SORBONNE UNIVERSITÉ
CRÉATEURS DE FUTURS
DEPUIS 1257

THÈSE DE DOCTORAT DE SORBONNE UNIVERSITÉ

Spécialité Informatique

École doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par
Thibaud ARNOUX

Pour obtenir le grade de

DOCTEUR de SORBONNE UNIVERSITÉ

Prédiction d'interactions dans les flots de liens

Combiner les caractéristiques structurelles et temporelles

Soutenue le 29 Juin 2018 devant le jury composé de :

Rapporteurs :	David Chavalarias	Directeur de recherches, CNRS
	Christophe Crespelle	Maître de conférences, Université Claude Bernard Lyon 1
Examinateurs :	Pierre Borgnat	Directeur de recherches, CNRS
	Anne Fladenmuller	Maître de conférences, Sorbonne Université
	Rushed Kanawati	Maître de conférences, Université Paris 13
Directeurs :	Matthieu Latapy	Directeur de recherches, CNRS
	Lionel Tabourier	Maître de conférences, Sorbonne Université

Table des matières

Introduction **7**

1 État de l'art **11**
1.1 Formulation de la prédiction de liens et de la récupération de liens manquants dans les graphes . . . 12
1.2 Méthodes de résolution . . . 12
1.3 Intégrer l'évolution temporelle . . . 14
1.4 Hybridation avec les séries temporelles . . . 15
1.5 Conclusion . . . 16

2 Positionnement **17**
2.1 Flots de liens . . . 17
2.2 Tâches de prédiction et évaluation . . . 18
2.2.1 Prédire le temps d'apparition de chaque lien . . . 19
2.2.2 Apparition du premier lien pour chaque paire . . . 19
2.2.3 Nombre de liens sur une période donnée . . . 21
2.2.4 Apparition d'au moins un lien sur une période donnée . . . 23
2.3 Envisager les différentes tâches . . . 24
2.3.1 Représenter les caractéristiques du flot . . . 24
2.3.2 Fonction de prédiction . . . 25
2.4 Utilisation des fonctions de prédiction . . . 25
2.4.1 Prédire les temps d'apparition des liens . . . 25
2.4.2 Prédiction du nombre de liens sur une période donnée . . . 26
2.5 Conclusion . . . 27

3 Protocole **29**
3.1 Combinaisons de métriques . . . 31
3.2 Prédiction de l'activité globale . . . 32
3.3 Répartition des liens . . . 33
3.4 Processus d'optimisation . . . 33
3.4.1 Flots d'entraînement, test, observation et prédiction . . . 34
3.4.2 Descente de gradient . . . 34
3.5 Conclusion . . . 35

4 Jeux de données **37**
4.1 Highschool . . . 37
4.2 Infocom . . . 38
4.3 Reality Mining . . . 39
4.4 Taxi . . . 40
4.5 Conclusion . . . 41

5 Métriques **43**
5.1 Normalisation . . . 43
5.2 Métriques de prédiction de liens . . . 44
5.2.1 Nombre de voisins commun . . . 44
5.2.2 Indice de Jaccard . . . 45
5.2.3 Indice de Sørensen . . . 46
5.2.4 Indice d'Adamic-Adar . . . 47
5.2.5 Indice d'allocation de ressource . . . 47
5.3 Métriques structurelles pondérées . . . 47
5.3.1 Nombre de voisins communs pondéré . . . 48
5.3.2 Autres métriques pondérées . . . 49
5.4 Métriques temporelles . . . 49
5.4.1 Extrapolation de l'activité . . . 49
5.4.2 Temps inter-contacts et évolution de l'activité . . . 53
5.5 Corrélations . . . 55
5.6 Conclusion . . . 57

6 Expérimentations **59**
6.1 Résultats expérimentaux . . . 60
6.1.1 Qualité de la prédiction . . . 60
6.1.2 Étude de la combinaison de métriques . . . 63
6.1.3 Cas des prédictions de faibles qualités . . . 67
6.2 Influence de la prédiction de l'activité globale . . . 70
6.3 Étude de la combinaison de paires de métriques . . . 71
6.3.1 Protocole expérimental simplifié . . . 71
6.3.2 Combinaisons de métriques temporelles et structurelles . . . 72
6.3.3 Nature des liens prédits . . . 74
6.4 Conclusion . . . 76

7 Classes **77**
7.1 Différentes approches des classes et catégories de paires de nœuds . . . 77
7.2 Réduction de l'espace des paramètres . . . 78
7.3 Classes par seuil d'activité . . . 80
7.3.1 Définition des classes . . . 80
7.3.2 Estimateur de qualité . . . 80
7.3.3 Résultats expérimentaux . . . 81
7.3.4 Résumé . . . 87

7.4 Méthode de clustering 88
7.4.1 Définition des classes 89
7.4.2 Résultats expérimentaux 92
7.5 Conclusion 98

Conclusion **101**

A Distributions cumulatives des métriques **103**

B Classes – Résultats détaillés **111**
B.1 Classes par seuils 111
B.1.1 Highschool 111
B.1.2 Infocom 113
B.1.3 Reality Mining 114
B.1.4 Taxi 116
B.2 Clustering – Diviser les clusters dans l'ordre inverse de leurs agrégations . 118
B.3 Clustering – Diviser les clusters suivant leur taille 120

C Documentation du code **123**
C.1 Getting Started 123
C.1.1 Pré-requis 123
C.1.2 Configuration par défaut 123
C.1.3 Structure des données 124
C.2 Lancer la prédiction 124
C.3 Output 125
C.4 Autres réglages 125

Introduction

De nombreux systèmes sont caractérisés par les interactions ayant lieu entre leurs éléments. Il peut s'agir de contacts entre individus, de la structure de sites internet, de comportements d'utilisateurs sur des réseaux sociaux ou des sites commerciaux, d'achats en ligne voire de systèmes physiques ou biologiques. Dans chaque cas, la structure des interactions a été étudiée pour identifier les caractéristiques de tels systèmes afin de comprendre leur fonctionnement et les relations des différents éléments les composant. De nombreux systèmes évoluent au cours du temps et les mécanismes derrière la création et la disparition d'interactions dans les réseaux ont été particulièrement étudiés [LBKT08]. Cette compréhension du fonctionnement des réseaux a permis de développer des méthodes visant à anticiper le comportement du système. En capturant les caractéristiques importantes du réseau, ces méthodes permettent de déterminer les éléments les plus susceptibles d'interagir dans le futur.

La prédiction de liens est un domaine de recherche très actif ces dernières années. Elle est au centre de nombreuses applications, dans des domaines très divers comme la sécurité, les protocoles de communication ou la gestion de réseaux de machines. Elle conduit dans ces systèmes à développer des algorithmes pour différentes tâches comme la recommandation ou la détection d'événements. Par exemple, elle intervient dans le cadre des réseaux tolérants aux délais [HSL10], dont les connexions entre machines évoluent au cours du temps. Elle a également des applications dans d'autres disciplines scientifiques, comme la biologie où elle a été utilisée pour prédire les interactions entre protéines [ABF+06] ou les connexions entre neurones [CALR13]. Elle a été particulièrement étudiée dans le domaine des réseaux sociaux où elle trouve de nombreuses applications comme des stratégies de recommandation pour les sites commerciaux [LSY03] ou la récupération de liens dans des données incomplètes [ZLZ09]. On la retrouve également dans les systèmes de recommandation des réseaux sociaux tels que Facebook [DTW+12] ou LinkedIn [XNR10] où elle permet de proposer à l'utilisateur de nouveaux contacts basés sur la structure du réseau et le profil des individus.

Ces réseaux ont été étudiés sous forme de graphes depuis longtemps. Cependant l'importance de la dynamique est devenue de plus en plus évidente [HS12]. D'autres représentations ont donc été développées pour mieux capturer l'évolution temporelle de ces systèmes. L'étude de réseaux dynamiques se fait souvent en agrégeant l'information au sein de fenêtres temporelles, engendrant une perte d'information. Le formalisme des flots

de liens est une approche permettant de conserver la dynamique du système tout en fournissant un cadre d'étude solide pour appréhender le comportement du système. Un flot de liens est une série de triplets (t, uv) indiquant qu'une interaction a eu lieu entre u et v au temps t. De nombreux jeux de données réels peuvent êtres modélisés et analysés en utilisant les flots de liens, comme des échanges d'emails, des appels téléphoniques ou du trafic IP [LVM17a, VL14, VFSML17]. Nous développons un formalisme de prédiction visant à utiliser l'information capturée par les flots de liens, à la fois structurelle et temporelle. Notre objectif est de définir un cadre de prédiction général permettant des études approfondies de la relation entre caractéristiques structurelles et temporelles dans les tâches de prédiction. Des expériences menées sur différents jeux de données nous permettront d'identifier les éléments cruciaux affectant la qualité de la prédiction et des pistes de réflexion pour aborder ces problèmes.

Dans l'optique de développer un formalisme de prédiction dans le cadre des flots de liens, il est important de conduire une réflexion sur les différentes approches possibles du problème. En effet, l'importance de la dynamique du système dans la prédiction dans les flots de liens la place au carrefour de plusieurs champs de recherches. Par construction, les flots de liens capturent l'information structurelle des systèmes, rapprochant la prédiction dans ce cadre de la prédiction de liens dans les graphes. D'autre part, l'importance donnée par ce formalisme à la dynamique du système nous incite à nous rapprocher du domaine de la prédiction de séries temporelles, dont les outils sont plus adaptés à capturer l'évolution des caractéristiques du système au cours du temps.

Nous passerons en revue au Chapitre 1 différentes approches développées dans la littérature concernant la prédiction de réseaux dynamiques, notamment les différentes formalisations utilisées pour représenter l'information temporelle au sein de protocoles de prédiction. Nous étudierons également les travaux combinant la prédiction de liens dans les graphes et la prédiction de séries temporelles.

Nous allons ensuite explorer différentes formalisations du problème de la prédiction dans les flots de liens au Chapitre 2. En effet, différentes tâches de prédiction sont possibles, de la prédiction des temps d'apparition de chaque lien dans le futur à la prédiction du graphe agrégé issue du flot prédit, les nœuds étant liés s'ils ont interagi au moins une fois. Dans chaque cas envisagé nous proposerons une approche possible pour évaluer la prédiction. Nous introduirons également des outils centraux de notre protocole, les fonctions de prédiction, et proposerons des protocoles de prédiction pour chaque tâche.

Dans la suite nous choisirons de développer une de ces approches et nous intéresserons à la prédiction de l'activité, c'est-à-dire prédire le nombre d'interactions apparaissant dans le futur pour chaque paire de nœuds durant une certaine période. Ce problème partage certaines caractéristiques avec le problème plus courant de la prédiction de liens. Il en diffère par le fait que nous prédisons non seulement qui interagira avec qui dans le futur mais également combien de fois.

Au Chapitre 3 nous introduirons le protocole développé, permettant de combiner de manière cohérente les caractéristiques des données afin d'effectuer la prédiction de l'activité.

Nous présentons également la méthode d'apprentissage supervisé optimisant la combinaison de métriques.

Afin d'évaluer la qualité de la prédiction nous présenterons au Chapitre 4 les jeux de données étudiés ainsi que certaines de leurs caractéristiques qui nous seront utiles pour comprendre le fonctionnement de notre protocole. Nous décrirons au Chapitre 5 des métriques structurelles issues des méthodes de prédiction de liens ainsi que des métriques temporelles permettant de capturer la dynamique des interactions des paires de nœuds.

Nous étudierons ensuite au Chapitre 6 le comportement de notre protocole sur diverses expériences sur les jeux de données présentés et évaluerons la qualité de chaque prédiction afin de déterminer les points forts de notre protocole ainsi que les défis à relever pour améliorer la prédiction. Nous nous appliquerons notamment à déterminer la nature des liens prédits par notre algorithme de prédiction ainsi que le rôle du choix de la combinaison de métriques dans la prédiction.

Observant que notre approche favorise la prédiction de liens spécifiques nous introduisons au Chapitre 7 des classes de nœuds aux comportements différents afin d'améliorer la prédiction. Nous étudierons comment l'utilisation de classes de nœuds permet de préserver la diversité dans le type de liens prédits tout en améliorant la prédiction. Différentes méthodes de définition seront explorées, permettant d'isoler des classes de paires de nœuds suivant différents critères notamment en utilisant des méthodes de clustering adaptées à notre problème.

Chapitre 1

État de l'art

Les problèmes de prédiction de liens se basent sur la modélisation des interactions entre les différents éléments des réseaux étudiés. De nombreux travaux ont étudié les mécanismes déterminant l'évolution de tels réseaux au cours du temps, ayant permis de mettre en évidence les caractéristiques temporelles des réseaux étudiés [PGPSV12], notamment ceux représentant des contacts entre individus. Ces études ont par exemple montré que les interactions entre individus ont lieu en rafales où de nombreux liens apparaissent à la suite [SBB10]. D'autres études se sont intéressées à l'évolution des caractéristiques structurelles du réseau au cours du temps, les apparitions et disparitions de ponts entre communautés par exemple, et comment les interactions changeantes des individus permettent tout de même une certaine stabilité de la structure au niveau global [KW06]. Ces études ont montré l'importance de l'étude de la dynamique des réseaux dans la prédiction de liens et différentes approches ont été développées afin de prendre en compte la temporalité des interactions.

La prédiction de l'activité, le nombre d'interactions ayant lieu entre chaque paire de nœuds, est au croisement de deux problèmes classiques dans la littérature, la prédiction de liens et la prédiction de séries temporelles, illustrées en Figure 1.1. Tandis que l'objectif de la prédiction de liens est de prédire les futures interactions des différents nœuds du réseau, la prédiction de séries temporelles vise à prévoir l'évolution d'une suites de valeurs au cours du temps. Nous présentons différentes approches de prédiction de liens permettant la prise

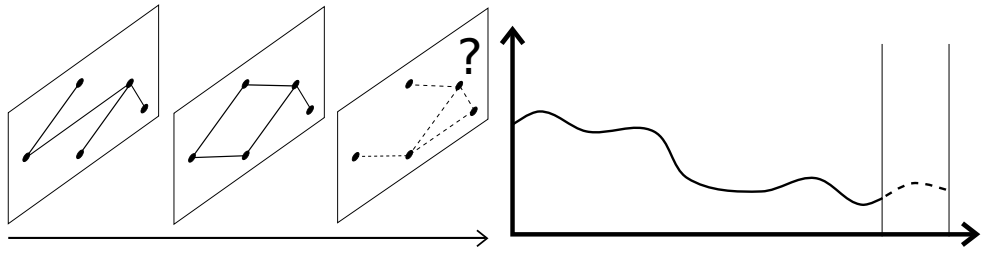


FIGURE 1.1 – Principe de base de la prédiction dans des graphes dynamiques (gauche) et dans les séries temporelles (droite).

en compte de la dynamique du réseau dans la prédiction. Nous explorons ensuite certaines études ayant adoptées une approche liant prédiction de liens et séries temporelles.

1.1 Formulation de la prédiction de liens et de la récupération de liens manquants dans les graphes

La prédiction de liens vise à déterminer les futures interactions du réseau. Dans sa formulation classique de Liben-Nowell et Kleinberg [LNK07], les données sont réparties en deux graphes représentant les interactions ayant eu lieu durant deux périodes de temps successives, la période d'entraînement et la période de test.

L'objectif est d'utiliser les caractéristiques des interactions de la période d'entraînement pour déterminer les nouveaux liens apparaissant durant la période de test. Pour cela, on utilise des mesures de similarités basées sur la topologie du graphe, permettant d'identifier des paires de nœuds susceptibles d'interagir. Dans ce domaine de nombreuses métriques ont été développées afin d'obtenir les informations les plus pertinentes pour la prédiction [AHCSZ06, LZ11]. Des mesures utilisant le voisinage local des paires de nœuds telles que le nombre de voisins communs entre deux nœuds et plusieurs variantes sont couramment utilisées. Il est également possible d'utiliser des métriques se basant sur la notion de plus court chemin, prenant donc en compte l'ensemble du réseau telles que Katz [Kat94] utilisant le nombre de chemins entre deux nœuds, pondérés par leur longueur ou *Hitting time* basé sur des marches aléatoires au sein du réseau. Ces métriques permettent alors de classer les paires de nœuds les plus susceptibles d'interagir durant la période de test. La prédiction s'effectue en sélectionnant les n paires ayant le meilleur score avec, dans le cas des travaux de Liben-Nowell et Kleinberg, n étant choisi comme le nombre de nouveaux liens apparus durant la période de test.

Cette tâche de prédiction est proche de la récupération de liens manquant dans les graphes [KL11, CMN08]. Lorsque seuls certains liens du réseau ont été capturés, la tâche consiste à prédire les liens n'ayant pas été observés afin de reconstituer le réseau réel. Ces problèmes sont formellement comparables et peuvent être résolus avec des méthodes très similaires. Dans la suite nous nous concentrerons plus particulièrement sur le problème de la prédiction de liens.

1.2 Méthodes de résolution du problème de la prédiction de liens

Comme dit précédemment, les données sont représentées sous la forme d'un graphe et la tâche consiste à prédire les liens les plus susceptibles d'apparaître dans ce graphe dans le futur. Ce problème ayant fait l'objet de nombreux travaux, de multiples approches ont

été résumées dans différents articles de synthèse [HLC05, WXWZ15, AHZ11]. Nous allons présenter brièvement ici certains des travaux classiques dans ce domaine.

Le réseau observé est représenté sous la forme d'un graphe $G = (V, E)$, où V représente les nœuds du réseau et E l'ensemble des liens, agrégeant les interactions ayant lieu durant la période d'entraînement. La tâche consiste alors à prédire un nouveau graphe $G' = (V', E')$ représentant les interactions durant la période de test à partir des caractéristiques du graphe G. Si de nombreuse méthodes se concentrent sur l'apparition de nouveaux liens dans réseau, considérant que les liens déjà apparus dans le graphe G seront encore présents dans le futur, certaines études se penchent également sur la récurrence des liens [SAS12] ou encore sur la disparition de ces liens [LBKT08, RLHC11].

Dans les études récentes, ce problème est communément vu comme une tâche de classification binaire, où la prédiction consiste à déterminer, pour chaque lien potentiel, son apparition ou non [AHCSZ06, DLC13] à partir des valeurs des différentes métriques utilisées. D'autres approches utilisent toutefois le classement des paires de nœuds en fonction du score des métriques au sein de méthodes de classification [PK12].

Tandis que de nombreuses méthodes classiques prennent la forme de classification non-supervisée, ces dernières années de nombreuse méthodes utilisant des algorithmes d'apprentissage supervisés ont été développées. Lichtenwalter et al. [LLC10] divisent par exemple la période d'entraînement en deux sous périodes. Un algorithme de classification supervisé est alors entrainé à prédire la deuxième sous période à partir des caractéristiques de la première. La prédiction est ensuite effectuée sur la période de test. Cela permet au protocole de prédiction d'adapter le poids de chaque métrique utilisée dans la prédiction à l'importance de la caractéristique qu'elle capture dans le comportement du système.

Cependant, contrairement à d'autres tâches de classification, le problème de la prédiction de liens présente des difficultés spécifiques. Le déséquilibre entre les classes est par exemple un problème classique. En effet la plupart des réseaux étudiés étant creux le nombre de liens apparaissant est faible comparé au nombre de liens possibles dans l'ensemble du réseau. Cela conduit à une tâche de classification significativement plus difficile que dans d'autres domaines. Nous reviendrons au Chapitre 7 sur certaines approches développées pour atténuer ce problème.

D'autres méthodes de prédiction ont été développées, basées sur des informations additionnelles sur les nœuds ou liens du graphe, par exemple, dans le cas de réseau de citations entre des articles scientifiques, les auteurs ou le journal ou la conférence dans lequel l'article a été publié [PU03] ou encore les similarités entre articles [BL11]. Ce type d'approche est également particulièrement adapté à l'étude de réseaux sociaux tels que Twitter [BFDD14] ou Facebook [BL11], où les données peuvent fournir plus d'informations que la structure du réseau, comme l'âge ou les relations personnelles ou familiales des individus.

1.3 Intégrer l'évolution temporelle

Lorsque l'évolution temporelle est un élément important des données, plusieurs approches ont été développées. Une approche possible est de répartir les données en plusieurs fenêtres temporelles T_i, puis agréger les données contenues dans celles-ci en une suite de graphes $G_i = (V, E_i)$ avec $E_i = \{(u, v) : \exists(t, u, v) \in E,\ t \in T_i\}$. Ces fenêtres temporelles permettent d'utiliser les méthodes de prédiction de liens classiques sur cette suite de graphes. Les informations contenues dans les données sont alors extraites en utilisant des mesures basées sur les graphes.

Plutôt qu'utiliser un ensemble de métriques prédéfini pour capturer l'information présente dans le réseau, certaines approches se basent sur des algorithmes d'apprentissage pour apprendre les métriques elles-mêmes afin de prédire l'apparition de liens dans le futur. Utilisant des fenêtres temporelles, Rahman et al [RSH$^+$18] condensent en un seul vecteur les informations structurelles et temporelles des données relatives à chaque paire de nœuds. Un algorithme d'apprentissage optimise alors une opération matricielle compressant chaque vecteur en un vecteur caractéristique de dimension inférieure dans le but de limiter la perte d'information lors de la compression. Les vecteurs caractéristiques sont alors utilisés par un algorithme de classification afin d'effectuer la prédiction. Cela permet ainsi de combiner l'information structurelle et temporelle des données.

Une autre approche de la prédiction de liens dans les systèmes variants au cours du temps consiste à agréger l'information temporelle au sein du système en attribuant des poids aux liens, basés sur les interactions précédentes. C'est notamment le cas dans les travaux de Muramata et Moriyasu [MM07], qui définissent des variantes pondérées de différentes mesures de similarité classiques. Tabourier et al. [TLL16] définissent également différentes métriques pondérées basées sur les caractéristiques d'appels téléphoniques, leur nombre ou leur durée afin de prédire les interactions futures dans des réseaux égo-centré. Ils définissent également des métriques permettant de capturer, par exemple, l'évolution du nombre d'appels au cours de la période d'entraînement. Ces procédés permettent de capturer l'information structurelle ainsi que d'utiliser indirectement l'information temporelle dans les données.

Ce procédé est particulièrement bien illustré par Scholz et al. [SAS12] où les données sont modélisées par le graphe $G < t$ représentant toutes les interactions ayant eu lieu avant le temps t, chaque lien étant pondéré par la durée du contact entre les individus. Ils définissent ensuite des variantes pondérées de mesures couramment utilisées afin de prédire la suite des interactions.

Dunlavy et Kolda [DKA11] explorent quant à eux une méthode pour agréger l'information de plusieurs fenêtres temporelles en une matrice de relation pondérée en donnant plus d'influences aux interactions récentes. Ils définissent également des approximations de la métrique Katz adaptée aux grands graphes.

Tylenda et al. [TAB09], explorent également l'utilisation de métriques pondérées dans des réseaux de collaborations scientifiques. Dans cette approche les poids ne sont pas

seulement en fonction du nombre de collaborations entre deux auteurs mais également suivant le temps écoulé depuis leur dernière collaboration et le nombre d'auteurs de chaque publication. Ils introduisent ainsi plusieurs extensions de métriques classiques permettant de prendre en compte ces différents poids. Ils parviennent ainsi à prédire les collaborations futures en prenant en compte l'aspect temporel des réseaux étudiés.

Par exemple, O'Madadhain et al. [OHS05] utilisent un algorithme de classement basé sur l'utilisation itérative de chaque événement dans le système. Cette étude se penchant sur des échanges d'emails, les événements sont dirigés d'un envoyeur vers un ensemble de nœuds. Toutes les paires de nœuds étant initialisées avec un potentiel de co-participation à un événement futur identique, l'algorithme itère ensuite sur chaque événement dans l'ordre chronologique. Le potentiel de chaque paire de nœuds impliquée est actualisé à chaque étape. L'algorithme s'arrête lors du traitement du dernier événement et les paires de nœuds sont alors classées par ordre de potentiel. La manière d'actualiser le potentiel de chaque nœud impliqué dans un événement donné peut dépendre de nombreux facteurs, tel que le temps depuis le dernier message reçu par l'envoyeur ou le dernier message que celui-ci a envoyé au moment de l'événement en question. Cette méthode permet à l'algorithme de prendre compte la dynamique des échanges d'emails avec précision.

1.4 Hybridation avec les séries temporelles

L'analyse des séries temporelles est un domaine de recherche actif. Une série temporelle est une séquence de valeurs évoluant au cours du temps. De nombreux travaux ont été faits concernant la prédiction de l'évolution de tels objets [Cha00]. Étant donnée une suite de valeurs $x_1, x_2, \ldots, x_N$, l'objectif est alors de prédire les valeurs x_{N+h}. Plusieurs études se sont penchées sur l'utilisation de la prédiction de séries temporelles dans les problèmes de prédiction de liens dans les graphes. En effet en faisant abstraction de la structure du réseau, la variation de nombreuses caractéristiques du système au cours du temps peut être vue comme des séries temporelles.

Par exemple, il est possible de se concentrer sur la fréquence d'apparition des liens dans le passé afin de prédire les futures interactions. Cette approche permet de prédire les apparitions futures de liens ayant été observés précédemment. C'est également la principale limitation de cette approche, puisqu'elle ne permet pas à elle seule de prédire l'apparition de nouveaux liens. La prédiction de séries temporelles est donc complémentaire de la prédiction de liens dans les graphes.

Plusieurs travaux ont étudié des approches permettant de relier ces deux domaines. Par exemple, Huang et Lin [HL09] adoptent une approche hybride permettant de combiner la prédiction de la fréquence d'interaction de chaque paire de nœuds grâce à des outils de séries temporelles et des mesures structurelles. Dans cette étude les données sont modélisées comme une série de graphes pondérés par le nombre d'interactions entre chaque paire durant chaque fenêtre temporelle. Un algorithme de prédiction classique est alors utilisé pour établir le score de chaque paire de nœuds, basé sur les métriques structurelles.

D'autre part, les auteurs utilisent un modèle répandu de prévision de série temporelles sur la suite des fréquences d'interactions de chaque paire de nœuds afin d'obtenir un score basé sur l'évolution des interactions. Les deux scores sont alors combinés afin d'obtenir une prédiction prenant en compte à la fois les informations temporelles et structurelles.

Une autre approche, adoptée par da Silva Soares et Prudêncio, est de se concentrer sur l'évolution des métriques structurelles au cours des différentes fenêtres temporelles et de modéliser celles-ci comme des séries temporelles [dSSP12]. La prédiction est alors effectuée grâce aux valeurs prédites de ces métriques en utilisant des méthodes de prédiction de liens classiques.

1.5 Conclusion

De nombreuses études ont été faites afin d'améliorer les méthodes de prédiction de liens et plus particulièrement pour permettre à ces méthodes de pendre mieux en compte la dynamique des réseaux évoluant au cours du temps. La plupart des études utilisent des fenêtres temporelles pour représenter l'évolution du réseau ou des méthodes de pondération agrégeant l'information temporelle afin de la rendre accessible à des méthodes de prédiction classiques.

Certains travaux se sont également attachés à développer des méthodes permettant de combiner des méthodes de prédiction de liens à des méthodes de prédiction de séries temporelles afin de mieux prédire l'évolution des interactions ou des caractéristiques du réseau.

Nous introduisons un protocole qui combine ces différentes sources d'information de manière cohérentes, tout en évitant la perte d'information due à l'utilisation de fenêtres temporelles ou de méthodes de pondération. Nous l'utilisons pour prédire les nouveaux liens ainsi que les répétitions de liens dans le flot.

Chapitre 2

Positionnement

Dans ce chapitre, nous introduisons le formalisme des flots de liens ainsi que les différentes approches envisagées concernant la prédiction d'interactions dans le cadre de ce formalisme. Nous discuterons des différentes options possibles et introduirons certains outils qui seront nécessaires par la suite. Finalement nous présenterons l'approche retenue et proposerons une définition formelle de celle-ci.

2.1 Flots de liens

Les réseaux dynamiques sont très étudiés et peuvent être représentés de multiples façons. Des formalismes ont été développés afin de mieux représenter les caractéristiques temporelles des données, évitant la perte d'information due à l'utilisation de fenêtres temporelles ou de graphes pondérés. C'est le cas des *temporal networks* [HS12], pour lesquels de nombreux travaux se sont attachés à étudier et formaliser cette approche [Hol15, BP16]. D'autres formalismes ont été développés comme les *time-varying graphs* [CFQS12] permettant de représenter ces caractéristiques. Puisqu'ils représentent le réseau sans perte, l'information qu'ils contiennent est équivalente. Ils correspondent à des approches différentes du même problème : représenter des réseaux évoluant au cours du temps.

Les flots de liens sont un autre formalisme visant à décrire les interactions dans des systèmes dynamiques [LVM17b]. Un flot de liens (voir Figure 2.1) est une séquence de triplets (t, uv), chaque triplet indiquant qu'une interaction entre u et v a eu lieu au temps t. Dans de tels flots de liens, la dynamique habituelle du réseau, comme des interactions quotidiennes entre collègues ou des transferts d'argent par exemple, est souvent mêlée à des dynamiques plus spécifiques comme des débats, des vacances, des attaques ou des fraudes. Dans ce contexte, la prédiction de liens vise à prédire de futures interactions en se basant sur les interactions précédentes, permettant d'anticiper l'évolution du système.

Plus formellement, un flot de liens est un triplet (T, V, E), où $T = [A, \Omega]$ est un intervalle

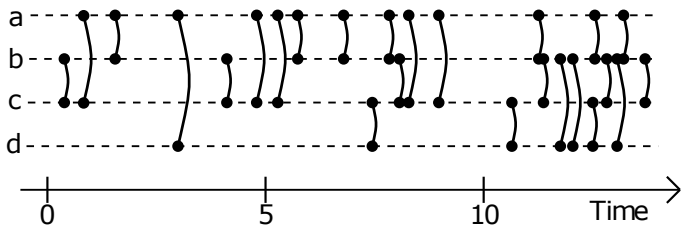


FIGURE 2.1 – Représentation schématique d'un flot de liens où les nœuds b et c ont interagi au temps 0.5, les nœuds a et c ont interagi au temps 1 et ainsi de suite.

de temps, V un ensemble de nœuds et $E \subseteq T \times V \otimes V$ est un ensemble de liens. $(t, uv) \in E$ signifie que les nœuds u et v ont interagi au temps t, où $V \otimes V$ désigne l'ensemble des paires d'éléments distincts non ordonnées de V.

Ce formalisme capture à la fois la structure et la dynamique des interactions. Notons en effet que la représentation classique en graphe peut être déduite du flot de liens, en considérant l'ensemble $\hat{E} = \{uv, \exists(t, uv) \in E)\}$ et le graphe induit $G = (V, \hat{E})$. De la même façon, on peut considérer la série temporelle décrivant l'ensemble du flot $f(t) = |\{uv, (t, uv) \in E\}|$.

2.2 Tâches de prédiction et évaluation

La prédiction d'interactions dans les flots de liens se trouve donc à mi-chemin entre la prédiction de liens dans les graphes et la prédiction de séries temporelles. L'étendue de ces deux domaines nous conduit à nous questionner sur les différentes approches possibles pour envisager la prédiction et à développer des méthodes adaptées pour manipuler les outils venant de ces différents domaines.

Dans chaque cas nous supposerons que l'information disponible est modélisée sous la forme d'un flot de liens $L = (T, V, E)$, où $E \subseteq T \times V \otimes V$ et $T = [A, \Omega]$. Dans la suite de ce manuscrit, nous nous référerons à L et T comme le *flot d'entrée* et la *période d'entrée* respectivement. La prédiction est associée avec le *flot de prédiction* $L_p = (T_p, V_p, E_p)$, où $E_p \subseteq T_p \times V_p \otimes V_p$ et $T_p = [A', \Omega']$, que l'on appellera *période de prédiction*, avec $\Omega \leq A' < \Omega'$. Nous distinguons ce flot du *flot réel*, que nous utiliserons pour évaluer la qualité de la prédiction, $L' = (T', V', E')$, où $E' \subseteq T' \times V' \otimes V'$ et nous considérons $T' = T_p = [A', \Omega']$.

Nous allons également définir un autre aspect du cadre de la prédiction : nos approches n'ont pas pour objectif de prédire l'apparition de nouveaux nœuds dans le système mais uniquement le comportement des nœuds observés. Nous supposerons donc que $V = V' = V_p$.

Nous allons ici présenter différentes options possibles pour définir la prédiction d'interactions dans les flots de liens puis considérer les premiers outils envisagés pour mettre

en place un protocole de prédiction. Au centre de la définition du problème se trouve la détermination de chacun des objets prédits.

2.2.1 Prédire le temps d'apparition de chaque lien

Cela correspondrait à l'approche la plus complète et, *a priori*, permettrait d'obtenir la prédiction la plus ambitieuse.

2.2.1.1 Formalisation

L'objectif est de prédire chaque lien apparaissant, c'est-à-dire le flot de liens L' dans son intégralité. Plus précisément, nous voulons prédire l'ensemble des liens $(t, uv) \in E_p$ tels que $E_p = E'$.

2.2.1.2 Évaluation

Afin d'évaluer la qualité d'une telle prédiction, une mesure de la distance entre L_p et L' est nécessaire. Différentes mesures de distance entre deux graphes ont été proposées dans la littérature [GXTL10] et peuvent être adaptées aux flots de liens. Nous allons ici considérer une adaptation de la distance d'édition de graphe au flot de liens, où une modification de la suite des interactions d'une paire de nœuds du flot est l'équivalent d'une modification d'un lien dans un graphe.

Les interactions entre chaque paire de nœuds peuvent être vues comme une suite de points. Une distance entre deux suites de points peut donc être utilisée afin d'évaluer la prédiction. Nous proposons par exemple d'utiliser la *spike time distance*, définie par Victor et Purpura [VP96]. Celle-ci fait intervenir des transformations du flot par le biais de deux étapes élémentaires : l'ajout ou le retrait d'un lien, ayant un coût 1, et le déplacement d'un lien d'un temps t_1 à un temps t_2, ayant un coût $q \cdot |t_2 - t_1|$ avec q un paramètre déterminant les couts relatifs de chaque étape. La distance entre les flots L_p et L' est ensuite définie comme le coût minimal nécessaire pour passer d'un flot à l'autre. Nous reviendrons plus en détail sur cette distance au Chapitre 7.

2.2.2 Apparition du premier lien pour chaque paire

Cette approche est une simplification du problème précédent, envisagée comme première étape avant un problème plus complexe. L'objectif est ici de prédire la prochaine interaction entre chaque paire de nœuds, s'il y en a effectivement une. Cette approche a l'avantage d'éviter un point délicat, le nombre de liens prédits pour chaque paire de nœuds. Bien qu'apportant moins d'information, il peut suffire dans certaines applications d'identifier les prochaines paires de nœuds actives. Par exemple, dans le contexte de la

diffusion d'information, l'utilisateur s'intéresse principalement au temps de la prochaine interaction, afin de diffuser l'information le plus rapidement possible.

2.2.2.1 Formalisation

Cette tâche consiste donc à déterminer un ensemble de temps $\{t_{uv}, uv \in V \otimes V\}$ pour les paires de nœuds dont nous prédisons l'apparition d'une interaction telles que chaque t_{uv} corresponde au temps d'apparition du premier lien de chaque paire : $\{t_{uv}, uv \in V \otimes V\} = \{min_{(t,uv)\in E'}(t), uv \in V \otimes V\}$.

2.2.2.2 Évaluation

Concernant l'évaluation de la qualité de la prédiction, les distances dans les processus temporels, vues précédemment, peuvent être également utilisées. L'application de la *spike time distance* est alors plus simple, considérant qu'une seule étape est nécessaire pour chacune des paires de nœuds. En définissant un lien non prédit comme prédit à un temps infini et un lien non apparu comme apparu à un temps infini, nous pouvons alors définir un indicateur de la qualité de la prédiction :

$$D(L', L_p) = \sum_{uv \in V \otimes V} min(|t'_{uv} - t_{uv,p}|, q)$$

q étant la pénalité dans le cas d'une interaction prédite dans L_p non apparue dans L' (et vice versa).

Dans ce type de tâche, déterminer si un lien a effectivement été prédit ou non nécessite la construction d'une méthode d'évaluation adaptée, capable d'estimer l'écart de temps entre un lien prédit et un lien apparu et d'en tirer une estimation de la qualité de la prédiction. De même, il n'est pas évident de traiter le cas de liens prédits en excès ou en défaut. Dans ce contexte il est difficile d'adapter les méthodes d'évaluation classiques de la prédiction de liens. En effet, dans le cadre d'une prédiction en temps continu un lien n'est jamais prédit avec une précision parfaite. Il convient donc de mesurer l'écart entre le lien apparu et le lien prédit.

Dans cet exemple de méthode d'évaluation, la distance dépend linéairement de la différence entre le temps d'apparition d'un lien et le temps prédit. On peut cependant considérer qu'une dépendance linéaire n'est pas appropriée pour décrire le problème précisément. Nous représentons en Figure 2.2 un exemple de distance utilisant une fonction sigmoïde de la distance temporelle. Cette méthode d'évaluation se prête à une interprétation de la distance temporelle entre l'événement prédit et apparu. Il est possible ici d'utiliser le vocabulaire des tâches de classification, similaire à ce qui est utilisé en prédiction de liens. En effet, si un lien est observé dans le flot réel à un instant t, alors qu'il n'a pas encore été prédit, il peut être interprété comme un équivalent à un *faux négatif*. De même un

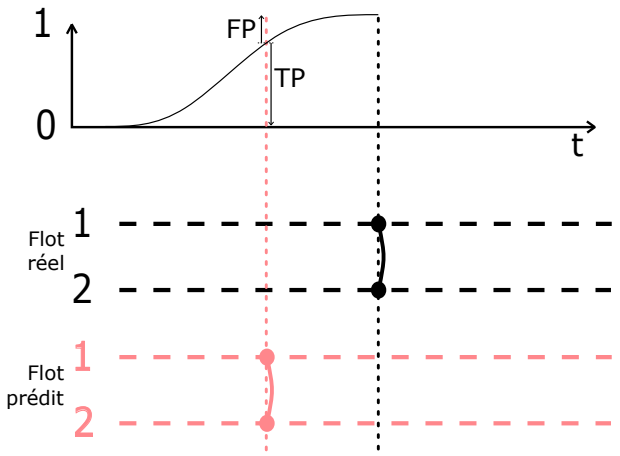


FIGURE 2.2 – Représentation de la distance entre un lien prédit et apparu et sont interprétation en termes de vrai et faux positifs.

lien prédit alors qu'il n'est pas encore apparu est un *faux positif*. Dans la pratique, un lien ne se produit jamais exactement au temps où il a été prédit. Une notion binaire de faux positif ou faux négatif n'est donc pas adaptée. Nous utiliserons plutôt un score dans un intervalle $[0, 1]$, rendant compte de la proximité temporelle de la prédiction par rapport au lien apparu. C'est le rôle de la fonction sigmoïde évoquée plus haut. Comme représenté en Figure 2.2, si nous supposons que $d_{uv}(L', L_p) = 1 - f(t'_{uv}, t_{uv,p})$ avec $t_{uv,p} < t'_{uv}$, alors $f(t'_{uv}, t_{uv,p})$ quantifierait la validité de la prédiction pour la paire de nœuds uv (correspondant donc à un *faux positif* (FP)), tandis que $1 - f(t'_{uv}, t_{uv,p})$ représenterait le degré d'erreur comme le ferait un *faux négatif* (FN). Si $t_{uv,p} > t'_{uv}$, alors $1 - f(t'_{uv}, t_{uv,p})$ représenterai le degré d'erreur comme le ferai un *faux positif* (FP). Dans ce contexte, un lien non prédit est équivalent à un lien prédit à un temps $t_{uv,p} = \infty$, et de même un lien prédit n'apparaissant pas est équivalent à un lien apparaissant à $t_{uv,p} = \infty$.

Notons cependant que VP, FP et FN sont, dans une tache de classification des valeurs binaires clairement définies, tandis qu'ils dépendent ici du choix de la fonction f. De plus, il n'existe pas ici d'équivalent direct aux *faux négatifs*.

2.2.3 Nombre de liens sur une période donnée

Dans les approches précédentes, nous nous sommes concentrés sur les temps d'apparition des liens. Une autre approche consiste à prédire combien de fois chaque paire de nœuds va interagir durant une période donnée. Cette approche est moins centrée sur l'aspect temporel que les précédentes, dans le sens où la précision temporelle de la prédiction dépend de la durée de la période de prédiction considérée. Cependant comme celle-ci peut être ajustée, il est possible d'adapter la précision temporelle de la prédiction.

2.2.3.1 Formalisation

Afin de formaliser l'objet de la prédiction, nous définissons le nombre de liens entre deux nœuds u et v dans le flot $L = (T, V, E)$ comme l'activité $\mathcal{A}_E(uv) = |\{(t, uv) \in E\}|$. Dans ce contexte l'objet de notre prédiction est de déterminer l'activité prédite $\mathcal{A}_{E_p}(uv)$ afin de minimiser $|\mathcal{A}_{E_p}(uv) - \mathcal{A}_{E'}(uv)|$ pour toute paire $uv \in V \otimes V$. Cela correspond à la prédiction du graphe induit, pondéré par le nombre d'interactions entre chaque paire de nœuds.

2.2.3.2 Évaluation

Comme dit précédemment, un des objectifs des méthodes d'évaluation est de capturer des informations pertinentes sur la prédiction tout en restant proche des méthodes existantes afin de pouvoir facilement adapter différents algorithmes de prédiction existants. Contrairement à certaines des tâches envisagées plus haut, le temps exact d'apparition des liens n'est pas prédit. Il n'est donc pas nécessaire de considérer une méthode d'évaluation prenant en compte cet aspect-là. L'évaluation consiste à comparer les activités prédites avec l'activité réelle.

Nous définissons les indicateurs de qualité dans le même esprit que précédemment, en cherchant des équivalents aux vrais positifs, faux positifs et faux négatifs. Les FP représentant les événements prédits mais n'apparaissant pas, il est légitime de traduire cette idée par la différence entre le nombre de liens prédits et le nombre de liens apparaissant réellement lorsque l'activité prédite est plus grande que l'activité réelle. De même, les FN correspondent aux événements apparaissant mais n'ayant pas été prédits, cela se traduit par l'opposé de la même différence lorsque le nombre de liens apparus est supérieur au nombre de liens prédits. Les TP correspondent au nombre d'événements apparaissant et ayant été prédits, donc le minimum entre les deux activités. Formellement :

$$\begin{cases} |VP(uv)| = \min(\mathcal{A}_{E_p}(uv), \mathcal{A}_{E'}(uv)) \\ |FP(uv)| = \max(\mathcal{A}_{E_p}(uv) - \mathcal{A}_{E'}(uv), 0) \\ |FN(uv)| = \max(\mathcal{A}_{E'}(uv) - \mathcal{A}_{E_p}(uv), 0) \end{cases}$$

Ces définitions sont illustrées en Figure 2.3. La somme de chacun de ces indicateurs pour toutes les paires de nœuds nous donne alors le nombre de VP, FP et FN global lors de notre prédiction. Ces définitions permettent de préserver les relations usuelles entre ces indicateurs, en particulier $VP + FP$ représente le nombre total de liens prédits et $VP + FN$ le nombre total de liens apparus durant T'.

Cela permet de définir des équivalents à des indicateurs de performance plus sophistiqués, utiles pour évaluer la qualité de la prédiction :

— la précision $\frac{VP}{VP+FP}$, qui, en prédiction de liens, représente la proportion de liens prédits correctement parmi tous les liens prédits

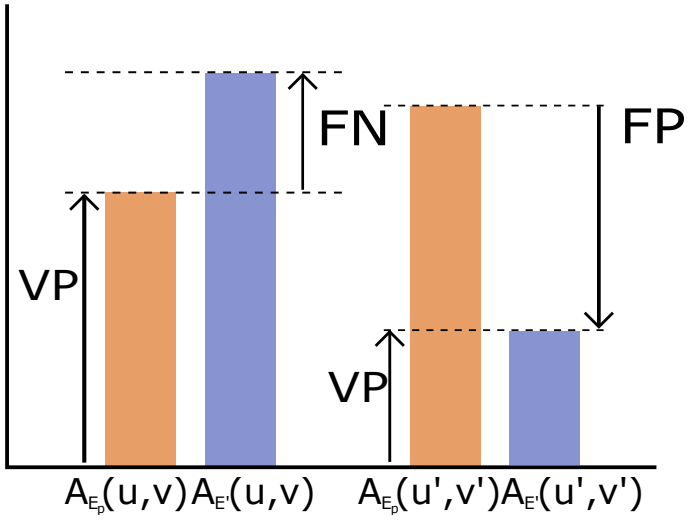


FIGURE 2.3 – Illustration de la méthode d'évaluation dans les cas $A_{E_p} < A_{E'}$ et $A_{E_p} > A_{E'}$.

— le rappel $\frac{VP}{VP+FN}$ qui représente la proportion de liens apparus correctement prédits.
— le F-score évaluant la qualité globale de la prédiction, défini comme la moyenne harmonique des deux précédents indicateurs : $2 \cdot \frac{precision \cdot rappel}{precision + rappel}$. Celui-ci nous servira d'indicateur à optimiser dans la suite.

Encore une fois, la notion de Vrais Négatifs n'a pas d'équivalent direct. Nous ne définissons donc pas d'autres indicateurs classiques utilisant cette notion (spécificité, sensibilité, courbe ROC, etc).

2.2.4 Apparition d'au moins un lien sur une période donnée

Finalement, considérons l'objectif de prédire si une paire interagit au moins une fois durant le flot réel. Cette tâche est intéressante car elle est en réalité similaire au problème de la prédiction de liens dans un graphe. La prédiction correspond à prédire la structure du graphe induit par le flot de liens L'.

La principale différence est ici l'utilisation des flots de liens, permettant d'utiliser à la fois l'information temporelle et structurelle afin d'améliorer la prédiction. En terme d'évaluation, ce problème a largement été étudié comme un problème de classification, permettant donc d'utiliser les méthodes d'évaluations utilisées habituellement pour ce genre de tâche comme vu au Chapitre 1.

2.3 Envisager les différentes tâches en utilisant des fonctions de prédiction

Nous présentons une façon d'approcher ces différents problèmes, en prenant avantage du fait que les données contiennent des informations structurelles et temporelles. Il est donc nécessaire de représenter les caractéristiques du flot d'une façon reflétant ces deux aspects. De plus, nous voulons construire une méthode permettant de combiner les informations issues de ces caractéristiques sans perte d'information.

2.3.1 Représenter les caractéristiques du flot

Nous capturons les caractéristiques structurelles et temporelles des flots de liens par des fonctions que nous appellerons métriques, qui représentent la vraisemblance de l'apparition d'un lien entre les nœuds u et v à chaque instant t donné dans la période de prédiction. Cela permettra par exemple de représenter des caractéristiques indiquant des périodes plus susceptibles de voir l'apparition de liens que d'autres. Par exemple, une métrique capturant le fait que les paires de nœuds ont des interactions à intervalles réguliers pourra être représentée sous la forme de pics périodiques dans comme illustré dans la figure 2.4.

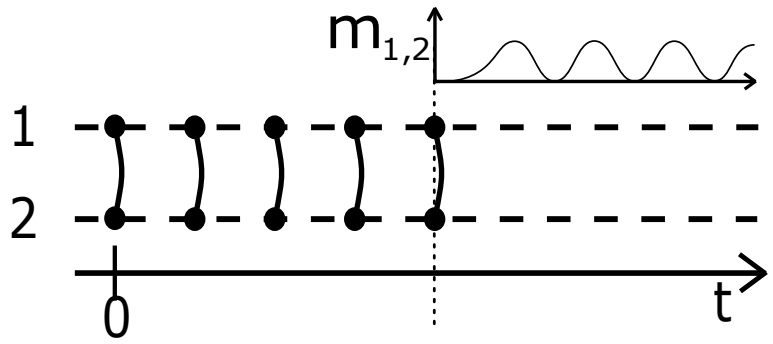


FIGURE 2.4 – Illustration d'une possible métrique dans le cadre de nœuds ayant des interactions à intervalles réguliers.

Cette représentation a également l'avantage de permettre de regrouper au sein d'une approche unifiée des outils provenant de la prédiction de liens et de la prédiction de séries temporelles. En effet, les métriques de base utilisées en prédiction de liens sont, la plupart du temps, représentées sous la forme d'un indice, comme le nombre de voisins communs. Ce type de métriques peut être facilement représenté en y associant une fonction indépendante du temps de la valeur de l'indice. Dans le cadre de la prédiction de séries temporelles, l'objectif est de prédire l'évolution d'une suite de valeurs au cours du temps, souvent sous la forme de fonction. Représenter celles-ci dans le cadre de notre protocole se fait alors de façon directe.

Nous verrons également au Chapitre 5 des exemples de métriques spécifiques aux flots de liens, ainsi que leurs fonctions associées.

2.3.2 Fonction de prédiction

Ces métriques sont ensuite combinées en une fonction $\mathcal{F}_{uv}(t)$ telle que pour tout $uv \in V \otimes V$, et pour tout t dans la période de prédiction, $\mathcal{F}_{uv}(t)$ représente la vraisemblance de l'apparition d'un lien (t, uv). Cette représentation permet de combiner de façon cohérente les informations apportées par des métriques provenant de la prédiction de liens dans les graphes comme de la prédiction de séries temporelles.

Nous verrons en détails au Chapitre 3 comment notre protocole effectue la combinaison des différentes fonctions de métriques et l'utilisation de la fonction de prédiction dans le cadre ce travail.

2.4 Utilisation des fonctions de prédiction pour chaque tâche

Nous avons vu en Section 2.2 que les problèmes présentés se répartissent en deux tâches distinctes : prédire l'apparition d'un ou plusieurs liens, c'est-à-dire prédire précisément les triplets (t, uv) (Sections 2.2.1 et 2.2.2), et la prédiction du nombre de liens apparaissant durant une période donnée (Sections 2.2.3 et 2.2.4). Nous allons voir dans cette section comment approcher les différentes tâches dans le cadre de l'utilisation de fonctions de prédiction. Nous allons supposer que celles-ci permettent effectivement de représenter la vraisemblance d'apparition de liens de chaque paire de nœuds au cours du temps et passer en revue les différents avantages et difficultés de chaque tâche.

2.4.1 Prédire les temps d'apparition des liens

Les fonctions de prédiction représentant la vraisemblance d'apparition de liens entre une paire de nœuds donnée, nous les utilisons afin d'identifier les temps les plus susceptibles de voir un lien apparaître. Une approche naturelle de ce problème pourrait être de rechercher les maximums locaux de la fonction de prédiction. De plus, afin de prédire efficacement l'apparition de liens, il conviendrait de déterminer un critère pertinent indiquant si un pic est assez significatif pour justifier la prédiction. Cela pourrait passer par la définition d'un seuil, déterminé par le flot d'entrée, éliminant les maximums locaux inférieurs au dit seuil. Cela reviendrait alors à détecter des pics dans une fonction sur les intervalles déterminés par le seuil. Cette approche est illustrée en Figure 2.5. La recherche de pics

dans une série temporelle est un domaine de recherche actif et des méthodes adaptées en provenant pourraient être utilisées [P+09].

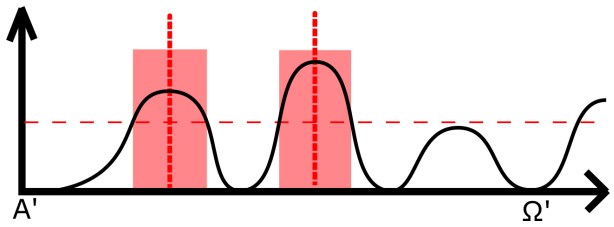


FIGURE 2.5 – Illustration d'une méthode de détermination d'apparition de liens, une recherche des maximums locaux associée à un seuil permettant d'éliminer les pics non significatifs.

Cependant, dans l'optique d'une application sur des systèmes réels, nous devons envisager certains points particuliers liés à cette approche :

— Le choix du critère de significativité des pics prend alors une importance critique dans le processus, déterminant le nombre de liens prédits pour chaque paire et dans le flot global. Il doit donc être choisi avec précaution d'après le flot d'entrée.
— La détection de pic ne répond pas exactement au même problème et peut poser certaines difficultés, par exemple pour une fonction de prédiction constante ou présentant un plateau sur une certaine durée. Tout d'abord, une détection de pics peut ne pas détecter ce plateau, menant à l'absence de prédiction durant cette période. De même, un plateau en dessous du seuil fixé ne conduirait à la prédiction d'aucun lien, quand bien même, intuitivement, il semble indiquer une certaine vraisemblance d'apparition de lien.

2.4.2 Prédiction du nombre de liens sur une période donnée

Afin de prédire le nombre de liens durant la période de prédiction, il est possible d'utiliser les fonctions de prédiction d'une manière différente. Ne prédisant plus le temps exact des interactions, il n'est plus nécessaire de détecter les pics de la fonction de prédiction.

Les fonctions de prédiction représentent la vraisemblance de l'apparition de liens entre chaque paire de nœuds durant la période de prédiction. Nous construisons ces fonctions pour faire en sorte que l'aire sous celles-ci soit proportionnelle au nombre d'interactions entre chaque paire de nœuds. En supposant que nous soyons capables de prédire le nombre total d'interactions dans le flot réel, il est donc possible de répartir les liens parmi les paires de nœuds, proportionnellement à l'aire sous leur fonction de prédiction. En pratique, la prédiction du nombre d'interactions dans le flot réel est un problème classique de prédiction dans les séries temporelles [BJRL15].

2.5 Conclusion

Dans ce chapitre, nous avons exploré différents problèmes concernant la prédiction dans les flots de liens et défini des outils centraux de notre protocole : la modélisation des caractéristiques du flot et les fonctions de prédiction. Finalement, nous avons formalisé la tâche de prédiction à effectuer : prédire l'activité de chaque paire de nœuds durant la période de prédiction.

Il est important de noter que nous n'avons pas exploré toutes les possibilités offertes par le formalisme des flots de liens et la prédiction dans le cadre de systèmes dynamiques. Il est possible d'imaginer d'autres formes de prédiction intéressantes faisant le lien entre la structure et la dynamique du système. Il est par exemple possible d'imaginer une prédiction à plusieurs niveaux, en utilisant des données modélisées comme un flot de liens pour prédire une structure sous-jacente modélisée par un graphe. Par exemple, modéliser les interactions passées entre un groupe d'individu, e-mails, appels téléphoniques, comme un flot de liens et inférer leurs relations sociales, modélisées comme un graphe. Dans le cas d'appels téléphoniques par exemple, deux individus ayant un grand nombre d'interactions durant les heures de travail sont sans doute des collègues, alors que des interactions concentrées lors des soirs et weekends indiquent probablement plutôt des amis ou connaissances.

Chapitre 3

Protocole

Nous présentons ici notre protocole de prédiction de l'activité dans les flots de liens. Nous insistons sur l'aspect modulaire de ce protocole, composé de différents modules indépendants permettant de l'adapter aux spécificités de différents systèmes ainsi que d'ouvrir la voie à l'intégration de méthodes plus performantes pour chaque module.

Suivant l'approche proposée pour ce type de tâche au chapitre 2, le protocole repose sur deux parties : d'une part, la combinaison des informations apportées par les métriques en une fonction de prédiction et d'autre part la prédiction du nombre global de liens dans le flot de prédiction. Le protocole utilise ensuite la comparaison entre les différentes fonctions de prédiction pour répartir les liens entre les paires de nœuds. Cela permet d'isoler un élément crucial du protocole de prédiction, l'évaluation du nombre global de liens prédits. Un algorithme d'apprentissage supervisé permet alors d'optimiser la construction de la fonction de prédiction.

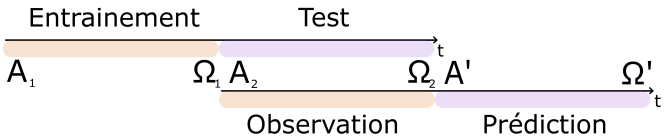


FIGURE 3.1 – Illustration des différentes périodes utilisées dans notre protocole : les périodes d'entraînement et de test (en haut) utilisées pour entraîner l'algorithme et les périodes d'observation et de prédiction (en bas) pour effectuer la prédiction elle-même.

Dans la pratique, nous procédons de la façon suivante, illustrée en Figure 3.1 : nous divisons le flot d'entrée L présenté au chapitre précédent en deux sous-flots : un *sous-flot d'entraînement* $L_1 = (T_1, V, E_1)$ avec $T_1 = [A_1, \Omega_1]$, utilisé pour calculer les métriques lors de la phase d'apprentissage, et un *sous-flot de test* $L_2 = (T_2, V, E_2)$ avec $T_2 = [A_2, \Omega_2]$. Les valeurs des paramètres sont alors calculées grâce à un algorithme d'apprentissage afin d'optimiser la prédiction de l'activité sur T_2 en utilisant les informations contenues dans T_1. Les métriques utilisées sont ensuite recalculées sur le sous-flot d'observation $L_2 =$

(T_2, V, E_2) avec $T_2 = [A_2, \Omega_2]$, pour prédire l'activité durant le flot de réel $L' = (T', V, E')$ avec $T' = [A', \Omega']$.

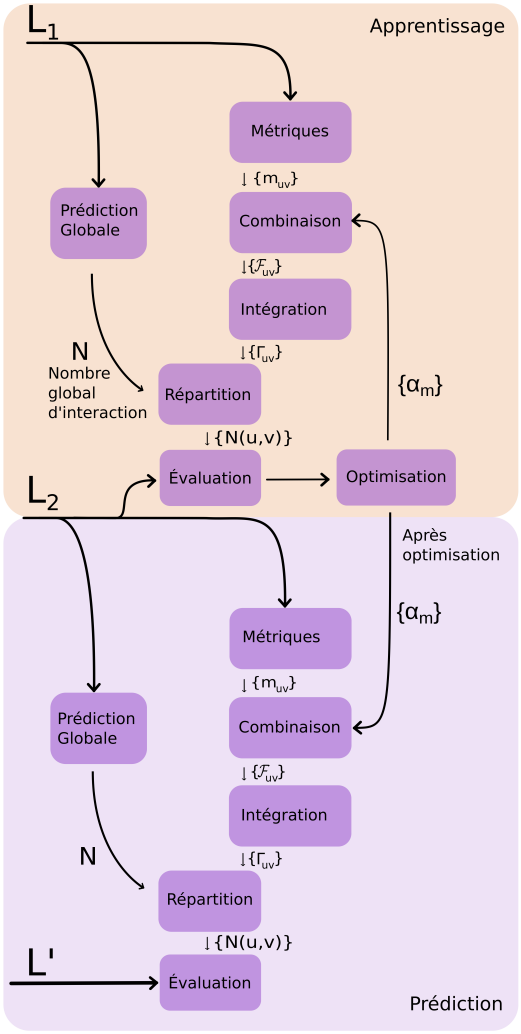


FIGURE 3.2 – Résumé de notre protocole, incluant les phases d'apprentissage et d'évaluation

L'architecture de notre protocole est présentée dans la Figure 3.2. Dans la partie supérieure, nous pouvons voir la phase d'apprentissage, permettant d'optimiser les poids de chaque métrique dans la combinaison α_m, prenant en entrée le flot L_1. Le protocole se divise ensuite en deux branches, la prédiction globale du nombre de liens N dans le flot L_2 et le calcul et l'utilisation des métriques. Le calcul des métriques nous permet d'obtenir l'ensemble des métriques $\{m_{uv}\}$ qui seront ensuite combinées pour former les fonctions de prédiction $\{\mathcal{F}_{uv}\}$. Celles-ci sont alors intégrées pour former l'ensemble $\{\Gamma_{uv}\}$ des scores d'apparition de liens.

Ces deux branches permettent ensuite de répartir ces liens et effectuer la prédiction, l'ensemble des $N(u, v)$, les nombres de liens prédits pour chaque paire de nœuds uv.

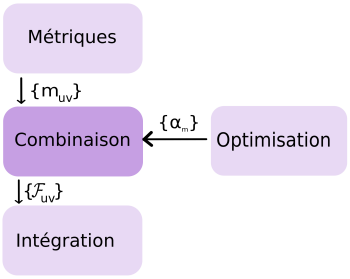
$N(u, v)$ est donc l'activité prédite $A_{E_p}(uv)$ pour chaque uv lors de la phase de prédiction et son équivalent sur la période T_2 lors de la phase d'entrainement. La prédiction est ensuite évaluée en utilisant le flot L_2 comme présenté au chapitre précédent. Cette première partie du protocole est utilisée par un algorithme d'apprentissage afin d'optimiser les paramètres α_m de la combinaison. Ces paramètres sont ensuite utilisés dans la seconde partie du protocole afin d'effectuer la prédiction de l'activité dans le flot L' à partir de L_2.

Nous allons d'abord nous concentrer sur la construction des fonctions de prédiction présentées dans le chapitre précédent à partir des métriques qui seront définies au Chapitre 5. Nous allons ensuite extrapoler le nombre global de liens à la période de prédiction, puis répartir ces liens entre les différentes paires de nœuds grâce aux fonctions de prédiction associées à ceux-ci.

Pour des raisons de clarté, nous allons présenter la phase d'apprentissage du protocole, en utilisant le flot d'entraînement L_1 pour prédire le flot de test L_2. La prédiction de L' se fait de manière similaire à partir du flot L_2.

3.1 Combinaisons de métriques

En considérant un ensemble $\mathcal{M}$ de métriques (qui seront définies au Chapitre 5), nous allons construire une fonction de prédiction pour chaque paire de nœuds uv dans l'ensemble des paires de nœuds du flot d'entraînement $V \otimes V$. Chaque métrique m_{uv} est une fonction positive du temps associée à une paire de nœuds, représentant la vraisemblance d'apparition de liens entre u et v d'après cette métrique durant la période de test au temps t.



Afin d'utiliser l'information apportée par chaque métrique, il nous faut établir un protocole pour combiner les métriques. Nous choisissons ici une méthode simple permettant d'analyser le comportement de notre protocole lors de l'apport d'informations de natures différentes : nous effectuons une combinaison linéaire des métriques. Nous construisons alors une fonction de prédiction $\mathcal{F}$, telle que pour toute paire $uv \in V \otimes V$, et pour tout t dans la période de test $[A_2, \Omega_2]$, $\mathcal{F}_{uv}(t)$ représente la vraisemblance d'apparition de liens entre u et v au temps t. Plus formellement :

$$\mathcal{F}_{uv}(t) = \sum_{m \in \mathcal{M}} \alpha_m \cdot m_{uv}(t) \tag{3.1}$$

où $m_{uv}(t)$ est la fonction associée à la métrique m. Les paramètres α_m contrôlent le poids de chaque métrique dans la fonction de prédiction. Étant donnée cette fonction de prédiction, une méthode classique consiste à apprendre sur une période d'entraînement les valeurs des paramètres α_m qui optimisent un critère d'évaluation donné. La méthode

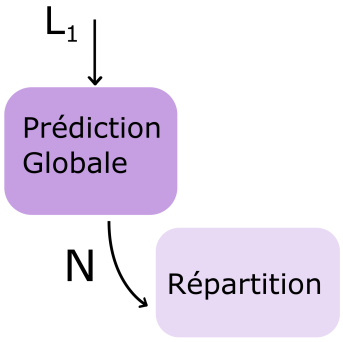
d'optimisation des poids est présentée en Section 3.4. Ces poids sont ensuite utilisés pour la prédiction réelle sur la période de prédiction.

Notons que l'information importante pour la prédiction n'est pas contenue dans la valeur absolue de la fonction $\mathcal{F}_{uv}$ mais dans la valeur de celle-ci relativement aux autres fonctions $\mathcal{F}_{u'v'}$. Comme nous le verrons dans la Section 3.3, ces fonctions nous permettront de répartir entre les différentes paires de nœuds le nombre total de liens prédits.

3.2 Prédiction de l'activité globale

Nous allons ici présenter notre méthode de prédiction du nombre global de liens dans L_2. Afin de prédire le nombre global d'interactions durant $T_2 = [A_2, \Omega_2]$, nous allons ici faire l'hypothèse que la fréquence moyenne d'apparition de liens dans L_2 est identique à celle dans L_1 comme illustré en Figure 3.3. Nous pouvons alors extrapoler linéairement l'activité du flot afin de déterminer le nombre global de liens N à prédire :

$$N = |E_1| \cdot \frac{\Omega_2 - A_2}{\Omega_1 - A_1} \tag{3.2}$$



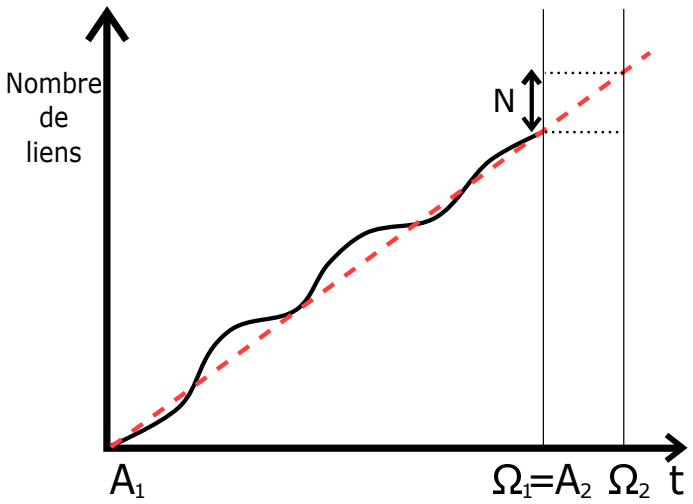
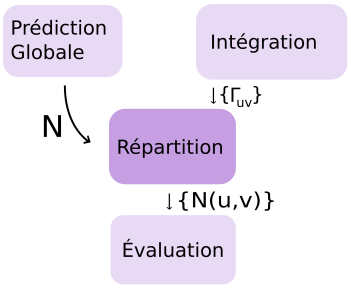


FIGURE 3.3 – Illustration du principe de la prédiction de l'activité globale, représentant le nombre de liens apparaissant durant la période d'entraînement. Le nombre de liens N à prédire pendant $[A_2, \Omega_2]$ est obtenu par extrapolation linéaire (ligne pointillée).

La prédiction de l'activité globale est une tâche de prédiction de séries temporelles classique. De ce fait, des méthodes plus élaborées seraient utilisable pour ce module. Notre ambition à ce stade se limite à une première approche privilégiant la simplicité.

3.3 Répartition des liens

La fonction de prédiction d'une paire de nœuds représentant la vraisemblance de l'apparition de liens entre ces deux nœuds, nous définissons une méthode pour les comparer. Il faut, pour cela, condenser, pour chaque paire de nœuds, l'information contenue dans la fonction de prédiction lors de la période de test afin d'obtenir un score permettant de répartir les N liens estimés précédemment. Nous définissons le *score d'apparition de liens* Γ_{uv} comme la vraisemblance d'apparition de liens entre u et v durant T_2 :



$$\Gamma_{uv} = \int_{A_2}^{\Omega_2} \mathcal{F}_{uv}(t)\, dt \tag{3.3}$$

Cela permet de prendre en compte les variations de la fonction de prédiction sur la période de test et ainsi de n'utiliser que l'information concernant la période considérée provenant des métriques.

Nous répartissons les N liens estimés proportionnellement au *score d'apparition de liens* pour obtenir $N(u, v)$, le nombre de liens prédits pour chaque paire uv :

$$N(u, v) = N \cdot \frac{\Gamma_{uv}}{\sum_{xy \in V \otimes V} \Gamma_{xy}} \tag{3.4}$$

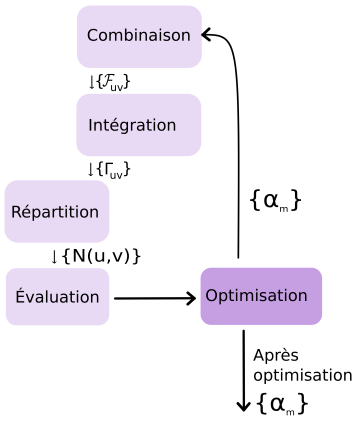
Comme mentionné précédemment, ce sont donc les valeurs relatives des Γ_{uv} qui nous sont utiles lors du calcul des $N(u, v)$, avec $\sum_{xy \in V \otimes V} N(x, y) = N$.

Ce protocole permet de prédire l'activité future des paires de nœuds, c'est-à-dire le nombre de liens apparaissant entre chaque paire de nœuds durant T_2. Il est important de noter que, dû aux spécificités de notre tâche de prédiction, et contrairement aux méthodes habituellement utilisées en prédiction de liens dans les graphes, ce nombre n'est pas nécessairement un entier. La qualité de la prédiction peut ensuite être évaluée en utilisant la méthode présentée en Section 2.2.3.2 du Chapitre 2.

3.4 Processus d'optimisation

Le choix des valeurs des paramètres α_m est une étape cruciale pour obtenir une prédiction efficace, car ils déterminent l'équilibre entre les différentes métriques combinées. Une méthode automatisée pour optimiser ces valeurs est également nécessaire pour l'utilisation simultanée de multiples métriques puisque dans ce cas, explorer systématiquement l'espace des paramètres serait trop coûteux. Nous suivons un protocole d'apprentissage supervisé et définissons des périodes d'entraînement et de test.

3.4.1 Flots d'entraînement, test, observation et prédiction



Comme présenté en introduction de ce chapitre nous utilisons une méthode dite de *hold-out* [K+95]. Nous divisons donc les données en sous-flots afin de définir une phase d'apprentissage durant laquelle nous optimisons les valeurs des paramètres, puis extrapolons ces paramètres afin d'effectuer la prédiction elle-même. Notons que la méthode de *k-fold validation*, est classiquement utilisée dans de nombreuses tâches de prédiction, considérée comme plus performante. Cela consiste à diviser les données en k sous ensembles, puis prédire un des sous-ensembles en utilisant les $k-1$ autres. Le processus est ensuite répété k fois. Cependant celle-ci est difficile à implémenter dans un contexte où l'agencement temporel des liens joue un rôle majeur [AC+10]. Cela conduirait ici à utiliser de nombreuses périodes de temps et à effectuer des prédictions sur des intervalles non contigus, menant à une importante perte de structure temporelle.

Comme présenté plus haut le flot d'entrée L est divisé en un *sous-flot d'entraînement* L_1 et un *sous-flot de test* L_2, qui sont utilisés pour optimiser les paramètres $\{\alpha_m\}$. La prédiction est ensuite effectuée sur le flot L' en utilisant le flot L_2. Dans le cadre de ce travail le flot d'entrée et de prédiction sont contigus et les sous-flots de test et d'observation sont identiques. Comme vu lors de la formalisation de la tâche de prédiction au Chapitre 2, ce choix n'est pas nécessaire. Cependant, il correspond au choix fait par la plupart des études de prédiction et correspond à de nombreux cas d'application. Les dernières observations sont utilisées pour prédire les prochains événements, en se basant sur l'hypothèse d'une certaine continuité dans le comportement du système au cours du temps.

3.4.2 Descente de gradient

Nous utilisons une implémentation standard d'un algorithme de descente de gradient pour explorer l'espace des paramètres et trouver un optimum pour chaque paramètre α_m. L'objectif est de maximiser le F-score obtenu lors de la prédiction en se déplaçant itérativement dans l'espace des paramètres. Nous choisissons cet indicateur car il représente un compromis entre la précision et le rappel. À chaque étape de l'algorithme, nous faisons varier chaque paramètre de $\pm\epsilon$, le pas de dérivation, afin de déterminer la direction de l'espace des paramètres maximisant le F-score. Nous nous déplaçons alors dans cette direction et recommençons le processus. L'algorithme s'arrête lorsque qu'aucun déplacement ne permet d'améliorer le F-score.

L'utilisation d'une combinaison linéaire de fonctions de métrique permet d'éviter certaines étapes lourdes en calculs. Par exemple, les métriques ne nécessitent pas d'être recalculées à chaque combinaison. De même, l'intégration étant linéaire, nous pouvons

effectuer celle-ci directement sur les métriques et éviter de la recalculer lors de l'exploration. Pour plus de clarté et parce que de nouvelles méthodes de combinaison peuvent être implémentées ceci n'est pas représenté en Figure 3.2. Cela permet de réduire le coût en calcul de la méthode significativement. L'ensemble initial de paramètres donné en entrée à l'algorithme d'apprentissage est tiré aléatoirement dans l'espace des paramètres pour chaque prédiction.

3.5 Conclusion

Notre protocole est capable de capturer différentes informations et de les combiner afin d'optimiser la prédiction. La modularité du protocole offre une grande flexibilité quant aux tâches de prédiction envisagées. Chacune des étapes présentées ici est conçue pour être indépendante des autres. Ainsi l'implémentation d'un nouvel algorithme d'apprentissage peut se faire sans modifier le reste du protocole. De même, de nouvelles méthodes de combinaison peuvent facilement être implémentées. Notre implémentation est disponible à l'adresse `https://github.com/ThibaudA/linkstreamprediction`. De plus amples informations concernant celle-ci sont données dans l'Annexe C

Dans une démarche exploratoire, certains choix effectués ici ont été faits par simplicité. Nous avons notamment fait l'hypothèse que la fréquence d'apparition de liens globale reste constante entre la période d'entrée et la période de prédiction en Section 3.2. Cependant, cette hypothèse n'est pas toujours valide et dépend fortement des données étudiées. Les modèles développés dans le contexte des séries temporelles, comme le modèle ARIMA, qui extrapole plus précisément l'activité passée [HL09], permettraient certainement de mieux évaluer le nombre de liens prédits.

Dans la suite, nous allons étudier le comportement du protocole lors de prédictions sur des jeux de données réels. Nous nous concentrons plus particulièrement sur la combinaison des métriques, notamment les poids déterminés par l'algorithme d'apprentissage et leur influence sur le type de liens prédits. Cela nous permettra d'identifier les limites du protocole ainsi que des pistes d'améliorations.

Chapitre 4

Jeux de données

Afin de mesurer les performances de notre protocole, nous allons conduire des expériences sur quatre jeux de données. Ceux-ci sont modélisés sous la forme de flots de liens. Chaque lien, non dirigé, (t, uv) indique que les nœuds u et v ont interagi au temps t. Nous appelons cela une interaction entre les nœuds u et v.

Nous utilisons des jeux de données de contacts. Cela permet une interprétation des interactions des nœuds intuitive. Chacun de ces jeux de données comporte entre 96 et 305 nœuds ayant des contacts durant des périodes allant de quelques jours à plusieurs mois. Nos expériences porteront sur des sous-parties de ces données afin d'effectuer les prédictions sur des intervalles de temps où nous pouvons nous attendre à ce que les paires de nœuds ne changent pas radicalement de comportement, par exemple, sur certains jeux de données, pendant la nuit. Comme dit précédemment, dans chacune des expériences présentées les durées des périodes d'entraînement, de validation, d'observation et de prédiction seront choisies égales.

4.1 Highschool

Les premières données utilisées ont été collectées dans un lycée en 2012 que nous noterons *Highschool* dans la suite. Les élèves des classes préparatoires de ce lycée ont porté pendant un peu plus d'une semaine des capteurs RFID lors de leur présence dans l'enceinte du lycée. Chaque lien (t, uv) indique que le capteur porté par le nœud u ou v a détecté le capteur porté par l'autre nœud au temps t, et donc que les deux nœuds ont été assez proches pour permettre la détection. Les capteurs sont calibrés pour détecter l'orientation des élèves, limitant l'apparition d'un lien à deux individus face à face, à une distance inférieure à un mètre pendant 20 secondes (voir [MFB15] pour une description détaillée).

Le flot de liens ainsi obtenu comporte 181 nœuds et 45047 liens, connectant 2220 paires de nœuds distinctes durant une période de 8 jours. Les contacts sont détectés toutes

les 20 secondes. Le profil d'activité de l'intégralité du jeu de données est représenté en Figure 4.1 (b).

Nous pouvons observer sur le profil du jeu de données total l'absence de liens durant la nuit et le weekend, les étudiants ne portant les capteurs que durant leur présence au lycée. Pour cette raison, nous nous concentrerons sur la première journée de l'expérimentation dont le profil d'activité est présenté en Figure 4.1 (a). La dynamique est rythmée par les emplois du temps de chaque classe, notamment marquée par un changement de structure durant la pause déjeuner : les contacts interclasses sont alors fréquents, tandis qu'ils sont rares lors des heures de classe [MFB15].

Concernant ce jeu de données, nous utiliserons des périodes longues d'une, deux et trois heures commençant le lundi à 8h30. Les différentes périodes utilisées sont résumées en Table 4.1.

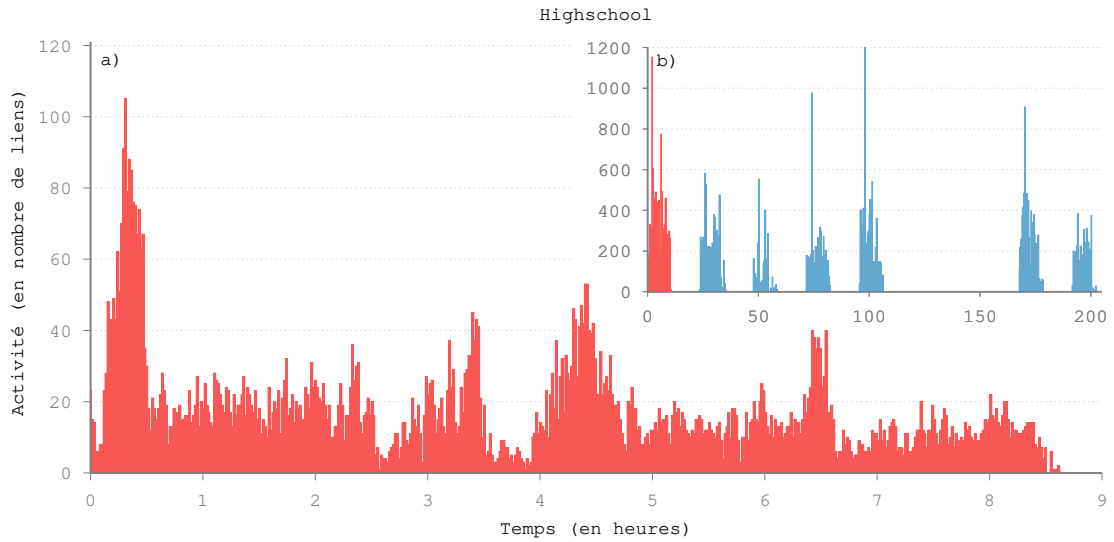


FIGURE 4.1 – Nombre de liens apparaissant au cours du temps dans le jeu de données *Highschool* lors de la première journée sur des intervalles de 60 secondes (a) et durant la totalité du jeu de données sur des intervalles de 25 minutes (b) (en rouge : la période étudiée).

4.2 Infocom

Le second jeu de données a été collecté durant la conférence IEEE INFOCOM 2006 à Barcelone (*Infocom*) – voir [SGC+09]. Des capteurs Bluetooth ont été utilisés pour détecter la proximité des différents participants à la conférence. Ces données comportent 98 nœuds et 283100 liens. Durant cette expérience de trois jours, 4338 paires de nœuds ont interagi. Les contacts sont détectés toutes les 120 secondes. Ce jeu de données de contacts implique moins de nœuds mais contient plus de liens et de paires de nœuds actives que le

précédent. Comparer ces deux jeux de données permettra d'obtenir des informations sur comment la densité d'interactions affecte les performances de la prédiction et la combinaison de métriques utilisée. Nous utiliserons des périodes de prédiction similaires au jeu de données précédent, avec des périodes d'une, deux et trois heures commençant le premier jour de la conférence à 9h. Les profils d'activités du jeu de données et de la période étudiée sont représentés en Figure 4.2. Nous observons bien un cycle jours-nuits marqué, en notant cependant l'apparition constante de liens durant les nuits. On remarque également que la fréquence globale d'apparition de liens semble plus stable au cours du temps que dans les données précédentes.

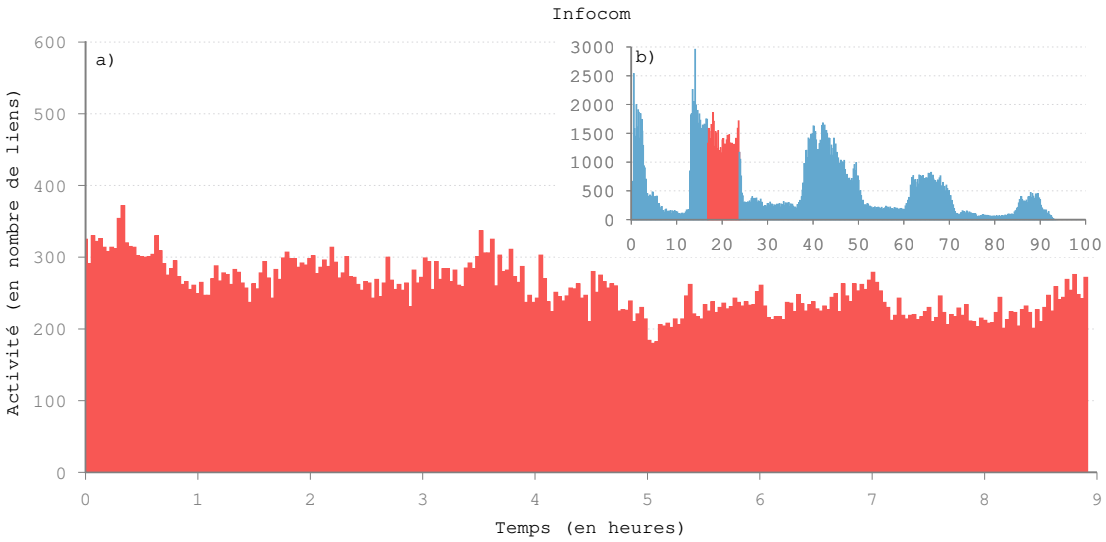


FIGURE 4.2 – Nombre de liens apparaissant au cours du temps dans le jeux de données *Infocom* lors du premier jour de la conférence sur des intervalles de 2 minutes (a) et durant la totalité du jeu de données sur des intervalles de 10 minutes (b) (en rouge : la période étudiée).

4.3 Reality Mining

Nous étudierons également le jeu de données *Reality Mining* – voir [EP05]. Il s'agit de données de contacts entre étudiants du Massachusetts Institute of Technology, enregistrées par leurs téléphones mobiles. Il contient 1063063 liens entre 96 nœuds pendant neufs mois, connectant 2539 paires de nœuds avec une granularité de 10 minutes.

L'expérience s'étendant sur une durée significativement plus longue, ce jeu de données permettra d'utiliser des périodes d'expérimentation de plusieurs jours, afin d'évaluer les performances de notre protocole pour différentes échelles de temps. Nous choisirons des périodes d'un, deux et trois jours, commençant un mardi à 1h30. Les profils d'activités sont représentés en Figure 4.3. Ici encore le cycle jour-nuit est particulièrement marqué.

On note également une forte baisse d'activité durant le cinquième et sixième jour de la période, correspondant au weekend. Il sera intéressant d'observer les conséquences d'un tel changement d'activité sur la qualité de la prédiction.

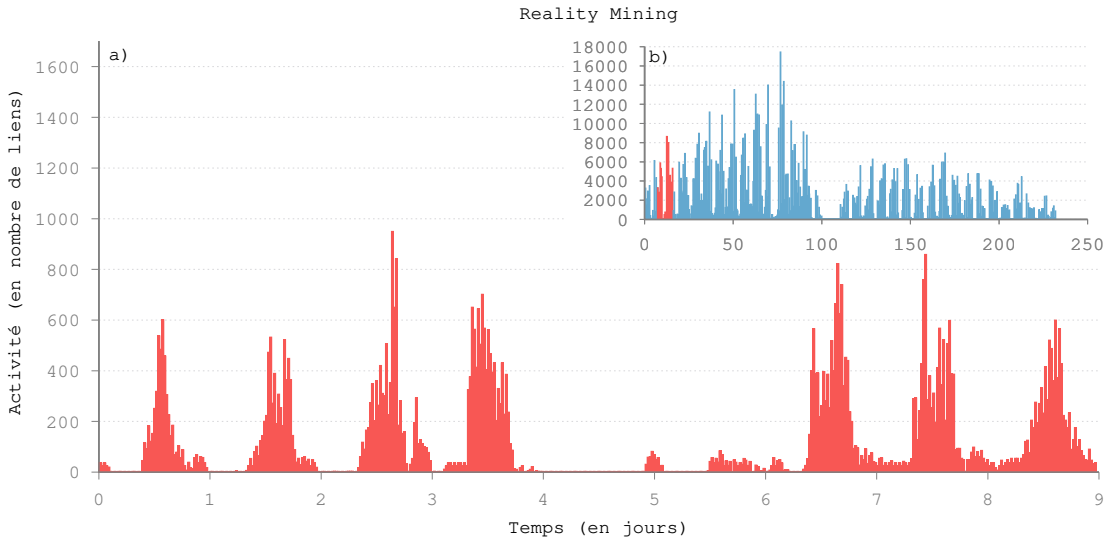


FIGURE 4.3 – Nombre de liens apparaissant au cours du temps dans le jeu de données *Reality Mining* durant les périodes utilisées lors des expériences sur des intervalles de 25 minutes (a) et durant la totalité du jeu de données sur des intervalles de 10 heures (b) (en rouge : la période étudiée).

4.4 Taxi

Le jeu de données *Taxi* est basé sur les positions GPS de 305 taxis dans la ville de Rome, enregistrées durant le mois de février 2014 – voir [BBL+14]. Tandis que les contacts des jeux de données précédents sont capturés à l'aide de capteurs de proximités entre individus, les contacts sont ici basés sur les positions de chaque voiture au cours du temps. Nous considérons ici que deux taxis ont interagi s'ils se situent à moins de 30 mètres l'un de l'autre.

Cela forme un flot de liens de 22364061 liens avec une précision temporelle d'une seconde. Durant l'expérience, 16799 paires de nœuds ont interagi. Ce jeu de données fait donc intervenir nettement plus de paires de nœuds que les autres jeux de données auxquels nous allons nous intéresser. De plus, le type de contact capturé devrait apporter une information différente des jeux de données précédents, la proximité de deux taxis étant, *a priori*, moins synonyme d'une interaction sociale entre ceux-ci qu'une proximité de deux élèves ou deux participants à une conférence. Les contacts seront sans doute plutôt influencés par les zones d'activité des différents taxis dans la ville de Rome ainsi que les trajets effectués par les clients. De façon similaire au jeu de données *Reality Mining* nous

allons nous pencher sur des périodes longue d'un, deux ou trois jours, commençant un mercredi à 8h00. Les profils d'activité sont représentés en Figure 4.4. L'activité du jeu de données semble comparable d'un jour sur l'autre, à l'exception du cinquième et sixième jours, où l'activité est plus faible, correspondant ici aussi au weekend. D'autre part notons que les cycles jours-nuits semblent moins marqués que dans les autres jeux de données.

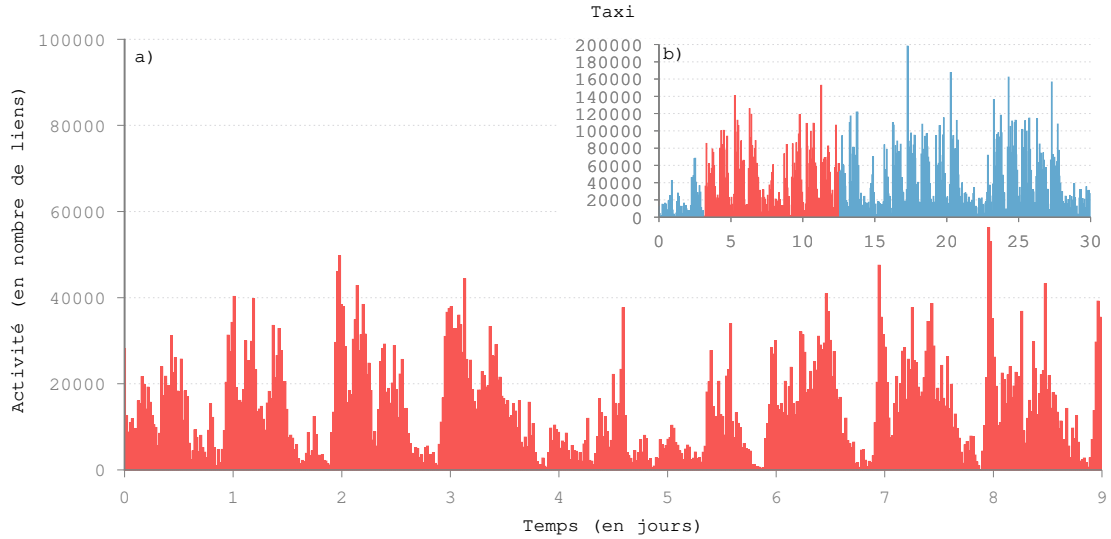


FIGURE 4.4 – Nombre de liens apparaissant au cours du temps dans le jeu de données *Taxi* durant les périodes utilisées lors des expériences sur des intervalles de 25 minutes (a) et durant la totalité du jeu de données sur des intervalles de 85 minutes (b) (en rouge : la période étudiée).

4.5 Conclusion

Ces quatre jeux de données nous permettront d'étudier les comportements des différents modules du protocole dans diverses conditions. Les périodes choisies pour chaque expérience sont résumées dans la Table 4.1. Pour chacune des périodes le nombre de nœuds, de liens et de paires actives de la période sont indiquées en Table 4.2.

Tous les jeux de données présentés ici sont basés sur une proximité physique entre des individus ou des véhicules. Compte tenu des nombreuses applications liées aux domaines de la prédiction de liens ou de séries temporelles, d'autres jeux de données capturés dans des contextes différents pourraient également apporter des éclairages intéressant sur le comportement de notre protocole lors de tâches de prédiction spécifiques. De nombreux jeux de données disponibles, comme des envois de paquets entre serveurs [Lab] ou des échanges d'e-mails [MPK11], se prêtent à ce type de tâche.

TABLE 4.1 – Temps de départs de fin de chaque période pour chaque jeu de données.

Données	Durée	A_1	$\Omega_1 = A_2$	$\Omega_2 = A'$	Ω'
Highschool	1h	8:30	9:30	10:30	11:30
	2h	8:30	10:30	12:30	14:30
	3h	8:30	11:30	14:30	17:30
Infocom	1h	9:00	10:00	11:00	12:00
	2h	9:00	11:00	13:00	15:00
	3h	9:00	12:00	15:00	18:00
Reality Mining	1j	Mardi	Mercredi	Jeudi	Vendredi
	2j	Mardi	Jeudi	Samedi	Lundi
	3j	Mardi	Vendredi	Lundi	Jeudi
Taxi	1j	Mercredi	Jeudi	Vendredi	Samedi
	2j	Mercredi	Vendredi	Dimanche	Mardi
	3j	Mercredi	Samedi	Mardi	Vendredi

TABLE 4.2 – Nombre de nœuds, de liens et de paires de nœuds actives pendant chaque période de chaque jeu de données.

Données	Durée	Nb nœuds	Nb liens	Nb paires actives
Highschool	1h	133	3927	407
	2h	151	6999	631
	3h	156	8899	750
Infocom	1h	95	25252	1893
	2h	97	47820	2360
	3h	97	68670	2606
Reality Mining	1j	74	20114	466
	2j	80	31509	581
	3j	81	60587	719
Taxi	1j	136	90680	876
	2j	186	211442	2000
	3j	251	299520	2907

Chapitre 5

Métriques

L'information contenue dans les flots de liens peut être quantifiée de plusieurs façons. Par exemple, on peut considérer le nombre d'interactions entre deux nœuds ou la densité du voisinage d'un nœud. Plus généralement, les méthodes existantes se concentrent soit sur les caractéristiques structurelles (dans le cas de la prédiction de liens) [AHCSZ06, LZ11], soit sur les caractéristiques temporelles (dans le cas de la prédiction de séries temporelles) [dSSP12]. Elles peuvent également utiliser des caractéristiques extérieure comme l'age ou le statut des individus pour déterminer les interactions les plus susceptibles d'avoir lieu. Cependant ce type d'information ne sera pas étudié dans le cadre de ce travail. Nous voulons que notre protocole soit capable d'utiliser de façon cohérente différents types de métriques capturant ces informations. De plus, nous voulons utiliser des métriques adaptées aux flots de liens, combinant l'information structurelle et temporelle.

Nous allons ici présenter la façon dont notre protocole représente l'information apportée par les métriques, ainsi que différentes métriques que nous utiliserons dans la suite.

5.1 Normalisation

Les valeurs absolues de chaque métrique pouvant être très différentes, il convient de normaliser celles-ci. Cette étape n'est pas nécessaire pour le fonctionnement du protocole mais permet de faciliter l'apprentissage des poids des différentes métriques durant la combinaison des fonctions par l'algorithme, qui fonctionne mieux lorsque les dimensions des paramètres sont du même ordre de grandeur. Elle permet aussi de faciliter la comparaison de l'importance de chaque métrique dans la prédiction. Nous obtenons les métriques en normalisant par le maximum des intégrales de la métrique sur chaque paire de nœuds durant la période durant laquelle celles-ci ont été calculées :

$$\overline{m}_{uv}(t) = \frac{m_{uv}(t)}{max_{uv} \int_{\Omega'}^{A'} m_{uv}(x)dx}$$

où m_{uv} est la fonction représentant la métrique m pour la paire uv, $\overline{m}_{uv}$ est la fonction de métrique normalisée et A' et Ω' sont les temps de début et fin de la période de validation ou prédiction. Nous désignons l'ensemble des fonctions de métriques normalisées par $\mathcal{M}$.

5.2 Métriques de prédiction de liens

Nous allons d'abord décrire des métriques issues de la prédiction de liens dans les graphes dans le contexte de la prédiction d'activité dans les flots de liens, notamment le nombre de voisins communs entre deux nœuds u et v, et d'autres métriques dérivées de celle-ci. Notons que les métriques présentées ici sont des fonctions indépendantes du temps. En effet, pour les métriques présentées, l'information capturée indique la vraisemblance d'apparition de liens, sans précision sur le temps d'apparition desdits liens. Ces métriques sont basées sur les propriétés du voisinage de chaque nœud dans un graphe. Nous définissons le voisinage d'un nœud u dans un flot de liens $L = (T, V, E)$ comme $\mathcal{N}_E(u) = \{v : \exists(t, uv) \in E\}$.

Nous définissons dans cette section les métriques utilisées et considérerons la distribution des valeurs obtenues pour chaque paire de nœuds pour chacun des jeux de données sur chacune des périodes définies au Chapitre 4.

5.2.1 Nombre de voisins commun

Le *nombre de voisins communs* (VC) est une mesure classique en prédiction de liens dans les graphes. Son importance en prédiction se base sur l'hypothèse que deux nœuds ayant un grand nombre de voisins en commun sont susceptibles d'avoir des interactions dans le futur. Nous définissons le nombre de voisins communs entre deux nœuds u et v comme suit :

$$VC_{E,uv} = |\mathcal{N}_E(u) \cap \mathcal{N}_E(v)|$$

Nous présentons en Figure 5.1, la distribution cumulative du nombre de voisins communs pour chaque période d'entraînement de chaque jeu de données. Notons que, comme pour la fonction de prédiction, plus que la valeur absolue des métriques présentées ici c'est leurs valeurs relatives qui importe. C'est la comparaison des métriques entre les différentes paires de nœuds qui nous permettra de déterminer l'activité prédite. De ce fait, nous nous intéresserons plus ici à la façon dont les valeurs de chaque métrique sont réparties qu'aux valeurs des métriques elles-mêmes.

Nous voyons que *Highschool*, *Reality Mining* et *Taxi* présentent des allures qualitativement similaires, la plupart des paires étant réparties sur les valeurs les plus faibles avec quelques paires ayant un nombre de voisins communs plus élevé. Sur le jeu de données

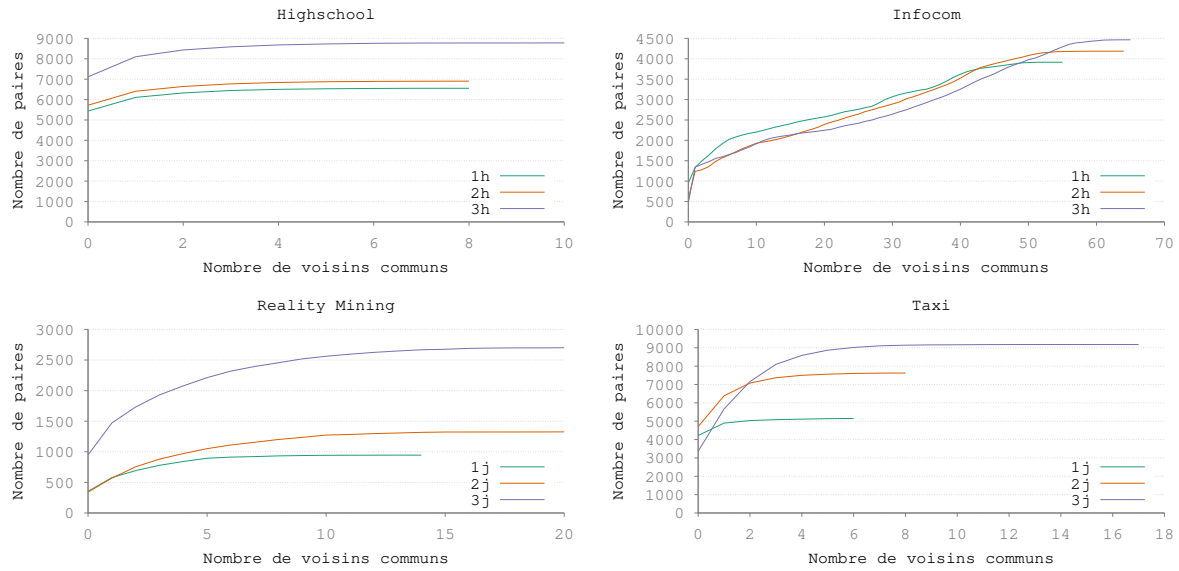


FIGURE 5.1 – Fonctions de distribution cumulative du nombre de voisins communs pour chaque période d'entraînement de chaque jeu de données.

Infocom, nous observons l'absence de paires ayant un nombre de voisins en commun significativement plus élevé que les autres. La distribution du nombre de voisins communs s'étale relativement uniformément pour des nombres de voisins communs allant jusqu'à 40 ou 55 suivant la période d'entraînement considérée.

Le nombre de paires n'ayant aucun voisin en commun varie significativement d'un jeu de données à l'autre ainsi que suivant la longueur de la période considérée. *Highschool* présente le plus grand pourcentage de paire n'ayant aucun voisin en commun, entre 81 et 82% lors des trois expériences. Seule l'expérience d'une heure du jeu de données *Taxi* présente un taux similaire avec 81% des paires n'ayant aucun voisin commun. Les autres expériences présentent des proportions de paires n'ayant pas de voisins en commun plus faible, particulièrement pour *Infocom*, où celles-ci sont entre 10 et 24%.

Ces courbes mettent en évidence des différences structurelles notables entre les différents jeux de données. Notons en particulier que le jeu de données *Highschool* présente les nombres de voisins commun les plus faibles tandis que le jeu de données *Infocom* semble présenter des paires de nœuds au voisinage commun plus dense.

5.2.2 Indice de Jaccard

L'*indice de Jaccard* (IJ) [Jac01], est proche du nombre de voisins communs. Il est défini comme le nombre de voisins communs divisé par le cardinal de l'union des voisins de chaque nœud. Il a pour but d'apporter une information similaire au nombre de voisins communs tout en diminuant l'impact des nœuds à fort degré.

$$IJ_{E,uv} = \frac{|\mathcal{N}_E(u) \cap \mathcal{N}_E(v)|}{|\mathcal{N}_E(u) \cup \mathcal{N}_E(v)|}$$

Les distributions des indices de Jaccard pour chaque jeu de données sont présentées en Figure 5.2.

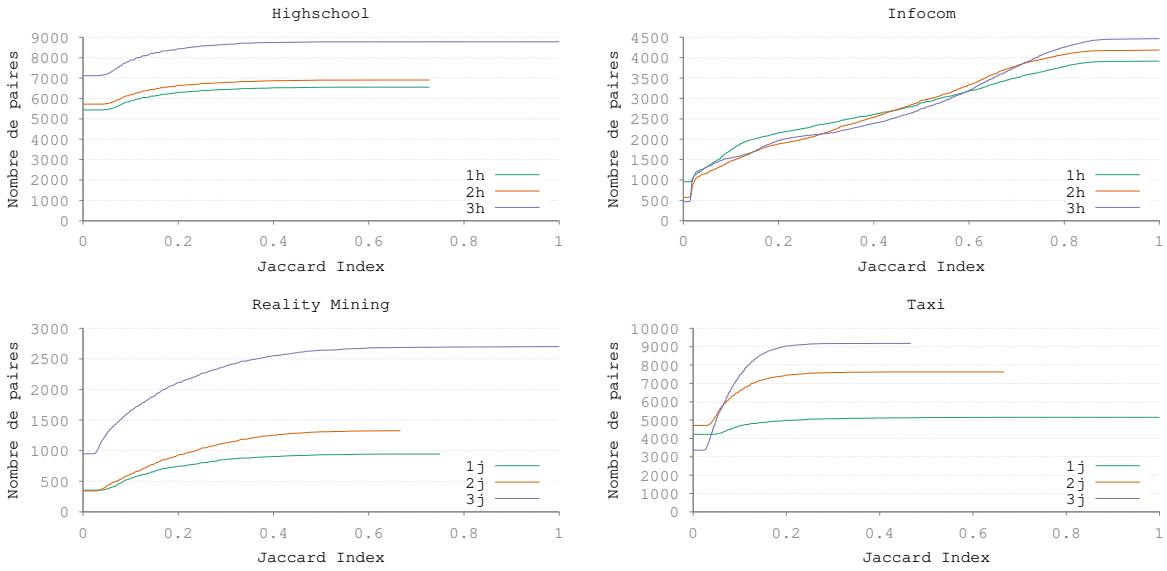


FIGURE 5.2 – Fonctions de distribution cumulative de l'indice de Jaccard pour chaque période d'entraînement de chaque jeu de données.

Nous observons des courbes aux allures similaires que pour le nombre de voisins communs. On peut noter que contrairement au nombre de voisins communs, les périodes d'entraînement plus longues ne sont pas nécessairement synonymes d'indice de Jaccard plus élevés. Cela peut s'observer sur le jeu de données *Reality Mining* entre les périodes d'entraînement d'une et deux heures, et plus particulièrement sur *Taxi*, où les plus courtes périodes d'entraînement présentent des indices maximum plus élevés.

5.2.3 Indice de Sørensen

L'*indice de Sørensen* [Sør48] peut être compris comme une alternative à l'indice de Jaccard. Le nombre de voisins communs n'est alors plus divisé par le cardinal de l'union des voisinages des nœuds mais par la somme de leur cardinal :

$$IS_{E,uv} = \frac{2 \cdot |\mathcal{N}_E(u) \cap \mathcal{N}_E(v)|}{|\mathcal{N}_E(u)| + |\mathcal{N}_E(v)|}$$

Étant très proche des autres mesures présentées, les distributions sont semblables aux courbes associées aux autres métriques structurelles. Celles-ci sont présentées en Annexe A.

5.2.4 Indice d'Adamic-Adar

L'*indice d'Adamic-Adar* [AA03] est spécifiquement conçu pour la prédiction de liens dans les réseaux sociaux, en diminuant le poids des paires de nœuds partageant un voisinage de fort degré, avec l'intuition que la significativité d'un lien pour un nœud de très fort degré est moindre que pour un nœud de faible degré :

$$AA_{E,uv} = \sum_{w \in \mathcal{N}_E(u) \cap \mathcal{N}_E(v)} \frac{1}{\log |\mathcal{N}_E(w)|}$$

Le choix du logarithme permet d'accentuer l'écart entre les paires de nœuds ayant peu de voisins par rapport à l'écart entre les paires ayant un large voisinage. Étant à nouveaux d'allures très similaires, les distributions sont présentées en Annexe A.

5.2.5 Indice d'allocation de ressource

L'*indice d'allocation de ressource* [ZLZ09] est proche de l'indice d'Adamic-Adar mais attribue différemment les poids associés au degré des voisins communs :

$$AR_{E,uv} = \sum_{w \in \mathcal{N}_E(u) \cap \mathcal{N}_E(v)} \frac{1}{|\mathcal{N}_E(w)|}$$

Nous présentons les distributions associées en Annexe A.

5.3 Métriques structurelles pondérées

Les métriques présentées précédemment décrivent des caractéristiques structurelles du graphe induit par le flot de lien. Nous allons ici présenter des métriques pondérées, permettant de capturer à la fois des informations structurelles et des informations agrégées sur la dynamique du système. Ce sont des variations pondérées des mesures de prédiction de liens présentées précédemment [TLL14], calculées sur le graphe induit par le flot de liens, la pondération de chaque lien traduisant le nombre d'interactions de chaque paire de nœuds.

Ces métriques utilisent la notion d'activité entre deux nœuds, présentée au Chapitre 2.2.3, définie comme $\mathcal{A}_E(u, v) = |\{(t, uv) \in E\}|$. Elles contiennent donc de l'information sur

l'activité de la paire de nœuds agrégée durant la période observée. Cependant, comme les métriques précédentes, elles ne permettent pas d'identifier des instants spécifiques durant lesquels l'apparition de liens est plus vraisemblable que dans d'autres. Elles seront donc également représentées par des fonctions constantes au cours du temps.

5.3.1 Nombre de voisins communs pondéré

Le *nombre de Voisins Communs Pondéré* (VCP) conserve l'idée que deux nœuds sont plus susceptibles d'interagir dans le futur s'ils ont de nombreux voisins en commun tout en insistant sur les voisins communs ayant eu un grand nombre d'interactions avec les deux nœuds :

$$VCP_{E,uv} = \sum_{w \in \mathcal{N}_E(u) \cap \mathcal{N}_E(v)} \mathcal{A}_E(u,w) \cdot \mathcal{A}_E(v,w)$$

Nous présentons en Figure 5.3 la distribution des valeurs de cette métrique. Nous pouvons voir que les valeurs sont nettement moins uniformément réparties sur les différentes paires de nœuds qu'avec la mesure structurelle d'origine. C'est notamment le cas pour les jeux de données *Highschool* et *Taxi* : entre 98 et 99% des paires ont des valeurs de moins d'un dixième de la valeur maximale obtenue pour chaque expérience. Concernant *Infocom* ces proportions sont entre 69 et 73% et pour *Reality Mining* entre 89 et 94%.

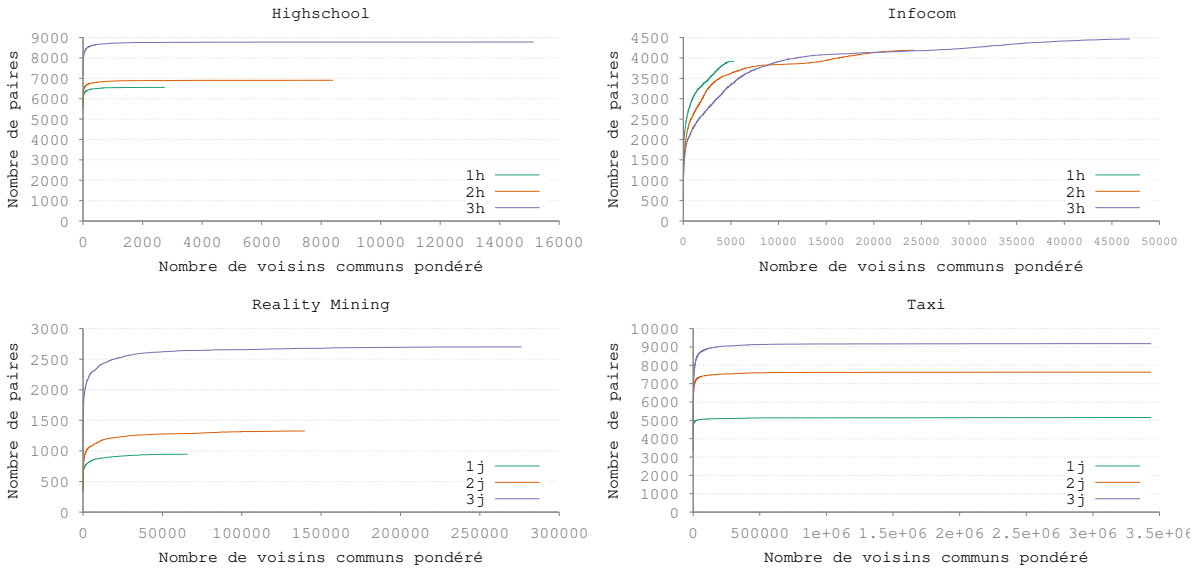


FIGURE 5.3 – Fonctions de distribution cumulative du nombre de voisins communs pondéré pour chaque période d'entraînement de chaque jeu de données.

5.3.2 Autres métriques pondérées

De la même façon, nous définissons les autres métriques pondérées :

— L'*Indice de Sørensen Pondéré* (ISP) est similaire à l'Indice de Sørensen mais prend en compte l'activité de chaque nœud :

$$ISP_{E,uv} = \frac{\sum_{w \in \mathcal{N}_E(u) \cap \mathcal{N}_E(v)} \mathcal{A}_E(u,w) + \mathcal{A}_E(v,w)}{\sum_{k \in \mathcal{N}_E(u)} \mathcal{A}_E(u,k) + \sum_{k \in \mathcal{N}_E(v)} \mathcal{A}_E(v,k)}$$

— L'*Indice d'Adamic-Adar Pondéré* (AAP) diminue le poids des voisinages de haut degré et de haute activité :

$$AAP_{E,uv} = \sum_{w \in \mathcal{N}_E(u) \cap \mathcal{N}_E(v)} \frac{1}{\sum_{k \in \mathcal{N}_E(w)} \log(\mathcal{A}_E(w,k))}$$

— L'indice d'*Allocation de Ressource Pondéré* (ARP) est similaire à l'indice d'Adamic-Adar pondéré, mais attribue les poids de façons différentes :

$$ARP_{E,uv} = \sum_{w \in \mathcal{N}_E(u) \cap \mathcal{N}_E(v)} \frac{1}{\sum_{k \in \mathcal{N}_E(w)} \mathcal{A}_E(w,k)}$$

Pour chacune de ces métriques, les distributions cumulatives montrent un comportement qualitativement similaire aux métriques structurelles associées. Les figures correspondantes sont reportées en Annexe A.

5.4 Métriques temporelles

Le formalisme des flots de liens permet de modéliser l'information temporelle sur le comportement du système. Nous allons définir des métriques capturant spécifiquement des caractéristiques temporelles pour permettre à notre protocole de l'utiliser pour améliorer la prédiction.

5.4.1 Extrapolation de l'activité

Nous allons tout d'abord définir une métrique, l'extrapolation de l'activité passée des paires de nœuds, qui permettra de capturer les répétitions de liens dans le flot. Afin de définir cette métrique nous allons utiliser l'activité $\mathcal{A}_E(u,v)$ entre u et v durant le flot L=(T,V,E). Dans la suite nous nous référerons à cette métrique comme l'*Extrapolation de l'Activité(EA)* :

$$EA_{E,uv} = |\{(t,uv) \in E\}|$$

Nous pouvons voir en Figure 5.4 les distributions de l'extrapolation de l'activité pour chaque période d'entraînement pour chaque paire ayant eu au moins une interaction durant la période d'entraînement. Nous pouvons voir que pour chaque jeu de données, les paires de nœuds ont des niveaux d'activité variés. Pour la distribution présentant le moins de diversité, sur le jeu de données *Taxi* pour des périodes d'une heure, 85% des paires considérées ont eu moins d'un dixième de l'activité de la paires ayant eu le plus grand nombre d'interactions. Notons particulièrement que le jeu de données *Infocom* présente bien plus de paires de nœuds actives qu'*Highschool*, prés de 2000 pour la période de trois heures contre un peu plus de 400. Notons également qu'avec 90 interactions durant une période de trois heures les paires les plus actives d'*Infocom* ont interagi à tout les pas de temps du jeu de données.

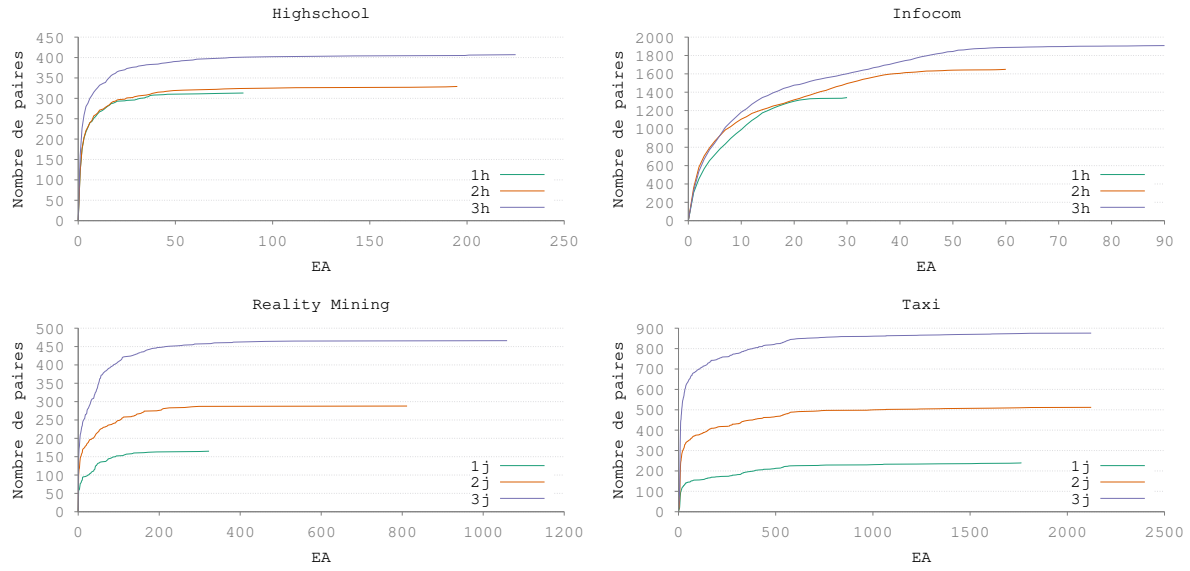


FIGURE 5.4 – Fonctions de distribution cumulative de l'extrapolation de l'activité pour chaque période d'entraînement de chaque jeu de données.

Nous allons ensuite adapter l'extrapolation de l'activité en nous concentrant sur l'activité la plus récente durant la période d'entraînement. Ce choix est fait en partant du principe que les interactions récentes affectent plus la dynamique que les plus anciennes [DKA11].

5.4.1.1 Activité pendant une durée fixée

Nous allons considérer l'activité durant une période de temps récente δ. Pour chaque paire de nœuds, nous calculons :

$$EA_{\delta S,E,uv} = |\{(t, uv) \in E \ : \ t \in [\Omega - \delta, \Omega]\}|$$

Où Ω est le temps de fin de la période d'entraînement. Afin d'étudier l'influence de δ nous allons considérer des périodes de 100, 1000 et 10000 secondes. L'objectif est de permettre

l'utilisation de métriques capturant des aspects spécifiques de la dynamique. Dans le cas présent, nous espérons capturer la dynamique durant la fin de la période d'entraînement relative à différentes échelles de temps.

Nous allons tout d'abord considérer l'activité durant la plus courte période, les 100 dernières secondes. Cette métrique capture l'information sur une très courte période avant la fin de la période d'observation. La granularité des jeux de données *Infocom* et *Reality Mining* ne permet pas d'utiliser la métrique EA_{100S} sur ceux-ci. Considérant les jeux de données *Highschool* et *Taxi*, seules 1% des paires ont interagi durant cette période.

Cela suggère que cette métrique ne permet pas de modéliser globalement le comportement du flot de liens, la grande majorité des paires n'ayant eu aucune interaction durant cette période. Cependant, il semble raisonnable d'avancer qu'une paire de nœuds ayant une interaction est fortement susceptible d'interagir de nouveau immédiatement après. Il parait justifié d'utiliser une métrique n'affectant qu'un faible pourcentage des paires de nœuds du jeu de données si celle-ci permet d'identifier des paires ayant une forte probabilité d'interagir dans la suite.

L'information capturée par ces métriques est très influencée par l'intervalle de temps minimal entre deux liens, spécifique à chaque jeu de données. Ainsi la métrique EA_{1000S} se comporte ainsi de la même manière sur le jeu de données *Reality mining* qu'EA_{100S} sur *Highschool* et *Infocom*, avec seulement 1% des paires ayant interagi durant les 1000 dernières secondes. Les jeux de données *Highschool* et *Taxi* présentent quant à eux des profils quantitativement similaires à ceux vu pour l'extrapolation de l'activité de l'intégralité de période d'observation. Les distributions associées sont reportées en Annexe A. La distribution des valeurs de cette métrique, en Figure 5.5 présente un profil plus singulier pour le jeu de données *Infocom*, avec entre 50 et 67% des paires ayant eu entre 1 et 8 interactions, réparties de façon homogène.

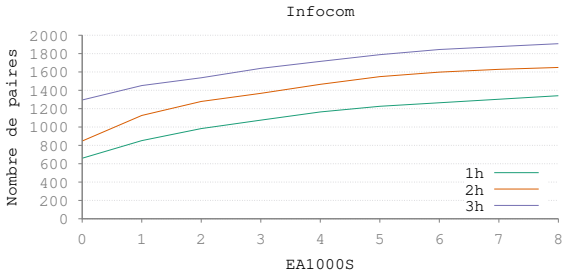


FIGURE 5.5 – Fonctions de distribution cumulative de l'extrapolation de l'activité sur 1000 secondes pour chaque période d'entraînement du jeu de données *Infocom*.

En allongeant la durée sur laquelle l'activité de chaque paire de nœuds est prise en compte, l'information capturée se rapproche de l'extrapolation de l'activité sur toute la période d'observation. Dans le cas des jeux de données *Highschool* et *Infocom*, pour les périodes d'une et deux heures, la métrique EA_{10000S} est équivalente à l'extrapolation de l'activité sur toute la période d'observation. Concernant les périodes de trois heures d'*Highschool*

et d'*Infocom*, ainsi que sur les expériences du jeu de données *Taxi*, les distributions sont très similaires à l'extrapolation de l'activité sur toute la période. Sur le jeu de données *Reality Mining*, nous observons encore une fois qu'entre 96 et 98% des paires de nœuds n'ont pas interagi durant les 10000 dernières secondes. Ces distributions sont de nouveau reportées en Annexe A.

Nous voyons ici que les métriques présentées permettent effectivement de capturer l'information durant la fin de la période d'entraînement. Cependant, il apparait que la distribution des valeurs de celles-ci sont particulièrement dépendantes du jeu de données utilisé. Cela dépend notamment de l'intervalle minimum entre deux contacts, comme dans le cas de EA_{100S} sur les jeux de données *Infocom* et *Reality Mining*. Elle est aussi dépendante de la quantité de liens apparaissant durant la période d'entraînement. Cela peut s'observer dans le cas de EA_{10000S}. Bien que très au-dessus de la granularité du jeu de données *Reality Mining*, il apparait qu'assez peu d'information n'est capturée, la plupart des paires de nœuds n'ayant pas interagi durant les 10000 dernières secondes.

5.4.1.2 Nombre de liens par seconde durant les k dernières interactions

L'objectif est ici d'obtenir une métrique permettant de capturer l'activité récente d'un lien, sur une période adaptée à sa fréquence d'interaction, moins sensible au pas de temps et au nombre d'interactions du jeu de données en évitant l'utilisation d'une durée fixe sur laquelle mesurer l'activité. Nous allons prendre en compte le nombre d'interactions par seconde de chaque paire de nœuds, entre Ω et le temps d'apparition du $k^{ème}$ liens avant Ω. Nous définissons le nombre de liens par seconde durant les k dernières interactions comme suit :

$$EA_{kL,E,uv} = k/(\Omega - t_k)$$

Avec t_k tel que $|\{(t, uv) \in L, \Omega \geq t \geq t_k\}| = k$. Cela permet de mesurer la fréquence d'interaction sur la fin de la période d'entraînement, tout en étant moins sensible à la granularité et en évitant certains des problèmes vus précédemment.

Nous prenons dans la suite le nombre de liens $k = 10$. Les courbes concernant les jeux de données *Highschool*, *Infocom* et *Taxi* présentent des allures assez similaires à celles de l'activité durant la période d'observation et sont reportées en Annexe A.

Nous pouvons cependant voir en Figure 5.6, que cette métrique permet d'obtenir, pour le jeu de données *Reality Mining*, une distribution radicalement différente de celles obtenues pour les métriques EA_{1000S} et EA_{10000S}, où la majorité des paires de nœuds présentaient des valeurs nulles. Nous observons en effet que, dans chaque expérience, 99% des paires de nœuds présentent un nombre de liens par seconde non nul, majoritairement réparti entre 0 et 0.001 liens par seconde.

Il est intéressant de constater que cette métrique permet d'obtenir des distributions de valeurs réparties sur une plage pour chaque période de chaque jeu de données, contrairement aux métriques précédentes qui étaient particulièrement dépendantes du δ considéré.

Cette métrique, qui semble moins sensible aux caractéristiques du jeu de données, ne rencontre pas de cas extrêmes, par exemple où l'immense majorité, voire l'intégralité, des paires présentent des valeurs nulles.

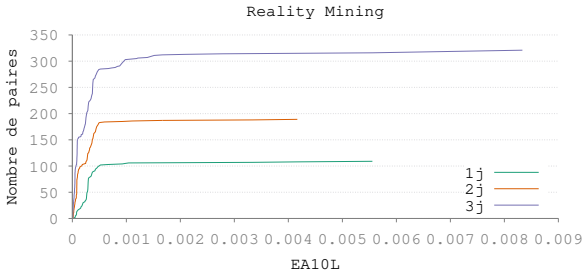


FIGURE 5.6 – Fonctions de distribution cumulative du nombre d'interactions par seconde de chaque paire de nœuds, entre Ω et le temps d'apparition du $10^{ème}$ lien avant Ω pour chaque période d'entraînement du jeu de données *Reality Mining*.

5.4.2 Temps inter-contacts et évolution de l'activité

Nous avons présenté des métriques qui nous serviront à capturer certaines caractéristiques des jeux de données étudiés. Cependant ces métriques ne tirent pas avantage de la capacité des fonctions de prédiction à représenter des variations de la vraisemblance d'apparition de liens au cours du temps.

Nous allons ici présenter des propositions de métriques adaptées aux flots de liens permettant de représenter la dépendance temporelle des fonctions de métriques afin de représenter la vraisemblance d'apparition de liens.

5.4.2.1 Distribution des temps inter-contacts

Nous allons nous concentrer sur la distribution des temps inter-contacts de chaque paire de nœuds. L'objectif ici est de définir une métrique représentant le comportement des paires de nœuds interagissant à intervalles réguliers, en faisant l'hypothèse que ce comportement se poursuivra dans le futur. Pour cela, nous allons utiliser la distribution des temps inter-contacts afin d'en calculer la moyenne et l'écart-type. Afin de représenter cette information nous allons créer un peigne de fonctions gaussiennes de l'écart-type correspondant centrées sur les emplacements supposés des prochains liens entre chaque paire de nœuds en se basant sur le temps d'apparition t_d du dernier lien entre la paire de nœuds :

$$IC_{E,uv}(t) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{((t-t_d-\mu/2)\%\mu-\mu/2)^2}{2\sigma^2}}$$

avec μ la moyenne des temps inter-contacts de la paire uv et σ leur écart-type. Cette métrique conduirait, dans le cas de nœuds ayant des interactions à des temps très réguliers,

à modéliser leur comportement par un pic à l'emplacement de chaque lien attendu, comme évoqué au Chapitre 2 et illustré en Figure 5.7.

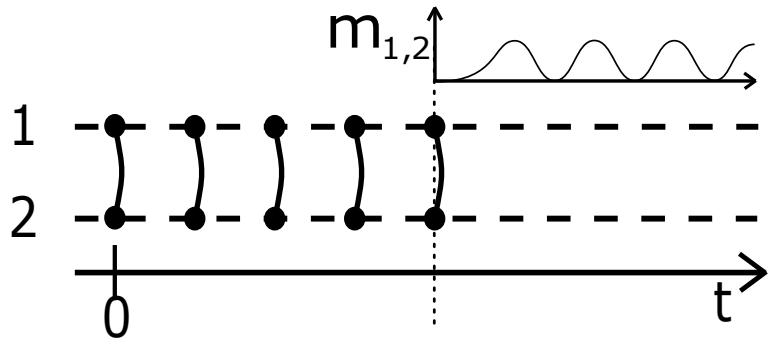


FIGURE 5.7 – Illustration d'une possible fonction de métrique dans pour deux nœuds ayant des interactions à intervalles réguliers.

Nous présentons en Figure 5.8 la distribution de l'écart-type des temps inter-contact pour chaque paire de nœuds pour le jeu de données *Infocom*. Les distributions associées aux trois autres jeux de données présentent des allures similaires et sont reportées en Annexe A, de même que les moyennes des temps inter-contacts.

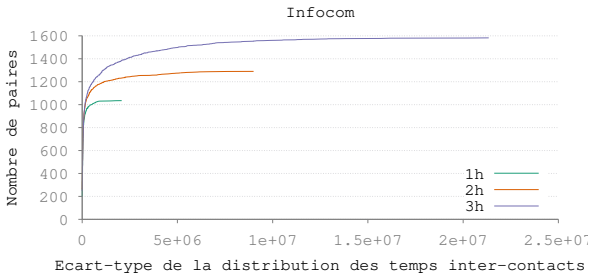


FIGURE 5.8 – Fonctions de distribution cumulative de l'écart-type des temps inter-contacts pour chaque période d'entraînement du jeu de données *Infocom*.

La distribution des écarts-types sur le jeu de données *Infocom* montre que ceux-ci sont distribués de manière hétérogène, avec entre 20 et 25% des paires dont l'écart type de la distribution des temps inter-contacts est inférieur à la longueur de la période d'observation. C'est un résultat attendu, peu de liens ayant des interactions très régulières. Nous nous attendons à ce que cette métrique permette de détecter un type spécifique d'interactions se produisant de manière quasi périodique.

5.4.2.2 Régression linéaire de l'activité

Une approche simple utilisant l'activité passée comme une série temporelle consiste à modéliser l'activité comme une fonction affine du temps et de minimiser l'écart de cette

fonction avec l'activité durant la période d'entraînement. Cela permet de saisir la tendance lors de l'étude de données non stationnaire.

La succession d'apparitions de liens entre une paire de nœuds étant un ensemble de temps d'apparition elle ne se prête pas directement à la modélisation en série temporelle. En effet, les liens étant instantanés nous ne disposons pas d'une variable variant durant la période de prédiction. Il convient alors de diviser la période de prédiction en plusieurs intervalles afin d'obtenir la moyenne du nombre de liens apparus au cours du temps sur chaque intervalle. Nous obtenons finalement pour chaque paire de nœuds une fonction de la forme :

$$FL_{E,uv}(t) = max(a_{E,uv} \cdot t + b_{E,uv}, 0)$$

Nous présentons en Figure 5.9 les distributions des valeurs des deux paramètres présentés pour le jeu de données *Highschool*. Les autres distributions sont reportées en Annexes A

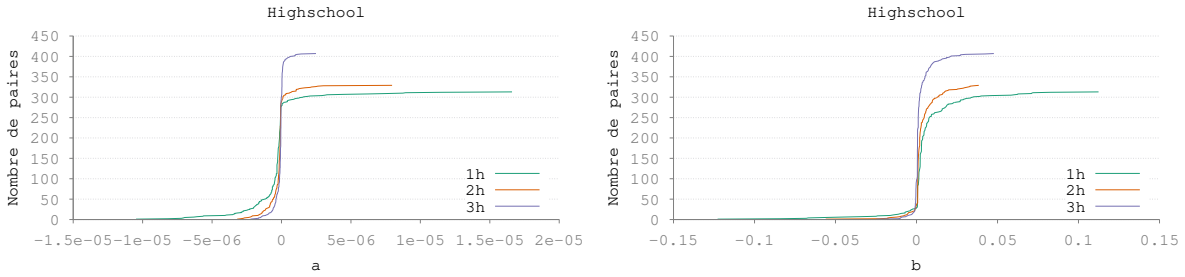


FIGURE 5.9 – Fonctions de distribution cumulative de coefficients $a_{E,uv}$ et $b_{E,uv}$ pour chaque période d'entraînement pour le jeu de données *Highschool*.

Concernant les distributions des paramètres $a_{E,uv}$, elles se concentrent sur de faibles valeurs du paramètre, de l'ordre de $\pm 10^{-5}$. La plupart des métriques sont donc des fonctions variant lentement au cours du temps durant les périodes de prédiction. Les paramètres $b_{E,uv}$ quant à eux se répartissent entre -0.15 et 0.15. L'allure de la distribution se rapproche de celle observée en Section 5.4.1 pour l'extrapolation de l'activité.

5.5 Corrélations

De nombreuses métriques présentées plus haut ne sont pas indépendantes les unes des autres. Les métriques structurelles présentées en Section 5.2 par exemple, sont toutes basées sur le voisinage local des paires de nœuds. Afin d'estimer la proximité des différentes métriques nous présentons en Figure 5.10 leurs matrices de corrélation sur les jeux de données étudiés. Les métriques ont été calculées sur les périodes d'entraînement de deux heures et deux jours et intégrées sur les périodes de validation correspondantes.

Nous observons pour chacun des jeux de données une forte corrélation des métriques structurelles et pondérées tandis que les métriques temporelles semblent également bien

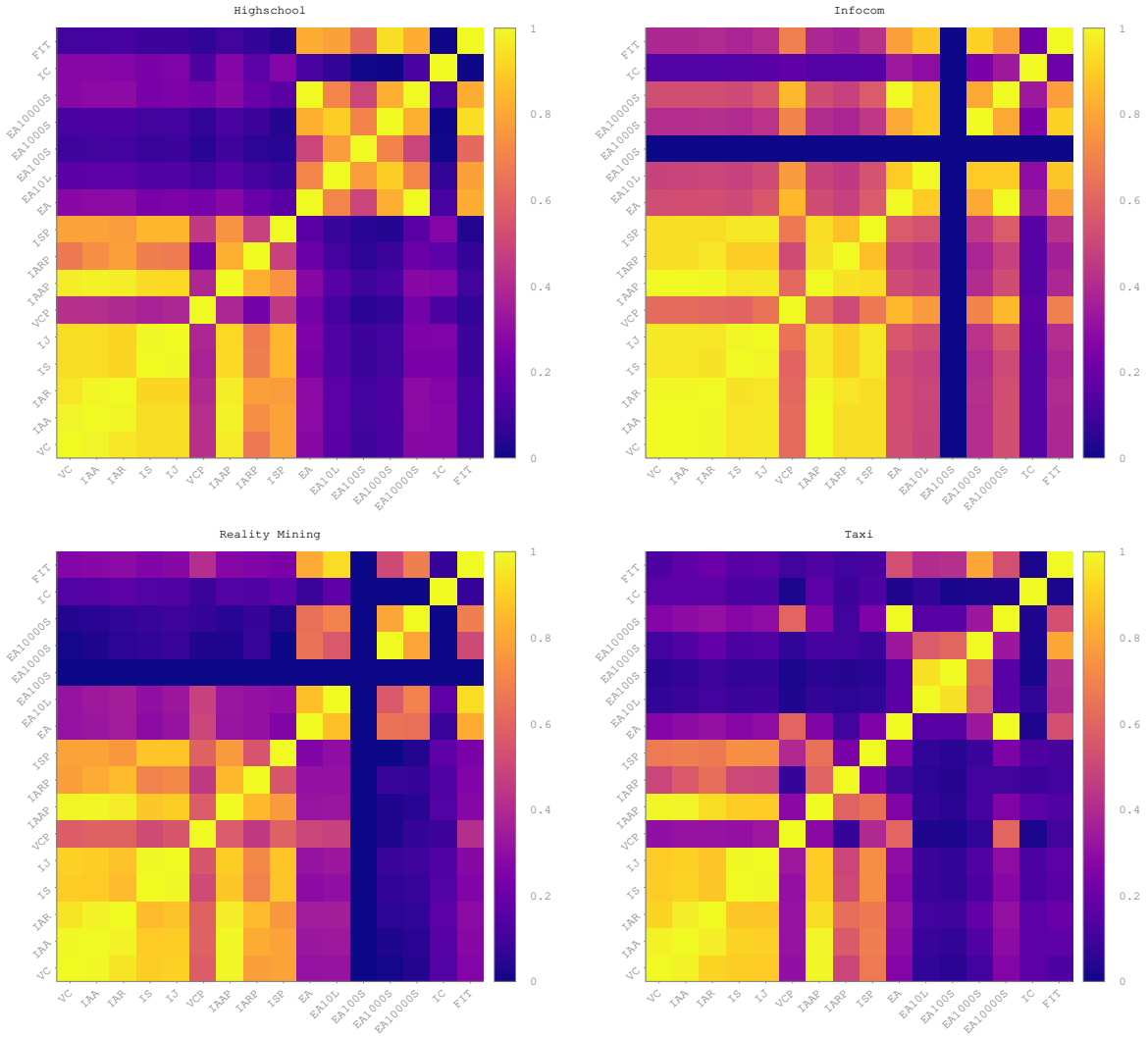


FIGURE 5.10 – Matrices de corrélation des valeurs des métriques calculées et intégrées sur les périodes de deux heures pour les jeux de données *Highschool* et *Infocom* et deux jours pour les jeux de données *Reality Mining* et *Taxi*.

corrélées entre elles. Notons cependant que le nombre de voisins pondéré et la métrique liée à la distribution des temps inter-contacts sont relativement décorrélées des autres.

Concernant *Infocom*, nous observons que les métriques temporelles et structurelles sont plus corrélées entre elles que dans le jeu de données *Highschool*. Il est également intéressant de noter que dans ce cas-ci le nombre de voisins communs pondéré est plus corrélé aux métriques temporelles qu'aux métriques structurelle et pondérées. Cela souligne le caractère hybride des métriques pondérées.

Pour le jeu de données *Reality Mining* nous notons que les métriques *EA*, *EA10L* et le fit de l'activité semblent plus corrélées aux métriques structurelles et pondérées que les autres métriques temporelles utilisées.

Sur le jeu de données *Taxi*, les démarcations entre les différents types de métriques semblent moins nettes, notamment concernant les métriques temporelles, assez peu corrélées ensemble à l'exception de *EA* et *EA10000S* et de *EA10L* et *EA100S*.

Comme attendu, dans chaque jeu de données, les métriques structurelles basées sur le *nombre de voisins communs* sont très corrélées. On constate moins de corrélation entre les différentes métriques pondérées. La corrélation des métriques temporelles entre elles semble plus dépendante du jeu de données utilisé, car plus sensible à la fréquence d'apparition des liens dans le système. Cela apparait nettement dans le jeu de données *Taxi* dans lequel seules quelques métriques pondérées semblent corrélées.

Notons également que la métrique basée sur la distribution des temps inter-contacts semble systématiquement décorrélée des autres métriques. Nous verrons dans la suite si cela indique que cette métrique permet de capturer des caractéristiques du comportement des paires de nœuds inaccessibles aux autres mesures.

5.6 Conclusion

Nous avons vu dans ce chapitre les différentes métriques que nous allons utiliser dans la suite ainsi que certaines propriétés de celles-ci. L'étude des matrices de corrélation nous permet d'identifier des groupes de métriques capturant des comportements proches les uns des autres. Intuitivement, nous pouvons nous attendre à voir notre algorithme combiner ces diverses informations pour optimiser notre prédiction. Nous verrons dans la suite les résultats de prédiction de notre protocole sur les différentes périodes des jeux de données en utilisant les métriques présentées ici.

L'étude des caractéristiques des flots de liens et la création de nouvelles métriques adaptées permettrait à notre protocole de modéliser des comportements plus complexes. Nous pourrions par exemple envisager d'intégrer des recherches de motifs dans le flot d'entraînement et d'identifier des motifs typiques afin d'améliorer la prédiction de la dynamique à court terme.

Chapitre 6

Expérimentations

Dans cette section nous étudions les résultats de notre méthode de prédiction de l'activité sur les différents jeux de données décrits au Chapitre 4 afin d'identifier les points forts de la procédure ainsi que les principaux enjeux pour améliorer la prédiction. Nous avons d'abord implémenté le protocole vu au Chapitre 3 et documentons cette implémentation en Annexe C. Nous étudierons plus précisément comment la méthode d'apprentissage répartit les poids entre les métriques combinées. Puis nous nous pencherons sur certains aspects du protocole, tel que l'influence du nombre total de liens prédits sur la qualité de la prédiction. Nous utiliserons enfin un protocole d'apprentissage simplifié afin d'étudier l'influence des différents types de métriques dans la prédiction de différentes catégories de liens.

Pour rappel, notre protocole présenté au Chapitre 3 divise le flot d'entrée L en deux sous-flots, le flot d'entraînement $L_1 = (T_1, V, E_1)$ avec $T_1 = [A_1, \Omega_1]$ et le flot de validation de la phase d'apprentissage et d'observation de la phase de prédiction $L_2 = (T_2, V, E_2)$ avec $T_2 = [A_2, \Omega_2]$. Les valeurs des paramètres α_m sont alors choisies grâce à un algorithme de descente de gradient optimisant la prédiction de l'activité durant T_2 en utilisant l'information contenue dans L_1. Les métriques de prédiction du Chapitre 5 sont ensuite calculées sur le sous-flot d'observation L_2, afin de prédire l'activité dans le sous flot réel $L' = (T', V, E')$ avec $T' = [A', \Omega']$.

Comme présenté au Chapitre 4, durant nos expériences, la durée des périodes d'entraînement, validation, observation et prédiction sont choisie égales. Nous utiliserons des périodes d'une, deux et trois heures commençant à 8h30 et 9h pour le jeu de données *Highschool* et *Infocom* respectivement et des périodes d'un, deux et trois jours commençant un jeudi pour à 1h30 pour *Reality Mining* et un vendredi à 8h00 pour le jeu de données *Taxi*.

6.1 Résultats expérimentaux

Nous allons étudier les résultats de notre protocole appliqué sur les différentes périodes des jeux de données présentés au Chapitre 4. Dans chaque expérience, toutes les métriques présentées au Chapitre 5 sont utilisées.

6.1.1 Qualité de la prédiction

Les résultats de ces différentes expériences sont présentés en Table 6.1. Nous rapportons le F-score, la précision et le rappel pour chaque expérience, ainsi que le nombre de liens prédits et apparus durant la période de prédiction. Pour plus de clarté nous noterons le nombre de liens prédits A_p et le nombre de liens réellement apparus A_r. Les valeurs présentées sont les moyennes et les écart-types obtenus sur 10 prédictions pour chacune des expériences.

TABLE 6.1 – F-score, précision, rappel et nombre de liens prédits et apparus.

Dataset	Durée	F-score	Précision	Rappel	A_p	A_r
Highschool	1h	0.45 ± 0.00	0.33 ± 0.00	0.70 ± 0.00	1123	534
	2h	0.16 ± 0.00	0.17 ± 0.00	0.15 ± 0.00	1751	2028
	3h	0.29 ± 0.00	0.22 ± 0.00	0.42 ± 0.00	3072	1608
Infocom	1h	0.59 ± 0.00	0.58 ± 0.00	0.59 ± 0.00	8167	8138
	2h	0.54 ± 0.00	0.50 ± 0.00	0.60 ± 0.00	16737	13939
	3h	0.66 ± 0.00	0.63 ± 0.00	0.70 ± 0.00	22568	20310
R.Mining	1j	0.47 ± 0.00	0.41 ± 0.00	0.55 ± 0.00	6105	4518
	2j	0.09 ± 0.00	0.05 ± 0.00	0.64 ± 0.01	18537	1434
	3j	0.17 ± 0.03	0.30 ± 0.05	0.12 ± 0.02	11395	28692
Taxi	1j	0.19 ± 0.00	0.20 ± 0.00	0.18 ± 0.00	872173	943173
	2j	0.10 ± 0.00	0.08 ± 0.00	0.16 ± 0.00	1904076	877069
	3j	0.17 ± 0.00	0.20 ± 0.00	0.14 ± 0.00	1843329	2702338

6.1.1.1 Highschool

Nous allons tout d'abord considérer les expériences sur le jeu de données *Highschool*. Nous pouvons voir que durant l'expérience d'une heure, l'algorithme prédit environ 50% plus de liens que réellement apparus. Nous obtenons un rappel de 0.70, ce qui veut dire que la plupart des liens apparus ont été correctement prédits. Cependant l'excès de liens prédits mène à une précision de 0.33, indiquant un fort taux de *faux positifs*, beaucoup de liens prédits n'étant pas apparus.

Concernant l'expérience de deux heures, nous observons une baisse significative de la qualité de la prédiction, visible aussi bien sur la précision que sur le rappel (et donc sur le

F-score). Cette baisse est probablement due au fait que la période d'observation s'étend sur la pause déjeuner des étudiants. Le nombre total de liens prédits sur la période de prédiction sous-estime le nombre de liens apparaissant dans le flot réel, autrement dit $A_p < A_r$. Cela reflète que l'activité du flot d'observation n'est pas aussi élevée que durant la période de prédiction.

Cependant nous pensons que la raison principale de la baisse de la qualité de la prédiction vient du fait que la pause déjeuner est une occasion pour des interactions qui diffèrent de celles apparaissant durant les heures de classes ce qui semble avoir un impact significatif. Nous explorons plus en détail cette question en Section 6.1.3.

L'expérience de trois heures présente une amélioration de la précision et du rappel comparé à l'expérience de deux heures. Nous pensons que les périodes d'observations plus longues diminuent l'effet des changements de comportement, comme celui ayant lieu durant la pause déjeuner. L'activité globale prédite est plus élevée que l'activité réelle. Bien que cela ait un impact sur la qualité de la prédiction, nous pouvons voir que cela ne conduit pas à une baisse de la qualité aussi significative que lors de l'expérience de deux heures.

Ces expériences montrent que la qualité des prédictions sur de longues périodes peut être négativement affectée par les fortes variations de comportement du système au cours du temps. Dans cet exemple elles sont principalement dues à l'emploi du temps du lycée mais on peut penser que l'on aurait le même genre de problème sur des données qui alterneraient entre différentes dynamiques au cours du temps.

6.1.1.2 Infocom

Pour le jeu de données *Infocom*, les performances d'une expérience à l'autre sont plus stables que pour *Highschool*, en particulier en comparant les nombres de liens prédits et apparus. En effet les différences entre A_p et A_r restent inférieures à 20% pour chaque expérience. Nous avons observé cela dans le profil d'activité du jeu de données, comme présenté au Chapitre 4. Nous observons également de meilleures performances, avec des F-score entre 0.54 et 0.66. Cela suggère que, lors la conférence durant laquelle ce jeu de données a été collecté, les différences de comportement entre les présentations et les pauses sont moins marquées que précédemment. À une bien plus petite échelle que pour *Highschoool*, nous observons également une baisse de la qualité de la prédiction durant les périodes de deux heures comparée aux périodes d'une heure, le F-score passant de 0.59 à 0.54, mais cet effet disparait sur les plus longues périodes : les périodes de trois heures présentent même un F-score plus élevé que dans le cas de périodes d'une heure, de 0.66.

6.1.1.3 Reality Mining

L'expérience d'une heure sur le jeu de données *Reality Mining* présente des performances correctes, avec un F-score de 0.47. Cependant nous pouvons voir une importante baisse

de la qualité dans les expériences plus longues, particulièrement concernant les périodes de deux jours. Dans ce dernier cas, cette chute de la qualité de la prédiction semble venir de l'importante surestimation du nombre de liens prédits comparé aux liens réellement apparus avec $A_p = 12837$ et $A_r = 1434$. Concernant le cas des périodes de trois jours, la prédiction globale est 60% plus basse que l'activité réelle. Nous observons également un écart-type pour les différents indicateurs nettement plus élevé que pour les autres expériences, où ils sont systématiquement inférieurs à 0.01. La raison de cette augmentation de l'écart-type n'est pas claire. Nous supposons que dans cette expérience les valeurs du F-score dans l'espace des paramètres α_m présentent des caractéristiques pour lesquelles l'algorithme d'apprentissage n'est pas adapté. Nous nous pencherons sur ces deux derniers cas plus en détails en Section 6.1.3.

6.1.1.4 Taxi

Considérant le jeu de données *Taxi*, nous pouvons voir que la qualité de la prédiction en termes de F-score est plus basse que pour les autres jeux de données, entre 0.10 et 0.19. Cela est dû au plus grand nombre de nœuds et d'interactions dans le jeu de données, menant à une tâche de prédiction plus difficile. C'est un problème connu en prédiction de liens, le déséquilibre de classes. Nous verrons au chapitre suivant des travaux ayant développé des méthodes pour atténuer ce problème. Notons également que, comme dit au Chapitre 4, la nature du jeu de données est différente, les liens ne représentant pas des interactions sociales et la proximité géographique entre deux taxis pouvant être moins significative que la proximité d'étudiants ou de participant à une conférence. L'expérience concernant les périodes longues d'un jour montre une activité prédite proche de l'activité réelle, menant à la meilleure prédiction sur ce jeu de données. Nous pouvons également voir que pour les prédictions sur deux et trois jours, la prédiction mène à des F-scores moins élevés de 0.10 et 0.17 respectivement, et dans chaque cas, la prédiction de l'activité globale a été largement mal estimée avec un écart entre A_p et A_r d'environ un million de liens dans chaque cas.

6.1.1.5 Résumé

Ces expériences tendent à montrer que le choix de la période de prédiction joue un rôle clé dans la prédiction. Les périodes choisies ont une forte influence sur la qualité de l'extrapolation globale ainsi que sur les variations des comportements des systèmes étudiés au cours du temps, menant à d'importantes variations sur la qualité de la prédiction. Notons également les faibles écart-types obtenus sur chacune des réalisations des expériences, ce qui suggère que l'algorithme d'apprentissage est stable sur ces expériences.

6.1.2 Étude de la combinaison de métriques

Dans cette section nous allons observer la façon dont notre algorithme combine les différentes métriques afin d'optimiser la prédiction. Dans chaque cas, la hauteur de chaque barre correspond à la moyenne des coefficients obtenus pour les expériences présentées plus haut sur dix réalisations de l'expérience. Les acronymes utilisés dans la suite sont indiqués en Table 6.2.

Table 6.2 – Acronymes des métriques utilisées

Acronyme	Métrique
VC	*Nombre de voisins communs*
AA	*Indice d'Adamic-Adar*
AR	*Indice d'Allocation de ressource*
IS	*Indice de Sørensen*
IJ	*Indice de Jaccard*
VCP	*Nombre de voisins communs pondéré*
AAP	*Indice d'Adamic-Adar Pondéré*
ARP	*Indice d'Allocation de ressource Pondéré*
ISP	*Indice de Sørensen Pondéré*
EA	*Extrapolation de l'Activité*
EA10L	*Activité par seconde durant les 10 links*
EA100S	*Extrapolation de l'Activité durant les 100 dernières secondes*
EA1000S	*Extrapolation de l'Activité durant les 1000 dernières secondes*
EA10000S	*Extrapolation de l'Activité durant les 10000 dernières secondes*
Fit	*Régression linéaire de l'activité*
IC	*Distribution des temps inter-contacts*

6.1.2.1 Highscool

Les combinaisons de métriques utilisées pour le jeu de données *Highscool* sont présentées en Figure 6.1. Considérant les périodes d'une heure nous pouvons voir que le protocole utilise à la fois l'extrapolation de l'activité sur les 1000 dernières secondes et de la régression linéaire de l'activité passée, avec une petite influence de l'activité pendant les 100 dernières secondes, environ 10% de participation de cette métrique dans la combinaison.

Pour l'expérience d'une durée de deux heures cependant nous pouvons voir une plus grande diversité parmi les métriques utilisées par l'algorithme. Notons cependant qu'à l'exception du nombre de voisins communs pondéré, ce sont toutes des métriques purement temporelles.

La combinaison des métriques sur des périodes de trois heures semble montrer moins de variété dans les métriques utilisées, avec l'absence de l'utilisation de la régression linéaire

de l'activité et de la distribution de temps inter-contacts ainsi qu'une utilisation réduite de l'activité durant les 10 derniers liens, seulement 1% participation dans la combinaison.

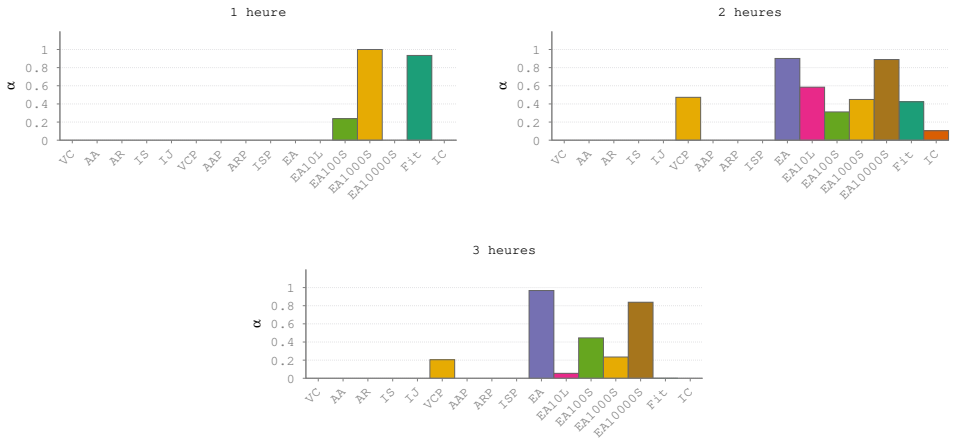


FIGURE 6.1 – Répartition des coefficients des métriques calculé par l'algorithme d'apprentissage pour le jeu de données *Highschool* pour des périodes d'une, deux et trois heures.

6.1.2.2 Infocom

Pour le jeu de données *Infocom* nous pouvons voir en Figure 6.2 que l'algorithme se concentre de nouveau sur les métriques temporelles.

Concernant la période d'une heure notons que l'activité durant la fin de période d'observation avec l'*EA1000S* représente 80% des métriques utilisées . Il utilise également faiblement l'extrapolation de l'activité sur l'ensemble de la période. Notons qu'étant donnée la période considérée, l'activité durant les 10000 dernières secondes est équivalente à l'*EA*.

Pour les périodes de deux heures, l'algorithme utilise encore une fois une variété de métriques temporelles différentes, notamment la régression de l'activité et l'activité durant les 10000 dernières secondes associées au *VCP*. Notons également la présence de la distribution des temps inter-contacts à hauteur de 10% de la combinaison.

Lors de l'expérience utilisant des périodes de 3 heures, nous observons encore une fois une diminution dans la variété des métriques utilisée, le nombre de voisins communs pondéré et l'extrapolation de l'activité sur les 1000 et 10000 dernières secondes représentant 86% de la combinaison.

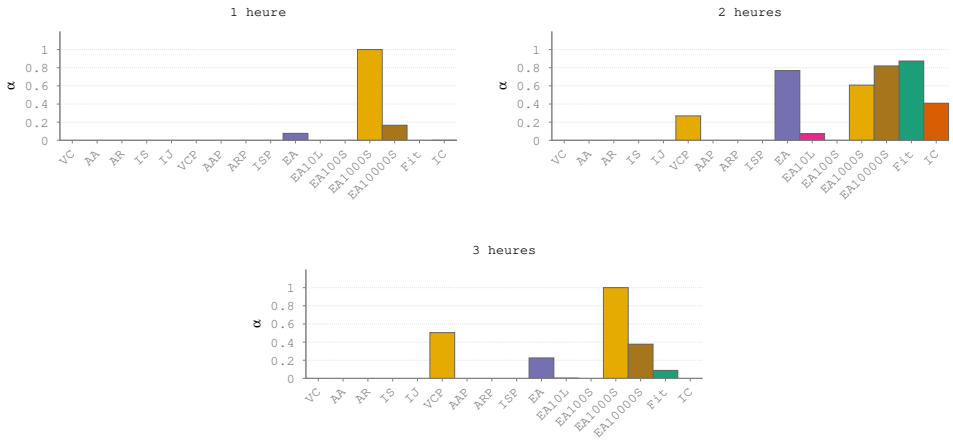


FIGURE 6.2 – Répartition des coefficients des métriques calculé par l'algorithme d'apprentissage pour le jeu de données *Infocom* pour des périodes d'une, deux et trois heures.

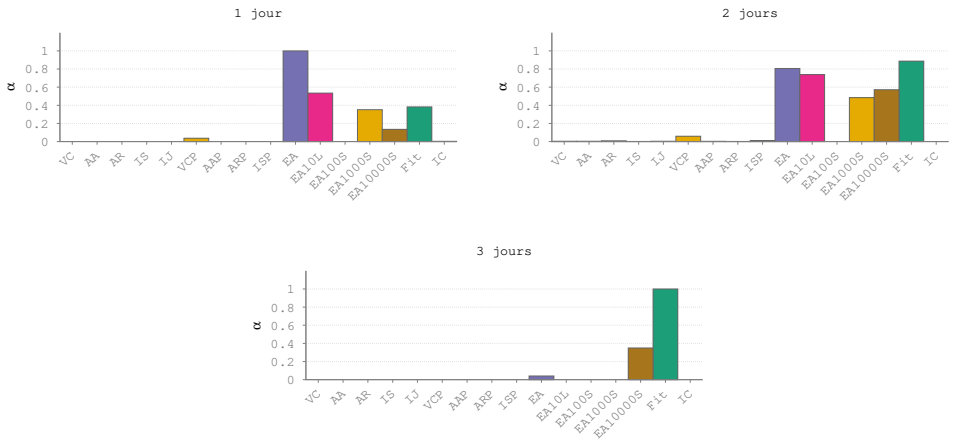


FIGURE 6.3 – Répartition des coefficients des métriques calculé par l'algorithme d'apprentissage pour le jeu de données *Reality Mining* pour des périodes d'un, deux et trois jours.

6.1.2.3 Reality Mining

Comme présenté en Figure 6.3, nous pouvons voir que les combinaisons de métriques pour les périodes d'un et deux jours privilégient de nouveau les métriques temporelles, avec en priorité l'extrapolation de l'activité pour la période d'un jour et la régression de l'activité pour celle de deux jours. Notons également l'utilisation importante de l'activité par seconde durant les 10 derniers liens dans ces expériences, qui représente 20% de la combinaison. Cela fait écho aux distributions cumulatives des valeurs des différentes métriques temporelles présentées au Chapitre 5, où nous avons vu que, pour ce jeu de données, l'*EA10S* présentait des valeurs réparties de façon plus homogène que les extrapolations de l'activité sur différentes durées où entre 96% et 98% des paires ayant eu au moins une

interaction durant la période d'observation présentaient des valeurs nulles. Notons tout de même que, malgré cela, celles-ci sont également utilisées ici notamment considérant les périodes de 2 jours.

Sur les périodes de 3 jours, la combinaison des métriques présente des caractéristiques particulières comparées aux précédentes. Durant cette expérience, notre algorithme utilise à 72% le fit de l'activité combinée avec une petite influence de l'*EA10000S*. Comme vu précédemment, nous étudierons plus en détail cette combinaison en Section 6.1.3.

6.1.2.4 Taxi

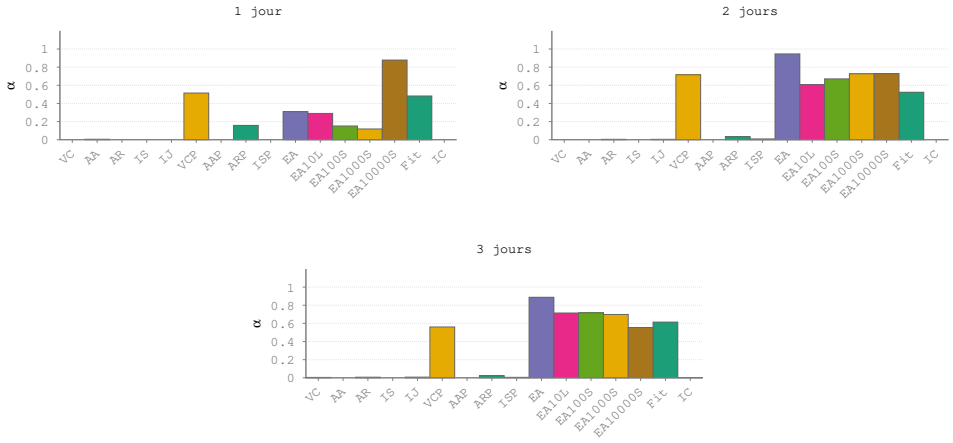


FIGURE 6.4 – Répartition des coefficients des métriques calculé par l'algorithme d'apprentissage pour le jeu de données *Taxi* pour des périodes d'un, deux et trois jours.

Finalement, considérant le jeu de données *Taxi* les combinaisons sont présentées en Figure 6.4. Pour les périodes d'une journée, nous pouvons voir que l'algorithme utilise principalement l'*EA10000S*, la régression de l'activité ainsi que le *VCP* qui représentent 48% de la combinaison. Cependant nous pouvons voir que les autres métriques temporelles sont également utilisées avec des poids plus faibles. Notons également une faible utilisation de l'indice Adamic-Adar pondéré, 5% de la combinaison.

Concernant les périodes de deux jours, nous observons une répartition équitable entre les métriques temporelles, dominées par l'*EA*. Cela tend à indiquer que toutes les échelles temporelles apportent certaines informations à la prédiction. Encore une fois nous observons une utilisation importante du *VCP*.

La combinaison sur les périodes de 3 jours présente un profil similaire. Bien qu'anecdotique étant donnés les poids des métriques en question, moins de 1% de la combinaison, notons l'apparition pour la seule fois dans cette série d'expériences de métriques structurelles, l'indice d'Adamic-Adar et de Jaccard.

6.1.2.5 Résumé

Comme nous avons pu le voir notre protocole tend à utiliser une certaine variété de combinaisons de métriques. Cependant, nous notons que dans chacune des expériences les métriques temporelles sont particulièrement favorisées. Les métriques purement structurelles sont globalement ignorées par l'algorithme, alors que certaines métriques pondérées, principalement le nombre de voisins communs pondéré, qui capturent à la fois de l'information temporelle et structurelle, sont quant à elles régulièrement utilisées. Il est également intéressant de noter que les différentes métriques temporelles, correspondant à différentes échelles de temps sont représentées, indiquant que combiner ces métriques permettent au protocole de capturer différentes informations.

Par ailleurs, un point important est que, extrapolant l'activité passée des paires de nœuds, les métriques temporelles ne sont simplement pas capables de prédire des liens entre des nœuds n'ayant eu aucune interaction dans le passé, tandis que les métriques structurelles et pondérées en sont capables. Or la prédiction de liens n'étant jamais apparu dans le passé est une tâche plus difficile que la prédiction d'observations déjà observées. Cela nous amène à penser que c'est la raison pour laquelle notre protocole tend à ignorer les métriques structurelles et à négliger la plupart des métriques pondérées. En privilégiant la prédiction de liens déjà observés, il améliore la prédiction globale au détriment de la prédiction de nouveau liens. Nous explorerons cette intuition en Section 6.3.

6.1.3 Cas des prédictions de faibles qualités

Dans cette section, nous allons nous intéresser plus en détail à trois cas vus en Section 6.1.1 qui présentent des prédictions de faibles qualités comparées aux autres expériences sur le même jeu de données.

Highschool – 2 heures

Nous allons tout d'abord nous pencher sur l'expérience de deux heures sur le jeu de données *Highschool* dont le F-score est de 0.16 tandis que les deux autres expériences présentent un F-score de 0.45 et 0.29. Nous avons noté précédemment une sous-estimation de l'activité durant la période de prédiction. Nous présentons en Figure 6.5 le profil d'activité durant les différentes périodes du protocole. Nous avons également vu au Chapitre 4 que la période de prédiction $[A', \Omega']$ s'étend de 12h30 à 11h30. Cette période correspond à la pause déjeuner, où les étudiants quittent leurs classes et ont l'opportunité de se mélanger avec les autres étudiants. Il est difficile de tirer une conclusion à partir de la Figure 6.5, étant donné que si l'activité diminue durant la période de validation/observation et augmente de nouveau durant la période de prédiction, l'activité prédite vue en Table 6.1, 1751 liens, ne semble pas assez éloignée de l'activité réelle, 2028 liens, pour expliquer seule la baisse de la qualité de la prédiction dans cette expérience.

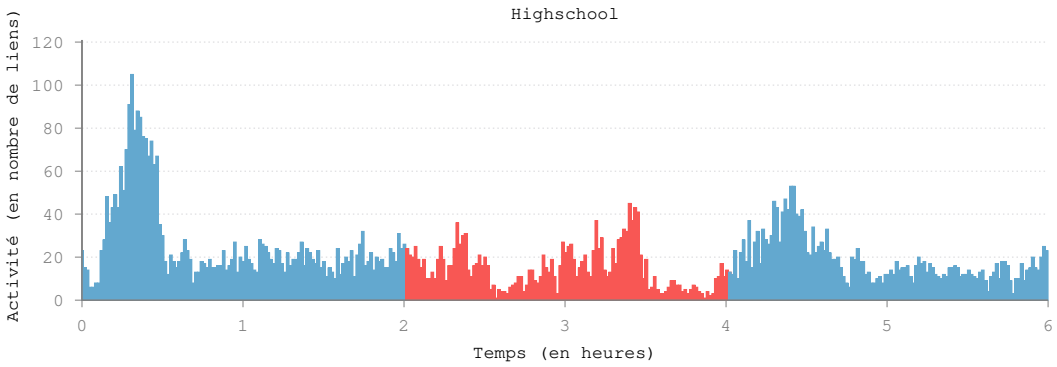


FIGURE 6.5 – Nombre de liens apparaissant au cours du temps durant l'expérience sur des périodes de 2h pour le jeu de données *Highschool* (en rouge la période de validation/observation)

Afin d'aller plus loin dans l'étude du problème, nous allons effectuer une tâche de prédiction non réaliste mais qui nous permettra d'avoir une meilleure idée des raisons derrière cette baisse de la qualité. Nous répétons l'expérience précédente, mais en fixant le nombre total de liens prédits durant la période de prédiction au nombre total de liens réellement apparus. Nous utilisons donc des informations qui ne sont normalement pas connues de l'expérimentateur lors d'une tâche de prédiction, le nombre de liens dans le flot prédit. L'évaluation de la qualité de cette expérience rapporte un F-score de 0.16, une précision de 0.16 et un rappel de 0.16. La précision et le rappel sont extrêmement proches des valeurs obtenues durant l'expérience originelle menant à un F-score identique. Cela nous montre que l'erreur dans la prédiction globale ne suffit pas à expliquer la faible qualité de la prédiction. Cela met en évidence que, dans ce cas-ci, la prédiction est particulièrement affectée par le changement de comportement des étudiants entre les heures de cours et de pause déjeuner.

Reality Mining – 2 jours

Considérons maintenant l'expérience sur des périodes de deux jours sur le jeu de données *Reality Mining*. Nous avons observé en Table 6.1 une importante baisse de l'activité globale durant la période de prédiction, de 18537 à 1434 liens. Nous montrons en Figure 6.6 le nombre de liens au cours de temps durant cette expérience. Nous observons une large baisse du nombre de liens apparaissant durant les deux derniers jours. L'expérience commençant un mardi, la période d'entraînement est donc du mardi au mercredi, la période de validation et d'observation du jeudi au vendredi, tandis que la période de prédiction s'étend du samedi au dimanche, ce qui mène à une importante baisse de l'activité.

En répétant la même expérience que précédemment, fixant le nombre global de liens prédits au nombre de liens apparus nous obtenons un F-score de 0.13, une précision de 0.13 ainsi qu'un rappel de 0.13. Bien que largement inférieures aux valeurs obtenues pour l'expérience sur des durées d'un jour, respectivement 0.47, 0.41 et 0.55, nous notons une

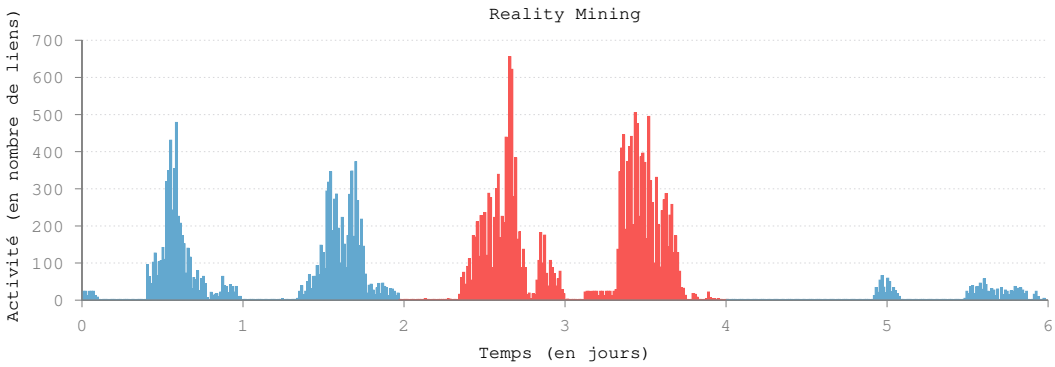


FIGURE 6.6 – Nombre de liens apparaissant au cours du temps durant l'expérience sur des périodes de 2 jours pour le jeu de données *Reality Mining* (en rouge la période de validation/observation)

amélioration significative de la qualité de la prédiction. Cela suggère que la prédiction du nombre de liens global, bien qu'affectant fortement la prédiction n'est pas ici non plus le seul facteur responsable de la faible qualité. Comme pour *Highschool*, le weekend semble montrer, en plus d'une chute de l'activité, un changement de comportement. Les interactions durant le weekend sont différentes de celles ayant lieu durant la semaine. Ces résultats mettent en évidence l'importance dans le choix de la période d'entraînement, de validation et de prédiction dans notre protocole.

Reality Mining – 3 jours

Nous allons maintenant nous pencher sur le cas plus particulier de l'expérience de trois jours sur le jeu de données *Reality Mining*. Nous avons vu que le nombre de liens prédits, 11395, est largement inférieur au nombre de liens apparus, 28692. En répétant la procédure précédente nous obtenons un F-score de 0.19, une précision de 0.19 et un rappel de 0.19. Nous notons bien ici une amélioration de la prédiction. Cependant, la particularité de cette prédiction est la combinaison de métrique utilisée, vue en Section 6.3, radicalement différente des deux autres expériences sur ce même jeu de données, privilégiant la régression linéaire de l'activité. Afin d'observer le rôle de cette métrique dans la qualité de la prédiction sur cette expérience nous allons procéder à une prédiction standard, sans fixer le nombre de liens prédits, en utilisant toutes les métriques sauf celle-ci. Nous obtenons alors un F-score de 0.41, une précision de 0.73 et un rappel de 0.29 avec la combinaison de métriques présentée en Figure 6.7, indiquant l'usage majoritaire de l'extrapolation de l'activité sur toute la période, 70% de la combinaison, et pendant les 10000 dernières secondes, 70%.

Ces valeurs significativement plus hautes indiquent que la cause réelle de la faible performance de notre protocole lors de cette expérience n'est pas due à la sous-estimation du nombre de liens dans le flot réel mais à l'usage du fit de l'activité. Alors qu'elle semble apporter une information utile dans le cadre des autres expériences, elle a ici pour ef-

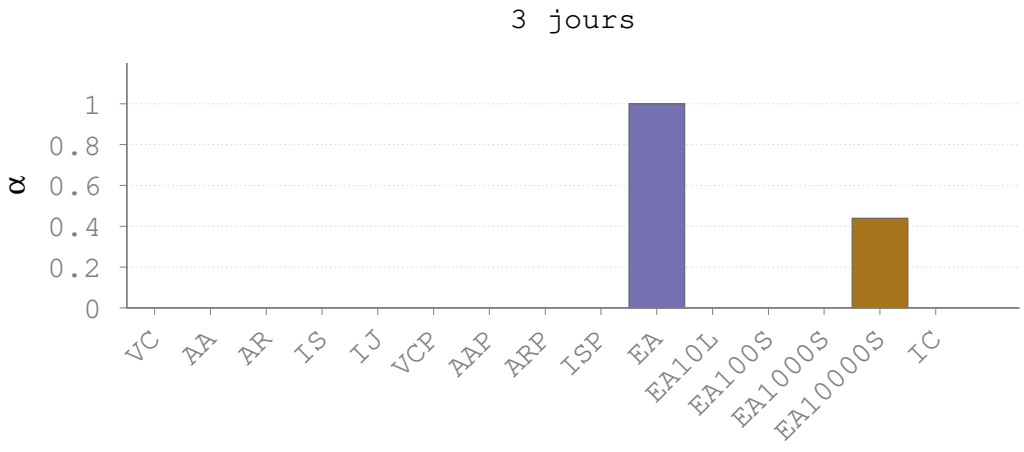


FIGURE 6.7 – Répartition des coefficients des métriques calculé par l'algorithme d'apprentissage pour le jeu de données *Reality Mining* pour des périodes de trois jours sans la régression de l'activité.

fet de pénaliser la prédiction. Elle apporte une information pertinente lors de la phase d'apprentissage de l'algorithme optimisant la prédiction du flot de validation à partir du flot d'entraînement et est donc choisie dans la combinaison des métriques. Cependant, le changement de comportement du système entre la phase d'apprentissage et la prédiction proprement dite affecte particulièrement la capacité de la métrique à capturer des informations pertinentes. Cela met en évidence le caractère critique du choix de la période d'apprentissage vis-à-vis de la période de prédiction. C'est un problème de sur-apprentissage, communément rencontré dans les tâches de prédiction.

6.2 Influence de la prédiction de l'activité globale

Nous avons vu précédemment que la prédiction de nombre de liens total durant la période de prédiction impacte la prédiction. Afin d'avoir une meilleure intuition de l'importance de ce facteur nous allons étudier de façon plus systématique son impact. En nous concentrant sur l'expérience présentant la meilleure performance, avec des périodes de trois jours sur le jeu de données *Infocom*, nous allons artificiellement faire varier le nombre de liens prédits.

Toutes les expériences présentées en Table 6.3 sont identiques à l'expérience de trois heures présentée en 6.1.1.2, excepté le nombre global de liens prédits. Prenant comme base le nombre de liens réellement apparu durant le flot de prédiction, nous allons successivement prédire 10%, 20% et 30% de liens en plus puis de liens en moins.

Comme nous pouvons le voir, la meilleure prédiction est obtenue lorsque l'on prédit le nombre exact de liens apparus ou 10% de liens en plus, avec un F-score de 0.67.

TABLE 6.3 – F-score, précision, rappel et nombre de liens prédits et apparus pour différentes sur ou sous-estimation du nombre de liens apparus.

Dataset	Pourcentage	F-score	Précision	Rappel	A_p	A_r
Infocom 3h	+30%	0.65	0.58	0.75	26403	20310
	+20%	0.66	0.61	0.73	24372	20310
	+10%	0.67	0.64	0.70	22341	20310
	0%	0.67	0.67	0.67	20310	20310
	−10%	0.66	0.70	0.63	18279	20310
	−20%	0.64	0.72	0.58	16248	20310
	−30%	0.62	0.75	0.52	14217	20310

Cependant, il est particulièrement intéressant de constater que, tandis que la précision et le rappel semblent varier significativement, de 0.58 à 0.75 et de 0.52 à 0.75 respectivement, le F-score semble nettement moins affecté par la variation du nombre de liens prédits qu'anticipé, baissant à 0.62 pour une sous-estimation de 30% et à seulement 0.65 pour une surestimation du même ordre.

6.3 Étude de la combinaison de paires de métriques

Dans cette section, nous utilisons un modèle simplifié du protocole, identique à celui présenté plus haut, excepté que nous ne mélangerons que deux métriques différentes à la fois. Cette simplification nous permet de représenter clairement les indicateurs de qualité de la prédiction en fonction des poids associés à chacune des métriques utilisées. Notre objectif est d'identifier le type de liens prédits par le protocole par les différentes combinaisons de métriques possibles. Nous allons nous concentrer sur les jeux de données *Highschool* et *Infocom*. Comme discuté dans la suite, cela nous donnera des indications supplémentaires expliquant la faible contribution des métriques structurelles durant la série d'expériences précédente.

6.3.1 Protocole expérimental simplifié

Notre protocole expérimental se concentre sur le gain de performance atteint par la combinaison de deux métriques. Nous nous concentrons ici sur les périodes d'entraînement et de validation seulement. Puis, afin de comprendre l'information apportée par chaque métrique nous combinons deux métriques à la fois, le poids de chacune étant associé à un paramètre α, selon l'équation suivante :

$$\mathcal{F}(t)_{uv} = \alpha \cdot m^1(t)_{uv} + (1 - \alpha) \cdot m^2(t)_{uv}$$

où m^1 et m^2 sont les deux métriques utilisées dans la combinaison.

6.3.2 Combinaisons de métriques temporelles et structurelles

Nous allons étudier comment différentes métriques avec différents poids affectent la prédiction sur les jeux de données *Highschool* et *Infocom*. Pour cela, nous combinons trois des métriques utilisées précédemment avec la métrique affectée du poids le plus important dans les expériences présentées ci-dessus pour chaque jeu de données. Dans chaque cas, nous combinerons en premier lieu cette métrique avec la deuxième métrique ayant le plus participé à la prédiction. Nous la combinerons ensuite avec le nombre de voisins communs et l'indice de Sørensen afin d'étudier comment notre algorithme combine une métrique temporelle avec des métriques structurelles.

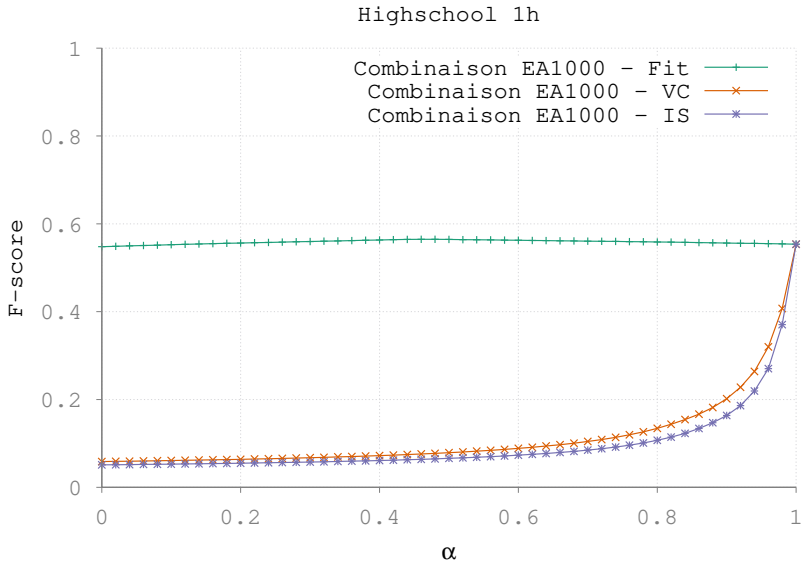


FIGURE 6.8 – F-score de la prédiction pour différents rapports de poids entre l'*EA1000S* et trois autres métriques pour le jeu de données *Highscool* sur des périodes d'une heure

Dans le cas du jeu de données *Highschool* nous utiliserons les périodes d'entraînement et de validation d'une heure. Nous prendrons $m^1 = EA1000S$, l'extrapolation de l'activité durant les 1000 dernières secondes, et combinerons successivement cette métrique avec la régression de l'activité, le nombre de voisins communs (VC) et l'indice de Sørensen (SI). Nous présentons les résultats de cette expérience en Figure 6.8. Dans chaque cas les valeurs de α les plus hautes représentent un plus grand poids de *EA1000S* dans la prédiction.

La courbe liée à la régression de l'activité montre que le F-score est presque constant sur l'intervalle considéré. Il atteint cependant un maximum pour $\alpha = 0.82$ avec un F-score de 0.56.

Pour les métriques structurelles nous observons des valeurs bien plus faibles du F-score pour les petits α. Le F-score augmente lentement jusqu'à $\alpha \simeq 0.8$ puis augmente rapidement à son maximum pour $\alpha = 1$.

Cela montre que dans cette expérience, notre algorithme ne bénéficie pas des métriques structurelles. En effet quand $\alpha = 1$, la prédiction n'utilise plus que l'*EA1000S*.

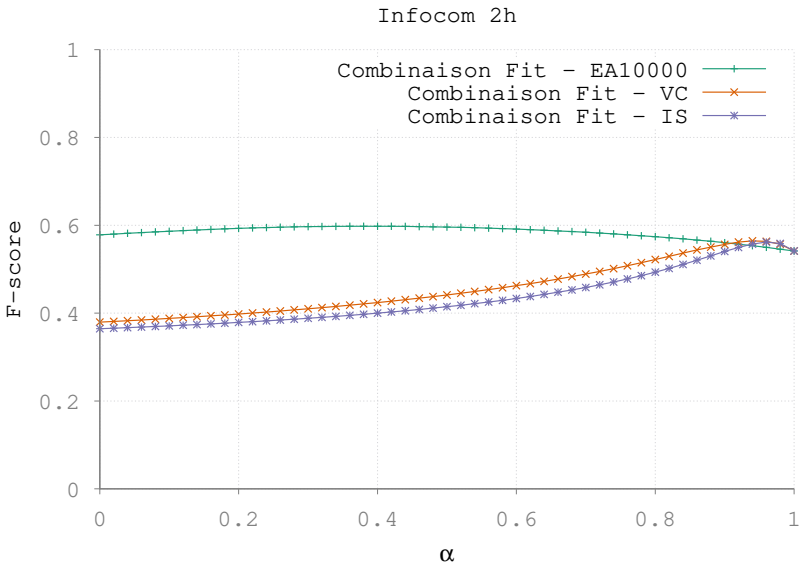


FIGURE 6.9 – F-score de la prédiction pour différents rapports de poids entre la régression de l'activité et trois autres métriques pour le jeu de données *Infocom* sur des périodes de deux heures

Nous appliquons ensuite le même protocole pour le jeu de données *Infocom* sur des périodes de deux heures. Nous combinons cette fois la régression de l'activité avec, successivement, l'extrapolation de l'activité durant les 1000 dernières secondes, le nombre de voisins communs et l'indice de Sørensen.

La combinaison entre les deux métriques temporelles, la régression de l'activité et l'*EA1000S*, se comporte globalement de la même façon que pour le jeu de données *Highschool*, avec une courbe légèrement convexe avec un maximum pour $\alpha \simeq 0.52$.

La combinaison avec les métriques structurelles, le nombre de voisins communs et l'indice de Sørensen, présente en revanche un profil différent. Les deux courbes montrent une augmentation du F-score jusqu'à atteindre un maximum pour $\alpha = 0.94$ puis décroît jusqu'à $\alpha = 1$.

Ces observations indiquent que, dans ces expériences, combiner une métrique temporelle avec une métrique structurelle peut en effet mener à une amélioration du F-score, cependant cette amélioration n'est pas ici suffisante pour justifier l'attribution d'un poids au *VC* au détriment de *EA10000*. Nous observons que dans chacune des expériences présentées notre protocole tend à largement favoriser les métriques temporelles. Nous allons maintenant nous pencher sur le type de liens prédits pour différentes combinaisons de métriques en évaluant séparément la prédiction de l'apparition de nouveaux liens et des liens récurrents.

6.3.3 Nature des liens prédits

En utilisant le même protocole de prédiction, nous divisons l'ensemble des paires de nœuds en deux catégories, pour observer quel type de liens sont prédits dans chaque combinaison de métriques.

Tout d'abord, certaines paires n'ont jamais interagi durant T_1, donc prédire une apparition de liens entre deux nœuds de ce type revient à prédire l'apparition d'un lien inédit dans le flot. Nous appelons *nouveaux liens* tout (t, uv) dans le flot d'entraînement L_1 tel que $\nexists\ x \in T_1, (x, uv) \in E_1$.

D'autre part, certaines paires ont interagi durant T_2 et prédire ce type d'interaction correspond à prédire une répétition de liens. Nous appelons *liens récurrents* tout (t, uv) dans le flot d'entraînement L_1 tel qu'il existe un lien $(x, uv) \in E_1$.

Le protocole est appliqué de la même façon que précédemment, puis la méthode d'évaluation est appliquée sur l'ensemble des paires de nœuds et sur chacun de ses sous-ensembles.

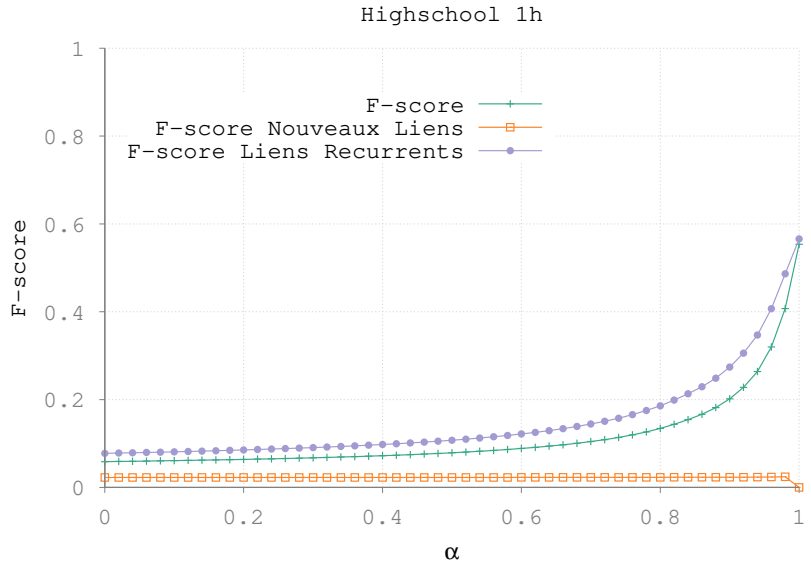


FIGURE 6.10 – F-score de la prédiction pour différentes rapports de poids entre l'*EA1000S* et *VC* pour différentes catégories de paires de nœuds pour le jeu de données *Highscool* sur des périodes d'une heure. croix vertes : tous les liens, carrés oranges : nouveaux liens, cercles violets : liens récurrents

Nous présentons en Figure 6.10 le F-score obtenu pour la combinaison de l'*EA1000S* et du nombre de voisins communs pour l'expérience d'une heure sur le jeu de données *Highschool* comme une fonction de α pour les deux catégories de paires mentionnées plus haut ainsi que pour l'ensemble complet des paires de nœuds.

Le F-score pour l'ensemble des paires de nœuds est donc le même que précédemment. Nous pouvons voir que le F-score correspondant à la catégorie des liens récurrents augmente

de façon similaire à celui associé à l'ensemble des paires nœuds, avec un maximum pour $\alpha = 1$, augmentant à mesure que plus de poids est attribué à l'*EA1000S*. Le F-score associé aux nouveaux liens quant à lui est très faible pour tout les α jusqu'à atteindre 0 pour $\alpha = 1$ puisque l'*EA1000S* n'est pas capable de prédire de nouveaux liens.

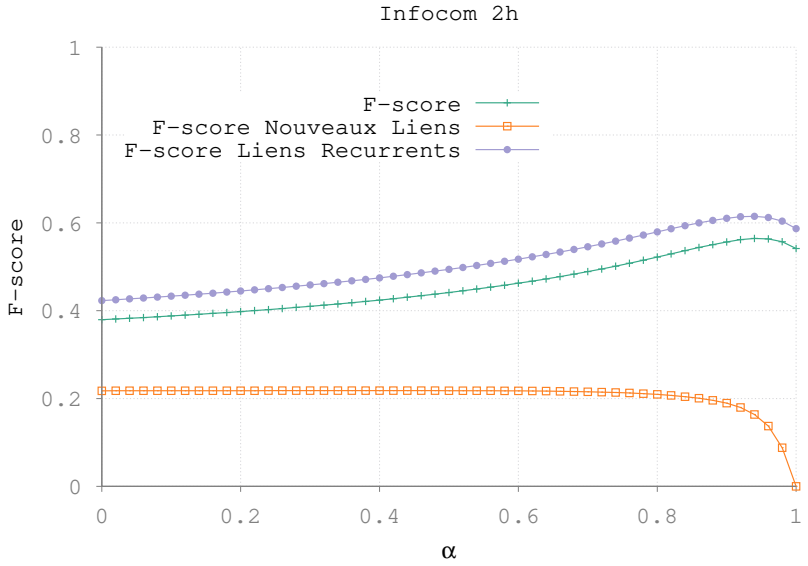


FIGURE 6.11 – F-score de la prédiction pour différentes rapports de poids entre le fit de l'activité et *VC* pour différentes catégories de paires de nœuds pour le jeu de données *Infocom* sur des périodes de deux heures croix vertes : tous les liens, carrés oranges : nouveaux liens, cercles violets : liens récurrents

Les courbes associés au jeu de données *Infocom*, présentées en Figure 6.11 présentent un comportement sensiblement différent. Nous pouvons voir que le F-score pour la prédiction des nouveaux liens commence à 0.22 pour $\alpha = 0$, puis diminue lentement jusqu'à $\alpha = 0.8$ où il décroit rapidement vers 0. Considérant les liens récurrents, le F-score commence à 0.41 puis atteint un maximum de 0.62 pour $\alpha \simeq 0.92$, puis décroit vers 0.56.

Nous observons que la qualité de la prédiction des nouveaux liens est significativement plus basse que celle des liens récurrents dans chaque expérience. La difficulté de cette tâche est principalement due au problème de déséquilibre de classe, qui est une difficulté connue dans le domaine de la prédiction de liens [LLC10]. Il est relié au fait que le rapport entre le nombre de liens apparaissant réellement et le nombre de liens potentiels, qui est simplement le nombre de paires de nœuds est plus défavorable dans le cas de la prédiction de nouveaux liens que dans la prédiction de liens récurrents. Par exemple, pour le jeu de données *Highschool*, 82 liens sont apparus entre les 6242 paires de nœuds n'ayant jamais interagi, alors que 452 liens sont apparus entre les 313 paires de nœuds ayant interagi dans le passé. Il apparait clairement que le nombre de liens prédits dans chaque catégorie joue un rôle important dans la qualité de la prédiction, les liens les plus difficiles à prédire impactant négativement la qualité de la prédiction globale.

6.4 Conclusion

Dans ce chapitre nous avons vu que notre protocole est capable de combiner différentes métriques afin d'améliorer la prédiction. Toutefois il semble favoriser les métriques temporelles sur les jeux de données utilisés. Il semble également relativement robuste aux imprécisions de la prédiction de liens globale.

Nous pouvons cependant noter que la qualité de la prédiction est fortement influencée par les choix des périodes de prédiction, notamment aux changements de comportement du système, comme un changement très important de l'activité globale, illustré par l'expérience sur les périodes de deux jours du jeu de données *Reality Mining* ou des interactions ayant lieu comme dans l'expérience sur les périodes de deux heures sur *Highschool*. De plus, nous avons noté qu'en choisissant des combinaisons de métriques spécifiques la prédiction se concentre sur les liens récurrents au détriment de la prédiction de nouveaux liens.

Cela nous amène à avancer une hypothèse concernant les poids faibles ou nuls des métriques structurelles dans les combinaisons calculées en Section 6.1.2. En effet, en ne sélectionnant quasi exclusivement que des métriques purement temporelles, notre protocole évite la difficile tâche de prédire les nouveaux liens, pour favoriser la prédiction des liens récurrents. Bien que cela améliore le F-score global, c'est également préjudiciable à la variété de liens prédits. Afin de corriger ce comportement potentiellement indésirable, nous allons explorer dans le prochain chapitre comment intégrer la notion de catégorie de liens dans notre protocole de prédiction.

Chapitre 7

Classes

Comme nous l'avons vu précédemment, certaines combinaisons de métriques spécifiques favorisent la prédiction de liens entre différents types de paires de nœuds. De fait, notre protocole utilise l'optimisation des paramètres de métriques pour sélectionner la tâche de prédiction la plus facile afin de maximiser la qualité globale de l'évaluation. Cela le conduit à privilégier la prédiction de liens récurrents, une tâche proche de la prédiction de séries temporelles et limite les bénéfices de notre approche hybride.

Afin d'introduire plus de variété dans le type de liens prédits, nous introduisons dans notre protocole différentes classes de paires de nœuds. Nous allons utiliser un ensemble de paramètres α_m distincts pour chacune de ces classes, permettant d'adapter la combinaison de métriques à chaque classe. Nous allons ici explorer le comportement de notre protocole vis-à-vis des possibilités offertes par l'introduction des classes. Nous allons dans un premier temps mettre en place un protocole de base pour la gestion de classes basé sur l'activité des paires de nœuds, puis nous verrons une méthode pour les définir, ouvrant la possibilité d'obtenir des classes définies par des critères plus avancés.

7.1 Différentes approches des classes et catégories de paires de nœuds

De nombreuses études se sont penchées sur différentes façons de définir et d'utiliser des catégories de nœuds ou de paires de nœuds. Il est possible d'intégrer les catégories de nœuds dans la prédiction pour créer des variantes des métriques classiques prenant en compte les attributs des nœuds ou la structure des communautés au sein du réseau [SH12].

Les catégories de nœuds peuvent également être utilisées pour réduire l'effet du déséquilibre de classes entre les liens apparaissant et le nombre de paires de nœuds possibles. Utiliser des catégories pour échantillonner le jeu de données et réduire le nombre de paires de nœuds possible est une méthode efficace pour améliorer la prédiction. Par exemple,

Lichtenwalter et al. [LLC10] établit des catégories de nœuds basées sur la distance entre ceux-ci. En considérant les voisinages de chaque nœud comme des problèmes distincts, ils réduisent le nombre de paires de nœuds à considérer et améliorent la prédiction.

Scholz et al. [SAS12] font la distinction entre la prédiction de nouveaux liens et de liens récurrents afin de mieux comprendre les différences entre ces deux types de prédiction. Ils observent notamment l'influence des durées de contacts entre des participants sur la prédictibilité des liens récurrents ouvrant la voie au développement de méthodes utilisant plus en détails l'aspect temporel des données.

7.2 Réduction de l'espace des paramètres

L'introduction des classes implique une forte augmentation de la dimension de l'espace des paramètres comparé aux expériences du Chapitre 6. Afin de garder des temps de calcul raisonnables, nous allons réduire l'ensemble des métriques disponibles. De nombreuses méthodes existent pour sélectionner les métriques à utiliser lors d'une tâche de prédiction [DL97, GE03]. Nous allons ici utiliser une méthode empirique, utilisant les matrices de corrélation entre métriques présentées au Chapitre 5 que nous reproduisons en Figure 7.1 ainsi que les résultats des combinaisons du chapitre précédent afin d'identifier les métriques les plus susceptibles de permettre une prédiction variée.

— Comme vu au Chapitre 5, les métriques structurelles sont fortement corrélées. Plus particulièrement le *nombre de voisins communs*, l'*indice d'Adamic-Adar* et l'*indice d'allocation de ressource* capturent des informations similaires.

— Nous voyons également que l'*indice de Sørensen* et l'*indice de Jaccard* sont très semblables. Nous allons donc choisir une métrique parmi chaque groupe, le *nombre de voisins communs* et l'*indice de Sørensen*. Cela va permettre une certaine variété dans les métriques structurelles disponibles.

— Pour plusieurs jeux de données les variations pondérées des métriques structurelles semblent proches des métriques structurelles, à l'exception du *nombre de voisins communs pondéré*. De plus seul le *nombre de voisins communs pondéré* a été utilisé durant les expériences du chapitre précédent. Nous allons donc garder ce dernier pour les expériences suivantes.

— Les métriques temporelles permettent de se concentrer sur des périodes temporelles différentes au sein des jeux de données et apportent des informations différentes. Nous allons cependant en écarter certaines en se basant principalement sur les combinaisons de métriques du chapitre précédent. Tout d'abord, l'*extrapolation de l'activité sur les 100 dernières secondes* n'est pas applicable sur la moitié des jeux de données présentés. La *distribution des temps inter-contacts* n'a été utilisée que dans que dans deux expériences sur les douze, nous allons donc également

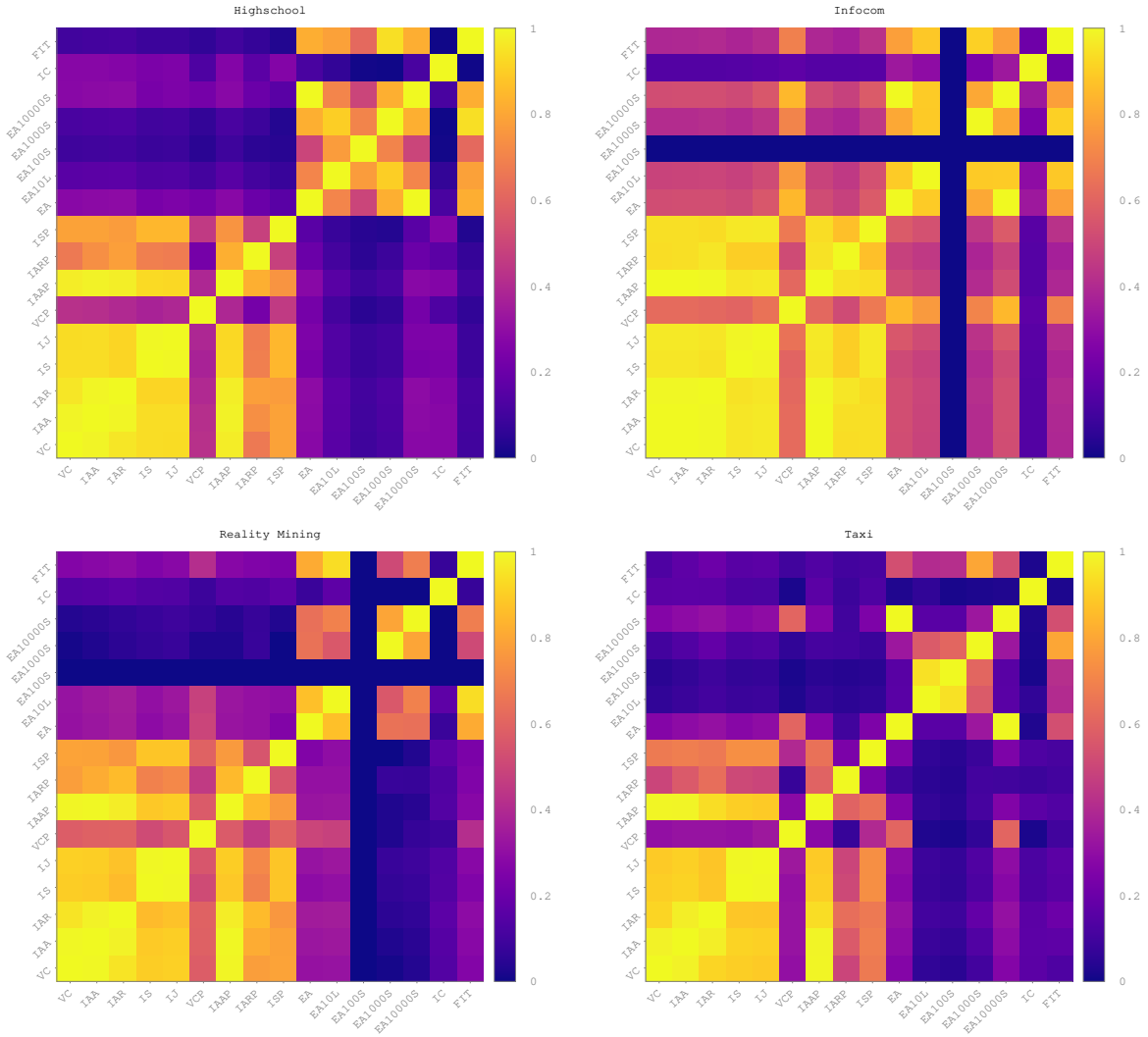


FIGURE 7.1 – Matrices de corrélation des valeurs des métriques calculées et intégrées sur les périodes de deux heures pour les jeux de données *Highschool* et *Infocom* et deux jours pour les jeux de données *Reality Mining* et *Taxi*.

écarter cette métrique. Finalement, la *régression linéaire de l'activité passée* a bien été utilisé dans plusieurs expériences. Cependant elle a également conduit, dans l'expérience sur des périodes de trois heures concernant le jeu de données *Reality Mining*, à une forte baisse de la qualité de la prédiction. Afin d'éviter cet écueil dans les expériences suivantes nous allons nous passer de cette métrique.

Dans les expériences de cette section nous utiliserons donc les métriques suivantes : le *nombre de voisins communs* (VC), l'*indice de Sørensen* (IS) et le *nombre de voisins communs pondéré* (VCP). À ces métriques nous associerons également des métriques temporelles, l'*extrapolation de l'activité* (EA), l'*activité par seconde durant les 10 derniers liens* (EA10L), l'*extrapolation de l'activité durant les 1000 dernières secondes* (EA1000S) et l'*extrapolation de l'activité durant les 10000 dernières secondes* (EA1000S).

Il y a une part d'arbitraire dans cette sélection. Une méthode systématique de sélection de métriques permettrait, à l'avenir, d'obtenir des ensembles de métriques potentiellement plus pertinents et de pouvoir sélectionner celles-ci à partir d'un plus grand nombre de métriques.

7.3 Classes par seuil d'activité

Nous allons tout d'abord considérer un système de classes simples, définies par l'activité de chaque paire de nœuds. Nous verrons ensuite comment notre protocole se comporte vis-à-vis de ce type de classes pour chacune des expériences.

7.3.1 Définition des classes

Nous choisissons ici de répartir les paires de nœuds dans différentes classes basées sur leur niveau d'activité. L'idée sous-jacente est de créer des classes reflétant, d'un côté la prédiction de liens dans les graphes et de l'autre la prédiction de séries temporelles. En effet la prédiction de liens entre des paires de nœuds n'ayant pas interagi dans le passé est une tâche liée à la prédiction de nouveaux liens dans les graphes alors que la prédiction de liens récurrents s'apparente à la prédiction de séries temporelles.

Nous définissons trois classes de paires de nœuds. La classe C1 est constituée de paires de nœuds n'ayant pas interagi durant la période d'entraînement La classe C2 rassemble les paires de nœuds avec moins d'un nombre de liens donné k. Finalement, les paires de nœuds restantes ayant eu une forte activité sont regroupées au sein de la classe C3. Plus formellement :

$$C1 = \{uv \in V \otimes V,\ |(t, uv) \in E_1| = 0\}$$

$$C2 = \{uv \in V \otimes V,\ 0 < |(t, uv) \in E_1| \leq k\}$$

$$C3 = \{uv \in V \otimes V,\ k < |(t, uv) \in E_1|\}$$

Le paramètre k sera fixé lors de la phase d'apprentissage, utilisant l'activité de chaque paire de nœuds durant L_1, de façon à obtenir un équilibre entre le nombre de paires de nœuds dans les classes C2 et C3. Bien que ce paramètre ait été choisi pour obtenir des classes C1 et C2 équilibrées, le ratio entre le nombre de paires au sein de celles-ci peut changer lors de la phase de prédiction.

7.3.2 Estimateur de qualité

Nous suivons le même protocole expérimental que dans le Chapitre 6, avec un ensemble de paramètres α_m pour chaque classe, calculé durant la phase d'apprentissage. Nous avons

noté au Chapitre 6 qu'en optimisant le F-score global, notre protocole favorise les classes à haute activité.

Nous allons redéfinir l'estimateur de qualité utilisé par notre algorithme d'apprentissage, de façon à s'assurer de la prédiction d'une large variété de liens. Ainsi, plutôt que d'utiliser le F-score global, nous définissons le score de prédiction $\overline{F}$ comme la moyenne harmonique des F-scores pour chaque classe, c'est-à-dire :

$$\frac{1}{\overline{F}} = \frac{1}{3}\left(\frac{1}{F(C_1)} + \frac{1}{F(C_2)} + \frac{1}{F(C_3)}\right)$$

Où $F(C_1)$, $F(C_2)$ et $F(C_3)$ sont les F-scores calculés sur chaque sous-ensemble de paires de nœuds définies par nos classes.

Une fois la phase d'apprentissage terminée et les paramètres optimisés, nous réassignons chaque paire de nœuds à nos différentes classes en utilisant leur activité durant le flot d'observation L_2. Enfin nous continuons le protocole, combinant les métriques pour chaque classe en utilisant leur ensemble de paramètres respectifs.

7.3.3 Résultats expérimentaux

Nous appliquons ce protocole sur chacune des expériences présentées au Chapitre 6, en utilisant les même périodes d'entraînement, validation, observation et prédiction. Nous présentons ici une expérience de chaque jeu de données mettant en lumière certains aspects du protocole. Les expériences restantes sont reportées en Annexe B.

Nous présentons les résultats pour chaque classe C1, C2 et C3, ainsi que les résultats globaux, obtenus en combinant la prédiction pour chaque classe, notés *All Class*. Nous présentons également les résultats obtenus sur les mêmes expériences par un protocole n'utilisant pas les classes, avec le même ensemble de métriques. Les résultats correspondants sont notés C0. Les F-scores C0-1, C0-2 et C0-3 représentent les scores obtenus par l'application de la méthode d'évaluation sur les sous-ensembles des paires de nœuds correspondants à chaque classe de paires. Comme précédemment, nous reportons la moyenne des différents indicateurs utilisés, le F-score, la précision et le rappel ainsi que l'activité prédite et réelle ainsi que le nombre de paires présentes dans chaque classe sur 10 réalisation de notre protocole. Dans chaque expérience, les écart-types concernant le F-score sont systématiquement inférieur à 0.03 et concernant la précision et le rappel systématiquement inférieurs à 0.01. On ne le précisera plus dans les tableaux pour ne pas alourdir la lecture.

7.3.3.1 Highschool

Nous présentons en Table 7.1 les résultats obtenus par notre protocole sur le jeu de données *Highschool* durant l'expérience utilisant des périodes d'une heure. Le paramètre

k est fixé à 3 afin d'obtenir un nombre de paires de nœuds comparable dans les classes C2 et C3.

Nous observons tout d'abord que, tandis qu'aucun lien n'a été prédit dans la classe C0-1, l'introduction des classes, ainsi que l'estimateur de qualité utilisé, conduit le protocole à prédire 126 liens dans la classe C1. Le faible F-score de 0.03 est explicable avec 82 liens réellement apparus parmi les 2935 paires de nœuds composant cette classe, c'est-à-dire environ 9% de la totalité des liens du flot réel. Dans cette expérience, la prédiction de nouveaux liens est une tâche particulièrement difficile.

La classe C2 présente un nombre de liens prédits particulièrement plus élevé que le nombre de liens apparus : 476.45 ont été prédits alors que seulement 30 sont apparus. En conséquence nous obtenons un rappel de 1, tous les liens apparus ayant été prédits, mais une précision de 0.06. Cela conduit à un F-score de 0.12, alors que le nombre de liens plus proche de l'activité réelle prédite dans C0-2 mène à un F-score de 0.18. Notons tout de même que la différence de qualité ne semble pas refléter l'ordre de grandeur de l'erreur dans la prédiction du nombre de liens dans cette classe.

La classe C3 quant à elle montre une nette amélioration du F-score, de 0.49 à 0.59, avec un nombre de liens prédits plus proche du nombre de liens apparus que pour C0-3, 519 liens prédits pour 422 apparus, contre 1098 pour la classe C0-3.

Malgré une amélioration significative de la prédiction de la classe C3 et marginale de la classe C1 nous observons une baisse notable de la qualité de la prédiction globale, de 0.45 pour C0 à 0.38 pour *All Class*. Cet effet était attendu, puisque l'algorithme n'optimise plus les différentes combinaisons afin de maximiser la qualité globale mais la moyenne harmonique des différentes classes. Nous verrons cela de façon plus nette lors de l'expérience suivante. Nous pouvons cependant noter que, dans cette expérience notre algorithme prédit nettement plus de liens dans la classe C2 que nécessaire, afin de mieux optimiser la prédiction sur les classes C1 et plus particulièrement C3.

TABLE 7.1 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Highschool* utilisant des périodes d'une heure.

Classe	F-score	Précision	Rappel	A_p	A_r	Nombre de paires
C0	0.45	0.34	0.71	1123	534	3003
C0-1	–	–	–	0	82	2935
C0-2	0.18	0.21	0.17	24.26	30	25
C0-3	0.49	0.34	0.88	1098.74	422	43
AllClass	0.38	0.28	0.59	1123	534	3003
C1	0.03	0.02	0.04	126.57 ± 17.22	82	2935
C2	0.12	0.06	1.00	476.45 ± 55.66	30	25
C3	0.59	0.54	0.66	519.97 ± 50.65	422	43

Nous présentons en Figure 7.2, les valeurs des paramètres α_m calculées par l'algorithme d'apprentissage pour chaque classe, ainsi que pour la prédiction sans classe, C0. Comme attendu, la méthode tend à utiliser différentes combinaisons afin d'effectuer la prédiction sur chaque classe.

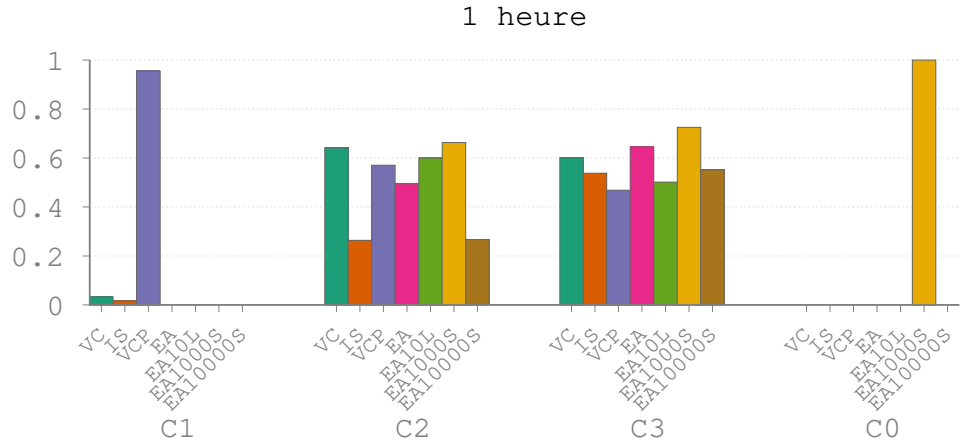


FIGURE 7.2 – Répartition des coefficients des métriques pour chaque classe calculée par l'algorithme d'apprentissage pour le jeu de données *Highschool* pour des périodes d'une heure.

Concernant la prédiction sans classe, nous observons un comportement similaire à celui présenté durant l'expérience correspondante au chapitre précédent. Notre algorithme utilise uniquement l'extrapolation de l'activité durant les 1000 dernières secondes pour prédire l'activité durant la période de prédiction. Concernant la classe C1 nous observons que le protocole utilise principalement le nombre de voisins communs pondéré. Les métriques temporelles ne permettant pas de prédire les nouveaux liens, celles-ci ne sont pas utilisées pour cette classe. Les métriques structurelles sont couramment utilisées dans la prédiction de liens dans les graphes, il est donc logique qu'elles soient privilégiées ici, la prédiction de nouveau liens étant proche de ce problème.

Les classes C2 et C3 présentent des répartitions proches, utilisant les métriques temporelles pour optimiser la prédiction mais également les métriques structurelles et pondérées. Il est cependant intéressant de constater que la prédiction de la classe C3 combine aussi les métriques temporelles avec des métriques structurelles. Cela contraste avec la prédiction sans classe, où aucun poids n'était attribué aux métriques structurelles et pondérées. Notre interprétation est que, durant le protocole sans classe, l'algorithme d'apprentissage favorisait les métriques temporelles afin d'exclure les paires de nœuds n'ayant jamais interagi, bien plus difficiles à prédire.

Avec l'introduction des classes, notre protocole est capable de concentrer la prédiction sur les liens récurrents, permettant d'attribuer des poids significatifs aux métriques structurelles pour les classes C2 et C3 sans impacter la prédiction de nouveaux liens.

7.3.3.2 Infocom

Nous présentons en Table 7.2, les résultats obtenus pour l'expérience sur les périodes de deux heures du jeu de données *Infocom*.

TABLE 7.2 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Infocom* utilisant des périodes de deux heures.

Classe	F-score	Précision	Rappel	A_p	A_r	Nombre de paires
C0	0.55	0.50	0.60	16737	13939	4278
C0-1	0.05	0.31	0.03	216.48 ± 28.87	2668	2833
C0-2	0.38	0.46	0.33	1704.02 ± 22.68	2346	739
C0-3	0.64	0.51	0.85	14816.50 ± 49.43	8925	706
AllClass	0.53	0.49	0.58	16737	13939	4278
C1	0.26	0.23	0.31	3607.44 ± 467.89	2668	2833
C2	0.41	0.38	0.46	2911.85 ± 652.59	2346	739
C3	0.65	0.61	0.70	10217.71 ± 810.15	8925	706

Nous observons tout d'abord que le nombre de liens prédits dans la classe C1 est plus de 15 fois plus élevé que dans la classe C0-1. La précision diminue mais le rappel est significativement plus élevé, permettant au protocole d'obtenir un F-score de 0.26, contre 0.05 pour la classe C0-1.

Nous notons également que l'activité prédite pour la classe C2 est presque deux fois plus importante, ce qui mène également à une faible amélioration du F-score, de 0.38 à 0.41.

Il est particulièrement intéressant de noter que l'introduction des classes améliore également la prédiction dans la classe C3, de 0.64 à 0.65. Le nombre de liens prédits dans cette classe diminue largement suite à l'augmentation du nombre de liens prédits dans les autres classes, le rapprochant de l'activité observée.

Contre-intuitivement, tandis que la qualité de la prédiction s'est améliorée dans chaque classe, le F-score obtenu sur l'union des classes diminue, de 0.55 à 0.53. Ce phénomène semble être de même nature que le paradoxe de Simpson. Cela peut conduire à une situation où la précision(ou le rappel) sur l'union des sous ensembles et la précision sur chaque sous-ensemble évoluent différemment.

Nous présentons sur la Figure 7.3, les valeurs des paramètres α_m durant ces expériences. Nous voyons que, pour la prédiction sans classe C0, notre algorithme utilise majoritairement les métriques temporelles. Notons toutefois l'utilisation du nombre de voisins communs pondéré, responsable de la prédiction de liens dans la classe C0-1. Encore une fois la classe C1 favorise les métriques structurelles et pondérées, principalement le nombre de voisins communs (standard et pondéré). Couplé à l'augmentation du nombre de liens prédits dans cette classe, cela explique la forte amélioration du F-score de la prédiction de l'activité de ces paires de nœuds. De nouveau, les classe C2 et C3 combinent des métriques structurelles et pondérées. Nous pensons que cette diversité de métriques,

combinée à une meilleure prédiction du nombre global de liens dans chaque classe est responsable de l'amélioration de la qualité de la prédiction observée.

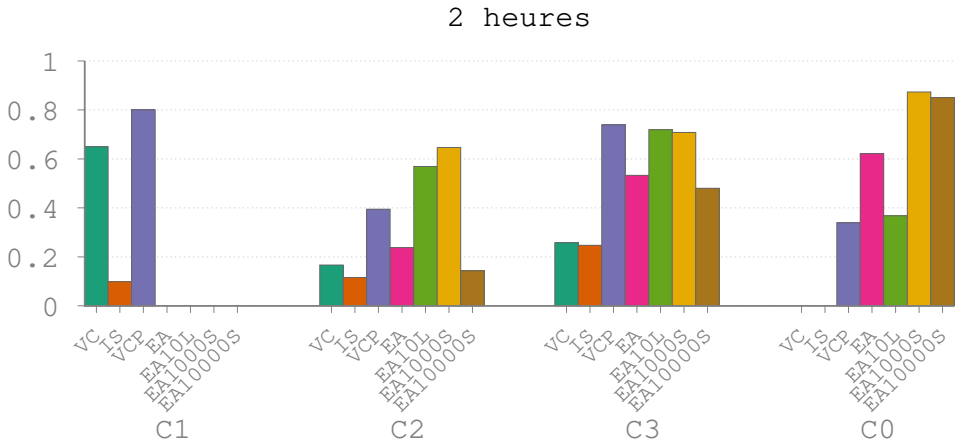


FIGURE 7.3 – Répartition des coefficients des métriques pour chaque classe calculée par l'algorithme d'apprentissage pour le jeu de données *Infocom* pour des périodes de deux heures.

7.3.3.3 Reality Mining

Nous présentons en Figure 7.3 les résultats obtenus lors de l'expérience utilisant des périodes d'un jour sur le jeu de données *Reality Mining*.

TABLE 7.3 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Reality Mining* utilisant des périodes d'une journée

Classe	F-score	Précision	Rappel	A_p	$A_p r$	Nombre de paires
C0	0.48	0.41	0.56	6105	4518	1035
C0-1	0.00	0.03	0.00	55.76 ± 46.43	1218	842
C0-2	0.14	0.13	0.15	313.73 ± 30.49	284	107
C0-3	0.57	0.43	0.82	5735.51 ± 63.76	3016	86
AllClass	0.30	0.26	0.35	6105	4518	1035
C1	0.07	0.05	0.10	2529.83 ± 437.85	1218	842
C2	0.17	0.12	0.28	668.96 ± 136.08	284	107
C3	0.47	0.48	0.46	2906.21 ± 355.81	3016	86

Nous observons une amélioration de la qualité de la prédiction dans les classes C1 et C2, passant respectivement de 0 à 0.07 et de 0.14 à 0.17. Cela semble être lié à l'augmentation du nombre de liens prédits dans chacune de ces classes, de 55.76 à 2529.83 pour C1 et

de 313.73 à 668.96 pour C2. Le très faible nombre de liens prédits dans la classe C0-1 menant en effet à l'absence quasi systématique de vrai positif, comme en témoignent la précision et le rappel. Concernant la classe C2 nous pouvons voir que l'augmentation du nombre de liens prédits conduit à un rappel bien plus élevé, presque le double que pour C0-2, tandis que la précision diminue légèrement.

Sur la classe C3, nous observons pour la première fois, sur une classe contenant de nombreuse paires de nœuds, une nette diminution du F-score lors de l'introduction des classes. Nous avons observé un phénomène similaire sur *Highschool* mais le faible nombre de paires de nœuds présentes dans la classe C2 semblait la principale raison de ce comportement particulier. Nous voyons ici que, tandis que le nombre de liens prédits dans cette classe semble se rapprocher du nombre de liens apparus, la précision augmente légèrement et le rappel passe de 0.82 à 0.46, menant à une baisse du F-score de 0.57 à 0.47. Nous observons également une baisse de la qualité de la prédiction globale, de 0.48 à 0.30.

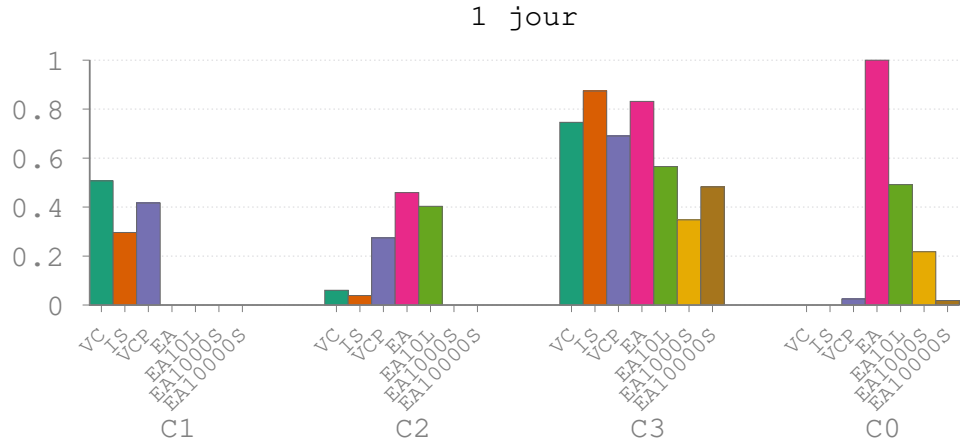


FIGURE 7.4 – Répartition des coefficients des métriques pour chaque classe calculés par l'algorithme d'apprentissage pour le jeu de données *Reality Mining* pour des périodes d'une journée.

Nous présentons en Figure 7.4 les combinaisons utilisées par notre algorithme durant cette expérience. La classe C0 utilise des métriques similaires à celles utilisées lors du Chapitre 6, privilégiant les métriques temporelles. De même que dans les autres expériences présentées plus haut nous voyons que la classe C1 se concentre sur les métriques structurelles et pondérées.

Concernant les classes C2 et C3 nous observons une nette différence entre la combinaison de métriques utilisées pour chaque classe. La classe C2 se concentre principalement sur trois métriques : le nombre de voisins communs pondéré, l'extrapolation de l'activité durant la période de prédiction et l'activité par unité de temps lors des 10 derniers liens. La classe C3 quant à elle utilise toutes les métriques disponibles, avec notamment une forte importance donnée aux métriques structurelles, le nombre de voisins communs et l'indice

de Sørensen. Cela semble indiquer que les classes C2 et C3 permettent d'isoler des paires de nœuds aux comportements sensiblement différents, modélisées par des combinaisons de métriques différentes.

7.3.3.4 Taxi

Enfin, nous présentons les résultats obtenus pour le jeu de données *Taxi* sur des périodes d'une journée en Figure 7.4.

TABLE 7.4 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Taxi* utilisant des périodes d'une journée

Classe	F-score	Précision	Rappel	A_p	A_r	Nb de paires
C0	0.18	0.19	0.17	872173	943173	42486
C0-1	0.10	0.15	0.08	314192.82 ± 48811.66	566861	36899
C0-2	0.15	0.17	0.13	80097.97 ± 7873.91	103565	2644
C0-3	0.28	0.22	0.38	477882.21 ± 55815.45	272747	2943
AllClass	0.14	0.15	0.14	872173	943173	42486
C1	0.12	0.12	0.11	499064.43 ± 45140.19	566861	36899
C2	0.16	0.15	0.17	120744.32 ± 26360.89	103565	2644
C3	0.19	0.20	0.18	252364.25 ± 48574.71	272747	2943

Nous observons dans cette expérience des résultats proches de ceux obtenus pour *Reality Mining*, avec une amélioration du F-score des classes C1 et C2 et une diminution de la qualité pour C3 et la prédiction globale.

Notons toutefois que la répartition des liens dans ce jeu de données présente une particularité par rapport aux autres jeux de données présentés. En effet, la majorité des liens apparus durant la période de prédiction sont entre des paires de nœuds de la classe C1, autrement dit, il apparait plus de nouveaux liens que de répétitions de liens. Cela explique le poids relativement élevé des métriques structurelles et l'importance du nombre de voisins communs pondéré dans la combinaison pour la prédiction sans classe C0, présentée en Figure 7.5 avec les combinaisons associées aux autres classes.

Concernant la combinaison des métriques des différentes classes nous pouvons voir une répartition semblable à celles observées précédemment, les éléments de la classe C1 étant prédits grâce aux métriques structurelles et pondérées tandis que les classes C2 et C3 combinent toutes les métriques disponibles pour représenter le comportement de ces paires de nœuds.

7.3.4 Résumé

À la lumière de ces expériences, nous pouvons voir que l'introduction des classes de paires de nœuds dans notre protocole est une option pertinente afin d'améliorer la diversité des

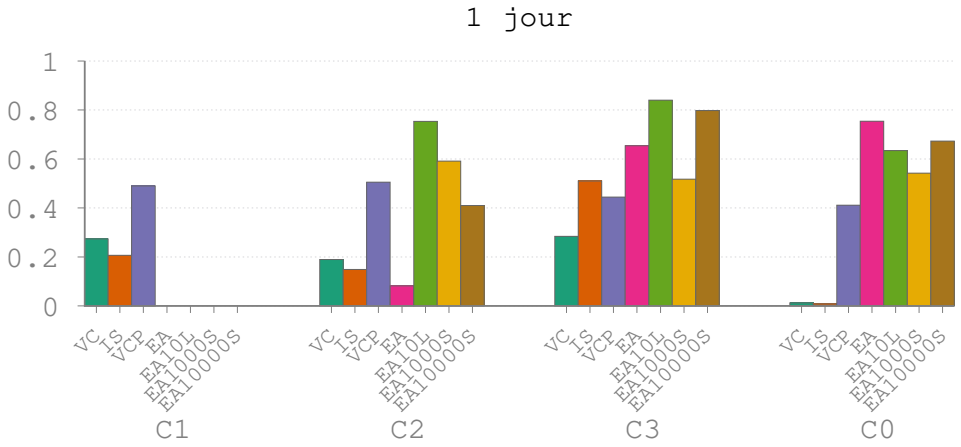


FIGURE 7.5 – Répartition des coefficients des métriques pour chaque classe calculée par l'algorithme d'apprentissage pour le jeu de données *Taxi* pour des périodes de deux heures.

liens prédits. Elle permet d'éviter à l'algorithme d'apprentissage de favoriser les tâches de prédiction les plus faciles et de prédire au sein d'un protocole unique différentes classes de nœuds aux comportements distincts.

Notons toutefois que l'introduction des classes de paires de nœuds présente également des points délicats. Comme nous l'avons vu, la volonté de diversifier le type de liens prédits entre en conflit avec l'estimation de la qualité globale de la prédiction. En effet, l'utilisation de la moyenne harmonique des F-scores pour optimiser la prédiction conduit à une baisse des F-scores calculés sur l'ensemble des paires de nœuds. De plus, comme nous l'avons vu pour le jeu de données *Infocom*, une amélioration du F-score sur les sous-ensembles de paires ne signifie pas forcément une amélioration du F-score global.

D'autre part, l'estimateur de qualité présenté en Section 7.3.2 n'est qu'un des estimateurs possible parmi d'autres. Il conduit notamment à donner une forte importance à l'amélioration des classes les plus difficiles, avec des F-score faibles. Cela mène parfois à des baisses importantes de la qualité dans la classe C3 pour de faibles améliorations de la prédiction des autres classes comme dans le cas du jeu de données *Taxi*. L'importance d'une prédiction diversifiée dépend du domaine d'application de l'algorithme, le choix de l'estimateur apparaît comme relevant de l'utilisateur, que ce soit par le biais d'une moyenne pondérée ou d'un estimateur plus spécifique à l'application considérée.

7.4 Méthode de clustering

L'introduction des classes présente des possibilités intéressantes pour regrouper des paires de nœuds aux comportements comparables. La méthode présentée précédemment se base

sur l'activité de chaque paire de nœuds durant la période d'entraînement et d'observation. Dans cette partie exploratoire, nous introduisons une méthode permettant de répartir les nœuds suivant le temps de chacune de leurs interactions. Nous utilisons une méthode de clustering hiérarchique pour définir des classes basées sur la distance entre les différentes paires de nœuds. Nous utilisons ensuite le même protocole que précédemment pour effectuer la prédiction.

Nous présentons la méthode utilisée puis étudions les résultats de prédictions pour différentes classes définies suivant ce procédé.

7.4.1 Définition des classes

Afin de répartir les liens dans les différentes classes nous allons utiliser la méthode appelée UPGMA (*Unweighted Pair Group Method with Arithmetic mean*) [Sok58]. Cette méthode permet de regrouper les éléments d'un système, en étant basée sur la distance entre ceux-ci.

L'algorithme calcule dans un premier temps la distance entre chaque paire de nœuds en utilisant la *spike time distance* évoquée au Chapitre 2 entre la suite des interactions de chaque paire durant la période d'entraînement. Les paires de nœuds sans interaction sont regroupées au sein de la classe C1. Nous définissons la classe de chacune des paires de nœuds restantes en utilisant les clusters calculés par la méthode de clustering. Ensuite, comme précédemment, l'algorithme d'apprentissage optimise alors les combinaisons de métriques utilisées pour chaque classe. Lors de la phase de prédiction, les paires sont réparties dans les différentes classes en utilisant la suite de leurs interactions durant la période d'observation afin d'effectuer la prédiction.

7.4.1.1 Distance entre deux ensembles de points

Nous allons tout d'abord définir la distance entre deux paires de nœuds basée sur les interactions entre chaque paire. Les interactions entre deux nœuds sont modélisées par une suite de points caractérisée par les temps d'apparition des liens entre les nœuds. Dans ce contexte, nous voulons définir une mesure de similarité entre deux suites de points. Plusieurs distances existent dans la littérature. Nous avons évoqué une telle mesure au Chapitre 2, introduite par Victor et Purpura [VP96], la *Purpura spike time distance*.

L'ensemble des interactions entre deux nœuds X_A et Y_A, $\{(t, X_A Y_A) \in E\}$ est notée S_A. La distance entre deux suites de points S_A et S_B est caractérisée par le nombre minimal d'étapes élémentaires nécessaires pour transformer la suite de point S_A en S_B. Deux étapes élémentaires sont possibles :

- L'ajout ou le retrait d'un lien, ayant un coût 1.
- Le déplacement d'un lien d'un temps a_k à un temps b_k. Le coût associé est $q \cdot |b_k - a_k|$.

Le paramètre q détermine le coût relatif de chaque étape. La distance entre S_A et S_B étant finalement :

$$D^{spike}(S_a, S_b) = min(K(S_A, S_1) + K(S_1, S_2) + \cdots + K(S_{n-1}, S_n) + K(S_n, S_B))$$

Où les S_k sont les suites de points intermédiaires et $K(S_k, S_{k+1})$ le coût de l'étape élémentaire permettant de passer de la suite S_k à la suite S_{k+1}. La *Purpura spike time distance* permet d'obtenir une mesure entre deux ensembles de points possédant les propriétés mathématiques d'une distance. Le paramètre q contrôle la précision temporelle de la distance, c'est-à-dire l'intervalle entre deux liens tel qu'il est moins coûteux de déplacer le lien d'un temps à un autre que de supprimer ce lien et l'ajouter au temps voulu.

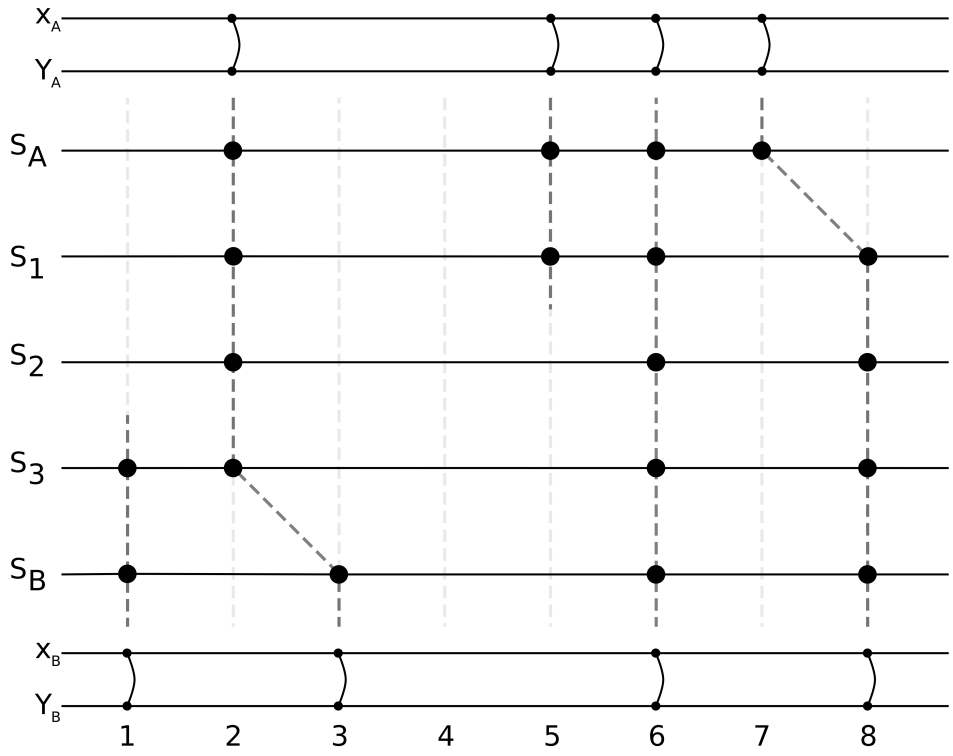


FIGURE 7.6 – Illustration de la *Purpura spike time distance* entre deux paires de nœuds X_AY_A et X_BY_B pour $2 \geq q \geq 1$.

Nous illustrons une suite minimale d'étapes élémentaires permettant de transformer l'ensemble des interactions de deux paires de nœuds X_AY_A et X_BY_B en Figure 7.6. Les interactions de chaque paire de nœuds sont représentées par les ensembles de points S_A et S_B. S_1, S_2 et S_3 représentent les suites de points intermédiaire permettant de passer de S_A à S_B pour $2 \geq q \geq 1$. La distance obtenue étant alors $2 \cdot q + 2$: deux ajouts ou retraits de points ainsi que le déplacement de deux points d'une unité de temps, pondéré par q.

Lorsque $q = 0$, le déplacement des liens n'ayant aucun coût, cette distance est équivalente à la différence du nombre de liens apparus entre les deux paires de nœuds. Cela conduirait

à une distance de 0 dans l'exemple présenté. À l'opposé, lorsque $q = \infty$ le déplacement d'un lien d'un temps à l'autre n'étant pas possible, la *spike time distance* mesure le nombre de liens non simultanés entre X_AY_A et X_BY_B, une distance de 6 dans notre exemple.

Dans le contexte de notre protocole nous choisirons un paramètre q tel que le coût d'enlever et de rajouter un lien soit équivalent au coût de déplacer le lien d'un tiers de la longueur des périodes considérées.

7.4.1.2 UPGMA

Ayant défini une distance, on peut appliquer une méthode de clustering agrégatif, l'UPGMA, permettant de construire un dendrogramme basé sur la structure de la matrice des distances entre les éléments d'un système, dans notre cas entre les paires de nœuds.

Les éléments sont initialement répartis dans des clusters à un élément et la matrice de distance est calculée en utilisant la *spike pike distance*. À chaque étape, l'algorithme identifie les deux clusters A et B les plus proches. Ceux-ci sont alors fusionnés et la distance entre l'union de ces deux clusters et chaque autre cluster X est calculée comme la moyenne des distances de chaque cluster à X, pondérée par le cardinal de chaque cluster :

$$d_{(A\cup B),X} = \frac{|A| \cdot d_{A,X} + |B| \cdot d_{B,X}}{|A| + |B|}$$

Le dendrogramme est construit itérativement jusqu'à ce que tous les éléments soient réunis dans un même cluster. Nous obtenons alors un arbre dont chaque noeud correspond à un cluster, autrement dit, un ensemble de paires de nœuds de notre jeu de données.

Les classes utilisées dans notre protocole sont sélectionnée en coupant le dendrogramme, garantissant que toutes les paires de nœuds sont dans un et un seul cluster. Nous explorerons différentes coupes du dendrogramme en Section 7.4.2.

7.4.1.3 Répartition des paires de nœuds

Les classes sont définies par l'algorithme de clustering, basé sur leurs interactions durant la période d'entraînement. Lors de la phase de prédiction les paires de nœuds observées durant la période d'observation doivent être réparties parmi les différentes classes.

Pour chaque paire de nœuds uv, nous calculons la distance entre uv et chaque paire de nœuds apparu durant la période d'entraînement Cela nous permet alors de calculer la distance entre uv et chacun des clusters obtenu lors de la coupe de dendrogramme. La paire est alors ajoutée à la classe correspondant au cluster le plus proche.

7.4.2 Résultats expérimentaux

Nous allons ici étudier les résultats de notre protocole de prédiction utilisant les classes calculées par notre algorithme de clustering suivant deux méthodes de coupe du dendrogramme, définissant des classes différentes. Dans cette démarche exploratoire toutes les expériences seront effectuées sur le jeu de données *Infocom* utilisant des périodes de deux heures. Nous définirons entre 3 et 6 classes grâce à notre algorithme de clustering, définies suivant deux coupes différentes du dendrogramme.

Nous avons vu précédemment que l'analyse du F-score dans les protocoles utilisant différentes classes de nœuds n'est pas évidente, notamment lorsque qu'une amélioration de la prédiction sur chaque classe ne se reflète pas sur la qualité de la prédiction globale. Notre objectif ici est de déterminer si les différents clusters calculés permettent de définir des classes aux comportements distincts, notamment en observant la combinaison de métriques calculée par notre algorithme d'apprentissage. Nous nous intéresserons également au nombre de paires de nœuds dans chaque classe.

7.4.2.1 Diviser les clusters dans l'ordre inverse de leurs agrégations

Une coupe possible du dendrogramme est de partir du cluster final contenant l'ensemble des paires de nœuds et de successivement diviser les clusters dans l'ordre inverse de l'algorithme de clustering agrégatif. Cela correspond à produire les clusters les plus éloignés au sens de la distance choisie et donc *a priori* des classes aux comportements distincts. Nous allons successivement effectuer la prédiction avec 3, 4, 5 et 6 classes, correspondant à diviser successivement 1, 2, 3 et 4 clusters.

Nous présentons en Figure 7.7 la succession des divisions de cluster. Le nombre de paires indiqué ici correspond aux paires présentes dans chaque cluster lors de la phase d'apprentissage. Chaque niveau représente les clusters utilisés dans chaque expérience, ordonnés suivant le nombre de paires présentes ainsi que le nom de la classe associée.

Nous observons que notre protocole tend à diviser les clusters contenant le moins de paires de nœuds. Nous voyons ainsi que le cluster contenant 1274 paires de nœuds est identique dans chacune des expériences. Cela conduit, pour l'expérience utilisant 6 classes, à deux clusters significativement plus volumineux que les trois autres, le plus petit ne contenant qu'une seule paire.

Nous présentons en Figure 7.8 la moyenne du nombre de paires de nœuds et du F-score sur 10 réalisations du protocole pour chaque expérience faite grâce aux clusters présentés ci-dessus. Dans chaque cas l'écart-type du F-score est inférieur à 0.06 excepté pour la classe $C_{III}5$ de l'expérience à 5 classes où il est de 0.10. Les résultats détaillés sont reportés en Annexe B. Nous pouvons tout d'abord observer que le nombre de classes utilisées n'a qu'une faible influence sur la qualité de la prédiction globale, variant de 0.51 à 0.54. L'influence sur la classe C1 est également faible, variant de 0.25 à 0.26. Cet effet est *a priori* dû à l'estimateur de qualité utilisé dans notre algorithme de prédiction : la

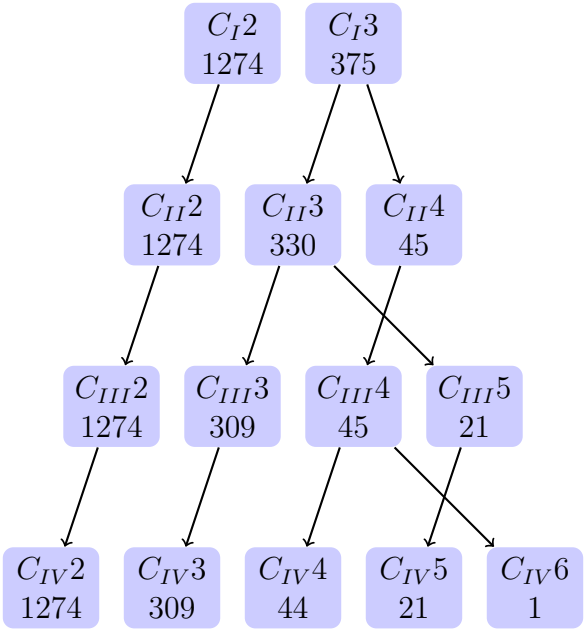


FIGURE 7.7 – Succession des divisions de clusters pour une coupe du dendrogramme dans l'ordre inverse de leur agrégation pour le jeu de données *Infocom* sur la période d'observation de 2 heures, avec le nombre de paires de nœuds dans chaque cluster.

moyenne harmonique favorise l'amélioration des F-scores les plus bas au détriment des autres classes. Notre protocole tend donc à affecter un nombre optimal de liens à cette classe afin d'en améliorer le F-score.

Concernant le nombre de paires de nœuds présentes dans chaque classe, nous observons bien le même phénomène que pour les clusters en Figure 7.7, avec des classes C2 et C3 contenant significativement plus de paires de nœuds que les autres, de l'ordre de vingt fois plus de nœuds pour la classe C2 et de six fois plus de nœuds pour la classe C3. Toutefois, l'effet semble moins prononcé que précédemment, avec entre 6 et 31 paires de nœuds supplémentaires dans chacune des plus petites classes.

Il est difficile de tirer des conclusions de l'évolution du F-score au fil les divisions. Notons toutefois que certaines divisions semblent permettre l'amélioration du F-score d'une des sous-classes, comme la division de la classe $C_{II}3$ en $C_{III}3$ et $C_{III}4$ lors de l'expérience utilisant 4 classes, passant de 0.68 pour le cluster d'origine à 0.69 et 0.66 pour les sous-classes ou encore pour la division de la classe $C_{III}4$ de l'expérience à 5 classes en $C_{IV}4$ et $C_{IV}6$, permettant un F-score supérieur ou égal au F-score d'origine dans chacune des sous-classes.

Cependant nous observons également que la division de $C_{II}3$ entre les expériences utilisant quatre et cinq classes s'accompagne d'une baisse du F-score dans chacune des classes ainsi que pour l'évaluation générale de la prédiction qui passe de 0.54 à 0.51, sa valeur la plus faible.

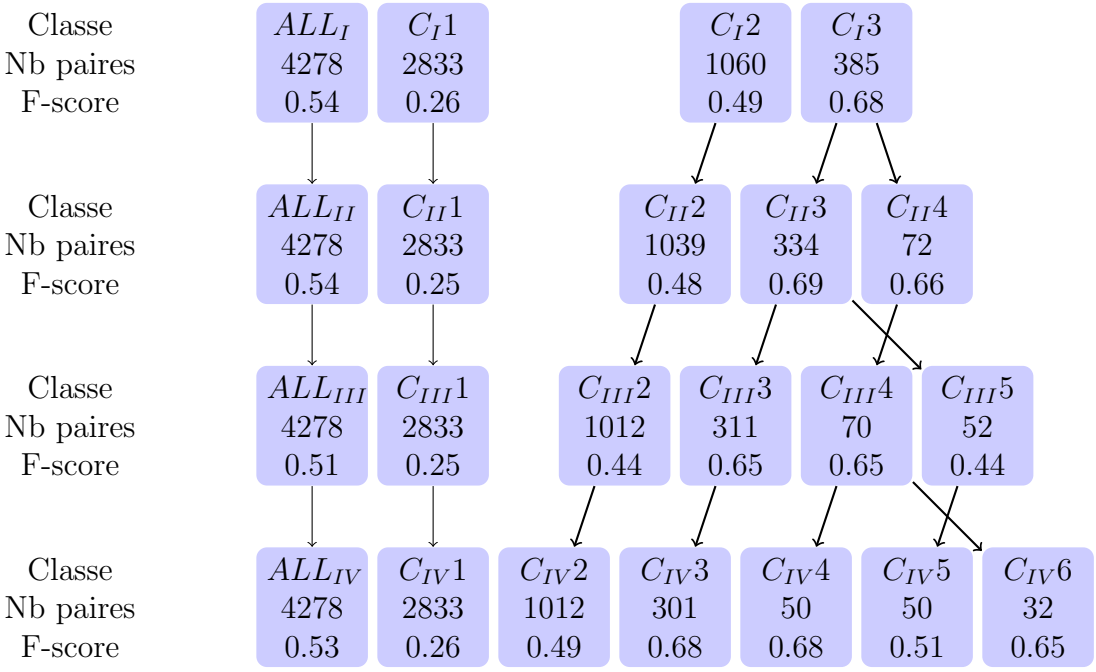


FIGURE 7.8 – Nombre de paires et F-score pour chaque classe durant la phase de prédiction pour des expériences utilisant 3, 4, 5 et 6 classes, ainsi que les relations entre les clusters associés pour le jeu de données *Infocom* sur des périodes de 2 heures.

Nous présentons en Figure 7.9 les combinaisons de métriques utilisées pour chacune des expériences présentées ci-dessus. Notre objectif ici est de déterminer si les différentes divisions de classes ont permis d'isoler des groupes de paires de nœuds aux comportements distincts. Nous faisons l'hypothèse que le choix des métriques parmi l'éventail proposé permettra d'identifier ce type de division.

La composition des classes C1 n'étant pas directement affectée par les différentes divisions ne change pas significativement durant ces expériences et utilise toujours exclusivement les métriques structurelles.

Nous pouvons tout d'abord observer que dans l'expérience utilisant 3 classes, les classes C_I2 et C_I3 semblent utiliser des combinaisons de métriques différentes. La classe C_I3 utilise l'intégralité des métriques proposées tandis que la classe C_I2 n'utilise que peu les métriques structurelles, indiquant des classes aux caractéristiques spécifiques.

La première division concerne la classe C_I3, se divisant en $C_{II}3$ et $C_{II}4$. Nous pouvons voir que cette division ne semble pas modifier significativement la répartition des métriques pour chaque classe, à l'exception d'une baisse de l'utilisation du nombre de voisins communs pondéré.

La seconde division scinde la classe $C_{II}3$ en $C_{III}3$ et $C_{III}5$. La nouvelle classe $C_{III}3$ montre encore une répartition similaire, toutes les métriques proposées étant utilisées. Toutefois la classe $C_{III}5$ présente une répartition très différente, la combinaison se concentrant très

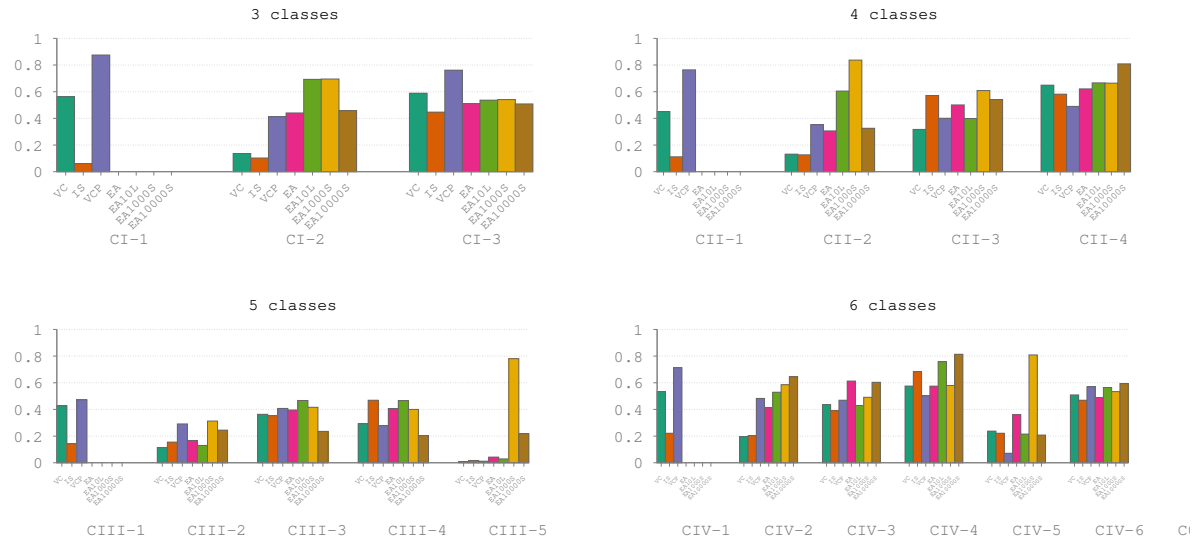


FIGURE 7.9 – Répartitions des coefficients des métriques pour chaque expérience calculées par l'algorithme d'apprentissage pour le jeu de données *Infocom* pour des périodes de deux heures.

largement sur l'extrapolation de l'activité durant les 1000 dernières secondes. Les autres métriques ne sont que très peu utilisées, à l'exception d'*EA10000S*.

La dernière division concerne la classe $C_{III}4$, créant les classes $C_{IV}4$ et $C_{IV}6$. Comme pour la première division, nous n'observons pas un changement significatif dans la composition des métriques.

Notons tout d'abord que l'expérience utilisant trois classes présente des profils de combinaison différents entre les classes C_I2 et C_I3 ainsi que des F-score similaires à ceux obtenus lors de l'utilisation des classes par seuil d'activité. Cela indique que l'UPGMA permet bien d'identifier des classes ayant des comportements distincts. Notre méthode de coupe du dendrogramme présente également des comportements intéressants, permettant d'obtenir des améliorations du F-score ou des changements significatifs dans la combinaison de métriques pour certaines classes. Nous avons observé que cette méthode pouvait conduire à l'apparition de cluster de très petites tailles, notamment dans le cas de l'expérience à six classes, avec un cluster C6 ne comprenant qu'une seule paire de nœuds.

7.4.2.2 Diviser les clusters suivant leurs tailles

Dans cette série d'expériences nous allons étudier une autre méthode de coupe du dendrogramme, moins susceptible de créer des clusters très déséquilibrés en taille. Nous répétons à l'identique le protocole précédent, en choisissant différemment les clusters à diviser pour chaque expérience. Plutôt que suivre l'ordre inverse de l'agrégation des clusters, nous allons entre chaque expérience diviser le cluster contenant le plus grand nombre de paires

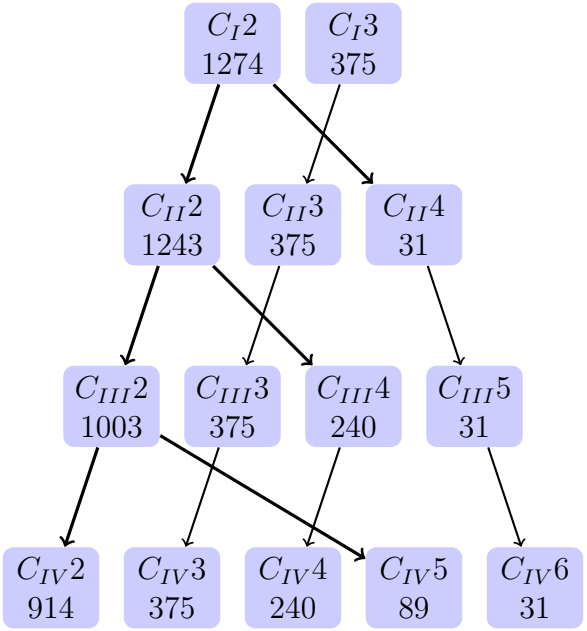


FIGURE 7.10 – Succession des divisions de clusters pour une coupe du dendrogramme par ordre de nombre de paires de nœuds pour le jeu de données *Infocom* sur des périodes d'observations de 2 heures, avec le nombre de paires de nœuds dans chaque cluster.

de nœuds. Bien que cette méthode ne garantisse pas l'absence de très petits clusters nous espérons que leur frequence d'apparition soit moindre.

Nous présentons en Figure 7.10, la succession des divisions de clusters. Nous observons que cette méthode de coupe conduit à la division successive du cluster de départ C2 en différents clusters. Cela mène à la diminution du nombre de liens dans le plus gros cluster C2 de 1274 pour la première expérience, à 914 pour l'expérience à 6 classes. Notons tout d'abord que, contre intuitivement, la première division conduit à la création du plus petit cluster de cette série d'expériences avec 31 paires de nœuds. Les divisions suivantes permettent cependant la création de clusters plus imposants avec 240 nœuds lors de la deuxième division et 89 lors de la troisième. Le cluster C3 n'est pas modifié au cours de ces divisions.

Nous présentons en Figure 7.11, le F-score obtenu dans cette série d'expériences ainsi que le nombre de paires de nœuds dans chaque classe. Les résultats détaillés sont présentés en Annexes B.

La classe C1 et la prédiction globale se comporte de la même façon que pour la méthode de coupe précédente, invariante à quelques fluctuations près. Les classes $C_{I-IV}3$, associées au même cluster durant toutes les expériences, montrent une diminution du nombre de paires de nœuds la composant, de 385 à 307 pour les deux dernières expériences. Le F-score concernant ces classes reste stable avec un F-score de 0.68.

Concernant l'évolution des classes $C_{I-IV}2$, le nombre de paires de nœuds dans la classe

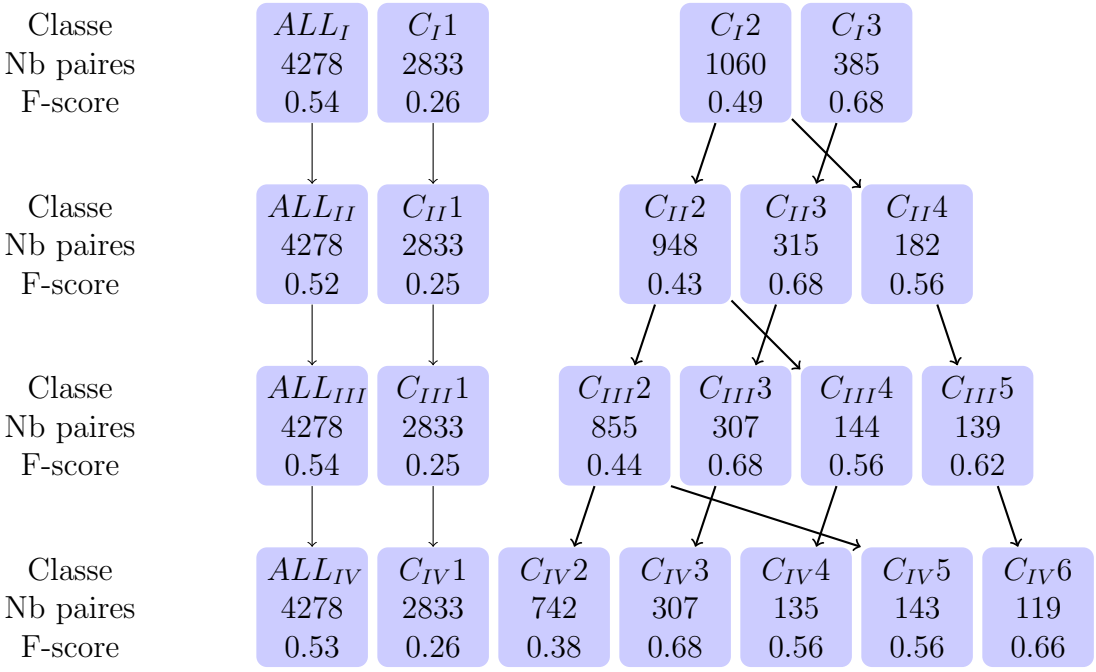


FIGURE 7.11 – Nombre de paires et F-score pour chaque classe durant la phase de prédiction pour des expériences utilisant 3, 4, 5 et 6 classes, ainsi que les relations entre les clusters associés pour le jeu de données *Infocom* sur des périodes de 2 heures.

diminue au cours des expériences de 1060 à 742. Cela s'accompagne d'une diminution du F-score de la sous-classe la plus peuplée à chaque division successive, de 0.49 à 0.38. Nous pouvons voir que chaque division s'accompagne d'une amélioration significative du F-score pour la plus petite sous-classe créée de 0.49 à 0.56 lors de la première division, de 0.43 à 0.56 pour la deuxième et de 0.44 à 0.56 lors de la dernière division. La classe issue de la première division montre également une amélioration du F-score à chaque sous-division de sa classe d'origine, jusqu'à 0.66 lors de la dernière expérience.

Nous observons également que cette méthode de coupe du dendrogramme crée des classes moins déséquilibrées qu'en Section 7.4.2.1, 119 pour la plus petite classe contre 32 dans la série d'expériences précédentes.

Les différentes combinaisons de métriques sont présentées en Figure 7.12. Suivant la même démarche que précédemment nous allons étudier les changements dans les combinaisons de métriques lors de chaque division de classe. La première division de la classe C_I2 conduit à la création des classes $C_{II}2$ et $C_{II}4$. Nous pouvons voir que la classe $C_{II}2$ lors de l'expérience à 4 classes tend à utiliser des métriques similaires à sa classe d'origine, majoritairement l'extrapolation de l'activité par seconde durant les dix derniers liens et l'extrapolation de l'activité durant les 1000 dernières secondes. Notons cependant que l'utilisation du nombre de voisins communs pondéré augmente nettement. La classe $C_{II}4$ nouvellement créée montre cependant une répartition différente avec une utilisation de toutes les métriques disponibles, notamment des métriques structurelles, très faiblement

utilisées dans la classe d'origine.

La seconde division, pour l'expérience de 5 classes, divisant la classe $C_{II}2$ en $C_{III}2$ et $C_{III}4$ ne semble pas engendrer une modification significative de la combinaison de chacune de ces classes par rapport à la classe d'origine.

Finalement la troisième division engendre les classes $C_{IV}2$ et $C_{IV}5$. Nous pouvons encore voir que la classe $C_{IV}2$ reste quantitativement similaire. La classe $C_{IV}4$ quant à elle voit à la fois une augmentation de l'utilisation des métriques structurelles ainsi qu'une forte augmentation de l'importance de l'extrapolation de l'activité sur la période de prédiction.

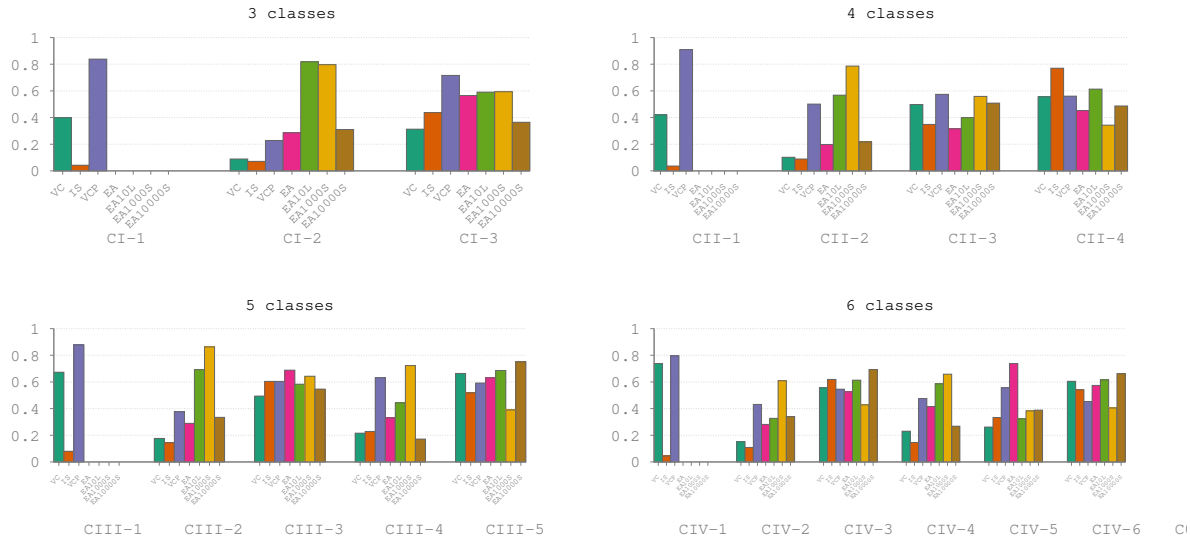


FIGURE 7.12 – Répartition des coefficients des métriques pour 6 classes calculés par l'algorithme d'apprentissage pour le jeu de données *Infocom* pour des périodes de deux heures.

Ces résultats montrent que ce type de coupe du dendrogramme permet de limiter l'apparition de classes regroupant un faible nombre de paires de nœuds tout en permettant d'obtenir des classes présentant des combinaisons de métriques variées. L'évolution de la qualité de la prédiction est également prometteuse créant des classes montrant des F-scores relativement élevés.

7.5 Conclusion

Dans ce chapitre nous avons introduit des classes de paires de nœuds dotées de leur propre ensemble de paramètres de combinaison afin de permettre à notre protocole de mieux modéliser le comportement des différents types de paires de nœuds dans les données. Afin de favoriser une prédiction répartie parmi les différentes classes nous avons défini

un nouvel estimateur de qualité dans notre algorithme d'apprentissage, forçant notre protocole à réaliser des prédictions dans chaque classe.

Nous avons tout d'abord défini des classes simples basées sur l'activité de chaque paire de nœuds afin d'adapter la combinaison de métriques à leur comportement. Les expériences effectuées montrent que l'introduction de classes est capable d'améliorer la prédiction dans les différentes classes tout en favorisant la variété de liens prédits. Il apparait cependant qu'il est difficile d'évaluer la qualité globale de la prédiction, le F-score global ne semblant pas adapté pour représenter l'amélioration de la prédiction lors de la prédiction de classes multiples. Comme nous l'avons vu, dans certaines situations il ne permet pas de traduire une amélioration de la prédiction sur chaque classe par une amélioration globale. L'utilisation d'un autre estimateur de qualité est donc nécessaire pour évaluer la qualité de la prédiction, notamment durant la phase d'apprentissage. Nous avons introduit un estimateur, la moyenne harmonique des F-scores de chaque classe, qui permet de prendre en compte ces améliorations. Cependant nous avons observé que celui-ci favorisait particulièrement l'amélioration des classes les plus difficiles à prédire. D'autres estimateurs de qualité sont possibles, permettant d'influencer le comportement du protocole de prédiction. En fonction du contexte d'application de l'algorithme une réflexion approfondie sur le choix de l'estimateur serait nécessaire.

Dans la seconde partie du chapitre nous avons exploré une méthode plus avancée pour définir des classes en regroupant les paires de nœuds suivant leurs temps d'interaction grâce à un algorithme de clustering et une mesure de distance dans les suites de points. Deux manières pour définir les classes à partir du dendrogramme ainsi créé ont été étudiées. Cette méthode semble prometteuse et permet dans certains cas la création de classe utilisant des combinaisons de métriques distinctes. Nous pensons cependant qu'une étude plus approfondie de cette méthode est nécessaire. Il parait crucial par exemple de mieux comprendre les caractéristiques des classes ainsi créées, l'utilisation de la combinaison de métriques ne permettant pas de réellement identifier quels comportements ont été isolés par notre méthode. Comprendre l'évolution de ces caractéristiques au fil des divisions des clusters permettrait par exemple d'améliorer la coupe du dendrogramme.

Les méthodes de clustering sont également particulièrement dépendantes de la mesure de distance utilisée. Il apparait nécessaire d'étudier comment la *time spike distance* se comporte dans notre contexte ainsi que d'analyser l'influence du paramètre q sur la création des classes. De nombreux algorithmes de clustering et de mesures de distance existants dans la littérature, une exploration plus systématique de ceux-ci pourra permettre la création de classes plus performantes ou plus adaptés à des jeux de données spécifiques.

Les classes présentées ici se sont concentrées sur l'activité des paires de nœuds. Il serait également possible d'utiliser des algorithmes de détection de communautés pour définir des classes basées sur la structure du réseau, par exemple en utilisant des combinaisons de métriques différentes pour les liens intra et inter communautés. De plus, dans de nombreuses applications des informations additionnelles sont disponibles sur les nœuds ou les liens, par exemple l'age ou le sexe des individus dans le cas de réseaux sociaux. Ce type d'informations est particulièrement adapté pour la définition de classes permettant

d'identifier des comportements particuliers.

Conclusion

Dans ce travail nous avons proposé un protocole de prédiction de l'activité adapté au formalisme des flots de liens, permettant de prendre avantage de l'information contenue dans cette représentation des données. Il s'appuie sur une méthode flexible de combinaison des informations provenant de métriques capturant les caractéristiques du flot et sur l'extrapolation du nombre total de liens dans le flot. Nous avons également proposé un protocole d'évaluation adapté à notre problème. Nos expériences montrent que notre protocole est capable de trouver des combinaisons de métriques temporelles et structurelles adaptées au comportement des paires de nœuds menant à des améliorations de la prédiction par rapport à l'utilisation de métriques uniques. Nous avons également étudié comment ce protocole tend à exclure des types de paires de nœuds spécifiques, menant à moins de variété dans les liens prédits. l'introduction de classes de paires de nœuds montre qu'il est possible de limiter ce problème, menant à un meilleur équilibre entre les différents types de liens prédits. En permettant d'adapter la combinaison de métriques à chaque classe nous parvenons également à améliorer la prédiction dans celles-ci. Nous avons également exploré différentes manières de définir ces classes afin de regrouper au mieux les paires de nœuds au comportement semblable. Notre protocole est conçu d'une façon modulaire, chaque partie étant indépendante des autres et pouvant être remplacée ou améliorée suivant l'application considérée.

Les expériences menées ont permis de mettre en lumière différents points méritant un approfondissement. Les métriques présentées dans ce travail sont des mesures classiques utilisées dans la prédiction de liens dans les graphes ou des métriques permettant de capturer l'information temporelle dans le flot. Notre protocole pouvant accepter des métriques variées, nous voulons définir des métriques plus raffinées permettant de capturer des dynamiques plus subtiles du flot, comme des techniques de détection de motifs afin d'identifier des structures d'interactions typiques. Par exemple, nous pourrions considérer que si trois nœuds u, v et w interagissent occasionnellement les uns les autres lors de courtes périodes de haute activité, l'apparition de liens entre les paires uv et uw suggère l'apparition de liens entre v et w peu après. Ce type de mesure permettrait d'utiliser des comportements particuliers des paires de nœuds pour la prédiction. Pour mettre en place ce type d'outils il faudra d'une part identifier les motifs récurrents significatifs apparaissant dans le flot ainsi que déterminer les échelles de temps des motifs qui nous intéressent pour chaque cas de prédiction. Par exemple, la détection de motifs s'étalant sur quelques secondes ne doit pas trop influencer la prédiction sur de longues durées, par

exemple plusieurs jours, tandis qu'à l'inverse un motif s'étalant sur une longue période ne semble pas adapté pour prédire des périodes plus courtes. Pour cela les fonctions de prédiction, pouvant spécifier les variations de la vraisemblabilité d'apparition de liens au cours du temps, semblent adaptées. Par exemple, le motif évoqué plus haut ne devrait alors influencer la fonction de prédiction que sur les premiers instants de la période de prédiction, durant une période similaire à la longueur des motifs détectés.

Il serait également intéressant d'introduire des métriques utilisant des outils de prédiction de séries temporelles. Par exemple, la modélisation de variation de métriques structurelles au cours du temps telle que présentée par da Silva Soares et Prudêncio [dSSP12] pourrait être introduite dans notre protocole. De même, modéliser l'évolution de l'activité des paires de nœuds au cours du temps de façon similaire à Huang et Lin [HL09] permettrait de mieux prédire les répétitions de liens.

Lors de l'extrapolation du nombre de liens prédits global nous avons également fait l'hypothèse que l'activité globale du flot reste constante d'une période à l'autre. Nous avons vu que la prédiction du nombre de liens global joue un rôle dans la qualité de la prédiction. Cette hypothèse n'est pas forcément satisfaite et dépend fortement des données considérées. Cependant le nombre de liens apparaissant à chaque pas de temps peut être considéré comme une série temporelle. Les modèles développés dans le contexte des séries temporelles permettraient alors d'améliorer l'évaluation du nombre de liens prédits. Par exemple, le modèle ARIMA permet de modéliser une série temporelle afin d'en prédire l'évolution. Cela permettrait, dans le cas de jeux de données dont l'activité globale évolue significativement au cours du temps, de prédire un nombre total de liens plus précis. C'est le cas notamment dans les jeux de données dont l'activité est fortement liée aux cycles jours/nuits ou aux jours de la semaine.

Pour finir nous voulons également étudier plus en détails comment notre protocole se comporte lors de l'utilisation des classes. Tout d'abord d'autres définitions des classes sont possibles, par exemple basées sur la structure du réseau ou des informations extérieures provenant du contexte du réseau étudié. Cela peut passer par l'utilisation de la distance entre les paires de nœuds [LLC10] afin de prédire différemment les liens reliant des nœuds proches et éloignés ou encore combiné à des algorithmes de détection de communautés. Il sera alors important d'étudier plus en détails les caractéristiques de chaque classe afin d'identifier les groupes de nœuds susceptibles d'être prédits selon les mêmes critères. Différentes caractéristiques peuvent être intéressantes à étudier, comme la régularité des interactions ou l'évolution de la fréquence d'interactions au cours du temps. Cela pousse à des analyses plus spécifiques des jeux de données pour isoler ces caractéristiques.

Annexe A

Distributions cumulatives des métriques

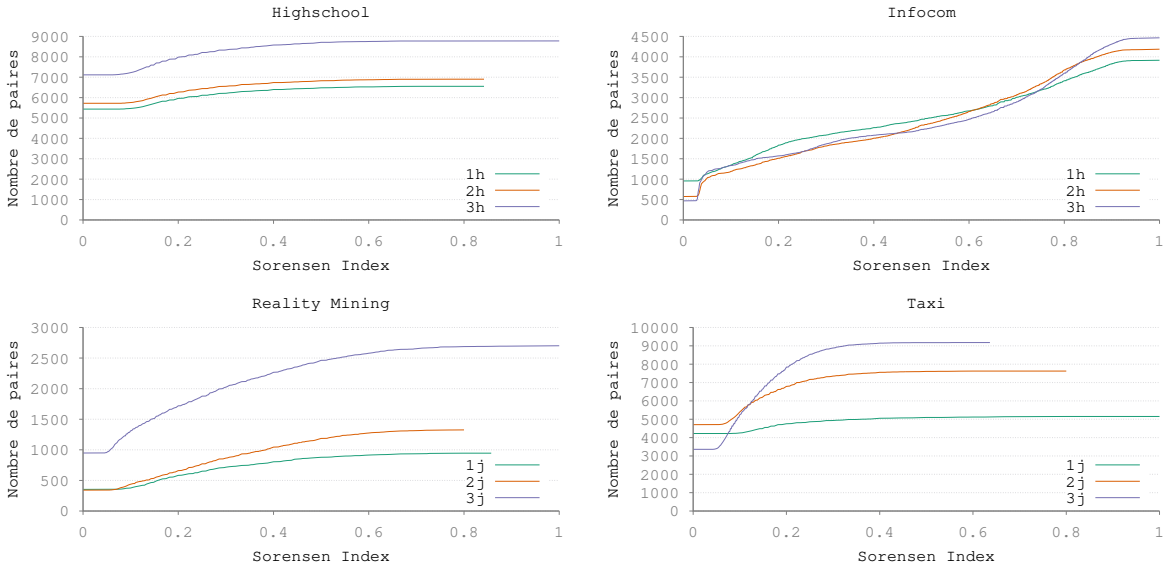


FIGURE A.1 – Fonctions de distribution cumulative de l'indice de Sørensen pour chaque période d'entraînement de chaque jeu de données.

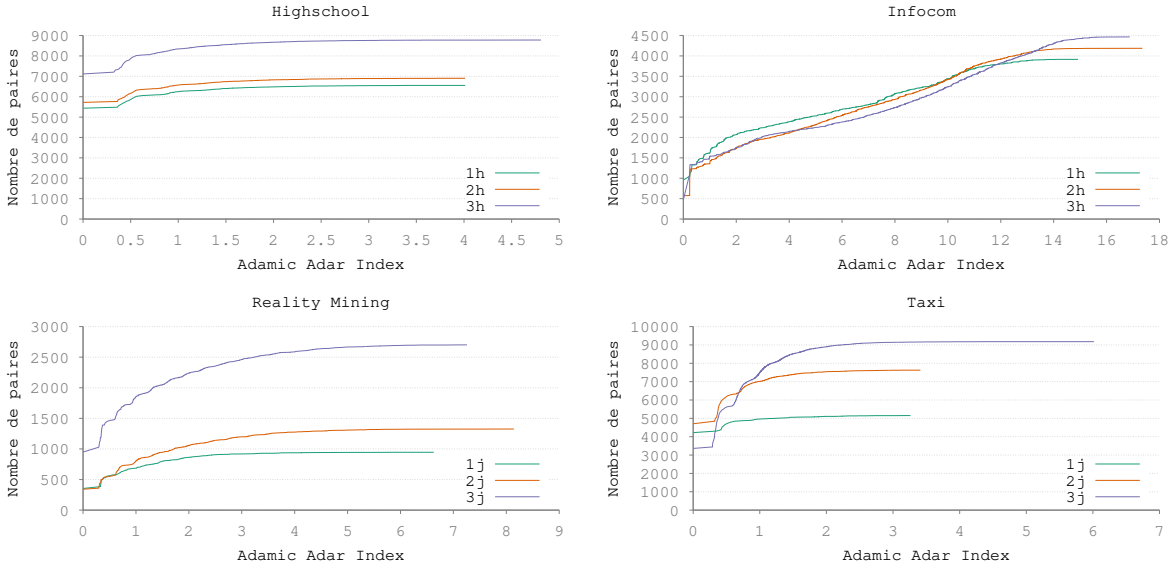


FIGURE A.2 – Fonctions de distribution cumulative de l'indice d'Adamic-Adar pour chaque période d'entraînement de chaque jeu de données.

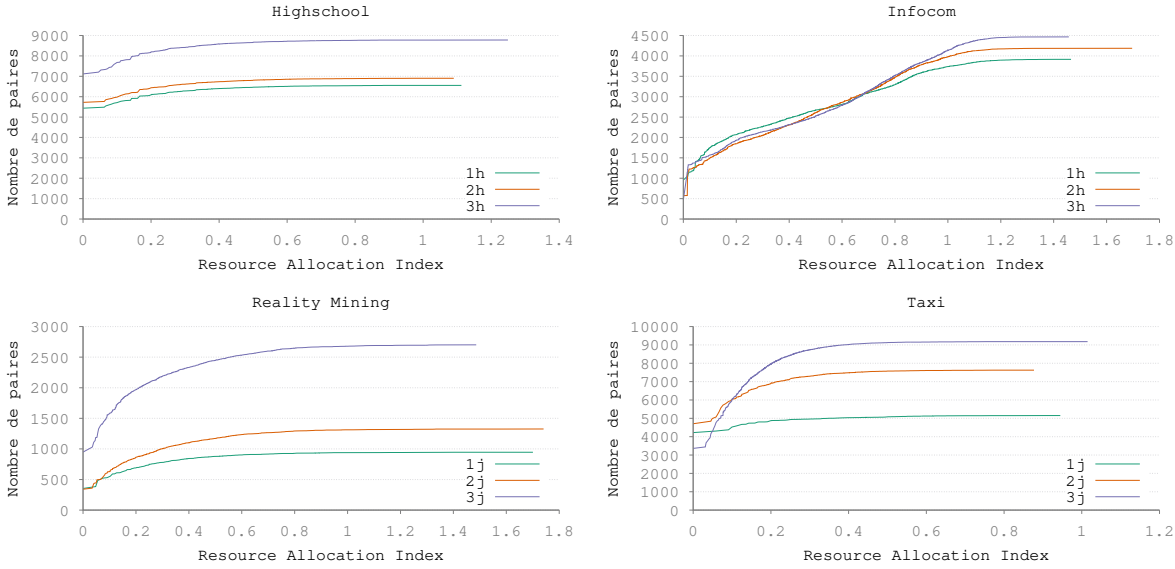


FIGURE A.3 – Fonctions de distribution cumulative de l'indice d'allocation de ressource pour chaque période d'entraînement de chaque jeu de données.

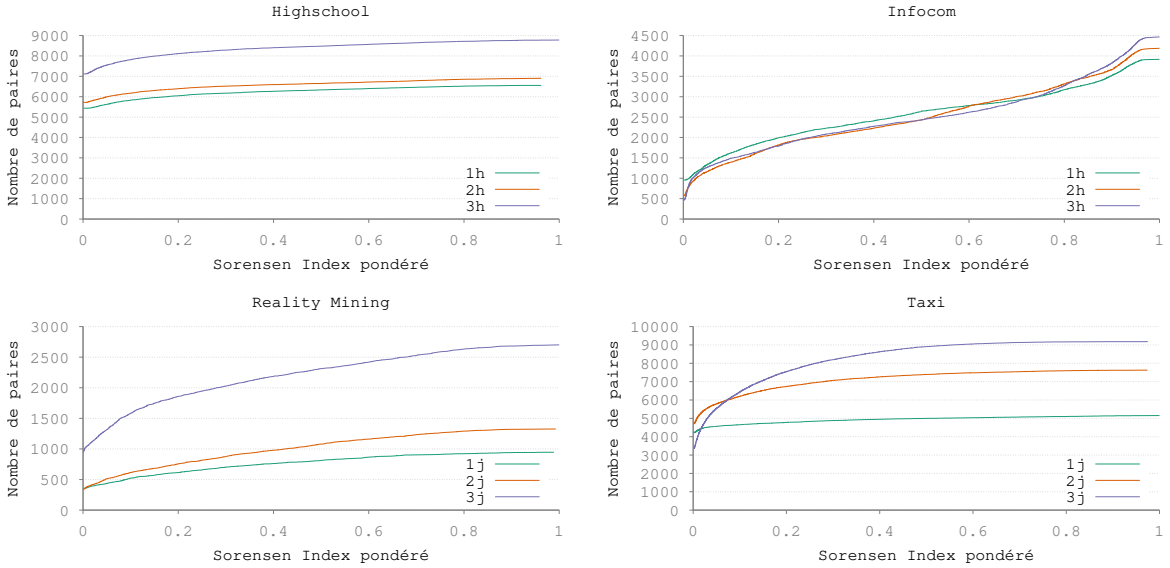


FIGURE A.4 – Fonctions de distribution cumulative de l'indice de Sørensen pondéré pour chaque période d'entraînement de chaque jeu de données.

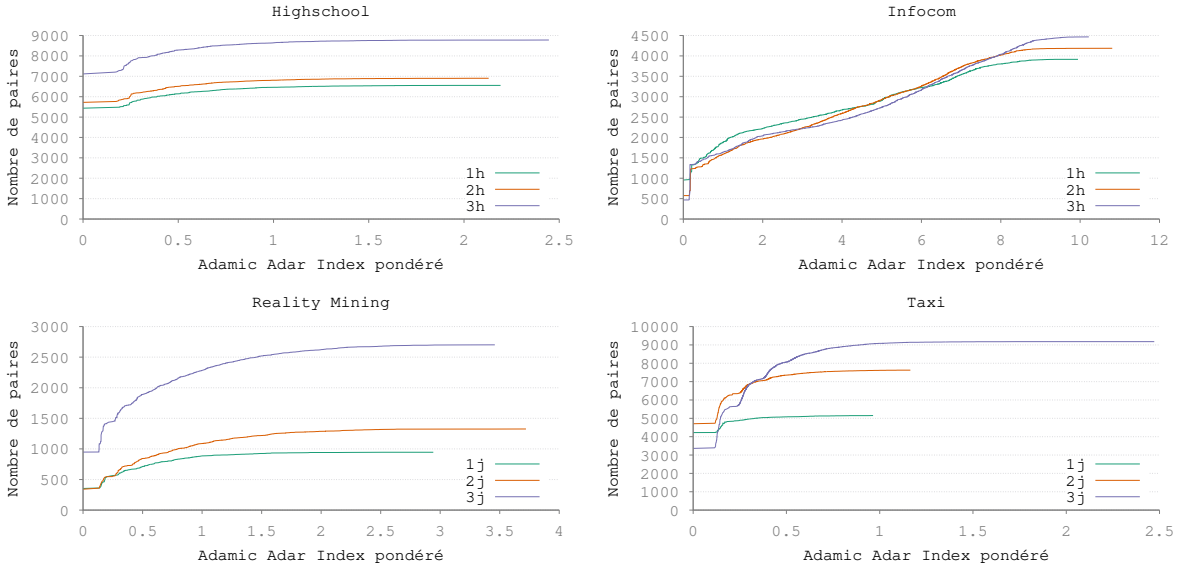


FIGURE A.5 – Fonctions de distribution cumulative de l'indice d'Adamic-Adar pondéré pour chaque période d'entraînement de chaque jeu de données.

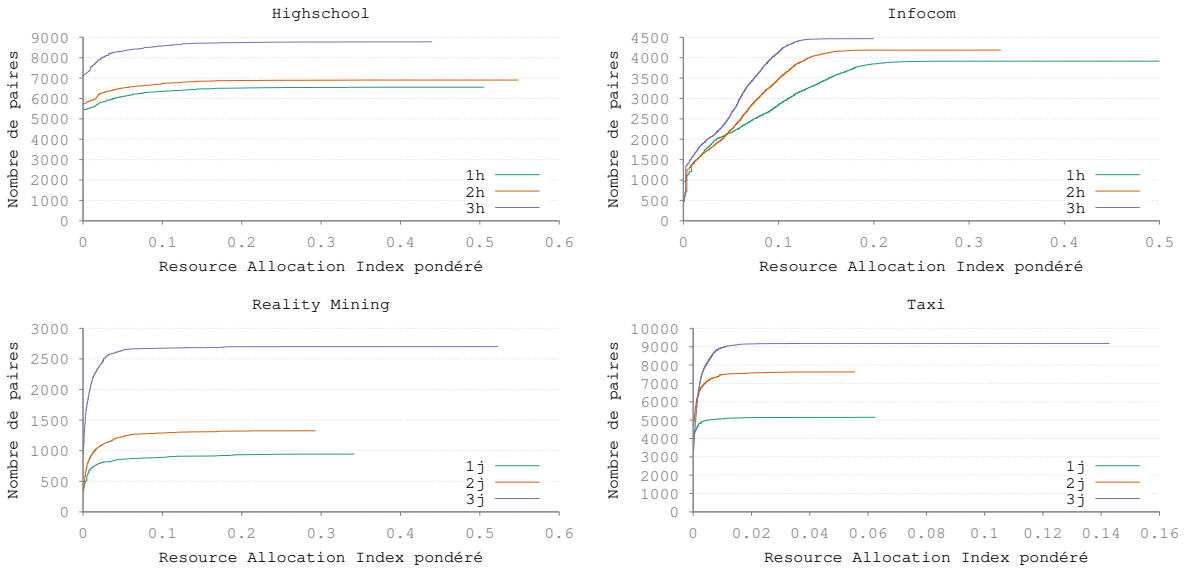


FIGURE A.6 – Fonctions de distribution cumulative de l'indice d'allocation de ressource pondéré pour chaque période d'entraînement de chaque jeu de données.

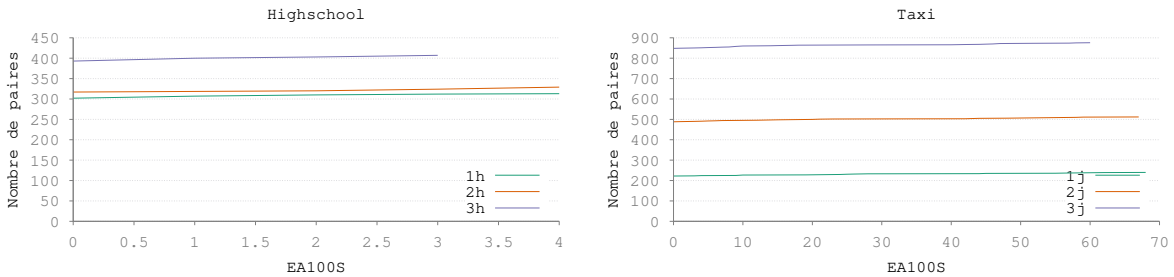


FIGURE A.7 – Fonctions de distribution cumulative de l'extrapolation de l'activité sur 100 secondes pour chaque période d'entraînement de chaque jeu de données.

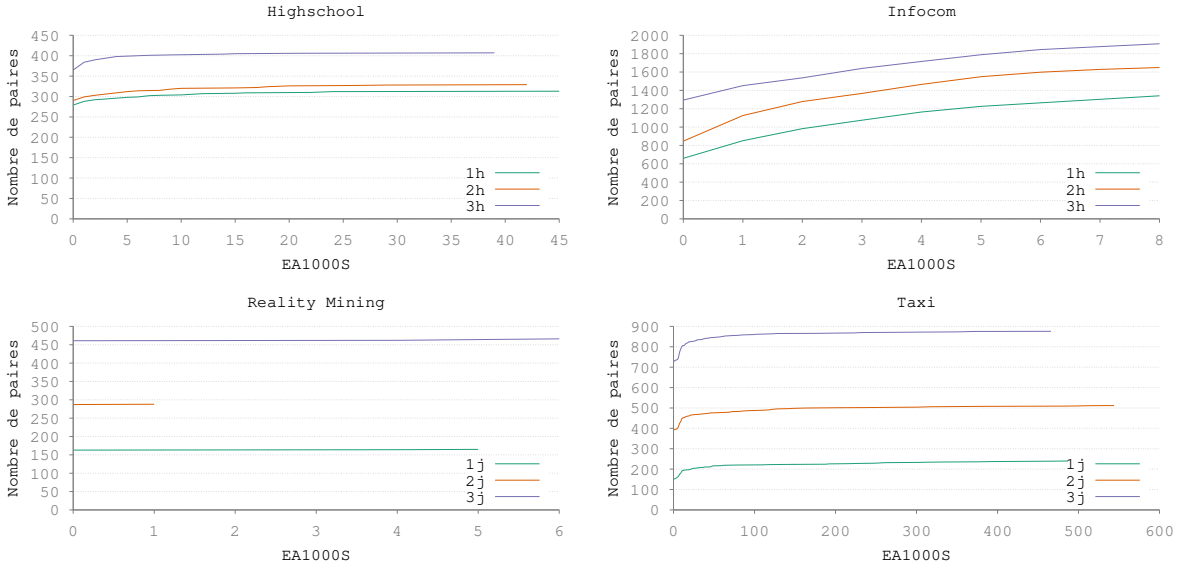


FIGURE A.8 – Fonctions de distribution cumulative de l'extrapolation de l'activité sur 1000 secondes pour chaque période d'entraînement de chaque jeu de données.

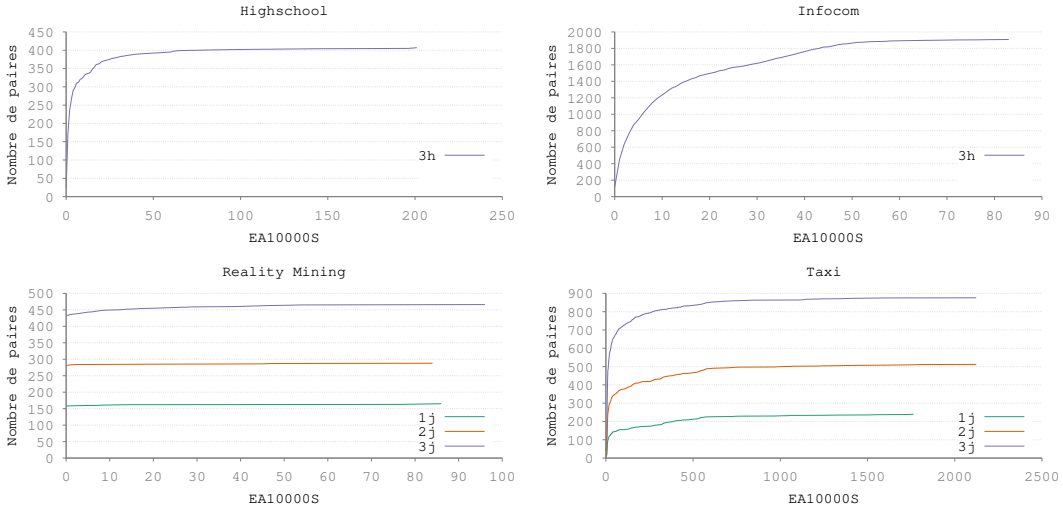


FIGURE A.9 – Fonctions de distribution cumulative de l'extrapolation de l'activité sur 10000 secondes pour chaque période d'entraînement de chaque jeu de données.

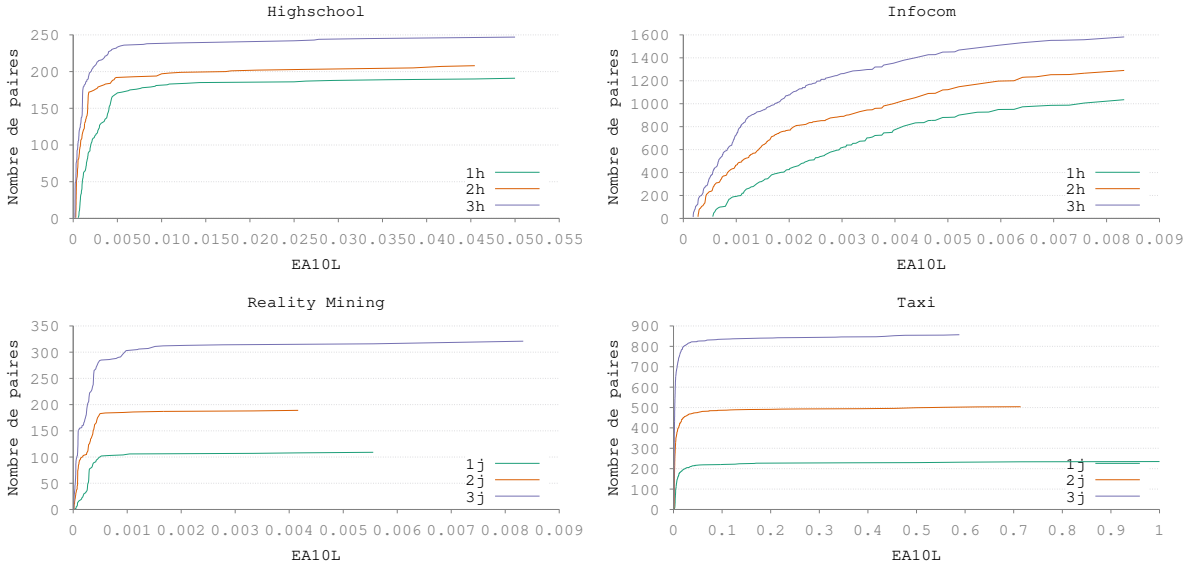


FIGURE A.10 – Fonctions de distribution cumulative du nombre d'interactions par seconde de chaque paires de nœuds, entre Ω et le temps d'apparition du $10^{ème}$ lien avant Ω pour chaque période d'entraînement de chaque jeu de données.

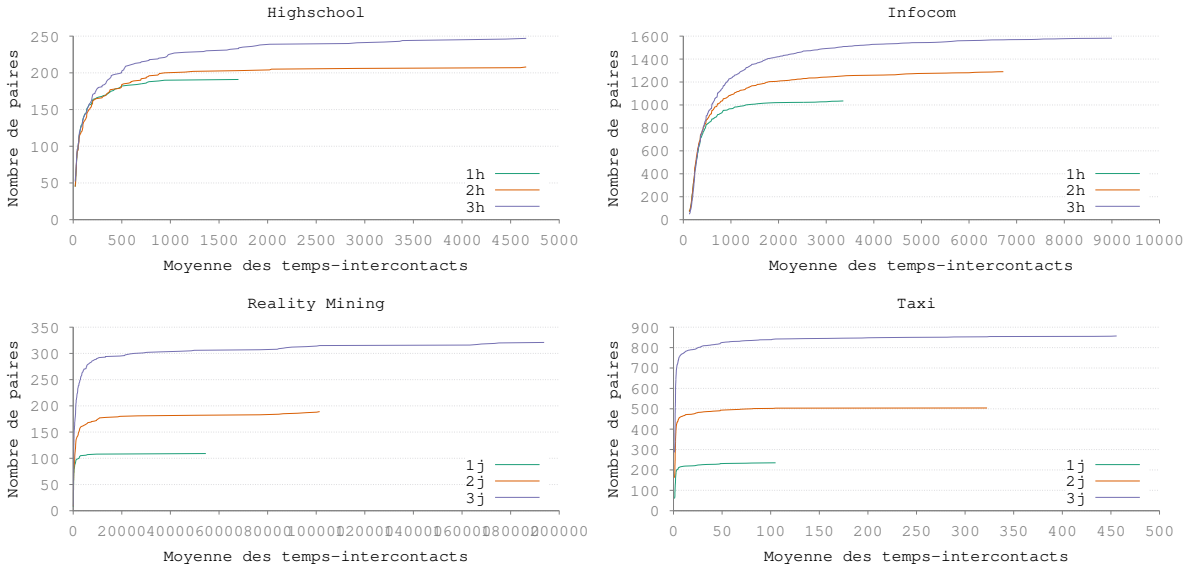


FIGURE A.11 – Fonctions de distribution cumulative de la moyenne des temps inter-contacts pour chaque période d'entraînement de chaque jeu de données.

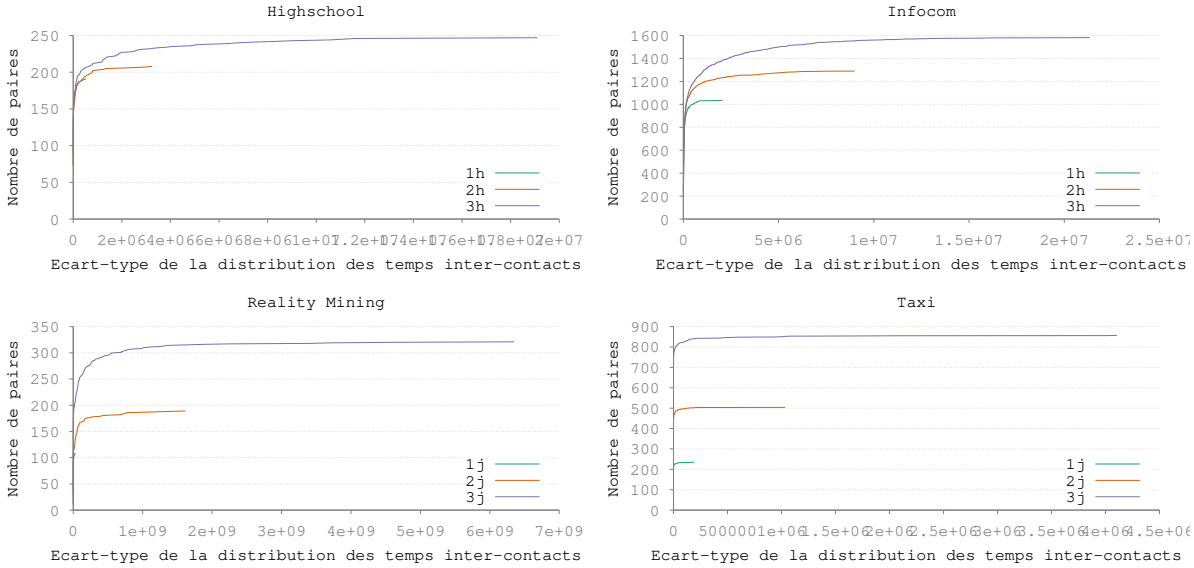


FIGURE A.12 – Fonctions de distribution cumulative de l'écart-type des temps inter-contacts pour chaque période d'entraînement de chaque jeu de données.

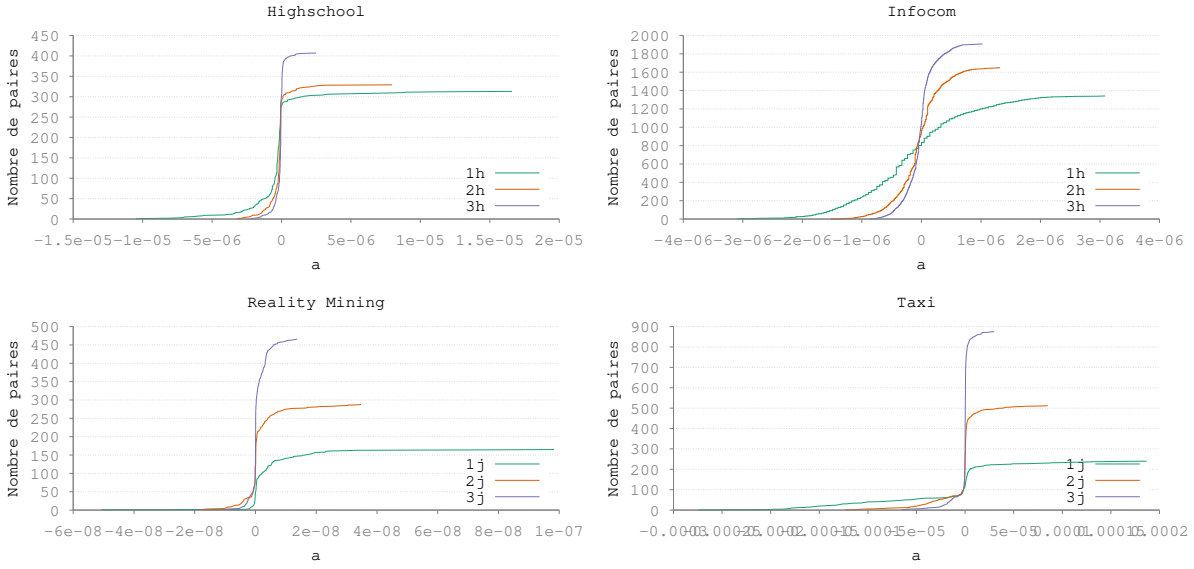


FIGURE A.13 – Fonctions de distribution cumulative du coefficient $a_{E,uv}$ pour chaque période d'entraînement de chaque jeu de données.

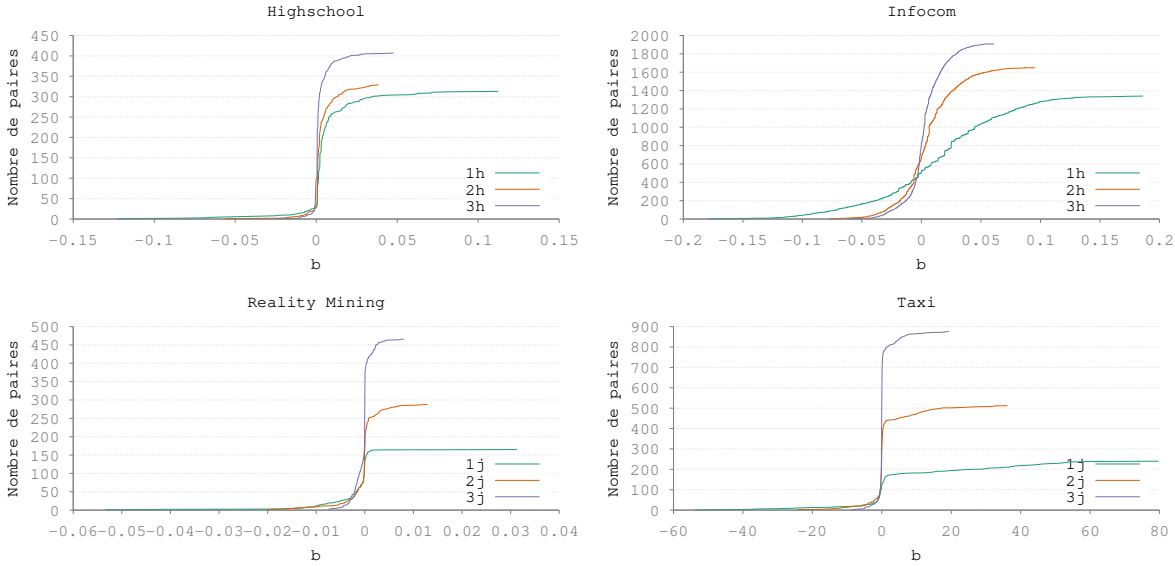


FIGURE A.14 – Fonctions de distribution cumulative du coefficient $b_{E,uv}$ pour chaque période d'entraînement de chaque jeu de données.

Annexe B

Classes – Résultats détaillés

B.1 Classes par seuils

B.1.1 Highschool

Table B.1 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Highschool* utilisant des périodes d'une heure.

Classe	F-score	Précision	Rappel	A_p	A_r
C0	0.45 ± 0.00	0.34 ± 0.00	0.71 ± 0.00	1123.00 ± 0.00	534
C0-1	–	–	–	0.00 ± 0.00	82
C0-2	0.18 ± 0.00	0.21 ± 0.00	0.17 ± 0.00	24.26 ± 0.00	30
C0-3	0.49 ± 0.00	0.34 ± 0.00	0.88 ± 0.00	1098.74 ± 0.00	422
AllClass	0.38 ± 0.03	0.28 ± 0.02	0.59 ± 0.04	1123.00 ± 0.00	534
C1	0.03 ± 0.00	0.02 ± 0.00	0.04 ± 0.00	126.57 ± 17.22	82
C2	0.12 ± 0.01	0.06 ± 0.01	1.00 ± 0.00	476.45 ± 55.66	30
C3	0.59 ± 0.03	0.54 ± 0.03	0.66 ± 0.05	519.97 ± 50.65	422

Table B.2 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Highschool* utilisant des périodes de deux heures.

Classe	F-score	Précision	Rappel	A_p	A_r
C0	0.17 ± 0.00	0.18 ± 0.00	0.15 ± 0.00	1751.00 ± 0.00	2028
C0-1	0.01 ± 0.00	0.06 ± 0.00	0.01 ± 0.00	104.32 ± 7.45	866
C0-2	0.12 ± 0.00	0.18 ± 0.00	0.09 ± 0.00	168.78 ± 1.92	348
C0-3	0.24 ± 0.00	0.19 ± 0.00	0.34 ± 0.00	1477.90 ± 7.37	814
AllClass	0.13 ± 0.01	0.14 ± 0.01	0.12 ± 0.00	1751.00 ± 0.00	2028
C1	0.04 ± 0.00	0.04 ± 0.00	0.04 ± 0.00	857.30 ± 65.15	866
C2	0.13 ± 0.01	0.14 ± 0.01	0.12 ± 0.01	295.69 ± 43.91	348
C3	0.25 ± 0.01	0.29 ± 0.01	0.22 ± 0.01	598.01 ± 42.19	814

TABLE B.3 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Highschool* utilisant des périodes de trois heures.

Classe	F-score	Précision	Rappel	A_p	A_r
C0	0.29 ± 0.00	0.22 ± 0.00	0.42 ± 0.00	3072.00 ± 0.00	1608
C0-1	0.03 ± 0.00	0.08 ± 0.00	0.02 ± 0.00	83.57 ± 6.88	405
C0-2	0.21 ± 0.00	0.21 ± 0.00	0.22 ± 0.00	206.24 ± 0.76	194
C0-3	0.33 ± 0.00	0.22 ± 0.00	0.61 ± 0.00	2782.19 ± 7.32	1009
AllClass	0.16 ± 0.01	0.12 ± 0.01	0.24 ± 0.01	3072.00 ± 0.00	1608
C1	0.08 ± 0.00	0.05 ± 0.00	0.19 ± 0.01	1653.56 ± 100.77	405
C2	0.21 ± 0.02	0.15 ± 0.02	0.36 ± 0.03	477.68 ± 85.25	194
C3	0.24 ± 0.01	0.25 ± 0.01	0.23 ± 0.02	940.75 ± 83.99	1009

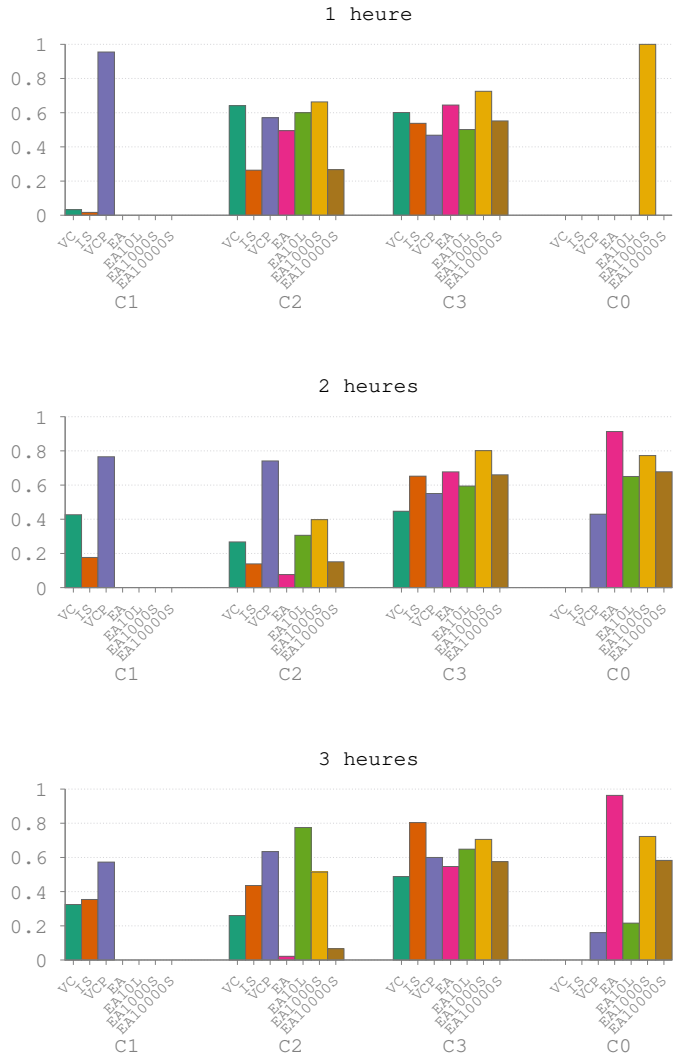


FIGURE B.1 – Répartition des coefficients des métriques pour chaque classe calculées par l'algorithme d'apprentissage pour le jeu de données *Highschool* pour chaque expérience.

B.1.2 Infocom

TABLE B.4 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Infocom* utilisant des périodes d'une heure.

Classe	F-score	Précision	Rappel	A_p	A_r
C0	0.59 ± 0.00	0.58 ± 0.00	0.59 ± 0.00	8167.00 ± 0.00	8138
C0-1	–	–	–	0.00 ± 0.00	1944
C0-2	0.51 ± 0.00	0.48 ± 0.00	0.54 ± 0.00	1795.68 ± 5.34	1621
C0-3	0.71 ± 0.00	0.61 ± 0.00	0.85 ± 0.00	6371.32 ± 5.34	4573
AllClass	0.59 ± 0.00	0.59 ± 0.00	0.60 ± 0.00	8167.00 ± 0.00	8138
C1	0.25 ± 0.00	0.27 ± 0.01	0.23 ± 0.01	1647.47 ± 107.87	1944
C2	0.53 ± 0.01	0.54 ± 0.02	0.52 ± 0.03	1568.27 ± 148.94	1621
C3	0.75 ± 0.00	0.72 ± 0.01	0.78 ± 0.02	4951.26 ± 190.65	4573

TABLE B.5 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Infocom* utilisant des périodes de deux heures.

Classe	F-score	Précision	Rappel	A_p	A_r
C0	0.55 ± 0.00	0.50 ± 0.00	0.60 ± 0.00	16737.00 ± 0.00	13939
C0-1	0.05 ± 0.01	0.31 ± 0.00	0.03 ± 0.00	216.48 ± 28.87	2668
C0-2	0.38 ± 0.00	0.46 ± 0.00	0.33 ± 0.01	1704.02 ± 22.68	2346
C0-3	0.64 ± 0.00	0.51 ± 0.00	0.85 ± 0.00	14816.50 ± 49.43	8925
AllClass	0.53 ± 0.01	0.49 ± 0.01	0.58 ± 0.01	16737.00 ± 0.00	13939
C1	0.26 ± 0.00	0.23 ± 0.01	0.31 ± 0.03	3607.44 ± 467.89	2668
C2	0.41 ± 0.01	0.38 ± 0.03	0.46 ± 0.06	2911.85 ± 652.59	2346
C3	0.65 ± 0.00	0.61 ± 0.02	0.70 ± 0.03	10217.71 ± 810.15	8925

TABLE B.6 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Infocom* utilisant des périodes de trois heures.

Classe	F-score	Précision	Rappel	A_p	A_r
C0	0.67 ± 0.00	0.63 ± 0.00	0.70 ± 0.00	22568.00 ± 0.00	20310
C0-1	0.14 ± 0.00	0.27 ± 0.00	0.09 ± 0.00	607.76 ± 25.32	1742
C0-2	0.48 ± 0.00	0.50 ± 0.00	0.47 ± 0.00	2603.94 ± 20.39	2730
C0-3	0.73 ± 0.00	0.66 ± 0.00	0.81 ± 0.00	19356.30 ± 45.22	15838
AllClass	0.56 ± 0.01	0.53 ± 0.01	0.59 ± 0.01	22568.00 ± 0.00	20310
C1	0.23 ± 0.00	0.16 ± 0.01	0.38 ± 0.02	4068.78 ± 447.42	1742
C2	0.45 ± 0.01	0.37 ± 0.02	0.59 ± 0.03	4337.37 ± 408.10	2730
C3	0.65 ± 0.01	0.69 ± 0.02	0.61 ± 0.02	14161.84 ± 715.29	15838

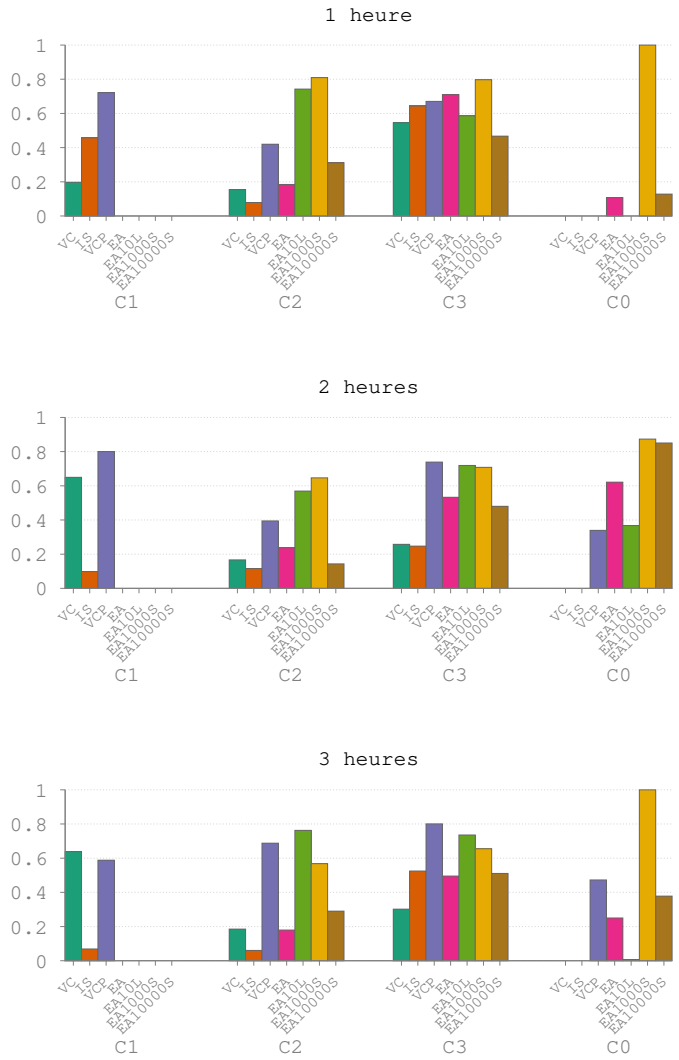


FIGURE B.2 – Répartition des coefficients des métriques pour chaque classe calculées par l'algorithme d'apprentissage pour le jeu de données *Infocom* pour chaque expérience.

B.1.3 Reality Mining

TABLE B.7 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Reality Mining* utilisant des périodes d'une heure.

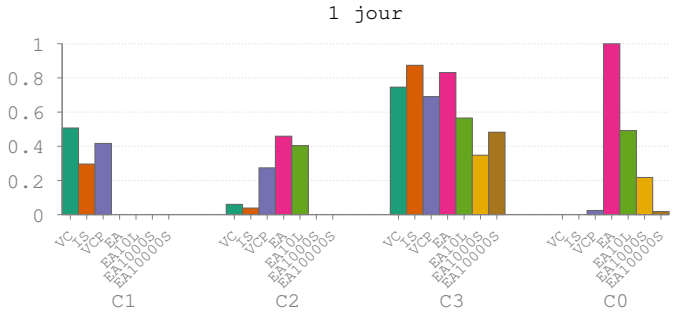
Classe	F-score	Précision	Rappel	A_p	A_r
C0	0.48 ± 0.00	0.41 ± 0.00	0.56 ± 0.00	6105.00 ± 0.00	4518
C0-1	–	–	–	55.76 ± 46.43	1218
C0-2	0.14 ± 0.02	0.13 ± 0.01	0.15 ± 0.02	313.73 ± 30.49	284
C0-3	0.57 ± 0.00	0.43 ± 0.00	0.82 ± 0.01	5735.51 ± 63.76	3016
AllClass	0.30 ± 0.02	0.26 ± 0.02	0.35 ± 0.03	6105.00 ± 0.00	4518
C1	0.07 ± 0.00	0.05 ± 0.00	0.10 ± 0.01	2529.83 ± 437.85	1218
C2	0.17 ± 0.01	0.12 ± 0.01	0.28 ± 0.05	668.96 ± 136.08	284
C3	0.47 ± 0.02	0.48 ± 0.03	0.46 ± 0.04	2906.21 ± 355.81	3016

TABLE B.8 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Reality Mining* utilisant des périodes de deux heures.

Classe	F-score	Précision	Rappel	A_p	A_r
C0	0.09 ± 0.00	0.05 ± 0.00	0.64 ± 0.00	18537.00 ± 0.00	1434
C0-1	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.01	203.49 ± 337.32	289
C0-2	0.02 ± 0.00	0.01 ± 0.00	0.04 ± 0.01	497.74 ± 61.53	121
C0-3	0.10 ± 0.00	0.05 ± 0.00	0.89 ± 0.00	17835.77 ± 391.33	1024
AllClass	0.07 ± 0.00	0.04 ± 0.00	0.52 ± 0.03	18537.00 ± 0.00	1434
C1	0.01 ± 0.00	0.00 ± 0.00	0.05 ± 0.01	3241.02 ± 668.35	289
C2	0.02 ± 0.00	0.01 ± 0.00	0.34 ± 0.05	3188.88 ± 641.07	121
C3	0.11 ± 0.01	0.06 ± 0.00	0.67 ± 0.05	12107.11 ± 815.49	1024

TABLE B.9 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Reality Mining* utilisant des périodes de trois heures.

Classe	F-score	Précision	Rappel	A_p	A_r
C0	0.41 ± 0.00	0.73 ± 0.00	0.29 ± 0.00	11395.00 ± 0.00	28692
C0-1	–	–	–	0.00 ± 0.00	8466
C0-2	0.12 ± 0.00	0.48 ± 0.00	0.07 ± 0.00	375.63 ± 0.00	2655
C0-3	0.57 ± 0.00	0.74 ± 0.00	0.46 ± 0.00	11019.37 ± 0.00	17571
AllClass	0.30 ± 0.01	0.52 ± 0.02	0.21 ± 0.01	11395.00 ± 0.00	28692
C1	0.07 ± 0.00	0.16 ± 0.00	0.05 ± 0.00	2444.36 ± 247.86	8466
C2	0.27 ± 0.01	0.29 ± 0.03	0.26 ± 0.03	2484.36 ± 531.13	2655
C3	0.40 ± 0.02	0.75 ± 0.02	0.28 ± 0.02	6466.27 ± 562.90	17571



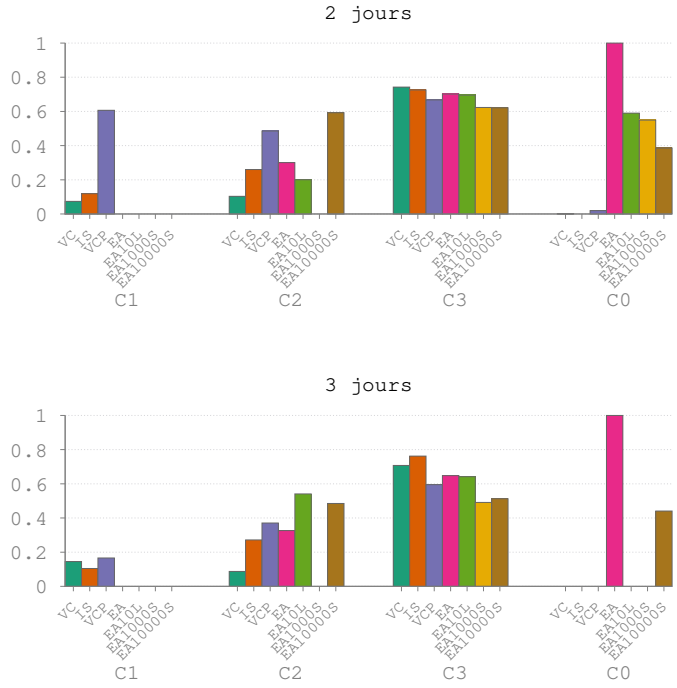


FIGURE B.3 – Répartition des coefficients des métriques pour chaque classe calculées par l'algorithme d'apprentissage pour le jeu de données *Reality Mining* pour chaque expérience.

B.1.4 Taxi

TABLE B.10 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Taxi* utilisant des périodes d'une.

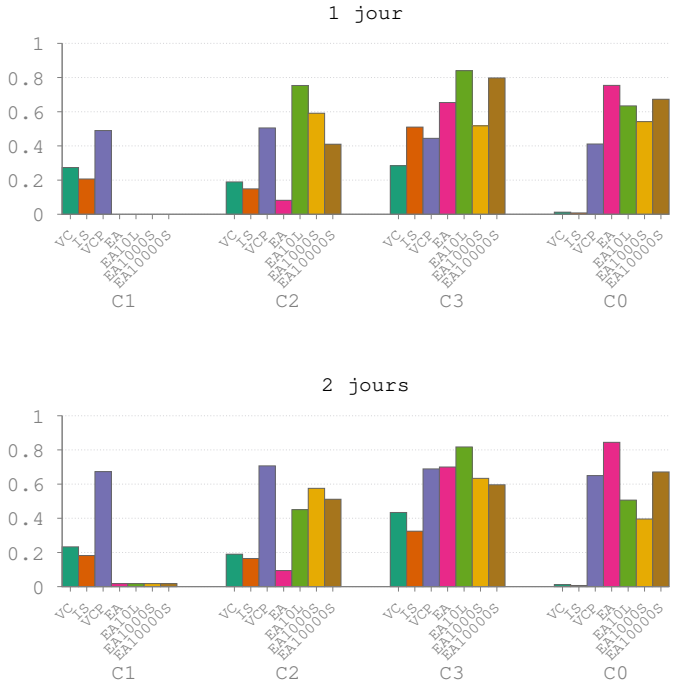
Classe	F-score	Précision	Rappel	A_p	A_r
C0	0.18 ± 0.00	0.19 ± 0.00	0.17 ± 0.00	872173.00 ± 0.00	943173
C0-1	0.10 ± 0.00	0.15 ± 0.01	0.08 ± 0.01	314192.82 ± 48811.66	566861
C0-2	0.15 ± 0.00	0.17 ± 0.01	0.13 ± 0.01	80097.97 ± 7873.91	103565
C0-3	0.28 ± 0.01	0.22 ± 0.01	0.38 ± 0.03	477882.21 ± 55815.45	272747
AllClass	0.14 ± 0.00	0.15 ± 0.00	0.14 ± 0.00	872173.00 ± 0.00	943173
C1	0.12 ± 0.00	0.12 ± 0.00	0.11 ± 0.01	499064.43 ± 45140.19	566861
C2	0.16 ± 0.00	0.15 ± 0.01	0.17 ± 0.02	120744.32 ± 26360.89	103565
C3	0.19 ± 0.00	0.20 ± 0.02	0.18 ± 0.02	252364.25 ± 48574.71	272747

TABLE B.11 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Taxi* utilisant des périodes de deux heures.

Classe	F-score	Précision	Rappel	A_p	A_r
C0	0.10 ± 0.00	0.07 ± 0.00	0.16 ± 0.00	1904076.00 ± 0.00	877069
C0-1	0.06 ± 0.00	0.05 ± 0.00	0.07 ± 0.00	604611.38 ± 44838.60	484035
C0-2	0.10 ± 0.00	0.08 ± 0.00	0.14 ± 0.01	250046.20 ± 15243.09	142368
C0-3	0.14 ± 0.00	0.09 ± 0.00	0.36 ± 0.01	1049418.42 ± 58936.50	250666
AllClass	0.09 ± 0.00	0.07 ± 0.00	0.14 ± 0.00	1904076.00 ± 0.00	877069
C1	0.06 ± 0.00	0.05 ± 0.00	0.09 ± 0.01	878717.64 ± 106700.94	484035
C2	0.10 ± 0.00	0.07 ± 0.01	0.17 ± 0.02	350423.49 ± 66061.58	142368
C3	0.13 ± 0.01	0.09 ± 0.01	0.23 ± 0.02	674934.87 ± 103942.77	250666

TABLE B.12 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Taxi* utilisant des périodes de trois heures.

Classe	F-score	Précision	Rappel	A_p	A_r
C0	0.17 ± 0.00	0.20 ± 0.00	0.14 ± 0.00	1843329.00 ± 0.00	2702338
C0-1	0.08 ± 0.01	0.17 ± 0.00	0.06 ± 0.00	566952.93 ± 55112.35	1677817
C0-2	0.15 ± 0.00	0.19 ± 0.01	0.13 ± 0.01	212408.70 ± 13551.18	313991
C0-3	0.27 ± 0.01	0.23 ± 0.01	0.34 ± 0.02	1063967.37 ± 67120.67	710530
AllClass	0.14 ± 0.00	0.17 ± 0.00	0.11 ± 0.00	1843329.00 ± 0.00	2702338
C1	0.11 ± 0.00	0.14 ± 0.00	0.08 ± 0.00	976051.73 ± 76662.41	1677817
C2	0.17 ± 0.00	0.17 ± 0.01	0.17 ± 0.01	322266.75 ± 40671.19	313991
C3	0.18 ± 0.01	0.21 ± 0.01	0.16 ± 0.01	545010.52 ± 57251.35	710530



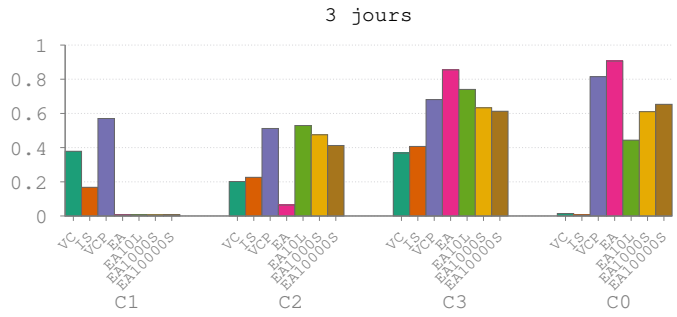


FIGURE B.4 – Répartition des coefficients des métriques pour chaque classe calculées par l'algorithme d'apprentissage pour le jeu de données *Taxi* pour chaque expérience.

B.2 Clustering – Diviser les clusters dans l'ordre inverse de leurs agrégations

TABLE B.13 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Infocom* utilisant des périodes de deux heures pour trois classes.

Classe	F-score	Précision	Rappel	A_p	A_r	Nb Paires
C0	0.55 ± 0.00	0.50 ± 0.00	0.60 ± 0.00	16737.00 ± 0.00	13939	4278
C0-1	0.04 ± 0.00	0.31 ± 0.00	0.02 ± 0.00	202.72 ± 11.47	2668	2833
C0-2	0.49 ± 0.00	0.47 ± 0.00	0.51 ± 0.00	5220.28 ± 20.52	4779	1060
C0-3	0.66 ± 0.00	0.52 ± 0.00	0.91 ± 0.00	11314.00 ± 25.07	6492	385
AllClass	0.54 ± 0.00	0.49 ± 0.00	0.59 ± 0.00	16737.00 ± 0.00	13939	4278
C1	0.26 ± 0.00	0.24 ± 0.00	0.28 ± 0.02	3082.34 ± 219.40	2668	2833
C2	0.49 ± 0.01	0.47 ± 0.02	0.52 ± 0.03	5306.99 ± 513.15	4779	1060
C3	0.68 ± 0.01	0.60 ± 0.02	0.78 ± 0.03	8347.67 ± 519.81	6492	385

TABLE B.14 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Infocom* utilisant des périodes de deux heures pour quatre classes.

Classe	F-score	Précision	Rappel	A_p	A_r	Nb Paires
C0	0.55 ± 0.00	0.50 ± 0.00	0.60 ± 0.00	16737.00 ± 0.00	13939	4278
C0-1	0.04 ± 0.00	0.31 ± 0.00	0.02 ± 0.00	203.30 ± 23.13	2668	2833
C0-2	0.48 ± 0.00	0.47 ± 0.00	0.50 ± 0.00	4835.53 ± 36.72	4567	1039
C0-3	0.66 ± 0.00	0.52 ± 0.00	0.90 ± 0.00	8930.26 ± 14.82	5109	334
C0-4	0.66 ± 0.00	0.52 ± 0.00	0.91 ± 0.01	2767.92 ± 41.57	1595	72
AllClass	0.54 ± 0.01	0.50 ± 0.00	0.60 ± 0.01	16737.00 ± 0.00	13939	4278
C1	0.25 ± 0.01	0.24 ± 0.01	0.27 ± 0.03	2969.29 ± 504.43	2668	2833
C2	0.48 ± 0.01	0.47 ± 0.02	0.49 ± 0.03	4712.19 ± 448.66	4567	1039
C3	0.69 ± 0.00	0.60 ± 0.02	0.80 ± 0.03	6862.96 ± 460.09	5109	334
C4	0.66 ± 0.01	0.57 ± 0.03	0.78 ± 0.04	2192.56 ± 207.90	1595	72

TABLE B.15 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Infocom* utilisant des périodes de deux heures pour cinq classes.

Classe	F-score	Précision	Rappel	A_p	A_r	Nb Paires
C0	0.55 ± 0.00	0.50 ± 0.00	0.60 ± 0.00	16737.00 ± 0.00	13939	4278
C0-1	0.05 ± 0.00	0.31± 0.00	0.03 ± 0.00	219.87 ± 13.34	2668	2833
C0-2	0.49 ± 0.00	0.48 ± 0.00	0.50 ± 0.00	4552.92 ± 24.43	4403	1012
C0-3	0.65 ± 0.00	0.51 ± 0.00	0.91 ± 0.00	8564.59 ± 9.81	4850	311
C0-4	0.67 ± 0.00	0.54 ± 0.00	0.91 ± 0.00	2690.63 ± 16.33	1595	70
C0-5	0.55 ± 0.00	0.44 ± 0.00	0.74 ± 0.00	708.99 ± 4.54	423	52
AllClass	0.51 ± 0.02	0.47 ± 0.02	0.56 ± 0.03	16737.00	13939	4278
C1	0.25 ± 0.02	0.23 ± 0.02	0.28 ± 0.07	3417.54 ± 1263.64	2668	2833
C2	0.44 ± 0.06	0.49 ± 0.09	0.45 ± 0.14	4384.23 ± 1968.66	4403	1012
C3	0.65 ± 0.05	0.59 ± 0.11	0.78 ± 0.15	6898.97 ± 2497.89	4850	311
C4	0.65 ± 0.03	0.64 ± 0.07	0.69 ± 0.10	1770.39 ± 426.38	1595	70
C5	0.44 ± 0.10	0.63 ± 0.10	0.37 ± 0.15	265.87 ± 145.22	423	52

TABLE B.16 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Infocom* utilisant des périodes de deux heures pour six classes.

Classe	F-score	Précision	Rappel	A_p	A_r	Nb Paires
C0	0.55 ± 0.00	0.50 ± 0.00	0.60 ± 0.00	16737.00 ± 0.00	13939	4278
C0-1	0.05 ± 0.00	0.31 ± 0.00	0.03 ± 0.00	214.88 ± 17.40	2668	2833
C0-2	0.49 ± 0.00	0.48 ± 0.00	0.50 ± 0.00	4540.81 ± 28.22	4403	1012
C0-3	0.65 ± 0.00	0.51 ± 0.00	0.91 ± 0.00	8263.92 ± 9.50	4613	301
C0-4	0.68 ± 0.00	0.53 ± 0.00	0.92 ± 0.00	2043.28 ± 20.02	1180	50
C0-5	0.55 ± 0.00	0.44 ± 0.00	0.73 ± 0.00	666.29 ± 3.15	401	50
C0-6	0.68 ± 0.00	0.57 ± 0.00	0.85 ± 0.00	1007.83 ± 8.40	674	32
AllClass	0.53 ± 0.01	0.49 ± 0.01	0.58 ± 0.02	16737.00	13939	4278
C1	0.26 ± 0.00	0.23 ± 0.01	0.29 ± 0.03	3328.64 ± 472.24	2668	2833
C2	0.49 ± 0.01	0.45 ± 0.03	0.53 ± 0.05	5202.32 ± 891.08	4403	1012
C3	0.68 ± 0.02	0.63 ± 0.06	0.76 ± 0.10	5644.77 ± 1140.94	4613	301
C4	0.68 ± 0.01	0.63 ± 0.04	0.74 ± 0.06	1405.85 ± 202.04	1180	50
C5	0.51 ± 0.04	0.49 ± 0.05	0.55 ± 0.11	460.09 ± 136.11	401	50
C6	0.65 ± 0.02	0.64 ± 0.03	0.66 ± 0.07	695.33 ± 97.13	674	32

B.3 Clustering – Diviser les clusters suivant leur taille

TABLE B.17 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Infocom* utilisant des périodes de deux heures pour trois classes.

Classe	F-score	Précision	Rappel	A_p	A_r	Nb Paires
C0	0.55 ± 0.00	0.50 ± 0.00	0.60 ± 0.00	16737.00 ± 0.00	13939	4278
C0-1	0.04 ± 0.00	0.31 ± 0.00	0.02 ± 0.00	191.56 ± 17.05	2668	2833
C0-2	0.49 ± 0.00	0.47 ± 0.00	0.51 ± 0.00	5210.98 ± 24.02	4779	1060
C0-3	0.66 ± 0.00	0.52 ± 0.00	0.91 ± 0.00	11334.46 ± 37.17	6492	385
AllClass	0.54 ± 0.00	0.50 ± 0.00	0.60 ± 0.00	16737.00 ± 0.00	13939	4278
C1	0.25 ± 0.00	0.24 ± 0.01	0.26 ± 0.01	2820.43 ± 216.22	2668	2833
C2	0.49 ± 0.01	0.49 ± 0.02	0.49 ± 0.03	4848.17 ± 460.32	4779	1060
C3	0.68 ± 0.01	0.58 ± 0.01	0.81 ± 0.02	9068.40 ± 383.66	6492	385

TABLE B.18 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Infocom* utilisant des périodes de deux heures pour quatre classes.

Classe	F-score	Précision	Rappel	A_p	A_r	Nb Paires
C0	0.55 ± 0.00	0.50 ± 0.00	0.60 ± 0.00	16737.00 ± 0.00	13939	4278
C0-1	0.05 ± 0.00	0.31 ± 0.00	0.02 ± 0.00	210.30 ± 17.61	2668	2833
C0-2	0.44 ± 0.00	0.45 ± 0.00	0.42 ± 0.00	3395.28 ± 23.03	3647	948
C0-3	0.66 ± 0.00	0.52 ± 0.00	0.92 ± 0.00	9746.62 ± 37.53	5493	315
C0-4	0.64 ± 0.00	0.52 ± 0.00	0.83 ± 0.00	3384.79 ± 12.03	2131	182
AllClass	0.53 ± 0.01	0.48 ± 0.01	0.58 ± 0.01	16737.00 ± 0.00	13939	4278
C1	0.25 ± 0.01	0.25 ± 0.01	0.25 ± 0.02	2730.82 ± 340.43	2668	2833
C2	0.43 ± 0.01	0.45 ± 0.01	0.42 ± 0.03	3409.68 ± 256.46	3647	948
C3	0.68 ± 0.01	0.64 ± 0.03	0.73 ± 0.04	6285.82 ± 648.97	5493	315
C4	0.59 ± 0.03	0.45 ± 0.04	0.89 ± 0.04	4310.68 ± 610.77	2131	182

TABLE B.19 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Infocom* utilisant des périodes de deux heures pour cinq classes.

Classe	F-score	Précision	Rappel	A_p	A_r	Nb Paires
C0	0.55 ± 0.00	0.50 ± 0.00	0.60 ± 0.00	16737.00 ± 0.00	13939	4278
C0-1	0.04 ± 0.01	0.31 ± 0.00	0.02 ± 0.00	205.20 ± 29.97	2668	2833
C0-2	0.42 ± 0.00	0.46 ± 0.00	0.39 ± 0.00	2564.65 ± 21.29	3017	855
C0-3	0.66 ± 0.00	0.51 ± 0.00	0.92 ± 0.00	9645.87 ± 47.42	5392	307
C0-4	0.56 ± 0.00	0.49 ± 0.00	0.66 ± 0.00	1585.50 ± 7.12	1177	144
C0-5	0.64 ± 0.00	0.51 ± 0.00	0.84 ± 0.00	2735.78 ± 11.19	1685	139
AllClass	0.53 ± 0.01	0.49 ± 0.01	0.59 ± 0.01	16737.00 ± 0.00	13939	4278
C1	0.25 ± 0.01	0.24 ± 0.01	0.27 ± 0.03	3002.10 ± 518.31	2668	2833
C2	0.43 ± 0.01	0.42 ± 0.03	0.45 ± 0.05	3301.11 ± 553.60	3017	855
C3	0.68 ± 0.00	0.64 ± 0.02	0.74 ± 0.04	6246.29 ± 531.12	5392	307
C4	0.55 ± 0.01	0.51 ± 0.03	0.60 ± 0.04	1392.55 ± 158.63	1177	144
C5	0.63 ± 0.02	0.51 ± 0.06	0.84 ± 0.06	2794.94 ± 444.51	1685	139

TABLE B.20 – F-score, précision, rappel et nombre de liens prédits et apparus dans chaque classe sur le jeu de données *Infocom* utilisant des périodes de deux heures pour six classes.

Classe	F-score	Précision	Rappel	A_p	A_r	Nb Paires
C0	0.55 ± 0.00	0.50 ± 0.00	0.60 ± 0.00	16737.00 ± 0.00	13939	4278
C0-1	0.05 ± 0.01	0.31 ± 0.00	0.03 ± 0.00	215.58 ± 27.45	2668	2833
C0-2	0.34 ± 0.00	0.41 ± 0.00	0.30 ± 0.00	1592.90 ± 14.04	2173	742
C0-3	0.66 ± 0.00	0.52 ± 0.00	0.92 ± 0.00	9617.42 ± 51.53	5392	307
C0-4	0.57 ± 0.00	0.50 ± 0.00	0.66 ± 0.00	1499.20 ± 11.57	1133	135
C0-5	0.61 ± 0.00	0.55 ± 0.00	0.69 ± 0.01	1344.57 ± 24.06	1082	142
C0-6	0.62 ± 0.00	0.50 ± 0.00	0.83 ± 0.00	2467.34 ± 16.55	1491	119
AllClass	0.53 ± 0.01	0.49 ± 0.01	0.58 ± 0.01	16737.00 ± 0.00	13939	4278
C1	0.26 ± 0.01	0.24 ± 0.01	0.27 ± 0.02	2975.74 ± 340.90	2668	2833
C2	0.38 ± 0.01	0.37 ± 0.01	0.40 ± 0.03	2413.79 ± 265.30	2173	742
C3	0.68 ± 0.01	0.65 ± 0.02	0.72 ± 0.03	6011.25 ± 418.80	5392	307
C4	0.55 ± 0.01	0.50 ± 0.01	0.62 ± 0.02	1396.05 ± 74.55	1133	135
C5	0.55 ± 0.02	0.47 ± 0.04	0.69 ± 0.04	1611.45 ± 232.07	1082	142
C6	0.63 ± 0.02	0.52 ± 0.04	0.81 ± 0.04	2328.72 ± 335.00	1491	119

Annexe C

Documentation du code

C.1 Getting Started

C.1.1 Pré-requis

```
Python3
Numpy
Scipy
Matplotlib
```

C.1.2 Configuration par défaut

Prédiction sans classes et classes par seuil

3 classes par activité des paires :

```
C0: Sans classes
C1: Paires sans interaction durant l'observation
C2: Moins que classthreshold=5 liens durant l'observation
C3: Plus que classthreshold=5 liens durant l'observation
AllClasses: Union de C1, C2 et C3
```

— Extrapolation de l'activité durant l'entrainement : Activité durant la période de test
— Extrapolation de l'activité durant la prédiction : Extrapolation de l'activité durant la période d'observation.
— Initialisation de la descente de gradient : Exploration aléatoire de l'espace des paramètres entre les paramètres indiqué dans le fichier de configuration pour chaque métrique.

C.1.3 Structure des données

Flot de liens non dirigés, suite de triplets :

```
t u v
...
```

<float :t> : temps d'apparition du liens

<int :u>,<int :v> : paire de nœuds

C.2 Lancer la prédiction

```
cat <data_file> | python main.py <config_file>
```

Structure du fichier de configuration :

```
<float:tstartobsT> #start time of observation training period
<float:tendobsT> #end time of observation training period
<float:tstartpredT> #start time of prediction training period
<float:tendpredT> #end time of prediction training period
<float:tstartobs> #start time of observation
<float:tendobs> #end time of observation
<float:tstartpred> #start time of pred
<float:tendpred> #end time of pred
Metrics #Metrics used:
Metric1 [parameters]
Metric2 [parameters]
Metric3 [parameters]
...
EndMetrics
[Options]
Commentaries:
Bla bla
```

Métriques disponibles :

```
PairActivityExtrapolation
commonNeighbors
weightedCommonNeighbors
resourceAlloc
```

```
weightedResourceAlloc
adamicAdar
weightedAdamicAdar
sorensenIndex
weightedSorensenIndex
PairActivityExtrapolationNbLinks<int:k>
PairActivityExtrapolationTimeInter<float:k>
fitnPointExtrapolation<int:n>
intercontactTimes
```

Paramètres : <float>,<float>

C.3 Output

Par défaut l'algorithme affiche la qualité de la prédiction et la combinaison de métriques utilisée durant la prédiction pour chaque classe.

Les activités prédites peuvent être extraites via l'option "Extract" (voir en dessous).

C.4 Autres réglages

- Extraction de la prédiction (Dans le fichier de configuration : [Option] = Extract <directory>). Exemple et format dans le dossier TestExtract
- Réglage de seuil (Dans le fichier de configuration : [Option] = Threshold <float>)
- Classes par UPGMA, coupe dans l'ordre inverse (Dans le fichier de configuration : [Option] = UPGMAINV <int :nbCluster> <float :VandPparameter>)
- Classes par UPGMA, coupe par taille (Dans le fichier de configuration : [Option] = UPGMASIZE <int :nbCluster> <float :VandPparameter>)
- Prédiction unique, utilisant les paramètre indiqué dans le fichier de configuration pour chaque métrique (Dans le fichier de configuration : [Option] = Onepred)

Bibliographie

[AA03] Lada A Adamic and Eytan Adar. Friends and neighbors on the web. *Social networks*, 25(3) :211–230, 2003.

[ABF+06] Edoardo M Airoldi, David M Blei, Stephen E Fienberg, Eric P Xing, and Tommi Jaakkola. Mixed membership stochastic block models for relational data with application to protein-protein interactions. In *Proceedings of the international biometrics society annual meeting*, volume 15, 2006.

[AC+10] Sylvain Arlot, Alain Celisse, et al. A survey of cross-validation procedures for model selection. *Statistics surveys*, 4 :40–79, 2010.

[AHCSZ06] Mohammad Al Hasan, Vineet Chaoji, Saeed Salem, and Mohammed Zaki. Link prediction using supervised learning. In *SDM06 : workshop on link analysis, counter-terrorism and security*, 2006.

[AHZ11] Mohammad Al Hasan and Mohammed J Zaki. A survey of link prediction in social networks. In *Social network data analytics*, pages 243–275. Springer, 2011.

[BBL+14] Lorenzo Bracciale, Marco Bonola, Pierpaolo Loreti, Giuseppe Bianchi, Raul Amici, and Antonello Rabuffi. CRAWDAD dataset roma/taxi (v. 2014-07-17). Downloaded from `https://crawdad.org/roma/taxi/20140717/taxicabs`, July 2014. traceset : taxicabs.

[BFDD14] Catherine A Bliss, Morgan R Frank, Christopher M Danforth, and Peter Sheridan Dodds. An evolutionary algorithm approach to link prediction in dynamic social networks. *Journal of Computational Science*, 5(5) :750–764, 2014.

[BJRL15] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis : forecasting and control*. John Wiley & Sons, 2015.

[BL11] Lars Backstrom and Jure Leskovec. Supervised random walks : predicting and recommending links in social networks. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 635–644. ACM, 2011.

[BP16] Vladimir Batagelj and Selena Praprotnik. An algebraic approach to temporal network analysis based on temporal quantities. *Social Network Analysis and Mining*, 6(1) :28, 2016.

[CALR13] Carlo Vittorio Cannistraci, Gregorio Alanis-Lobato, and Timothy Ravasi. From link-prediction in brain connectomes and protein interactomes to the local-community-paradigm in complex networks. *Scientific reports*, 3 :1613, 2013.

[CFQS12] Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, and Nicola Santoro. Time-varying graphs and dynamic networks. *International Journal of Parallel, Emergent and Distributed Systems*, 27(5) :387–408, 2012.

[Cha00] Chris Chatfield. *Time-series forecasting*. CRC Press, 2000.

[CMN08] Aaron Clauset, Cristopher Moore, and Mark EJ Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453(7191) :98–101, 2008.

[DKA11] Daniel M Dunlavy, Tamara G Kolda, and Evrim Acar. Temporal link prediction using matrix and tensor factorizations. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 5(2) :10, 2011.

[DL97] Manoranjan Dash and Huan Liu. Feature selection for classification. *Intelligent data analysis*, 1(3) :131–156, 1997.

[DLC13] Darcy Davis, Ryan Lichtenwalter, and Nitesh V Chawla. Supervised methods for multi-relational link prediction. *Social network analysis and mining*, 3(2) :127–141, 2013.

[dSSP12] Paulo Ricardo da Silva Soares and Ricardo Bastos Cavalcante Prudêncio. Time series based link prediction. In *Neural Networks (IJCNN), The 2012 International Joint Conference on*, pages 1–7. IEEE, 2012.

[DTW+12] Yuxiao Dong, Jie Tang, Sen Wu, Jilei Tian, Nitesh V Chawla, Jinghai Rao, and Huanhuan Cao. Link prediction and recommendation across heterogeneous social networks. In *Data mining (ICDM), 2012 IEEE 12th international conference on*, pages 181–190. IEEE, 2012.

[EP05] Nathan Eagle and Alex (Sandy) Pentland. CRAWDAD dataset mit/reality (v. 2005-07-01). Downloaded from `https://crawdad.org/mit/reality/20050701`, July 2005.

[GE03] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar) :1157–1182, 2003.

[GXTL10] Xinbo Gao, Bing Xiao, Dacheng Tao, and Xuelong Li. A survey of graph edit distance. *Pattern Analysis and applications*, 13(1) :113–129, 2010.

[HL09] Zan Huang and Dennis KJ Lin. The time-series link prediction problem with applications in communication surveillance. *INFORMS Journal on Computing*, 21(2) :286–303, 2009.

[HLC05] Zan Huang, Xin Li, and Hsinchun Chen. Link prediction approach to collaborative filtering. In *Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries*, pages 141–142. ACM, 2005.

[Hol15] Petter Holme. Modern temporal network theory : a colloquium. *The European Physical Journal B*, 88(9) :234, 2015.

[HS12] Petter Holme and Jari Saramäki. Temporal networks. *Physics reports*, 519(3) :97–125, 2012.

[HSL10] Theus Hossmann, Thrasyvoulos Spyropoulos, and Franck Legendre. Know thy neighbor : Towards optimal mapping of contacts to social graphs for dtn routing. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9. IEEE, 2010.

[Jac01] Paul Jaccard. *Etude comparative de la distribution florale dans une portion des Alpes et du Jura*. Impr. Corbaz, 1901.

[K+95] Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Montreal, Canada, 1995.

[Kat94] J Sylvan Katz. Geographical proximity and scientific collaboration. *Scientometrics*, 31(1) :31–43, 1994.

[KL11] Myunghwan Kim and Jure Leskovec. The network completion problem : Inferring missing nodes and edges in networks. In *Proceedings of the 2011 SIAM International Conference on Data Mining*, pages 47–58. SIAM, 2011.

[KW06] Gueorgi Kossinets and Duncan J Watts. Empirical analysis of an evolving social network. *science*, 311(5757) :88–90, 2006.

[Lab] ANT Lab. `https://ant.isi.edu/datasets/`.

[LBKT08] Jure Leskovec, Lars Backstrom, Ravi Kumar, and Andrew Tomkins. Microscopic evolution of social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 462–470. ACM, 2008.

[LLC10] Ryan N Lichtenwalter, Jake T Lussier, and Nitesh V Chawla. New perspectives and methods in link prediction. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 243–252. ACM, 2010.

[LNK07] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7) :1019–1031, 2007.

[LSY03] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations : Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1) :76–80, 2003.

[LVM17a] Matthieu Latapy, Tiphaine Viard, and Clémence Magnien. Stream graphs and link streams for the modeling of interactions over time. *CoRR*, abs/1710.04073, 2017.

[LVM17b] Matthieu Latapy, Tiphaine Viard, and Clémence Magnien. Stream graphs and link streams for the modeling of interactions over time. *arXiv preprint arXiv :1710.04073*, 2017.

[LZ11] Linyuan Lü and Tao Zhou. Link prediction in complex networks : A survey. *Physica A : Statistical Mechanics and its Applications*, 390(6) :1150–1170, 2011.

[MFB15] Rossana Mastrandrea, Julie Fournet, and Alain Barrat. Contact patterns in a high school : a comparison between data collected using wearable sensors, contact diaries and friendship surveys. *PloS one*, 10(9) :e0136497, 2015.

[MM07] Tsuyoshi Murata and Sakiko Moriyasu. Link prediction of social networks based on weighted proximity measures. In *Proceedings of the IEEE/-WIC/ACM international conference on web intelligence*, pages 85–88. IEEE Computer Society, 2007.

[MPK11] Radosław Michalski, Sebastian Palus, and Przemysław Kazienko. Matching organizational structure and social network extracted from email communication. In *Lecture Notes in Business Information Processing*, volume 87, pages 197–206. Springer Berlin Heidelberg, 2011.

[OHS05] Joshua O'Madadhain, Jon Hutchins, and Padhraic Smyth. Prediction and ranking algorithms for event-based network data. *ACM SIGKDD explorations newsletter*, 7(2) :23–30, 2005.

[P+09] Girish Palshikar et al. Simple algorithms for peak detection in time-series. In *Proc. 1st Int. Conf. Advanced Data Analysis, Business Analytics and Intelligence*, pages 1–13, 2009.

[PGPSV12] Nicola Perra, Bruno Gonçalves, Romualdo Pastor-Satorras, and Alessandro Vespignani. Activity driven modeling of time varying networks. *Scientific reports*, 2 :469, 2012.

[PK12] Manisha Pujari and Rushed Kanawati. Supervised rank aggregation approach for link prediction in complex networks. In *Proceedings of the 21st International Conference on World Wide Web*, pages 1189–1196. ACM, 2012.

[PU03] Alexandrin Popescul and Lyle H Ungar. Statistical relational learning for link prediction. In *IJCAI workshop on learning statistical models from relational data*, volume 2003. Citeseer, 2003.

[RLHC11] Troy Raeder, Omar Lizardo, David Hachen, and Nitesh V Chawla. Predictors of short-term decay of cell phone contacts in a large scale communication network. *Social Networks*, 33(4) :245–257, 2011.

[RSH+18] Mahmudur Rahman, Tanay Kumar Saha, Mohammad Al Hasan, Kevin S Xu, and Chandan K Reddy. Dylink2vec : Effective feature representation for link prediction in dynamic networks. *arXiv preprint arXiv :1804.05755*, 2018.

[SAS12] Christoph Scholz, Martin Atzmueller, and Gerd Stumme. On the predictability of human contacts : Influence factors and the strength of stronger ties. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, pages 312–321. IEEE, 2012.

[SBB10] Juliette Stehlé, Alain Barrat, and Ginestra Bianconi. Dynamical and bursty interactions in social networks. *Physical review E*, 81(3) :035101, 2010.

[SGC+09] James Scott, Richard Gass, Jon Crowcroft, Pan Hui, Christophe Diot, and Augustin Chaintreau. CRAWDAD dataset cambridge/haggle (v. 2009-05-29). Downloaded from `http://crawdad.org/cambridge/haggle/20090529`, May 2009.

[SH12] Sucheta Soundarajan and John Hopcroft. Using community information to improve the precision of link prediction methods. In *Proceedings of the 21st International Conference on World Wide Web*, pages 607–608. ACM, 2012.

[Sok58] Robert R Sokal. A statistical method for evaluating systematic relationship. *University of Kansas science bulletin*, 28 :1409–1438, 1958.

[Sør48] Thorvald Sørensen. A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons. *Biol. Skr.*, 5 :1–34, 1948.

[TAB09] Tomasz Tylenda, Ralitsa Angelova, and Srikanta Bedathur. Towards time-aware link prediction in evolving social networks. In *Proceedings of the 3rd workshop on social network mining and analysis*, page 9. ACM, 2009.

[TLL14] Lionel Tabourier, Anne-Sophie Libert, and Renaud Lambiotte. Rankmerging : Learning to rank in large-scale social networks. In *DyNakII, 2nd International Workshop on Dynamic Networks and Knowledge Discovery (PKDD 2014 workshop)*, 2014.

[TLL16] Lionel Tabourier, Anne-Sophie Libert, and Renaud Lambiotte. Predicting links in ego-networks using temporal information. *EPJ Data Science*, 5(1) :1, 2016.

[VFSML17] Tiphaine Viard, Raphaël Fournier-S'niehotta, Clémence Magnien, and Matthieu Latapy. Discovering patterns of interest in ip traffic using cliques in bipartite link streams. *arXiv preprint arXiv :1710.07107*, 2017.

[VL14] Tiphaine Viard and Matthieu Latapy. Identifying roles in an ip network with temporal and structural density. In *Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 801–806. IEEE, 2014.

[VP96] Jonathan D Victor and Keith P Purpura. Nature and precision of temporal coding in visual cortex : a metric-space analysis. *Journal of neurophysiology*, 76(2) :1310–1326, 1996.

[WXWZ15] Peng Wang, BaoWen Xu, YuRong Wu, and XiaoYu Zhou. Link prediction in social networks : the state-of-the-art. *Science China Information Sciences*, 58(1) :1–38, 2015.

[XNR10] Rongjing Xiang, Jennifer Neville, and Monica Rogati. Modeling relationship strength in online social networks. In *Proceedings of the 19th international conference on World wide web*, pages 981–990. ACM, 2010.

[ZLZ09] Tao Zhou, Linyuan Lü, and Yi-Cheng Zhang. Predicting missing links via local information. *The European Physical Journal B*, 71(4) :623–630, 2009.