THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse Capitole (UT Capitole)*

Présentée et soutenue le *18 juin 2024* **par :**

Thibault Fourez

Analyser la Mobilité avec un Système Multi-Agents Ensembliste d'Apprentissage par Contexte

---

JURY

Frédéric AMBLARD	Professeur, Université Toulouse Capitole	Directeur
Frédéric SCHETTINI	Responsable de l'innovation digitale, Citec	Co-directeur du monde socio-économique
Nicolas VERSTAEVEL	Maître de conférences, Université Toulouse Capitole	Co-directeur
Flavien BALBO	Professeur, Mines Saint-Étienne	Rapporteur et président du jury
Emmanuelle GRISLIN-LE STRUGEON	Professeure, INSA Hauts-de-France	Rapportrice
Patrice AKNIN	Directeur scientifique, IRT SystemX	Examinateur
Frédéric ARMETTA	Maître de conférences, Université Claude Bernard Lyon 1	Examinateur
Margot PERIARD	Ingénieure, Cerema	Invitée

École doctorale et spécialité :
MITT : Domaine STIC : Intelligence Artificielle

Unité de Recherche :
Institut de Recherche en Informatique de Toulouse

Directeur(s) de Thèse :
Frédéric AMBLARD, *Frédéric SCHETTINI* et *Nicolas VERSTAEVEL*

Rapporteurs :
Flavien BALBO et *Emmanuelle GRISLIN-LE STRUGEON*

Aucun de nous ne sait ce que nous savons tous, ensemble.
Euripide

Résumé

La connaissance de l'état de la mobilité sur un territoire est un enjeu majeur pour les collectivités locales. Savoir comment et pourquoi les usagers se déplacent peut notamment influer sur les choix d'urbanisme et de développement en matière de transports en commun, de sécurité et de lutte contre le changement climatique.

L'émergence de sources de données de mobilité à grande échelle, en particulier de mesures effectuées directement sur les smartphones des usagers, rend possible l'utilisation de modèles d'apprentissage automatique pour produire des analyses de l'état de mobilité à n'importe quel moment là où les enquêtes classiques sont chères et très peu fréquentes.

En considérant le cas de la détection du mode de transport, qui est un problème de classification supervisée non linéaire, des pré-requis sont établis pour la mise en place d'un modèle d'apprentissage automatique. Ces pré-requis mènent à la construction de Smapy, un classifieur coopératif d'un genre nouveau qui se positionne à l'intersection entre système multi-agents adaptatif (AMAS) d'apprentissage par contexte (SACL), agrégation ensembliste et constructiviste de classifieurs, modèle de voisinage et apprentissage par renforcement.

Une première expérimentation met en concurrence Smapy, dont les agents sont munis de modèles linéaires, avec ces mêmes modèles linéaires seuls sur trois jeux de données de degrés de linéarité variables. Les résultats montrent que Smapy parvient à résoudre les problèmes non linéaires en les transformant en problèmes de coopération entre agents munis de modèles linéaires, là où ces mêmes modèles seuls en sont incapables.

Deux autres expérimentations introduisent une chaîne de détection du mode de transport de bout en bout à partir de données de géolocalisation et d'accélération collectées sur des usagers via des applications smartphone, et comparent les performances de Smapy avec d'autres classifieurs de la littérature. Les résultats montrent que Smapy présente des performances semblables aux autres modèles de référence dans la majorité des cas, tout en disposant d'un fort potentiel d'explicabilité dû à ses caractéristiques géométriques.

Des perspectives de recherche sont établies, notamment en ce qui concerne l'explicabilité de Smapy. Des pistes d'amélioration des contributions et de résolution du problème de classification de l'activité aux arrêts sont également présentées.

Abstract

Knowledge of the state of mobility in a given area is a major challenge for local authorities. Knowing how and why users travel can influence urban planning and development choices in terms of public transport, safety and the climate change policy.

The emergence of large-scale mobility data sources, in particular measurements taken directly from users' smartphones, makes it possible to use machine learning models to produce analyses of the state of mobility at any given time, where conventional surveys are expensive and very infrequent.

Considering the use case of transport mode detection, which is a non-linear supervised classification problem, prerequisites are established for the implementation of a machine learning model. These prerequisites have led to the construction of Smapy, a new kind of cooperative classifier that lies at the intersection of adaptive multi-agent systems (AMAS), self-adaptative context learning (SACL), ensemblist and constructivist aggregation of classifiers, neighbourhood model and reinforcement learning.

A first experiment pitted Smapy, whose agents are equipped with linear models, against these same linear models alone on three data sets of varying degrees of linearity. The results show that Smapy succeeds in solving non-linear problems by transforming them into cooperation problems between agents equipped with linear models, whereas these same models alone are incapable of doing so.

Two other experiments introduce an end-to-end transport mode detection chain based on geolocation and acceleration data collected from users via smartphone applications, and compare Smapy's performance with other classifiers in the literature. The results show that Smapy performs similarly to other reference models in the majority of cases, while having a high potential for explanability due to its geometric characteristics.

Prospects for further research are identified, particularly with regard to Smapy's explicability. Avenues for improving the contributions and solving the problem of classifying activity at stops are also presented.

Ça y est, nous y sommes. Ma thèse touche à sa fin et vous vous apprêtez à lire l'aboutissement de 1160 jours de ma vie à travers ce manuscrit. Ou du moins, une conclusion, car cette thèse fut avant tout un voyage de trois ans qui ne saurait être retranscrit pleinement dans ce document, aussi dense et néanmoins (je l'espère) agréable à lire soit-il. Ma véritable richesse, c'est de pouvoir regarder en arrière et de voir à quel point la personne que j'étais il y a trois ans a grandi. Ces quelques lignes sont pour toutes les personnes qui, de près ou de loin, m'ont accompagné dans mon périple.

Je tiens tout d'abord à remercier mes deux rapporteurs, Emmanuelle Grislin-Le Strugeon et Flavien Balbo, pour leur travail de relecture et leurs précieux retours. Les autres membres du jury, Patrice Aknin et Frédéric Armetta, pour avoir accepté d'évaluer mon travail de thèse. Margot Periard, pour sa présence lors de ma soutenance. Je remercie également mon comité de suivi de thèse, composé de Thomas Thévenin et Laurent Vercouter, pour leur accompagnement.

Bien sûr, j'exprime ma plus profonde gratitude à mes directeurs de thèse, Nicolas et ses drôles de Frédérics. Nicolas tout d'abord, pour son soutien constant, sa bonne humeur et sa disponibilité, alors même qu'il a entrepris un voyage beaucoup plus long et beau que le mien le jour où il est devenu papa (encore félicitations à toi !). Tu m'as transmis ta vision de la recherche et tu as su me guider pendant ces trois dernières années.

Vient ensuite le premier des Frédérics, que nous nommeront Fred A dans la suite de ce chapitre. Je te remercie pour ton implication et tes précieux conseils tout au long de ma thèse. Tu m'as montré les ficelles du monde de la recherche et j'espère avoir hérité de ta sérénité à toute épreuve. Fred M, qui m'a accompagné durant la première moitié de ma thèse et que je tiens à remercier chaleureusement. Je retiens notamment nos longues séances de brainstorming aussi agréables qu'enrichissantes.

Enfin, j'ai une pensée particulière pour Fred S, qui m'a accompagné dès mon arrivée à Citec en 2020 et bien entendu tout au long de ma thèse. Tu es probablement celui qui peut le plus témoigner de mon évolution ces trois dernières années et tu as su, malgré les très nombreux projets sur lesquels tu travailles, occuper un véritable rôle de mentor pour lequel je te serai toujours reconnaissant.

Mes remerciements vont également à Citec et à tous les collègues avec qui j'ai eu la chance de travailler et partager des moments de convivialités durant ma thèse. Plus particulièrement, l'équipe de Toulouse avec Sonia, Sylvain, Léandre, Dominika, Yazid, Brigitte. L'équipe Modelity, bien entendu, avec Mansour, Camille, Camille, Murray et Hugues qui m'ont tous aidé dans mes travaux de recherche. Merci également aux stagiaires que j'ai eu l'occasion de (co)encadrer, Yasser, Salamata et Léonard. Enfin, merci à Nicolas Chiabaut pour ses conseils de recherche durant son passage au sein de Citec.

Côté labo, la bonne ambiance de l'équipe SMAC a rendu les jours passés à l'IRIT particulièrement agréables et stimulants intellectuellement. Merci aux permanents de l'équipe, Marie-Pierre, Elsy, Stéphanie, Jean-Pierre (dit JPEG) et

Benoît, qui œuvrent à développer une véritable dynamique autour d'un sujet de recherche aussi vaste que passionnant. Merci à Kamal avec qui j'ai eu la chance de collaborer durant son post-doctorat à l'IRIT. Merci également à Bruno dont le travail a largement influencé ma thèse, et de qui j'ai eu le privilège d'assister à sa soutenance. Merci à Sébastien, mon plus fidèle colocataire de bureau. J'ai également une pensée pour Walid, Guilhem, Davide et Ha Nhi qui ont brillement soutenu avant moi, ainsi que Quentin, Damia, Kristell, Kévin, Maxence et Axel qui sont les prochains sur la liste... Bon courage à Timothée, avec qui j'ai eu peu d'occasions d'échanger, mais qui soutient sa thèse le même jour que moi. Félicitations par avance. Tous mes encouragements à Clément et Jordan qui continuent les travaux sur l'apprentissage par Contexte dans leurs propres thèses. Enfin, je remercie Benoît, Manon, Léo, Valentin, Livia, Pierre, Benjamin, ainsi que tous ceux que j'ai oublié. Merci pour ces sessions de jeux de société, pour ces escapes rooms, pour ces afterworks et aussi (un peu) pour le travail réalisé ensemble !

A mes meilleurs amis, pour la plupart rencontrés à l'INSA, mais dont j'ai su très tôt que je resterai proche tout au long de ma vie, un immense merci pour les moments passés ces trois dernières années. A ceux tout d'abord qui ont choisi le chemin de la thèse, ~~Héloise~~ ~~Eloase~~ Eloïse qui s'est envolée pour Montréal et qui me donne un prétexte pour y retourner moi aussi, Philippe qui aura finalement soutenu six jours avant moi dans le seul but d'éviter mes questions, Alice avec qui je passe mon temps à râler (et je suis sûr que ça continuera bien après la thèse) et Ilinka dont j'ai l'immense honneur d'être l'ami depuis une décennie (!) tout en ayant réussi à esquiver ses deux derniers déménagements.

Aux autres, à Vincent qui me supporte depuis encore plus longtemps et que j'ai hâte de revoir au Canada ou ailleurs, à Marine avec qui je forme l'inimitable duo de charisme, à Fanny mon éternel binôme de TP avec qui l'adage « qui aime bien châtie bien » est constamment vérifié, à ~~Vincent~~ Kronk qui m'impressionnera toujours par sa capacité à se passionner pour tout type de sujet et qui est la seule personne que je connais à regarder un film avec les mains, à Mathilde dite la maman du groupe ou FOTS, toujours aussi prévoyante, à Louis le futur homme le plus drôle de France, bien que j'ai pu être assez dur avec certaines de tes blagues, à Jacques dont j'espère connaître un jour l'ensemble de ses titres de noblesse et à Léa que j'ai appris à connaître plus récemment, je suis fier et heureux de vous connaître, merci du fond du cœur.

Au groupe du CNES et d'ailleurs, Clara, Paul, Tom, Timothée, Hugo et Valentin, avec qui j'ai eu la chance de me lier d'amitié juste avant le début de ma thèse, merci pour tous ces moments partagés ces dernières années. Vous êtes la preuve que de grandes amitiés peuvent naître en très peu de temps.

Enfin, merci à David et Andrea qui ont été parmi mes rencontres les plus marquantes de ses trois dernières années et qui m'ont fait grandir.

Et bien sûr, je garde mes derniers mots pour ma famille, qui a toujours été à mes côtés et à laquelle je suis fier d'appartenir. Maman, Papa, merci d'avoir été les meilleurs parents du monde, je vous aime. C'est grâce à vous que je suis qui je suis aujourd'hui, je vous en serai éternellement reconnaissant.

A mes deux grand-mères, Eliane et Marie, à mon grand-père Jean qui nous a quittés, à Yvan, Dominique et leurs filles Eléonore (dite Léo) et Esther, merci d'avoir été à mes côtés tout au long de ma vie. Je n'ai pas besoin de beaucoup de lignes pour vous dire à quel point je suis fier de vous avoir à mes côtés. Je vous aime tous énormément.

Enfin, merci Anne-Lise de supporter (dans tous les sens du terme) ton petit frère depuis toujours. Tu m'impressionnes sans cesse. Je t'aime.

Table des matières

Introduction générale

La connaissance de l'état de la mobilité sur un territoire est primordiale pour la prise de décisions des collectivités locales [Bac+19]. Quels sont les besoins en termes de transport en commun ? Vers où se dirigent les principaux flux domicile/travail ? La réponse à ces questions s'obtient généralement par de longues enquêtes sur les déplacements, très coûteuses et peu fréquentes. D'un autre côté, la multiplication de capteurs embarqués sur les usagers, notamment à travers leurs smartphones, constitue une source de données importante et fiable pour analyser les déplacements effectués [SB+17]. Le volume de ces données potentielles pose plusieurs problèmes classiques dans le domaine du Big Data. Tout d'abord, la nécessité de mettre en place une chaîne de traitements fiable et automatisée pour réaliser les analyses souhaitées à l'échelle d'un quartier, d'une ville ou même d'un pays. Ces chaînes de traitements sont basées sur des modèles d'apprentissage automatique plus ou moins élaborés selon l'objectif de l'analyse.

Ces modèles doivent notamment s'adapter à d'importants changements de comportement des usagers dans le temps et dans l'espace [Fou+22a]. En effet, les infrastructures, les habitudes et la législation sont largement dépendantes du territoire et de la période temporelle considérés.

De plus, le résultat de ces analyses ayant un impact sur les choix d'urbanisme, de développement économique et sur la qualité de vie des usagers, il est primordial d'anticiper les besoins d'explicabilité et d'interprétabilité des modèles d'apprentissage automatique utilisés [Mon+22].

Contributions de la thèse

Une solution envisagée dans ce projet de thèse est d'utiliser des systèmes multi-agents coopératifs pour analyser l'état de la mobilité. De tels systèmes possèdent une capacité d'adaptation rapide, et sont capables de transformer un problème d'apprentissage automatique en un problème de coopération basée sur un ensemble restreint de règles. Cette simplicité rend le comportement des systèmes multi-agents coopératif interprétables par un utilisateur humain et constitue un premier pas vers l'explicabilité souhaitée dans l'analyse de la mobilité à grande échelle.

Cette thèse présente deux types de contributions (c.f. l'axe horizontal de la figure 0.1). D'un point de vue méthodologique, un nouvel algorithme d'apprentissage automatique répondant à ces problématiques est introduit sous la forme d'un système multi-agent coopératif et ensembliste nommé Smapy. Des expérimentations sont menées pour prouver sa capacité à résoudre des problèmes de classification supervisée grâce à des règles de coopération entre agents.

D'un point de vue applicatif, le problème de détection du mode de transport est choisi pour illustrer le potentiel d'un tel modèle dans l'analyse de la mobilité. Plusieurs expérimentations sont menées pour évaluer les performances et la capacité de généralisation d'une chaîne de traitements de bout en bout. Les résultats des expérimentations menées sont détaillés et mis en perspective en vue d'une généralisation à d'autres cas d'étude de l'état de la mobilité.

Organisation du manuscrit

Le manuscrit est structuré de la manière suivante :

— Chapitre 1 : Ce chapitre introduit la notion d'état de la mobilité selon cinq axes et positionne le projet industriel de l'entreprise Citec dans lequel s'inscrit la thèse par rapport à ces axes. Les trois grand objectifs de la thèse sont établis : caractériser et classifier les données de mobilité disponibles, étudier les pré-requis des modèles d'analyse automatique de la mobilité à grande échelle, et proposer une solution algorithmique appliquée au cas d'usage de la détection du mode de transport.

— Chapitre 2 : Dans ce chapitre, je propose une classification des données de mobilité utilisées dans la littérature en trois grandes catégories : les données contextuelles qui sont relatives au contexte géographique et culturel du territoire d'étude, les données utilisateur qui sont des séries temporelles de mesures effectuées sur des capteurs embarqués par les usagers, et les données agrégées qui sont des projections de données utilisateur sur des données contextuelles à but statistique.

— Chapitre 3 : Ce chapitre justifie la construction d'un système multi-agents coopératif et ensembliste pour répondre aux pré-requis introduits dans le chapitre 1. Pour cela, différents concepts d'intelligence artificielle issus de la littérature sur lesquels le système s'appuie, tels que l'apprentissage automatique, l'agrégation de classifieurs et l'apprentissage par contexte, sont présentés.

— Chapitre 4 : Dans ce chapitre, je présente un cas d'application de l'analyse de la mobilité abondamment traité dans la littérature : la détection du mode de transport. J'introduis une classification des solutions utilisant les données utilisateur en reprenant le traitement de données étape par étape (données d'entrée, pré-traitement, calcul de features, traitement et post-traitement).

— Chapitre 5 : Ce chapitre présente Smapy, le système multi-agents ensembliste conçu pour répondre aux pré-requis listés dans le chapitre 1. Après une description de la structure du système et de ses règles de fonctionnement et coopération, je détaille les motivations derrière chaque mécanisme. Les caractéristiques de l'implémentation d'un tel système sont également présentées.

— Chapitre 6 : Dans ce chapitre, je démontre par l'expérimentation que Smapy permet de transformer un problème de classification non linéaire

en problème de coopération, plus simple à résoudre. Sur des jeux de données jouets ayant plusieurs degrés de linéarité, des classifieurs linéaires sont utilisés seuls ou intégrés dans des agents Contexte pour discriminer deux classes. L'intégration dans des agents permet de résoudre même les problèmes non linéaires.

— Chapitre 7 : Ce chapitre présente une méthode de détection du mode de transport conçue pour s'intégrer dans le contexte industriel de la thèse. Les différentes étapes de la chaîne de traitement sont détaillées, et plusieurs classifieurs dont Smapy sont entraînés sur des données de géolocalisation et d'accélération collectées à l'aide d'une application smartphone. Les résultats de cette expérimentation sont présentés et discutés.

— Chapitre 8 : Ce chapitre décrit la validation de la méthode de détection du mode de transport présentée dans le chapitre 7 sur des données de géolocalisation et d'accélération plus fournies, collectées via une application smartphone développée par Citec. Les caractéristiques de ces données diffèrent de celles utilisées dans le chapitre 7, et plusieurs scénarios d'apprentissage sont étudiés pour qualifier la généralisation de la méthode dans le temps et dans l'espace. Les mêmes classifieurs sont entraînés sur ces nouvelles données, et le comportement de Smapy est discuté pour chacun des scénarios.

— Chapitre 9 : Dans cet ultime chapitre, une conclusion générale synthétise l'ensemble du manuscrit. Les contributions de la thèse sont listées selon trois domaines : l'analyse de la mobilité, l'apprentissage par contexte dans les systèmes multi-agents adaptatifs et l'apprentissage automatique. Les perspectives de poursuite du sujet de thèse sont ensuite établies et les premières pistes explorées sont détaillées.

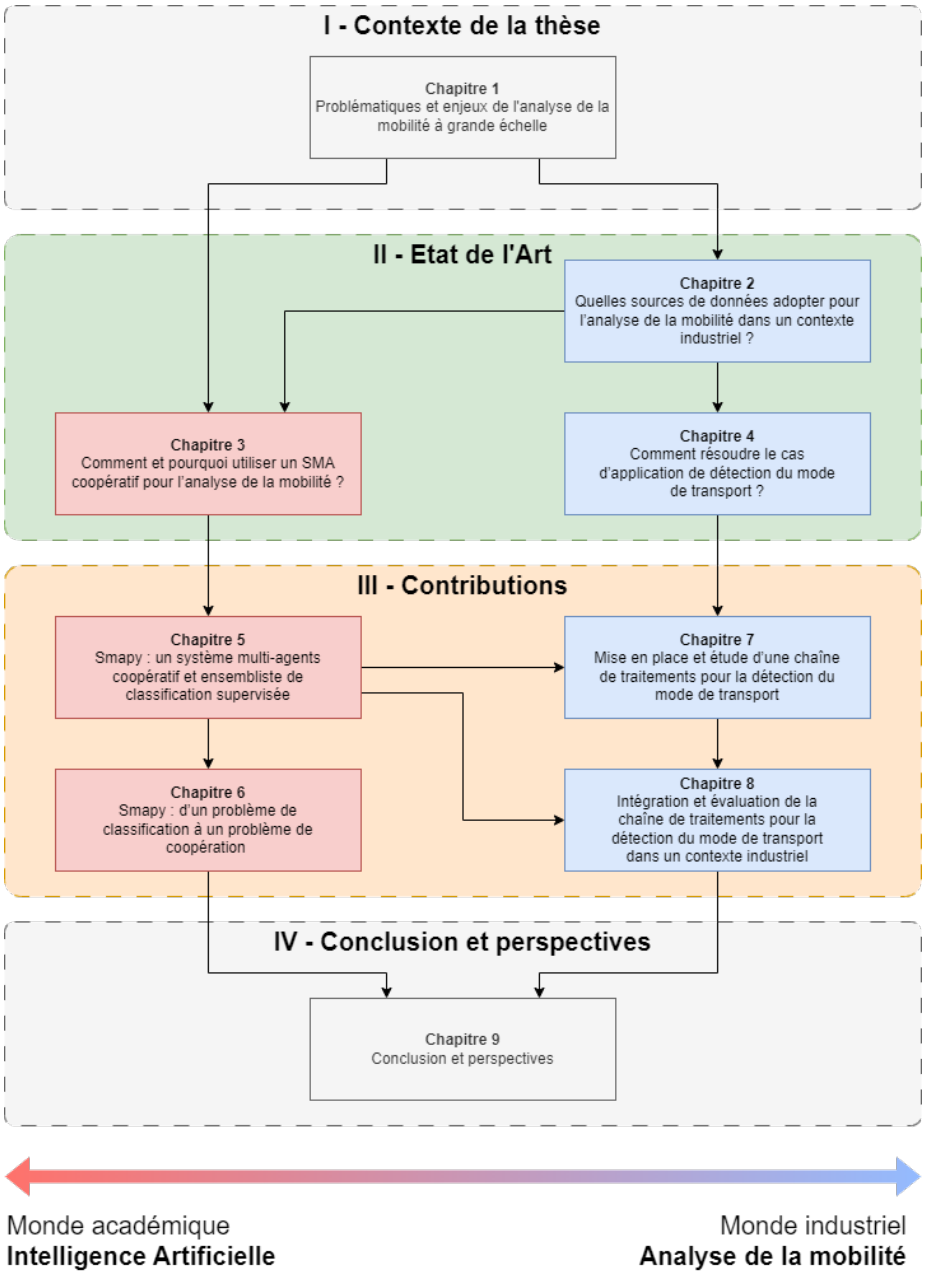


FIGURE 0.1 – Schéma de l'organisation du manuscrit.

Première partie

Contexte de la thèse

Chapitre 1

Problématiques et enjeux de l'analyse de la mobilité à grande échelle

Qu'est-ce que la mobilité ? Si l'Académie française la définit sobrement comme la "capacité à se mouvoir ou à être mû" [Aca], le sens du terme mobilité dépend avant tout du sujet du mouvement, ainsi que l'espace dans lequel l'action est effectuée. Le sujet du mouvement peut être un objet ou une personne, et l'espace peut être physique (mobilité spatiale), virtuel (e.g. mobilité de l'information) ou social (mobilité sociale). Dans le cadre de cette thèse, nous nous intéressons à la mobilité spatiale des individus, c'est-à-dire leurs déplacements dans l'espace géographique que constitue le monde réel.

Ce chapitre présente le contexte dans lequel s'effectue cette thèse. Dans un premier temps, la section 1.1 introduit la notion d'état de la mobilité et les enjeux associés à sa connaissance. Le contexte industriel de la thèse est ensuite présenté dans la section 1.2. Dans un dernier temps, les objectifs de la thèse sont introduits et détaillés dans la section 1.3 avant d'être traités dans la suite du manuscrit.

1.1 Etat de la mobilité

Tout déplacement possède une origine et une destination, deux lieux qui présentent *a priori* un intérêt pour son auteur. De plus, il s'étend sur une période temporelle continue allant du départ de l'origine à l'arrivée à la destination. Le déplacement est associé à un vecteur, appelé mode de transport, et à un réseau composé de routes et chemins. Chaque mode de transport peut circuler sur tout ou partie de ce réseau. Enfin, un déplacement est nécessairement motivé par un besoin qui justifie la dépense d'énergie associée. Toutes ses caractéristiques dépendent de l'auteur du déplacement, de ses motivations et des ressources dont il dispose. Ainsi, analyser la mobilité, c'est se poser, pour chaque déplacement, tout ou partie de cinq questions [Fou+23] :

— Qui est l'auteur du déplacement ?
— Où se déplace-t-il ? (entre quels lieux ?)
— Quand se déplace-t-il ? (à quel moment ?)
— Comment se déplace-t-il ? (par quel mode ?)
— Pourquoi se déplace-t-il ? (pour quel motif ?)

Répondre à ces questions sur un territoire et une période donnés, c'est en connaître l'état de la mobilité [Fou+23]. Il s'agit d'un enjeu majeur pour les

collectivités locales, car l'état de la mobilité conditionne les choix d'urbanisme et les stratégies de développement d'un quartier, d'une ville ou même d'une région.

Des enquêtes de déplacement exhaustives sont effectuées auprès des citoyens, mais elles sont peu fréquentes (une à deux fois par décennie) car coûteuses et essentiellement déclaratives (les données proviennent des déclarations des usagers et ne sont pas observées de façon précise et objective) [Bac+19] [Gon+12]. Les données collectées au cours de ces enquêtes peuvent rapidement devenir obsolètes car les pratiques liées à la mobilité évoluent sans cesse, au fil des changements sociétaux (e.g. apparition de nouveaux modes de transport), législatifs (e.g. mesures visant à limiter les impacts du changement climatique) ou du fait d'événements exceptionnels (e.g. pandémie mondiale du COVID-19, organisation des Jeux Olympiques).

Un enjeu de taille pour les collectivités est donc d'avoir une bonne estimation de l'état de la mobilité sans avoir à investir dans une enquête de grande ampleur, sur une période et un territoire donnés. Dans la section suivante, le contexte industriel dans lequel cette thèse s'inscrit est présenté.

1.2 Contexte industriel de la thèse

Cette thèse s'effectue dans le cadre d'un contrat CIFRE entre l'entreprise Citec Ingénieurs Conseils et l'Institut de Recherche en Informatique de Toulouse (IRIT). Citec est un bureau d'ingéniérie présent sur trois pays (Suisse, France et Italie) et spécialisé dans les études de mobilité. Les projets sur lesquels travaillent Citec couvrent un large spectre allant de la modélisation du trafic à l'aménagement du territoire ou les plans de mobilité d'entreprise. Les principaux clients de Citec sont des collectivités locales qui ont besoin de connaître l'état de la mobilité sur leur territoire et d'orienter leur politique d'urbanisme afin d'atteindre leurs objectifs en termes de sécurité, de qualité de vie, ou encore de lutte contre le changement climatique.

L'équipe Citec Digital est chargée de développer des solutions numériques d'analyse et de valorisation de la donnée à destination des ingénieurs Citec. A travers plusieurs partenariats avec des fournisseurs de données de mobilité, l'équipe Digital produit des analyses statistiques et géographiques utilisées dans de nombreux projets et occupe un rôle transversal.

Ce projet de thèse se caractérise par une forte dimension applicative liée aux activités de Citec. Plus particulièrement, les objectifs de la thèse sont établis et détaillés dans la prochaine section.

1.3 Objectifs de la thèse

Dans ce contexte, le principal objectif de cette thèse est d'étudier le potentiel de modèles d'apprentissage automatique pour connaître l'état de la mobilité sur un territoire à partir de données collectées passivement sur les usagers. En d'autres termes, l'enjeu est de pouvoir générer des résultats similaires à ceux obtenus via une véritable enquête de déplacements, sans les coûts temporels et financiers associés, et sans les biais liés à leur nature déclarative.

1.3.1 Sur quelles données peut s'appuyer l'analyse de la mobilité ?

Une des premières questions à se poser est la nature des données à utiliser pour étudier l'état de la mobilité sur un territoire et une période donnés. La réponse à cette question, qui est également la première contribution de cette thèse, consiste en une nouvelle classification à la fois académique et industrielle des sources de données de mobilité disponibles. Chaque source de données présente des avantages et inconvénients, au niveau de leur représentativité, de leur disponibilité dans un contexte industriel ou encore de leur précision. Elles peuvent êtres classées selon trois catégories :

— Les **données contextuelles** qui sont indépendantes des utilisateurs et donnent des informations sur le contexte spatio-temporel des déplacements.

— Les **données utilisateur** qui sont des séries temporelles de mesures effectuées sur les utilisateurs.

— Les **données agrégées** qui sont des projections des données utilisateurs sur des données contextuelles, généralement dans un but d'analyse statistique.

1.3.1.1 Finalité du projet industriel

L'objectif à long terme pour l'équipe Digital est de générer des résultats similaires à une enquête de déplacement exhaustive telle que réalisée par des collectivités, uniquement à partir de données innovantes à moindre coût. Le résultat de ces analyses aurait ainsi la même structure que des données agrégées, comme indiqué sur la figure 1.1. Dans la pratique, des données agrégées de références (e.g. les enquêtes de déplacements précédentes) sont généralement utilisées pour combler le manque de représentativité des données collectées.

1.3.1.2 Formalisme des données utilisateur

Les données utilisateur constituent la principale source de données sur laquelle je m'appuie pour produire des analyses de l'état de la mobilité dans le cadre de cette thèse. Les caractéristiques de ces données sont présentées et discutées dans la suite de ce manuscrit.

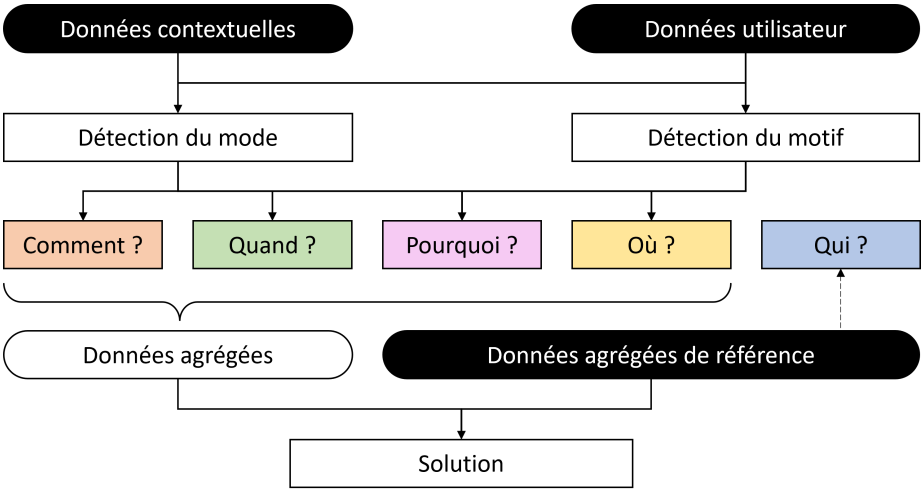


FIGURE 1.1 – Schéma du fonctionnement de la solution envisagée dans le cadre du projet de thèse

Les données utilisateur caractérisent les mouvements effectués par les usagers. Pour qualifier ces mouvements, je me base dans cette thèse sur la terminologie introduite par [BLVO13] :

- La **trace** d'un utilisateur (*Movement Trajectory* dans [BLVO13]) qui contient l'ensemble des points de données collectés sur un utilisateur. Elle est composée de trajets et d'arrêts.
- Le **trajet** (*Journey* dans [BLVO13]) qui correspond au déplacement entre deux points d'intérêt et qui peut être composé de plusieurs segments.
- Le **segment** qui correspond à un déplacement mono-modal (i.e. avec un seul mode de transport).
- Le **point d'intérêt** (*Relevant Location* dans [BLVO13]) qui correspond à l'origine ou la destination d'un trajet, et où l'utilisateur reste le temps d'effectuer une activité (e.g. domicile, lieu de travail, magasin, etc.).

1.3.2 Quel modèle utiliser pour l'analyse automatique de la mobilité à grande échelle ?

L'utilisation de l'apprentissage automatique et plus généralement de l'intelligence artificielle dans l'analyse de la mobilité n'est pas nouvelle [Abd+19]. Dans une deuxième contribution, plusieurs types de modèles d'intelligence artificielle de la littérature sont introduits afin de répondre à trois pré-requis majeurs de l'analyse automatique de la mobilité à grande échelle :

- **Apprentissage en ligne et capacité d'adaptation rapide** : le comportement des usagers évolue dans le temps et l'espace. En raison

d'événements (e.g. la pandémie de COVID-19, l'adoption d'une nouvelle loi, etc.) ou de différences d'infrastructures d'un territoire à l'autre (e.g. environnement urbain, environnement rural), la distribution statistique de toutes les mesures effectuées sur des utilisateurs change et les modèles d'apprentissage automatique doivent s'adapter à ces changements. Une solution immédiate à cette problématique est l'apprentissage en ligne qui permet d'affiner un modèle à partir d'observations plus récentes sans devoir le ré-entraîner depuis le début. Cependant, certains modèles possèdent une grande inertie et nécessitent de nombreuses observations pour s'adapter aux nouveaux comportements.

— **Performance sur des problèmes fortement non linéaires** : la mobilité individuelle des personnes est complexe et difficilement prévisible. Sur des problèmes de classification (e.g. mode de transport, but du déplacement, etc.), les frontières entre les classes sont souvent floues et non linéaires. Les modèles d'apprentissage envisagés doivent avoir de bonnes performances sur des problèmes fortement non linéaires pour garantir une précision acceptable dans un contexte industriel.

— **Explicabilité** : les utilisateurs des résultats produits sont majoritairement des collectivités locales ayant un pouvoir décisionnel en termes d'urbanisme sur leur territoire. Ces décisions ont un impact direct sur la qualité de vie des usagers (e.g. temps de trajet, apaisement de la circulation, etc.) et leur sécurité (e.g. protection des piétons et cyclistes, gestion des mouvements de foule, etc.). Il est donc primordial que les modèles d'apprentissage automatique utilisés ne soient pas des "boîtes noires" [Phi+20], et que leurs prédictions puissent être interprétées et expliquées par des ingénieurs et chargés de mobilité.

Les modèles d'apprentissage automatiques introduits présentent tous des avantages et inconvénients, et conduisent à la construction d'un nouveau type de classifieur, à l'intersection entre système multi-agents adaptatif (AMAS) [Cap+03], agrégation ensembliste, modèle de voisinage et apprentissage par renforcement. L'implémentation d'un tel classifieur, appelé Smapy, constitue la troisième contribution de cette thèse. Son fonctionnement est détaillé et plusieurs expérimentations sont menées pour étudier son potentiel selon les pré-requis introduits dans ce chapitre.

1.3.3 Cas d'application de la détection du mode de transport

Par rapport aux cinq axes de connaissance de la mobilité, il a été choisi d'illustrer le potentiel de Smapy et d'autres classifieurs issus de la littérature sur le cas d'application de la classification supervisée du monde de transport à partir de données utilisateur. Après un état de l'art qui constitue la quatrième contribution de cette thèse, une chaîne de traitements de bout en bout a été développée. A travers deux expérimentations basées sur plusieurs jeux de données, les performances de plusieurs classifieurs (dont Smapy) sont comparées afin d'évaluer le potentiel de mise en production dans le contexte industriel de

Citec.

Trois problématiques sont établies dans ce chapitre. Quelles données permettent de produire des analyses de l'état de la mobilité qui s'inscrivent dans le contexte industriel de Citec ? Quel modèle utiliser pour produire ces analyses ? Comment caractériser les performances de ce modèle sur le cas de la détection du mode de transport ? Les trois chapitres suivants constituent l'état de l'art de cette thèse et présentent respectivement les approches de la littérature pour répondre à ces trois problématiques.

Deuxième partie

Etat de l’Art

Chapitre 2

Quelles sources de données adopter pour l'analyse de la mobilité dans un contexte industriel ?

Telle que nous l'avons présentée dans le chapitre 1, la connaissance de l'état de la mobilité s'inscrit dans un contexte spatio-temporel et s'articule autour de 5 axes ("Qui ?", "Quand ?", "Où", "Comment ?" et "Pourquoi ?"). L'axe "Qui ?" concerne le profilage des utilisateurs se déplaçant sur une zone et une période d'étude. Les informations recherchées sont généralement relatives à la catégorie socio-professionnelle de l'utilisateur, son âge et son genre [ris21]. Ces informations étant sensibles et intrusives, un des premiers constats est l'impossibilité de disposer et d'utiliser de telles données. L'axe "Qui ?" a donc été écarté de l'analyse de la mobilité envisagée.

Dans cette thèse, nous nous concentrons sur les quatre autres axes de l'analyse de l'état de la mobilité. Les données permettant de les caractériser peuvent être classées selon la typologie suivante :

— Les **données contextuelles** qui sont indépendantes des utilisateurs et donnent des informations sur le contexte spatio-temporel des déplacements.
— Les **données utilisateur** qui sont des séries temporelles de mesures effectuées sur les utilisateurs.
— Les **données agrégées** qui sont des projections des données utilisateurs sur des données contextuelles, généralement dans un but d'analyse statistique.

Dans ce chapitre, nous présentons et formalisons ces trois types de données de mobilité, leurs intérêts et limites respectives. En particulier, nous détaillons les données utilisateur de géolocalisation qui par définition répondent aux deux axes "Où ?" et "Quand ?" présentés dans le chapitre 1. Répondre aux deux axes restants revient à considérer les deux problèmes suivants :

— La détection du mode de transport ou Transport Mode Detection (TMD) pour l'axe "Comment ?".
— La détection du motif des déplacements pour l'axe "Pourquoi ?".

La section 2.1 présente les différents types de données contextuelles et leur intérêt en analyse de la mobilité. Dans la section 2.2, les données agrégées sont introduites. La section 2.3 présente de façon détaillée les données utilisateur disponibles pouvant servir à l'analyse de la mobilité. Les attributs relatifs à

chaque type de données utilisateur sont également présentés et formalisés. Enfin, dans la section 2.4, une synthèse des différents types de données de mobilité est présentée.

2.1 Données contextuelles

Tout déplacement d'un individu s'effectue dans un contexte spatial, temporel et social. Les informations relatives au réseau de transport, son état et son infrastructure, sont des exemples de données contextuelles [Fou+23]. Elles sont indépendantes de la trajectoire de l'individu mais peuvent fortement la conditionner : un usager pourra par exemple privilégier un mode de transport à un autre, choisir son heure de départ ou adapter son itinéraire en fonction de l'état de travaux sur la route. Les données contextuelles font partie d'un Système d'Information Géographique (SIG) lorsqu'elles sont relatives à un territoire et peuvent être cartographiées. Cette section présente successivement les différents types de données contextuelles :

— L'infrastructure de transport (réseau routier, réseau de transports en commun)

— Le découpage administratif

— L'occupation du sol

— Les lieux d'intérêt

— Les événements (de nature sociale, météorologique, etc.)

2.1.1 Infrastructure de transport

Deux éléments principaux composent l'infrastructure de transport d'un territoire. Tout d'abord, le réseau de routes et axes de circulation que les véhicules, cyclistes, piétons peuvent emprunter. Ensuite, le réseau de transports en commun (TC) composé en partie d'infrastructures propres (e.g. arrêts/gares, voies ferrées) et d'un calendrier de dessertes.

2.1.1.1 Réseau routier

L'ensemble des routes et chemins peut être représenté par un graphe orienté dont les noeuds sont les intersections et les liens les arcs routiers permettant de connecter les noeuds entre eux. Les arcs sont donc les unités permettant de construire un itinéraire entre deux points géographiques. Dans certaines représentations, les routes peuvent être subdivisées en arcs pour respecter un critère de longueur maximale de ces unités, et à chaque intersection. Chaque arc routier peut posséder plusieurs attributs, dont :

— Une géométrie composée d'une suite de points de coordonnées reliés entre eux. Le sens allant du premier au dernier point de cette suite est appelé sens de digitalisation de l'arc.

— Un sens de circulation. Par convention, on note F (*from*) le sens de digitalisation, T (*to*) le sens inverse et B (*both*) les deux sens.

— Une vitesse réglementaire.

— Des règles d'accessibilité sous la forme d'une variable binaire pour chaque mode de transport.

D'autres attributs comme le niveau hiérarchique (e.g route principale, route secondaire, etc) ou le nombre de voies peuvent également être intégrés selon la base de données choisie. Il existe plusieurs bases de données proposant un graphe routier à l'échelle mondiale. Parmi les plus célèbres, on peut citer HERE Map Content (anciennement Navstreets) [HER24] fourni par l'entreprise HERE et le réseau d'OpenStreetMap (OSM) [Ope17], une initiative collaborative et *open source* de cartographie.

2.1.1.2 Réseau de transports en commun

Les réseaux de transport en commun sont constitués d'arrêts reliés entre eux par des lignes de métro, bus, tramway, etc. Le fonctionnement de chaque ligne est assuré par des tournées de véhicules tout au long de la journée, à des horaires prédéfinis et communiqués aux usagers. Ces tournées peuvent ne pas être identiques tout au long de la journée, en raison par exemple d'itinéraires alternatifs ne desservant pas tous les arrêts. De plus, les horaires des transports en commun changent souvent au cours de l'année (notamment pendant l'été et les jours fériés). Ainsi, pour décrire un réseau de transport en commun, il est plus pertinent de décrire les différentes tournées de véhicules planifiées plutôt que les lignes elles-mêmes.

En 2006, Google présente le General Transit Feed Specification (GTFS) [Goo23a], un standard informatique de stockage et de communication d'informations relatives aux réseaux de transport en commun. Le GTFS d'un réseau est constitué de plusieurs fichiers contenant les informations sur ses opérateurs de transport (agency.txt), ses lignes (routes.txt), ses arrêts (stops.txt), ses itinéraires (trips.txt), ses régimes d'horaires (calendar.txt) et les horaires de chaque tournée pour chaque arrêt (stops_times.txt). D'autres fichiers optionnels spécifient des jours de perturbation (calendar_dates.txt), des correspondances entre arrêts (transfers.txt), les formes géométriques des itinéraires (shapes.txt) et les fréquences de desserte d'une tournée pour les arrêts n'ayant pas d'horaire fixe (frequencies.txt). Tous ces fichiers sont organisés en base de donnée relationnelle (c.f. figure 2.1), ce qui permet d'effectuer des requêtes sur les horaires ou de représenter le réseau sur une carte. Les calculateurs d'itinéraires fusionnent les données GTFS au réseau routier afin de générer des feuilles de route pour les usagers. Enfin, le format GTFS-realtime étend le format GTFS en y ajoutant la possibilité d'obtenir la position des véhicules en direct, et donc de fournir une estimation précise des temps d'attente aux arrêts.

La plupart des grandes villes disposent d'une représentaton GTFS de leur réseau de transport en commun, et fournissent généralement les fichiers en

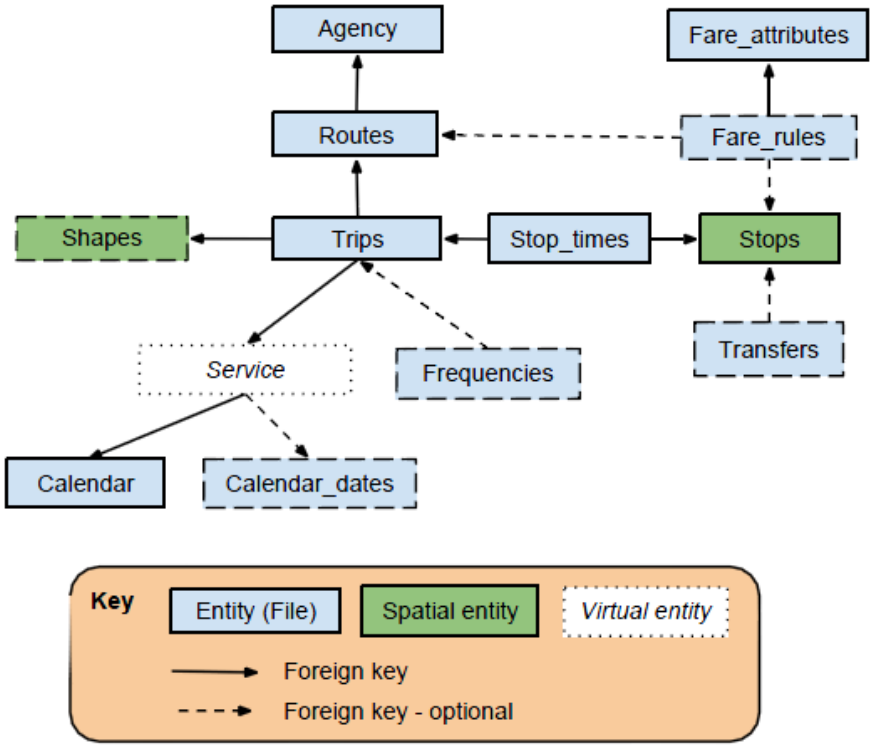


FIGURE 2.1 – Schéma de l'organisation des fichiers GTFS en base de données relationnelle (source : *data-transport.org*)

open data. En France, le site *transport.data.gouv* recense tous les fichiers GTFS disponibles et à jour.

2.1.2 Découpage administratif

Chaque pays possède des frontières et des subdivisions politiques de son territoire. En France, on distingue par exemple les régions, départements et communes (voire communautés de communes, agglomérations, métropoles, etc.). A cela s'ajoutent des divisions territoriales qui n'ont pas de sens politique. Les Ilots Regroupés pour l'Information Statistique (IRIS), par exemple, sont des unités territoriales en France, d'environ 2000 habitants chacune, utilisés dans les enquêtes de l'Insee pour recenser et mesurer des indicateurs démographiques [Ins16]. Dans certains pays, la notion de commune n'existe pas, et peut-être remplacée par celle de code postal. Toutes ces données sont généralement distribuées en *open data* par les services publics.

2.1.3 Occupation du sol

L'observation spatiale de la Terre à partir d'images RADAR ou optiques permet de connaître la nature des sols dans le monde entier. Par exemple, les satellites de la mission européenne Sentinel orbitent autour de la Terre et mettent régulièrement à jour une base de données d'image. Ces images peuvent ensuite être utilisées pour classifier chaque pixel du sol afin d'identifier le *land cover* biogéographique (e.g. champ, forêt, étendue d'eau, zone construite) et le *land use* socio-économique (e.g. zone agricole, zone industrielle, zone résidentielle) [FBN19].

La connaissance de l'emplacement des bâtiments et de leurs fonctions permet d'obtenir des informations sémantiques sur la nature d'un déplacement. Des bases de données telles que la BD-TOPO [Ins24b] fournie par l'Institut national de l'information géographique et forestière (IGN) en France agrègent les emprises au sol des bâtiments issues du cadastre, les zones d'activité labélisées et les informations SIG classiques (noms des lieux, services publics, etc.).

2.1.4 Lieux d'intérêt

Il existe plusieurs sources de données de points d'intérêt (POI) géographiques, chacune ayant ses propres avantages et inconvénients. Les principales bases de données possèdent des API pour effectuer des requêtes de points d'intérêt sur un territoire :

— OpenStreetMap [Ope17] : tout comme le réseau routier, les points d'intérêts sont disponibles gratuitement et mis à jour par une communauté active. En revanche, la qualité des données peut varier en fonction des régions en raison de la nature collaborative du projet.

— Google Places API [Goo23b] : un service qui permet d'accéder à des données sur des POI tels que des entreprises, monuments, commerces autres lieux. Il offre une grande quantité de données, est facile à utiliser et intègre d'autres services de Google. Il fournit également des données riches, comme des avis d'utilisateurs, des taux de fréquentation horaire et des photos. Cependant, l'API n'est pas gratuite et peut avoir des restrictions sur la manière dont les données peuvent être utilisées.

— HERE Places API [HER23] : un service qui fournit des données et des informations de navigation sur des POI. L'API offre une couverture mondiale, des données en temps réel et une intégration facile avec d'autres services Here. Comme Google Places, elle n'est pas gratuite.

En dehors de ces API, il existe des jeux de données connus dans la littérature, collectés à partir de services de référencèrent de POI tels que Yelp [Yel]. De plus, les services publics recensent les adresses et raisons sociales de toutes les entreprises sur leur territoire, ce qui permet de constituer des bases de données exhaustives des commerces et autres lieux d'activité. En France, la base de données Sirène [Ins24a] permet d'accéder librement à ces informations.

2.1.5 Evénements

Des événements indépendants du réseau de transport et des usagers modifient l'état de la mobilité sur un territoire. La météo, par exemple, influence les choix de mode de transport, la vitesse de circulation et le comportement de conduite [Ely+23]. Dans [Naw+20a], les données météorologiques sont couplées à des données de géolocalisation pour améliorer la prédiction du mode de transport en raison d'une forte corrélation du choix modal avec la météo.

La mobilité est également influencée par des événements sociaux réguliers (e.g. jours de repos traditionnels sur le territoire observé, heures de pointe, etc.) ou exceptionnels (e.g. grèves, accidents, événements sportifs, etc.). La connaissance de la sociologie d'un territoire est donc primordiale pour analyser les déplacements des usagers. Toujours en détection du mode de transport, la variable *weekday*, qui désigne le jour de la semaine, est utilisée en entrée des modèles de classification utilisés dans [Naw+20b] et [Gir+22].

Les données contextuelles sont indépendantes des usagers, mais elles apportent des informations sur les tendances de déplacements et les comportements en vigueur sur un territoire et une période données. La prochaine section présente les données agrégées, qui sont des projections des données individuelles d'un groupe d'usagers sur des données contextuelles.

2.2 Données agrégées

Les données agrégées sont le résultat de l'analyse du comportement d'un groupe d'usagers dans un contexte spatial, temporel et social. Elles sont généralement reliées à un ou plusieurs types de données contextuelles, et viennent y apporter des informations statistiques macroscopiques sur la population.

2.2.1 Comptages en section

Pour caractériser la fréquentation d'une route, en déterminer les heures de pointe et mesurer d'éventuelles congestions, il faut compter les usagers qui y circulent. Selon la nature de la route et des modes de transport autorisés, il existe plusieurs méthodes pour mesurer le nombre d'usagers. Pour les véhicules, des systèmes de tubes pneumatiques permettent de détecter et différencier les voitures des motos et vélos, et d'en mesurer la vitesse moyenne. Des algorithmes de reconnaissance d'images basés sur du *Deep Learning* permettent de compter les usagers à partir d'images provenant de caméras installées sur la zone d'intérêt [Cia+22]. De plus, certains fournisseurs de service de navigation tels que HERE Technologies ou TomTom disposent des trajectoires de leurs utilisateurs. Grâce à un algorithme de *map matching*, les points de localisation sont agrégés aux tronçons routiers afin de calculer, pour un tronçon et une période donnée, le nombre de véhicules captés ainsi que leurs vitesses. Bien que ces données soient disponibles en temps réel dans de nombreux pays, elles ne sont pas

exhaustives contrairement aux méthodes traditionnelles de comptage, et le nombre de véhicules captés doit être redressé.

2.2.2 Flux origine-destination

Compter les usagers sur une section de route tout au long de la journée permet de répondre à la question : "Quand les usagers se déplacent-ils ?" (c.f. chapitre 1). Cependant, pour répondre aux questions "Où les utilisateurs se déplacent ?", c'est-à-dire déterminer leur origine et leur destination, il est nécessaire de connaître au préalable leurs points d'entrée et de sortie sur la zone d'étude.

Avec des données géolocalisation et un algorithme de *map matching*, il est possible de générer une matrice origine-destination (OD) entre la zone d'intérêt et les zones adjacentes ou sur un ensemble de zones.

Dans les réseaux de transport payants tels que les autoroutes ou les réseaux de transport en commun, les points d'entrée sont relevés à l'entrée (péage ou portique). Si la sortie est également relevée, il est possible de générer des matrices OD entre les noeuds du réseau (péages ou arrêts). Dans le cas où seule l'entrée est relevée (e.g. dans un bus), il faut utiliser un modèle de simulation basé sur des hypothèses de fréquentation pour estimer les flux OD.

2.2.3 Enquêtes ménage-déplacement

Les enquêtes ménage-déplacement (EMD) sont des études réalisées par les autorités publiques pour comprendre les habitudes de déplacement des individus dans une zone géographique spécifique. Ces enquêtes recueillent des informations détaillées sur les voyages effectués par les personnes ("Qui ?"), y compris le mode de transport utilisé ("Comment ?"), la destination ("Où ?") et l'heure ("Quand ?") et la raison du déplacement ("Pourquoi ?") [ris21]. Les données recueillies sont ensuite utilisées pour planifier et améliorer les infrastructures de transport, pour évaluer l'impact des politiques de transport existantes et pour développer de nouvelles stratégies de mobilité. Les EMD sont essentielles pour une planification efficace et durable des transports. Elles permettent aux décideurs de prendre des décisions éclairées basées sur les besoins réels des citoyens et contribuent à créer des villes plus accessibles et plus durables.

Ces enquêtes sont cependant peu fréquentes (une à deux fois par décennie) car très coûteuses et essentiellement déclaratives (les données proviennent des déclarations des usagers et ne sont pas observées de façon précise et objective) [Bac+19] [Gon+12]. L'outil en ligne Mobiliscope [Val+23] recense les résultats d'EMD conduites en France, au Canada ou en Amérique latine dans un tableau de bord cartographié interactif.

2.2.4 Tableaux de bord d'analyse de la mobilité

De nombreux fournisseurs de service numériques (i.e. applications smartphone et réseaux cellulaires) collectent des données utilisateur et les agrègent de manière à reproduire les indicateurs d'une EMD en temps réel et à moindre coût

sous la forme de tableaux de bord interactifs de l'état de la mobilité. Comme illustré sur la figure 2.2, ces différents produits ne se positionnent pas tous de la même manière sur les cinq axes de connaissance de la mobilité présentés dans le chapitre 1.

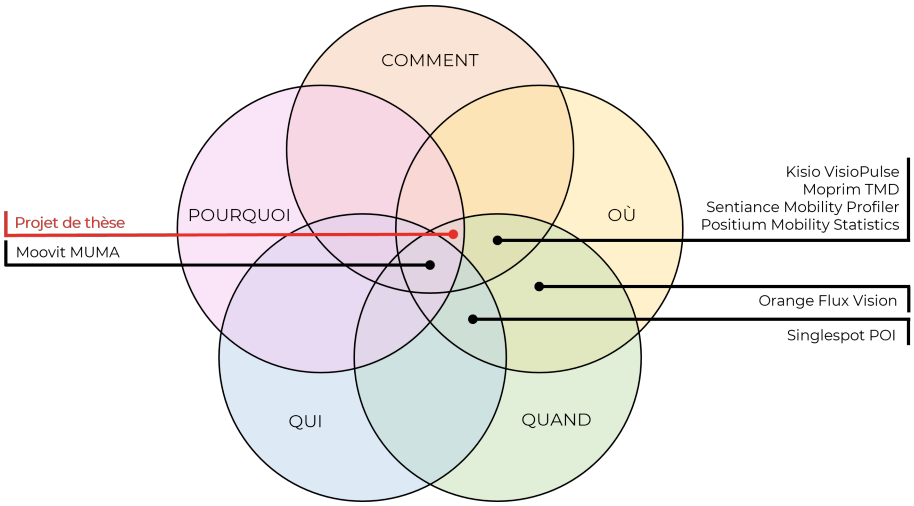


FIGURE 2.2 – Positionnement d'un échantillon de produits industriels d'analyse de la mobilité par rapport aux cinq axes présentés dans le chapitre 1.

La prochaine section présente les différents types de données utilisateur et les attributs associés. Les sources de données disponibles en analyse de la mobilité, leurs avantages et inconvénients sont également détaillés.

2.3 Données utilisateur

Les données utilisateur sont des points de mesure successifs collectés par des capteurs embarqués sur des personnes. Pour un utilisateur donné, les données collectées peuvent être vues comme une série temporelle de mesures à des instants $(t_i)_{i\in I} = \mathcal{T}$. A chaque instant t_i, on dispose des valeurs mesurées par tout ou partie des capteurs embarqués par l'utilisateur. Dans la suite, on note $I = [\![1, n]\!] \subset \mathbb{N}$ l'ensemble des indices des instants t_i, $n = |I|$ le nombre d'observations temporelles et $T = t_n - t_1$ la durée d'observation.

Cette section présente successivement les différents types de données utilisateur, leurs attributs, leurs sources et leur utilisation en analyse de la mobilité.

2.3.1 Géolocalisation

La géolocalisation consiste à déterminer la position géographique d'un objet ou d'une personne, c'est-à-dire à calculer ses coordonnées géographiques (latitude, longitude et parfois altitude). En analyse de la mobilité, on cherche à connaître la position d'utilisateurs au cours du temps sous la forme d'une série temporelle de points dans l'espace. Dans cette thèse, on note $I_p \subset I$ l'ensemble des indices des instants d'observation de la position d'un utilisateur, et $n_p = |I_p|$ le nombre d'observations. On considère la série temporelle des coordonnées suivantes :

$$(p_i)_{i \in I_p} = (\text{lon}_i, \text{lat}_i, \text{alt}_i)_{i \in I_p} = (\lambda_i, \varphi_i, h_i)_{i \in I_p} \tag{2.1}$$

où λ, φ et h symbolisent respectivement les longitude, latitude et altitude. La connaissance de la position d'un utilisateur au cours du temps permet de calculer des attributs de localisation, qui sont présentés dans les paragraphes suivants.

Distance Il existe plusieurs manières de calculer la distance géographique entre deux points à partir de leurs coordonnées, selon le niveau d'approximation et le référentiel utilisés. Dans le cadre de nos travaux, nous avons choisi d'utiliser la formule de haversine [Inm49] qui relie la distance d entre deux points A et B d'une sphère uniquement à leurs latitudes φ_A, φ_B et longitudes λ_A, λ_B :

$$\text{hav}(\frac{d}{r}) = \text{hav}(\varphi_B - \varphi_A) + \cos(\varphi_A)\cos(\varphi_B)\text{hav}(\lambda_B - \lambda_B) \tag{2.2}$$

où r est le rayon de la sphère et hav : $\theta \mapsto \sin^2(\frac{\theta}{2})$ la fonction haversine. Le rayon de la Terre étant d'environ 6371 kilomètres, on obtient la distance en mètres entre deux positions observées aux instants t_i et $t_{i'}$, $i, i' \in I_p$:

$$d_{i,i'} \approx 12742.10^3 \arcsin\left(\sqrt{\sin^2(\frac{\varphi_{i'} - \varphi_i}{2}) + \cos(\varphi_{i'})\cos(\varphi_i)\sin^2(\frac{\lambda_{i'} - \lambda_i}{2})}\right) \tag{2.3}$$

Vitesse La vitesse est définie comme étant le rapport entre la distance parcourue et le temps de parcours. Elle se mesure sur un intervalle temporel $[t_i, t_i']$ avec $i, i' \in I_p$:

$$\bar{v}_{i,i'} = \frac{\|p_{i'} - p_i\|}{t_{i'} - t_i} \tag{2.4}$$

où p_i, p_j sont les vecteurs de position aux instants $t_i, t_{i'}$ et $\|.\|$ une norme dans l'espace des coordonnées. On parle alors de vitesse moyenne $\bar{v}_{i,i'}$ sur l'intervalle $[t_i, t_{i'}]$. En passant à la limite, c'est-à-dire en réduisant les intervalles temporels de mesure à des instants $(t_i)_{i \in I_p}$, on obtient les vitesses instantanées $(v_i)_{i \in I_p}$ définies comme :

$$\forall i \in I_p, \quad v_i = \lim_{\Delta t \to 0} \frac{\|p_{i'} - p_i\|}{t_{i'} - t_i} = \lim_{\Delta t \to 0} \frac{d_{i,i'}}{\Delta t} = \frac{d}{dt} p_i \quad \text{avec} \quad t_{i'} = t_i + \Delta t \quad (2.5)$$

Certains appareils estiment la vitesse instantanée dans la direction du relèvement en plus des coordonnées géographique des points collectés. Les vitesses sont généralement exprimées en mètres par seconde ($m.s^{-1}$) ou kilomètres par heure (km/h).

Relèvement Le relèvement (aussi appelé *Bearing* ou *Heading*) est l'angle dans lequel un objet se déplace par rapport au Nord géographique. Sa valeur s'exprime en degrés sur l'intervalle $[0, 360[$ ou $[-180, 180[$. Un relèvement de zéro signifie que l'objet se déplace exactement vers le Nord, et la valeur augmente dans le sens horaire. Les valeurs de relèvement $(\theta_i)_{i \in I_p}$ peuvent être mesurées en même temps que les coordonnées géographiques.

Dans les sous-parties suivantes, nous présentons les différentes sources de données de géolocalisation, leurs avantages et inconvénients, ainsi que leurs utilisations possibles en analyse de la mobilité.

2.3.1.1 Géolocalisation par satellite

Le Global Positioning System (GPS) est un système de navigation satellite qui permet de déterminer la position géographique précise d'un récepteur n'importe où sur Terre grâce à une constellation de satellites en orbite. Actuellement, il y a 31 satellites GPS opérationnels en orbite, répartis sur six plans orbitaux. Les récepteurs GPS sont des appareils portables ou embarqués dans des véhicules, des téléphones portables, etc. Ils captent les signaux émis par les satellites pour calculer leur position en utilisant le principe de trilatération. En recevant des signaux de plusieurs satellites, le récepteur peut calculer sa propre position en mesurant le temps que mettent les signaux pour atteindre la terre.

Le GPS a été développé par le Département de la Défense des États-Unis dans les années 1970 pour un usage militaire. Il est devenu opérationnel en 1978. En 1983, le président Ronald Reagan a annoncé que le GPS serait rendu accessible aux civils, bien que la précision des signaux ait été délibérément dégradée pour des raisons de sécurité nationale. Cette dégradation a été levée en 2000, permettant ainsi une précision plus élevée pour les utilisateurs civils. La précision du GPS peut varier en fonction de plusieurs facteurs, y compris le nombre de satellites visibles, l'obstruction du signal par des bâtiments ou des arbres, et la qualité du récepteur. En conditions idéales, le GPS peut fournir une précision d'une dizaine de mètres.

Dès son ouverture au public, le GPS a été largement utilisé dans la navigation terrestre, maritime ou aérienne. Les récepteurs GPS ont commencé à être intégrés dans des appareils grand public tels que les systèmes de navigation de voiture. Les données GPS ont été de plus en plus utilisées pour analyser

la mobilité des individus et des véhicules. Ces données sont précieuses pour la planification urbaine, la gestion des flottes, la recherche en sciences sociales, etc. Elles permettent de suivre les déplacements, d'optimiser les itinéraires, de surveiller la vitesse, et même de prédire la congestion routière.

Au fil des ans, le GPS a évolué avec l'ajout de signaux améliorés, comme le GPS différentiel, qui augmente la précision. De plus, d'autres systèmes de navigation par satellite, tels que le GLONASS russe, le Galileo européen, et le Beidou chinois, sont également devenus opérationnels, élargissant ainsi les options disponibles pour la collecte de données de localisation. C'est pourquoi on parle aujourd'hui dans la communauté scientifique de GNSS (Global Navigation Satellite Systems) pour désigner les différents systèmes de manière neutre et indifférenciée.

Avantages Les GNSS ont une précision élevée en extérieur et en surface, de l'ordre d'une dizaine de mètres [BSG19]. De plus, les constellations de satellites assurent une couverture mondiale et résiliente, avec notamment l'interopérabilité entre le GPS et Galiléo qui permet à l'un des systèmes de prendre le relai sur l'autre en cas de défaillance.

Inconvénients L'utilisation de satellites nécessite une ligne de vue directe avec le récepteur, ce qui limite l'utilisation d'un GNSS en intérieur ou au milieu d'obstacles du type canyon qui dévient le signal et dégradent fortement la précision de localisation. En analyse de la mobilité, ce problème est bien connu pour les zones urbaines denses aussi appelées canyons urbains en raison de la grande hauteur des immeubles autour du réseau de routes.

De plus, l'utilisation d'un GNSS consomme beaucoup d'énergie comparativement à d'autres méthodes de géolocalisation, ce qui peut poser problème lorsque le récepteur est un appareil mobile avec une batterie limitée.

Les données GNSS permettent de façon immédiate de reconstituer des matrices origine-destination (OD) à partir des trajectoires observées pour connaître les flux de déplacement des usagers. Elles ont également d'autres applications dans la littérature, comme la détection du mode de transport [Naw+20b] ou de l'activité [Irs+21].

2.3.1.2 Téléphonie mobile

La communication sans fil entre appareils mobiles tels que des téléphones ou des tablettes passe par un réseau cellulaire. Le territoire est divisé en cellules, qui sont des zones géographiques couvertes par des antennes-relais. Chaque antenne-relais est reliée à une station de transmission de base (*Base Transceiver Station* ou BTS), qui gère les communications dans sa cellule. Les antennes-relais émettent et reçoivent des ondes radio, qui sont des signaux électromagnétiques utilisés pour transmettre des informations (voix, données, texte, etc.).

Les réseaux cellulaires ont évolué au fil du temps, en passant par différentes générations, qui correspondent à des améliorations technologiques. La première génération (1G) était basée sur la transmission analogique de la voix. La

deuxième génération (2G) a introduit la transmission numérique de la voix et des données. La troisième génération (3G) a permis l'accès à Internet et aux services multimédias. La quatrième génération (4G) a augmenté les débits et la bande passante. La cinquième génération (5G) apporte des vitesses encore plus rapides et une meilleure connectivité.

Pour gérer la mobilité des utilisateurs qui se déplacent d'une cellule à une autre, les stations de base communiquent entre elles et avec des centraux téléphoniques, qui sont des équipements qui acheminent les appels vers leur destination. Lorsqu'un utilisateur change de cellule, il y a un transfert intercellulaire, qui consiste à basculer la liaison vers la nouvelle antenne-relais, sans interrompre la communication. La connaissance des coordonnées géographiques des BTS les plus proches de l'appareil mobile permet de connaître la position approximative de son utilisateur. On distingue deux types de données collectées par le biais du réseau cellulaire [Bac+19] :

— *Call Detail Records* (CDR) : collecte active (i.e. suite à une action de l'utilisateur) de l'identifiant des BTS les plus proches lors d'événements tels que des appels, SMS, l'utilisation du réseau 3G/4G, etc.
— *Location Area Updates* (LAU) : collecte passive (i.e. sans intervention de l'utilisateur) de l'identifiant des BTS les plus proches à intervalle régulier.

Avantages Le principal avantage de l'utilisation des réseaux cellulaires pour géolocaliser les utilisateurs est l'ubiquité des appareils mobiles. La quasi-totalité des personnes possédant au moins un appareil mobile, les principaux fournisseurs d'accès à un réseau cellulaire se retrouvent mécaniquement en possession d'une base de données représentative de la population globale. De plus, ces fournisseurs connaissent généralement leurs parts de marché sur un territoire et peuvent effectuer un redressement pour estimer le nombre d'utilisateurs présents à un instant donné.

Inconvénients Contrairement à la géolocalisation par satellite, on ne mesure pas directement la position de l'utilisateur, mais uniquement les positions des antennes relais les plus proches de lui. Il est possible de trianguler la position approximative de l'utilisateur en utilisant la force du signal capté pour chacune des antennes relais. La précision de méthode de géolocalisation varie fortement avec la qualité de couverture du réseau, et ne dépasse généralement pas plusieurs centaines de mètres [Kys14]. Cependant, cette pratique est interdite dans de nombreux pays comme la France par souci de protection des données personnelles. De manière générale, il est désormais inenvisageable d'accéder à de telles données en France, et dans d'autres pays européens, en raison du durcissement de la règlementation en vigueur (particulièrement dans un contexte industriel).

De nombreuses recherches portent sur la reconstruction de matrices OD à partir d'un découpage territorial et de données de géolocalisation provenant de réseaux cellulaires ([Ale+15], [Too+15], [DL+14], [NWC18], [Li+18]). Dans

[DL+14], un tableau de bord interactif est produit pour visualiser les flux de déplacement à Abidjan, en Côte d'Ivoire.

D'autres approches tentent de comprendre et modéliser le comportement des usagers ([GHB08], [Cal+13]), notamment en étudiant la corrélation des trajectoires issues du réseau cellulaire avec des données agrégées de nature socio-démographiques ou économiques ([Pap+15], [NWC18]). Dans [Wan+18], les auteurs vont même plus loin et étudient la corrélation entre la mobilité de l'utilisateur et ses habitudes de navigation sur internet (ses sites les plus fréquentés, son temps de navigation, etc.).

Les données de téléphonie mobile permettent de modéliser la demande de transport ([Too+15], [Wan+13], [Hua+18]), de reconstruire des itinéraires par le biais d'un mapping au réseau routier ([Asg+16]), mais également d'estimer la densité de population ([Bac+17], [Kho+16], [Kho+18]), l'état du trafic ([Don+15]) ou encore les flux de passagers ([Zho+16]).

Enfin, ces données sont utilisées pour inférer le mode de transport (c.f. chapitre 4) et le but du déplacement ([JFG17], [CBM14], [Ale+15]).

2.3.1.3 WiFi

A l'instar de la triangulation cellulaire, la géolocalisation par WiFi repose sur la mesure de la puissance du signal WiFi émis par les bornes d'accès environnantes, et sur la comparaison avec une base de données qui associe chaque borne à une localisation géographique. La géolocalisation par WiFi présente plusieurs avantages par rapport aux autres méthodes, comme le GNSS ou la triangulation cellulaire. Elle est intéressante en milieu urbain ou en intérieur, où les signaux satellites sont souvent perturbés par les obstacles. Elle est aussi plus précise que la triangulation par données cellulaires [Kys14]. Enfin, elle est peu gourmande en énergie, car elle ne nécessite pas d'activer un récepteur dédié. Les données de géolocalisation par WiFi sont notamment utilisées pour les études de mobilité à l'intérieur de bâtiments [Shu+15].

2.3.1.4 Bluetooth

La géolocalisation par Bluetooth à faible énergie (*Bluetooth Low Energy* ou BLE) est une technique qui permet de déterminer la position d'un appareil mobile à partir de la mesure de sa distance avec d'autres appareils équipés de la même technologie. Le principe est basé sur l'émission et la réception de signaux radio à faible puissance entre les appareils Bluetooth, qui peuvent ainsi calculer leur distance relative en fonction du temps de propagation et de la force du signal à l'instar de la géolocalisaiton par WiFi. La géolocalisation par Bluetooth présente plusieurs avantages, comme une faible consommation d'énergie, une bonne précision dans les espaces intérieurs et une facilité d'utilisation. Elle peut être utilisée pour des applications variées, comme le suivi d'objets, la localisation de personnes ou la réalité augmentée. Cependant, elle est peu adaptée à des environnements extérieurs en raison de la faible puissance des signaux radios

utilisés. Dans des conditions optimales, un récepteur ne capte pas de signal Bluetooh au-delà d'une centaine de mètres.

Dans [Ver+12], les auteurs mènent une étude de cas sur la dynamique spatio-temporelle des mouvements humains lors d'événements de masse en utilisant la technologie Bluetooth. En déployant des scanners Bluetooth à 22 endroits, ils extraient les trajectoires des visiteurs et analysent leurs schémas. [Del+12] se concentrent sur l'analyse des séquences spatio-temporelles dans les données de suivi Bluetooth. Ils appliquent des méthodes d'alignement de séquences pour examiner les schémas comportementaux des visiteurs suivis par Bluetooth lors d'une foire commerciale en Belgique. [YM19] analysent les données Bluetooth dans le cadre d'une étude de cas à Austin au Texas pour comprendre les schémas de déplacement et les niveaux de congestion dans la ville. Les données Bluetooth peuvent également être utilisées non pas pour reconstituer des trajectoires, mais pour calculer des temps de parcours (et donc des vitesses). [Hag+10] présentent un système composé de deux bornes Bluetooth séparées de deux miles permettant de calculer les temps de parcours des véhicules connectés sur des autoroutes. Ils démontrent le potentiel des capteurs Bluetooth pour la collecte de données de vérité terrain dans la recherche sur les transports.

2.3.2 Centrale inertielle

Une centrale inertielle est un instrument de mesure initialement utilisée en navigation, capable de mesurer les mouvements angulaires et l'accélération. De nos jours, de nombreux appareils mobiles tels que les smartphones modernes sont équipés d'une *Inertial Measurement Unit* (IMU) qui s'apparente à une centrale inertielle et intègre deux types de capteurs :

— Trois accéléromètres pour mesurer l'accélération dans chaque direction de l'espace.

— Trois gyroscopes pour mesurer la vitesse angulaire autour de chaque axe de l'espace.

Dans la suite de cette thèse, on considère l'ensemble d'indices $I_a \subset I$ (de cardinal $n_a = |I_a|$) des instants d'observation de mesures IMU collectées sur un utilisateur donné.

Accélération L'accélération est la variation de la vitesse. Elle s'exprime généralement en mètres par seconde au carré ($m.s^{-2}$) et se mesure dans les trois directions de l'espace x, y et z. Les valeurs d'accélération collectées sur un utilisateur constitue une série temporelle en trois dimensions :

$$(a_i)_{i \in I_a} = (a_i^x, a_i^y, a_i^z)_{i \in I_a} \tag{2.6}$$

L'accélération étant un vecteur réel en trois dimensions, il est possible de calculer sa magnitude à l'aide de la norme euclidienne $|.|$:

$$\forall i \in I_a, \quad \|a_i\| = \sqrt{a_i^{x\,2} + a_i^{y\,2} + a_i^{z\,2}} \tag{2.7}$$

Vitesse de rotation La vitesse de rotation se mesure autour de chaque axe en radians par seconde (rad/s) et constitue également une série temporelle à trois dimensions dont on peut calculer la magnitude :

$$(g_i)_{i \in I_a} = (g_i^x, g_i^y, g_i^z)_{i \in I_a} \tag{2.8}$$

$$\forall i \in I_a, \quad \|g_i\| = \sqrt{g_i^{x\,2} + g_i^{y\,2} + g_i^{z\,2}} \tag{2.9}$$

Les données d'accélération et de gyroscope sont notamment utilisées pour résoudre le problème de détection du mode de transport ([Car+18], [Alo20] et [IG20]).

2.3.3 Autres capteurs présents dans les smartphones

En plus d'une centrale inertielle, les smartphones modernes intègrent d'autres types de capteurs qui présentent un fort potentiel d'information sur la mobilité des utilisateurs avec une faible consommation d'énergie.

Magnétométrie Les smartphones modernes intègrent trois magnétomètres qui mesurent l'intensité du champ magnétique dans les trois directions de l'espace. Les mesures issues des magnétomètres sont initialement utilisées en complément de l'accélération tri-axiale dans des problèmes de HAR ([KGQ12], [AB10]). De plus, les bâtiments étant construits avec des matériaux ferromagnétiques et remplis d'appareils électroniques, les mesures du champ magnétique sont très bruitées ([SSR10], [Goz+11]). Chaque pièce possède une signature électromagnétique propre en raison de ces disparités de mesure. Dans [Shu+15], les auteurs utilisent les intensités magnétiques combinées aux signaux WiFi collectés de façon opportuniste pour identifier la pièce dans laquelle se trouve un utilisateur avec une faible consommation énergétique (par rapport aux techniques similaires utilisant seulement les signaux WiFi).

Barométrie Parmi les capteurs intégrés dans les smartphones modernes, on trouve également un baromètre qui mesure la pression atmosphérique afin d'améliorer la mesure de l'altitude. Des changements soudains de pressions mesurées peuvent indiquer que l'utilisateur se déplace, et la variation d'altitude associée donne des informations sur le mode de déplacement utilisé. Dans [San+14], la mesure de la pression est utilisée pour détecter si un utilisateur de smartphone est immobile, s'il marche ou s'il se déplace avec un véhicule.

Audiométrie Le microphone est un composant de base de tout téléphone. L'environnement sonore d'un utilisateur en déplacement est fortement corrélé au mode de transport utilisé (bruit de moteur, bruit ambiant des transports en

commun, etc). Dans [LLL22], les auteurs présentent un modèle de *Deep Learning* capable de détecter le mode de transport à partir d'enregistrements audio. Pour minimiser la consommation énergétique, le microphone n'est activé que lorsque certains critères d'accélération sont remplis, mais seuls les enregistrements audio interviennent dans la classification.

Intensité lumineuse Les capteurs d'intensité lumineuse des smartphones permettent de réduire la consommation énergétique en adaptant la luminosité de l'écran en fonction de l'éclairage ambiant. De la même manière que pour l'intensité magnétique, les différentes pièces d'un bâtiment possèdent un environnement lumineux qui leur est propre. Des systèmes basés sur des balises LED émettant une signature lumineuse spécifique dans chaque pièce permettent de géolocaliser un utilisateur à l'intérieur d'un bâtiment ([Yan+15], [Kuo+14]), mais nécessitent l'installation d'un matériel dédié. Dans [ZZ16], les auteurs s'affranchissent de balises lumineuses additionnelles et parviennent à discriminer différentes pièces à partir de la seule signature lumineuse de tubes fluorescents. D'autres systèmes fonctionnent pour des sources de lumière arbitraires en combinant les intensités lumineuses aux accélérations mesurées ([XZH15], [Zha+17]). Dans [WYM18], un modèle de *Deep Learning* est utilisé pour la géolocalisation en intérieur à partir des intensités lumineuses et magnétiques mesurées par un smartphone.

Les données utilisateur couvrent un large spectre d'informations relatives aux usagers. Les deux principaux types de données utilisateur sont la géolocalisation et les mesures inertielles d'un usager dans le temps. D'autres types de données utilisateur basés notamment sur les capteurs disponibles dans les smartphones récents sont utilisés en analyse de la mobilité. La prochaine section propose une courte synthèse des données de mobilité présentées dans ce chapitre.

2.4 Synthèse

Il existe de nombreux types de données de mobilité, avec pour chacun plusieurs sources de données potentielles. Comme illustré par la figure 2.3, nous proposons dans ce chapitre une typologie permettant de distinguer données contextuelles, données utilisateur et données agrégées. Cette dernière catégorie est en réalité la projection de données utilisateur sur des données contextuelles. C'est également le type de données que nous cherchons à reproduire dans le projet de thèse, dans le but d'obtenir des résultats similaires à ceux d'une EMD à moindre coût et avec des données observées (et non pas déclaratives) et plus récentes.

Les deux principales sources de données utilisateur utilisées dans la littérature sont la géolocalisation et les capteurs intégrés dans les smartphones. Dans le chapitre 4, nous montrons comment ces données servent notamment à résoudre le problème de détection du mode de transport (TMD).

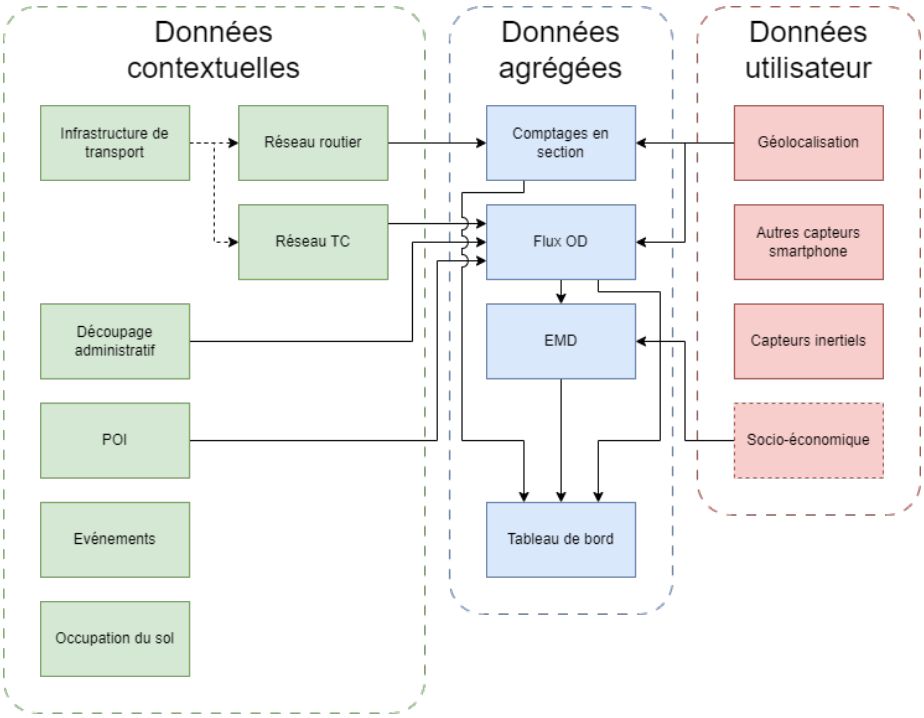


FIGURE 2.3 – Hiérarchie des différents types de données de mobilité selon la typologie introduite dans ce chapitre.

Chapitre 3

Comment et pourquoi utiliser un SMA coopératif pour l'analyse de la mobilité ?

Dans le chapitre 1, le contexte industriel de Citec a été présenté. En particulier, la volonté de caractériser les axes "Comment" et "Pourquoi" de l'état de la mobilité sur un territoire à partir de données de mobilité introduites dans le chapitre 2 a permis d'isoler deux cas d'usage :

— La classification du mode de transport, ou *Transport Mode Detection* (TMD), lors des déplacements des utilisateurs.

— La classification de l'activité lors des arrêts des utilisateurs.

Dans le cadre de cette thèse, le choix a été fait de se focaliser sur le cas d'application de la détection du mode de transport afin d'illustrer le potentiel des données utilisateur présentées dans le chapitre 2 avec un problème de classification supervisée. Avant toute chose, il convient de rappeler les trois pré-requis introduits dans le chapitre 1 pour un modèle d'analyse de la mobilité :

— **L'apprentissage en ligne et capacité d'adaptation rapide** : les comportements des usagers peuvent évoluer dans le temps et l'espace, notamment en raison de différences de législations (e.g. sens de circulation, vitesse limite, etc.).

— **Une bonne performance sur des problèmes fortement non linéaires** : les mesures utilisées pour détecter les modes de transport ne permettent généralement pas de discriminer clairement des modes tels que le vélo, le bus et la voiture en milieu urbain (e.g. les itinéraires et vitesses sont sensiblement équivalents). Les frontières de décisions sont floues, et le problème de classification en découlant est fortement non linéaire.

— **Le besoin d'explicabilité** : les analyses de la mobilité sont généralement réalisées à la demande de collectivités locales, et leurs résultats peuvent avoir une influence sur des choix d'urbanisme ayant un impact direct sur la vie des usagers. Il est donc primordial que le modèle utilisé puisse prendre des décisions interprétables et explicables par un utilisateur humain.

Ce chapitre présente successivement différents concepts de résolution d'un problème de classification tel que la détection du mode de transport, nécessaires pour introduire la solution algorithmique proposée dans le chapitre 5. La section 3.1 présente la notion d'apprentissage automatique, et notamment la notion de classification supervisée dans lequel s'inscrit le problème de détection du mode de transport. Dans la section 3.2, différentes techniques d'agrégation de modèles

sont introduites, ainsi que leurs avantages et inconvénients. La section 3.3 est dédiée aux systèmes multi-agents et plus particulièrement à l'apprentissage par contexte. Ces deux notions sont à la base de la solution proposée dans le prochain chapitre. Enfin, le positionnement de la solution algorithmique développée dans le projet de thèse est présenté dans la section 3.4 et vient conclure le chapitre.

3.1 Apprentissage automatique

Le *Machine Learning* (ML) ou apprentissage automatique est un domaine de l'intelligence artificielle qui vise à concevoir des systèmes capables d'apprendre à effectuer une tâche dépendante de données d'entrées, sans être explicitement programmés pour cela, à la manière du processus d'apprentissage observé chez l'humain ou l'animal [Sam59]. La tâche considérée est généralement un processus d'association ou de prise de décision. Dans les deux cas, elle peut être ramenée à un problème de prédiction d'une variable cible (ou **variable à expliquer**) à partir des variables des données d'entrée (ou **variables explicatives**). La variable à expliquer est dite qualitative si elle prend des valeurs discrètes, et quantitative dans le cas contraire.

L'apprentissage automatique consiste à associer p variables explicatives d'un individu X à une variable expliquée y, c'est-à-dire à connaître la relation qui relie X à y. Mathématiquement, cette relation est représentée par la **fonction objectif** f :

$$f(X) = y \tag{3.1}$$

Dans la pratique, il est généralement impossible de connaître explicitement la fonction objectif. On cherche alors la modéliser à partir de n observations $(X_i, y_i)_{i \in [\![1,n]\!]}$ en construisant un modèle $\hat{f}$:

$$\forall i \in [\![1, n]\!], \quad \hat{f}(X_i) = \hat{y}_i \sim y_i = f(X_i) \tag{3.2}$$

La valeur approchée $\hat{y}_i$ est la prédiction de la vraie valeur y_i par le modèle $\hat{f}$.

Dans le cas où la variable à expliquer est qualitative, la fonction objectif est discrète et le modèle $\hat{f}$ est appelé **classifieur**. Le problème d'apprentissage du modèle est alors appelé problème de classification. Si la variable à expliquer est quantitative, la fonction objectif est continue au moins par morceaux et le modèle est appelé **régresseur**. On parle alors de problème de régression.

Il existe quatre principaux types d'apprentissage automatique :

— L'apprentissage supervisé
— L'apprentissage non supervisé
— L'apprentissage semi-supervisé
— L'apprentissage par renforcement

3.1.1 Apprentissage supervisé

L'apprentissage supervisé se fait généralement en deux étapes [Ami20] :

— L'entraînement, durant laquelle le modèle de prédiction est construit à partir d'un sous-ensemble des observations $(X_i, y_i)_{i\in[\![1,n]\!]}$.

— La validation, durant laquelle on génère des prédictions avec le modèle entraîné pour les comparer aux vraies valeurs de la variable cible à l'aide d'une fonction de distance pour évaluer ses performances.

Une fois l'apprentissage terminé, le modèle est utilisé pour prédire la variable cible à partir de nouvelles observations X_i pour lesquelles on ne connaît plus les vraies valeurs y_i. Dans le cas de l'apprentissage en ligne (*online*), il est possible d'alterner les phases d'apprentissage et d'exploitation du modèle pour l'améliorer sur de nouvelles observations sans le ré-entraîner depuis le début [FR+13].

Un problème récurrent de l'apprentissage supervisé est le sur-ajustement (aussi appelé **sur-apprentissage**). Un modèle est dit sur-ajusté lorsqu'il modélise trop finement les données d'entraînement, et donc leur bruit, au point de mal prédire de nouveaux points dans des zones non explorées ou dont la distribution statistique est légèrement différentes de celles des données d'entrée. Pour éviter cela, il est conseillé de constituer une base d'apprentissage représentative par rapport aux données susceptibles d'être prédites lors de l'exploitation du modèle. En particulier, si le comportement statistique des données d'entrée évolue avec le temps, il est important que le modèle soit entraîné avec des données récentes pour tenir compte de cette évolution. Lorsque le modèle ne fait pas de sur-apprentissage sur un ensemble de points, on dit qu'il a une bonne **généralisation** sur cet ensemble.

Parmi les modèles d'apprentissage supervisé, on peut distinguer les modèles paramétriques, qui possèdent une structure fixée et connue à l'avance, des modèles non paramétriques dont la structure dépend des données d'apprentissage. Parmi les modèles non paramétriques, les modèles de voisinage proposent un apprentissage local basé sur des opérations géométriques dans l'espace des variables d'entrée.

3.1.1.1 Modèles paramétriques

Nous appelons modèles paramétriques les modèles statistiques dont l'expression est connue explicitement et ne dépend pas du nombre d'individus dans l'ensemble d'apprentissage [Aze22]. Ces modèles contiennent un nombre fini de paramètres dont les valeurs sont ajustées durant l'entraînement.

Régression linéaire Historiquement, les premières approches d'apprentissage automatique visent à résoudre des problèmes de régression à l'aide de modèles statistiques permettant d'approcher la valeur d'une variable à expliquer à partir des variables explicatives avec l'hypothèse qu'elles lui sont statistiquement corrélées. Dans le cas le plus simple de la régression linéaire, introduit par Francis Galton au début du XIX$^{\text{ème}}$ siècle [Sta01], le modèle $\hat{f}$ est une fonction linéaire (i.e. ses images ne sont que des combinaisons linéaires des variables d'entrée) :

$$\forall i \in [\![1, n]\!], \quad \hat{y}_i = \hat{f}(X_i) = \beta_0 + \sum_{j=1}^{p} \beta_j X_i^j \tag{3.3}$$

L'apprentissage du régresseur consiste alors à optimiser les coefficients $(\beta_j)_{j\in[\![0,p]\!]}$ de ces combinaisons linéaires afin de minimiser l'écart entre les prédictions et les observations réelles. Le problème de régression devient un problème d'optimisation dans lequel la **fonction de coût** considérée correspond au critère des moindres carrés (pour des raisons de différentiabilité) :

$$\min_{(\beta_j)_{j\in[\![0,p]\!]}} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \min_{(\beta_j)_{j\in[\![0,p]\!]}} \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j X_i^j)^2 \tag{3.4}$$

Des approches plus récentes proposent d'améliorer la régression linéaire par l'ajout à la fonction de coût d'une pénalisation par la norme l_2 (ridge [HK70]) ou l_1 (LASSO [Tib96]) du vecteur des coefficients $(\beta_j)_{j\in[\![0,p]\!]}$ afin d'augmenter la généralisation du modèle. Ces deux termes sont combinés dans l'approche Elastic Net [ZH05].

Les modèles linéaires peuvent être adaptés à des problèmes de classification. Si la variable qualitative à expliquer est binaire, il est possible d'entraîner un régresseur linéaire afin de prédire la probabilité d'appartenance à une des deux classes. Les Modèles Linéaires Généralisés (GLM) [NW72] étendent le domaine des régresseurs linéaires en introduisant une fonction de lien dans la résolution du problème des moindres carrés. Dans le cas de la régression logistique (logit) où cette fonction est une sigmoïde ($x \mapsto \frac{1}{1+e^{-x}}$), les prédictions sont homogènes à des probabilités et le modèle est également un classifieur binaire. De plus, il est possible de généraliser la régression logistique à une variable explicative pouvant prendre m modalités en effectuant m régressions successivement entre une classe et toutes les autres afin d'obtenir m probabilités d'appartenance en sortie (régression logistique multi-classes).

Algorithmes Passifs Agressifs Les algorithmes Passifs Agressifs (PA) [Cra+06] sont une famille de modèles linéaires d'apprentissage en ligne qui ne se comportent pas de la même manière selon que la prédiction obtenue à partir d'un nouvel individu est correcte ou non. Dans le premier cas, les coefficients du modèle ne changent pas (comportement passif). Dans le second cas, les coefficients sont incrémentés de façon à ce que le modèle prédise correctement et avec une marge unitaire un nouveau point identique au dernier point observé (comportement agressif). L'agressivité de ces algorithmes est contrôlée par un coefficient devant l'incrément des poids à chaque itération. On distingue deux principales variantes de ces algorithmes : PA-I lorsque l'incrément est linéaire pour le nouveau point (fonction de coût *Hinge*) et PA-II lorsqu'il est quadratique (fonction de coût *squared Hinge*).

Support Vector Machines Les *Support Vector Machines* (SVM) [CST+00] sont une technique de classification supervisée dans laquelle le problème initial est transformé en une recherche d'hyperplans séparant deux classes via un noyau définissant un espace intermédiaire. Dans ce nouvel espace, le problème à résoudre est supposé linéaire. Lorsque le noyau utilisé n'est pas linéaire, les hyperplans de l'espace intermédiaire définissent ainsi des frontières non linéaires dans l'espace initial, bien que la résolution algorithmique soit celle d'un problème linéaire. Cependant, lorsque le noyau est linéaire, le modèle obtenu est également linéaire. Parmi toutes les frontières possibles, les SVM recherchent l'hyperplan maximisant la confiance, c'est-à-dire la distance à la frontière, de chaque côté. De plus, dans le cas où les classes ne sont pas linéairement séparables dans l'espace intermédiaire, il est possible d'accepter les erreurs de séparation pour les points proches de la frontière générée via un paramètre d'équilibrage par souci de généralisation.
Le passage des SVM à m classes peut se faire selon plusieurs stratégies : *un contre tous* qui consiste en la résolution de m problèmes binaires (la classe d'intérêt contre toutes les autres), *un contre un* qui consiste à entraîner $\frac{m(m-1)}{2}$ SVM (une par couple de classes) puis à procéder par vote pour la prédiction, et la stratégie *Crammer-Singer* [CS03] qui est une reformulation du problème de minimisation des SVM intégrant plusieurs classes pour un coût de calcul quadratique.

Intérêt Les modèles paramétriques présentent l'avantage d'être simples à appréhender car leur structure est statique et l'apprentissage consiste à ajuster les coefficients correspondant à chaque variable (ou combinaison de variables). L'analyse de ces coefficients donne des informations sur l'importance relative des variables explicatives, ainsi que la sensibilité du modèle selon ces mêmes variables, ce qui le rend particulièrement explicable au sens de l'interprétabilité [MCB18].

De plus, l'ajustement de ces coefficients est généralement indépendant de l'ordre des individus de la base d'entraînement, ce qui rend les modèles paramétriques compatibles avec l'apprentissage en ligne.

Les modèles paramétriques peuvent avoir de bonnes performances dans certains problèmes fortement non linéaires à condition d'en connaître la structure au préalable (e.g. pour le choix du noyau à utiliser dans un SVM [DRP12]).

Enfin, la capacité d'adaptation aux changements de comportement des individus (se traduisant par des changements de distribution statistique dans les variables explicatives) est limitée par la structure statique des modèles paramétriques.

3.1.1.2 Modèles de voisinage

Contrairement aux modèles paramétriques, les modèles de voisinage n'ont pas d'expression explicitement connue. L'approximation de la fonction objectif est obtenue principalement sur des critères géométriques ou de voisinage entre les individus de la base d'apprentissage.

K plus proches voisins L'algorithme des k plus proches voisins, ou *K-Nearest Neighbors* (KNN) [CH67] est une méthode d'apprentissage supervisé qui permet de classer des données en fonction de leur similarité avec les données d'entraînement. Le principe est de trouver les k individus de l'ensemble d'entraînement les plus proches (au sens d'une fonction de distance) de l'individu à classer, et de lui attribuer la classe majoritaire parmi ces k voisins. L'algorithme KNN peut être utilisé pour des problèmes de classification ou de régression, selon que la classe ou la valeur moyenne des k voisins est renvoyée.

L'algorithme KNN présente l'avantage d'être simple à implémenter et intuitif à comprendre. Cependant, il fait partie des modèles d'apprentissage dits "paresseux" (*Lazy Learning*) car il nécessite de stocker toutes les données d'entraînement en mémoire. Ces données sont traitées à chaque itération via des calculs de distances, ce qui peut être coûteux en termes de mémoire et de temps [Aha97].

Arbres de décision Les arbres de décision de classification/régression ou *Classification And Regression Trees* (CART) sont basés sur la notion d'arbre de décision, qui est une structure hiérarchique composée de nœuds et de branches [Bre84]. Chaque nœud représente un test sur une variable explicative, optimisé de façon a séparer les individus en deux groupes les plus distincts possibles (au sens d'un critère de variance), et chaque branche représente le résultat du test. Les feuilles de l'arbre sont les valeurs prédites par le modèle. Les CART peuvent être utilisés pour des problèmes de classification ou de régression [Bre84]. Dans le cas de la classification, les feuilles de l'arbre correspondent à des classes, et le critère d'optimisation est généralement l'indice de Gini ou l'entropie. Dans le cas de la régression, les feuilles de l'arbre correspondent à des valeurs numériques, et le critère d'optimisation est généralement l'erreur quadratique moyenne. Dans l'espace des variables d'entrée du modèle, la classification par arbre de décision construit un pavage à base d'hypercubes et constitue donc bien un modèle de voisinage.

Les CART présentent plusieurs avantages, tels que la facilité d'interprétation, la robustesse aux données manquantes ou aberrantes, la capacité à gérer des variables qualitatives ou quantitatives, et la possibilité de réaliser une sélection automatique des variables pertinentes [Gil+18]. Ils présentent aussi quelques inconvénients tels que la sensibilité au choix des paramètres (e.g. profondeur maximale, nombre minimal d'observations par nœud, etc.), le risque de sur-apprentissage (i.e. des changements mineurs dans les données d'apprentissage peuvent nécessiter des changements importants dans la structure de l'arbre) et la non-linéarité des frontières de décision [Kot13].

Intérêt Contrairement aux modèles paramétriques, les modèles de voisinage résolvent les problèmes de manière locale dans l'espace des variables explicatives. Ils peuvent présenter de bonnes performances sur des problèmes linéaires seulement localement, et fortement non linéaires globalement.

Les variables explicatives constituent un espace géométrique dans lequel les modèles de voisinage se construisent lors de l'apprentissage. Il est aisé de

visualiser les frontières de décision du modèle ainsi que leur sensibilité par rapport à chacune des variable explicative. Les modèles de voisinage présentent donc un fort potentiel d'explicabilité.

Les modèles de voisinage sont dépendants de l'ordre des individus dans la base d'entraînement, ce qui leur confère une capacité d'adaptation rapide à des changements de comportement à condition d'être compatibles avec l'apprentissage en ligne. Des algorithmes comme les arbres de Mondrian (basés sur les processus statistiques de Mondrian [RT+08]) ont pour but d'adapter les modèles de voisinage (en l'occurrence les CART) à l'apprentissage en ligne [LRT14].

3.1.2 Apprentissage non supervisé

Dans l'apprentissage non supervisé, les valeurs de la variable cible ne sont pas connues dans lors de l'entraînement. Les méthodes de classification non supervisée regroupent les individus en fonction des similarités de leurs variables explicatives.

La classification ascendante hiérarchique (CAH) [Nie16] est une méthode de construction de classes successives à partir des données d'entrée de manière hiérarchique grâce à des calculs de distances entre les individus. Un graphe appelé dendrogramme représente les clusters ainsi constitués et permet de choisir le nombre de classe qui convient le mieux à la résolution du problème considéré.

D'autres méthodes comme k-*means* [For65] sont des équivalents non supervisés de modèles de voisinage comme KNN et créent des *clusters* en allouant dynamiquement les individus à des centres de classes mobiles.

3.1.3 Apprentissage semi-supervisé

L'apprentissage semi-supervisé désigne les méthodes d'apprentissage sur des données dont l'oracle est connu seulement pour une partie d'entre eux [VEH20]. Dans le cas de la classification, les algorithmes de détection d'anomalie consituent un exemple d'apprentissage semi-supervisé. La base d'apprentissage contient uniquement des points considérés comme "normaux", et un classifieur mono-classe (*One Class*) est entraîné à détecter les points aberrants d'un point de vue statistique (*outliers*).

3.1.4 Apprentissage par renforcement

Comment apprendre à un modèle à prendre des décisions dans des problèmes complexes nécessitant de construire une stratégie sur du long terme ? L'apprentissage par renforcement est une méthode d'apprentissage automatique qui permet à un système autonome d'apprendre à optimiser une récompense quantitative en interagissant avec son environnement [Sze22]. Le système, appelé agent, choisit des actions en fonction de son état courant ou de l'état de l'environnement auquel il a accès et observe les conséquences de ses choix sur la récompense et l'état futur. L'apprentissage par renforcement se distingue des autres méthodes d'apprentissage automatique par le fait que l'agent n'a pas accès

à des exemples de comportement optimal (l'existence de celui-ci n'étant même pas garantie), mais doit découvrir par lui-même les actions les plus bénéfiques.

L'élément central de l'apprentissage par renforcement est la récompense. Pour chaque action possible de l'agent, l'environnement dans lequel il évolue lui renvoie une récompense (positive ou négative). L'objectif de l'agent est de trouver une politique, c'est-à-dire une règle de décision, qui maximise la récompense cumulée sur le long terme.

L'apprentissage par renforcement a connu un essor important ces dernières années, grâce aux progrès des techniques d'apprentissage profond et aux applications dans des domaines variés, tels que les jeux vidéo, la robotique, le contrôle industriel ou la finance. Parmi les exemples les plus célèbres, on peut citer le programme AlphaGo de Google DeepMind [Sil+16], qui a battu le champion du monde de go en 2016, ou le programme OpenAI Five [Ber+19], qui a affronté des joueurs professionnels du jeu vidéo Dota 2 en 2018.

Il existe plusieurs approches de l'apprentissage automatique selon le problème considéré. Le cas d'application de la détection du mode de transport est un problème de classification supervisée dans lequel on cherche à entraîner un modèle à partir de données d'entrée dont le mode de transport est connu, pour ensuite prédire le mode de transport sur de nouvelles données où il est inconnu. La section suivante présente des méthodes d'agrégation de modèles pour améliorer la performance de classification sur des problèmes non linéaires tels que la détection du mode de transport.

3.2 Agrégation de modèles

L'agrégation de modèle consiste à regrouper plusieurs instances d'un même modèle, voire de modèles différents, afin de combiner leurs performances et diminuer l'erreur de prédiction. Dans ma classification de méthodes d'agrégation de modèles, je distingue deux approches fondamentales :

— L'approche connexionniste, dans laquelle les modèles sont chaînés en série les uns par rapport au autres et sont entraînés sur un même jeu de données.

— L'approche ensembliste, dans laquelle les modèles sont entraînés en parallèle les uns des autres sur des jeux de données non identiques.

Il est important de noter qu'une méthode d'agrégation de classifieurs peut combiner connexionnisme et ensemblisme. Les classifieurs y sont vus comme les briques unitaires d'un classifieur plus grand, et ainsi de suite. De plus, une agrégation de modèles n'a pas nécessairement une structure connue à l'avance. Lorsque celle-ci dépend des données d'entrée, le modèle global est dit constructiviste [Ver16].

3.2.1 Connexionnisme

L'approche connexionniste peut être illustrée par un circuit électrique en série (c.f. figure 3.1). Les données d'entrée passent successivement par une chaîne de classifieurs et la fonction de décision du classifieur global est une composition des fonctions de décision des classifieurs internes. Dans l'apprentissage supervisé, l'information sur la sortie (oracle) remonte la chaîne de classificateurs dans la direction opposée.

L'approche connexionniste est associée aux réseaux neuronaux artificiels (ANN) dans lesquels les informations des données de sortie sont renvoyées par l'algorithme de rétropropagation du gradient [RHW86]. Les classificateurs internes qui composent le réseau neuronal sont linéaires. Un réseau est généralement composé de plusieurs couches (perceptron multicouche). La multiplication de ces couches est à l'origine de l'apprentissage profond.

FIGURE 3.1 – Schéma d'un modèle connexionniste dans lequel trois classifieurs $C1$, $C2$ et $C3$ sont chaînés.

La principale limitation de l'approche connexionniste est le très grand nombre de paramètres des modèles, nécessitant un grand nombre d'observations pour optimiser les poids associés à chaque classificateur interne [Bel+19]. Cela peut rendre difficile l'adaptation de tels modèles dans des problèmes dynamiques où le comportement des individus évolue dans le temps.

Intérêt Les modèles connexionnistes, et plus particulièrement les réseaux de neurones, sont reconnus pour leurs très bonnes performances sur des problèmes fortement non linéaires. Leur apprentissage nécessite un grand nombre d'individus dans la base d'entraînement. La capacité d'adaptation de ces algorithmes à des changements de comportement soudains des individus est donc limitée. De plus, les réseaux de neurones sont généralement incompatibles avec l'apprentissage en ligne.

Enfin, les réseaux de neurones profonds sont constitués de milliers, voire de millions de coefficients qu'il est impossible d'interpréter pour quantifier l'importance d'une variable ou d'un individu dans l'apprentissage. Ces modèles sont régulièrement qualifiés de "boîtes noires" [Gil+18] à cause de ce manque d'explicabilité et d'interprétabilité.

3.2.2 Ensemblisme

L'apprentissage ensembliste est une méthode d'apprentissage automatique qui combine plusieurs modèles d'apprentissage simples dits "faibles" [Don+20] pour obtenir une meilleure performance de prédiction. L'idée est de créer un

ensemble de modèles diversifiés et complémentaires qui peuvent se corriger mutuellement et réduire les erreurs individuelles. La structure d'un classifieur ensembliste peut-être assimilée à celle d'un circuit électrique en parallèle (c.f. figure 3.2). Il existe plusieurs techniques d'apprentissage ensembliste, telles que le bagging (*bootstrap aggregating*), le boosting, le stacking ou le voting [Don+20]. Ces techniques diffèrent par la façon dont elles génèrent, pondèrent et agrègent les modèles de base.

L'apprentissage ensembliste a été introduit par Leo Breiman dans les années 1990 avec le développement de l'algorithme de bagging [Bre96]. Le bagging consiste à créer des échantillons bootstrap (c'est-à-dire des échantillons aléatoires avec remise) à partir du jeu de données d'origine, et à entraîner un modèle de base sur chaque échantillon. Le modèle final est obtenu en moyennant les prédictions des modèles de base. Le bagging permet de réduire la variance et d'éviter le sur-apprentissage.

Le boosting est une autre technique d'apprentissage ensembliste, qui vise à améliorer la performance d'un modèle faible en lui ajoutant progressivement des modèles complémentaires. Le boosting ajuste les poids des observations en fonction des erreurs commises par le modèle précédent, de sorte que les modèles suivants se concentrent sur les cas difficiles. Le modèle final est obtenu en combinant les prédictions des modèles pondérées par leur précision. Le boosting permet de réduire le biais et d'augmenter la complexité du modèle.

Le stacking est une technique d'apprentissage ensembliste qui utilise un méta-modèle pour combiner les prédictions de plusieurs modèles de base. Le méta-modèle est entraîné sur un jeu de données constitué des prédictions des modèles de base comme variables explicatives, et de la variable cible comme variable à expliquer. Le stacking permet d'exploiter les forces et les faiblesses de chaque modèle de base, et d'obtenir un modèle plus robuste et généralisable.

Le voting est une technique d'apprentissage ensembliste qui consiste à faire voter les modèles de base pour la classe ou la valeur à prédire. Le voting peut être simple (chaque modèle a le même poids) ou pondéré (les modèles sont pondérés par leur précision ou leur confiance). Le voting permet de réduire l'impact des modèles aberrants ou biaisés, et d'obtenir un consensus entre les modèles.

L'apprentissage ensembliste est un domaine actif de recherche en apprentissage automatique, qui vise à développer des méthodes efficaces et adaptatives pour combiner plusieurs sources d'information et améliorer la qualité des prédictions.

Intérêt L'agrégation ensembliste de classifieurs permet d'obtenir une bonne performance sur des problèmes non linéaires, même lorsque les classifieurs internes sont des modèles simples [Don+20]. Les prédictions d'un modèle ensembliste sont difficiles à interpréter car les classifieurs internes ne sont pas nécessairement entraînés sur les mêmes données, voire sur les mêmes variables. Si l'algorithme *Random Forest* [Bre01] propose un score d'importance des variables explicatives, les frontières de décision ne sont pas visualisables et l'explicabilité reste faible.

La compatibilité des approches ensemblistes avec l'apprentissage en ligne, ainsi que leur capacité d'adaptation rapide à des changements de comportement

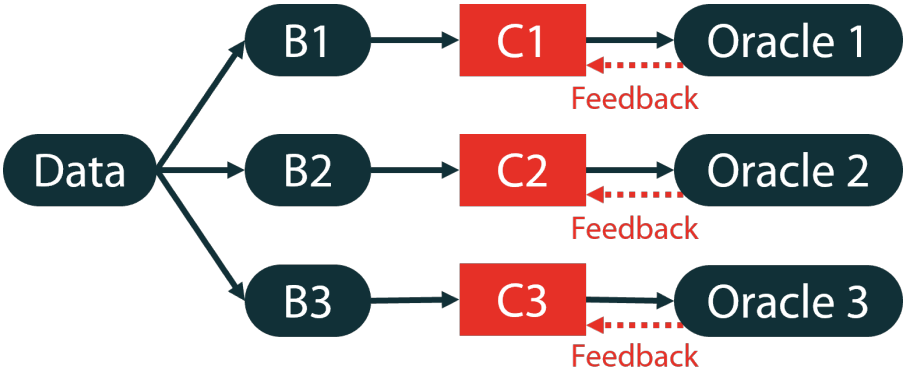


FIGURE 3.2 – Schéma d'un modèle ensembliste dans lequel trois échantillons bootstrap ($B1$, $B2$ et $B3$) sont constitués à partir des données d'entrée pour entraîner trois classifieurs $C1$, $C2$ et $C3$.

des individus, dépendent des classifieurs internes utilisés.

3.2.3 Constructivisme

Dans l'approche d'agrégation auto-constructive, le graphe des classifieurs internes (connexionniste ou ensembliste) n'a pas de structure fixe (c.f. figure 3.3). Cette structure est construite au cours du processus d'apprentissage et peut évoluer dans le temps [Cor19]. Lorsque les changements apportés au graphe des classificateurs sont effectués par les classificateurs eux-mêmes, le système s'auto-organise. L'approche constructiviste peut être vue comme la version "non paramétrique" de l'agrégation de classifieurs.

Les algorithmes tels que les réseaux neuronaux auto-constructifs [CN+09] se situent à l'intersection du connexionnisme et du constructivisme, car la structure des couches de neurones est construite de manière incrémentale. D'autres approches telles que l'apprentissage par schéma [Hol+04] possèdent des architectures construites par l'interaction avec l'environnement (i.e. les données d'entrée). L'approche constructiviste permet de concevoir des classifieurs non linéaires auto-organisés adaptés à la résolution de problèmes dynamiques.

FIGURE 3.3 – Schéma d'un modèle constructiviste

La section suivante introduit les systèmes multi-agents, et plus particulièrement la théorie de l'apprentissage par contexte sur laquelle se base la solution algorithmique développée dans le cadre de cette thèse.

3.3 Systèmes multi-agents

Les systèmes multi-agents (SMA) sont une approche de modélisation et de résolution de problèmes complexes en informatique et en intelligence artificielle. Inspirés par les interactions observées dans les sociétés naturelles, les SMA regroupent plusieurs agents autonomes, dotés de capacités de perception, de raisonnement et d'action individuelles. Ces agents interagissent dynamiquement au sein d'un environnement partagé, souvent pour atteindre des objectifs communs ou pour résoudre des problèmes complexes qui dépassent les capacités individuelles de chaque agent. Les SMA sont utilisés dans divers domaines tels que la robotique, la gestion de systèmes distribués, la simulation sociale ou la logistique.

3.3.1 Systèmes multi-agents adaptatifs

La théorie des systèmes multi-agents adaptatifs (AMAS) [Cap+03] propose une approche coopérative des interactions entre agents. Les critères de conception présentés pour ces interactions mènent à un résultat satisfaisant, mais pas nécessairement optimal, dans la résolution du problème posé (théorème d'adéquation fonctionnelle).

3.3.2 Apprentissage par contexte

L'apprentissage par contexte consiste à explorer l'espace défini par les variables d'entrée du modèle à l'aide d'agents en coopération. L'approche *AMAS for Context Learning* (AMAS4CL) se base sur la théorie des AMAS et plus particulièrement sur le paradigme *Self-Adaptative Context Learning* (SACL) [Boe+15] pour définir les règles de coopération entre agents et propose une structure composée de plusieurs types d'agents pour explorer l'espace des variables du problème.
Les algorithmes basés sur l'approche SACL sont utilisés pour résoudre des problèmes variés tels que l'apprentissage par démonstration en robotique [Ver+15] ou l'optimisation du fonctionnement d'un moteur de voiture thermique [Boe+15].

Les architectures SACL sont généralement composées de trois types d'agents :

— Les agents Contexte : ce sont les briques unitaires du modèle. Ils définissent des hypercubes de l'espace des variables d'entrée. Lorsqu'un nouveau point appartient à une de ces zones, l'agent Contexte correspondant est dit activé et propose une décision du système selon ses propres connaissances.

— Les agents Percept : ils font office d'interface entre les données d'entrée et le système. Ils récupèrent les valeurs des variables d'entrées (capteurs) à chaque itération et les transmettent aux agents Contexte.

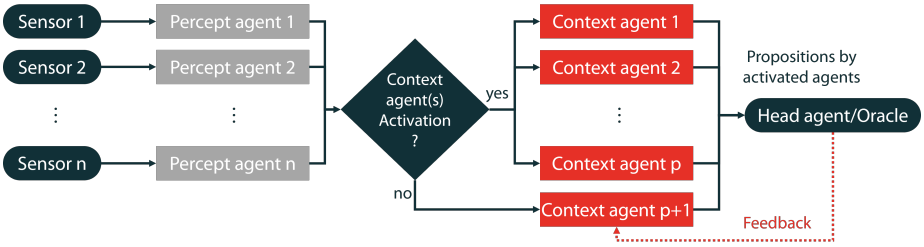


FIGURE 3.4 – Schéma d'un SMA d'apprentissage par contexte

— L'agent Head : c'est le superviseur du système. Il reçoit les propositions des agents Contexte activés, décide de la prochaine action du système et leur renvoie un feedback.

Un SMA d'apprentissage par contexte, comme tout modèle d'apprentissage, possède deux modes de fonctionnement : l'exploration (ou apprentissage) qui consiste à instancier et agencer les agents Contexte grâce aux feedback de l'agent Head, et l'exploitation (ou prédiction) qui consiste à prendre une décision sans mettre à jour le système. Le fonctionnement de cette architecture est présenté dans la figure 3.4 pour le cas coopératif en phase d'exploration (i.e. le cas de fonctionnement optimal). Lorsque le comportement du système n'est pas optimal au regard des objectifs de l'utilisateur, on dit que la situation est non-coopérative (SNC). Le système doit alors s'adapter pour maximiser la coopération entre agents pour revenir dans le cas coopératif. En apprentissage par contexte, cette coopération se traduit par les tailles (i.e. les dimensions des hypercubes), positions et connaissances des agents Contexte. Les SNC définies dans la contribution ainsi que leurs résolutions sont détaillées dans le chapitre 5.
L'architecture ELLSA [Dat21] s'appuie sur SACL et introduit des règles de coopération pour explorer l'espace des variables d'entrée. Les agents Contexte implémentent des régressions linéaires permettant d'approximer la fonction sous-jacente du problème à résoudre, supposée localement linéaire sur leurs zones d'activation. Ces modèles internes de régression linéaire permettent également de résoudre des problèmes de classification supervisée dans lesquels chaque agent Contexte est associé à une classe.

Les architectures SACL sont des modèles de voisinage car l'apprentissage consiste à construire un pavage efficace de l'espace des variables d'entrée afin de résoudre localement le problème considéré. De plus, les feedbacks peuvent être vus comme des récompenses au sens de l'apprentissage par renforcement dans lequel chaque agent Contexte a pour objectif de maximiser sa coopération avec les autres agents Contexte. Enfin, le nombre d'agents Contexte, leurs positions et leurs taille relatives ne sont pas connues *a priori*. Les modèles de SACL se placent donc dans le paradigme du constructivisme, et peut également être vue comme une approche ensembliste. Dans ELLSA [Dat21], les agents Contexte peuvent être considérés comme des classifieurs linéaires (bien qu'ils ne puissent prédire qu'une seule classe) avec des règles coopératives.

L'apprentissage par contexte constitue la dernière brique sur laquelle se base la solution algorithmique proposée dans le cadre de cette thèse. La section suivante détaille le positionnement de la solution par rapport aux différents concepts présentés dans ce chapitre.

3.4 Synthèse

Au sens mathématique du terme, un modèle d'apprentissage automatique est une fonction associant un label (valeur numérique dans le cas de la régression, catégorie dans le cas de la classification) à un vecteur de l'espace des variables explicatives du problème considéré. Cette fonction a pour but d'approcher la fonction objectif, c'est-à-dire la véritable relation entre variables d'entrée et labels.

Les modèles paramétriques, caractérisés par une expression analytique connue, permettent une grande spécialisation (et donc potentiellement une grande performance) sur des modèles très divers (y compris non linéaires), à condition d'en avoir au préalable étudié la structure. En revanche, dans le cas où cette structure évolue dans le temps, les modèles paramétriques présentent une faible capacité d'adaptation en raison de leur rigidité. L'entraînement des modèles paramétriques consiste à ajuster les valeurs des coefficients associés à des groupes de variables explicatives. Les modèles paramétriques sont donc compatibles avec l'apprentissage en ligne, et très fortement explicables en termes d'importances de variables d'entrée ou de sensibilité.

A l'inverse, les modèles de voisinage n'ont pas d'expression analytique connue. L'ordre des individus lors de l'entraînement influence grandement la construction des frontières de décisions dans l'espace des variables explicatives ou dans un espace dérivé. Cette propriété augmente la capacité et la vitesse d'adaptation de ces modèles en cas de changement de comportement des utilisateurs. De plus, les frontières de décision sont immédiatement disponibles, ce qui confère aux modèles de voisinage une grande explicabilité géométrique. L'approche SACL est un exemple de modèle de voisinage car les agents Contexte sont instanciés et évoluent dans l'espace des variables d'entrée, sans qu'une expression mathématique globale soit connue.

De plus, les modèles SACL se rapprochent de l'apprentissage par renforcement car ils sont composés d'agents cherchant à maximiser leur coopération grâce à un système de récompenses orchestré par l'agent Head.

Les modèles d'apprentissage automatique peuvent être vus comme les briques unitaires de modèles d'agrégation, plus complexes et potentiellement plus performants dans le cas de problèmes fortement non linéaires. Les deux principaux paradigmes d'agrégation sont le connexionisme et l'ensemblisme. Dans les deux cas, la compatibilité avec l'apprentissage en ligne et la rapidité d'adaptation sont conditionnées par les types de modèle interne utilisés. La complexification de ces modèles se traduit généralement par une faible explicabilité.

Enfin, la méthode d'agrégation de classifieurs ou régresseurs, pouvant être une combinaison d'approches connexionistes et ensemblistes, n'est pas nécessairement connue *a priori*. Dans le paradigme du constructivisme, l'organisation interne des modèles unitaires se construit lors de l'entraînement en fonction des propriétés des individus.

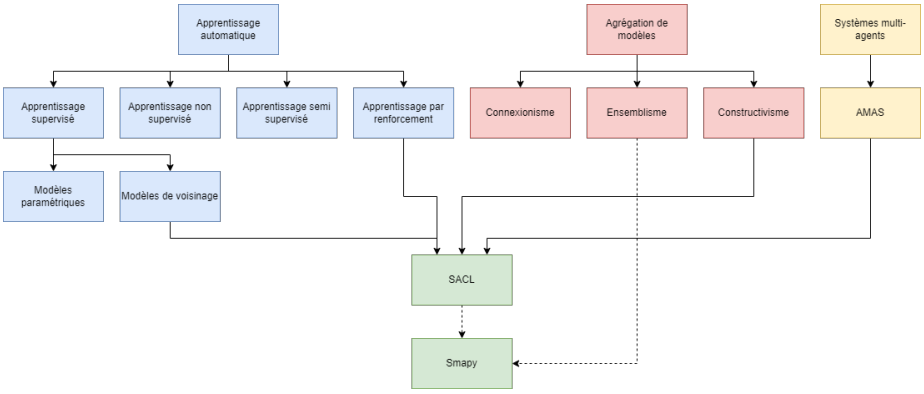


FIGURE 3.5 – Hiérarchie des concepts à l'origine du paradigme SACL et de Smapy, le système multi-agent coopératif et ensembliste développé dans le cadre du projet de thèse.

Ce chapitre a permis d'introduire les différents concepts d'apprentissage automatique et d'agrégation de modèles nécessaire à la construction de la solution algorithmique proposée dans le cadre de la thèse. Cette solution s'inscrit dans la continuité de la théorie des AMAS et de l'apprentissage par contexte. Dans le chapitre 5, nous proposons d'adapter l'approche SACL, un modèle de voisinage, que nous voyons également comme une agrégation constructiviste de modèles paramétriques simples, à des problèmes de classification supervisée (c.f. figure 3.5). **En particulier, nous choisissons de bénéficier des propriétés de l'apprentissage ensembliste en intégrant à l'intérieur des agents Contexte des classifieurs arbitraires, compatibles avec l'apprentissage en ligne. Le modèle résultant, appelé Smapy, se place donc à l'intersection entre système multi-agent coopératif, modèle de voisinage, ensemblisme et constructivisme.**

Chapitre 4

Comment résoudre le cas d'application de détection du mode de transport ?

Dans le chapitre 1, le cas d'application de la détection du mode de transport (TMD), qui est un problème de classification supervisée, est proposé pour illustrer le potentiel d'un modèle répondant aux pré-requis énoncés dans le même chapitre pour l'analyse de la mobilité. Les données de mobilité disponibles sont présentées et commentées dans le chapitre 2. Deux types de données de mobilité peuvent être utilisés dans ce but :

— Les données utilisateur qui sont des séries temporelles d'observations d'un utilisateur (e.g. GNSS, accéléromètre).

— Les données contextuelles qui sont des informations additionnelles sur le contexte spatial (données SIG), temporel (e.g. tableaux horaires des transports en commun, heures d'ouverture de lieux publics) ou spatio-temporel (e.g. accès en temps réel aux positions des bus ou taxis).

Ce chapitre présente l'état d'avancement de la recherche dans la résolution du problème de TMD. Dans la section 4.1, les caractéristiques du problème, notamment les données d'entrée et de sortie, sont présentées. Les section qui suivent présentent le processus de résolution dans l'ordre. La phase de pré-traitement est détaillée dans la section 4.2, puis le traitement de données, qui se traduit dans la majorité des cas par l'apprentissage d'un modèle de *Machine Learning*, est présenté dans la section 4.3. Des techniques de post-traitement visant à corriger les prédictions des modèles utilisés sont introduites dans la section 4.4. Dans un dernier temps, la résolution du problème de TMD est synthétisée dans la section 4.5.

4.1 Caractérisation du problème

Cette section présente les caractéristiques du problème de détection du mode de transport (TMD) tel qu'il est généralement abordé dans la littérature. Il s'agit d'un problème de classification supervisée dans lequel des données utilisateur (c.f. section 2.3) sont utilisées en entrée d'un classifieur pour déterminer un mode de transport en sortie, c'est-à-dire une classe.

Nous nous plaçons dans le contexte de l'analyse des traces des utilisateurs comme source primaire de données. Parmi les différentes terminologies utilisées pour la notion de trace, j'ai fait le choix de me baser sur celle introduite par [BLVO13] (c.f. chapitre 1) :

Definition 4.1.1 (Trace d'un utilisateur)**.** La trace d'un utilisateur est une série temporelle de points d'observations (potentiellement multi-dimensionnels et de dimensions différentes) mesurés sur un seul utilisateur. Elle est composée d'**arrêts** et de **trajets**.

Definition 4.1.2 (Trajet)**.** Un trajet est une période durant laquelle un utilisateur est en déplacement. Il est délimité par une origine et une destination qui sont des lieux d'intérêts pour l'utilisateur (dans la mesure où ils justifient son déplacement). Un trajet peut être multi-modal et est composé de *segments* mono-modaux.

Definition 4.1.3 (Segment)**.** Un segment est une période durant laquelle un utilisateur se déplace avec un seul mode de transport. Il est délimité par un point de départ et un point d'arrivée. Ces deux points peuvent être des points d'intérêt pour l'utilisateur ou des correspondances entre deux modes de transport différents.

4.1.1 Données d'entrée

Les deux principaux types de données utilisateur pour la résolution de ce problème sont les données de géolocalisation (c.f. section 2.3.1) et les données provenant d'une centrale inertielle (c.f. section 2.3.2) ou d'autres capteurs typiquement présents dans les smartphones (c.f. section 2.3.3). D'autres types de données utilisateur peuvent être utilisés, et la prédiction peut être améliorée par l'ajout de données contextuelles (c.f. section 2.1) en entrée du classifieur.

4.1.1.1 Données de géolocalisation

Sources de données Les données de géolocalisation sont très utilisées dans la littérature pour classifier les modes de transport. La plupart des données de géolocalisation sont issues d'un système GNSS (c.f. section 2.3.1.1) et sont collectées à travers une balise embarquée ([Bol+12], [Gon+12], [FT13], [DS17]), le réseau cellulaire ([CXC+22]) ou d'une application de smartphone (TRAC-IT [Gon+10], Routecoach [Sem+17], TransGPS [Gao+20], TravelVU [NA20]). La plupart des applications utilisées sont développées *ad hoc* pour les besoins du projet de recherche ([Ste+11], [WGJ17]). Il existe également des jeux de données publics de références utilisés pour comparer les différentes méthodes entre elles. En particulier, le dataset Microsoft GeoLife [Zhe+09] est abondamment utilisé pour évaluer des méthodes de classification du mode de transport ([Naw+20a], [Li+20], [Naw+20b]).

Attributs La vitesse et l'accélération font partie des attributs les plus régulièrement utilisés pour classifier le mode de transport à partir de données de géolocalisation [Yan+18]. Elles sont généralement estimées par rapport au points de données précédents. Le relèvement est également utilisé car il est lié à la sinuosité caractéristique de certains modes de transports comme le vélo ou la marche. D'autres attributs comme l'à-coup (ou *jerk*), qui est la dérivée de l'accélération par rapport au temps ([Naw+20a], [Li+20]), la longueur du

segment ([Gon+10], [NA20]) ou encore l'heure et le jour de la semaine ([Naw+20a], [Gir+22]) peuvent être utilisés pour inférer le mode de transport. Dans le cas où les données de géolocalisation proviennent du réseau cellulaire, le nombre d'antennes relais impliquées dans la mesure des coordonnées peut constituer un nouvel attribut [CXC+22].

Les coordonnées elles-mêmes peuvent également servir à classifier le mode de transport lorsqu'elles sont mises dans le contexte spatial de la zone d'étude, c'est-à-dire lorsque les données de géolocalisation sont combinées à des données contextuelles. De précédentes recherches ont montré que l'intégration de données SIG telles que les tronçons routiers ([Gon+12], [Ste+11], [BLVO13], [Ras+15], [Sem+17]) ou les points d'intérêt (POI) ([SN+16]) améliorent la classification du mode de transport. Les données issues des opérateurs de transport en commun (TC) telles que les arrêts ([Ste+11], [SN+16], [Sem+17], [NA20]), le tracé des lignes ([Ras+15], [XCZ19], [CXC+22]) voire les positions des bus en temps réel ([Ste+11]) permettent également d'affiner la classification des modes TC.

Par ailleurs, plusieurs auteurs ont tenté d'exploiter les fréquences des instants d'observation. Dans [BLVO13], le temps écoulé depuis le dernier arrêt est utilisé comme variable du modèle. Dans [FT13], la variable STEPS, qui est le temps moyen d'immobilité par minute, est introduite. Dans [Fou+23], nous utilisons les écarts temporels et spatiaux entre deux points consécutifs comme attributs, et nous montrons qu'ils font partie des attributs les plus discriminants pour la détection du mode de transport.

Critique Les différentes méthodes de collecte de données de géolocalisation présentent toutes des avantages et inconvénients. Les GNSS nécessitent une grande consommation d'énergie de l'appareil de mesure [Che+17], mais garantissent une précision satisfaisante dans le monde entier (sauf en milieu souterrain). Les méthodes basées sur les réseaux WiFi et Bluetooth consomment beaucoup moins d'énergie et sont plus précises en milieu urbain (au point de permettre une géolocalisation en intérieur [Shu+15]), mais sont très limitées dans les zones rurales. La géolocalisation par réseau cellulaire a une précision plus faible, mais les opérateurs de téléphonie mobile possèdent des parts de marchés importantes leur permettant d'effectuer des mesures représentatives à l'échelle de la population totale (e.g. En France en 2023, seuls quatre opérateurs se partagent l'ensemble du marché des télécommunications avec des parts de marché allant de 15% à 40% [Doz23]). Les données de géolocalisation peuvent être combinées à des données contextuelles pour améliorer les performances des classifieurs.

4.1.1.2 Données de smartphones

Sources de données La généralisation de centrales inertielles et d'autres types de capteurs dans les smartphones modernes permet de caractériser les mouvements d'un utilisateur (*Human Activity Recognition* ou HAR). Le problème de classification du mode de transport peut-être vu comme une extension du HAR. La majorité des études sont basées sur des applications de collecte *ad hoc*

dans les smartphones des utilisateurs ([SH16], [Guo+22]), mais il existe également des jeux de données de référence dans la littérature (US-Transportation Mode [Car+18], [Alo20], HTC [IG20]).

Attributs Les attributs régulièrement utilisés dans la littérature pour caractériser le mode à partir de données smartphone sont l'accélération selon les trois axes provenant des accéléromètres, la vitesse de rotation selon les trois axes provenant des gyroscopes, l'intensité du champ magnétique dans les trois directions de l'espace provenant des magnétomètres et la pression atmosphérique provenant du baromètre ([San+14]) liée à l'altitude de l'utilisateur. D'autres attributs tels que l'intensité sonore provenant du microphone ([Alo20]) sont plus rarement utilisés pour la classification.

Critique Les capteurs mentionnés dans cette section se généralisent à tous les smartphones modernes et consomment moins d'énergie qu'un GNSS. De plus, les attributs relatifs au mouvements de l'utilisateur permettent de discriminer des modes ayant un profil de vitesse similaire. Cependant, le contexte spatial est perdu car on ne connaît pas les localisations successives de l'utilisateur, ce qui limite l'intégration de données contextuelles dans la classification (c.f. section 2.1).

4.1.1.3 Données hybrides

La compatibilité des smartphones avec les GNSS permet de collecter conjointement des données de géolocalisation et des mesures de capteurs intégrés (accéléromètres, gyroscopes, magnétomètres, etc.). De nombreuses approches se basent sur ces deux sources de données pour classifier le mode de transport. Il existe peu de jeux de données publics combinant géolocalisation et mesures de capteurs (SHL Transportation Dataset [Gjo+18], OCC-TMD [Fou+23]), et les études existantes s'appuient sur des applications smartphone *ad hoc* pour collecter simultanément les positions et mesures liées à un utilisateur ([JR14], [ZYS16], [SB+17]). La précision de classification obtenue est supérieure à celles obtenues par les deux sources de données prises séparément ([Fou+23]).

4.1.1.4 Autres types de données utilisateur

Plus rarement, d'autres types de données utilisateur ne provenant ni d'un GNSS ni d'un smartphone sont utilisés. Dans [WGJ17], des données socio-économiques sur les utilisateurs, comme le nombre de voitures et vélos dans le foyer ou la possession d'un abonnement de transport en commun, sont utilisées en plus de leurs positions GPS pour inférer le mode de transport. Ces informations augmentent significativement la précision de classification (notamment pour le mode vélo électrique) et réduisent la confusion entre les modes bus et voiture. Cependant, il est souvent inenvisageable d'avoir à disposition des informations aussi intrusives sur les utilisateurs qui sont généralement anonymisés dans les données disponibles.

Dans [Gir+22], les utilisateurs sont équipés d'une série de capteurs GNSS et inertiels ainsi que de patchs mesurant leur fréquence cardiaque. Il apparaît que l'ajout de cet attribut augmente la précision de classification uniquement pour le mode vélo, mais cette étude pose les bases d'une utilisation potentielle des données médicales collectées par les montres et bracelets connectés pour inférer le mode de transport utilisé.

4.1.2 Données de sortie

Le problème de détection du mode de transport (TMD) peut être vu comme une extension du problème de reconnaissance de l'activité humaine (HAR). L'objectif est de fournir en sortie une classe correspondant au mode de transport utilisé. En HAR, les classes considérées sont généralement l'immobilité (*still*), la marche (*walk*) et le déplacement en véhicule (*vehicle*). D'autres classes plus fines comme la course (*run*) ou les différentes positions (assise, debout, allongée) sont parfois considérées en HAR, mais beaucoup plus rarement en TMD.

La complexification des méthodes de résolution ont permis de séparer la classe *vehicle* en classe vélo (*bike*) et véhicule motorisé (*motorized*), voire même à faire la distinction entre vélo et vélo électrique, et entre transports en commun (TC) et véhicules privés (VP). Certaines approches compilées dans le tableau 4.1 proposent de distinguer tout ou partie des différents modes TC (métro, bus, tram, train voire avion) et VP (voiture, moto) comme le montre la figure 4.1. Le choix des classes dépend du territoire considéré et du contexte de l'étude. Plus les classes sont affinées, plus le problème de classification est complexe à résoudre.

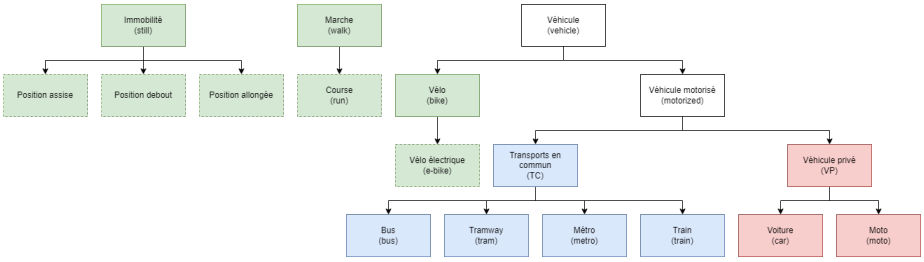


FIGURE 4.1 – Hiérarchie des classes utilisées en détection du mode de transport. Les modes actifs sont représentés en vert, les modes de transports en commun en bleu et les véhicules privés motorisés en rouge. La différenciation des positions d'immobilité, la course et le vélo électrique sont rarement considérés dans la littérature.

Les approches de résolution du problème de TMD diffèrent par la nature des données d'entrée utilisées, mais également par la typologie de modes considéré en sortie. Dans la prochaine section, les étapes et méthodes de pré-traitement,

c'est-à-dire tout ce qui sépare les données brutes de la phase d'apprentissage du classifieur, sont introduites.

4.2 Pré-traitement

Cette section présente les deux principales étapes de pré-traitement d'un modèle de classification du mode de transport. La première étape consiste à segmenter les trajets et arrêts et est spécifique aux données de mobilité. La seconde étape est le calcul de *features*. Il est généralement effectué avant tout entraînement d'un modèle d'apprentissage automatique.

4.2.1 Segmentation

La trace d'un utilisateur est une série temporelle de mesures successives et a donc une cohérence temporelle. Une trace est constituée de trajets, pouvant être séparés en segments (i.e. groupes de points successifs correspondant à des déplacements de l'utilisateur selon un seul mode), et d'arrêts (i.e. groupes de points successifs qui correspondent à des périodes d'immobilité de l'utilisateur). Il existe alors deux manières d'aborder le problème de classification du mode de transport :

— En considérant l'immobilité (*still*) comme une classe à part entière et en entraînant le modèle à la prédire en même temps que les autres modes pour chaque point de donnée.
— En procédant d'abord à une **segmentation** des trajets grâce à un algorithme de détection des arrêts, puis classifier le mode de chaque segment.

4.2.1.1 Méthodes sans détection des arrêts

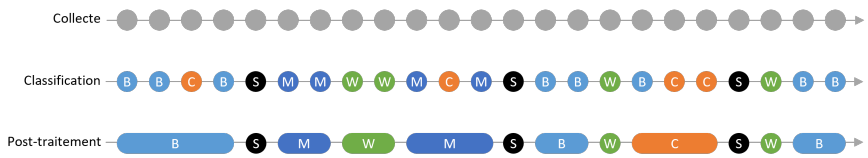


FIGURE 4.2 – Schéma de résolution du problème de TMD sans segmentation des trajets. Les couleurs et lettres représentent des modes différents (S pour *still*, W pour *walk*, M pour *metro*, B pour *bus*, C pour *car*).

La méthode de résolution directe du problème de TMD consiste à classifier le mode pour chaque point de donnée, en considérant la classe **still**. Cela nécessite donc que les points d'arrêts soient inclus et labélisés dans la base d'apprentissage. Cette méthode a l'avantage d'être simple à mettre en oeuvre, mais peut conduire à des résultats bruités. Par exemple, des attributs telles que

la vitesse ou l'accélération d'un utilisateur peuvent fortement varier d'un point à l'autre sans que le mode n'ait changé (e.g. arrêt à un feu rouge, embouteillages, etc.), mais conduire à des prédictions différentes en sortie du classifieur. La figure 4.2 illustre ce bruit dans le résultat de la classification sans segmentation.

Afin d'atténuer ce bruit, on évite généralement de passer directement les attributs mesurés pour chaque point en entrée du classifieur, au profit d'indicateurs statistiques (*features*) calculés sur un groupe de points à l'aide d'une fenêtre temporelle statique ou glissante [Yan+18]. Ce procédé permet de lisser les valeurs des attributs afin d'éviter des signaux trop bruités.

4.2.1.2 Méthodes avec détection des arrêts

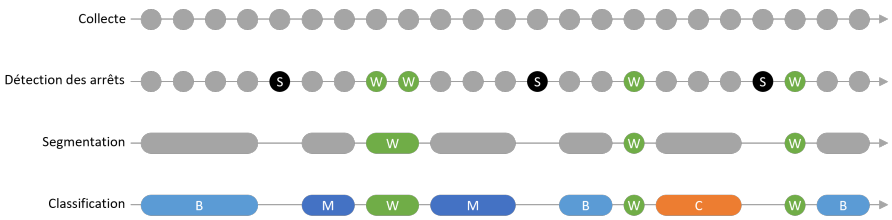


FIGURE 4.3 – Schéma de résolution du problème de TMD avec segmentation des trajets basée sur la détection des arrêts (et points de marche). Les couleurs et lettres représentent des modes différents (S pour *still*, W pour *walk*, M pour *metro*, B pour *bus*, C pour *car*).

Il est possible d'éviter le bruit résultant de la classification point par point et la détection de faux arrêts en procédant au préalable à une segmentation des trajets. Comme illustré par la figure 4.3, la première étape est de détecter les arrêts, généralement avec des règles logiques sur les attributs, basées sur des seuils de temps et de distance ([Gon+12], [WNB12], [BLVO13], [Ras+15], [SN+16], [SB+17], [Xia+17], [DS17], [Li+20], [Gao+20], [Naw+20b]). La détection ou l'annotation des arrêts peut également être réalisée directement dans l'application smartphone utilisée pour la collecte ([Gon+10], [Sem+17], [WGJ17], [XCZ19], [NA20], [CXC+22]). La classification du mode est ensuite réalisée sur les segments délimités par les arrêts, que l'on suppose être mono-modaux.

De nombreuses approches proposent d'aller plus loin et de détecter également les points correspondant à la marche (*walk*) en faisant l'hypothèse que tout segment mono-modal commence et se termine par un arrêt ou un trajet de marche à pied [Naw+20b]. Ainsi, plusieurs segments différents peuvent être identifiés entre deux arrêts s'ils sont délimités par des segments à pied. Cette méthode a l'avantage de supprimer le bruit induit par la résolution directe. De plus, dans le cas de données de géolocalisation, les emplacements des arrêts peuvent être combinés à des données contextuelles pour affiner la prédiction du mode, voire de classifier le motif de l'arrêt (i.e. répondre à l'axe "Pourquoi" présenté dans le chapitre 1). En revanche, la mise en oeuvre de cette méthode nécessite un algorithme de détection des arrêts robuste et adapté aux données utilisées. En

raison de méthodes de collecte différentes (e.g. fréquence d'acquisition, critères de collecte basés sur la distance parcourue ou le temps écoulé, etc.), ces algorithmes sont très difficilement transposables d'une source de données à une autre.

4.2.2 Calcul de features

Nous avons vu dans la section précédente que les valeurs d'attributs des points de données ne sont pas utilisées telles quelles en entrée du classifieur. Afin de lisser les données et d'exploiter les propriétés statistiques des groupes de points adjacents dans le temps, les points de données sont regroupés en segments à l'aide d'une fenêtre temporelle ou d'un algorithme de segmentation basé sur une détection des arrêts. De nouvelles variables appelées *features* sont calculées pour chaque segment et servent de variables d'entrée pour le classifieur utilisé.

4.2.2.1 Indicateurs statistiques

Les *features* sont généralement des indicateurs statistiques sur les attributs, tels que la moyenne, la médiane, le minimum, le maximum, l'écart-type, les quantiles (notamment le quantile à 95%), le coefficient d'asymétrie (*skewness*) ou le coefficient d'aplatissement (*kurtosis*). L'idée derrière le calcul d'indicateurs statistiques est que le mode ne dépend pas uniquement des valeurs instantanées des attributs mesurés sur un utilisateur, mais de leur variation dans le temps. Par exemple, des modes tels que la voiture ou le vélo sont caractérisés par une grande variabilité de la vitesse ou de l'accélération par rapport au train.

D'autres indicateurs tels que les coefficients de Fourier de variables quantitatives, obtenus numériquement par transformée de Fourier rapide, peuvent également être calculés sur les fenêtres temporelles et utilisés comme variables d'entrées pour la classification du mode de transport ([ZYS16])

4.2.2.2 Features profondes

Certaines approches basées sur des modèles d'apprentissage profond proposent de calculer des *features* profondes (*deep features*), c'est-à-dire des indicateurs n'ayant pas de signification statistique, issus d'une couche d'un réseau de neurones artificiel. Ces indicateurs n'ont pas de formule explicite car leur calcul dépend de la structure du réseau, et donc de l'apprentissage. Ils sont toutefois supposés être discriminants pour les classes considérées. Dans [Naw+20a], le modèle ConvLSTM, un réseau de neurones convolutionnel et récurrent, est utilisé pour extraire des *features* à partir de deux couches internes. La première renvoie des *features* dérivées d'indicateurs statistiques calculés sur les attributs. La seconde couche renvoie des *features* dérivées d'attributs "auxiliaires" sur lesquels il est impossible de calculer des indicateurs statistiques (e.g. l'heure, le jour de la semaine, etc.).

4.2.2.3 Sélection de features

Le calcul de plusieurs indicateurs statistiques sur chaque attribut donne un grand nombre de *features*, c'est-à-dire une grande dimension du problème de classification. Pour simplifier le problème et éviter les *features* redondantes, certaines approches de la littérature proposent de réduire cette dimension en sélectionnant les *features* les plus discriminantes selon les classes considérées. Plusieurs méthodes de sélection de *features* sont utilisées en TMD, comme l'Analyse en Composantes Principales (ACP) ([IG20]), l'analyse de la variance (ANOVA) ([Bol+12]), le test statistique du χ^2 ([Ste+11]) ou des algorithmes de sélection automatique ([Red+10]). Les importances relatives des variables d'entrées calculées avec le critère de Gini à l'issue de l'algorithme *Random Forest* (RF) peuvent être utilisées pour sélectionner les *features* les plus discriminantes ([ZYS16], [Xia+17], [IG20]). De plus, les méthodes de calcul de *features* profondes peuvent également être vues comme un moyen de contrôler le nombre de dimensions du problème et donc de sélectionner les *features* les plus discriminantes.

Dans la résolution du problème de TMD, le pré-traitement est constitué d'une étape de segmentation visant à regrouper les points de données en segments temporels, et d'une phase de calcul de *features* qui servent de variables d'entrée du classifieur. La prochaine section présente la phase d'entraînement du classifieur, et fait un état des différents modèles utilisés dans la littérature.

4.3 Traitement

Il existe dans la littérature une importante diversité de classifieurs utilisés pour résoudre le problème de détection du mode de transport. Une première approche consiste à définir un algorithme basé sur des règles issues du domaine de l'analyse de la mobilité ([SN+16]). Dans [Gon+12], ces règles sont basées sur les mesures et sur la prise en compte de données SIG. Dans [BLVO13], les auteurs utilisent un système expert flou comme classifieur.

De nombreuses références utilisent des techniques d'apprentissage automatique (*Machine Learning*). Le but est de rechercher par apprentissage sur des données $(X_i, y_i)_{i\in[\![1,n]\!]}$ une estimation $\hat{f}$ de la fonction objectif théorique f telle que pour toute observation X_i et son mode de transport y_i associé, la relation $y_i = f(X_i)$ est vérifiée. Une fois la fonction $\hat{f}$ apprise par l'algorithme de Machine Learning, une estimation $\hat{y}_i = \hat{f}(X_i)$ des modes de transport y_i de nouvelles observations X_i peuvent être obtenues.

Parmi les algorithmes les plus utilisés dans ce cadre d'étude, on peut lister la régression logistique (logit) ([DS17]), les arbres de décision (CART) ([Ste+11], [Red+10], [Zhe+08] et [Alo20]), les classifieurs bayésiens naïfs ([Ste+11] et [Nic+10]), les réseaux bayésiens ([Ste+11] et [Zhe+08]), les machines à vecteurs de support (SVM) ([JR14] et [Nic+10]) et les modèles de Markov cachés ([WNB12] et [Red+10]). Des approches ensemblistes sont également étudiées, parmi lesquelles le bagging ([Alo20], les forêts aléatoires (RF) ([Ste+11], [Alo20]

et [SH16]) et l'algorithme Gradient Boosting ([Alo20]). Des réseaux de neurones artificiels (ANN) denses ([Ste+11] et [Gon+10]) et récurrents (LSTM) ([IG20]) sont régulièrement utilisés dans la littérature.

De nombreux classifieurs sont utilisés dans la littérature pour résoudre le problème de TMD. La prochaine section présente une méthode de post-traitement permettant d'améliorer la précision de classification notamment dans le cas où aucune détection des arrêts n'a été réalisée durant l'étape de segmentation.

4.4 Post-traitement

Dans le cas où la segmentation n'est pas basée sur une détection des arrêts, les prédictions en sortie peuvent être bruitées (c.f. section 4.2.1.1). Il est possible de réduire ce bruit pour améliorer la qualité de prédiction en mettant en place un post-traitement des données de sortie, comme illustré sur la dernière ligne de la figure 4.2. Ce post-traitement consiste à délimiter des segments à partir des points stationnaires (*still*) ou de marche (*walk*) détectés, pour ensuite réaliser un vote majoritaire des modes prédits. Les vainqueurs des votes majoritaires sont ensuite choisis comme prédictions finales du modèle. Dans [Guv+17], cette méthode est implémentée sous le nom d'algorithme de *Healing*. L'utilisation du post-traitement permet d'améliorer le score *recall* de 7 points.

Dans [Gir+22], un algorithme similaire appelé "homogénéisation" est utilisé, mais les segments de prédiction ne sont plus définis en fonction des points stationnaires ou de marche détectés. Les auteurs utilisent une fenêtre temporelle glissante pour réaliser le vote majoritaire et lisser les prédictions. Plusieurs tailles de fenêtre ont été testées, et les plus grandes améliorations du score de précision sont obtenues pour des fenêtres de 4 à 5 minutes (près de 11 points de plus sur le score de précision). Les modes stationnaires (*still*) et véhicules privés (VP) sont ceux qui bénéficient de la plus grande amélioration du score de précision à l'issue du processus d'homogénéisation.

Les différentes étapes d'une solution de détection du mode de transport ont été présentées. Dans la prochaine section, une synthèse de l'état de l'art et du cheminement de résolution est réalisée.

4.5 Synthèse

La classification supervisée du mode de transport à partir de données utilisateur est un problème abondamment traité dans la littérature. Le tableau 4.1 compile et classifie de nombreuses solutions proposées. Les approches diffèrent par la nature de leurs données d'entrée (géolocalisation, mesures de capteurs de smartphone, etc.), mais également sur les choix réalisés dans chacune des quatre étapes de la résolution du problème (c.f. figure 4.4).

La segmentation consiste à regrouper les points de données consécutifs en segments. Le regroupement peut se faire à l'aide de fenêtres temporelles ou avec un algorithme de détection des arrêts. Dans ce dernier cas, les segments ont également un sens sémantique, car ils correspondent à un segment réel avec un seul mode de transport.

Les attributs de chaque point forment des séries temporelles sur l'ensemble du segment, et l'on extrait des *features* d'ordre statistique ou profond afin de réduire le bruit des prédictions en sortie et d'exploiter les propriétés statistiques sous-jacentes. Cependant, des *features* trop nombreuses peuvent dégrader les performances des modèles d'apprentissage automatique. Certaines approches proposent de procéder à une sélection de *features* basée sur des métriques d'importances ou des heuristiques. A l'issue de cette éventuelle sélection, les *features* constituent les variables d'entrée du classifieur considéré.

La plupart des approches récentes utilisent des classifieurs d'apprentissage automatique. Il existe également des résolutions hiérarchiques basées sur des règles de décision ainsi que des méthodes hybrides, combinant règles de décision pour certaines classes et modèles d'apprentissage automatique pour les autres.

Dans le cas où la segmentation est réalisée avec une fenêtre temporelle, les prédictions en sortie peuvent être très bruitées. Des algorithmes de post-traitement comme le *Healing* permettent de lisser et même de corriger ces prédictions pour augmenter la précision de classification globale.

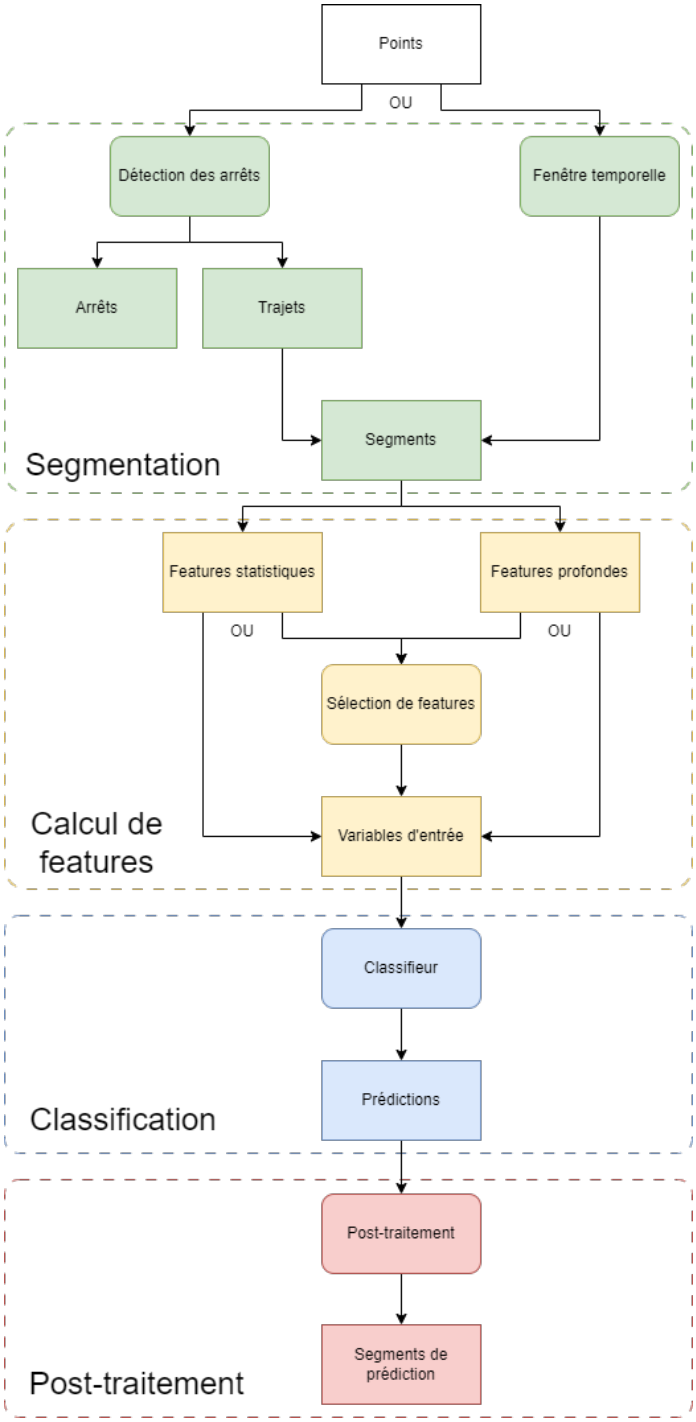


FIGURE 4.4 – Schéma récapitulatif des différentes étapes de résolution du problème de détection du mode de transport.

TABLEAU 4.1 – Etat de l'art comparatif d'approches de résolution du problème de détection du mode de transport.

Référence	**Année**	**Données**				**Modes**	**Segments**	**Précision**	**Sélection features**	**Post traitement**
		Géoloc.	Smart.	Context.	Autres					
[Gon+10]	2010	✓				3	N/A	91.23%		
[Red+10]	2010	✓	✓			5	Fenêtre temporelle (1s)	93.70%	CFS	DHMM
[Nic+10]	2010		✓			3	Fenêtre temporelle (4s)	97%		
[Ste+11]	2011	✓		✓	Live traffic bus	6	Fenêtre temporelle (30s)	93.70%	Chi-squared, information gain	
[Bol+12]	2012	✓				6	Fenêtre temporelle	88%	ANOVA	Matrice de transition
[Gon+12]	2012	✓		✓		5	Détection des arrêts	82.60%		
[WNB12]	2012	✓	✓			8	Détection des arrêts	N/A		
[FT13]	2013	✓	✓			11	Fenêtre temporelle (60s)	92%		
[BLVO13]	2013	✓		✓		9	Détection des arrêts	91.60%		
[San+14]	2014		✓			3	Fenêtre temporelle (200s)	69%-93%		

Tableau 4.1 – Etat de l'art comparatif d'approches de résolution du problème de détection du mode de transport.

Référence	**Année**	**Données**				**Modes**	**Segments**	**Précision**	**Sélection features**	**Post traitement**
		Géoloc.	Smart.	Context.	Autres					
[JR14]	2014	✓	✓			5	Fenêtre temporelle (1s)	98.86%		
[Ras+15]	2015	✓		✓		6	Détection des arrêts	92.40%		
[SH16]	2016		✓			6	Fenêtre temporelle	99.96%		
[ZYS16]	2016	✓	✓			5	Fenêtre temporelle (20s)	93.80%	RF	Healing
[SN+16]	2016	✓		✓		3	Détection des arrêts	N/A		
[Sem+17]	2017	✓		✓		5	N/A	95.20%		
[SB+17]	2017	✓	✓			5	Détection des arrêts	N/A		
[Xia+17]	2017	✓				6	Détection des arrêts	90.77%	RF	
[DS17]	2017	✓				5	Détection des arrêts	90%		
[WGJ17]	2018	✓			Socio-éco.	5	N/A	93.11%	Optimisation combinatoire	
[XCZ19]	2019	✓		✓		6	N/A	93.04%		
[Naw+20a]	2020	✓		✓		4	Fenêtre temporelle (20min)	83.81%		

Tableau 4.1 – Etat de l'art comparatif d'approches de résolution du problème de détection du mode de transport.

Référence	**Année**	**Données**				**Modes**	**Segments**	**Précision**	**Sélection features**	**Post traitement**
		Géoloc.	Smart.	Context.	Autres					
[NA20]	2020	✓		✓		5	N/A	89.10%		
[Li+20]	2020	✓				5	Détection des arrêts	86.70%		
[Gao+20]	2020	✓		✓	Live traffic	4	Détection des arrêts	N/A		
[Naw+20b]	2020	✓				4	Détection des arrêts	93.99%		
[Alo20]	2020		✓			5	Fenêtre temporelle (5s)	95.60%		
[IG20]	2020		✓			13	Fenêtre temporelle	91.10%	ACP, RF	Healing
[CXC+22]	2022	✓		✓		3	N/A	N/A		
[Guo+22]	2022		✓			5	Fenêtre temporelle (2s)	97%		
[Gir+22]	2022	✓	✓		Fréquence cardiaque	5	Fenêtre temporelle (1min)	90%		Healing

Chacun des choix énoncés dans ce chapitre est conditionné par les propriétés des données d'entrée disponibles, ainsi que les contraintes et besoins métier en termes d'implémentation, de performance et d'explicabilité. Dans le chapitre 7, nous reprenons ce processus en justifiant nos choix par rapport au contexte du projet de thèse. La formalisation de ce processus et la classification des références retenues selon cette typologie (c.f. tableau 4.1) peut également être vue comme une contribution à part entière, tant du point de vue scientifique qu'industriel.

Troisième partie

Contributions

Chapitre 5

Smapy : un système multi-agents coopératif et ensembliste de classification supervisée

Les pré-requis présentés dans le chapitre 1 auxquels doit répondre la solution algorithmique développée durant la thèse ont conduit à introduire différentes notions d'apprentissage automatique dans le chapitre 3. Ces notions ont permis de positionner la solution à l'intersection des domaines de l'apprentissage ensembliste, des modèles de voisinage, des systèmes multi-agents d'apprentissage par contexte et même de l'apprentissage par renforcement. Ce chapitre présente le classifieur Smapy ([Fou+22a] et [Fou+22b]). Bien qu'ayant été développé pour répondre au problème de l'analyse de la mobilité, il s'agit d'une contribution au domaine de l'apprentissage automatique et des systèmes multi-agents qui peut être utilisée indépendamment pour tout problème de classification supervisée.

Dans un premier temps, la section 5.1 rappelle les motivations derrière la création de Smapy. Dans la section 5.2, le principe général, les différents éléments de l'architecture, les règles de fonctionnement et les paramètres du système sont introduits et détaillés. Enfin, la section 5.3 est dédiée aux implémentations du système réalisées pendant la thèse.

5.1 Motivations

Dans le chapitre 3, nous avons établi les avantages et inconvénients de différents types de modèles d'apprentissage automatique par rapport aux trois pré-requis présentés au chapitre 1 :

— Capacité d'adaptation rapide aux changements de comportement dans les données et compatibilité avec l'apprentissage en ligne

— Performance sur des problèmes fortement non linéaires

— Explicabilité des prédictions

Afin de répondre à ces critères, nous avons décidé d'utiliser les propriétés des modèles paramétriques pour l'apprentissage en ligne et des modèles de voisinage pour leur résolution locale du problème considéré qui les rend rapidement adaptables aux changements. De plus, l'agrégation ensembliste de classifieurs est reconnue pour ses performances sur des problèmes non linéaires.

Le modèle proposé répond aux trois pré-requis en se plaçant à l'intersection entre apprentissage ensembliste et résolution locale du problème avec des modèles paramétriques, compatibles avec l'apprentissage en ligne. Pour cela, nous

étendons l'architecture SACL (c.f. section 3.3.2) utilisée dans ELLSA [Dat21], en intégrant des modèles d'apprentissage automatique indépendants dans les agents Contexte. Le système, appelé Smapy (Système Multi-Agents ensembliste d'apprentissage par contexte développé en PYthon), est également un classifieur auto-constructeur avec des caractéristiques inspirées de l'apprentissage par renforcement. Il résout des problèmes non linéaires en les transformant en problèmes de coopération locale entre classifieurs plus simples.

Smapy est une évolution des architectures SACL intégrant une dimension ensembliste par le biais des classifieurs internes aux agents Contexte. Dans la prochaine section, le fonctionnement détaillé de Smapy est présenté et formalisé.

5.2 Définition formelle

Cette partie présente le fonctionnement général de Smapy, les différents types d'agents et leurs rôles respectifs et précise les motivations derrière chaque mécanisme de coopération. Enfin, les paramètres internes du système sont rappelés.

5.2.1 Principe général

A l'instar des autres architectures type SACL, Smapy possède deux modes de fonctionnement :

— L'exploration durant laquelle la couverture de l'espace des variables d'entrée par les agents Contexte est modifiée en fonction de nouvelles observations labélisées disponibles. C'est la phase d'entraînement du modèle à partir des données d'entrée.

— L'exploitation durant laquelle le système utilise sa couverture de l'espace des variables d'entrée pour classifier un nouveau point. A ce stade, le modèle est déjà entraîné et il prédit une classe en utilisant la couverture de l'espace des variables d'entrée construite durant la phase d'exploration.

Dans les deux cas, le fonctionnement du système est itératif et chaque cycle démarre avec une nouvelle observation $X_i \in \mathbb{R}^p$. En phase d'exploration, la classe y_i correspondant à la nouvelle observation est également fournie au système. De plus, toujours lors de l'exploration, les agents Contexte activés mettent à jour leur modèle interne avec la dernière observation après avoir proposé une classe de sortie à l'agent Head et reçu un *feedback* (positif ou non). Le *feedback* reçu par un agent Contexte lui permet de mettre à jour sa perception de lui-même au sein du collectif à travers une métrique de performance explicitée dans la section 5.2.3. Il lui permet également de savoir s'il a un comportement non coopératif au regard de l'objectif du système et, le cas échéant, d'agir sur lui-même ou ses voisins pour revenir à un état coopératif (voir section 5.2.4).

5.2.2 Agents

Cette partie présente les trois types d'agents intervenant dans l'architecture SACL de Smapy, dont les relations ont été décrites dans la figure 3.4, à savoir :

— Les agents Percept

— Les agents Contexte

— L'agent Head

5.2.2.1 Percept

Les p agents Percept récupèrent les valeurs des p variables d'entrée de chaque nouvelle observation, et les transmettent aux agents Contexte. Ils stockent également les extrema observés pour chaque variable (c.f. algorithme 1).

Algorithme 1 Lecture d'un nouveau point d'observation et mise à jour des p agents Percept

Entrée: Nouveau point $X_i \in \mathbb{R}^p$
  pour tout $j \in [\![1, p]\!]$ **faire**
    $\text{percept}_j.\text{last\_value} \leftarrow {X_i}^j$
    $\text{percept}_j.\text{min} \leftarrow \min({X_i}^j, \text{percept}_j.\text{min})$
    $\text{percept}_j.\text{max} \leftarrow \max({X_i}^j, \text{percept}_j.\text{max})$
  fin pour

5.2.2.2 Contexte

Un agent Contexte l définit un hypercube dans l'espace des variables d'entrée à p dimensions (i.e. un parallélépipède de dimension p). Pour chaque dimension j, il possède deux paramètres $r_{l,j,0}$ et $r_{l,j,1}$ qui définissent les bornes inférieure et supérieure d'un intervalle d'activation. Un agent Contexte est dit **activé** lorsque le dernier point d'observation du système est à l'intérieur de sa zone d'activation (c.f. algorithme 2). Chaque dimension de l'hypercube d'un agent Contexte est associée à l'agent Percept responsable de la lecture de la variable d'entrée correspondante.

L'agent peut calculer à tout instant v_l, le volume de son hypercube d'activation, selon la formule suivante :

$$v_l = \prod_{j=1}^{p} (r_{l,j,1} - r_{l,j,0}) \tag{5.1}$$

L'agent Contexte possède également un niveau de confiance c_l, fonction de son historique $\mathcal{H}_l$ (ensemble de ses cycles d'activations depuis sa création), de ses propositions de classe $(\hat{y}_l^i)_{i \in \mathcal{H}_l}$ pour les observations $(y_i)_{i \in \mathcal{H}_l}$ sur cet historique, et de deux paramètres externes F_+ et F_- qui pondèrent respectivement les *feedbacks* positifs et négatifs de l'agent Head :

Algorithme 2 Activation des q agents Contexte à l'apparition d'un nouveau point d'observation

last_activated_contexts = []
pour tout $l \in [\![1, q]\!]$ **faire**
  pour tout $j \in [\![1, p]\!]$ **faire**
    si $r_{l,j,0} \leq \text{percept}_j.\text{last\_value} \leq r_{l,j,1}$ **alors**
      last_activated_contexts.append(l)
    fin si
  fin pour
fin pour
Sortie: last_activated_contexts

$$c_l = \sum_{i \in \mathcal{H}_l} (F_+ * \mathbb{1}_{\hat{y}_l^i = y_i} - F_- * \mathbb{1}_{\hat{y}_l^i \neq y_i}) \tag{5.2}$$

A partir du niveau de confiance, nous définissons le score s_l d'un agent Contexte à l'aide d'une fonction de normalisation N_c qui est un paramètre externe de Smapy :

$$s_l = N_c(c_l) \tag{5.3}$$

Enfin, l'agent Contexte possède un modèle interne de classification supervisée $\hat{f}_l$ appris sur les observations l'ayant activé. Ce modèle lui permet d'apprendre localement à classifier les points qui apparaissent dans sa zone d'activation, à partir de l'historique des points qu'il a observés. L'implémentation python de Smapy rend possible l'utilisation de modèles type *scikit-learn* à condition qu'ils supportent l'apprentissage en ligne nécessaire à l'adaptation de l'agent à de nouvelles observations. Pour la suite de cet article, nous définissons plusieurs propriétés des agents Contexte :

Definition 5.2.1 (Dilatation/Rétractation)**.** Un agent Contexte se dilate (resp. rétracte) d'un facteur α lorsque qu'il augmente (resp. diminue) ses frontières de manière à multiplier son volume par $1 + \alpha$ (resp. $1 - \alpha$). Le mécanisme de dilatation (resp. rétractation) est illustré par la figure 5.1 (resp. 5.2). On dit que la dilatation (resp. rétractation) se fait vers un point X_i lorsque la dimension et la direction sont telles que la distance du point X_i au nouveau centre de l'agent est minimisée (resp. maximisée).

Definition 5.2.2 (Poussée)**.** Un agent Contexte l_1 pousse un agent Contexte l_2 lorsque l_2 se rétracte de façon que la zone d'intersection de l_1 et l_2 se retrouve totalement en dehors de l_2 (et donc contenue uniquement dans l_1). Le mécanisme de poussée est illustré par la figure 5.3. La dimension de la poussée est celle qui minimise la perte de volume de l'agent Contexte l_2.

Definition 5.2.3 (Absorption)**.** Un agent Contexte l_1 absorbe un agent contexte l_2 lorsque l_1 se dilate de façon à contenir entièrement la zone recouverte par l_2

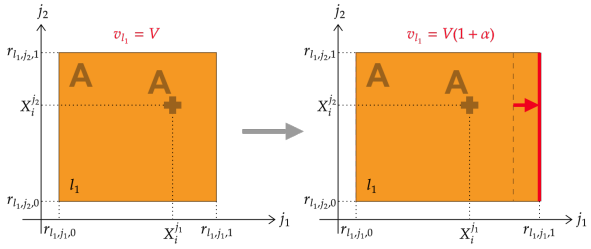


FIGURE 5.1 – Schéma de la dilatation d'un agent Contexte l_1 ayant prédit la classe A d'un facteur α vers le point X_i de classe A.

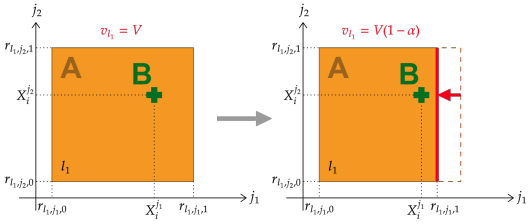


FIGURE 5.2 – Schéma de la rétractation d'un agent Contexte l_1 ayant prédit la classe A d'un facteur α vers le point X_i de classe B.

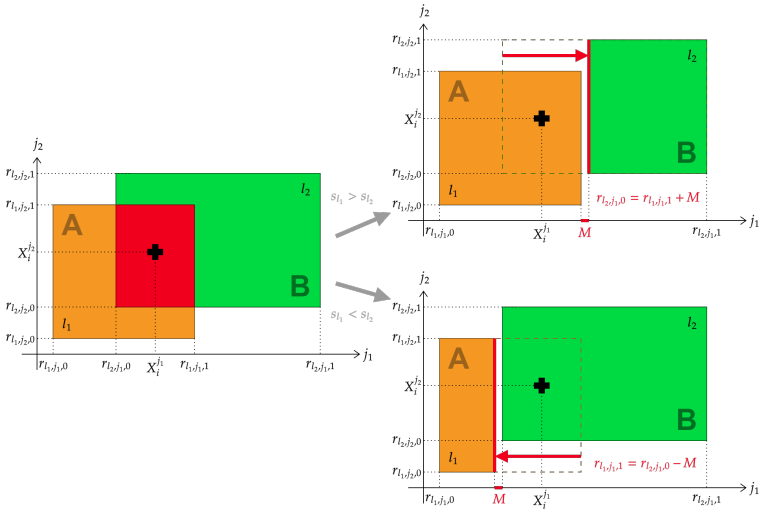


FIGURE 5.3 – Schéma d'une poussée entre deux agents Contexte l_1 et l_2 ayant prédit des classes différentes A et B.

et que l'agent l_2 est détruit (c.f. figure 5.4).

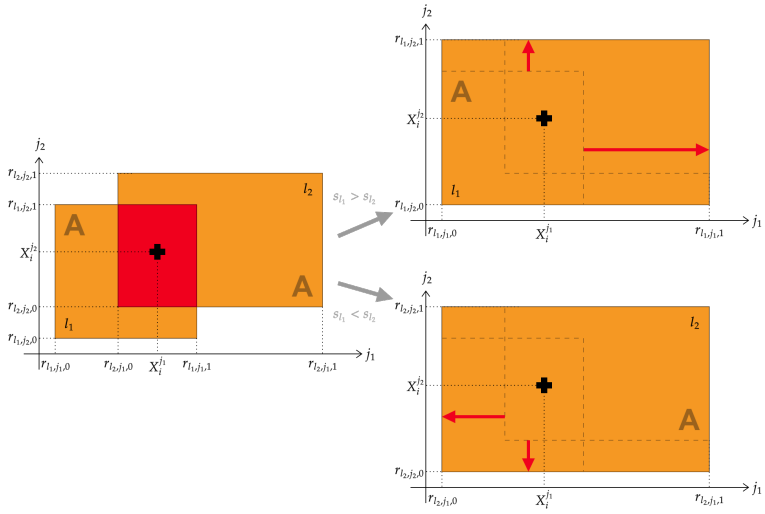


FIGURE 5.4 – Schéma d'une absorption entre deux agents Contexte l_1 et l_2 ayant prédit la même classe A.

Definition 5.2.4 (Exclusion de point)**.** Un agent Contexte l_1 exclut une observation X_i lorsque l_1 se rétracte de façon à ce que X_i se retrouve en dehors de l_1. L'exclusion de point est contrôlée par un paramètre booléen externe E (c.f. figure 5.5). La dimension de l'exclusion est celle qui minimise la perte de

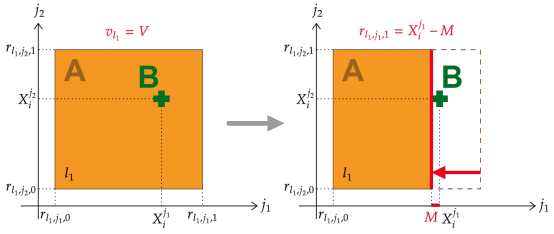


FIGURE 5.5 – Schéma de l'exclusion du point X_i de classe B d'un agent Contexte l_1 ayant prédit la classe A.

volume de l'agent Contexte l_1.

Definition 5.2.5 (Indice de superposition)**.** L'indice de superposition o_{l_1,l_2} est le rapport entre le volume de l'intersection de deux agents Contexte l_1 et l_2 et le minimum des volumes de ces agents :

$$o_{l_1,l_2} = o_{l_2,l_1} = \frac{v_{l_1 \cap l_2}}{\min(v_{l_1}, v_{l_2})} \tag{5.4}$$

Definition 5.2.6 (Réinitialisation)**.** Un agent contexte l_1 se réinitialise lorsqu'il « oublie » son modèle interne (i.e. un nouveau modèle interne est instancié et entraîné uniquement sur le dernier point observé) et remet à zéro son niveau de confiance c_{l_1}.

5.2.2.3 Head

L'agent Head organise la coopération des agents Contexte. A chaque itération, il reçoit les propositions de classes des agents Contexte activés, sélectionne la classe proposée par l'agent Contexte ayant le score le plus élevé (ou procède par vote majoritaire en cas d'égalité) et renvoie un *feedback* à tous les agents activés durant la phase d'exploration (voir section 5.2.3). L'agent Head peut également créer de nouveaux agents Contexte en cas d'incompétence du système (c.f. section 5.2.4). La gestion d'une itération du système par l'agent Head est résumée dans l'algorithme 3.

5.2.3 Feedback

Lorsque les agents Contexte sont activés, ils proposent une prédiction à l'agent Head. Ce dernier sélectionne la prédiction de l'agent ayant le score le plus élevé. Durant la phase d'exploration (apprentissage), l'agent Head renvoie un *feedback* aux agents Contexte ayant proposé une prédiction.
Si la prédiction est bonne (par rapport au label du nouveau point), alors la confiance de l'agent contexte augmente de F_+ et il se dilate d'un facteur α (paramètre externe) vers ce point.
Si la prédiction est mauvaise, alors la confiance de l'agent contexte diminue de F_-. Si l'exclusion de point est autorisée (i. e. E est vrai), alors l'agent Contexte exclut le nouveau point. Sinon, l'agent Contexte se rétracte d'un facteur α vers ce point.
A l'issue du *feedback* (positif ou négatif), si le point d'observation se trouve encore dans la zone d'activation de l'agent Contexte, le modèle local de ce dernier apprend partiellement le nouveau point (au sens de l'apprentissage en ligne). La gestion des *feedbacks* est détaillée dans l'algorithme 4.

Lorsque le taux de mauvaises prédictions par rapport aux bonnes prédictions d'un agent Contexte est supérieur à un paramètre externe optionnel Z, il est réinitialisé. Pour connaître ce taux, il faudrait *a priori* stocker en mémoire les nombres de bonnes et mauvaises prédictions de chaque agent. Cependant, la connaissance des valeurs des *feedbacks* positif F_+ et négatif F_- ainsi que l'initialisation du score de confiance à zéro nous permet d'établir la formule suivante pour le niveau de confiance critique $c^*(Z, z_l)$ de tout agent Contexte l :

$$c^*(Z, z) = \frac{z}{Z+1}(F_+ - ZF_-) \tag{5.5}$$

Algorithme 3 Prise de décision de l'agent Head

Entrée: Nouveau point $X_i \in \mathbb{R}^p$
Sortie: Décision $\hat{y}$ (pour le mode exploitation)

```
Mise à jour des agents Percept                                  ▷ c.f. algorithme 1
Activation des agents Contexte                                  ▷ c.f. algorithme 2
A ← agents Contexte activés
si len(A) = 1 alors
    b = A[0]
    ŷ ← f̂_b(X_i)
sinon si len(A) > 1 alors
    S ← []                                                      ▷ Liste des scores
    pour l ∈ A faire
        S.append(s_l)                                           ▷ Ajout du score de l'agent l
        B ← argmax(S)                                           ▷ Liste des meilleurs agents Contexte
    fin pour
    si len(B) = 1 alors
        b ← B[0]                                                ▷ Meilleur agent Contexte
        ŷ ← f̂_b(X_i)
    sinon
        P ← []                                                  ▷ Liste des prédictions des meilleurs agents
        pour b ∈ B faire
            P.append(f̂_b(X_i))                                  ▷ Ajout de la prédiction de l'agent b
        fin pour
        ŷ ← mode(P)                                             ▷ Vote majoritaire
    fin si
    si mode exploration alors
        Feedback aux agents Contexte activés                    ▷ c.f. algorithme 4
        Résolution des SNC de concurrence et de conflit         ▷ c.f. algorithme 6
    fin si
sinon                                                           ▷ Aucun agent activé
    Résolution d'une SNC d'incompétence                         ▷ c.f. algorithme 5
fin si
```

Par exemple, si $Z = 3$, $F_+ = 1$ et $F_- = 2$, alors le niveau de confiance c_l d'un agent Contexte l ayant été activé $z_l = 4$ fois passe en-dessous du niveau de confiance critique $c^*(3, 4) = -5$ lorsque son nombre de mauvaises prédictions est au moins 3 fois supérieur à son nombre de bonnes prédictions. Dans ce cas, il est réinitialisé.

Motivation Si un agent Contexte l prédit la bonne classe d'un point d'observation, alors son modèle pourrait également faire de bonnes prédictions au voisinage de ce point. Il est donc pertinent de dilater sa zone d'activation pour explorer une portion d'espace supplémentaire au-delà de ce point. A l'inverse, si la prédiction est mauvaise, l'agent Contexte a peut-être exploré trop loin au-delà

d'une frontière de classe et son modèle n'y est plus adapté. La rétractation sert donc à réduire la taille de la zone d'activation de façon à revenir du bon côté de la frontière de classe théorique. Ce double mécanisme est similaire à celui implémenté dans ELLSA [Dat21] et se produit de façon itérative jusqu'à stabiliser la zone d'activation des agents Contexte de part et d'autre d'une frontière de classe. Cependant, Smapy introduit un paramètre additionnel d'exclusion E qui permet, en cas de mauvaise prédiction, d'exclure le point mal classifié au lieu de simplement rétracter la zone d'activation de l'agent Contexte.
Enfin, la mise-à-jour du niveau de confiance c_l (et donc du score s_l) permet de comparer les agents Contexte entre-eux sur la base de leur performances passées, afin de résoudre des situations de non coopération (c.f. section 5.2.4). L'asymétrie potentielle des *feedbacks* positif (F_+) et négatif (F_-) permet de contrôler la sévérité du système vis-à-vis des agents Contexte à la manière d'un modèle d'apprentissage par renforcement. La réinitialisation est utilisée en dernier recours lorsqu'un agent Contexte n'est plus capable de s'adapter aux changements et effectue trop de mauvaises prédictions.

Algorithme 4 Gestion des *feedbacks* par les agents Contexte

Entrée: Nouveau point $X_i \in \mathbb{R}^p$, agents Contexte activés A, classe réelle y_i, facteur de dilatation α, *feedback* positif F_+, *feedback* négatif F_-, paramètre d'expulsion de point E, facteur de réinitialisation Z

pour tout $l \in A$ **faire**
  si $\hat{f}_l(X_i) = y_i$ **alors** ▷ Bonne prédiction
    L'agent Contexte l se dilate vers X_i d'un facteur α
    $c_l \leftarrow c_l + F_+$
  sinon ▷ Mauvaise prédiction
    si E est vrai **alors**
      L'agent Contexte l exclut X_i
    sinon
      L'agent Contexte l se rétracte vers de X_i d'un facteur α
    fin si
    $c_l \leftarrow c_l - F_-$
  fin si
  si X_i est dans la zone d'activation de l **alors**
    Le modèle interne de l intègre X_i dans son apprentissage
  fin si
  si $c_l \leq c^*(Z, z_l)$ **alors**
    L'agent Contexte l est réinitialisé
  fin si
fin pour

5.2.4 Situations de non coopération

Les AMAS abordent les problèmes à résoudre comme des problèmes de coopération entre agents. Les situations de non coopération (SNC) sont des états durant lesquels le comportement du système doit évoluer pour atteindre l'objectif fixé par l'utilisateur. En apprentissage par contexte, cela se traduit par le ré-agencement des agents Contexte afin d'améliorer le pavage de l'espace des variables d'entrée. Nous présentons dans cette partie les trois types de SNC qui peuvent survenir durant le fonctionnement de Smapy ainsi que leur résolution.

5.2.4.1 Incompétence

L'incompétence apparaît lorsqu'aucun agent Contexte n'a été activé. En phase d'exploration, un nouvel agent Contexte est créé autour du nouveau point (c.f. figure 5.6) et les SNC éventuellement engendrées sont résolues. Le rayon initial du nouvel agent est contrôlé par un paramètre externe R. En phase d'exploitation, l'agent Contexte le plus proche du nouveau point (au sens de la distance Euclidienne entre le point et la frontière de l'agent) propose sa prédiction (c.f. algorithme 5).

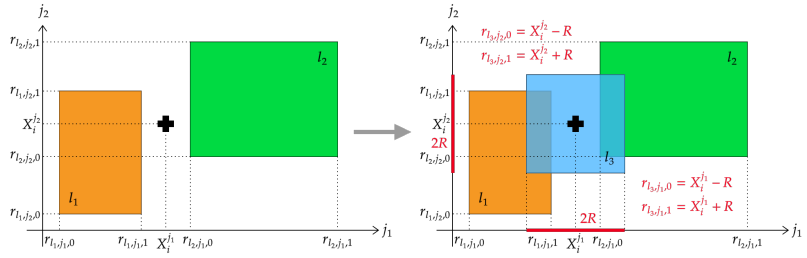


FIGURE 5.6 – Schéma de la résolution d'une situation d'incompétence en phase d'exploration par la création d'un nouvel agent Contexte l_3 autour de l'observation X_i.

Motivation Tout comme dans ELLSA [Dat21], l'incompétence en phase d'exploitation est résolue en choisissant l'agent Contexte le plus proche du point d'observation, car l'on suppose que son modèle interne est le plus adapté. Les agents Contexte ayant généralement un modèle interne linéaire, cela revient à faire l'hypothèse de la linéarité locale du problème de classification à résoudre. En phase d'exploration, il est nécessaire d'instancier un nouvel agent Contexte pour poursuivre l'apprentissage du modèle.

5.2.4.2 Concurrence

La concurrence apparaît pendant la phase d'exploration lorsque deux agents Contexte activés proposent une même prédiction (en l'occurrence, une même

classe). Si un seuil de superposition est défini à travers le paramètre externe O, et si l'indice de superposition des deux agents est supérieur à ce seuil, l'agent avec le score le plus élevé absorbe l'autre. Sinon, l'agent Contexte avec le score le plus élevé pousse l'autre agent. La résolution d'une SNC de concurrence est détaillée dans l'algorithme 6.

Motivation Si deux agents Contexte l_1 et l_2 sont activés simultanément, il existe nécessairement une intersection non nulle de leurs zones de superposition. Dans le cas où les deux agents prédisent la même classe, nous faisons l'hypothèse qu'il est inutile qu'ils soient tous les deux présents sur cette intersection. Ainsi, de la même manière que dans ELLSA, l'intersection est éliminée par une poussée dans le cas où le score de superposition des agents o_{l_1,l_2} est inférieur au paramètre de seuil de superposition O, c'est-à-dire dans le cas où les deux zones d'activation sont faiblement superposées. En revanche, contrairement à ELLSA, lorsque les agents Contexte sont fortement superposés (relativement à leurs volumes), nous considérons que l'un des deux agents peut être absorbé, c'est-à-dire être supprimé et voir sa zone d'activation ré-allouée à l'agent Contexte de meilleur score.

5.2.4.3 Conflit

Un conflit apparaît pendant la phase d'exploration lorsque deux agents Contexte activés proposent des prédictions différentes (en l'occurrence, des classes différentes). L'agent de score le plus élevé pousse alors l'autre agent (c.f. algorithme 6).

Motivation Il est problématique que deux agents Contexte superposés prédisent des classes différentes. La résolution d'un conflit, qui consiste à supprimer l'intersection entre les deux agents à l'aide d'une poussée, est similaire à celle implémentée dans ELLSA. Contrairement à la résolution d'une concurrence, l'absorption n'est pas envisagée car la présence d'agents en conflit traduit l'existence d'une frontière de classes au niveau de leur intersection. Il n'est donc pas pertinent d'agrandir la zone d'activation d'un des agents au-delà de cette frontière par le biais d'une absorption, car son modèle interne y serait *a priori* inadapté.

5.2.5 Paramètres internes

Les paramètres internes de Smapy présentés dans ce chapitre sont résumés dans le tableau 5.1. Lors d'une poussée, un agent Contexte réduit sa zone d'activation de façon à éliminer l'intersection avec l'autre agent Contexte. Cependant, même avec une intersection de mesure nulle, il est possible qu'un nouveau point active encore les deux agents s'il se trouve exactement sur leur frontière commune. C'est pourquoi lors de l'implémentation, un paramètre additionnel M, qui désigne la marge minimale des agents Contexte, est ajouté afin de s'assurer d'une distance minimale de non couverture entre les zones

Algorithme 5 Résolution d'une SNC d'incompétence

Entrée: Nouveau point $X_i \in \mathbb{R}^p$, liste des agents Contexte C
Sortie: Décision $\hat{y}$ (pour le mode exploitation)
  si mode exploitation **alors**
    $D \leftarrow []$ ▷ Liste des distances aux frontières des agents Contexte
    pour tout agent Contexte l **faire**
      D.append($\|l - X_i\|$) ▷ Distance entre X_i et la frontière de l'agent l
    fin pour
    $b = C[\text{argmin}(D)]$ ▷ Sélection de l'agent le plus proche
    $\hat{y} \leftarrow \hat{f}_b(X_i)$
  sinon si mode exploration **alors**
    Création d'un nouvel agent Contexte $q + 1$ autour du point X_i
  fin si

d'activation des agents Contexte. La valeur souhaitée de M est comparable à l'erreur machine.

A travers la gestion des SNC et des *feedbacks*, Smapy transforme un problème de classification sur des données *a priori* non linéaires en un problème de coopération entre agents Contexte dont le rôle est d'assurer une résolution locale du problème sur des zones supposées plus linéaires. On intégrera donc généralement des modèles paramétriques (c.f. section 3.1.1.1) linéaires dans les agents Contexte. Il est donc nécessaire de fournir en plus des neuf paramètres internes présentés dans le tableau 5.1 deux paramètres additionels :

— Le modèle interne par défaut à instancier dans les agents Contexte au moment de leur création (`local_model_default` dans l'implémentation).
— Les paramètres internes de ce modèle (`local_model_default_params` dans l'implémentation).

Le fonctionnement de Smapy a été présenté, formalisé et détaillé, de la même manière que les motivations sous-jacentes. Dans la prochaine section, la façon dont Smapy a été implémenté durant la thèse est présentée.

5.3 Implémentation

Cette section présente les principales caractéristiques des deux implémentations de Smapy réalisées pendant le projet de thèse :

— Une implémentation "explicite" qui reprend la structure de l'architecture SACL telle que présentée dans la section 3.3.2.
— Une implémentation plus abstraite, dite "implicite", qui vise à optimiser les temps de calcul et les échanges de données d'une classe à l'autre.

Algorithme 6 Résolution des SNC de concurrence et de conflit

Entrée: Nouveau point $X_i \in \mathbb{R}^p$, agents Contexte activés A, seuil de superposition O

```
pour tout l1, l2 ∈ A, l1 ≠ l2 faire
    si f̂_l1(X_i) = f̂_l2(X_i) alors                    ▷ SNC de concurrence
        si o_l1,l2 ≥ O alors
            si s_l1 > s_l2 alors
                l1 absorbe l2
            sinon si s_l1 < s_l2 alors
                l2 absorbe l1
            sinon                                      ▷ Egalité des scores
                L'agent Contexte le plus récent absorbe l'autre
            fin si
        sinon
            si s_l1 > s_l2 alors
                l1 pousse l2
            sinon si s_l1 < s_l2 alors
                l2 pousse l1
            sinon                                      ▷ Egalité des scores
                L'agent Contexte le plus récent pousse l'autre
            fin si
        fin si
    sinon                                              ▷ SNC de conflit
        si s_l1 > s_l2 alors
            l1 pousse l2
        sinon si s_l1 < s_l2 alors
            l2 pousse l1
        sinon                                          ▷ Egalité des scores
            L'agent Contexte le plus récent pousse l'autre
        fin si
    fin si
fin pour
```

5.3.1 Implémentation explicite

L'implémentation explicite a été réalisée au début de la thèse. Elle reprend exactement la structure des systèmes SACL et associe une classe à chaque type d'agents, ainsi qu'une classe `SmapyClassifier` qui gère le système global. Le diagramme de classes 5.7 illustre les relations entre les différentes classes.

Dans cette première implémentation, les p agents Percept, instances de la classe `Percept`, lisent les valeurs d'un nouveau point d'observation sur chacune de ses dimensions et mettent à jour leurs extrema. L'agent Head, unique instance de la classe `Head`, est en charge de la prise de décision à travers la méthode `decide`. La classe `Context` est la plus riche. Chaque agent Contexte en est une instance, et peut :

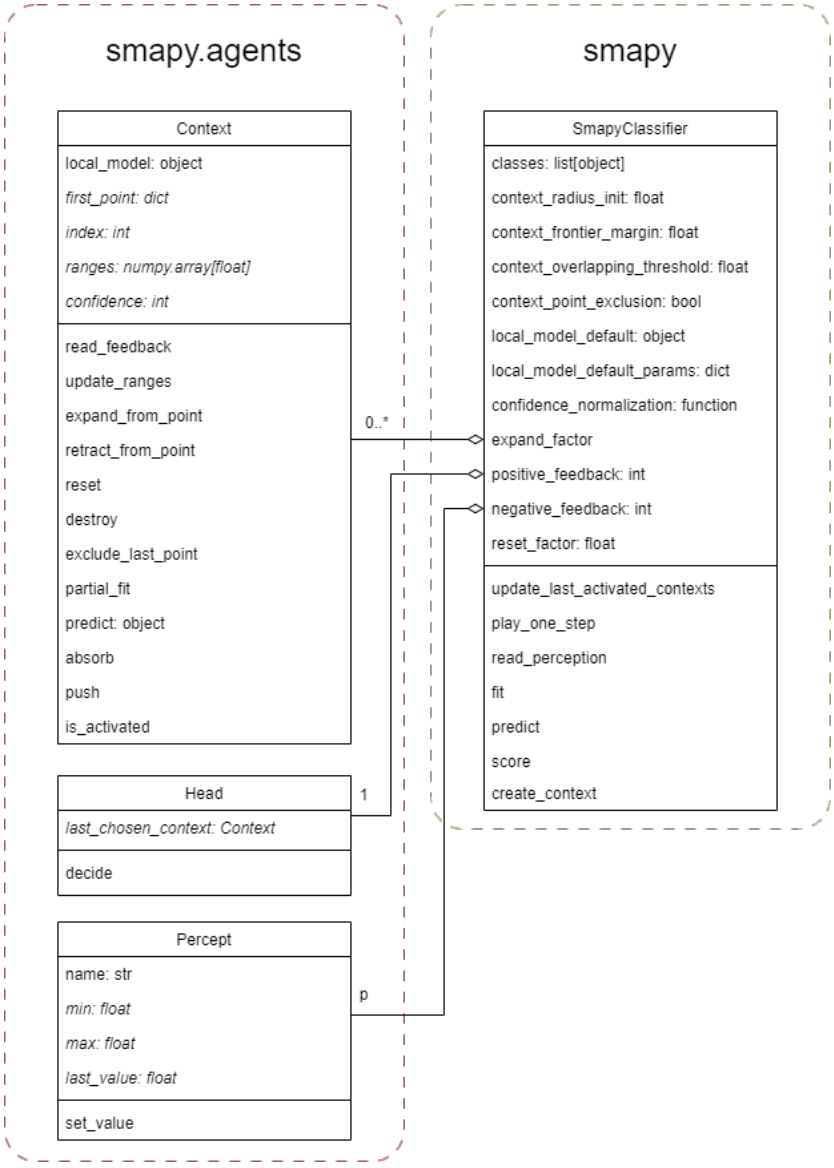


FIGURE 5.7 – Diagramme de classes de l'implémentation explicite de Smapy

TABLEAU 5.1 – Paramètres internes de Smapy.

Paramètre	Symbole	Implémentation	Type
Rayon initial des agents Contexte	R	`context_radius_init`	réel positif
Marge minimale des agents Contexte	M	`context_frontier_margin`	réel positif
Seuil de superposition pour l'absorption	O	`context_overlapping_thershold`	réel positif
Exclusion de point	E	`context_point_exclusion`	booléen
Fonction de normalisation du niveau de confiance	N_c	`confidence_normalization`	fonction réelle croissante
Facteur de dilatation/-rétractation	α	`expand_factor`	réel positif
Feedback positif	F_+	`positive_feedback`	entier positif
Feedback négatif	F_-	`negative_feedback`	entier positif
Facteur de réinitialisation	Z	`reset_factor`	réel strictement positif

— Prédire la classe du dernier point observé avec son modèle interne `local_model` et la méthode `predict`.

— Lire les *feedbacks* renvoyés par l'agent Head avec la méthode `read_feedback` et mettre à jour sa zone d'activation `ranges` avec la méthode `update_ranges`.

— Appliquer les mécanismes de résolution des SNC de concurrence et de conflit avec les méthodes `push` et `absorb`.

La classe `SmapyClasifier` est chargée de recevoir les paramètres du système (c.f. section 5.2.5) d'organiser les itérations avec la méthode `play_one_step`. De plus, la méthode `create_context` permet d'instancier un nouvel agent Contexte pour résoudre une SNC d'incompétence.

Le choix de python pour l'implémentation de Smapy est motivé par la disponibilité de très nombreux classifieurs à travers tout autant de librairies. En particulier, la classe `SmapyClassifier` a été conçue de manière à s'interfacer avec les classes d'hyperparamétrisation de la librairie *scikit-learn* [Ped+11] comme `GridSearchCV` et les structures de chaînage de classifieurs comme la classe `Pipeline` qui permet d'ajouter des étapes de pré-traitement et de normalisation de données en amont du modèle d'apprentissage. De plus, la classe `Context` accepte des classifieurs *scikit-learn* comme modèle interne à condition qu'ils soient compatibles avec l'apprentissage en ligne.

5.3.2 Implémentation implicite

Dans un objectif de simplification et d'optimisation du temps de calcul, une seconde implémentation de Smapy a été réalisée. La principale nouveauté est la suppression de la classe `Head` et la ré-assignation du rôle d'agent Head à la classe principale `SmapyClassifier2`, c'est-à-dire au système global. En particulier, la fonction de décision `_decide` est désormais une méthode de la classe principale, qui gère toujours les itérations à travers la méthode `_play_one_step` (c.f. figure 5.8). Ce choix répond au constat que l'agent Head peut conceptuellement être assimilé au système général dans la mesure où il coordonne en partie le fonctionnement des agents Contexte. De plus, les échanges d'informations entre objets sont réduits.

Une deuxième différence notoire est la construction de la classe principale `SmapyClassifier2` comme héritière de la classe `BaseEstimator` de *scikit-learn*. Ainsi, certaines méthodes obligatoires pour la compatibilité de Smapy avec des structures telles que `GridSearchCV` ou `Pipeline` de *scikit-learn* sont héritées afin de réduire la taille du code. Des fonctions génériques de vérification de paramètres issues de *scikit-learn* sont également utilisées dans Smapy. Globalement, l'implémentation implicite de Smapy se veut conçue dans la philosophie de *scikit-learn* et de l'ensemble de règles PEP8.

En termes d'optimisation de temps de calcul, cette implémentation se différencie de la version explicite par l'agrégation de tous les agents Contexte dans une unique instance de la nouvelle classe `Contexts` qui permet de remplacer des calculs itératifs par des calculs matriciels. En particulier, les calculs géométriques tels que l'activation d'agents, les calculs de volume, les dilatations ou encore les rétractations sont résolus simultanément à chaque itération grâce à l'attribut `boundaries` qui compile dans un tableau *numpy* toutes les frontières de décision de tous les agents Contexte. De la même manière, l'agrégation de tous les modèles internes dans l'attribut `models` permet d'entraîner simultanément les modèles des agents activés à chaque itération. Enfin, l'utilisation d'une unique classe `Contexts` permet de simplifier significativement les créations et suppression d'agents Contexte qui ne sont plus des instances séparées.

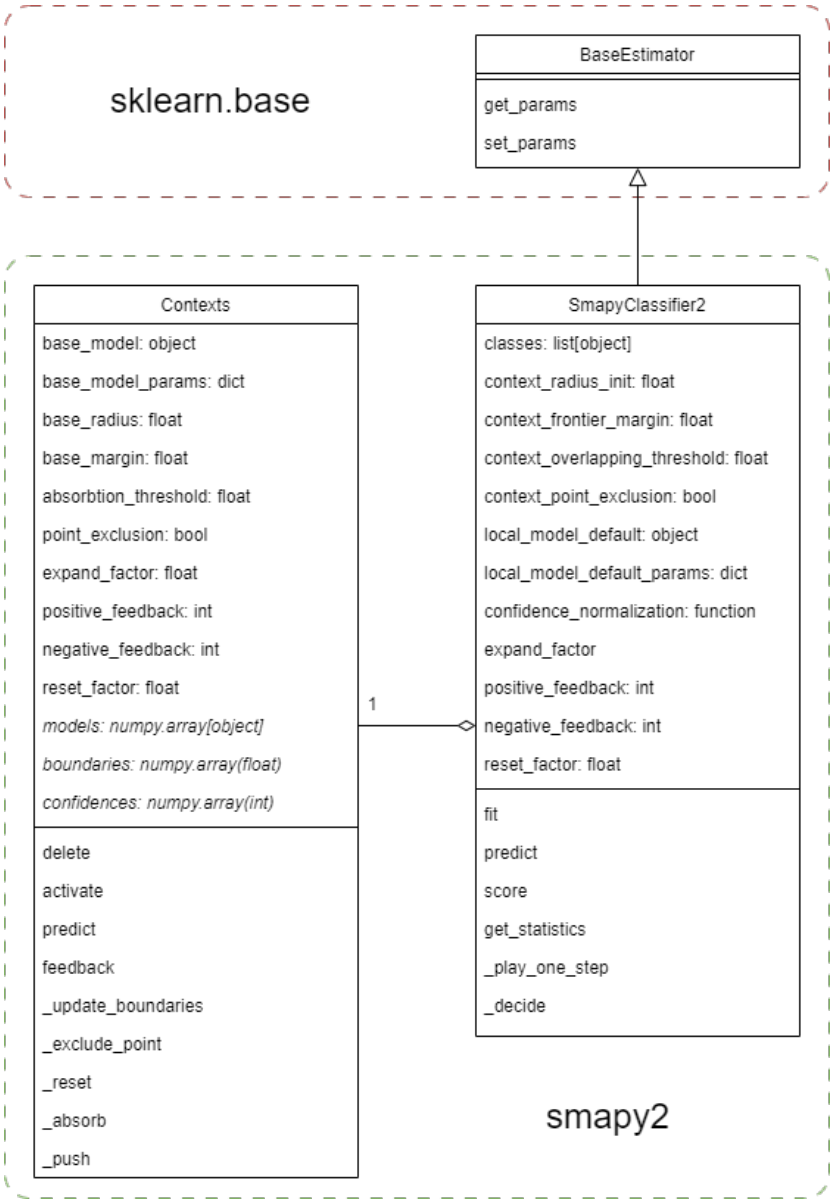


FIGURE 5.8 – Diagramme de classes de l'implémentation implicite de Smapy

5.3.3 Modules complémentaires

Afin de valider le fonctionnement de Smapy et d'interpréter le positionnement des agents Contexte les uns par rapport aux autres, un module de représentation graphique a été développé en python pour les trois premières dimensions (i.e. $1 \leq p \leq 3$) comme illustré sur la figure 5.9.

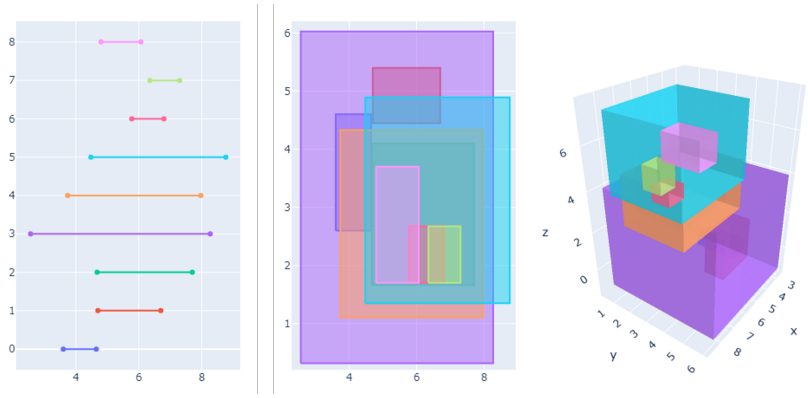


FIGURE 5.9 – Visualisation des agents Contexte sur des instances de Smapy entraînées sur des données jouet à une (gauche), deux (milieu) et trois (droite) dimensions. Sur la représentation en une dimension, l'axe vertical représente les indices des agents Contexte par souci de lisibilité.

De plus, des scripts ont été développés pour mener des tests reproductibles de Smapy sur plusieurs datasets réels ou jouets. Une interface graphique a été développée avec *tkinter* pour contrôler le choix du dataset, les valeurs des paramètres et visualiser le positionnement des agents Contexte (c.f. figure 5.10).

Le classifieur Smapy est ensembliste car il agrège les classifieurs internes contenus dans les agents Contexte. Cette agrégation s'effectue de manière géométrique dans l'espace des variables d'entrée, ce qui en fait également un modèle de voisinage. Ces caractéristiques géométriques sont dues à un ensemble de règles de fonctionnement ayant pour but d'assurer la coopération entre les agents Contexte afin de résoudre localement des problèmes de classification supervisée. Dans le chapitre 6, une expérimentation est menée pour montrer que ces règles de fonctionnement permettent de transformer un problème de classification non linéaire en un problème de coopération entre agents qu'il est possible de résoudre même si leurs classifieurs internes sont des modèles linéaires.

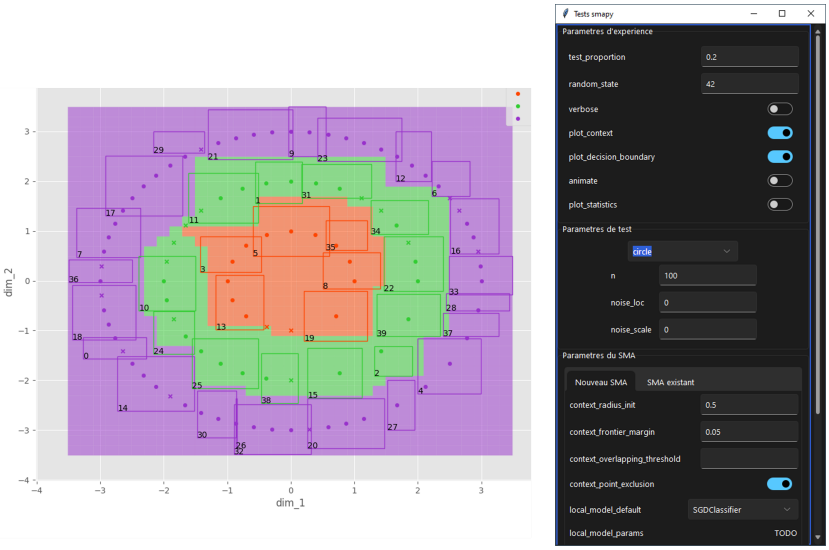


FIGURE 5.10 – Interface graphique de contrôle des paramètres d'expérimentation de Smapy (droite) et visualisation des agents Contexte (gauche).

Chapitre 6

Smapy : d'un problème de classification à un problème de coopération

Le chapitre 5 a introduit Smapy et ses règles de fonctionnement basées sur la coopération entre les agents Contexte. Dans ce chapitre, nous présentons l'expérimentation de comparaison entre quatre des modèles linéaires présentés dans la section 3.1.1.1 (régression logistique, SVM linéaire, PA-I et PA-II) et des instances de Smapy avec ces mêmes modèles à l'intérieur des agents Contexte dans des problèmes simples de classification supervisée. La motivation de cette expérience est de vérifier si la transformation d'un problème de classification en problème de coopération multi-agents sans changer les techniques d'apprentissage automatique permet d'améliorer les performances de ces dernières. En particulier, il est montré que des modèles linéaires, lorsqu'ils sont intégrés à des agents Contexte dans Smapy, sont capables de résoudre des problèmes de classification non linéaire grâce aux règles de coopération.

Les données d'entrée de cette expérimentation sont présentées dans la section 6.1. Le protocole expérimental, basé sur une double validation croisée, est ensuite détaillé dans la section 6.2. La section 6.3 présente les résultats de l'expérimentation sur chaque dataset, qui sont ensuite discutés dans la section 6.4.

6.1 Données d'entrée

L'expérimentation est menée sur trois jeux de données de classification binaire à deux dimensions, inclus dans la librairie *scikit-learn* [VMG] et régulièrement utilisés à des fins de comparaison de modèles :

— *Moons* : Deux nuages de points entrelacés expliqués par les deux variables (`noise=0.3`)

— *Circles* : Un nuage de points circulaire enclavé dans un autre nuage en forme d'anneau (`noise=0.2`, `factor=0.5`)

— *Linearly separable* : Deux nuages de points avec une frontière linéaire expliquée par une seule des deux variables

Chaque jeu de données contient 100 points. Ils sont centrés et réduits à l'aide du modèle `StandardScaler` de *scikit-learn* avant apprentissage.

La prochaine section détaille le protocole expérimental pour tester la différence de performances entre modèles seuls et instances de Smapy sur chacun des trois datasets présentés.

6.2 Protocole expérimental

Etape 1 L'expérimentation se déroule en deux parties. Dans un premier temps, nous recherchons pour chacun des 4 modèles linéaires la combinaison optimale de paramètres parmi une grille de paramètres présentée dans le tableau 6.1 à l'aide d'une validation croisée à cinq blocs.

Etape 2 Une fois ces combinaisons obtenues, nous recherchons pour chaque modèle linéaire la combinaison optimale de paramètres de Smapy parmi une grille de paramètres présentée dans le tableau 6.2 avec une validation croisée à cinq blocs. Les agents Contexte des instances de Smapy ont pour modèle interne le modèle linéaire correspondant, entraîné avec les paramètres de sa combinaison optimale obtenue précédemment.

Ce protocole est répété pour chaque jeu de données afin d'obtenir 12 instances de Smapy ainsi que 12 instances de modèles linéaires correspondantes, dont tous les paramètres ont été optimisés par validation croisée. On compare ensuite les instances de Smapy optimisées avec les modèles linéaires à l'aide de deux métriques d'évaluation :

— Précision de classification (multi-classes) moyenne sur les cinq itérations de la validation croisée (étape 1 pour les modèles linéaires, étape 2 pour les instances de Smapy).

— Frontières de décision des modèles (linéaires ou Smapy) entraînés avec les meilleures combinaisons de paramètres obtenues par validation croisée.

Tableau 6.1 – Liste des grilles de valeurs pour la recherche des combinaisons optimales de paramètres des modèles linéaires étudiés (les autres paramètres gardent leur valeur par défaut dans l'implémentation *scikit-learn*).

Paramètre	**Grille de valeurs**		
	Régression logistique & SVM linéaire		
`alpha`	0.0001	0.001	0.01
`penalty`	l_1	l_2	ElasticNet
	PA-I & PA-II		
`C`	0.5	1.0	2.0

Ce protocole expérimental permet de comparer des instances de modèles linéaires seuls et de Smapy hyperparamétrées, c'est-à-dire entraînées sur les mêmes données avec des combinaisons de paramètres optimales. Dans la prochaine

Tableau 6.2 – Liste des grilles de valeurs pour la recherche des combinaisons optimales de paramètres des Smapy instanciés.

Paramètre	Grille de valeurs		
R	0.1	0.2	0.5
M	10^{-6}		
O	0.2	0.5	
E	Faux	Vrai	
N_c	Sigmoïde		
α	0.0	0.1	0.2
F_+	1.0		
F_-	0.5	1.0	2.0
Z	Aucun		

section, les performances des différents modèles testés sont comparées sur les données de test.

6.3 Résultats

Dans cette section, nous présentons les résultats comparatifs entre modèles linéaires seuls et instances de Smapy selon les deux métriques introduites précédemment (c.f. section 6.2).

Précision de classification Le tableau 6.3 compile les précisions de classification obtenues. Nous remarquons qu'elles diffèrent grandement selon le jeu de données d'entrée. Avec le dataset *Linearly separable* (cas linéaire), aucune différence significative de la précision n'est observée après le passage au SMA. Les modèles linéaires étudiés permettent déjà d'obtenir un score élevé malgré le bruit dans les données, car la frontière entre les deux nuages de points est linéaire.
Pour les deux autres jeux de données, on observe une amélioration de la précision avec les instances de Smapy. En particulier, on observe une très forte amélioration pour le dataset *Circles*, pour lequel les modèles linéaires seuls donnent un résultat proche du hasard (50%). Cette mauvaise performance de ces modèles s'explique par la nature du dataset dans lequel la frontière entre les deux nuages de points est circulaire, donc très peu approximable par des méthodes linéaires. Cependant, nous voyons que l'intégration de ces modèles dans un SMA permet de palier la non-linéarité du problème initial de recherche d'une frontière quadratique en un problème de coopération localement linéaire.
Les précisions obtenues sur le dataset *Moons* sont légèrement meilleures avec l'approche SMA, mais les modèles linéaires seuls permettent d'obtenir des scores satisfaisants. Les données d'entrée ont en effet un comportement qui se rapproche du cas linéaire présenté plus tôt.

Tableau 6.3 – Comparaison des précisions de classification obtenues pour chaque dataset et pour chaque modèle.

	Modèle seul	**Modèle + SMA**
	Moons	
Logit	0.83	**0.89**
SVM linéaire	0.86	**0.89**
PA-I	0.82	**0.89**
PA-II	0.84	**0.87**
	Circles	
Logit	0.49	**0.83**
SVM linéaire	0.53	**0.83**
PA-I	0.53	**0.83**
PA-II	0.53	**0.83**
	Linearly separable	
Logit	**0.92**	0.91
SVM linéaire	0.90	**0.91**
PA-I	0.89	**0.90**
PA-II	**0.89**	**0.89**

Frontières de décision Afin de mieux comprendre le potentiel de l'approche SMA dans le cas de modèles linéaires, nous représentons les frontières de décision obtenues pour chaque modèle et pour chaque dataset (meilleurs cas des validations croisées) dans la figure 6.1.
Nous constatons que les modèles linéaires seuls donnent des frontières linéaires qui sont adaptées au problème de classification pour les jeux de données *Moons* et *Linearly separable*. Les approches SMA sur ces deux datasets reproduisent d'ailleurs ce comportement linéaire dans les frontières.
En revanche, avec le dataset *Circles*, les modèles linéaires seuls sont incapables de séparer les deux nuages de points enclavés, contrairement à l'approche SMA. De plus, ces nuages étant invariants par rotation autour de leur centre, toute séparation par une frontière linéaire passant par ce centre donne un résultat proche du hasard car les classes sont statistiquement équiréparties de chaque côté.

Les précisions de classification et frontières de décision permettent d'appréhender le potentiel d'une intégration de classifieurs linéaires dans des agents Contexte de Smapy. Dans la prochaine section, les résultats obtenus sont discutés.

FIGURE 6.1 – Représentations graphiques des frontières de décision obtenues pour chaque dataset (lignes) et pour chaque modèle linéaire seul ou dans une instance de Smapy (colonnes). La précision de classification obtenue est indiquée en bas à droite pour chaque cas.

6.4 Discussion

En passant par une approche SMA, le problème de classification initial est résolu localement à l'échelle des agents Contexte. Ainsi, même si les agents possèdent des modèles internes ne pouvant générer que des frontières linéaires, ils se sont positionnés et dimensionnés entre-eux de façon à approximer localement une frontière non linéaire grâce aux différents mécanismes de coopération présentés dans le chapitre 5.

Nous pouvons cependant nous demander si les agents Contexte ne se sont pas sur-spécialisés à l'échelle locale en observant des groupes homogènes d'individus (au sens de la classe). L'existence du mécanisme d'exclusion de point, bien qu'il soit souvent sélectionné par validation croisée, a tendance à renforcer cette sur-spécialisation des agents en excluant les points de nouvelle classe de leurs zones d'activation.

Cependant, le comportement idéal recherché pour Smapy est de construire des agents Contexte qui couvrent des zones homogènes de l'espace des variables d'entrées exploré, notamment pour des raisons d'explicabilité. Il y a donc un équilibre à trouver entre interprétabilité géométrique de la disposition des agents Contexte et généralisation du système à des problèmes dynamiques dans lequel de nouvelles classes peuvent apparaître.

Chapitre 7

Mise en place et étude d'une chaîne de traitements pour la détection du mode de transport

Le problème de détection du mode de transport (TMD) est un problème de classification supervisée abondamment traité dans la littérature (c.f. chapitre 4). Il s'intègre dans un projet plus vaste d'analyse de la mobilité, pour lequel des pré-requis ont été énoncés dans le chapitre 1. Pour répondre à ces pré-requis, le classifieur Smapy, système multi-agents ensembliste, a été développé pendant le projet de thèse et présenté dans le chapitre 5. L'expérimentation conduite dans le chapitre 6 a montré que Smapy parvient à transformer des problèmes de classification non linéaires en problèmes de coopération entre agents dont les modèles internes sont linéaires.

L'objectif de l'expérimentation présentée dans ce chapitre est de comparer les performances de plusieurs classifieurs, dont Smapy, pour résoudre le problème de détection du mode de transport à partir de données de géolocalisation et d'accéléromètre mesurées sur un smartphone. Cette classification porte sur les points de données sans détection des arrêts lors de la segmentation (c.f. section 4.2.1.1). Une fenêtre temporelle glissante est utilisée pour fusionner les deux sources de données et calculer des indicateurs statistiques sur les attributs [Fou+23].

La section 7.1 présente les données collectées dans le cadre de cette expérimentation et leurs caractéristiques. Le pré-traitement réalisé est détaillé dans la section 7.2. Le protocole d'entraînement de plusieurs classifieurs, dont Smapy, est introduit dans la section 7.3. Les résultats obtenus sur le jeu de données collecté (disponible en *open source* par ailleurs) et sur deux autres jeux de données de référence sont ensuite présentés dans la section 7.4. Enfin, une discussion dans la section 7.5 pose les limites et opportunités de ce type de méthode pour résoudre le problème de TMD.

7.1 Données d'entrée

Cette section introduit le jeu de données collecté pour tester la chaîne de traitement de détection du mode de transport. Ces données ont été collectées grâce à une application smartphone développée dans le cadre du projet de recherche en mobilité Vilagil [IRI19]. Cette application collecte passivement les positions GPS et les signaux d'accéléromètrie du smartphone. Cette étude porte sur les données collectées par un utilisateur masculin de 25 ans. Cet utilisateur a ensuite ajouté un label correspondant au mode de transport utilisé pour chaque

point d'observation (c.f. section 7.1.2.2). Le smartphone utilisé pour la collecte est un Samsung Galaxy A32 avec le système d'exploitation Android 11. Dans la suite du manuscrit, le jeu de données collecté est appelé OCC-TMD (Occitanie TMD) [Fou+23]. Il est disponiblement en *open source*.

7.1.1 Règles d'acquisition des données

Dans cette partie, les règles d'acquisition de l'application utilisée pour la collecte sont détaillées pour les données de géolocalisation et les données d'accélération.

7.1.1.1 Données de localisation

Tableau 7.1 – Attributs des données de localisation

Attribut	**Description**	**Unité**
timestamp	Date et heure	Heure Unix (ms)
lat	Latitude	Système de coordonnées WGS 1984
lon	Longitude	Système de coordonnées WGS 1984
location_accuracy	Précision des coordonnées	m (mètres)
location_speed	Vitesse (depuis le point précédent)	$m.s^{-1}$ (mètres par seconde)
location_speed_accuracy	Précision de la vitesse	$m.s^{-1}$ (mètres par seconde)
location_heading	Relèvement (par rapport au Nord)	degrés [0,360)

Les attributs collectés par le module GPS du smartphone sont les coordonnées de l'utilisateur (lat, lon) et leur précision (location_accuracy), l'instant d'acquisition (timestamp), la vitesse (location_speed) et sa précision (location_speed_accuracy) et le relèvement (location_heading). Les unités de chaque attribut sont détaillées dans le tableau 7.1.

Les données de localisation de l'utilisateur sont collectées lorsque les deux conditions ci-dessous sont réunies :

— L'utilisateur s'est déplacé d'au moins 50 mètres depuis le point précédent.

— Il s'est écoulé au moins 10 secondes depuis le point précédent.

En d'autres termes, deux points consécutifs ne peuvent être à moins de 10 secondes d'intervalle, ou à moins de 50 mètres de distance. Ces critères ont pour but d'économiser la batterie du smartphone en évitant l'acquisition de points jugés inutiles (i.e. lors des moments d'immobilité). Les seuils de temps et de distance ont été fixés de manière empirique en fonction du retour des utilisateurs.

7.1.1.2 Données d'accéléromètrie

Tableau 7.2 – Attributs des données d'accéléromètrie

Attribut	Description	Unité
timestamp	Date et heure	Heure Unix (ms)
x	Accélération sur l'axe x corrigée de Gx	$m.s^{-2}$ (mètre par seconde au carré)
y	Accélération sur l'axe y corrigée de Gy	$m.s^{-2}$ (mètre par seconde au carré)
z	Accélération sur l'axe z corrigée de Gz	$m.s^{-2}$ (mètre par seconde au carré)

Le module d'accéléromètre du smartphone renvoie l'accélération selon les trois axes (x, y et z) corrigée par rapport à la gravité, ainsi que l'instant d'acquisition (timestamp). Les unités sont détaillées dans le tableau 7.2.

Comme pour la localisation, les données d'accéléromètre de l'utilisateur sont collectées lorsque :

— L'accélération détectée a un écart d'au moins 1 $m.s^{-2}$ avec le point précédent.

— Il s'est écoulé au moins 1 seconde depuis le point précédent.

Le signal d'accéléromètre d'un utilisateur est donc composé de points distants d'au moins une seconde. Le seuil d'accélération porte sur l'écart d'accélération entre deux points consécutifs, et non sur l'accélération elle-même. Ainsi, dans le cas théorique où l'accélération est constante, aucun nouveau point ne sera enregistré malgré la variation de vitesse potentielle de l'utilisateur. Comme dans le cas de la localisation, ces seuils ont été fixés empiriquement à partir des retours d'expérience des utilisateurs.

7.1.2 Collecte des données

Cette partie présente le déroulement de la collecte de données. Plus particulièrement, la période d'acquisition et la méthode de labélisation sont détaillées.

7.1.2.1 Période d'acquisition

Les données du dataset ont été collectées par un seul utilisateur de manière discontinue du 26 juillet 2022 au 10 août 2022. Cet utilisateur s'est déplacé dans la région Occitanie dans le sud de la France (entre Toulouse et Montpellier), en notant pour chaque déplacement l'heure de début, l'heure de fin et le mode de transport utilisé.

Hypothèse 1 (Continuité des données)**.** *Soit une période d'observation* $\mathcal{T}_k = [t_k^0, t_k^1]$*. A tout instant* $t \in \mathcal{T}_k$*, les déplacements de l'utilisateur sont collectés, i.e.*

FIGURE 7.1 – Distribution temporelle des points en fonction des modes de transport.

il porte en permanence sur lui un smartphone allumé avec l'application en état de fonctionnement.

Pour satisfaire l'hypothèse 1, les déplacements de l'utilisateur ont été tracés sur 4 périodes d'observation $\mathcal{T}_1$ à $\mathcal{T}_4$ de manière continue. Au total, le dataset contient **73 heures et 37 minutes** de déplacements, dont 44 heures et 43 minutes pour la période $\mathcal{T}_1$, 26h et 6 minutes pour la période $\mathcal{T}_2$, 1 heure et 47 minutes pour la période $\mathcal{T}_3$ et 1 heure pour la période $\mathcal{T}_4$.

TABLEAU 7.3 – Effectifs des points par période d'observation et par mode.

Obs. phase	still	walk	bike	car	bus	metro	train	Total
$\mathcal{T}_1$	3572	6169	1037	0	1181	947	0	12906
$\mathcal{T}_2$	4892	10944	385	3789	0	0	343	20353
$\mathcal{T}_3$	694	92	0	206	0	0	877	1869
$\mathcal{T}_4$	0	858	0	0	527	239	0	1624
Total	9158	18063	1422	3995	1708	1186	1220	**36752**

7.1.2.2 Labélisation

Dans les jours suivant chaque période d'observation, l'utilisateur a passé en revue l'historique de ses coordonnées spatiales sur la période à l'aide d'un logiciel SIG. Chaque point de coordonnées a été associé à un des 7 modes de transports suivants : *still* (immobilité), *walk* (marche), *bike* (vélo), *car* (voiture), *bus*, *metro* et *train*.

Hypothèse 2 (Intervalle de labélisation)**.** *Soient deux points de localisation consécutifs p_i et p_{i+1} aux instants t_i et t_{i+1} et modes m_i et m_{i+1} respectifs. Tout point d'observation de l'intervalle $(t_i, t_{i+1}]$ a pour mode m_{i+1}.*

Dans le cas où le mode change entre deux points de localisation consécutifs, le label d'un point correspond au mode utilisé entre le point précédent **exclu** et lui-même **inclus** selon l'hypothèse 2. Les labels associés aux points de localisation sont ensuite propagés aux points d'accéléromètre en utilisant le champ temporel associé. La figure 7.2 montre la répartition des points d'observation (localisations + accéléromètre) par mode de transport. Les nombres de points observés par période et par label sont présentés dans le tableau 7.3.

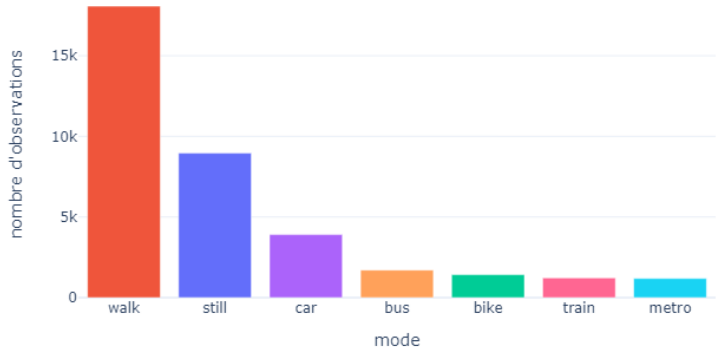


FIGURE 7.2 – Diagramme en barres du nombre de points par mode de transport.

Les règles d'acquisition et les hypothèses présentées dans cette section constituent une contribution à la mise en place d'un système de détection du mode de transport à partir d'un programme de collecte de données sur des smartphones. La prochaine section détaille le pré-traitement des données GPS et accéléromètre collectées sur les utilisateurs. Les différentes étapes sont illustrées à l'aide du dataset introduit précédemment.

7.2 Pré-traitement

Cette section présente la chaîne de pré-traitement des données permettant d'extraire des features statistiques des données brutes collectées par l'utilisateur. Les données brutes contiennent 40830 points d'accéléromètre et 1168 points de localisation sur l'ensemble des 4 périodes d'observation, soit un total de **41998 points d'observation** de l'utilisateur.

7.2.1 Filtrage des données

Les données de localisation sont fournies avec un champ de précision (`locationAccuracy`). Afin d'éliminer les observations de localisation aberrantes,

les points dont la précision de localisation est strictement supérieure à 100 mètres sont arbitrairement supprimés. Le nombre de points d'accéléromètre reste inchangé, mais 18 points de localisation sont éliminés. Après l'étape de filtrage des données, il reste au total **41980 points d'observation** de l'utilisateur.

7.2.2 Ajout des labels

Les données sont collectées sur des périodes d'observation $(\mathcal{T}_k)_{k\in[\![1,4]\!]}$. Les labels ont été définis par l'utilisateur à partir des points de localisation. Une période d'observation $\mathcal{T}_k = [t_k^0, t_k^1]$ est divisée par l'utilisateur en segments $(t_i, t_{i+1}]$ caractérisés par un point de localisation de début p_i, un point de localisation de fin p_{i+1} et un mode m_{i+1} (c.f. hypothèse 2). Chaque point de localisation est ainsi labélisé avec le mode du segment auquel il appartient, à l'exception de **14 points** qui ne sont inclus dans aucun segment (ils sont en dehors de toute période d'observation). En revanche, les points d'accéléromètre a_i ont une fréquence d'acquisition différente (et largement supérieure) à celle des points de localisation. Ils ne sont donc pas nécessairement inclus dans les périodes d'observation, et donc dans les segments $(t_i, t_{i+1}]$ labélisés par l'utilisateur. Le tableau 7.4 illustre la façon dont les points d'accélération n'appartenant à aucun segment sont enlevés du dataset. **5153 points d'accéléromètre** ne sont détectés dans aucun segment. A l'issue de cette étape, il reste donc **36813 points d'observation** de l'utilisateur.

Tableau 7.4 – Exemple de propagation des labels aux points d'accéléromètre. Seuls les points a_3, a_4,a_7, a_8, a_9 et a_{10} sont inclus dans une période d'observation et sont associés à un label. Les autres points d'accélération n'ont pas de mode connu et sont enlevés du dataset.

a_1	a_2	p_1^0	a_3	a_4	p_1^1	a_5	p_2^0	p_2^1	a_6	p_3^0	a_7	a_8	a_9	p_i	p_3^1	p_4^0	a_{10}	p_4^1	a_{11}
?	?	$\mathcal{T}_1$				?	$\mathcal{T}_2$		?	$\mathcal{T}_3$						$\mathcal{T}_4$			?

7.2.3 Calcul d'attributs

En plus des attributs des points de localisation et d'accéléromètre, 4 attributs supplémentaires (écart temporel et distance pour les points de localisation, écart temporel et magnitude pour l'accéléromètre) sont calculés. La magnitude de l'accéléromètre est calculée à partir de la formule suivante :

$$\text{accelerometer\_magnitude} = \sqrt{x^2 + y^2 + z^2} \tag{7.1}$$

où x, y et z désignent les valeurs d'accélération corrigées selon les trois axes (c.f. tableau 7.2).

Les descriptions et unités des attributs additionnels sont présentées dans le tableau 7.5.

Les attributs d'écart temporel ou de distance imposent de réaliser les calculs séparément pour chaque période d'observation pour respecter l'hypothèse 1 de

Tableau 7.5 – Attributs additionnels calculés sur les points de localisation (2 premières lignes) et sur les points d'accéléromètre (2 dernières lignes).

Attribut	Description	Unité
location_time_delta	Ecart temporel avec le point précédent	s (secondes)
location_distance_delta	Distance (à vol d'oiseau) au point précédent	m (mètres)
accelerometer_time_delta	Ecart temporel avec le point précédent	s (secondes)
accelerometer_magnitude	Magnitude de l'accéléromètre	$m.s^{-2}$

continuité des données. Une valeur d'écart temporel ou de distance entre le dernier point d'une période et le premier de la période suivante n'a pas de sens car l'utilisateur a potentiellement réalisé de nombreux autres déplacements dont il n'existe aucune observation. Ainsi, pour chaque période d'observation, le premier point de localisation et le premier point d'accéléromètre sont enlevés du dataset car on ne peut pas calculer d'attribut différentiel par rapport au point précédent (soit **8 points enlevés** au total). A l'issue du calcul des attributs additionnels, le dataset contient donc **36805 points d'observation** (soit 90.1% du dataset original).

7.2.4 Fusion des données et calcul de features

A ce stade, le dataset contient des points de localisation et des points d'accéléromètre avec des attributs différents. Afin d'entraîner des classifieurs, il est nécessaire de fusionner ces deux sources de données afin de calculer les mêmes features statistiques sur l'ensemble des points. Pour chaque période d'observation, l'étape de fusion des données et l'étape de calcul de features sont réalisées simultanément grâce à une fenêtre glissante selon le processus suivant :

1. Concaténation des n_p^k points de localisation $(p_i)_{i\in[\![1,n_p^k]\!]}$ et des n_a^k points d'accélération $(a_i)_{i\in[\![1,n_a^k]\!]}$ de la période d'observation $\mathcal{T}_k$ en une unique série temporelle ordonnée de n^k points d'observations aux instants $(t_i)_{i\in[\![1,n^k]\!]}$.
2. Parcours de l'ensemble des données avec des fenêtres temporelles $T_i = (t_j)_{j\in[\![1,n^k]\!],|t_i-t_j|\leq\frac{T}{2}}$ de durée T centrées autour des points d'observation aux instants t_i.
3. Pour chaque point d'observation d'instant t_i, calcul des p features statistiques $(f_j(T_i))_{j\in[\![1,p]\!]}$ sur la fenêtre temporelle T_i.
4. Complétion des données manquantes par remplissage arrière (i.e. valeur du point suivant).

L'utilisation de fenêtres temporelles dans l'étape 2 a pour but de rassembler les points d'observation dans un même voisinage temporel pour obtenir des informations de localisation et d'accéléromètre. Les features statistiques sont ensuite calculées sur ce voisinage temporel. La durée T du voisinage temporel doit être suffisamment grande pour capter la variation de l'accélération (dont la fréquence minimale est de 1 seconde) et des informations de localisation, tout en étant plus courte que la durée d'un déplacement type pour éviter la confusion

entre les modes dans le calcul statistique des features. Une durée de 60 secondes semble respecter ces critères. Avec une telle durée, 14 points d'observation n'ont pas d'information d'accéléromètre dans leur fenêtre temporelle (i.e. pas de features statistiques issues des attributs d'accéléromètre), et 12367 points d'observation n'ont pas d'information de localisation, soit 33.6% du total des points.

Les features calculées à l'étape 3 sont les cinq indicateurs statistiques `min` (minimum), `max` (maximum), `mean` (moyenne), `median` (médiane) et `std` (écart-type) calculés sur quatre attributs de localisation de base (`speed` (vitesse), `speedAccuracy` (précision de la vitesse), `locationAccuracy` (précision de la localisation) et `heading` (relèvement)) et les 4 attributs additionnels décrits dans la section 7.2.3. Au total, **40 features statistiques** sont calculées pour chaque point d'observation.

Afin de gérer les données manquantes à l'étape 4, un remplissage arrière (*backward fill*) est réalisé, c'est-à-dire que pour chaque attribut manquant la première valeur trouvée dans les points suivants est propagée. Le choix du sens de remplissage provient de l'hypothèse 2 qui implique que le mode à un instant t non observé est le mode du point observé à l'instant $t_i = \min \{t_i \mid i \in [\![1, n]\!], t < t_i\}$, soit le premier point observé après l'instant t.

A l'issue de ce processus, les points d'observation de chaque période sont rassemblés. Par construction, chaque période d'observation a comme dernier élément un point de localisation (c.f. section 7.2.2). Dans le cas où ce point n'a aucun point d'accéléromètre dans son voisinage temporel (et donc des valeurs manquantes pour les features issues des informations d'accéléromètre), il est enlevé du dataset. Dans les données collectées par l'utilisateur, ce cas de figure ne s'est présenté pour aucune des 4 périodes d'observation. En conséquent, le dataset final est composé de $n = 36805$ **lignes (points d'observation), et** $p = 40$ **colonnes (features statistiques)** auxquelles s'ajoutent le mode de transport et la période d'observation.

Le processus de pré-traitement présenté dans cette section intègre une dimension temporelle à la construction des variables d'apprentissage d'un modèle de classification. D'une part, un attribut d'écart temporel entre points successifs est calculé pour chaque signal de donnée, et d'autre part la fusion des données et le calcul de features statistiques sur l'ensemble des attributs est réalisé à l'aide d'une fenêtre temporelle glissante. La section suivante présente la classification du mode de transport à partir des features calculées sur le dataset collecté et sur deux autres datasets issus de la littérature.

7.3 Apprentissage

Nous proposons de comparer les performances de plusieurs algorithmes de *Machine Learning* issus de la littérature d'une part, et de Smapy ([Fou+22b], [Fou+22a] et chapitre 5) d'autre part. Dans la suite de cette section, l'expérience

est reproduite sur deux datasets publics : Microsoft GeoLife [Zhe+09] pour les données GPS, et US-TMD [Car+18] pour les données d'accéléromètre.

7.3.1 Dataset collecté

Le dataset collecté combine des informations de localisation et d'accéléromètrie. Afin d'étudier la contribution de chaque type d'information à la détection du mode de transport, il est nécessaire de définir plusieurs sous-ensembles du dataset pour les expérimentations. Le dataset complet est noté D_{full}. Les datasets avec les features issues des données de localisation et d'accéléromètre sont notés respectivement D_{location} et $D_{\text{accelerometer}}$. De plus, afin de comparer les résultats avec ceux issus du dataset GeoLife (c.f. section 7.3.2), le dataset D'_{location} contenant les features issues de la localisation sans les informations de précision sur la position ou la vitesse est défini. Les variables d'entrée du problème de classification du mode de transport pour chaque dataset sont présentées dans le tableau 7.6.

Tableau 7.6 – Sous-ensembles de features des différentes variantes du dataset. 5 features sont calculées à partir de chaque attribut (moyenne, minimum, maximum, écart-type et médiane).

Attribut	D_{full}	D_{location}	D'_{location}	$D_{\text{accelerometer}}$
location_accuracy	✓	✓		
location_speed	✓	✓	✓	
location_speed_accuracy	✓	✓		
location_heading	✓	✓	✓	
location_distance_delta	✓	✓	✓	
location_time_delta	✓	✓	✓	
accelerometer_magnitude	✓			✓
accelerometer_time_delta	✓			✓
Nombre de features	**40**	**30**	**20**	**10**

7.3.1.1 Comparaison d'approches de *Machine Learning*

Sur chaque dataset, les performances de 4 classifieurs issus du *Machine Learning* sont comparées pour la résolution du problème de classification du mode de transport :

— KNN : k plus proches voisins [CH67]

— RF : Forêt aléatoire [Bre01]

— ANN : Réseau de neurones artificiel [RHW86]

— SVM : Machine à vecteurs de support [CST+00]

Les implémentations utilisées proviennent de la librairie python *scikit-learn* [Ped+11]. Pour chaque dataset, les paramètres de chaque classifieur sont choisis

par validation croisée à 5 blocs à partir des grilles de paramètres présentées dans le tableau 7.7.

TABLEAU 7.7 – Grilles de valeurs pour l'optimisation des paramètres par validation croisée. Les autres paramètres gardent leurs valeurs par défaut dans l'implémentation de *scikit-learn*.

Paramètre	**Grille de valeurs**		
	KNN		
`n_neighbors`	5	10	15
`weights`	uniform	distance	
	RF		
`n_estimators`	100	200	300
	ANN		
`hidden_layer_sizes`	100	200	500
`alpha`	1e-4	1e-3	1e-2
`learning_rate_init`	1e-3	1e-2	
	SVM		
`kernel`	rbf	poly	
`C`	1	50	100

7.3.1.2 Smapy

Le classifieur Smapy, qui est à l'intersection entre système multi-agents coopératif et modèle d'apprentissage ensembliste, est également testé. Les paramètres utilisés sont présentés dans le tableau 7.8. Le modèle interne retenu est un SVM avec un noyau linéaire (i.e. un classifieur linéaire). L'entraînement

TABLEAU 7.8 – Valeurs des paramètres de Smapy utilisées pour la classification du mode de transport.

Paramètre	**Valeur**
R	0.2
M	10^{-6}
O	10%
E	Oui
N_c	Sigmoïde
α	5%
F_+	1
F_-	0.5
Z	Aucun
Modèle interne	SVM linéaire

s'effectue sur 80% des données choisies aléatoirement, et l'évaluation des performances sur les 20% de données restantes.

7.3.2 Dataset GeoLife

GeoLife [Zhe+09] est un dataset créé par Microsoft Research Asia entre avril 2007 et août 2012. Il compile les trajectoires GPS de près de 180 utilisateurs sur un total de plus de 48000 heures, collectées à l'aide de traceurs GPS ou de téléphones équipés d'un GPS. Chaque point du dataset est défini par des coordonnées GPS (lat, lon, alt) et un mode de transport (marche, vélo, bus, voiture, métro, train, avion, bateau, course, moto ou taxi).

Afin de comparer GeoLife avec le dataset collecté, il faut lui appliquer le pré-traitement présenté en section 7.2 :

1. Filtrage des données non labélisées ou avec des modes absents du dataset collecté (avion, bateau, course, moto et taxi).
2. Calcul des attributs de localisation additionnels.
3. Calcul de features statistiques sur les attributs (moyenne, minimum, maximum, écart-type et médiane) avec des fenêtres glissantes de durée $T = 60$ secondes.

A l'issue du filtrage des données à l'étape 1, le dataset contient 5.17 millions de points d'observation. Contrairement au dataset collecté, le mode *still* (absence de déplacement) est absent.

En plus des attributs additionnels `location_time_delta` et `location_distance_delta`, on calcule dans l'étape 2 une estimation des attributs `location_speed` et `location_heading` entre deux points consécutifs car le dataset GeoLife ne contient pas d'information de vitesse ou de relèvement. Le calcul des attributs doit être fait pour chaque période d'observation. Les données GeoLife sont segmentées par période d'observation à chaque changement d'utilisateur ou à chaque écart temporel de plus de 24 heures. 890 périodes d'observation sont obtenues, avec pour effectif moyen 5809 points d'observation et pour effectif médian 1504 points d'observation. Après le calcul des attributs, les points d'observation dont la vitesse estimée est supérieure à 400km/h sont filtrés.

Les features issues des attributs de localisation sont calculées de la même manière que pour les données du dataset collecté lors de l'étape 3. Le dataset (noté D_{geolife}) contient 4.7 millions de points d'observation à l'issue du pré-traitement. Les deux modes les plus fréquemment observés sont la marche (*walk*) et le bus (*bus*). La répartition des autres modes est représentée dans la figure 7.3.

En raison d'un nombre de points beaucoup plus important que dans le dataset collecté, un sous-ensemble $D^{\text{test}}_{\text{geolife}}$ de 40000 points est sélectionné aléatoirement pour l'apprentissage des classifieurs. Parmi ces points d'observation, 80% sont utilisés pour l'entraînement et 20% pour le test des modèles. La précision de classification de chaque modèle introduit dans la section 7.3.1.1 est ensuite étudiée sur l'ensemble du dataset. Pour chacun des algorithmes, la

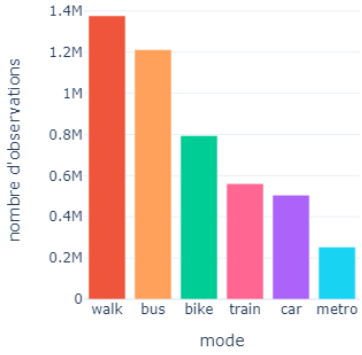


FIGURE 7.3 – Diagramme en barres du nombre de points par mode de transport dans le dataset GeoLife à l'issue du pré-traitement.

combinaison de paramètres sélectionnée sur le dataset collecté est utilisée. De plus, afin d'évaluer l'apport des attributs différentiels (`location_time_delta` et `location_distance_delta`), les sous-ensembles D'_{geolife} et $D'^{\text{test}}_{\text{geolife}}$ privés des features calculées à partir de ces deux attributs sont définis.

7.3.3 Dataset US-Transportation Mode

US-Transportation Mode (US-TMD) [Car+18] est un dataset collecté par l'Université de Bologne à l'aide d'une application smartphone de traçage utilisée par 13 volontaires. Chaque utilisateur a lancé plusieurs acquisitions correspondant à certains de ses déplacements. Les points collectés sont ensuite labélisés avec le mode de transport utilisé durant cette période (absence de déplacement, marche, voiture, bus ou train).

A partir des données brutes, le même pré-traitement que pour le dataset collecté est appliqué :

1. Filtrage des données par validité et par type de capteur.
2. Calcul des attributs d'accéléromètre (magnitude et écarts temporels) sur chaque période d'observation.
3. Calcul de features statistiques sur les attributs (moyenne, minimum, maximum, écart-type et médiane) avec des fenêtres glissantes de durée $T = 60$ secondes pour chaque période d'observation.

Les données brutes contiennent des mesures acquises par différents capteurs des smartphones des utilisateurs. Dans le cadre de cette expérimentation, seuls les capteurs d'accéléromètre et de vitesse sont retenus afin de tester la qualité de la chaîne de traitement mise en place (les positions GPS ne sont pas fournies). Cependant, en raison du très faible nombre de valeurs de vitesse disponibles,

seules les informations issues de l'accéléromètre sont gardées. Après sélection des points d'observation issus de l'accéléromètre et suppression des données manquantes ou invalides, le dataset contient 1.4 millions de points.

Les périodes d'observation sont définies de la même manière que pour le dataset GeoLife (c.f. section 7.3.2), avec en plus une segmentation entre les différents fichiers d'acquisition d'un même utilisateur. De ce fait, chaque période d'observation possède un unique mode de transport.

A l'issue du calcul des features, le dataset (noté D_{ustmd}) contient 1.4 millions de points. La répartition des modes de transport est illustrée dans la figure 7.4. Contrairement au dataset collecté, le mode *metro* est absent.

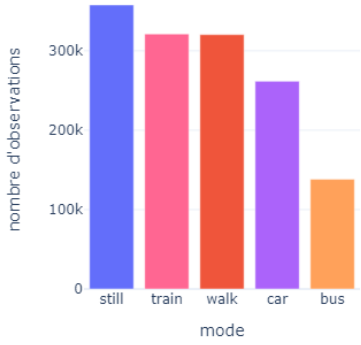


FIGURE 7.4 – Diagramme en barres du nombre de points par mode de transport dans le dataset US-TMD à l'issue du pré-traitement.

Comme avec le dataset GeoLife, un sous-ensemble $D^{\text{test}}_{\text{ustmd}}$ de 40000 points (dont 80% pour l'apprentissage et 20% pour le test) est sélectionné aléatoirement pour entraîner les modèles avant d'étudier leurs performances sur le dataset complet. Les modèles sont entraînés avec les paramètres sélectionnés sur le dataset collecté. On définit également les sous-ensembles D'_{ustmd} et $D'^{\text{test}}_{\text{ustmd}}$ privés des features issues de l'attribut `accelerometer_time_delta` afin d'en étudier la pertinence.

Le protocole d'expérimentation, basé sur une hyperparamétrisation des classifieurs à comparer, vise à étudier les performances de la chaîne de traitements sur plusieurs sous-ensembles du dataset collecté et de deux autres datasets de la littérature. Dans la prochaine section, les résultats obtenus sur les jeux de test sont présentés.

7.4 Résultats

Cette section présente les résultats obtenus à l'issue de la classification supervisée du mode de transport avec les classifieurs présentés dans la section 7.3 pour chacun des trois jeux de données. Une analyse quantitative et qualitative de la pertinence de l'utilisation des écarts temporels est également réalisée.

7.4.1 Dataset collecté

Tableau 7.9 – Comparaison des précisions de classification. Pour les 4 premiers classifieurs, la meilleure précision sur la validation croisée présentée dans la section 7.3.1.1 a été retenue. L'instance de Smapy est entraînée sur 80% des données et la précision retenue est évaluée sur les 20% restants.

	D_{full}	D_{location}	D'_{location}	$D_{\text{accelerometer}}$
RF	**99.57%**	**97.06%**	**97.02%**	**98.65%**
SVM	96.57%	94.36%	88.91%	74.49%
ANN	98.14%	95.24%	90.41%	85.33%
KNN	99.01%	**96.44%**	96.41%	**94.79%**
Smapy	99.25%	96.67%	96.67%	92.92%

Le tableau 7.9 présente les précisions de classification obtenues pour le dataset collecté selon la formule suivante :

$$\text{précision} = \frac{\text{nombre de prédictions correctes}}{\text{nombre de prédictions}} \tag{7.2}$$

Le classifieur Random Forest obtient systématiquement la meilleure précision, quel que soit le sous-ensemble du dataset utilisé. Lorsque toutes les features sont utilisées, l'algorithme discrimine le mode de transport avec 99.57% de précision. Les performances de l'approche multi-agents Smapy sont toutefois similaires. L'utilisation des SVM donne les performances les plus basses, en particulier lorsque l'on considère uniquement les features issues des données d'accéléromètre (74.49% de précision).

Pour chaque classifieur, les meilleures performances sont atteintes avec le dataset complet D_{full}. Les données d'accéléromètre semblent moins discriminantes que les données GPS car la précision obtenue avec le dataset $D_{\text{accelerometer}}$ est moins élevée que celle obtenue avec D_{location} (sauf avec Random Forest). Parmi les données GPS, la prise en compte des attributs de précision dans D_{location} augmente systématiquement la précision obtenue avec D'_{location}. Pour tous les classifieurs, l'utilisation conjointe des données GPS et accéléromètre permet d'obtenir la meilleure précision.

La matrice de confusion présentée dans la figure 7.5 montre des erreurs très faibles pour chaque mode prédit. Les deux modes ayant le plus de confusion (0.6%) sont l'immobilité (*still*) et le métro (*metro*). Ces deux modes sont caractérisés

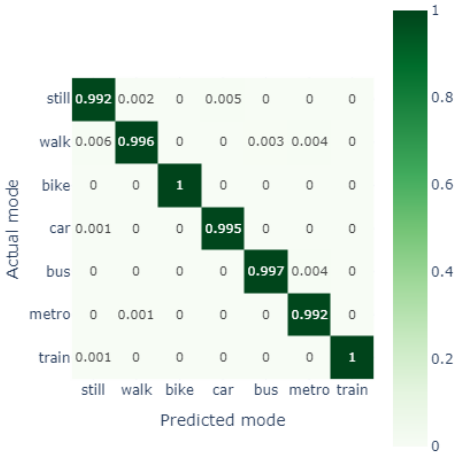


FIGURE 7.5 – Matrice de confusion normalisée avec le classifieur Random Forest sur le dataset D_{full}.

par un faible nombre d'observations, par règle d'acquisition pour le mode *still* et par perte de signal pour le mode *metro*.

L'importance des features est évaluée avec la mesure d'impureté de Gini calculée lors de l'apprentissage de l'instance de Random Forest ayant donné la meilleure précision. L'objectif est d'évaluer la pertinence de la chaîne de prétraitement, et en particulier le calcul des attributs d'écarts temporels. Les valeurs d'importance sont illustrées dans la figure 7.6. La feature la plus importante est `std_location_distance_delta`, c'est-à-dire l'écart-type de la distance entre deux points GPS successifs. Sur l'ensemble des features, celles issues des écarts temporels ont une importance moyenne plus élevée que les autres (0.0296 contre 0.0235). Les features issues de l'accéléromètre ont une importance moyenne légèrement supérieure à celle des autres (0.0258 contre 0.0247), ce qui explique les performances plus élevées du classifieur Random Forest sur le sous-ensemble $D_{\text{accelerometer}}$.

Les deux features suivantes dans l'ordre d'importance sont issues des attributs d'écarts temporels dans les données d'accéléromètre et GPS. Afin de de visualiser la capacité de discrimination de ces deux features, leurs distributions pour chaque mode sont présentées dans la figure 7.7 sous forme de boxplot. Il apparaît que les écarts temporels dans les données d'accéléromètre permettent de discriminer facilement les modes *train* et *bus*, tandis que ceux issus des données GPS permettent de séparer les modes *train*, *metro* et *still* des autres mode de manière significative. Cela s'explique par la différence de fréquence d'observation pour ces modes en particulier, en raison des règles d'acquisition du dataset pour les modes *still* (fréquence très faible) et *bus* (fréquence très élevée), et de la perte de signal GPS pour les modes *metro* et *train*.

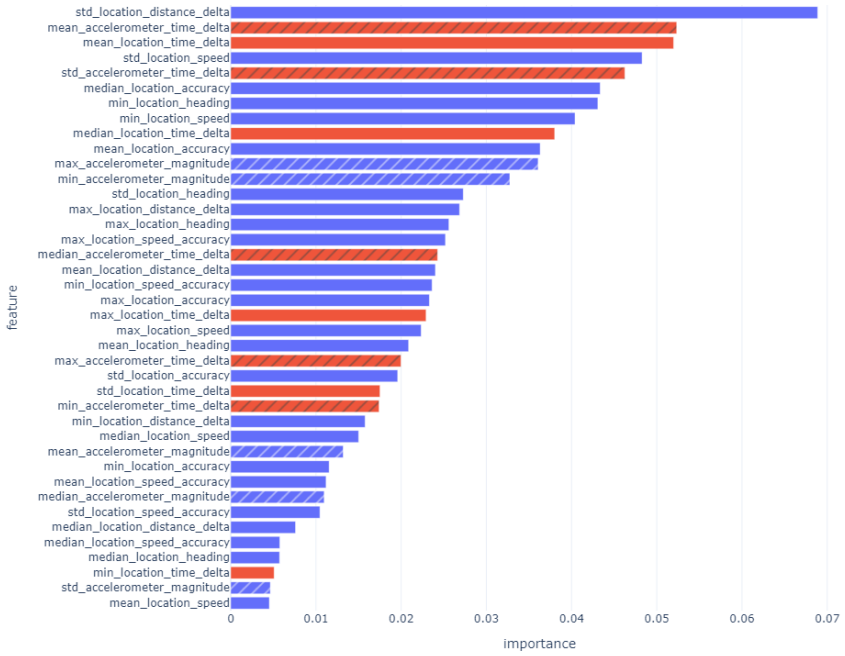


FIGURE 7.6 – Importances de Gini pour le classifieur Random Forest entraîné avec la meilleure combinaison de paramètres (sélectionnée par validation croisée). Les features d'écarts temporels sont représentées en rouge. Les features issues des données d'accéléromètre sont représentées hachurées.

7.4.2 Dataset GeoLife

Les précisions obtenues avec le dataset GeoLife et présentées dans le tableau 7.10 sont maximales avec le classifieur Random Forest dans la majorité des cas (jusqu'à 77.39% avec le dataset de test complet $D^{\text{test}}_{\text{geolife}}$) et minimales avec le classifieur Smapy (63.29% avec les mêmes données). Les performances sur les sous-échantillons de test sont légèrement plus élevées que sur les datasets complets. L'utilisation des features issues des écarts temporels dans les sous-ensembles $D^{\text{test}}_{\text{geolife}}$ et D_{geolife} donne systématiquement des meilleures performances que dans les sous-ensembles $D'^{\text{test}}_{\text{geolife}}$ et D'_{geolife} qui ne les contiennent pas.

La matrice de confusion obtenue avec le classifieur Random Forest (c.f. figure 7.8) permet d'évaluer les erreurs de prédiction commises pour chaque classe prédite. Ces erreurs sont quasiment toujours inférieures à 10%, sauf dans le cas du mode *walk* (11.5% des prédictions correspondent au mode *bus*) et du mode *car* (10.1% des prédictions correspondent au mode *bus*). La confusion *car*/*bus* peut s'expliquer par des vitesses de circulation similaires, tandis que les trajets de modes *car* et *walk* sont caractérisés par des moments d'immobilité (le dataset

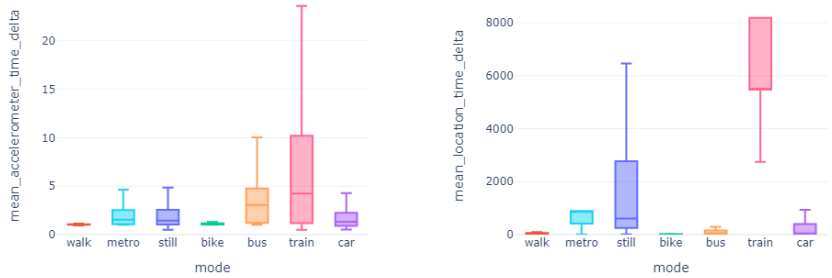


FIGURE 7.7 – Distribution des features `mean_accelerometer_time_delta` (gauche) et `mean_location_time_delta` (droite) en fonction du mode de transport. Les données sont représentées sous forme de boxplot sans les outliers.

TABLEAU 7.10 – Comparaison des précisions de classification sur les différents sous-ensembles du dataset GeoLife. Les précisions des datasets complets D_{geolife} et D'_{geolife} sont obtenues respectivement avec les modèles entraînés sur les sous-ensembles $D^{\text{test}}_{\text{geolife}}$ et $D'^{\text{test}}_{\text{geolife}}$.

	$D^{\text{test}}_{\text{geolife}}$	D_{geolife}	$D'^{\text{test}}_{\text{geolife}}$	D'_{geolife}
RF	**77.39%**	70.42%	**73.08%**	**71.66%**
SVM	71.49%	69.87%	71.19%	69.81%
ANN	73.43%	**71.83%**	71.23%	70.14%
KNN	70.15%	68.87%	69.74%	68.78%
Smapy	63.29%	62.96%	60.59%	60.26%

GeoLife ne possède pas de classe *still*).

7.4.3 Dataset US-TMD

Les résultats obtenus avec le dataset US-TMD confirment ceux obtenus avec le dataset GeoLife. Comme illustré dans le tableau 7.11, Random Forest fait partie des classifieurs les plus performants (aux côtés de KNN), avec une précision maximale de 99.39% atteinte sur l'ensemble du dataset de test. Smapy a des performances beaucoup moins bonnes que les autres classifieurs (71.09% sur l'ensemble des données) qui peut s'expliquer par l'inadéquation des paramètres utilisés avec le dataset US-TMD. Les features d'écarts temporels (dans les sous-ensembles $D^{\text{test}}_{\text{ustmd}}$ et D_{ustmd}) améliorent les précisions obtenues dans presque tous les cas.

Les résultats sont comparés à ceux obtenus dans [Car+18] (auteurs du dataset) et [VGR20]. La précision de classification est fortement améliorée avec l'approche présentée précédemment (de l'ordre de 10% avec Random Forest), même sans

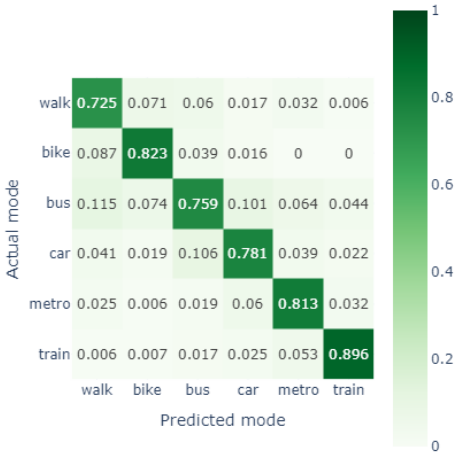


FIGURE 7.8 – Matrice de confusion normalisée avec le classifieur Random Forest sur le dataset $D^{\text{test}}_{\text{geolife}}$.

TABLEAU 7.11 – Comparaison des précisions de classification sur les différents sous-ensembles du dataset US-TMD avec les précisions obtenues dans [Car+18] et [VGR20]. Les précisions des datasets complets D_{ustmd} et D'_{ustmd} sont obtenues respectivement avec les modèles entraînés sur les sous-ensembles $D^{\text{test}}_{\text{ustmd}}$ et $D'^{\text{test}}_{\text{ustmd}}$.

	$D^{\text{test}}_{\text{ustmd}}$	D_{ustmd}	$D'^{\text{test}}_{\text{ustmd}}$	D'_{ustmd}	[Car+18]	[VGR20]
RF	**99.39%**	96.82%	**98.46%**	96.88%	**89.00%**	**85.00%**
SVM	91.70%	91.25%	80.89%	81.56%	86.00%	79.00%
ANN	95.74%	95.04%	91.76%	91.81%	87.00%	75.00%
KNN	98.64%	**98.61%**	98.15%	**97.92%**		80.00%
Smapy	71.09%	71.30%	62.04%	62.60%		

utiliser les écarts temporels. Les paramètres utilisés dans nos modèles sont ceux sélectionnés par validation croisée sur le dataset collecté. Les paramètres utilisés dans les modèles des deux articles ont été optimisés par leurs auteurs respectifs.

La figure 7.9 présente les erreurs commises pour chaque classe prédite. Elles sont toutes inférieures à 1%. La précision obtenue lors de la prédiction du mode *car* est légèrement moins élevée que pour les autres modes (98.3% contre plus de 99.5% pour toutes les autres classes).

Les résultats présentés dans cette section montrent que les règles d'acquisition présentées dans la partie 7.1.1, ainsi que le processus de pré-traitement basé sur l'analyse des écarts temporels présenté dans la section 7.2 sont pertinents pour la détection du mode de transport. Le classifieur Random Forest semble avoir

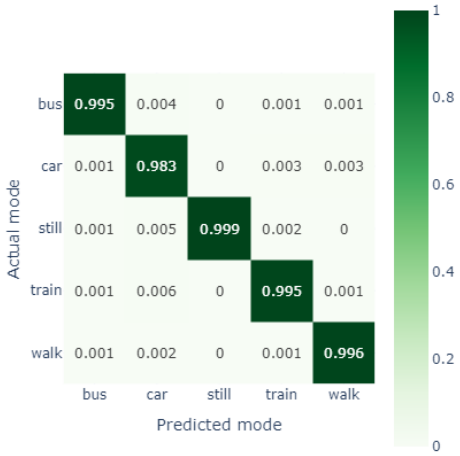


FIGURE 7.9 – Matrice de confusion normalisée avec le classifieur Random Forest sur le dataset $D^{\text{test}}_{\text{ustmd}}$.

de meilleures performances dans la plupart des cas. De plus, la combinaison des données GPS et accéléromètre permet d'améliorer la précision de classification par rapport à l'utilisation d'une seule de ces deux sources de données. La prochaine section est une discussion des résultats obtenus à l'issue de cette expérimentation.

7.5 Discussion

Cette expérimentation illustre une nouvelle méthodologie de classification supervisée du mode de transport à partir de données de géolocalisation et d'accéléromètre. Elle est composée des éléments suivants :

— Un ensemble de règles d'acquisition présentées dans la section 7.1.1.

— Un ensemble d'hypothèses à respecter sur la collecte des données, présentées dans la section 7.1.2.

— Une chaîne de pré-traitement des données explicitée dans la section 7.2, basée notamment sur l'analyse des écarts temporels entre deux points successifs.

Notre méthodologie est illustrée avec un jeu de données collecté dans le cadre d'un projet de recherche sur la mobilité, et avec les jeux de données publics GeoLife et US-TMD. Les différentes étapes des contributions sont détaillées afin d'assurer leur reproductibilité sur d'autres territoires. Le choix de n'utiliser que des données GPS et d'accélérométrie, sans les compléter par des données contextuelles (c.f. section 2.1), est dû à la volonté de rendre notre méthodologie agnostique du territoire considéré (les données contextuelles n'étant pas disponibles sous la même forme sur l'ensemble des territoires) d'une part, et aux bonnes performances rencontrées avec ces deux sources de données utilisateur d'autre part.

Nous obtenons des précisions de classification élevées sur notre dataset et sur US-TMD qui tendent à valider la pertinence de la méthodologie présentée. Les résultats montrent que les règles d'acquisition mises en place ainsi que l'utilisation des écarts temporels améliorent la précision de classification. Cependant, les écarts temporels entre les points ne sont exploitables que lorsque l'hypothèse 1 sur la continuité des données est vérifiée. En effet, si l'acquisition des données est interrompue sur un intervalle temporel, les points précédant et suivant cet intervalle auront un écart temporel similaire aux écarts observés pour des modes tels que le train ou le métro.

Dans le cas du dataset US-TMD, l'utilisation des écarts temporels ne suffit pas à expliquer le gain significatif de performance par rapport aux résultats obtenus par les auteurs. Une étude plus approfondie de l'influence de la méthode de labélisation, de la segmentation des trajets ou encore du calcul des features statistiques à partir des attributs est nécessaire. L'ajout de nouveaux modes de transport tels que le bateau, l'avion ou les vélos et trottinettes électriques pourrait faire évoluer les règles d'acquisition et les features utilisées dans la chaîne de traitement. Concernant les règles d'acquisition introduites, une étude de l'impact de la fréquence de collecte des capteurs sur la précision et la consommation énergétique du smartphone permettrait d'établir un équilibre entre qualité des données et acceptabilité par les usagers.

Enfin, l'utilisation de Smapy illustre le potentiel des approches multi-agents pour résoudre des problèmes d'analyse de la mobilité. Sur le dataset collecté, les performances obtenues avec Smapy sont élevées et comparables à celles des classifieurs de la littérature. Sur les deux autres datasets en revanche, les performances sont inférieures. Cette différence peut être due à une mauvaise combinaison de paramètres pour l'entraînement des instances de Smapy sur les deux datasets de la littérature. Une étude des performances de Smapy avec une hyperparamétrisation par validation croisée permettrait d'explorer le potentiel de Smapy sur ces jeux de données, mais nécessiterait en amont une optimisation du temps de calcul dans l'implémentation. De plus, il serait intéressant d'approfondir l'approche multi-agents pour introduire la notion d'explicabilité dans la prédiction du mode de transport. En effet, les analyses de mobilité ayant un impact potentiel sur les politiques d'urbanisme et de développement, il est nécessaire de pouvoir interpréter et justifier les prédictions obtenues à partir des modèles d'apprentissage automatique utilisés. Ce point est discuté et approfondi dans le chapitre 9.

Le prochain chapitre présente une nouvelle expérimentation de validation de la chaîne de traitement mise en place sur de nouvelles données. Plusieurs scénarios d'apprentissage et des résultats plus approfondis sur les performances de Smapy y sont introduits.

Chapitre 8

Intégration et évaluation de la chaîne de traitements pour la détection du mode de transport dans un contexte industriel

Dans le chapitre 7, une chaîne de traitements basée sur des données de géolocalisation et d'accélération a été introduite pour résoudre le problème de détection du mode de transport (TMD). Plusieurs classifieurs, dont Smapy, ont été testés sur des données collectées à l'aide d'une application développée dans le cadre du projet de recherche Vilagil [IRI19].

Dans ce chapitre, une seconde expérimentation de comparaison de classifieurs sur le problème de TMD est menée. Elle répond à deux objectifs :

— Valider et quantifier les performances de la méthodologie mise en place pour l'expérimentation présentée dans le chapitre 7 dans le contexte industriel de Citec (notamment avec une collecte de données plus importante et contrôlée de bout en bout).

— Explorer plusieurs scénarios d'apprentissage (et notamment de découpage entre données d'entraînement et données de test).

Dans un premier temps, la section 8.1 présente la méthode de collecte de données et le dataset ainsi constitué, en caractérisant les différences avec le dataset OCC-TMD issu de l'application développée dans le cadre du projet Vilagil. Les étapes de pré-traitement sont ensuite rappelées et adaptées aux nouvelles données dans la section 8.2. L'hyperparamétrisation des classifieurs ainsi que les différents scénarios d'apprentissage sont détaillés dans la section 8.3, avant de présenter les résultats obtenus sur les différents scénarios dans la section 8.4. Enfin, ces résultats sont discutés dans la section 8.5.

8.1 Données d'entrée

Cette section présente la méthode de construction du dataset utilisé dans cette expérimentation. Dans un premier temps, le fonctionnement de l'application abitrack utilisée pour la collecte est détaillé. Les caractéristiques des données collectées sont ensuite présentées et discutées.

8.1.1 Règle d'acquisition des données

Abitrack est une application pour smartphone développée par Citec en 2023 afin de permettre à des volontaires de collecter des données utilisateur sous la

TABLEAU 8.1 – Attributs collectés par l'application abitrack

Attribut	Description	Unité
timestamp	Date et heure	Heure Unix (ms)
lat	Latitude	Système de coordonnées WGS 1984
lon	Longitude	Système de coordonnées WGS 1984
location_accuracy	Précision des coordonnées	m (mètres)
speed	Vitesse (depuis le point précédent)	$m.s^{-1}$ (mètres par seconde)
speed_accuracy	Précision de la vitesse	$m.s^{-1}$ (mètres par seconde)
heading	Bearing (par rapport au Nord)	degrés [0,360)
acceleration_x	Accélération sur l'axe x corrigée de Gx	$m.s^{-2}$ (mètre par seconde au carré)
acceleration_y	Accélération sur l'axe y corrigée de Gy	$m.s^{-2}$ (mètre par seconde au carré)
acceleration_z	Accélération sur l'axe z corrigée de Gz	$m.s^{-2}$ (mètre par seconde au carré)

forme de traces contenant points de géolocalisation (grâce au système GNSS) et mesures tri-axiales de l'accélération (grâce aux accéléromètres intégrés dans les smartphones). D'autres attributs comme le relèvement (*heading*), la précision de géolocalisation, la vitesse et la précision de la vitesse sont collectés (c.f. tableau 8.1). Contrairement aux données collectées avec l'application développée dans le cadre du projet Vilagil (c.f. section 7.1.1), les attributs d'accélération et de géolocalisation sont mesurés simultanément à la fréquence fixe de $0.1Hz$ (une mesure toutes les dix secondes).

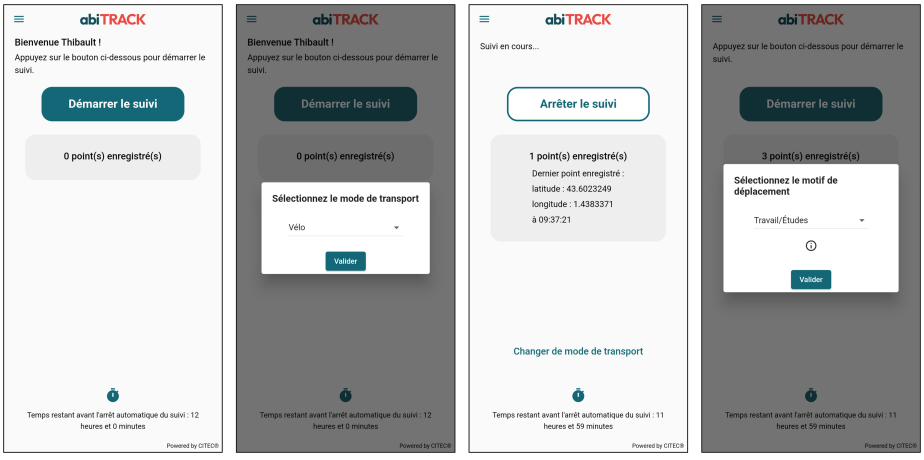


FIGURE 8.1 – Captures d'écran du fonctionnement d'abitrack. De gauche à droite : **(1)** l'écran d'accueil et bouton de démarrage du suivi, **(2)** la sélection du mode de transport au démarrage du suivi, **(3)** l'écran de suivi en cours avec le bouton de changement de mode et **(4)** la sélection du motif d'arrêt lors de l'arrêt du suivi.

De plus, contrairement à l'application du projet Vilagil, abitrack intègre

Tableau 8.2 – Caractéristiques des utilisateurs impliqués dans la collecte de données avec l'application abitrack.

Utilisateur	Genre	Âge	Voiture	Vélo	Abo. TC	Abo. Train	Abo. VLS	Points
A	M	26	✓		✓	✓	✓	2509
B	M			N/A	N/A	N/A	N/A	33
C	M	36		✓		✓	✓	7783
D	M	55	✓	✓				1770
E	M	34		✓		✓	✓	4275

l'étape de labélisation dans la collecte comme illustré dans la figure 8.1. Après s'être connecté avec ses identifiants, l'application invite l'utilisateur à démarrer le suivi de son déplacement **(1)** à condition qu'il ait renseigné le mode de transport qu'il s'apprête à utiliser **(2)**. Durant le suivi, l'utilisateur peut déclarer un changement de mode de transport ou arrêter la collecte lorsqu'il arrive à destination **(3)**. Dans le cas d'un arrêt du suivi, l'utilisateur doit obligatoirement indiquer le motif de son arrêt (i.e. la nature de l'activité réalisée à destination) pour des besoins futurs de classification automatique du motif **(4)**. Un tel fonctionnement implique que l'hypothèse de continuité des données présentée dans la section 7.1.2 n'est plus respectée, car seuls les déplacements sont mesurés. Lors de la déclaration d'un arrêt, un point de mesure est généré avec l'information du motif de l'arrêt, mais aucun autre point ne sera mesuré jusqu'à ce que l'utilisateur décide de redémarrer le suivi. Il peut donc se passer plusieurs jours entre deux déplacements mesurés sans que l'utilisateur ait été immobile durant cette période. Enfin, en raison de l'absence de mesures lors des arrêts, la classe d'immobilité (*still*) n'est pas considérée dans cette expérimentation, bien qu'elle fasse partie des choix possibles dans abitrack pour des raisons de *debug*.

8.1.2 Données collectées

Les données utilisées dans cette expérimentation ont été collectées par 5 ingénieurs de Citec entre septembre 2023 et janvier 2024. Les traces collectées couvrent une zone géographique comprenant la France métropolitaine, la Suisse et l'Angleterre (c.f. figure 8.2). Huit modes de transport sont observés : la marche (*walk*), le vélo (*bike*), le bus, le métro, l'avion (*plane*), la voiture (*car*), le train et le tramway (*tram*).

Les informations compilées dans le tableau 8.2 indiquent que les profils socio-démographiques des volontaires sont assez peu diversifiés (uniquement des hommes occupant un poste de cadre). L'âge moyen des volontaires est de 37,8 ans. Deux d'entre eux possèdent une voiture, tandis que trois d'entre eux possèdent un vélo. Un seul des utilisateurs est abonné à un réseau de transports en communs (TC) durant la période d'étude, trois d'entre-eux possèdent un abonnement ferroviaire et un abonnement à un service de vélos en libre service (VLS).

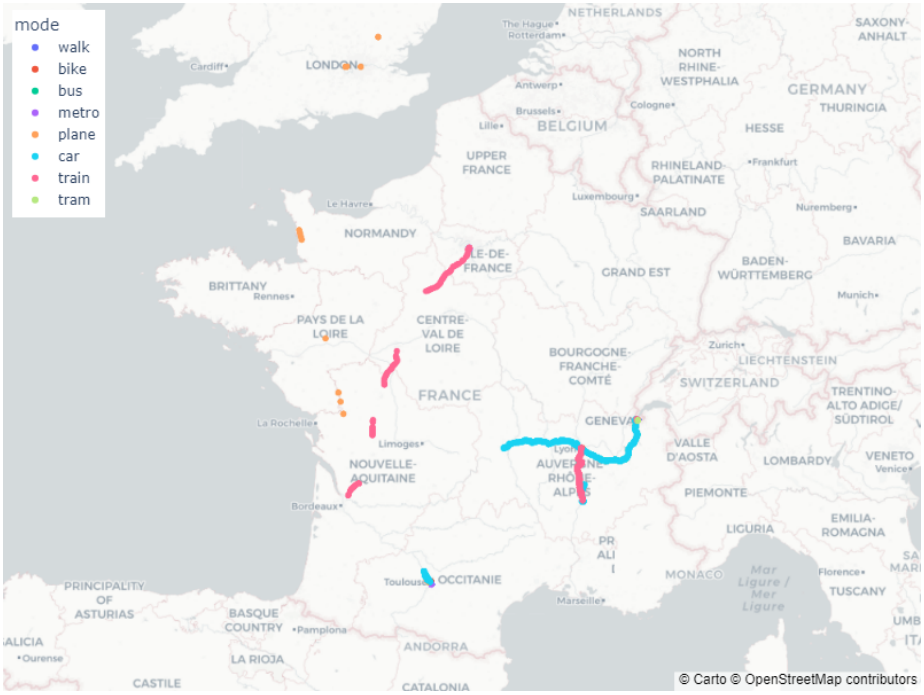


FIGURE 8.2 – Cartographie des traces de géolocalisation collectées avec abitrack entre septembre 2023 et janvier 2024.

L'application abitrack a permis de collecter un nouveau jeu de données selon des règles d'acquisition contrôlées dans l'optique d'une intégration au contexte industriel de Citec. Dans la prochaine section, les étapes de pré-traitement présentées dans le chapitre 7 sont adaptées à ces nouvelles données.

8.2 Pré-traitement

8.2.1 Filtrage des données

Les données brutes collectées avec abitrack contiennent 20153 points. Quatre étapes de filtrage sont mises en place :

— Suppression des valeurs invalides (e.g. vitesses et précisions négatives).

— Suppression des 3336 points dont la précision de géolocalisation (`location_accuracy`) est strictement supérieure à 100 mètres.

— Suppression des 408 points ayant le mode marche (*walk*) et une vitesse (`speed`) strictement supérieure à 4 mètres par seconde (14.4 km/h).

— Suppression des 39 points ayant le mode vélo (*bike*) et une vitesse (`speed`) strictement supérieure à 10 mètres par seconde (36 km/h).

A l'issue du filtrage, il reste 16370 points dans le dataset.

8.2.2 Calcul de features

Le calcul des attributs et des features s'effectue de la même manière que pour l'expérimentation précédente (c.f. section 7.2.3). Cependant, en raison de la synchronisation des mesures de géolocalisation et d'accélération, les attributs additionnels d'écart temporel `location_time_delta` et `accelerometer_time_delta` sont strictement identiques. Le second n'est donc pas calculé et seuls trois attributs additionnels (`location_time_delta`, `location_distance_delta` et `accelerometer_magnitude`) sont ajoutés aux attributs de base présentés dans le tableau 8.1.

Tableau 8.3 – Sous-ensembles de features des différentes variantes du dataset abitrack. 5 features sont calculées à partir de chaque attribut (moyenne, minimum, maximum, écart-type et médiane).

Attribut	D_{full}	D_{location}	D'_{location}	$D_{\text{accelerometer}}$
location_accuracy	✓	✓		
location_speed	✓	✓	✓	
location_speed_accuracy	✓	✓		
location_heading	✓	✓	✓	
location_distance_delta	✓	✓	✓	
time_delta	✓	✓	✓	✓
accelerometer_magnitude	✓			✓
Nombre de features	**35**	**30**	**20**	**10**

En utilisant la méthode décrite dans la section 7.2.4, cinq indicateurs statistiques (minimum, maximum, moyenne, médiane, écart-type) sont calculés sur chaque voisinage de point avec une fenêtre temporelle de 60 secondes. Le nombre total de features s'élève donc à 35, et plusieurs sous-ensembles du dataset sont générés pour pouvoir comparer l'impact des features issues de la géolocalisation à celles issues de l'accélération (c.f. tableau 8.3).

A l'issue du calcul de features avec la fenêtre temporelle, les points ayant des valeurs manquantes (i.e. les valeurs qui ont été supprimées durant la première étape du filtrage, et qui n'ont pas été complétées durant l'étape de remplissage arrière décrite dans la section 7.2.4) sont également supprimés. Le dataset final contient 16256 points.

La chaîne de traitements introduite dans le chapitre 7 a été adaptée aux données collectées via l'application abitrack dans le contexte industriel de Citec. La prochaine section présente le protocole d'entraînement des classifieurs à comparer, dont Smapy, selon trois scénarios.

8.3 Entraînement des classifieurs

Tout comme dans l'expérimentation précédente (c.f. section 7.3.1.1), nous comparons les performances de cinq classifieurs (KNN, RF, ANN, SVM et Smapy) selon trois scénarios de découpage entre dataset d'entraînement et dataset de test :

— Découpage aléatoire : les données d'entraînement et de test sont séparées de façon aléatoire.

— Découpage temporel : les données d'entraînement couvrent une période temporelle antérieure aux données de test.

— Découpage spatial : les données d'entraînement couvrent une zone géographique distincte de celle des données de test.

8.3.1 Hyperparamétrisation

La sélection des paramètres optimaux se fait de la même manière qu'avec les données du dataset OCC-TMD, c'est-à-dire par validation croisée en cinq parties sur une grille de paramètres (à l'aide de la classe `GridSearchCV` de *scikit-learn*). Les valeurs des paramètres testés sont les mêmes que dans le tableau 7.7 pour les quatre classifieurs issus de la littérature. Pour Smapy, les valeurs testées pour chaque paramètre sont présentées dans le tableau 8.4.

Tableau 8.4 – Liste des grilles de valeurs pour la recherche des combinaisons optimales de paramètres de Smapy sur les sous-ensembles d'hyperparamétrisation de chaque dataset abitrack. Les valeurs en gras sont les valeurs les plus souvent choisies. Les valeurs en rouge ont été choisies pour tous les datasets.

Paramètre	**Grille de valeurs**		
R	**0.1**	0.2	0.5
M	10^{-6}		
O	0.1	**0.2**	0.5
E	Faux	**Vrai**	
N_c	**Sigmoïde**		
α	**0.0**	0.1	0.2
F_+	**1.0**		
F_-	0.5	1.0	**2.0**
Z	Aucun	**2**	3

La combinaison optimale de paramètres est recherchée pour chacun des quatre datasets D_{full}, D_{location}, D'_{location} et $D_{\text{accelerometer}}$. Pour des raisons d'économie de calcul, l'hyperparamétrisation est réalisée sur des sous-ensembles de ces datasets contenant 1000 points aléatoirement sélectionnés.

Dans le cas de Smapy, certaines valeurs de paramètres, comme le rayon initial des agents Contexte R de 0.1 ou l'activation de l'exclusion de points E, ont été systématiquement fixées pour les quatre datasets. La dilatation et la rétractation

des agents Contexte à la suite de feedback par le biais du paramètre α ont quant à elles été désactivées dans la majorité des cas.

Les combinaisons de paramètres sélectionnées sont ensuite utilisées pour l'apprentissage des classifieurs sur l'ensemble des données dans les trois scénarios considérés.

8.3.2 Découpage aléatoire

D'une manière assez classique, nous séparons le dataset en deux sous-ensembles :

— Le jeu d'entraînement contenant 80% des points sélectionnés aléatoirement, soit 13004 points.

— Le jeu de test contenant les 20% des points restants (également ordonnés aléatoirement), soit 3252 points.

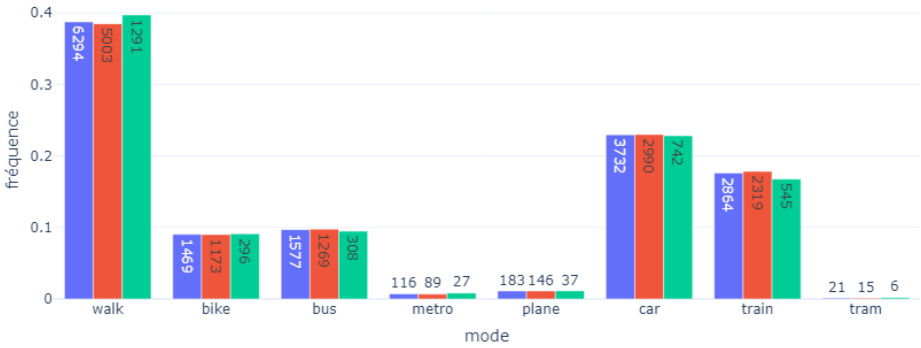


FIGURE 8.3 – Diagramme en barres des parts modales dans le dataset complet (bleu), dans le dataset d'entraînement (rouge) et dans le dataset de test (vert) dans le scénario du découpage aléatoire. Les effectifs sont notés dans les barres.

Les répartitions modales des sous-ensembles sont représentées sur la figure 8.3. La répartition des classes dans les deux sous-ensembles est sensiblement identique à celle du dataset complet. Ainsi, ce scénario offre un jeu d'entraînement dont la représentativité est théoriquement optimale. L'objectif de ce scénario est de servir de référence pour étudier la généralisation de la méthode dans le temps et l'espace.

8.3.3 Découpage temporel

Tout en conservant la même proportion de données de test, nous définissions un autre découpage basé sur la temporalité afin d'étudier la capacité de notre méthode à se généraliser dans le temps :

— Le jeu d'entraînement contenant les 80% premiers points observés, soit 13004 points.

— Le jeu de test contenant les 20% derniers points observés (également ordonnés dans l'ordre chronologique), soit 3252 points.

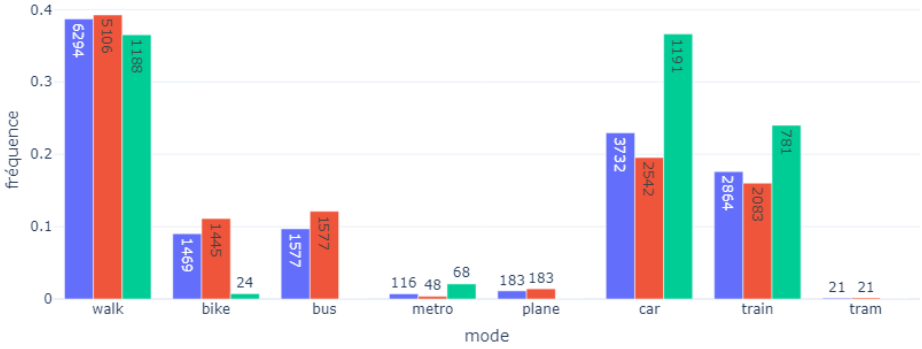


FIGURE 8.4 – Diagramme en barres des parts modales dans le dataset complet (bleu), dans le dataset d'entraînement (rouge) et dans le dataset de test (vert) dans le scénario du découpage temporel. Les effectifs sont notés dans les barres.

En raison du déroulement de la collecte de données, la répartition des classes du sous-ensemble d'entraînement (i.e. tous les points collectés jusqu'au 8 décembre à 12h08) est sensiblement différente de celle du sous-ensemble de test (i.e. tous les points collectés après le 8 décembre à 12h08). Les parts de la voiture et du train sont beaucoup plus importantes dans le sous-ensemble de test, tandis que les modes vélo, bus, avion et tramway y sont absents (c.f. figure 8.4).

Du fait du faible nombre d'utilisateurs et de la discontinuité des mesures dans le temps, il est difficile d'évaluer si cette forte variabilité est due à un changement de comportement dans le temps. Une telle affirmation nécessiterait un jeu de données beaucoup plus exhaustif.

8.3.4 Découpage spatial

Le dernier scénario vise à étudier la généralisation de notre méthode d'un territoire géographique à l'autre. Le découpage suivant est proposé :

— Le jeu d'entraînement contenant l'ensemble des 12703 points collectés à Toulouse et ses alentours (i.e. de latitude strictement inférieure à 44), soit environ 78,14% des points.

— Le jeu de test contenant les 3553 points collectés en dehors de Toulouse (i.e. de latitude supérieure à 44), soit environ 21,86% des points.

La zone de Toulouse a été choisie pour le jeu de test car elle représente à peu près la même proportion que les sous-ensembles de test des deux autres scénarios, et dispose d'infrastructures couvrant l'ensemble des classes considérées (métro, tramway, aéroport, etc.). La figure 8.5 illustre toutefois une importante différence de répartition des classes entre les données de Toulouse et le reste des points collectés. En particulier, la part du mode vélo est trois fois plus importante dans le jeu de test, tandis que les modes bus, train et tramway y

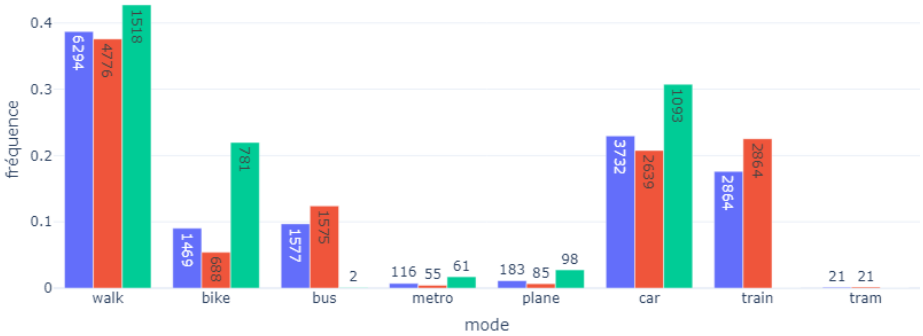


FIGURE 8.5 – Diagramme en barres des parts modales dans le dataset complet (bleu), dans le dataset d'entraînement (rouge) et dans le dataset de test (vert) dans le scénario du découpage spatial. Les effectifs sont notés dans les barres.

TABLEAU 8.5 – Comparaison des précisions de classification sur les différents sous-ensembles du dataset abitrack pour le scénario de découpage aléatoire.

	D_{full}	D_{location}	D'_{location}	$D_{\text{accelerometer}}$
KNN	74.66%	73.74%	73.06%	70.39%
RF	**76.91%**	**76.78%**	**75.12%**	**74.82%**
ANN	72.17%	69.65%	60.58%	61.62%
SVM	73.40%	68.63%	61.07%	61.72%
Smapy	70.88%	70.05%	52.31%	57.38%

sont absents.

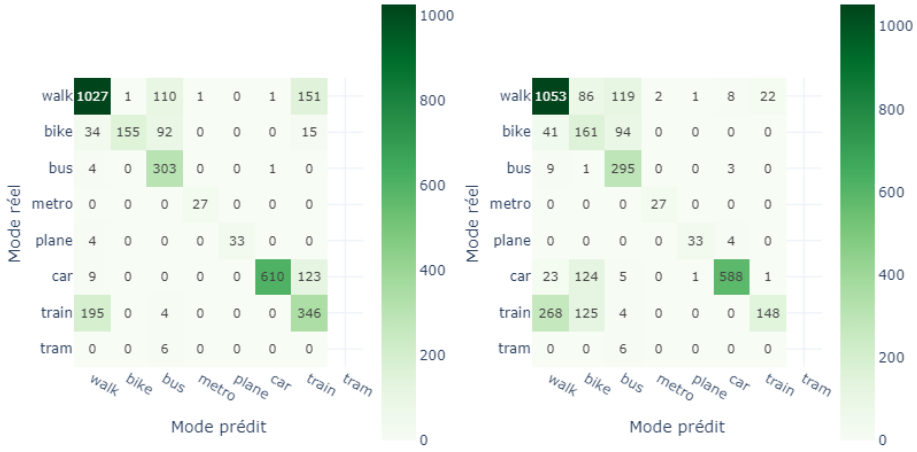
Le protocole d'apprentissage présenté dans cette section est basé sur une hyperparamétrisation des classifieurs et trois scénarios d'entraînement. Dans la prochaine section, les résultats obtenus pour chacun de ces scénarios sont présentés.

8.4 Résultats

Nous comparons dans cette section les résultats obtenus pour chaque scénario en termes de précision de classification. Nous étudions également l'importance des features en entrée du classifieur Random Forest, ainsi que des statistiques concernant les mécanisems impliqués dans l'apprentissage des instances de Smapy.

8.4.1 Découpage aléatoire

Le scénario de découpage aléatoire assure la plus forte représentativité des données d'entraînement par rapport au jeu de test. Tous les classifieurs



(a) Random Forest (meilleure précision) (b) Smapy

FIGURE 8.6 – Comparaison des matrices de confusion obtenues avec le meilleur classifieur et Smapy pour le scénario de découpage aléatoire sur D_{full}.

implémentés parviennent à dépasser les 70% de précision lorsqu'ils sont entraînés sur l'ensemble des features disponibles (D_{full}). Les résultats compilés dans le tableau 8.5 indiquent que Random Forest atteint la plus forte précision avec 76,91% sur D_{full}, et surpasse les autres classifieurs dans tous les autres cas. La précision obtenue avec Smapy est équivalente à celle des classifieurs issus de la littérature sur D_{full} et D_{location}. Elle se dégrade lorsque les features liées à la précision de géolocalisation sont enlevées, et avec les features issues de l'accélération uniquement.

La figure 8.6a représente la matrice de confusion obtenue avec Random Forest sur D_{full} pour le scénario du découpage aléatoire. Les principales confusions observées sont :

— La confusion train/marche (195 points de train mal classifiés en marche, 151 points de marche mal classifiés en train), probablement due à des erreurs de labélisation. Tout déplacement en train est nécessairement précédé et suivi d'un déplacement à pied au sein de la gare, et le changement de classe est généralement déclaré dans abitrack à l'entrée dans le train pour le départ, et à la sortie de la gare pour l'arrivée. Ainsi, des tronçons de marche ou des moments d'arrêts (attente du départ) sont labélisés comme des déplacements en train dans le jeu de données.

— La confusion voiture/train uniquement dans un sens (123 points de voiture mal classifiés en train). Ces deux modes peuvent avoir des profils de vitesse et d'accélération similaires et une faible variation du relèvement (*Heading*) sur de longs trajets (autoroute versus trains à vitesse modérées comme les Intercités).

— Les confusions entre les modes marche, vélo et bus (110 points de marche

Tableau 8.6 – Comparaison des précisions de classification sur les différents sous-ensembles du dataset abitrack pour le scénario de découpage temporel.

	D_{full}	D_{location}	D'_{location}	$D_{\text{accelerometer}}$
KNN	38.71%	**55.72%**	38.65%	36.99%
RF	**47.85%**	45.91%	**44.77%**	38.41%
ANN	41.36%	40.47%	37.39%	**40.50%**
SVM	43.67%	40.13%	39.58%	40.47%
Smapy	40.74%	44.19%	34.50%	35.12%

mal classifiés en bus, 92 points de vélo mal classifiés en bus et 34 points de vélo mal classifiés en marche). Ces trois modes sont généralement les plus difficiles à classifier dans un contexte urbain où les arrêts sont fréquents (e.g. feux rouges, embouteillages), les vitesses faibles et l'accélération instable.

Ces confusions sont comparées à celles obtenus avec Smapy sur la figure 8.6b. Le nombre de points de marche mal classifiés en train est beaucoup plus faible (22 points), et les modes voiture et train sont souvent mal classifiés en vélo (124 et 125 points respectivement). Les confusions entre les modes marche, vélo et bus subsistent.

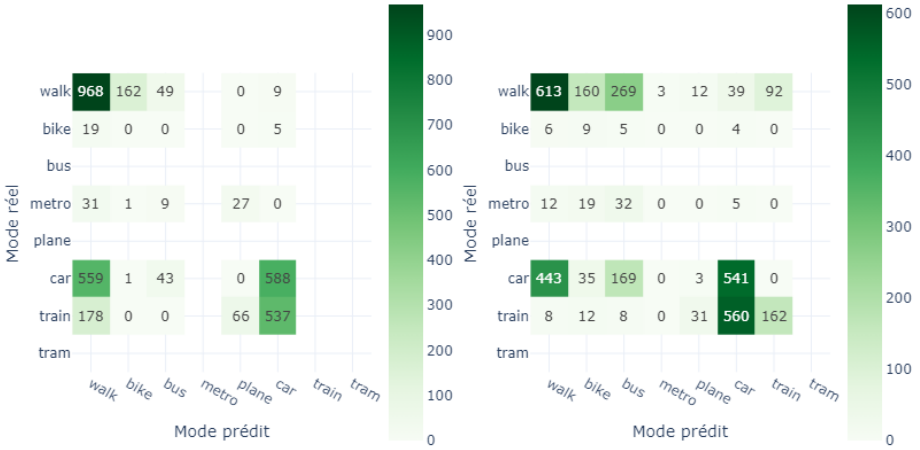
Ces résultats indiquent que les deux classifieurs ne basent pas leur prédictions sur les mêmes features. Dans les deux cas, aucune prédiction n'a été faite pour le mode tramway, ce qui n'est guère étonnant quand on voit que seuls 21 points correspondant à ce mode sont présents dans l'ensemble des données.

8.4.2 Découpage temporel

Le scénario temporel a pour but d'étudier la généralisation dans le temps des classifieurs implémentés pour le problème de TMD. La première chose que nous remarquons en étudiant les précisions obtenues dans le tableau 8.6 est la forte dégradation des performances de tous les classifieurs par rapport au scénario aléatoire. Le meilleur score obtenu est celui de KNN sur D_{location} avec 55,72% de précision. Sur l'ensemble des features (D_{full}), Random Forest est le classifieur le plus performant avec 47,85% de précision. Smapy obtient des précisions équivalentes à celles des autres classifieurs sur toutes les combinaisons de features.

La figure 8.7 représente les matrices de confusion obtenues sur D_{full} avec Random Forest et Smapy. Les principales confusions observées sont :

— La confusion train/voiture (respectivement 537 et 560 points de train mal classifiés en voiture), très importante, pouvant être due à la faible représentativité du dataset d'entraînement. En effet, le seul long trajet en train à grande vitesse (entre Bordeaux et Paris) est observé après le 8 décembre 2023, et donc uniquement dans les données de test.



(a) Random Forest (meilleure précision) (b) Smapy

FIGURE 8.7 – Comparaison des matrices de confusion obtenues avec le meilleur classifieur et Smapy pour le scénario de découpage temporel sur D_{full}.

— La confusion bus/voiture, qui sont deux modes difficiles à discerner en milieu urbain ou sur l'autoroute (les cars longue distance étant labélisés comme des bus durant la collecte).

— Les confusions entre les modes marche, vélo et bus, probablement pour les mêmes raisons que dans le scénario précédent.

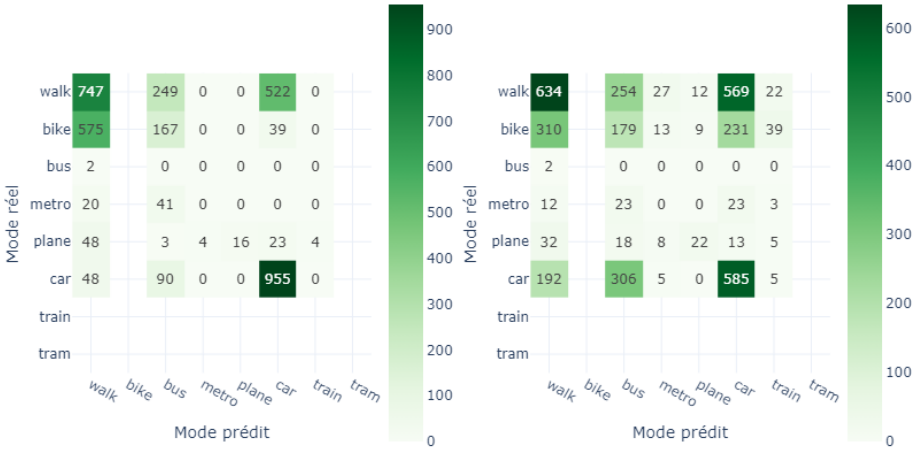
L'instance de Random Forest n'a prédit aucun point pour les modes métro et train, ce qui est surprenant dans la mesure où le mode train est le troisième mode le plus représenté à la fois dans les données d'apprentissage et les données de test (après la marche et la voiture). Le métro n'a pas non plus été prédit par Random Forest, mais il est pratiquement absent de la base d'apprentissage pour ce scénario. Dans les deux cas, le mode tramway n'a jamais été prédit (à raison car aucun point de tranmway n'est présent dans les données de test). Le mode avion a été prédit par les deux classifieurs alors qu'absent dans la base de test, ce qui peut-être du à une assimilation à ce mode des outliers en termes de vitesse.

8.4.3 Découpage spatial

Les précisions obtenues pour le scénario de découpage spatial sont indiquées dans le tableau 8.7. La dégradation de la précision par rapport au scénario aléatoire est moins importante que celle du scénario de découpage temporel, mais les précisions obtenues d'un sous-ensemble de features à l'autre et d'un classifieur à l'autre sont beaucoup plus instables. Le meilleur score est atteint par SVM sur D'_{location} (i.e. les features issues de la géolocalisation sans la précision de géolocalisation) avec 69,35%. Sur l'ensemble des features D_{full}, le meilleur score est atteint par Random Forest avec seulement 48,35%. Smapy occupe de

TABLEAU 8.7 – Comparaison des précisions de classification sur les différents sous-ensembles du dataset abitrack pour le scénario de découpage spatial.

	D_{full}	D_{location}	D'_{location}	$D_{\text{accelerometer}}$
KNN	33.80%	40.53%	46.24%	25.13%
RF	**48.35%**	**51.93%**	53.36%	16.52%
ANN	41.35%	45.45%	60.17%	24.04%
SVM	36.70%	37.49%	**69.35%**	26.91%
Smapy	34.93%	41.37%	28.23%	**34.73%**



(a) Random Forest (meilleure précision) (b) Smapy

FIGURE 8.8 – Comparaison des matrices de confusion obtenues avec le meilleur classifieur et Smapy pour le scénario de découpage spatial sur D_{full}.

bonnes positions, avec la première place atteinte pour $D_{\text{accelerometer}}$ (34,73%), sauf sur D'_{location} où il performe très mal avec seulement 28,23% de précision.

Tout comme pour le scénario de découpage temporel, de fortes confusions sont observées pour Random Forest et Smapy avec l'ensemble des features (c.f. figure 8.8) :

— La confusion marche/voiture (respectivement 522 et 569 points de marche mal classifiés en voiture, et respectivement 48 et 192 points de voiture mal classifiés en marche), très importante, qui traduit potentiellement une différence de comportement de conduite (en termes de vitesse notamment) entre Toulouse et le reste de la zone d'étude, bien que la représentativité du dataset ne soit pas suffisante pour l'affirmer.

— La confusion voiture/bus déjà observée dans le scénario de découpage temporel.

— Les confusions entre les modes marche, vélo et bus observées dans les deux

autres scénarios.

L'ensemble de test contient uniquement les points observés à Toulouse. Les modes train et tramway y sont absents et le mode bus ne correspond qu'à deux points. Si aucun point de tramway n'a été prédit dans les deux cas, pour les mêmes raisons que dans le scénario de découpage aléatoire, les prédictions de bus ont conduit à de fortes confusions et une importante dégradation de la précision de classification. Enfin, aucun des deux classifieurs n'a prédit le mode vélo alors qu'il s'agit de la troisième classe la plus représentée dans les données de test (781 points). Les données d'entraînement contenaient pourtant 688 points de vélo (c.f. figure 8.5).

8.4.4 Importance des features

Tout comme pour l'expérience sur le dataset OCC-TMD, nous étudions l'importance des features obtenues par le critère d'impureté de Gini calculé lors de l'apprentissage de les instances de Random Forest sur D_{location}. La figure 8.9 montre les sommes des importances de chaque feature sur les trois scénarios, par ordre décroissant. Le "classement" est dominé par les features issues de la vitesse (positions 1, 2, 3, 8 et 20), en particulier la vitesse maximale sur les segments définis par les fenêtres temporelles (`max_speed`). Ces features sont celles ayant le plus participé à la discrimination des modes de transport sur les données considérées. En comparaison, les features liées à la norme de l'accélération (`accelerometer_magniture`) sont un peu moins importantes que dans l'expérimentation précédente.

Les features basées sur les écarts temporels sont moins importantes que pour le dataset OCC-TMD, ce qui était un résultat attendu. En effet, dans le fonctionnement normal d'abitrack, la fréquence de collecte des points est constante et égale à 0.1Hz (un point toutes les dix secondes). Les valeurs de l'attribut `time_delta` sont donc constantes sauf lors des pertes de signal qui peuvent donner des informations sur l'utilisation de certains modes comme le métro et l'avion. Le pouvoir discriminant des features liées aux écarts temporels est donc plus faible que lors de l'expérimentation précédente (positions 5, 7, 18, 22 et 24).

Les features liées à la précision de géolocalisation sont relativement mal classées en termes d'importance (positions 21, 25, 26, 27 et 29) et ne permettent donc pas de discriminer les modes avec Random Forest. De façon plus surprenante, les features liées au relèvement (*Heading*) sont les moins importantes (positions 28, 30, 31, 32 et 34) alors qu'elles sont censées traduire la forte variabilité des directions prises avec des modes tels que la marche ou le vélo.

8.4.5 Comportement de Smapy

En dehors de la capacité de généralisation dans le temps ou dans l'espace, nous avons voulu étudier les mécanismes impliqués dans l'apprentissage de Smapy sur les différents scénarios. Grâce à des métriques internes intégrées dans l'implémentation python de Smapy, nous représentons sur la figure 8.10

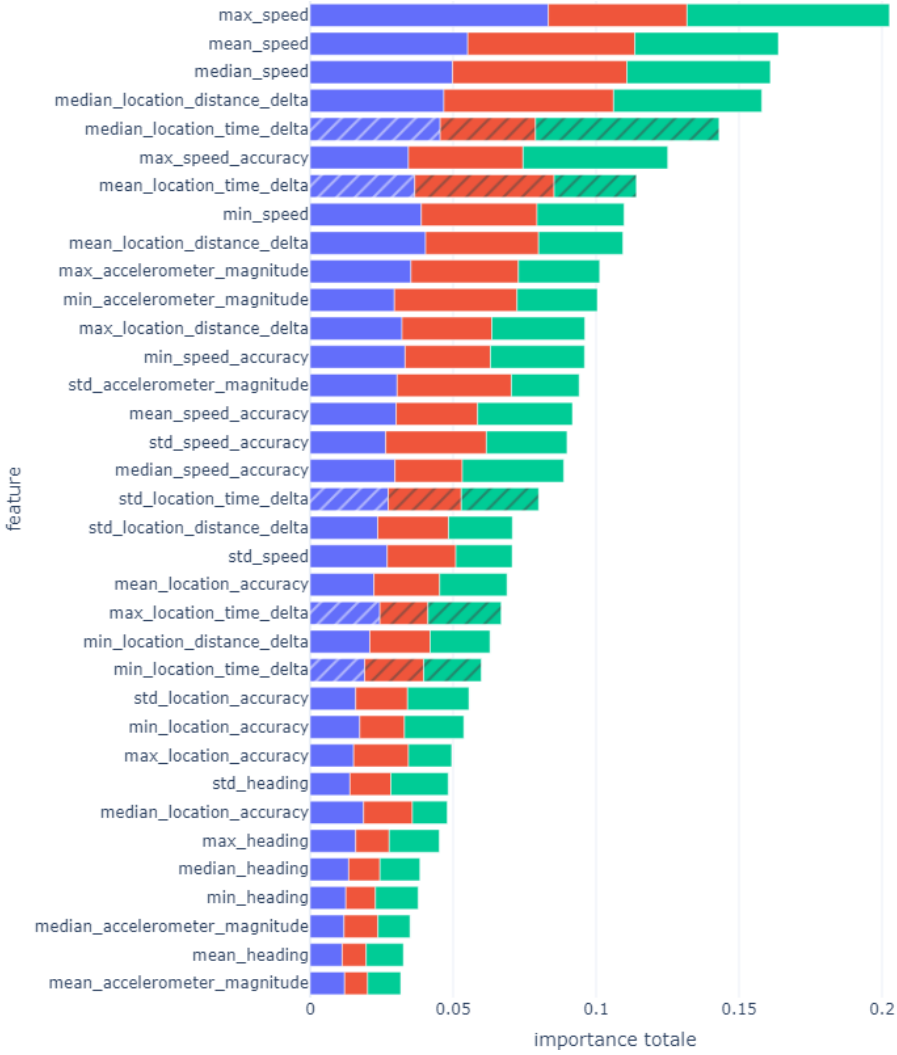


FIGURE 8.9 – Sommes des importances de Gini selon les trois scénarios (découpage aléatoire en bleu, découpage temporel en rouge et découpage spatial en vert) pour le classifieur Random Forest entraîné sur D_{full} avec la meilleure combinaison de paramètres (sélectionnée par validation croisée). Les features d'écarts temporels sont représentées hachurées.

le nombre d'occurrence des mécanismes de coopération et des situations non coopératives (SNC) de conflit et de compétition (c.f. sections 5.2.3 et 5.2.4) à la fin de l'apprentissage sur D_{location} pour les trois scénarios.

La seule différence notable entre les scénarios est le nombre de dilatations et de rétractations. Avec le découpage aléatoire, les agents Contexte se dilatent deux

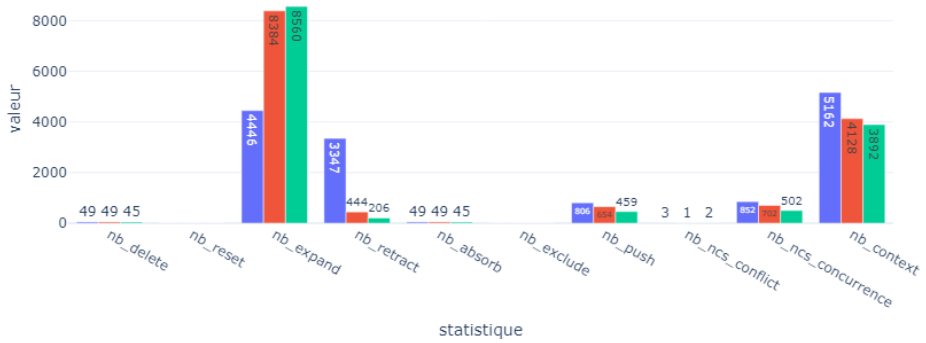


FIGURE 8.10 – Nombre d'occurrence de différents mécanismes selon les trois scénarios (découpage aléatoire en bleu, découpage temporel en rouge et découpage spatial en vert) dans l'instance de Smapy entraînée sur D_{full} avec la meilleure combinaison de paramètres (sélectionnée par validation croisée).

fois moins souvent et se rétractent respectivement 8 à 16 fois plus souvent qu'avec les découpages temporel et spatial. Cela s'explique par la plus forte discontinuité des données (au niveau de leur ordre d'apparition) lues par Smapy avec le découpage aléatoire, et donc un nombre d'oscillations des zones d'activation des agents Contexte plus important de part et d'autre des frontières de décision. Dans les scénarios de découpage temporel et spatial, les points sont lus par Smapy dans un ordre chronologique, et ont une cohérence temporelle et spatiale beaucoup plus grande. Ainsi, au cours d'un trajet, les points successifs activent les mêmes agents Contexte qui se dilatent plus souvent et plus loin dans l'espace des variables d'entrée. Il est également intéressant de constater que le nombre d'agents Contexte à la fin de l'apprentissage est plus important dans le scénario de découpage temporel, ce qui indique des agents Contexte plus étendus et moins nombreux dans les deux autres scénarios (et confirme les dilatations plus nombreuses).

Les résultats obtenus montrent une dégradation de la performance de tous les classifieurs entre le scénario de découpage aléatoire et les deux autres scénarios. L'importance relative des variables ainsi que le comportement de Smapy sont également étudiés. Une discussion de ces résultats est proposée dans la prochaine section.

8.5 Discussion

Le premier objectif de cette expérimentation était de valider et évaluer la qualité de la méthode de classification du mode de transport mise en place sur les données OCC-TMD et décrite dans le chapitre 7. Cette fois-ci, la collecte de données est plus importante et contrôlée de bout en bout par Citec à travers

une application dédiée. L'expérimentation sur les données abitrack permet de montrer la faisabilité d'une collecte de données et d'une labélisation intégrée qui peuvent être mises en entrée d'une chaîne de pré-traitement et d'un classifieur.

Le second objectif était d'explorer trois scénarios d'apprentissage pour étudier la généralisation de la solution proposée dans le temps et dans l'espace. Le scénario de référence, qui consiste à sélectionner aléatoirement 80% des points pour constituer la base d'apprentissage, est celui pour lequel tous les classifieurs testés obtiennent les meilleurs précisions (entre 70% et 77%). Les deux autres scénarios traduisent une forte dégradation de la précision de classification lorsque le jeu de test est collecté à une période différente ou sur une zone d'étude différente. Cette mauvaise généralisation est toutefois à fortement nuancer, car le dataset collecté est très peu fourni (notamment pour des modes tels que le métro, le tram, le train et l'avion) et les cinq volontaires peu représentatifs de la population globale (uniquement des hommes dans la vie active et à des postes de cadre). Afin de quantifier la difficulté de généralisation dans le temps et l'espace, il faudrait tester les mêmes classifieurs et features sur des datasets beaucoup plus denses et importants. Cette expérimentation nous permet tout de même d'identifier les défis liés à l'industrialisation d'une méthode de détection du mode de transport en l'absence d'une source de données utilisateur fiable et statistiquement représentative de la population des zones d'étude.

En ce qui concerne la méthode de pré-traitement en elle-même, nous constatons que les importances relatives des features ont évolué par rapport à l'expérimentation sur les données OCC-TMD. Cela met en lumière la nécessité d'adapter le pré-traitement à la nature des données, et en particulier aux règles d'acquisition des mesures sur les utilisateurs (e.g. la fréquence d'observation qui peut être constante ou dynamiquement gérée selon les déplacements détectés). De plus, certaines features évoquées dans le chapitre 4, liées notamment à des données contextuelles (e.g. distance au réseau de chemins de fer, à un arrêt de bus, etc.) ou à des indicateurs différents (e.g. coefficients de Fourier, features profondes, etc.), pourraient être testées dans ce contexte pour évaluer leur pertinence sur les données abitrack.

Enfin, les algorithmes de post-traitement évoqués dans la section 4.4 pourraient être testés pour le scénario de découpage temporel (car cela nécessite la continuité temporelle des prédictions à corriger) pour évaluer leur intérêt. A l'inverse, une étape préalable de détection des arrêts (et donc de segmentation des trajets et arrêts) pourrait être ajoutée pour diminuer le bruit des prédictions en sortie et anticiper la résolution du problème de détection des motifs de déplacement (i.e. les activités réalisées aux arrêts) pour laquelle abitrack a également été conçue.

Quatrième partie

Conclusion et perspectives

Chapitre 9

Conclusion et perspectives

Cette thèse fait le lien entre le domaine applicatif de l'analyse de la mobilité et le paradigme des systèmes multi-agents d'apprentissage par contexte. Dans une première partie, le contexte industriel et la notion de connaissance de l'état de la mobilité ont été présentés. Une seconde partie de la thèse a permis d'établir un état de l'art multi-disciplinaire portant sur l'analyse de la mobilité, avec le cas d'usage de la détection du mode de transport, ainsi que le positionnement scientifique de Smapy, la solution algorithmique proposée. Ce nouveau classifieur, à l'intersection entre systèmes multi-agents d'apprentissage par contexte et ensemblisme, constitue avec une série d'expérimentations sur la détection du mode de transport les contributions présentées dans une troisième partie.

Dans cette dernière partie, une conclusion générale reprenant chacun des chapitre de ce manuscrit est proposée. Dans un second temps, les contributions de cette thèse sont rappelées et discutées par rapport aux objectifs initiaux. Enfin, des perspectives de recherche sont présentées.

Conclusion générale

Le chapitre 1 a défini le concept d'état de la mobilité. Cette notion représente un enjeu pour les collectivités locales car la connaissance de l'état de la mobilité sur un territoire influence les choix d'urbanisme et de développement. Cette thèse CIFRE a pour but d'explorer des solutions d'analyse de la mobilité à partir de données innovantes, principalement mesurées sur les usagers. Trois objectifs ont ainsi été introduits :

— Caractériser les données de mobilité existantes et étudier leur potentiel.

— Proposer une solution algorithmique sous la forme d'un modèle d'apprentissage automatique pour l'analyse de la mobilité.

— Etudier et résoudre le cas d'application de la détection du mode de transport.

Le chapitre 2 répond au premier objectif en dressant un état de l'art des données de mobilité existantes selon une typologie nouvelle : les données contextuelles, les données agrégées et les données utilisateur. De nombreuses approches d'analyse de la mobilité se basent sur les données utilisateur, parfois couplées avec des données contextuelles et même redressées avec des données agrégées de référence issues d'enquêtes. Dans le cas d'application industriel de Citec, il a été choisi de retenir les données utilisateur comme source de données principale de la solution algorithmique développée.

Le chapitre 3 établit le positionnement de la solution algorithmique proposée pour répondre au deuxième objectif et justifie ainsi l'approche utilisée, à

l'intersection entre modèle ensembliste, modèle de voisinage, système multi-agents d'apprentissage par contexte et même modèle d'apprentissage par renforcement. En particulier, trois pré-requis sont établis pour la solution algorithmique proposée :

— La compatibilité avec l'apprentissage en ligne et une capacité d'adaptation rapide.

— De bonnes performances sur des problèmes fortement non linéaires.

— Un fort potentiel d'explicabilité.

Certains modèles d'apprentissage automatique présentés dans cette section sont également utilisé dans les expérimentations de la thèse, comme modèles internes des agents Contexte de Smapy ou comme modèles concurrents.

Le problème de détection du mode de transport, qui s'inscrit dans le cadre de la classification supervisée, est l'objet du troisième objectif de la thèse. Dans le chapitre 4, un état de l'art de la classification du mode de transport est réalisé, détaillant toutes les étapes des chaînes de traitement mises en place dans la littérature. Les données les plus couramment utilisées pour résoudre ce problème sont les mesures de géolocalisation et d'accélération, c'est-à-dire deux types de données utilisateur.

Smapy, la solution algorithmique proposée, est introduite dans le chapitre 5. La structure du système, le rôle des différents types d'agents ainsi que les règles de fonctionnement assurant la coopération sont détaillés et formalisés. En particulier, les motivations derrière chacun des mécanismes implémentés sont présentées. Ce nouveau classifieur s'inscrit dans la continuité des systèmes multi-agents d'apprentissage par contexte, à la différence que chaque agent Contexte possède un classifieur interne résolvant localement le problème de classification considéré.

Le chapitre 6 présente une expérimentation dont l'objectif est de montrer que Smapy est capable de transformer un problème de classification non linéaire en problème de coopération entre agents. Pour cela, des classifieurs linéaires sont testés seuls et à l'intérieur d'une instance de Smapy sur trois problèmes de classification de linéarité différentes. Les résultats indiquent que l'intégration des classifieurs linéaires dans les agents Contexte de Smapy permet de résoudre les problèmes les moins linéaires, là où les classifieurs linéaires seuls en sont incapables. Cette expérimentation vient donc valider le deuxième pré-requis établi dans le chapitre 3.

Le chapitre 7 introduit une chaîne de traitement pour la résolution du problème de détection du mode de transport basée notamment sur les écarts-temporels entre les observations provenant de données de géolocalisation et d'accélération mesurées sur un usager. Les performances de Smapy sur ces données sont comparées à celles de quatre autres classifieurs issus de la littérature. Ces quatre classifieurs sont également testés sur deux datasets de référence dans la littérature afin de valider la pertinence de la chaîne de traitement mise en place.

Dans le chapitre 8, une nouvelle expérimentation est menée afin de valider la chaîne de traitement présentée dans le chapitre 7 sur de nouvelles données

collectées par Citec. Trois scénarios d'entraînement ont été testés afin d'évaluer la capacité de généralisation dans le temps et dans l'espace de Smapy et des quatre autres classifieurs de référence, afin de caractériser l'adéquation de la solution proposée avec le premier des pré-requis rappelés dans le chapitre 3. Une dégradation de la précision de classification dans le temps et d'une zone géographique à l'autre a été observée avec chaque modèle. Ces résultats mettent en avant la nécessité d'entraîner les classifieurs sur des données plus représentatives en termes de périodes et de zones géographiques, mais également d'importants biais présents dans les données en raison du nombre réduit d'usagers observés.

Contributions

D'un point de vue applicatif, cette thèse présente plusieurs contributions dans le domaine de l'analyse de la mobilité. A travers Smapy, cette thèse constitue également une évolution des sysèmes multi-agents d'apprentissage par contexte, qui s'inscrivent eux-même dans la théorie des systèmes multi-agents adaptatifs (AMAS) [Cap+03]. Smapy est un nouveau type de classifieur ensembliste qui vient apporter une contribution au domaine de l'apprentissage automatique.

Contribution à l'analyse de la mobilité

Cette thèse présente plusieurs contributions au domaine de l'analyse de la mobilité. Tout d'abord, une classification des données de mobilité ainsi qu'un état de l'art de leur utilisation sont proposés dans le chapitre 2. Cette réponse au premier objectif de la thèse a pour vocation d'éclairer sur la diversité des sources de données disponibles, ainsi que de la diversité de leurs utilisations.

De plus, le problème plus spécifique de la détection du mode de transport à partir de données utilisateur est abordée à travers un état de l'art dans le chapitre 4 et deux expérimentations d'une chaîne de traitement développée dans les chapitres 7 et 8. Les résultats expérimentaux indiquent que cette chaîne de traitement, basée notamment sur des attributs d'écarts temporels entre les observations, apporte un gain de précision de classification par rapport à d'autres approches de la littérature.

Contribution à l'apprentissage par contexte dans la théorie des AMAS

Smapy se place dans la lignée des systèmes multi-agents adaptatifs (AMAS) [Cap+03] conçus selon le paradigme de l'apprentissage par contexte auto-adaptatif (SACL) [Boe+15]. En particulier, Smapy reprend certains mécanismes d'ELLSA [Dat21] en ajoutant des classifieurs à l'intérieur des agents Contexte. Le système est alors capable de résoudre des problèmes de classification supervisée en faisant coopérer les agents entre eux.

Un des objectifs de ce manuscrit est de transmettre les intuitions ayant conduit à la création de chacun des mécanismes de Smapy. Contrairement à d'autres AMAS, l'apprentissage par contexte impose une forte couche d'abstraction sur

la structure des agents. Ces derniers ne représentent pas d'objets ou d'individus réels et l'environnement dans lequel ils évoluent est un espace abstrait composé des variables d'entrée du problème. Les intuitions derrière les mécanismes de coopération n'ont donc pas d'interprétation dans le monde réel, mais ont un sens lorsque l'on considère le problème à résoudre comme la recherche d'un pavage optimal de l'espace des variables d'entrée, et les agents Contexte comme des formes géométriques qu'il faut agencer le mieux possible pour "capter" les points d'observation à venir.

Contribution à l'apprentissage automatique

Smapy a été conçu comme une agrégation ensembliste de modèles "faibles" à l'image d'algorithmes tels que Random Forest. Cependant, les propriétés de coopération entre agents se présentant comme des hypercubes dans l'espace des variables d'entrée rappelle le fonctionnement des modèles de voisinage. La possibilité d'asymétrie des *feedbacks* renvoyés aux agents Contexte en cas de bonne ou de mauvaise prédiction peut être vue comme une version simplifiée du système de *feedbacks* utilisé dans l'apprentissage par renforcement. Le système doit s'auto-construire de manière à maximiser la coopération globale, c'est-à-dire de maximiser les *feedbacks* positifs.

Enfin, en intégrant des classifieurs dans les agents Contexte, Smapy fait le lien entre les architectures SACL et la classification supervisée. L'expérimentation conduite dans le chapitre 6 montre que le système est capable de résoudre des problèmes de classification non linéaires en les transformant en un problème de coopération entre modèles linéaires. La notion d'apprentissage automatique, et plus particulièrement d'apprentissage en ligne, est intégrée à l'intérieur d'une architecture SACL par le biais des modèles internes des agents Contexte.

Perspectives

Bien qu'apportant de multiples contributions, cette thèse laisse également derrière elle de nombreuses pistes à explorer ou approfondir, tant au niveau de Smapy que de la notion d'explicabilité ou de la connaissance des autres axes de l'état de la mobilité.

Améliorations de Smapy

Optimisation du temps de calcul

Plusieurs aspects de Smapy restent à améliorer. Tout d'abord, afin de conduire plus facilement des tests de validation sur de gros jeux de données, le système devrait être implémenté de manière plus optimisée pour fortement diminuer le temps de calcul. Deux implémentations python ont été proposées pendant cette thèse : une explicite et une implicite. L'implémentation explicite a une structure de classes très proche de la conception schématique des architectures SACL. L'implémentation implicite est une tentative d'optimiser les performances de

Smapy et de maximiser la compatibilité avec *scikit-learn* en faisant abstraction de certains éléments tels que l'agent Head, ou en rassemblant tous les agents Contexte dans un même objet. Cependant, il est possible d'aller plus loin dans l'abstraction, avec python ou un autre langage de plus bas niveau, afin d'optimiser les temps de calculs de Smapy.

Agrégation de différents classifieurs

Dans sa version actuelle, les agents Contexte d'une instance de Smapy ont tous le même type de classifieur interne, avec les mêmes paramètres par défaut (à travers les paramètres internes `local_model_default` et `local_model_default_params`). En réalité, rien dans les implémentations n'empêche d'agréger plusieurs classifieurs différents dans les agents Contexte, mais aucun critère de sélection de modèle n'a encore été implémenté. En d'autres termes, en l'absence d'un critère pour choisir tel ou tel classifieur au moment de la création d'un agent Contexte, cette fonctionnalité n'a jamais été testée.

Pourtant, l'agrégation de classifieurs différents pourrait présenter un intérêt pour les problèmes linéaires par endroits et fortement non linéaires à d'autres endroits. Si le système possède deux modèles locaux par défaut, un "faible" (i.e. un modèle linéaire) et un plus "puissant" (i.e. un modèle non linéaire), et qu'il est possible d'évaluer si la zone de l'espace dans laquelle un agent Contexte doit être créé est linéairement séparable (i.e. dans laquelle les classes peuvent être séparées par des hyperplans), alors le système pourrait être optimisé. Dans les zones linéairement séparables, un modèle "faible" suffit, tandis que les zones qui contiennent des frontières de classes fortement non linéaires nécessitent des modèles locaux plus complexes. Ainsi, il serait possible d'augmenter la précision de classification dans les zones fortement non linéaires, tout en privilégiant des modèles plus simples nécessitant moins de calculs dans les zones linéairement séparables.

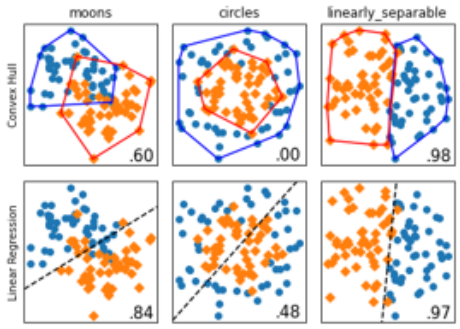


FIGURE 9.1 – Visualisation en deux dimensions du critère de l'enveloppe convexe (ligne du haut) et de la régression linéaire (ligne du bas) sur les trois jeux de données introduits dans la section 6.1.

Durant cette thèse, deux critères d'évaluation de la linéarité locale ont été

imaginés et représentés sur la figure 9.1. Dans les deux cas, la première étape consiste à quadriller l'espace des variables d'entrées et à calculer un indice de "séparabilité linéaire" sur chacune des zones ainsi constituées.

Le premier critère consiste à estimer les enveloppes convexes de chaque classe présente dans la zone, puis de calculer leur indice de superposition (c.f. définition 5.2.5). Si les zones sont parfaitement séparables linéairement, cette intersection est nécessairement vide, donc l'indice nul. Dans le cas extrême du dataset *Circles* où le nuage de points d'une classe est intégralement contenu dans celui d'une autre classe, l'indice de superposition est égal à 1. Pour désigner la séparabilité linéaire, on prend donc 1 − indice de superposition comme critère.

Le second critère envisagé est l'utilisation d'un modèle de régression linéaire dans chaque zone. Si le modèle parvient à classifier parfaitement les classes dans cette zone (i.e. avec une précision égale à 1), alors celle-ci est parfaitement séparable linéairement. Dans le cas extrême du dataset *Circles*, la performance du modèle est la même que le hasard. Le critère se présente alors comme la précision du modèle de régression linéaire appliqué sur la zone.

Dans les deux cas, il n'est pas nécessaire de stocker les points observés dans le système, car les enveloppes convexes ou les modèles de régression linéaires peuvent être mis à jour en utilisant uniquement la dernière observation.

Terme de généralisation dans le score des agents Contexte

Une autre piste d'amélioration de Smapy est l'ajout d'un terme de généralisation dans le calcul du score d'un agent Contexte. Un modèle de voisinage comme Smapy a une bonne généralisation s'il fait de bonnes prédictions sur des zones de l'espace encore inexplorées. De façon géométrique, cela se traduirait par un agent Contexte de taille importante plutôt qu'un petit agent Contexte resserré autour d'un seul point qui représenterait un risque de sur-apprentissage. C'est d'ailleurs pour augmenter l'exploration, et donc la généralisation, que le mécanisme de dilatation en cas de *feedback* positif a été mis en place.

Une façon intuitive de favoriser les agents Contexte de grande taille est d'intégrer un terme dépendant de son volume dans le calcul de son score :

$$s_l = (1 - G) * N_c(c_l) + G * N_v(v_l) \tag{9.1}$$

L'ajout de ce terme implique la création de deux nouveaux paramètres internes :

— La fonction de normalisation du volume N_v pour avoir une valeur comprise entre 0 et 1 de la même manière qu'avec le niveau de confiance.

— Le terme de généralisation G entre 0 et 1, qui contrôle le poids donné au terme de généralisation par rapport à celui issu du niveau de confiance. S'il est à 0, seul le niveau de confiance est pris en compte (et donc la spécialisation des agents Contexte), tandis que s'il est à 1, seul le volume est pris en compte (et donc la généralisation des agents Contexte).

Explicabilité

La compréhension des décisions prises par les algorithmes d'apprentissage automatique est devenue un enjeu crucial dans de nombreux domaines, notamment avec l'émergence des notions de responsabilité, de transparence et d'interprétabilité des algorithmes [Phi+20]. Dans le domaine de l'analyse de la mobilité, certaines prédictions peuvent orienter des choix d'urbanisme ou de développement qui influent sur la qualité de vie des usagers. Il est donc primordial pour les décisionnaires des collectivités de pouvoir interpréter les analyses produites par des modèles d'apprentissage automatique.

Dans ce contexte, le domaine de l'explicabilité de l'IA (XAI) vise à rendre les décisions prises par les systèmes d'IA compréhensibles pour les humains [Phi+20]. En effet, des modèles abondamment utilisés tels que les réseaux de neurones profonds peuvent être souvent perçus comme des "boîtes noires" dans laquelle il est difficile de comprendre comment une décision est prise.

Un des objectifs initiaux de la thèse était d'intégrer la notion de XAI dans la solution algorithmique proposée. Bien que les données disponibles et les cas d'application rencontrés pendant le déroulement de la thèse n'aient pas permis d'explorer cette voie dans son intégralité, plusieurs pistes ont été dégagées.

Tout d'abord, une des idées de base derrière l'utilisation d'une architecture SACL était d'apporter une interprétation géométrique liée au positionnement des agents Contexte. En effet, les règles de fonctionnement de Smapy poussent les agents Contexte à se rétracter dans les zones où les frontières de classe sont les plus complexes (i.e. les zones de l'espace des variables d'entrée les moins linéaires). La dimension de rétractation, c'est-à-dire la variable d'entrée selon laquelle un agent Contexte réduit sa zone d'activation, est liée à une mauvaise prédiction ayant conduit au *feedback* négatif. Sachant cela, mon intuition est que les variables les plus sensibles, c'est-à-dire celles pour lesquelles de petites modifications entraînent le plus de changements de classe, correspondent aux dimensions ayant le plus souvent entraîné des rétractations d'agents Contexte. En d'autres termes, si les agents Contexte d'une instance de Smapy sont en moyenne plus petits selon une de leurs dimensions (par rapport à leur taille initiale selon cette même dimension), cela signifie que la variable associée joue un rôle **important** dans la classification. A l'inverse, dans le cas extrême où un agent Contexte s'étend sur l'ensemble de l'espace exploré sur une de ses dimensions, alors la variable associée à cette dimension a peu d'importance dans la classification.

Une approche proposée serait donc d'étudier les volumes moyens relatifs $\overline{v_j}$ selon chaque dimension j (par rapport au volume initial $2R$ selon chaque dimension où R est le rayon initial sur toutes les dimensions) calculés sur l'ensemble des q agents Contexte :

$$\overline{v_j} = \frac{1}{2Rq} \sum_{l=1}^{q} (r_{l,j,1} - r_{l,j,0})$$

La figure 9.2 affiche les volumes moyens relatifs obtenus sur l'instance de

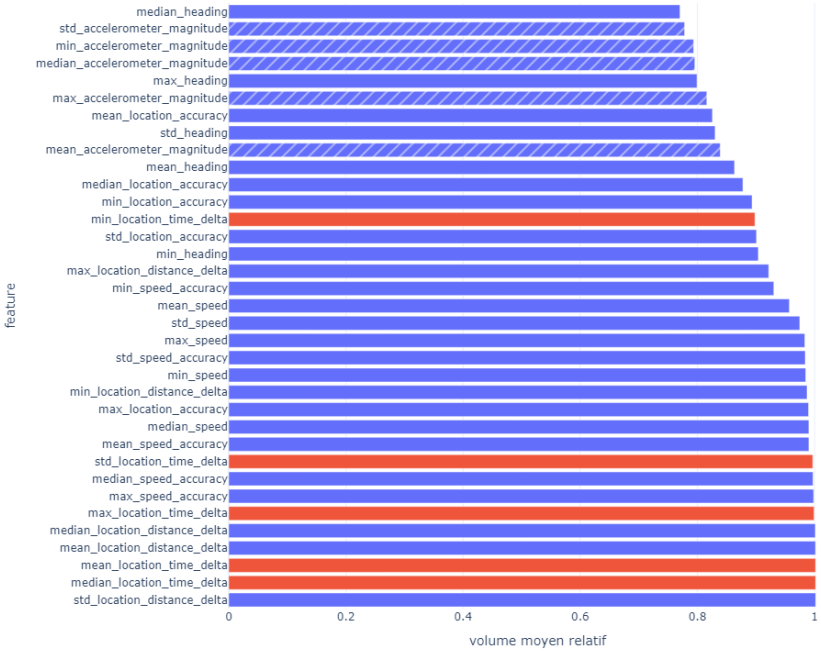


FIGURE 9.2 – Volumes moyens relatifs par rapport au volume initial des agents Contexte selon chaque dimension de l'espace des variables d'entrée, mesurés sur l'instance de Smapy entraînée dans la section 8.3 sur les données abitrack avec le scénario de découpage aléatoire. Les features d'écarts temporels sont représentées en rouge. Les features issues des données d'accéléromètre sont représentées hachurées.

Smapy entraînée dans la section 8.3 avec le scénario de découpage aléatoire. Une étude plus approfondie pourrait être menée pour comparer de faibles valeurs de volume moyen relatif à des scores d'importance élevés obtenus par exemple avec le critère de Gini sur un classifieur Random Forest.

Une autre approche possible est basée sur le fait que Smapy est un classifieur ensembliste. Dès lors, il est possible d'agréger les métriques d'importance des modèles internes des agents Contexte pour générer des scores d'importance à l'échelle du système tout entier. En particulier, dans le cas où les modèles internes sont linéaires (e.g. avec le classifieur logit), les coefficients associés à chaque variable d'entrée sont corrélés à leur importance dans le calcul des probabilités d'appartenance aux différentes classes.

Enfin, des algorithmes tels que LIME [RSG16] ou SHAP [LL17] permettent de rendre interprétables des modèles complexes. Ils pourraient être utilisés pour évaluer le degré d'explicabilité apporté par les différentes métriques proposées

pour Smapy.

Analyse de la mobilité

Cette thèse constitue la première étape d'un projet d'analyse de la mobilité plus vaste. La chaîne de traitement de détection du mode de transport peut être améliorée, et le problème de détection du motif de déplacement (axe "Pourquoi ?" de la connaissance de l'état de la mobilité) peut être résolu parallèlement.

Utilisation de données contextuelles

La littérature abonde d'exemples d'utilisation de données contextuelles pour l'analyse de la mobilité (c.f. section 2.1). Dans le problème de la détection du mode de transport, la connaissance de la géographique du réseau routier ou ferroviaire permettrait par exemple de corriger les prédictions pour les modes voiture et train respectivement. De plus, l'intégration de données relatives aux transports en commun (TC) avec les fichiers GTFS du territoire considéré permettrait d'améliorer la détection des modes TC (métro, bus, tramway). Enfin, il serait intéressant d'étudier si des données événementielles telles que l'historique météorologique permet de caractériser le choix modal des utilisateurs (e.g. par temps de pluie, on peut s'attendre à observer moins d'utilisateurs à pied). Ainsi, pour quantifier l'intérêt de l'utilisation des données contextuelles, il est possible de mener une étude comparative de la chaîne de traitement développée avec les méthodes de référence de la littérature basée sur des données contextuelles. Dans un second temps, l'intégration de données contextuelles dans la chaîne de traitement développée pourrait être testée.

Post-traitement dans la détection du mode de transport

Tout d'abord, la chaîne de traitement mise en place pour la détection du mode de transport est basée sur un groupement des points de données sans détection des arrêts (c.f. section 4.2.1.1). Dans ces cas-là, il existe des solutions de post-traitement pour améliorer la précision de classification en sortie à partir d'un algorithme de *Healing* [Guv+17] ou d'homogénéisation [Gir+22]. Ces solutions doivent être mises en places dans le cas où les données à corriger s'étendent sur une période temporelle continue (et sont donc incompatibles avec des scénarios de découpage aléatoire des données d'entraînement et de test (c.f. section 8.3). Afin de mettre en place un algorithme de post-traitement, il est préférable de disposer au préalable d'un accès à une source de données utilisateur collectées en temps réel pouvant être classifiées en ligne.

Détection des arrêts

A l'inverse, un moyen d'améliorer la précision de classification du mode de transport sans implémenter de post-traitement est l'intégration d'une étape de détection des arrêts en amont. Cette détection présente deux avantages. D'une part, les segments constitués sont mono-modaux (à condition de détecter les

segments de marche comme des transitions entre les modes) et peuvent donc être classifiés en une seule fois ce qui élimine le bruit dans les données de sortie. D'autre part, les arrêts détectés peuvent être utilisés pour détecter des activités et en déduire le but de chaque trajet. Plusieurs solutions de la littérature présentées dans la section 4.2.1.2 utilisent une segmentation basée sur la détection des arrêts.

Classification des motifs de déplacement

Un des objectifs du projet industriel dans lequel s'inscrit cette thèse est la caractérisation de l'axe "Pourquoi ?" de la connaissance de l'état de la mobilité. Cela revient à classifier le but d'un trajet à partir des données utilisateur associées. Il y a deux façons de considérer le problème :

— Classifier le but d'un trajet à partir des données associées au déplacement.

— Classifier les activités réalisées aux destinations de chaque trajet à partir du lieu de l'arrêt.

La deuxième approche est celle qui a été envisagée pendant la thèse. Elle est basée sur la connaissance de la géolocalisation des usagers à leurs destinations, mise en correspondance avec des données contextuelles de lieux d'intérêt (c.f. section 2.1.4).

FIGURE 9.3 – Exemple de détection de l'activité lors d'un arrêt avec l'algorithme à base de règles logiques développé durant la thèse.

Un algorithme de détection des activités aux arrêts basé sur des règles logiques et illustré sur la figure 9.3 a été développé pendant la thèse. Tout d'abord, une liste d'activités possibles est définie (e.g. loisirs, achats, éducation, travail, résidentiel). L'idée de l'algorithme est d'estimer pour chaque activité de cette liste la probabilité qu'elle corresponde à l'arrêt d'un usager. Pour qu'un usager effectue une activité, il faut qu'il soit à proximité d'un lieu proposant cette activité, et que le lieu en question soit ouvert. L'algorithme consiste donc à trouver quel est le lieu d'intérêt où l'usager s'est arrêté parmi un ensemble de lieux candidats. La probabilité d'activité est alors le produit de deux probabilités :

— Une **probabilité de distance**, qui désigne la probabilité que l'usager se soit arrêté sur le lieu candidat. Cette probabilité suit une loi de Rayleigh [Wei], qui correspond à la norme d'un vecteur gaussien à deux dimensions dont les coordonnées sont indépendantes. En supposant que les erreurs de géolocalisation en latitude et longitude suivent des lois normales indépendantes et centrées (i.e. d'espérance nulle), alors la norme des erreurs de géolocalisation est une variable aléatoire X de loi de Rayleigh de paramètre σ. La probabilité de distance est donc la probabilité que la distance d de l'usager au lieu candidat soit inférieure à l'erreur de géolocalisation, soit $\mathbb{P}(X > d) = 1 - \mathbb{P}(X \leq d) = 1 - F_X(d)$ où F_X désigne la fonction de répartition de X. Comme la variance de X est égale à $\frac{4-\pi}{2}\sigma^2$, il est possible de "forcer" l'algorithme à ne considérer que les lieux candidats dont la distance avec l'usager est inférieure à un paramètre de seuil δ. Pour cela, il faudrait choisir comme paramètre de la loi de Rayleigh σ_δ tel que $F_X(\delta) = 0.99$, c'est-à-dire tel que δ soit le quantile à 99% de la loi de probabilité de l'erreur de géolocalisation.

— Une **probabilité de disponibilité**, qui désigne la probabilité qu'un lieu candidat soit ouvert pendant l'intégralité de la durée de l'arrêt d'un usager. Pour cela, une base de données des horaires d'ouverture de différents lieux d'intérêt issus du dataset Yelp [Yel] a été constituée. Les activités liées aux lieux d'intérêt ont été mises en correspondance avec celles de la liste pré-établie à l'aide du modèle de *word embedding* Word2Vec [Mik+13]. Des probabilités de disponibilité de chaque activité ont été calculées sur chaque tranche horaire pour tous les jours type de la semaine. A partir de cette base de données, il est possible d'obtenir la probabilité de disponibilité d'une activité sur une période donnée en faisant le produit des probabilités de disponibilité sur l'ensemble des tranches horaires couvertes par la durée de présence de l'usager à son arrêt.

La dernière étape est d'obtenir les lieux candidats pour calculer les probabilités d'activité. L'algorithme développé fait appel à l'API OverPass d'OpenStreetMap [Ope17] via la librairie python *overpy* [Phi14] afin de récupérer les lieux d'intérêts dans un rayon de δ autour des coordonnées de l'arrêt de l'usager.

Cet algorithme n'a pas pu être validé durant la thèse pour plusieurs raisons. Tout d'abord, aucune méthode robuste de détection des arrêts n'a été mise en place durant la thèse. De plus, le dataset Yelp sur lequel s'appuie le calcul de la probabilité de disponibilité et les données de lieux d'intérêt d'OpenStreetMap ne contiennent pas les lieux associés aux activités "résidentiel" et "travail" (dans le cas de bureaux). Or ces deux activités occupent généralement la majorité de la journée des usagers. Une solution pourrait être d'étudier la régularité de présence en journée et la nuit pour déduire les lieux de domicile et de travail d'un usager afin de détecter les activités "résidentiel" et "travail" séparément des autres.

Enfin, une autre piste d'amélioration est l'intégration d'une troisième couche de probabilité provenant de données agrégées type enquêtes ménage-déplacement (EMD). Les EMD fournissent généralement des statistiques sur les activités

réalisées sur une semaine type pour une zone géographique donnée. Il serait donc possible de récupérer ces statistiques sur la zone de l'arrêt afin d'affiner les probabilités d'activité calculées par l'algorithme.

Les classifieurs sont-ils tous des agents qui s'ignorent ?

Cette thèse devrait être vue comme l'ajout d'une pierre à l'édifice des systèmes multi-agents d'apprentissage par contexte et de l'apprentissage ensembliste. En se plaçant à l'intersection entre ces deux domaines de l'intelligence artificielle, Smapy a pour ambition de répondre aux pré-requis établis pour le cas d'application de l'analyse de la mobilité. L'étude des performances d'un tel modèle selon ces pré-requis doit être approfondie, mais également la caractérisation de son comportement et de ses performances en tant que modèle d'apprentissage automatique agnostique de l'application considérée.

La typologie de paradigmes d'apprentissage automatique présentée dans ce manuscrit pose de nouvelles questions. En particulier, d'aucuns pourraient être amenés à se demander si tous les modèles d'apprentissage ensembliste, qui sont des agrégation de classifieurs indépendants, ne sont pas également des systèmes multi-agents. En réalité, les notions d'agent et de classifieur sont interchangeables dans de nombreux cas, chacune faisant référence à une entité disposant d'un certain degré d'autonomie, de son propre espace de visibilité et d'un modèle interne lui permettant de prendre une décision à partir d'une nouvelle observation.

La démocratisation de l'intelligence artificielle dans notre quotidien, à une échelle que l'on peine encore à appréhender, amène à considérer des approches d'apprentissage fédéré [Kai+21] pour satisfaire des exigences de confidentialité de la donnée et de scalabilité du système. Un futur travail de recherche permettrait d'étudier la décentralisation de modèles d'apprentissage automatique en les considérant du point de vue des systèmes multi-agents.

Bibliographie personnelle

[Fou+22a] FOUREZ, Thibault et al. « An ensemble Multi-Agent System for non-linear classification ». In : *arXiv preprint arXiv :2209.06824* (2022).

[Fou+22b] FOUREZ, Thibault et al. « How to solve a classification problem using a cooperative tiling Multi-Agent System ? » In : *International Conference on Practical Applications of Agents and Multi-Agent Systems*. Springer. 2022, p. 166-178.

[Fou+23] FOUREZ, Thibault et al. « Transport Mode Detection on GPS and accelerometer data : a temporality based workflow ». In : (2023).

Bibliographie

[Abd+19] Abduljabbar, Rusul et al. « Applications of artificial intelligence in transport : An overview ». In : *Sustainability* t. 11, n° 1 (2019), p. 189.

[Aca] Académie française. *Mobilité*. fr.

[Aha97] Aha, David W. « Lazy learning ». In : *Lazy learning*. Springer, 1997, p. 7-10.

[Ale+15] Alexander, Lauren et al. « Origin–destination trips by purpose and time of day inferred from mobile phone data ». In : *Transportation research part c : emerging technologies* t. 58 (2015), p. 240-250.

[Alo20] Alotaibi, Bandar. « Transportation mode detection by embedded sensors based on ensemble learning ». In : *IEEE Access* t. 8 (2020), p. 145552-145563.

[AB10] Altun, Kerem et Barshan, Billur. « Human activity recognition using inertial/magnetic sensor units ». In : *Human Behavior Understanding : First International Workshop, HBU 2010, Istanbul, Turkey, August 22, 2010. Proceedings 1*. Springer. 2010, p. 38-51.

[Ami20] Amini, Massih-Reza. *Principes de base en apprentissage supervisé*. 2020.

[Asg+16] Asgari, Fereshteh et al. « CT-Mapper : Mapping sparse multimodal cellular trajectories using a multilayer transportation network ». In : *Computer Communications* t. 95 (2016), p. 69-81.

[Aze22] Azencott, Chloé-Agathe. « 5. Régressions paramétriques ». FR. In : *Introduction au Machine Learning*. T. 2e éd. InfoSup. Paris : Dunod, 2022, p. 74-86.

[Bac+17] Bachir, Danya et al. « Using mobile phone data analysis for the estimation of daily urban dynamics ». In : *2017 IEEE 20th international conference on intelligent transportation systems (ITSC)*. IEEE. 2017, p. 626-632.

[Bac+19] Bachir, Danya et al. « Inferring dynamic origin-destination flows by transport mode using mobile phone data ». In : *Transportation Research Part C : Emerging Technologies* t. 101 (2019), p. 254-275.

[Bel+19] Belkin, Mikhail et al. « Reconciling modern machine-learning practice and the classical bias–variance trade-off ». In : *Proceedings of the National Academy of Sciences* t. 116, n° 32 (2019), p. 15849-15854.

[Ber+19] Berner, Christopher et al. « Dota 2 with large scale deep reinforcement learning ». In : *arXiv preprint arXiv :1912.06680* (2019).

[BSG19] Bijjahalli, Suraj, Sabatini, Roberto et Gardi, Alessandro. « GNSS performance modelling and augmentation for urban air mobility ». In : *Sensors* t. 19, n° 19 (2019), p. 4209.

[BLVO13] Biljecki, Filip, Ledoux, Hugo et Van Oosterom, Peter. « Transportation mode-based segmentation and classification of movement trajectories ». In : *International Journal of Geographical Information Science* t. 27, n° 2 (2013), p. 385-407.

[Boe+15] Boes, Jérémy et al. « The self-adaptive context learning pattern : Overview and proposal ». In : *International and Interdisciplinary Conference on Modeling and Using Context*. Springer. 2015, p. 91-104.

[Bol+12] Bolbol, Adel et al. « Inferring hybrid transportation modes from sparse GPS data using a moving window SVM classification ». In : *Computers, Environment and Urban Systems* t. 36, n° 6 (2012), p. 526-537.

[Bre84] Breiman, Leo. *Classification and regression trees*. Routledge, 1984.

[Bre96] Breiman, Leo. « Bagging predictors ». In : *Machine learning* t. 24 (1996), p. 123-140.

[Bre01] Breiman, Leo. « Random forests ». In : *Machine learning* t. 45, n° 1 (2001), p. 5-32.

[Cal+13] Calabrese, Francesco et al. « Understanding individual mobility patterns from urban sensing data : A mobile phone trace example ». In : *Transportation research part C : emerging technologies* t. 26 (2013), p. 301-313.

[Cap+03] Capera, Davy et al. « The AMAS theory for complex problem solving based on self-organizing cooperative agents ». In : *WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies : Infrastructure for Collaborative Enterprises, 2003.* IEEE. 2003, p. 383-388.

[CN+09] Carmo Nicoletti, Maria do et al. « Constructive neural network algorithms for feedforward architectures suitable for classification tasks ». In : *Constructive neural networks*. Springer, 2009, p. 1-23.

[Car+18] Carpineti, Claudia et al. « Custom dual transportation mode detection by smartphone devices exploiting sensor diversity ». In : *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE. 2018, p. 367-372.

[CBM14] Chen, Cynthia, Bian, Ling et Ma, Jingtao. « From traces to trajectories : How well can we guess activity locations from mobile phone traces ? » In : *Transportation Research Part C : Emerging Technologies* t. 46 (2014), p. 326-337.

[CXC+22] Chen, Jiatao, Xiong, Chen, Cai, Ming et al. « A Travel Mode Identification Framework Based on Cellular Signaling Data ». In : *Mobile Information Systems* t. 2022 (2022).

[Che+17] Chen, Ke-Yu et al. « Mago : Mode of transport inference using the hall-effect magnetic sensor and accelerometer ». In : *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* t. 1, n° 2 (2017), p. 1-23.

[Cia+22] Ciampi, Luca et al. « Multi-camera vehicle counting using edge-AI ». In : *Expert Systems with Applications* t. 207 (2022), p. 117929.

[Cor19] Corbacho, Fernando J. « Towards self-constructive artificial intelligence : Algorithmic basis (Part I) ». In : *arXiv preprint arXiv :1901.01989* (2019).

[CH67] Cover, Thomas et Hart, Peter. « Nearest neighbor pattern classification ». In : *IEEE transactions on information theory* t. 13, n° 1 (1967), p. 21-27.

[CS03] Crammer, Koby et Singer, Yoram. « Ultraconservative online algorithms for multiclass problems ». In : *Journal of Machine Learning Research* t. 3, n° Jan (2003), p. 951-991.

[Cra+06] Crammer, Koby et al. « Online passive aggressive algorithms ». In : (2006).

[CST+00] Cristianini, Nello, Shawe-Taylor, John et al. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[DS17] Dalumpines, Ron et Scott, Darren M. « Making mode detection transferable : extracting activity and travel episodes from GPS data using the multinomial logit model and Python ». In : *Transportation planning and technology* t. 40, n° 5 (2017), p. 523-539.

[Dat21] Dato, Bruno. « Lifelong Learning by Endogenous Feedback, application to a Robotic System ». Thèse de doct. Université Toulouse 3-Paul Sabatier, 2021.

[Del+12] Delafontaine, Matthias et al. « Analysing spatiotemporal sequences in Bluetooth tracking data ». In : *Applied Geography* t. 34 (2012), p. 659-668.

[DL+14] Di Lorenzo, Giusy et al. « AllAboard : Visual exploration of cellphone mobility data to optimise public transport ». In : *Proceedings of the 19th international conference on Intelligent User Interfaces*. 2014, p. 335-340.

[DRP12] Dioşan, Laura, Rogozan, Alexandrina et Pecuchet, Jean-Pierre. « Improving classification performance of support vector machine by genetically optimising kernel shape and hyper-parameters ». In : *Applied Intelligence* t. 36 (2012), p. 280-294.

[Don+15] Dong, Honghui et al. « Traffic zone division based on big data from mobile phone base stations ». In : *Transportation Research Part C : Emerging Technologies* t. 58 (2015), p. 278-291.

[Don+20] Dong, Xibin et al. « A survey on ensemble learning ». In : *Frontiers of Computer Science* t. 14 (2020), p. 241-258.

[Doz23] Dozias, Arthur. « La concurrence dans le marché français des communications électroniques ». fr. In : *Trésor-Eco* t. 321 (2023), p. 4-8.

[Ely+23] Elyoussoufi, Ayman et al. « The Relationships between Adverse Weather, Traffic Mobility, and Driver Behavior ». In : *Meteorology* t. 2, n° 4 (2023), p. 489-508.

[FT13] Feng, Tao et Timmermans, Harry JP. « Transportation mode recognition using GPS and accelerometer data ». In : *Transportation Research Part C : Emerging Technologies* t. 37 (2013), p. 118-130.

[FR+13] Fontenla-Romero, Óscar et al. « Online machine learning ». In : *Efficiency and Scalability Methods for Computational Intellect*. IGI global, 2013, p. 27-54.

[For65] Forgy, Edward W. « Cluster analysis of multivariate data : efficiency versus interpretability of classifications ». In : *biometrics* t. 21 (1965), p. 768-769.

[Fou+22a] Fourez, Thibault et al. « An ensemble Multi-Agent System for non-linear classification ». In : *arXiv preprint arXiv :2209.06824* (2022).

[Fou+22b] Fourez, Thibault et al. « How to solve a classification problem using a cooperative tiling Multi-Agent System ? » In : *International Conference on Practical Applications of Agents and Multi-Agent Systems*. Springer. 2022, p. 166-178.

[Fou+23] Fourez, Thibault et al. « Transport Mode Detection on GPS and accelerometer data : a temporality based workflow ». In : (2023).

[FBN19] Furberg, Dorothy, Ban, Yifang et Nascetti, Andrea. « Monitoring of urbanization and analysis of environmental impact in Stockholm with Sentinel-2A and SPOT-5 multispectral data ». In : *Remote Sensing* t. 11, n° 20 (2019), p. 2408.

[Gao+20] Gao, Liangpeng et al. « Effectiveness of Public Transport Networks in Motorized Mode Detection : A Case Study of a Planning Survey in Nanjing ». In : *2020 IEEE 5th International Conference on Intelligent Transportation Engineering (ICITE)*. IEEE. 2020, p. 37-43.

[Gil+18] Gilpin, Leilani H et al. « Explaining explanations : An overview of interpretability of machine learning ». In : *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*. IEEE. 2018, p. 80-89.

[Gir+22] Giri, Santosh et al. « Application of machine learning to predict transport modes from GPS, accelerometer, and heart rate data ». In : *International Journal of Health Geographics* t. 21, n° 1 (2022), p. 19.

[Gjo+18] Gjoreski, Hristijan et al. « The university of sussex-huawei locomotion and transportation dataset for multimodal analytics with mobile devices ». In : *IEEE Access* t. 6 (2018), p. 42592-42604.

[Gon+12] Gong, Hongmian et al. « A GPS/GIS method for travel mode detection in New York City ». In : *Computers, Environment and Urban Systems* t. 36, n° 2 (2012), p. 131-139.

[GHB08] Gonzalez, Marta C, Hidalgo, Cesar A et Barabasi, Albert-Laszlo. « Understanding individual human mobility patterns ». In : *nature* t. 453, n° 7196 (2008), p. 779-782.

[Gon+10] Gonzalez, Paola A et al. « Automating mode detection for travel behaviour analysis by using global positioning systems-enabled mobile phones and neural networks ». In : *IET Intelligent Transport Systems* t. 4, n° 1 (2010), p. 37-49.

[Goo23a] Google. *General Transit Feed Specification Reference*. fr. Mars 2023.

[Goo23b] Google. *Places API*. fr-x-mtfrom-en. Nov. 2023.

[Goz+11] Gozick, Brandon et al. « Magnetic maps for indoor navigation ». In : *IEEE Transactions on Instrumentation and Measurement* t. 60, n° 12 (2011), p. 3883-3891.

[Guo+22] Guo, Lin et al. « Convolutional neural network-based travel mode recognition based on multiple smartphone sensors ». In : *Applied Sciences* t. 12, n° 13 (2022), p. 6511.

[Guv+17] Guvensan, M Amac et al. « A novel segment-based approach for improving classification performance of transport mode detection ». In : *Sensors* t. 18, n° 1 (2017), p. 87.

[Hag+10] Haghani, Ali et al. « Data collection of freeway travel time ground truth with bluetooth sensors ». In : *Transportation Research Record* t. 2160, n° 1 (2010), p. 60-68.

[HER23] HERE. *Places (Search) API - Developer Guide*. Nov. 2023.

[HER24] HERE. *HERE Map Content - Schema*. Fév. 2024.

[HK70] Hoerl, Arthur E et Kennard, Robert W. « Ridge regression : Biased estimation for nonorthogonal problems ». In : *Technometrics* t. 12, n° 1 (1970), p. 55-67.

[Hol+04] Holmes, Michael et al. « Schema learning : Experience-based construction of predictive action models ». In : *Advances in Neural Information Processing Systems* t. 17 (2004).

[Hua+18] Huang, Zhiren et al. « Modeling real-time human mobility based on mobile phone and transportation data fusion ». In : *Transportation research part C : emerging technologies* t. 96 (2018), p. 251-269.

[Inm49] Inman, James. *Navigation and nautical astronomy : For the use of British seamen*. F. et J. Rivington, 1849.

[Ins16] Insee. *Définition - IRIS*. Oct. 2016.

[Ins24a] Insee. *Le répertoire Sirene et sa diffusion*. Fév. 2024.

[Ins24b] Institut national de l'information géographique et forestière. *BD TOPO®*. Fév. 2024.

[IRI19] IRIT. *Action « Territoire d'innovation » du PIA3 : le projet VILAGIL retenu !* 2019.

[Irs+21] Irshaid, Hafez et al. « User activity and trip recognition using spatial positioning system data by integrating the geohash and gis approaches ». In : *Transportation research record* t. 2675, n° 4 (2021), p. 391-405.

[IG20] Iskanderov, Jemshit et Guvensan, M Amac. « Breaking the limits of transportation mode detection : Applying deep learning approach with knowledge-based features ». In : *IEEE Sensors Journal* t. 20, n° 21 (2020), p. 12871-12884.

[JR14] Jahangiri, Arash et Rakha, Hesham. « Developing a support vector machine (SVM) classifier for transportation mode identification by using mobile phone sensor data ». In : *Transportation Research Board 93rd Annual Meeting*. T. 14. 2014, p. 1442.

[JFG17] Jiang, Shan, Ferreira, Joseph et Gonzalez, Marta C. « Activity-based human mobility patterns inferred from mobile phone data : A case study of Singapore ». In : *IEEE Transactions on Big Data* t. 3, n° 2 (2017), p. 208-219.

[Kai+21] Kairouz, Peter et al. « Advances and open problems in federated learning ». In : *Foundations and Trends® in Machine Learning* t. 14, n° 1–2 (2021), p. 1-210.

[Kho+16] Khodabandelou, Ghazaleh et al. « Population estimation from mobile network traffic metadata ». In : *2016 IEEE 17th international symposium on a world of wireless, mobile and multimedia networks (WoWMoM)*. IEEE. 2016, p. 1-9.

[Kho+18] Khodabandelou, Ghazaleh et al. « Estimation of static and dynamic urban populations with mobile network metadata ». In : *IEEE Transactions on Mobile Computing* t. 18, n° 9 (2018), p. 2034-2047.

[KGQ12] Kim, Ji-Sun, Gračanin, Denis et Quek, Francis. « Sensor-fusion walking-in-place interaction technique using mobile devices ». In : *2012 IEEE Virtual Reality Workshops (VRW)*. IEEE. 2012, p. 39-42.

[Kot13] Kotsiantis, Sotiris B. « Decision trees : a recent overview ». In : *Artificial Intelligence Review* t. 39 (2013), p. 261-283.

[Kuo+14] Kuo, Ye-Sheng et al. « Luxapose : Indoor positioning with mobile phones and visible light ». In : *Proceedings of the 20th annual international conference on Mobile computing and networking*. 2014, p. 447-458.

[Kys14] Kysela, Jiří. « Comparison of web applications geolocation services ». In : *2014 IEEE 15th International Symposium on Computational Intelligence and Informatics (CINTI)*. IEEE. 2014, p. 449-453.

[LRT14] Lakshminarayanan, Balaji, Roy, Daniel M et Teh, Yee Whye. « Mondrian forests : Efficient online random forests ». In : *Advances in neural information processing systems* t. 27 (2014).

[LLL22] Lee, Sungyong, Lee, Jinsung et Lee, Kyunghan. « DeepVehicleSense : An Energy-Efficient Transportation Mode Recognition Leveraging Staged Deep Learning Over Sound Samples ». In : *IEEE Transactions on Mobile Computing* (2022).

[Li+20] Li, Linchao et al. « Coupled application of generative adversarial networks and conventional neural networks for travel mode detection using GPS data ». In : *Transportation Research Part A : Policy and Practice* t. 136 (2020), p. 282-292.

[Li+18] Li, Zufen et al. « Identifying temporal and spatial characteristics of residents' trips from cellular signaling data : Case study of Beijing ». In : *Transportation research record* t. 2672, n° 42 (2018), p. 81-90.

[LL17] Lundberg, Scott M et Lee, Su-In. « A unified approach to interpreting model predictions ». In : *Advances in neural information processing systems* t. 30 (2017).

[Mik+13] Mikolov, Tomas et al. « Efficient estimation of word representations in vector space ». In : *arXiv preprint arXiv :1301.3781* (2013).

[MCB18] Molnar, Christoph, Casalicchio, Giuseppe et Bischl, Bernd. « iml : An R package for interpretable machine learning ». In : *Journal of Open Source Software* t. 3, n° 26 (2018), p. 786.

[Mon+22] Monje, Leticia et al. « Deep learning XAI for bus passenger forecasting : A use case in Spain ». In : *Mathematics* t. 10, n° 9 (2022), p. 1428.

[Naw+20a] Nawaz, Asif et al. « Convolutional LSTM based transportation mode learning from raw GPS trajectories ». In : *IET Intelligent Transport Systems* t. 14, n° 6 (2020), p. 570-577.

[Naw+20b] Nawaz, Asif et al. « Mode Inference using enhanced Segmentation and Pre-processing on raw Global Positioning System data ». In : *Measurement and Control* t. 53, n° 7-8 (2020), p. 1144-1158.

[NW72] Nelder, John Ashworth et Wedderburn, Robert WM. « Generalized linear models ». In : *Journal of the Royal Statistical Society : Series A (General)* t. 135, n° 3 (1972), p. 370-384.

[NA20] Nguyen, Minh Hieu et Armoogum, Jimmy. « Hierarchical process of travel mode imputation from GPS data in a motorcycle-dependent area ». In : *Travel behaviour and society* t. 21 (2020), p. 109-120.

[NWC18] Ni, Linglin, Wang, Xiaokun Cara et Chen, Xiqun Michael. « A spatial econometric model for travel flow analysis and real-world applications with massive mobile phone data ». In : *Transportation research part C : emerging technologies* t. 86 (2018), p. 510-526.

[Nic+10] Nick, Theresa et al. « Classifying means of transportation using mobile sensor data ». In : *The 2010 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2010, p. 1-6.

[Nie16] Nielsen, Frank. « Hierarchical Clustering ». In : fév. 2016, p. 195-211.

[Ope17] OpenStreetMap contributors. *Planet dump retrieved from https ://planet.osm.org*. 2017.

[Pap+15] Pappalardo, Luca et al. « Using big data to study the link between human mobility and socio-economic development ». In : *2015 IEEE International Conference on Big Data (Big Data)*. IEEE. 2015, p. 871-878.

[Ped+11] Pedregosa, F. et al. « Scikit-learn : Machine Learning in Python ». In : *Journal of Machine Learning Research* t. 12 (2011), p. 2825-2830.

[Phi14] PhiBo. *Python Overpass API*. 2014.

[Phi+20] Phillips, P Jonathon et al. « Four principles of explainable artificial intelligence ». In : *Gaithersburg, Maryland* t. 18 (2020).

[Ras+15] Rasmussen, Thomas Kjær et al. « Improved methods to deduct trip legs and mode from travel surveys using wearable GPS devices : A case study from the Greater Copenhagen area ». In : *Computers, Environment and Urban Systems* t. 54 (2015), p. 301-313.

[Red+10] Reddy, Sasank et al. « Using mobile phones to determine transportation modes ». In : *ACM Transactions on Sensor Networks (TOSN)* t. 6, n° 2 (2010), p. 1-27.

[RSG16] Ribeiro, Marco Tulio, Singh, Sameer et Guestrin, Carlos. « " Why should i trust you ?" Explaining the predictions of any classifier ». In : *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2016, p. 1135-1144.

[ris21] risques l'environnement, la mobilité et l'aménagement Centre d'études et d'expertise sur les, éd. *Les enquêtes mobilité certifiées Cerema, EMC² : principes méthodologiques*. fr. Références. Lyon : Cerema Territoires et villes, 2021.

[RT+08] Roy, Daniel M, Teh, Yee Whye et al. « The Mondrian Process. » In : *NIPS*. T. 21. 2008.

[RHW86] Rumelhart, David E, Hinton, Geoffrey E et Williams, Ronald J. « Learning representations by back-propagating errors ». In : *nature* t. 323, n° 6088 (1986), p. 533-536.

[Sam59] Samuel, Arthur L. « Some studies in machine learning using the game of checkers ». In : *IBM Journal of research and development* t. 3, n° 3 (1959), p. 210-229.

[San+14] Sankaran, Kartik et al. « Using mobile phone barometer for low-power transportation context detection ». In : *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*. 2014, p. 191-205.

[SB+17] Sauerländer-Biebl, Anke et al. « Evaluation of a transport mode detection using fuzzy rules ». In : *Transportation research procedia* t. 25 (2017), p. 591-602.

[Sem+17] Semanjski, Ivana et al. « Spatial context mining approach for transport mode recognition from mobile sensed big data ». In : *Computers, Environment and Urban Systems* t. 66 (2017), p. 38-52.

[SH16] Shafique, Muhammad Awais et Hato, Eiji. « Travel mode detection with varying smartphone data collection frequencies ». In : *Sensors* t. 16, n° 5 (2016), p. 716.

[Shu+15] Shu, Yuanchao et al. « Magicol : Indoor localization using pervasive magnetic field and opportunistic WiFi sensing ». In : *IEEE Journal on Selected Areas in Communications* t. 33, n° 7 (2015), p. 1443-1457.

[SN+16] Siła-Nowicka, Katarzyna et al. « Analysis of human mobility patterns from GPS trajectories and contextual information ». In : *International Journal of Geographical Information Science* t. 30, n° 5 (2016), p. 881-906.

[Sil+16] Silver, David et al. « Mastering the game of Go with deep neural networks and tree search ». In : *nature* t. 529, n° 7587 (2016), p. 484-489.

[Sta01] Stanton, Jeffrey M. « Galton, Pearson, and the peas : A brief history of linear regression for statistics instructors ». In : *Journal of Statistics Education* t. 9, n° 3 (2001).

[Ste+11] Stenneth, Leon et al. « Transportation mode detection using mobile phones and GIS information ». In : *Proceedings of the 19th ACM SIGSPATIAL international conference on advances in geographic information systems*. 2011, p. 54-63.

[SSR10] Storms, William, Shockley, Jeremiah et Raquet, John. « Magnetic field navigation in an indoor environment ». In : *2010 Ubiquitous Positioning Indoor Navigation and Location Based Service*. IEEE. 2010, p. 1-10.

[Sze22] Szepesvári, Csaba. *Algorithms for reinforcement learning*. Springer Nature, 2022.

[Tib96] Tibshirani, Robert. « Regression shrinkage and selection via the lasso ». In : *Journal of the Royal Statistical Society : Series B (Methodological)* t. 58, n° 1 (1996), p. 267-288.

[Too+15] Toole, Jameson L et al. « The path most traveled : Travel demand estimation using big data resources ». In : *Transportation Research Part C : Emerging Technologies* t. 58 (2015), p. 162-177.

[VGR20] Vakili, Meysam, Ghamsari, Mohammad et Rezaei, Masoumeh. « Performance analysis and comparison of machine and deep learning algorithms for IoT data classification ». In : *arXiv preprint arXiv :2001.09636* (2020).

[Val+23] Vallée, Julie et al. *Mobiliscope, a geovisualization platform to explore cities around the clock*. Avr. 2023.

[VEH20] Van Engelen, Jesper E et Hoos, Holger H. « A survey on semi-supervised learning ». In : *Machine learning* t. 109, n° 2 (2020), p. 373-440.

[VMG] Varoquaux, Gaël, Müller, Andreas et Grobler, Jaques. *Classifier comparison*.

[Ver+12] Versichele, Mathias et al. « The use of Bluetooth for analysing spatiotemporal dynamics of human movement at mass events : A case study of the Ghent Festivities ». In : *Applied Geography* t. 32, n° 2 (2012), p. 208-220.

[Ver16] Verstaevel, Nicolas. « Self-organization of robotic devices through demonstrations ». Thèse de doctorat dirigée par Gleizes, Marie-Pierre et Régis, Christine Intelligence artificielle Toulouse 3 2016. Thèse de doct. 2016.

[Ver+15] Verstaevel, Nicolas et al. « Principles and experimentations of self-organizing embedded agents allowing learning from demonstration in ambient robotic ». In : *Procedia Computer Science* t. 52 (2015), p. 194-201.

[WGJ17] Wang, Bao, Gao, Linjie et Juan, Zhicai. « Travel mode detection using GPS data and socioeconomic attributes based on a random forest classifier ». In : *IEEE Transactions on Intelligent Transportation Systems* t. 19, n° 5 (2017), p. 1547-1558.

[Wan+13] Wang, Ming-Heng et al. « Estimating dynamic origin-destination data and travel demand using cell phone network data ». In : *International Journal of Intelligent Transportation Systems Research* t. 11 (2013), p. 76-86.

[WYM18] Wang, Xuyu, Yu, Zhitao et Mao, Shiwen. « DeepML : Deep LSTM for indoor localization with smartphone magnetic and light sensors ». In : *2018 IEEE international conference on communications (ICC)*. IEEE. 2018, p. 1-6.

[Wan+18] Wang, Yihong et al. « Understanding travellers' preferences for different types of trip destination based on mobile internet usage data ». In : *Transportation Research Part C : Emerging Technologies* t. 90 (2018), p. 247-259.

[Wei] Weisstein, Eric W. *Rayleigh Distribution*. en. Text. Publisher : Wolfram Research, Inc.

[WNB12] Widhalm, Peter, Nitsche, Philippe et Brändie, Norbert. « Transport mode detection with realistic smartphone sensor data ». In : *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*. IEEE. 2012, p. 573-576.

[XCZ19] Xiao, Guangnian, Cheng, Qin et Zhang, Chunqin. « Detecting travel modes using rule-based classification system and Gaussian process classifier ». In : *IEEE Access* t. 7 (2019), p. 116741-116752.

[Xia+17] Xiao, Zhibin et al. « Identifying different transportation modes from trajectory data using tree-based ensemble classifiers ». In : *ISPRS International Journal of Geo-Information* t. 6, n° 2 (2017), p. 57.

[XZH15] Xu, Qiang, Zheng, Rong et Hranilovic, Steve. « IDyLL : Indoor localization using inertial and light sensors on smartphones ». In : *Proceedings of the 2015 ACM international joint conference on pervasive and ubiquitous computing*. 2015, p. 307-318.

[Yan+18] Yang, Xue et al. « A review of GPS trajectories classification based on transportation mode ». In : *Sensors* t. 18, n° 11 (2018), p. 3741.

[Yan+15] Yang, Zhice et al. « Wearables can afford : Light-weight indoor positioning with visible light ». In : *Proceedings of the 13th annual international conference on mobile systems, applications, and services*. 2015, p. 317-330.

[Yel] Yelp. *Yelp Dataset*.

[YM19] Yuan, Yihong et Mills, David. « Exploring Urban Dynamics from Bluetooth Tracking Data : A Case Study of Austin, Texas ». In : *Proceedings of the ICA*. T. 2. Copernicus Publications Göttingen, Germany. 2019, p. 153.

[ZZ16] Zhang, Chi et Zhang, Xinyu. « LiTell : Robust indoor localization using unmodified light fixtures ». In : *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*. 2016, p. 230-242.

[Zha+17] Zhao, Zenghua et al. « NaviLight : Indoor localization and navigation under arbitrary lights ». In : *IEEE INFOCOM 2017-IEEE Conference on Computer Communications*. IEEE. 2017, p. 1-9.

[Zhe+08] Zheng, Yu et al. « Learning transportation mode from raw gps data for geographic applications on the web ». In : *Proceedings of the 17th international conference on World Wide Web*. 2008, p. 247-256.

[Zhe+09] Zheng, Yu et al. « Mining interesting locations and travel sequences from GPS trajectories ». In : *Proceedings of the 18th international conference on World wide web*. 2009, p. 791-800.

[Zho+16] Zhong, Gang et al. « Characterizing passenger flow for a transportation hub based on mobile phone data ». In : *IEEE Transactions on Intelligent Transportation Systems* t. 18, n° 6 (2016), p. 1507-1518.

[ZYS16] Zhou, Xiaolu, Yu, Wei et Sullivan, William C. « Making pervasive sensing possible : Effective travel mode sensing based on smartphones ». In : *Computers, Environment and Urban Systems* t. 58 (2016), p. 52-59.

[ZH05] Zou, Hui et Hastie, Trevor. « Regularization and variable selection via the elastic net ». In : *Journal of the royal statistical society : series B (statistical methodology)* t. 67, n° 2 (2005), p. 301-320.

Liste des algorithmes

Liste des figures

Liste des tableaux

Acronymes

ACP Analyse en composantes principales. 61, 67

AMAS Adaptative multi-agent system (système multi-agent adaptatif). v, vii, xvi, 13, 48, 51, 82, 145

ANN Artificial neural network (réseau de neurones artificiels). 45, 62, 109, 110, 114, 117, 118, 128, 131, 133, 135

ANOVA Analysis of variance (analyse de la variance). 61, 65

BTS Base transceiver station (station de transmission de base). 29, 30

CAH Classification ascendante hiérarchique. 43

CART Classification and regression tree (arbre de décision de régression et de classification). 42, 43, 61

CDR Call detail records (statitiques d'appel). 30

ELLSA Endogenous Lifelong Learner by Self-Adaptation (apprentissage permanent et endogène par auto-adaptation). 49, 74, 81, 82, 83, 145

EMD Enquête ménage-déplacement. 25, 34, 153

GNSS Global navigation satellite system (système de positionnement par satellites). 29, 31, 53, 54, 55, 56, 57, 124

GPS Global positioning system (géo-positionnement par satellite). 28, 29, 56, 101, 102, 105, 109, 111, 112, 114, 115, 119

GTFS General Transit Feed Specification (spécification générale pour les flux relatifs aux transports en commun). 21, 22, 151, 174

HAR Human activity recognition (reconnaissance de l'activité humaine). 55, 57

IA Intelligence artificielle. 149

IMU Inertial measurement unit (unité de mesure inertielle). 32

IRIS Ilots Regroupés pour l'Information Statistique. 22

KNN K-nearest neighbors (k plus proches voisins). 42, 43, 109, 110, 114, 117, 118, 128, 131, 133, 135

LAU Location area update (mise à jour de la localisation mobile). 30

LSTM Long short-term memory (réseau récurrent à mémoire court et long terme). 60, 62

OD Origine-destination. 25, 29, 30

PA Passive aggressive (algorithme passif agressif). 40, 93, 94, 96

POI Point of interest (point d'intérêt). 23, 55

RF Random Forest (forêt aléatoire). 61, 66, 67, 109, 110, 114, 117, 118, 128, 131, 133, 135

SACL Self-adaptative context learning (apprentissage par contexte auto-adaptatif). v, vii, 48, 49, 50, 51, 74, 75, 84, 85, 145, 146, 149, 174

SIG Système d'information géographique. 20, 23, 53, 55, 61, 104

SMA Système multi-agents. xiv, 37, 38, 40, 42, 44, 46, 48, 49, 50, 95, 96, 98, 174

Smapy Système Mult-Agents ensembliste d'apprentissage par contexte développé en PYthon. xv, xvi, xvii, 1, 2, 3, 13, 51, 73, 74, 75, 76, 78, 80, 81, 82, 83, 84, 86, 87, 88, 89, 90, 91, 93, 94, 95, 96, 97, 98, 101, 108, 110, 114, 116, 117, 118, 120, 127, 128, 131, 132, 133, 134, 135, 136, 138, 144, 145, 146, 147, 148, 149, 150, 151, 174, 175, 176, 177, 178, 179

SNC Situation de non coopération. 49, 80, 82, 83, 84, 85, 87, 137, 173

SVM Support vector machine (machine à vecteurs de support). 41, 61, 93, 94, 96, 109, 110, 114, 117, 118, 128, 131, 133, 134, 135

TC Transports en commun. 20, 55, 57, 125, 151

TMD Transport mode detection (détection du mode de transport). xvi, 19, 34, 37, 53, 56, 57, 58, 59, 61, 62, 101, 102, 109, 112, 113, 117, 118, 119, 120, 123, 128, 133, 136, 138, 139, 174, 175, 179

VP Véhicules privés. 57, 62

XAI eXplainable artificial intelligence (intelligence artificielle explicable). 149

Notations mathématiques

a_i Vecteur d'accélération associé au point d'observation d'indice i. 32, 33

β_0 Intercept dans une régression linéaire. 40

β_j Coefficient associé à la j-ième variable explicative dans une régression linéaire. 40

c_l Niveau de confiance de l'agent Contexte l. 75, 76, 80, 81, 148

c^* Niveau de confiance critique de réinitialisation (fonction du facteur de réinitialisation Z et du nombre d'activation t). 79, 80, 81

$d_{i,i'}$ Distance entre les coordonnées géographiques associées aux points d'observation d'indices i et i'. 27, 28

f Fonction objectif reliant des observations X_i à leurs classes respectives y_i. 38, 61

$\hat{f}$ Modèle d'apprentissage automatique d'une fonction objectif f. 38, 39, 40, 61

g_i Vecteur de vitesses de rotation associé au point d'observation d'indice i. 33

h_i Altitude associée au point d'observation d'indice i. 27

$\mathcal{H}_l$ Historique d'activation d'un agent Contexte l (i.e. ensemble des itérations dans lesquelles l'agent Contexte l a été activé). 75, 76

$i : 1 \mapsto n$ (Indice d'un point d'observation parmi les n points d'observations). 26, 27, 28, 32, 33, 38, 39, 40, 61, 75, 76, 107, 108

I Ensemble des indices des points d'observations X_i (et de leurs temps d'observation t_i). 26, 27, 32

I_a Ensemble des indices des points d'observations X_i possédant un vecteur d'accélération (et de leurs temps d'observation t_i). 32, 33

I_p Ensemble des indices des points d'observations X_i possédant des coordonnées géographiques (et de leurs temps d'observation t_i). 27, 28

$j : 1 \mapsto p$ (Indice d'une variable d'entrée parmi les p variables d'entrée). 40, 75, 76, 107, 149

$k : 1 \mapsto .$ (Indice d'une période d'observation). 106

$l : 1 \mapsto q$ (Indice d'un agent Contexte l parmi les q agents Contexte d'une instance de Smapy). 75, 76, 79, 80, 81, 84

λ_i Longitude associée au point d'observation d'indice i. 27

$m : . \mapsto m$ (Nombre de classes dans un problème de classification supervisée). 40, 41

n_a^k Nombre de points d'observation possédant un vecteur d'accélération dans une période d'observation $\mathcal{T}_k$. 107

n^k Nombre de points d'observation dans une période d'observation $\mathcal{T}_k$. 107

n_p^k Nombre de points d'observation possédant des coordonnées géographiques dans une période d'observation $\mathcal{T}_k$. 107

o_{l_1,l_2} Indice de superposition des agents Contexte l_1 et l_2. 78, 79, 83, 85

φ_i Latitude associée au point d'observation d'indice i. 27

p_i Vecteur de coordonnées géographiques associé au point d'observation d'indice i. 27, 28

$r_{l,j,0}$ Borne inférieure de la frontière de la zone d'activation de l'agent Contexte l selon la dimension j. 75, 76, 149

$r_{l,j,1}$ Borne supérieure de la frontière de la zone d'activation de l'agent Contexte l selon la dimension j. 75, 76, 149

$\mathbb{R}^p$ Ensemble des réels à p dimensions. 74, 75, 80, 81, 84, 85

s_l Score de l'agent Contexte l. 76, 80, 81, 148

θ_i Relèvement associé au point d'observation d'indice i. 28

t_i Temps associé au point d'observation d'indice i. 26, 27, 28, 104, 106, 107, 108

$\mathcal{T}_k$ Période d'observation d'indice k, durant laquelle des mesures sont collectées en continu sur un utilisateur. 103, 106, 107

v_i Vitesse instantanée au point d'observation d'indice i. 27, 28

$\bar{v}_{i,i'}$ Vitesse moyenne entre les points d'observation d'indices i et i'. 27

$\overline{v_j}$ Volume moyen relatif (par rapport au volume initial) des agents Contexte selon la dimension j. 149

v_l Volume de l'agent Contexte l. 75, 148

X_i Point d'observation (de dimension p). 38, 39, 40, 61, 74, 75, 76, 77, 78, 80, 81, 82, 84, 85, 174, 175

X_i^j j-ième coordonnée du point d'observation X_i. 40

y_i Variable expliquée (ou classe) correspondant au point d'observation X_i. 38, 39, 40, 61, 74, 75, 76, 81

$\hat{y}_i$ Prédiction de la variable expliquée (ou classe) y_i avec un modèle d'apprentissage automatique $\hat{f}$. 38, 40

z_l Nombre d'activations de l'agent Contexte l. 80, 81

Paramètres de Smapy

α Facteur de dilatation/rétractation (réel positif). 76, 77, 79, 81, 87, 95, 110, 128, 129, 174, 175

F_- Feedback négatif (entier positif). 75, 76, 79, 80, 81, 87, 95, 110, 128

F_+ Feedback positif (entier positif). 75, 76, 79, 80, 81, 87, 95, 110, 128

G Terme de généralisation dans le calcul du score des agents Contexte (réel entre 0 et 1). 148

M Marge minimale des agents Contexte (réel positif). 83, 84, 87, 95, 110, 128

N_c Fonction de normalisation du niveau de confiance (fonction réelle croissante). 76, 87, 95, 110, 128, 148

N_v Fonction de normalisation du volume (fonction réelle croissante). 148

O Seuil de superposition pour l'absorption (réel positif). 83, 85, 87, 95, 110, 128

E Exclusion de point (booléen). 78, 79, 81, 87, 95, 110, 128

R Rayon initial des agents Contexte (réel positif). 82, 87, 95, 110, 128, 149

Z Facteur de réinitialisation (réel positif). 79, 80, 81, 87, 95, 110, 128