Forum Jeunes Chercheurs à Inforsid 2016

Cécile Favre[1], **Chloé Artaud**[2], **Clément Duffau**[3]
Ophélie Fraisier[4,5], **Roland Kotto Kombi**[6]

1. Université de Lyon, Université Lyon 2, ERIC - EA 3083
5 av. Pierre Mendès-France, 69676 Bron Cedex, France
cecile.favre@univ-lyon2.fr

2. Université de La Rochelle, L3i - EA 2118 & DGA
av. Michel Crépeau, 17042 La Rochelle Cedex 1, France
chloe.artaud@univ-lr.fr

3. Université Côte d'Azur, CNRS, I3S - UMR 6070
2000 route des Lucioles, 06903 Sophia Antipolis Cedex, France
duffau@i3s.unice.fr

4. Université de Toulouse, Université Toulouse 3 Paul Sabatier, IRIT - UMR 5505
118 route de Narbonne, 31062 Toulouse, France

5. CEA Tech Midi-Pyrénées,
135 av. de Rangueil, 31400 Toulouse, France
Ophelie.Fraisier@cea.fr

6. INSA de Lyon, LIRIS - UMR 5205
20 av. Albert Einstein, 69100 Villeurbanne, France
roland.kotto-kombi@liris.cnrs.fr

RÉSUMÉ. *La huitième édition du Forum Jeunes Chercheurs du congrès INFORSID s'est déroulée en 2016 à Grenoble. Cette édition a accueilli 19 doctorant sélectionnés parmi 32 candidats, de première ou deuxième année, effectuant leur recherche dans le domaine des systèmes d'information. Cet article coordonné par Cécile Favre (responsable de l'organisation du Forum) présente une sélection des quatre meilleures contributions à ce forum.*

ABSTRACT. *The eighth edition of the Forum Jeunes Chercheurs of the INFORSID congress held in 2016 in Grenoble, France. It hosted 19 first-year or second-year PhD students, among 32 candidates, working in the Information Systems field. This article coordinated by Cécile Favre (in charge of the organisation of the Forum) presents a selection of the four best contributions.*

MOTS-CLÉS : *Forum jeunes chercheurs, Inforsid, système d'information.*

KEYWORDS: *PhD symposium, Inforsid, information systems.*

Introduction

La huitième édition du Forum Jeunes Chercheurs a été organisée à Grenoble le 31 mai 2016, dans le cadre du 34e congrès INFORSID [1]. L'objectif de cette manifestation scientifique était double :

– permettre aux jeunes chercheuses et chercheurs en première ou deuxième année de doctorat de présenter leur problématique de recherche et d'établir des contacts avec des équipes travaillant sur les domaines similaires ou connexes,

– offrir un aperçu des axes de recherche actuels et ainsi élargir le champ des connaissances des jeunes chercheuses et jeunes chercheurs, et des participants à la conférence plus largement.

Trente-deux soumissions ont été proposées par des doctorants de première ou deuxième année. Les actes du Forum Jeunes Chercheurs (Favre, 2016) recueillent les 19 articles sélectionnés, rédigés par des doctorants issus de divers laboratoires de recherche en France et en Tunisie (figure 1). Ces articles ont fait l'objet d'une présentation lors de la session plénière du congrès consacrée au forum, session qui s'est poursuivie par un temps d'échange avec les participants de la conférence lors d'un moment dédié. Durant toute la durée du congrès, l'exposition des posters préparés par les doctorants a permis également d'offrir un espace de rencontres et de discussions avec les chercheuses et chercheurs de la communauté des systèmes d'information.

Le nombre et la qualité des articles ainsi que la diversité des thématiques, à la fois dans les soumissions et les contributions sélectionnées des participants au Forum Jeunes Chercheurs, sont autant d'éléments qui démontrent l'importance, l'actualité et la dynamique des recherches dans le domaine des systèmes d'information.

Cet article met en avant les travaux de quatre jeunes chercheuses et chercheurs ayant participé au forum[2]. Ces contributions ont été sélectionnées sur la base de la qualité de l'exposé et de l'article de quatre pages publié dans les actes du Forum Jeunes Chercheurs. Les doctorants sélectionnés ont été invités à détailler leur travail de doctorat sur six pages, ces dernières étant assemblées dans les sections suivantes. Chaque section présente le contexte de la recherche, un bref état de l'art, la problématique de recherche abordée, les actions réalisées, ainsi que les actions futures. Ainsi, Chloé Artaud traite du processus d'authentification de documents sur la base de vérifications d'informations textuelles. Clément Duffau aborde la question de la production de logiciels fiables à partir du recueil d'expertises, et en particulier dans le cadre des systèmes de neurostimulation médicale. Ophélie Fraisier présente l'analyse de sentiments sur Twitter en intégrant les contextes spatiotemporel et social. Et, dans la dernière section, Roland Kotto Kombi considère le traitement élastique

1. INFormatique des ORganisation et Systèmes d'Information et de Décision : http://inforsid.fr/

2. Les doctorants· ayant contribué à cet article sont listés alphabétiquement.

des flux de données interrogés par des utilisateurs. Le présent article a été coordonné par Cécile Favre, responsable de l'organisation du Forum Jeunes Chercheurs 2016 se déroulant dans le cadre de la conférence INFORSID.

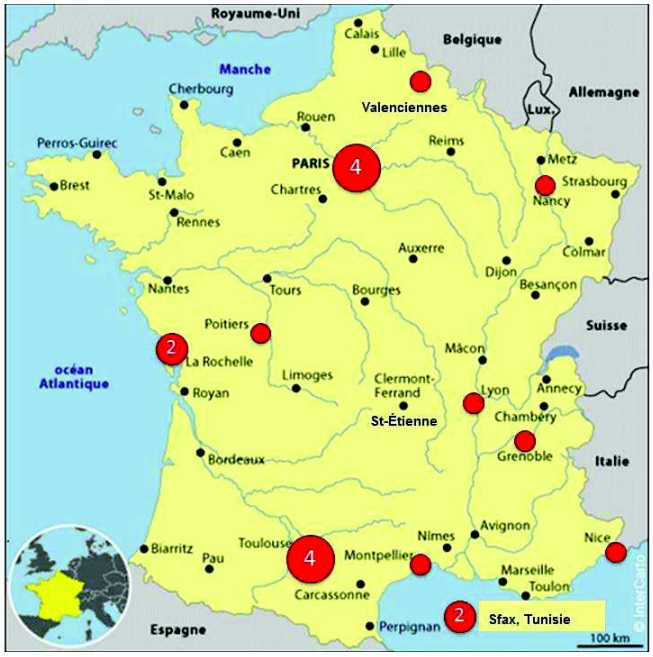

Figure 1. Répartition géographique des affiliations des 19 doctorants ayant participé au Forum Jeunes Chercheurs à Inforsid 2016

Authentification de documents par la vérification des informations textuelles

CHLOÉ ARTAUD

Doctorante de première année au Laboratoire Informatique, Image, Interaction (L3I)[3] de La Rochelle,
sous la direction d'Antoine Doucet et de Jean-Marc Ogier,
dans le cadre d'un travail réalisé avec la DGA.

1. Contexte

En 2015, 68 % des entreprises françaises interrogées dans le cadre d'une étude de PricewaterhouseCoopers[4] déclarent avoir été victimes d'une fraude au cours des

3. http://l3i.univ-larochelle.fr/
4. Global Economic Crime Survey 2016 « La fraude explose en France. La cybercriminalité au cœur de toutes les préoccupations », PwC.

24 derniers mois. Ces fraudes sont très variées (changements de RIB, fraudes comptables, fraudes aux achats, etc.) et peuvent être réalisées par des personnes externes ou internes à l'entreprise. Dans un autre registre, les administrations publiques sont également victimes de fraudes à travers de fausses déclarations. La Sécurité sociale a ainsi détecté en 2013 un préjudice de 350,5 millions d'euros lié à la fraude aux prestations sociales[5]. Les institutions ne sont pas les seules à être victimes de fraudes : les particuliers eux-mêmes peuvent être victimes d'arnaques. Lors d'un achat de voiture, certains fraudeurs peuvent par exemple falsifier la carte grise en changeant la date de construction de la voiture, lors de l'achat d'une maison, les diagnostics énergétiques...

Dans le cadre des échanges entre les individus, les entreprises et les administrations, les documents sont très importants et constituent souvent des preuves. À ce titre, il est fréquent que certains documents soient modifiés ou imités, afin d'obtenir des avantages (gagner plus d'argent ou de temps, obtenir un emploi ou une allocation) ou d'éviter des inconvénients (payer des taxes, faire des démarches pour renouveler des certificats ou des ordonnances, être expulsé d'un pays, etc.).

Chaque document doit donc être contrôlé afin de détecter les faux documents (créés de toute pièce en imitant un style ou un modèle) ou les documents falsifiés. Si des solutions existent pour sécuriser et contrôler les documents d'identité, les autres types de documents, qu'ils soient électroniques ou papier, souvent en format A4 et simplement imprimés, sont facilement modifiables.

Le sujet de notre travail s'inscrit donc dans le contexte de la lutte contre les fraudes : détecter les documents faux ou falsifiés, notamment en vérifiant les informations qu'ils contiennent.

2. État de l'art

La détection des fraudes sur les documents est très peu étudiée, malgré des besoins importants. Nos recherches traversent donc plusieurs domaines, de l'analyse de document à l'ingénierie des connaissances en passant par l'analyse d'image, la recherche d'information et les *digital forensics*, que l'on pourrait traduire par « informatique légale », par analogie à « médecine légale », ou « criminalistique numérique ». Ce dernier champ disciplinaire consiste à regrouper des preuves d'actions illégales en rapport avec l'informatique, comme la cybercriminalité ou l'usage de logiciels à des fins illégales. Trouver la preuve qu'un document nmérique a été modifié après sa date d'émission officielle, par exemple, fait partie des *digital forensics* : il s'agit de mener des investigations sur des outils informatiques afin de trouver les traces des modifications.

5. Chiffres extraits de la page http://www.securite-sociale.fr/La-fraude-sociale.

La plupart des travaux sur la détection des faux documents concernent des indices graphiques, comme la différence d'inclinaison, de taille, d'alignement ou de bruit d'un caractère par rapport aux autres (Bertrand *et al.*, 2013), la différence de police ou d'espacements des caractères au sein d'un mot (Bertrand *et al.*, 2015), le décalage d'une ligne par rapport aux marges (Beusekom *et al.*, 2010) ou encore l'inclinaison différente des lignes les unes par rapport aux autres (Beusekom *et al.*, 2009). Dans ces travaux, l'hypothèse de départ est que les fraudeurs doivent modifier les éléments du document de façon rapide et simple, souvent dans la précipitation, et que la modification n'est donc pas toujours parfaite, ce qui permet de la détecter.

D'autres travaux en analyse d'image permettent de détecter ou de suspecter des modifications, comme une étude sur les imprimantes et scanners utilisés dans la fabrication du document (Elkasrawi, Shafait, 2014) ou la vérification des tampons (Micenková *et al.*, 2014). Poisel et Tjoa (2011) dressent un état de l'art des recherches de fraudes sur les données multimédias, dont celles sur les images. Plus largement, de nombreux travaux existent sur les modifications des images de scènes naturelles, comme les *Copy-Move Forgeries* (falsifications par copie d'un morceau de l'image et déplacement dans cette même image) ou la retouche d'image.

Parallèlement à l'analyse d'image, nous nous intéressons aux informations contenues dans le document. Si les journalistes s'intéressent de plus en plus au *fact checking* (vérification des faits), il est difficile de trouver des travaux scientifiques sur la vérification d'information. Nous pouvons cependant citer Goasdoué *et al.* (2013) et leur outil FactMinder, à la croisée de l'extraction d'information, de la recherche d'information sur le Web et de l'ingénierie des connaissances.

3. Problématique

La recherche sur les faux documents se concentre donc essentiellement sur des analyses graphiques. Cependant, les documents n'étant pas seulement une matrice de pixels, nous pensons qu'il est possible de lier la détection d'indices graphiques à la détection de faux contenus ou de contenus incohérents. En effet, le contenu d'un document est composé d'informations diverses, textuelles et graphiques, ayant toutes un sens. Ces informations sont liées sémantiquement les unes aux autres et peuvent être vérifiées, seules et collectivement. Par exemple, l'adresse d'une entreprise est liée à son nom, et souvent à un numéro de téléphone, un numéro de SIRET...

Aujourd'hui, lorsqu'une personne veut vérifier une information, son premier réflexe est de faire une recherche sur Internet, à l'aide de moteurs de recherche. Généralement, lorsque les mots de la requête sont bien choisis, la page de résultats (SERP, pour *Search Engine Results Page*) donne une idée de la véracité de l'information. Sinon, il faut explorer les liens ou reformuler la requête. Nous envisageons d'automatiser ce processus afin que tout un chacun puisse vérifier simplement les documents reçus.

Nous pensons donc qu'il serait intéressant de créer un processus capable de comparer automatiquement l'information extraite du document à vérifier avec l'information accessible en ligne, afin de donner une indication sur la fiabilité du document. Ce processus, que nous présentons dans la partie suivante, serait ensuite intégré à un système plus vaste combinant les indices graphiques aux indices de véracité de l'information et de cohérence de l'information, comme nous pouvons le voir dans la figure 2.

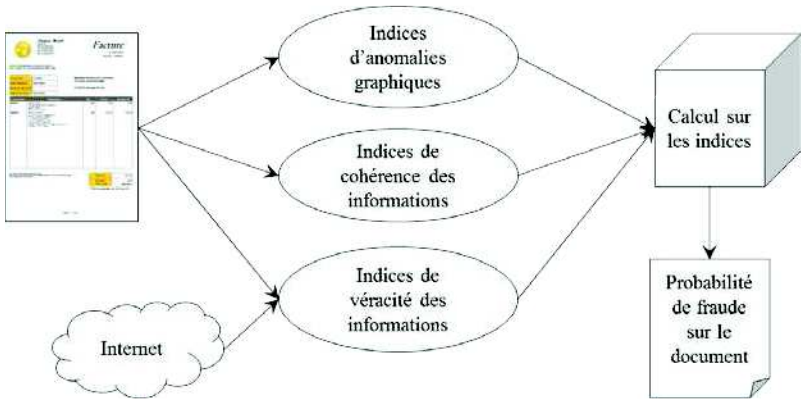

Figure 2. Schéma du système de détection de faux documents

4. Actions réalisées

Nous avons élaboré le plan d'un processus qui permettrait de vérifier les informations simples. Il s'agit tout d'abord d'extraire les informations du document (sur un document déjà OCRisé) et de les annoter afin d'en peupler une ontologie. Cette tâche fait partie du domaine de l'extraction d'information. Nous pouvons pour cela créer des règles d'extraction pour chaque information récurrente des documents. Cette extraction peut se faire avec des outils comme Unitex.

Une fois les informations extraites et identifiées, il s'agit d'en faire des requêtes pour un moteur de recherche. Les requêtes peuvent être composées d'une seule information, qui correspondra alors à une instance de l'ontologie (comme « 12345678901234 », instance de la classe « numéro SIRET »), ou de plusieurs mots combinant instances et classes (comme « numéro SIRET Exentrep ») qui associe la classe « numéro SIRET » et l'instance « Exentrep » de la classe « nom société ». Nous avons choisi d'utiliser une ontologie pour pouvoir justement associer des informations grâce aux propriétés.

Ces requêtes permettent de récupérer une SERP dont on peut extraire les informations, en utilisant les mêmes règles d'extraction que précédemment. Il faut

ensuite vérifier que les informations extraites sont identiques en les normalisant. Si elles sont identiques, elles deviennent une information unique que nous pouvons donc comparer avec celle du document à vérifier. S'il y a plusieurs résultats différents en revanche, il faut alors les traiter selon le type d'informations recherchées. Dans le cas de l'exemple précédent, le numéro de SIRET le plus probable pour une entreprise est celui qui revient le plus fréquemment dans la recherche associée au nom de l'entreprise. Dans le cas d'une recherche de prix de produit pour la vérification d'une facture ou d'un récépissé, par exemple, il ne s'agit plus de prendre le résultat le plus probable, mais de prendre en compte l'ensemble des prix trouvés, soit en en faisant la moyenne, soit en créant un intervalle acceptable. On peut également vérifier les différents résultats en les requêtant à leur tour et en analysant les résultats de la recherche.

Une fois les résultats extraits, les informations du document et celles de la SERP doivent être comparées. Lorsqu'il s'agit d'une information composée d'une suite invariable de caractères, comme une entité nommée ou un numéro, nous pouvons effectuer une comparaison caractère par caractère. De cette comparaison, nous pourrons affecter un indice de fiabilité à chaque information : plus la différence est grande, plus la fiabilité est faible.

Certaines informations sont plus complexes à traiter. Les chaînes de caractères pouvant contenir des abréviations par exemple doivent faire l'objet d'un traitement linguistique particulier : il s'agit de prendre en compte la variabilité lexicale et syntaxique pour éviter de considérer des informations vraies mais écrites différemment comme fausses. Par exemple, dans le cas de la vérification d'une adresse sur une facture de taxi, il ne faudrait pas qu'une abréviation comme « av. G^{al} de Gaulle » ne soit pas reconnue comme équivalente à l'expression « avenue du Général de Gaulle ». En effet, cela pourrait conduire à penser que la note de taxi présente une fausse adresse, et donc est un faux document.

Le système ne doit pas seulement vérifier que chaque information du document existe, mais également vérifier les liens entre les informations, c'est-à-dire les propriétés entre les instances et les classes de l'ontologie du document. Les informations extraites d'Internet pourront donc peupler une deuxième ontologie afin de pouvoir la comparer à celle représentant les informations du document. Cela permettra de vérifier les relations inférées lors des deux extractions d'informations en utilisant des méthodes d'alignement d'ontologies.

5. Actions futures

Le processus proposé se fera en deux phases : nous chercherons d'abord à évaluer chaque étape en les réalisant manuellement, puis nous tenterons d'automatiser intégralement le processus, si la première phase se révèle concluante. Il s'agira donc à terme d'automatiser le processus tout en laissant à l'utilisateur.trice la possibilité de le superviser et d'intervenir à chaque étape pour corriger d'éventuelles erreurs. Il faudra

également évaluer les performances de ce système en comparant les résultats avec ceux obtenus manuellement.

Nous pensons également explorer les liens proposés dans la SERP : nous pouvons pour des informations plus complexes avoir besoin de fouiller le web plus en profondeur. Ainsi, nous pouvons imaginer un système qui tenterait d'extraire les informations sur chaque page explorée et continuer à explorer les liens tant qu'il n'a pas trouvé suffisamment d'informations pertinentes.

Dans un dernier temps, nous chercherons à associer ces indices sémantiques aux indices graphiques décrits dans l'état de l'art pour réaliser un système de détection des fraudes complet où l'utilisateur(trice) n'aura qu'à entrer le document à vérifier pour que le système lui donne un taux de fiabilité pour ce document. La combinaison de ces indices permettra de localiser les falsifications dans le document et d'augmenter la précision et le rappel du système : si le document contient une anomalie graphique ainsi qu'une anomalie sémantique au même endroit, la probabilité que ce document soit faux sera très forte.

Il serait également intéressant de pouvoir rentrer dans le système non pas un seul document mais un lot de documents liés sémantiquement les uns aux autres pour vérifier la cohérence des informations d'un document à l'autre, ainsi que pour vérifier graphiquement que des caractères ou des mots n'ont pas été copiés-collés d'un document à l'autre. Cela permettrait de vérifier par exemple que les notes de frais de taxi ou d'hôtel présentées pour remboursement à un employeur ou une administration sont cohérentes avec les horaires et les lieux des obligations professionnelles de l'individu, inscrites dans son agenda ou dans d'autres documents, de type invitations, ordres de mission, convocations.

Du recueil d'expertises à la production de logiciels fiables : application aux systèmes de neurostimulation médicale

CLÉMENT DUFFAU	Doctorant de première année au Laboratoire d'informatique, signaux et systèmes de Sophia Antipolis (i3S)[6], sous la direction de Mireille Blay-Fornarino

1. Contexte

La problématique portée par la société AXONIC est la production de systèmes de neurostimulation médicale (SNSM) dédiés à différentes pathologies (*e.g.,* obésité, douleur fantôme). Un SNSM se présente comme un neurostimulateur (partie *hardware*), ses électrodes de stimulation et un logiciel de pilotage. Ce logiciel est un

6. http://www.i3s.unice.fr/

système d'information utilisé par différents corps de métiers (chirurgien, clinicien, patient). Les systèmes ainsi développés doivent répondre à des normes strictes qui reposent entre autres sur un cycle de développement qui inclut de nombreuses phases de tests et en particulier, une fois la vérification & validation passée, sur des campagnes d'expérimentations mettant en jeu différents sujets (de l'animal à l'homme avec différentes caractéristiques). En fonction des phases de tests, des propriétés différentes sont recherchées (*e.g.* seuils de douleur, efficacité, identification des effets indésirables). On parle d'expérimentation, lorsque l'on mène sur un même sujet un ensemble de stimulations. Chaque stimulation donne lieu à des résultats physiques « bruts » (*e.g.* ressenti patient, imagerie médicale). Les résultats obtenus dans le cadre d'une expérimentation sont ensuite interprétés par des experts métiers (*e.g.* cliniciens, chirurgiens) qui en déduisent alors la validation ou non des propriétés recherchées ; l'ensemble stimulation, résultat, et interprétation devient alors une *connaissance*.

Sur la base de ces connaissances, il s'agit alors de contraindre les SNSM pour, par exemple, interdire les stimulations qui conduisent à de la douleur. Aujourd'hui, cette étape repose principalement sur le développeur qui doit configurer le logiciel pour traduire ces différentes « connaissances » en contraintes et recommandations. Or chaque SNSM présente ses propres caractéristiques (*e.g.,* type de stimulateur, implantation ou non, forme de l'onde électrique), ce qui conduit à une grande variabilité des SNSM. Les systèmes eux-mêmes présentent des plages de paramétrages différentes en fonction des sujets, des pathologies, des propriétés recherchées.

Dans cette thèse nous nous intéressons à faciliter la production de SNSM adaptés aux sujets et aux pathologies en intégrant la prise en compte des expérimentations dans le cycle de production de ces SNSM. Bien qu'appliquée dans le contexte de cette thèse au domaine médical porté par la société AXONIC, cette problématique est généralisable à la mise en œuvre de grands systèmes, dès que la connaissance sur ces systèmes dépend d'expérimentations.

2. État de l'art

Les lignes de produits logiciels (LPL) sont des solutions bien adaptées pour produire des logiciels fiables, automatiquement (Pohl *et al.*, 2005). Dans notre cas d'étude, la ligne doit évoluer pour intégrer non seulement de nouvelles fonctionnalités (*e.g.*, nouveau stimulateur, nouvelle pathologie) mais également et surtout, s'enrichir des connaissances acquises. En cela, nous nous approchons d'une problématique d'écosystème (Bosch, 2009). Toute la difficulté est alors de faire évoluer les assets associés (au sens de la LPL, artefacts nécessaires à la production de codes) (Seidl *et al.*, 2012).

L'évolution du logiciel a largement été étudiée (Mens *et al.*, 2005). Plus spécifiquement, nous nous intéressons aux évolutions au cours du cycle de vie, à quel

Cette analyse sociale pourrait donc apporter un éclairage nouveau sur certains messages, et notamment permettre de révéler certains sentiments implicites invisibles pour les autres modèles de détection de sentiments. La problématique de ma thèse consiste donc à assembler ces deux composantes que sont l'analyse de sentiments et l'analyse sociale afin de proposer une synthèse multi-factorielle utile et lisible.

4. Actions réalisées

Afin de me familiariser avec les méthodes d'évaluation du domaine de la recherche d'information, j'ai réalisé un état de l'art des campagnes d'évaluation et des tâches proposées par celles-ci. Ceci a également permis de repérer quelles tâches pourraient être intéressantes, qu'il s'agisse de tâches actuelles auxquelles il serait possible de participer dans le futur ou de tâches passées pour lesquelles les jeux de données et les résultats sont disponibles (Fraisier, 2016). L'étape suivante a été de définir le front de recherche actuel afin de savoir où situer mon travail et mes potentielles contributions dans le domaine (voir section 4.2).

J'ai effectué une première expérimentation pour découvrir les algorithmes classiques, avec une approche à base d'apprentissage supervisé sur une tâche de classification automatique de tweets en opinions positives/négatives/neutres. Les données d'évaluation proviennent de la tâche 2 de la campagne SemEval 2013 et sont composées de 9 655 tweets d'entraînement et de 3 813 tweets de test. Les résultats, présentés dans la figure 5, montrent que l'on obtient de bons résultats avec cette approche assez simple, le meilleur des modèles testé permettant d'obtenir un F-score de 0,60 alors que le meilleur résultat obtenu par les participant de la campagne est de 0,69.

Tableau 1. Résultats des expérimentations sur la classification des tweets de la tâche 2 de la campagne SemEval 2013 (le score F1 du meilleur participant était de 0,69)

Classifieur	Score F1	Classifieur	Score F1
SVM à noyau sigmoïde	**0,60**	Forêt aléatoire	0,55
SVM à noyau linéaire	0,59	Arbre de décision	0,55
Bayésien naïf multinomial	0,57		

Afin d'étudier les liens entre sentiments et communautés, j'ai décidé d'analyser un jeu d'environ 330 000 tweets provenant du mouvement #Gamergate (Baio, 2014). Ce *hashtag* rassemblait les *Gamergaters* qui affirmaient vouloir dénoncer le manque de déontologie des journalistes du monde du jeu vidéo et leurs détracteurs qui soutenaient qu'il s'agissait d'un mouvement de harcèlement des femmes du milieu. Les interactions entre utilisateurs ont été modélisées grâce à un graphe de mentions, un graphe de retweets et un graphe global prenant en compte ces deux interactions. L'algorithme de Louvain a été utilisé pour détecter les communautés (voir tableau 2).

Tableau 2. Synthèse des communautés détectées (un graphe est considéré comme ayant une structure de communauté significative à partir d'une modularité ≥ 0,3)

Indicateur	Citations	Retweets	Global
Modularité	0,46	0,33	0,35
Nombre de communautés de plus de 10 utilisateur·trices	34	43	45
Nombre total de communautés	764	1 360	1 414

Afin de catégoriser les sentiments exprimés, j'ai constitué un jeu d'entraînement équilibré de 6 millions de critiques négatives, neutres et positives tirées de Leskovec et Krevl (2014)[11] Ce jeu de données a servi à entraîner un SVM à noyau linéaire. Une fois appliqué sur notre jeu de données, nous obtenons 62 % de tweets négatifs, 20 % positifs et 18 % neutres. En aggrégeant les sentiments au niveau auteur e avec une règle de majorité, nous avons 61 % d'auteurs exprimant des sentiments négatifs, 15 % des sentiments positifs et 24 % des sentiments neutres.

En analysant ces données au niveau des communautés, bien que les sentiments exprimés soient majoritairement négatifs et qu'on retrouve le même vocabulaire employé, le sujet de ces sentiments n'est pas le même selon qu'on se trouve dans une communauté pro ou anti Gamergate[12]. Les personnes de la communauté pro Gamergate expriment des sentiments négatifs principalement envers les journalistes et les féministes alors que les personnes de la communauté anti Gamergate expriment des sentiments négatifs envers les Gamergaters.

5. Actions futures

Les premiers travaux réalisés ont mis en évidence la nécessité de détecter le sujet des sentiments afin d'exploiter au mieux les communautés. Il sera donc nécessaire de rajouter cet élément au modèle en plus d'autres caractéristiques pouvant améliorer la classification (présence de ponctuation, d'émoticône, de négation, etc.). Je tenterai d'appliquer ce nouveau modèle à un jeu de tweets portant sur la loi travail ou le Brexit afin d'étudier les communautés présentes sur ce sujet sur Twitter, leurs préoccupations et leurs interactions.

Je déterminerai également comment évaluer les différentes composantes de ce nouveau modèle. Afin de comparer notre modèle à d'autres, une participation à une campagne d'évaluation est fortement envisagée. Il pourrait s'agir de la tâche

11. Le choix d'un jeu d'entraînement composé d'un grand nombre de critiques plutôt que d'un nombre plus restreint de tweets se base sur Mansour *et al.* (2013) qui indiquent qu'un classifieur bien entraîné sera robuste aux changements de domaines. Le jeu de données de Leskovec et Krevl (2014) étant reconnu pour sa qualité, j'ai choisi de m'appuyer sur celui-ci.
12. Malgré un vocabulaire et des sentiment similaires, le camp est facilement identifiable grâce aux utilisateurs les plus influents et aux tweets les plus représentatifs de la communauté (déterminés grâce à IraMuTeQ (Ratinaud, 2009)).

« Sentiment Analysis in Twitter » de SemEval ou la nouvelle tâche TREC combinant « Microblog » et « Temporal Summarization ». L'une des limites rencontrée par rapport aux jeux de données disponibles est liée à la nouveauté de notre tâche : nous avons à la fois besoin des annotations liées aux sentiments et des interactions sociales entre utilisateurs.

À plus long terme, ce modèle pourra être enrichi par une première étape de détection des thèmes. En effet, il paraît peu probable que les communautés des membres Twitter restent les mêmes quel que soit le sujet abordé.

Remerciements

Ce travail a été réalisé grâce à l'obtention d'un financement Contrat Laboratoire - Entreprise numéro 14050975 soutenu par la Région Midi-Pyrénées.

Traitement élastique des flux de données guidées par les flux et par les ressources

ROLAND KOTTO KOMBI | Doctorant de deuxième année au Laboratoire d'InfoRmatique en Image et Systèmes d'Information (LIRIS)[13] à l'INSA de Lyon, sous la direction de Philippe Lamarre et Nicolas Lumineau

1. Contexte

Avec la prolifération de sources de flux de données (capteurs, objets connectés, etc.), les méthodes d'acquisition, stockage et traitement de ces données ont évolué. Ces sources émettent des données éphémères qui nécessitent un traitement à la volée en temps réel. Un flux de données est une séquence d'items potentiellement infinie et imprévisible en termes de débit et de distribution des valeurs parmi les items. L'interrogation de ces flux *via* des requêtes, dites *continues*[14], représente un défi majeur en termes de performance pour le traitement en temps réel : passage à l'échelle pour la gestion de flux massifs et robustesse pour permettre le traitement continu des données. Afin de répondre à ces enjeux, des systèmes de gestion de flux de données (Schneider *et al.*, 2009 ; Peng *et al.*, 2015 ; Xu, Peng, 2016) ont été développés. Dans la suite de cette section, nous nous intéressons aux approches qui permettent l'exécution de requêtes continues de manière parallèle et distribuée.

13. http://liris.cnrs.fr/

14. Requêtes émettant des résultats au fur et à mesure que des items arrivent.

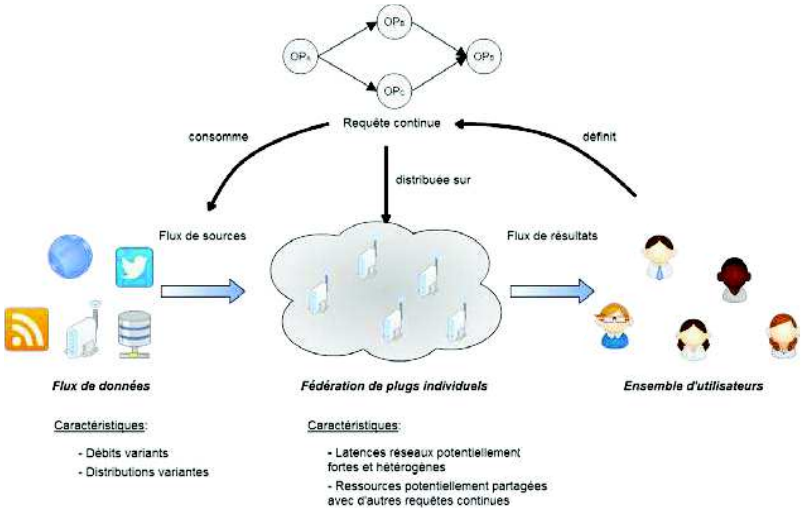

Figure 5. Fonctionnement d'une communauté d'utilisateurs

Dans le cadre du projet ANR Socioplug[15], un ensemble d'utilisateurs souhaite interroger des services générant des flux de données (voir figure 5). Pour cela, chaque utilisateur dispose d'une unité de traitement aux ressources limitées en terme de capacité de calcul et de stockage. Dans notre approche, les utilisateurs sont organisés en groupes, appelés *communautés*. Chaque communauté regroupe les utilisateurs intéressés par le résultat de requêtes équivalentes (Dufromentel *et al.*, 2015). Une fois la communauté constituée, la requête continue peut alors être exécutée de manière parallèle et distribuée sur l'ensemble des unités de traitement associées aux utilisateur trices de la communauté. Cela permet donc de mutualiser les ressources des utilisateur trices afin de permettre le traitement de la requête qui n'aurait pu être géré individuellement.

2. Problématique

Dans un contexte de traitement parallèle et distribué de requêtes continues sur des flux à débit et distribution variant, trois principaux problèmes se posent : la performance du traitement (latence), la qualité des résultats (en fonction des pertes de données) et la consommation des ressources (adéquation entre ressources et traitements).

D'un point de vue de la performance du traitement, le système doit être en mesure de renvoyer des résultats à l'utilisateur dans un délai acceptable. En

15. http://socioplug.univ-nantes.fr/index.php/SocioPlug_Project

considérant par exemple une requête continue calculant, toutes les 30 secondes, la moyenne d'une valeur sur les cinq dernières minutes. Si le calcul de la moyenne prend 45 secondes, alors les résultats observés par l'utilisateur trice ne correspondront plus à son besoin car ils auront été calculés avec des valeurs obsolètes.

D'un point de vue de la qualité des résultats, le système doit être en mesure d'évaluer voire maîtriser les pertes de données. En effet, les items d'un flux étant éphémères, des pertes irréversibles peuvent subvenir. Ces pertes peuvent être volontaires, par exemple si le système dispose d'une stratégie d'échantillonnage. Dans le cas contraire, des pertes involontaires peuvent subvenir si les ressources allouées pour les traitements ne sont pas adaptées.

Au niveau de la consommation des ressources, il est important que le système n'alloue que les ressources nécessaires pour l'exécution de chaque requête continue. Au-delà de l'aspect *Green IT* qui vise à économiser des ressources et donc de l'énergie, il est nécessaire de rappeler que les utilisateurs peuvent appartenir à plusieurs communautés, ce qui impliquera une gestion partagée des ressources.

Enfin, dans le contexte Socioplug, l'utilisateur trice n'a pas une connaissance complète sur la ou les communautés auxquelles il/elle appartient. Il ne peut donc pas avoir une influence sur les critères précédents en guidant le système. Il est donc nécessaire que le système soit capable de prendre des décisions d'auto-adaptation sans intervention d'untiers.

L'adaptation des ressources par rapport aux traitements, dit *traitement élastique* (Hirzel *et al.*, 2014), a pour objectif de maintenir une qualité et une performance satisfaisante du système quelles que soient les variations des flux d'entrée. Ce sont sur ces aspects liés à l'élasticité que nous nous focaliserons ici.

3. État de l'art

Une requête continue peut être représentée comme un graphe orienté acyclique. Chaque nœud correspond à un opérateur atomique et les arêtes définissent l'ordre d'exécution des opérateurs. Nous considérons les flux de données comme une séquence potentiellement infinie d'items. Chaque item est décrit par un *schéma* S et une estampille. Par exemple, les items d'un flux de messages sur un réseau social, peuvent être décrit par le schéma S : <auteur,sujet,destinataire,message> et l'estampille correspond à la date d'émission du message avec une précision de l'ordre de la seconde. Ainsi, plusieurs items peuvent donc partager une même estampille. Les flux que nous considérons ont un débit et une distribution des valeurs variant au cours du temps.

Le défi auquel nous nous attaquons consiste à adapter les ressources allouées à une requête continue par rapport aux variations du débit des flux et l'évolution de leurs distributions. Deux niveaux d'adaptation dynamique pour le traitement

élastique en environnement distribué sont considérés : la parallélisation et le placement des opérateurs.

D'un côté, (Schneider *et al.*, 2009 ; Xu, Peng, 2016) se sont intéressés à la parallélisation des opérateurs. En effet, en modifiant le nombre de répliques d'un même opérateur, ces systèmes influent sur sa capacité d'absorption des flux. Par exemple, un opérateur exécuté par une seule réplique et étant capable de traiter 50 items par seconde, devrait pouvoir traiter près de 100 items par seconde s'il est exécuté par deux répliques sur deux unités de traitement distinctes. La parallélisation des opérateurs permet donc d'adapter dynamiquement la capacité de traitement. Cependant, augmenter le nombre de répliques d'un opérateur peut engendrer des échanges réseaux (Xu *et al.*, 2014) plus importants, si les répliques sont distribuées sur plusieurs unités de traitement, ce qui peut dégrader les performances. De plus, les solutions existantes ne prennent pas en compte la sélectivité des opérateurs. La sélectivité d'un opérateur se définit comme le rapport entre le nombre d'items renvoyés en sortie et le nombre d'items traités. En effet, de nombreux opérateurs effectuent un filtrage des items reçus en entrée par rapport à leurs valeurs. L'algorithme présenté par Xu et Peng (2016) permet d'évaluer l'effet de la réplication d'un opérateur sur la requête continue mais ne prend pas en compte la sélectivité des opérateurs. Il apparaît alors nécessaire de prendre en compte ce facteur pour définir un mécanisme dynamique de gestion de la parallélisation des opérateurs qui pourra reposer sur une gestion pertinente de la congestion des opérateurs.

D'un autre côté, le choix d'une stratégie de placement des opérateurs a un impact direct sur les performances du système. Comme présenté dans (Xu *et al.*, 2014 ; Peng *et al.*, 2015), plusieurs stratégies de placement existent. Elles peuvent être regroupées en deux grandes catégories : les stratégies basées sur la répartition de la charge entre les unités de traitement et celles basées sur le trafic réseau. La première catégorie de stratégie de placement vise à répartir le plus équitablement possible la charge entre les unités de traitement. Du point de vue de la performance, ces stratégies sont intéressantes car elles vont permettre de répartir équitablement les augmentations de charge dues aux variations de débit des flux d'entrée. Cela assure donc une bonne stabilité du système et réduit les chances de création de goulots d'étranglement. Cependant, cette répartition de charge engendre naturellement une forte dispersion des opérateurs sur les unités de traitement et génère un trafic réseau important. La seconde catégorie de stratégies tend à regrouper dynamiquement, sur de mêmes unités de traitement, les opérateurs échangeant les volumes de données les plus importants tout en évitant de saturer ces unités. Les résultats présentés dans (Xu *et al.*, 2014 ; Peng *et al.*, 2015) ont montré que le trafic réseau est un facteur prédominant sur les performances d'un système distribué de traitement de flux. Une conséquence de ces stratégies, est la dispersion des traitements par rapport aux stratégies basées sur la répartition decharge.

4. Actions réalisées

Afin de traiter les problématiques soulevées en section 5.2, une étude bibliographique a été réalisée et les résultats ont été synthétisés dans (Kotto-Kombi *et al.*, 2015). De cette étude bibliographique, nous avons établi une classification basée sur trois grands critères : le paradigme selon lequel les requêtes continues sont représentées (graphe orienté acyclique, job MapReduce, hybride), le support d'exécution des requêtes continues (centralisé ou distribué) et enfin le type d'opérateurs supportés.

Suite à ce travail de classification, nous avons pu mettre en évidence les limites des solutions existantes. En effet, à notre connaissance, les solutions distribuées de traitement de flux nécessitent l'expertise et l'intervention de l'utilisateur pour adapter la parallélisation des opérateurs d'une requête continue. Cela présente, dans notre contexte, une contrainte non acceptable. De plus, ces systèmes ne sont pas guidés par la qualité des résultats mais uniquement par les performances. Enfin, même du point de vue de la performance, ces systèmes se basent sur une stratégie de placement sans faire de compromis entre stabilité, dispersion des traitements et performance.

Afin de répondre au problème de l'adaptation dynamique du système, nous avons proposé une architecture générique basée sur l'observation des capacités d'absorption et de traitements des opérateurs. L'intérêt principal de cette architecture est de permettre une optimisation modulaire du processus de traitement des requêtes continues.

Comme illustré sur la figure 6, cette architecture se divise en trois couches. Une première couche, dite couche de *réorganisation logique*, prend en entrée une requête continue définie soit par un graphe soit par un langage déclaratif comme CQL (Arasu *et al.*, 2006) et renvoie le graphe orienté acyclique qui est considéré comme le graphe optimal pour représenter la requête continue. La seconde couche, dite *d'allocation virtuelle*, prend en entrée le graphe optimal et définit dynamiquement le nombre de répliques associées à chaque opérateur. Cette couche est donc celle qui va influer directement sur la qualité des résultats. Pour maîtriser cette qualité, nous observons à intervalles réguliers le nombre d'items reçus et traités pour chaque opérateur. De ce fait, nous sommes en mesure d'identifier quels opérateurs disposent de ressources insuffisantes. De cette manière, lorsqu'un opérateur est considéré comme *congestionné*[16], le système augmente dynamiquement le nombre de répliques de cet opérateur et les place sur d'autres unités de traitement afin d'augmenter la capacité globale de traitement de l'opérateur. De cette manière, dans la limite des ressources dont le système dispose, nous sommes en mesure de maîtriser la qualité des résultats. Enfin, la troisième couche, dite *d'allocation physique*, place les différentes répliques d'opérateurs sur les unités de traitement en suivant une stratégie d'allocation prédéfinie. Cette couche influe donc directement sur la stabilité et la performance du

16. Opérateur générant une perte de données du fait de la saturation de sa file d'exécution.

système. Nous sommes actuellement en train d'étudier une stratégie d'allocation tenant à la fois compte de la dispersion des traitements et de la stabilité du système.

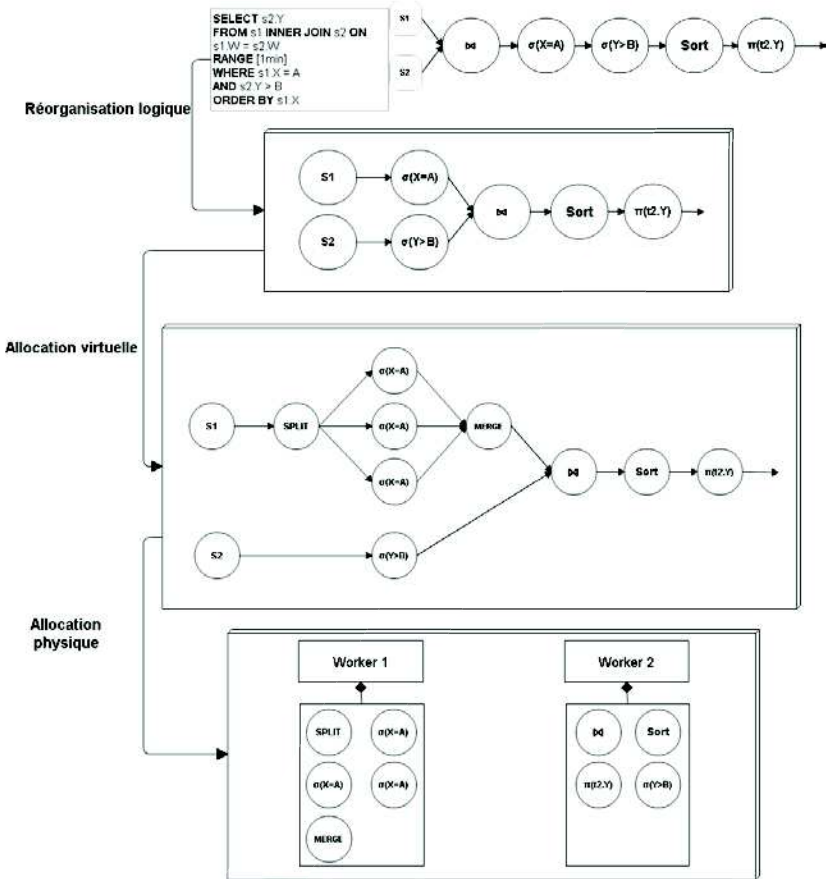

Figure 6. Architecture générique

Nous avons choisi d'implémenter la première version de notre architecture générique en surcouche du système de traitement distribué de flux de données Apache Storm[17]. Storm est un système permettant l'écriture de requêtes continues opérateur par opérateur dans un langage de programmation haut niveau (JAVA ou Clojure). Nous avons choisi ce système pour ses performances supérieures à la plupart des solutions existantes mais également pour sa robustesse (gestion automatique des pannes, forte extensibilité). Nous avons mis en place la couche d'allocation virtuelle qui permet de modifier dynamiquement le nombre de répliques

17. http://storm.apache.org/

associées à chaque opérateur sans intervention de l'utilisateur. Nous travaillons actuellement sur les indicateurs qui permettent de détecter les opérateurs congestionnés afin d'améliorer significativement les performances suite à un changement de degré de parallélisme. De plus, nous évaluons également les métriques à considérer pour réduire dynamiquement le nombre de répliques lorsqu'un opérateur est exécuté par un nombre important de répliques faiblement chargées.

5. Actions futures

En plus de nos travaux en cours, nous nous intéressons à deux axes complémentaires ayant un impact direct sur la performance du système et la qualité globale des résultats : la stabilité et la capacité d'anticipation.

Bien que le trafic réseau soit un facteur prédominant sur la latence globale d'une requête continue, la stabilité du système joue un rôle important dans les performances. Un système adaptant très fréquemment l'allocation des opérateurs sur les unités de traitement, engendre des surcoûts importants en termes de reconfiguration et potentiellement des pertes de données involontaires dégradant considérablement la qualité globale des résultats. La stratégie d'allocation que nous allons définir intégrera une fonction de coût permettant d'estimer le bénéfice d'une reconfiguration des opérateurs sur les unités de traitement par rapport au temps nécessaire pour la réaliser et l'impact sur la qualité des résultats. Cette stratégie serait alors capable d'effectuer des reconfigurations n'affectant que les opérateurs nécessitant une modification des ressources allouées pour un coût de reconfiguration maîtrisé.

En complément, nous réfléchissons également sur l'anticipation de congestion. En effet, lorsqu'une congestion est détectée, il est souvent déjà trop tard pour réagir. En effet, l'opérateur concerné doit déjà gérer une quantité d'items supérieure à sa capacité de traitement. Il apparaît alors intéressant d'adapter le mécanisme de détection des congestions afin d'estimer une congestion future à partir de l'historique des observations les plus récentes.

Enfin, cette dernière perspective nous permettrait de nous intéresser à la modification dynamique du nombre de répliques associées à un opérateur collectant un ensemble d'items pour produire un seul résultat tel que l'opérateur de jointure ou le produit cartésien. Étant donné que ces opérateurs peuvent produire de grands volumes de données en sortie par rapport aux données en entrée, il est crucial de pouvoir anticiper la congestion de ce type d'opérateurs ainsi que le volume des données en sortie.

Bibliographie

Allisio L., Mussa V., Bosco C., Patti V., Ruffo G. (2013). Felicittà: Visualizing and Estimating Happiness in Italian Cities from Geotagged Tweets. In *ESSEM@AI*IA*, vol. 1096, p. 95-106. CEUR-WS.org

Arasu A., Babu S., Widom J. (2006, juin). The cql continuous query language: Semantic foundations and query execution. *The VLDB Journal,* vol. 15, n° 2, p. 121-142. http://dx.doi.org/10.1007/s00778-004-0147-z

Baio A. (2014, octobre). *72 Hours of #Gamergate – Digging through 316,669 tweets from three days of Twitter's two-month-old trainwreck.* https://medium.com/message/72-hours-of -gamergate-e00513f7cf5d

Bertrand R., Gomez-Krämer P., Terrades O. R., Franco P., Ogier J.-M. (2013). A system based on intrinsic features for fraudulent document detection, p. 106-110. http://dblp.uni-trier.de/db/conf/icdar/icdar2013.html#BertrandGTFO13

Bertrand R., Terrades O. R., Gomez-Krämer P., Franco P., Ogier J.-M. (2015). A conditional random field model for font forgery detection, p. 576-580. http://dblp.uni-trier .de/db/conf/icdar/icdar2015.html#BertrandTGFO15 (relocated from Tunis, Tunisia).

Beusekom J. (van), Shafait F., Breuel T. (2009). Automatic line orientation measurement for questioned document examination. C*omputational forensics: Third international workshop,* IWCF 2009, the Hague, the Netherlands, August 13-14, proceedings, Z. J. M. H. Geradts, K. Y. Franke, C. J. Veenman (Eds.), p. 165-173. Berlin, Heidelberg, Springer Berlin Heidelberg. http://dx.doi.org/10.1007/978-3-642-03521-0_15

Beusekom J. (van), Shafait F., Breuel T. M. (2010). Document inspection using text-line alignment, p. 263-270. http://dblp.uni-trier.de/db/conf/das/das2010.html# BeusekomSB10

Blondel V. D., Guillaume J.-L., Lambiotte R., Lefebvre E. (2008). Fast unfolding of communities in large networks. *JSTAT*, vol. 2008, n° 10, p. P10008.

Bosch J. (2009). From software product lines to software ecosystems, vol. 1, p. 111-119. http://dl.acm.org/citation.cfm?id=1753235.1753251

Buckley J., Mens T., Zenger M., Rashid A., Kniesel G. (2005). Towards a taxonomy of software change. *Journal of Software Maintenance and Evolution*, vol. 17, n° 5, p. 309-332.

Donoghue J. O., Roantree M., Boxtel M. V. (2015). A configurable deep network for high-dimensional clinical trial data. In *Neural networks (IJCNN), 2015 International Joint Conference on*, July, p. 1-8.

Dufromentel S., Cazalens S., Lesueur F., Lamarre P. (2015). Qtor: A flexible publish/subscribe peer-to-peer organization based on query rewriting. In *Database and expert systems applications 26th international conference, DEXA 2015,* Valencia, Spain, September 1-4, 2015, proceedings, part II, p. 507-519. http://dx.doi.org/10.1007/978-3-319-22852-5_41

Elkasrawi S., Shafait F. (2014). Printer identification using supervised learning for document forgery detection, p. 146-150. http://dblp.uni-trier.de/db/conf/das/das2014.html #ElkasrawiS14

Favre C. (2016). *Actes du 8e Forum Jeunes Chercheurs du congrès INFORSID* (Ed.). https://eric.univ-lyon2.fr/~cfavre/ForumJC_INFORSID16/Actes_ForumJC_INFORSID2016.pdf.

Favre J.-M., Establier J., Blay-Fornarino M. (Eds.) (2006). *L'ingénierie dirigée par les modèles : au-delà du MDA*, Cachan, France, Hermes-Lavoisier.

Feldman R. (2013). Techniques and applications for sentiment analysis. *Communications of the ACM*, vol. 56, n° 4, p. 82.

Fraisier O. (2016, février). *Information Retrieval – Evaluation Campaigns* n° IRIT/RR–2016–04–FR. https://www.irit.fr/publis/IRIS/2016_R_F.pdf.

Goasdoué F., Karanasos K., Katsis Y., Leblay J., Manolescu I., Zampetakis S. (2013, June). *Fact Checking and Analyzing the Web.* http://hal.inria.fr/hal-00814285

Hirzel M., Soulé R., Schneider S., Gedik B., Grimm R. (2014, mars). A catalog of stream processing optimizations. *ACM Comput. Surv.*, vol. 46, n° 4, p. 46:1-46:34. http://doi.acm.org/10.1145/2528412

Hu X., Tang L., Tang J., Liu H. (2013). Exploiting social relations for sentiment analysis in microblogging. In *WSDM*, p. 537-546.

Hubaux A., Classen A., Heymans P. (2009). Formal Modelling of Feature Configuration Workflows. In *SPLC'09*, IEEE, p. 221-230.

Irsoy O., Cardie C. (2014). Opinion Mining with Deep Recurrent Neural Networks. In *EMNLP*, p. 720-728.

Kang K. C., Cohen S. G., Hess J. A., Novak W. A., Spencer Peterson A. (1990). *Feature-Oriented Domain Analysis (FODA) feasibility study*. Rapport technique, November. The Software Engineering Institute. http://www.sei.cmu.edu/reports/90tr021.pdf

Kotto-Kombi R., Lumineau N., Lamarre P., Caniou Y. (2015, octobre). *Parallel and Distributed Stream Processing: Systems Classification and Specific Issues.* https://hal.archives-ouvertes.fr/hal-01215287 (working paper or preprint)

Kuo Y.-H., Fu M.-H., Tsai W.-H., Lee K.-R., Chen L.-Y. (2016). Integrated microblog sentiment analysis from users' social interaction patterns and textual opinions. *Applied Intelligence*, vol. 44, n° 2, p. 399-413.

Leskovec J., Krevl A. (2014, jun). *SNAP Datasets: Stanford large network dataset collection.* http://snap.stanford.edu/data.

Liu B. (2012). Sentiment Analysis and Opinion Mining. *Synthesis Lectures on HLT*, vol. 5, n° 1, p. 1-167.

Mansour R. H., Refaei N., Gamon M., Sami K., Abdel-Hamid A., Elbahtemy A. (2013). Revisiting The Old Kitchen Sink: Do We Need Sentiment Domain Adaptation? In *RANLP*. https://www.microsoft.com/en-us/research/publication/revisiting-the-old-kitchen-sink-do-we-need-sentiment-domain-adaptation/

Mens T., Demeyer S., Wermelinger M., Hirschfeld R., Ducasse S., Jazayeri M. (2005). Challenges in Software Evolution. *International Workshop on Principles of Software Evolution (IWPSE)*, p. 13-22.

Micenková B., Beusekom J. van, Shafait F. (2014). Stamp verification for automated document authentication, vol. 8915, p. 117-129. http://dblp.uni-trier.de/db/conf/iwcf/iwcf2014.html#MicenkovaBS14

Peng B., Hosseini M., Hong Z., Farivar R., Campbell R. H. (2015). R-storm: Resource-aware scheduling in storm. In *Proceedings of the 16th annual middleware conference,* Vancouver, BC, Canada, December 07-11*,* p. 149-161. http://doi.acm.org/10.1145/281457 6.2814808

Pohl K., Böckle G., Linden F. J.v.d. (2005). *Software product line engineering: Foundations, principles and techniques*. Secaucus, NJ, USA, Springer-Verlag New York, Inc.

Poisel R., Tjoa S. (2011). Forensics investigations of multimedia data: A review of the state-of-the-art.,p. 48-61. http://dblp.uni-trier.de/db/conf/imf/imf2011.html#PoiselT11

Polacsek T. (2016). *Argumentation tree: A new player in your diagrams,* p. 223-224.

Ratinaud P. (2009). *IRaMuTeQ : Interface de R pour les Analyses Multidimensionnelles de Textes et de Questionnaires.* http://www.iramuteq.org

Schneider S., Andrade H., Gedik B., Biem A., Wu K.-L. (2009, May). Elastic scaling of data parallel operators in stream processing. In *Parallel distributed processing, 2009. IPDPS 2009. IEEE international symposium on*, p. 1-12.

Seidl C., Heidenreich F., Assmann U. (2012). Co-evolution of Models and Feature Mapping in Software Product Lines. *Proceedings of the 16th International Software Product Line Conference (SPLC)*, vol. 1, p. 76-85. http://doi.acm.org/10.1145/2362536 .2362550

Tang J., Chang Y., Liu H. (2014). Mining social media with social theories: a survey. *ACM SIGKDD Explorations Newsletter*, vol. 15, n° 2, p. 20-29.

West R., Paskov H. S., Leskovec J., Potts C. (2014). Exploiting Social Network Structure for Person-to-Person Sentiment Analysis. *Transactions of the ACL*, vol. 2, p. 297-310.

Wohlin C., Runeson P., Höst M., Ohlsson M. C., Regnell B., Wesslén A. (2000). *Experimentation in software engineering: An introduction*. Norwell, MA, USA, Kluwer Academic Publishers.

Wu F., Huang Y., Song Y. (2016). Structured microblog sentiment classification via social context regularization. *Neurocomputing*, vol. 175, p. 599-609.

Xia R., Zong C., Li S. (2011). Ensemble of feature sets and classification algorithms for sentiment classification. *Information Sciences*, vol. 181, n° 6, p. 1138-1152.

Xu, Peng G. (2016). Stela: Enabling stream processing systems to scale-in and scale-out on-demand. In *Proc. IEEE International Conference on Cloud Engineering (IC2E).*

Xu J., Chen Z., Tang J., Su S. (2014, June). T-storm: Traffic-aware online scheduling in storm. In *Distributed computing systems (ICDCS), 2014 IEEE 34th International Conference on*, p. 535-544.

Zhang K., Yang Y., Sun A., Liu H. (2014). A Systematic Framework for Sentiment Identification by Modeling User Social Effects. In *WI-IAT*, vol. 2, p. 172-179.