

THESE DE DOCTORAT DE

L'ÉCOLE NATIONALE SUPERIEURE MINES-TELECOM ATLANTIQUE

BRETAGNE PAYS DE LA LOIRE - IMT ATLANTIQUE

COMUE UNIVERSITE BRETAGNE LOIRE

ECOLE DOCTORALE N° 601

Mathématiques et Sciences et Technologies

de l'Information et de la Communication

Spécialité : *Informatique*

Par

Alassane Samba

Science des données au service des réseaux d'opérateur

Proposition de cas d'utilisation, d'outils et de moyens de déploiement

Thèse présentée et soutenue à « Lieu », le « date » (6)

Unité de recherche : (7)

Thèse N° : (8)

Rapporteurs avant soutenance :

Isabelle Chrisment
Lyes Khoukhi

Professeur, LORIA / Telecom Nancy
Maitre de Conférences (HDR), Université de Technologie de Troyes

Composition du Jury :

Alberto Blanc
Yann Busnel
Christelle Caillouet
Isabelle Chrisment
Philippe Dooze
Lyes Khoukhi
Gerardo Rubino
Gwendal Simon

Maître de Conférences, IMT Atlantique
Professeur, IMT Atlantique
Maitre de Conférences, Université de Nice Sophia Antipolis
Professeur, LORIA / Telecom Nancy
Chercheur senior, Orange Labs
Maitre de Conférences HDR, Université de Technologie de Troyes
Directeur de Recherche, INRIA
Professeur, IMT Atlantique

Résumé

L'évolution des télécommunications a mené aujourd'hui à un foisonnement des appareils connectés et une massification des services multimédias. Face à cette demande accrue de service, les opérateurs ont besoin d'adapter le fonctionnement de leurs réseaux, afin de continuer à garantir un certain niveau de qualité d'expérience à leurs utilisateurs.

Pour ce faire, les réseaux d'opérateur tendent vers un fonctionnement plus cognitif voire autonomique. Il s'agit de doter les réseaux de moyens d'exploiter toutes les informations ou données à leur disposition, les aidant à prendre eux-mêmes les meilleures décisions sur leurs services et leur fonctionnement, voire s'autogérer. Il s'agit donc d'introduire de l'intelligence artificielle dans les réseaux. Cela nécessite la mise en place de moyens d'exploiter les données, d'effectuer sur elles de l'apprentissage automatique de modèles généralisables, apportant les informations qui permettent d'optimiser les décisions. L'ensemble de ces moyens constituent aujourd'hui une discipline scientifique appelée science des données. Cette thèse s'insère dans une volonté globale de montrer l'intérêt de l'introduction de la science des données dans différents processus d'exploitation des réseaux.

Premièrement, la thèse comporte deux contributions algorithmiques montrant des cas d'utilisation concrets de la science des données pour le réseau. Il s'agit, d'une part, de modèles permettant une prédiction instantanée du débit pour les utilisateurs du réseau cellulaire d'un opérateur. D'autre part, il s'agit d'une approche permettant d'analyser automatiquement les résultats de tests de performances des réseaux virtualisés.

En second lieu, la thèse présente deux contributions logicielles. Il s'agit d'abord, d'un nouvel outil facilitant l'analyse et la visualisation de corrélation, et ensuite, de l'implémentation de nouveaux moyens facilitant le déploiement un algorithme issu de la science des données en tant que service.

Les résultats concluants de ces différents travaux ont démontré l'intérêt et la faisabilité de l'utilisation de la science des données pour l'exploitation des réseaux d'opérateur. Ces résultats ont aussi fait l'objet de plusieurs utilisations par des projets connexes.

Mots-clefs

science des données, apprentissage automatique, analyse de résultats de tests, analyse de corrélations, réseau cellulaire, réseau virtualisé, réseau cognitif, prédiction de débit, déploiement d'algorithmes, API.

Abstract

The evolution of telecommunications has led today to a proliferation of connected devices and a massification of multimedia services. Faced with this increased demand for service, operators need to adapt the operation of their networks, in order to continue to guarantee a certain level of quality of experience to their users.

To do this, operator networks tend towards a more cognitive or autonomic functioning. It is about giving the networks the means to exploit all the information or data at their disposal, helping them to make the best decisions about their services and operations, and even self-manage. It is therefore a question of introducing artificial intelligence into networks. This requires setting up means to exploit the data, to carry out on them the automatic learning of generalizable models, providing information that can optimize decisions. All these means today constitute a scientific discipline called data science. This thesis fits into a global desire to show the interest of the introduction of data science in different network operating processes.

It firstly contains two algorithmic contributions showing concrete usecases of data science for the network. These are, on the one hand, models allowing an instantaneous throughput prediction for users of an operator's cellular network. On the other hand, it is a approach to automatically analyze performance test results for virtualized networks.

Second, the thesis presents two software contributions. It is first about a new tool to facilitate correlation analysis and visualization, and then, the implementation of new ways to facilitate the deployment of an algorithm derived from data science as a service.

The conclusive results of these various studies have demonstrated the interest of data science for the exploitation of operator networks and have been used for several purposes. The conclusive results of these various studies have demonstrated the interest and feasibility of using data science for the operation of operator networks. These results have also been used by several related projects.

Keywords

data science, machine learning, test results analysis, correlation analysis, cellular network, virtualized network, cognitive network, throughput prediction, deployment of algorithms, API.

Table des matières

1	Introduction	1
	Contexte et objectifs de recherche	1
	Contributions de la thèse	3
	Structure du document	7
I	État de l'art	9
2	Analyse et apprentissage automatique de données	11
2.1	Représentation des données statistiques	12
2.2	Apprentissage	13
2.3	Focus sur l'apprentissage supervisé	15
2.4	Outils et langages pour l'apprentissage de données	19
3	Les réseaux d'opérateur d'aujourd'hui et de demain	21
3.1	Modèles et protocoles de communication modernes	22
3.2	Réseau d'opérateur	26
3.3	Réseau mobile	28
3.4	Qualité de Service	30
3.5	Réseaux programmables, cognitifs et autonomiques	34
II	Contributions algorithmiques	41
4	Prédiction instantanée de débit dans le réseau mobile	43
4.1	Introduction	44
4.2	État de l'art	45
4.3	Données collectées	48
4.4	Études des corrélations bivariées avec le débit	54
4.5	Modèles de prédiction instantanée du débit mobile	57
4.6	Discussion : Opportunités et Limites	60
4.7	Pistes d'implémentation	61
4.8	Conclusion	63
5	Analyse de résultats de tests de performance de réseaux virtualisés	65
5.1	Développement de la NFV et automatisation de réseau	66
5.2	La vérification d'infrastructure NFV et le projet Yardstick	67
5.3	Approche d'analyse proposée	68
5.4	Outil TOM	70
5.5	Application aux résultats de tests de bande passante mémoire	72
5.6	Travaux connexes	78

5.7 Discussion	78
5.8 Conclusion	78
III Contributions logicielles	81
6 Linkspotter : outil d'analyse et de visualisation de corrélation	83
6.1 Mesure de corrélation entre deux variables	84
6.2 BeEF : algorithme de discréétisation	85
6.3 MaxNMI : information mutuelle normalisée maximale	86
6.4 Regroupement de variables utilisant la matrice de corrélation	87
6.5 Visualisation de la matrice de corrélation	87
6.6 Conclusion	89
7 Déploiement d'algorithmes de Data Science en tant que service	91
7.1 Travaux connexes	93
7.2 Motivations et contexte	93
7.3 Approche « Package & Containerize »	97
7.4 Implémentation avec Rapp & RappServer	99
7.5 Discussion et Perspectives	111
8 Conclusion	113
8.1 Bilan	113
8.2 Perspective : vers une QoS prévisible pour les automobiles connectés	114
Liste des figures	117
Liste des tableaux	119
Liste des acronymes	121
Références	125

Chapitre 1

Introduction

Sommaire

Contexte et objectifs de recherche	1
Une nouvelle ère pour les réseaux de télécommunication	1
Vers des réseaux d'opérateur cognitifs	2
Émergence de la Data Science au service des réseaux	2
Positionnement de la thèse	3
Contributions de la thèse	3
Structure du document	7

Ce chapitre introductif présente d'abord le contexte, les objectifs et le résumé des contributions de la thèse. Nous y décrivons aussi la structure de ce document.

Contexte et objectifs de recherche

Pour présenter le contexte et les objectifs de la thèse, nous commençons par un rappel historique, suivi d'une argumentation sur les nouveaux besoins de gestion cognitive pour les réseaux et l'importance de la science des données dans ce cadre. Ensuite, nous présentons le positionnement de la thèse, avant d'aborder ses contributions.

Une nouvelle ère pour les réseaux de télécommunication

Le monde a connu au cours des dernières années un agrandissement considérable des besoins en télécommunication. Ceci est expliqué par une large démocratisation des smartphones et des objets connectés et par le foisonnement des services multimédias. Tout cela impose aux opérateurs et aux fournisseurs de contenus multimédias de moderniser les technologies qui soutiennent leurs réseaux, pour continuer à garantir un certain niveau de qualité de service. Sur les réseaux mobiles, ces évolutions ont donné lieu à une succession de générations de technologie. On est ainsi passé du [Global System for Mobile Communications \(GSM\)](#), 2^{ème} génération de réseau mobile, à la [Long Term Evolution \(LTE\)](#), 4^{ème} génération, entre les années 90 et les années 2010. Aujourd'hui le monde prépare la 5G (5^{ème} génération de réseau mobile). Elle comporte à l'heure actuelle plusieurs exigences en termes de meilleur débit, de moindre latence, de garantie de bonne

expérience utilisateur, de support d'un nombre massif de connections, de support d'une grande diversité de types de terminaux et d'efficacité énergétique. Plusieurs technologies accompagnent la 5G telles que la **Network Function Virtualization (NFV)** qui permet de faire face au défi de performance, en termes de « scalabilité »¹, de réactivité et d'efficacité énergétique. La **NFV** consiste en l'utilisation d'équipements informatiques standards, et non plus d'équipements dédiés très coûteux, pour assurer les fonctions du réseau. Cela revient à amener l'intelligence des réseaux au niveau logiciel et ainsi s'appuyer sur les techniques de virtualisation et de gestion de « Cloud » et de « Data center » pour assurer les services réseau avec une certaine garantie de qualité.

Vers des réseaux d'opérateur cognitifs

Parmi les moyens étudiés par les chercheurs pour garantir la qualité de service dans ce contexte évolutif des réseaux de télécommunication, on note les *réseaux cognitifs*. **THOMAS et collab.** [2007] les définissent comme des réseaux avec un processus cognitif capable de percevoir les conditions actuelles du réseau, de planifier, de décider, d'agir sur ces conditions, d'apprendre des conséquences de ses actions tout en suivant les objectifs de bout en bout. Cette boucle cognitive sonde l'environnement réseau, planifie les actions en fonction des données collectées par des capteurs et en fonction des politiques établies pour le réseau, décide du scénario correspondant le mieux à son objectif de bout en bout à l'aide d'un moteur de raisonnement, et finalement agit sur le scénario choisi. Le système apprend donc du passé (situations, plans, décisions, actions) et utilise ces connaissances pour améliorer les décisions à l'avenir.

Les réseaux cognitifs sont donc l'application aux réseaux de l'*intelligence artificielle* définie par **RUSSELL et NORVIG** [2010] comme « l'étude des agents qui reçoivent des perceptions de l'environnement et réalisent des actions, chacun de ces agents implémentant une fonction qui relie les séquences de perception aux actions ».

L'intelligence artificielle trouve en son centre la science des données, plus connue de nos jours sous son anglicisme *Data Science* que nous utiliserons tout au long du document.

Émergence de la Data Science au service des réseaux

La Data Science est un terme qui s'est démocratisé au cours de la dernière décennie. Elle est la preuve de la prise de conscience de l'importance et du potentiel que comportent les données dans tous les domaines métier. Le développement de la Data Science en tant que discipline a été facilité par les progrès récents dans la capacité à apprendre sur les données grâce à la miniaturisation des processeurs, à l'évolution des technologies de stockage, au développement de nouvelles architectures de traitement et à la sophistication des algorithmes d'apprentissage. C'est une discipline qui inclut divers savoir-faire autour des données, tels que leur collecte, leur gestion, leur modélisation, leur apprentissage, leur visualisation et leur intégration à un système d'intelligence artificielle.

La composante de la Data Science qui nous intéresse le plus dans le contexte de ce manuscrit est le *Machine Learning*, en d'autres termes, l'apprentissage automatique. Le Machine Learning consiste à apprendre des données, c'est-à-dire en extraire une connaissance qui se traduit sous la forme d'un modèle, qu'on utilise pour comprendre, classifier ou prédire une mesure, un phénomène ou un événement.

1. la « scalabilité » est la capacité d'un dispositif informatique à s'adapter au rythme de la demande, à passer à l'échelle

Cette thèse utilise largement ces notions liées à la Data Science qui se trouvent utiles pour la gestion des réseaux notamment pour bâtir des réseaux d'opérateur plus cognitifs. Nous montrons dans les paragraphes suivants - de positionnement et de résumé des contributions - comment la thèse s'insère au milieu de ces différentes notions.

Positionnement de la thèse

La multiplication des terminaux connectés et l'exigence de connectivité accrue imposée par la 5G poussent les réseaux d'opérateur à évoluer vers la virtualisation. Cette dernière permet de faire face au défi de performance, sans augmenter massivement les investissements et les coûts d'exploitation. Cet avènement coïncide dans le temps avec celui du développement de la Data Science, où les capacités d'exploiter les données se sont largement développées. Dans ce contexte évolutif, la Data Science crée une opportunité d'améliorer l'exploitation des réseaux de télécommunication.

Au sein d'Orange, le projet de recherche nommé [One Cognitive Operation \(COGITO\)](#), auquel la thèse est rattachée, vise à construire les moyens d'une gestion cognitive pour les réseaux et services de la 5G, grâce notamment à la Data Science. De même, le projet européen [Cognitive Networks \(CogNet\)](#), auquel j'ai pu participer au cours de la thèse, a proposé des architectures de réseaux permettant d'avoir un système intelligent de connaissance et de gestion d'un réseau 5G.

Ainsi, cette thèse est née du besoin de montrer le potentiel intérêt de la Data Science pour la gestion des réseaux. Elle s'insère au sein de ces deux projets de recherche. Elle porte sur des cas d'utilisation concrets de la Data Science pour le réseau et sur la proposition de nouveaux outils d'analyse et d'apprentissage de données, mais aussi de déploiement de la Data Science en tant que service.

Contributions de la thèse

Les contributions de la thèse sont de deux ordres : algorithmique et logiciel. Toutes les contributions entrent dans le cadre de l'insertion de la Data Science dans le réseau, soit par la proposition de cas concret d'utilisation, soit par la proposition d'outils facilitant l'analyse ou le déploiement de moyens de traitement de données dans le réseau.

Les contributions algorithmiques sont **(i)** la prédiction instantanée du débit du réseau mobile et **(ii)** l'automatisation de l'analyse des résultats de test de performance d'un réseau virtualisé.

(i) Prédiction instantanée du débit du réseau mobile Le débit d'un client sur le réseau cellulaire peut varier significativement au cours du temps, avec un effet non négligeable sur la qualité d'expérience. Les fournisseurs de contenu abordent cette problématique en utilisant différentes représentations d'un même contenu (par exemple, la résolution d'image, la résolution vidéo, etc.) et basculent entre celles-ci en fonction des mesures de débit collectées lors de la connexion. S'il était possible de connaître le débit accessible avant l'établissement d'une connexion, les fournisseurs de contenu pourraient choisir la représentation la plus appropriée dès le début. Nous avons réalisé une campagne de mesure impliquant 60 utilisateurs connectés au réseau d'Orange en production en France. Grâce aux données collectées, nous avons montré la capacité d'exploiter les mesures sur le réseau de l'opérateur et sur le terminal mobile pour prédire avec précision le débit accessible avant d'établir une connexion. Cela a permis d'introduire des stratégies de coopération

entre le client, l'opérateur, et le fournisseur de contenu pour parvenir à une solution de livraison cognitive de données multimédia.

(ii) Automatisation de l'analyse des résultats de test de performance d'un réseau virtualisé

La virtualisation des réseaux apporte aux opérateurs une flexibilité qui va permettre de réduire leurs coûts d'exploitation. En effet, de nombreuses fonctions du réseau peuvent être réduites à leur simple aspect logiciel de manière à pouvoir les déployer au sein de Data Centers sous forme de machines virtuelles. Cela nécessite donc de pouvoir activer et désactiver à la demande des machines virtuelles, afin de gérer de manière dynamique le déploiement et les ressources du réseau virtualisé. Afin de garantir un certain niveau de qualité de service, ces opérations imposent à l'opérateur de pouvoir tester chaque déploiement à la demande aussi. Des outils de test tels que *Yardstick* ([YARDSTICK \[2016a\]](#)) sont aujourd'hui en cours de développement, notamment au sein du projet « open source » *OPNFV* ([OPNFV \[2014\]](#)). Notre contribution a porté sur la fourniture de moyen d'analyser automatiquement des données de test de performance de déploiement issues de *Yardstick*. Il s'agit d'algorithmes capables d'effectuer, premièrement un « benchmarking » entre différents déploiement répertoriés dans les fichiers journaux, et en second lieu, une analyse comparative de l'effet des paramètres qui ont potentiellement influé sur la performance du réseau virtuel. Cette contribution a d'ailleurs été partagée avec la communauté du projet *OPNFV* sous forme de code.

Sur les aspects logiciels, les contributions sont constituées **(iii)** d'une part, d'un outil générique d'analyse et de visualisation de corrélations et **(iv)** d'autre part, d'un outil de déploiement automatique et efficace d'application de Data Science dans un réseau.

(iii) Linkspotter : outil d'analyse et de visualisation de corrélations L'exploration des relations entre les variables est une étape importante dans le processus d'exploration de données. Il permet à l'analyste de mieux comprendre les phénomènes décrits dans les jeux de données avant toute modélisation. Afin de faciliter l'exploration des données, j'ai développé et utilisé le package R nommé *Linkspotter* au cours de la thèse. Il offre plusieurs fonctionnalités permettant d'analyser et de visualiser de manière exhaustive en utilisant un graphe toutes les corrélations bi-variées d'un fichier de données. Outre la nouveauté du type de visualisation de corrélation permise, *Linkspotter* introduit un nouveau coefficient de corrélation nommé « Maximal Normalized Mutual Information (MaxNMI) ». Il se base sur une discrétisation supervisée des variables continues. Son intérêt est d'être facile à calculer et à comparer quel que soit le type de couple de variables (continue-catégorielle, continue-continue, catégorielle-catégorielle).

(iv) Rapp : outil de déploiement Data Science en tant que service La Data Science et la gestion de « Cloud » sont deux métiers distincts. D'un côté les Data Scientists, de part les outils qui composent leur écosystème, ont la capacité de développer des applications de Data Science qui sont le fruit de leurs analyses, leurs modélisations et leurs visualisation de données. Il peut s'agir d'un modèle prédictif à déployer sous forme d'une interface de programmation applicative (« API ») ou d'une interface de visualisation de données. Cependant, il manque toujours aujourd'hui aux Data Scientists des outils pour mettre en production ces applications qu'ils produisent au sein d'une infrastructure Cloud qui les met à disposition de leur utilisateurs finaux. Le package R nommé *Rapp* et son concept, que j'ai développés et utilisés au cours de cette thèse, constituent un pont entre les deux métiers cités. *Rapp* offre des outils permettant d'encapsuler une application développée sous R dans un « contai-

ner » et de la publier dans un Cloud au travers d'une API réceptrice. Le principe de « containerisation » utilisé, facilité par des technologies existantes telles que *Docker* ([MERKEL \[2014\]](#)), permet l'isolation de l'application de Data Science, qui va bénéficier au sein de son container de toutes les dépendances dont elle a besoin. Elle permet aussi au gestionnaire de Cloud de mieux gérer l'application, du point de vue de la sécurité et de son accessibilité.

Publications

Les travaux de la thèse ont donné lieu plusieurs publications et communications sous diverses formes. Les majeures sont : 1 article de journal, 3 articles de conférence, 2 dépôts de brevets dont l'un est encore à l'étude, 2 présentations acceptées à un séminaire de programmeurs (*Rencontres R*) sur deux années successives, 2 publications de code source acceptées respectivement dans le [Comprehensive R Archive Network \(CRAN\)](#), dépôt officiel des packages du langage R, dans le projet open source OPNFV, et enfin 2 contributions à la rédaction de deux livrables du projet CogNet.

Article de journal

- « Predicting File Downloading Time From Scratch : Large-Scale Analysis of Statistical Approaches on Cellular Network », écrite par Alassane Samba, Yann Busnel, Alberto Blanc, Philippe Dooze et Gwendal Simon, en cours d'examen par l'éditeur de *Elsevier Computer Networks*, suite à la réalisation de sa demande de révision mineure.

Articles de conférence

- « Instantaneous Throughput Prediction in Cellular Networks : Which Information Is Needed ? » écrite par Alassane Samba, Yann Busnel, Alberto Blanc, Philippe Dooze et Gwendal Simon, présenté par Alassane Samba à la conférence internationale *IFIP/IEEE International Symposium on Integrated Network Management (IM)* à Lisbonne, Portugal, en mai 2017.
- « CogNitive 5G Networks : Comprehensive Operator Use Cases with Machine Learning for Management Operations » écrite par Imen Grida Ben Yahia, Jaafar Bendriss, Alassane Samba et Philippe Dooze, présentée par Alassane Samba à la conférence internationale *20th ICIN Conference Innovations in Clouds, Internet and Networks (ICIN)* à Paris, France, en mars 2017.
- « Throughput Prediction in Cellular Networks : Experiments and Preliminary Results », écrite par Alassane Samba, Yann Busnel, Alberto Blanc, Philippe Dooze, Gwendal Simon, présentée par Alassane Samba aux *Rencontres Francophones sur la Conception de Protocoles, l'Évaluation de Performance et l'Expérimentation des Réseaux de Communication (CoRes)* à Bayonne, France, en mai 2016.

Brevets

- « Procédé et dispositif d'actualisation d'un modèle prédictif d'une variable relative à un terminal mobile », inventeurs : Philippe Dooze, Alassane Samba, Eric Debeau, numéro INPI : FR1660991, officiellement déposé le 14/11/2016.

- « Sélection de l'interface radio d'un smartphone selon des critères de QoS et de consommation électrique », inventeurs : Joel Penhoat, William Diego, Alassane Samba, demande de brevet soumise le 20/02/2017

Présentations à Rencontres R

- « Linkspotter : an R package for correlations analysis and visualization », présenté par Alassane Samba à *Rencontres R* en juin 2017², <https://hal.archives-ouvertes.fr/hal-01836428>
- « Construction et déploiement d'applications web basées sur R », présenté par Alassane Samba à *Rencontres R* en juillet 2018³, <https://hal.archives-ouvertes.fr/hal-01836400>

Codes sources

- « linkspotter : Bivariate Correlations Calculation and Visualization », package R créé et maintenu par Alassane Samba, publié en juillet 2018 dans le CRAN : <https://cran.r-project.org/web/packages/linkspotter/index.html>
- « TOM : Machine Learning-Based Test Results Analysis », première version du code ayant fait l'objet d'une contribution acceptée dans le projet open source Open Platform for NFV (OPNFV), en mars 2017 <https://github.com/opnfv/qtip/tree/master/contrib/TOM>

La publication du code source des outils Rapp & RappServer constituant également une de nos contributions logicielles est aujourd'hui à l'étude.

Contributions au projet européen CogNet

- « D3.2 – Feature and Structure modeling, Structured Input/Output, Unsupervised Learning and Domain Adaptation (Accompanying Document for the Engineering Release) », contributeurs : Tymoshenko, K., O. Uryupina, A. Moschitti, D. Bonadiman, I. Haponchyk, A. Passerini, F. Granelli, L. Xu, H. Assem, T. S. Buda, A. Samba, P. Dooze, Y. B. Ben Slimen, S. Allio, J. Jacques, B. Zhu, B. Ordozgoiti, S. Gómez Canaval, A. Mozo, M. Quartulli, A. Martin, N. Oses, J. Odriozola et I. Arruabarrena, lien : http://www.cognet.5g-ppp.eu/wp-content/uploads/2015/08/D3.2_FINAL.pdf, octobre 2016
- « D3.3 – Feature and Structure modeling, Structured Input/Output, Unsupervised Learning and Domain Adaptation (Accompanying Document for the Engineering Release) », contributeurs : Moschitti, A., A. Passerini, F. Granelli, O. Uryupina, I. Haponchyk, D. Bonadiman, A. Abad, L. Chen, H. Assem, T. S. Buda, B. Çağlayan, L. Xu, B. Ordozgoiti, J. R. Sánchez-couso, A. Mozo, H. Arregui, E. Loyo, I. Segall, M. Goldstein, Y. Ben Slimen, S. Allio, J. Jacques, M. Michel, S. Senecal, V. Grollemund, A. Samba et P. Dooze, lien : http://www.cognet.5g-ppp.eu/wp-content/uploads/2017/05/CogNet_D33_Final.pdf, avril 2017

2. Sixièmes Rencontres R : <http://angletr2017.com/programme.html>

3. Septièmes Rencontres R : <https://r2018-rennes.sciencesconf.org/details>

Ces contributions aux livrables D3.2 et D3.3 du projet ont été accompagnées de codes sources⁴. Nous avons également été *reviewer* sur le livrable D3.1 de ce projet.

Structure du document

Le document s'articule en trois parties. La première traite de l'état de l'art, d'abord sur l'analyse et l'apprentissage de données (Chapitre 2) et ensuite sur les réseaux de télécommunication, notamment cognitifs (Chapitre 3). La deuxième partie traite des contributions algorithmes à savoir le modèle de prédiction de débit (Chapitre 4) et le modèle d'analyse des données de test de réseau virtualisé (Chapitre 5). Enfin, la troisième partie traite des contributions logicielles à savoir les outils Linkspotter (Chapitre 6) et Rapp (Chapitre 7). A la fin du document, une conclusion générale des travaux est fournie (Chapitre 8). Chaque chapitre se résume de la manière suivante :

Le Chapitre 2 présente un état de l'art général dans le domaine de l'analyse et de l'apprentissage de données. J'y insiste sur les méthodes d'analyse de corrélation et d'apprentissage supervisé qui ont été utilisées au cours des travaux.

Le Chapitre 3 traite de l'état de l'art des réseaux cognitifs. Il décrit l'évolution des réseaux télécommunication d'un passé récent à aujourd'hui. Et expose les stratégies proposées dans la littérature pour intégrer la notion de Data Science dans la gestion des réseaux.

Le Chapitre 4 présente la première contribution algorithmique de la thèse correspondant à un cas d'utilisation de gestion cognitive du réseau, notamment cellulaire. Il s'agit de la prédiction instantanée du débit cellulaire pour les besoins des services adaptatifs. Un état de l'art spécifique à ce cas d'utilisation, la démarche expérimentale adoptée et les résultats de l'étude de la prédictibilité du débit en utilisant des données accessibles à différents niveau du réseau y sont présentés.

Le Chapitre 5 traite de la deuxième contribution algorithmique, à savoir l'automatisation de l'analyse des tests de performance d'un réseau virtualisé. Il comporte un état de l'art spécifique sur le sujet, une présentation de la démarche scientifique, une présentation de l'algorithme d'analyse statistique automatique proposé et l'évaluation de l'algorithme par son application à différents fichiers de données de test.

Le Chapitre 6 ouvre la partie concernant les contributions logicielles. Il traite de Linkspotter, un outil générique que j'ai développé, qui facilite l'analyse des corrélations entre les variables d'un fichier de données quel que soit leurs types. Un état de l'art y est proposé, d'abord sur les mesures de corrélation et ensuite sur les outils de visualisation de corrélation. Par la suite, sont présentés l'algorithme de discrétilisation **Best Equal-Frequency (BeEF)** et le coefficient **Maximal Normalized Mutual Information (MaxNMI)** implementés par Linkspotter, tous deux proposés dans le cadre de la thèse. Enfin, avant la conclusion du chapitre, une évaluation du **MaxNMI** est présentée en comparaison avec d'autres mesures de corrélation.

4. Codes sources du projet européen CogNet : <https://github.com/CogNet-5GPPP>, rubrique CSE non accessible publiquement à ce jour : <https://github.com/CogNet-5GPPP/WP3-CSE>

Le Chapitre 7 traite du déploiement des applications de Data Science, c'est-à-dire qui permettent d'analyser, de visualiser des données, d'effectuer un apprentissage automatique ou d'exécuter un modèle prédictif. Un état de l'art spécifique sur le sujet y est présenté. Ensuite l'architecture proposée y est décrite. Enfin, une implémentation sous la forme de l'outil nommé Rapp y est présentée.

Le Chapitre 8 conclut ce manuscrit. Il fait d'abord le bilan des contributions de la thèse et dégage des perspectives quant aux améliorations possibles et aux nouvelles problématiques de recherche suscitées.

Première partie

État de l'art

Chapitre 2

Analyse et apprentissage automatique de données

Sommaire

2.1 Représentation des données statistiques	12
2.1.1 Qu'est-ce qu'une donnée statistique?	12
2.1.2 Notion d'individu et de variable	12
2.1.3 Les types de variable	12
2.2 Apprentissage	13
2.2.1 Apprentissage supervisée	13
2.2.2 Apprentissage non supervisée	13
2.2.3 Apprentissage semi-supervisé	14
2.2.4 Apprentissage par renforcement	14
2.2.5 Apprentissage en ligne	14
2.3 Focus sur l'apprentissage supervisé	15
2.3.1 Quelques méthodes d'apprentissage supervisé utilisées	15
2.3.2 Évaluation d'un modèle issu d'un apprentissage supervisé	16
2.4 Outils et langages pour l'apprentissage de données	19
2.4.1 Le langage R	19
2.4.2 Les APIs	19
2.4.3 Visualisation de données	20

L'apprentissage automatique constitue aujourd'hui une discipline scientifique à part entière. Composante de la "Data Science", cette discipline constitue le cœur de l'intelligence artificielle présentée comme un phénomène qui va révolutionner l'économie mondiale. L'apprentissage automatique tire son existence des méthodes statistiques largement développées au cours des dernières décennies et des récents progrès du monde de l'informatique en capacité de calcul, notamment grâce à l'accroissement des performances des processeurs.

L'apprentissage automatique consiste en une extraction de connaissance à partir d'un ensemble de données. La connaissance extraite peut avoir deux finalités. La première est l'analyse et la compréhension d'un ou de phénomènes décrits par les données. La seconde est l'apprentissage d'une réalité expliquée par les données qui se traduit sous la

forme d'un modèle que l'on utilise par la suite pour effectuer de manière automatique une prédiction, une classification ou un regroupement.

Dans la suite de ce chapitre, nous revenons sur la notion de représentation de données, qui permettra de mieux caractériser cette dernière. Cela permet de présenter aisément la suite. D'abord, nous présentons les méthodes d'apprentissage classées par leur type déterminé selon leur finalité. Ensuite nous faisons un focus sur l'apprentissage supervisé qui a le plus servi au cours de nos travaux. Enfin, nous présentons les notions et outils informatiques permettant de mettre en œuvre un apprentissage automatique et d'utiliser ses résultats.

2.1 Représentation des données statistiques

2.1.1 Qu'est-ce qu'une donnée statistique ?

La donnée est la représentation numérique d'une information sous une forme qui permet d'en empiler plusieurs, de faciliter sa lecture, de faciliter l'enregistrement d'une nouvelle et de faciliter le traitement de l'ensemble. Il peut s'agir de la trace écrite d'un évènement, d'une description, d'un texte, d'une image, d'un son, etc. Ainsi, un ensemble de données peut prendre différentes formes et différents formats de fichier existent pour faciliter leurs échanges et leurs traitements.

2.1.2 Notion d'individu et de variable

Un *individu* ou *instance* au sens statistique est l'entité élémentaire du sujet d'étude. Par exemple, il peut s'agir d'un client, d'une connexion, d'un terminal, etc. On collecte sur chaque individu des informations appelés variables. Chaque variable traduit une caractéristique de l'individu. Des exemples de variable seraient la tranche d'âge d'un client, son type de forfait, sa quantité de données consommée, etc. La représentation des données sous la forme d'un tableau avec les individus en ligne et les variables en colonne est la plus courante.

2.1.3 Les types de variable

Une variable en statistique peut être des types suivants : quantitatif ou qualitatif.

Variable quantitative

Une variable quantitative, comme son nom l'indique, traduit une notion de quantité. Elle est par définition numérique. On peut en calculer un indicateur de tendance centrale, tel que la moyenne, et la médiane, un indicateur de dispersion, tel que la variance et l'écart-type. On distingue deux types de variables quantitatives : les quantitatives continues et les quantitatives discrètes. Cette différenciation intervient souvent dans les choix de leur modélisation et dans leur représentation graphique où des types de figures différents sont généralement utilisés. Les variables quantitatives continues sont celles dont les valeurs possibles (le support) sont dans l'ensemble des nombres réels \mathbb{R} . Par contre, le support des variables quantitatives discrètes est l'ensemble des entiers relatifs \mathbb{Z} .

Variable qualitative

Une variable qualitative, par ailleurs qualifiée de *catégorielle*, est une variable dont les valeurs prises sont un ensemble fini de cas décrivant des qualités ou des catégories. Un exemple peut être le type de terminal ou la couleur d'un objet.

Discrétisation

On est souvent appelé à discrétiser une variable quantitative. Discrétiser une variable quantitative consiste par abus de langage à la rendre catégorielle (i.e. qualitative) en définissant des bornes d'intervalles sur celle-ci. Ainsi chaque intervalle peut être considéré comme une catégorie de la nouvelle variable obtenue. Cette opération est souvent effectuée pour simplifier une représentation ou se projeter vers une forme de la variable adaptée au type de traitement voulu.

2.2 Apprentissage

L'apprentissage automatique, plus connu sous son appellation anglophone *Machine Learning*, est la démarche d'apprendre sur des données, grâce à des algorithmes, des modèles transposables à de nouveaux cas. Le but peut être d'établir un modèle de prédiction, de classification, de recommandation ou de regroupement. Selon l'objectif et l'approche, on peut classer les méthodes d'apprentissage automatique en plusieurs catégories. Les paragraphes suivants présentent chacune d'elles à savoir : l'apprentissage supervisé, non supervisé, semi-supervisé et par renforcement.

2.2.1 Apprentissage supervisé

Un apprentissage supervisé consiste à mettre en face à face un groupe de variables appelées *variables explicatives* ou *prédicteurs* et un autre en général constitué d'une seule variable, appelée *variable cible* ou *à prédire*. L'objectif, en effet, est d'apprendre le lien entre les variables explicatives et la variable à prédire grâce à des algorithmes d'optimisation et de traduire sous forme de modèle. Étant donné que c'est la famille de méthode qui a le plus été utilisée au cours de nos travaux, nous la détaillerons plus que les autres, en lui consacrant la section suivante [2.3](#) de ce chapitre.

2.2.2 Apprentissage non supervisé

Les différentes méthodes de « clustering » existant, aussi appelées « regroupement » ou « partitionnement » en français, partagent le même objectif. Il s'agit toujours de déterminer un regroupement des observations optimal selon des critères définis pour obtenir différentes classes d'individus. Ainsi, en pratique, un « clustering » peut servir à la recherche d'une typologie, ou d'une segmentation. Pour ce faire, on optimise le partitionnement selon des critères objectifs d'homogénéité au sein de chaque classe et d'hétérogénéité entre les classes. En d'autres termes, on cherche à regrouper au sein d'une même classe les individus les plus ressemblants et à obtenir les classes les plus distinctes possibles.

Le *K-means* [MACQUEEN et collab., 1967], utilisé sur nos travaux précédents la thèse [SAMBA et DOOZE, 2015] est un exemple simple d'apprentissage non supervisé. C'est plus précisément une méthode d'agrégation autour de centres mobiles. Son algorithme implémente le principe de ré-allocation dynamique des individus à des centres de classes,

eux-mêmes recalculés à chaque itération. Il s'agit de voir les données comme une représentation vectorielle des observations dans \mathbb{R}^p muni d'une métrique. C'est-à-dire que l'on peut voir les observations comme un nuage de points dans un espace de p dimensions. La métrique utilisée, souvent la distance euclidienne, permet de mesurer l'éloignement entre deux points du nuage. C'est une méthode itérative. : après une initialisation des centres consistant à tirer aléatoirement k individus, l'algorithme répète deux opérations jusqu'à atteindre une convergence : (*i*) chaque individu est affecté à la classe dont le centre est le plus proche au sens de la distance euclidienne qui est notre métrique ensuite (*ii*) on calcule les coordonnées des k centres des classes ainsi constituées.

Pour juger de la qualité d'un regroupement ou apprentissage non supervisé, il faut munir l'espace d'une métrique. La distance euclidienne est la plus courante. Deux types d'indicateur sont calculés à partir de la métrique. Il s'agit (*i*) de l'*inertie inter-classe* qui mesure l'éloignement entre les différents groupes obtenus et (*ii*) de l'*inertie intra-classe* qui mesure la proximité des individus appartenant au même groupe. Le choix du meilleur regroupement doit s'effectuer en respectant un compromis entre ces deux types d'indicateur. C'est la raison pour laquelle il existe des dizaines d'autres indicateurs proposés dans la littérature et calculés à partir de ces deux derniers. Les plus courants sont détaillés par [CHARRAD et collab. \[2014\]](#).

2.2.3 Apprentissage semi-supervisé

Ce sont les méthodes qui s'inspirent à la fois de l'apprentissage supervisé et du non supervisé pour s'adapter à certaine situation. En effet, l'apprentissage semi-supervisé est pertinent quand la variable cible n'est renseignée que pour une partie de l'échantillon. Comme détaillé pour les modèles présentés par [ZHU \[2006\]](#), le modèle d'apprentissage supervisée obtenu à partir du sous-ensemble où la variable cible est présente est combiné à une approche non supervisé effectuée sur le reste de l'échantillon. Cette combinaison renforce le modèle qui aurait été obtenu avec une approche supervisée seule sur le sous-ensemble comportant la variable cible.

2.2.4 Apprentissage par renforcement

L'apprentissage par renforcement [[SUTTON et BARTO, 1998](#)] est un cas particulier de l'apprentissage en ligne. Il s'agit d'une modélisation particulière d'un système comprenant un *agent autonome* et son *environnement*. L'agent mène des *actions* continuellement dans le temps au sein de cette environnement. Chaque action est menée au temps t de manière à maximiser une *récompense quantitative*. Celle-ci est renvoyée par l'environnement selon l'action menée et peut être positive ou négative. Ainsi, au fil des itérations, l'agent apprend de ses succès et de ses échecs et optimise ses actions en fonction.

2.2.5 Apprentissage en ligne

Les modèles d'apprentissage en ligne sont nés de deux besoins : (*i*) pouvoir apprendre sur un flux de données et non sur une base de données statique et (*ii*) produire un modèle qui change dans le temps avec les nouvelles données qui arrivent. [SHALEV-SHWARTZ et collab. \[2012\]](#) détaillent plusieurs types d'apprentissage en ligne.

2.3 Focus sur l'apprentissage supervisé

Nous nous arrêtons un instant sur l'apprentissage supervisé pour fournir plus de détails sur cette méthode. Ces détails permettront d'introduire l'essentiel des techniques d'apprentissage utilisées dans le cadre de la thèse que nous présenterons dans les chapitres de contribution. On distingue deux types d'apprentissage supervisé. La différence est au niveau du type de la variable à prédire. Si elle est qualitative, on parle de *classification*. Si elle est quantitative, on parle de *régression*.

2.3.1 Quelques méthodes d'apprentissage supervisé utilisées

Nous citons dans cette partie trois méthodes d'apprentissage supervisé qui ont servi aux travaux de thèse.

Modèle linéaire

L'approche d'un modèle linéaire est basée sur l'analyse statistique, elle vise à ajuster la prédiction en une équation linéaire $Y + \epsilon = \beta X$ où Y est la variable à prédire, X la matrice de prédicteurs, β les coefficients à estimer et ϵ l'erreur. Tout en vérifiant les hypothèses suivantes : (i) l'erreur suit une loi Gaussienne pour chaque individu, (ii) l'erreur pour un individu est indépendant de celle pour les autres et (iii) la variance de l'erreur est la même entre les individus. Cette approche utilise un algorithme d'optimisation pour estimer les coefficients β . Ainsi, l'équation obtenue permet pour tout nouvel individu d'effectuer une prédiction noté \hat{Y} . L'avantage d'un modèle linéaire est la simplicité et la rapidité. Par contre, il peut être moins performant quand on est en face de relations trop complexes entre variables explicatives et variables à expliquer ou en face de trop de valeurs extrêmes ou « outlier » .

PLS-GLM

Il s'agit d'un modèle linéaire généralisé (GLM) utilisant les moindres carrés partiels [GE-LADI et KOWALSKI, 1986]. Il aide à corriger quelques défauts du modèle linéaire tout en conservant ses avantages. Un GLM permet d'envisager d'autres types de relation entre la variable cible et les variables explicatives, c'est-à-dire en considérant plutôt l'équation $\log(Y + \epsilon) = \beta X$ plus adaptée à certain cas. L'utilisation des moindres carrés partiels permettent d'effectuer l'apprentissage même en présence de corrélations fortes entre les variables explicatives. On effectue l'apprentissage plutôt sur les variables latentes appelées composantes principales résultant d'une réduction de dimension préalablement effectuée. Le PLS-GLM est plus efficace qu'un modèle linéaire dans la plus part des cas réels, du fait que les interactions sont prises en compte et que des relations plus complexes peuvent être modélisées. Il conserve les avantages des modèles linéaires tels que la robustesse, la clarté des critères d'optimisation, la facilité et la rapidité d'apprentissage et de déploiement.

Random Forest

Introduit par BREIMAN [2001a], il s'agit d'une extension des arbres de décision ou de régression. Le principe de fonctionnement est le suivant : premièrement, un ensemble d'arbres est construit à partir d'un nombre réduit de prédicteurs choisis aléatoirement, et en second lieu, les prédictions issues de tous les arbres sont agrégées. Chaque arbre

est construit à partir d'un échantillon *bootstrap*, notion détaillée par **EFRON et TIBSHIRANI [1994]**, et d'un nombre restreint de prédicteurs choisi aléatoirement. L'ensemble des arbres de régression est appelé *forêt aléatoire* (random forest). La prédiction considérée est la moyenne des sorties de chaque arbre dans le cas d'une régression. Un modèle Random Forest est assez flexible pour capturer des relations complexes entre les prédicteurs et la variable à prédire. De plus, le Random Forest n'est pas sujet au sur-apprentissage et les valeurs aberrantes n'impactent pas considérablement sa performance.

Réseaux neuronaux artificiels (NN)

Nous avons utilisé un design particulier de NN, appelé Feed-Forward Neural Networks [**BEBIS et GEORGOPOULOS, 1994; VENABLES et RIPLEY, 2013**], qui contient une unique couche cachée. Le principal avantage de cette technique d'apprentissage est sa capacité à s'adapter à des corrélations complexes entre la variable à expliquer et les variables explicatives. De plus, la conception basée sur une seule couche cachée conduit à des algorithmes faciles à coder et rapide à exécuter. Cependant, les NN sont plus sensibles au sur-apprentissage que les autres algorithmes utilisés.

2.3.2 Évaluation d'un modèle issu d'un apprentissage supervisé

Il existe divers moyens d'évaluer la qualité d'un modèle d'apprentissage supervisé. Nous en présentons d'abord les métriques les plus utilisées selon la catégorie d'apprentissage supervisé (classification ou régression). Ensuite, nous présentons la démarche de validation croisée qui utilise ces métriques pour évaluer le sur-apprentissage.

Mesures de qualité de Classification supervisée

Les mesures de qualité de classification permettent de juger de la confiance qu'on peut avoir par rapport au résultat obtenu d'un modèle produit par l'apprentissage. Cela donne aussi des éléments objectifs pour comparer plusieurs modèles.

Pour une classification, la base de l'évaluation est la *matrice de confusion*. Elle comporte toutes les notions de base qui permettent de bâtir les critères d'évaluation. C'est le tableau qui croise les classes prédictes et les classes observées. Considérant, un ensemble de données composé de N instances, on sait pour chaque instance quelle est la classe observée et quelle est la classe prédictée. La matrice de confusion permet de croiser les classes prédictes et les classes observées sous forme d'un tableau de fréquences. Pour étayer notre explication, prenons l'exemple d'une classification binaire. On considère 100 tests de débit avec l'issue « fort » pour 80 cas et l'issue « faible » pour les 20 restant. Un modèle de classification dans ce cas pourrait consister en un algorithme qui en fonction de données de contexte d'une connexion mises en entrée, va la désigner « de fort débit » ou « de faible débit ». Supposons qu'on évalue ce modèle en utilisant les 100 tests et que 70 des 80 cas de débit fort et 5 des 20 cas de débit faible sont classés « de fort débit ». Le reste est classé « de faible débit ». Dans ce cas où la classification est binaire, on dit qu'il y a 70 *vrais positifs (VP)*, 10 *faux négatifs (FN)*, 5 *faux positifs (FP)* et 15 *vrais négatifs (VN)*. La matrice de confusion est la suivante.

Ces notions sont la base de plusieurs indicateurs tels que la *précision*, le *rappel*, la *sensibilité*, la *spécificité*, etc. qui permettent de bâtir des critères de jugement et de comparaison des algorithmes de classification.

TABLEAU 2.1 – Example de matrice de confusion

		Débit observé	
		faible	fort
débit prédit	faible	15	10
	fort	5	70

Mesures de qualité de régression

Mesurer la qualité d'une régression consiste à évaluer la distance globale entre les valeurs prédites et les valeurs observées. Pour les modèles de régression paramétrique, des indicateurs, comme Akaike Information Criterion (AIC) [PAN, 2001], utilisent la fonction de vraisemblance du modèle.

Critère d'information d'Akaike (AIC) L'AIC est une mesure de la performance basée sur la fonction de vraisemblance du modèle. Il pénalise les modèles avec un plus grand nombre de prédicteurs. Il est calculé comme suit :

$$AIC = 2k - 2\ln(L),$$

où L est la valeur maximale de la fonction de vraisemblance pour le modèle et k est le nombre de paramètres estimés dans le modèle.

Erreur quadratique moyenne (MSE : Mean Square Error) C'est la moyenne arithmétique des carrés des écarts entre les valeurs prédites et les valeurs observées. Cette valeur est à minimiser dans le cadre d'une régression simple ou multiple.

Soit un échantillon de n instances, y_i est la valeur observée de la variable cible pour chaque instance i , \hat{y}_i est la valeur prédite et \bar{y} est la moyenne empirique de la variable cible. La MSE est calculée de la manière suivante :

$$MSE = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2$$

Erreur-type (RMSE) C'est la racine carrée de l'erreur quadratique moyenne définie ci-dessus. La RMSE se calcule par :

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2}$$

Erreur absolue moyenne (MAE pour Mean Absolute Error) Il s'agit de la moyenne arithmétique des valeurs absolues des écarts entre les valeurs prédites et les valeurs observées.

$$MAE = \frac{1}{n} \sum_{i=1}^n |\hat{y}_i - y_i|$$

Erreur absolue moyenne en pourcentage (Mean Absolute Percentage Error, alias MAPE) C'est la moyenne des écarts en valeur absolue par rapport aux valeurs observées. C'est donc un pourcentage et par conséquent un indicateur pratique de comparaison. Son inconvénient est qu'il ne peut s'appliquer qu'à des valeurs strictement positives.

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^n \left| \frac{\hat{y}_i - y_i}{y_i} \right|$$

Erreur absolue médian en pourcentage (Median Absolute Percentage Error, alias MedAPE) C'est la médiane des écarts en valeur absolue par rapport aux valeurs observées. Il est légèrement différent du MAPE du fait qu'on considère la médiane plutôt que la moyenne. De même, elle ne peut s'appliquer qu'à des valeurs strictement positives.

$$\text{MedAPE} = \text{median}_{1 \leq i \leq n} \left(\left| \frac{\hat{y}_i - y_i}{y_i} \right| \right)$$

Coefficient de détermination R² Le coefficient de détermination, R², représente la proportion de la variance de la variable cible expliquée par les prédicteurs.

Il est calculé de la manière suivante :

$$R^2 = 1 - \frac{\sum_{i=1}^n (\hat{y}_i - y_i)^2}{\sum_{i=1}^n (\bar{y} - y_i)^2}$$

Validation croisée

La validation croisée sert à évaluer la performance d'un modèle en situation réelle, c'est-à-dire quand il est déployé sur de nouvelles instances. En effet, il peut arriver qu'un modèle soit trop spécifique aux instances qui ont servi à le construire. C'est ce qu'on appelle le *sur-apprentissage*. Pour vérifier qu'un modèle ne tombe pas dans ce cas, les critères d'évaluation décrits ci-dessus sont plutôt utilisés sur des sous-ensembles d'instances nouvelles, c'est-à-dire différentes de ceux qui ont servi dans la phase d'apprentissage. Cette démarche constitue le principe général des différentes méthodes de validation croisée qu'on trouve dans la littérature telles que les suivantes :

Validation par sous-ensembles « Apprentissage-Test » Le critère le plus évident pour estimer la performance d'un classificateur est sa précision sur les nouvelles instances. Le nombre de nouvelles instances est potentiellement grand (sinon infini), donc une estimation doit être calculée sur un sous-ensemble de test. Ceci est communément appelé validation croisée. L'ensemble de données est divisé en un ensemble d'apprentissage et un ensemble de tests. L'ensemble d'apprentissage représentant généralement plus de 60% est utilisé pour apprendre un modèle tandis que l'ensemble de test est utilisé pour évaluer ce modèle sur des instances nouvelles. Un inconvénient de la validation basée sur un sous-ensemble de test est qu'il réduit les données consacrées à l'apprentissage d'un modèle. Un deuxième inconvénient est que la performance de la mesure du modèle avec les données de test est définie comme une partie de la variabilité selon le caractère aléatoire. En fait, les instances représentant le sous-ensemble de test sont sélectionnées au hasard à partir de l'ensemble de données initial. Donc, la validation basée sur un sous-ensemble de test n'est appropriée que lorsqu'on dispose d'un ensemble de données initial important.

« Leave-one-out » La validation croisée « leave-one-out » apporte des solutions aux inconvénients de validation basé sur un sous-ensemble de test. En fait, il s'agit de valider le modèle en apprenant n fois sur $n - 1$ instances de l'échantillon initial ; n est le nombre

initial d'instances. À chaque fois, l'instance exclue de l'apprentissage est utilisée pour tester le modèle obtenu sans elle. Et une mesure de performance agrégée est calculé à la fin à partir des n mesures de performance obtenues pour les n modèles. Un inconvénient d'une validation croisée « leave-one-out » est que l'apprentissage doit être effectué n fois. Ainsi, cette phase peut prendre beaucoup de temps quand n est trop grand.

« K-fold » La validation croisée « K-fold » peut être considérée comme une validation « leave $\frac{n}{K}$ out ». Il s'agit d'un compromis entre la validation en utilisant des sous-ensembles « Apprentissage-Test » qui dépend de l'échantillon de test sélectionné de manière aléatoire et de la validation croisée « leave-one-out » où on effectue l'apprentissage autant de fois qu'il y a d'instances. Il s'agit de diviser l'échantillon en K sous-ensemble, d'effectuer l'apprentissage sur chaque combinaison de $K-1$ sous-ensemble et de tester le modèle obtenu sur le $K^{\text{ème}}$ sous-ensemble laissé à la marge. Les mesures de qualité calculées pour les K modèles obtenus sont agrégées (moyennées la plupart du temps) pour calculer une mesure globale.

2.4 Outils et langages pour l'apprentissage de données

Dans l'informatique, le traitement de données est un domaine particulier. Certains langages de programmation comme Python ou Scala se spécialisent de plus en plus pour traiter ce type de besoin. D'autres, eux, ont été créés spécialement dans ce but. C'est le cas de R et SAS. Dans le paragraphe qui suit, nous allons présenter plus spécifiquement R que j'ai le plus utilisé au cours de la thèse.

2.4.1 Le langage R

Né en 1993 dans un contexte de recherche, R [R CORE TEAM, 2016] est un langage spécialisé dans la statistique et la science des données. Son code est libre d'utilisation. Il dispose d'un emplacement officiel où on peut retrouver des milliers de packages qui l'étendent. Ces derniers sont open source et peuvent être téléchargés à partir d'emplacement reconnus comme le CRAN¹, Bioconductor² et GitHub³. Les deux premiers sont les emplacements officiels des binaires et sources des packages open source, tandis que GitHub constituent l'emplacement courant où les développement des packages open source sont menés.

2.4.2 Les APIs

Une Application Programming Interface (API) est une solution informatique qui permet à des applications de communiquer entre elles et de s'échanger mutuellement des services. Peu utilisé dans le monde la Data Science, les APIs constitueront un outil incontournable pour les applications d'intelligence artificielle pour dialoguer avec leur environnement. En effet, la finalité de tout algorithme issu d'un apprentissage est d'être déployé en vue d'apporter une information lisible par l'homme ou directement utilisée par une autre application dans le but d'optimiser une décision ou un fonctionnement. La

1. CRAN (The Comprehensive R Archive Network) : <https://cran.r-project.org>
2. Bioconductor : <https://www.bioconductor.org>
3. GitHub : <https://github.com>

seconde alternative constitue ce que l'on appelle la « Closed Loop », notion très importante pour dans les réseaux cognitifs, que l'on détaillera dans le Chapitre 3. Ainsi, les APIs me semblent constituer un candidat incontournable pour réaliser proprement la phase de déploiement et interfaçer facilement des applications issues de la Data Science avec le reste le reste d'un système d'information. Des travaux récents en développement informatique ont donné naissance à des programmes facilitant la création d'APIs en utilisant des langages auxquels les Data Scientists sont familiers, notamment R. Nous avons étendu ces développements dans le cadre de cette thèse pour proposer des outils facilitant encore plus la phase de déploiement. Nous présentons ces résultats dans le Chapitre 7.

2.4.3 Visualisation de données

La visualisation constitue un pan important de la Data Science. Il s'agit de la phase qui permet de représenter les données de manière synthétique dans le but d'en faciliter la lecture, l'analyse et l'appréciation humaine. Dans de nombreux cas la finalité de l'analyse ou de l'apprentissage de données est d'apporter une vision compréhensible par l'homme d'un ou de phénomènes traduits par les données. La visualisation est utile par exemple dans un outil de supervision de réseau qui serait capable de prévenir une panne en affichant les détails alertant les agents de supervision qui seront ainsi capable de dépêcher un technicien pour une intervention. Cette exemple est typiquement un cas d'utilisation de ce que l'on appelle l'« Open Loop » pour les réseaux cognitifs, que nous détailleront dans le Chapitre 3.

Chapitre 3

Les réseaux d'opérateur d'aujourd'hui et de demain

Sommaire

3.1 Modèles et protocoles de communication modernes	22
3.1.1 Modèle OSI : approche conceptuelle	22
3.1.2 La suite TCP/IP : approche pratique	23
3.1.3 IP et routage	24
3.1.4 UDP	25
3.1.5 TCP	25
3.2 Réseau d'opérateur	26
3.2.1 Réseau d'accès	27
3.2.2 Réseau cœur	28
3.3 Réseau mobile	28
3.3.1 Qu'est ce qu'un réseau mobile ou cellulaire ?	29
3.3.2 Moyens de supervision de réseau mobile	29
3.4 Qualité de Service	30
3.4.1 Définition de la QoS	30
3.4.2 De la QoS à la QoE	31
3.4.3 Paramètres de QoS	32
3.5 Réseaux programmables, cognitifs et autonomiques	34
3.5.1 Virtualisation des fonctions réseau	34
3.5.2 Notion de réseaux cognitifs et autonomiques	35
3.5.3 CogNet : architecture de réseau cognitif et autonomique	36

Ce chapitre vise à introduire d'une manière simple des notions importantes permettant de comprendre le fonctionnement actuel des réseaux d'opérateur, ainsi que leur évolution. Ainsi, nous commencerons par les modèles conceptuels et les protocoles majeurs qui sous-tendent les réseaux de communication modernes. Ensuite, nous verrons une présentation générale d'un réseau d'opérateur et des entités qui le composent. Par la suite, nous nous arrêterons sur les réseaux mobiles pour entrer un peu plus dans les détails sur cette entité du réseau d'opérateur qui a particulièrement intéressé nos travaux.

notamment concernant le chapitre 4 sur la prédition de débit. Ensuite, nous présenterons le concept de qualité de service et ses composantes. Enfin, nous introduirons des concepts nouveaux dans les réseaux tels que la programmabilité et la cognitivité qui sont en train d'orienter l'évolution des réseau, notamment ceux des opérateurs.

3.1 Modèles et protocoles de communication modernes

Les systèmes de télécommunication n'ont pas cessé de se développer au cours de l'histoire de l'humanité. Des procédés datant d'avant notre ère sont aujourd'hui connus à l'instar des sons de tambours en Afrique [HERZOG, 1945] et des signaux de fumée en Amérique [CLARK, 1884] qui on permis aux populations de se transmettre des informations parfois complexes sur des distances relativement grandes. Depuis lors, la révolution industrielle a accru les besoins de télécommunication, et la science s'est développée dans le sens de sa facilitation. Entre la fin du 18^{ème} siècle et la moitié du 20^{ème} la télécommunication a connu de nombreuses avancées scientifiques. Dès le début de cette période, les premiers télégraphes sont nés. L'évolution s'est poursuivie avec l'invention du code Morse, la pose des premiers câbles télégraphiques sous-marins en cuivre, l'invention du téléphone, ainsi que le développement de la connaissance des ondes électromagnétiques. Ce dernier a permis la découverte des ondes radioélectriques suivie par les premières transmissions sans fil par ce moyen. Les premières tentatives fructueuses de transmission d'information grâce à la fibre optique ont aussi eu lieu à la fin de cette période. La Seconde Guerre et ses enjeux technologiques ont accéléré le développement des moyens de télécommunication, notamment grâce à l'invention des premiers ordinateurs pour déchiffrer les signaux ennemis. Après cette période les télécommunications ont commencé à prendre une forme numérique. Ensuite, dès les années 1960, les premiers réseaux informatiques sont apparus. A partir de ce moment, la télécommunication s'est de plus en plus confondu avec l'informatique. Ainsi, des protocoles et des standards de communication informatique ont été développés, permettant l'interopérabilité des différents éléments des réseaux informatiques. Ces normes, telles que le modèle [Open Systems Interconnection \(OSI\)](#) et la pile de protocoles TCP/IP, sont le socle de nos modes de télécommunication d'aujourd'hui, et notamment du réseau informatique mondial appelé « internet » qui a fini de transformer le monde en un village planétaire. Dans les paragraphes suivants nous verrons succinctement quelques unes des notions les plus importantes de ces standards et protocoles sous-jacents aux réseaux de télécommunication informatiques d'aujourd'hui. Il s'agit du modèle [OSI](#), le modèle TCP/IP, le protocole réseau nommé [Internet Protocol \(IP\)](#) et les protocoles de transport nommés [Transmission Internet Protocol \(TCP\)](#) et [User Datagram Protocol \(UDP\)](#).

3.1.1 Modèle OSI : approche conceptuelle

Le modèle [OSI](#) signifiant littéralement *modèle d'interconnexion des systèmes ouverts* est un modèle conceptuel qui caractérise et normalise les fonctions de communication d'un système informatique ou de télécommunication sans tenir compte de la structure et de la technologie sous-jacentes. Il est décrit par un standard [ISO](#) [1994]. Son objectif est l'interopérabilité de divers systèmes de communication avec des protocoles normalisés. Le modèle partitionne un système de communication en couches d'abstraction. La version originale du modèle a défini sept couches encapsulées de la plus basse à la plus hautes. Chaque couche résout un certain nombre de problèmes relatifs à la transmission de données, et fournit des services bien définis aux couches supérieures. Les couches

TABLEAU 3.1 – Modèle OSI

	Unité de données	Couche
Couches hautes	Donnée	7 - Application : Point d'accès aux services réseau
		6 - Présentation : Conversion et chiffrement des données
		5 - Session : Communication interhost
Couches matérielles	Segment	4 - Transport : Connexion de bout en bout et contrôle de flux
	Paquet	3 - Réseau : Parcours et adressage logique
	Trame	2 - Liaison : Adressage physique
	Bit	1 - Physique : Transmission binaire physique ou analogique

hautes sont plus proches de l'utilisateur et gèrent des données plus abstraites, en utilisant les services des couches basses qui mettent en forme ces données afin qu'elles puissent être émises sur un médium physique.

Les sept couches, illustrées sur le tableau 3.1, sont les suivantes dans un ordre ascendant :

1. la couche Physique : c'est la couche la plus basse gérant les transmission des signaux sous forme numérique ou analogique.
2. la couche Liaison : elle gère l'adressage physique (adresse [Media Access Control \(MAC\)](#)).
3. la couche Réseau : elle détermine le parcours des données et gère l'adressage logique (adresse [IP](#)).
4. la couche Transport : c'est la couche qui gère connexion de bout en bout. Elle assure la connectabilité et contrôle de flux. Les protocoles [TCP](#) et [UDP](#) que nous verrons plus en détail dans la suite appartiennent à cette couche..
5. la couche Session : elle gère les sessions entre les différentes applications.
6. la couche Présentation : elle gère le chiffrement et le déchiffrement des données et convertit les données machine en données exploitables par n'importe quelle autre machine.
7. la couche Application : c'est la couche la plus haute constituant le point d'accès aux services réseau.

3.1.2 La suite TCP/IP : approche pratique

La suite TCP/IP, encore appelée *suite des protocoles Internet* est une implémentation elliptique du modèle [OSI](#). C'est l'ensemble des protocoles utilisés aujourd'hui pour le transfert des données sur Internet. Elle est décrite par le document de référence de l'[Internet Engineering Task Force \(IETF\)](#) [BRADEN, 1989] dont la première version date d'octobre 1989. Les trois couches supérieures du modèle [OSI](#) (Application, Présentation et Session) sont considérées comme une seule couche Application dans TCP/IP. Comme TCP/IP n'a pas de couche Session unifiée sur laquelle les couches plus élevées peuvent s'appuyer, ces fonctions sont généralement remplies par chaque application ou des fois ignorées. Une vue simplifiée des couches TCP/IP, de la plus haute à la plus basse, est donc :

5. Couche Application

4. Couche Transport
3. Couche Réseau
2. Couche Liaison
1. Couche Physique.

Une autre approche du modèle TCP/IP consiste à mettre en avant un modèle en 2 couches, traduisant mieux le fonctionnement d'internet. Dans cette approche, Le protocole **IP** que nous verrons dans la suite (section 3.1.3) fait abstraction du réseau physique et les applications s'appuient directement sur la couche Transport (sur **TCP** ou **UDP** en pratique). On a donc la représentation suivante :

- Applications
- 2. Couche Transport (**TCP** ou **UDP**)
- 1. Couche Internet (**IP**)
- Accès réseau.

Dans les sous-sections qui suivent, nous présentons, à l'instar de **VALLET [2015]**, les protocoles les plus courants aujourd'hui pour remplir les rôles des couches Internet et Transport, à savoir respectivement l'**IP** d'une part, et d'autre part l'**UDP** et le **TCP**.

3.1.3 IP et routage

IP est un protocole permettant l'adressage et le routage des paquets. Une adresse **IP** permet d'identifier de manière unique une interface dans une machine connectée sur un réseau. Les paquets **IP** sont composés d'une entête, comportant principalement les adresses **IP** source et destination.

On trouve dans les réseaux des machines appelées *routeurs*, qui sont dotées de plusieurs interfaces et qui sont chargées de transmettre les paquets qu'elles reçoivent. Ces composants sont des points d'aiguillage qui forment le squelette du réseau. Chaque routeur dispose d'une table de routage lui permettant de savoir sur quelle interface il doit transmettre le paquet qu'il a reçu, afin que ce dernier puisse atteindre sa destination. Chaque paquet est alors routé de proche en proche jusqu'à la machine destinataire. Les échanges d'informations entre les routeurs sont codifiées par les protocoles de routage. Ces échanges permettent aux routeurs de construire une image du réseau, et donc leur table de routage. Le réseau internet est composé de groupes distincts de routeurs que l'on nomme **Autonomous System (AS)**. Chaque **AS** regroupe des routeurs placés sous la même autorité administrative, et est géré indépendamment. Le nombre d'**AS** ne cesse de croître, et en avril 2018, le nombre d'**AS** présents sur internet est d'environ 60700 selon les informations de **BATES et collab. [2018]**.

L'architecture du fonctionnement d'un routeur peut être vue comme composée de deux fonctions : le plan de contrôle et le plan de données. Le plan de contrôle consiste en l'élaboration de la topologie du réseau qui définit ce qu'il faut faire avec les paquets reçus par le routeur. En pratique, on peut aussi voir le plan de contrôle comme le fonctionnement du réseau qui transporte du trafic de signalisation et qui est responsable du routage. Le trafic de signalisation constitue l'ensemble des messages échangés entre les routeurs et les terminaux des clients, dans un but de contrôle et de gestion des transmissions, en dehors des paquets du client à transmettre effectivement. Ces messages de signalisation peuvent aussi servir a posteriori dans un but plus global de supervision du réseau. Le plan de données ou plan de commutation, quant à lui, fait référence au mouvement effectif des paquets sur les chemins définis grâce au plan de contrôle et aux processus de transfert de ces paquets d'une interface à une autre.

3.1.4 UDP

UDP [POSTEL, 1980] est un protocole léger de la couche Transport du modèle OSI. Il fournit des mécanismes de base permettant à une application d'envoyer des données à d'autres applications sur le réseau. Il utilise la notion de *port* pour distinguer les applications utilisant UDP sur une même machine. Un message UDP est associé à un port sur la machine source et à un port sur la machine destinataire. Lors de la réception, les paquets sont automatiquement redirigés par le système d'exploitation vers la bonne application en fonction du port destinataire utilisé. UDP est encapsulé dans les paquets IP. Tout comme ce dernier, UDP ne fournit aucun mécanisme supplémentaire pour assurer la fiabilité de la transmission. Les transmissions s'effectuent alors dans un mode déconnecté et non fiable. Ainsi, les messages UDP peuvent être perdus, dupliqués, ou même arriver à destination de façon désordonnée. De plus, si la source envoie des paquets trop rapidement pour que l'application réceptrice puisse tous les traiter, des pertes de paquets seront observées. L'intégrité des données transportées par un paquet est la seule garantie apportée par le protocole grâce à l'utilisation d'un *checksum*¹ intégré à l'entête UDP (tout comme IP). L'absence de mécanisme évolué de contrôle permet à UDP de proposer la transmission la plus légère possible. Une application utilisant le protocole UDP, mais souhaitant une gestion plus poussée des paquets devra nécessairement intégrer ce traitement à l'intérieur même de l'application. Les applications de type *streaming*, qui intègrent le contrôle d'erreur dans leurs algorithmes, sont de bons exemples de l'utilisation d'UDP. Les applications ne souhaitant pas gérer par elle-même le contrôle des paquets peuvent se tourner vers le protocole TCP, que nous abordons dans le paragraphe suivant.

3.1.5 TCP

De même que UDP, le protocole TCP [POSTEL, 1981] est un protocole de la couche Transport du modèle OSI. Il fournit aux applications le moyen de communiquer entre elles sur le réseau. Il repose aussi sur le protocole IP pour transférer ses paquets, et utilise également la notion de port pour différencier les différentes destinations possibles s'exécutant sur une même machine. Cependant, contrairement à UDP, le protocole TCP est un protocole de transport complexe, fiable, fonctionnant en mode connecté. Nous n'abordons ici que les grandes lignes de son fonctionnement.

La procédure de connexion suivie par TCP est composée de trois étapes connues sous le nom de « Three-way handshake ». La source commence par envoyer une demande de connexion vers le destinataire. Si le destinataire accepte la connexion, elle envoie un message vers la source, qui lui répond enfin pour confirmer l'ouverture de la connexion. Une fois la connexion établie, les applications peuvent communiquer. Le protocole utilise des messages d'acquittement appelés « ACK », envoyés par la machine destinataire à la réception d'un paquet, pour détecter les erreurs de transmissions. Si le message d'acquittement correspondant à un message transmis précédemment n'est pas reçu après une certaine période de temps, le message est ré-émis. De la même manière, à la réception, TCP procède à un contrôle des paquets : les messages dupliqués sont ignorés et les messages désordonnés sont réordonnés avant leur transfert à l'application destinataire.

Le protocole TCP adapte automatiquement le débit de transfert en procédant à un contrôle de congestion. Il utilise pour cela une fenêtre glissante de taille variable définissant le nombre maximal de paquets pouvant transiter sur le réseau avant qu'un paquet

1. La somme de contrôle ou *checksum* en anglais, parfois appelée « empreinte », est un nombre qu'on ajoute à un message à transmettre pour permettre au récepteur de vérifier que le message reçu est bien celui qui a été envoyé.

d'acquittement ne soit envoyé vers la source. La taille de la fenêtre varie différemment suivant deux phases successives : la phase **Slow Start (SS)** puis phase **Congestion Avoidance (CA)**. Nous présentons la succession des deux phases sur la figure 3.1. La phase SS correspond à une augmentation exponentielle du débit d'émission, jusqu'à ce qu'un certain seuil soit atteint, ou qu'une perte de paquet soit constatée. Si un paquet est perdu, la phase redémarre depuis le début. Par contre si aucune perte de paquet n'est constatée, l'algorithme passe alors dans la phase CA. La taille de la fenêtre est alors incrémentée progressivement, jusqu'à ce qu'une perte de paquet soit détectée. Dès qu'une perte est constatée, la taille de la fenêtre glissante est divisée par 2, puis l'algorithme recommence son augmentation progressive.

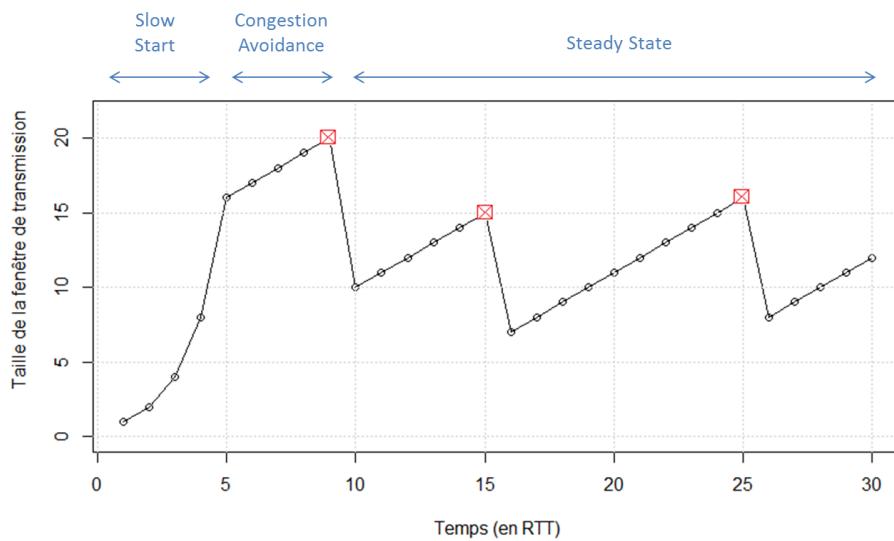


FIGURE 3.1 – Illustration de l'évolution de la taille de la fenêtre TCP à ses différentes phases

3.2 Réseau d'opérateur

Un opérateur est une organisation dont la mission est de fournir des services d'accès et d'utilisation des réseaux de télécommunication. Son rôle est de mettre en place, de gérer et d'exploiter des infrastructures constituant une bonne partie des réseaux de télécommunication, en respectant certaines normes internationales et locales. Les normes internationales existent pour assurer l'inter-connectivité entre les réseaux mis en place par différents acteurs et sont définies par des consortiums de industriels dans les télécommunications (opérateurs, constructeurs, fournisseurs de contenus, etc.). Les normes locales quant à elles sont régies par les états au travers d'organismes de régulation telles que l'[Autorité de Régulation des Communications Electroniques et des Postes \(ARCEP\)](#) en France et l'[Autorité de Régulation des Télécommunications et des Postes \(ARTP\)](#) au Sénégal. Elles régulent les télécommunications en fonction du contexte de chaque région, par exemple pour gérer la répartition du spectre radio.

Le réseau d'opérateur est un réseau géré par un opérateur de télécommunication. Il est généralement composé du réseau d'accès et du réseau cœur, comme illustré sur la figure 3.2. Dans les paragraphes suivants nous décrirons d'un point de vue général cette architecture.

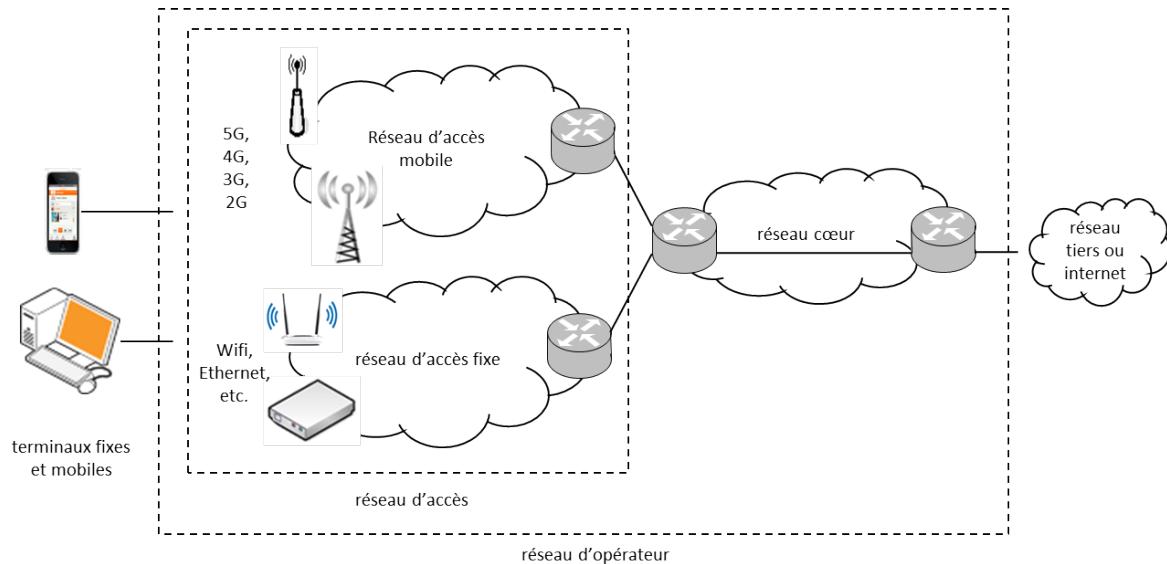


FIGURE 3.2 – Vue générale d'un réseau d'opérateur

3.2.1 Réseau d'accès

Le réseau d'accès d'un opérateur peut être composé d'un réseau d'accès cellulaire pour connecter les utilisateurs mobiles et d'un réseau d'accès fixe pour connecter notamment les réseaux résidentiels des clients. Nous définissons chacun d'eux dans les paragraphes suivants.

Réseau d'accès mobile

Le réseau d'accès mobile de l'opérateur, souvent appelé en anglais, [Radio Access Network \(RAN\)](#) est la partie radioélectrique, sans fil, du réseau opéré. C'est celle qui permet de connecter les différents appareils mobiles des clients quand ils sont hors des réseaux résidentiels tels que les réseaux Wifi. Le [RAN](#) inclut tous les équipements permettant d'acheminer les flux des utilisateurs mobiles jusqu'au réseau cœur, mais aussi de gérer leurs connectivité sur le réseau en toute mobilité. Nous détaillerons un peu plus le réseau mobile dans la section [3.3](#).

Réseau d'accès fixe

On appelle réseau d'accès fixe de l'opérateur, l'ensemble composé des réseaux résidentiels permettant d'offrir de la connectivité au client, grâce notamment au câble Ethernet, à la fibre optique et au Wifi, et de tous les équipements chargé acheminer leurs flux jusqu'au réseau cœur. C'est le réseau filaire dont les extrémités en cuivre ou en fibre optique sont tirées jusqu'au plus proche de l'utilisateur, de manière à ce qu'il puisse y connecter ses propres points d'accès tels qu'un modem Wifi. Pour le réseau d'accès fixe, les technologies généralement utilisées au niveau de la couche Physique du modèle [OSI](#) sont les celles de type [Digital Subscriber Line \(DSL\)](#) utilisant les lignes de cuivre et les celles de type [Fiber to the Home/Building/Desktop/etc. \(FTTx\)](#) utilisant la fibre optique tirée jusqu'au plus proche du client.

3.2.2 Réseau cœur

Le réseau cœur est l'autre partie du réseau géré par l'opérateur, au delà du réseau d'accès. On trouve dans le réseau cœur tous les équipements et les fonctions permettant à l'opérateur d'offrir des services à valeur ajoutée, de superviser son réseau, de facturer les services et de gérer la connexion avec des réseaux tiers. On peut en citer les suivants :

NMS

Le **Network Management System (NMS)** est un équipement que l'on retrouve à différent niveau du réseau pour superviser et contrôler différentes composantes physiques et logicielles de ce dernier. Il enregistre généralement les données provenant de différents éléments du réseau pour constituer des bases de données et des rapports de supervision destinés à l'administrateur du réseau.

Sondes

Une sonde est un équipement placé de manière à être traversé par les flux du réseau. Elle permet d'évaluer la **Qualité de Service (QoS)** réalisée, grâce aux métriques qu'elle calcule à partir de l'observation des flux.

IMS

L'**IP multimedia subsystem (IMS)** est une fonction réseau standardisée [ETSI, 2000] qui permet aux opérateurs de fournir des services multimédia fixes et mobiles. Son architecture permet, entre autres, d'utiliser la technologie *voix sur IP*, ainsi qu'une implémentation de **Session Initiation Protocol (SIP)**² fonctionnant sur les protocoles standards IP.

CDN

Un **Content Delivery Network (CDN)** ou *réseau de diffusion de contenu* est constitué de machines reliées en réseau à travers l'internet et qui coopèrent afin de mettre à disposition du contenu ou des données à des utilisateurs. Leur but est d'approcher encore plus des utilisateurs les contenus multimédias de fournisseurs distants afin d'accroître leur qualité d'expérience. Les **CDN** sont quelques fois en dehors du réseau d'opérateur, mais bénéficient de liaisons spéciales avec ce dernier pour garantir une bonne qualité de service.

3.3 Réseau mobile

Dans cette partie, nous entrons un peu plus dans le détail de la caractérisation de ce qu'on appelle un réseau mobile ou encore réseau cellulaire. La première contribution de la thèse portant sur le réseau mobile, nous verrons dans cette section un ensemble de notions utiles pour mieux comprendre l'exposé de la contribution du Chapitre 4.

2. SIP est un protocole standard ouvert de gestion de sessions souvent utilisé dans les télécommunications multimédia. Il est couramment utilisé pour la téléphonie par internet (voix sur IP).

3.3.1 Qu'est ce qu'un réseau mobile ou cellulaire ?

Le réseau cellulaire est une type de réseau d'accès qui est apparu dès la fin des années 1970. Son principe est d'assimiler les zones géographiques à couvrir à des cellules de plusieurs kilomètres de rayon. Ces cellules se superposent partiellement pour assurer une couverture complète du territoire cible. Au centre de chaque cellule se trouve une antenne jouant le rôle d'émetteur-récepteur pour communiquer avec les terminaux mobiles des utilisateurs grâce à des ondes radioélectriques communément qualifiées de « radio » tout court. L'interface radio utilise une bande fréquence généralement spécifique au pays dans lequel le réseau est opéré. Derrière chaque antenne se trouve une ce que l'on appelle une station de base. C'est l'équipement qui gère le fonctionnement de l'antenne. Les stations de base sont reliées à d'autres équipements du réseau d'accès radio qui les contrôlent et les gèrent conjointement pour par exemple assurer les « handover ». On parle de handover ou de transfert intercellulaire quand un utilisateur mobile en déplacement passe d'une cellule à une autre. Les équipements de contrôle du réseau d'accès radio déplacent des mécanismes permettant d'assurer ces transferts intercellulaires sans interruption de service.

Plusieurs générations de réseau d'accès radio se sont succédées. De légères modifications de l'architecture et l'utilisation de bandes de fréquence différentes ont permis d'améliorer au fil des générations la qualité de service. Ainsi, on a connu la 1G, la 2G, la 3G, la 4G et bientôt la 5G qui comporte déjà des exigences plus fortes de qualité de service et de résilience permettant d'accueillir notamment les nouveaux réseaux d'objets communément appelés [Internet of Things \(IoT\)](#).

3.3.2 Moyens de supervision de réseau mobile

La partie radio des réseaux cellulaires est celle qui fait sa spécificité. La qualité de service dans ce type de réseau est très dépendante du contexte de l'utilisateur et sa mobilité. Néanmoins, plusieurs mécanismes bien définis et étudiés au cours des dernières années, concernant la modulation de fréquence et le handover permettent d'assurer de bons niveaux de qualité de service. Ces mécanismes s'appuient sur une connaissance précise de la puissance et de la qualité de réception des ondes radio, mesurées au niveau du terminal utilisateur. Pour le réseau 4G, les indicateurs classiquement calculés pour ce faire sont le [Received Signal Strength Indicator \(RSSI\)](#), le [Reference Signal Received Power \(RSRP\)](#), le [Reference Signal Received Quality \(RSRQ\)](#), le [Signal-to-noise ratio \(SNR\)](#), le [Signal-to-interference-plus-noise ratio \(SINR\)](#) et le [Channel Quality Indicator \(CQI\)](#). Ils sont définis dans le paragraphe suivant. Des procédés comme le [Minimization of Drive Tests \(MDT\)](#) et les speed-tests que nous décrirons par la suite permettent d'effectuer et d'enregistrer ces mesures pour des besoins à plus long terme.

Indicateurs de puissance et de qualité de lien radio

Ici nous présentons quelques indicateurs de puissance et de qualité sur l'interface radio des réseaux 4G.

Received Signal Strength Indicator (RSSI) mesure de la puissance d'un signal reçu par le terminal client sur la bande de fréquence concerné. Il s'agit donc de la puissance mesurée en provenance de toutes les stations de base captées par le terminal.

Reference Signal Received Power (RSRP) mesure la puissance du signal de référence reçu par le terminal client en provenance d'une station de base.

Reference Signal Received Quality (RSRQ) est une mesure de qualité de la réception du signal obtenue d'un ratio entre le RSRP et le RSSI.

Signal-to-noise ratio (SNR) ou rapport signal sur bruit est un indicateur générique de la qualité de la transmission d'une information.

Signal-to-interference-plus-noise ratio (SINR) ou rapport signal sur interférence + bruit est aussi est un indicateur générique de la qualité de la transmission d'une information. Il se différencie en prenant en compte l'interférence.

Channel Quality Indicator (CQI) est aussi une mesure de la qualité de l'onde radio reçue par un téléphone mobile. C'est une fonction du SINR et est utilisé dans les réseaux 3G et 4G.

Drive testing

Le « drive testing » est une méthode de mesure et d'évaluation de la couverture, de la capacité et de la [QoS](#) d'un réseau cellulaire. La technique consiste à utiliser un véhicule motorisé contenant un équipement de mesure d'interface radio sur le réseau cellulaire capable de détecter et d'enregistrer une grande variété de paramètres des couches Physique et Application du modèle [OSI](#) relatifs à la qualité de service sur la zone géographique donnée. En mesurant ce qu'un abonné du réseau cellulaire pourrait rencontrer dans une zone spécifique, les opérateurs mobile peuvent apporter des changements dirigés sur leurs réseaux offrant une meilleure couverture et un meilleur service à leurs clients.

Mécanisme « Minimization of Drive Tests »

[MDT](#) fait partie des mécanismes de mesure standardisé par le consortium [3rd Generation Partnership Project \(3GPP\)](#) et décrits par l'[ETSI \[2016\]](#). Il vise à éviter à l'opérateur de réseau le recours aux drive tests, plus coûteux et moins écologiques, pour superviser le réseau. [MDT](#) permet de collecter des mesures concernant les terminaux des utilisateurs mobiles telles que les indicateurs de puissance et de qualité sur les interfaces radio. Son esprit est donc d'utiliser le terminal du client comme sonde au lieu de voitures spécifiques équipées pour les drive tests. Le principe de fonctionnement du [MDT](#) est de s'appuyer sur les messages de signalisation entre les terminaux mobiles et les stations de base dans le cadre du plan de contrôle pour collecter des informations sur l'interface radio et sur le contexte des terminaux mobile. Toutes les données sont capturées au niveau des équipements du réseau d'accès et envoyées vers des bases de données centralisées dans un but de supervision.

3.4 Qualité de Service

La qualité de service souvent abrégée [QoS](#) est un élément important à prendre en considération dans l'exploitation des réseaux et des services qu'ils supportent. On lui donne plusieurs définitions selon les cas.

3.4.1 Définition de la QoS

Les organismes de normalisation fournissent plusieurs définitions du terme [QoS](#) adaptées à différentes situations mais se rejoignant souvent sur plusieurs aspects. L'[IETF](#) définit la [QoS](#) comme « un ensemble d'exigences de service que le réseau doit satisfaire lors du

transport d'un flux» [CRAWLEY et collab., 1998]. L'Union Internationale des Télécommunications - Secteur de la normalisation des télécommunications (UIT-T), quant à elle, la définit comme « la totalité des caractéristiques d'un service de télécommunication qui influent sur sa capacité à satisfaire les besoins exprimés et implicites de l'utilisateur du service » [ITU-T, 2008]. Enfin, l'European Telecommunications Standards Institute (ETSI) la définit comme « l'effet collectif des performances de service qui déterminent le degré de satisfaction d'un utilisateur du service » [ETSI, 1994]. La première définition met l'accent sur les aspects de performance de réseaux, alors que les deux dernières incluent l'utilisateur dans la définition, en le plaçant parmi les agents qui détermine le jugement de la qualité.

On a tendance à utiliser le mot « QoS » dans des situations différentes où elle ne représente pas exactement la même chose. Néanmoins, un modèle de QoS [ITU-T, 2001] proposé par l'UIT-T unifie les définitions et clarifie les différents types de QoS que l'on peut observer. La figure 3.3, dont nous commentons les éléments dans les paragraphes suivants, illustre ce modèle.

La QoS intrinsèque La QoS intrinsèque correspond au type de QoS désigné par la définition de l'IETF. Il s'agit de la QoS émanant de la capacité du réseau.

La QoS offerte La QoS offerte est celle correspondant au niveau de service que le fournisseur met à disposition au client selon la situation.

La QoS réalisée Malgré une certaine volonté et la spécification d'un certain niveau de qualité à offrir, la QoS peut être borné à une certaine limite à cause de divers facteurs. Le niveau atteint malgré tout est appelée la QoS réalisée.

La QoS requise La QoS requise est celle qui répondrait au besoin du client, en l'occurrence celle établie dans son contrat de service ou Service-Level Agreement (SLA)³.

La QoS perçue La QoS perçue, comme son nom l'indique est celle que perçoit le client, au vu de la QoS réalisée, de la QoS requise et de divers facteurs extérieurs influençant son ressenti. Elle s'apparente ainsi à la Qualité d'Expérience (QoE).

3.4.2 De la QoS à la QoE

Les opérateur et les utilisateurs des réseaux ont de plus en plus tendance à privilégier comme moyen de jugement des services la qualité d'expérience (QoE) de l'utilisateur à la QoS. La QoS consiste en des mesures objectives, techniques, qualifiant les performances du réseau. La QoE quant à elle consiste en des mesures subjectives dépendant des utilisateurs et reflétant la satisfaction de ces derniers. Néanmoins, la QoE et QoS ont une forte corrélation dépendant des utilisateurs et des types de service. La plusieurs modèles permettant d'estimer la QoE à partir de la QoS existent et des architectures adéquates de mesure et d'estimation de la QoE, telles que le modèle ARCU [ETSI, 2014] issu des travaux du projet Celtic QuEEN sont standardisée.

3. Le SLA ou entente de niveau de service est un document qui définit le niveau de qualité et de prestation prescrit entre un fournisseur de service et un client. Il s'agit de clauses basées sur un contrat définissant les objectifs précis attendus et le niveau de service que souhaite obtenir un client de la part du prestataire et fixe les responsabilités.

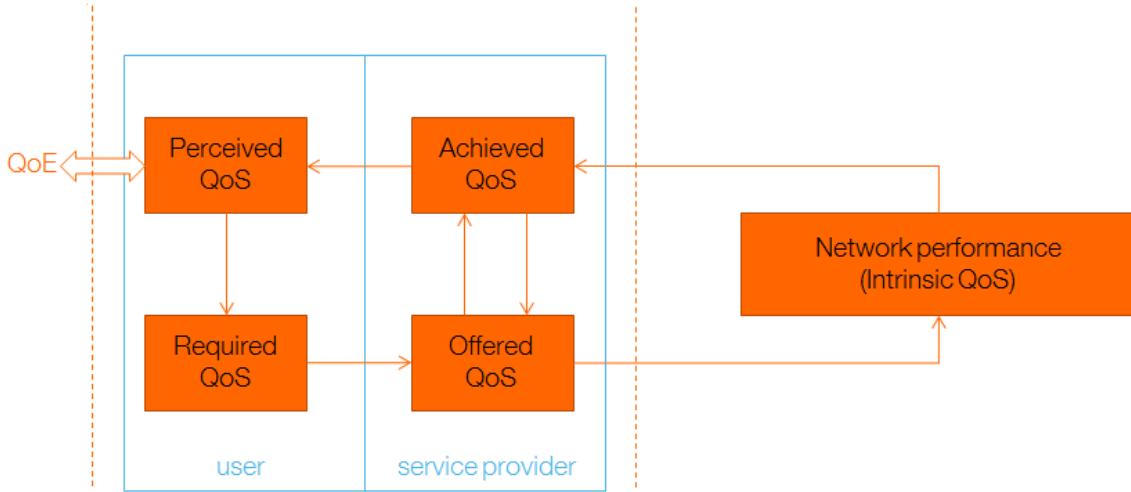


FIGURE 3.3 – Les différents types de QoS

3.4.3 Paramètres de QoS

La qualité de service se mesure grâce à des paramètres classiques tels que, la capacité d'un lien, la bande passante disponible, le débit, la latence, le taux perte de paquets et la gigue. Les trois premiers sont souvent confondus abusivement et nécessite d'être distingués.

Capacité d'un lien

Il s'agit de la notion dont dépendent nécessairement la bande passante disponible et le débit. Elle indique la quantité maximale de données pouvant transiter sur le lien par unité de temps. Il s'agit d'une propriété du lien, donc d'un paramètre de **QoS intrinsèque**, qui dépend aussi bien de ses propriétés physiques que de sa configuration. Le gestionnaire du lien peut délibérément décider de limiter cette capacité en jouant sur la configuration logicielle. On parlera alors de **QoS offerte**. Cette capacité est considérée comme fixe tant qu'aucune modification n'est opérée sur le lien.

Bandé passante disponible pour une liaison

Elle se rapporte à la capacité inutilisée d'un lien par unité de temps. Elle dépend de la capacité du lien mais aussi de la quantité de données qui y circule effectivement. Cette métrique est une variable dépendante du temps. Il est bon de différencier la bande passante d'un lien de la bande passante d'un chemin. Ainsi la bande passante d'un chemin est égale à la bande passante minimale de tous les liens constituant le chemin, et donc de son "goulot d'étranglement".

Débit effectif

C'est est la quantité de trafic maximale pouvant être transportée par un flux sur le lien considéré sur une période de mesure. C'est la raison pour laquelle on parle souvent de débit moyen sur une période. Cette notion est dépendante de plusieurs facteurs tels que le type de trafic transitant sur le lien (**UDP** ou **TCP** par exemple). S'il s'agit d'un trafic **TCP**, le débit peut aussi dépendre du nombre de connexions **TCP**, de la taille des buffers

TCP, de la congestion le long du chemin de retour pour l'envoi des paquets ACK ou de l'algorithme utilisé pour l'implémentation de **TCP**. Un débit calculé sur une connexion **TCP** qui a saturé le lien sur la période de mesure va donc refléter la quantité de données que l'on peut faire transiter par unité de temps sur le lien quand il se trouve dans les mêmes conditions que lors de la période de mesure.

Latence

La latence dans les réseaux se définit très simplement : il s'agit du temps total nécessaire à la transmission d'un paquet entre sa source et sa destination. On parle de **Round Trip Time (RTT)** pour qualifier le temps nécessaire à un aller-retour. Cette latence est tout d'abord occasionnée par un phénomène physique : les paquets sont transmis sous la forme d'un signal qui ne peut dépasser la vitesse de la lumière (300000 km/s dans le vide). Ainsi, si l'on considère une transmission entre les villes de Paris et de Los Angeles (distantes d'environ 9000 km), la latence de transmission sera nécessairement supérieure à 30 ms. Les communications par satellites sont particulièrement touchées par ce phénomène, en raison de la grande distance qui les sépare du sol (35000 km environ en position géostationnaire). Mais d'autres contraintes s'ajoutent à ce délai minimum de transmission, car les paquets doivent transiter par de nombreux équipements pour être acheminés jusqu'à leur destination. Chaque équipement ne dispose que d'une capacité de traitement finie qui détermine le nombre maximum de paquets pouvant être traités par unité de temps. Les paquets doivent patienter dans les files d'attente de chaque équipement avant d'être traités. Les conséquences de la latence sont particulièrement visibles pour tous les services qui fonctionnent en temps réel : transmission vidéo, communication vocale, applications interactives de type jeux-vidéo, etc. La gigue, qui représente les variations de latence au cours du temps, a également une forte influence sur ces applications.

Taux de perte de paquets

Il correspond au nombre de paquets qui n'arrivent pas correctement jusqu'à leurs destinations. Ce phénomène peut être principalement causé par deux facteurs. Le premier est le dépassement des capacités intrinsèques de traitement du réseau de transmission : si les équipements traversés deviennent trop congestionnés (remplissant leur files d'attente plus rapidement qu'ils ne peuvent les vider) des paquets peuvent se perdre, à défaut de pouvoir être stockés en file d'attente. Un deuxième facteur réside dans la corruption éventuelle des paquets lors de leur transmission. Les signaux transportant les paquets peuvent en effet subir des influences externes (interférences par exemple) causant une corruption du signal transmis. Les connexions sans fil (Wifi ou satellitaire) sont particulièrement concernées. La somme de contrôle employée par les protocoles de la pile TCP/IP est conçue pour détecter ce type de corruption. Lorsque l'équipement traversé calcule une somme de contrôle différente de celle attendue, le paquet est rejeté. Comme pour le délai, le taux de perte influence négativement la qualité de service des applications temps réel comme la communication audio ou vidéo. Bien qu'ils intègrent ce problème dans leur conception, des taux de pertes élevées dégradent la qualité perçue par l'utilisateur (dégradation de la qualité vidéo ou audio, coupures intempestives). Les connexions **TCP**, qui sont immunisées contre les pertes de paquets, subissent indirectement leur influence : le débit maximal atteignable avec **TCP** est en effet fortement influencé par ce paramètre, comme nous l'avons vu ci-dessus.

3.5 Réseaux programmables, cognitifs et autonomiques

A l'aube de la 5G, plusieurs technologies en développent aujourd'hui accompagnent l'évolution des réseaux d'opérateur. En particulier, on peut parler de la *programmabilité* permise par la virtualisation des fonctions réseau et de la *cognitivité* permise par les architectures de réseau autonome et la Data Science. La programmabilité et la cognitivité sont liées. La première facilite la mise en pratique la seconde. Enfin, la Data Science et l'intelligence artificielle donnent les moyens aux réseaux d'être cognitifs.

Nous verrons dans les paragraphes suivants la notion de virtualisation des fonctions réseau, la notion de réseau cognitif et autonome et une architecture récente pour arriver à l'implémentation de réseaux cognitifs et autonomes.

3.5.1 Virtualisation des fonctions réseau

Comme expliqué dans son livre blanc [CHIOSI et collab., 2012], le principe de la virtualisation des fonction réseau est né comme une initiative de plusieurs opérateurs de réseau pour réduire le recours à du matériel propriétaire qui était nécessaire jusque là quand il s'agit de déployer une fonction réseau. À partir de l'année 2012, le développement des technologies Cloud, notamment dans le domaine de la virtualisation, a été tel qu'elles ont été identifiées comme un moyen de rendre les réseaux plus programmables. En fait, l'idée est de dissocier la fonction réseau du matériel sur lequel elle tourne. Il consiste à réduire la fonction réseau à son aspect logiciel, afin qu'il puisse être exécuté comme un ou plusieurs machines virtuelles au dessus de n'importe quel serveur suffisamment provisionné, et notamment dans un data center.

Plusieurs cas d'utilisation de la virtualisation des fonctions réseau sont décrits dans les standards [ETSI, 2013b]. Par exemple, il est prévu la virtualisation de fonctions telles que les CDN, l'IMS, les stations de base ou le réseau cœur dans son intégralité.

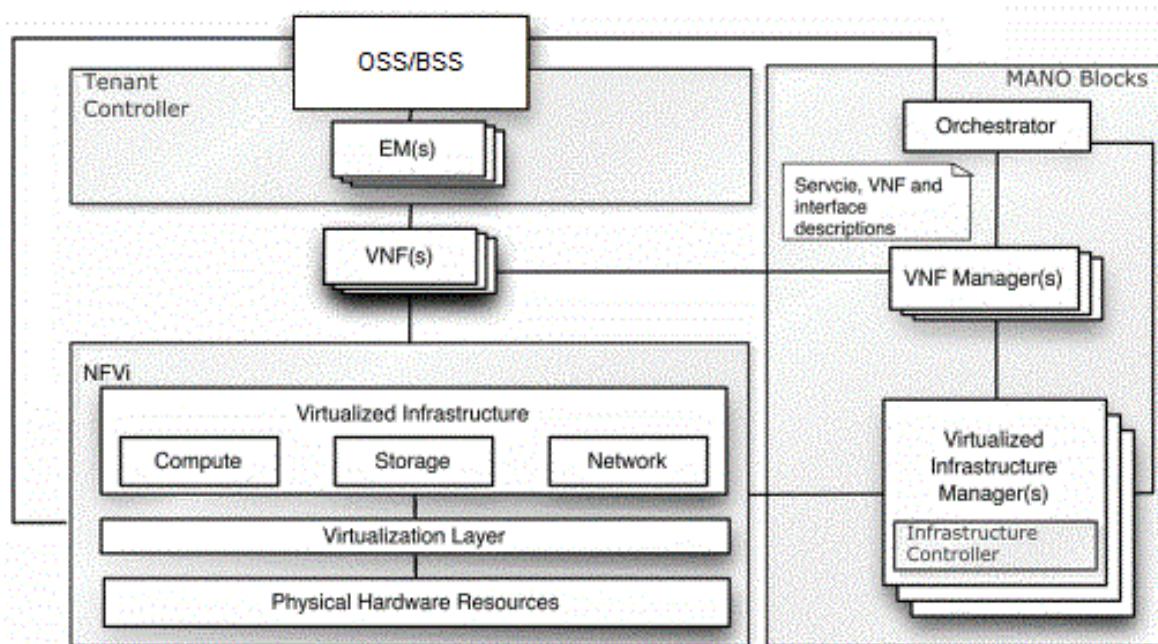


FIGURE 3.4 – Architecture NFV

Cette idée initiale a donné naissance à une architecture de référence [ETSI, 2013a]. Comme le montre la figure 3.4, cette architecture prévoit quatre blocs, à savoir le **Network**

Function Virtualization Infrastructure (NFVI), les blocs Management and Orchestration (MANO), les Virtual Network Function (VNF) et les Element Management System (EMS).

Infrastructure NFV

NFVI représente les ressources physiques et virtuelles sur lesquelles les instances de fonctions réseau doivent être déployées. C'est le socle du réseau virtuel.

Fonction réseau virtuelle

Une VNF est une fonction réseau, comme un Home Subscriber Server (HSS) ou un Evolved Packet Core (EPC), réduite à son simple aspect logiciel. La VNF est hébergée sur une ou différentes machines virtuelles ou conteneurs (Docker par exemple). Ainsi, une fonction réseau donnée peut nécessiter plusieurs VNF. Une VNF peut également être composites et nécessiter un ensemble de machines virtuelles ou de conteneurs.

Element Management System

L'EMS est d'une manière générale un ensemble de systèmes et d'applications destiné à gérer les éléments du réseau. Dans le cas de fonctions réseau virtualisées, il assure la gestion d'un ou de plusieurs VNF.

Blocs MANO

Les blocs MANO assurent la gestion continue du cycle de vie du service réseau, des VNF, de leurs composants, ainsi que leur infrastructure virtuelle et physique sous-jacente. Les blocs MANO englobent trois fonctions principales : Network Function Virtualization Orchestration (NFVO), Virtual Network Function Manager (VNFM) et Virtual Infrastructure Manager (VIM).

NFVO gère le cycle de vie des services réseau. Il comprend différentes fonctions, à savoir l'enregistrement d'un service réseau dans les inventaires et la vérification de tous les descripteurs du service, ainsi que ses informations associées. Le bloc NFVO instancie également le service réseau en suivant son modèle prédéfini. Le bloc NFVO a également la responsabilité du passage à l'échelle et des ressources des services. Les opérations typiques du bloc NFVO peuvent être résumées par la création, la suppression, l'interrogation et la mise à jour des fonctions réseau.

VNFM gère le cycle de vie des VNF. Il est responsable de leur instantiation, de leur passage à l'échelle et de leur mise à jour.

VIM quant à lui gère les ressources de calcul et de stockage.

3.5.2 Notion de réseaux cognitifs et autonomiques

La notion d'informatique autonome n'est pas nouvelle. En 2005 déjà, IBM et collab. [2005] publiait un livre blanc sur le sujet. Cependant, au cours des dernières années, avec l'avènement de la 5G, les opérateurs ont vu de plus en plus l'intérêt de transposer les notions d'informatique autonome à la gestion des réseaux. En effet, donner plus de réactivité et d'autonomie au réseau semble être une condition importante pour arriver à gérer les milliards d'objets connectés dont les cabinet d'études font la prévision pour la prochaine décennies. Dans le domaine des réseaux du futur, les termes *réseau autonome*, *réseau cognitif* et *réseau auto-géré* font référence au même concept. Il s'agit

d'apporter une intelligence à la gestion des réseaux qui permet de les rendre plus autonomes, voire autonomiques.

Différence entre autonome et autonomique

SHARROCK [2010] dans son manuscrit de thèse définit l'*autonomie* comme le fait de déléguer de la responsabilité au système autonome pour atteindre des objectifs définis. C'est à dire que l'autonomie est l'automatisation de la prise de responsabilité et de la prise de décision pour que les différentes tâches du système soient réussies. L'*autonomique*, quant à lui, est défini comme le fait que le système se contrôle lui-même, sa spontanéité résulte en général de stimulus. C'est à dire que l'autonomique est l'automatisation de la prise de responsabilité et de la prise de décision pour que tout le processus d'opération du système fonctionne.

Boucle MAPE-K

La boucle **Monitor, Analyse, Plan, Execute, Knowledgee (MAPE-K)** est un principe de fonctionnement qui peut permettre à un système d'être autonomique. Elle a été introduite par la société IBM dans son livre blanc [**IBM et collab., 2005**]. Les différentes fonctions qui composent la boucle sont les suivantes :

Monitor consiste premièrement à collecter des informations sur les ressources gérées, telles que la topologie et les métriques de **QoS** entre autres. En second lieu, il s'agit d'agréger toutes ces informations, de les mettre en forme, d'évaluation leur corrélations et de détecter des symptômes qui ont besoin d'être analyser.

Analyze consiste à analyser en détail les symptômes détectés par la fonction Monitor. Le résultat de cette analyse est envoyé à la fonction Plan.

Plan définit et ajuste les actions nécessaires pour atteindre les objectifs du système en fonction des éléments obtenus de l'analyse.

Execute consiste à exécuter les actions définies par la fonction Plan.

Knowledge , dernier point non moins important pour la boucle, traduit le principe du partage et de l'actualisation continue de l'information entre les différentes fonctions décrites ci-dessus pour permettre leur bon déroulement.

Cette architecture **MAPE-K** est devenue un socle commun à tous les systèmes qui se veulent autonomiques, y compris les réseaux.

Le caractère autonomique pour les réseaux

Ainsi pour un réseau, le caractère autonomique est celui qui lui permet d'avoir une surveillance permanente de son état, de savoir détecter les événements qui peuvent avoir un effet négatif sur cet état et de savoir prendre les décisions permettant de maintenir son bon fonctionnement. la programmabilité des réseaux leur donne plus de flexibilité pour acquérir un caractère autonomique. Plusieurs architecture on été proposées pour fournir une base commune permettant d'insérer facilement dans le réseau des fonctions autonomiques. L'architecture CogNet, que nous présenterons dans la suite, en est un exemple.

3.5.3 CogNet : architecture de réseau cognitif et autonomique

5G-SON est la première initiative tendant à apporter des aspects autonomique au réseau du futur. Les **Self Organizing Networks (SON)** adressent uniquement le réseau d'ac-

cès par contre. L'architecture [Generic Autonomic Networking Architecture \(GAN\)](#) [MERIEM et collab., 2016], standardisée par l'ETSI, dont le premier document de référence est paru en octobre 2016 donne une vision globale sur l'architecture attendu pour supporter des fonctions autonome au sein des réseaux. En parallèle un projet européen débuté en 2015 et que j'ai rejoins au cours de mes travaux de thèse traite du même sujet de l'autonomie. En effet, il propose une architecture, qu'on peut par ailleurs voir comme une implémentation de [GAN](#), dédiée à supporter les nombreux cas d'utilisation de fonctions autonomes qui pourraient à l'avenir être implémentées dans le réseau. Les paragraphes suivants décrivent plus précisément l'architecture CogNet proposée.

CogNet est d'abord le nom du projet 5G PPP H2020 co-financé par la Commission Européenne, que j'ai rejoins au cours de mes travaux. Dans ce projet, une architecture de gestion cognitive des réseaux 5G est proposée. Sa vision globale est représentée sur la figure 3.5. La publication [XU et collab. \[2016\]](#) décrit en détail l'architecture. C'est une architecture qui s'appuie sur la notion de réseau virtualisé illustré avec la vue standardisée de [NFV](#).

D'un point de vue global, l'architecture CogNet repose sur l'existence d'agents de collecte de données qui sont distribués au sein des éléments composant l'architecture [NFV](#) et qui transmettent régulièrement les données concernant l'état et la performance du réseau à l'infrastructure CogNet. Au sein de ce dernier, les données sont stockées, analysées, utilisées pour produire et exécuter des modèles issus d'algorithme d'apprentissage comme ceux décrits dans le Chapitre 1. Les résultats d'analyse et d'exécution de modèles à partir des données permettent, tirer des connaissances du réseau (e.g. l'identification d'une dégradation de performance) afin de déclencher des actions de correction, d'amélioration ou de maintien du fonctionnement du réseau (e.g. augmenter la capacité du réseau). Pour ce faire, l'infrastructure CogNet est composée de plusieurs éléments essentiels à savoir un collecteur et entrepôt de données appelé *Data Collector*, un moteur d'apprentissage et d'exécution de modèles appelé *CogNet Smart Engine* et un gestionnaire de règles appelé *Policy Manager*. Je détaillerai dans la suite le rôle de chacun de ces composants.

Data Collector

Son rôle est de collecter des données issues de différents éléments du réseau. Il assure également le nettoyage et filtrage des données (réduction de dimension, suppression de données dupliquées et inexploitées, traitement de valeurs manquantes, etc. Un entrepôt de données permet de stocker ces données nettoyées destinées à être utilisées par le CogNet Smart Engine.

CogNet Smart Engine

Le CogNet Smart Engine est le composant central où l'intelligence de la gestion cognitive réside. Il inclut plusieurs fonctionnalités à savoir le *pré-traitement de données*, la *sélection d'algorithme*, le *traitement batch*, le *traitement (quasi) temps-réel*. Elles sont décrites dans les paragraphes suivants.

- **Le pré-traitement** peut être automatique ou manuel et vise à préparer les données destinées directement au moteur de traitement batch. Il inclut la notion de « Feature Processing », qu'on peut traduire par « sélection de variables », qui permet de générer automatiquement à partir des données des variables intéressantes destinées à l'apprentissage de modèles.

- **La sélection d'algorithmes** a le même rôle que le pré-traitement. Il est capable de sélectionner et identifier les modèles qui doivent être déployés automatiquement en fonction de besoins métier ou des services à demandés par les clients.
- **Le traitement batch** permet surtout d'implémenter tous les algorithmes d'apprentissage supervisé et non supervisé, mais aussi d'exécuter les modèles qu'il produisent dans les cas où le résultat attendu dépend de plusieurs entrées qu'il faut stocker. Prenons l'exemple d'une détection d'anomalie par *scoring*. Ce cas précis peut nécessiter le *scoring* de plusieurs évènements suivi d'une combinaison linéaire des scores obtenus.
- **Le traitement (quasi) temps-réel** est plus dédié à l'exécution des modèles appris en traitement batch. Cependant il permet aussi d'implémenter des algorithmes d'apprentissage en ligne et/ou par renforcement qui apprennent sur des flux de données. Des logiciels open source candidats tels que *Apache Storm* et *Apache Spark Streaming* permettent respectivement d'effectuer des traitements le temps-réel et en quasi temps-réel.

Policy Manager

Son rôle est de traduire les résultats issues du CogNet Smart Engine en préconisation d'actions à mener grâce à une approche à base de règles. Ces préconisations sont destinées au **MANO** décrit dans la section 3.5.1, comme illustré dans la figure 3.5. Des interfaces standardisées par l'**IETF** permettent la communication entre le Policy Manager et le **MANO**. Ce dernier est chargé d'exécuter les actions préconisées, dans le but par exemple de corriger un problème de congestion, une dégradation de performance, une violation de **SLA**, etc. Le Policy Manager a aussi des sous-composants que nous appelons le *moteur de règles*, la *base de données de règles* et le *transmetteur et l'exécuteur de règles*.

- **Le moteur de règles** fait le lien entre les résultats reçus du CogNet Smart Engine et les préconisations à transmettre au réseau via le **MANO**. Il peut fonctionner d'une manière semi automatique. C'est-à dire, d'une part, qu'elle peut inclure des fonctions d'optimisation et des algorithmes d'apprentissage qui lui permettent de décider de manière autonome des politiques à adopter selon les situations, et d'autre part, qu'elle peut nécessiter l'intervention des équipes opérationnelles pour y implémenter des règles de gestion.
- **La base de données de règles** est celle qui répertorie les règles et politiques générées et/ou utilisées par le moteur de règles.
- **Le transmetteur et l'exécuteur de règles** s'occupent de donner l'ordre au réseau via le **MANO**, par une simple transmission de l'information ou une utilisation des **API** du **MANO** pour déclencher les actions.

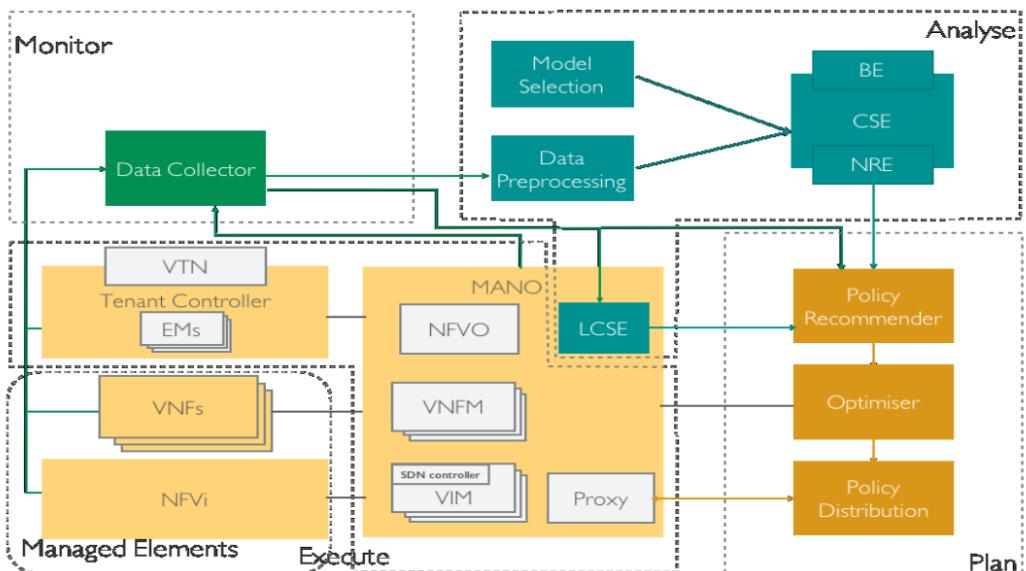


FIGURE 3.5 – Vue globale de l'architecture CogNet

Deuxième partie

Contributions algorithmiques

Chapitre 4

Prédiction instantanée de débit dans le réseau mobile

Sommaire

4.1 Introduction	44
4.2 État de l'art	45
4.2.1 Prédiction basée sur la connaissance des caractéristiques du chemin réseau de bout en bout	46
4.2.2 Prédiction basée sur la connaissance de l'historique de débit	46
4.2.3 Prédiction basée sur la connaissance de mesures de la couche physique	47
4.2.4 Synthèse	48
4.3 Données collectées	48
4.3.1 Campagne de mesure	48
4.3.2 Description des données	50
4.3.3 Pré-traitement des données	52
4.4 Études des corrélations bivariées avec le débit	54
4.5 Modèles de prédiction instantanée du débit mobile	57
4.5.1 Méthodologie	57
4.5.2 Évaluation de performance	58
4.5.3 Résultats	59
4.6 Discussion : Opportunités et Limites	60
4.7 Pistes d'implémentation	61
4.8 Conclusion	63

Le débit descendant peut varier de manière significative dans les réseaux cellulaires, avec éventuellement, un effet non négligeable sur l'expérience de l'utilisateur. Les fournisseurs de contenu multimédia résolvent ce problème en utilisant différentes représentations (par exemple, plusieurs résolutions d'une image ou d'une vidéo) d'un même contenu et en basculant parmi celles-ci en fonction de mesures de bande passante effectuées lors de la connexion. Une connaissance du débit réalisable avant l'établissement de la connexion aiderait les fournisseurs de contenu à choisir la représentation la plus appropriée dès le début. La prédiction de ce débit réalisable constitue l'objet de notre étude.

Nous avons mené plusieurs campagnes de mesure impliquant un panel d'utilisateurs connectés à un réseau de production en France, pour déterminer s'il est possible de prédire le débit réalisable en utilisant des mesures recueillies sur le réseau de l'opérateur et sur les terminaux mobiles des utilisateurs, avant d'établir une connexion. L'analyse de ces données montre qu'il est effectivement possible d'exploiter ces mesures pour prédire le débit réalisable avec une précision acceptable. Cela a permis d'élaborer des stratégies de coopération entre le terminal mobile, le réseau d'opérateur et le réseau de diffusion de contenu pour pouvoir mettre en œuvre des solutions d'anticipation.

4.1 Introduction

Dans les réseaux cellulaires, la qualité de service (QoS), en particulier le débit, peut varier considérablement en fonction de facteurs tels que les conditions du canal sans fil (radio), l'activité des utilisateurs et leur localisation. Pour faire face à la modification de la qualité de service, les fournisseurs de contenu implémentent des stratégies adaptatives de livraison, où la qualité et les caractéristiques du contenu fourni sont ajustés pour correspondre à la QoS réalisable de chaque utilisateur. Ces stratégies adaptatives sont réactives : les caractéristiques du contenu délivré à l'instant t sont basées sur des mesures recueillies entre le début de la connexion et t .

Pourtant, les fournisseurs de contenu doivent prendre quelques décisions clés au début de la livraison. Par exemple, la plupart des services Web ont plusieurs feuilles de style pour leur pages web, avec un nombre variable d'éléments [HAVA MUNTEAN et MC MANIS, 2006; VETRO et TIMMERER, 2005]. Le choix de la feuille de style à livrer doit être prise au tout début de la connexion, alors qu'aucun historique sur la performance du chemin réseau n'est disponible. Un autre exemple est le streaming vidéo adaptatif, où la vidéo est divisée en plusieurs morceaux, et chaque morceau codé à un débit différent, correspondant à un niveau de qualité. Tout au long de la livraison, le client sélectionne la représentation de la plus haute qualité en fonction d'une estimation de la bande basée sur l'historique de performance la plus récente. Au tout début, cependant, aucun historique n'est disponible. La livraison commence souvent par une représentation de qualité moyenne ou faible pour ne pas prendre de risque [MOK et collab., 2016]. Que ce soit pour les feuilles de style de la page Web ou pour les morceaux de vidéo, les fournisseurs de contenu pourraient pallier l'ignorance de la qualité réalisable au début de la connexion, en bénéficiant d'une estimation raisonnablement précise du débit réalisable pour un client donné, juste avant le début de la connexion.

La littérature abondante sur la prédiction de débit, qui est détaillée dans la section 4.2, ne répond pas à notre demande pour l'estimation sans historique du débit moyen, c'est-à-dire celui qui traduit le temps nécessaire pour télécharger une certaine quantité de données. De nombreuses propositions antérieures de prédiction de débit sont basées sur les métriques de bout en bout (ou de chemin réseau), telles que le temps d'aller-retour (RTT) et le taux de perte de paquets. Bien que ces propositions puissent être très efficaces, elles nécessitent la mesure de métriques de bout en bout, qui ne sont pas disponibles au tout début de la connexion. Un deuxième groupe de propositions repose sur l'historique du débit. Ces techniques, qui sont principalement utilisées dans le streaming vidéo adaptatif, permettent de prédire le débit dans l'avenir immédiat [CHEN et collab., 2013; HUANG et collab., 2015; ZOU et collab., 2015]. Comme dans le cas des propositions basées sur des mesures de bout en bout, les propositions basées sur l'historique ne sont pas une option au tout début d'une connexion. Plus récemment, LU et collab. [2015] ont proposé d'exploiter les mesures de la couche physique pour prédire le débit à très court terme

(quelques centaines de millisecondes au plus). Bien que ces prédictions à court terme puissent être utiles pour le contrôle de la congestion et à des fins de planification de paquets, ils sont moins pertinents pour les fournisseurs de contenu car les échelles de temps sont trop courtes : les clients ont besoin de quelques secondes pour télécharger une page Web ou un segment vidéo typique.

L'objectif dans cette recherche est d'étudier s'il est possible d'obtenir une estimation assez précise du temps nécessaire pour télécharger un fichier ayant une taille typique de 10 Mo (e.g. le premier segment d'une vidéo streaming). En d'autres termes, nous étudions la prédition du débit moyen sur une période de temps plus longue que ce qui a été étudié dans les propositions récentes susmentionnées. Nous nous concentrerons sur le cas du réseau cellulaire, qui est un environnement difficile puisque la qualité de service peut changer radicalement au cours du temps.

L'idée clé de notre approche de prédition est de mettre en œuvre un apprentissage automatique de modèles de prédition sur un ensemble de données provenant de différentes sources, telles que le canal radio (par exemple la force et la qualité du signal reçu), les coordonnées [Global Positioning System \(GPS\)](#) du client, sa vitesse, la catégorie de son terminal, la bande de fréquences cellulaire et les performance du réseau d'accès cellulaire (par exemple, le débit cellulaire moyen, le nombre moyen d'utilisateurs, taux de réussite de la connexion). Ces différents éléments sont disponibles, soit sur le terminal mobile, soit au niveau du système de gestion du réseau cellulaire ([NMS](#)).

Dans ce chapitre, nous étudions l'idée d'utiliser l'apprentissage automatique pour prédire, sans mesure préalable de bout en bout, le temps nécessaire pour télécharger un fichier. Par souci d'exhaustivité, nous utilisons un grand ensemble de données, que nous avons collecté à différentes périodes dans un réseau mobile de production, et quatre algorithmes d'apprentissage automatique. Nos principales contributions sont les suivantes : (i) nous fournissons une analyse de la relation bivariée entre chaque variable d'entrée et le débit, afin de mieux comprendre la contribution de chaque variable à la prédition du débit cellulaire ; (ii) nous montrons qu'une prédition instantanée du débit réalisable par l'utilisateur est possible. Nous soulignons que la précision dépend du type d'information en entrée ; (iii) nous discutons des implémentations possibles de cette approche pour améliorer les stratégies de livraison adaptative.

Le reste du chapitre est organisé de la manière suivante : après un examen de la littérature dans la section 4.2, nous décrivons les campagnes de mesure et les données collectées dans la section 4.3. Nous analysons la corrélation bivariée entre différentes variables d'entrée et le débit dans la section 4.4. Nous exploitons ensuite ces résultats dans la comparaison de différents algorithmes d'apprentissage automatique pour prédire le débit dans la section 4.5. Enfin, nous discutons de notre approche à la section 4.6 et introduisons des stratégies de mise en œuvre dans la section 4.7.

4.2 État de l'art

Dans cette partie, nous allons passer en revue dans les travaux précédents sur la prédition de débit. On distingue trois grandes familles de proposition concernant la prédition de débit : la prédition basée sur une connaissance des caractéristiques du chemin réseau de bout en bout ([RTT](#), taux de perte de paquets, etc.), la prédition basée sur l'historique du débit pendant la session et la prédition basée sur les mesures de la couche physique. Nous allons décrire l'état de l'art pour chacune de ces familles de propositions. Ensuite, nous résumerons les principales différences entre notre proposition et les travaux précédents.

4.2.1 Prédiction basée sur la connaissance des caractéristiques du chemin réseau de bout en bout

Nos homologues ont analysé le comportement du protocole TCP (Transmission Control Protocol) pour obtenir des modèles précis pour la prédiction du débit basé sur la connaissance des caractéristiques du chemin réseau de bout en bout. Divers modèles ont été étudiés au cours des vingt dernières années [BUI et collab., 2014; CARDWELL et collab., 2000; MATHIS et collab., 1997; PADHYE et collab., 2000; PRASAD et collab., 2003; REN et LIN, 2011; SIKDAR et collab., 2003]. Les résultats théoriques incluent les liens forts entre le débit, le RTT et le taux de perte de paquets. C'est possible d'obtenir une estimation précise de chacun de ces paramètres de bout en bout à partir des deux autres. Ces études ne répondent toutefois pas aux besoins qui motivent nos travaux à savoir, pouvoir estimer le débit avant le début de la connexion, quand aucun historique de mesure n'est disponible. Puisque nous considérons que le système n'a connaissance ni du RTT ni du taux de perte de paquets, nous ne pouvons pas utiliser directement ces modèles pour obtenir une estimation du débit moyen sur une période typique de 10 s par exemple. Une autre famille d'études [HE et collab., 2005; MIRZA et collab., 2007] a visé à remplacer ces paramètres de bout en bout dans les modèles précédents par un ensemble d'autres mesures qui peuvent être obtenues rapidement par sondage du chemin réseau. La prédiction est cependant moins précise qu'avec le RTT et le taux de perte de paquets en raison de l'incertitude sur la validité des métriques sondées. De plus, notre motivation est de ne pas avoir à déployer de moyens de sonder le réseau en amont des connexions, mais plutôt de s'appuyer exclusivement sur l'information disponible instantanément avant le début de la connexion.

4.2.2 Prédiction basée sur la connaissance de l'historique de débit

Une des lignes de recherche concernant la prédiction de débit est constitué par le développement de modèles basés sur l'historique de débit, c'est-à-dire une série de mesures de débit effectuées précédemment. Leur approche est d'utiliser des techniques d'analyse de séries chronologiques (par exemple, le lissage de la série ou le calcul d'une moyenne mobile) pour prédire le débit en observant son évolution dans le passé. Par exemple, XU et collab. [2013] fournissent un cadre dédié à la prévision de la performance du réseau, y compris la prévision du débit au cours des 500 ms suivants sur la base des vingt dernières secondes. Dans le même sillage, JIANG et collab. [2012] utilisent une moyenne harmonique des observations précédentes de débit. LI et collab. [2014] utilisent également une fonction de lissage pour prédire le débit. Enfin, SUN et collab. [2016] divisent une session en périodes de 6 s et utilisent un modèle de Markov caché ([Hidden Markov Model \(HMM\)](#)) pour prédire le débit de chaque période au cours du temps.

Les technologies de vidéo streaming adaptatif utilisent une variante de la prédiction du débit basée sur l'historique pour déterminer quel débit est le plus approprié pour les x prochaines secondes (x peut valoir entre 1 à 15 s) [JIANG et collab., 2012; LI et collab., 2014]. La prédiction est basée sur les techniques de surveillance du *buffer* [HUANG et collab., 2015]. Le niveau du buffer pilote la sélection du débit. Cependant, lorsque le buffer est vide au début de la session, une estimation du débit réalisable est nécessaire, comme le soulignent MOK et collab. [2016].

Tout en ayant prouvé leur efficacité dans les réseaux fixes [JIANG et collab., 2012; LI et collab., 2014; XU et collab., 2013], ces stratégies ne sont pas adaptées aux réseaux cellulaires en raison de la variation fréquente des conditions du canal radio, qui influent sur

le débit réalisable. **YAO et collab.** [2008] ont mis en évidence cette faiblesse en montrant que, dans les réseaux cellulaires, le débit réalisable devrait plutôt être prédit à partir de la localisation du client qui a un impact plus fort sur la qualité du canal radio. Plus récemment, **ZOU et collab.** [2015] ont étudié des méthodes plus précises de prédition du débit réalisable en prenant en compte les variations de qualité du canal radio.

Cependant, ces approches, du fait qu'elles dépendent de l'historique de débit, ne répondent pas au besoin qui motive notre étude, c'est-à-dire le besoin de prédire le débit instantanément sans historique de mesures. De plus, nous étudions de manière globale la combinaison de diverses informations disponibles, relatives à la localisation du client, au canal radio et au réseau d'accès de l'opérateur, pour identifier les paramètres qui ont le plus d'influence sur le débit réalisable.

4.2.3 Prédiction basée sur la connaissance de mesures de la couche physique

Une approche plus récente consiste à prédire le débit en utilisant des mesures collectées sur la couche physique. Notre étude tombe dans cette catégorie, qui a été récemment explorée pour les réseaux cellulaires dans une série d'articles [**LU et collab.**, 2015; **XIE et collab.**, 2015].

LU et collab. [2015] s'appuient sur le fait que les terminaux mobiles et les stations de base échangent périodiquement des mesures sur le canal radio, qui sont utilisées par la station de base pour prendre des décisions de planification. Parmi ces mesures, **LU et collab.** [2015] suggèrent d'utiliser le **CQI**, dont ils déduisent une estimation de débit. L'estimation proposée n'est valable que pour une courte période de temps (500 ms est la plus longue période testée [**LU et collab.**, 2015]), car le **CQI** est très dynamique. Cette approche n'est pas adaptée à nos besoins puisque notre motivation est d'adresser le téléchargement de blocs de données pouvant durer plusieurs secondes (jusqu'à 15 s pour un segment vidéo [**LEDERER et collab.**, 2013]). Plus généralement, nous nous sommes intéressés à prédire le temps nécessaire pour le téléchargement d'un fichier volumineux.

Le cadre présenté par **XIE et collab.** [2015] vise à déduire la bande passante disponible à partir du taux d'utilisation des blocs de ressources physiques (**Physical Resource Block (PRB)**). Leur proposition consiste à estimer une « durée de séjour » de la bande passante du réseau, d'autres termes la durée de validité de l'estimation de la bande passante, en utilisant un *modèle de Pareto* [**XIE et collab.**, 2015]. L'approche consistant à estimer l'utilisation des **PRB** équivaut à celle d'exploiter directement le **RSRQ**, qui fournit déjà l'information sur la qualité du canal sans avoir besoin de mettre en œuvre un autre système de surveillance des ressources radio. Cette approche ne permet pas une prédition instantanée de la bande passante, étant donné que celle-ci ainsi que sa durée de validité doivent être enregistrées en permanence.

La principale différence entre ces approches et la nôtre est que nous collectons des données à la fois du terminal client et des stations de base, à savoir des mesures de la couche physique et des mesures de performance du réseau d'accès agrégées par cellule au quart d'heure. Notre objectif est d'étendre cette série de travaux en ayant une analyse systématique de toutes les mesures disponibles instantanément avant les transmissions (sans avoir à sonder la connexion de bout en bout) et en évaluant leur pertinence comme prédicteurs du débit client.

4.2.4 Synthèse

La popularité des techniques de livraison adaptative de contenu a renforcé le besoin d'une prédition du débit réalisable à la fois raisonnablement précise et instantanée (ne dépendant pas de l'historique récente de débit, ni d'une connaissance des paramètres de bout en bout de la connexion). Nous avons examiné les approches existantes, qui peuvent être basées sur l'analyse de la qualité de la connexion de bout en bout, sur l'historique de mesures de débit, ou sur certaines informations en temps réelles de la couche physique. Seule la dernière peut correspondre une prédition instantanée et est adaptée aux réseaux cellulaires. Cependant, les propositions que nous avons trouvées sur ce créneau sont limitées à de petites échelles de temps (de l'ordre de 500 ms dans les travaux de [LU et collab. \[2015\]](#)) où seul le débit en régime permanent (« steady state ») est adressé. Dans notre étude, nous complétons l'approche consistant à utiliser les informations de la couche physique par l'ajout d'autres informations concernant la performance du réseaux d'accès radio ([RAN](#)), et le contexte de l'utilisateur. Cela permet d'obtenir une prédition raisonnablement précise du débit moyen pour une échelle de temps plus longue, notamment pour les session [TCP](#) connaissant un démarrage lent (« slow start »). Nous évaluons également l'importance de chaque catégorie d'informations en entrée dans la prédition.

4.3 Données collectées

Pour évaluer notre approche sur des jeux de données réels, nous avons effectué plusieurs campagnes de mesure, impliquant plus de 50 bénévoles répartis dans plusieurs villes en France. Dans ce qui suit, nous donnons d'abord un aperçu des différentes campagnes menées en 2016. Ensuite, nous présentons dans la section [4.3.2](#) une description détaillée des données collectées. Enfin, le pré-traitement des données est détaillée dans la section [4.3.3](#).

4.3.1 Campagne de mesure

Durant les campagnes de test, chaque terminal utilisateur télécharge périodiquement un fichier grâce à une application Android [\[V3D, 2016\]](#) à partir d'un serveur dédié suffisamment approvisionné. La taille du fichier téléchargé varie entre 4 et 50 Mo selon la campagne. Les téléchargements utilisent le protocole [TCP](#) et sont effectués grâce au réseau mobile de production réel de l'opérateur Orange en France. Pour chaque mesure, l'application collecte diverses données grâce à l'[API](#) du système d'exploitation Android, au début et à la fin de chaque connexion. En parallèle, une série de mesures concernant la performance du réseau cellulaire est recueillie grâce au [NMS](#) du [RAN](#) de l'opérateur. Ces mesures de performance de cellule comprennent une trentaine d'indicateurs que nous avons analysés et triés. Une description exhaustive des données collectées est fournie à la section [4.3.2](#). La figure [4.1](#) présente de manière succincte l'architecture réseau qui soutient les campagnes de mesure. En effet, nous avons organisé trois campagnes de mesure (désignées respectivement dans la suite par A, B et C), avec des contextes, des tailles de fichier à télécharger et des niveaux de mobilité différents. Cette diversité est représentative des conditions réelles d'utilisation des services mobiles de l'opérateur. Au cours de chaque campagne, l'application est programmée pour effectuer des mesures tout au long de la journée pour tous les terminaux utilisateurs concernés. Le tableau [4.1](#) présente les détails de chaque campagne.

CHAPITRE 4. PRÉDICTION INSTANTANÉE DE DÉBIT DANS LE RÉSEAU MOBILE

TABLEAU 4.1 – Résumé du fichier de données

	campaign				
	A	B	C	all	
Informations générales					
dates	early 2016	early 2016	end of 2016		
nombre de testeurs	40	5	5		
nombre d'observations	5 441	2 632	15 011	23 084	
nombre de modèles de terminal	23	5	5	26	
nombre de cellules	390	40	122	497	
nombre de villes	112	14	41	147	
Taille de fichier					
4 Mo / 20 Mo / 50 Mo	5 441 / 0 / 0	0 / 2 632 / 0	0 / 12 437 / 2 574	5 441 / 15 069 / 2 574	
Bandé de fréquence					
proportion of LTE2600 / LTE800	0,58 / 0,42	0,88 / 0,12	0,63 / 0,37	0,64 / 0,36	
Catégorie de terminal					
proportion de cat,3	0,23	0,00	0,00	0,05	
proportion de cat,4	0,70	0,59	0,77	0,74	
proportion de cat,6	0,07	0,41	0,23	0,21	
Context					
proportion d'indoor / outdoor / inconnu	0,41 / 0,15 / 0,44	0,47 / 0,31 / 0,22	0,59 / 0,26 / 0,15	0,52 / 0,24 / 0,23	
proportion de statique / mobile / inconnu	0,57 / 0,08 / 0,35	0,60 / 0,20 / 0,20	0,83 / 0,02 / 0,15	0,74 / 0,06 / 0,20	
médiane / moyenne vitesse (in m/s)	0,00 / 0,45	0,00 / 0,39	0,00 / 0,14	0,00 / 0,23	
médiane / moyenne de la distance à la station de base (en m)	857 / 1 302	702 / 1 387	2 931 / 2 777	1 879 / 2 336	
Radio					
médiane / moyenne de RSRQ (en dB)	-7,7 / -8,3	-6 / -7	-8 / -8	-8 / -8	
médiane / moyenne de RSRP (en dBm)	-106 / -104	-110 / -109	-111 / -109	-111 / -108	
médiane / moyenne de RSSI (en dBm)	-79 / -80	-83 / -83	-80 / -82	-80 / -82	
RAN					
médiane / moyenne de débit moyen cellulaire (en Mbit/s)	29 / 27	30 / 29	29 / 28	29 / 28	
médiane / moyenne nombre moyen d'utilisateurs	14 / 20	2 / 9	9 / 16	7 / 15	
médiane / moyenne taux d'erreur de bloc	0,05 / 0,06	0,05 / 0,05	0,08 / 0,09	0,07 / 0,07	
médiane / moyenne du taux de succès de connexion	0,999 / 0,997	1,000 / 0,999	0,999 / 0,998	0,999 / 0,998	
Débit (en Mbit/s)					
1er quartile / médiane / 3ème quartile	13 / 23 / 36	25 / 37 / 46	11 / 24 / 34	12 / 26 / 36	
moyenne	25	34	23	25	

La campagne A s'est déroulée début 2016. Elle a impliqué 40 volontaires répartis sur plusieurs villes en France. Cette campagne présente une grande diversité de cellules, avec un accent sur les contextes urbains. Pour éviter une trop grande consommation du forfait des volontaires, nous avons limité la taille de téléchargement à 4 Mo.

La campagne B a eu lieu pendant la même période que la campagne A, mais sur cinq smartphones de test dédiés avec des cartes SIM à forfait illimité. Le taille du fichier à télécharger a alors été fixée à 20 Mo pour ces terminaux. Cette campagne a eu lieu dans des conditions plus favorable, avec des tests le plus souvent localisés dans des zones plutôt touristiques, semi-urbaines, bien couvertes par le réseau. Cependant, il présente une plus faible diversité de cellules.

La campagne C s'est déroulée fin 2016, en utilisant les mêmes cinq smartphones de test que dans la campagne B. Elle a été réalisée pour collecter des données dans des conditions plus diverses. Dans cette campagne, la taille du fichier téléchargé a été définie à 50 Mo pour un sixième des mesures et à 20 Mo pour le reste. Il s'agit principalement de tests en statique dans divers endroits (par exemple, à domicile), mais surtout dans des zones rurales.

Les données collectées comprennent 23 084 tests. On observe au total 26 modèles de smartphones différents répartis selon la catégorie de terminal (cf. section 4.3.2 et 3GPP

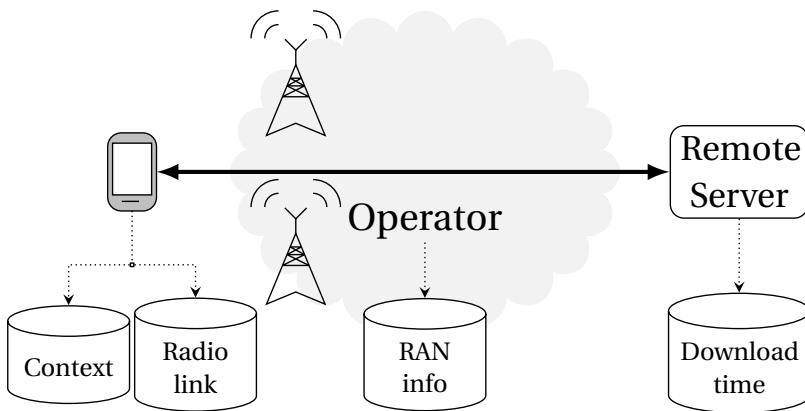


FIGURE 4.1 – Architecture générale des campagnes de collecte de données

[2016]) de la manière suivante : 6 de « catégorie 3 », 18 de « catégorie 4 » et 2 de « catégorie 6 ». Environ 500 cellules, réparties sur 147 communes de France, apparaissent dans les données. La bande fréquence de ces cellules (cf. section 4.3.2) est LTE2600 pour 64% d'entre elles et LTE800 pour les 36% restant. Pour éviter tout artefact de mesure, les sessions ayant subi un *handover*¹ durant le transfert ne sont pas prises en compte dans cette étude.

4.3.2 Description des données

Nous décrivons dans cette partie les données collectées, regroupées selon leur source dans le réseau.

Débit moyen

Nous appelons *débit moyen* la mesure qui fait l'objet de notre étude de prédictibilité. Il s'agit du débit calculé par la division de la taille du fichier par le temps de téléchargement. TCP (cf. section 3.1.5) est conçu pour adapter automatiquement le débit d'envoi des paquets en fonction de la bande passante de la connexion. De ce fait, les facteurs qui limitent la bande passante auront un impact sur le débit moyen d'une session. Une session TCP comprend deux phases : une initialisation appelée *slow start* et une seconde phase de stabilité appelée *steady state*. L'initialisation correspond à une augmentation rapide du débit d'envoi dès le début de la session jusqu'à ce qu'un plafond soit atteint. Ensuite, la *steady state* est la phase où la congestion est évitée. Elle correspond à une série d'augmentations et de diminutions du débit, qui visent à le maintenir aussi proche que possible de la bande passante disponible. Pour calculer le débit moyen, nous tenons en compte la durée totale de la session TCP.

Nous avons utilisé différentes tailles de fichiers lors de nos campagnes de mesure afin de couvrir différents cas : les petits fichiers permettent de mesurer le débit moyen sans étendre la durée de la session dans les cas où le réseau a une faible bande passante. De même, les fichiers volumineux permettent de mesurer le débit moyen avec une durée de session pas trop courte dans les cas où la bande passante est élevée. Quelques uns de nos téléchargements, notamment pour la campagne A-4 Mo peuvent avoir fini avant la fin du *slow start* TCP. Néanmoins, puisque 70% des sessions ont duré plus de 1 s pour cette campagne et que l'étendue des durées de session est assez large (cf. figure 4.2), nous estimons que les mesures de débit ont su capturer les variations de la bande passante

1. un changement de cellule ou de technologie au sein d'un même cellule pour un client donné

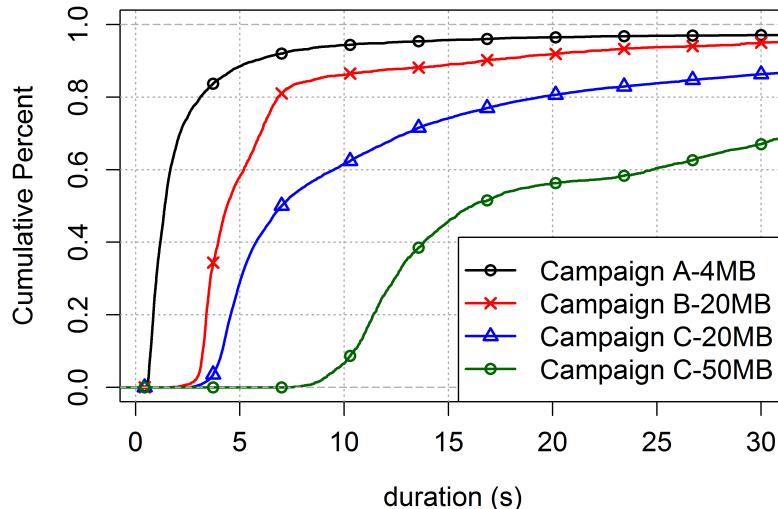


FIGURE 4.2 – Cumulative distribution function of the file download time

du réseau. La fonction de distribution cumulative empirique du débit moyen mesuré est représentée sur la figure 4.2. Pour la campagne B-20 Mo (respectivement C-20 Mo et C-50 Mo), toutes les sessions ont duré plus de 1,5 s (respectivement 2,5 s et 7 s).

Catégorie du terminal et bande de fréquence de la cellule

La norme LTE définit des catégories de terminaux qui déterminent leurs spécifications en termes de performance et permettent aux stations de base de connaître le niveau de performance attendue pour chaque terminal. Les réseaux cellulaires utilisent plusieurs bandes de fréquences : par exemple 2,6 GHz pour la LTE2600 et 800 MHz pour les stations de base LTE800. La LTE2600 est particulièrement utilisée dans les zones urbaines car elle utilise une bande plus large (20 MHz) et ne couvre pas de zone très étendue. La LTE800 utilise une bande de fréquence plus petite (10 MHz). Elle est généralement utilisée en zone rurale pour profiter de sa couverture plus large.

Mesures de la couche physique

Sur le terminal, nous collectons le RSSI, le RSRP et le RSRQ présentés dans le chapitre 3.

Alors qu'il existe d'autres métriques telles que le SINR et le CQI, nous avons observé que la plupart des terminaux ne reportent pas de valeurs précises, probablement à cause du système d'exploitation ou de l'appareil lui-même. L'API fournie par le système d'exploitation permet aux applications d'obtenir des informations sur le terminal. Sur les terminaux *Android*, la disponibilité des données radio dépend de la version du système d'exploitation et des politiques d'économie d'énergie du fabricant. Par exemple, nous avons découvert que la mise en veille de l'écran, sur certains terminaux, bloque la fourniture de certaines données concernant le canal radio au travers de l'API. Du fait que nos mesures ont été effectuées sur chaque terminal sans intervention spécifique de l'utilisateur, de nombreux tests ont été effectués alors que le terminal était inactif, entraînant quelques pertes de données.

En ce qui concerne les données de CQI, elles ne sont plus accessibles sur l'API du système d'exploitation Android. Cela explique pourquoi Lu et collab. [2015] utilisent une

configuration complexe en ayant recours au logiciel [Qualcomm eXtensible Diagnostic Monitor \(QXDM\)](#) pour capturer des données à partir de la couche physique, y compris des données de [CQI](#). Étant donné que nous nous intéressons à une approche de collecte de données et de prédiction du débit réalisable, simple et facile à implémenter, nous nous passons du [CQI](#) dans cette étude et nous concentrons sur le [RSRQ](#) qui est également un indicateur de qualité.

Mesures du réseau d'accès radio

Les opérateurs disposent de systèmes de gestion de réseau ([NMS](#)) qui permettent entre autres de collecter des données de performance sur chaque station de base pour surveiller leur bon fonctionnement au cours de temps. Ces systèmes collectent des compteurs bruts d'événements sur le réseau qu'ils agrègent par la suite sur par période temporelle. Nous avons étudié de nombreuses métriques auxquelles nous avons eu accès. Nous nous sommes concentrés sur les métriques qui sont bien corrélées au débit sans être trop corrélés entre elles. Parmi la trentaine d'indicateurs de performance de réseau d'accès collectés dont certains ont de forts corrélations entre eux, nous avons gardé le débit moyen cellulaire, le nombre moyen d'utilisateurs sur la cellule, le [Block Error Rate \(BLER\)](#) de la cellule et son taux de réussite de la configuration « [Radio Resource Control \(RRC\)](#) ». Chaque indicateur est calculé sur une période de quinze minutes. Enfin, malgré la relative stabilité relative des [NMS](#), nous avons également observé des valeurs manquantes sur ces données concernant le réseau d'accès radio. Nous désignons ces mesures du réseau d'accès radio sous l'appellation de « données de RAN » dans le reste du document.

Les informations de contexte utilisateur

Intuitivement, sur le réseau mobile, le contexte de l'utilisateur pendant le téléchargement peut aider à prédire le débit auquel il peut prétendre. Ces informations de contexte sont des indicateurs pertinents comme le montrent les travaux de [Hu et collab. \[2014\]](#) et de [Hahn et collab. \[2015\]](#). Dans nos travaux, nous considérons les indicateurs suivants :

Intérieur / Extérieur : fourni par l'application que nous avons utilisée sur les terminaux (heuristique basée sur le nombre de satellites [GPS](#) visibles).

Distance à la cellule : basée sur les coordonnées [GPS](#) fournies par l'application et sur la base de données de topologie des cellules de l'opérateur, incluant l'emplacement des stations de base.

Vitesse : estimée grâce au [GPS](#) et à l'accéléromètre.

4.3.3 Pré-traitement des données

Chaque fois qu'un terminal télécharge le fichier, il génère une entrée. Cette entrée contient un horodatage et le temps qu'il a fallu pour télécharger le fichier complet. C'est celui qui nous permet de calculer le débit moyen. Une entrée contient également les mesures que nous avons présentées dans la section précédente.

Certaines entrées sont incomplètes. Par exemple, selon la marque, le modèle ou le système d'exploitation du terminal mobile, il se peut qu'une partie des données radio soit indisponible. L'emplacement (coordonnées GPS) n'est pas toujours disponible non plus. Enfin, certains appareils ne signalent pas les mesures sur le canal radio lorsque leur écran est en veille. Nous devons donc faire face à certaines données manquantes.

TABLEAU 4.2 – Les dix schémas de disponibilité de données les plus fréquents

Schéma cat. de terminal bande de fréquence	radio			context			RAN			nb. d'occurrences
	RSRQ	RSRP	RSSI	distance	vitesse	in/outdoor	débit moyen de la cellule	nb. moyen d'utilisateurs de la cellule	BLER de la cellule	
s_1	✓	✓	✓	✓	✓	✓	✓	✓	✓	9,459
s_2	✓	✓	✓	✓	✓	✓	✗	✗	✗	3,347
s_3	✓	✓	✗	✗	✗	✗	✓	✓	✓	1,445
s_4	✓	✓	✓	✓	✗	✗	✓	✓	✓	1,232
s_5	✓	✓	✓	✓	✗	✓	✓	✓	✓	1,224
s_6	✓	✓	✓	✗	✗	✓	✓	✓	✓	1,079
s_7	✓	✓	✓	✓	✗	✓	✓	✗	✓	731
s_8	✓	✓	✗	✗	✗	✓	✗	✗	✓	572
s_9	✓	✓	✓	✓	✗	✓	✗	✗	✗	540
s_{10}	✓	✓	✗	✗	✗	✗	✗	✗	✗	502
Total missing	0	0	3 587	4 877	9 351	4 638	4 679	5 234	7 543	7 543
									5 357	5 357

Nous distinguons dix cas différents, en fonction des métriques manquantes. Nous appelons ces différents cas des schémas de disponibilité des données. Chaque schéma est caractérisé par les métriques manquantes : toutes les entrées appartenant à un schéma ont les mêmes métriques manquantes et sont valides pour les autres métriques. Le tableau 4.2 montre les dix schémas les plus fréquents dans nos campagnes de mesure. Le schéma s1, dans lequel toutes les métriques sont présentes, est le plus fréquent avec 9459 entrées. Nous avons aussi pu exploiter les autres schémas, sachant que l'utilisation des algorithmes de prédiction (présentés dans la section 4.5) nécessitent des données complètes. Par exemple, lorsqu'on utilise les informations radio et les informations de contexte comme données d'entrée, nous incluons les schémas s1 et s2.

Nous avons préféré cette approche de mélange de différents schémas selon les données d'entrées pour conserver le plus d'observations possible sans introduire d'éventuel bruit par imputation. D'autres approches pour traiter les données manquantes pourraient être l'*imputation* — les valeurs sont remplacées par la moyenne de la variable, sa médiane ou une autre valeur découlant d'un apprentissage des données [BUUREN et GROOTHUIS-OUDSHOORN, 2011] —, ou la suppression de toutes des observations contenant des données manquantes — l'instance entière dans le fichier de données est ignorée. Les deux approches ne correspondent pas bien à notre problème. En cas d'imputation, le choix de la valeur d'imputation peut avoir un impact significatif sur les résultats alors que son choix est plutôt arbitraire. L'approche basée sur la suppression des observations contenant des valeurs manquantes réduirait considérablement le volume de données pour l'apprentissage.

4.4 Études des corrélations bivariées avec le débit

Nous calculons d'abord différents coefficients de corrélation bivariée pour étudier le lien entre le débit et chacune des données collectées. Ainsi, nous avons utilisé linkspotter [SAMBA, 2017a], une extension du logiciel statistique R [R CORE TEAM, 2016] qui est issue de nos travaux et que nous présenterons dans le chapitre 6. Nous fournissons un résumé des corrélations dans le tableau 4.3.

- Le coefficient de corrélation de Pearson indique un lien linéaire positif (ou négatif) entre deux variables quantitatives. Plus la valeur du coefficient est proche de 1 (respectivement -1), plus la corrélation (respectivement, la corrélation inverse) est élevée.
- Le coefficient de corrélation de Spearman est comparable à celui de Pearson, mais indique plutôt un lien monotone entre deux variables quantitatives, donc y compris linéaire.
- Le **Maximal Information Coefficient (MIC)** [RESHEF et collab., 2011] mesure entre deux variables quantitatives plusieurs types de corrélation, y compris linéaire, monotone, sinusoïdale, etc. Il s'agit d'une valeur entre 0 et 1. Plus elle est proche de 1, plus la corrélation entre les deux variables est élevée.
- Le coefficient de corrélation Point-Biserial est un cas particulier du coefficient de corrélation de Pearson s'appliquant aux variables binaires.
- **MaxNMI** [SAMBA, 2017a] : comme son nom l'indique, c'est une mesure d'information mutuelle [FANO et HAWKINS, 1961] normalisée basée sur la discrétisation à fréquence égale qui permet d'obtenir sa plus forte valeur. Il permet d'évaluer la corrélation entre deux variables quel que soit de leurs types (qualitative ou quantitative). Toute variable quantitative est discrétisée en intervalles de fréquence égale

TABLEAU 4.3 – Corrélation entre le débit et les autres variables collectées

	Radio			Context			RAN					
	UE cat.	cell freq. band	RSRQ	RSRP	RSSI	distance	speed	location	débit moyen de la cellule	nb. moyen d'utilisateurs de la cellule	BLER de la cellule	taux de succès de connexion de la cellule
Pearson correlation coef.	-	-	0.67	0.39	-0.17	-0.64	0.00	-	0.59	-0.53	-0.39	0.07
Spearman correlation coef.	-	-	0.71	0.45	-0.15	-0.66	0.11	-	0.62	-0.57	-0.43	0.47
MIC	-	-	0.37	0.18	0.33	0.62	0.07	-	0.49	0.30	0.20	0.24
Point-biserial correlation coef.	-	0.69	-	-	-	-	-	0.09	-	-	-	-
MaxNMI	0.09	0.48	0.32	0.14	0.32	0.49	0.02	0.03	0.38	0.24	0.13	0.19

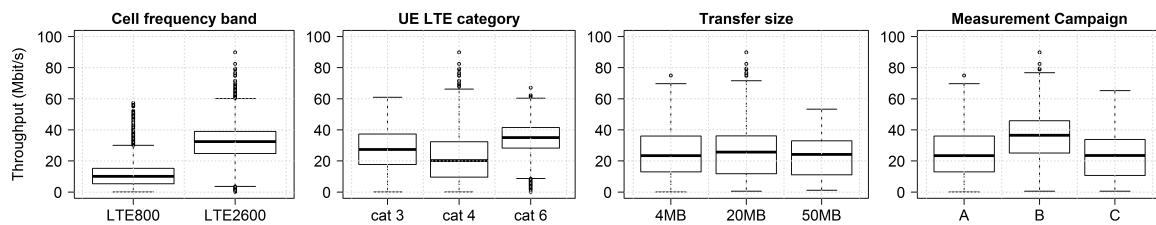


FIGURE 4.3 – Débit moyen en fonction de quelques facteurs

de manière à maximiser l'information mutuelle normalisée du couple. Avec cette normalisation, on obtient un coefficient entre 0 et 1 (plus il est proche de 1, plus la corrélation est forte).

Sur la base des résultats donnés dans le tableau 4.3, nous identifions certaines données qui sont faiblement corrélées au débit : il s'agit de l'emplacement de l'utilisateur (« indoor » ou « outdoor »), la vitesse de l'utilisateur et la catégorie de son terminal. Ces mesures sont difficiles à obtenir avec précision, il est donc possible que la faible corrélation soit due à une défaillance dans le processus de collecte de données.

Nous fournissons une vue graphique générale de la corrélation entre chaque variable et le débit moyen de la session : sur la figure 4.3 pour la bande de fréquence cellulaire, la catégorie du terminal, la taille du fichier de transfert et la campagne de mesure, sur la figure 4.4 pour les données radio, sur la figure 4.5 pour les données de contexte, et sur la figure 4.6 pour les données RAN. Nous présentons les résultats en utilisant des *box-plots* qu'on appelle aussi *boîtes à moustache*. Ils décrivent la distribution complète du débit moyen sur chaque intervalle obtenu par une discréétisation à fréquence égale des variables, pour celles qui sont quantitatives. Une boîte d'un boxplot délimite les quartiles et ses moustaches l'étendu des données (à l'exclusion des valeurs potentiellement aberrantes).

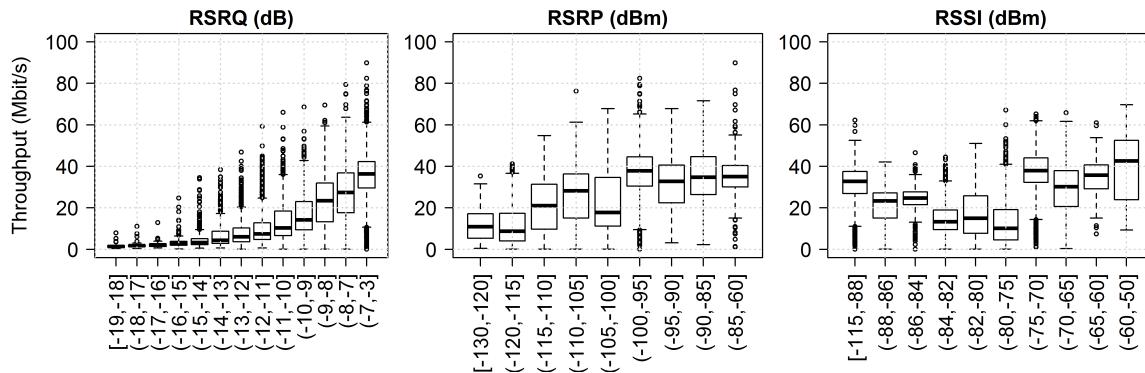


FIGURE 4.4 – Débit moyen en fonction des données radio

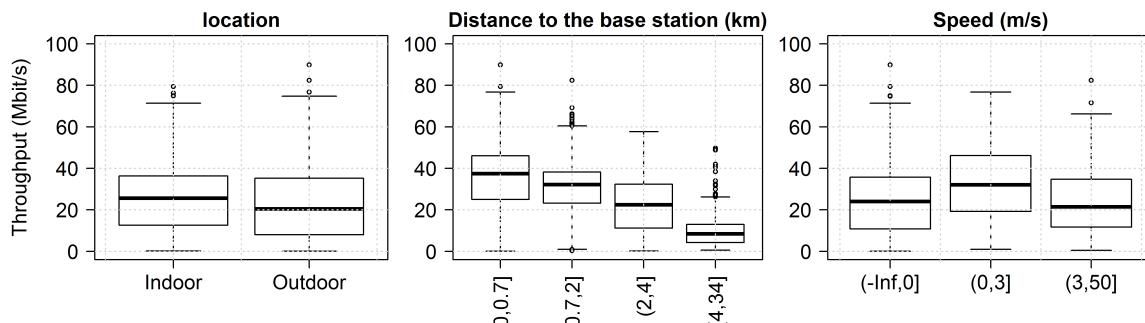


FIGURE 4.5 – Débit moyen en fonction des données de contexte

Nos observations sont les suivantes : (i) la forte corrélation entre la bande de fréquence cellulaire et le débit est visible sur la figure 4.3 où les cellules LTE2600 présentent de meilleur débits que les cellules LTE800. En effet, les cellule LTE2600 ont une plus grande bande passante, permettant d'atteindre des débits plus élevés. (ii) La figure 4.3 illustre également la faible corrélation entre le débit et la taille du fichier transféré ou la campagne de test. La principale différence entre les trois campagnes de test, que représente la taille du fichier transféré, n'a pas impacté significativement le débit. Néanmoins, la campagne B a obtenu de meilleurs débits du fait de la sur-représentation des cellules LTE2600 et des cellules faiblement chargées (cf. tableau 4.1). (iii) Les données radio ont une forte corrélation avec le débit, comme lisible sur la figure 4.4. On remarque cependant un important nombre de valeurs extrêmes au delà des boxplots qui montrent que le débit n'est

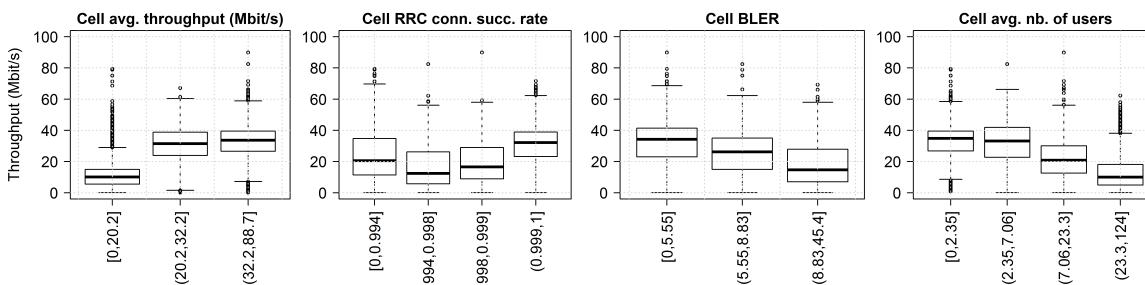


FIGURE 4.6 – Débit moyen en fonction des données de RAN

pas influencé par les données radio uniquement. (iv) Sur la figure 4.5, on lit une corrélation inverse entre la distance à la station de base et le débit. Cela laisse penser que plus on est loin de la station de base, plus la puissance de réception baisse, favorisant un débit plus faible. Cependant il faut aussi noter que les plus grandes valeurs de distance correspondent plus fréquemment à des tests effectués sur des cellules LTE800. Ces dernières ont une couverture plus large et fournissent généralement un débit plus faible que celui des cellules LTE2600, comme expliqué dans la section 4.3.2. (v) La figure 4.6 met en évidence que les quatre métriques que nous avons choisies parmi les mesures de performance du réseau d'accès ont une forte corrélation avec le débit. Ces statistiques reflètent plusieurs dimensions liées à la cellule qui peuvent impacter le débit réalisable pour un utilisateur.

4.5 Modèles de prédition instantanée du débit mobile

Dans la partie précédente, nous avons analysé la corrélation bi-variée entre chaque variable et le débit. Maintenant, nous étendons notre étude en imaginant une prédition de débit basée sur une combinaison de ces variables. Dans cette étude, nous vérifions la précision de la prédition de débit en fonction de la combinaison de données pris en entrée. Nous appelons *prédicteur* chaque combinaison de données pris en entrée. Nous conservons dans cette analyse la typologie des données présentée précédemment, à savoir leur regroupement en donnée de **contexte**, **radio** et de **RAN**. Pour chaque prédicteur, nous effectuons un apprentissage supervisé comme décrit sur la figure 4.7. La première phase consiste à construire le modèle prédictif à partir d'un ensemble de données d'entraînement. La deuxième phase consiste à calculer un débit appelé *débit prédit* à partir du modèle et des prédicteurs sur un ensemble de données de test. La différence entre le débit prédit et la valeur réelle de débit observée sur l'ensemble de test permet de calculer des mesures de précision et de qualité du modèle prédictif.

4.5.1 Méthodologie

Les prédicteurs sont construits comme suit. D'abord, nous incluons toujours la catégorie du terminal et la bande de fréquence cellulaire puisque les deux mesures sont toujours disponibles. Ensuite nous considérons les différentes variables regroupées comme décrit dans la section 4.3.2 : contexte, radio et RAN. Soit i une observation correspondant à une session de téléchargement du fichier. Soit y_i le débit moyen associé à l'observation i . Chaque observation correspond à un schéma de disponibilité des données comme ceux présentés sur la table 4.2. Le schéma de disponibilité de chaque observation détermine les modèles dans lesquels elle sera utilisée, comme élément de l'ensemble d'apprentissage ou dans l'ensemble de test, selon le prédicteur considéré. Pour chaque prédicteur, des modèles prédictifs sont appris en utilisant plusieurs modèles d'apprentissage. Ensuite chaque modèle prédictif permet de calculer le débit prédit \hat{y} sur l'ensemble de test. Nous comparons ensuite le débit prédit \hat{y} au débit réel y_i pour juger de la précision du modèle.

Nous avons utilisé quatre algorithmes d'apprentissage supervisé présenté dans le chapitre 2 pour nous assurer que la précision de la prédition d'un prédicteur donné n'est pas due à un biais dans l'apprentissage, mais plutôt à la corrélation entre ce prédicteur et le débit. Ces algorithmes sont : les Forêts aléatoires (**Random Forests (RF)**) [BREIMAN, 2001b], un modèle linéaire (**Linear model (LM)**), un modèle linéaire généralisé [**McCULLAGH, 1984**] sur les moindres carrés partiels (**Partial Least Square Generalized Linear Model**

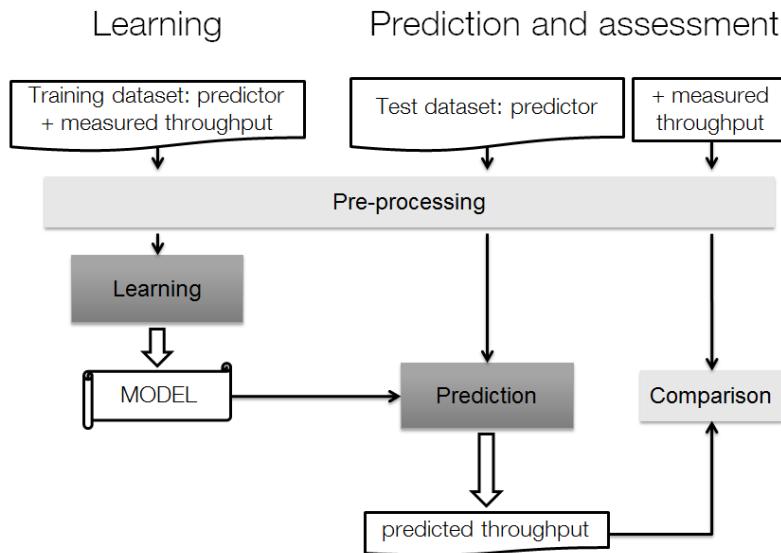


FIGURE 4.7 – Le processus d'apprentissage supervisé

(PLS-GLM)) et un réseau de neurones artificiels (Artificial Neural Networks (NN)) [VENABLES et RIPLEY, 2013]. Nous décrivons chacun de ces algorithmes dans le chapitre 1. Nous avons utilisé le logiciel statistique R [R CORE TEAM, 2016] et plusieurs paquets qui implémentent ces algorithmes : le package *randomForest* [LIAW et WIENER, 2002], le package *stats* [R CORE TEAM, 2016], une extension du package *pls* [MEVIK et collab., 2013] que nous avons codée, et le package *nnet*.

Pour évaluer les algorithmes, nous avons séparé aléatoirement l'ensemble de données en deux sous-ensembles : un sous-ensemble d'entraînement (60%) et un sous-ensemble de test (40%). Le sous-ensemble d'entraînement est utilisé pour construire les modèles en utilisant les algorithmes d'apprentissage supervisé. Le sous-ensemble de test est utilisé pour évaluer le modèle. Cette méthodologie de validation est traditionnelle en apprentissage supervisé. Elle permet, comme nous l'avons expliqué dans le chapitre 1, de vérifier qu'un modèle est généralisable. En d'autres termes, il permet de vérifier que le modèle reste précis quand on l'utilise pour effectuer une prédiction sur une nouvelle observation.

4.5.2 Évaluation de performance

Nous utilisons deux mesures d'évaluation de modèle prédictif de type régression (cf. chapitre 1) : le coefficient de détermination R^2 et le taux d'erreur absolu médian *MedAPE*.

Nous présentons également les résultats de manière graphique avec, premièrement, une représentation de la fonction empirique de la distribution cumulative (Empirical Cumulative Distribution Function (ECDF)) de l'erreur de prédiction sur la figure 4.8. En second lieu, on présente un *heatmap* comparant les valeurs de débit réelles et prédites dans la figure 4.9. Pour les deux, nous nous limitons aux trois principaux prédicteurs : « radio seulement », « RAN seulement », et « radio & RAN ».

TABLEAU 4.4 – Précision de la prédiction selon les données prises en entrée. B est le prédicteur de référence incluant comme variables explicatives la bande de fréquence de la cellule et la catégorie du terminal

Predictors	# metrics	training		test		RF		LM		PLS-GLM		NN	
		# entries	# entries	R ²	MedAPE								
B	2	13,851	9,233	0.49	0.26	0.50	0.26	0,49	0,26	0.50	0.26	0.50	0.26
B + Radio	5	8,210	5,472	0.75	0.19	0.72	0.19	0.74	0.18	0.73	0.19		
B + RAN	6	9,340	6,201	0.72	0.17	0.59	0.23	0.60	0.21	0.62	0.21		
B + Context	5	10,710	7,140	0.59	0.24	0.57	0.25	0.60	0.24	0.58	0.25		
B + Radio + RAN	9	5,872	3,903	0.87	0.11	0.79	0.16	0.81	0,15	0.79	0.16		
B + Radio + Context	8	7,870	5,246	0.82	0.15	0.74	0.19	0.77	0,17	0.75	0.19		
B + RAN + Context	9	7,320	4,863	0.86	0.13	0.67	0.21	0.72	0,19	0.73	0.19		
B + Radio + RAN + Context	12	5,682	3,777	0.89	0.10	0.79	0.17	0.82	0,14	0.81	0.17		

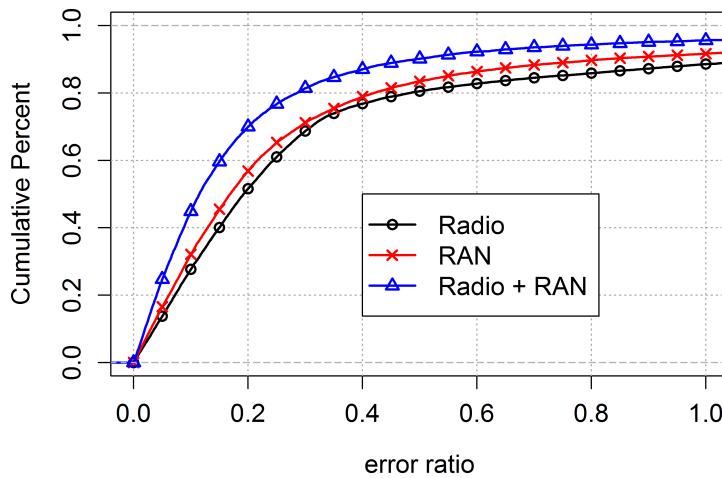


FIGURE 4.8 – Fonction empirique de distribution cumulative du taux d'erreur

4.5.3 Résultats

Nous présentons dans le tableau 4.4 les performances des modèles prédictifs obtenus de chacun des quatre modèles d'apprentissage appliqués à chacune des combinaisons considérées de prédicteurs. Le prédicteur de référence, noté B, correspond à la catégorie du terminal et la bande de fréquence de la cellule. Ensuite, nous augmentons au fur et à mesure la quantité d'informations disponibles.

Nous analysons d'abord les performances des algorithmes d'apprentissage supervisé. On remarque qu'il réagissent tous de la même façon face à l'ajout des différents prédicteurs. En effet, cela montre que chaque combinaison de variable mise en entrée détiennent une certaine capacité explicative du débit considéré, et certaines combinaisons en possèdent plus que d'autres. C'est le cas de la combinaison « Radio + RAN » qui a une meilleure capacité explicative du débit que la combinaison « Radio + Context » par exemple, au vu de leur valeurs correspondantes de R² (respectivement 0,87 et 0,82 pour l'algorithme RF). Le RF culmine la performance avec une valeur R² égale à 0,89 et un taux d'erreur médian égale à 0,10 quand tous les prédicteurs sont utilisés. Les autres approches fournissent des résultats à peu près équivalents. En particulier, l'approche PLS-GLM améliore sensiblement le LM. Cependant le RF reste l'algorithme qui fournit la meilleure précision de prédiction dans notre cas.

Nous analysons ensuite la précision de la prédiction. Les résultats (en particulier un coefficient de détermination calculé sur le sous-ensemble de test égal à 0,89 et un taux

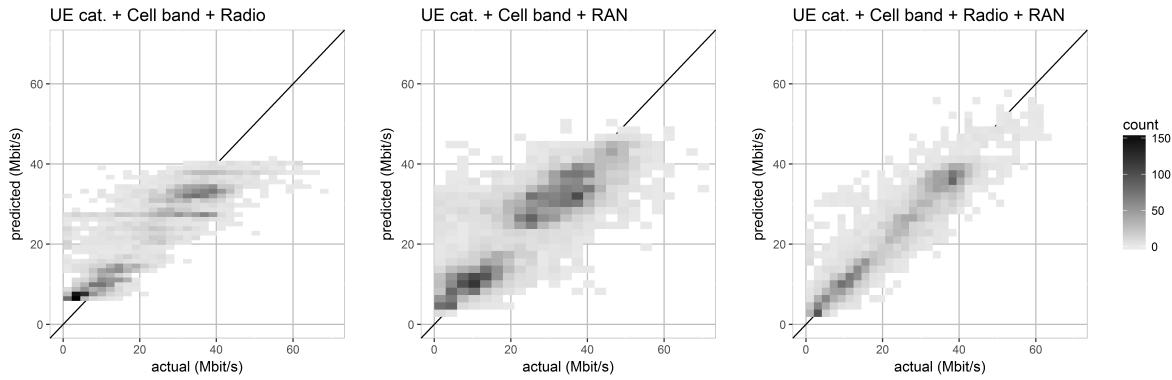


FIGURE 4.9 – Débit moyen prédit versus débit moyen réel

d'erreur médian égal à 0,10 pour les algorithmes de prédition les plus efficaces) sont dans le même niveau de performance que les techniques de prédition non-instantanées [MIRZA et collab., 2007] ne prenant pas en compte l'effet de la partie sans fil (radio) de la connexion en bout de réseau. Notre étude révèle ainsi que la prise en compte des données concernant le contexte de l'utilisateur, le canal radio et l'état du réseau d'accès de l'opérateur conduit à une prédition plus précise du débit. Le *heatmap* représenté sur la figure 4.9 fournit une vue graphique de cette précision. Ce résultat valide notre approche, qui consiste à obtenir une estimation du débit moyen d'un téléchargement suffisamment précis pour permettre aux fournisseurs de contenu de sélectionner une classe de service pour chaque utilisateur final.

Nous analysons enfin la qualité des différents prédicteurs. Nous devons trouver l'équilibre entre la précision de la prédition et la complexité du système de prédition. Nous montrons d'abord que la bande de fréquence cellulaire et la catégorie du terminal ne permettent pas de prime abord une prédition précise. L'information de contexte améliore la performance de la prédition, mais les deux principales familles de données recueillies qui conduisent à une prédition plus précise sont les données de RAN et les données sur le canal radio (le coefficient de détermination est respectivement de 0,75 et de 0,72 avec l'algorithme RF). Ces résultats valident la corrélation que nous avons calculée dans la section 4.4. Deuxièmement, les données de RAN et de radio sont complémentaires puisque leur combinaison augmente le R^2 à 0,87 et limite le taux d'erreur médian à 0,11 avec l'algorithme RF. Ces résultats sont proches des meilleurs obtenus, c'est-à-dire lorsque le prédicteur comprend également les informations de contexte. En d'autres termes, la combinaison des informations sur le canal radio et des informations sur le RAN permet une prédition précise. La figure 4.8 et la figure 4.9 donnent un aperçu de cette observation. Ce résultat ouvre de larges perspectives pour la mise en œuvre d'une prédition instantanée de débit réalisable pour aider les fournisseurs de contenu multimédias à mieux optimiser leur services adaptatifs et à surmonter les aléas de connexion afférant aux réseaux mobiles.

4.6 Discussion : Opportunités et Limites

Nous discutons maintenant les opportunités et les limites de notre proposition. Le débit réalisable d'une connexion sur un réseau cellulaire dépend de la performance de tous les composants impliqués dans la transmission : le terminal mobile, le canal radio, la capacité et l'état de la cellule, le réseau cœur de l'opérateur et même le serveur du four-

nisseur de contenu. À tout moment, l'un de ces composants peut être le goulot d'étranglement qui limite le débit réalisable. Une assertion communément acceptée est que le goulot d'étranglement sur le réseau mobile est le plus souvent situé dans ce que l'on appelle le « dernier kilomètre », c'est-à-dire la partie du réseau située entre le réseau d'accès de l'opérateur et le terminal de l'utilisateur mobile. En effet les opérateurs et les fournisseurs de contenu ont tendance à sur approvisionner leur réseau cœur et leur [CDN](#). Si c'est effectivement le cas, prédire le débit pour le dernier kilomètre équivaut quasiment à prédire le débit de bout en bout.

Dans le même temps, certains chercheurs ont également étudié le cas où le réseau cœur [[SAID et collab., 2013](#)] ou le [CDN](#) [[LIU et collab., 2014](#)] sont sous approvisionnés, auquel cas prédire le débit grâce aux données du dernier kilomètre ne suffit pas. Mais une telle prédition peut encore être exploitée quand elle est combinée avec des informations liées à l'état du réseau cœur et du [CDN](#). En règle générale, depuis que les fournisseurs de contenu utilisent plusieurs [CDN](#) [[ADHIKARI et collab., 2012](#)], des solutions de surveillance de [CDN](#) spécifiques émergent. Il serait donc possible d'intégrer les résultats de ces solutions de surveillance comme des entrées supplémentaires de notre algorithme de prédition de débit.

Une autre source d'amélioration pour notre algorithme est d'utiliser d'autres types d'informations relatives au téléphone portable de l'utilisateur. Sur la couche physique, la sensibilité de réception et la puissance d'émission sont deux variables qui peuvent impacter la qualité de service. Sur la couche applicative, le tampon et même la performance du système d'exploitation sont des éléments critiques. Dans notre modèle, nous prenons en compte la catégorie du terminal mobile, mais nous pourrions aller plus loin dans cette analyse et évaluer si l'écart de performance de prédition peut être expliqué par d'autres paramètres liés au terminal.

Enfin, une limitation de notre étude est la disponibilité relativement faible des donnée radio. En effet, les mesures sur le canal radio sont accessibles par les systèmes d'exploitation des terminaux mobiles, à travers des [API](#) spécifiques. Malheureusement, les développeurs des systèmes d'exploitation mobiles restreignent de plus en plus l'accès à ces [API](#). De plus, la documentation sur ces dernières est relativement pauvre. Dans notre proposition, nous supposons une disponibilité totale des données sur le canal radio, mais cela pourrait nécessiter des efforts pour mettre en œuvre leur collecte. La norme [MDT](#) [[ETSI, 2016](#)] est une opportunité à considérer pour résoudre ce problème car il permet à l'opérateur de réseau d'accéder à des informations sur le canal radio et sur le contexte de chaque abonné du réseau mobile.

4.7 Pistes d'implémentation

Nous abordons maintenant un cadre plus général en considérant notre solution comme l'un des éléments d'une chaîne de distribution impliquant les fournisseurs de contenu, les opérateurs et les utilisateurs mobiles. Nous estimons qu'une telle solution est un net positif pour toutes les parties impliquées, justifiant les éventuels coûts d'implémentation. Nous avons montré que la prédition de débit instantanée et suffisamment précise est possible. Cependant, nous n'avons pas encore expliqué comment une telle prédition pourrait être partagée et exploitée. Nous fournissons ici quelques détails sur le processus qui pourrait être envisagé, en nous appuyant sur la figure 4.10, étant donné qu'une mise en œuvre complète n'est pas le but à ce niveau. Nous discutons également de futurs travaux possibles sur divers aspects ces pistes d'implémentation.

D'abord, on distingue trois acteurs principaux :

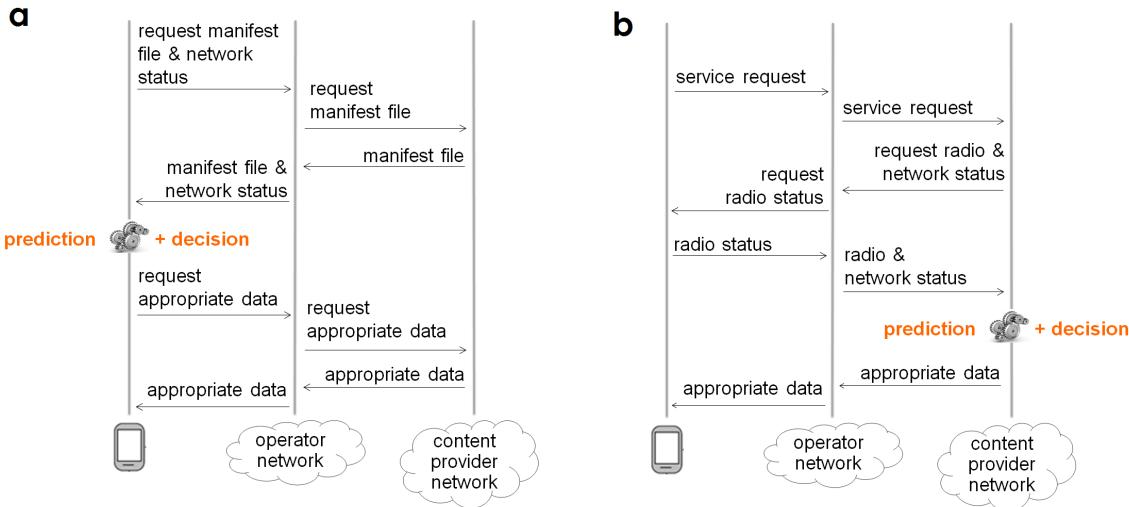


FIGURE 4.10 – Implementation strategies. a : client-based pull delivery, b : server-based push delivery

- L'utilisateur mobile, qui tire parti de l'[API](#) du système d'exploitation pour obtenir des informations sur le canal radio. L'utilisateur mobile peut accorder l'accès à ces données aux autres acteurs.
- Le fournisseur de contenu, qui prépare plusieurs représentations possible de ses contenus correspondant à différents niveaux de qualité de service. Dans les systèmes tels que [Dynamic Adaptive Streaming over HTTP \(DASH\)](#), le fournisseur de contenu décrit ces qualités dans un fichier appelé *manifest file*, de sorte que les autres acteurs soient au courant des différentes options pour la livraison d'un contenu.
- L'opérateur, qui peut partager des informations sur l'état du réseau cellulaire. Un tel partage d'information n'existe pas aujourd'hui et l'une des principales contributions de notre travail est de mettre en évidence les avantages du partage de cette information. Ces dernières années, plusieurs propositions ont déjà été discuté avec des objectifs similaires. En particulier, le groupe de standardisation *MPEG DASH* a ouvert une activité pour s'occuper de la notion de [Server and Network Assisted DASH \(SAND\)](#) [THOMAS et collab., 2015]. Cette proposition définit un élément de réseau appelé [DASH-Aware Network Element \(DANE\)](#) [COFANO et collab., 2016], qui sera typiquement constitué d'un ensemble d'[API](#) permettant aux opérateurs d'ouvrir des données à jour sur l'état du réseau. Une extension d'une telle proposition aux réseaux cellulaires et à toute application mobile devrait être considérée.

[JIANG et collab. \[2014\]](#) ont proposé un cadre plus général de partage d'information entre fournisseurs de contenu, opérateurs et utilisateurs. Ces trois acteurs doivent collaborer pour améliorer la qualité d'expérience globale des utilisateurs finaux. On peut prévoir deux manières de mettre en œuvre la prédiction de débit pour aider la livraison adaptative de contenus : une prédiction côté client et une prédiction côté serveur.

Dans le cas de la prédiction côté client (cf. la figure 4.10-a), le client doit d'abord demander le manifest file auprès du fournisseur de contenu, ainsi que l'état du réseau cellulaire auprès de l'opérateur. Ensuite, le client combine ces informations à ses propres données collectées à propos du canal radio pour effectuer la prédiction de débit. Enfin, sur cette base il sélectionne sur le manifest file la meilleure représentation compatible avec son niveau de débit réalisable prédit. Les avantages de ce scénario comprennent

que le mobile ne révèle pas d'informations privées et que le traitement est effectué du côté client, allégeant le fardeau du serveur. Cependant, les informations sur le réseau cellulaire devraient être disponibles à tout client et cela générerait beaucoup d'accès au système d'information de l'opérateur puisque chaque nouvelle connexion nécessiterait une nouvelle requête.

Dans le cas de la prédiction côté serveur (cf. la figure 4.10-b), le fournisseur de contenu a accès aux données concernant le canal radio depuis le terminal du client ou depuis l'opérateur si un mécanisme tel que MDT est implémenté. Il récupère alors les informations sur l'état du réseau cellulaire de l'opérateur. Ensuite, en fonction de la prédiction, il sélectionne le niveau de qualité de service le plus adapté au contexte instantané du client. Les avantages de ce scénario sont que les informations collectées sur le réseau de l'opérateur peuvent être valables pour plusieurs utilisateurs s'ils se trouvent au même endroit. Elles peuvent donc être utilisées simultanément sans besoin d'une nouvelle requête pour chaque session. Cependant, cela signifie que les fournisseurs de contenu peuvent avoir accès aux informations sur le canal radio des terminaux mobiles.

4.8 Conclusion

Dans les réseaux cellulaires, le dernier kilomètre est généralement le principal goulot d'étranglement. Pour s'assurer la qualité du service dans ces réseaux, les fournisseurs de contenu adoptent des stratégies de livraison adaptative, comme pour la *video streaming* adaptative. S'appuyer sur une prédiction de débit fiable est un facteur clé de succès dans ce contexte. Des moyens de prédire ce débit ont été étudiés dans le passé. Cependant la plupart des solutions proposées sont basées sur une série de mesures antérieures. Pourtant, la prédiction peut ne dépendre d'aucun sondage ni d'aucune nécessité de connaître l'historique de débit. Une telle prédiction instantanée est utile notamment en début de session pour optimiser l'expérience de l'utilisateur. Par ailleurs, la prédiction doit être précise pendant une durée suffisamment longue, correspondant au temps de téléchargement typique d'une page Web ou d'un segment vidéo. La plupart des propositions dans la littérature concernent une prédiction de débit qui n'est valable qu'à court terme.

Dans ce chapitre, nous abordons une solution de prédiction de débit qui remplit ces exigences susmentionnées. Nous étudions la relation inhérente entre le débit et une collection de statistiques qui sont déjà disponibles avant le début d'une connexion. Enfin nous tirons parti de différentes techniques d'apprentissage supervisé pour proposer une prédiction reproductible. Nous avons pu montrer qu'une prédiction précise est possible en combinant les informations sur la qualité du lien cellulaire de l'utilisateur et celles sur la performance du réseau d'accès. Une telle prédiction permettra aux fournisseurs de contenu de mieux adapter leur livraison de contenus dès le début des sessions. Par ailleurs, le partage d'informations entre les parties concernées représente une condition de succès du mécanisme. C'est la raison pour laquelle, nous introduisons des stratégies d'implémentation现实的 pour atteindre cet objectif. La conception et l'évaluation de telles stratégies peuvent constituer d'intéressants travaux à mener dans l'avenir.

Les travaux effectués ont été tout d'abord publiés en version courte dans un workshop [SAMBA et collab., 2016] et dans une conférence internationale [SAMBA et collab., 2017], puis nous les avons étendus en un papier journal en cours de soumission. Le code permettant d'implémenter la prédiction utilisant l'algorithme Random Forest a aussi été publié [MOSCHITTI et collab., 2017; TYMOSHENKO et collab., 2016] dans le cadre du projet de recherche CogNet. En ce qui concerne le transfert de technologie, ces travaux ont pour le moment servi à la société Orange à développer une application [ORANGE SERVICES SRL,

2018] permettant à ses clients d'évaluer leur qualité de service accessible en temps réel. Nous avons donc également présenté notre proposition ainsi que l'application au Salon de la Recherche d'Orange en 2016.

Chapitre 5

Analyse de résultats de tests de performance de réseaux virtualisés

Sommaire

5.1 Développement de la NFV et automatisation de réseau	66
5.1.1 OPNFV	66
5.1.2 ONAP	67
5.2 La vérification d'infrastructure NFV et le projet Yardstick	67
5.3 Approche d'analyse proposée	68
5.3.1 Analyse comparative	68
5.3.2 Analyse explicative : algorithme de classement des paramètres de configuration	69
5.4 Outil TOM	70
5.4.1 Implémentation en couches	70
5.4.2 Fonctionnement en déploiement	71
5.5 Application aux résultats de tests de bande passante mémoire	72
5.5.1 Cas de test et données	72
5.5.2 Benchmarking	72
5.5.3 Analyse	74
5.6 Travaux connexes	78
5.7 Discussion	78
5.8 Conclusion	78

Le principe de la **NFV** est de dissocier les fonctions réseau des équipements matériels qui leur sont dédiés. Dans cette nouvelle manière de concevoir le réseau, les fonctions réseau sont réduites à leur simple aspect logiciel et hébergées dans des machines virtuelles qui peuvent donc être lancées sur des serveurs standards. Ce principe permet de s'affranchir des équipements dédiés traditionnels, puisqu'un data center standard suffit pour instancier un réseau avec toutes ses fonctions. Ce nouveau concept permettra aux opérateurs de réseau de réduire considérablement leurs dépenses de capital (CAPEX) et d'exploitation (OPEX), étant donné qu'ils pourront bénéficier du développement des technologies Cloud pour assurer la flexibilité de leur réseau et leur capacité à passer à l'échelle.

Pour continuer à maintenir une bonne qualité de service avec ce nouveau type de réseau, il est nécessaire pour les opérateurs d'être en mesure d'effectuer automatiquement des tests de performance. Ensuite, les résultats bruts de ces tests doivent subir une analyse permettant, d'une part, la supervision du réseau par ses administrateurs (open loop), et d'autre part, d'alimenter une boucle de contrôle qui donne au réseau son caractère autonome (« closed loop »). Les deux perspectives nécessitent la mise en place de services à valeur ajoutée composés d'algorithmes automatisant l'analyse des données brutes issues des tests et d'outils permettant la mise à disposition de ces algorithmes et leurs résultats aux chaînes de supervision et de rétroaction dans le réseau autonome. L'approche et l'outil TOM que nous proposons s'inscrivent dans ce cadre. L'approche répond à deux besoins. Le premier est d'identifier les configurations qui mènent selon les résultats du test aux pires ou aux meilleures performances. Nous préconisons pour ce faire une méthodologie d'analyse de la distribution des valeurs de la mesure constituant le résultat, en utilisant la notion de boxplot pour la visualisation des distributions. Le second est d'identifier les paramètres qui expliquent les résultats obtenus. Cela revient à analyser la prédictibilité du résultat du test en fonction de ses paramètres connus. Nous proposons pour ce faire une procédure de régression exploratoire pas à pas automatisable et adaptée aux tests de performance. Elle s'inspire de l'algorithme de sélection de variable Stepwise et du modèle linéaire général. Ces notions statistiques sont ici combinées et personnalisées pour des besoins opérationnelles d'analyse automatisée des résultats de tests de performance. L'outil TOM implémente notre approche et fournit une [API](#) composée, d'une part, d'un *Benchmark* capable de fournir un classement des configurations de test, et d'autre part, un *Analyser* capable de déterminer un modèle explicatif du résultat de test.

La deuxième section de ce chapitre revient brièvement sur les avancées actuelles dans le domaine de la virtualisation des fonctions réseau et introduit les activités des projets open source [OPNFV](#) et [Open Network Automation Platform \(ONAP\)](#) qui développent les solutions phares dans ce domaine. La troisième fait un focus sur la vérification d'infrastructure [NFV](#) et décrit les solutions développées dans le sous-projet Yardstick d'[OPNFV](#) qui fournit des moyens tester automatiquement les plateformes. La quatrième section présente les approches d'analyse que nous adaptons au contexte d'analyse automatique de tests. Dans la cinquième section, nous présentons l'implémentation de ces approches à travers l'outil TOM qui facilite leur utilisation dans un contexte de réseau virtualisé. Ensuite, un exemple d'utilisation de TOM pour le cas de test de bande passante mémoire de Yardstick est présenté dans la sixième section. Enfin, avant la conclusion, nous jetons un regard critique sur nos propositions dans la sixième section et rappelons l'état de l'art sur ce sujet dans la septième section.

5.1 Développement de la NFV et automatisation de réseau

L'implémentation de l'architecture [NFV](#) et ses composants a requis que plusieurs opérateurs de réseau et beaucoup de sociétés concernées joignent leurs forces autour de projets open source tels que [OPNFV](#) et [ONAP](#). Dans cette section, nous décrirons les ambitions et quelques réalisations de ces projets open sources.

5.1.1 OPNFV

[OPNFV \[2014\]](#) est une initiative open source qui se vise à impulser le développement et l'évolution des composantes de la [NFV](#) en s'appuyant sur l'écosystème open source. Son ambition est d'accélérer la transformation des réseaux en créant une plateforme [NFV](#)

de référence, facilitant l'intégration, le déploiement et la vérification. [OPNFV](#) inclut quatre groupes de travail et plus de cinquante sous-projets. Dans la section 5.2, nous détaillerons un de ces projets, à savoir [Yardstick](#), qui se focalise sur la vérification d'infrastructure.

5.1.2 ONAP

Dans la même veine, [ONAP](#) [2017] étend l'implémentation de la notion de [NFV](#) et s'intéresse plus particulièrement à l'automatisation, consistant au bloc [MANO](#) de [NFV](#) décrit dans le chapitre 3 à la section 3.5.1. [ONAP](#) fournit une plateforme complète pour l'orchestration à base de règles en temps réel et l'automatisation des fonctions réseau virtuelles. Son ambition est de permettre aux développeurs et fournisseurs de logiciel, de réseau, de solutions système et de Cloud d'être capables d'automatiser le lancement de nouveaux services et le support complet du cycle de vie de ces derniers.

Un des composants phares d'[ONAP](#) est le celui nommé [Data Collection, Analytics, and Events \(DCAE\)](#). En conjonction avec les autres composants, le [DCAE](#) recueille les données de performance, d'usage et de configuration en provenance des environnements opérés et de ses outils de test et de vérification. Ces données alimentent par la suite divers applications d'analyse. Les résultats de ces analyses alimentent automatiquement à leur tour la *boucle de contrôle*, qui peut être ouverte ou fermée. La *boucle fermée* est une forme de boucle de contrôle où les résultats d'analyse déclenchent automatiquement des actions appropriées définies par un *Policy Manager* (cf. chapitre 3, section 3.5.3) qui peut lui-même par ce biais être mis à jour. La *boucle ouverte* est la forme de contrôle où les résultats d'analyse sont destinés aux outils de supervision qui peuvent aussi déclencher des alertes alertant un superviseur humain pour qu'il prenne les décisions appropriées.

5.2 La vérification d'infrastructure NFV et le projet [Yardstick](#)

Les tests et la vérification sont des éléments clés de l'exploitation d'un réseau. C'est encore plus le cas lorsqu'il s'agit d'un réseau virtuel, qui est censé être plus autonome, flexible et tolérant aux pannes.

En conséquence, parallèlement à la mise en œuvre de la norme [NFV](#) [ETSI, 2013a], les opérateurs de réseaux développent en interne et à travers des projets open source des outils à prendre en compte dans les blocs [MANO](#) permettant d'effectuer des vérifications de plateformes et de services. Par exemple, [SHIN et collab.](#) [2015], [SPINOSO et collab.](#) [2015] et [PELAY et collab.](#) [2017] illustrent des propriétés clés à vérifier et fournissent des éléments de formalisme pour les processus de vérification. Par ailleurs, dans les projets open source, des approches plus pratiques ont été prises et des outils de test et de vérification de fonctions réseau virtuelles ont été proposés, conformément aux recommandations de [ETSI](#) [2016]. [Yardstick](#) [YARDSTICK, 2016a] est un de projets de [OPNFV](#), implémentant ces recommandations, à travers un outil de test automatique et des scénarios de test. Son objectif est de permettre une vérification de la conformité de l'infrastructure pendant que les fonctions réseau virtuelles fonctionnent par dessus. Son principe est de décomposer les métriques typiques de performance de fonctions réseau virtuelles en plusieurs caractéristiques et métriques de performance représentées par des scénarios de test distincts. Plusieurs cas de test [YARDSTICK, 2016b] sont considérés, tels que les tests de CPU, de cache, de stockage, de bande passante, de mémoire, de latence et de variation de délai machines virtuelles.

Chaque cas de test peut être présenté comme composé de plusieurs paramètres d'entrée, qui représentent le contexte de test, et une sortie, qui est le résultat du test. Les données de test sont donc composées des valeurs de ces paramètres en entrée et en sortie. A l'heure actuel, une fois le test effectué et les données collectées, il manque encore des moyens d'analyser et d'interpréter automatiquement les résultats. Pourtant, il est important d'intégrer ces moyens dans le processus global de vérification, de validation et de décision afin de réaliser la boucle de contrôle dont nous parlions dans l'introduction et dans la section 5.1.2.

5.3 Approche d'analyse proposée

L'approche d'analyse que nous proposons et son implémentation à travers l'outil TOM décrit dans la section 5.4 vise à constituer un chaînon dans la boucle de contrôle ouverte (« open loop »). En effet, au delà des décisions automatiques qui pourraient découler demain des résultats tests d'infrastructure, il demeure aujourd'hui un besoin pour les opérateurs de tirer des connaissances opérationnelles sur le fonctionnement de leur plateforme grâce à ces tests. Il s'agit d'exploiter tout l'historique de résultats de tests enregistré dans les bases de données pour premièrement comparer les différentes configurations testées, et en second lieu déterminer les paramètres qui influencent le résultat. Par ailleurs, les connaissances tirées de ces analyses exploratoires peuvent aussi alimenter construction ou l'amélioration d'algorithmes décisionnaires. Ainsi, notre approche comporte deux volets dédiés respectivement à l'analyse comparative (comparaison des configurations selon la distribution du résultat de test et visualisation grâce à des boxplots augmentés) et à l'analyse explicative (classement des variables explicatives à l'aide du modèle linéaire général).

5.3.1 Analyse comparative

Nous présentons ici, premièrement l'approche d'analyse comparative de distribution de la mesure de test selon les différentes configurations testées et en second lieu la visualisation par *boxplot* mettant en œuvre l'approche.

Analyse de la distribution d'une variable

La mesure constituant le résultat d'un scénario de test présente toujours une certaine variabilité. En statistique, il est considéré comme l'observation d'une variable représentant le concept mesuré. Outre l'observation de la variabilité de la métrique au fil du temps, des informations importantes peuvent découler d'une comparaison de la distribution de ses valeurs en fonction des différentes configurations testées. Les indicateurs clés à observer sont la moyenne, l'écart type, la médiane, les quantiles révélant l'étendue de la distribution, les valeurs minimales et maximales.

Boxplot

Un boxplot ou boîte à moustaches est un type de figure destiné à la visualisation de distribution d'une variable quantitative. Il est apparu à la fin des années 1970 [TUKEY, 1977]. Il nous permet ici de synthétiser les indicateurs clés énumérés ci-dessus dans un seul graphique. Les figures de la section 5.5 (figures 5.3, 5.4 et 5.5) en sont des exemples.

Il s'agit de dessiner un rectangle (« la boîte ») dont la largeur est limitée par les 25^{ème} et 75^{ème} percentiles de la mesure, respectivement nommé Q_1 et Q_3 . Un trait à l'intérieur du rectangle représente la médiane de la mesure. Les segments représentés d'un côté et de l'autre du rectangle (les « moustaches ») sont limités respectivement par les statistiques suivantes : $Q_1 - 1.5(Q_3 - Q_1)$ et $Q_3 + 1.5(Q_3 - Q_1)$. Ces statistiques sont voisines respectivement du de 1^{er} et du 99^{ème} percentile d'une distribution suivant une loi normale [TUKEY, 1977], car un boxplot permettait à sa création de détecter rapidement la présence de valeurs potentiellement aberrantes en étudiant une mesure qui suit une loi normale. Si la valeur minimale de la mesure est supérieure à $Q_1 - 1.5(Q_3 - Q_1)$, alors c'est elle qui détermine la borne du segment inférieur. Et respectivement, si la valeur maximale de la mesure est inférieure à $Q_3 + 1.5(Q_3 - Q_1)$, alors c'est elle qui détermine la borne du segment supérieur.

Un boxplot peut aussi être se présenter différemment comme le montrent FRIGGE et collab. [1989] et KAMPSTRA et collab. [2008]. Ainsi, pour apporter plus d'éléments de comparaison, nous avons étendu l'implémentation des boxplots dans le logiciel R en fournissant une fonction permettant d'ajouter une représentation de la moyenne et de l'écart type sur un boxplot comme sur les exemples de la figure 5.3.

Ainsi, comparer plusieurs boxplots correspondant aux différentes configurations, dans un ordre approprié, comme sur les figures 5.3, 5.4 et 5.5, permet une comparaison facile et équitable de ces dernières. Néanmoins, l'effectif de mesures pour chaque configuration, permettant de construire son boxplot doit être soigneusement pris en compte. En fait, plus le nombre est élevé, plus les conclusions de comparaison possibles seront fiables.

5.3.2 Analyse explicative : algorithme de classement des paramètres de configuration

Ici, nous présentons l'algorithme de classement des paramètres de configuration que nous proposons. Il s'agit d'un algorithme destiné à classer les paramètres de configuration par ordre d'importance dans l'explication de la variance de la mesure de test. Il construit et évalue plusieurs modèles de régression linéaire (cf. section 2.3.1) pour déterminer pas à pas les collections de paramètres de configurations expliquant le mieux la variance de la mesure de test. Notre algorithme est assimilable à la méthode de sélection de variable « Stepwise Forward » [HOCKING, 1976]. Cependant, à la différence de ce dernier, son ambition est purement exploratoire et il n'exclut aucune variable explicative du modèle final. Nous le décrivons comme suit.

Soit un test où les paramètres de configurations testés sont les variables explicatives X_1, \dots, X_p et la mesure du test est la variable Y. L'objectif de l'algorithme est de déterminer p modèles explicatifs de Y successifs comportant respectivement les 1 à p variables les plus à même d'expliquer la variance de Y. L'algorithme construit pas à pas le modèle complet $Y = f(X_1, \dots, X_p)$, en cherchant à déterminer à chaque étape $l \in \{1, \dots, p\}$, les l variables explicatives permettant d'obtenir le meilleur modèle explicatif $M_{i_1^*, \dots, i_l^*}$ de Y au sens du coefficient de détermination R^2 (cf. section 2.3.2). Nous considérons donc une fonction EVAL qui évalue le R^2 d'un modèle et qui permet de juger de sa qualité. Nous utilisons le R^2 car il peut être interprété comme le part de variance de Y expliquée par le modèle. Il détermine ainsi l'ambition de l'algorithme qui est, à chaque étape l , de déterminer la variable supplémentaire $X_{i_l^*}$ permettant d'obtenir le modèle expliquant le mieux la variance de la mesure de test.

Ainsi, à l'étape 1, on cherche

$$M_{i_1^*} : Y = f(X_{i_1^*}), \text{ EVAL}(M_{i_1^*}) = \underset{i_1}{\operatorname{argmax}} \text{ EVAL}(M_{i_1}) \quad \forall i_1 \in \{1, \dots, p\}. \quad (5.1)$$

Ensuite, à chaque étape $l \in \{2, \dots, p\}$, on cherche

$$\begin{aligned} M_{i_1^*, \dots, i_l^*} : Y = f(X_{i_1^*}, \dots, X_{i_l^*}), \text{ EVAL}(M_{i_1^*, \dots, i_l^*}) = \underset{i_l}{\operatorname{argmax}} \text{ EVAL}(M_{i_1^*, \dots, i_l}) \\ \forall i_l \in \{1, \dots, p\}, i_l \notin \{i_1^*, \dots, i_{l-1}^*\}. \end{aligned} \quad (5.2)$$

Enfin, le vecteur résultat $(\text{EVAL}(M_{i_1^*}), \dots, \text{EVAL}(M_{i_1^*, \dots, i_p^*}))$ est analysé.

L'avantage de cet algorithme réside dans son aspect générique et applicable à tous les cas similaires. Son résultat permet en un coup d'œil à un analyste de déterminer les paramètres de configuration ayant réellement influé sur le résultat du test. Un calcul de la corrélation bivariée entre chaque paramètre et la mesure, par exemple, ne permettrait pas une telle interprétation, car la corrélation entre deux variables peut être due à leur corrélation commune avec une autre ou même d'autres.

Néanmoins, le pré-requis à l'utilisation de cette algorithme et l'interprétation de ses résultats est d'avoir mis en œuvre un protocole de test convenable où toutes les modalités des paramètres de configuration sont testées un nombre suffisant de fois.

Un inconvénient que nous pouvons également noter est que pour une nombre de variables explicatives croissant, le nombre de modèles à construire et à évaluer croît plus vite. Quand nous considérons p variables explicatives, en suivant notre algorithme incrémental, le nombre de modèles à construire et à évaluer au total est $p^{i\text{ème}}$ nombre triangulaire $t_p = \frac{p(p+1)}{2} \forall p \in \mathbb{N}$. Par exemple, si les paramètres de configuration considérés sont au nombre de 3, l'algorithme évaluera au total 6 modèles. Et s'ils sont au nombre de 10, l'algorithme évaluera au total 55 modèles. Cependant, cela ne pose problème que dans les rares cas de test où les paramètres de configuration sont trop nombreux et où la capacité à paralléliser le calcul à chaque étape limiterait le problème.

5.4 Outil TOM

Nous avons implémenté l'approche décrite dans la section 5.3 à travers un outil appelé TOM. C'est un outil d'analyse et d'aide à la décision fournissant un ensemble de visualisation de données et de résultats d'analyse aidant l'opérateur humain à prendre des décisions à partir de résultats de tests de performance convenablement organisés. Il se présente sous trois formes : un package R, une API et une interface web. Sous toutes ses versions, TOM fournit à ce jour deux services : le premier intitulé *Benchmarking* correspondant à l'approche d'analyse comparative et le second intitulé *Analysis* correspondant à l'approche d'analyse explicative.

Dans les paragraphes suivants, nous détaillons l'architecture logicielle de TOM et quelques informations sur son implémentation et son déploiement.

5.4.1 Implémentation en couches

TOM est développé en langage R. Pour donner plus de flexibilité à son développement, nous avons pris l'option de le décomposer en plusieurs composants qui peuvent être utiles dans différents cas d'utilisation. La figure 5.1 illustre les dépendances entre les différents composants de TOM.

Le composant principal est le package R *tom*. Il contient les deux fonctions principale de l'approche de TOM à savoir *compare* pour le *Benchmarking* et *explain* pour la partie *Analysis*. Le package *tom* est lui même dépendant du package R *bbplot* que nous avons développé également.

Le package *bbplot* est un composant annexe de TOM. Il contient la fonction *boosted-boxplot* utilisée par le package *tom* dans la partie *Benchmarking*. Le choix a été fait de séparer cette fonction du reste car elle peut ainsi être utile à d'autres projet indépendant de *tom*. Le package *bbplot* a pour ambition de regrouper diverses fonctions de visualisation ajoutant des fonctionnalités intéressantes à celles fournies par les packages de base de R.

L'autre composant majeur est l'[API](#) TOM. Il utilise et adapte les fonctions du package *tom* pour les fournir en tant que web service. Ainsi, au dessus de l'[API](#) TOM, on peut construire diverses applications clientes utilisant ses fonctionnalités. Par exemple, nous avons créé une interface web graphique constituant une application cliente de l'[API](#) TOM grâce à l'outil Swagger [SMARTBEAR SOFTWARE, 2011].

Par ailleurs, nous avons également développé une application Shiny¹ [[CHANG et collab., 2017](#)] utilisant aussi le package R *tom*. Cette application web offre une interface graphique permettant d'interagir avec une session R lancée sur le serveur distant et donc de paramétriser de manière dynamique les fonctionnalités de TOM.

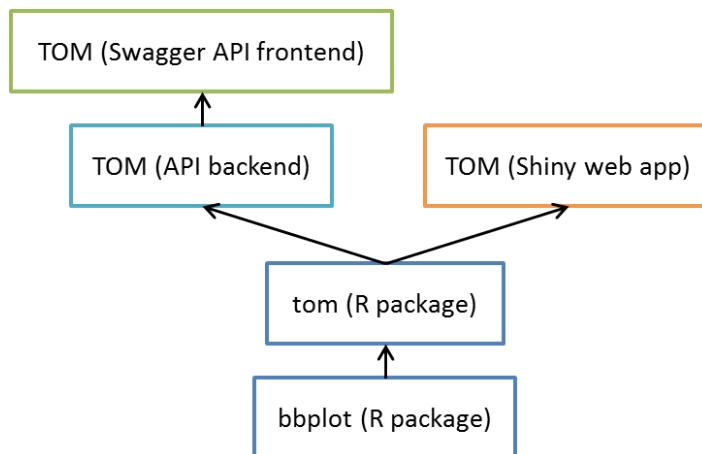


FIGURE 5.1 – Diagramme de dépendance entre les composants

5.4.2 Fonctionnement en déploiement

Les différents composants indispensables (les packages *bbplot* et *tom*) sont utilisés pour déployer l'[API](#) de TOM [SAMBA, 2018b]. Comme illustré sur la figure 5.2, ces différents composants peuvent être déployés au dessus du logiciel R, qui peut lui même être déployé au dessus d'un serveur Linux. Cette entité ainsi obtenue, constitue une instance à part entière de l'[API](#) de TOM. Il y a deux manières d'utiliser cette [API](#) dépendant des données à analyser. Si ces dernières se trouvent dans une base de données InfluxDB comme c'est le cas pour celles du projet Yardstick, il s'agira de faire une requête GET incluant les paramètres de la fonctionnalité que l'on souhaite utiliser et les paramètres de connexion

1. Une application Shiny est une application web graphique développée en langage R grâce au package *shiny* de R. Elle offre une interface web graphique permettant d'interagir avec une session R lancée sur le serveur.

à la base de données InfluxDB. La seconde option correspond à la possibilité offerte de charger du client vers le serveur les données à analyser. Il s'agira d'utiliser une requête POST incluant le fichier de données et les paramètres de la fonctionnalité que l'on souhaite utiliser.

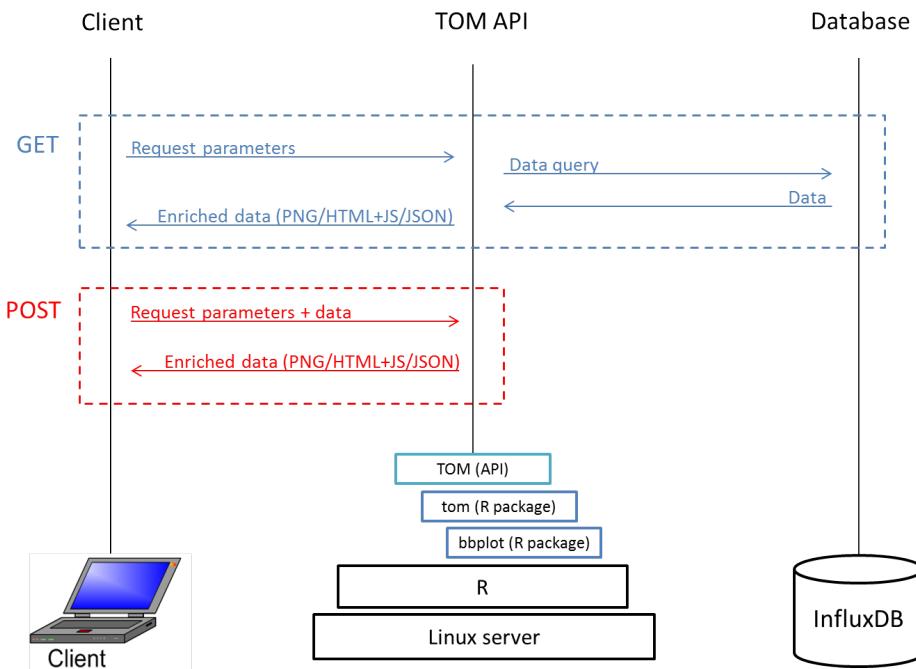


FIGURE 5.2 – Diagramme de séquence de l'API TOM en déploiement

5.5 Application aux résultats de tests de bande passante mémoire

Les travaux d'[OPNFV](#) et notamment les données collectées sur le projet Yardstick ont favorisé la conception et le développement de TOM. Il se place au dessus de tout outil de test et fournit des services d'analyse des résultats des tests. Parmi les nombreux cas de test de Yardstick, nous pouvons prendre en exemple le test de bande passante mémoire pour illustrer l'utilisation et les résultats de TOM. Nous commençons de ce fait par présenter ce cas de test et les données qui en résultent.

5.5.1 Cas de test et données

Le cas de test étudié vise à mesurer la vitesse à laquelle les données peuvent être lues et écrites dans la mémoire. Dans ce cas de test, la vitesse de lecture et d'écriture de données est mesurée en megaoctets par seconde (MBps) sur un même emplacement mémoire.

Les données étudiées ont été collectées entre le 21 septembre 2017 et le 15 janvier 2018. Elles correspondent à la version de Yardstick nommée *Euphrates*.

5.5.2 Benchmarking

La partie « Benchmarking » correspond à la comparaison de la distribution de la métrique entre les différentes configurations. La comparaison se fait grâce aux *boxplots* pré-

sentées ci-après. Les paramètres de configurations de test étudiés sont le scénario de déploiement (*deploy_scenario*), le [Point of Delivery \(PoD\)](#) (*pod_name*) et l'installateur (*installer*). Notre outil TOM nous permet de produire les figures 5.3, 5.4, 5.5, 5.6, 5.7, 5.8 et 5.9 ci-dessous.

Comparaisons selon un seul paramètre

A l'aide de l'outil TOM, nous comparons le résultat de test d'abord pour les différents scénario de déploiement (Figure 5.3), ensuite pour les différent [PoD](#) (Figure 5.4) et enfin pour les différents installateurs (Figure 5.5). Le boxplot produit pour chacun de ces paramètres de configuration fournit une visualisation de la distribution du résultat de test pour chacune de leurs modalités, avec plusieurs indicateurs représentés tels que la moyenne (le point rouge) et l'écart type (la longueur du trait d'une part et d'autre du point rouge).

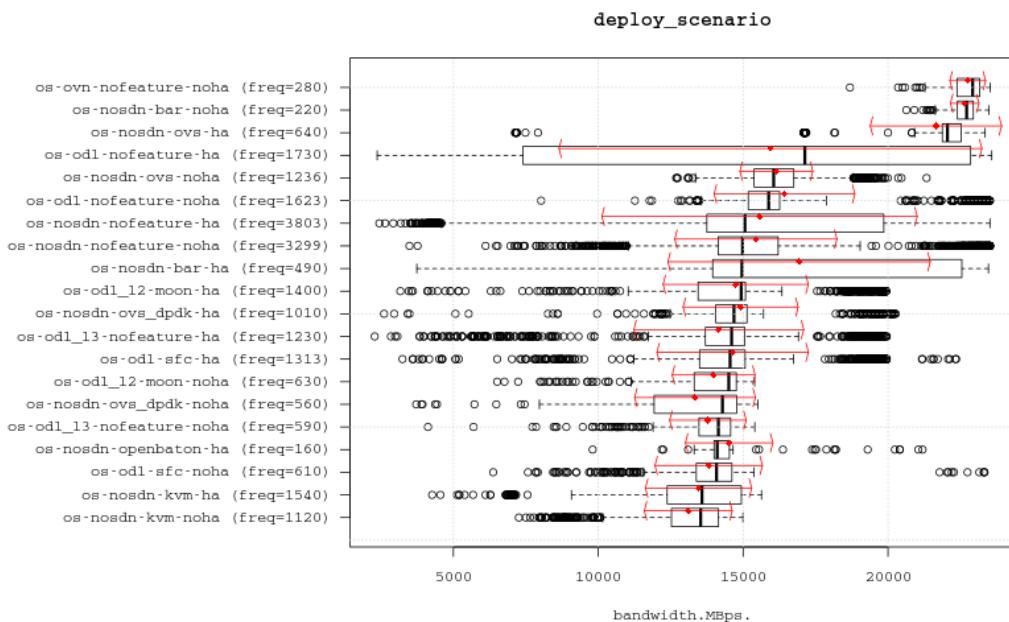


FIGURE 5.3 – Résultats de test de bande passante mémoire selon le scénario de déploiement

Comparaisons selon plusieurs paramètres

L'outil TOM permet aussi de comparer différentes configurations définies par plusieurs paramètres. Par exemple la figure 5.6 compare le résultat de bande passante mémoire entre les différentes combinaisons de scénarios de déploiement et de [PoD](#), la figure 5.7, entre les différentes combinaisons de scénarios de déploiement et d'installateur, et la figure 5.8, entre les différentes combinaisons de [PoD](#) et d'installateur. Cela permet par exemple de visualiser, grâce aux figures 5.4 et 5.8, une des caractéristiques des données qui sera décelée par l'analyse explicative qui suit. Cette caractéristique est la redondance entre le paramètre « *pod_name* » et le paramètre « *installer* ». En effet, sur chaque [PoD](#) a été utilisé un seul installateur.

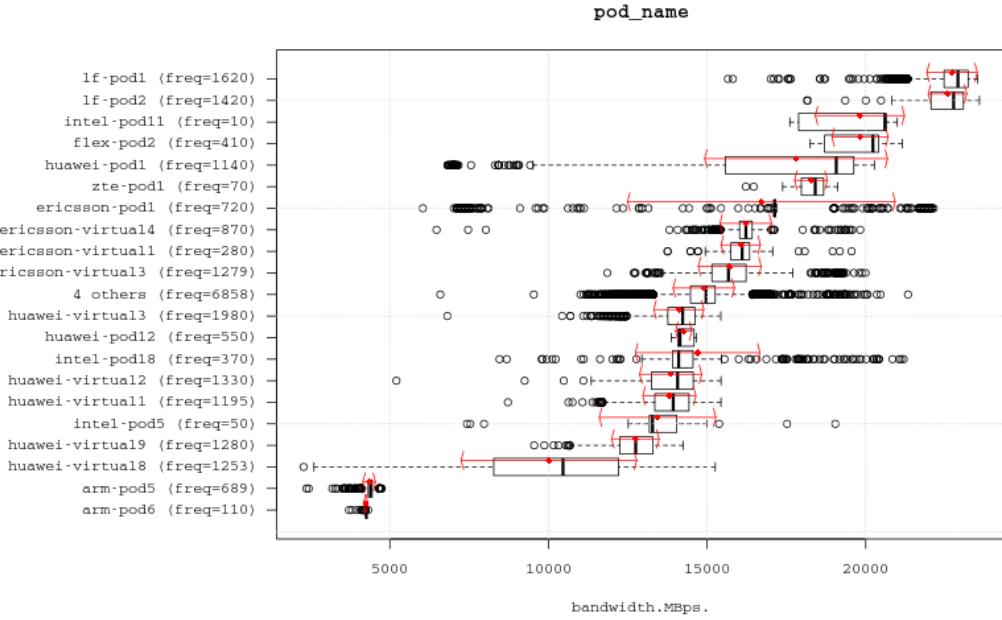


FIGURE 5.4 – Résultats de test de bande passante mémoire selon le PoD

TABLEAU 5.1 – Modèle à 1 variable explicative de la bande passante mémoire

paramètre	R^2
pod_name	0.88
deploy_scenario	0.20
installateur	0.32

5.5.3 Analyse

Dans cette partie, nous présentons les résultats de l'algorithme de classement des paramètres de configuration, en s'aidant de la figure 5.10, et des tableaux 5.1 et 5.2.

Le tableau 5.1 présente le coefficient de détermination obtenu en tantant d'expliquer la variance du résultat de test de bande passante mémoire par chacun des paramètres de configuration. Le tableau 5.2 présente les meilleures valeurs de coefficient de détermination obtenu en tentant d'expliquer la variance du résultat de test par d'abord un seul paramètre, ensuite 2 paramètres et enfin 3 paramètres. Les valeurs de ce dernier tableau sont celles utilisées pour dessiner la figure 5.10 qui apporte un aspect plus visuel de la même information.

On remarque, au vu de son coefficient de détermination qui s'élève à 0.88, que le **PoD** est le paramètre qui explique le plus la variance de la bande passante mémoire.

Malgré la valeur relativement élevée de 0.32 obtenue comme coefficient de détermination de la bande passante mémoire par l'installateur, on constate que ce dernier paramètre n'apporte aucune capacité explicative supplémentaire à un modèle explicatif de bande passante mémoire une fois que ce modèle inclut déjà l'information sur le **PoD** et sur le scénario de déploiement. Cela est dû à la redondance entre le **PoD** et l'installateur déjà évoquée dans la section 5.5.2.

L'utilisation de l'approche d'analyse que nous proposons et que nous avons implantée à travers l'outil TOM permet notamment de déceler facilement ce type de phénomène sur des données.

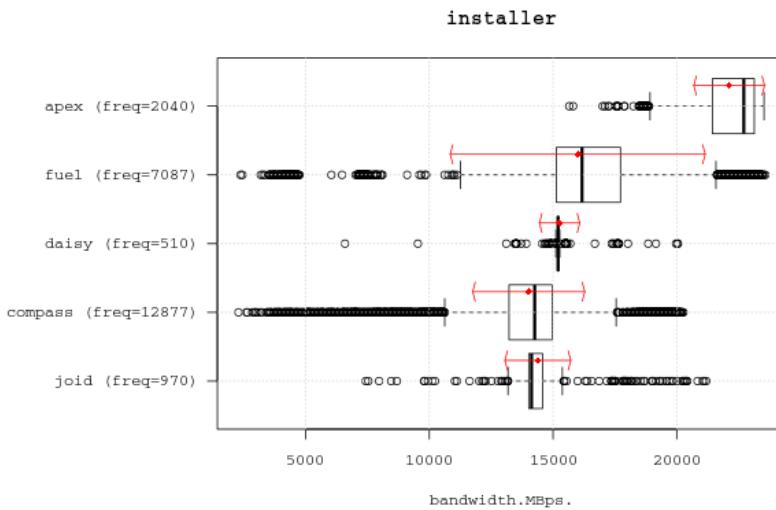


FIGURE 5.5 – Résultats de test de bande passante mémoire selon l'installleur

TABLEAU 5.2 – Modèle à plusieurs variables explicatives de la bande passante mémoire

paramètre	R ²
pod_name	0.88
pod_name + deploy_scenario	0.92
pod_name + deploy_scenario + installer	0.92

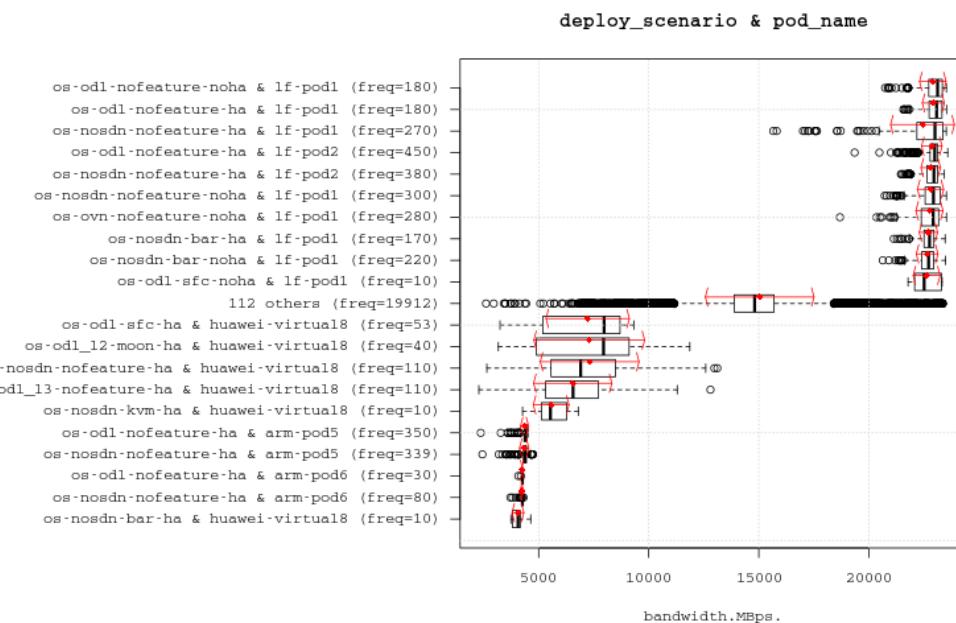


FIGURE 5.6 – Résultats de test de bande passante mémoire selon le scénario de déploiement et le PoD

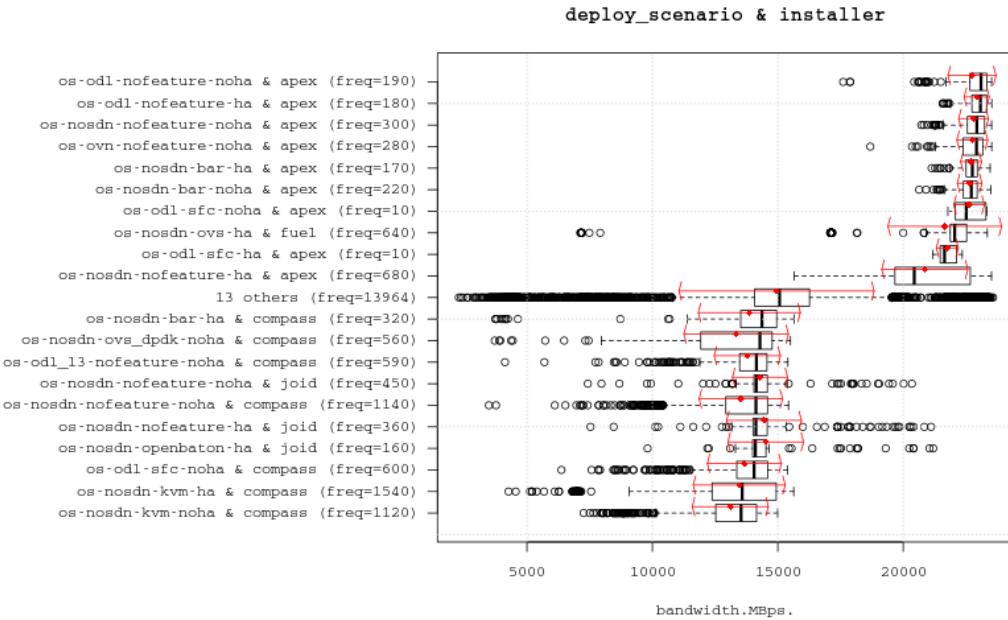


FIGURE 5.7 – Résultats de test de bande passante mémoire selon le scénario de déploiement et l'installleur

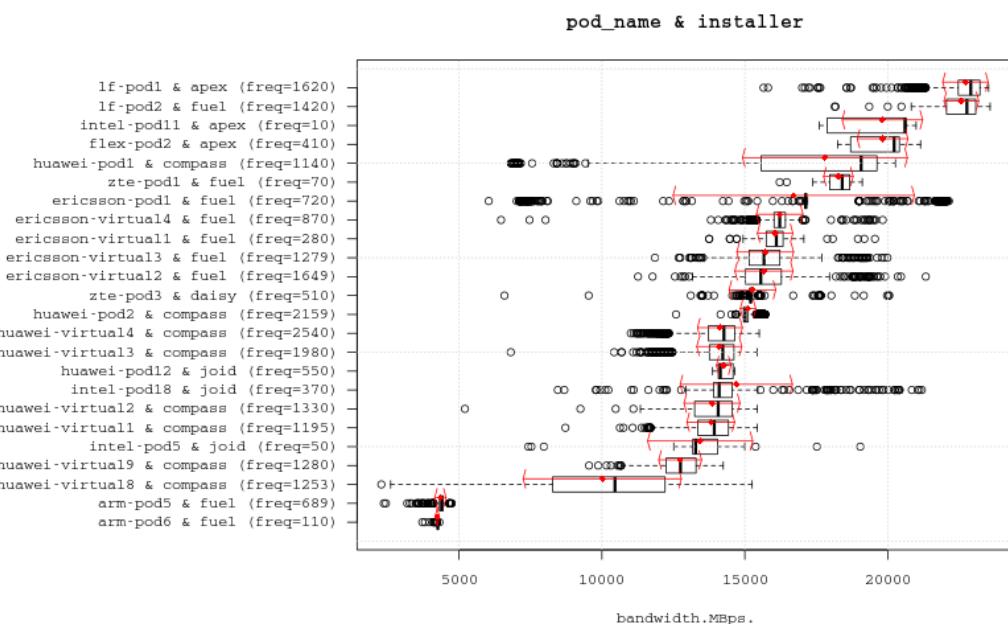


FIGURE 5.8 – Résultats de test de bande passante mémoire selon le PoD et l'installleur

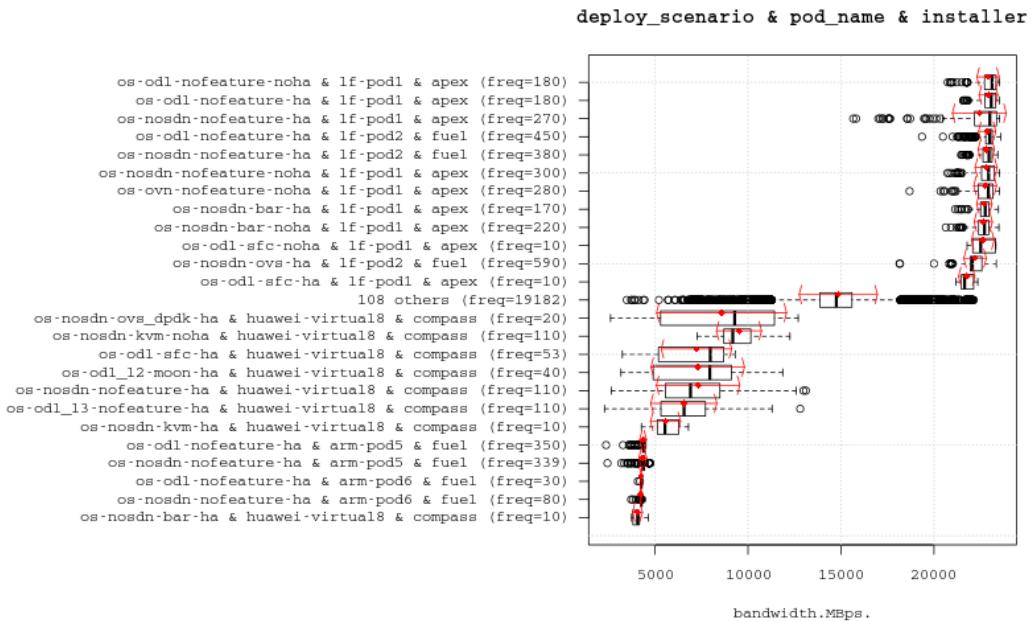


FIGURE 5.9 – Résultats de test de bande passante mémoire selon le scénario de déploiement, le PoD et l'installateur

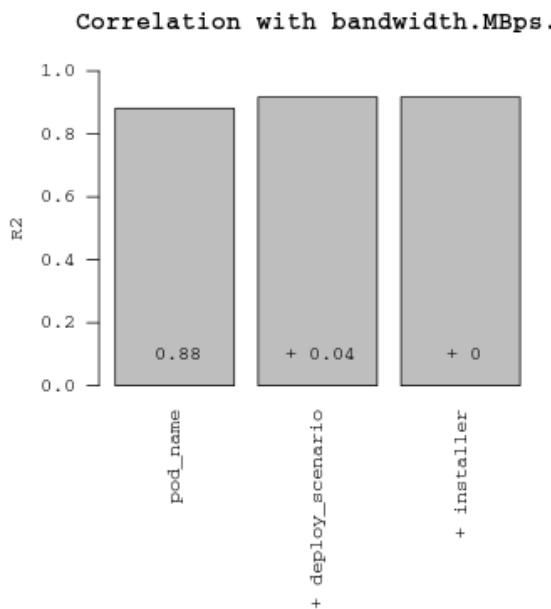


FIGURE 5.10 – Capacité explicative de la bande passante mémoire par les paramètres de configuration choisis

5.6 Travaux connexes

Nos travaux se trouvent dans le même sillage que plusieurs propositions trouvées dans la littérature. **JIANG et collab.** [2009] et **MALIK** [2010] proposent des méthodologies d'analyse de résultats de tests. Les deux traitent des tests de charge.

JIANG et collab. [2009] présente une approche d'analyse automatique de *logs* de test de charge pour analyser les problèmes de performance. Les auteurs infèrent d'abord la référence de performance du système à partir des exécutions précédentes. Ensuite, ils effectuent une comparaison des performances par rapport à la référence déterminée.

MALIK [2010] vise à aider les analystes à identifier automatiquement les compteurs de performance importants pour un test de charge et à les comparer à travers plusieurs tests pour déterminer les gains et les pertes de performance. Il vise aussi à les aider à comprendre la cause d'un échec de test de charge. Les auteurs utilisent pour ce faire une méthode classique d'[Analyse en Composantes Principales \(ACP\)](#).

La différence entre notre proposition et ces dernières est, premièrement qu'elle a l'ambition d'être générique et adaptée à différents types de test, et en second lieu, elle permet de fournir un classement des paramètres les plus influents sur le résultat du test, qu'ils soient des paramètres quantitatives ou qualitatives.

5.7 Discussion

Trois points peuvent être notés pour discuter des précaution d'usage et des améliorations possibles concernant la méthodologie et l'outil proposés.

Premièrement, nos travaux ont uniquement porté sur l'extraction de connaissance à partir des données. Autant les tests à réaliser en amont que la prise de décision en aval sont à la marge de notre solution. Ainsi, pour que les résultats des analyses issues de notre solution soient pertinents, il est nécessaire d'avoir au préalable établi un protocole de test adéquat. Il s'agira d'avoir par exemple un nombre de test suffisant par modalité de configuration et par croisement des modalités de configuration.

Deuxièmement, l'étude a aussi fait ressortir la nécessité d'inclure dans les données à collecter par **Yardstick**, des paramètres décrivant plus précisément chaque **PoD** et chaque scénario de déploiement. Cela permettrait d'aller plus loin dans l'explication des résultats de tests. En effet il est intéressant de savoir que la variance du résultat est surtout dû à la différence entre les **PoD**, mais il serait encore plus intéressant d'avoir dans l'analyse des détails sur les **PoD** (les caractéristiques qui les différencient et celles qui les confondent), afin de pouvoir déterminer les caractéristiques qui ont réellement influé sur le résultat de test.

Enfin, nous pourrions aussi aller plus loin dans l'analyse en incluant plusieurs métriques de test dans une même analyse. Cela permettrait de donner une dimension plus large à la notion de performance d'une configuration. Ainsi la performance serait analysée sous son aspect multidimensionnel. Cela permettrait par exemple de proposer une autre manière de juger de la performance d'un configuration différente de l'approche à base de seuil par métrique.

5.8 Conclusion

Nos travaux offrent une méthodologie pour tirer des connaissances à partir de résultats de tests de performance d'infrastructure **NFV**. La méthodologie et l'outil développés

sont assez génériques pour s'adapter à différents cas de test. Leur utilisation illustre la notion de boucle ouverte dans la gestion des fonctions réseau virtualisées, où un opérateur (humain ou machine) peut tirer des connaissances de données bruts issues de tests afin de prendre les meilleures décisions selon le contexte.

Le code source de la première version de l'outil TOM a été partagé avec la communauté OPNFV, notamment au sein du projet QTIP [OPNFV, 2016]. Des développements supplémentaires que nous avons effectué ont donné lieu à deux applications web déployées [SAMBA, 2018b,c], directement interfacée avec la base de données InfluxDB de Yardstick, pour analyser les résultats de tests enregistrés par le projet en temps réel.

Des travaux futurs consisteront à réaliser un apprentissage non supervisé incluant plusieurs métriques. Cela donnera la capacité d'isoler les configurations fonctionnant le moins bien selon toutes les métriques considérées.

Troisième partie

Contributions logicielles

Chapitre 6

Linkspotter : outil d'analyse et de visualisation de corrélation

Sommaire

6.1 Mesure de corrélation entre deux variables	84
6.1.1 Quantitative-Quantitative	84
6.1.2 Catégorielle-Catégorielle	84
6.1.3 Quantitative-Catégorielle	85
6.2 BeEF : algorithme de discréttisation	85
6.2.1 Algorithme	86
6.2.2 Discussion	86
6.3 MaxNMI : information mutuelle normalisée maximale	86
6.4 Regroupement de variables utilisant la matrice de corrélation	87
6.5 Visualisation de la matrice de corrélation	87
6.5.1 Graphe dynamique	87
6.5.2 Interface utilisateur dynamique	88
6.6 Conclusion	89

L'exploration des relations entre les variables est une étape importante dans le processus d'exploration de données. Il permet au Data Scientist de mieux comprendre les phénomènes décrits dans les jeux de données. Cela aide également à éviter le sur-apprentissage et à se prémunir contre le « concept drift » dans la modélisation.

Afin de faciliter l'exploration des données, Linkspotter est un package R offrant plusieurs fonctionnalités permettant d'analyser et de visualiser de manière exhaustive, en utilisant un graphe, toutes les corrélations bi-variées d'un fichier de données.

Ses fonctionnalités principales sont : (*i*) le calcul de plusieurs matrices de corrélation correspondant à différents coefficients et (*ii*) le partitionnement des variables par apprentissage non supervisé.

Il offre également une interface utilisateur personnalisable permettant de : (*a*) visualiser les corrélations à l'aide d'un graphe (les variables correspondant aux nœuds et les corrélations correspondant aux arcs). C'est une nouvelle approche de visualisation qui est proposée plus conviviale que l'affichage d'une matrice de corrélations coloriée, (*b*) visualiser la distribution de chaque variable grâce à son histogramme ou à son diagramme

en barres, (c) visualiser un lien entre un couple de variables à l'aide de nuage de points, de boîtes-à-moustaches, etc.

En outre, un nouveau coefficient de corrélation que nous appelons **MaxNMI** basé sur l'information mutuelle et la discrétisation supervisée des variables continues est également introduit par Linkspotter. L'intérêt de ce coefficient est qu'il peut être calculé et comparé quel que soit le type de couple de variables (continue vs catégorielle, continue vs continue, catégorielle vs catégorielle).

Notre première section reviendra sur la notion de corrélation entre deux variables et les différents types de méthodes existants pour évaluer cette corrélation. La deuxième section présente notre algorithme de discrétisation **BeEF**. Dans la troisième section, nous présentons notre algorithme **MaxNMI**. Dans la quatrième section nous détaillons le regroupement de variables proposé. Notre cinquième section présente la la méthodologie de visualisation d'une matrice de corrélation proposée, ainsi que les interfaces produits par le logiciel pour faciliter son utilisation.

6.1 Mesure de corrélation entre deux variables

La mesure de la corrélation entre deux variables dépend de leur nature. Comme présenté dans le chapitre 2 une variable peut être quantitative ou catégorielle. Le couple de variables peut donc avoir trois formes pour lesquelles nous présentons dans les paragraphes qui suivent les méthodes de mesure de corrélation utilisables.

6.1.1 Quantitative-Quantitative

C'est la forme de couple de variables la plus courante. Il existe plusieurs coefficients de corrélation pour ce cas. Nous pouvons en citer le r de Pearson, le ρ de Spearman, le τ de Kendall et le MIC que Linkspotter implémente.

Le coefficient de corrélation de Pearson mesure la corrélation linéaire entre les deux variables. Les coefficients de corrélation de Spearman et de Kendall mesurent une corrélation monotonique - et pas nécessairement linéaire - c'est-à-dire celle entre deux variables qui s'influencent positivement ou négativement uniquement. Le **MIC** [RESHEF et collab., 2011] par contre est une coefficient permettant d'évaluer plusieurs types de corrélation - qui ne sont pas nécessairement monotoniques - où le nuage de points entre les deux variables peut prendre différentes formes (e.g. sinusoïde, deux lignes croisée et ellipse).

6.1.2 Catégorielle-Catégorielle

Pour évaluer le lien entre deux variables catégorielles on calcule généralement l'information mutuelle. La normalisation de cette dernière pour se ramener à des valeurs entre 0 et 1 permet d'obtenir notion assimilable à un coefficient de corrélation.

Information mutuelle

L'information mutuelle de deux variables est une mesure de la dépendance statistique entre elles. Son calcul est basé sur la notion d'entropie de Shannon. Elle quantifie la « quantité d'information » exprimée à la fois par les deux variables.

Soient, par exemple, X et Y deux variables catégorielles. L'information mutuelle mesure la quantité d'information apportée en moyenne par une réalisation de X sur les probabilités de réalisation de Y. Elle s'exprime par :

$$I(X, Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) = H(X) + H(Y) - H(X, Y) \quad (6.1)$$

où H est l'entropie de Shannon. Si $I(X, Y) = 0$ alors X et Y expriment deux types d'information totalement différents.

Normalisation de l'information mutuelle

Il existe plusieurs façons de normaliser une information mutuelle (cf. [WIKIPEDIA \[2018\]](#)). Nous considérons dans Linkspotter la suivante :

$$C_{XY} = \frac{I(X, Y)}{\min[H(X), H(Y)]}. \quad (6.2)$$

C_{XY} est donc une variante de l'information mutuelle, assimilable à un coefficient de corrélation. Son calcul consiste à diviser l'information mutuelle par l'entropie minimale entre les deux variables. Cette normalisation permet de considérer l'information mutuelle comme un cas particulier de corrélation totale. Cette information mutuelle normalisée est ainsi définie entre 0 et 1. Plus elle est proche de 1 (respectivement 0) plus la corrélation entre les deux variables est élevée (faible).

6.1.3 Quantitative-Catégorielle

Le cas où seule l'une des variables est catégorielle nécessite généralement deux alternatives : on peut effectuer une *ANOVA* à un facteur ou procéder à la discrétisation supervisée de la variable quantitative grâce à la variable catégorielle.

L'[Analyse de la variance \(ANOVA\)](#) est une méthodologie statistique classique consistant à construire un modèle linéaire généralisé sur les données pour tester une hypothèse de dépendance entre une variable quantitative et une variable catégorielle appelée facteur. Le coefficient de détermination (R^2) obtenu du modèle est assimilable à une mesure de corrélation entre les deux variables [\[RESHEF et collab., 2011\]](#).

Comme expliqué dans le chapitre 2, la discrétisation consiste à transformer une variable quantitative en une variable catégorielle. Elle est supervisée quand elle est effectuée en fonction d'une autre variable. Nous proposons dans ce chapitre un algorithme simple pour discrétiser une variable de manière supervisée. Nous l'appelons [BeEF](#).

6.2 BeEF : algorithme de discrétisation

L'idée sous-jacente à l'algorithme [BeEF](#) est de trouver, pour deux variables, la discrétisation de l'une (ou des deux) qui maximise l'information mutuelle normalisée obtenue entre les deux variables catégorielles. Si les deux variables sont continues, elles sont toutes discrétisées et l'information mutuelle normalisée est calculée pour toutes les combinaisons de discrétisation obtenues. Nous avons utilisé une discrétisation à fréquence égale (« Equal-Frequency »), i.e. la discrétisation est faite en ne considérant que des intervalles équiprobables. Cela simplifie l'algorithme puisque que le seul paramètre à faire varier est le nombre d'intervalles. Un critère d'arrêt proposé par [RESHEF et collab. \[2011\]](#) permet de limiter ce nombre pour éviter de sur-discrétiser et d'allonger le temps de calcul. Nous allons le détailler dans la description suivante de l'algorithme.

6.2.1 Algorithme

L'algorithme distingue deux cas dépendant du type de couple de variables : (*quantitatif, catégoriel*) ou (*quantitatif, quantitatif*).

Si une seule variable est continue, nous effectuons la discréétisation en regroupant des valeurs dans x intervalles de fréquences égales. L'algorithme fait varier x jusqu'à trouver x^* correspondant au nombre d'intervalles respectant le critère d'arrêt qui donne lieu à la plus grande information mutuelle normalisée. Le critère d'arrêt (cf. [KINNEY et ATWAL \[2014\]](#); [RESHEF et collab. \[2011\]](#)) est : $x * y < n^{0,6}$, où y est le nombre de classe de la variable catégorielle. Si les deux variables sont quantitatives, les deux sont discréétisées. Le nombre respectif d'intervalles x et y sont tous deux optimisés, en respectant le même critère d'arrêt. Le couple (x, y) qui atteint l'information mutuelle normalisée la plus élevée est conservé.

6.2.2 Discussion

Le caractère équiprobable des intervalles comporte à la fois des avantages et des inconvénients. L'inconvénient est le risque de ne pas trouver certaines corrélations correspondant à des signaux faibles. En effet, la technique [BeEF](#) ne vise pas à réaliser la meilleure discréétisation possible pour le couple de variables. Néanmoins, il présente l'avantage de réduire considérablement la complexité d'algorithme. Comme les intervalles potentiels sont supposés équiprobables, Le seul paramètre à faire varier pour la discréétisation supervisée est le nombre d'intervalles à considérer. Cela simplifie considérablement l'algorithme.

6.3 MaxNMI : information mutuelle normalisée maximale

Le [MaxNMI](#) vise à fournir un coefficient permettant d'évaluer le niveau du lien entre deux variables quel que soit leur type. Il s'appuie sur une discréétisation supervisée préalable des variables continues et calcule l'information mutuelle normalisée du couple discret. La discréétisation supervisée adopté pour Linkspotter est celle issue de l'algorithme [BeEF](#) présenté ci-dessus. L'algorithme [MaxNMI](#) consiste à discréétiser toute variable quantitative du couple et à calculer l'information mutuelle normalisée du couple de variables qualitatives obtenu. Une discréétisation supervisée par [BeEF](#) est utilisée dans Linkspotter.

L'algorithme générique du [MaxNMI](#) est le suivant :

```

switch Détermination du type du couple (X, Y) do
  case (catégoriel, catégoriel)
    Calcul de  $C_{X,Y}$ 
  case (quantitatif, catégoriel)
    Détermination de  $X^*$ , résultat de la discréétisation BeEF de X en fonction de Y
    Calcul de  $C_{X^*,Y}$ 
  case (catégoriel, quantitatif)
    Détermination de  $Y^*$ , résultat de la discréétisation BeEF de Y en fonction de X
    Calcul de  $C_{X,Y^*}$ 
  case (quantitatif, quantitatif)
    Détermination du couple  $(X^*, Y^*)$ , résultat de la discréétisation BeEF du couple (X, Y)
    Calcul de  $C_{X^*,Y^*}$ 

```

6.4 Regroupement de variables utilisant la matrice de corrélation

Le regroupement vise à trouver des groupes de variables similaires. La similitude de deux variables est déterminée par la ressemblance de leur corrélation avec les autres variables. Par conséquent, le regroupement est basé sur la matrice de corrélation. Le principe est d'effectuer un algorithme de clustering classique comme un *K-means* [MACQUEEN et collab., 1967] ou un *clustering statistique* [FRALEY et RAFTERY, 2002] sur la matrice de corrélation. Ainsi, le résultat dépend du coefficient considéré pour calculer la matrice de corrélation.

Seul le *clustering statistique* implémenté par le package *mclust* de R (cf. [SCRUCCA et collab., 2016]) est fourni dans la première version de Linkspotter. Le nombre de groupes à construire peut être déterminé par l'utilisateur. Sinon, le nombre entre 2 et 9 donnant le meilleur regroupement est choisi.

6.5 Visualisation de la matrice de corrélation

Linkspotter permet de visualiser toute matrice de corrélation en utilisant un graphe. Il fournit également une interface utilisateur qui permet de personnaliser le graphe de manière interactive, de visualiser la distribution de chaque variable et de d'afficher les tableaux correspondant aux matrices de corrélation calculées.

6.5.1 Graphe dynamique

Sur le graphique produit par Linkspotter, les variables correspondent aux nœuds et leurs corrélations correspondent aux arêtes entre les nœuds. Une arête n'apparaît que si la corrélation à laquelle elle correspond est supérieure à la valeur minimale choisie comme paramètre lors du tracé du graphe. Un exemple sur l'ensemble de données *Iris* [FISHER et MARSHALL, 1936] est fourni sur la figure 6.1. Lorsque la souris d'ordinateur de l'utilisateur passe sur une arête, la valeur de la corrélation correspondante est affichée. Selon le choix de l'utilisateur, la couleur des nœuds et des arêtes peut dépendre respectivement du résultat de la mise en grappes et de la direction de corrélation (si les coefficients de corrélation de Pearson r, Rho de Spearman ou Kendall sont calculés). Un bord bleu correspond à une corrélation positive et un bord rouge correspond à une corrélation négative.

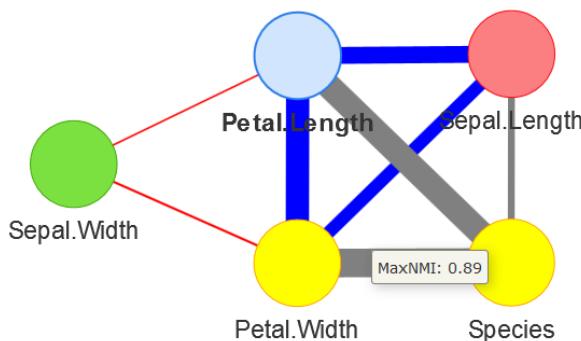


FIGURE 6.1 – Graphe de corrélation obtenu sur le fichier de données Iris

6.5.2 Interface utilisateur dynamique

L'interface graphique comprend deux onglets : "Graphs" et "Tables".

Onglet « Graph »

L'onglet "Graphiques" comprend le graphique et plusieurs autres vues et éléments de personnalisation pour l'analyser. Il s'agit des éléments suivants :

Les fonctionnalités principales sont les suivantes :

- La liste déroulante *Correlation coefficient* : permet de choisir le coefficient de corrélation à visualiser.
- le curseur *Minimum Correlation* : permet de définir le seuil minimal de corrélation nécessaire pour dessiner un lien.
- La liste déroulante *Interest variable* : permet de choisir une variable d'intérêt. Si une variable d'intérêt est choisie, un nouveau curseur *Minimum Correlation with interest variable* s'affiche. Il donne la possibilité de définir un nouveau seuil minimal de corrélation spécifique à cette variable d'intérêt s'enclenche, en plus du seuil général. Ce seuil spécifique a comme borne supérieure le seuil général.

Les options à cocher sont les suivantes :

- *Highlight variable on click* : si elle est cochée, il devient possible de faire un focus sur un nœud en cliquant dessus ou en le sélectionnant sur la liste déroulante qui s'affiche.
- *Variable Clustering* : si elle est cochée, les nœuds sont colorés selon le groupe auquel ils appartiennent. Si elle n'est pas cochée, tous les nœuds sont bleus.
- *Color Edges by correlation direction* : si elle est cochée, les arêtes sont colorées selon le sens de la corrélation (bleu pour positive, rouge pour négative et gris pour non applicable) trouvée entre les deux variables. Le cas non applicable correspond au cas où au moins une des variables est qualitative.
- *Smooth edges* : si elle est cochée, on autorise les arêtes à se courber au besoin. Sinon, ils restent droits.
- *Dynamic nodes stabilization* : si elle est cochée, le graphe se repositionne selon l'algorithme de stabilisation de graphe, après chaque mouvement par l'utilisateur.
- *Re-stabilize* : bouton qui permet de re-stabiliser le graphe (centrage et redistribution des nœuds sur le plan de manière « quasi-optimale »)

Les informations générales sont :

- *nb. observations* : le nombre d'entrées dans l'ensemble de données
- *nb. variables* : le nombre de variables dans l'ensemble de données
- *nb. couples* : le nombre de couples
- *nb. current edges* : le nombre actuel d'arêtes tracées en fonction des seuils choisis

Clic sur un lien du graphe produit les éléments suivants :

Figure de résumé de lien (en bas à droite du graphe) Son type dépend de la nature du lien correspondant :

- variable quantitative vs variable quantitative : un nuage de point
- variable quantitative vs variable qualitative : un boxplot
- variable qualitative vs variable qualitative : non encore traité

Table de résumé de lien (sous les informations générales) Quel que soit le type de lien, les valeurs pour ce lien de tous les coefficients de corrélation inclus à l'appel de la fonction R sont affichées. Quand au moins une des variables est qualitative, seul le MaxNMI possède une valeur, les autres coefficients restent non applicables.

Clic sur un noeud du graphe produit les éléments suivants :

Figure de résumé de variable correspondante (en bas à gauche du graphe) Son type dépend de la nature de la variable correspondante :

- variable quantitative : un histogramme
- variable qualitative : un diagramme en barre

Table de résumé de la variable correspondante (sous les informations générales)
Son type dépend de la nature de la variable :

- variable quantitative : tableau contenant le min, le 1er quartile, la médiane, la moyenne, le 3ème quartile et le max de la variable.
- variable qualitative : tableau contenant la fréquence de chaque modalités de la variable.

Onglet « Tables »

Cet onglet permet d'afficher 2 tableaux :

- la matrice de corrélation correspond au coefficient de corrélation sélectionné
- le tableau décrivant le groupe auquel est affecté chaque variable selon le regroupement effectué.

L'option *Correlation coefficient* permet de choisir le coefficient de corrélation à considérer parmi ceux qui sont calculés au départ.

6.6 Conclusion

Ce chapitre traite de l'outil Linkspotter réalisé au cours des travaux et qui a servi dans les phases d'analyse exploratoire des autres parties de la thèse, en particulier le chapitre sur la prédiction instantanée de débit sur le réseau mobile. Cet outil fournit aux analystes des moyens, notamment graphiques, d'analyser les corrélations entre les variables. Il est produit sous forme de package R et est disponible sur internet [SAMBA, 2017a] pour les utilisateurs de ce langage. Nous avons aussi mis en place un site internet [SAMBA, 2017b] accessible à tous, permettant de le présenter et de fournir son guide d'utilisation. En juillet 2017, Linkspotter a fait l'objet d'une présentation aux *Sixièmes Rencontres R*¹.

1. Rencontres R : conférence sur le langage de programmation R (cf. <http://angletr2017.com/>, <https://angletr2017.sciencesconf.org/>)

Chapitre 7

Déploiement d'algorithmes de Data Science en tant que service

Sommaire

7.1 Travaux connexes	93
7.2 Motivations et contexte	93
7.2.1 Développement de la Data Science	93
7.2.2 Qu'appelle-t-on algorithme de Data Science?	94
7.2.3 Sur le déploiement des algorithmes de Data Science	95
7.2.4 Concepts utiles	96
7.3 Approche « Package & Containerize »	97
7.3.1 Approche	97
7.3.2 Recommandations complémentaires	98
7.3.3 Avantages	99
7.4 Implémentation avec Rapp & RappServer	99
7.4.1 Architecture	100
7.4.2 Coder l'application R	101
7.4.3 Installer Rapp & RappServer	101
7.4.4 Packager avec Rapp	102
7.4.5 Containeriser avec Rapp	102
7.4.6 Lancer un RappServer	103
7.4.7 Description de l'API de RappServer	104
7.4.8 Déployer	106
7.5 Discussion et Perspectives	111

La Data Science est une discipline récente. Le terme a été utilisé pour la première fois par CLEVELAND [2001]. Depuis, il s'est beaucoup démocratisé notamment à partir de 2012 où des dirigeants de grandes entreprises ont commencé à publiquement évoquer intérêt de la discipline. Par exemple, Hal Varian, économiste en chef chez Google, est connu pour avoir affirmé en 2009 que le métier de statisticien serait « le métier le plus sexy des dix prochaines années ». Depuis, la terminologie de « statisticien » a évolué vers celle de « Data Scientist », comme le préconisait déjà C.F. Jeff Wu en 1997 à sa conférence

inaugurale intitulée « Statistics = Data Science ? » pour sa nomination à la chaire H.C. Carver à l’Université du Michigan. Dans cette conférence, il a caractérisé le travail statistique comme une trilogie entre (*i*) la collecte, (*ii*) la modélisation et l’analyse de données, et (*iii*) la prise de décision.

En effet, la Data Science est aujourd’hui définie comme une discipline regroupant les compétences permettant d’accomplir cette trilogie. De plus en plus d’outils, accompagnés par l’essor du Big Data, enrichissent le bagage des Data Scientists. Des langages de programmation comme R, nés pour faciliter l’application de la statistique, se dotent de plus en plus d’extensions permettant d’effectuer des tâches différentes et complémentaires à la statistique et l’apprentissage automatique. Parallèlement, des langages de programmation généralistes, comme Python se dotent de plus en plus d’extensions permettant de faire de la statistique et de l’apprentissage automatique. Ainsi, certains langages de programmation, notamment ceux cités ci-dessus, se sont hissés comme les langages préférés des Data Scientists, du fait qu’il comportent les outils qui leur permettent d’accomplir leurs tâches habituelles. Contrairement au « statisticien » traditionnel, le Data Scientist est donc un développeur également. Cependant, il semble manquer encore aujourd’hui aux Data Scientists des outils et les approches qui leur permettent d’arriver jusqu’à l’industrialisation de leurs travaux. En effet, tout algorithme issu du travail des Data Scientists, qu’il soit destiné à apprendre des modèles à partir des données, à exécuter ces modèles, ou à fournir une visualisation à partir des données, est voué à être déployé dans un système d’information pour servir à des tiers (d’autres applications ou des humains) comme illustré sur les figures 7.1 et 7.2. Cette phase de déploiement ou d’industrialisation est habituellement laissée à la charge des équipes de développement et de gestion de système d’information. Aujourd’hui, avec l’émergence et la démocratisation du Cloud Computing [ZHANG et collab., 2010], on peut entrevoir d’intégrer un peu plus les compétences liées à ce domaine à la boîte à outils du Data Scientist. Ainsi, ce dernier qui aura développé un algorithme de Data Science tel qu’une API de prédiction ou une algorithme de visualisation sera aussi capable de le déployer facilement dans un Cloud, d’une manière qui garantit une certaine agilité, critère important aujourd’hui dans les systèmes d’information.

Pour répondre à cette problématique, nous avons proposé une approche qui a permis pendant ma thèse de déployer facilement les algorithmes que nous avons proposés tels que TOM et celui de la prédiction instantanée de débit. Nous avons implémenté cette approche dans des packages R que nous avons nommés *Rapp* et *RappServer*. *Rapp* offre des outils permettant d’encapsuler tout algorithme développé sous R dans un « conteneur » et de la publier dans un Cloud au travers d’une API réceptrice. *RappServer* permet d’instancier l’API réceptrice au sein du Cloud. Le principe de « conteneurisation » que nous utilisons, facilité par des technologies existantes telles que *Docker* [MERKEL, 2014], permet l’isolation de l’algorithme de Data Science, qui va bénéficier au sein de son conteneur de toutes les dépendances dont elle a besoin. L’approche comporte plusieurs autres bénéfices que nous verrons dans la suite du chapitre.

La première section de ce chapitre présente quelques travaux connexes que l’on peut considérer comme l’état de l’art sur ce sujet précis. Le deuxième section revient sur la motivation et le contexte qui donne naissance à cette proposition. Nous y définissons ce que nous appellons « algorithme de Data Science », ensuite nous y présentons les notions de d’API, de *conteneurisation* et de *livraison continue* qui sous-tendent la proposition. Dans la troisième section, l’approche *package & containerize* proposée est présentée. Dans la quatrième section, nous présentons les packages *Rapp* et *RappServer*. Enfin, la dernière section propose une conclusion, ainsi qu’une discussion des travaux et une vision sur les

perspectives sur ce sujet.

7.1 Travaux connexes

Dans la littérature, nous trouvons plusieurs travaux connexes au notre. En effet, certains chercheurs ont déjà depuis un certain temps le réflexe de conteneuriser leurs algorithmes surtout dans un but de reproductibilité et de portabilité à l'instar de **GURALNICK et collab.** [2014] et **FILGUEIRA et collab.** [2016]. De plus, comme les codes sources de leurs algorithmes sont le plus souvent ouverts, ils les publient sur la plateforme *The Docker Hub* [**DOCKER.INC.**, 2015] qui permet de les importer directement sous forme d'image Docker de base. Plusieurs travaux similaires peuvent être cités. Par exemple, **HOLDGRAF et collab.** [2017] décrit une utilisation des technologies Cloud pour faciliter l'enseignement de la Data Science. Ils évoquent l'intérêt d'une architecture basée sur le Cloud pour la pédagogie et la reproductibilité des algorithmes de Data Science. Le papier de **GYMREK et FARJOUN** [2016] a pour but de dresser des recommandations pour faire de la Data Science ouverte. Il se focalise sur la reproductibilité et mentionne, sans aller dans le détail, l'intérêt de conteneuriser les analyses en utilisant Docker. **VOGELSTEIN et collab.** [2016], dans un but de reproductibilité également, mentionnent aussi la possibilité de mettre ensemble toutes les dépendances nécessaires à un algorithme d'apprentissage dans un conteneur Docker, afin de pouvoir le lancer dans un Cloud. [**KACAMARGA et collab.**, 2015] décrivent la manière dont ils ont utilisé Docker pour bâtir leur plateforme de bio-informatique pouvant exécuter des algorithmes de Data Science. Toujours dans un but de reproductibilité, **CHIRIGATI et collab.** [2016] ont bâti un outil qui permet aux chercheurs qui publient dans les conférences SIGMOD¹ de packager facilement leur expérimentation en utilisant Docker afin que les relecteurs puissent la réimplémenter facilement. Cette démarche rejoint l'approche que nous proposons, à la différence qu'au delà de la reproductibilité, nous nous intéressons également à la mise en production automatisée des algorithmes, en l'occurrence celles de Data Science. Des travaux qui visent aussi à rendre la recherche scientifique plus reproductible aboutissent souvent à des plateformes permettant aux chercheurs de travailler en commun sur une même expérimentation et d'avoir un outil d'intégration continue qui automatise la validation et la génération des résultats expérimentaux.

Tous ces travaux récents justifient encore plus l'importance de notre démarche d'implémenter dans les outils habituels des Data Scientists, des notions qui facilitent la portabilité, l'intégration et la livraison continues de leurs développements. Notre contribution vise à définir une approche standard de déploiement des algorithmes de Data Science et à apporter des outils logiciels (*Rapp* et *RappServer*) qui facilitent l'appropriation de cette dernière.

7.2 Motivations et contexte

7.2.1 Développement de la Data Science

La Data Science a connu un essor important au cours des dernières années sur plusieurs aspects, encouragé par l'augmentation des capacités de calcul.

1. ACM Sigmod (The ACM Special Interest Group on Management of Data). URL : <https://sigmod.org/>

Sur le stockage, la lecture et l'écriture des données (SGBD)

Les systèmes de gestion de base de données ont évolué avec la naissance du NoSQL [[STO-NEBRAKER, 2010](#)] qui s'écarte du paradigme classique des bases relationnelles et propose des manières plus flexibles de stocker les données. Des gains importants sont observés sur la rapidité de lecture et d'écriture, et sur la capacité à passer à l'échelle.

Sur les algorithmes

Les algorithmes de Machine Learning se sont un peu plus démocratisés. De nouveaux algorithmes ont vu le jour avec de très bonnes performances sur des cas d'utilisation historiquement difficiles, tels que la reconnaissance d'images. La naissance et la performance de ces algorithmes sont surtout favorisées par l'augmentation des capacités de calcul.

Sur les systèmes de traitement des données

On a connu un développement du calcul distribué avec l'émergence des concepts de MapReduce [[DEAN et GHEMAWAT, 2008](#)], entre autres. On a aussi vu la naissance de systèmes de traitement tels que Hadoop [[APACHE, 2011](#)], Spark [[APACHE, 2014](#)], etc.

Sur les langages de programmation

Les langages dédiés à l'analyse de données, comme R, ont décuplé de nombre d'utilisateurs au cours des dernières années. Ces langages se développent plus rapidement et deviennent de plus en plus performants.

Le développement de la Data Science a pour conséquence le fait que de plus en plus de personnes se forment à cette discipline. Elle est même devenue un métier à part entière. On parle aujourd'hui de « Data Scientist ». Les Data Scientists sont formés pour maîtriser plusieurs éléments des chaînes de procédures qui permettent de valoriser des données. La finalité d'une valorisation de données peut être :

- la construction d'un modèle (prédictif, de prévision, de classification, de scoring, de regroupement, etc.)
- le déploiement du modèle
- la réalisation d'une étude
- la construction de moyens d'analyse (graphiques, visualisation, interface homme-machine de requêtage, etc.)
- le déploiement de ces moyens d'analyse.

7.2.2 Qu'appelle-t-on algorithme de Data Science ?

Dans ce document, le terme « algorithme de Data Science » fait référence à tout programme informatique destiné à remplir une fonction qui relève de la Data Science. Elle peut s'agir d'un apprentissage machine, de l'exécution d'un modèle appris, de la collecte de données à partir d'un source tiers, de la mise en forme ou de la visualisation de données. L'algorithme de Data Science se présente souvent sous la forme d'une fonction qui prend en entrées des paramètres et qui renvoie en sortie un résultat. Ce résultat peut être

celui d'une prédiction, d'une classification, d'un regroupement, d'un classement, d'une mise en forme de données, un modèle issu d'un apprentissage machine, le code source d'une visualisation (par exemple une image PNG ou un code HTML/Javascript), etc. Ce résultat est souvent destiné à une utilisation automatique par d'autres applications (Close Loop) ou à un analyste humain (Open Loop).

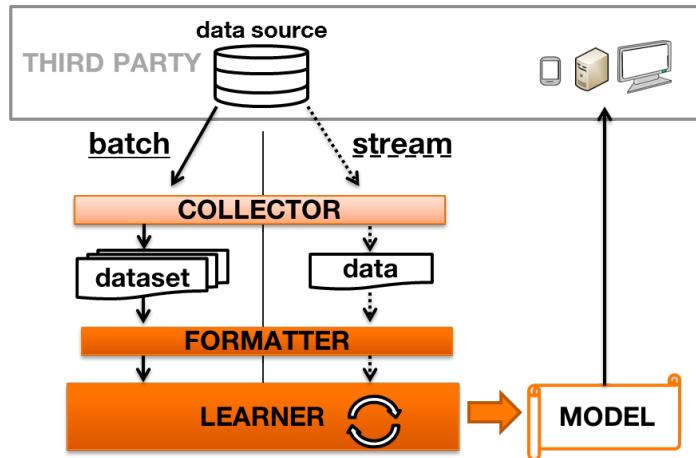


FIGURE 7.1 – Apprentissage de modèle

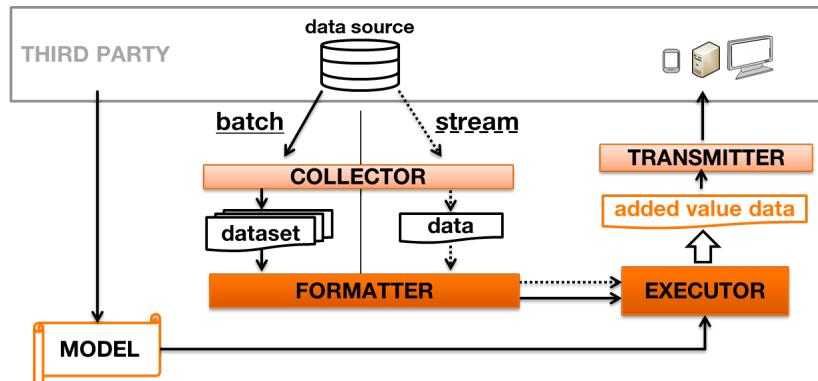


FIGURE 7.2 – Exécution de modèle

7.2.3 Sur le déploiement des algorithmes de Data Science

Au sujet du déploiement, nous estimons qu'il manque encore des éléments dans la boîte à outils du Data Scientist. En effet, au cours des travaux de cette thèse, s'est posée la question de déploiement des algorithmes proposés, tels que TOM et l'algorithme de prédiction instantanée du débit. La problématique est la suivante : Comment déployer les algorithmes de Data Science dans des systèmes d'information existant ? Comment les rendre accessibles à leurs utilisateurs finaux, notamment les applications qui consomment leurs résultats ? En sondant les habitudes, nous nous ronds compte qu'il y a principalement deux manières de faire :

La première consiste à se doter d'une plateforme dédiée dite « Big Data » (Oracle, Cloudera, etc.) qui utilisent généralement les environnements de calcul Hadoop [APACHE, 2011] et Spark [APACHE, 2014]. L'intérêt de ces plateformes dédiées est qu'elles fournissent

des logiciels permettant de déployer sous forme de tâche le script constituant l'algorithme de Data Science. Cependant, leur inconvénient est que les applications qui consomment les résultats de ce script doivent aller chercher son résultat sauvegardé sur la plateforme. Or cela nécessite de coder de manière spécifique chaque application dépendante.

La deuxième approche est de passer le relais aux développeurs purs. Il s'agit d'écrire les spécifications de l'algorithme afin qu'il soit entièrement recodé dans un langage habituellement utilisé pour le déploiement comme Java. Ainsi, l'algorithme peut être déployé de la même manière que les applications traditionnelles.

Il existe une troisième approche qui est celle que nous proposons consistant à faire entrer dans la culture des Data Scientists les notions de *DevOps* et de *Cloud*. Il s'agit de doter les langages de programmation préférés des Data Scientists, tels que R et Python, de plus en plus de fonctionnalités facilitant la mise en production de leurs algorithmes. En effet, ces derniers bénéficient aujourd'hui de plus en plus d'extensions qui vont dans ce sens. Beaucoup de concepts existants déjà dans le monde Cloud et DevOps, tels que les APIs Web, la conteneurisation et la notion de [Continuous Integration/Continuous Delivery \(CI/CD\)](#), facilitent cette démarche. De brèves descriptions de ces concepts sont fournies dans la sous-section suivante. Le principal avantage de l'approche que nous soutenons grâce à ce chapitre de la thèse est de renforcer l'*agilité* dans le déploiement des algorithmes de Data Science dans les systèmes d'information classiques. En effet, nous pourrons espérer des déploiements plus rapides, ainsi que la possibilité de mener une intégration et une livraison automatique et continue des algorithmes de Data Science.

7.2.4 Concepts utiles

Dans cette sous-section, est fourni un peu plus de détails sur quelques concepts inspirés des technologies Cloud et de la culture DevOps dont l'appropriation par les Data Scientists permettra d'amener vers eux plus d'agilité dans le déploiement de leurs algorithmes.

API

Comme décrit à la fin du chapitre 2, une API est un ensemble de services offert par un programme informatique à d'autres programmes informatiques. En effet, beaucoup de logiciels sont des APIs et/ou des clients d'API.

API Web

Un API Web est une forme d'API utilisant le standard HTTP pour l'interaction entre les programmes. Ils ont l'intérêt de fournir un vocabulaire standard et largement démocratisé, adapté à l'interaction entre des logiciels.

API sans état

Le terme « sans état » fait référence au fait que l'API transmet pour chaque requête une réponse qui ne dépend que de la requête et des paramètres de cette dernière. Chaque couple *requête-réponse* est traitée comme une transaction indépendante, sans lien avec les requêtes précédentes ou suivantes.

Conteneurisation

La conteneurisation est une alternative légère à la virtualisation [TURNBULL, 2014]. Elle consiste à encapsuler une application dans un conteneur avec son propre environnement d'exploitation. Des outils très développés aujourd'hui tels que Docker [MERKEL, 2014] permettent de mettre en œuvre la conteneurisation.

Livraison continue

La livraison continue est une approche permettant de produire des logiciels dans des cycles courts. Cela permet d'automatiser la mise à disposition d'un logiciel et de ses mises à jour à ses utilisateurs. Le but est de construire, tester et diffuser un logiciel plus rapidement. Elle aide à réduire le coût, le temps et les risques associés au déploiement du logiciel en adoptant une approche plus incrémentale des modifications en production. Des outils justement nommés de CI/CD (intégration continue et livraison continue) sont communément utilisés pour y arriver. Nous pouvons en citer *Travis CI* [TRAVIS.CI.COMMUNITY, 2011] et *Jenkins* [JENKINS.COMMUNITY, 2011].

7.3 Approche « Package & Containerize »

Les concepts présentés ci-dessus comportent plusieurs avantages en ce qui concerne le développement et le déploiement de logiciel. Le domaine de la Data Science pourrait également en bénéficier notamment pour le déploiement des algorithmes. Dans la section-ci nous détaillons l'approche de déploiement proposée, ainsi que ses avantages.

7.3.1 Approche

L'approche est constituée de trois étapes :

1. Coder l'algorithme de Data Science comme une API sans état. La nature d'un algorithme de Data Science telle que définie dans sous-section 7.2.2, lui donne par définition la capacité d'être implémenté comme une API sans état. En effet, le résultat de l'algorithme ne dépendra que des paramètres de la requête. Ainsi, deux requêtes identiques, n'auront des résultats différents que si les autres logiciels dont dépend l'algorithme donnent des résultats différents. Cela permet d'alléger la mémoire vive liée au processus de l'algorithme. Sous R, d'intéressants outils existants permettent de lancer un serveur exécutant du code R à l'instar de *httpuv* [CHENG et collab., 2017] et *Rserve* [URBANEK, 2013]. D'autres packages R à l'instar de *plumber* [ALLEN, 2017], *jug* [SMEETS, 2017] et *openCPU* [OOMS, 2014] s'appuient notamment sur *httpuv* pour fournir des cadres intéressants pour servir facilement un algorithme codé sous R comme une API web. Sous Python des outils équivalents existent à l'instar de *Flask* JOURNEY [2017].
2. Étape « Test » : elle consiste à tester le code pour s'assurer que l'algorithme fonctionne convenablement. Il existe souvent des outils pour l'automatiser. Par exemple, sous R, le package *testthat* [WICKHAM, 2011] permet de définir des test unitaires.
3. Étape « Package » : elle consiste à servir l'algorithme sous forme de package du langage de programmation utilisé, complété par une fonction qui permet de lancer l'API de l'algorithme. Cette étape est importante car elle permet de gérer facilement les dépendances vers d'autres packages existants. Cela permet également à

l'algorithme de bénéficier de tous les avantages d'un package, principalement le fait d'être facilement réutilisable. A l'issue de cette étape, on obtient un package dont l'on peut facilement compresser le code source sous forme d'un fichier qu'on peut facilement transporter et installer. Sous R, le package *devtools* [WICKHAM et CHANG, 2017] par exemple facilite cette étape.

4. Étape « Containerize » : elle consiste à créer un fichier descriptif qui permet la construction du conteneur qui lance l'API de l'algorithme automatiquement. Pour cela il faut s'appuyer sur une des solutions existantes de gestion de conteneur. *Docker* [MERKEL, 2014] est la plus utilisée aujourd'hui. Le fichier descriptif du conteneur accompagné du fichier compressé du code source du nouveau package qu'on obtient après cette étape sont tout ce qu'il faudra à la solution de gestion de conteneur pour déployer l'algorithme de Data Science dans un Cloud. Cette étape permet la gestion facile de l'algorithme de Data Science au sein du Cloud tel un conteneur classique. En particulier, la gestion des droits d'accès, l'équilibrage de charge et l'association à un nom de domaine, etc. peuvent être gérés de manière classique par le Cloud.

Les étapes (2) et (3) peuvent être automatisées dans un contexte de livraison continue de l'algorithme, notamment en utilisant un outil de CI/CD tel que *Travis CI* [TRAVIS.CI.COMMUNITY, 2011] ou *Jenkins* [JENKINS.COMMUNITY, 2011].

7.3.2 Recommandations complémentaires

Dans cette partie, nous listons quelques recommandations pour développer un algorithme de Data Science qui complètent l'approche.

Utilisation des langages de programmation appropriés

Les algorithmes de Data Science se développent plus facilement aujourd'hui qu'avant grâce à des langages de programmation qui se spécialisent de plus en plus sur le sujet, tels que R et Python. Du fait qu'ils comportent déjà beaucoup de packages dédiés aux Data Science - facilitant la collecte des données, leur mise en forme, leur apprentissage, leur visualisation, etc. - ces langages simplifient le développement d'algorithmes émanant de cette discipline.

Préférer le stockage dans une base de données

Utiliser les bases de données, quelque soit leurs types (NoSQL, SQL, orienté graphe, etc.) comme moyen de stockage permanent et à moyen terme des données permet de séparer les préoccupations. En effet, l'accessibilité, la sécurité et l'intégrité des données pourront être gérées grâce aux fonctionnalités dédiées de chaque système de gestion de base de données. Contrairement à un stockage direct sur le disque dur du serveur où l'algorithme fonctionne, l'utilisation d'une base de données externe permet en général une plus grande rapidité d'entrée-sortie. De plus, les langages de programmation adaptés à la Data Science comme R et Python disposent d'interfaces facilitant la communication avec la majorité des types de bases de données existant.

Libérer la mémoire vive

Il est aussi recommandé que le Data Scientist code ses algorithmes d'une manière qui charge le moins possible la mémoire vive de la machine exécutant le code. Cela permet de

gagner en rapidité et en économie de ressource RAM dans les Clouds où les algorithmes sont déployés.

Découper les algorithmes complexes en fonctions élémentaires réplifiables

Cela facilite les cas où les tâches peuvent être répliquées et distribué à plusieurs machines.

7.3.3 Avantages

Les avantages de l'étape « Package » sont : (*i*) une meilleure gestion des dépendances en bénéficiant du principe utilisé pour les packages R, (*ii*) une standardisation les programmes pour assurer leur portabilité.

Les avantages de l'étape « Containerize » sont : (*iii*) l'algorithme peut être déployé dans un système d'information sans se soucier de son système d'exploitation, pourvu qu'il utilise un gestionnaire de conteneur tel que Docker, (*iv*) l'algorithme bénéficie d'un environnement qui lui est dédié, (*v*) il est isolé du reste du système d'information et limite ainsi les risques de failles de sécurité sur ce dernier, (*vi*) il s'intègre au système d'information en tant que simple port et peut ainsi être géré comme tel, de manière classique, du point de vue de la sécurité et de l'accessibilité.

De plus, coder l'algorithme comme une API sans état permet (*vii*) d'utiliser une forme normalisée de communication entre les services, notamment HTTP, et (*viii*) de faciliter la réplicabilité de l'algorithme et la vérification de son résultat.

Enfin, l'approche dans sa globalité (*ix*) facilite la livraison continue des algorithmes de Data Science.

Dans cette section, l'approche proposée a été décrite et commentée, accompagnée de recommandations constituées d'un ensemble de bonnes pratiques pour développer efficacement des algorithmes de Data Science. Dans la section qui suit, nous décrivons l'implémentation qui a été faite de cette approche à travers des logiciels qui permettent d'automatiser ses différentes phases en allant jusqu'au déploiement de l'algorithme.

7.4 Implémentation avec Rapp & RappServer

L'approche proposée et décrite dans la section 7.3 est implémentée sous la forme de deux logiciels Rapp et RappServer. Ces derniers sont codés en langage R et livrés sous la forme de packages qui étendent ce langage.

Rapp permet de packager, conteneuriser et déployer les applications codées sous R avec *plumber* [ALLEN, 2017] ou *shiny* [CHANG et collab., 2017]. Le package plumber permet de transformer en API web tout algorithme écrit sous la forme d'une fonction en langage R. Le package shiny, en plus de l'algorithme, permet de développer en même temps une interface homme-machine. Dans cette section, une « application R » fait référence à l'un de ces deux types d'application.

Dans les sous-sections qui suivent sont décrits, d'abord (*i*) deux architectures de déploiement alternatives conformes à notre approche, ensuite (*ii*) comment coder proprement une application R, (*iii*) comment installer Rapp ou RappServer, (*iv*) comment packager une application R en utilisant Rapp, (*v*) comment conteneuriser une application R en utilisant Rapp et (*vi*) comment lancer un RappServer. Par la suite on fournit une (*vii*)

description de l'API receptrice de RappServer. Enfin, on décrit (*viii*) comment déployer une application R en utilisant différents moyens.

7.4.1 Architecture

Nous décrivons dans cette sous-section quelques cas d'utilisation alternatifs de l'approche présentés sur la figure 7.3 et la figure 7.4. Ces cas d'utilisations comprennent tous les deux les étapes de l'approche décrites dans la section 7.3.

Scénario 1 : Containerisation côté client

Dans ce scénario (cf. Figure 7.3), les étapes « Package » et « Containerize » de l'approche décrites dans la section 7.3 sont réalisées par l'utilisateur, c'est-à-dire « côté client ». Au niveau du serveur qui doit déployer l'application, une API réceptionne l'application paquettisée et conteneurisée pour le déployer convenablement.

L'automatisation de chacune de ces étapes faciliterait une démarche d'intégration et de livraison continues de l'application de Data Science comme illustré sur les figures 7.3 et 7.4.

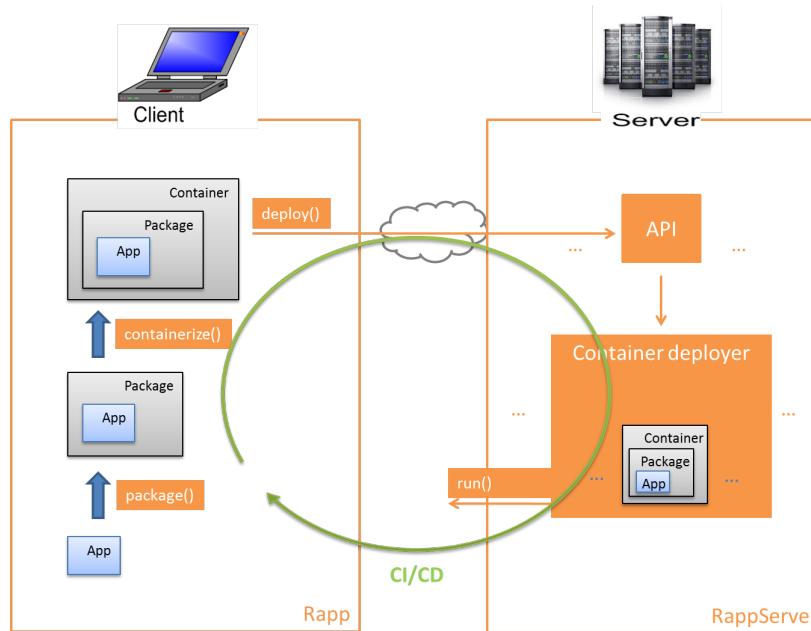


FIGURE 7.3 – Exemple d'utilisation de Rapp & RappServer - Scénario 1 : conteneurisation côté client

Scénario 2 : Containerisation côté serveur

Dans ce scénario (cf. Figure 7.4), toutes les étapes de l'approche proposée dans la section 7.3 se déroulent « côté serveur ». L'utilisateur envoie une requête à l'API receptrice comportant le code source brut de l'application à déployer (sous forme d'un fichier zip par exemple). A la réception de la requête, le serveur se charge à son tour, de packager, de conteneuriser et de déployer l'application.

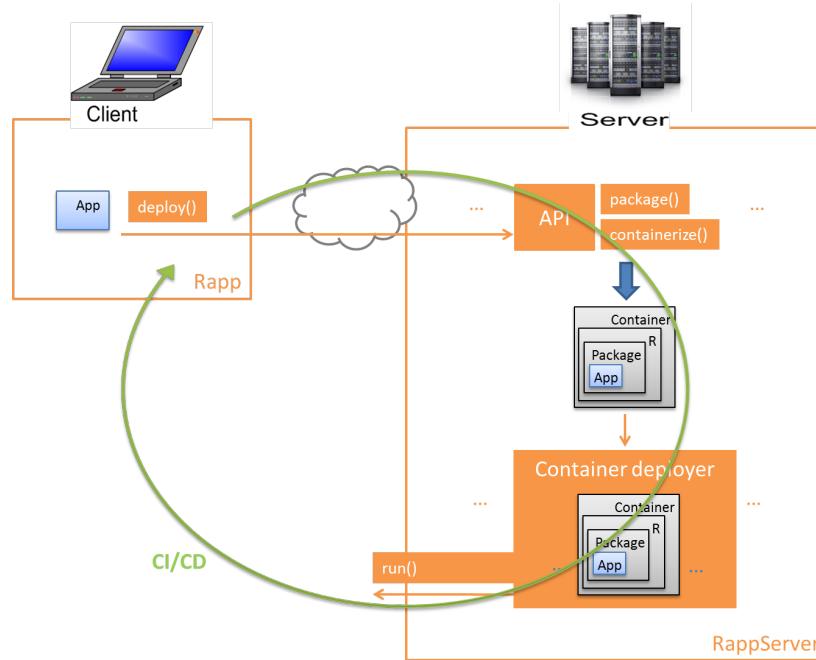


FIGURE 7.4 – Exemple d'utilisation de Rapp & RappServer - Scénario 2 : conteneurisation côté serveur

7.4.2 Coder l'application R

Cette sous-section liste les bonnes pratiques nécessaires au codage des applications R attendues par Rapp et RappServer. Elles complètent les recommandations de la sous-section 7.3.2.

Tout d'abord, cette première version de Rapp fonctionne uniquement avec les applications R développées avec les packages *pumber* et *shiny*. De plus, Les dépendances qu'on peut préciser à Rapp sont celles disponibles dans le CRAN.

Concernant le code en tant que tel, si le script R constituant le point d'entrée de l'application intègre un autre script, ce dernier doit se situer dans le même dossier que le point d'entrée ou, de préférence dans un sous-dossier de celui-ci. Il en est de même pour les éventuelles données chargées par l'application. Il est recommandé d'avoir respectivement des sous-dossiers *src/* et *data/* pour ces deux types de dépendance.

Par ailleurs, l'application doit être codée dans une perspective de déploiement dans une machine quelconque. Par exemple le pointage vers un fichier ou un dossier doit se faire en considérant que le répertoire de travail de R en déploiement sera le dossier où se trouve le script constituant le point d'entrée de l'application.

7.4.3 Installer Rapp & RappServer

Rapp et RappServer s'installent de la même manière comme tout package R, grâce notamment à la fonction *install.packages()*. Pour installer une version hébergée dans le CRAN de Rapp, on peut exécuter sous R la commande suivante :

```
> install.package('Rapp')
```

Pour l'installer à partir de son code source disponible sur le disque de l'utilisateur sous la forme du fichier compressé *Rapp_0.1.0.tar.gz*, on peut exécuter sous R la commande suivante :

```
> install.package('Rapp_0.1.0.tar.gz', repos=NULL)
```

RappServer doit être installé dans le serveur qui aura pour rôle de déployer les applications R qu'on lui envoie. Ce dernier doit au préalable comporter le logiciel Docker et un kernel Python dont dépend RappServer. Rapp quant à lui peut être installé sur tout client qui pourra alors packager, conteneuriser et envoyer des applications R sur un serveur où RappServer est lancé.

7.4.4 Packager avec Rapp

Une fois le package Rapp installé sur un client, on peut l'utiliser pour packager une application R. Packager une application R dans notre contexte consiste à la placer dans le dossier *inst/* du package - destiné aux codes sources extérieurs utiles au package - et à créer la fonction principale du package destinée à lancer l'application. La fonction *package()* permet d'automatiser cette procédure. On l'utilise de la manière suivante :

```
> Rapp::package(  
+   appFolder = "app/TOM/",  
+   type = "plumber",  
+   newPackageName = "TOM",  
+   appMainFileName = "app.R",  
+   dependenciesFromCRAN = NULL,  
+   defaultHost = '0.0.0.0',  
+   defaultPort = 8000,  
+   destinationPath = 'package/',  
+   checkBeforeBuilding = TRUE,  
+   keepPackageReleaseOnly = FALSE,  
+   quietly = FALSE,  
+   desc = " The description of the package.",  
+   title = "What the Package Does",  
+   version = "0.1.0",  
+   authors_at_R = "c(person('First','Last',  
+     role = c('aut','cre'),  
+     email = 'fl@mail.c'))",  
+   depends = "R (>= 3.3.2)",  
+   license = "GLP3",  
+   encoding = "UTF-8",  
+   lazydata = "true")
```

7.4.5 Containeriser avec Rapp

Une fois packagée, on peut conteneuriser une application conformément à l'approche proposée (cf. Section 7.3). La fonction de conteneurisation offerte par *Rapp* s'utilise de la manière suivante :

```
> Rapp::containerize(  
+   tarGzFile = "targz/TOM_0.1.0.tar.gz",  
+   packageName = "TOM",  
+   dependenciesFromCRAN = NULL,
```

```
+   sourceDockerImage = "sambaala/rapps",
+   host="0.0.0.0",
+   portToExpose = 8000,
+   maintainer = "c(person('First','Last',
+                         role = c('aut','cre'),
+                         email = 'fl@mail.c'))",
+   destinationPath = "cont/")
```

On peut aussi utiliser directement la fonction `package_and_containerize()` qui enchaîne les étapes « Package » et « Containerize » :

```
> Rapp::package_and_containerize(
+   appFolder = 'app/TOM/',
+   type = 'plumber',
+   newPackageName = 'TOM',
+   appMainFileName = 'app.R',
+   dependenciesFromCRAN = NULL,
+   sourceDockerImage = 'sambaala/rapps',
+   host='0.0.0.0',
+   portToExpose = 8000,
+   destinationPath = 'cont/',
+   checkBeforeBuilding = TRUE,
+   keepDockerFolderOnly = FALSE,
+   quietly = FALSE,
+   desc = " The description of the package.",
+   title = "What the Package Does",
+   version = "0.1.0",
+   authors_at_R = "c(person('First','Last',
+                           role = c('aut','cre'),
+                           email = 'fl@mail.c'))",
+   depends = "R (>= 3.3.2)",
+   license = "GLP3",
+   encoding = "UTF-8",
+   lazydata = "true")
```

7.4.6 Lancer un RappServer

Un RappServer est une API web capable de déployer une application R sur le serveur où il est installé. Il utilise pour ce faire l'approche que nous décrivons dans la section [7.3](#) et emploie Docker comme gestionnaire de conteneur. L'ordre de déploiement lui est donné grâce une simple requête POST.

Pour lancer l'API de RappServer sur un serveur de type Linux, il faut avoir installé R et le package RappServer. Ensuite, l'API se lance de la manière suivante sur la console *bash* :

```
$ R -e "RappServer::start(port=8002,hostName='http://<hostname>')" &
```

Le résultat de cette commande est la suivante, si elle est réussie.

```
Starting server to listen on port 8002
```

7.4.7 Description de l'API de RappServer

L'API de RappServer contient un seul endpoint « / », accessible en requête POST et GET. Le document JSON (Listing 7.1) ci-dessous décrit le type de requête attendu par l'API. La description s'appuie sur le standard Swagger version 2.0 [SMARTBEAR SOFTWARE, 2011].

Listing 7.1 – Description de l'API de RappServer

```

1  {
2      "swagger" : "2.0",
3      "info" : {
4          "description" : "RappServer API enabling R app
5              deployment",
6          "version" : "0.1",
7          "title" : "RappServer"
8      },
9      "host" : "hostname",
10     "schemes" : [ "http" ],
11     "produces" : [ "application/json" ],
12     "paths" : {
13         "/" : {
14             "post" : {
15                 "summary" : "R app deployer",
16                 "description" : "Deploy an R app",
17                 "consumes" : [ "multipart/form-data" ],
18                 "parameters" : [ {
19                     "name" : "deployMode",
20                     "in" : "formData",
21                     "description" : "a character string corresponding
22                         to 'deployFromZip' or 'deployFromDockerfile'
23                         according to the files used for the deployment
24                         ",
25                     "required" : true,
26                     "type" : "string",
27                     "enum" : [ "deployFromDockerfile", "deployFromZip
28                         " ]
29                 }, {
30                     "name" : "dockerfile",
31                     "in" : "formData",
32                     "description" : "the R app Dockerfile (required
33                         if deployMode is 'deployFromDockerfile')",
34                     "required" : false,
35                     "type" : "file"
36                 }, {
37                     "name" : "packagetargz",
38                     "in" : "formData",
39                     "description" : "the R app tar.gz file (required
40                         if deployMode is 'deployFromDockerfile')",
41                     "required" : false,
42                     "type" : "file"
43                 }
44             }
45         }
46     }
47 }
```

```

36     },
37     "name" : "uploadedzip",
38     "in" : "formData",
39     "description" : "the R app zip file (required if
40                     deployMode is 'deployFromZip')",
41     "required" : false,
42     "type" : "file"
43   },
44   {
45     "name" : "type",
46     "in" : "formData",
47     "description" : "a character string 'plumber' or
48                     'shiny' (required if deployMode is '
49                     'deployFromZip')",
50     "required" : false,
51     "type" : "string"
52   },
53   {
54     "name" : "appMainFileName",
55     "in" : "formData",
56     "description" : "a character string corresponding
57                     to the name of the main file of the Plumber
58                     API or Shiny App. For Shiny it will correspond
59                     to the 'app.R' file and for Plumber to the ,
60                     'plumber.R' file (required if deployMode is ,
61                     'deployFromZip')",
62     "required" : false,
63     "type" : "string",
64     "default" : "app.R"
65   },
66   {
67     "name" : "dependenciesFromCRAN",
68     "in" : "formData",
69     "description" : "a character string corresponding
70                     to the packages the Plumber API or Shiny App
71                     depends , separated by spaces (required if
72                     deployMode is 'deployFromZip')",
73     "required" : false,
74     "type" : "string"
75   ],
76   "responses" : {
77     "200" : {
78       "description" : "Successful request"
79     },
80     "default" : {
81       "description" : "Unexpected error"
82     }
83   }
84 },
85 "get" : {
86   "summary" : "RappServer GUI",
87   "description" : "Show the GUI of the RappServer",
88 }
```

```

74     "responses" : {
75         "200" : {
76             "description" : "Successful request"
77         },
78         "default" : {
79             "description" : "Unexpected error"
80         }
81     }
82 }
83 }
84 }
85 }
```

Requête POST

En requête POST, l'API attend des paramètres incluant le code source (fichier zip d'une part ou Dockerfile et package compressé en tar.gz d'autre part) de l'application R à déployer et différentes informations permettant d'effectuer et de personnaliser le déploiement.

Requête GET

En requête GET, l'API renvoie une page web correspondant à une interface graphique qui permet de déployer facilement les applications. En d'autres termes, elle permet d'effectuer les requêtes POST décrites ci-dessus, en passant par une interface graphique dédiée. L'utilisation de cette dernière sera décrite dans la sous-section [7.4.8](#) suivante.

Dès lors que RappServer fournit une API, on peut développer à souhait des interfaces graphiques qui s'appuient sur cette dernière avec plus ou moins de fonctionnalités.

7.4.8 Déployer

Déployer une application R consiste à la lancer dans un serveur où elle sera accessible à ses utilisateurs. Le package RappServer permet de déployer dans un serveur doté de Docker une API capable de recevoir d'une machine cliente le code source d'une application R - sous forme d'un fichier zip ou d'un Dockerfile accompagné du package compressé de l'application - et de la lancer dans le serveur où il est installé.

Déployer en utilisant l'interface graphique dédiée de RappServer

L'interface graphique (cf. figures [7.5](#), [7.7](#) et [7.6](#)), codée en HTML et Javascript, permet de requêter avec la méthode POST l'API de RappServer. Elle comporte deux rubriques constituée chacune d'un formulaire permettant de renseigner les paramètres de déploiement et de transférer les fichiers (zip, Dockerfile ou tar.gz) nécessaires vers le serveur. L'une des rubriques (cf. Figure [7.5](#)) permet de déployer une application à partir de son Dockerfile et l'autre (cf. Figure [7.7](#)) à partir de son fichier zip.

Déployer à partir d'un Dockerfile : ce mode correspond au Scénario 1 de déploiement décrit dans la sous-section [7.4.1](#) (cf. Figure [7.3](#)). L'utilisateur package et conteneurise d'abord

son application en local. Il obtient les fichiers *Dockerfile* et *tar.gz* qu'il peut dès lors transférer vers le serveur via le formulaire de l'interface pour le déploiement de l'application par le RappServer distant. Le résultat du déploiement, comportant l'URL attribuée à l'application, apparaît tel que sur la figure 7.6.

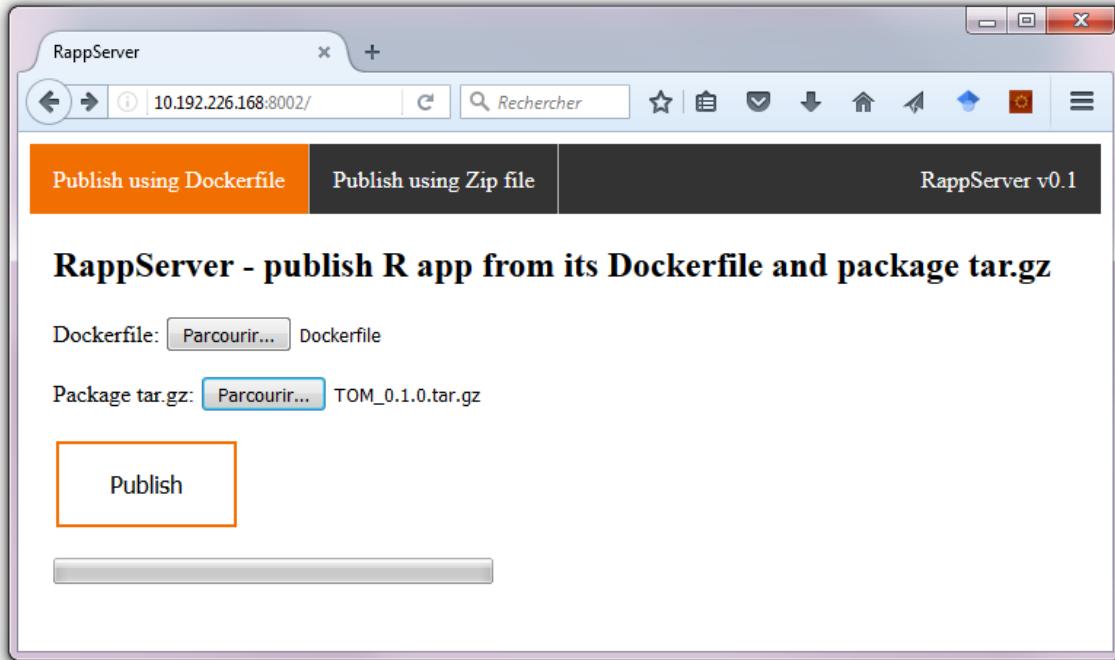


FIGURE 7.5 – Déploiement à partir d'un Dockerfile via l'interface graphique

Déployer à partir d'un fichier zip : ce mode correspond au Scénario 2 de déploiement décrit dans la sous-section 7.4.1 (cf. Figure 7.4). L'utilisateur transfère vers le serveur le fichier zip qu'il aura déjà constitué, contenant tous les fichiers et dossiers de l'application R grâce au formulaire. Il renseigne les paramètres tels que le type de l'application (plumber ou shiny) le nom du fichier principal de l'application ou en d'autres termes son point d'entrée, et enfin, le nom des packages qu'on peut installer à partir du CRAN [CRAN, 2017] dont dépend l'application. Le résultat est similaire à celle d'un déploiement à partir d'un Dockerfile (cf. Figure 7.6).

Déployer en utilisant Rapp

Le package Rapp comporte une fonction *deploy()* qui peut servir de client pour tout RappServer. Elle peut être utilisée de trois manières différents : (i) pour un déploiement de l'application R à partir d'un Dockerfile, (ii) à partir d'un fichier zip et (iii) à partir d'un dossier local comportant l'application R.

Déployer à partir d'un Dockerfile : La commande R de déploiement à partir d'un Dockerfile est la suivante :

```
> Rapp::deploy(  
+   type = "shiny",  
+   appDockerfile = "cont/example1_docker/Dockerfile",
```

```
+   appTarGz = "cont/example1_docker/example1_0.1.0.tar.gz",
+   url = "http://10.192.226.168:8002")
```

Le paramètre *url* représente l’URL du RappServer qui doit être à l’écoute au préalable. Cette première manière d’utiliser la fonction *deploy()* nécessite d’avoir d’abord packagé et conteneurisé l’application. Dans l’exemple, le Dockerfile et le package compressé obtenus sont stockés dans le dossier *cont/example1_docker*.

Déployer à partir d’un fichier zip : Le fichier zip doit comporter à sa racine le fichier R constituant le point d’entrée de l’application. La commande R de déploiement à partir du fichier zip est la suivante :

```
> Rapp::deploy(
+   type = "plumber",
+   appZip = "zip/TOM.zip",
+   url = "http://10.192.226.168:8002")
```

Cette commande nécessite d’avoir d’abord compressé tous les fichiers et dossiers composant l’application R (plumber ou shiny) sous forme d’un fichier zip. Dans l’exemple, le fichier zip *TOM.zip* est contenu dans le dossier *zip*.

Déployer à partir d’un dossier local : Dans ce cas Rapp se charge de compresser les fichiers et dossiers constituant l’application avant de l’envoyer à l’API du RappServer. La commande R de déploiement à partir du dossier contenant l’application R en local est :

```
> Rapp::deploy(
+   type = "shiny",
+   appFolder = "app/example1",
+   url = "http://10.192.226.168:8002")
```

Dans l’exemple, tous les fichiers et dossiers de l’application R sont contenus dans le dossier *app/example1*.

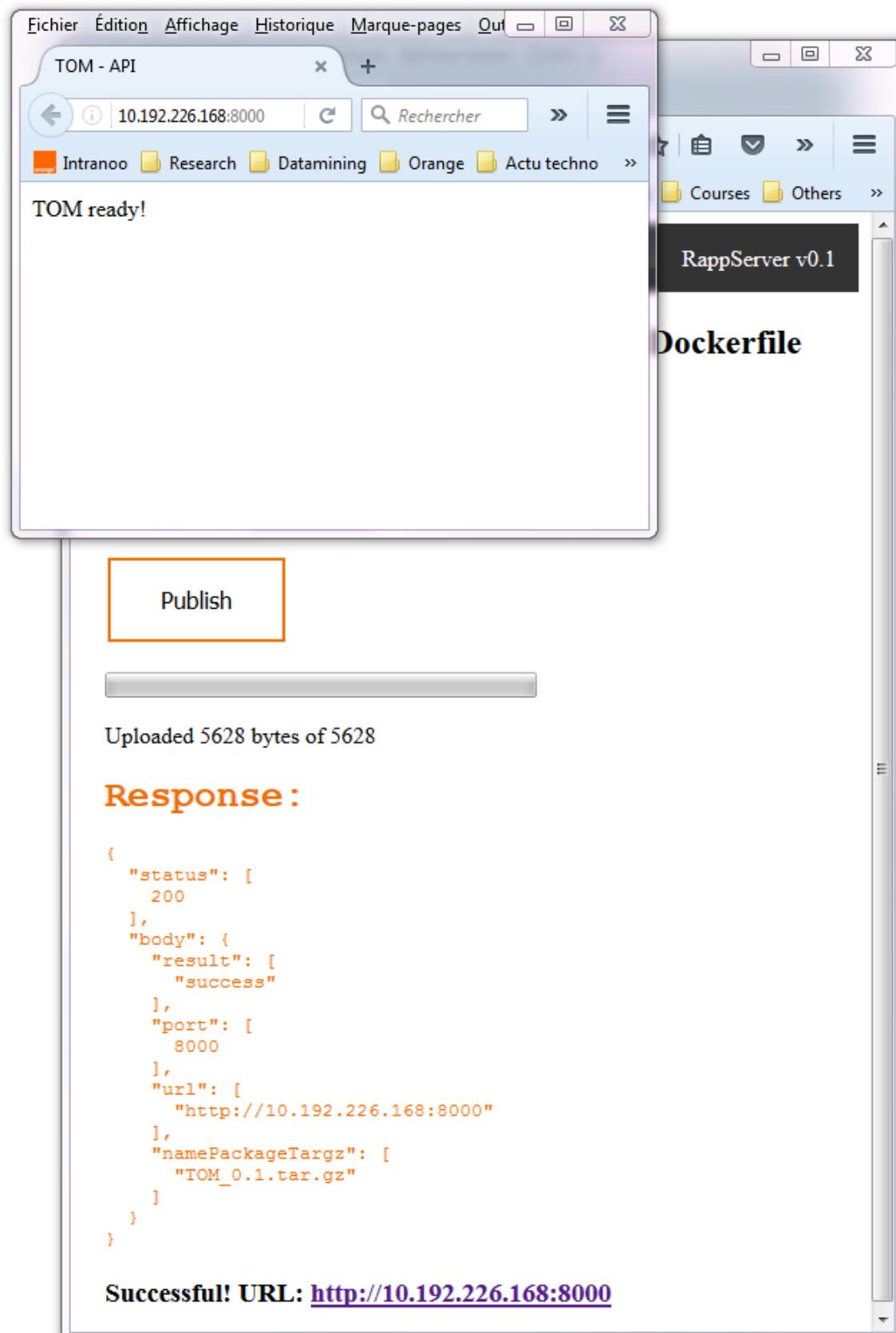


FIGURE 7.6 – Résultat de déploiement via l'interface graphique

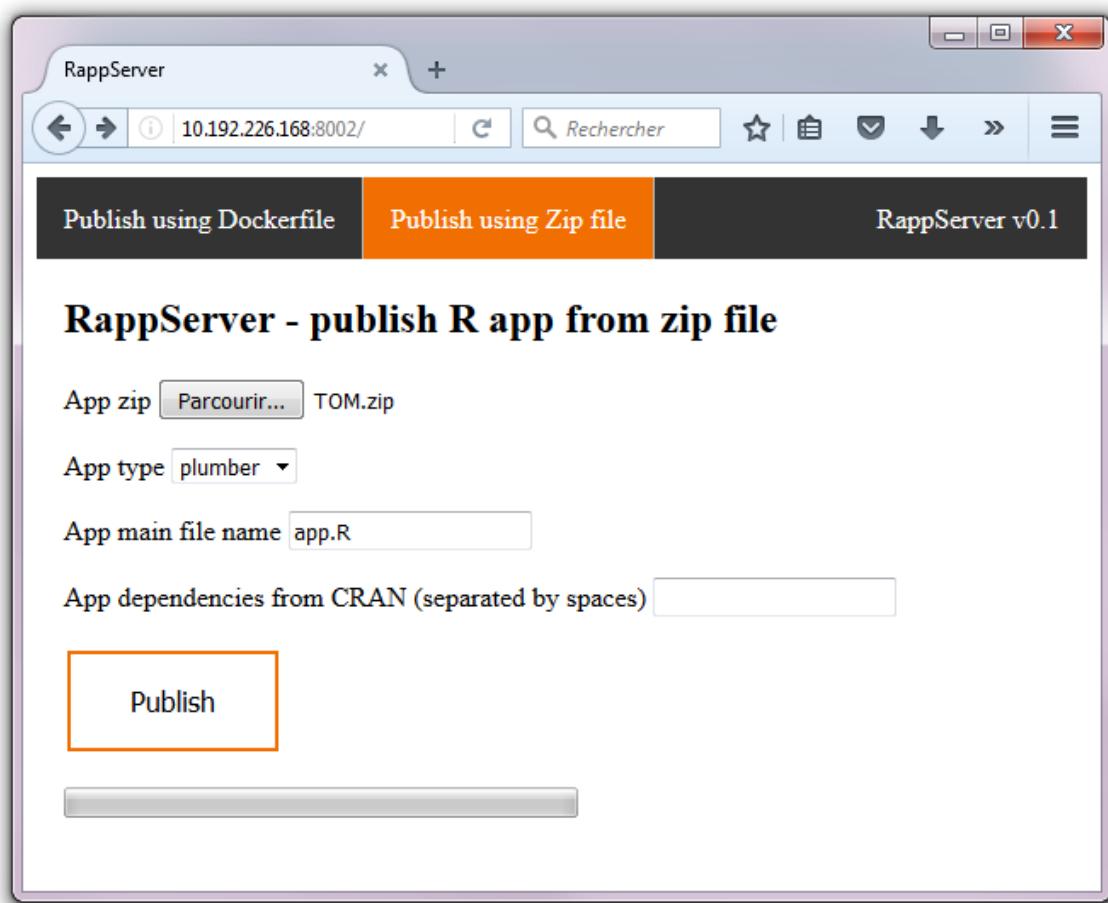


FIGURE 7.7 – Déploiement à partir d'un zip via l'interface graphique

7.5 Discussion et Perspectives

Dans cette section de conclusion, nous pointons sur les éléments de l'approche et son implémentation qui pourraient être améliorés, et dégagons des pistes de réflexion et de développement.

Les travaux connexes trouvés dans la littérature portent essentiellement sur la reproducibilité des analyses et modèles produits par les Data Scientists. Des plateformes sont souvent proposées pour faciliter le partage de code et la portabilité des travaux. Sur le sujet précis du déploiement d'algorithmes de Data Science, l'approche claire que nous proposons et le fait que nous fournissons des moyens d'utiliser et d'automatiser cette dernière dans un langage tel que R largement utilisé par les Data Scientists, pourraient avoir une incidence positive sur la manière dont les algorithmes de Data Science se développent, se déplacent et interagissent.

Néanmoins, nous notons déjà quelques pistes d'amélioration qui constitueront la poursuite de ce travail. D'abord, l'étape de test de l'approche n'est pas implémentée à l'heure actuelle dans le logiciel Rapp. Aujourd'hui, l'utilisateur doit respecter au mieux les recommandations et les bonnes pratiques, et s'assurer du bon fonctionnement de son algorithme avant de le soumettre pour un déploiement. Dans une version ultérieure, une batterie de tests tout au long du processus pourra être prévue. Cela permettra d'informer l'utilisateur des éventuels dysfonctionnements à corriger. Par ailleurs, l'implémentation que nous proposons qui est adaptée aux algorithmes et applications R codées avec *plumber* [ALLEN, 2017] et *shiny* [CHANG et collab., 2017], peut être généralisée à d'autres types d'application voire d'autres langages de programmation.

Une autre piste de réflexion est constituée par la possibilité de définir précisément et de standardiser les différentes tâches qui composent les activités de Data Science. L'idée est d'arriver à fournir par exemple un « apprentissage supervisé en tant que service » ou un « clustering en tant que service » quels que soient les modèles utilisés, avec des APIs unifiées et standards. Ce sera dans un but d'apporter plus d'interopérabilité entre les outils et les langages utilisés dans ce domaine. La démarche qui a donné naissance au *Predictive Model Markup Language (PMML)* [THE DATA MINING GROUP, 2008] est une bonne entrée en matière sur ce sujet, pour le cas des modèles d'apprentissage supervisé.

Chapitre 8

Conclusion

Sommaire

8.1 Bilan	113
8.2 Perspective : vers une QoS prévisible pour les automobiles connectés	114

Dans ce chapitre de conclusion, nous établissons un bilan de nos travaux de thèse et leurs retombées et donnons une vision des perspectives à travers la présentation succincte d'un nouveau travail entamé élargissant un des sujets de la thèse.

8.1 Bilan

Cette thèse a constitué un passionnant travail qui avait pour objectif d'évaluer la pertinence de l'utilisation des techniques issues du monde de la science des données dans la gestion des réseaux d'opérateur. Grâce à deux cas d'utilisation, à savoir la prédiction de débit instantanée sur le réseau mobile et l'analyse automatique des résultats de test d'infrastructure de réseau virtualisé, nous avons pu concevoir et utiliser des algorithmes et approches innovants pouvant apporter une plus-value dans la gestion des réseaux d'opérateur. Une volonté de partage et de vulgarisation de nos méthodologies et approches nous ont poussés à développer également des outils, qui ont été utiles à nos cas d'utilisation, et qui pourront à l'avenir être utiles à d'autres travaux. C'est le cas de Linkpotter, Rapp & RappServer, ainsi que TOM.

Nos travaux présentés dans le chapitre 4, portant sur la prédiction de débit ont fait émerger plusieurs idées qui sont indirectement liées à celle-ci et qui ont donné lieu à un brevet que nous avons déposé portant sur la mise à jour des modèles prédictifs embarqués par les terminaux mobiles [DOOZE et collab., 2016] et une demande de brevet, aujourd'hui en cours, sur le choix d'interface de réseau mobile respectueux de l'efficacité énergétique et de la qualité de service, en collaboration avec des collègues spécialistes sur les aspects énergétiques des réseaux. Les résultats de nos travaux sur la prédiction de débit ont aussi été utilisés pour réaliser une application mobile nommée Calireso [ORANGE SERVICES SRL, 2018] qui est aujourd'hui mise à disposition des clients de l'opérateur mobile Orange en France, afin qu'ils puissent avoir une estimation de leur qualité de service accessible en temps réel, selon leur situation géographique. L'application exploite les données radio collectées du terminal du client. Par ailleurs, le code source du modèle pré-

dictif final a été publié dans le cadre du projet européen CogNet [MOSCHITTI et collab., 2017; TYMOSHENKO et collab., 2016].

Nos travaux portant sur l'analyse automatisée des résultats de tests de réseau virtualisé, présentés dans le chapitre 5, ont donné naissance à l'outil d'analyse TOM. Ce dernier s'est décliné sous plusieurs formes à savoir, à ce jour, un package R, une API [SAMBA, 2018b] déployée et une application shiny [SAMBA, 2018c] déployée également. Ces travaux ont aussi fait en partie l'objet d'une présentation invitée DEBEAU et SAMBA [2017] effectuée par Eric Debeau à l'[ONAP Mini Summit](#)¹, en juin 2017. Les travaux se poursuivent sur ce sujet pour étendre la logique d'analyse à la prise en compte simultanée de plusieurs métriques de test.

L'outil Linkspotter issu de nos travaux présentés dans le chapitre 6 a été publié sous forme de package R [SAMBA, 2017a] et a fait l'objet d'une présentation aux *Sixièmes Rencontres R*² en juillet 2017.

L'approche *Package & Containerize* que nous avons proposée dans le chapitre 7 s'est révélée être précurseur d'une vision qui a donné naissance au projet open source Acumos [[LINUX FOUNDATION, 2018](#)] lancé en 2018, auquel nous participons pour le compte d'Orange. Ce projet vise à accroître les moyens techniques à disposition des industriels pour déployer des algorithmes d'intelligence artificielle dans des Clouds. L'approche et les outils issues de notre thèse sur ce domaine sont en droite ligne avec les ambitions du projet. Notre participation à ce dernier permet d'y intégrer nos propositions afin de les partager avec toute la communauté intéressée. Dans un autre registre, une partie de nos propositions sur ce sujet ont fait l'objet de notre présentation [SAMBA, 2018a] aux *Sep tièmes Rencontres R* en juillet 2018, dans la session « R extensions ».

Par ailleurs, de nouveaux travaux pouvant être considérés comme une continuité de la thèse ont démarré au sein d'Orange Labs que j'ai par la suite intégré en tant qu'ingénieur de recherche, affecté à l'équipe qui m'a accueilli en [Convention Industrielle de Formation par la REcherche \(CIFRE\)](#) au cours de la thèse. Ces nouveaux sujets dans lesquels je me suis engagé sont en partie alimentés par mes résultats de thèse. La présentation succincte de l'un d'entre eux constitue le but de la section de perspective suivante.

8.2 Perspective : vers une QoS prévisible pour les automobiles connectés

Nos travaux sur la prédiction de débit sur le réseau mobile ont intéressé des collègues d'Orange qui sont déjà intégrés au [5G Automotive Association \(5GAA\)](#). Il s'agit d'une organisation internationale et interprofessionnelle regroupant des entreprises des secteurs de l'automobile, de la technologie et des télécommunications, qui travaillent ensemble pour développer des solutions de bout en bout pour les futurs services de mobilité et de transport. Nos travaux ont été présentés aux réunions plénières et nous participons à la rédaction du livrable pourtant sur la « qualité de service prédictible ».

L'intérêt de pouvoir prédire la qualité de service sur le réseau mobile afin d'assurer la connectivité des automobiles s'est manifesté dans les travaux de l'association. Nous envisageons alors d'élargir l'approche qui nous a permis de construire des modèles de prédiction de débit (cf. chapitre 4) à la construction de modèles prédictifs de la qualité

1. ONAP Mini Summit 2017 : <http://events17.linuxfoundation.org/events/opnfv-summit/extend-the-experience/onap>

2. Rencontres R : conférence portant sur le langage de programmation R (cf. <http://angletr2017.com/>, <https://angletr2017.sciencesconf.org/>)

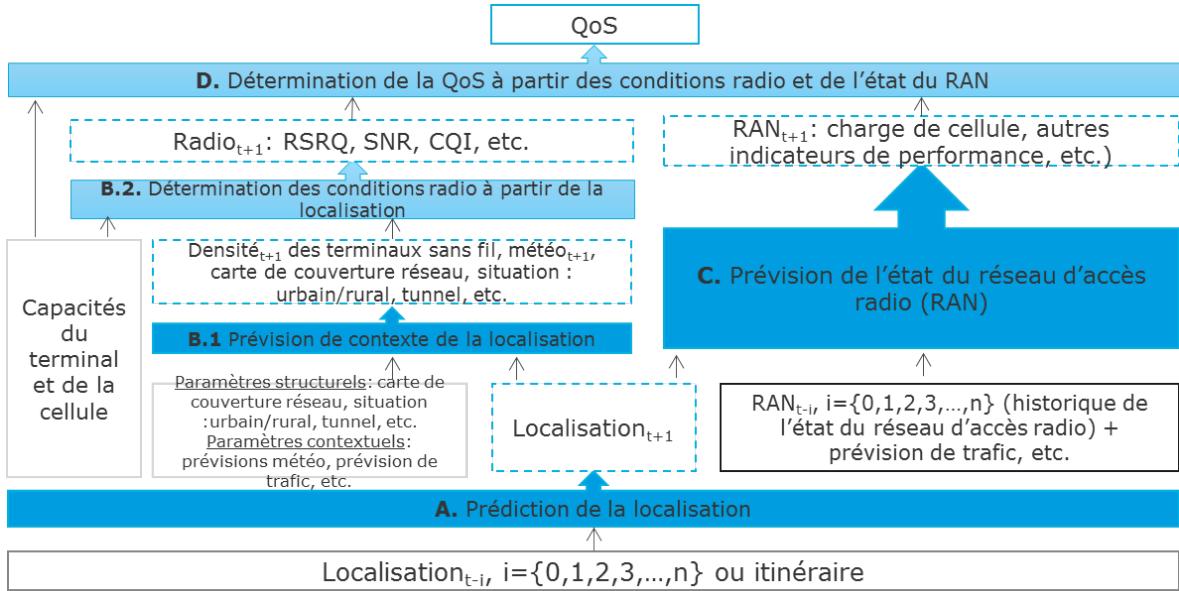


FIGURE 8.1 – Vue générale de l'approche de prédiction spatio-temporelle de la QoS

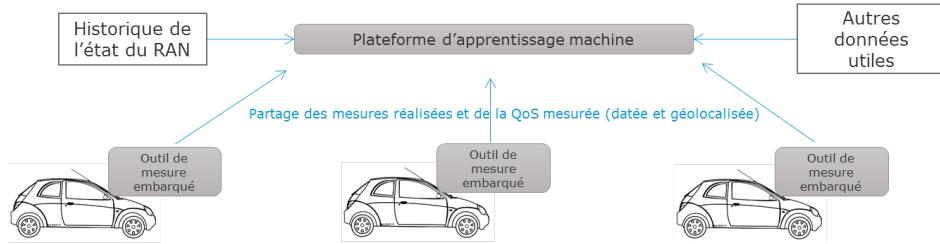


FIGURE 8.2 – Premier scénario d'apprentissage

de service, en intégrant la dimension spatio-temporelle à la prédiction. Nous avons déjà imaginé l'approche généralisée que nous illustrons sur la figure 8.1 et envisageons les architectures potentielles illustrées sur les figures 8.2 et 8.3.

En effet, la prédiction spatio-temporelle de la qualité de service peut être découpé en plusieurs blocs imbriqués et interdépendants comme illustré sur la figure 8.1, constituant les étapes qui nous paraissent nécessaires pour arriver à inclure dans la prédiction les deux dimensions espace et temps. Ces blocs sont les suivants :

Prédiction de la localisation

La localisation du véhicule connecté peut être prédite grâce à un modèle de trajet [BARTH et collab., 2012], exploitant l'historique de localisation, la vitesse, etc. Elle peut aussi être simplement déduite dans le cas où l'itinéraire du véhicule est connu grâce à son GPS par exemple.

Prévision des conditions radio

Les conditions radio expérimentées par le terminal mobile au temps t dépendent des facteurs d'interférence et de bruit qui affectent le terminal au temps t (contexte de l'emplacement : densité de terminaux mobiles, météo, couverture réseau, situation : urbaine/rurale, tunnel, etc.), et des capacités du terminal et de la cellule à laquelle il est

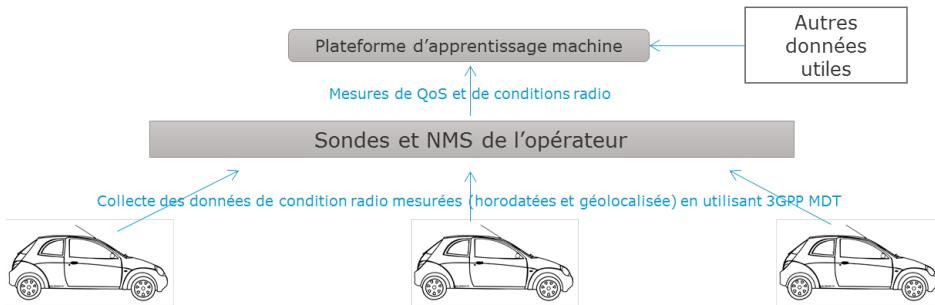


FIGURE 8.3 – Second scénario d'apprentissage

connecté. Ce bloc peut être subdivisé en deux, selon la méthodologie utilisée. Il s'agira d'avoir d'abord une prédiction du contexte de l'emplacement et ensuite, la détermination des conditions radio selon ce contexte. Un modèle issue de l'apprentissage de mesures collectées grâce à [MDT \[ETSI, 2016\]](#) ou un crowdsourcing pourront servir dans ce cadre.

Prévision de l'état du RAN

La littérature regorge de travaux intéressants sur le sujet de la prédiction de l'état du réseau d'accès comme en témoignent [BUI et collab. \[2017\]](#). Nous avons lancé un stage au sein d'Orange qui vise à renforcer ce type de travaux au sein de notre équipe, qui feront partie de l'approche globale permettant d'arriver à une prédictibilité spatio-temporelle de la qualité de service. Il s'agit d'introduire et d'adapter des modèles d'apprentissage de séries temporelles telles que le modèle [Seasonal AutoRegressive Integrated Moving Average \(SARIMA\)](#) et le modèle [Vector AutoRegression \(VAR\)](#) qui semblent convenir au cas étudié.

Détermination de la QoS à partir des conditions radio et de l'état du RAN

Cette étape consiste en l'exécution d'un modèle issu d'un apprentissage supervisé de la [QoS](#) en fonction des conditions radio et de l'état du [RAN](#). En effet, comme nous l'avons montré dans le chapitre 4, la [QoS](#) accessible à un terminal en temps réel dépend surtout de ces deux paramètres. Il s'agit donc d'élargir nos propositions du chapitre 4 à une notion plus globale de [QoS](#), incluant la latence, le taux de perte de paquet, la gigue, etc.

L'intégration de tous ces modèles évoqués permettra d'arriver à une prévisibilité dans le temps et dans l'espace de la qualité de service pour les automobiles connectés. C'est un besoin de plus en plus exprimé par leurs constructeurs et qui donnera aux automobiles connectés et leurs fournisseurs de contenus multimédias la capacité d'avoir un comportement proactif face aux changements de qualité de connexion qu'ils peuvent subir durant un trajet.

Liste des figures

3.1 Illustration de l'évolution de la taille de la fenêtre TCP à ses différentes phases	26
3.2 Vue générale d'un réseau d'opérateur	27
3.3 Les différents types de QoS	32
3.4 Architecture NFV	34
3.5 Vue globale de l'architecture CogNet	39
4.1 Architecture générale des campagnes de collecte de données	50
4.2 Cumulative distribution function of the file download time	51
4.3 Débit moyen en fonction de quelques facteurs	55
4.4 Débit moyen en fonction des données radio	56
4.5 Débit moyen en fonction des données de contexte	56
4.6 Débit moyen en fonction des données de RAN	56
4.7 Le processus d'apprentissage supervisé	58
4.8 Fonction empirique de distribution cumulative du taux d'erreur	59
4.9 Débit moyen prédit versus débit moyen réel	60
4.10 Implementation strategies. a : client-based pull delivery, b : server-based push delivery	62
5.1 Diagramme de dépendance entre les composants	71
5.2 Diagramme de séquence de l'API TOM en déploiement	72
5.3 Résultats de test de bande passante mémoire selon le scénario de déploiement	73
5.4 Résultats de test de bande passante mémoire selon le PoD	74
5.5 Résultats de test de bande passante mémoire selon l'installeur	75
5.6 Résultats de test de bande passante mémoire selon le scénario de déploiement et le PoD	75
5.7 Résultats de test de bande passante mémoire selon le scénario de déploiement et l'installeur	76
5.8 Résultats de test de bande passante mémoire selon le PoD et l'installeur	76
5.9 Résultats de test de bande passante mémoire selon le scénario de déploiement, le PoD et l'installeur	77
5.10 Capacité explicative de la bande passante mémoire par les paramètres de configuration choisis	77
6.1 Graphe de corrélation obtenu sur le fichier de données Iris	87
7.1 Apprentissage de modèle	95
7.2 Exécution de modèle	95
7.3 Exemple d'utilisation de Rapp & RappServer - Scénario 1 : conteneurisation côté client	100

7.4 Exemple d'utilisation de Rapp & RappServer - Scénario 2 : conteneurisation côté serveur	101
7.5 Déploiement à partir d'un Dockerfile via l'interface graphique	107
7.6 Résultat de déploiement via l'interface graphique	109
7.7 Déploiement à partir d'un zip via l'interface graphique	110
8.1 Vue générale de l'approche de prédition spatio-temporelle de la QoS	115
8.2 Premier scénario d'apprentissage	115
8.3 Second scénario d'apprentissage	116

Liste des tableaux

2.1	Example de matrice de confusion	17
3.1	Modèle OSI	23
4.1	Résumé du fichier de données	49
4.2	Les dix schémas de disponibilité de données les plus fréquents	53
4.3	Corrélation entre le débit et les autres variables collectées	55
4.4	Précision de la prédiction selon les données prises en entrée. B est le prédicteur de référence incluant comme variables explicatives la bande de fréquence de la cellule et la catégorie du terminal	59
5.1	Modèle à 1 variable explicative de la bande passante mémoire	74
5.2	Modèle à plusieurs variables explicatives de la bande passante mémoire . .	75

Liste des acronymes

- 3GPP** 3rd Generation Partnership Project. [30](#)
- 5GAA** 5G Automotive Association. [114](#)
- ACP** Analyse en Composantes Principales. [78](#)
- ANOVA** Analyse de la variance. [85](#)
- API** Application Programming Interface. [19, 38, 48, 51, 61, 62, 65, 70, 71, 114](#)
- ARCEP** Autorité de Régulation des Communications Electroniques et des Postes. [26](#)
- ARTP** Autorité de Régulation des Télécommunications et des Postes. [26](#)
- AS** Autonomous System. [24](#)
- BeEF** Best Equal-Frequency. [7, 84, 85, 86](#)
- BLER** Block Error Rate. [52](#)
- CA** Congestion Avoidance. [25](#)
- CDN** Content Delivery Network. [28, 34, 60, 61](#)
- CI/CD** Continuous Integration/Continuous Delivery. [96, 97, 98](#)
- CIFRE** Convention Industrielle de Formation par la REcherche. [114](#)
- COGITO** One Cognitive Operation. [3](#)
- CogNet** Cognitive Networks. [3](#)
- CQI** Channel Quality Indicator. [29, 47, 51](#)
- CRAN** Comprehensive R Archive Network. [5, 6](#)
- DANE** DASH-Aware Network Element. [62](#)
- DASH** Dynamic Adaptive Streaming over HTTP. [62](#)
- DCAE** Data Collection, Analytics, and Events. [67](#)
- DSL** Digital Subscriber Line. [27](#)
- ECDF** Empirical Cumulative Distribution Function. [58](#)
- EMS** Element Management System. [34, 35](#)
- EPC** Evolved Packet Core. [35](#)
- ETSI** European Telecommunications Standards Institute. [30, 36](#)
- FTTx** Fiber to the Home/Building/Desktop/etc.. [27](#)
- GANA** Generic Autonomic Networking Architecture. [36](#)
- GPS** Global Positioning System. [45, 52, 115](#)

- GSM** Global System for Mobile Communications. [1](#)
- HMM** Hidden Markov Model. [46](#)
- HSS** Home Subscriber Server. [35](#)
- IETF** Internet Engineering Task Force. [23, 30, 31, 38](#)
- IMS** IP multimedia subsystem. [28, 34](#)
- IoT** Internet of Things. [29](#)
- IP** Internet Protocol. [22, 23, 24, 25, 28](#)
- LM** Linear model. [57, 59](#)
- LTE** Long Term Evolution. [1, 51](#)
- MAC** Media Access Control. [23](#)
- MANO** Management and Orchestration. [34, 35, 38, 67](#)
- MAPE-K** Monitor, Analyse, Plan, Execute, Knowledgee. [36](#)
- MaxNMI** Maximal Normalized Mutual Information. [7, 54, 84, 86](#)
- MDT** Minimization of Drive Tests. [29, 30, 61, 63, 115](#)
- MIC** Maximal Information Coefficient. [54, 84](#)
- NFV** Network Function Virtualization. [1, 37, 65, 66, 67, 78](#)
- NFVI** Network Function Virtualization Infrastructure. [34, 35](#)
- NFVO** Network Function Virtualization Orchestration. [35](#)
- NMS** Network Management System. [28, 45, 48, 52](#)
- NN** Artificial Neural Networks. [57](#)
- ONAP** Open Network Automation Platform. [66, 67, 114](#)
- OPNFV** Open Platform for NFV. [6, 66, 67, 72](#)
- OSI** Open Systems Interconnection. [22, 23, 24, 25, 27, 30](#)
- PLS-GLM** Partial Least Square Generalized Linear Model. [57, 59](#)
- PMML** Predictive Model Markup Language. [111](#)
- PoD** Point of Delivery. [72, 73, 74, 78](#)
- PRB** Physical Resource Block. [47](#)
- QoE** Qualité d'Expérience. [31](#)
- QoS** Qualité de Service. [28, 30, 31, 32, 36, 116](#)
- QXDM** Qualcomm eXtensible Diagnostic Monitor. [51](#)
- RAN** Radio Access Network. [27, 47, 48, 55, 116](#)
- RF** Random Forests. [57, 59, 60](#)
- RRC** Radio Resource Control. [52](#)
- RSRP** Reference Signal Received Power. [29, 51](#)
- RSRQ** Reference Signal Received Quality. [29, 47, 51](#)

RSSI Received Signal Strength Indicator. [29](#), [51](#)

RTT Round Trip Time. [33](#), [44](#), [45](#)

SAND Server and Network Assisted DASH. [62](#)

SARIMA Seasonal AutoRegressive Integrated Moving Average. [116](#)

SINR Signal-to-interference-plus-noise ratio. [29](#), [51](#)

SIP Session Initiation Protocol. [28](#)

SLA Service-Level Agreement. [31](#), [38](#)

SNR Signal-to-noise ratio. [29](#)

SON Self Organizing Networks. [36](#)

SS Slow Start. [25](#)

TCP Transmission Internet Protocol. [22](#), [23](#), [24](#), [25](#), [32](#), [33](#), [47](#), [48](#), [50](#)

UDP User Datagram Protocol. [22](#), [23](#), [24](#), [25](#), [32](#)

UIT-T Union Internationale des Telecommunications - Secteur de la normalisation des télécommunications. [30](#), [31](#)

VAR Vector AutoRegression. [116](#)

VIM Virtual Infrastructure Manager. [35](#)

VNF Virtual Network Function. [34](#), [35](#)

VNFM Virtual Network Function Manager. [35](#)

Références

- 3GPP. 2016, «Technical specification group radio access network; evolved universal terrestrial radio access (e-utra); user equipment (ue) radio access capabilities (release 14)», *Technical Specification*, vol. 36. [49](#)
- ADHIKARI, V. K., Y. GUO, F. HAO, M. VARVELLO, V. HILT, M. STEINER et Z.-L. ZHANG. 2012, «Unreeling netflix : Understanding and improving multi-cdn movie delivery», dans *IEEE INFOCOM*. [61](#)
- ALLEN, J. 2017, *plumber : An API Generator for R*. URL <https://CRAN.R-project.org/package=plumber>. [97](#), [99](#), [111](#)
- APACHE. 2011, *Apache Hadoop : open-source software for reliable, scalable, distributed computing*. URL <http://hadoop.apache.org>. [94](#), [95](#)
- APACHE. 2014, *Apache Spark : Lightning-fast cluster computing*. URL <https://spark.apache.org>. [94](#), [95](#)
- BARTH, D., S. BELLAHSENE et L. KLOUL. 2012, «Combining local and global profiles for mobility prediction in lte femtocells», dans *Proceedings of the 15th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems*, ACM, p. 333–342. [115](#)
- BATES, T., P. SMITH et G. HUSTON. 2018, *Cidr report*. URL <http://www.cidr-report.org/as2.0/>. [24](#)
- BEBIS, G. et M. GEORGIOPoulos. 1994, «Feed-forward neural networks», *IEEE Potentials*, vol. 13, n° 4, p. 27–31. [16](#)
- BRADEN, R. 1989, «Requirements for internet hosts-communication layers», . [23](#)
- BREIMAN, L. 2001a, «Random forests», *Machine learning*, vol. 45, n° 1, p. 5–32. [15](#)
- BREIMAN, L. 2001b, «Random forests», *Machine learning*, vol. 45, n° 1, p. 5–32. [57](#)
- BUI, N., M. CESANA, S. A. HOSSEINI, Q. LIAO, I. MALANCHINI et J. WIDMER. 2017, «A survey of anticipatory mobile networking : Context-based classification, prediction methodologies, and optimization techniques», *IEEE Communications Surveys & Tutorials*, vol. 19, n° 3, p. 1790–1821. [116](#)
- BUI, N., F. MICHELINAKIS et J. WIDMER. 2014, «A model for throughput prediction for mobile users», dans *European Wireless Conference*. [46](#)
- BUUREN, S. et K. GROOTHUIS-OUDSHOORN. 2011, «mice : Multivariate imputation by chained equations in r», *Journal of statistical software*, vol. 45, n° 3. [54](#)

- CARDWELL, N., S. SAVAGE et T. ANDERSON. 2000, «Modeling tcp latency», dans *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, IEEE, p. 1742–1751. 46
- CHANG, W., J. CHENG, J. ALLAIRE, Y. XIE et J. MCPHERSON. 2017, *shiny : Web Application Framework for R*. URL <https://CRAN.R-project.org/package=shiny>, r package version 1.0.5. 71, 99, 111
- CHARRAD, M., N. GHAZZALI, V. BOITEAU et A. NIKNAFS. 2014, «NbClust : An R package for determining the relevant number of clusters in a data set», *Journal of Statistical Software*, vol. 61, n° 6, p. 1–36. URL <http://www.jstatsoft.org/v61/i06/>. 14
- CHEN, J., R. MAHINDRA, M. A. KHOJASTEPOUR, S. RANGARAJAN et M. CHIANG. 2013, «A scheduling framework for adaptive video delivery over cellular networks», dans *ACM MobiCom*. 44
- CHENG, J., H. C. BRAVO, J. OOMS et W. CHANG. 2017, *httpuv : HTTP and WebSocket Server Library*. URL <https://CRAN.R-project.org/package=httpuv>, r package version 1.3.5. 97
- CHIOSI, M., D. CLARKE, P. WILLIS, A. REID, J. FEGER, M. BUGENHAGEN, W. KHAN, M. FAR-GANO, C. CUI, H. DENG et collab.. 2012, «Network functions virtualisation : An introduction, benefits, enablers, challenges and call for action», dans *SDN and OpenFlow World Congress*, p. 22–24. 34
- CHIRIGATI, F., R. RAMPIN, D. SHASHA et J. FREIRE. 2016, «Reprozip : Computational reproducibility with ease», dans *Proceedings of the 2016 International Conference on Management of Data*, ACM, p. 2085–2088. 93
- CLARK, W. P. 1884, *The Indian Sign Language : With Brief Explanatory Notes of the Gestures Taught Deaf-mutes in Our Institutions for Their Instruction and a Description of Some of the Peculiar Laws, Customs, Myths, Superstitions, Ways of Living, Code of Peace and War Signals of Our Aborigines*, Philadelphia, Pa. : LR Hamersly & Company. 22
- CLEVELAND, W. S. 2001, «Data science : an action plan for expanding the technical areas of the field of statistics», *International statistical review*, vol. 69, n° 1, p. 21–26. 91
- COFANO, G., L. DE CICCO, T. ZINNER, A. NGUYEN-NGOC, P. TRAN-GIA et S. MASCOLO. 2016, «Design and experimental evaluation of network-assisted strategies for HTTP adaptive streaming», dans *MMSys*. 62
- CRAN. 2017, CRAN <https://cran.r-project.org/>. URL <https://cran.r-project.org/>. 107
- CRAWLEY, E., H. SANDICK, R. NAIR et B. RAJAGOPALAN. 1998, «A framework for qos-based routing in the internet», . 31
- DEAN, J. et S. GHEMAWAT. 2008, «Mapreduce : simplified data processing on large clusters», *Communications of the ACM*, vol. 51, n° 1, p. 107–113. 94
- DEBEAU, E. et A. SAMBA. 2017, *ONAP and OPNFV Collaboration : from Infrastructure to Machine-Learning*. URL https://wiki.onap.org/download/attachments/13598894/opnfv_summit_2017_opnfv_onap_collaboration.pdf. 114

RÉFÉRENCES

- DOCKER.INC. 2015, *The Docker Hub*. URL <https://hub.docker.com>. 93
- DOOZE, P., A. SAMBA et E. DEBEAU. 2016, «Procédé et dispositif d'actualisation d'un modèle prédictif d'une variable relative à un terminal mobile», European Patent, WO2018087491. 113
- EFRON, B. et R. J. TIBSHIRANI. 1994, *An introduction to the bootstrap*, CRC press. 16
- ETSI. 1994, «General aspects of quality of service (qos) and network performance (np)», *ETSI Network Aspects*. 31
- ETSI. 2000, «Digital cellular telecommunications system (phase 2+) ; universal mobile telecommunications system (umts) ; lte ; ip multimedia subsystem (ims) ; stage 2», *ETSI TS 23.228 v1.0.0*. 28
- ETSI. 2013a, «Network Functions Virtualisation (NFV) ; Architectural Framework», *GS NFV 002 V1.1.1 (2013-10)*. 34, 67
- ETSI. 2013b, «Network Functions Virtualisation (NFV) ; Use Cases», *GS NFV 001 V1.1.1 (2013-10)*. 34
- ETSI. 2014, «Speech and multimedia transmission quality (stq) ; quality of experience ; a monitoring architecture», *ETSI TS 103 294 V1.1.1*. 31
- ETSI. 2016, «Digital cellular telecommunications system (phase 2+) ; universal mobile telecommunications system (umts) ; telecommunication management ; subscriber and equipment trace ; trace data definition and management», *3GPP TS 32.423 version 13.0.0 Release 13*. 30, 61, 116
- ETSI. 2016, «Network Functions Virtualisation (NFV) ; Pre-deployment Testing ; Report on Validation of NFV Environments and Services », *GS NFV-TST 001 V1.1.1*. 67
- FANO, R. M. et D. HAWKINS. 1961, «Transmission of information : A statistical theory of communications», *American Journal of Physics*, vol. 29, n° 11, p. 793–794. 54
- FILGUEIRA, R., R. F. DA SILVA, A. KRAUSE, E. DEELMAN et M. ATKINSON. 2016, «Asterism : Pegasus and dispel4py hybrid workflows for data-intensive science», dans *Data-Intensive Computing in the Clouds (DataCloud), 2016 Seventh International Workshop on*, IEEE, p. 1–8. 93
- FISHER, R. et M. MARSHALL. 1936, «Iris data set», *RA Fisher, UC Irvine Machine Learning Repository*, vol. 440. 87
- FRALEY, C. et A. E. RAFTERY. 2002, «Model-based clustering, discriminant analysis, and density estimation», *Journal of the American statistical Association*, vol. 97, n° 458, p. 611–631. 87
- FRIGGE, M., D. C. HOAGLIN et B. IGLEWICZ. 1989, «Some implementations of the box-plot», *The American Statistician*, vol. 43, n° 1, p. 50–54. 69
- GELADI, P. et B. R. KOWALSKI. 1986, «Partial least-squares regression : a tutorial», *Analytica chimica acta*, vol. 185, p. 1–17. 15

- GURALNICK, R., T. CONLIN, J. DECK, B. STUCKY et N. CELLINESE. 2014, «The trouble with triplets in biodiversity informatics : A data-driven case against current identifier practices», *PLoS ONE*, vol. 9, n° 12, p. e114 069. [93](#)
- GYMREK, M. et Y. FARJOUN. 2016, «Recommendations for open data science», *GigaScience*, vol. 5, n° 1, p. 22. [93](#)
- HAHN, S., D. GOTZ, S. LOHMULLER, L. SCHMELZ, A. EISENBLÄTTER et T. KÜRNER. 2015, «Classification of cells based on mobile network context information for the management of SON systems», dans *IEEE VTC Spring*. [52](#)
- HAVA MUNTEAN, C. et J. McMANIS. 2006, «Fine grained content-based adaptation mechanism for providing high end-user quality of experience with adaptive hypermedia systems», dans *ACM Proc. of the 15th Int. Conf. on World Wide Web WWW*, p. 53–62. [44](#)
- HE, Q., C. DOVROLIS et M. AMMAR. 2005, «On the predictability of large transfer tcp throughput», dans *ACM SIGCOMM Computer Communication Review*, vol. 35, ACM, p. 145–156. [46](#)
- HERZOG, G. 1945, «Drum-signaling in a west african tribe», *Word*, vol. 1, n° 3, p. 217–238. [22](#)
- HOCKING, R. R. 1976, «A biometrics invited paper. the analysis and selection of variables in linear regression», *Biometrics*, vol. 32, n° 1, p. 1–49. [69](#)
- HOLDGRAF, C., A. CULICH, A. ROKEM, F. DENIZ, M. ALEGRO et D. USHIZIMA. 2017, «Portable learning environments for hands-on computational instruction : Using container-and cloud-based technology to teach data science», dans *Proceedings of the Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact*, ACM, p. 32. [93](#)
- HU, X., X. LI, E. C. H. NGAI, V. C. M. LEUNG et P. KRUCHTEN. 2014, «Multidimensional context-aware social network architecture for mobile crowdsensing», *IEEE Communications Mag.*, vol. 52, n° 6, p. 78–87. [52](#)
- HUANG, T.-Y., R. JOHARI, N. MCKEOWN, M. TRUNNELL et M. WATSON. 2015, «A buffer-based approach to rate adaptation : Evidence from a large video streaming service», *ACM SIGCOMM Computer Communication Review*, vol. 44, n° 4, p. 187–198. [44, 46](#)
- IBM et collab.. 2005, «An architectural blueprint for autonomic computing», *IBM White paper*. [35, 36](#)
- ISO. 1994, «Information technology—open systems interconnection—basic reference model : The basic model», *ISO standard ISO/IEC 7498-1*, p. 7498–1. [22](#)
- ITU-T. 2001, «Communications quality of service : A framework and definitions», *Recommendation G.1000*. [31](#)
- ITU-T. 2008, «Recommendation e800 : Definitions of terms related to quality of service», *International Telecommunication Union's Telecommunication Standardization Sector (ITU-T) Std.* [31](#)
- JENKINS.COMMUNITY. 2011, *Jenkins : Continuous Integration and Continuous Delivery*. URL <https://jenkins.io>. [97, 98](#)

- JIANG, J., X. LIU, V. SEKAR, I. STOICA et H. ZHANG. 2014, «EONA : Experience-Oriented Network Architecture», dans *ACM HotNet*. [62](#)
- JIANG, J., V. SEKAR et H. ZHANG. 2012, «Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive», dans *Proceedings of the 8th international conference on Emerging networking experiments and technologies*, ACM, p. 97–108. [46](#)
- JIANG, Z. M., A. E. HASSAN, G. HAMANN et P. FLORA. 2009, «Automated performance analysis of load tests», dans *Software Maintenance, 2009. ICSM 2009. IEEE International Conference on*, IEEE, p. 125–134. [78](#)
- JURNEY, R. 2017, *Agile Data Science 2.0 : Building Full-stack Data Analytics Applications with Spark*, " O'Reilly Media, Inc.", 70–72 p.. [97](#)
- KACAMARGA, M. F., B. PARDAMEAN et H. WIJAYA. 2015, «Lightweight virtualization in cloud computing for research», dans *International Conference on Soft Computing, Intelligence Systems, and Information Technology*, Springer, p. 439–445. [93](#)
- KAMPSTRA, P. et collab.. 2008, «Beanplot : A boxplot alternative for visual comparison of distributions», . [69](#)
- KINNEY, J. B. et G. S. ATWAL. 2014, «Equitability, mutual information, and the maximal information coefficient», *Proceedings of the National Academy of Sciences*, vol. 111, n° 9, p. 3354–3359. [86](#)
- LEDERER, S., C. MÜLLER, C. TIMMERER, C. CONCOLATO, J. L. FEUVRE et K. FLIEGEL. 2013, «Distributed DASH dataset», dans *Proc. of ACM Multimedia Systems Conference (MM-Sys)*, p. 131–135. [47](#)
- LI, Z., X. ZHU, J. GAHM, R. PAN, H. HU, A. C. BEGEN et D. ORAN. 2014, «Probe and adapt : Rate adaptation for http video streaming at scale», *IEEE Journal on Selected Areas in Communications*, vol. 32, n° 4, p. 719–733. [46](#)
- LIAW, A. et M. WIENER. 2002, «Classification and regression by randomforest», *R News*, vol. 2, n° 3, p. 18–22. URL <http://CRAN.R-project.org/doc/Rnews/>. [58](#)
- LINUX FOUNDATION. 2018, *Acumos : Making Artificial Intelligence Accessible To Everyone*. URL <https://www.acumos.org/>. [114](#)
- LIU, J., G. SIMON, C. ROSENBERG et G. TEXIER. 2014, «Optimal delivery of rate-adaptive streams in underprovisioned networks», *IEEE Journal on Selected Areas in Communications*. [61](#)
- LU, F., H. DU, A. JAIN, G. M. VOELKER, A. C. SNOEREN et A. TERZIS. 2015, «CQIC : Revisiting cross-layer congestion control for cellular networks», dans *ACM HotMobile Workshop*. [44, 47, 48, 51](#)
- MACQUEEN, J. et collab.. 1967, «Some methods for classification and analysis of multivariate observations», dans *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, Oakland, CA, USA, p. 281–297. [13, 87](#)
- MALIK, H. 2010, «A methodology to support load test analysis», dans *Software Engineering, 2010 ACM/IEEE 32nd International Conference on*, vol. 2, IEEE, p. 421–424. [78](#)

- MATHIS, M., J. SEMKE, J. MAHDAVI et T. OTT. 1997, «The macroscopic behavior of the TCP congestion avoidance algorithm», *SIGCOMM Comput. Commun. Rev.*, vol. 27, n° 3. [46](#)
- MCCULLAGH, P. 1984, «Generalized linear models», *European Journal of Operational Research*, vol. 16, n° 3, p. 285–292. [57](#)
- MERIEM, T. B., R. CHAPARADZA, B. RADIER, S. SOULHI et A. P. LÓPEZ. 2016, «Gana-generic autonomic networking architecture», . [37](#)
- MERKEL, D. 2014, «Docker : lightweight linux containers for consistent development and deployment», *Linux Journal*, vol. 2014, n° 239, p. 2. [5, 92, 97, 98](#)
- MEVIK, B.-H., R. WEHRENS et K. H. LILAND. 2013, *pls : Partial Least Squares and Principal Component regression*. URL <https://CRAN.R-project.org/package=pls>, r package version 2.4-3. [58](#)
- MIRZA, M., J. SOMMERS, P. BARFORD et X. ZHU. 2007, «A machine learning approach to TCP throughput prediction», dans *ACM Sigmetrics conference*. [46, 60](#)
- MOK, R. K. P., W. LI et R. K. C. CHANG. 2016, «Irate : Initial video bitrate selection system for HTTP streaming», *IEEE Journal on Selected Areas in Communications*, vol. 34, n° 6, p. 1914–1928. [44, 46](#)
- MOSCHITTI, A., A. PASSERINI, F. GRANELLI, O. URYUPINA, I. HAPONCHYK, D. BONADIMAN, A. ABAD, L. CHEN, H. ASSEM, T. S. BUDA, B. ÇAĞLAYAN, L. XU, B. ORDOZGOITI, J. R. SÁNCHEZ-COUSO, A. MOZO, H. ARREGUI, E. LOYO, I. SEGALL, M. GOLDSTEIN, Y. BEN SLIMEN, S. ALLIO, J. JACQUES, M. MICHEL, S. SENECAL, V. GROLLEMUND, A. SAMBA et P. DOOZE. 2017, «D3.3-Feature and Structure modeling, Structured Input/Output, Unsupervised Learning and Domain Adaptation (Accompanying Document for the Engineering Release)», URL http://www.cognet.5g-ppp.eu/wp-content/uploads/2017/05/CogNet_D33_Final.pdf. [63, 114](#)
- ONAP. 2017, *ONAP : Open Network Automation Platform*. URL <https://www.onap.org>. [67](#)
- OOMS, J. 2014, «The opencpu system : Towards a universal interface for scientific computing through separation of concerns», *arXiv preprint arXiv:1406.4806*. [97](#)
- OPNFV. 2014, *OPNFV: Open Platform for NFV*. URL <https://www.opnfv.org>. [4, 66](#)
- OPNFV. 2016, *QTIP Project*. URL <https://github.com/opnfv/qtip>. [79](#)
- ORANGE SERVICES SRL. 2018, *Calireso*. URL <https://play.google.com/store/apps/details?id=com.orange.labs.calireso>. [63, 113](#)
- PADHYE, J., V. FIROIU, D. F. TOWSLEY et J. F. KUROSE. 2000, «Modeling TCP reno performance : A simple model and its empirical validation», *IEEE/ACM Trans. Netw.*, vol. 8, n° 2, doi :10.1109/90.842137, p. 133–145, ISSN 1063-6692. URL <http://dx.doi.org/10.1109/90.842137>. [46](#)
- PAN, W. 2001, «Akaike's information criterion in generalized estimating equations», *Biometrics*, vol. 57, n° 1, p. 120–125. [17](#)

- PELAY, J., F. GUILLEMIN et O. BARAIS. 2017, «Verifying the configuration of virtualized network functions in software defined networks», dans *Network Function Virtualization and Software Defined Networks (NFV-SDN), 2017 IEEE Conference on*, IEEE, p. 223–228. [67](#)
- POSTEL, J. 1980, «User datagram protocol. RFC 768, Internet Engineering Task Force», *IETF*. [25](#)
- POSTEL, J. 1981, «Transmission control protocol. RFC 793, Internet Engineering Task Force», *IETF*. [25](#)
- PRASAD, R., C. DOVROLIS, M. MURRAY et K. CLAFFY. 2003, «Bandwidth estimation : metrics, measurement techniques, and tools», *IEEE Network*, vol. 17, n° 6, doi :10.1109/MNET.2003.1248658, p. 27–35, ISSN 0890-8044. [46](#)
- R CORE TEAM. 2016, *R : A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria. URL <https://www.R-project.org/>. [19, 54, 58](#)
- REN, F. et C. LIN. 2011, «Modeling and improving TCP performance over cellular link with variable bandwidth», *IEEE Transactions on Mobile Computing*, vol. 10, n° 8, doi :10.1109/TMC.2010.234, p. 1057–1070, ISSN 1536-1233. [46](#)
- RESHEF, D. N., Y. A. RESHEF, H. K. FINUCANE, S. R. GROSSMAN, G. MCVEAN, P. J. TURNBAUGH, E. S. LANDER, M. MITZENMACHER et P. C. SABETI. 2011, «Detecting novel associations in large data sets», *science*, vol. 334, n° 6062, p. 1518–1524. [54, 84, 85, 86](#)
- RUSSELL, S. J. et P. NORVIG. 2010, *Artificial Intelligence : A Modern Approach*, Prentice Hall, viii p.. [2](#)
- SAID, S. B. H., M. R. SAMA, K. GUILLOUARD, L. SUCIU, G. SIMON, X. LAGRANGE et J.-M. BONNIN. 2013, «New Control Plane in 3GPP LTE/EPC Architecture for On-Demand Connectivity Service», dans *IEEE CloudNet*. [61](#)
- SAMBA, A. 2017a, *Linkspotter*. URL <https://github.com/sambaala/linkspotter>. [54, 89, 114](#)
- SAMBA, A. 2017b, *Linkspotter : Correlation Analysis and Visualization*. URL <https://linkspotter.sigmant.net/>. [89](#)
- SAMBA, A. 2018a, *Construction et déploiement d'applications web basées sur R*. URL <https://r2018-rennes.sciencesconf.org/204125>. [114](#)
- SAMBA, A. 2018b, *TOM API Swagger : Yardstick test result analysis*. URL <https://api.tomyardstick.sigmant.net/swagger/>. [71, 79, 114](#)
- SAMBA, A. 2018c, *TOM shiny app : Yardstick test result analysis*. URL <https://tomyardstick.sigmant.net/>. [79, 114](#)
- SAMBA, A., Y. BUSNEL, A. BLANC, P. DOOZE et G. SIMON. 2016, «Throughput prediction in cellular networks : Experiments and preliminary results», dans *1ères Rencontres Francophones sur la Conception de Protocoles, l'Évaluation de Performance et l'Expérimentation des Réseaux de Communication (CoRes 2016)*. [63](#)

- SAMBA, A., Y. BUSNEL, A. BLANC, P. DOOZE et G. SIMON. 2017, «Instantaneous throughput prediction in cellular networks : Which information is needed?», dans *Integrated Network Management (IM), 2017 IFIP/IEEE International Symposium on*, IEEE. [63](#)
- SAMBA, A. et P. DOOZE. 2015, «Détermination de la classe de qos des connexions ip du réseau mobile par l'apprentissage non supervisé», dans *Journée Clustering*. [13](#)
- SCRUCCA, L., M. FOP, T. B. MURPHY et A. E. RAFTERY. 2016, «mclust 5 : Clustering, classification and density estimation using gaussian finite mixture models», *The R journal*, vol. 8, n° 1, p. 289. [87](#)
- SHALEV-SHWARTZ, S. et collab.. 2012, «Online learning and online convex optimization», *Foundations and Trends in Machine Learning*, vol. 4, n° 2, p. 107–194. [14](#)
- SHARROCK, R. 2010, *Gestion autonome de performance, d'énergie et de qualité de service. Application aux réseaux filaires, réseaux de capteurs et grilles de calcul*, thèse de doctorat, Institut National Polytechnique de Toulouse-INPT. [36](#)
- SHIN, M.-K., Y. CHOI, H. H. KWAK, S. PACK, M. KANG et J.-Y. CHOI. 2015, «Verification for nfv-enabled network services», dans *Information and Communication Technology Convergence (ICTC), 2015 International Conference on*, IEEE, p. 810–815. [67](#)
- SIKDAR, B., S. KALYANARAMAN et K. S. VASTOLA. 2003, «Analytic models for the latency and steady-state throughput of tcp tahoe, reno, and sack», *IEEE/ACM Transactions On Networking*, vol. 11, n° 6, p. 959–971. [46](#)
- SMARTBEAR SOFTWARE. 2011, *Swagger : World's Most Popular API Framework*. URL <https://swagger.io>. [71, 104](#)
- SMEETS, B. 2017, *jug : A Simple Web Framework for R*. URL <https://CRAN.R-project.org/package=jug>, r package version 0.1.7. [97](#)
- SPINOSO, S., M. VIRGILIO, W. JOHN, A. MANZALINI, G. MARCHETTO et R. SISTO. 2015, «Formal verification of virtual network function graphs in an sp-devops context», dans *European Conference on Service-Oriented and Cloud Computing*, Springer, p. 253–262. [67](#)
- STONEBRAKER, M. 2010, «Sql databases v. nosql databases», *Communications of the ACM*, vol. 53, n° 4, p. 10–11. [94](#)
- SUN, Y., X. YIN, J. JIANG, V. SEKAR, F. LIN, N. WANG, T. LIU et B. SINOPOLI. 2016, «Cs2p : Improving video bitrate selection and adaptation with data-driven throughput prediction», dans *Proceedings of the 2016 conference on ACM SIGCOMM 2016 Conference*, ACM, p. 272–285. [46](#)
- SUTTON, R. S. et A. G. BARTO. 1998, *Reinforcement learning : An introduction*, vol. 1, MIT press Cambridge. [14](#)
- THE DATA MINING GROUP. 2008, *PMML : Predictive Model Markup Language*. URL <http://dmg.org/pmmml/v4-3/GeneralStructure.html>. [111](#)
- THOMAS, E., M. VAN DEVENTER, T. STOCKHAMMER, A. C. BEGEN et J. FAMAHEY. 2015, «Enhancing mpeg dash performance via server and network assistance», . [62](#)

RÉFÉRENCES

- THOMAS, R. W., D. H. FRIEND, L. A. DASILVA et A. B. MACKENZIE. 2007, «Cognitive networks», dans *Cognitive radio, software defined radio, and adaptive wireless systems*, Springer, p. 17–41. 2
- TRAVIS.CI.COMMUNITY. 2011, *Travis CI : Test and Deploy Your Code with Confidence*. URL <https://travis-ci.org>. 97, 98
- TUKEY, J. W. 1977, *Exploratory data analysis*, vol. 2, Reading, Mass. 68, 69
- TURNBULL, J. 2014, *The Docker Book : Containerization is the new virtualization*, James Turnbull. 97
- TYMOSHENKO, K., O. URYUPINA, A. MOSCHITTI, D. BONADIMAN, I. HAPONCHYK, A. PAS-SERINI, F. GRANELLI, L. XU, H. ASSEM, T. S. BUDA, A. SAMBA, P. DOOZE, Y. B. BEN SLIMEN, S. ALLIO, J. JACQUES, B. ZHU, B. ORDOZGOITI, S. GÓMEZ CANAVAL, A. MOZO, M. QUARTULLI, A. MARTIN, N. OSSES, J. ODRIozola et I. ARRUBARRENA. 2016, «D3.2-Feature and Structure modeling, Structured Input/Output, Unsupervised Learning and Domain Adaptation (Accompanying Document for the Engineering Release)», URL http://www.cognet.5g-ppp.eu/wp-content/uploads/2015/08/D3.2_FINAL.pdf. 63, 114
- URBANEK, S. 2013, *Rserve : Binary R server*. URL <https://CRAN.R-project.org/package=Rserve>, r package version 1.7-3. 97
- V3D. 2016, *EQual One - V3D*. URL <http://www.v3d.fr/solution/equal-one/>. 48
- VALLET, J. 2015, *Optimisation dynamique de réseaux IP/MPLS*, thèse de doctorat, Toulouse, INSA. 24
- VENABLES, W. N. et B. D. RIPLEY. 2013, *Modern applied statistics with S-PLUS*, Springer Science & Business Media. 16, 58
- VETRO, A. et C. TIMMERER. 2005, «Digital item adaptation : overview of standardization and research activities», *IEEE Trans. Multimedia*, vol. 7, n° 3, p. 418–426. 44
- VOGELSTEIN, J. T., B. MENSCH, M. HÄUSSER, N. SPRUSTON, A. C. EVANS, K. KORDING, K. AMUNTS, C. EBELL, J. MULLER, M. TELEFONT et collab.. 2016, «To the cloud ! a grass-roots proposal to accelerate brain science discovery», *Neuron*, vol. 92, n° 3, p. 622–627. 93
- WICKHAM, H. 2011, «testthat : Get started with testing», *The R Journal*, vol. 3, p. 5–10. URL http://journal.r-project.org/archive/2011-1/RJournal_2011-1_Wickham.pdf. 97
- WICKHAM, H. et W. CHANG. 2017, *devtools : Tools to Make Developing R Packages Easier*. URL <https://CRAN.R-project.org/package=devtools>, r package version 1.13.2. 98
- WIKIPEDIA. 2018, «Mutual information — Wikipedia, the free encyclopedia», https://en.wikipedia.org/wiki/Mutual_information. [Online ; accessed 01-May-2018]. 85
- XIE, X., X. ZHANG, S. KUMAR et L. E. LI. 2015, «pistream : Physical layer informed adaptive video streaming over lte», dans *ACM MobiCom*. 47

- XU, L., H. ASSEM, I. G. B. YAHIA, T. S. BUDA, A. MARTIN, D. GALICO, M. BIANCANI, A. PASTOR, P. A. ARANDA, M. SMIRNOV et collab.. 2016, «Cognet : A network management architecture featuring cognitive capabilities», dans *Networks and Communications (EuCNC), 2016 European Conference on*, IEEE, p. 325–329. [37](#)
- XU, Q., S. MEHROTRA, Z. MAO et J. LI. 2013, «PROTEUS : Network Performance Forecast for Real-time, Interactive Mobile Applications», dans *ACM MobiSys*. [46](#)
- YAO, J., S. S. KANHERE et M. HASSAN. 2008, «An empirical study of bandwidth predictability in mobile computing», dans *ACM WINTECH workshop*. [47](#)
- YARDSTICK. 2016a, *Yardstick : Infrastructure Verification*. URL <https://wiki.opnfv.org/yardstick>. [4, 67](#)
- YARDSTICK. 2016b, *Yardstick Test Cases*. URL <http://docs.opnfv.org/en/stable-euphrates/submodules/yardstick/docs/testing/user/userguide/15-list-of-tcs.html>. [67](#)
- ZHANG, Q., L. CHENG et R. BOUTABA. 2010, «Cloud computing : state-of-the-art and research challenges», *Journal of internet services and applications*, vol. 1, n° 1, p. 7–18. [92](#)
- ZHU, X. 2006, «Semi-supervised learning literature survey», *Computer Science, University of Wisconsin-Madison*, vol. 2, n° 3, p. 4. [14](#)
- ZOU, X. K., J. ERMAN, V. GOPALAKRISHNAN, E. HALEPOVIC, R. JANA, X. JIN, J. REXFORD et R. K. SINHA. 2015, «Can accurate predictions improve video streaming in cellular networks ?», dans *ACM HotMobile*. [44, 47](#)

Titre : Science des données au service des réseaux d'opérateur

Mots clés : science des données, réseaux, prédiction de débit, déploiement.

Résumé : L'évolution des télécommunications a mené aujourd'hui à un foisonnement des appareils connectés et une massification des services multimédias. Face à cette demande accrue de service, les opérateurs ont besoin d'adapter le fonctionnement de leurs réseaux, afin de continuer à garantir un certain niveau de qualité d'expérience à leurs utilisateurs. Pour ce faire, les réseaux d'opérateur tendent vers un fonctionnement plus cognitif voire autonome. Il s'agit de doter les réseaux de moyens d'exploiter toutes les informations ou données à leur disposition, les aidant à prendre eux-mêmes les meilleures décisions sur leurs services et leur fonctionnement, voire s'autogérer. Il s'agit donc d'introduire de l'intelligence artificielle dans les réseaux. Cela nécessite la mise en place de moyens d'exploiter les données, d'effectuer sur elles de l'apprentissage automatique de modèles généralisables, apportant l'information

qui permet d'optimiser les décisions. L'ensemble de ces moyens constituent aujourd'hui une discipline scientifique appelée science des données. Cette thèse s'insère dans une volonté globale de montrer l'intérêt de l'introduction de la science des données dans différents processus d'exploitation des réseaux. Elle comporte deux contributions algorithmiques correspondant à des cas d'utilisation de la science des données pour les réseaux d'opérateur, et deux contributions logicielles, visant à faciliter, d'une part l'analyse, et d'autre part le déploiement des algorithmes issus de la science des données. Les résultats concluants de ces différents travaux ont démontré l'intérêt et la faisabilité de l'utilisation de la science des données pour l'exploitation des réseaux d'opérateur. Ces résultats ont aussi fait l'objet de plusieurs utilisations par des projets connexes.

Title: Data science at the service of operator networks

Keywords: data science, operator network, throughput prediction, deployment.

Abstract: The evolution of telecommunications has led today to a proliferation of connected devices and a massification of multimedia services. Faced with this increased demand for service, operators need to adapt the operation of their networks, in order to continue to guarantee a certain level of quality of experience to their users. To do this, operator networks tend towards a more cognitive or autonomic functioning. It is about giving the networks the means to exploit all the information or data at their disposal, helping them to make the best decisions about their services and operations, and even self-manage. It is therefore a question of introducing artificial intelligence into networks. This requires setting up means to exploit the data, to carry out on them the automatic learning of generalizable models, providing information

that can optimize decisions. All these means today constitute a scientific discipline called data science. This thesis fits into a global desire to show the interest of the introduction of data science in different network operating processes. It includes two algorithmic contributions corresponding to usecases of data science for the operator networks, and two software contributions, aiming to facilitate, on the one hand, the analysis, and on the other hand the deployment of the algorithms produced through data science. The conclusive results of these various studies have demonstrated the interest and the feasibility of using data science for the exploitation of operator networks. These results have also been used by related projects.