Université Pierre et Marie Curie

[PLDAC - 2015]

Rapport de projet

Diffusion de l'information inter-communautés dans les réseaux sociaux.

Author :
Niki Rohani

Supervisor :
Sylvain Lamprier
Simon Bourigaults

10 juillet 2016

Table des matières

Résumé

Depuis l'émergence des réseaux sociaux en ligne, les problématiques liées à la diffusion d'informations ont, dans la plupart des cas, été traitées avec une granularité utilisateur. En prenant en compte le fait que les informations disponibles sur les liens entre utilisateurs ou leurs actions externes au réseau sont souvent inconnus, il devient très difficile de pouvoir prévoir le comportement d'un utilisateur unique durant le processus de diffusion. Nous proposons ici de traiter le processus de diffusion comme une diffusion inter-communautés en regroupant les utilisateurs ayant le même comportement en une entité unique.

0.1 Introduction

L'émergence des réseaux sociaux en ligne a motivé un grand nombre de recherches récentes. Le sujet auquel nous allons nous intéresser est celui de la propagation de l'information. Le but est d'étudier les intéractions entre utilisateurs, comme le partage d'une information ou le "j'aime" de Facebook, affectant la manière dont un contenu est propagé.

Afin de modéliser la diffusion d'information dans ces réseaux, plusieurs modèles ont vu le jour. Parmi ces modèles, plusieurs sont issus de l'épidémiologie, les plus courants étant l'independent cascade model (IC) [18] et le linear Threshold Model (LT) . Ceux-ci considèrent l'information comme un virus infectant l'un après l'autre les individus d'une population en passant de l'un à l'autre en suivant les arcs d'un graphe social. IC est un modèle réprésentant la diffusion comme des cascades [1] d'infections, modélisant la probabilité d'un utilisateur d'infecter ses voisins, tandis que LT détermine l'infection d'un utilisateur suivant un seuil d'influence de ses voisins.

Nous nous focaliserons ici sur le modèle IC, modèle basé sur les cascades, car ce type de modèle est connu pour décrire le processus de diffusion de manière plus précise que ses homologues [9]. Il y a quelques années que [18] a proposé une méthode afin d'apprendre les paramètres (probabilités de diffusion utilisateur à utilisateur) du modèle IC de manière plus efficace (par rapport à l'article [7]). Depuis, le modèle IC a servi de base pour d'autres modèles de diffusion orientés cascades, différant de par leur manière de concevoir les temps d'infection [17] ou en ajoutant des meta-données sur les utilisateurs ou leurs cascades [19][11][8]. Nous fonderons notre étude sur un modèle IC basique. IC se sert d'un graphe social et représente la diffusion d'information comme une chaîne où l'information est propagée d'un utilisateur à un autre à travers les liens du graphe [18].

Le graphe social étant soit incomplet, quand il est connu grâce aux liens d'amitié ou aux relations, soit inconnu pour des raisons de propriété privée, il doit donc être induit. Une raison de ce besoin de déduire le graphe social est

1. Une cascade représente une suite de triplets contenant l'utilisateur infecté, son temps d'infection, et l'information par laquelle il a été infecté.

que celui que nous pouvons posséder a priori grâce aux données des réseaux, telles que les liens d'amitié etc, n'est pas représentatif du phénomène de diffusion. Certaines méthodes se fondent sur des graphes où les utilisateurs sont tous reliés [15][5], la nôtre se fondera sur un graphe appris grâce aux participations aux cascades [2]. Une fois le graphe connu, IC peut apprendre un modèle probabiliste, où chaque utilisateur infecté au temps t à une probabilité d'infecter un de ses voisins au temps t+1. Si pendant le processus d'infection, l'utilisateur A échoue dans la transmission au temps t+1 d'un voisin B, alors A ne peut plus infecter B durant le reste de la diffusion[18].

Le fait de considérer une diffusion d'information à l'échelle utilisateur pose certains problèmes. Un de ces problèmes est celui du manque de généralisation. En effet, considérer le phénomène de diffusion à cette échelle avec le peu de de données que l'on possède sur l'utilisateur et sur la diffusion ne nous permet pas de modéliser le mecanisme sous-jaccent de diffusion. De ce fait, les modèles issus de l'étude à l'échelle utilisateur risquent très souvent d'être trop proches des données d'apprentissage et par conséquent ne permettent pas de calculer de nouvelles diffusions. Les graphes sociaux ne sont pas comme des graphes aléatoires, ils possèdent des propriétés qui les différencient de ces derniers [14]. Une de leur propriété est celle des *communautés*. Une communauté est constituée de personnes similaires. Cependant la notion de similarité est très compliquée à définir. Une grande partie de la littérature de la détection de communauté se base sur des communautés dites *Cohésives* [16] [4], c'est-à-dire qu'elle crée des communautés de noeuds fortement reliés au sein de cette dernière et faiblement reliés en dehors. Ce principe de regroupement joue le rôle de frein dans la diffusion d'information quand il s'agit de s'intéresser à la diffusion inter-communautés. Un autre type de communauté que [22] présente (2-mode community) permet de regrouper les utilisateurs communiquant avec les mêmes groupes d'utilisateurs.

Dans ce rapport, nous allons dans un premier temps tenter d'apporter une brève vision de l'état de l'art dans le domaine du regroupement par communauté dans l'analyse d'une diffusion, nous allons ensuite essayer de nous inspirer des modèles et vision des différents articles étudiés afin de présenter quelques modèles d'expériences, que nous allons par la suite expérimenter afin

2. Nous considérons qu'il existe un lien orienté de a à b si a est présent, dans un temps inférieur à b, à une cascade ou b est présent.

de pouvoir mettre en lumière quelques-unes des propriétés que recèlent nos modèles et plus généralement la diffusion inter-communautés. Notre rapport est structuré en 3 parties. La première consistera à décrire les concepts de diffusion d'informations et de cascades d'informations, de présenter quelques algorithmes d'apprentissage des poids de diffusions d'information, puis de faire un rapide tour de la littérature de la détéction de communautés classiques et de celle qui utilise les informations de diffusions. La deuxième partie consistera à présenter nos modèles d'expériences et recherches découlant de nos lectures et la dernière à apporter une interprétation des résultats obtenus.

Chapitre 1

Etude bibliographique

1.1 Introduction aux modèles de diffusions

1.1.1 Définitions

Cette section sert à définir de manière formelle quelques concepts indispensables à la compréhension de ce document.

Graphe social

Comme décrit dans la section 1., nous pouvons modéliser le réseau social, c'est à dire les utilisateurs, par un graphe où les noeuds correspondent aux utilisateurs et les arrêtes aux liens sociaux. Définition formelle : Soit $G = (N, A)$ un graphe G, N étant les noeuds et $A \in NxN$ les arrêtes dirigés représentant les liens entre les utilisateurs du réseau social.

Cascade d'informations

Soit U un utilisateur et I une information, le fait que U ait été infecté par I c'est-à-dire que U adopte ou diffuse I à un temps t, est défini par le triplet : (U, I, t). Une cascade est alors une suite de triplets où chaque utilisateur est infecté par une information à un temps t peut être continue ou discret. Quand t est discret on parle alors de pas de temps.

1.1.2 Un modèle de diffusion simple : Independant Cascade Model

Quand il s'agit d'étudier la diffusion d'informations dans un réseau, un modèle très utilisé est l'*independant cascade model* [**?**]. Ce modèle considère le graphe social comme un graphe de diffusion. Chaque lien du graphe $(u, u') \in A$ est associé à une probabilité de diffusion $K_{u,u'}$. Cette probabilité représente la probabilité que u a d'infecter un de ses voisins. La diffusion commence avec un ensemble D_0 d'utilisateurs infectés. Quand un noeud $u \in D_t$ est infecté au temps t, il a une chance d'infecter un de ses voisins au temps $t + 1$. Dans le modèle classique, il est considéré que si u échoue dans l'infection d'un de ses voisins au temps $t + 1$ alors qu'il a été infecté au temps t, alors il ne tentera plus d'infecter ce voisin. Nous verrons plus tard qu'il est possible de modifier cette condition et que sa modification entraîne un accroissement des performances de IC. Le processus de diffusion s'arrête une fois qu'il n'y a plus de diffusion possible, c'est-à-dire une fois que le modèle est stable.

Apprentissage des paramètres

Les paramètres de IC sont donc les probabilités de diffusion. Afin d'apprendre ces paramètres [18] propose la méthode suivante. Si nous considérons une cascade d'informations D, où D_T est l'ensemble des utilisateurs infectés au temps t, pour chaque lien $e = (u, u') \in A$, $u \in D_T$ et $u' \in D_{T+1}$ il est probable que u est infecté u' à travers le lien e. Cependant, il faut considérer la probabilité qu'un autre utilisateur y ait infecté u' avec $y \in D_T$ et $y \in B'_u$

où $B_u = \{u' : (u', u) \in A\}$. La probabilité qu'un utilisateur w soit infecté au temps $t+1$ est donnée par l'équation 1.

$$P_w(t+1) = 1 - \prod_{v \in B_w \cap D_T} (1 - K_{v,w}) \tag{1.1}$$

La vraisemblance est définie à l'équation 2.

$$L(\theta; D) = (\prod_{t=0}^{T-1} \prod_{w \in D_{t+1}} P_w(t+1))(\prod_{t=0}^{T-1} \prod_{w \in D_t} \prod_{w \in F_v C_{t+1}} (1 - K_{v,w})) \tag{1.2}$$

Avec $F_v = w : (v, w) \in A$ et C_t l'ensemble des utilisateurs infectés depuis 0 à t. En considérant l'ensemble des cascades indépendantes, nous pouvons définir la vraisemblance comme suit : $L(\theta) = \sum_{D_s \in cascade} logL(\theta; D_s)$.

La formule de la vraisemblance n'étant pas facilement optimisable, l'apprentissage se fait selon un algorithme EM. L'estimation des paramètres est basée sur les probabilités de succès des infections, chaque probabilité $K_{v,w}$ est modifiée à chaque pas de EM.

1.1.3 Le modèle Linear Threshold vs IC

Un autre modèle de diffusion est le modèle Linear Threshold [6] modélise la diffusion d'informations comme un phénomène de pression sociale. Ainsi, l'infection d'un utilisateur u est déterminée par la pression de ses voisins. Le modèle admet qu'il existe un seuil λ : s'il est dépassé par la pression des voisins déjà infectés de u, alors u sera à son tour infecté. [10] a proposé un modèle généralisant LT et proposant une méthode pour la diffusion d'informations dans un réseau social. Nous considérons comme pour IC un réseau $G = (N, A)$ où chaque lien $(u, u') \in A$ est dirigé et à un poids $b_{u,u'}$ correspond à la probabilité d'influence de u envers u', donc $b_{u,u'} <= 1$. La propagation se fait donc de manière suivante : si un utilisateur u n'est pas infecté au temps t, il sera infecté si $\sum_{u' \in V(u)} b_{u',u} >= \theta_u$, où $V(u)$ est l'ensemble des utilisateurs u' tel que $u' : (u', u) \in A$ et u' est infecté. θ_u représente le seuil de u, représentant la propension de u à être influencé par ses voisins. Le processus est aussi itératif comme IC.

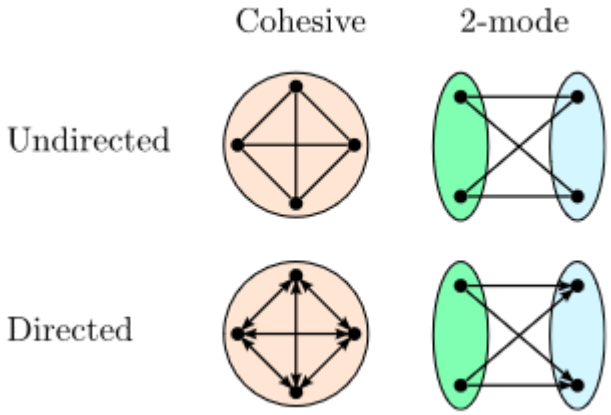


FIGURE 1.1 – Un exemple de communautés dîtes cohésives et 2-mode

Alors que IC se focalise sur les capacités des utilisateurs à influencer leurs voisins, LT se concentre sur la propension d'un utilisateur à être infecté par ses voisins. L'étude [9] montre que IC permet de mieux représenter la dynamique de diffusion que LT. C'est pourquoi nous avons choisi d'utiliser le modèle IC lors de nos expériences afin de représenter le phénomène de diffusion d'informations.

1.2 Détection de communautés pour la diffusion d'informations

1.2.1 Types de communautés

Un graphe est un modèle abstrait général pour représenter des objets reliés entre eux. Une manière efficace de modéliser et étudier un graphe est de découvrir les objets qui suivent le même schéma de connectivité. Ce travail est principalement celui de découvrir les ensembles de communautés dans un graphe. Ce sujet d'étude est encore aujourd'hui en plein essort. [22] dénote deux familles d'étude au sujet de la détéction de communautés :

— La détéction de communautés **Cohésives** : Ensemble d'objets fortement reliés entre eux mais très peu reliés avec l'extérieur.
— La détéction de communautés **2-mode** : Structure réunissant des objets réliés entre eux et par leur similarité de comportement vis-à-vis des objets extérieurs.

La détéction de communautés Cohésives a longtemps été le sujet d'étude principale pour ce qui est de la découverte de communautés dans un graphe [1][3][4]. Le problème de ces structures est qu'elles ne sont pas adaptées au phénomène de diffusion d'informations de par leur nature. C'est pour cette raison que certains chercheurs ont tenté d'étendre la définition de communautés, une de ces extensions est le concept de communautés 2-mode [22].

Ces communautés ne sont plus constituées de noeuds fortements reliés, mais prennent en compte le comportement des noeuds interieurs vis-à-vis des noeuds exterieurs. L'idée est de pouvoir modéliser des communautés dont les noeuds intéragissent de manière similaire avec l'extérieur. La définition de communauté varie donc fortement en fonction du besoin d'analyse. Un autre aspect variant est celui de l'assignement d'un noeud à un cluster. Cette assignement peut être fait de manière unique comme Kmean [12] ou encore le clustering hiérarchisé, ou de manière multiple, aussi nommé *overlapping communities* qui peut se traduire par communautés chevauchées [21] considérant qu'un noeud ou un utilisateur peut appartenir à plusieurs communautés.

1.2.2 CCN : Utilisation des cascades pour inférer les communautés

Maintenant que nous avons présenté une liste non exhaustive de type de communauté que la littérature répertorie, nous allons présenter un article qui traite le sujet de la détéction de communautés en utilisant l'information contenue dans les cascades afin d'en déduire des communautés chevauchées. Cette méthode a été proposée par [2].

L'idée de cette méthode est de détecter des communautés du graphe social qui expliquent aussi les cascades. Les auteurs de l'article déplorent le fait que très peu de recherches mettent en commun la détection de commu-

nautés et l'étude du phénomène de contagion de l'information. Selon eux, la nature des cascades et des communautés sont naturellements opposées : les communautés bloquent la propagation des cascades et, quand une cascade vient à s'estomper, une communauté peut en expliquer la raison. Leur intuition vienst donc du fait que si une communauté peut expliquer la raison de l'arrêt d'une cascade, alors, en observant des cascades passées, nous pouvons en déduire des communautés.

Le modèle considère que chaque observation d'une cascade est le résultat de l'action d'un utilisateur u selon sa communauté c qui représente un de ses points d'intérêt. Un utilisateur u possède un degré de participation à une communauté c. Ce degré est déterminé par deux paramètres : le degré $\pi_u^{c,s}$ représentant la participation active et $\pi_u^{c,d}$ la participation passive. $\pi_u^{c,s}$ caractérise le potentiel de u à propager de l'information dans c, donc à être acteur. $\pi_u^{c,d}$ caractérise le potentiel de u à être infecté par une information dans c. La propagation d'information se fait donc en se propageant dans une communauté c , puis en se propageant à travers les liens de cette communauté et enfin peut infecter une communauté différente c' que si il y a un utilisateur de c qui est aussi dans c' et ayant un assez haut degré de participation passive dans c ainsi qu'un assez haut degré de participation active dans c'. Les liens inter-communautés sont donc établis par les utilisateurs qui se trouvent dans le chevauchement de plusieurs communautés.

Définition formelle du modèle CCN

Nous partons des mêmes définitions que dans la section 1. Soit $t_i(u)$ le pas d'activation de l'utilisateur u pour l'information i, si l'utilisateur n'a jamais été infecté par i, alors $t_i(u) = \infty$. $D_i(t)$ l'ensemble des utilisateurs qui sont infectés par i au temps t, et $C_i(t)$ l'ensembe des utilisateurs qui ont déjà été infectés par i au temps t. Nous notons aussi un seuil d'infection qui démarque le temps maximum pendant lequel un utilisateur u peut infecter un autre utilisateur, nous notons ce temps Δ. Nous en déduisons donc $F_i(t)$ l'ensemble des utilisateurs pouvant propager i à l'instant t.
$F_i(t) = \{u \in N | t - t_i(u) <= \Delta\}$
Nous pouvons déduire de la formule précédente, $F_{i,u}(t)$ représentant les utilisateurs potentiels dans la cause de l'infection de u par i au temps t.

$F_{i,u}(t) = \{v \in N | (v,u) \in A \wedge 0 <= t_i(u) - t_i(v) <= \Delta\}$
Le modèle CCN ne s'intéressant qu'aux phénomènes d'influences, nous pouvons supprimer les observations pour lesquelles aucun influenceur n'est la cause de l'infection, i.e. $F_{i,u} = \emptyset$.
$D = \{(u,i,t) \in L | F_{i,u}! = \emptyset\}$

La propagation d'informations à travers le modèle entre u et v ne peut se faire que si une communauté c peut influencer u et que v est influençable par u au regard de leur degré d'activité et de passivité. A noter qu'un lien (u,v) ne peut exister si il n'y pas pas une communauté c dans laquelle u a un haut degré d'activité et v un haut degré de passivité. Une manière intéressante de modéliser les cascades D, est de les voir comme un graphe bi-partie, où les liens ne peuvent être établis qu'entre les informations I et les utilisateurs N. Les poids sont les $t_i(u)$, c'est-à-dire le temps auquel u a été activé par i. De ce point de vue, un lien (i,u) ne peut exister s'il n'y a pas une communauté c et un influenceur u' qui détient un haut ratio d'influence, et u un haut ratio de passivité. Ce ratio est défini par deux distributions multinomiales : ϕ^c le ratio de passivité et ϑ^c le ratio d'activité dans c. Les formules sont données en 2 et 3.

$$\vartheta_u^c = \frac{exp^{\pi_u^{c,s}}}{\sum_{u' \in N} exp^{\pi_{u'}^{c,s}}} \tag{1.3}$$

$$\phi_u^c = \frac{exp^{\pi_u^{c,d}}}{\sum_{u' \in N} exp^{\pi_{u'}^{c,d}}} \tag{1.4}$$

Ces ratios peuvent se traduire par la capacité à être influenceur/influençable dans c par rapport à la somme des capacités des utilisateurs $u \in N$ dans c. De même à chaque activation a, il existe un influenceur et un influencé. Vu que le modèle est génératif, ils sont tirés selon deux distributions multinomiles :

$$\theta_u^{c,s} = \frac{exp^{\pi_u^{c,s}}}{\sum_{u' \in F_{i_a}(t_a)} exp^{\pi_u^{c,s}}} \tag{1.5}$$

$$\Phi_{u,v}^{c,s} = \frac{exp^{\pi_u^{c,d}}}{\sum_{v':(u,v') \in A, v'/\in C_{i_a}(t_a - 1)} exp \pi_u^{c,s}} \tag{1.6}$$

Notez que t_a et i_a dénotent le temps t et l'information i de a. Les influenceurs u sont choisis en fonction des influenceurs potentiels. Un utilisateur v ne peut être influencé s'il est déjà actif où s'il n'est pas connecté à u. On peut voir la génération des nouveaux liens entre i et des utilisateurs v dans le graphe des cascades comme un modèle markovien où l'on choisit d'abord les utilisateurs u qui ont déjà été connectés à i et en choisissant v dans l'ensemble des noeuds tel que $(u, v) \in A$ dans le graphe social.
Nous avons là les paramètres du modèle :
$\Pi = \{\pi_1, ..., \pi_C\}$ représente la distribution a priori des communautés c.
$\Pi^s = \{\pi_1^s, ..., \pi_C^s\}$ représente pour chaque utilisateur son degré d'activité dans c
$\Pi^d = \{\pi_1^d, ..., \pi_C^d\}$ représente pour chaque utilisateur son degré de passivité dans c
Nous pouvons maintenant décrire la phase d'apprentissage du modèle.

Apprentissage du modèle CCN

Le but est maintenant d'apprendre les paramètres $\Theta = \{\Pi, \Pi^s, \Pi^d\}$. Les auteurs supposent les liens sociaux et les cascades indépendantes, la vraisemblance des données par rapport à Θ est donnée dans l'equation 1.7.

$$P(G, D|\Theta) = \prod_{(u,v)\in A} P(u, v|\Theta) \prod_{u\in D} P(a|\Theta) \tag{1.7}$$

$$P(u, v|\Theta) = \sum_k \theta_u^k \phi_u^k \pi_k \tag{1.8}$$

$$P(a|\Theta) = \sum_k \sum_{u\in F_{i_a}, v_a} \pi_k \theta_u^{k,a} \Phi_{u,v_a}^{k,a} \tag{1.9}$$

L'algorithme étant un EM, l'estimation des paramètres cachés se fait comme sur la figure 1.2. L'algorithme est donné à la figure 1.3.

$$Q(\Theta, \Theta') = \sum_{(u,v)\in A} \sum_{k} \gamma_{u,v,k}(\Theta') \left[\log \pi_k + \log \vartheta_u^k + \log \varphi_v^k\right]$$
$$+ \sum_{a\in\mathbb{D}} \sum_{k} \sum_{u\in\mathcal{F}_{i_a,v_a}} \eta_{u,a,k}(\Theta') \left(\log \pi_k + \log \theta_u^{k,a} + \log \phi_{u,v_a}^{k,a}\right)$$

$$\begin{aligned}\gamma_{u,v,k}(\Theta) &= P(z_\ell^k | \ell \equiv (u,v) \in A, \Theta) \\ &= \frac{\vartheta_u^k \varphi_v^k \pi_k}{\sum_{k'} \vartheta_u^{k'} \varphi_v^{k'} \pi_{k'}}\end{aligned}$$

$$\begin{aligned}\eta_{u,a,k}(\Theta) &= P(z_a^k, w_a^u | a \in \mathbb{D}, \Theta) \\ &= \frac{P(a \in \mathbb{D} | w_a^u, z_a^k, \Theta) P(w_a^u | z_a^k, \Theta) P(z_a^k | \Theta)}{P(a \in \mathbb{D} | \Theta)} \\ &= \frac{\phi_{u,v_a}^{k,a} \theta_u^{k,a} \pi_k}{\sum_{k'} \sum_{u'\in\mathcal{F}_{i_a,v_a}} \phi_{u',v_a}^{k',a} \theta_{u'}^{k',a} \pi_{k'}}\end{aligned}$$

FIGURE 1.2 – Estimation des paramètres de CCN

Algorithm 1: CCN model parameters estimation

Input : Graph $G = (N, A)$, propagation log $\mathbb{D}$, and $K \in \mathbb{N}^+$.
Output: The set of all parameters, $\Theta = \{\Pi, \Pi^s, \Pi^d\}$.

init(Π, Π^s, Π^d); //Initialization of parameters
repeat
  E-step
  forall the $k \in \{1, \dots K\}$ **do**
    forall the $\ell \equiv (u, v) \in A$ **do**
      compute $\gamma_{u,v,k}$ according to eq. 9
    end
    forall the $a \equiv (i_a, v_a, t_a) \in \mathbb{D}$ **do**
      forall the $u \in N$ **do**
        compute $\eta_{u,a,k}$ according to eq. 10
      end
    end
  end
  M-step
  forall the $k \in \{1, \dots K\}$ **do**
    compute π_k according to equation 11
    forall the $u \in N$ **do**
      compute δ_u^k and λ_u^k according to eqs. 5-6
      $\pi_u^{k,s(new)} \leftarrow \pi_u^{k,s} + \delta_u^k$
      $\pi_u^{k,d(new)} \leftarrow \pi_u^{k,d} + \lambda_u^k$
    end
  end
until convergence;

FIGURE 1.3 – Estimation des paramètres

1.2.3 CSI : Un modèle d'influence de communautés basé sur les études de cascades.

L'article précédent nous proposait un modèle permettant de trouver des communautés grâce aux cascades et au graphe social. Nous allons tenter d'étudier une méthode exploitant elle aussi les cascades mais, cette fois-ci, à l'instar d'une modélisation optimale des communautés, à la modélisation des influences inter-communautés. La méthode présentée dans [13] *Community-level Social Influence analysis* permet l'analyse du phénomène de diffusion grâce à un clustering hiérarchisé et de l'algorithme IC. Le modèle s'intéresse à la tendance des utilisateurs à être influenceurs, et cherche à déterminer les groupes d'influenceurs qui peuvent exister dans le réseau. CSI se base sur des cascades observées afin d'induire ces tendances. Selon les auteurs, leur technique fournit une aide à l'analyse de réseaux large échelles en regroupant les utilisateurs jouant le même rôle dans le phénomène diffusion. Dans un premier temps, le modèle CSI modifie quelques caractéristiques d'IC.

Modification d'IC

Une des caractéristiques qui ont été critiquées est celle du fait que l'infection d'un utilisateur u' par un utilisateur u ne peut être faite qu'à $t+1$ avec t = temps d'infection de u. Dans le modèle CSI un utilisateur u' peut être infecté par un utilisateur u si u' est infecté avant un delai Δ. Plus formellement, $F_{\alpha,u'}+ = \{u|(u;u') \in A; 0 <= t_\alpha(u') - t_\alpha(u) <= \Delta\}$ représente l'ensemble des voisins de u' qui ont pu infecté u'.

Description du modèle

CSI se base non pas sur une analyse de l'influence utilisateur-utilisateur, mais sur une analyse communauté-communauté. Pour ce faire, le modèle utilisateur un clustering hiérarchisé, le but étant de trouver un arbre H de G. H est un arbre ayant comme racine un cluster regroupant tout les utili-

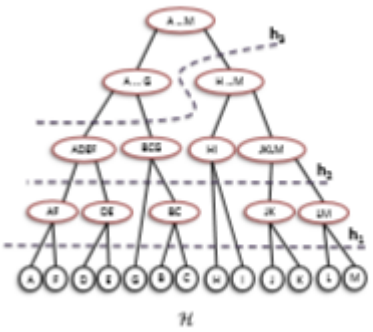


FIGURE 1.4 – Exemple d'un clustering hiérarchisé. Les feuilles correspondent aux utilisateurs. Chaque noeuds est une communauté regroupant les feuilles descendantes. Une coupure h de H corresponds en la section d'arrêtes ayant comme noeud fils les communautés.

sateurs avec les noeuds N comme feuilles et un nombre variable de noeuds intermédiaires. Une coupure h de H est un ensemble de noeuds intermédiaires tel que pour $n \in N$, un et un seul noeud intermédiaire $e \in h$ connecte n à r la racine de H. Ainsi chacun des noeuds $e \in h$ représente une communauté potentielle. Le but est alors de trouver la coupure optimale de H. Notons cette coupure h^*. Notons $C(H)$ toutes les coupures de H. $h \in C(H)$ induit, donc P_h une partition de G tel que toutes les feuilles $n \in H$ connectées à un noeud $e \in h$ appartiennent au même cluster $c_e \in N$. Les auteurs définissent $a(n)$ le cluster du noeud $n \in N$. Dans le modèle CSI toutes les arrêtes d'un même cluster ont la même probabilité d'influencer les autres clusters, c'est-à-dire que l'on peut ne considérer qu'un seul lien d'un cluster vers un autre. Etant donné la fonction $p_h : P_h * P_h -> [0, 1]$ assignant la probabilité d'influence à chaque cluster, les auteurs définissent la vraisemblance.

$$logL(D|p) = \sum_{\alpha \in D} logL_\alpha(p) \tag{1.10}$$

$$L_\alpha(p) = \prod_{v \in V} [1 - \prod_{u \in F^+_{\alpha,v}} 1 - p(u,v)][\prod_{u \in F^-_{\alpha,v}} 1 - p(u,v)] \tag{1.11}$$

Avec $\alpha \in D$ une cascade.

Apprentissage du modèle

Le problème est alors de trouver la meilleure coupe h^* tel que $h^* = argmin_{h \in C(H)} f(L(D, p_h), h)$ avec $L = vraissemblancedeD$, D un ensemble de cascades. Les auteurs ne précisent pas les algorithmes qu'ils ont utilisés pour la coupe. Une fois la coupe connue, il faut apprendre le poids des liens entre communautés, qui determinera le poids d'influence d'une communauté à une autre. Cette tâche est effectuée avec un aglorithme EM. La formule de la vraisemblance de l'espérance est définie comme suit :

$$\begin{aligned} Q(p_h, p^o ld_h) = \sum_{a \in D} \sum_{u} \{ & \sum_{u' \in F^+_{\alpha,u}} \phi_{\alpha,u,u'} log p_h(a(u), a(u')) + \\ & (1 - \phi_{\alpha,u,u'}) log(1 - p_h(a(u), a(u'))) + \\ & \sum_{u \in F^-_{\alpha,u}} log(1 - p_h(a(u), a(u'))) \} \end{aligned} \tag{1.12}$$

Où $\phi_{\alpha,u,u'}$ la probabilité que l'infection de u soit due à u' dans la cascade α. Le but étant donc de maximiser Q pour $\delta Q = 0$ pour chaque $p_h(x, y)$ avec $x, y \in P_h$. Ce qui nous donne

$$p_h(x, y) = \frac{1}{|S^+_{x,y}| + |S^-_(x, y)|} \sum_{\alpha \in D} \sum_{v \in A(y)} \sum_{u \in F^+_{\alpha,v} u \in A(x)} \phi_{\alpha,v,u} \tag{1.13}$$

Où $S^+_{x,y} = \sum_\alpha \sum_{v \in A(y)} \sum_{u \in A(x)} I(u \in F^+_{\alpha,v})$ et $S^+_{x,y} = \sum_\alpha \sum_{v \in A(y)} \sum_{u \in A(x)} I(u \in F^-_{\alpha,v})$, il reste à définir $\phi_{\alpha,u,v}$. Pour déterminer la probabilité, il suffit d'estimer les probabilités $p(u, v)$ du modèle IC sur l'ensemble des utilisateurs, puis de déduire la probabilité d'influence dans une cascade.

$$\phi_{\alpha,u,v} = \frac{p(u, v)}{1 - \prod_{w \in F^+_{\alpha,v}} 1 - p(w, v)} \tag{1.14}$$

La procédure d'apprentissage est définie ainsi :

- On utilise l'algorithme d'apprentissage IC sur l'ensemble des utilisateurs et des cascades.
- On déduit $\phi_{\alpha,u,v}$
- On calcule $\phi(x, y)$ selon l'équation 10.

Algorithme pour la coupe optimale

L'exploration de l'espace des possibilités $C(H)$ est exponentielle en N. Les auteurs ont trouvé une heuristique permettant de ne pas avoir à effectuer une recherche exhaustive. L'opération commence à la coupe correspondant aux feuilles, h1 sur la figure 1.4. A chaque itération, le calcul de toutes les coupes possibles obtenues à partir de la fusion des communautés partageant le même parent est fait. Plus formellement, à partir de $h = a_1, ..., a_h \in C(H)$, les candidats à la fusion sont définis par $M(h) = \{< y, y' >: (x, y) \in hand(x, y') \in h, x! = r\}$.La fonction de similarité $similarity(y, c, p) = \sum_z (p(y, z)p(c, z) + p(z, y)p(z, c))$ permet de calculer le meilleur candidat pour une fusion. Cette similarité peut être vue comme un produit des vecteurs d'influence des communautés y et c. Ainsi les utilisateurs sont regroupés selon leur tendance à influencer les autres.

Expérience et conclusion du modèle CSI

Les expériences du modèle CSI ont étés faites sur trois différents jeux de données. Chacune de ces données ayant des propriétés qui leur sont propres. Ce que cette méthode a d'intéressant, c'est qu'elle permet de fournir une vue d'ensemble d'un réseau social et surtout d'en dégager les tendances de fonctionnement.

Chapitre 2

Présentation des modèles

2.1 Introduction

Maintenant que nous avons présenté quelques-unes des techniques de détection de communautés informées, nous allons présenter nos modèles, qui serviront de base à l'analyse de notre problématique.

La prédiction de la diffusion d'informations à l'échelle utilisateur reste encore aujourd'hui incertaine. Ceci est dû au fait qu'il est difficile de modéliser le comportement d'un utilisateur face à une information et face au comportement de ses voisins car il dépend de trop de facteurs inconnus. Cependant, nous pensons que des schémas de comportements peuvent être modélisés sur un groupe d'utilisateurs ayant un comportement similaire face au phénomène de diffusion, et ainsi pouvoir augmenter nos performances.

Dans ce chapitre nous allons présenter nos modèles, inspirés de ceux de l'état l'art que nous avons étudiés, qui serviront de base à nos expériences.

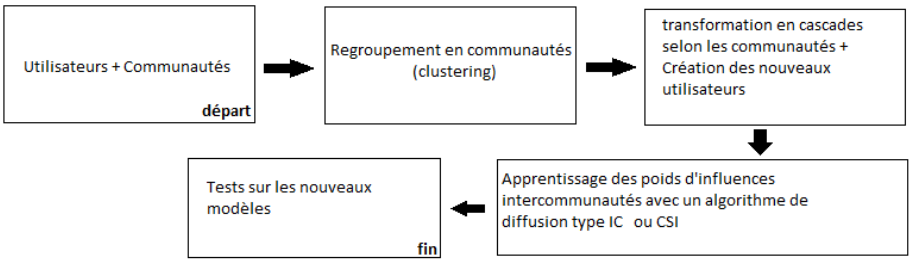


FIGURE 2.1 – Chaîne de traitement

2.2 Notations

Nous considérons les notation du chapitre 1.

— Nous considérons un ensemble d'utilisateurs U
— un ensemble de cascades C, dont un sous ensemble d'apprentissage noté C_a et de test C_t.
— Notons $t_c(u)$ le temps d'infection de u dans c.
— Un ensemble de communautés P où $P(u)$ rend la communauté de u. P_i désigne l'ensemble des utilisateurs de la $i$ème communauté.
— Une fonction de similarité afin de regrouper les utilisateurs notée Φ

2.3 Chaine de traitement

Le but étant de pouvoir analyser le comportement des communautés, nous devons dans un premier temps regrouper les utilisateurs en communautés, puis en déduire des liens entre communautés à partir de ceux connus des utilisateurs. De ces liens nous pourons déduire un poids représentant l'influence d'une communauté sur une autre. Pour la phase de regroupement nous avons utilisé un algorithme de clustering. Il nous a donc fallu définir des similarités entre utilisateurs. La chaîne de traitement est présentée sur la figure 2.

2.4 Phase de regroupement

Afin de regrouper les utilisateurs en communautés, nous avons utilisé l'algorithme de clustering K-means se basant sur un calcul de similarité entre utilisateurs.

Pour cela, nous avons dû définir une fonction de similarité entre utilisateurs. Nous avons choisi de tester 4 fonctions de similarités.

1. La première, $\phi_{coCascade}$, est plutôt simple, elle consiste à calculer l'indice de jaccard défini comme suit : $sim(u, u') = \frac{|c_u \cup c_v|}{|c_u|+|c_v|}$ où c_u correspond à l'ensemble des cascades de u. C'est une similarité basé sur les co-participations aux cascades, ce qui peut dans certains cas créer des communautés cohésives.
2. La deuxième similarité, $\phi_{timeStep}$, est basée sur la position des utilisateurs dans les cascades. Pour cela nous avons discrétisé le temps d'une cascade en 10 pas. Pour chaque cascade nous avons normalisé les pas de temps des infections.
3. La troisième, $\phi_{successor}$, est basé sur les successeurs communs. Il s'agit d'un regroupement par influence, où l'on regroupe les utilisateurs selon leur propension à influencer les mêmes groupes d'utilisateurs. Nous avons essayé de représenter l'intuition de CSI qui tente une modélisation des influences.
4. La dernière enfin, $\phi_{predecessor}$, est basé sur les prédecesseurs communs. Elle fonctionne de la même manière que son homologue mais cette fois-ci elle tente de modéliser la propension à se faire influencer par le même groupe d'utilisateur.

Une fois le regroupement effectué, nous avons créé des utilisateurs représentant ces groupes, en fusionnant la liste des successeurs et prédécesseurs de chaque utilisateur tel que $suc_{P_i} = \cup_{u \in P_i} suc(u)$ où $suc(u)$ représente l'ensemble des successeurs de u, $u \in U$, $P_i \in P$. $pred_{P_i} = \cup_{u \in P_i} pred(u)$.

2.5 Phase d'apprentissage

2.5.1 Apprentissage IC

Une fois le regroupement fait, nous devons transformer les cascades C_a et C_t en fonction des nouveaux groupes d'utilisateurs.

Pour cela, nous avons défini un seuil λ_p représentant le taux d'utilisateurs de $p \in P$ qu'il faut infecter dans une cascade $c \in C$ pour que p soit infecté par i où $c = \cup(u, i, t)$. L'infection est alors déterminée par : $t_c(p) = min(t_c(u))$ où $u \in p$. Pour chaque valeur de k, on apprend un modèle IC. Nous ne nous intéresserons pas ici à l'optimisation des paramètres. Voici les paramètres choisies :

- nbIteration = 100 : nombre d'itérations.
- nbCascade = -1 : on prend en compte toutes les cascades.
- reg : 0.3 : Terme de régularisation
- $\Lambda = \infty$: temps pendant lequel un utilisateur u peut infecter un utilisateur. Cela signifie qu'un utilisateur infecté peut infecter ses voisins durant tout le phénomène de propagation. Ce paramètre a été choisi par recommendation d'un encadrant et par l'utilisation [13] de ce paramètre. Un paramètre cependant à faire varier : $lambda_p$.

Chapitre 3

Experiences

3.1 Données

3.1.1 Digg

Digg est un site internet de " social news ". Ses utilisateurs postent des stories, des articles issus de blogs ou de journaux en ligne. Si un utilisateur apprécie une story, il peut lui attribuer un digg (l'équivalent d'un " +1 " ou d'un " like "). Les stories ayant reçu le plus de Diggs apparaissent sur la page d'accueil du site. Digg fournit une API de streaming permettant de surveiller le site et de récupérer l'ensemble des stories postées et des diggs attribués. Nous avons ainsi étudié des données provenant de l'ensemble de l'activité du site sur six jours. Nous avons considéré que chaque story correspondait à une cascade. L'ensemble des Diggs effectués par les utilisateurs permettent de calculer la matrice de contamination C. Pour échantillonner les timestamps, nous utilisons un pas de une heure et un horizon de deux jours.

Nous avons donc 4500 utilisateurs, un ensemble de 19172 cascades d'apprentissage et 8072 cascades de test. Le graphe social est obtenu en déduisant les liens sociaux, tel que pour tout $u, v \in U$ si u, v apparaissent dans la même cascade et $t_c(u) < t_(v)$ alors un lien (u, v) est crée.

3.1.2 Mesure d'évaluation

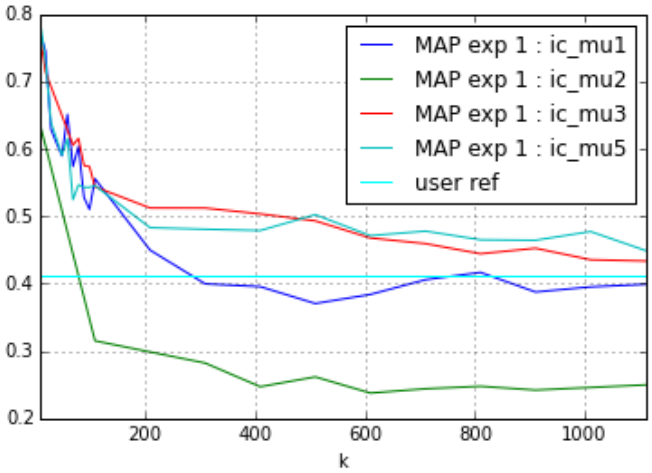
Afin d'effectuer nos mesure nous utiliserons le Mean Average Precision (MAP). Cette mesure calcule la moyenne des Average Precision, donc la moyenne de la mesure AP pour les cascades de test. Etant donné une cascade de test, l'AP de la prédiction est la somme pondérée de la précision de chaque bonne prédiction. Le set de prédiction est ordonné selon les infections. Ainsi les utilisateurs infectés sont placés en haut de la liste. Donné un ensemble de bonne prédiction, notons $n(r)$ le nombre de bonne prédiction dans la liste ordonnée au rang de r. L'équation de l'average precision est la suivante : $\sum_r \frac{n(r)}{rank(r)}$ avec $rank$ le rang, la place de r dans la liste.

3.2 Expérience avec IC

nous allons suivre la chaîne suivante :

1. Définition de la représentation vectorielle d'un utilisateur pour la similarité
2. Regroupement avec l'algorithme Kmean en k communautés, avec 5 groupements pour Kmean, c'est-à-dire que Kmean effectue 5 fois la même tâche et choisti le résultat qui maximise la similarité.
3. Transformation des cascades selon le paramètre λ, i.e. le seuil d'utilisateurs infectés pour infecter une communauté
4. Création des nouveaux utilisateurs en fusionnant les communautés, leurs successeurs et leurs prédécesseurs.
5. Apprentissage des poids entre chaque utilisateur avec l'algorithme IC paramétré selon les paramètres génériques définis à la section 2.5.1
6. Evaluation des performances avec MAP en prédiction de diffusion finale avec étude qualitative et quantitative.

Le but étant donc de faire varier k, pour chaque choix fixe de lambda et des stratégies de similarités.



(a) performances MAP pour IC

FIGURE 3.1 – Performances en MAP, en fonction des différentes stratégies. User ref représente la performance MAP sans communauté.

Nous allons aussi faire varier lambda, pour chaque choix fixe de k et des stratégies de similarités.

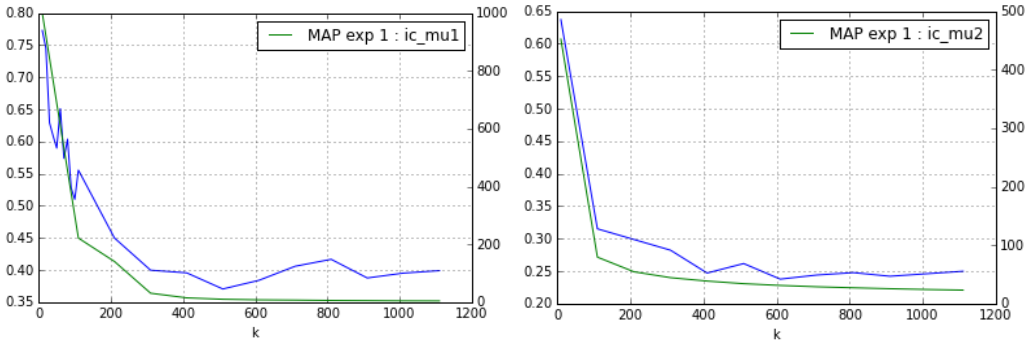
Définition vectorielle des utilisateurs : Choix des similarité

Nous avons donc établi 4 définitions vectorielles que nous noterons μ.

— μ_1 : Cette stratégie correspond à $\phi_{coCascade}$
— μ_2 : Cette stratégie correspond à $\phi_{timeStep}$
— μ_3 : Cette stratégie correspond à $\phi_{coSuccessor}$
— μ_5 : Cette stratégie correspond à $\phi_{coPredecessor}$

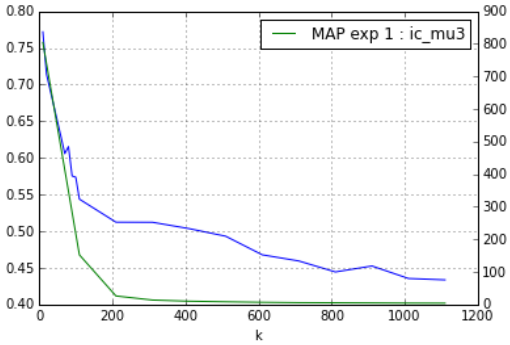
3.2.1 Evaluation des résultats

Les évaluations ont été effectuées sur la base transformée de cascades C_t. Nous avons fait varier k de 10 à 1200. L'évaluation a été faite avec IC sur

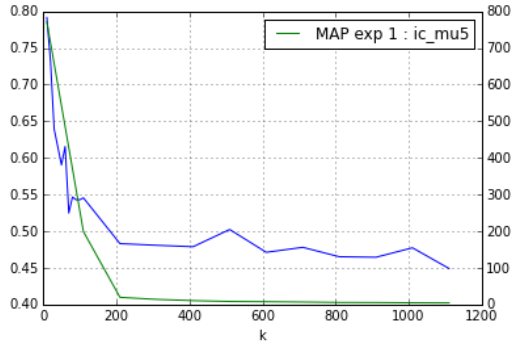


(a) MAP μ_1

(b) MAP μ_2

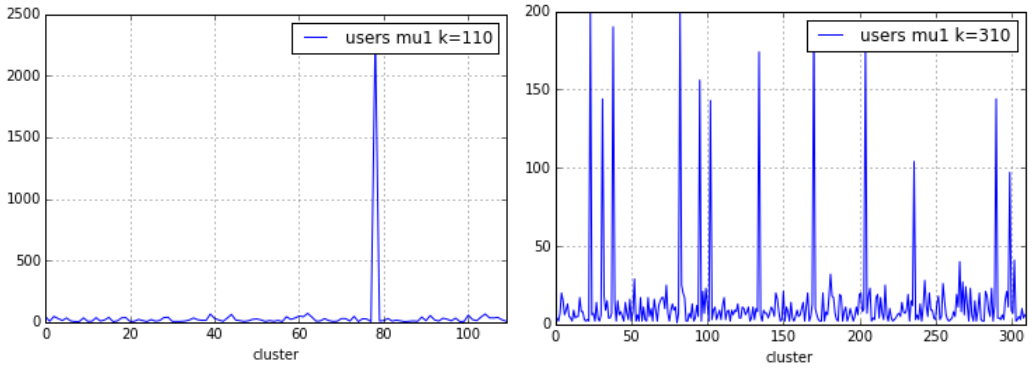


(c) MAP μ_3



(d) MAP μ_5

FIGURE 3.2 – Performances en MAP (bleue) et variance des tailles de communautés (en vert), en fonction des différentes stratégies.



(a) μ_1 k = 110 (b) μ_1 k = 300

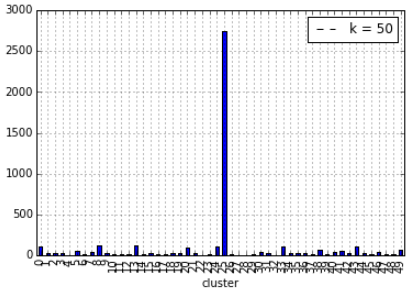
FIGURE 3.3 – Répartition en terme de nombre pour chaque communautés. Chaque communauté est représentée par un point en abscisse. Sa coordonnée en ordonnée correspond à la taille.

une prédiction avec 5000 simulations de Monte Carlo.

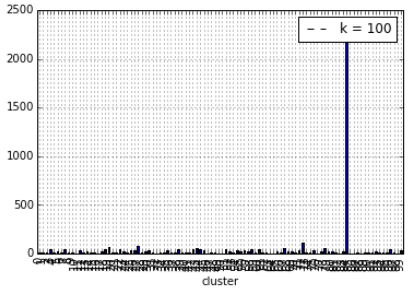
Evaluation pour $\lambda = 0$

Nous avons tout d'abord évalué les différentes stratégies μ en fixant λ à 0 où pour chaque communauté il suffit d'un seul utilisateur infecté pour infecter la communauté. Cette stratégie est de fait très sensible à l'infection et a donc tendance à être moins représentative du comportement communautaire si celles ci sont trop grandes.

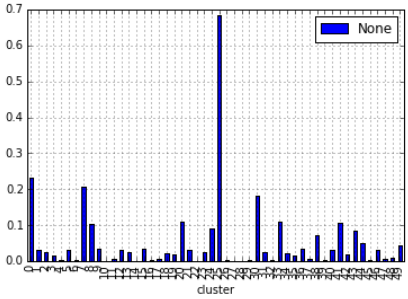
Sur la figure 3.1 nous pouvons voir un comparatif des résultats obtenus selon les différentes stratégies. Ce que nous pouvons voir, c'est une nette différence de performance en fonction des stratégies de similarités, donc de regroupement d'utilisateurs, pour les mêmes paramètres d'apprentissages et de tests. Cette différence prouve le fait que le critère de regroupement en communauté, et non simplement le regroupement en lui même, est une donnée importante à ne pas négliger, qui influence considérablement les prédictions. Selon notre interprétation, certaines des stratégies de similarités ne représentent pas bien le modèle de diffusion et donc créent des groupes non représentatifs de ce phénomène.



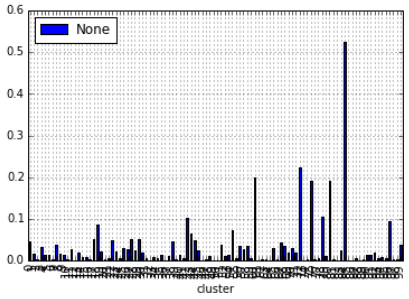
(a) users/communautés, k = 50



(b) users/communautés, k = 100

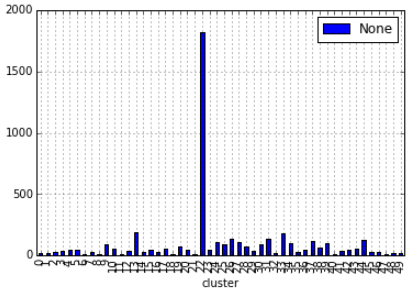


(c) infections/communautés, k = 50

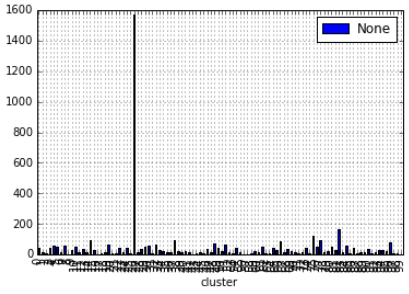


(d) infections/communautés, k = 100

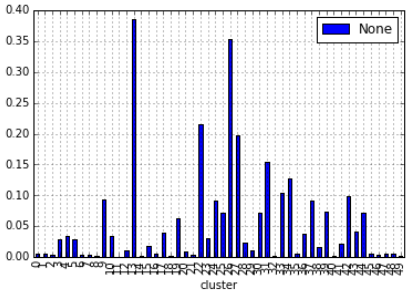
FIGURE 3.4 – Rapport entre le nombre de users par communauté, sur la première ligne, et la propension d'infection par communauté, deuxième ligne, pour μ_1. Les communautés sont représentées par leur numéro en abscisses. On voit ici que les communautés dominantes sont celles qui sont le plus infectées.



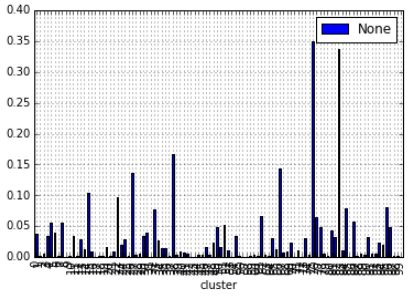
(a) users/communautés, k = 50



(b) users/communautés, k = 100

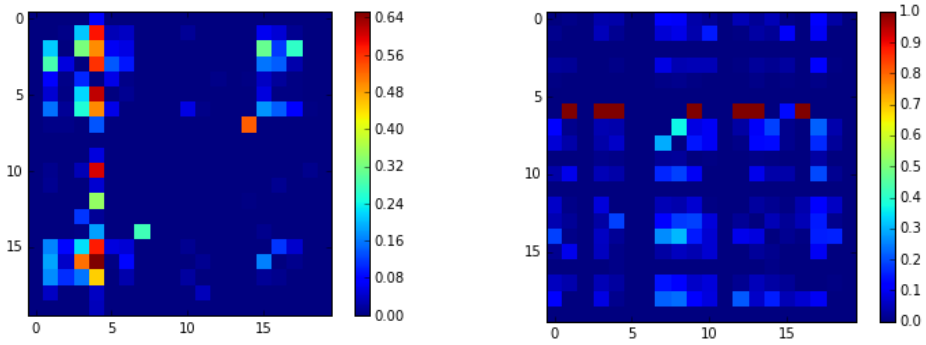


(c) infections/communautés, k = 50

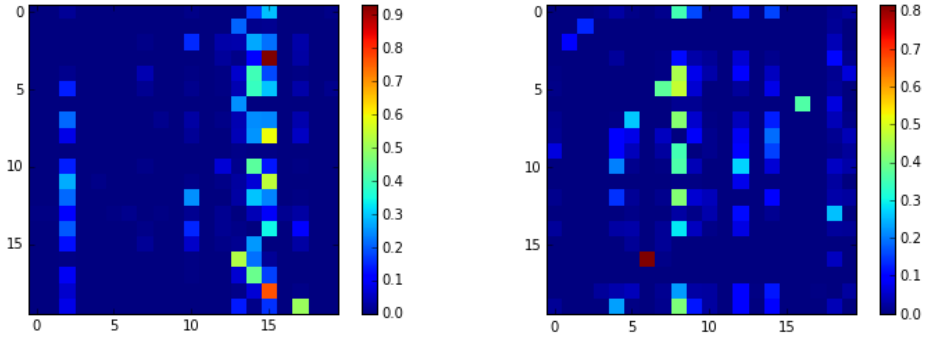


(d) infections/communautés, k = 100

FIGURE 3.5 – Rapport entre le nombre de users par communauté, sur la première ligne, et la propension d'infection par communauté, deuxième ligne, pour μ_3. Les communautés sont représentées par leur numéro en abscisses.



(a) μ_1 (b) μ_2



(c) μ_3 (d) μ_5

FIGURE 3.6 – Matrice des poids appris par IC inter communautés pour k = 20. Les lignes représentent l'influenceur.

La figure 3.1.a. permet de voir que les stratégies μ_1 et μ_2 passent en dessous des performances sans communauté à $k > 300$. Les stratégies μ_3 et μ_5 semblent obtenir des meilleurs performances en prédiction ce qui nous laisse penser qu'elles sont adaptées pour représenter le problème de diffusion.

Un point commun à toutes les stratégies est que leurs performances chutent considérablement entre $10 < k < 100$, puis décroissent lentement. Une de nos intuitions est ici confirmée, pour les stratégies étant représentatives du phénomène de diffusion, plus on grossit l'échelle d'analyse, plus les performances en prédiction augmentent. Cependant, ces performances restent très en dessous de nos espérances, avec un gain de 0.1 en MAP pour μ_3 en passant de 4500 utilisateurs à 500 groupes,soit un faible gain de performances face à la perte d'informations. Néanmoins, n'oublions pas que le fait que $\lambda = 0$ rend le modèle très sensible, ainsi nous pouvons nous attendre à des résultats biaisés pour $10 < k < 100$.

Sur la figure 3.2, nous pouvons voir la variance de la taille des communautés en fonction de k. Nous pouvons voir que peu importe la stratégie, les performances sont très fortements liées à la variance des communautés. Plus les communautés sont inégalitairement réparties, plus les performances augmentent. Si nous étudions plus en détail les k où la performance change de pente, c'est-à-dire entre 100 et 300, nous nous apercevons qu'il y a un réel changement dans la structure des communautés. Sur la figure 3.3 nous voyons bien que pour $k < 100$ la plupart des communautés sont constituées de peu d'utilisateurs, tandis qu'une est constituée d'un grand nombre d'utilisateurs, mais plus on augmente k à partir de $k > 100$ et plus il y a des communautés de tailles similaires. Nous pouvons penser que ceci est la raison de la plus grande performance entre $10 < k < 100$, moins il y a de communautés, moins il y a de probabilités de se tromper dans la prediction mais surtout si une grande communauté regroupe plus de 80% du nombre. Nous pouvons penser que la majeure partie du temps cette communauté est infectée, donc les poids appris par IC en direction de ce modèle sont fort élevés. Cependant, cette intuition s'est révélée fausse sur certaines stratégies, le fait qu'une communauté contienne la majeure partie ne signifie pas que cette communauté intervienne dans le processus.

Si les communautés sont regroupées selon $\phi_{coCascade}$ en prenant un k compris entre 10 et 100, nous avons bien une communauté dominante, mais qu'en

est-il de sa participation aux cascades ? Sur la figure 3.4 nous pouvons voir qu'en effet les communautés dominantes sont aussi celles qui sont le plus infectées, et donc cela biaise les résultats de performances, car l'on ne fait que prédire une infection d'une communauté qui a d'énormes chances d'être infectée et dont l'infection ne dépend que d'une seule personne. Mais ce résultat n'est pas le cas pour la similarité $\phi_{coSuccesseur}$, c'est la figure 3.5 qui prouve le fait qu'une communauté dominante n'est pas forcément la plus infectée. On voit bien que la communauté dominante est rarement présente dans les cascades, et ce sont des comunautés de faible nombre qui sont très présentes. Cela démontre le fait que nos regroupements pour μ_3 et μ_5 ont permis séparer les utilisateurs actifs dans les cascades de ceux qui ne le sont pas vraiment ; en corrélant cette information et la performance, on démontre aussi que ce type de regroupement permet de capter des tendances communautaires exploitables pour prédire la diffusion. Ces résultats confirment aussi ce que [13] démontre, la modélisation sous forme d'influenceurs est un modèle pertinent quant à la diffusion d'informations.

Afin de confirmer nos résultats nous avons réeffectué les tests sur k variant de 10 à 100 en supprimant des cascades les groupes dominants, i.e. supérieur 1000 utilisateurs. Ainsi nous avons un modèle appris avec des cascades dont les groupes étaient existants, mais où ils ne le sont plus en test. Cela permet de déterminer l'importance de ces groupes dans le processus de diffusion. Plus la performance est en baisse par rapport à la version sans suppression des communautés dominantes, plus l'impact de ces communautés sur la diffusion est grande : d'où une signification moindre des performances car $\lambda = 0$. Nous avons effectué ces tests sur les k où il existait effectivement ce phénomène de communauté dominante, c'est à dire $10 < k < 210$. Sur la figure 3., on vois comme prévue que les performances en tests chuttent pour μ_1 quand on n'a pas pris en compte les communautés dominantes. Ces communautés ont donc un impact sur μ_1, et les performances sur les autres communautés sont donc beaucoup moins grandes que ce que l'on pouvait espérer. En revanche les performances en tests pour μ_3 sont semblablent à la version normale. Cela confirme notre hypothèse selon laquelle les communautés dominantes pour $k < 200$ ne jouent pas un rôle important dans la diffusion.

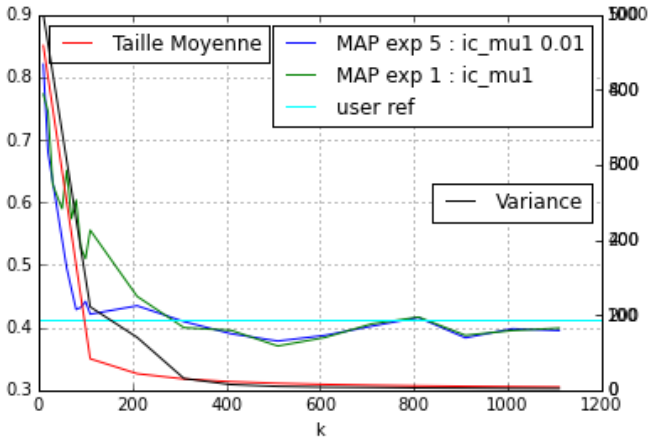
En règle générale, parmis nos 4 stratégies, il s'est avéré que les performances en prédiction étaient plus grandes quand on prenait en compte les arrêtes du graphe social. Ainsi, comme nous pouvons les voir sur 2.5 et 2.6,

les stratégies μ_3 et μ_5 ont tendences à former des petites communautés très actives, contrairement à μ_1 qui ne forme en réalité qu'une communauté active et d'autre peu actives qui au final n'intervienent pas dans le processus. La figure 3.6 représente la matrice des poids appris par IC. Sur cette figure on s'apperçoit que la stratégie μ_2 qui est la moins performante modélise une communauté très influenceur et d'autres qui n'ont aucuns pouvoir d'influence. Les autres stratégies ont toutes des communautés qui influences et d'autres qui se font influencer. Cela favorise grandement la circulation de l'information, c'est en cela que les stratégies μ_3 et μ_5 sont plus performantes.

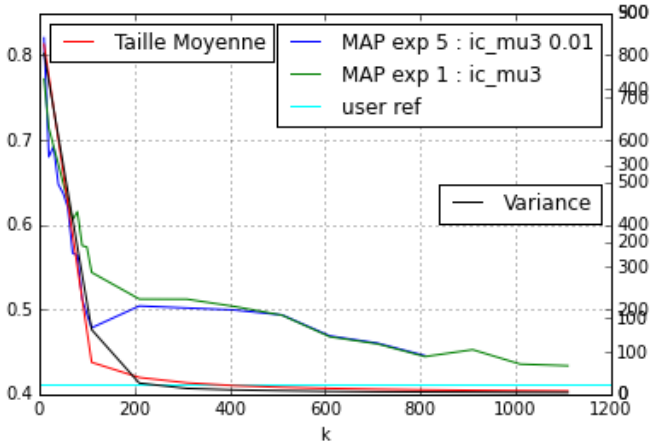
La stratégie μ_2 quant à elle a été un echec, les résultats sont très en dessus de la référence. On peut en conclure que le fait de rassembler des groupes selon leur place dans la diffusion (au début, ou à la fin), donc en fonction de leur capaciter à être influenceur ou influencé, sans prendre en compte le lien social n'est pas un critère représentatif du phénomène de diffusion d'information.

$\lambda = 0.01$

Nous avons poursuivis nos expériences sur μ_1, μ_3 et μ_5 en transformant les cascades selon $\lambda = 0.01$. Il faut donc maintenant 1% des utilisateurs dans une communauté afin de considérer celle-ci infectée. Dans la figure 3.8 nous pouvons voir les nouveaux taux de participations aux cascades des différents groupes. Nous voyons clairement que les groupes dominants ont été évincés des cascades, et plus les groupes sont grands moins ils participent aux cascades. Sur la figure 3.7 nous voyons que les résultats pour toutes les stratégies ont chuttés pour $k < 300$. On voit sur les graphiques que les performances pour $\lambda = 0.01$ dépendent de la variance, en noir, et de la moyenne, en rouge, des tailles des communautés. Les figure 3.8 et 3.9 montrent qu'entre k = 50 et k = 210, il y a une différence pour $\lambda = 0.01$ et $\lambda = 0$; le taux de participation aux cascades est modifié pour k = 50, la propension des groupes à participer est modifiée et c'est ce point qui justifie la baisse de performances, cette propension n'étant pas modifié pour k = 210. Ceci s'explique assez facilement : moins il y a de communautés, plus ces communautés sont grandes, moins il y a de chance de trouver 1% de leurs utilisateurs dans une même cascade, sachant que la taille moyenne d'une cascade est de 3 à 4 participants.

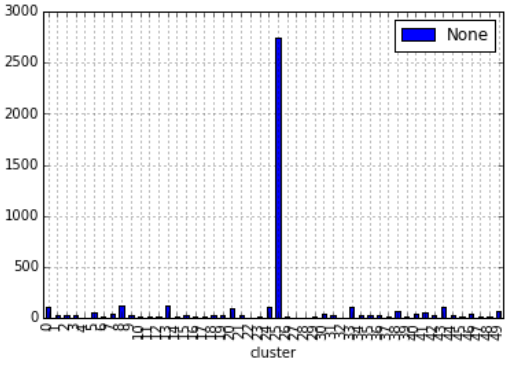


(a) performances MAP μ_1

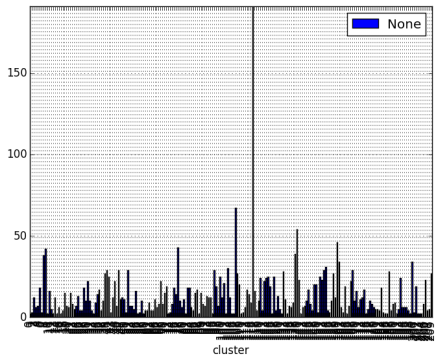


(b) performances MAP μ_3

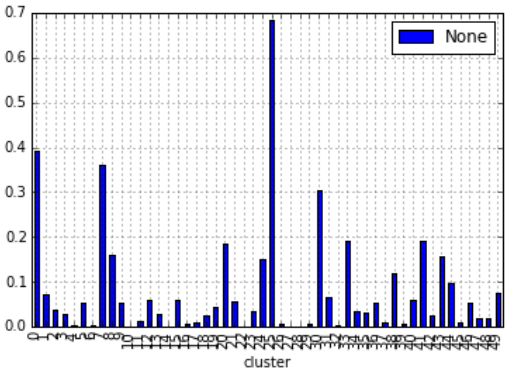
FIGURE 3.7 – Performances en MAP, en fonction des différentes stratégies. User ref représente la performance MAP sans communauté. La variance et la taille moyenne représentent les variances et tailles des communautés en fonction de k. La courbe en bleue est la performance sans λ et celle en verte avec $\lambda = 0.01$



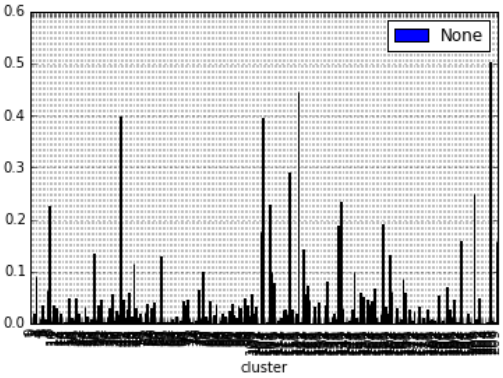
(a) Repartition utilisateur, k = 50



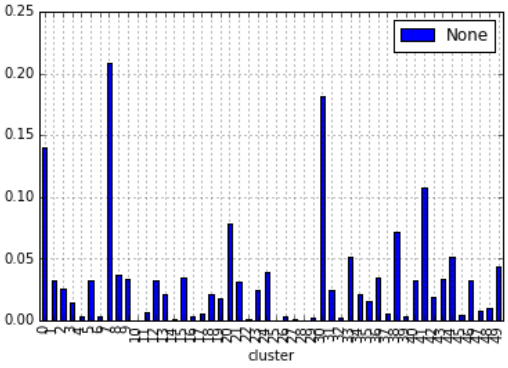
(b) Repartition utilisateur, k = 210



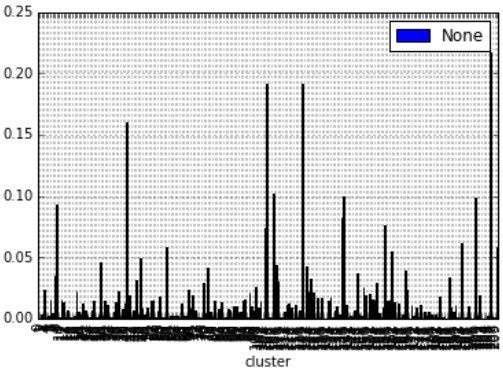
(c) Taux de participation aux cascade



(d) Taux de participation aux cascade

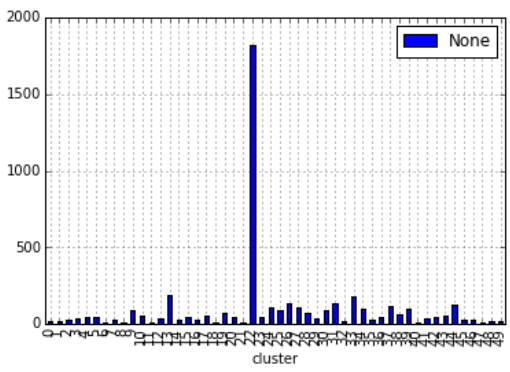


(e) Taux avec λ 0.01

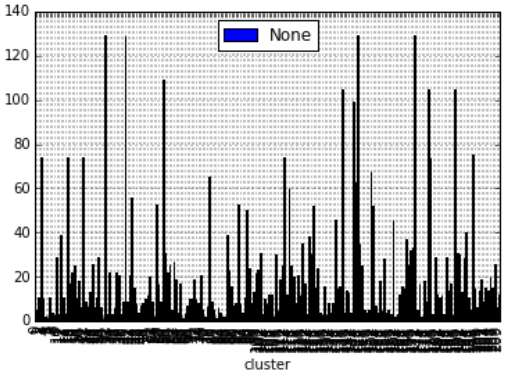


(f) Taux avec λ 0.01

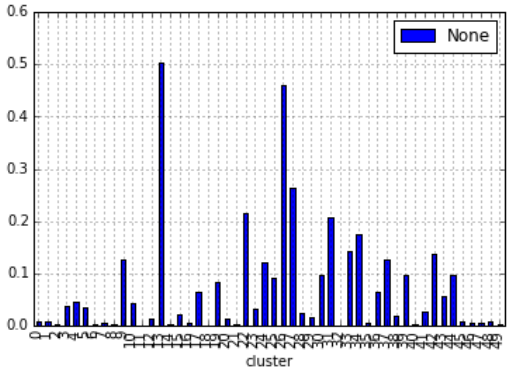
FIGURE 3.8 – Etude comparative sur μ_1 entre le taux de participation aux cascade pour $\lambda = 0$ sur la ligne 2, et $\lambda = 0.01$ sur la ligne 3. Sur la ligne 1 une référence au nombre d'utilisateurs. La colonne de gauche représente k = 50 et la colonne de droite k = 210



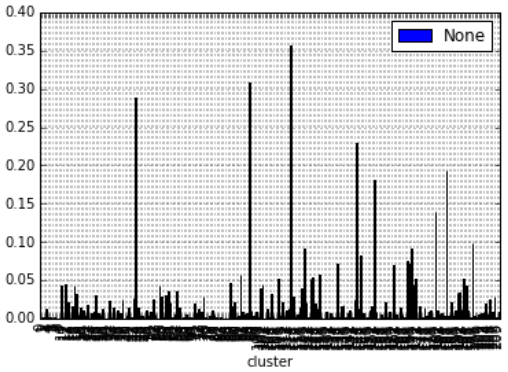
(a) Repartition utilisateur, k = 50



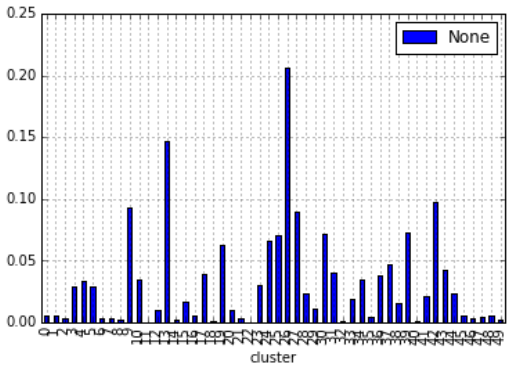
(b) Repartition utilisateur, k = 210



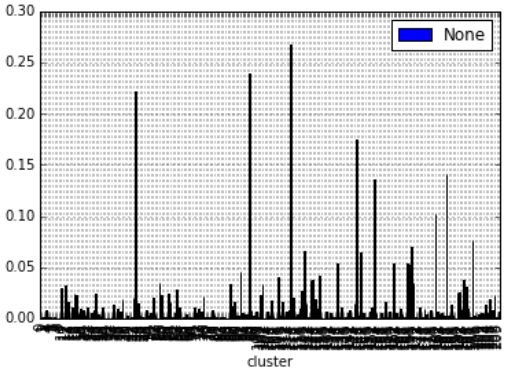
(c) Taux de participation aux cascade



(d) Taux de participation aux cascade



(e) Taux avec λ 0.01



(f) Taux avec λ 0.01

FIGURE 3.9 – Etude comparative sur μ_3 entre le taux de participation aux cascade pour $\lambda = 0$ sur la ligne 2, et $\lambda = 0.01$ sur la ligne 3. Sur la ligne 1 une référence au nombre d'utilisateurs. La colonne de gauche représente k = 50 et la colonne de droite k = 210

Pour une communauté de taille ¿ 200, cela signifie au moins 2 utilisateurs de la communauté dans la cascade, ce qui n'est que très rarement le cas pour beaucoup d'entre elles. En revenche pour une communauté de 50 à 100 utilisateurs, cela signifie au moins 1 utilisateur infecté.

Quand nous étudions la diffusion d'information inter-communautés de notre manière, il devient difficile pour des situations réelles de rencontrer tout une communauté infectée par une même information. Ceci est dû notamment à la taille des cascades de notre dataset. C'est pour cela que notre manière de modéliser une communauté pour un dataset comme celui digg n'est pas adaptée à l'étude d'une infection intra communauté.

3.3 Conclusion des expériences

Notre étude expérimentale a permis de rendre compte de certaines tendances dans la diffusion inter-communautés. Le but de notre démarche peut être défini comme un grossissement de l'échelle d'analyse où l'on ne s'intéresse non plus à l'utilisateur, mais plutôt à un groupe d'utilisateurs. De ce point de vue, une composante majeure dans l'évaluation, et qui déterminera notre angle de recherche, est la définition de contamination d'une communauté. Nous avons vu que l'étude communautaire du dataset digg ne permettait pas d'améliorer les performances en prédiction de diffusion finale, ceci étant dû au fait que les cascades d'informations sont relativement petites. Ainsi, pour des jeux de données où les cascades d'informations sont petites, définir une infection à l'échelle communautaire n'est pas très fructueux. D'une part, pour les communautés de petites tailles, il semblerait que le regroupement en communautés n'apporte aucune ou peu d'informations sur le comportement en diffusion. Il faut donc agrandir les tailles des communautés, et surtout réduire le nombre de ces communautés pour réussir à dégager une certaine tendance dans leur comportement. D'autre part, le fait même d'agrandir les tailles des communautés rend l'analyse plus floue et on perd une grande quantité d'informations.

Là où réside l'intérêt de notre étude, c'est dans la mise en lumière des tendances communautaires dans une diffusion d'informations. Le fait de re-

grouper des utilisateurs selon qu'ils influencent ou sont influencés, a, dans notre cas, formé des groupes peu actifs, qui ont été laissés de côté, et d'autres très actifs, de taille d'environ 60 à 400 utilisateurs, dont l'une des particularités était d'être très reliés entre eux. Ces communautés fortement reliées non seulement permettent de dégager des tendances d'influences, mais aussi favorisent la diffusion d'informations. Le constraste est net entre les expériences faisant intervenir les arrêtes du graphe social, et les autres : les premières sont plus adaptées à l'analyse de la diffusion.

Une utilisation intéressante de ces communautés serait de les utiliser comme un a priori sur la propension d'un utilisateur à être infecté par une information.

Bibliographie

[1] Yong-Yeol Ahn, James P Bagrow, and Sune Lehmann. Link communities reveal multiscale complexity in networks. *Nature*, 466(7307) :761–764, 2010.

[2] Nicola Barbieri, Francesco Bonchi, and Giuseppe Manco. Cascade-based community detection. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '13, pages 33–42. ACM, 2013.

[3] Michele Coscia, Giulio Rossetti, Fosca Giannotti, and Dino Pedreschi. Demon : a local-first discovery method for overlapping communities. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 615–623, 2012.

[4] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3) :75–174, 2010.

[5] Manuel Gomez Rodriguez, Jure Leskovec, and Andreas Krause. Inferring networks of diffusion and influence. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10. ACM, 2010.

[6] M Granovetter. Threshold models of collective behavior. *American Journal of Sociology*, 83(6) :1420, 1978.

[7] D Gruhl, R Guha, David Liben-Nowell, and A Tomkins. Information diffusion in blog space.

[8] Adrien Guille and Hakim Hacid. A predictive model for the temporal dynamics of information diffusion in online social networks. In *Proceedings of the 21st international conference companion on World Wide Web*, WWW '12 Companion. ACM, 2012.

[9] Adrien Guille, Hakim Hacid, Cécile Favre, and Djamel A Zighed. Information diffusion in online social networks : A survey. 42(2) :17–28.

[10] David Kempe, Jon Kleinberg, and Eva Tardos. Maximizing the spread of influence through a social network. 2003.

[11] Cédric Lagnier, Ludovic Denoyer, Eric Gaussier, and Patrick Gallinari. Predicting information diffusion in social networks using content and users profiles. In *European Conference on Information Retrieval*, ECIR '13, 2013.

[12] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297, 1967.

[13] Yasir Mehmood, Nicola Barbieri, Francesco Bonchi, and Antti Ukkonen. Csi : Community-level social influence analysis. In *Machine Learning and Knowledge Discovery in Databases*, pages 48–63. Springer.

[14] Alan Mislove, Massimiliano Marcon, Krishna P Gummadi, Peter Druschel, and Bobby Bhattacharjee. Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, IMC '07, pages 29–42. ACM, 2007.

[15] Anis Najar, Ludovic Denoyer, and Patrick Gallinari. Predicting information diffusion on social networks with partial knowledge. In *Proceedings of the 21st international conference companion on World Wide Web*, WWW '12 Companion, pages 1197–1204. ACM, 2012.

[16] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043) :814–818, 2005.

[17] Kazumi Saito, Masahiro Kimura, Kouzou Ohara, and Hiroshi Motoda. Learning continuous-time information diffusion model for social behavioral data analysis. In *Advances in Machine Learning*, pages 322–337. Springer.

[18] Kazumi Saito, Ryohei Nakano, and Masahiro Kimura. Prediction of information diffusion probabilities for independent cascade model. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5179 LNAI, pages 67–75, 2008.

[19] Kazumi Saito, Kouzou Ohara, Yuki Yamagishi, Masahiro Kimura, and Hiroshi Motoda. Learning diffusion probability based on node attributes in social networks. In Marzena Kryszkiewicz, Henryk Rybinski, Andrzej Skowron, and Zbigniew W Ras, editors, *ISMIS*, volume 6804 of *Lecture Notes in Computer Science*, pages 153–162. Springer, 2011.

[20] Liaoruo Wang, Stefano Ermon, and John E Hopcroft. Feature-enhanced probabilistic models for diffusion network inference. In *Proceedings of the 2012 European conference on Machine Learning and Knowledge Discovery in Databases - Volume Part II*, ECML PKDD'12, pages 499–514. Springer-Verlag, 2012.

[21] Jierui Xie, Stephen Kelley, and Boleslaw K Szymanski. Overlapping community detection in networks : The state-of-the-art and comparative study. 45(4) :43.

[22] Jaewon Yang, Julian McAuley, and Jure Leskovec. Detecting cohesive and 2-mode communities indirected and undirected networks. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, WSDM '14, pages 323–332. ACM, 2014.