PHD THESIS

In Partial Fulfilment of the Requirements for the
Degree of Doctor of Philosophy from Sorbonne University

Specialization: Data Science

Semantic Data Driven Approach for Merchandizing Optimization

Amine DADOUN

Defended on 21/10/2021 before a committee composed of:

Reviewer	**Armelle Brun**, Universite de Lorraine, France
Reviewer	**Oscar Corcho**, Universidad Politécnica de Madrid, Spain
Examiner	**Virginie Lurkin**, Technische Universiteit Eindhoven, Eindhoven, The Netherlands
Examiner	**Thrasyvoulos Spyropoulos**, EURECOM, Sophia Antipolis, France
Thesis Director	**Bernard Merialdo**, EURECOM, Sophia Antipolis, France
Thesis Co-Director	**Raphäel Troncy**, EURECOM, Sophia Antipolis, France
Thesis Co-Director	**Michael Defoin-Platel**, Amadeus, Sophia Antipolis, France

Dedicated to my family

Acknowledgements

First of all, I would like to thank Amadeus, the company that allowed me to carry out this thesis in the best possible conditions by putting in place everything necessary to make it a success. I would particularly like to thank my supervisors at Amadeus, especially Michael Defoin-platel who brought me constructive criticism to get the best out of me, Lionel Gotti, Gerardo Ayala Solano and Olivier Ratier for their support but also their relevant remarks and comments from a business point of view which allowed me to understand the challenges of this thesis. I would also like to thank Thomas Fiig who has been of great help and involvement in conducting one of my best research works of this thesis. In addition, I would like to thank all my Amadeus colleagues who helped me in the business and operational part: Riccardo Petitti and Marco Manghisi.

Secondly, I would like to thank my supervisor Raphael Troncy: his contribution in the scientific part as well as the scientific methodology that he transmitted to me are only a few points in which he helped me to be able to carry out the research work during the thesis with success. In addition, I would like to thank all my Eurecom colleagues among them Lucas and Ismail who have not only contributed to make a pleasant work environment but also become genuine friends for me.

I would particularly like to thank the encouragements of all the people who supervised this thesis from near or far during the difficult periods.

Finally, I would like to say that without the unfailing help of my parents since the moment I was born, their unparalleled encouragement and their faith in me, I would never have been able to achieve what I was able to do. Lastly, I am extremely grateful to my mother, my first supporter and my daily source of motivation.

Sophia Antipolis, 01 August 2021 ***Amine Dadoun***

Abstract

With the recent advances in the field of Artificial Intelligence and its successful practical applications in various domains such as Natural Language Processing [101], Computer Vision [82] or Recommender Systems [169], many industrial sectors have started to adopt these technologies as part of their production workflow.

Recommender Systems, in particular, have demonstrated their huge impact when systematically applied in situations where enough data are available for Machine Learning algorithms to build accurate models. It is for example the case for the online retailing industry that has been drastically transformed with the emergence of automatic recommendations.

The travel industry being often cited as one of the best candidates to benefit from the Artificial Intelligence revolution [23], we propose to review the usage of Recommender Systems (past, present and future) in the context of the travel industry. More specifically, we focus on the airline travel industry because of its preponderance inside the travel market, and, more importantly because it is a good representative of the numerous challenges that this industry will face in terms of automatic recommendation.

A Recommender System is a component interfacing between customers and a catalog of products. From a customer point of view, a Recommender System helps to easily find the products fulfilling needs without having to express them explicitly. From the owner of the catalog point of view, a Recommender System is also a way to increase the visibility of products, to improve the customer experience and to build and maintain loyalty.

The scientific literature already reports a huge body of work around recommendation algorithms. However, most of them are not considered in production systems due to their lack of interpretability and scalability [16]. Conversely, thanks to their ability to overcome these two issues, Matrix Factorization [79] and Nearest Neighbor algorithms [128] are among the few algorithms that have been proven to be successful in industrial contexts. The digitization of our lives, and the incremental usage of the internet has pushed the airlines to invest in digital channels for selling their products. Moreover, online bookings represent now more than 80% of leisure airline bookings [52], which underlines the need for user-friendly websites guiding travelers toward the products they are looking for and this is exactly what a Recommender System is made for. With the progress in Artificial Intelligence, the trend is going toward more and more personalization. It is not about tailoring an offer for large market segments anymore but rather for a specific individual in a particular context. This move towards extreme person-

alization requires next generation Machine Learning techniques such as Deep Learning [105], making intense use of hardware acceleration and web-scale datasets.
To fully benefit from the power of Recommender Systems, it is necessary for the airlines to identify the potential recommendation use cases and then, to implement the corresponding technologies to customize their offers. More specifically, it is crucial to address the following points: what product to offer, to which customer, when to recommend an offer, at which price, and finally, how this offer should be presented to the customer and on which touchpoint.
The aim of this thesis is to provide answers to the aforementioned questions, to analyze the benefits of recommender systems for the airline travel industry and to propose novel recommender systems adapted to the airline industry with the objective to optimize airlines' offers conversion rate and improve the travelers experience.
In the first place, we explore the usefulness of enabling machine learning in airline specific recommendation use-cases that cover the traveler journey. More specifically, we propose Deep Knowledge Factorization Machine (DKFM) [29], an approach that leverages contextual, collaborative and content information in order to recommend personalized destinations to travelers. We compare our approach with a set of collaborative filtering methods and state-of-the-art recommender systems based on deep learning. In addition, we developed an API and a web service to demonstrate the usefulness of a personalized next trip recommender.
The use of collaborative filtering and hybrid recommender systems in the airline industry showed some limitations due to the nature of data such as data sparsity, cold start problem or even popularity bias [27]. To overcome these issues, we propose to use knowledge graphs as a means to represent all information used in recommender systems and to develop knowledge graph-based recommender systems to address some recommendation use-cases. In this context, we propose an approach that uses knowledge graph embeddings to better target the right audience in email marketing campaigns for airline products recommendation [28]. We conduct extensive experiments to compare our approach with the currently in-production rule-based system used by airline marketers and a supervised machine learning model based on handcrafted features as another baseline. The results demonstrate the impact of using knowledge graph embeddings as input of the machine learning model that predicts the target audience for a given marketing campaign.
Finally, in the same context, we propose Knowledge graph multi-task learning for recommendation (KGMTL4Rec) [27], a multi-task learning model based on a neural network architecture that leverages knowledge graph to recommend the next destination to a traveler. We experimentally evaluated our proposed approach by comparing it against the currently in-production system and state-of-the-art travel destination recommendation algorithms in an offline setting. The results confirm the significant contribution of using knowledge graphs as a means of representing the heterogeneous information used for the recommendation task, as well as the benefit of using a multi-task learning model in terms of recommendation performance and training time.

Résumé

Avec les récentes avancées dans le domaine de l'intelligence artificielle et ses applications pratiques réussies dans divers domaines tels que le traitement du langage naturel [101], la vision par ordinateur [82] ou les systèmes de recommandation [169], de nombreux secteurs industriels ont commencé à adopter ces technologies dans le cadre de leur flux de production. Les systèmes de recommandation, en particulier, ont démontré un impact considérable lorsqu'ils sont systématiquement appliqués dans des situations où suffisamment de données sont disponibles pour que les algorithmes d'apprentissage automatique puissent construire des modèles précis. C'est par exemple le cas du secteur de la vente au détail en ligne, qui a été radicalement transformé par l'émergence des recommandations automatiques.

L'industrie du voyage étant souvent citée comme l'un des meilleurs candidats pour bénéficier de la révolution de l'Intelligence Artificielle [23], nous proposons de passer en revue l'utilisation des Systèmes de Recommandation (passé, présent et futur) dans le contexte de l'industrie du voyage. Plus précisément, nous nous concentrons sur l'industrie du voyage aérien en raison de sa prépondérance au sein du marché du voyage, mais surtout, parce qu'elle est bien représentative des nombreux défis que cette industrie doit relever en matière de recommandation automatique.

Un système de recommandation est un composant faisant l'interface entre les clients et un catalogue de produits. Du point de vue du client, un système de recommandation permet de trouver facilement les produits répondant à des besoins sans avoir à exprimer explicitement sa volonté. Du point de vue du marchand, un système de recommandation est un moyen d'augmenter la visibilité des produits, d'améliorer l'expérience du client et de le fidéliser.

La littérature scientifique fait déjà état d'un grand nombre de travaux sur les algorithmes de recommandation. Beaucoup d'entre eux ne sont pas pris en compte dans les systèmes de production en raison de leur manque d'interprétabilité, d'évolutivité ou de passage à l'échelle. A l'inverse, grâce à leur capacité à surmonter ces problèmes, la factorisation matricielle [79] et la recherche de plus proches voisins (KNN) [128] font partie des rares algorithmes qui ont fait leurs preuves dans des contextes industriels. La numérisation de nos vies et l'utilisation croissante d'Internet ont poussé les compagnies aériennes à investir dans des canaux numériques pour vendre leurs produits. De plus, les réservations en ligne représentent désormais plus de 80 % des réservations des compagnies aériennes de loisirs [52], ce qui souligne la nécessité de sites Web conviviaux guidant les voyageurs vers les produits qu'ils recherchent

et c'est exactement ce à quoi sert un système de recommandation. Avec les progrès de l'intelligence artificielle, la tendance est de plus en plus à la personnalisation. Il ne s'agit plus d'adapter une offre à de larges segments de marché, mais plutôt à un individu spécifique dans un contexte particulier. Cette évolution vers une personnalisation extrême nécessite des techniques d'apprentissage automatique de nouvelle génération, telles que l'apprentissage profond (Deep Learning), rendue possible avec l'usage intensif de l'accélération matérielle et la mise à disposition de gigantesque jeux de données à l'échelle du Web.
Pour profiter pleinement de la puissance des systèmes de recommandation, les compagnies aériennes doivent identifier les cas d'utilisation potentiels de la recommandation, puis mettre en œuvre les technologies correspondantes pour personnaliser leurs offres. Plus précisément, il est crucial d'aborder les points suivants : quel produit proposer, à quel client, quand recommander une offre, à quel prix, et enfin, comment cette offre doit être présentée au client et sur quel point de contact.
L'objectif de cette thèse est d'apporter des réponses aux questions susmentionnées, d'analyser les avantages des systèmes de recommandation pour l'industrie du voyage aérien et de proposer de nouveaux systèmes de recommandation adaptés à l'industrie du voyage aérien dans le but d'optimiser le taux de conversion des offres des compagnies aériennes et ainsi améliorer l'expérience des voyageurs.

Contents

List of Figures

List of Tables

List of Abbreviations

AI artificial intelligence.

CA Context-aware.

CB Content-based Filtering.

CF Collaborative Filtering.

CTR click through rate.

CWA closed world assumption.

DLRS Deep Learning-based Recommender System.

DMOs Destination Marketing Organizations.

DRS destination recommender system.

EBSN Event-based social network.

EMD Electronic Miscellaneous Document.

GDS global distribution system.

GRU Gated Recurrent Unit.

HIN heterogeneous information network.

IATA International Air Transport Association.

KG Knowledge Graph.

KGE knowledge graph embedding.

KGRS knowledge graph-based recommender system.

LBSN Location-based social network.

LOD Linked Open Data.

MF Matrix Factorization.

ML Machine Learning.

MLP Multi-layer perceptron.

NDC New Distribution Capability.

NLP Natural Language Processing.

OWA open world assumption.

OWL Web Ontology Language.

PNR Passenger Name Record.

PoI Point of Interest.

RDF Resource Description Framework.

RDFS RDF Schema.

RFIC Reason for Issuance Code.

RMS Revenue Management Systems.

RNN Recurrent Neural Network.

ROI return on investment.

RS recommender system.

SB Session-based.

SKOS Simple Knowledge Organization System.

TF-IDF Term frequency-inverse document frequency.

URI Uniform Resource Identifier.

W3C World Wide Web Consortium.

Chapter 1

Introduction

1.1 Airlines in the digital age

The travel industry generally focuses on the sale of individual products even when these products are interdependent. The heterogeneous and complex nature of this industry does not allow to offer in an obvious way flexible travel experiences in which all the products needed by the traveler would be grouped into personalized packages representing the completeness of a trip. In order to create such an offer, it is necessary to understand the traveler's motivations, preferences and the way decisions are made.

The challenge consists in offering travelers inspiring and personalized offers in order to build and maintain their loyalty. However, travelers make decisions for various reasons: some are rational, while others are more emotional [2]; some are based on prior experiences, and some are based on objective characteristics of the offer such as the price, the travel time, etc. Understanding why the traveler takes a particular decision is therefore crucial. We hypothesize that the travel industry could take inspiration from other industries such as retail or entertainment in order to narrow the gap between travelers' needs and what is offered to them while keeping in mind the particularities of this industry that we detail further in this work.

In this thesis, we focus on the airline travel industry whose business is included in the travel sector. Airlines started and followed the deregulation taking place in the air transportation industry from the 70s and they have heavily invested in revenue management systems. For airlines, these systems are responsible for defining the price for which seats in airplanes should be sold at, taking into consideration the demand and the supply at the same time as shown in figure 1.1.

In the meantime, airlines have seen significant changes in the way their offer is being structured. Selling at the beginning air tickets which includes a wide selection of services, airlines

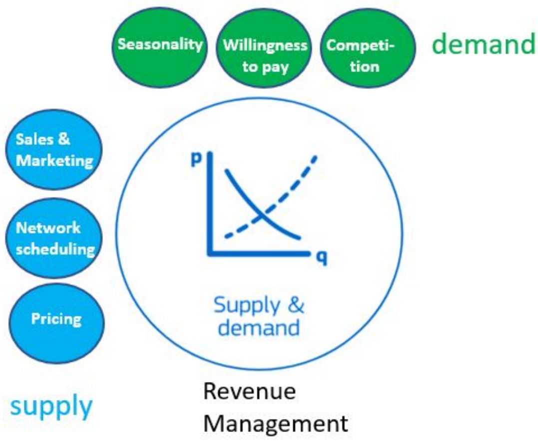

Figure 1.1 – Revenue Management is about reaching the best match between supply and demand.

are now selling significant volumes of ancillary services[1], ranging from flexibility options to additional comfort on board. Airlines went further by distributing as well, especially on their website, items sold by third party providers (rental cars, hotels, excursion, activities, etc.), aiming at making their offer cover the entire traveler journey. Selling now a much more diverse set of products, in order to maximize their revenues, airlines have to decide not only about the price of air tickets but to decide as well, **what** to offer, to which customer (**who**), **when** to offer, at which price, and finally **how** this offer should be presented to the customer and on which touchpoint.

In short, airlines have become retailers. The selling – or rather merchandising – processes of airlines encompass therefore many more aspects than it used to at the time of revenue management systems emergence. In parallel, as a result of both from the increase of computational power and from the digital transformation, airlines are collecting tremendous amounts of data about their customers, be it about their traveling history, their purchasing behavior, the way they engage with the airlines or the impact they have on social media. This data collection phase is primary and should be enabled first for airlines to become data-driven and thus start developing personalized offers for travelers.

Other industries with large inventory and broad digital penetrations such as web retailers have deployed advanced selling techniques, often data-driven and thus heavily relying on machine learning methods such as Recommender systems (RS), enabling them to pick the right offer for the right customer and increase their revenues as well as their customer satisfaction. Following this trend, the airline travel industry must be able to bridge the gap between travelers' motivations and the way services are proposed, drawing inspiration from these

[1]Ancillary services are all products offered by the airline beyond air tickets. They can be flight-related (e.g. extra baggage, preferred seat, etc.) or standalone services (e.g. lounge access)

other industries. Recent advances in artificial intelligence have impacted the development of a new generation of recommender systems in providing more accurate, contextualized and personalized offers to users. Hence, enabling recommender systems in the airline travel industry can help to adapt to the change of traveler's motivations and continuously generate concise and personalized offers. In figure 1.2, we present how offers can be now presented to travelers thanks to recommender systems.

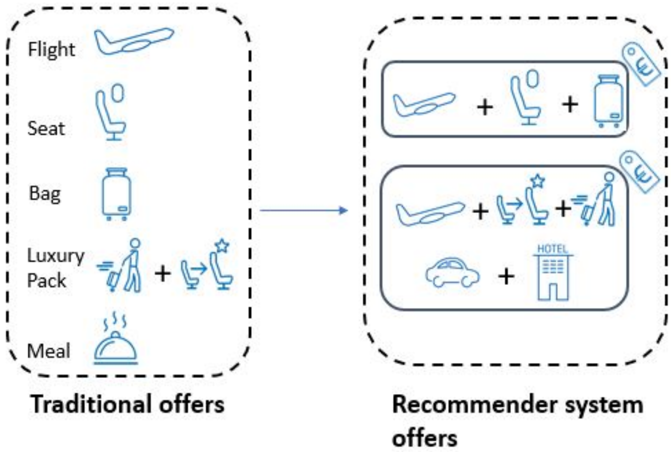

Figure 1.2 – Recommender systems are transforming the way airlines are selling products.

1.2 Recommender systems

A recommender system can be seen as an algorithm to compute the probability that a user (customer) would like to interact with an item (product or service). These systems were originally introduced to overcome the problem of information overload that customers face when exposed to a large catalog of products or services. By providing customers with contextualized and personalized recommendations, recommender systems aim at narrowing down the search to a manageable subset of products that are relevant to the customer.

Recommender systems have proven to be popular for both customers and sellers, particularly for online retail [124]. The most representative example is Amazon that has become one of the largest retailers in the world because, among other important things such as a large selection of products and a fast and reliable delivery chain, it offers best-of-breed customer experience as a result of an extensive use of recommender systems.

Recommender systems result in a more personalized shopping experience, giving customers the feeling of being understood and recognized which contributes in building trust and in maintaining loyalty. From the seller's point of view, recommender systems offer the possibility to control and to increase the exposure of their catalog by driving customers toward products lacking visibility. Recommender systems are also notoriously good at decreasing bounce rate and at increasing average time spent on a web page for online selling [137]. Finally, recommender systems have also proved to be very effective offline in email marketing campaigns allowing sellers to run so-called "one-to-one marketing" at scale [69].

Recommender systems are growing in popularity in the travel industry to address the complex set of decisions customers face when booking a flight, selecting a hotel or finding relevant events and activities at their destination. For example, Airbnb[2] is now offering real-time personalization of search rankings within its marketplace [47].

Travel agencies or brokers have recently called upon the research community to work further on the particularities of making recommendations in the context of travel. The online hotel booking platform Trivago[3] sponsored the 2019 Recommender Systems Challenge as part of the ACM RecSys yearly conference in order to improve their current recommender system for online hotels recommendation. However, despite the successful application of recommender systems across many industries, airline offer construction and retailing remains quite rudimentary with little or no differentiation in how products and services are selected, retailed, or priced across customers.

We believe the current approach is inadequate and that the key to profitability is to manage offers consistently in an integrated Offer Management System (OMS) encompassing recommender systems and thus serving the customer throughout the traveler journey from inspiration to post-trip.

1.3 The traveler journey

From inspiration, departure time to post-trip, recommendation can be triggered in any phase of the traveler journey (figure 1.3). The traveler journey is a key consideration to understand the customer needs and intents (figure 3.6). Research from Frost and Sullivan [94] indicates that there "are certain moments when the customer is in a purchasing mindset and thinking about his trip and what he will need". For example, at the booking stage, the customer is in a "planning" mindset. At this stage, the airline can approach the customer with more "expensive" offers such as cabin upgrade, or flexibility options. Close to departure (48h/24h), the customer has a different mindset - making the final preparations for his trip. At this moment, airlines could propose the customer with extra baggage, airport transfer, parking, priority check-in, or fast track access.

Therefore, at each phase of the traveler journey, one or more recommendation system use-

[2] https://www.airbnb.com/
[3] https://www.trivago.com/

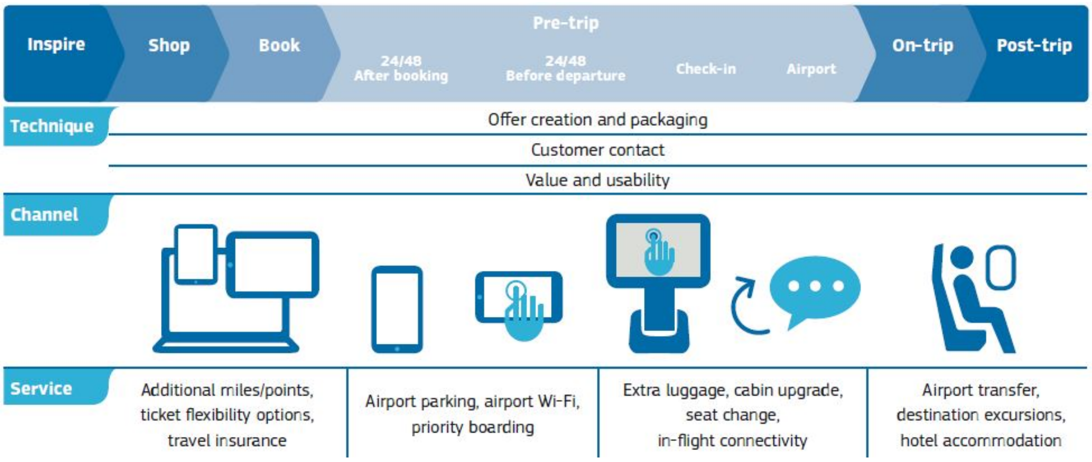

Figure 1.3 – The figure presents the merchandizing opportunities offered to airlines through the traveler journey. Source: https://amadeus.com/documents/en/blog/pdf/2014/12/report-thinking-like-a-retailer-airline-merchandising.pdf

cases can be addressed to build an end-to-end personalized traveler experience. The research challenge is therefore to look for novel and common methods that will eventually contribute to the development of recommender system algorithms that address the different use-cases.

Recommender systems in the airline industry usually suffers from the cold start problem and data sparsity [29]. Hence, establishing user profiles can be a difficult task as individual travel planning are typically much less frequent like, for example, book purchases or video watches. As a consequence, sophisticated recommendation techniques as widely used by Amazon or Netflix for instance cannot be directly applied to the airline travel domain [39]. In comparison with e-commerce or entertainment scenario (Netflix, YouTube, etc.) in which users' interactions are quite numerous (an average YouTube viewer watches 5 hours of videos a month[4], Amazon prime-members make 24 orders per year and non-members make 13 orders per year[5]), in the airline industry, travel interactions form a very sparse dataset. For example, UK travelers take in average 6.5 flights per year[6] and less than 5% of travelers purchase an ancillary service for a given flight in the European market (based on an in-house data analysis of historical sales, see section 3.1). The lack of travelers interactions with airlines products catalog confirm the sparsity in the dataset and using only travelers' historical bookings as input information of a recommender system may not be sufficient to build accurate recommendations.

Therefore, incorporating additional information such as travel context, travelers' demographics, or destination metadata into the recommender system could be valuable in address-

[4]https://www.comscore.com/
[5]https://www.statista.com/
[6]https://www.news24.com/

ing the above-mentioned issues. To integrate this heterogeneous information into a single data structure, knowledge graphs are an appropriate approach to consider. Indeed, recent works [113, 115, 134] have illustrated the effectiveness of using knowledge graph embeddings for items recommendation.

Furthermore, knowledge graphs can provide a unique data structure to gather all the information needed to develop a recommender system and thus be an input of it, to address various recommendation use-cases as shown in figure 1.3. Having a knowledge graph as a common data structure and a common input to all use-cases is a precious time saver for researchers and data scientists when they want to address each time a new use-case.

1.4 Knowledge graphs

According to [117], a Knowledge Graph (KG) (i) mainly describes real world entities and their interrelations, organized in a graph, (ii) defines possible classes and relations of entities in a schema, (iii) allows for potentially interrelating arbitrary entities with each other and (iv) covers various topical domains. Knowledge graphs are graphs in the sense that they store facts under the form of directed links between entities. For example, consider the fact that 'Eiffel tower' is located in 'Paris'. Both 'Eiffel tower' and 'Paris' are represented as nodes of the graph, whereas the property 'is located in' is represented by a typed edge connecting the two nodes. A fact is thus represented by a triple: (subject, predicate, object), e.g. (Eiffel tower, is located in, Paris) as shown in figure 1.4. Later in this thesis, we will present how properties and entities are referenced and defined through an ontology based on good semantic web practices.

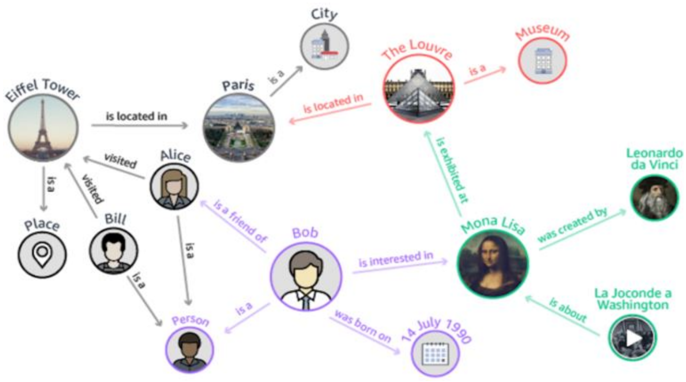

Figure 1.4 – An excerpt of a knowledge graph representing the city Paris as an entity in addition to some Paris landmarks also represented as entities. Properties are represented as typed edges connecting the entity to other entities. Source: https://www.kaggle.com/ferdzso/knowledge-graph-analysis-with-node2vec

KGs became an increasingly popular research direction towards cognition and human-level intelligence, and are now used in many AI applications such as semantic search or automatic fraud detection. In recent years, KGs have also been introduced in recommender system realm as Knowledge graph-based recommender systems (KGRS) [50] in order to enrich the graph of user-item interactions with more complex and structured information about the users, the items, and the interactions themselves.

One of the research challenges of this thesis is to build a comprehensive knowledge graph that first represents a complete traveler's journey from the time he/she enters airlines' website to his/her boarding in the airplane. The knowledge graph should contain information such as the travel context, travelers' demographics, or destination metadata but also contain descriptions of events and activities, places and sights, transportation facilities as well as social activities relevant to a destination (see figure 5.5). Those datasets are collected from numerous static, near- and real-time local and global data providers in the domain of tourism and culture. Entities in those knowledge graphs are automatically de-duplicated, interlinked and enriched using semantic web technologies (see section 2.2).

The construction of a knowledge graph will allow us to have a common data structure to be used for the different recommendation use-cases that cover the entire traveler journey, saving time and effort in building a dataset for each use-case. In addition, it will allow the data to be enriched with additional information to overcome the problem of data sparsity which is prevalent in the airline industry as shown in section 3.1.

This thesis is located at the intersection between the Recommender Systems and the Knowledge graphs research fields with an application in the airline industry, showing how recommender systems can be put in place in this industry and transform the way airlines are constructing and retailing their products. The sparsity of data collected in the airline industry in contrast with the wealth of data available on the Web leads us to use knowledge graphs as structure to incorporate the data coming not only from airlines' databases but also from the web. This approach allows to leverage the advances in recommender systems in the use of knowledge graph-based algorithms to improve the recommendations suggested to the travelers across their journeys.

In this context, several research challenges and questions arise, which will be the focus of the thesis research work. We describe the research challenges and contributions of the thesis in the following section.

1.5 Research challenges and contributions

Numerous studies show that travelers are now looking for personalized travel experience. For example, the Sabre Cooperation reveals that travelers are willing to spend 100$ on airline ancillaries to get a personalized travel experience[7]. Airlines maintain a huge data lake containing lists of possible destinations, ancillary services as well as possible bookable activities at destination provided by third parties such as Viator[8].

Beyond travel destinations and ancillary services, airlines can propose a whole set of tourism related offers ranging from car rentals, accommodations, activities and tickets for attraction and events. In this context, we formulate the first research question of this thesis:

- ***RQ1***: How can we propose personalized items (travel destinations, ancillary services, third party content) to travelers using recommender system algorithms? (Chapter 4)

  Addressing this research question requires implementing and testing hybrid recommender systems mixing content-based and collaborative filtering algorithms in order to suggest relevant offers for each traveler based on user profile inferred from the purchase history and insights gained on a broad collection of products.

  More specifically, in order to answer this research question, we have developed a systematic approach which consists in defining a list of use-cases of airline-specific recommendations. This approach consists of listing products already on the market that address the use-case, then collecting the data from the product data feed, and finally implementing a recommender system algorithm and setting up an evaluation protocol to assess the effectiveness of the implemented recommender system. In the following, we break down the research question ***RQ1*** into different research sub-questions that address a specific recommendation use-case.

  In the first place, the airline is interested at recommending the next destination to a traveler. This task that has been at the forefront of the airline industry for a long time is still an open problem as current airline solutions which provide recommendations on travel destinations lack contextualization and, more importantly, personalization. They either use a solution that suggests their most popular destinations to all travelers, or an interactive inspiration tool that matches travelers' criteria (budget, interests, etc.) with travel destinations. Therefore, there is a real opportunity to research methods to personalize the recommendation of travel destinations. In this context, we formulate the following research sub-question to address the problem of personalization in travel destination recommendation:

  - *RQ1.1*: What travel destination should be recommended to each traveler? (Section 4.1)

    To address this research question, we formulate the following problem: Given a traveler, his demographics information (age, nationality, etc.), his historical

[7]https://www.sabre.com/insights/releases/global-study-reveals-travelers-would-spend-100-on-airline-ancillaries-to-personalise-travel-experience/

[8]https://www.viator.com/

bookings and the contextual data related to those bookings (departure day of week, number of passengers, stay duration, etc.), we aim to recommend to this traveler a ranked list of destinations he/she would like to go to.

To tackle this problem, we propose Deep-Knowledge Factorization Machines [29] (DKFM) (section 4.1.4), a deep learning-based recommender system that leverages contextual, collaborative and content information in order to recommend personalized travel destinations to travelers. DKFM is a deep neural network that incorporates factorization machines in its core fed by contextual information, and a multi-layer perceptron fed by travelers' past interactions and also destinations and travelers' metadata. We compare our approach with a set of collaborative filtering methods and state-of-the-art deep learning-based recommender systems. We show that our hybrid deep learning-based recommender system outperforms state-of-the-art deep learning-based recommender systems for travel destination recommendation and obtains accurately decent results for the metrics defined in the experiments (see section 4.1.6). We also determine through an ablation study the contribution of every type of information with respect to the recommendation performance. Finally, a REST API is developed and integrated in a web service for demonstration purpose.

Once a traveler has booked an airline ticket, he or she may be interested in purchasing an ancillary service such as baggage, lounge or even third-party content like a hotel or airport transfer. Hence, the airline has an important role to play in personalizing the offers provided to travelers during this up-sell phase. We formulate the second research sub-question that arises from the above:

- *RQ1.2*: What ancillary service should be recommended to each traveler? (Section 4.2)

One of the most used retailing techniques in the airline industry to up-sell products is the use of email marketing campaigns, used to provide travelers with additional airline products to buy such as ancillary services, travel attractions, car rentals, etc. To address the research question *RQ1.2*, we consider the following problem: Given a notification campaign aimed at a large audience of travelers who have already booked a flight in a given context, we aim to target the relevant travelers among all the travelers that the email marketing campaign will reach.

To tackle this problem, we propose an approach [28] that leverages travelers' historical purchases and travelers' data to better target the audience in email marketing campaigns for ancillary services recommendation. We conduct extensive experiments to compare our approach with the currently in-production rule-based system used by airline marketers. Results show that using a machine learning algorithm instead of a rule-based algorithm leads to a better conversion rate of the email marketing campaigns sent to travelers.

Considering the travel flow shown in figure 1.3, after deciding which ancillary service to purchase, the traveler begins to think about which hotel would be best for him/her to stay at the destination. When a traveler wants to search for a place where to stay in the destination he/she travels to, he/she often looks to metasearch engines in order to compare different options he/she can be offered. Similarly, some airlines integrates the same metasearch engines in their booking flow with the goal to facilitate the traveler with the heavy search process. Based on some search criteria, metasearch engines propose a ranked list of hotel accommodations to propose to their users. This ranking can be customized by the airline and more importantly personalized to each traveler based on other available information than search data which the airline already has at its disposal. In this context, we address the following research question in order to make the accommodation search experience more personalized:

- *RQ1.3*: How can we personalize the suggested list of accommodations for each traveler?

  To address this research question, we make use of a public dataset of hotel search sessions released by the hotel booking platform Trivago provided as part of the RecSys 2019 Challenge[9] where the goal is to predict which accommodations (items) have been clicked in the search result during the last part of a user session in an offline evaluation setup with two objectives: improve the click-through rate (CTR) of Trivago navigation sessions and personalize search results for Trivago users.

  To address this problem, we propose a two-stages approach composed of a many-to-one recurrent neural network (RNN) that learns the probability that a user will click on an accommodation based on the sequence of actions the user has performed during the browsing session and a rule-based algorithm that reorders the list of accommodations based on a pattern obtained through comprehensive analysis of session data. In this work, we demonstrate the usefulness of using supervised machine learning to improve the CTR of Trivago users' sessions and conduct an extensive analysis on users' browsing sessions that lead to important conclusions and lessons to consider in order to improve the user experience in Trivago's metasearch engine.

To summarize, the three sub-research questions (RQ1.1, RQ1.2, RQ1.3) have been addressed using modern collaborative, content-based and hybrid recommender system algorithms. However, even if the proposed algorithms have shown relatively better results than other approaches (see chapter 4), they suffer from a number of shortcomings, including data sparsity

[9] http://www.recsyschallenge.com/2019/

and popularity bias and the limitations mentioned in section 1.3 and 1.4. Moreover, from a more abstract point of view, if we think about how does a traveler take the decision to go to a certain destination (e.g. Paris) or how does his/her brain perceive this city, we may suppose that all the information leading to the answer are interconnected. Hence, the challenge is to use all those concepts and relationships and associate them to build a relevant input data structure used by the recommender systems.

Therefore, the second part of this thesis is dedicated to the exploration of knowledge graph-based recommender systems. We aim to experimentally demonstrate the benefits of adopting this family of recommender system algorithms over the more traditional ones to revisit the previous research sub-questions. In this context, we formulate the following two research question:

- ***RQ2***: How can we build a comprehensive knowledge graph intended for the airline domain? (Section 5.1)

Adopting semantic web technologies, and relying on the many data sources available on the web and airline databases that contain millions of travelers' bookings, we develop an ontology that defines several classes corresponding to the high-level entities available in the collected data (see section 5.1.2). Then, based on the ontology, we build a large knowledge graph that contains travelers' bookings for two-year flights for a partner airline, and then use this knowledge graph as input source of the recommender systems developed to address two recommendation use-cases (see chapter 5).

- ***RQ3***: How can we leverage knowledge graphs to improve the predictions for each of the recommendation use-cases addressed in this thesis and overcome the standard recommender system limitations? (Chapter 5)

  To address this research question, we propose to develop novel knowledge graph-based recommender systems suited for the airline recommendation use-cases addressed in this thesis. For each use-case, we formulate the following research sub-questions:

  - *RQ3.1*: How does the use of knowledge graph embeddings compare to the use of handcrafted features used as input of a supervised machine learning model trained to target the relevant audience for an email marketing campaign? (Section 5.2)

    To address this research question, we propose TKE4Rec [28], an approach that leverages knowledge graph embeddings to better target the right audience in email marketing campaigns for airline products recommendation. More formally, the proposed approach consists of two stages: first, we compute KG embeddings of travelers and flight reservations; second, we use these embeddings in addition to flight contextual features as input of an XGBoost [19] classifier to learn what is the relevant audience to target for a given marketing campaign. We conduct extensive experiments to compare our approach with the currently in-production rule-based system used by airline marketers and a supervised machine learning model based on handcrafted features as another baseline. The results suggest that the use of knowledge graph embeddings is the most effective approach.

- *RQ3.2*: What is the benefit of using a knowledge graph as a unique data structure containing all the input information of the recommender system for travel destination recommendation? (Section 5.3)

  To address this research question, we propose KGMTL4Rec [27]: a multi-task learning model based on a neural network architecture that leverages knowledge graph to recommend the next destination to travelers. We experimentally evaluated our model by comparing it against the currently in-production recommender system and state-of-the-art travel destination recommendation algorithms including DKFM [29] in an offline setting. The results confirm the significant contribution of using knowledge graphs as a means of representing the heterogeneous information used for the recommendation task, as well as the valuable benefits of using a multi-task learning model in terms of recommendation performance and training time.

In the next section, we summarize the thesis structure.

1.6 Thesis structure

As illustrated in figure 1.5, the thesis is divided into six chapters, addressing research challenges within the Recommender Systems and Knowledge Graphs research fields applied to the airline industry.

Chapter 1 provides the general context in which the thesis is grounded, describes the research challenges and contributions of the thesis and provides an outline of the work.

In Chapter 2, a literature review of recommender systems and knowledge graphs is provided, going from general notions and concepts about RSs and KGs to the most recent and advanced works in the fields.

Chapter 3 presents a systematic review of recommender systems in the airline travel industry and how those can transform the construction and retailing of airlines' offers.

Chapter 4 describes the recommender systems developed and implemented to address the airline specific recommendation use-cases addressed in the thesis.

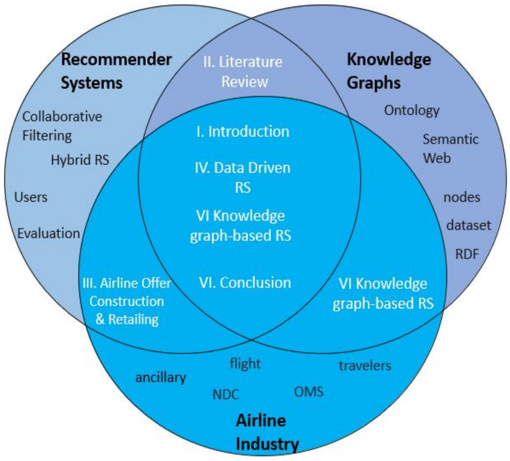

Figure 1.5 – The thesis is divided in 6 chapters covering three topics: recommender systems, knowledge graphs and the airline industry.

In chapter 5, we first present the ontology developed to build the knowledge graph that contains the data collected from the airlines and through Linked Open data, used as input of the knowledge graph-based recommender systems developed to address two different airline-specific recommendation use-cases. Then, we describe the work conducted to address these use-cases to answer *RQ3.1* and *RQ3.2*.

In Chapter 6, we summarize the findings of our research work, draw the main conclusions and outline possible short term and long term future work of this thesis.

Chapter 2

Literature Review

This chapter aims to introduce a set of notions that are important for the understanding of the work carried out in this thesis. We provide definitions for well established concepts such as Knowledge Graphs and Recommender Systems, as well as a up-to-date and relevant literature review of the most recent work in these research fields.

2.1 Recommender Systems

Information retrieval, as a scientific research field, is tightly coupled with recommender systems. Recommender systems address the problem of information overload that users normally encounter by providing them with personalized recommendations on content and service.

2.1.1 Principles

In the terminology of recommender systems, the customers are referred to as *users* and the products in the catalog are referred to as *items*. Hence, a recommender system can be seen as a way to compute the probability that a user would like to interact with an item and use this probability to recommend the most relevant subset of items to this user. Depending on the context, an interaction would correspond to the act of searching, buying, visiting, watching, etc.

In its most simple form, a recommender system is typically built in three consecutive steps: *information collection*, *learning* and *recommendation* [66]. The information collection phase consists in building a weighted graph $G = (U, I, E, w)$, where U, the set of users, and I, the set of items, are the nodes in the graph and E corresponds to the set of edges. These edges represent the past interactions between users and items. There are no edges between the users nor the items, hence the graph is bipartite. The strength of these past interactions is given by the function $w : E \mapsto [0,1]$.

In the learning phase, a Machine Learning (ML) algorithm is used to train a model $\mathscr{W}$ that approximates w in G. Finally, in the recommendation phase, the trained model is used to predict, for every possible pair $(u,i) \in (U \times I)$, the strength of the interaction between user u and item i. From these predictions, it is then possible to derive the list of items that could be recommended to the users.

From Tapestry [45], introduced in the early 90's that is considered as the first example of a working collaborative filtering algorithm, to the massive usage of deep learning algorithms [170], the research on recommender systems is now one of the most prolific topics in the Artificial intelligence (AI) literature. ML models designed to predict user-item interactions have evolved from using simple linear and logistic regression to deep neural network models that endow them non-linearity, and thus allow them to find non-linear patterns in the data. However, each of these approaches has its own specificities and it is important to understand their strengths and limitations when addressing a particular recommendation problem. In the remaining of this section, we review the main families of recommender systems [70]. Since this thesis is applied to the airline industry, we have chosen to illustrate our explanations of the different families of recommender systems by using products but also airline-specific terminology.

Collaborative Filtering (CF) Recommender Systems

CF algorithms are among the most widely used algorithms in the field of recommender systems [128] and have been applied in industries such as e-commerce or online entertainment to recommend the most relevant products (e.g. movies) to their customers. In the original formulation, a CF algorithm relies only on the interactions present in the graph G without any additional knowledge or information about the items or the users.

Figure 2.1 is an illustrative example of the bipartite user-item graph G for ancillary products. The graph contains interactions between users (travelers) and items (e.g. seat, baggage, etc.) represented by the solid arrows, while the dashed arrow represents the recommendations obtained from a CF algorithm. Let us consider the item i_1 (baggage) for example. Users u_1 and u_2 both purchased this item. Furthermore, user u_1 also purchased item i_2, thus item i_2 is recommended to user u_2.

We can divide CF algorithms into two different classes of methods: the first one relies on *Matrix Factorization (MF)* techniques [63] while the second one, named *Neighborhood Methods* [128], relies on computing the similarity between users or items.

Over the years, significant progress has been made to improve CF algorithms, for example, in

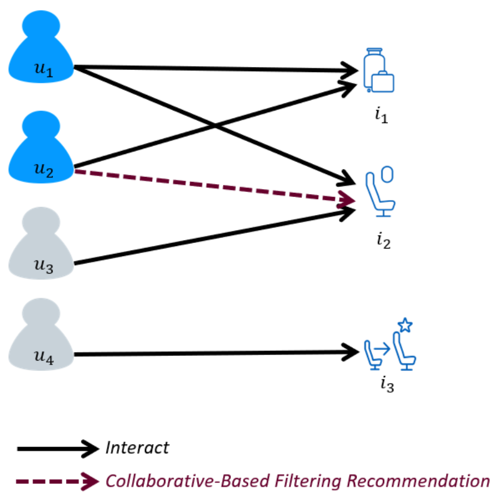

Figure 2.1 – CF Recommender Systems: Bipartite graph between users and items showing how item i_2 is recommended to user u_2 through a CF algorithm.

terms of learning speed [56] or accuracy [54, 121]. Nevertheless, despite their proven overall effectiveness and usability, CF algorithms are still limited especially when users interact with a restricted number of items (*data sparsity*) or when new users or new items frequently enter the system and, consequently, past interactions are not available (the user or item *cold start problem*).

Content-based Filtering (CB) Recommender Systems

CB filtering algorithm [85] aims at building user preference profiles based not only on historical user-to-item interactions but also on a form of description of these items that is often represented by a set of keywords or properties. Conversely, it is also possible to associate items to user profiles by looking at the description of the users interacting with them.

In figure 2.2, we present the graph G enriched with item properties required for the use of CB recommender system. Each item (ancillary product) is characterized by a set of properties: for example, the baggage item has the value "C" for the Reason for Issuance Code (RFIC)[1] and the value "A" for the Electronic Miscellaneous Document (EMD)[2] category, as it is a flight-associated product. In this example, the CB algorithm recommends item i_3 (premium seat) to user u_3 because item i_3 has the same characteristics of item i_2 which user u_3 has interacted with (added in user's cart) in the past.

With CB filtering, even new items without any previously observed interactions will have at

[1] RFIC is a categorization of ancillary services proposed by airlines. For further details, see https://www.atpco.net/resource/optional-services-industry-sub-codes.

[2] EMD is a ticket that contains information about ancillary services purchased in addition to the flight.

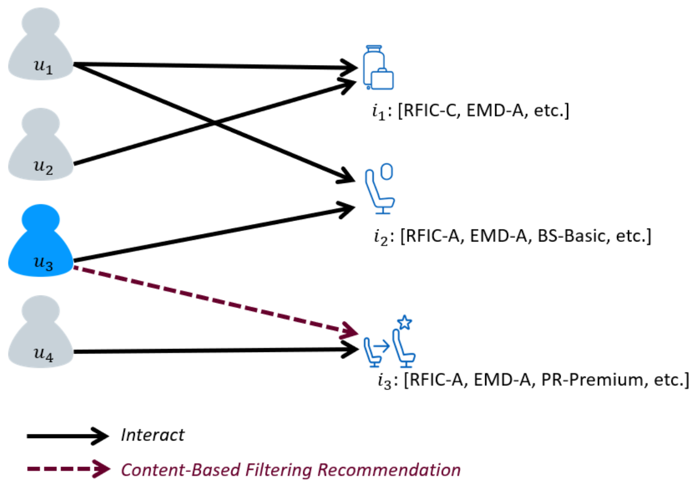

Figure 2.2 – CB Recommender Systems: Bipartite graph between users and items enriched with item descriptions showing how item i_3 is recommended to user u_3 through CB algorithm.

least a description that can be used by the system to provide recommendations. Hence, the problem of item *cold start* is mitigated. Nevertheless, CB filtering methods also have some shortcomings. For example, building and maintaining relevant representations for every item can turn into a heavy feature engineering task. Also, introducing novelty into what is being recommended to a given user is not possible since the system works only by looking at content associated with the user's past interactions.

One of the alternatives to deal with the above mentioned limitations such as the lack of novelty consists in mixing CB and CF techniques in what is referred to as Hybrid recommender systems in the literature [74, 98]. The shift of predictive models during recent years from using simple linear or logistic regression to models that incorporate deep networks [169] in order to consider many types of data such as categorical data projecting them into embedding spaces and numerical data in one model improved drastically models' performances. Following this trend, many deep learning-based recommender systems [21, 29, 105] have emerged taking into consideration numerous types of data. However, these models need the data to be pre-processed which can be a heavy task, especially when there are many features.

Context-aware (CA) Recommender Systems

CF or CB algorithms model the users' behavior by relying on past user-item interactions or on the content of the items. However, to better capture the complex decision-making process that the users are following when exposed to a selection of items (e.g. the offer set construction by the offer management system), it is crucial to consider the overall context of this process. For instance, a user who wants to travel during summer with four people for two weeks (likely leisure travel) will not have the same needs when traveling alone for two days during a winter week (likely business travel).

A CA recommender system should first be able to collect contextual information and then make use of it to better tailor the offers depending on the circumstances. In figure 2.3, we present the graph G enriched with contextual information. As an illustration, let us consider that the user u_1 who purchased both items i_1 (baggage) and i_2 (seat) for his trip to Paris which will last 8 days with a flight duration of 6 hours. On the other hand, we consider the user u_2 that will travel from New York to Paris on a similarly long flight (7 hours) for 12 days and purchased item i_1 in addition to the flight ticket. Item i_2 is being recommended to user u_2, as contexts C_1 & C_2 are closely related.

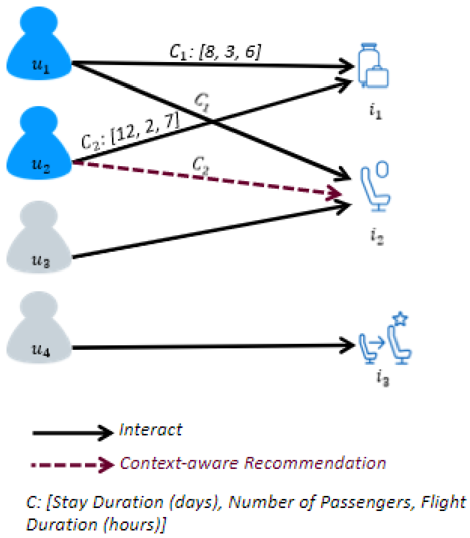

Figure 2.3 – CA Recommender Systems: Bipartite graph between users and items enriched with contextual information showing how item i_2 is recommended to user u_2 through CA algorithm.

Several initiatives have been conducted to enrich existing recommendation approaches with contextual information. We can categorize them into three different groups [4]: (i) Contextual Pre-filtering [3] where the contextual information is used only to filter out the graph of user-item interactions to keep only the data pertaining to a particular context; (ii) Contextual Post-filtering [116] where the context is used to produce contextualized recommendations on top of what a traditional recommender system suggests; and finally (iii) Contextual Modeling [72, 120, 156] where the context itself is considered by the model as input information together with the user-item interaction graph.

2.1.2 Knowledge Graph-based (KG) Recommender Systems

A Knowledge Graph can be seen as a directed heterogeneous graph in which nodes correspond to entities and edges correspond to relations (see section 2.2 for more details). In recent years, knowledge graphs have been used in recommender systems in order to overcome the problem of user-item interactions sparsity and the cold start problem which CF methods suffer from by leveraging properties about items and users and representing them in one single data structure [114]. In figure 2.4, an example of a Knowledge graph is depicted.

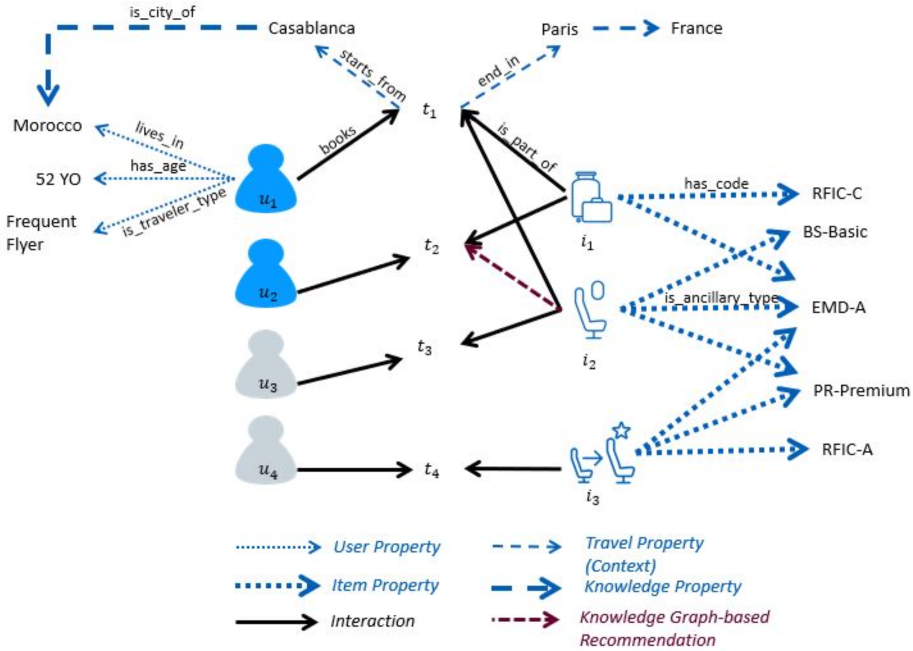

Figure 2.4 – KG Recommender Systems: Knowledge graph representing user-item interactions in addition to information about users, items and the context of each interaction showing how item i_2 is recommended to the user u_2 via KG recommender system algorithm over the knowledge graph.

Beyond the simple lists of properties already managed by previous versions of recommender systems, KGs represent and leverage semantically rich relations between entities. We see that travel t_1 booked by user u_1 starts from Casablanca, a city in Morocco, which is also the country where user u_1 lives. By construction, KGs can easily be linked between each other. For example, it would be straightforward to extend the graph from figure 2.4 to include cities' main Points of Interest (PoIs) [103]. One remarkable thing about KG recommender systems is their ability to make use of the KG structure to provide better recommendations [134].

In general, existing KG-based recommendation can be classified into two main categories [50]:

- *Embedding-based methods*, which are a subclass of knowledge graph-based recommender systems, consist in pre-processing a KG using knowledge graph embedding algorithms [12] and then incorporating the learned entity embeddings into a recommendation framework [29, 114, 168] (we describe in more details knowledge graph embedding algorithms in section 2.2.3). By using knowledge graph embedding algorithms, it is now possible to turn virtually any type of information into a vector which the system can learn.

In [168], the authors propose a two stages approach that consists in first computing the embeddings coming from a knowledge base composed of structural knowledge, image and text representing the items, then use the generated embeddings as input of a CF algorithm. In [114], the authors propose a two stages approach where they first compute some relatedness scores between embeddings of entities learned through node2vec [48], then they use Adarank as a learning to rank framework to rank the items they recommend for a given user. In [172], the authors make use of various types of side information about the items (e.g. review, brand, category, bought-together, etc.) to construct a knowledge graph which is subsequently used to construct items and users embeddings. As a second step, the recommender system rank candidate items j in an ascending order of a distance between u_i and v_j.

To summarize, many of the embedding-based methods are composed of two stages: first, entities' (items, users, etc.) embeddings are learned; second, embeddings are incorporated into a recommendation learning algorithm.

- *Path-based methods* which explore the various patterns of connections among items in a KG to provide additional guidance for recommendations. However, they heavily rely on manually designed meta-paths which are hard to optimize in practice.

  In [110], the authors present a hybrid graph-based data model to predict top-n recommendation by first extracting meta path-based features from a KG enriched through Linked Open Data (LOD), then train a learning to rank algorithm using co-occurring path metrics as features of the algorithm. In [166], the authors use matrix factorization method to compute latent representation of entities for different sub-graphs extracted from a heterogeneous KG, and then use an aggregation method to group all the generated latent representation to compute a recommendation probability. Inspired by the work proposed in [166], in [173], the authors considers the KG as a heterogeneous information network (HIN). They extract path based latent features to represent the connectivity between users and items along different types of relation paths. The drawback of these methods is that they commonly need expert knowledge to define the type and number of meta-paths. With the development of deep learning algorithms, different models [62, 134, 154] have been proposed to automatically encode KG meta-paths through embeddings to overcome the above mentioned limitations.

  Finally, recently, path-based methods have also been used to bring explainable recommendations [132, 150] benefiting from the fruitful information contained in the KGs.

Another category of KG recommender systems is worth mentioning. This category of recommender systems considers the whole structure of the KG instead of KG triples. In [146], the authors propose *RippleNet*, an end-to-end framework that naturally incorporates the knowledge graph into recommender systems. Similar to actual ripples propagating on the water, RippleNet stimulates the propagation of user preferences over the set of knowledge entities by automatically and iteratively extending a user's potential interests along links in the knowledge graph. In [140], the authors propose *AKUPM* a method that categorizes relationships into two types: inter-entity interaction and intra-entity interaction in order to avoid that the recommendation results suffer from some unrelated entities. A model is created for each category of relations. Hence, AKUPM is able to figure out the most related part of incorporated entities.

2.1.3 SB Recommender Systems

Recommender system approaches based on historical user-item interactions are very powerful because they are able to exploit long-term user profiles [93]. However, in many real-world applications such as e-commerce platforms, a large number of new users visit the system every day for which no historical information is available (user *cold start* problem).

It is therefore necessary to analyze users' live sequence of actions (for instance, sequence of their clicks) to identify patterns and generate recommendations [87]. This approach can range from simply detecting frequently co-occurring actions [5] to a more in-depth modeling of the sequence itself with deep learning techniques [59].

In figure 2.5, user u_1 starts a browsing session looking for a flight (event e_1), then chooses his flight (e_2) and adds it to the shopping cart, and he decides to add two ancillaries (seat and baggage) which represent events e_3 and e_4, to finally make his booking e_5. On the other hand, user u_2 follows the same path as u_1 for his first two events and decides at $t-1$ to add a seat to his shopping cart. Since adding seat and baggage in the same shopping cart are two co-occurring events, a SB recommender system will propose to user u_2 to add a baggage to his cart.

2.1.4 Recommender Systems in Tourism

Tourism is a trip for leisure or business. It involves complex decision-making from travelers to select destinations, hotels, events, activities, etc. On the other side, travel industry players (e.g.

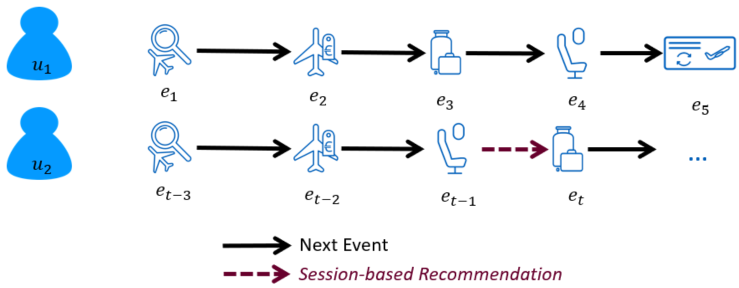

Figure 2.5 – Session-based recommender systems: Sequence of user events (interaction with the catalog), user u_2 is being recommended a bag at t through SB algorithm

travel agents) are helping travelers to find the most suitable options.

Early works have focused on personalized techniques in order to provide recommendation based on users' preferences and interests [125]. In [89], the authors proposed PersonalTour, a recommender system which is used by travel agencies to find suitable travel packages in accordance with the customer preference. In [129], the authors introduced MyTravelPal, a system providing travel destination recommendations in accordance with the affinity to user areas of interest. In [77], a Naive Bayes model is used to recommend travel destinations in a hotel booking platform based on multi-criteria rating data provided by previous users.

In [152], the authors propose an approach to generate sequence of Points of Interests (POIs) when visiting a city based on three user's inputs: start and end point plus his interests. However, user preferences or item characteristics are in many cases insufficient to have accurate recommendation. In [96], the authors propose to use contextual signals provided by Location Based Social Networks (LBSNs) such as time or location for events recommendation. In [78], the authors propose to compute an interest score for places and events in case of an in-car use, based on user preferences (given explicitly by the user) and weather conditions (contextual information). In [161], the authors use a neural network to learn user preferences, then used a context graph in order to regularize the obtained user preferences embedding.

2.1.5 Dataset for Tourism Recommendation

Several tourism recommendation use-cases have been addressed in recent years and consequently a number of datasets have been made public in order to replicate results or even improve on existing findings. In [7], the authors collected a very large-scale hotel recommendation dataset, based on TripAdvisor[3], containing 50 million reviews on hotels. In the hotel booking domain, Trivago[4] has released a public dataset of hotel search sessions as

[3]https://www.tripadvisor.com/
[4]https://www.trivago.com/

part of the ACM RecSys 2019 Challenge[5], with the goal to build a recommender system that predicts which hotels (items) the user has clicked on among the search results provided by the metasearch during the last part of the user session. In [103], the authors used Location Based Social Networks to build users' trails, where a trail is a succession of check-ins (the user share his/her location) made by a user in a venue when visiting a city. The Booking.com[6] platform recently released a dataset for the next destination recommendation task as part of the ACM WSDM 2021 WebTour[7] considering different contextual information related to the hotels bookings.

2.1.6 Evaluating Recommender Systems

In this section, we present the metrics and evaluation protocols that are often used to evaluate recommender systems and that will accordingly be used in this thesis. Techniques that claim to improve prediction accuracy in specific contexts must be evaluated following a rigorous experimental protocol and above all should permit the research community to be able to reproduce the experimental results as much as possible through open data and open source code.

Evaluation Metrics

In recent years, most of the research work carried out in the field of recommender systems is partly evaluated using accuracy metrics coming from the information retrieval field such as precision ($P@K$) and recall ($R@K$) [115, 134, 167] whose formulas are provided hereafter:

$$P@K = \frac{1}{n}\sum_{u=1}^{n}\sum_{j=1}^{K}\frac{hit(i_j, u)}{K} \tag{2.1}$$

$$R@K = \frac{1}{n}\sum_{u=1}^{n}\sum_{j=1}^{K}\frac{hit(i_j, u)}{|Rel(u)|} \tag{2.2}$$

where n and m represent respectively the number of users and the number of items, the set of items $i_1, i_2, ..., i_K$ are the items ranked from 1 to K, the value of hit is 1 if the recommended item i_j is relevant to user u, otherwise 0. $Rel(u)$ represents the set of relevant items for user u

[5] https://recsys.trivago.cloud/challenge/
[6] https://www.booking.com/
[7] https://www.bookingchallenge.com/

in the test set.

The purpose of these evaluation metrics in the context of product recommendation is to identify the K most relevant items for a given user and to measure the quality of retrieving with precision relevant information. In the special case of recommending only one item to the user, as in SB recommender systems where we want to measure the correctness of the immediate next item, $hitrate@K$ [54, 59] (Equation 2.3):

$$hitrate@K = \frac{1}{n}\sum_{u=1}^{n}\sum_{j=1}^{K} hit(i_j, u) \tag{2.3}$$

Similarly to $R@K$, $hitrate@K$ measures the correctness or accuracy of a recommender system.

Three other metrics are widely used in the literature to assess the accuracy of a recommender system, and more particularly, capture how well the hit is ranked in the list [54, 115, 134]:

- Mean average Precision ($MAP@K$): This metric measures how the order of relevant items is given by the recommender system:

$$MAP@K = \frac{1}{n}\sum_{u=1}^{n}\sum_{j=1}^{K} \frac{hit(i_j, u)}{|Rel(u)| \times |Rel_j(u)|} \tag{2.4}$$

- Top-K Mean Reciprocal Rank ($MRR@K$): This metric is a specific case of $MAP@K$, where there is only one relevant item. It measures how the recommender system rank well the relevant item against the irrelevant ones:

$$MRR@K = \frac{1}{n}\sum_{u=1}^{n}\sum_{j=1}^{K} \frac{1}{rank(u, i_j)} \tag{2.5}$$

  where i_j is the relevant recommended item within the top-K recommended items.

- Normalized discounted cumulative gain ($NDCG@K$): Order matters for both of $MAP@K$ and $NDCG@K$, but the main difference is that the mean average precision measure the binary relevance (an item is either of interest or not), while $NDCG@K$ allows relevance scores in form of real numbers:

$$DCG@K = \frac{1}{n}\sum_{u=1}^{n}\sum_{j=1}^{K} \frac{hit(i_j, u)}{log_2(i_j + 1)} \tag{2.6}$$

$$NDCG@K = \frac{DCG@K}{IDCG@K} \tag{2.7}$$

where $IDCG@K$ is the ideal discounted cumulative gain, defined as follows ($Rel(u)$) contains only relevant item of user u):

$$IDCG@K = \frac{1}{n} \sum_{u=1}^{n} \sum_{j=1}^{|Rel(u)|} \frac{hit(i_j, u)}{log_2(i_j + 1)} \tag{2.8}$$

Despite the relevance of these metrics in the assessment of recommender systems, recommending the same kind of products can be sometimes counter productive and not sufficient in real world applications (Netflix, Youtube, etc.). For instance, on Netflix, the user might be attracted by new kind of movies and series; On Youtube, the user often wants to watch new videos. The user must be surprised, and a good recommender system should have the ability to recommend unexpected and attractive items. The idea of not relying solely on precision-based metrics is also supported in [43]. In their work, the authors state that the purpose of an evaluation protocol is to assess the quality of the recommended items, and not only their accuracy or utility. In this context, only an online experiment where users of the system can judge the quality of the recommendations can reliably evaluate the recommendations. Therefore, when evaluating offline, it is necessary to consider other metrics than the sole accuracy.

In [57], the authors compare several metrics that could be exploited to compare different algorithms. After discussing accuracy-based metrics, they argue that in order to draw a reliable conclusion about the quality of recommendations, it is necessary that the recommender system should also be able to provide not only accurate but also useful suggestions. Indeed, an extremely popular item may be an accurate suggestion but not interesting for a user. Serendipity, novelty as well as diversity are alternative metrics to accuracy metrics. The serendipity metric [31] captures if a recommender system have the ability to recommend unexpected and attractive products. The novelty metric proposed in [144] measures the ability of a recommender system to suggest items that have a low probability of being known by a user.

In most of the research work carried out in recommender systems, the evaluation protocol is made in an offline setting where the above mentioned metrics are measured based solely on past interactions. However, this has shown to be not sufficient [69] in reality. More specifically, recommender system algorithms influence decisions made by users, thus their preferences, which in turn affect the data (generated from user interactions) used to train the recommender system creating a feedback loop. To overcome this limitation, evaluating the recommender system based on an online protocol becomes necessary. Some research works that use A/B testing technique to measure online business metrics as return on investment (ROI) or click through rate (CTR) have been published. In [21], the authors proposed to use an online metric called "online acquisition gain" for a Mobile Application Platform made available on the

Google Play Store[8]) to measure if there is a gain on the number of downloaded applications. They developed an A/B testing framework over three weeks and demonstrated that for their model called "Wide & Deep Learning", the acquisition gain was 3.9% greater than the previous model used in Google Play Store.

Evaluation Protocol and Dataset

Despite the relevance of using an online protocol to evaluate a recommender system, it is sometimes impossible or difficult to do so in the context of academic research because it requires having full control on a platform and respecting privacy regulations regarding the use of consumers' personal information[9].

In their work on Recurrent knowledge graph embedding for item recommendation [134], the authors use two real world datasets, namely MovieLens 1M[10] and the Yelp Challenge Dataset[11] to evaluate their approach and compare it with baseline models. They split the data randomly in the order of their timestamps and they used 80% of the feedback as training set and 20% as the test set.

In [55], the authors used the same MovieLens 1M dataset and Pinterest[12] to evaluate the performance of their model. They adopt a leave-one-out procedure which has been used in [120]. Technically, for each user, they hold-out the latest interaction and consider it in the test set following the strategy used in [37] which consists of randomly sampling items that are not interacted with by the user and rank the relevant item over these items.

In their work on property specific knowledge graphs [113], the authors used three different data sets: MovieLens 1M, LastFM[13] and LibraryThing[14] to evaluate their model. They splitted the data into three different sets: Training (70%), Validation (10%) and Test (20%). The idea of using a validation set is a technique used to avoid overfitting on the training set (once the training loss tends to differ a lot from the validation loss, then the training is stopped). They considered different metrics to evaluate their models, and in particular, the serendipity and novelty metrics in addition to the accuracy metrics (Precision, Recall and MAP).

More generally, in recent years, evaluation frameworks have been proposed to evaluate recommender systems and more particularly to provide the researchers with a tool to ease (and make them reproducible) all the experimental evaluation phases, from data reading to results

[8]https://play.google.com/store?hl=en
[9]https://eugdpr.org
[10]http://grouplens.org/datasets/movielens
[11]https://www.yelp.com/dataset_challenge
[12]https://sites.google.com/site/xueatalphabeta/dataset-1
[13]http://files.grouplens.org/datasets/hetrec2011/hetrec2011-lastfm-readme.txt
[14]https://www.librarything.com/

collection.

MymediaLite [42] is one of the first libraries in the field to provide a framework containing CF algorithms able to train on two different prediction tasks: rating prediction (e.g. on a scale of 1 to 5 stars) and item prediction from positive-only implicit feedback (e.g. from clicks or purchase actions), to the complete evaluation of the recommender systems. In [6], the authors propose *Eliott* an evaluation framework that optimizes hyper-parameters (51 strategies) for several recommendation algorithms (50 algorithms), selects the best models, compares them with the baselines providing intra-model statistics, computes metrics (36 metrics) spanning from accuracy to beyond-accuracy, bias, and fairness, and conducts statistical analysis (Wilcoxon and Paired t-test). Similarly, in their work around sequence-based recommender systems [102], the authors proposed an offline evaluation framework to evaluate any recommender system that returns sequence of items as output. They propose dividing the dataset randomly by allocating 80% for the training set and 20% for the test set and used metrics that cover novelty, accuracy, perplexity, and coverage, among others.

Considering the size of airlines' catalog (Section 3.1) and the use-cases that are addressed in this thesis (Chapter 4), we will focus on precision metrics such as $hitrate$ and MRR. Depending on the use-case, we will use either the leave-one-out evaluation protocol or a temporal split to constitute the training and test datasets. Finally, even if we believe that online evaluation is the only way to properly validate a recommender system [9, 106], we were not able to do so in this thesis for several reasons, including the fact that we do not have control over partner airlines' platforms and the Covid-19 conditions has slowed down the discussions with partner airlines to convince them to put into production the recommender systems developed in this thesis.

2.1.7 Summary

In this section, we have first introduced Recommender Systems using a definition making use of graphs. Then, we have presented a set of basic notions and concepts related to the field of Recommender Systems, illustrating the different families of algorithms used, and the most commonly used models for RSs. We have highlighted the pros and cons of the different algorithms, showing how the research trend is directed towards hybrid systems that combine the best of collaborative and content-based filtering. We have shown how KGs are ideal to incorporate different information in one single data structure and leverage those information to improve the recommendation.

Then, we have presented how we can recommend items in an online setting through SB recommender systems, a particular type of recommender system where basic algorithms can be extended and new algorithms based on recurrent neural networks are appropriate to use.

Finally, we have discussed the importance of RSs evaluation, by choosing the relevant metrics and evaluation protocol for each recommendation problem.

Beyond the different families of algorithms described in this section, the field of recommender systems is in constant evolution with more and more complex approaches being regularly proposed to address the limitations of the previous generation of algorithms. As an example, a promising research direction mixing reinforcement learning [136] and recommender systems [126, 174] is being explored with the ambition to focus on long-term returns and break the pernicious feedback loop of recommendation as described in [17].

In the next chapter of this thesis, we will discuss how we can transform the airline offer construction and retailing by enabling recommender systems in a new offer distribution mechanism. We will confront the families of recommender system algorithms presented in this section with a set of airline-specific use-cases, and propose a theoretical systematic approach on how we can match those.

2.2 Knowledge Graphs

The term “Knowledge Graph” was popularized by Google in 2012 to describe its graph-structured knowledge base containing hundreds of millions of entities and relationships. It has played a major role in the company’s search engine since 2012, providing detailed answers to search queries questions by displaying a sidebar containing specific information about entities mentioned in users queries [112].

More generally, a knowledge graph is a graph-structured knowledge base that stores factual information in the form of relationship between entities (or literal values), enabling the modeling of real world entities and their relations, and consequently powering search engines, natural language understanding systems and more recently recommender systems [149].

2.2.1 Principles

A number of definitions have emerged from prior works varying from specific technical definitions to broader general definitions. In this thesis, we adopt the same definition as in [60]: ‘A knowledge graph is a graph of data intended to accumulate and convey knowledge[15] of the real world, whose nodes represent entities of interest and whose edges represent relations between these entities’. Knowledge can be extracted from external sources, or from the knowledge graph itself.

[15]A number of specific definitions for knowledge have been proposed in the literature on epistemology

Knowledge graphs are usually assembled from multiple data sources, and as a result, can be highly diverse in terms of structure and granularity. Hence, the use of semantic web technologies and ontologies is required to rigorously build a knowledge graph. In practice, we can categorize the existing knowledge graphs into two main types [60]:

- *Open knowledge graphs* that are available online and constructed through communities of volunteers or automatically built from other sources such as Wikipedia. Dumps of these knowledge graphs are available in the web and can used by anyone. Among these knowledge graphs, we note those that are well known and widely used by the research community such as DBpedia[16], Wikidata[17], YAGO[18], etc. and a selection of knowledge graphs published within specific domains such as media [119], geography [133] and also tourism [90, 103].
- *Enterprise knowledge graphs* that are internal to a company and applied for commercial use-cases such as Airbnb [18] or Amazon [81] knowledge graphs.

In this thesis, we will make use of different publicly available open knowledge graphs [103, 145] to build our own enterprise knowledge graph used to address the airline specific recommendation use-cases defined in section 3.3.

A number of data models are reported in the literature to build a knowledge graph from existing data sources. Inspired by the work presented in [60], we list below the different existing options:

- *Multi-relational graphs* are directed labeled graphs where the nodes represent the entities and each edge between two nodes is labeled by a relation. This data representation offers a high flexibility in integrating new data into the knowledge graph in comparison with other data structure such as relational data bases, trees, etc. Resource Description Framework (RDF)[19] is the standard semantic web technology based on directed labeled graph and recommended by W3C. RDF is based on the idea of making statements about resources (in particular web resources) in expressions of the form subject–predicate–object, known as triples. The subject denotes the resource and can be represented either by a Uniform Resource Identifiers (URIs) which allow global identification in the web or to represent blank nodes, and the predicate denotes traits or aspects of the resource, and expresses a relationship between the subject and the object which can be represented as an URI or a literal which allow for representing strings (with or without language tags) or another datatype value (integers, dates, etc.).

[16]DBpedia:https://www.dbpedia.org/
[17]Wikidata: https://www.wikidata.org/wiki/Wikidata:Main_Page
[18]Yago: https://yago-knowledge.org/
[19]RDF: https://en.wikipedia.org/wiki/Resource_Description_Framework

- *Heterogeneous graphs* are graphs that contain typed nodes and edges. An edge is called homogeneous if it is between two nodes of the same type (e.g. borders); otherwise it is called heterogeneous. However, they support only a one-to-one relation between types and nodes.

- *Property graphs* were introduced to provide additional flexibility when modeling more complex relations. A property graph allows a set of property–value pairs and a label to be associated with both nodes and edges.

In this thesis, we choose the *multi-relational graphs* data model for its predominance in the research community and simplicity to represent data in the graph as triples and we use RDF framework to build our knowledge graph.

Once the data model chosen, the next step for building a KG is to define a semantic schema to ease the reasoning over the knowledge graph. A semantic schema allows for defining the meaning of high-level terms (vocabulary) used in the graph, which facilitates reasoning over graphs using those terms. A prominent standard for defining a semantic schema for RDF graphs is the RDF Schema (RDFS) standard [15], which allows for defining sub-classes, sub-properties, domains, and ranges amongst the classes and properties used in an RDF graph.

Defining the ontology to use in the construction of the knowledge graph based on the dataset is the logical next step, after having decided which schema to use. The Web Ontology Language (OWL)[20], recommended by the World Wide Web Consortium (W3C) and compatible with RDF graphs is the most widely adopted ontology language. Before building the knowledge graph, we need to define a set of elements that form the ontology. We start first by defining the classes to which the individuals belong to, as most often an entity description contains a classification of the entity with respect to a class hierarchy. Then, we define the properties that characterize the individuals. The properties can either be sub-properties of already existing properties defined in OWL (for instance) or can be defined by assigning a range and a domain.

In this thesis, we have often used the RDFs schema and the OWL ontology in the design of our own ontology.

Finally, interpreting non-existing triples in a knowledge graph is not obvious, indeed a triple (s, p, o) that does not exist in the knowledge graph does not mean it is not a true relationship. Two alternative assumptions are normally proposed namely Closed world assumption (CWA) where non-existing triples indicate false relationships and Open world assumption (OWA) where non-existing triples do not mean that the relationship is false or true, but simply unknown. Given that large-scale web KGs are incomplete, Semantic Web and RDF follow the

[20]OWL: https://www.w3.org/OWL/

OWA, which we also follow in this thesis.

2.2.2 Knowledge Graphs in Tourism

Planning a visit to the Louvre Museum or simply a visit to the city of Paris requires some prior knowledge of the events and activities taking place in the city, the places and sites to be discovered, the means of transportation to be used to get around, etc. All this information can be collected and represented in a knowledge graph within the domain of tourism. In recent years, several tourism knowledge graphs [73, 97, 155, 171] have been created to help Destination Marketing Organizations (DMOs) to promote more easily their destinations to leisure travelers thanks to the wealth of content available in knowledge graphs. In [142], the authors presented the process of building such comprehensive knowledge graphs collecting data from numerous static, near-and real-time local and global data providers, including hyper local sources. In [91], the authors proposed to build a travel knowledge graph collecting data coming from DBpedia, Wikidata and Geonames in order to group data about travel attractions with the objective to recommend these attractions to tourists.

Industrials in the tourism sector such as Airbnb [18] also started to build their own knowledge graph to transform trip planning to be more intuitive and personalized than before. Airbnb wants to provide the inspiration for the trip and then inform people about how to make the most of their trip to a particular destination by recommending local travel content to their travelers among other suggestions. Even Google are now suggesting tools to their users to help them find activities to do on the destination, or to plan their trips through trip planner tools or even book an hotel, all this thanks to their knowledge graph.

More recently, in [103], the authors used Location-based Social Networks to build users' trails. A trail is a succession of check-ins (the user share his location) made by a user in a venue. Each venue is categorized using Schema.org[21] (Restaurant, Civic structure, etc.) and foursquare[22] category which which has a higher level of granularity (Italian restaurant, Indian restaurant, etc.). The authors released Semantic Trails Datasets[23] (STD) which is a knowledge graph that contains data for users' check-ins collected in 2013 and in 2018 foursquare. The knowledge graph represents the interaction between users and point of interests, through the property 'visiting' as well as the relations between the venue and the other entities, namely: category, schema and city. Some research works have followed [111] to propose to travelers 'paths' of activities and places to visit based on STD knowledge graphs.

In this thesis, we collect data from the different knowledge graphs mentioned above to con-

[21] Schema.org: https://schema.org
[22] https://foursquare.com/
[23] Semantic Trails Datasets: https://figshare.com/articles/Semantic_Trails_Datasets/7429076

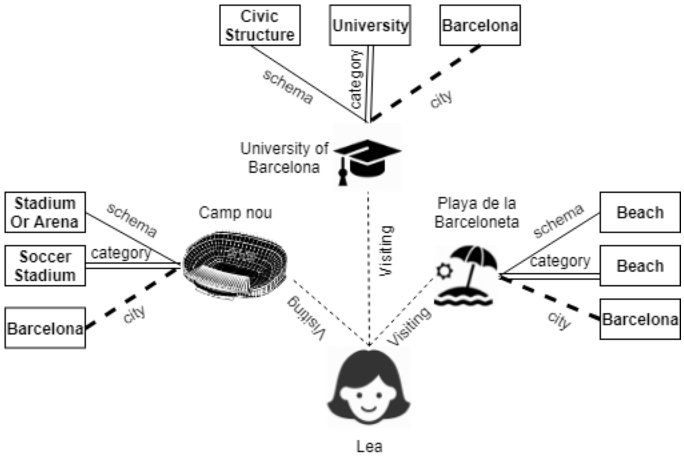

Figure 2.6 – Semantic Trail Knowledge Graph

struct our own knowledge graph that contains not only airlines' data but also external data.

2.2.3 Knowledge Graph Embeddings

Knowledge graphs are effective in representing structured data and incorporating data coming from different sources, however the underlying symbolic nature of knowledge graph triples usually makes KGs hard to manipulate for Machine Learning applications.

A Knowledge graph embedding (KGE) is a representation of a KG element into a continuous vector space. The objective of learning those embeddings is to ease the manipulation of graph elements (entities, relations) for prediction tasks such as entity classification, link prediction or recommender systems.

Most of the proposed methods rely solely on graph triples with the goal to embed KG entities and relations into continuous vector space. The idea is to preserve the inherent structure of the KG and simplify the use of KG elements. Once KG elements are represented as embeddings, a scoring function is used to measure the plausibility of a triple.

Embeddings have become popular thanks to the release of Word2vec [101] in 2013. Word2vec efficiently learns word embeddings by training a shallow neural network to predict the context of a word included in a vocabulary, defined by a sliding window of amplitude c with the key idea to preserve the semantic of the words. Two different architectures are proposed as shown in figure 2.7, namely *Continuous Bag-of-Words* [99] (CBOW) that implements a neural network where the input corresponds to the context words $w_{t-c}, w_{t-c+1}...w_{t+c-1}, w_{t+c}$ and the output to predict is the target word w_t and *Skip-Gram* [101] that implements a two layer neural network where the input corresponds to the target word w_t and the output to the context words $w_{t-c}, w_{t-c+1}...w_{t+c-1}, w_{t+c}$.

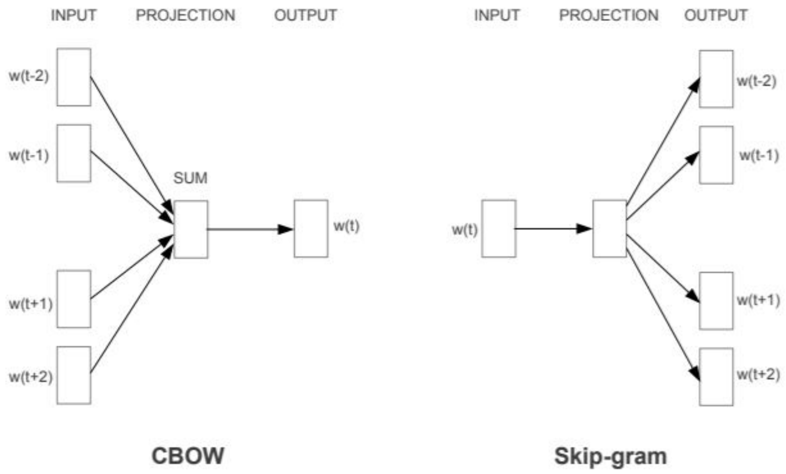

Figure 2.7 – The CBOW architecture predicts the current word based on the context, and the Skip-gram model predicts surrounding words given the current word. Source:https://arxiv.org/pdf/1301.3781.pdf

Following the same logic, the authors of DeepWalk [118] and node2vec [48] generalized embeddings to graphs by suggesting to make use of neural language models such as Word2vec to build graph embeddings. In DeepWalk, the authors proposed to extract sequences of nodes - which represent entities - in the graph by relying on a random uniform walk in the graph. This sequence of nodes can be seen as a text, and then CBOW or SkipGram is applied to construct embeddings of these nodes. Node2vec went further by introducing a more sophisticated random walk strategy that can be more easily adapted to a diversity of graph connectivity patterns, outperforming DeepWalk in link prediction and knowledge graph completion tasks.

Considering only the graph structure to encode KG elements is nevertheless not sufficient, hence other methods [12, 131, 151] have emerged to consider also properties and entity types of the graph. In [148], the authors classified the knowledge graph embedding algorithms into two main categories namely *translational distance models* that are based on a scoring function that measures the plausibility of a triple by measuring distances in the vector space, typically after performing a translation operation and *semantic matching models* that are based on a similarity-based scoring function that measures the plausibility of a triple by matching the semantics of the latent representations of entities and relations.

For the first category, TransE [12] is often mentioned as the most used translational distance model. TransE represents both entities and relations vectors in the same space R^d. Given a triple (s, p, o), the relation is interpreted as a translation vector r so that the embedded entities s (subject) and o (object) can be connected by p with low error, i.e., $s + p \approx o$ when the triple (s, p, o) holds in the knowledge graph. In other terms, the goal is to minimize the scoring function represented below.

$$f_{TransE}(s,p,o) = ||\vec{w}_s + \vec{w}_p - \vec{w}_o||_{L_{1,2}} \tag{2.9}$$

TransH [151] introduces relation-specific hyper-planes, each property p being represented on a hyperplane as w_p its normal vector. TransR [86] follows the same idea as TransH, but instead of projecting the relations into a hyper-plane, it proposes to create a specific space per relation. We represent in figure 2.8 the embedding space of the different translational distance models presented above.

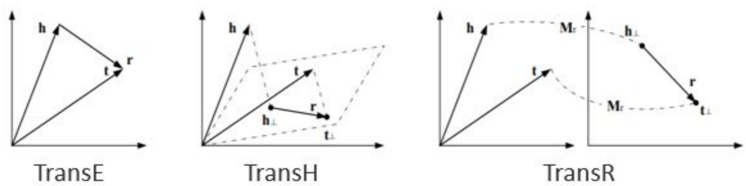

Figure 2.8 – Distance between embeddings are computed in the same embedding space for TransE regardless the relation while for TransH and TransR, they are computed in relation specific spaces. (h, r, t) is a triple in the KG. Source:https://persagen.com/files/misc/Wang2017Knowledge.pdf

On the other hand, semantic matching models exploit similarity-based scoring functions. In [107], the authors proposed *RESCAL*, a model that associates each entity with a vector to capture its latent semantics. Each relation is represented as a matrix that models pairwise interactions between latent factors. The score of a triple (s, p, o) is defined by a bi-linear scoring function minimized through tensor factorization based on ALS optimization technique. Other methods that extend RESCAL emerged. *NTN* [131] (Neural Tensor Network) is a neural network that learns representations using non-linear layers. ER-MLP (Multi layer perceptron) proposed in [35], where each relation (as well as entity) is associated with a single vector. More specifically, given a triple (s, p, o), the vector embeddings of s, p, and o are concatenated in the input layer, and mapped to a non-linear hidden layer. *DistMul* [160] simplifies RESCAL by representing relations with diagonal matrices, thus reducing its complexity. *ComplEX* [143] extends DistMul using complex numbers in place of real numbers. As mentioned in [112], recent work has dropped the assumption of embedding in a Euclidean space, showing that using hyperbolic spaces can lead to better performance, especially in modeling hierarchies [108].

We use knowledge graph embeddings as a mean to represent KG elements and use them as input of recommender system algorithms to address some airline specific recommendation use-cases as shown in section 5.2. Recently, a large number of new knowledge graph embedding [30] algorithms have emerged, but we highlight only those used in the thesis.

2.2.4 Summary

In this section, we have introduced some concepts and notions of semantic web and knowledge graph used for the construction of our knowledge graph. Then, we presented a set of knowledge graphs in the tourism domain that will be used and integrated with other data sources as part of the knowledge graph construction. Finally, we presented a literature review of knowledge graph embeddings which are a mean to manipulate knowledge graph elements and use them in recommender system algorithms. In this thesis, knowledge graphs are used as a mean to integrate data coming from different sources and used as input of recommender systems. Knowledge graph embedding algorithms are used to generate KG embeddings as input of recommender systems 5.2. In chapter 5, we describe how we build the knowledge graph that is used as input of KG recommender systems designed to overcome limitations of traditional recommender systems (e.g. CF, hybrid RSs) described in the conclusion of chapter 4.

Chapter 3

Recommender Systems in the Airline Travel Industry

In this chapter, we focus on the airline travel industry by first presenting its specifics in order to understand what is different from other industries that make regular use of recommender systems. Then we provide the necessary background for understanding the objectives behind the new distribution standards, known as the New Distribution Capability (NDC), and more especially how NDC can simplify the adoption of recommender systems in the airline industry. Afterwards, we list some airline specific recommendation use-cases and discuss how they can be implemented in practice using the families of recommender system algorithms described in section 2.1.

3.1 The 4Ws of the Airline Industry

With the digital transformation of retail commerce from physical sale points to virtual stores, recommender systems have shown their value in easing the search and purchase decision process of customers who are facing an ever increasing amount of products [25].

By selling more tickets online, airlines are also moving towards digitization, whether through their websites, direct online providers or even online travel agencies, allowing them to move from physical sales points to online sales. While this transformation is taking place, the airline industry has become the leading online industry in the travel sector. The online sales now represents more than 55% of total sales in the Europe travel market and more than 61% in the US market[1]. Moreover, airlines are now using different sales channel to reach travelers: Mobile applications through push-up notifications, online devices through email marketing campaigns, social media, etc. These latest developments give the airlines the opportunity to reach more and more travelers but also raise some questions for airlines that are new to

[1] https://www.phocuswright.com/US-Airlines-2019-Market-Sizing-and-Landscape/US-Airlines-2019-Market-Sizing-and-Landscape/39217?subr=1

the online retail: **w**hat product to offer, to **w**hich customer, **w**hen to recommend an offer, at **w**hich price, and finally how this offer should be presented to the customer and on which touchpoint.
Thanks to the New Distribution Capability (see section 3.2), airlines have never been that close to addressing these questions by enabling dynamic offer construction and dynamic pricing through machine learning techniques and recommender systems.

The sale of flight tickets remains the main activity of airlines because first and foremost the airplane is a means of transportation like many others. Sales of flight tickets represented in 2018 89.3%[2] of total revenue for airlines, however recent studies, from the International Air Transport Association (IATA) & Mckinsey Group [10], revealed that airlines can enhance their revenue by 7$ per passenger, by putting in place modern retailing techniques which comprises:

1. Creation of new offers (1) and creation of product bundling (2) through the dynamic offer build component.
2. Use of better targeting and retailing techniques (3) through the offer retailing component.
3. Enhancement of revenue management through dynamic pricing techniques (4) through the dynamic pricing offer component.

These components are all part of the new logic of Dynamic offer execution which constitures one of the main strength of NDC (see section 3.2). (1), (2) and (3) are what motivate the use of recommender systems in the airline travel industry. It will allow airlines to create the offer (**what**), target best the audience (**who**) for a given offer, use optimal sales channel to sell their products (**where**), deliver recommendation at best time (**when**). Answering the 4Ws questions consists in developing a recommender system that acts in the offer creation stage (what & who), and in the offer retailing stage (when & where) of the new airline offer management system (figure 3.2).

In the last two decades, recommender system research community have explored plenty of methods to improve the accuracy of recommender systems, leveraging historical data by using cutting edge AI algorithms. However, very little research has been carried out in the travel sector [125], and reasons are manifold: a difficult access to the data, the nature of data that implies cold start problem, data sparsity as mentioned in section 1 and a small number of products in the airline catalog.

Airline catalog comprises air tickets, but also ancillaries that represented 10.7%[3] of airline

[2] https://www.cartrawler.com/ct/ancillary-revenue/2019-cartrawler-ancillary-yearbook/
[3] https://www.cartrawler.com/ct/ancillary/cartrawler-ancillary-revenue-yearbook-ideaworkscompany/

revenue in 2018 as shown in figure 3.1. The increase of ancillary revenue as a percentage of total revenue (from 6.5% in 2010 to 10.7% in 2018) and the increase of the average ancillary revenue per passenger that jumped from 12.68$ in 2010 to 17.02$ in 2018 demonstrate that airlines have put an effort to sell more ancillaries over the years.

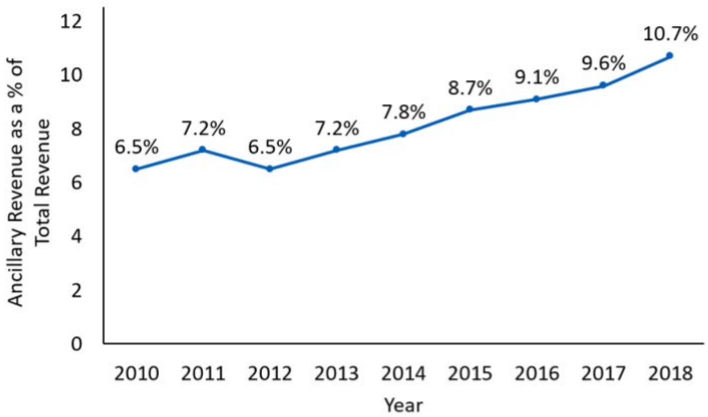

Figure 3.1 – Trend of Ancillary revenue as a percentage of total revenue. Source: CarTrawler Ancillary Yearbook

Despite this willingness to sell more ancillaries, from a scientific perspective one can wonder why we need to use sophisticated machine learning-based recommender systems considering the small catalog of products and the highly imbalanced airline sales as shown in figure 3.2.

The answer is that a recommender system is not only needed to filter out useful products for a given user from a large catalog, it is also used to send right offer to the right customer, to reach traveler in optimal time, and create new offers through bundling techniques. Moreover, the move of airlines to NDC and creation of new offers which will increase the airline catalog justify current recommender system needs in the airline travel industry.

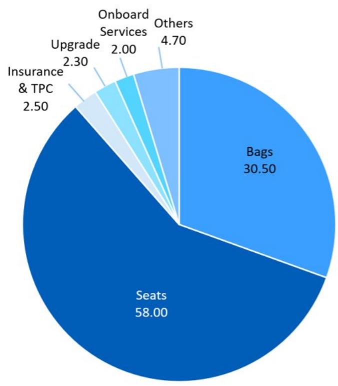

Figure 3.2 – Airline Ancillary sales distribution

Currently, very few airlines are using recommender system techniques based on a survey [41], where 45 representative airlines were studied 3.3; indeed only 10% are effectively using recommender systems. The same study has revealed that airlines promote their ancillary products at different stages of traveler journey, therefore there is an opportunity to make use of recommender systems to send the offer in the optimal time. More than 90% of the airlines are using email marketing campaigns to promote ancillaries, however, they do not use recommender systems to select the right offer that should be put in the email. Finally, over 60% of the airlines surveyed are doing travel packaging which again opens the door to creating dynamic packages through recommendation system techniques.

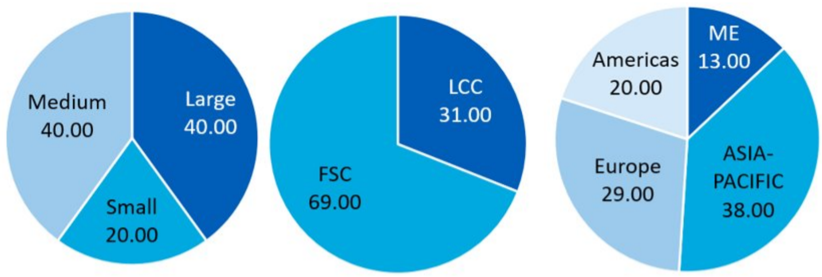

Figure 3.3 – Distribution of studied airlines by size, market and type

From a traveler point of you, finding the right offer each time we want to travel can get very frustrating and turn into a very complex and time-consuming problem; Recommender systems can be adopted as it will ease the search task by getting personalized offers based on our preferences and will make us more confident that we are getting best value each time.

From the airline point of you, understanding travelers' motivations is not obvious; In [67], the authors analyze "how do people make choices": if a traveler is an attribute-based decision maker, then these attributes reduces the total set of options to a smaller consideration set on the basis of items attributes, typically if the traveler is confronted between flight A that arrives at 8 pm and costs 120$, and flight B that arrives at 11:30 pm and costs 90$, the traveler will choose flight B because it is cheaper, while a traveler that is a consequence-based decision maker who evaluates and anticipates the consequences of an action, will choose flight A to catch last train in the train station.

In addition to understanding what drives human decisions, understanding the trip purpose, identifying the social, geographical and more generally the users' demographics is key to recommend the right destination or the right ancillary product to the user. In [2], the authors claimed that travelers' motivations are complex to identify. Indeed, if the traveler is a student, we probably want to sell an adventure trip, if the travelers are parents accompanied by their children, we probably want to sell basic ancillary needs: Food, water, comfort, etc.

An European airline travel market surveys that 55% of people are traveling for relaxing reasons, while 38% are traveling for visiting family, and 24% are traveling to find a romantic gateway.

Each destination in the world is characterized by various places to visit and activities to do (museums, beaches, etc.) and therefore can be the reason for travel. Thus, when recommending a destination, we need to identify what drives the user to travel and take into account what characterizes a destination in order to recommend the appropriate destination. Indeed, knowing the purpose of the trip is essential, for example millennials travel more than other generations, and on average travel two times more for leisure trip than business trip [38]; this study shows that millennials value experiences over things: 70% of millennials agree they would rather spend on amazing experiences vs. things.

When it comes to trip planning and even booking, the most used channel is online travel agency websites for all generations before metasearch websites and airline websites; This helps airlines identify what channel to target. Moreover, 75% of travelers, said would like to receive contextualized and personalized offers. The tendency is now to move to 'à la carte' sell of ancillary products which can make it complicated from an airline point of view, as it is not a straightforward task to de-bundle the existing fare families; the use of a recommender system can help to suggest the right ancillary for a traveler in a given context (see section 4.2). All the aforementioned elements demonstrate the need of enabling recommender systems to find the right offer for the right traveler based on the travel context and choose the right retailing technique to propose the offer.

However, in the current airline distribution model, airlines have delegated control of the offer construction to content aggregators, such as Global distribution systems (GDSs). Real-time interactions with the airline systems are quite limited, and the pricing function which is used to create offers on behalf of the airline is governed by industry standards that only enable very few parameters to differentiate the content based on who the traveler is. Therefore, airlines cannot provide personalized and contextualized offers in a meaningful way. Moreover, the responsibility of the offer construction and retailing has historically been managed across separate departments within the airline organization. Offer construction and retailing were therefore never part of a broader and holistic customer experience management strategy.

We believe the current approach is inadequate and that the key to profitability is to manage offers consistently in an integrated Offer Management System (OMS) serving the customer throughout the traveler journey from inspiration to post-trip. This advancement will happen as part of IATA's New Distribution Capability, which will allow airlines to move towards customer centric airline retailing. NDC is an enabler for the application of airline OMS including recommender systems. Industry adoption of NDC has continued to grow in recent years. As of August 2020, 40 airlines, 20 aggregators and 10 sellers are NDC certified level 4 (the highest level) covering booking of NDC content as well as supporting changes of the order [65].

3.2 Towards a New Distribution Capability in the Airline Industry

In this section, we first detail the traditional airline distribution model which will provide the necessary background for understanding IATA's NDC, which we discuss subsequently. We demonstrate that NDC is an enabler for the application of the airline OMS including recommender systems.

3.2.1 Traditional Distribution Model

Figure 3.4 shows how a customer's request for an itinerary is passed from a retailing platform (Airline Retailing platform, or Other Retailing platforms), possibly through a distributor, and to the airline's Inventory system for evaluation, using the distribution model in place today. For the direct channel (Direct Connect), the airline fully controls the shopping and pricing flow. However, for the indirect channels, the current distribution paradigm relies on a two-step process. First, the airline files fares with data distributors such as ATPCO or SITA. These filed fares drive the construction and pricing of the products that can be offered to the customers. Then, the availability computation within the airline's Inventory system (Flight Execution) determines which of the filed fares are made available for sale. The airlines control the availability computation via their Revenue Management Systems (RMSs), which essentially can be performed using offline optimization (Airline planning).

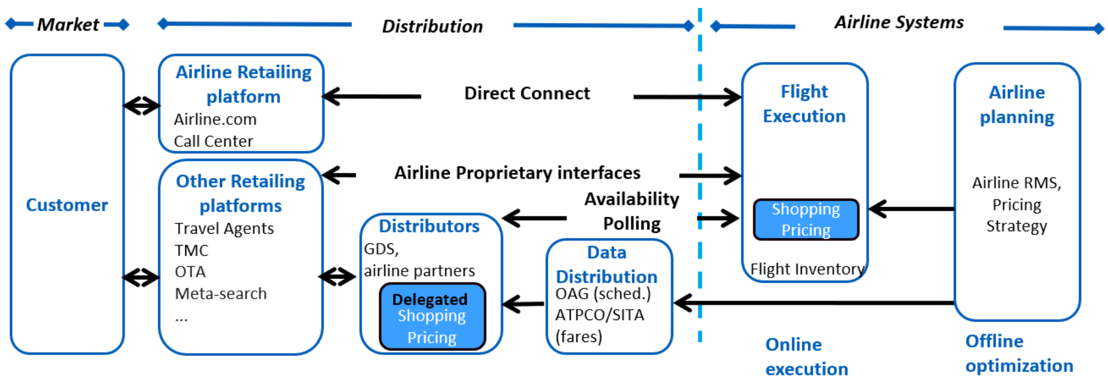

Figure 3.4 – Traditional Distribution Model

Other retailing platforms may interact directly with the airline's Flight Execution layer via proprietary interfaces. Distributors such as the GDSs acquire the filed fares content and have the authorization to build offers on behalf of the airlines (Delegated Shopping & Pricing). The distributors then poll the airline's availability to determine which fare products are available for sale. Consistency across indirect channels is enabled by highly standardized content and associated processing logic that the GDSs adopt and implement when accepting airline content and developing their shopping and pricing engines. This means that there is a limited ability for customer-specific information to be used in the indirect distribution channel. In principle, even if the airlines could create contextualized and personalized offers in the direct channel, this would create inconsistency that cannot be resolved among the distribution channels.

3.2.2 New Distribution Capability (NDC)

The New Distribution Capability is a set of new technical communication standards that was initiated almost a decade ago by the IATA. The vision with NDC is to modernize airline distribution and enable airlines to have better control of their offers and their retailing. We list below the most important benefits for airlines that are adopting NDC, which are of particular relevance for this thesis. For further information on the objectives and benefits of NDC, we refer the reader to [61].

- **Personalized and contextualized offers**. The airlines will have access to customer and contextual information in a shopping or booking request, which will allow for personalized and contextualized offers.
- **Dynamic Offers**. The airlines will be able to create, distribute, and fulfill dynamic offers as described in the next section.
- **Dynamic Pricing**. The airlines can employ dynamic pricing using a continuous price.
- **Retailing**. The airlines can provide the retailing platforms with product description that encompasses retailing preferences and information. For instance, rich media content that further complements their offers using visual elements, such as infographics, photos, videos, etc.
- **Merchandising**. The airlines will be able to employ merchandising techniques to affect customers purchase behavior.

Figure 3.5 shows how airlines are aspiring to take control of the offer creation, at scale and across all distribution channels.

In the NDC environment, airlines still make the decision of distributing via direct channels and/or via indirect channels with third-party intermediation. However, delegation of the offer creation to intermediaries no longer exists. Instead, each customer shopping request in an agent's front-office system is passed to the airline's OMS, either directly in the case of NDC Direct Connect distribution, or via an aggregator in the case of NDC Intermediated distribution. Note that the Airline Proprietary Interfaces and Availability Polling arrows in figure 3.4 have been replaced by NDC Direct Connect and NDC Intermediated arrows in figure 3.5, enabling a cost efficient deployment at scale for the distribution network actors.

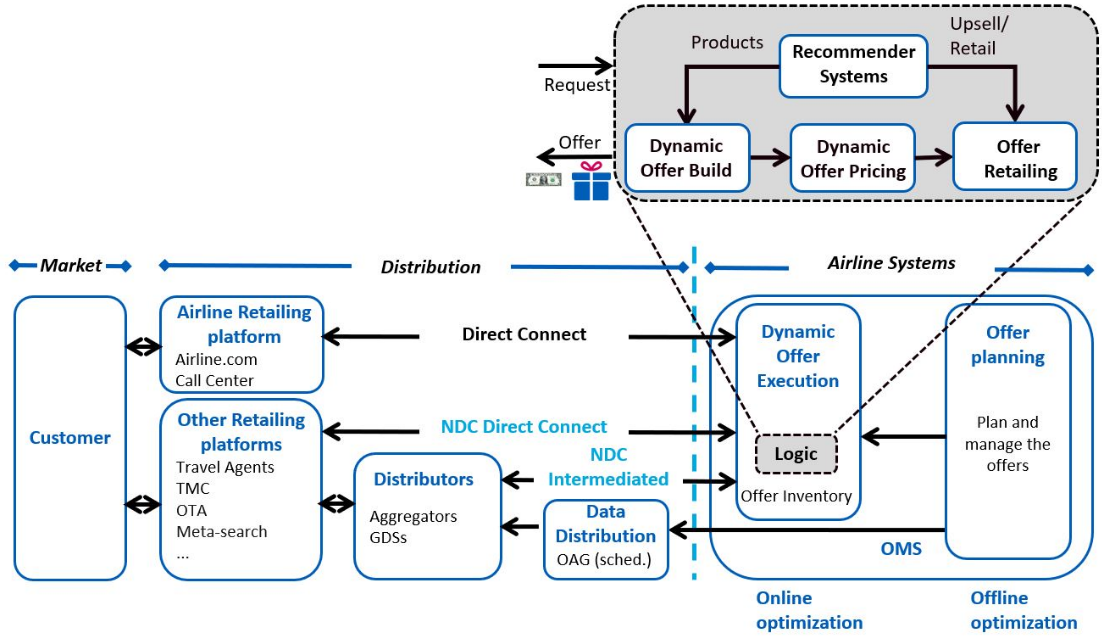

Figure 3.5 – Distribution model using NDC

The airline's OMS creates a set of one or more offers that are returned to the customer. Each offer is individually tagged with an offer ID that can be used in any subsequent request on that offer. If the customer accepts an offer, the offer is converted into an order and the contract with the customer is established.

3.2.3 The Offer Management System (OMS)

As seen in figure 3.5, the airline's OMS controls the offer construction and retailing for both the direct channel and the indirect channel in NDC. We can think about OMS as an extension of the airline's RMS in several dimensions.

The main extensions are as follows. First, RMS optimizes only the prices (actually the availabilities) of the pre-filed flight products, while OMS optimizes both product components (flight products, ancillaries, third-party content) and prices. Second, unlike RMS which provides the same price to all customers for a given flight and fare product, OMS may differentiate among customers and construct personalized and contextualized offers. Third, and not considered by RMS, OMS may construct one or multiple offers in a so-called *offer set* that will be displayed together as options. For further information, we direct readers to [40].

Finally, because RMS does not differentiate among customers, the price computation can essentially be pre-computed during the offline optimization processes and the on-line process is a lightweight execution logic. For OMS, this is not the case, as computing personalized and contextual offers is designed to be a real-time decision and the optimization logic must be moved to the online domain. This has significant ramifications for the IT system design of the OMS, which we will discuss below.

The online optimization logic of the OMS is comprised of the following components, which is illustrated in the inset in figure 3.5. In particular, we would like to draw attention to the role of recommender systems in guiding both the Dynamic Offer Build and the Offer Retailing, which has also been exemplified with the recommender system use-cases presented.

- **Dynamic Offer Build**. This module makes the determination of the relevant set of products (flights, ancillaries, and third party content) to be returned at the individualized customer level.
- **Dynamic Offer Pricing**. This module takes as input the offers that were built by "Dynamic Offer Build" and determines for each of these offers the selling price that maximizes the contribution considering both customer and contextual information.
- **Offer Retailing**. This module aims to increase conversion rates by applying merchandizing techniques to affect the customer's purchasing behavior.

In the description above, we have seen the different functional steps of an OMS to dynamically construct, price and retail an offer. However, we also need to consider the ecosystem that will trigger and support this process. In particular, online search engines have strict performance requirements. As these engines generate thousands of search transactions per booking, these IT systems need to be extremely cost-effective, scalable and resilient, to provide real-time dynamic offer construction and retailing while providing consistency across all distribution channels. Recent advancements in technology and infrastructure capabilities can enable airlines and system providers to accomplish these goals. For example, cloud infrastructure and real-time worldwide data synchronization and processing power allow data centers across continents to host and run local instances of the online optimization logic, accessible to any distribution channel, while continuously being under airline control.

3.3 Enabling Recommender Systems across the Traveler Journey

The traveler journey is a key consideration to understand the customer needs and intents (figure 3.6). Research from Frost and Sullivan [94] indicates that there "are certain moments when the customer is in a purchasing mind-set and thinking about his trip and what he will need". For example, at the booking stage, the customer is in a "planning" mind-set. At this stage, the airline can approach the customer with more "expensive" offers such as cabin upgrade, or flexibility options. Close to departure (48h/24h), the customer has a different mind-set - making the final preparations for his trip. At this moment, airlines could propose the customer with extra baggage, airport transfer, parking, priority check-in, or fast track access. In this section, we detail some use-cases for recommender systems along different phases of the traveler journey.

In order to provide more in-depth discussion, we focus on recommender systems that are under airline control. These use-cases cover customers that actively search and book travel products through the standard distribution channels enabled by NDC – both direct and indirect channels. Thus, use-cases for recommender systems regarding customer acquisition through the Internet giants' web interfaces, social media, and search engines will not be covered, since in these cases, the recommender systems reside outside the airline's control.

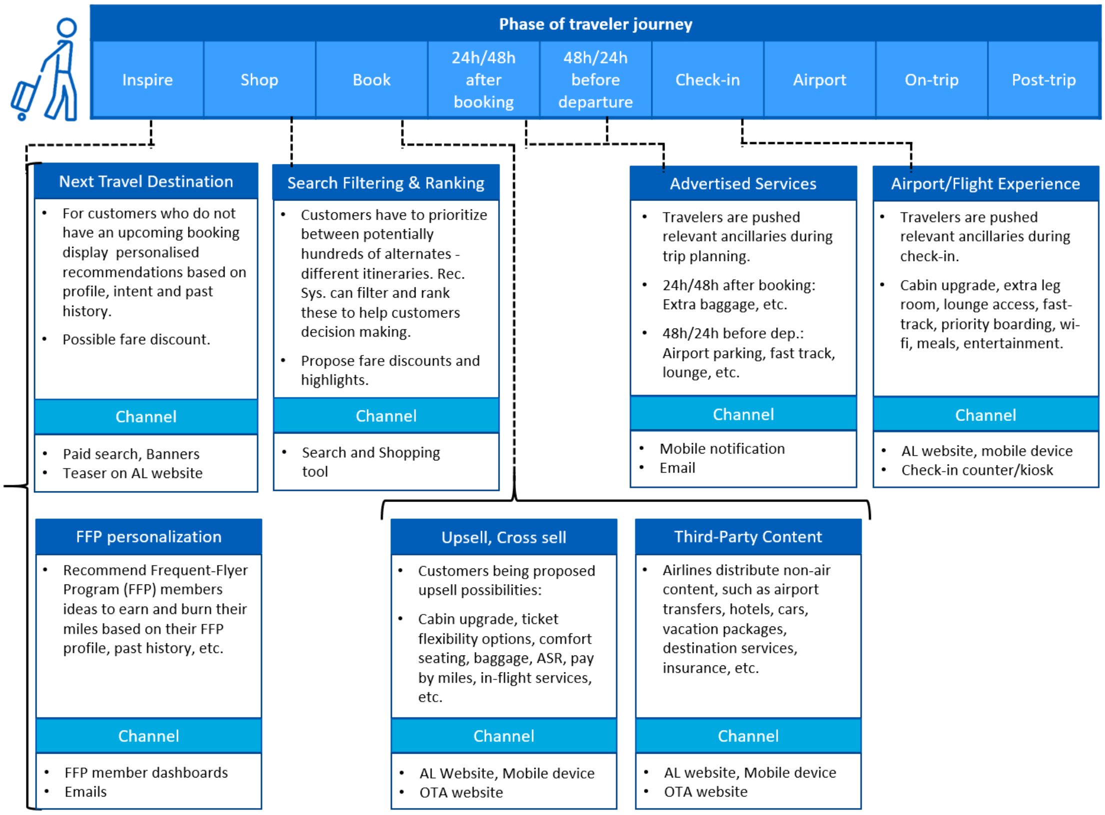

Figure 3.6 – Recommender system use-cases throughout the traveler journey

Next Travel Destination

The inspiration phase is a key opportunity to influence the customer decision making process. We distinguish between *passive inspiration* and *interactive inspiration*. The former represents the case when a customer (typically anonymously) lands on a web page (or receive marketing emails) and receives travel inspiration simply because some routes are popular in general, while the latter corresponds to the case where the customer interacts with the recommender system by providing personalized search criteria. In the following, in order to be concrete, we take the assumption that the customer stays anonymous and is engaged in interactive inspiration, providing the recommender system with more leverage.

Affinity shopping tools can be employed to create a personalized shopping experience. Rather than selecting the traditional criteria of origin/destination and calendar dates, these tools enable inspiration based on personalized criteria, such as customers' budget and interests (events or destination type such as beach, city, etc.). A recommender system with access to information of upcoming events (e.g. jazz festivals, sport events, exhibitions, etc.), and real-time information about flight prices and promotional fares (campaigns) could be used to recommend the most appropriate destinations and dates that match the customers criteria. Further, it could also recommend how the offers should be retailed using rich format such as infographics, photos and videos. For example, a trip during the summer to Nice Côte d'Azur in France, should have a very different presentation depending on if the customer is interested in beach, nightlife or a culinary experience.

FFP Personalization

The Frequent-Flyer Program (FFP) business model is dependent on FFP members having sufficient incentive to earn and burn their points. However, in reality, this may not be so easy. Premium-tier members with large point balances may not be able to find availability on attractive flights or premium classes due to blackouts or lack of award availability, while low-tier members with small point balances often cannot afford a redemption ticket and see no value in the program.

Recommender systems are in a good position to increase the number of points burned by using information about both the members' point balance and the availability of award tickets. For example, the premium-tier member may be offered to burn points for upgrades for his/her family on their annual vacation trip (to mitigate the dilution risk of the award ticket substituting a commercial ticket) or non-air content not readily accessible for purchase on the open market (e.g. backstage passes to concerts, games, etc.). For the low-tier member, recommender systems could offer a "discount" towards the fare of a commercial ticket.

Several other use-cases for recommender systems can also be identified, such as incentivizing members to earn points to reach the next tier level or burn points that are close to expiration. In all these cases, the system may be able to increase the value of the program by sending personalized emails to members with the right offer at the right time.

Search Filtering and Ranking

For a customer who makes searches by comparison shopping, booking air travel can be a daunting experience. He or she must prioritize among potentially hundreds of itineraries, with different prices and product characteristics across multiple partner airlines. As a result, it becomes almost impossible for the customer to make a purchase decision. Today, most search algorithms aim at finding the lowest fares but, in doing so, create irrelevant or unattractive itineraries that distract or overwhelm the customer.

A recommender system can filter the choice set into a manageable number of alternatives and rank them in order of relevancy based on an understanding of the customer's stated criteria. In this way, the recommender system both guides the customer in his decision process and benefits the airline through improved conversion rates. We may also add new customized criteria beyond the usual origin-destination, date range, flying time, ground time and overnight stay criteria to incorporate product attributes such as cabin, ticket flexibility, seat reservation and baggage allowance that are not typically considered in comparison shopping requests today.

Upsell, Cross-sell and Third-Party Content

When the customer has decided on his preferred itinerary, he enters the booking stage. During the booking stage, the recommender system has ideal information about the customer and his travel party – not only the current trip destination, duration, and already-selected ancillary services, but also the customer's profile and historic purchases. At the booking stage, the customer is in a planning mindset and this is an ideal opportunity to both increase ancillary revenues for the airlines as well as offer a one-stop shopping experience that covers the customer's full journey.

Examples of products that could be recommended at this stage include upsell offers such as cabin upgrades or ticket flexibility options, as well as cross-sell offers such as baggage, advance seat reservations or in-flights services (e.g. meals). In addition, the airline can also offer third-party content. Based on the customer needs, the commercial relation with the third parties, the prices and availabilities for the relevant resources, the recommender system can propose simple products such as insurance, airport transfers, etc., or even more complex bundled travel such as vacation packages that include hotels and rental cars.

Advertised Services

During the post-shopping period, the airline has an opportunity to push offers to customers through unsolicited mail or via notification on a mobile device. This period is a critical phase for the customers' last-minute decisions and preparations for their trip. Customers can be approached with ancillary services such as extra luggage, airport parking, seat selection, priority check-in, etc., and also be informed of availability of cabin upgrades that are aligned with their preferences. Again, the offer and communication would be very different between a family of four traveling long-haul from Frankfurt to New York City in economy class for a two weeks' vacation, versus a business purpose customer traveling the same itinerary and cabin, but staying only for two days. A recommender system would propose not only the most relevant offers but also the most relevant channel and time to push these offers with the benefit of increased adoption rates and customer satisfaction.

Airport/Flight Experience

During check-in, the customers actively interact with the airline via employees at the check-in counter, the kiosk, or on mobile devices. During this phase, the customer is focusing on the practicalities before takeoff. This may regard logistics of how to navigate through the airport, but the customer may also wish to indulge themselves with restaurants, lounge access, or cabin upgrades, which could be paid for example using FFP points.

Considering the personas mentioned before, the family of four returning from their vacation in New York City may have excess baggage, while the business purpose customer returning from New York City on a red-eye flight may be looking for an upgrade to the business cabin. These examples serve to illustrate that customers' needs may vary significantly and that the airline has an opportunity to approach the customers with relevant offers based on a deep understanding of their needs, preferences and intent.

3.4 Matching Airline Industry Use-Cases With Appropriate Recommendation Algorithms

In this section, we revisit the use-cases introduced in the previous section and we discuss how they can be implemented in practice using the families of recommender system algorithms described in section 2.1. We identify the most appropriate algorithms given the non-functional requirements, such as (i) the available input data, (ii) the output data, (iii) the chosen objectives, and (iv) the operational constraints (e.g. response times). For each use-case, we also provide relevant metrics that could be used to assess the quality of each recommender system. figure 3.7 provides a summary of this analysis.

		Next Travel Destination	FFP Personalization	Search Filtering & Ranking	Upsell, Cross sell & Third Party content	Advertised Services	Airport/Flight Experience
Input Data	Past user-item interactions	-	✓	-	✓	✓	✓
	User information	-	✓	-	✓	✓	✓
	Item Information	✓	✓	✓	✓	✓	✓
	Context Information	✓	✓	✓	✓	✓	✓
	Knowledge Graph	-	✓	-	✓	✓	✓
	Live interactions	✓	-	✓	✓	-	-
	Extra information	**RMS, Interests, Budget, Upcoming events**	**FFP, RMS**	**RMS**	**Third Party**	-	**FFP, RMS**
Output Data	Offer build	destination, date range	action to burn points	ranking of offers	ranking of offers	ancillary proposition	ranking of offers
	Offer retail	presentation, infographics, description	offers timing	offers highlight	offers highlight	offer timing	presentation, infographics, description
Objectives	Travelers' Loyalty	-	✓	-	✓	✓	✓
	Air product conversion	✓	✓	✓	-	-	-
	Ancillary product Conversion	-	-	-	✓	✓	✓
	Third Party Conversion	-	-	-	✓	-	-
	Miles burned	-	✓	-	-	-	-
Specifics	Response Time	-	-	✓	✓	-	✓
	Data Acquisition	-	-	✓	✓	-	✓
Algorithms Family		**CB, CA**	**CA, KG**	**CB, CA, SB**	**SB, CA, KG**	**KG**	**CA, KG**

Figure 3.7 – Summary of recommender system algorithms for each use-case given the input data, outputs, objectives and constraints. Algorithms in brackets are feasible, while the algorithms without bracket are preferred

Next Travel Destination

We take the assumption that the customer (user) is anonymous at this stage of the traveler journey. Hence, for this use-case, we cannot rely on the past interactions of the user and we discard the use of sophisticated algorithms such as KGRS that are most effective with this information. Instead, we consider using CA algorithms in a post-filtering fashion starting with CB or SB algorithms to rank destinations based on either the content of the destinations (CB) or the user's clicks through his live interactions (SB). The outputs of the CB/SB algorithms can then be filtered according to the criteria specified by the user from the search tool. Metrics used to evaluate the recommendations could be Click-Through Rate and Conversion Rate.

FFP Personalization

In this use-case, the customer identity is known and we can therefore leverage on individual FFP data - such as tier level, point balance, point expiration dates, recency, frequency, and monetary value - but also on price/point conversion rates for the recommended itineraries and services in order to produce meaningful recommendations. The algorithm must also be able to mix this information with a variety of other data from different sources, ranging from the product catalog of air and non-air products, the customer travel history, and the product availability and prices provided by the RMS.

Hence, because of their data integration capabilities, KGRS algorithms appear to be the natural choice for this complex use-case. Moreover, as demonstrated in [164], KGRS can be extended to include contextual information allowing the algorithm to capture the travel intent of the user. Metrics used to evaluate the recommendations could be conversion rate and FFP points burned.

Search Filtering & Ranking

We take the assumption that the customer (user) is anonymous during this stage. In this situation, the recommender system will have to rely on stated criteria (origin-destination, date range, stops, etc.), the context of the search (search time and date, type of the device being used, etc.), product attributes (cabin, flexibility, baggage allowance, etc.), and possible extended criteria depending on the capabilities of the search tool. The recommender system may also employ user navigation behavior to better understand the travel intent. Given the input data available, CA/SB recommender systems [120, 128] seem to be judicious choices provided that session data can be acquired and response time kept within acceptable limits. Metrics used to evaluate the recommendations could be Click-Through Rate, Conversion Rate, and sales.

Upsell, cross sell and Third-Party Content

At this stage, the customer identity is known. However, the customer travel history will, in many cases, still be absent or rather limited. In this case, SB/CA algorithms could be considered. On the other hand, when customer travel history is present, hybrid approaches integrating personalized recommendations could be investigated using for example the KGRS algorithms. Response time and data acquisition are important specifics of this use-case and must be taken into consideration before the preferred algorithm is chosen. Of note, the SB algorithms have a very fast execution time compared to CA and KGRS, which may impact the choice. Metrics used to evaluate the recommendations could be conversion rate, ancillary/third party revenue and adoption rates.

Advertised services

Targeting customers with unsolicited notifications can be counter-productive and lead to adversarial effects on customer loyalty if done incorrectly. It is therefore critical to identify the customers that we expect to react positively to an advertised service. This problem can be seen as an inverse recommendation scenario – recommending a user to an item.

This problem is well-suited for KGRS algorithms. Indeed, in this use-case where the customer identity is known, the algorithm can take advantage of a diverse set of data: collaborative information (e.g. historical ancillary purchases), user-related information (e.g. number in party), item-related information (e.g. product descriptions), and context-related information (e.g. attributes of the current order). Additionally, other ML approaches such as contextual multi-armed bandits [84] could also be employed to find the best timing and channel for sending the notifications. Metrics used to evaluate the recommendations could be Click-Through Rate, Conversion Rate, and incremental revenue.

Airport/Flight Experience

The time period spent at the airport or during the flight itself is a particularly favorable window of opportunity for the airlines to approach the traveler with personalized and contextualized offers. The algorithms of choice could be CF or CB given their ability to learn the preferences of the travelers and provide near real-time recommendations, especially when the product catalog is rather limited. Alternatively, the CA algorithm should be also considered, since this algorithm is able to capture travel intent which may well be of importance in this use-case. The conversion rate, incremental revenue, FFP points burned are the most appropriate metrics to evaluate how these algorithms perform.

3.5 Summary

Recommender systems have already been introduced in several industries such as retailing and entertainment, where their capability to display personalized and contextualized recommendations have provided benefits to customers and sellers alike. However, their application in the airline industry remains in its infancy. In this chapter, we explain that this is primarily a result of the limitations of IT systems that delegate airline control of offer creation to content aggregators. The traditional distribution paradigm relies on a two-step process - fare filing which drives the product and price construction, followed by the availability computation - which provides airlines with limited control over offer construction and retailing. Further, the airlines are unaware of the customer's identity and therefore unable to generate personalized recommendations.

NDC is an enabler for the airlines to provide contextualized and personalized offers, thereby opening the door for the application of recommender systems via the airlines Offer Management Systems (OMS). We believe that recommender systems hold the key to customer centricity with their ability to understand and respond to the needs of the customers throughout all touchpoints during the traveler journey, which we have exemplified with airline-specific recommender system use-cases.

We have explained how recent advances in ML have enabled the development of a new generation of recommender systems to provide more accurate, contextualized and personalized offers to users. However, choosing one family of algorithms over another can be a complex task for a travel industry expert because of the large number of algorithms described in the literature and the particularities of the travel domain. Therefore, we have for each of the use-cases, provided guidance by identifying the appropriate algorithms.

While we have discussed how the application of recommender systems can provide "short-term" (or transactional) benefit to the airline through increased ancillary adoption rates and revenue, we believe that recommender systems may have an even greater opportunity for improving customer experience and increasing customer loyalty by enabling airlines to understand their customers' needs, preferences and intent. The impacts of effective recommendations and retailing on customer loyalty in the airline industry have yet to be explored.

The next step in the thesis consists in developing recommender systems to address some of the airline specific recommendation use-cases described in section 3.3, then performing an empirical study of the different recommender system algorithms described in the previous chapter (chapter 2). The empirical work will serve not only to develop recommender systems that helps addressing the research questions mentioned in section 1.5 but also help us to assess the performance of the algorithms using actual airline data. This requires to partner with airlines in order to acquire real life data. This empirical work will be the main content of the next chapters of the thesis.

Chapter 4

Developing Recommender Systems across the Traveler Journey

In this chapter, we tackle three airline specific recommendation use-cases; for each of the use-cases we develop an appropriate recommender system algorithm based on user profile inferred from their booking history and content information about items which help gain insights on the collection of airline products. We conducted extensive experiments to compare the developed recommender system algorithms with a set of baseline algorithms and, for each of the use-cases, we address the research sub-questions that derives from ***RQ1***.
Each section in this chapter is dedicated to a use-case. We structure the sections as follows: we first formulate the problem we want to solve after introducing the use-case and presenting some related works, then we present the collected dataset that will be used to address the use-case, followed by a description of the algorithm developed to address the problem. Finally, we present the experiments performed to demonstrate the effectiveness of the model and lastly we give some conclusions and outline some limitations that are being addressed in the following chapter (see chapter 5.

4.1 Next Trip Recommendation

In this section we focus on the use-case of Next Trip recommendation (see section 3.3) where the objective is to recommend next travel destinations to past travelers.

Inspiring users who became exposed to many inspirational tourism posts and advertisements in social media, travel forums, travel agencies and airline websites is not an easy task. Indeed, although inspirational, many of these posts might not fit a particular user's profile and, thus, they may not be relevant to him/her.
In the recent years, destination recommender systems (DRSs) have been proposed to suggest a ranked list of destinations, sometimes composed of sights, events and destinations to visit, based on information provided by the user [74, 77, 162].
In this work, we tackle the use-case of 'Next Trip Recommendation', where the goal is to

recommend relevant travel destination to travelers. The use-case addressed in this work is slightly different from the one presented in section 3.3. Also during the inspiration phase, we focus on a passive inspiration scenario, in which travel destinations are sent to travelers through airline email marketing campaigns, with the goal of making the search process easier for travelers.

In addition to travelers' history, recommender systems can also consider contextual information, for example, by leveraging Location-based social networks (LBSNs) or Event-based social networks (EBSNs) data [96]. LBSNs allow users to publicly or privately share their position by performing a check-in when visiting a certain venue or a POI. Leveraging these information enable to first know what a destination is best characterized by (restaurants, sport events, museums, parks, etc.), and then to identify the user's interests [103].

In the recent years, The use of knowledge graph embeddings [115, 134, 168] and neural networks [21, 53, 55] for item recommendation has proven to be efficient by improving the recommendation performance. To tackle the problem of travel destination recommendation, we propose Deep Knowledge Factorization Machines (DKFM) a neural network-based algorithm for travel destination recommendation that combine two existing deep learning-based recommender systems [21, 53]. Our model leverages content, collaborative and contextual information related to travelers' bookings.
Travel destination content is enriched through the use of textual embeddings representing destinations based on their Wikipedia content description and the use of KGE coming from STD knowledge graph [103].
Our approach relies on learning i) a representation of destinations using different data sources including Wikipedia and STD, ii) the long-term user's behavior using his/her booking history and iii) a representation of the context associated with each past trips.

4.1.1 Related Work on DLRS

In section 2.1, we provide a literature review of recommender system algorithms specifying that many deep learning-based recommender systems emerged. In this section we give more details on DLRSs since the model proposed DKFM is based on deep learning. Recently deep learning algorithms have demonstrated their effectiveness when applied to information retrieval and recommender system [169].
In [36], the authors used a Multi-layer perceptron (MLP) that takes as input the (user, item) interactions and learn an arbitrary function that replaces the inner product of MF at the same time as the latent feature vectors (user and item embeddings).
In [55], the authors combined a MLP with a generalized MF in the form of a neural network represented by a single layer perceptron. The algorithm proposed by the authors is considered as a state-of-the-art CF recommender system.

In [21], the authors proposed *Wide and Deep learning* model for Mobile application recommendation[1]. The wide learning component is a linear model represented by a one-layer perceptron which enables to capture *memorization*, while the deep learning component is a non-linear model represented by a multi-layers perceptron which enables to capture *generalization*.
In [49], the authors proposed *DeepFM*, a model that combines factorization machines (FM) and a MLP. The idea is to model the high-order feature interactions via a multi-layer perceptron and low-order interactions with FM [123].
In [53], the authors proposed *NFM* which is similar to DeepFM, but they use a pooling layer that computes the first order feature interaction term in FM formula, instead of using the whole FM model.
Other neural network architecture has been used for recommendation, such as Recurrent Neural Networks (RNNs) [59] (e.g. session-based recommendation) or Convolutional Neural Networks (CNNs) used for example to capture images representation in order to enrich item representations [83].
Inspired by DeepFM [49] and NFM [53], We propose DKFM, a feed-forward neural network that combines two existing deep learning based recommender systems [21, 55].

4.1.2 Problem Formulation & Preliminaries

Problem Formulation:

To address the research question *RQ1.1*, we formulate the following problem: Given a traveler, his demographics (age, nationality, etc.), his historical bookings and the contextual data related to those bookings (day of week, number of passengers, stay duration, etc.), we aim to recommend to this traveler a ranked list of destinations he/she would like to go to. In this work, a destination is represented by an airport[2]. Figure 4.2 illustrates the recommendation task we want to tackle. This work is tightly coupled with a real world application where the aim is to suggest a ranked list of destinations where travelers would like to go to, as shown in Figure 4.1.

Preliminaries:

In recommender system realm, there are two different types of feedback: explicit feedback where the user gives a rating on how he/she liked an item, and the implicit feedback where we know only the interest of a user for an item. Concretely, in our case, the implicit feedback denotes the fact that a traveler t visited a destination d.

[1] Google Play Store: https://play.google.com/store
[2] Airport IATA Code: https://www.iata.org/en/publications/directories/code-search/

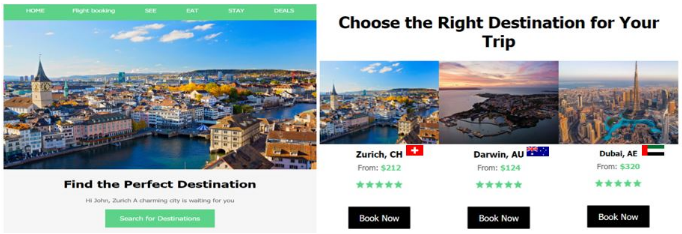

Figure 4.1 – Top-3 travel destinations recommendation included in a marketing email.

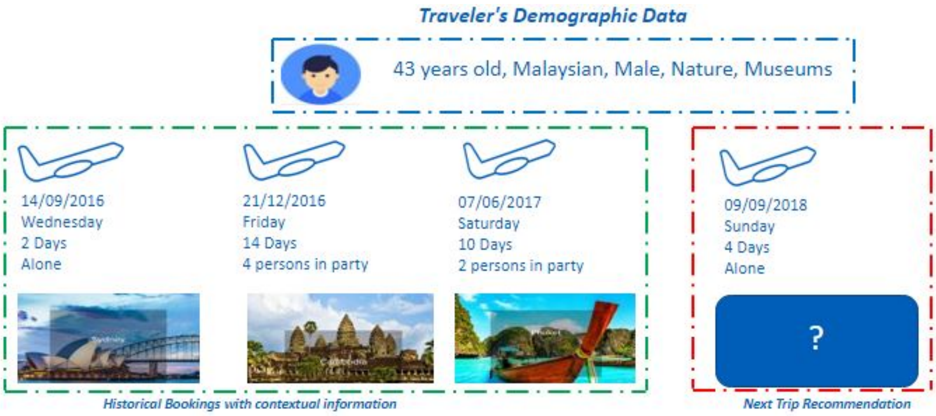

Figure 4.2 – The recommendation task is to predict the next trip destination a traveler would go to, given his/her historical bookings.

Definition 4.1.1 *Given a matrix* $\boldsymbol{M} \in \mathbb{R}^{n\times m}$*, where* m_{td} *is the number of times traveler* t *travelled to destination* d*,* n *the number of travelers and* m *the number of different destinations. We define the traveler binary feedback matrix* $\boldsymbol{R} \in \mathbb{R}^{n\times m}$ *as follows:*

$$r_{td} = \begin{cases} 1 & \textit{if } m_{td} > 0 \\ 0 & \textit{otherwise}\,. \end{cases} \tag{4.1}$$

Definition 4.1.2 *The sparsity of the traveler binary feedback matrix* $\boldsymbol{R} \in \mathbb{R}^{n\times m}$ *is defined as follows:*

$$\rho(\boldsymbol{R}) = 1 - \frac{\#interactions}{m \times n} \tag{4.2}$$

where, n *is the number of travelers and* m *is the number of different destinations*

4.1.3 Data

In this section, we present the source of the data used in this work, then we describe how the dataset is pre-processed and filtered to be used as input of our model DKFM and the baseline models used to compare our model with.
We work on a real-world production dataset of bookings from T-DNA database[3]. Each booking contains one or several air ticket purchases, and is stored using Passenger Name Record (PNR) information. A PNR is created at reservation time by the airline reservation system and contains information about the purchased air ticket (e.g. travel itinerary, payment information), traveler demographics and additional services (e.g. preferred seat, extra bag) if purchased.
The original dataset considered for this work contains ~ 4.1 Million bookings for 405302 unique travelers[4].

Customer Segmentation Model

According to [33], the approach to recommend destinations to Business/Leisure travelers is expected to be different. In this work, we focus only on recommending travels for leisure purpose. Hence, we keep only leisure bookings based on a market segmentation (travel purpose).
More formally, given a set of historical bookings labeled by their trip purpose [104] (business or leisure), we build a binary classifier based on Random Forest algorithm [13] in order to segment our bookings into business or leisure bookings. Table 4.1 shows the features used as input of the classification.

Table 4.1 – Features used for Business/Leisure Classification

Feature Name	Type	Range
Number Passenger	Numerical	{1..9}
Stay Duration	Numerical	[0,99]
Saturday Stay	Binary	{0,1}
Purchase Anticipation	Numerical	{0..364}
Age	Numerical	{0..99}
Gender	Categorical	{Female, Male, Unknown}

This dataset contains 122242 bookings (60% leisure). Random Forest hyper-parameters are tuned using grid-search algorithm over the following: maximum Tree depth $\in [5,8,10]$, maximum sample features per tree $\in [0.6, 0.65, 0.7, 0.75]$, minimum samples per leaf $\in [1,2]$, number of Trees $\in [10,20,50,100,150,200]$. Finally, to evaluate our classifier, we use a Cross Fold Validation (k=10) by splitting our data into training and validation set (90% training, 10% validation

[3] T-DNA: Traveler DNA is a database which contains bookings of travelers over a dozen of airlines. The dataset used in the experiments is GDPR compliant and do not include any personal identifiable information.
[4] Statistics of the pre-processed dataset are given in table 4.3 and table 4.4

set) and compute the accuracy, precision and recall metrics for the best performing classifier (optimal hyper-parameters). We report the results in table 4.2.

Table 4.2 – Business/Leisure Classification Performance.

Metric	Score
Accuracy	0.87
Precision	0.87
Recall	0.91

We also compute the importance of each feature of the classification task based on the relative information gain of each feature (see figure 4.3).

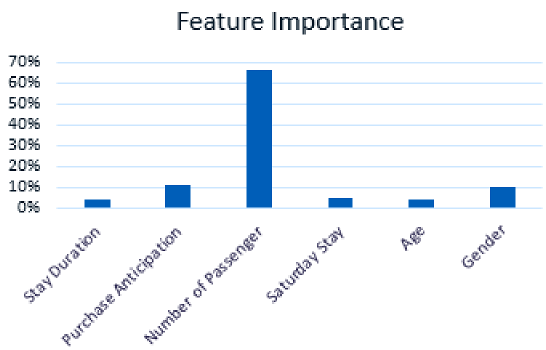

Figure 4.3 – Histogram representing the importance of each feature for Business/Leisure Classification.

The classifier was then used to classify bookings of the considered dataset into Business/Leisure bookings. We keep only leisure bookings: we obtain ~ 2 Million bookings for ~ 629156 unique travelers, which represent 48% of the original dataset.

Data Filtering for Recommendation

Despite the huge amount of available bookings that could be used as input of DKFM and the other baseline models, the feedback matrix R is highly sparse. Indeed, based on eq 4.2 the sparsity is equal to 99.6%. Moreover, more than 65% of the travelers have traveled only two times.

Similarly to [55,64], in order to cope with the limitation of data sparsity and without any further mention, we keep only travelers that has at least **5** different destinations in their history, and destinations that were visited at least **20** times. Applying these filters decrease considerably the size of bookings that are considered in this study and raise some questionings on how we can deal with the problem of data sparsity (see Chapter 5). Tables 4.3 and 4.4 show statistics of the pre-processed dataset.

Table 4.3 – Statistics of the experimental dataset

#Feedbacks	#Interactions	#destinations	#Travelers	Sparsity
304019	152547	119	26019	95%

Table 4.4 – Statistics of the experimental dataset

Variable	Min	Max	Std	Mean	Median
#Visiting same destination	1	354	3.34	2	1
#Different travelers per destination	20	19496	2452	1282	293
#Different destinations per traveler	5	37	1.49	5.86	5

4.1.4 DKFM: Deep Knowledge Factorization Machines

DKFM is deep neural network model that leverages information coming from different sources by enriching implicit interactions between travelers and destinations (Collaborative information) with external knowledge. We present the model architecture in figure 4.4.

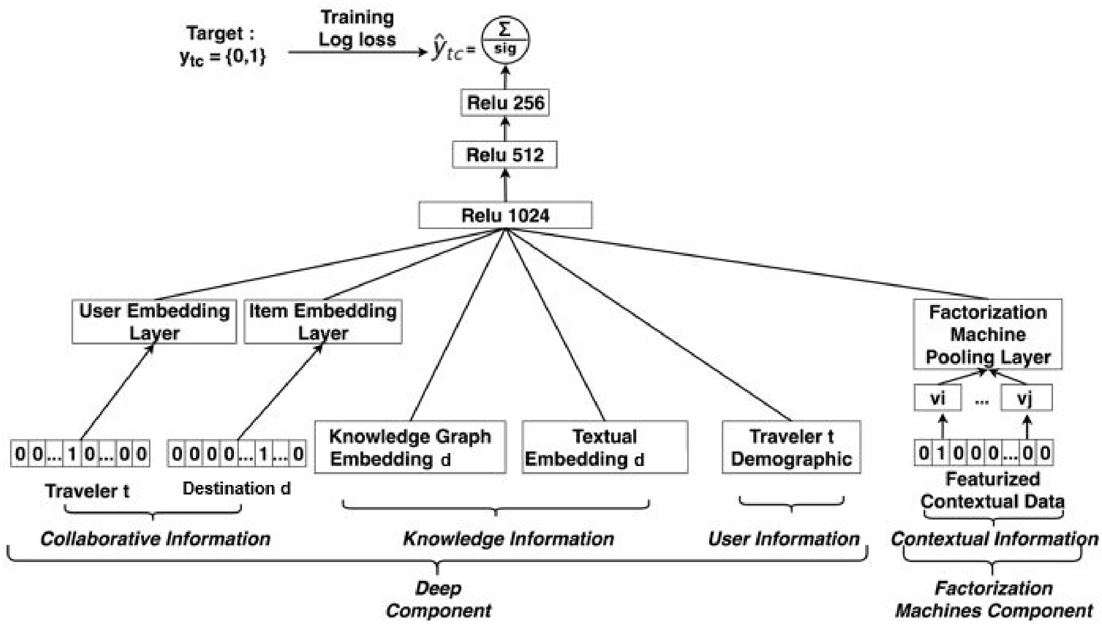

Figure 4.4 – Deep Knowledge Factorization Machines architecture.

In addition to knowledge information, we incorporate contextual travel information related to a travelers' bookings which can be an important factor to consider when doing recommenda-

tion in the airline industry [26].
DKFM is a combination of a deep component which is a MLP that takes as input the implicit travel-destination interactions and content information, with a FM component that takes as input contextual information.
The deep and factorization machines components are combined by concatenating:

- Traveler and destination embeddings;
- Textual and knowledge graph destination embeddings;
- User demographics;
- Contextual feature vectors computed by a pooling operation.

The concatenated vectors are fed into a MLP. The two components are jointly trained using back-propagation algorithm to learn the weights of the deep and factorization machines components, and also travelers' and destinations' embeddings.

In the remaining of this section, we first present how destinations are enriched with external knowledge resources, and describe in details the different components of our model. Finally, we present how we combine the two existing deep learning based recommender systems to build our model.

Textual Embedding

We use the Wikipedia Python API[5] to retrieve all the Wikipedia pages of the 119 destinations contained in our dataset. Once all documents describing travel destinations retrieved, we define a method to construct an embedding for each wikipedia page.
In the recent years, many competing algorithms designed to learn sentence or document representations have emerged. In [76], the authors propose to learn unsupervised sentence embeddings based on a recurrent neural network encoder-decoder trained to reconstruct the surrounding sentences from the current sentence, similarly to what is done in skip-gram model for word embeddings. In [88], the authors proposed a faster way to learn unsupervised sentence representations by reformulating the problem as a classification task, where the classifier has to choose the right next sentence among a set of possible words.
While these approaches have shown good performance, simple baseline models like averaging pre-trained word embeddings give also strong results [100]. We propose to encode a sentence as a weighted sum of word vectors, where the weight of each word vector corresponds to the Term frequency-inverse document frequency (TF-IDF) of the word based on all travel

[5] Python Wikipedia Api: https://pypi.org/project/wikipedia/

destinations Wikipedia pages. We use the fastText pre-trained word vectors [100]. FastText word vectors are learned using Wikipedia 2017, UMBC webbase corpus[6] and statmt news[7] datasets.

Knowledge Graph Embedding

In table 4.5, we present the original features included in the STD dataset used to build STD knowledge graph [103] (see section 2.6). We use STD knowledge graph to compute travel destination KGE.

Table 4.5 – Feature description of STD

Feature Name	Type	Range or #different values
Trail ID	Numerical	{1..Number of Trails}
User ID	Numerical	{1..Number of Users}
Venue ID	Categorical	~ 4.4 Million
Venue Category	Categorical	934
Venue Schema	Categorical	162
Venue destination	Categorical	43833
Venue Country	Categorical	207
Time Stamp	Date	2012-04-03 To 2018-10-19

In [115], the authors present an empirical comparison of translational distance models for items recommendation, the results have shown that TransE [12], the model with the least parameters in comparison with other translational distance models [148] obtains the best scores over a set of metrics. We propose to use TransE to learn embeddings for the entities and relations in the STD knowledge graph, and extract the travel destinations KGE. Finally, we use Wikidata ids[8] in order to match travel destinations of our dataset with destinations KGEs obtained using STD knowledge graph.

Deep Component

As shown in figure 4.4, the deep component of DKFM is a feed-forward neural network, that takes as input the one-hot encoded vector of the traveler and the travel destination (t, d) and transform these two vectors into low-dimensional and dense vectors through an embedding layer which is a single-layer perceptron, whose weights are initialized randomly. Weights are updated during the back-propagation phase. In addition to the collaborative information, user demographics in addition to travel destination KGE and textual embeddings are fed into

[6]https://ebiquity.umbc.edu/resource/html/id/351
[7]http://statmt.org/
[8]Wikidata: https://www.wikidata.org/wiki/Wikidata:Main_Page

the deep component of DKFM.

Factorization Machines Component

In [53], the authors modeled the first order feature interaction of factorization machines term by using two neural network layers. The first layer takes as input the vector $\mathbf{x}$ corresponding to contextual information, and create an embedding vector of each feature of $\mathbf{x}$. More formally, the first layer computes a vector $\mathbf{v}_i \in \mathbb{R}^k$ for each feature i, where k is the dimension of features vector. In the second layer, the following pooling operation is performed $\mathbf{x}$:

$$f(x) = \sum_{i=1}^{k} \sum_{j=i+1}^{k} x_i \mathbf{v}_i \odot x_j \mathbf{v}_j \tag{4.3}$$

where, $\odot$ is the element-wise product.

As shown in Figure 4.4, we use the same two layers in order to compute the factorization machines feature vectors interaction term.

Deep Knowledge Factorization machines

The obtained vectors from the deep component and the factorization machines component are concatenated and fed into a MLP that contains different hidden layers. In each hidden layer l, we perform the following computation:

$$a[l] = Relu(\mathbf{W}[l-1]^T \mathbf{a}[l-1] + b[l-1]) \tag{4.4}$$

where, $ReLu(x) = max(0, x)$ is the rectified linear unit function. It is used as the activation function for each layer of the MLP. While, there are other functions that can be used as activation function: sigmoid or hyperbolic tangent, ReLu function is proven to avoid vanishing gradient problem, and has shown better results in the experiments. $a[l-1], W[l-1], b[l-1]$ are respectively the activation functions, weights and bias of previous layer (layer l-1).

Finally, at the end of the last hidden layer L, we compute the prediction $\hat{y}_{td}$ by applying a sigmoid function to restrict the value between 0 and 1 which represents the probability to recommend the destination d to the traveler t:

$$P(t,d|\mathbf{X}) = \hat{y}_{td} = \sigma(\mathbf{h}^T\mathbf{a}[L]) \tag{4.5}$$

where, $\sigma(x) = \frac{1}{1+e^{-x}}$, h is the weight vector of the last neuron, $\mathbf{X}$ is the input vector of the MLP.

The objective function of the back-propagation algorithm is to minimize the logistic loss defined as the negative log-likelihood of the observation given the model's predictions (binary classification):

$$Loss(\hat{y}_{td}, y_{td}) = -log(P(\hat{y}_{td}|\mathbf{W}, b, \mathbf{h})) \tag{4.6}$$

$$= -\sum_{u=1}^{n}\sum_{i=1}^{m} y_{td} \times (log(\hat{y}_{td}) + (1-y_{td}) \times log(1-\hat{y}_{td})) \tag{4.7}$$

where, $P(\hat{y}_{td}|\mathbf{W}, b, \mathbf{h})$ is the likelihood function of $\hat{y}_{td}$, n is the number of users, and m the number of items.

4.1.5 Experimental Setup

In this section, we present the settings of the experiments and the baseline models implemented to compare our model with.

Dataset: We experiment our model with the dataset obtained in section 4.1.3. The characteristics of the dataset are shown in Tables 4.3 and 4.4.

Training & Test Sets: The recommendation task consists in predicting the next travel destination of a given traveler based on his/her previous trips, hence the dataset must be split in such a way that the test set must contain the last trip for each traveler (leave-last-out).
To do so, we adopt the leave-one-out strategy as used in [55]. Formally, for each traveler, we put his/her last trip in the test set and keep the remaining trips for the training set. N_s random destinations where the traveler never went to are considered as negative samples. The experiments showed that N_s was working well for a value of 3.

Evaluation Metrics: The output of the recommender system is a ranked list of 10 destinations, where at best, one element of the 10 recommended destinations is a relevant one and corresponds to the 'next' travel destination of the traveler. Given that, we think it is judicious to use Hit Rate metric to measure whether or not the relevant destination is in the top-10 list and use Mean Reciprocal Rank metric to capture how well the hit is ranked in the list. Even if these two metrics have been defined more generally in section 2.1.6, we decide to redefine them in the context of this use-case:

- **HR@K**:

$$HR@K = \frac{1}{n}\sum_{t=1}^{n}\sum_{j=1}^{K} hit(t, d_j) \tag{4.8}$$

- **MRR@K**:

$$MRR@K = \frac{1}{n}\sum_{t=1}^{n}\sum_{j=1}^{K} \frac{1}{rank(rel_t)} \tag{4.9}$$

where n represents the number of travelers, K the length of the ranked list and $hit(t, d_j)$ is equal to 1 if the traveler t traveled to the destination d_j. In equation 4.9, $rank(rel_t)$ is the rank of the relevant destination where the traveler t has traveled to. The rank is only considered if the relevant destination is in the top-K list.

Baseline Models: We compare our model DKFM with a set of baseline models that includes collaborative filtering algorithms, factorization machines algorithm and also two state-of-the-art DLRSs [169]. All baseline models used in this work are summarized below (we use the terms 'user' and 'item' to refer to travelers and destinations):

- **MostPop:** Items are ranked by their popularity. the popularity of an item is measured by the number of interactions of this item. The Top-k items are recommended to users in this case.
- **ItemKNN [128]:** This is a neighborhood based collaborative filtering algorithm based on items similarities. The idea is to compute an item-item similarity matrix based Pearson correlation coefficient, and then recommend to each user similar items to the ones available in the history of the user.
- **ImplicitMF [64]:** This method is proposed to deal with implicit feedback data when using MF algorithm. The authors propose to add a weight term to consider the confidence of an item and proposed an alternating least square algorithm to learn user's and item's latent vectors.
- **BPRMF [122]:** *BPRMF* is also a MF method tailored for implicit feedback where the authors propose to minimize a pairwise ranking loss rather than minimizing a mean squared error between the predicted and the observed 'rating' as usually done in Matrix Factorization algorithm.

- **NCF [55]:** *Neural Collaborative Filtering* is a state-of-the-art CF method. It combines the (user, item) interaction as input of a multi-layer perceptron and a single layer perceptron that models the matrix factorization method.
- **FM [123]:** *Factorization Machines* model was proposed to incorporate contextual information in the recommender system. The author propose a method that computes not only users' and items' latent vectors but also contextual features latent vectors.
- **WDL [21]:** *Wide & Deep Learning* model is a hybrid recommender system. It is a deep learning based recommender system that combines a deep component (feed forward neural network) plus a wide component that can be seen as a linear model that computes cross products between input features.
- **NFM [53]:** *Neural Factorization machines* is a state-of-the-art model for context-aware recommendation. The factorization component used in our model represents a part of the neural factorization machines, the other part is a MLP to which we add the linear term of factorization machines formula.

Implementation Framework & Parameter Settings: Our model plus all the baselines were implemented using Python and Tensorflow library[9]. The hyper-parameters of all the models are tuned using grid-search algorithm. First, we initialized all the weights randomly with a Gaussian Distribution ($\mu = 0$, $\sigma = 0.01$), and we use mini-batch Adam optimizer [75]. It is worth mentioning that other optimizers could be used in order to minimize the loss function defined in (6), however, Adam Optimizer has shown to be the most efficient in time and also accuracy. We evaluate our model using different values for hyper-parameters the size of traveler and destination embedding layers (els) $\in \{32, 64, 128\}$, the features vector size of the factorization machines component $\in \{16, 32, 64\}$, the batch size $\in \{64, 128, 256, 512, 1024\}$, the number of epochs $\in \{5, 10, 15, 20\}$ and the learning rate (lr) $\in \{0.001, 0.005, 0.006, 0.008, 0.1\}$.

4.1.6 Results

In this section, we report the results obtained from the experiments.

Deep component performance We present in figure 4.6 the recommendation performance of DKFM using different input information as input of the deep component with respect to the metrics we defined previously. We can notice that the traveler demographics information remarkably improved the performance of deep component that has only the traveler and destination embedding as input. The scores of HR@10 and MRR@10 increased by 15%. As for the destination embeddings, we can notice that using the textual embeddings improved the

[9] Python Tensorflow Api: https://www.tensorflow.org

results by 9% and 6%. When considering the traveler demographics in addition to the textual destination embedding the results improved by 27% for HR@10 and 28% for MRR@10. Finally, when concatenating all the input information, we improve HR@10 and MRR@10 by respectively 30% and 28%. In this experiment, we use a batch size of 256, the number of epochs used is 8. Even if the loss defined in equation 4.6 decreases after 5 epochs for both the training and validation set, DKFM starts over-fitting after 8 epochs and the metrics HR@10 and MRR@10 starts decreasing. Finally we use 0.006 as learning rate and 128 as the size of the traveler and destination embedding layers.

Demographics Data	Textual Embedding	Knowledge Graph Embedding	HR@ 10	MRR@ 10
✖	✖	✖	0.72	0.34
✖	✔	✖	0.79	0.37
✖	✖	✔	0.80	0.38
✔	✖	✖	0.82	0.38
✔	✔	✖	0.84	0.41
✔	✖	✔	0.85	0.42
✔	✔	✔	**0.88**	**0.44**

Figure 4.5 – Contribution of each input of the deep component with respect HR@10 and MRR@10.

Finally, in figure 4.6 we measure the effect of the embedding size on the performance of DKFM.

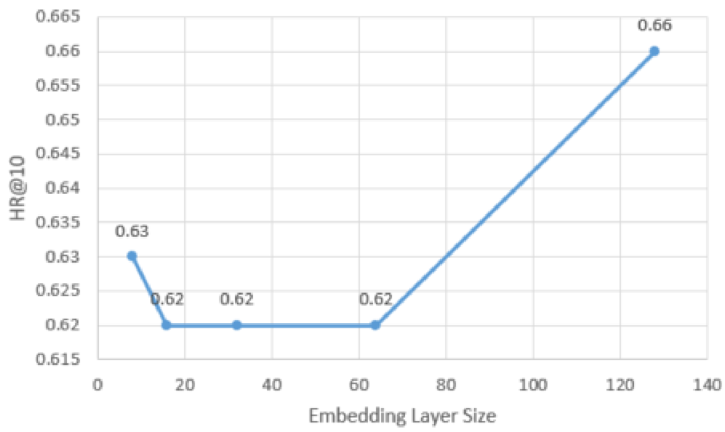

Figure 4.6 – HR@10 for different embedding layer size 8, 16, 32, 64, 128

Factorization machines performance The factorization component is fed by *the number of passengers* in the booking in addition to *the departure day of week* and *stay duration* as contextual data. Similarly to figure 4.5, we report the contribution of each input in the table 7.

Table 4.6 – Contribution of each input of DKFM with respect to the recommendation performance.

Model	HR@10	MRR@10	#Layers	1st Layer size
DKFM_CTXT	0.72	0.34	2	256
DKFMTE	0.79	0.37	2	512
DKFMKGE	0.80	0.38	2	512
DKFMU	0.82	0.38	2	512
DKFMUTE	0.84	0.41	2	1024
DKFMUKGE	0.85	0.42	2	1024
DKFM	**0.88**	**0.44**	3	1024

It is worth to notice that adding the contextual data increase the score of HR@10 and MRR@10 (DKFM_CTXT).

HR@10 and MRR@10 increased by 10% and 9% respectively when adding the textual embeddings (DKFMTE). As for, the knowledge graph embeddings both scores increased by 11% (DKFMKGE). When considering the user demographics data (DKFMU), we notice that the results improved by 13% and 11%. Finally, when considering all input information of DKFM model, the results were 22% for HR@10 and 30% for MRR@10 better than the DKFM with only contextual data. We tested different values of k from 8 to 128, and we did not notice any change neither on the test loss, nor on the metrics HR@10 and MRR@10.

DKFM against Baseline Models

We computed HR@10 and MRR@10, for the different baseline algorithms implemented and for our model DKFM, and we presented the results in figure 4.7.

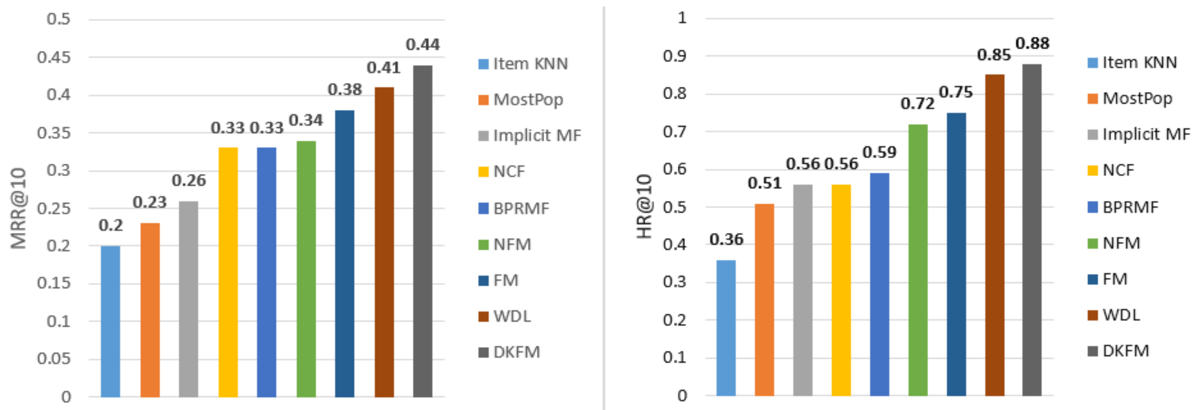

Figure 4.7 – Recommendation Performance of DKFM and baseline models with respect to HR@10 and MRR@10

As shown in figure 4.7, our model outperforms the collaborative filtering methods demonstrating the importance of adding the destination embeddings, traveler demographics data and the contextual data. It also shows a slight improvement over Wide and Deep Learning and Factorization machines where one is using destination embeddings and traveler demographics data and the other is using contextual data. Considering that training time is also an important aspect to consider when doing recommendation, we measured training times for both DKFM and WDL models. For each epoch, the training time is equal to 24 seconds for DKFM and 15 seconds for WDL. For our experiments, we used an NVIDIA Tesla K40C GPU with 12 GB of memory.
Finally, we compute the two evaluation metrics HR@K and MRR@K for different values of K and we report the results in figure 4.8

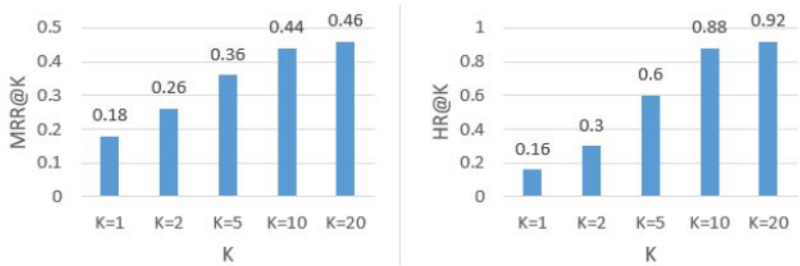

Figure 4.8 – Recommendation of DKFM model with respect to HR@K & MRR@K for different values of K.

4.1.7 Summary

In this work, we have developed DKFM a neural network-based model to recommend personalized travel destinations to past travelers. We have leveraged two external data sources in order to enrich the characteristics of the recommended travel destinations. We conducted several experiments to answer the following questions that address some aspects of the model:

1. What is the contribution of the deep component with respect to the recommendation performance of DKFM?
   The results of the experiments presented in figure 4.5 show that when considering travelers demographics and destinations embeddings (textual or graph) enhance significantly the recommendation performance of the deep component which consequently improve the recommendation performance of DKFM.

2. What is the contribution for each input used in the deep component (e.g. traveler demographics data, destination embeddings)?

Table 4.7 – Optimal Hyper-parameters for DKFM.

Hyper-parameter	els	lr	k	Ns	Batch size	Epoch
Value	128	0.06	8	3	256	8

The experiments demonstrate that using travelers' demographics improve remarkably the performance of the deep component. The use of textual embedding also improves the performance, but less than travelers' demographics. Finally, it shows that destination KG embeddings improve more the results of the two metrics in comparison with the two other inputs (demographics and textual embeddings).

3. How our model perform in comparison with the baseline algorithms?
   Our model outperforms all CF algorithms in addition to DLRSs [21,53]. One can notice that the baseline MostPop has a relatively good score for HR@10. This can be explained by the very few number of travel destinations (119) which is a few number of items, hence recommending the 10 most popular destinations is performing well: at least one time out of two, MostPop is recommending relevant travel destinations.

4. How the performance of DKFM is affected by the hyper-parameters?
   We ran grid-search on all the DKFM's hyper-parameters. The optimal hyper-parameters for our dataset are presented in table 4.7. We also compared range of values for the size of the embedding layers, where 128 showed to be the value that has the highest HR@10, and we also compared different values for k: the size of feature vector from factorization machines component.

By developing and evaluating DKFM, we addressed the first research sub-question of the thesis: *RQ1.1. What travel destination should be recommended to each traveler?* (Section 1.5). Nevertheless, as presented in section 4.1.3, the data sparsity is a strong limitation in this work as we worked only with travelers that have more than 5 historical trips which represents less than 20% of the airline traffic of our partner airline. In the next chapter (Chapter 5), we explore novel KG recommender systems for travel destination recommendation. This family of algorithms that can incorporate different types of information into one single structure (knowledge graph) has proven to be effective to deal with data sparsity and also cold start problems [146].

4.2 Advertised Ancillary Services

In this section we focus on the use-case of Advertised Services which is the subsequent step in the traveler journey after the inspiration and the shopping phase (see section 3.3). The objective is to approach travelers with ancillary services such as extra luggage, airport parking, seat selection, etc. through unsolicited mail or via push-up notifications on a mobile device.

The use of email marketing campaigns in e-commerce is an essential aspect for driving sales and establishing or maintaining good customer service. In order to embrace e-commerce techniques and boost their revenues, some airlines are using the so-called Amadeus Anytime Merchandizing (AAM) Notification System [10] which is an IT solution that allows airline marketers to effectively define, deploy, monitor and adjust airline email marketing campaigns sent to travelers in real-time. Customized notifications can be defined and sent to travelers after booking a flight, to suggest them additional services to buy (e.g. extra baggage, specific meal, preferred seat). The solution acts as a bridge between the travelers retailing touchpoints and the airline's service and delivery system.
As shown in figure 4.9, when using this solution, the airline marketer can choose the appropriate time **when** to send the notification (e.g. 5 days before departure), **what** product to recommend (e.g. Leg Space Seat), **how** to send the offer (e.g. via an Email) and to **whom** this offer should be sent (matching targeting criteria).

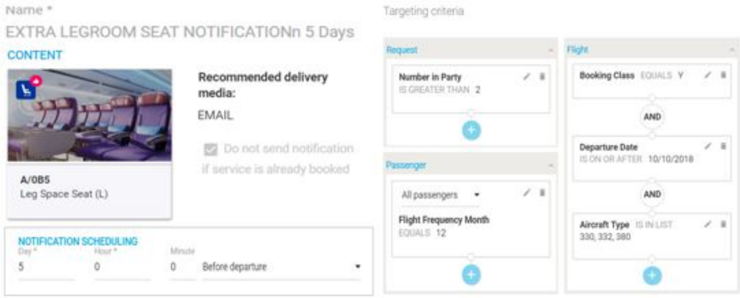

Figure 4.9 – AAM Notification System. Notification information (e.g. media used to send the notification, time of notification, etc.).

However, ensuring the success of the marketing campaigns in the airline industry is challenging, and the risk is high that the marketing campaign content is not suited to the needs of specific travelers. Indeed, despite all the functionalities included in the AAM Notification System, it is difficult for an airline to find the optimal audience to target for a given offer.

We conducted an analysis of historical sales triggered by some notification campaigns during the period 14 May 2019 - 17 Dec 2019 ran by one of our partner airlines and we observed a poor conversion of the notification offers (Section 4.2.2). This is partly due to the challenging decision-making process that an airline marketer faces when it comes to deciding which values (belonging to large value intervals) are appropriate for the criteria to be used (e.g. sending time, flight itineraries, flight departure point, etc.). Targeting customers with unsolicited notifications can be counter-productive and lead to adversarial effects on customer loyalty if

[10] https://amadeus.com/en/portfolio/airlines/anytime-merchandising

done incorrectly as the traveler may rapidly feel spammed. It is therefore critical to identify the customers that we expect to react positively to an advertised service in order to avoid spamming them with non-personalized emails. This problem can be seen as an inverse recommendation scenario, i.e. recommending a user to an item.

Inspired by recent works that have illustrated the effectiveness of using gradient boosting algorithms [68, 130] for item recommendation, we propose a gradient boosting-based algorithm [28] that leverages travelers' historical purchases and travelers' data to better target the audience in email marketing campaigns for ancillary services recommendation.

Related Work on Email Marketing Campaigns

Emails allow marketers to send messages to their customers at very low cost. They generally generate faster responses and create an opportunity for interactive communication with customers [22]. In [127], the authors analyze 70 randomized field experiments and find that email promotions not only increase customers' average purchase spending during the promotion window but also carry over to the week after the promotion expires. In our study, we will focus on personalized email marketing campaigns. In [1], the authors performed an empirical comparison of supervised machine learning models based on decision tree and logistic regression algorithms in order to improve the open rate and conversion rate of email marketing campaigns. In [34], the authors propose to use transactional features and instant messaging metadata to train a boosting tree regression algorithm to timely anticipate the needs of consumers in order to increase their level of engagement as well as the rate at which they repurchase products.

In the tourism domain, even if widely used, limited research is conducted on email marketing campaigns. In [139], the authors found that customers' favorite emails contain special offers, discounts and coupons as well as real-time communication tools. When customers perceived these emails as meeting their personal preferences, they developed a strong relationship with the sender. In [163], the authors demonstrated that the personalization, interactivity and price were important predictors of the possibility of revisiting the same accommodation. To the best of our knowledge, we are the first to propose a supervised machine learning approach to enable personalized email marketing in the airline travel industry.

Related Work on Gradient Boosting Algorithms for Recommendation

Gradient boosting is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weaker prediction models.

In [68], the authors used multiple additive Regression Trees (Dart) which is an ensemble model that uses boosted regression trees and handles the overspecialization. Their approach for online accommodation recommendation ranked 1^{st} in the famous RecSys 2019 Challenge[11]. In [130], the authors used XGBoost [19] which is an implementation of gradient boosting decision trees to predict tweet engagement, they have intensively used exploratory data analysis to extract and compute relevant features to feed XGBoost algorithm. Their solution ranked 1^{st} in the RecSys 2020 Challenge[12]. In this work, we use XGBoost algorithm as a supervised machine learning algorithm to better target the audience in email marketing campaigns based on flight contextual features and handcrafted features.

4.2.1 Problem Formulation & Preliminaries

Preliminaries:

Definition 4.2.1 *A notification campaign is a set of notifications sent to an audience of travelers within a given period of time and under some criteria to recommend an ancillary product.*

Definition 4.2.2 *We define the conversion rate of a notification campaign as follows:*

$$CR = \frac{1}{N_o} \sum_{i=1}^{N_o} hit(N_i) \tag{4.10}$$

where N_o is the number of notifications sent through the notification campaign, and $hit(N_i)$ is equal to 1 if the notification N_i triggers a purchase. In our work, we focus on optimizing the conversion rate.

Problem Formulation:

To address the research question *RQ1.2*, we formulate the following problem: Given a notification campaign aimed at a large audience of travelers who have already booked a flight in a given context, we aim to target the relevant travelers among all the travelers that the notifications will reach. As part of the study, we address the following questions:

1. How to extract the relevant sample of travelers to target for a given notification campaign?(Figure 4.10).
2. How does a supervised machine learning approach perform compared to a rule-based approach to target the relevant audience for a notification campaign?

[11] https://recsys.acm.org/recsys19/challenge/
[12] https://recsys.acm.org/recsys20/challenge/

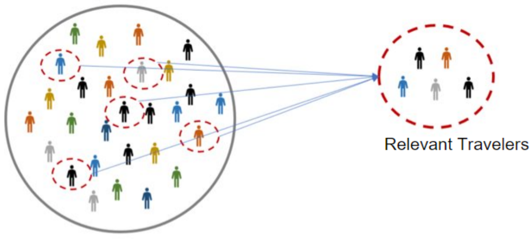

Figure 4.10 – The task is to extract the relevant travelers among the whole set of travelers that were initially targeted by the notification campaign through AAM Notification System.

4.2.2 Data

In this section, we first start by describing the notification campaigns analyzed and used as part of this work, then we present the constructed dataset used as input of the machine learning models.

Notification Campaign Analysis

We analyzed three notification campaigns involving approximately 8.2 million notifications sent by one of our partner airlines to its travelers between 14 May 2019 and 17 December 2019 in order to understand the behavior of travelers in response to the notification campaigns, and to compute the conversion rates of these notification campaigns.

As shown in Table 4.8, there are three different types of ancillaries that are advertised in three notification campaigns. By analyzing airline sales data over the same period, we can see that only 3 out of 34 different types of purchased ancillaries were offered in the notification campaigns. This shows an untapped sales potential. Moreover, we observed that 50% of sales triggered by a notification happens on the same day (< 24 hours) after receiving the notification. This demonstrates the effect of a notification on the purchases.

While the prepaid baggage notification campaign was aimed at all travelers who booked a flight during the period indicated in Table 4.8, the notification campaigns for Lounge access and Extra leg room seat contain a number of filtering criteria that explain the large discrepancy in terms of the number of notifications sent out. Indeed, for these two notification campaigns, the airline marketer chose to send the notification to a quite restrictive audience by combining a number of criteria (fare family, aircraft type, no chargeable seat in their booking, etc.)

Table 4.8 – Conversion rates of notification campaigns: rule-based approach.

Notification Campaign	Notification time	Date Range	Number of Notifications	Sales	CR
Extra leg room seat	5 days before Departure	19 May - 23 December 2019	~355 K	~2.8K	0.8%
Prepaid baggage	2 days before Departure	14 May - 17 December 2019	~7.5 M	~11K	0.15%
Lounge	Right after air ticket purchase	16 October - 17 December 2019	~338 K	104	0.03%
All Notifications	-	-	-	~13.8K	0.18%

Airline Travel Notification Dataset

We conducted experiments on a real-world production dataset of bookings from the T-DNA database[13]. Each booking contains one or several air ticket purchases, and is stored using Passenger Name Record (PNR) information. This is the same source of data used in the work of 'Next Trip Recommendation' presented earlier in section 4.1.3. The considered dataset contains approximately 2.33 million bookings for approximately 2.85 million unique travelers.

The Airline Travel Notification (ATN) dataset is produced by joining the notification dataset and the historical bookings dataset from T-DNA. This dataset contains information about the shopping and booking context (e.g. search date, number of passenger, departure date, etc.) and information about travelers (e.g. demographics and loyalty membership information). In total, the dataset contains 42 columns and ~ 8.2 million rows. For our experiments, the dataset was broken down into three different sub-datasets that correspond to the three different notification campaigns (Table 4.8).

4.2.3 Machine Learning-based Notification Targeting

We propose to develop a supervised machine learning model that includes as input contextual features and additional handcrafted travelers' features that capture travelers' preferences which we think could be particularly significant for model accuracy (hypothesis proven in the ablation study) as another baseline to compare with. Handcrafted travelers' features are features designed to capture travelers' preferences for ancillaries, destinations, points of sale, etc. and also customer lifetime value.

We compute several features based on travelers purchase history, such as preferred ancillary, preferred destination, etc. We list below the features computed that leverage travelers' history:

[13]T-DNA: traveler DNA is a database that contains bookings of travelers over a dozen of airlines

- *Bookings count*: Number of bookings already purchased by the traveler with the airline.
- *Average flight revenue*: The average booking price tickets for all historical bookings of the traveler.
- *Preferred ancillary*: This feature corresponds to the most sold ancillary to the traveler.
- *Preferred destination*: This feature corresponds to the most visited destination (airport) by the traveler.
- *Preferred seat characteristic*: This feature represents the seat characteristic that is the most purchased by the traveler. There are three types of seat characteristic namely Upper deck, Exit Row, Leg Space.
- *Extra leg room seat already purchase*: For each seat characteristic, we create a binary feature that represents if a traveler has already purchased an Extra leg room seat or not.
- *Seat sales count*: This feature represents the number of times a seat has been purchased by the traveler.
- *Prepaid baggage sales count*: This feature represents the number of times a prepaid baggage has been purchased by the traveler.
- *Lounge sales count*: This feature represents the number of times a lounge access has been purchased by the traveler.
- *Notification response rate*: This feature is equal to the number of sales divided by the number of notifications sent to the traveler (regardless of the recommended service).

The handcrafted features and the features available in the ATN dataset are used as input of a gradient boosting decision tree classifier. We use the official implementation[14] of XGBoost [19] to train a binary classifier to predict if the notified travelers will convert or not. In Section 4.2.4, we give more details about the hyper-parameters used in XGBoost.

4.2.4 Experimental Setup

In this section, we present the different settings and the evaluation protocol (evaluation metrics and split of the dataset) used to conduct the experiments.

Training & Test Sets: The three datasets corresponding to the three notification campaigns are split using the same strategy. Each dataset is sorted temporally, and 80% of the first rows of each dataset are used as training/validation sets. We use a cross-fold validation to train and

[14] XGBoost:https://XGBoost.readthedocs.io/

validate all models (k=5, a split of 80% for training and 20% for validation). The remaining 20% are used as test set to evaluate the model. The split between training and validation set is performed randomly in order to avoid a seasonality effect that is usually occurring in the travel industry.

Evaluation metrics: The output of our approach is the probability of purchasing the recommended ancillary a included in the notification N:

$$P(purchase = a|N) = P(purchase|Context, travelers' features) \tag{4.11}$$

To evaluate and compare, the different approaches implemented, we used the conversion rate defined at definition 4.2.2 and the three metrics defined as follows:

- **TPR**: The true positive rate is the percentage of correct positive predictions. It represents the ratio of travelers that the algorithm suggests to send the notification and effectively purchase the ancillary. TPR is defined as follows:

$$TPR = \frac{TP}{(TP + FN)} \tag{4.12}$$

- **TNR**: The true negative rate is the percentage of correct negative predictions. It represents the ratio of travelers that the algorithm suggest to not send the notification and effectively do not purchase the ancillary. TNR is defined as follows:

$$TNR = \frac{TN}{(TN + FP)} \tag{4.13}$$

- **ROC-AUC**: The area under ROC curve (FPR, TPR) helps to choose what is the optimal probability threshold that maximizes the CR and TPR and is defined as follows:

$$ROC\text{-}AUC = \int_0^1 TPR\, d(FPR) \tag{4.14}$$

  where, $FPR = 1 - TPR$ is the false positive rate

It is noteworthy that the conversion rate was measured offline as well as all the metrics based on the test set. According to equation 4.10, N_o represents the number of predicted positives and each hit hit_i corresponds to a true positive prediction.

Implementation Framework & Parameter Settings: The hyper-parameters of all the models were tuned using a combination of random-search and grid-search algorithms. We optimize the following hyper-parameters of XGBoost classifier: the max depth of a tree $\in [5, 50]$, the number of trees $\in [10, 100]$, the sub-sample of each tree $\in [0.65, 0.85]$ and the col-sample of each tree $\in [0.65, 0.85]$. In addition to these hyper-parameters, we compute a weighted score (ratio of number of negative class to the positive class) that we use in XGBoost to approach the problem as a cost-sensitive learning problem due to the high class imbalance between positive (purchase) and negative (no purchase) classes (Table 4.8).

4.2.5 Results

In this section, we discuss the results obtained from the experiments. Results of the experiments conducted are presented in table 4.9. TPR, TNR and ROC-AUC metrics are not provided for the rule-based approach implemented in AAM Notification System. The reason behind this is that the dataset used in the experiments is generated by the AAM notification system, which is different from the original dataset that contains all travelers used for the rule-based approach to identify the travelers matching the targeting criteria.

Empirical Comparison of Machine Learning Binary Classifiers

We perform an empirical comparison of different Machine Learning algorithms for the task of predicting the probability in Eq 4.11. We report the results in table 4.9.Results show that XGBoost algorithm is the best performing algorithm for this task with respect the the four metrics defined to compare the different ML algorithms. Furthermore, we show that in general the use of machine learning algorithms gives better performance than the use of a rule-based algorithm (the system currently in production). It is important to note that the probability computed by the ML algorithms is used by the AAM system to decide whether to recommend a certain ancillary service or not.

Table 4.9 – Recommendation performance of different machine learning classifiers with respect to 3 different ancillary services. LR: *Logistic Regression* [153]; DT: *Decision Tree* [14]; RF: *Random Forest* [13]; XGB: *XGBoost* [19]. The average standard deviation (by varying the seed when splitting the dataset) of each metric is as follows: $AUC-ROC: \pm 0.02$, $TPR: \pm 3\%$, $TNR: \pm 2\%$, $CR: \pm 0.1\%$

Model	Extra leg room seat				Prepaid baggage				Lounge			
	AUC-ROC	TPR	TNR	CR	AUC-ROC	TPR	TNR	CR	AUC-ROC	TPR	TNR	CR
Rule-based	-	-	-	0.8%	-	-	-	0.15%	-	-	-	0.03%
LR	0.77	78%	70%	2.4%	0.80	80%	68%	0.36%	0.81	80%	65%	0.31%
DT	0.73	75%	67%	2.2.%	0.78	77%	67%	0.29%	0.77	77%	62%	0.29%
RF	0.79	80%	67%	2.6%	0.83	81%	68%	0.38%	0.83	82%	65%	0.32%
XGB	0.83	85%	65%	2.8%	0.88	86%	74%	0.56%	0.89	88%	65%	0.36%

Ablation Study

Table 4.10 shows that using the features from the contextual features in addition to the travelers handcrafted features (*C+T*) as input of XGBoost performs better than using only one of them (*C* or *T*) as input for all notification campaigns. We also observe that using travelers handcrafted features as input information of XGBoost gives better results than using the entire ATN dataset that contains the contextual features for all the notification campaigns. We compute the information gain of all the features to determine the most important features of XGBoost model for each notification campaign and we report below the three most important ones with their respective information gain:

- Extra Leg Room Seat: {*Preferred Seat Characteristic*: 0.31, *Preferred ancillary*: 0.12, *Ticket amount*: 0.08}.
- Prepaid Baggage: {*Preferred destination*: 0.21, *Destination*: 0.12, *Prepaid Baggage sales Frequency*: 0.10}.
- Lounge: {*Average Flight Revenue*: 0.22, *Destination*: 0.20, *Age*: 0.15}.

Table 4.10 – Recommendation Performance of XGBoost algorithm for different inputs; *C* represents the contextual features, *T* represents the handcrafted travelers' features.

Input Features	Extra leg room seat				Prepaid baggage				Lounge			
	AUC-ROC	TPR	TNR	CR	AUC-ROC	TPR	TNR	CR	AUC-ROC	TPR	TNR	CR
C	0.75	78%	58%	2.2%	0.83	80%	71%	0.38%	0.76	80%	62%	0.18%
T	0.79	81%	60%	2.37%	0.85	82%	74%	0.4%	0.84	86%	67%	0.22%
C+T	0.83	85%	65%	2.8%	0.88	86%	**74**%	0.56%	0.89	88%	65%	0.36%

4.2.6 Summary

In this section, we have presented a recommender system developed to recommend a given ancillary to the right audience. The recommender system is based on simple ML classifier that computes the probability to recommend the ancillary service that an airline wants to suggest to their travelers that have booked a flight ticket within a given period. By recommending the right ancillary to the right traveler, we addressed the second research sub-question of '*RQ1.2* : What ancillary service should be recommended to each traveler?'. Moreover, we conducted extensive experiments to answer the questions raised in section 4.2.1:

1. How to extract the relevant sample of travelers to target for a given notification campaign?

The results of the experiments presented in table 4.9 show that extracting the relevant audience for a given notification campaign is not an easy task. Indeed, despite the fact that the conversion rate increases significantly with our approach, it remains relatively small. However, thanks to our approach, notification campaigns are better targeted and we manage to avoid recommending an unsuitable ancillary service to at least 65% of passengers.

2. How does a supervised machine learning approach perform compared to a rule-based approach to target the relevant audience for a notification campaign?
   Experiments have shown that the handcrafted features based supervised machine learning approach gives better results than the rule-based one. Indeed, in table 4.9, we can observe that the conversion rate is multiplied by more than 3 for Extra Leg Room Seat, almost 4 for Prepaid Baggage, and 12 for Lounge. Hence, we prove the benefit of using supervised machine learning over a simpler rule-based approach while it is the currently adopted mechanism used by airline marketers. It should be noted that the list of possible criteria available in AAM Notification System (see figure 4.9) is the same as the list of features used in the supervised machine learning approach.

Even if our approach has shown very good performances in the ancillary service recommendation, it is important to say that the features engineering step is very costly not only in terms of time because it requires an important time of reflection and a participation of functional experts of the domain, but also in terms of memory where the features computed in the dataset must be stored and added to the dataset. It is therefore quite logical to think about designing an automated features engineering process. The representation of features in a latent form (such as embeddings) is an option that we consider and which is explored in the next chapter (see section 5.2). The idea is to be able to build a knowledge graph that will contain all the information used to compute the handcrafted features (see section 5.1.2), but this time, instead of computing them, we will use KG embedding algorithms and represent all the information as embeddings. Thus, we will replace the handcrafted features by KG embeddings and evaluate the recommendation performance of using them as input of XGBoost algorithm.

4.3 Hotel Recommendation

In this section, we focus on the "hotel recommendation" use-case, which is a specific use-case for cross-selling third-party content presented earlier in section 3.3.

When the traveler has decided on his/her preferred itinerary, he/she enters the booking stage. At the booking stage, the traveler is in a planning mindset and this is an ideal opportunity to both increase ancillary revenues for the airlines as well as offer a one-stop shopping experience that covers the traveler's full journey. At this stage, the airline can take the role of Hotel metasearch engine to recommend to the traveler a ranked list of hotels based on the traveler's preferences, navigation data, etc.

To tackle this use-case, we make use of a public dataset of hotel search sessions released by Trivago[15] provided as part of the RecSys 2019 Challenge[16] where the goal is to predict which hotels (items) the user has clicked on among the search results provided by Trivago metasearch during the last part of the user session in an offline evaluation setup. Two objectives are aimed at: improve the click-through rate of Trivago navigation sessions and personalize search results for Trivago users. In the remaining part of this section, we speak of *accommodation* rather than hotel according to the terminology of Trivago platform.

Personalizing search results for each traveler is key and can lead to better conversion of offers presented to them, however the task is not trivial. Indeed, with the growing desire to travel on a day-to-day basis, the number of accommodation offers users can find on the web is increasing significantly. Therefore, it becomes important to help travelers choose the right accommodation to stay among the multitude of available choices. Recommender systems play an important role in filtering out undesired content first and keeping only content that the user might like, and then reordering the remaining choices in a personalized way.
In particular, in a user's navigation session, SB recommender systems (see section 2.1) can help the user to find more easily the elements he/she wants based on the actions he/she has performed. In such setting, users are not always known and identified, and we do not necessarily have long-term user profile for all users. Traditional models propose to use item nearest neighbor schemes to overcome this user cold start problem [87] or association rules in order to capture the frequency of two co-occurring events in the same session [5]. In recent years, some research works have focused on Recurrent Neural Networks (RNN) [59] considering the sequence of user's actions as input of the RNN. The RNN learns to predict the next action that will be made by a user given a sequence of past actions.

Inspired by these recent advances, we propose to develop a many-to-one RNN which predicts whether or not the last element of an action sequence is clicked by a user, as shown in figure 4.11.
Our approach consists in two stages: we first learn to compute the probability that a user clicks on an item (accommodation) given the previous actions made during the session based on a multi-architecture neural network composed of a RNN and a MLP, then we apply a rule-based algorithm that explicitly places the elements seen in the previous steps at the top of the accommodation list displayed to the user.

[15] https://www.trivago.com/
[16] http://www.recsyschallenge.com/2019/

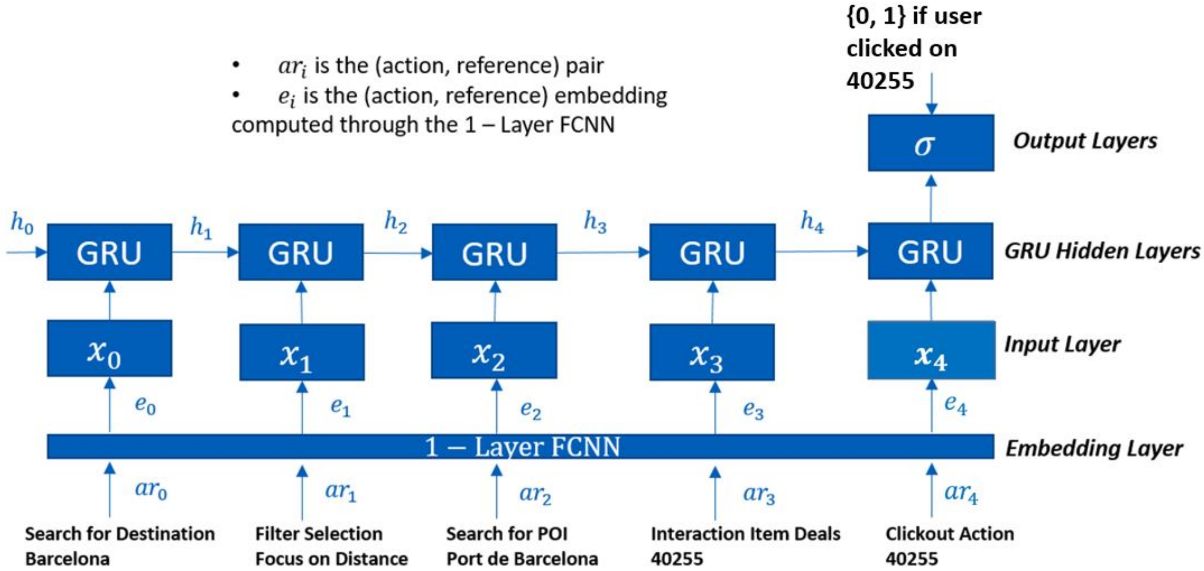

Figure 4.11 – Many-to-one Recurrent Neural Network for hotel recommendation.

Problem Formulation

What is the feeling of a person who says “Rooms of the hotel are enormous, the staff is friendly and efficient”? It is a positive one. Similarly to the sequence of words in a sentence where one can affirm what the feeling is, analyzing a sequence of actions performed by a user in a navigation session can lead to predict what will be the item the user will add to his cart at the end of the shopping session.

Based on this analogy with the language realm, where the use of sequential models and more especially RNNs has proven its efficiency [32] on several Natural Language Processing (NLP) tasks, we propose to develop a many-to-one recurrent neural network that learns the probability that a user will click on a hotel accommodation based on the sequence of actions he/she has performed during his/her browsing session.

More specifically, we combine a rule-based algorithm with a Gated Recurrent Unit (GRU) RNN in order to sort the list of accommodations that is shown to the user.

4.3.1 Data

The first part of our work consists in conducting an exploratory data analysis to understand users’ behavior on Trivago website. The dataset published for the challenge consists of interactions of users browsing Trivago website collected from 01-11-2018 to 09-11-2018 (9 days). More precisely, for a given session, we have in the dataset: the sequence of actions performed by the user during the session, the filters that were applied during the session, the accommodation list displayed to the user when performing a click-out action, plus the price of each accommodation in the impression list. In addition to these information, we have two contextual features: the device and the platform used by the user to perform the searches. The remainder of this section presents statistics and overviews of training data. We report in table 4.11 some statistics on important dataset variables.

Table 4.11 – Dataset Characteristics

Variable	Value
Number of users	730 803
Number of sessions	310 683
Number of actions	15 932 992
Number of different accommodations	927 142
Number of different destinations	34 752
Number of different platforms	55

General Statistics on Training Data

We report in figure 4.12, 5 summary statistics of different variables that characterize user sessions. The statistic tables highlight two important observations:

- *Dispersion*: The number of actions per session has a high standard deviation which means that the data is highly spread. For all the variables, we also have a very high maximum value which demonstrates the skewness of users' behaviors.
- *Actions required to end up on a 'click-out' action*: On average, a user performs 17.5 actions in a session. However, the average number of actions needed to perform a 'Click-out Action' is only 8, so what does the rest of the clicks correspond to? In more than 72% of cases, the last performed action in a session is a 'click-out' action. However, in 28% of all sessions, there are other actions following the 'click-out' action.

Sorting and Filtering Actions

In figure 4.13, we plot the histogram of most of the 15 filters used. The observed distribution does not follow a long-tail distribution and all filters are more or less used in similar proportion. We can thus infer that there are different types of user behaviors. More specifically, we compute the ratio of sessions where users use filter or sort buttons: this ratio is equal to 14% which represents a significant subset of the data. We also compute the average number of clickout actions performed per session for each platform and notice that there is a significant difference between people that are searching for accommodation using the Japan platform (8.7 clickout actions) and the Brazil one (23.9 clickout actions). Finally, we compute the average time a user

Number of sessions per User	
Max	201
Min	1
Mean	1.25
Std	0.75
Median	1

Number of actions per Session	
Max	3522
Min	1
Mean	17.5
Std	48.2
Median	4

Number of Clickout per User	
Max	284
Min	0
Mean	2.2
Std	2.8
Median	1

Number of Clickout per session	
Max	97
Min	0
Mean	1.74
Std	1
Median	2

Figure 4.12 – Statistics on Trivago dataset

spends in a session (8 minutes), and we noticed that there is a high standard deviation for this variable (22 minutes) which again demonstrates the dispersion in users' sessions. This leads us to the conclusion that there are different user profiles and behaviors. For example, we have users who need a lot of actions to finally perform a clikckout, users who perform volatile clicks, users who have to look at the images of the accommodation and then click on it, etc. Explicitly adding this information to our model can help to more effectively predict the user's clickout element. In addition, the idea of having a different model for each type of user is something that should be experimented with.

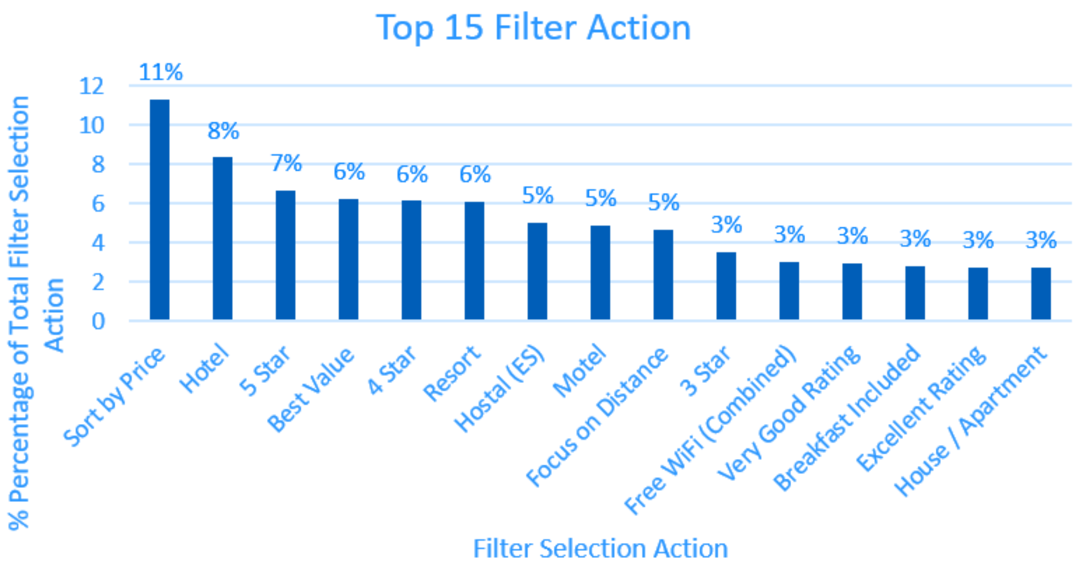

Figure 4.13 – Top 15 used filters

Accommodation Content

In addition to information on user sessions, a description of the accommodation is also provided. This enriches the input data of our recommendation system. We have 157 different properties that describe an accommodation (e.g. Wifi, swimming pool, good rating, etc.). We use these properties to enrich the input data of the neural network as explained in figure 4.16.

Number of properties per item	
Max	112
Min	1
Mean	19.7
Std	18.4
Median	15

Number of items per properties	
Max	533 286
Min	349
Mean	116 310
Std	123 047
Median	65 514

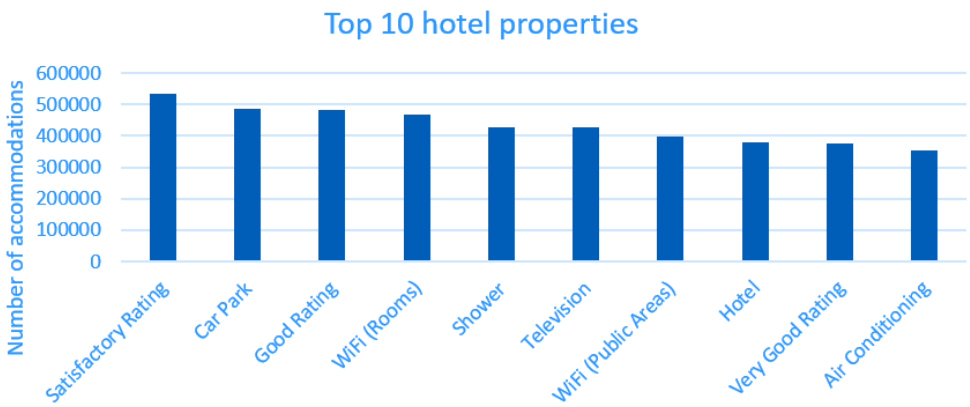

Figure 4.14 – Statistics on properties that define accommodations

4.3.2 Combining Rule-based and Supervised Learning Algorithms for Hotel Search Ranking

Our approach is a two-stage model that consists in computing a score for each element of the impression list displayed to the user when he/she performs a clickout action based on a supervised learning model, then applying a rule-based algorithm to the ranked list returned by the supervised learning model to reorder the ranked list of hotels(see figure 4.15). The objective of the model is to compute a list of probabilities:

$$P(a = clickout, r = c_t) = P(r_t = c_t | ar_{t-1}, ar_{t-2}, ..., ar_0), \tag{4.15}$$

where r is the reference of the accommodation and a the action, $c_t \in D_{acc}$.

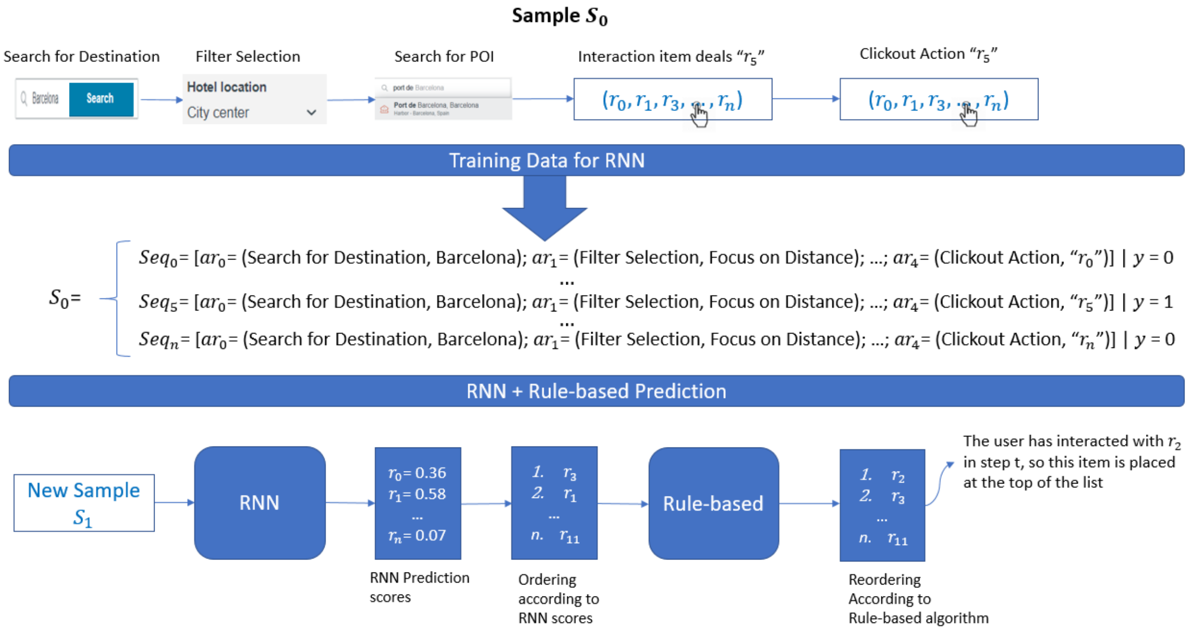

Figure 4.15 – Combining Rule-based and Supervised learning algorithms for Hotel ranking.

4.3.3 A multiple Neural Network Architecture for Hotel Search Ranking

The supervised learning model is a multiple neural network architecture model combining a RNN that considers the users' navigation actions (e.g. clickout, filtering, etc.) and a MLP that incorporates multiple information about the context of the navigation session and also the content of the accommodations that the user interacted with during the session. We present the model architecture in figure 4.16. The implementation of our method is publicly available at https://gitlab.eurecom.fr/dadoun/hotel_recommendation.

Recurrent Neural Network

Recurrent neural networks are widely used for many NLP tasks such as named entity recognition, machine translation or semantic classification [138]. Indeed, this neural network architecture works very well when it comes to recognizing sequence-based patterns and predicting the following element from a sequence of previous elements. It is therefore a natural choice to use this neural network architecture for the next click prediction based on the sequence of actions performed by the user. However, unlike [135], we consider our problem as a binary classification instead of a multi-label classification problem. More precisely, the RNN takes as input a sequence of actions with their corresponding references, represented by a one-hot encoding vector and fed into a one fully connected neural network in order to compute the (action, reference) embeddings, plus the last action that corresponds to a clickout with its reference, and then returns $P(r_t|ar_{t-1}, ar_{t-2}, ..., ar_0)$, where ar_i indicates the

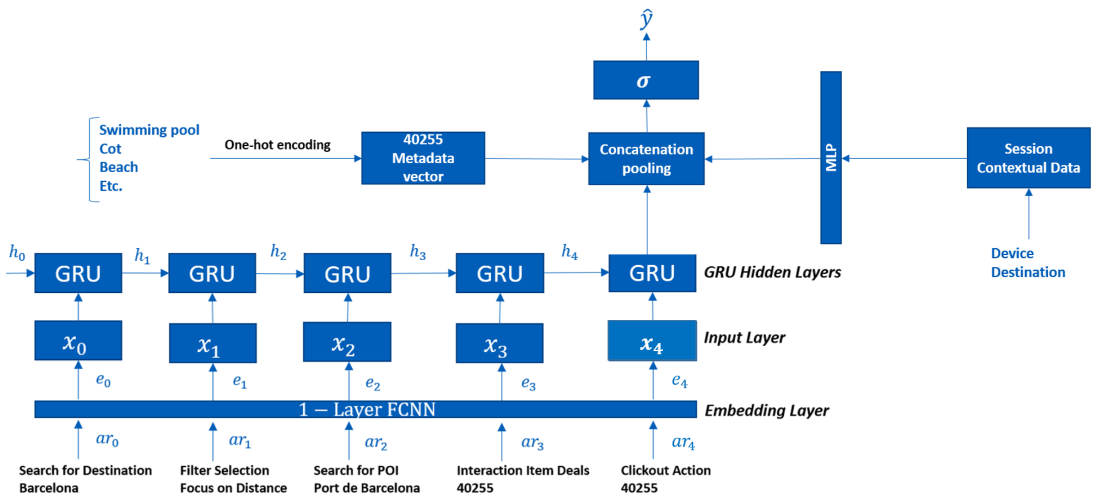

Figure 4.16 – RNN and MLP combination for content and context information

(action, reference) pair made by the user at step i. This probability indicates if the user has clicked in the accommodation r_t given the sequence of previous (action, reference) pairs $(ar_{t-1}, ar_{t-2}, ..., ar_0)$. Therefore, for each clickout action, our RNN returns a score for each item in the impression list, then the list is sorted in a decreasing way according to the score corresponding to each accommodation.

Multi-Layer Perceptron

In addition to the sequence of actions performed by the user, we first enrich our input data with the content of the accommodation, and then we add the contextual information of the session as shown in figure 4.16. The content information are represented using one hot-encoding technique where each element of the vector corresponds to a property (e.g. Wifi, restaurant, etc.) that represents the accommodation. We use the device and the platform as session-contextual information. These two categorical features are one-hot encoded as well and fed into a MLP as represented in figure 4.16. The MLP used is a 2 layers feed-forward neural network. The size of each layer is being optimized using Grid search as specified in section 4.3.5. We use GRU cells [24] in order to compute hidden states h_t for each step t and a sigmoid function in order to compute the probability score $\hat{y} = \sigma(W_y h_t + b_y)$, where $\sigma(x) = \frac{1}{1+e^{-x}}$.

Reordering Search Results via Rule-based Algorithm

In the opposite to the supervised learning algorithm which predicts a score for each element in the impression list, the rule-based algorithm simply reorders the accommodation list based on explicit prior items the user interacted with in previous actions. The motivation to use this rule-based algorithm comes from the data analysis made beforehand which shows an interesting and recurrent pattern: in several sessions, users who have interacted with an accommodation have performed a clickout action on this accommodation slightly later, where interacting with an accommodation is among the following actions: I_{acc} = {Interaction item ratings, Interaction item deals, Interaction item image, Interaction item information, Search for item, Clickout item}. The closer the element in I_{acc} is to the clickout action, the higher it is placed at the top of the list as illustrated in figure 4.15.

4.3.4 Experimental Setup

Evaluation Protocol and Metrics

Trivago published a set of training data used to train our model and a set of test data[17] that is split into validation set in order to compute scores of our model in local and confirmation set which is the subset of the data used to submit the results in the submission page[18].
As proposed by the challenge organizers, we use mean reciprocal rank metric to evaluate our model (see section 4.1.5).

We also implement a set of baseline models (see section 4.3.4) with which we compare our model.

Baseline Models

Most used recommender systems are based on a long-term user history which can lead to implement algorithms such as *matrix factorization* [80] as a baseline algorithm. However, in SB recommender systems, we do not have such long user past interactions [93]. Different baselines are implemented in the setting of SB recommendation as proposed in [93], and are described bellow:

- *Association rules* [5]: The association rules algorithm is designed to capture the frequency of two co-occurring events in the same session. The output of this algorithm is a ranking of next items based on the current clicked item.
- *Markov Chains* [109]: Similarly to the association rules algorithm, markov chains also captures the co-occurring events in the same session, but only takes into account two events that follow one after the other(in the same session).

[17] http://www.recsyschallenge.com/2019/Dataset
[18] http://www.recsyschallenge.com/2019/submission

- *Sequential rules* [71]: This method is similar to association rules and markov chains since it tries to capture frequency of co-occurring events, but it adds a weight term that captures the distance between two occurring events.
- *IKNN* [58]: In this algorithm, each item is represented by a sparse vector V_i of a length equal to the number of sessions, where $V_i[j] = 1$ if the item is seen in the session j and 0 otherwise.

Implementation Framework & Parameter Settings:

Our model and all baseline algorithms are implemented using Python and Tensorflow library[19]. The hyper-parameters of the RNN were tuned using grid-search algorithm. First, we initialize all the weights randomly with a Gaussian Distribution ($\mu = 0$, $\sigma = 0.01$), and we use mini-batch Adam optimizer [75]. It is worth mentioning that other optimizers are also tested. However, Adam Optimizer has shown to be the most efficient in time and also accuracy. We evaluate our model using different values of hyper-parameters:

- Size of ar embeddings: $E\_size \in \{64, 128, 256, 512\}$
- Hidden state: $h\_size \in \{64, 128\}$
- Batch size: $B\_size \in \{32, 64, 128, 256, 512\}$
- Number of epochs: $epochs \in \{5, 10, 15, 20\}$
- Learning rate: $l_r \in \{0.0001, 0.0005, 0.001, 0.005, 0.01\}$
- MLP layers size: $l\_sizes \in \{[256, 128], [128, 64], [64, 32]\}$

4.3.5 Results

Empirical Comparison

The results are reported in table 4.12. The scores correspond to an average of numerous experiments of Mean Reciprocal Rank metric computed on the validation set proposed by the organizers. After running extensive experiments to tune the hyper-parameters, our approach has shown to be the most accurate with a score of 0.59. Association rules and sequential rules give promising results: 0.52 and 0.51 respectively, when the Markov chains give only a score of 0.34. This shows that it is more important to consider all the elements seen in a session as close to each other than to consider only those seen sequentially close to each other.

[19] Python Tensorflow API: https://www.tensorflow.org

Table 4.12 – MRR scores on Validation set

Model	MRR
Association Rules	0.52
Markov Chains	0.34
Sequential Rules	0.51
IKNN	0.54
RNN only	0.49
RNN-MLP (Metadata only)	0.50
RNN-MLP (Context only)	0.49
RNN-MLP	0.50
Rule-based only	0.56
RNN-MLP + Rule-based	0.59

Lessons Learned

The task of predicting which element the user will click on based on performed actions is treated in a similar way to predicting the next word in a sentence. However, while the context in a sentence is very important and plays a big role in considering that two consecutive words have a sense, hence the use of RNN, we cannot be sure that the context is just as important for our task, especially when we look at the volatility of actions made by users in the same session. This leads us to question ourselves, especially when we look at the results obtained from the association rules and the method of the K-nearest neighbors (IKNN) which are better than the Markov chains method or the sequential rule method. This demonstrates that the succession of actions is not as important as it is assumed at the beginning of this work, and that the simple fact of considering the set of actions than the sequence of actions could have probably lead to better results.

The second important point to emphasize is the dispersion of user behavior in the website: indeed, when analyzing the data, we noticed that there are several types of users, which makes it complicated and difficult to build a model for all types of users and to find a pattern that generalizes all the different behaviors in order to make accurate predictions for our task. The idea proposed during the data analysis which is to create different models per user seems to be a good idea as well.

Lastly, the simple rule-based method is the most efficient one if we consider independently each algorithm and is not as far from the method that obtained the best result in this challenge (0.648 against 0.689 in the validation set as shown in the leaderboard). Given that this method does not require any learning, nor much computation time, it is worth using this method for simple cases as the example shown in Figure 4.15.

4.3.6 Summary

In this section, we worked with publicly available dataset that comes from one of the most demanding recommender system challenge. The aim of the challenge is to help the user find easily the accommodation in which he/she wants to stay, and to place it in the top of a list of different accommodations that are proposed to him/her, given previous performed actions in a session.
However, by using as input only the actions performed by the user in addition to some information related to the context of the session such as the user's device or country platform (e.g. '.fr', '.en'), such a task becomes hard.
Especially, in the travel sector where the context is very important: the seasonality effect or the number in party (e.g. traveling alone, in a group or with a family). Moreover, even if the accommodation properties are provided in the dataset, it is not possible to enrich the accommodations with external data due to the anonymization of the dataset. This could have been very beneficial for improving the recommendations provided to the users as the dataset used in SB recommender system are very sparse and more importantly contains many new users (cold start problem). In the following chapter (see chapter 5), we demonstrate how knowledge graphs help in overcoming these limitations through the enrichment of items.

4.4 Summary

In this chapter, we described the development of recommender systems that allow us to tackle different airline specific recommendation system use-cases that cover some phases of the traveler journey, and thus to personalize the offer suggested to travelers across all phases they go through. This has allowed to address the research question ***RQ1*** raised in section 1.5.
The use of historical traveler data, as well as the descriptive content of the products offered by the airlines, has proven to be beneficial in the recommendation performance through the development of hybrid recommender systems and the different ablation studies conducted in the experiments (see sections 4.1.6 and 4.2.5). However, this ad-hoc process of integrating data from different sources as done in the case use of 'Next trip recommendation' is not sustainable in the long run. Additionally, the heavy feature engineering work done to build the recommender system that tackles the ancillary recommendation use-case is heavy and demanding not only in terms of time but also in terms of memory. Finally, the intrinsic nature of the data: heterogeneous and highly sparse data due to the specificity of the airline industry represents serious limitations that need to be considered by using appropriate recommender system algorithms.

The integration of data in a knowledge graph allows us to overcome many of these limitations, including the extensive work involved in feature engineering which provides us with the advantage of data augmentation using semantic web technologies to integrate data coming from

different sources.
Moreover, the use of knowledge graphs allows us to integrate heterogeneous data from different sources into a single data structure, which allows us to use KG embedding algorithms to avoid going through the heavy task of feature engineering. Finally, the use of KG recommender systems alleviate the problem of data sparsity by leveraging the numerous side information added into the knowledge graph and their beneficial interactions to understand what characterizes a product and draw users' preferences [146].

In the next chapter of the thesis (see chapter 5), we first describe how we build the knowledge graph that contains information about travelers' bookings, demographic data, contextual information, and also numerous information about KG entities coming from different sources on the web (e.g. Wikidata). Then, we revisit two airline specific recommendation use-cases addressed in this chapter using KG recommender systems in order to demonstrate the improvements they bring over traditional hybrid recommender systems and more particularly how they overcome the limitations mentioned above.

Chapter 5

Knowledge Graph-based Recommender Systems in the Airline Travel Industry

In this chapter, we revisit two airline specific recommendation use-cases namely 'Next Trip Recommendation' and 'Advertised services' with the objective to overcome the limitations presented earlier in section 4.4; for each of the use-cases we develop a KG recommender system algorithm using as input the knowledge graph described in section 5.1. We conduct extensive experiments to compare these KG recommender systems with the ones developed in chapter 4 additionally to a set of baseline algorithms that belongs to the same family of recommender systems and, for each of the use-cases, we address the research sub-questions that derives from ***RQ3***.
Each section in this chapter is dedicated to a use-case. We structure the sections as follows: we first remind the reader of the problem we want to answer after introducing the subject and presenting some related works, then we briefly present the knowledge graph that will be used as input of the recommender system, followed by a description of the model developed to address the problem. Finally, we present the experiments performed to demonstrate the effectiveness of the model and lastly we give some conclusions and outline some future work about the use-case.

5.1 Airline Travel Knowledge Graph

In this section, we present the methodology used to design the 'Airline Travel Knowledge Graph' that we use as input of the KG recommender systems developed to address the two recommendation use-cases treated in this chapter and more importantly helpful to address the limitations raised in the previous chapter.

The knowledge graph that we intend to build must first represent the traveler journey from the time he/she is searching for his/her flight to the time he/she reaches his/her destination. In order to capture all the information about the flight search, the context of the flight and the traveler's demographics, we use the large customer relationship management database

T-DNA (described in section 4.1.3) that contains travelers' bookings of dozens of airlines. In this thesis, we partner with a major Asian airline to collect their data from T-DNA database and use it to build our knowledge graph.
In addition to T-DNA database, we use other data sources that contain transactional information including relevant information related to the context of the flight, the purchase context (which can be useful for travel recommender systems) and pricing information. We present in details the data sources in section 5.1.1.
Moreover, in order to bring additional knowledge to the different entities in the knowledge graph, we leverage numerous data available in the web (e.g. Wikidata) and incorporate them into the knowledge graph. We present the different sources and information integrated in the knowledge in section 5.1.3. Finally, we present some statistics about the knowledge graph that we built in sections 5.3.3 and 5.2.2 .

5.1.1 Data Sources

We leverage different datasets from Amadeus in order to collect travelers' information, travels contextual data, and bookings information.

T-DNA Database

Reminder: Amadeus traveler DNA identifies travelers, builds profiles and store information about travelers bookings. In the airline industry realm a booking is referred to Passenger Name Record. Each booking contains one or several air ticket purchases, and is stored using PNR information. The PNR is created at reservation time by airline reservation system and contains information about the purchased air ticket (e.g. travel itinerary, payment information), traveler demographics and ancillaries information if purchased comprised in the EMD ticket explained later in this section.

We present in tables 5.1 and 5.2 some elements of selected tables contained in T-DNA database by taking as example dummy data.

In table 5.1, we present some demographics of three different travelers including the gender, nationality and birth date.

Table 5.1 – Excerpt of travelers demographics table.

TID	Gender	Birth Date	Nationality	Country
T-21354	Male	05-05-1988	MY	MY
T-21652	Male	27-03-1994	SG	MY
T-21123	Female	13-06-1976	CN	AU

In table 5.2, we present some booking information about the same travelers presented in the previous table. We can notice that the two travelers ‘T21354’ and ‘T21652’ booked a one way flight and are traveling together from Kuala Lumpur (‘KUL’ airport) to Melbourne (‘MEL’ airport).

Table 5.2 – Excerpt of travelers bookings table. NIP: Number in party.

TID	PNR	Trip Category	Origin	Destination	Stay Duration	NIP
T-21354	PJ936	One way	KUL	MEL	-	2
T-21652	PJ936	One way	KUL	MEL	-	2
T-21123	QF348	Round Trip	SYD	TPE	10	1

Airlines Tickets Database

In addition to T-DNA database, we use a second database that contains transactional information, purchase information, flight contextual information and finally pricing information about the purchased ticket. We categorize the tickets available in the database into two tickets: air tickets that contain information about the flight and EMD tickets that contain information about additional services purchased along with an air ticket (e.g. baggage, preferred seat, etc.). We represent in the two tables bellow some information about purchased air tickets and EMD tickets contained in the airline tickets database.

We consider again the two bookings referenced by ‘PJ936’ and ‘QF348’: In table 5.3, we can observe that travelers ‘T-21354’ and ‘T-21652’ have booked their air tickets 58 days before the flight departure using a credit card. The table contains also information about the price of the air ticket in addition to the booking class and flight contextual information such as the flight distance.

Table 5.3 – Excerpt of air ticket table.

tkt_nbr	PNR	bpt_airport	off_airport	Adv_purchased	ticket_fop	Distance	bkg_class	tkt_price
2718XXX3	PJ936	KUL	MEL	58	Credit Card	6371	Y	165
2718XXX4	PJ936	KUL	MEL	58	Credit Card	6371	Y	165
2713XXX1	QF348	SYD	TPE	22	Credit Card	7260	C	145
2713XXX2	QF348	TPE	SYD	32	Credit Card	7260	C	205

In table 5.4, we can observe that the traveler ‘T-21123’ has purchased a service referenced by ‘0BX’ which is a lounge access (see table 5.5) for a price of 45$ 2 days before the departure date through the mobile application. This traveler has also purchased a prepaid baggage (ancillary referenced by ‘0AA’) in the airline website the same day of the flight booking.

Table 5.4 – Excerpt of EMD ticket table.

tkt_nbr	PNR	associated_air_tkt_nbr	RFISC	RFIC	sales_channel	Adv_purchased	tkt_price
2613XXX3	QF348	2713XXX1	'0BX'	'C'	DAPI Mobile	2	45
2613XXX4	QF348	2713XXX1	'0AA'	'C'	1A E-Retail	22	15
2613XXX5	QF348	2713XXX2	'0AA'	'C'	1A E-Retail	32	15

Ancillary Document Description

Additionally to airlines data, collected through the booking process, we use the ancillary optional documents provided by ATPCO[1] that contains the currently defined Reason For Issuance Sub Code (ancillary identifiers) available for use in the optional services online application by airlines. The list also contains the recommended RFIC for each industry sub code. This table is useful to enrich the content of the ancillaries that airlines recommend to their travelers. We present some examples of ancillary description in table 5.5.

Table 5.5 – Excerpt of Ancillary Document Description.

RFISC	RFIC	Category	Group	Sub-Group	Description 1	Description 2	Commercial Name
0AA	C	Baggage	BG - Baggage	-	-	-	PRE-PAID BAGGAGE
0BX	G	Airport Services	LG - Lounge	-	-	-	LOUNGE ACCESS
029	D	Financial Impact	TS-Travel Services	FT-Fast Track	-	-	Fast Track

5.1.2 Ontology Design

The Airline Travel KG is constructed from the different data sources described in the previous section. We develop an ontology which is defined and available in the Turtle format[2]. To design the KG, we have defined 7 classes corresponding to top level entities and based on the various tables available in the databases presented in section 5.1.1:

- **Traveler**: A traveler is identified uniquely by a T-DNA id. A traveler has a booking history of purchases (e.g. air tickets). An instance of traveler is a `schema:Person`[3].
- **Trip Reservation**: A trip reservation (PNR) represents the booking of all travelers contained in the PNR. It contains information such as the number of passengers, the destination, etc.

[1] https://www.atpco.net/sites/atpco-public/files/digital-resource-library/Opt_Scvs_Industry_Sub_Codes_Online_C.pdf
[2] http://bit.ly/kg-ontology
[3] The prefix `schema` is used for concepts defined by https://schema.org

- **Journey**: A journey is linked to a trip reservation. Each journey has a stay duration, a departure and an arrival airport.

- **Air Ticket**: An air ticket is contained in a PNR and contains flight and transactional information.

- **EMD Ticket**: An Electronic Miscellaneous Document ticket is linked to an air ticket. It contains information on the ancillary purchased by the traveler (e.g. ancillary type, ancillary price, etc.).

- **Ancillary**: An ancillary is a service purchased by a traveler (associated to a flight) in addition to the air ticket. It is identified by a sub-code (RFISC), labeled by a commercial name, defined by ATPCO[4]. It belongs to a group of ancillaries (Group, RFIC). We propose to model the different ancillaries as Simple Knowledge Organization System (SKOS)[5] concepts and we create an ancillary thesaurus represented as a concept scheme.

- **Airport**: It represents the airport where the traveler travels to. An airport serves one or several cities.

5.1.3 Knowledge Graph Enrichment

In addition to 'Airline Travel KG', we leverage the property 'owl:sameas' to make use of Linked Open Data to enrich the knowledge graph with destinations metadata. More formally, we make use of two publicly available knowledge graphs:

- Wikidata[6] is a free and open knowledge base that acts as central storage for the structured data of Wikipedia, Wikivoyage, etc. This knowledge graph is used to provide information on geographical characteristics of travel destinations.

- STD[7] is a knowledge graph containing semantically annotated trails created starting from check-ins performed on the Foursquare social network[8].

In section 5.3.3, we give further details on the properties and entities added to the knowledge graph.

[4]ATPCO Ancillary description: https://www.atpco.net/resource/optional-services-industry-sub-codes
[5]https://www.w3.org/TR/skos-reference/
[6]https://www.wikidata.org/
[7]https://ndownloader.figshare.com/files/14209556
[8]https://foursquare.com/

5.1.4 Summary

In this section, we presented the different sources used to build the so-called 'Airline Travel Knowledge Graph'. Then we described the top level entities in the ontology that we used to construct the knowledge graph from the different above mentioned data sources. Finally, we presented the external data sources used to enrich semantically our knowledge graph using additional entities. In sections 5.3.3 and 5.2.2, we present some statistics about the knowledge graph used to revisit the recommendation use-cases tackled in this chapter.

5.2 Advertised Ancillary Services

In this section we revisit the use-case 'Advertised ancillary services' which has the objective to approach travelers with personalized ancillary services such as extra luggage, airport parking, seat selection, etc. through unsolicited mail or via push-up notifications on a mobile device.

Earlier, in section 4.2, we demonstrated through extensive experiments the benefit of using ML algorithms for improving ancillary services recommendation to travelers. However, one major limitation was raised: Data augmentation through feature engineering is very costly, not only in terms of time because it requires an important time of reflection and a participation of functional experts of the domain, but also in terms of memory where the features computed in the dataset must be stored and added in the database. Hence, in this section we develop an embedding-based KG recommender system that first computes knowledge graph embeddings from the 'Airline Travel KG' with the objective to replace handcrafted features by KG embeddings as input of XGBoost algorithm. We conduct extensive experiments to compare our approach with the currently in-production system and the ML algorithms presented in section 4.2. The results suggest that the use of KG embeddings is the most effective approach.

Inspired by recent works that have illustrated the effectiveness of using KG embeddings [113, 115, 134] for item recommendation, we propose **T**ravel **K**nowledge Graph **E**mbeddings for email marketing campaigns (TKE) framework to better target the audience for a service the airline wishes to recommend through email marketing campaigns (see figure 5.1). More especially, in [114], the authors propose to use property-specific KG embeddings generated from node2vec algorithm [48] in order to compute relatedness scores between items and users. Similarly, we propose to use translational distance and semantic matching models to generate KG embeddings and use them as latent features of a XGBoost algorithm.

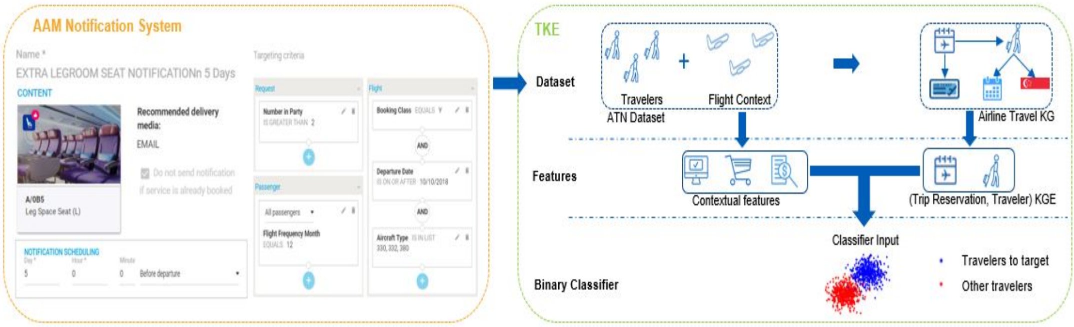

Figure 5.1 – On the left side: AAM Notification System. On the right side: Flowchart of our proposed TKE framework. Notification dataset used in this study is generated from the AAM Notification system. Contextual features include booking context (e.g. number of passengers, date of departure, etc.), notification information (e.g. media used to send the notification, time of notification, etc.).

5.2.1 Problem Formulation

To address the research question *RQ3.1*, we formulate the following problem: Given a notification campaign aimed at a large audience of travelers who have already booked a flight in a given context, we aim to use KG embeddings as input of XGBoost algorithm in order to target the relevant travelers among all the travelers that the notifications will reach.

The probability of recommending a given ancillary a in a notification N is revisited and is defined as follows for what remains:

$$P(purchase = a|N) = P(purchase|Context, TE, RE) \tag{5.1}$$

where, TE and RE are the traveler and Trip reservation embeddings.

5.2.2 Knowledge Graph

The knowledge graph used to tackle this use-case contains 41 different properties as shown in figure 5.2, ~ 80 million edges and ~ 9 million nodes.

For each notification campaign (see table 4.8), we extract a sub-graph from the Airline Travel KG that contains only information linked to the notification campaign. We present some statistics of these sub-graphs in table 5.6.

In figure 5.3, an excerpt of the KG is depicted, where a Malaysian traveler identified by T21354, born on "1988-05-05" has booked a one way flight for two people from Kuala lumpur to Melbourne. The EMD ticket identified by 23143 and linked to the air ticket 21563 represents

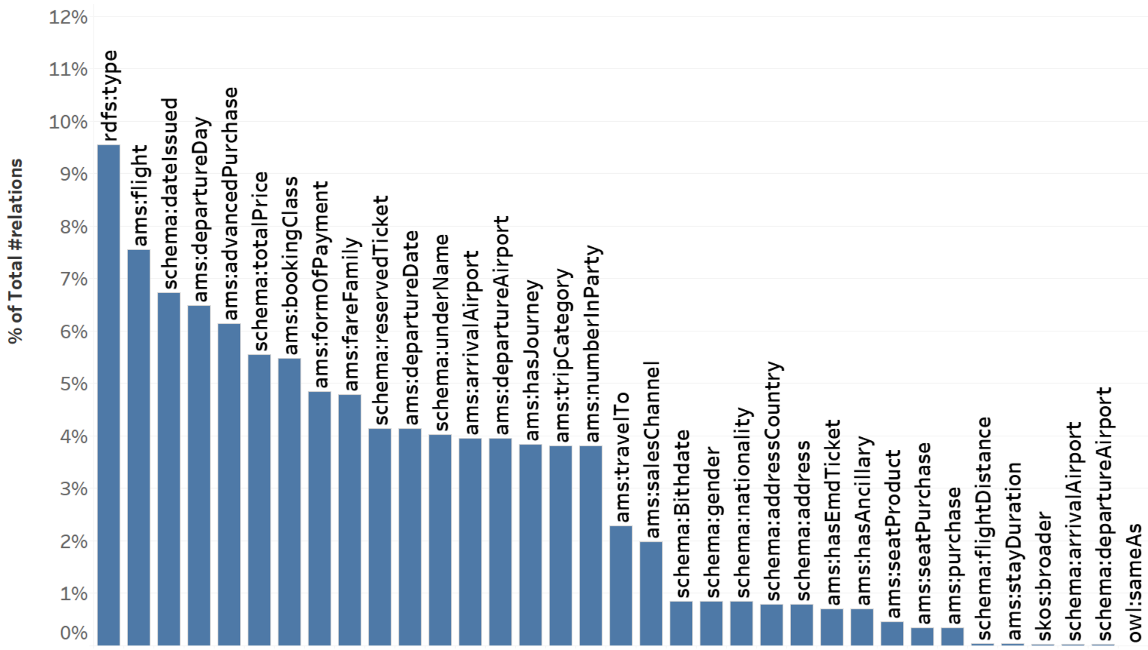

Figure 5.2 – Distribution of #relations of properties in the Airline Travel KG. All prefixes can be found in the ontology definition.

Table 5.6 – Statistics of subgraphs

Subgraph	#Edges	#Nodes	#travelers	#PNRs
Extra leg room seat	7M	800K	67K	205K
Prepaid baggage	64M	7.6M	572K	2.2M
Lounge	6.7M	789K	42K	203K

the purchase of an ancillary (a preferred seat).

5.2.3 TKE4Rec: Travel Knowledge Graph Embeddings for Recommendation

Our proposed framework TKE can be seen as a two-stage approach as presented in figure 5.1. In the first stage, we extract contextual features from the ATN dataset and compute KG embeddings of travelers and trip reservations from the Airline Travel KG. In the second stage, contextual features and KG embeddings are used as input of an XGBoost classifier in order to predict, for a given user, whether the notification should be sent or not. We use KG embeddings as latent features representation of travelers and trip reservations computed based on KG embedding algorithms such as TransE [12].

More formally, we use translational distance models to compute travelers and trip reservations embeddings as shown in figure 5.1. More formally, we learn the KG embeddings based on a

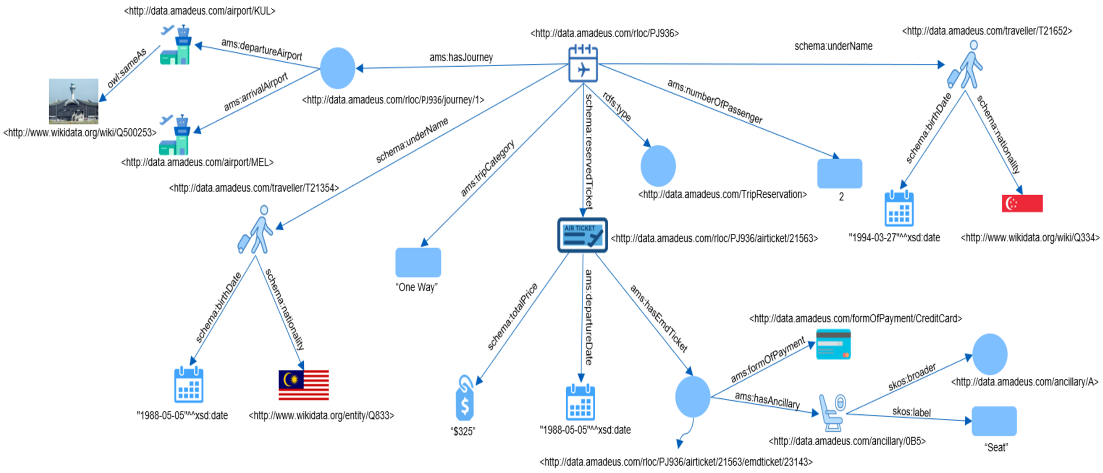

Figure 5.3 – Excerpt of the knowledge graph representing the travelers included in a Trip reservation through the property `schema:underName`, as well as other properties and relations to other entities. Literals are represented in blue rectangle, whereas other entities are represented in blue circle. In this depiction, some properties that links travelers, trip reservations, air tickets and EMD tickets are represented as an example, but more properties are included in the graph.

link prediction task, where some links of ancillary purchases and seat products are hidden in the training set, and put in the test set. Translational distance models are trained under the closed world assumption [148] using a pairwise loss that penalizes negative instances. More concretely, ancillaries that were not purchased by a traveler are considered as negative instances under the closed world assumption. Translational distance models are evaluated using ranking metrics such as hit rate or mean reciprocal rank. Hence, these models will return a high similarity score (low euclidean distance) for the ancillaries that are close in the graph embedding space to the embeddings of the ancillaries historically purchased by the travelers. As an example, we obtain a hit rate of ~ 0.42 with the TransE algorithm on the Airline Travel KG. In addition to translational distance models, we implemented a single-hidden MLP as proposed in [35] (ER-MLP), where each relation (as well as entity) is associated with a single vector. More specifically, given a fact (h, r, t), the vector embeddings of h, r, and t are concatenated in the input layer, and mapped to a non-linear hidden layer. The score is then generated by a linear output layer. The generated embeddings are used as input of XGBoost classifier in addition to the contextual features as shown in figure 5.1. We carry out a thorough empirical comparison of the aforementioned KG embedding algorithms and select the KG embeddings that allow the classifier to predict with the highest accuracy.

5.2.4 Experimental Setup

The objective of the experiments is to compare the use of handcrafted features (a) with the use of KG embeddings (b). (a) helps in interpreting the results and predictions obtained by the algorithm, while (b) lacks interpretation (latent features), but is easier to compute and maintain. We publish our code as open source in order to ease reproducibility[9].

Dataset: We experiment both approaches (a) and (b) with the three datasets presented in table 4.8. We use the Airline Travel KG presented in section 5.2.2 to generate the KG embeddings useful for our main approach TKE.

Training & Test Sets: We use the same setting presented in section 4.2.4 as evaluation protocol. For Knowledge graph-based algorithms, as described in [149], KG embedding algorithms are often designed to solve a link prediction task. We consider appropriate to split the KG by removing some edges that are included in the set of properties that link travelers with ancillaries and consider them as test sets, in order to evaluate the quality of the computed embeddings.

Evaluation metrics: We use exactly the same evaluation metrics presented in section 4.2.4.

Implementation Framework & Parameter Settings: For KG embedding algorithms, we use the deep learning framework pytorch[10] to implement ER-MLP [35] and the library pykg2vec [165] for all the other KG embedding algorithms. The hyper-parameters of all the models were tuned using a combination of random-search and grid-search algorithms. We apply grid-search algorithm on the implemented algorithms using the following values: the embedding size $k \in \{32, 64, , 96, 128, 256\}$, the batch size $\in \{128, 256, 512, 1024\}$, the number of epochs $\in \{50, 100, 200\}$, the learning rate $lr \in \{0.001, 0.003, 0.01, 0.03, 0.1, 0.3\}$ and negative samples Ns $\in [2, 10]$ for MLP algorithm.

5.2.5 Results

We present the results of the conducted experiments in table 5.7.

We observe in table 5.7 that using KG embeddings (concatenation of traveler and reservation KG embeddings) with contextual features as input of XGBoost performs better than using travelers handcrafted features regardless of the notification campaign and the KG embedding algorithm used to compute the embeddings. Moreover, KG embeddings computed from **ER-MLP** shows to perform better than KG embeddings computed from translational distance models except for the lounge notification campaign, where the use of KG embeddings com-

[9]https://gitlab.eurecom.fr/amadeus/tke4rec
[10]Pytorch:https://pytorch.org/

Table 5.7 – Evaluation results of the different approaches. (a) represents the results of XGBoost for different inputs; (b) represents the results of the TKE approach for different KG embedding algorithms. The average standard deviation (by varying the seed when splitting the dataset) of each metric is as follows: $AUC - ROC: \pm 0.02$, $TPR: \pm 3\%$, $TNR: \pm 2\%$, $CR: \pm 0.1\%$

Features	Extra leg room seat				Prepaid baggage				Lounge			
	AUC-ROC	TPR	TNR	CR	AUC-ROC	TPR	TNR	CR	AUC-ROC	TPR	TNR	CR
(a) C	0.75	78%	58%	2.2%	0.83	80%	71%	0.38%	0.76	80%	62%	0.18%
(a) T	0.79	81%	60%	2.37%	0.85	82%	74%	0.4%	0.84	86%	67%	0.22%
(a) C+T	0.83	85%	65%	2.8%	0.88	86%	**74**%	0.56%	0.89	88%	65%	0.36%
(b) TransE	0.85	86%	69%	3.1%	0.91	92%	65%	0.6%	0.90	89%	78%	0.35%
(b) TransH	0.84	85%	67%	3%	0.90	91%	65%	0.59%	**0.95**	**96%**	**85%**	**0.59**%
(b) TransR	0.84	85%	67%	2.9%	0.90	91%	65%	0.6%	0.92	92%	80%	0.52%
(b) ER-MLP	**0.87**	**88%**	**69%**	**3.2%**	**0.92**	**94%**	65%	**0.62%**	0.91	90%	81%	0.56%

puted from **TransH** model gives the best results. In table 5.8, we present the values of the hyper-parameters that lead to the best results given in table 5.7.

Table 5.8 – Best performing hyper-parameters and model for our knowledge graph embedding approach.

Notification Campaign	Model	k	lr	Ns	batch size	epochs
Extra leg room seat	ER-MLP	128	0.003	4	512	100
Prepaid baggage	ER-MLP	128	0.001	4	512	50
Lounge	TransH	96	0.03	-	128	50

5.2.6 Summary

In this section, we revisit the use-case of ancillary services recommendation through email marketing campaigns by using knowledge graph and KG embeddings instead of tabular data and handcrafted features. We have developed a two stage approach **TKE** (see figure 5.1) to address this use-case: first, we compute KG embeddings of travelers and trip reservations; second, we use these embeddings in addition to contextual features as input of an XGBoost classifier to learn what is the relevant audience to target for a given notification campaign. We conduct several experiments to address the research question ***RQ3.1***: Experiments show that regardless of the KG embedding algorithm tested, the KG embedding approach is better than the handcrafted features approach. This is very interesting from a scientific point of view, as it shows the added value of having a KG in the travel domain that could be used not only for ancillary recommendation task, but also other recommendation tasks (e.g. Trip recommendation as shown in section 5.3). It is worth noticing that when dealing with a cold-start problem (new user or item) for on-line usability, a rule-based approach is more appropriate.

5.3 Next Trip Recommendation

In this section we revisit the use-case of 'Next Trip Recommendation' by using KG recommender systems instead of traditional hybrid recommender systems as presented in section 4.1.
As already presented in section 4.1, several factors influence a user's decision when faced with a variety of travel destination choices: geographic context, best time to go, personal experiences, places to visit, scheduled events, etc. We think that the challenge of recommending the right travel destination lies in efficiently integrating and leveraging all of this information into the recommender system. In this section, we try to show the benefit of using knowledge graph as a means of representing all the heterogeneous information used for the recommendation task by evaluating experimentally our proposed knowledge graph-based recommender systems by comparing it against the currently in-production system and hybrid recommender systems in an offline setting.

The use of *CF* methods for travel destination recommendation suffers from the cold start problem and data sparsity [29]. Indeed using only travelers' historical bookings as input information of the recommender system may not be sufficient. Therefore, incorporating additional information such as travel context, traveler demographics, or destination metadata into the recommender system could be valuable in addressing the above-mentioned issues. To integrate these heterogeneous information into a single data structure, the knowledge graph is an appropriate candidate to consider. Indeed, recent works [113, 115, 134] have illustrated the effectiveness of using knowledge graph embeddings for items recommendation. However, as pointed out in [44], not all knowledge graph embedding algorithms are effective in combining different types of literals and most of them do not have a proper mechanism to handle multi-valued literals (text, image, numerical value, etc.). Inspired by the work proposed in [141], where the authors propose an approach for both relational learning and non-discrete attribute prediction on knowledge graphs, we propose **K**nowledge **G**raph-based **M**ulti **Task L**earning For **Rec**ommendation (KGMTL4Rec[11]), a neural network-based multi-task learning algorithm for travel destination recommendation that leverages knowledge graph[12] information. We present the model architecture in figure 5.4.

[11]https://gitlab.eurecom.fr/amadeus/KGMTL4Rec
[12]https://gitlab.eurecom.fr/amadeus/KGMTL4Rec/-/blob/master/ontology/ontology.ttl

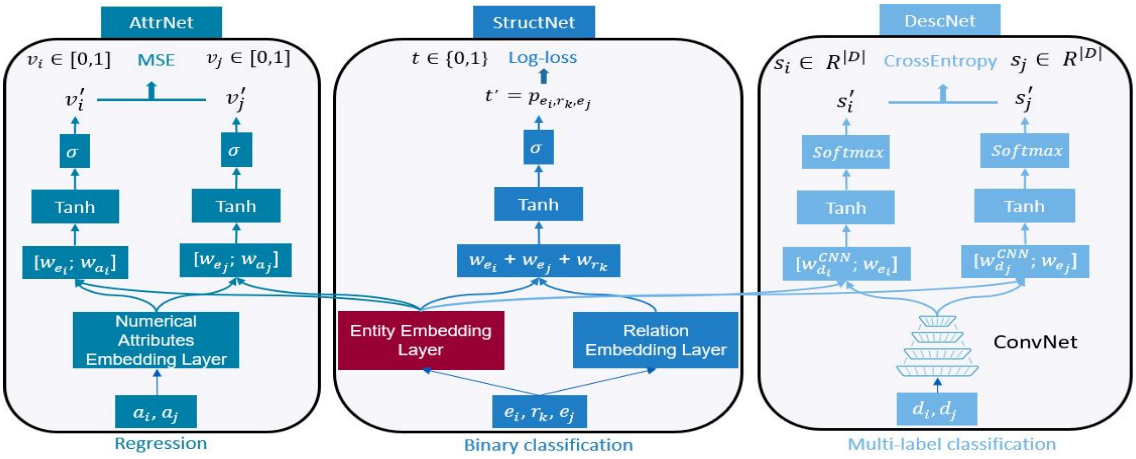

Figure 5.4 – KGMTL4Rec Architecture: A neural network composed of three sub-networks, each sub-network being specialized in a learning task. The same color is used for different elements of a sub-network (e.g. Turquoise color for AttrNet). Red color is assigned to the 'Entity Embedding Layer' as its weights are shared across the different sub-networks.

5.3.1 Related Work on Multi-Task Learning for Recommendation

Some research work focused on integrating MTL algorithms with traditional CF models such as matrix or tensor factorization [92, 148] in order to generate explainable recommendations. However, these factorization-based models cannot fully exploit the information available in the knowledge graph. In [95], the authors proposed a learning framework composed of two auxiliary tasks (click-through rate and conversion rate optimization) to deal with the extreme data sparsity problem of conversion rate optimization. In [51], the authors proposed a MTL framework to learn simultaneously parameters of two recommendation tasks namely ranking task and rating task. In [8], to deal with the sparsity of the interaction matrix, the authors used MTL to train the model for a combination of content recommendation and item metadata prediction. Similarly to these previous works, we use a neural network with shared parameters learned through different tasks as model architecture. In [147], the authors propose a neural network-based MTL algorithm to predict not only user-item interactions but also missing links in a knowledge graph. Similarly, in [158], the authors mixes a relational modeling algorithm with a recommendation one in a MTL fashion based on a neural network. Nevertheless, the models proposed in the two above-mentioned works do not incorporates literals, thus missing a valuable opportunity for data enrichment. In the opposite, KGMTL4Rec takes into account several types of inputs which constitutes its main strength in comparison with existing MTL algorithms for recommendation.

In the previous chapter, we presented DKFM a hybrid recommender systems that make use of numerous data (collaborative data, content and contextual information, external data enrichment), however, not all the data used as input of DKFM model (see figure 4.4) comes from a single data structure. This represents the major difference between the work carried out in this section and the one presented in section 4.1.

The dataset released for this challenge is completely anomymized. Hence, it cannot be used in our work since destinations (referenced by ids) are unknown. To the best of our knowledge, there is no public available dataset that addresses the task of travel destination recommendation that can benefit from the type of data augmentation we are proposing in this work. We describe the experimental dataset we use later in section 5.3.3.

5.3.2 Problem Formulation

In this section, we focus on recommending not only next travel destination to travelers but also **new** travel destinations. Hence the objective is to provide leisure travelers with travel destinations that they have never visited yet. We consider past bookings of travelers, booking contexts and travelers' and destinations' metadata as information to be used in our recommender system. These information are collected and stored in the airline travel knowledge graph described in section 5.1. The task of recommending the next travel destination to a traveler is formulated as a link prediction task in a knowledge graph. We address the following questions that derives from the problem formulation:

1. What is the benefit of using a knowledge graph as a unique data structure containing all the input information of the recommender system?
2. Given the heterogeneous nature of the information included in the knowledge graph (numerical values, dates, texts, etc.), what is the best performing approach for travel destination recommendation?

5.3.3 Knowledge Graph

We extract a sample from the knowledge graph constructed. The sample contains 486.000 bookings from November 2018 to December 2019, made by 40.965 unique travelers and covering 136 different destinations.

A destination where a traveler traveled to is described by a property which we name `travelTo`. The objective of the recommender system is to predict the correct links labeled by the property `travelTo` between travelers and destinations.

In addition to this Airline Travel KG, we make use of the property `owl:sameas` to enrich the knowledge graph with destinations metadata. In practice, we re-use the Wikidata[13] knowledge graph, the Semantic Trails Dataset (STD) knowledge graph [103] and Wikipedia textual description of the travel destinations to populate our original airline travel KG. In the end, the KG used to tackle our recommendation task contains 48 different properties, ~ 13.7 million edges (~ 634.000 nodes) of which ~ 11.9 Millions come from the Original Airline Travel KG (32 Properties about PNRs, travelers' information, etc.), ~ 1.7 Millions from the STD knowledge graph (5 properties) and ~ 100K from Wikidata (11 properties) and finally ~ 486K edges are travel interactions (property `travelTo`).

In figure 5.5, an excerpt of the KG is depicted, where a Singaporean traveler, born on "1994-03-27" booked a one-way flight from Kuala Lumpur to Melbourne (the property 'travelto' coming from the traveler points at Melbourne airport).

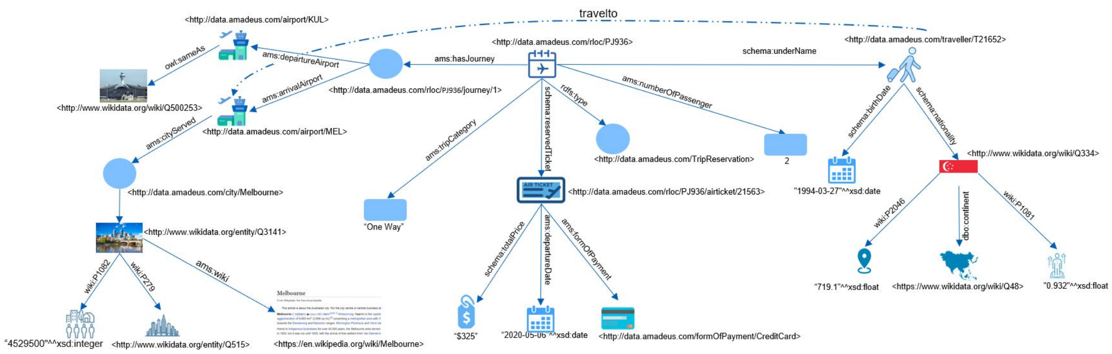

Figure 5.5 – Excerpt of the knowledge graph representing a traveler included in a Trip reservation through the property `schema:underName`, as well as other properties and relations to other entities. Literals are represented in blue rectangle, whereas other entities are represented in blue circle. In this depiction, some properties which links travelers, trip reservations, air tickets, travel destinations are represented as an example, but more properties are included in the graph.

5.3.4 KGMTL4Rec: Knowledge Graph-based Multi-Task Learning for Recommendation

As mentioned at the beginning of this section, MT-KGNN [141] has recently proven to be an effective approach to deal with non-discrete values in knowledge graphs for representation learning. The authors proposed a multi-objective neural network model trained using a multi-task learning algorithm that includes two regression tasks to predict numerical attributes of KG entities and one classification task to predict when a triplet (head, relation, tail) holds in the KG. In our work, we propose to extend MT-KGNN model by adding a sub-network called DescNet (see figure 5.4) that predicts the correct entity described by a textual description given as input

[13] https://www.wikidata.org/

of DescNet. Inspired by DKRL model proposed in [157], we decide to use a convolutional neural network to reduce the dimension of word vectors of the textual descriptions and train DescNet sub-network along with two other sub-networks (StructNet & AttrNet). We present the model architecture of KGMTL4Rec in figure 5.4. We describe below the different learning tasks and present the multi-task learning algorithm used to train KGMTL4Rec.

Structural Learning (StructNet): The first learning task of KGMTL4Rec corresponds to a binary classification task which is used to model the structural aspect of the knowledge graph. Each element of the input triplet (e_i, r_k, e_j) of StructNet is first passed into an embedding lookup layer, then the embeddings $(w_{e_i}, w_{r_k}, w_{e_j}) \in R^d$ are summed and passed into a *hyperbolic tangent* (tanh) nonlinear layer. Finally, a sigmoid linear layer is added to compute the probability $p_{e_i,r_k,e_j} = P((e_i, r_k, e_j) \in T_r)$, where T_r is the set of existing triples in the knowledge graph. More formally, the probability p_{e_i,r_k,e_j} is computed as follows:

$$p_{e_i,r_k,e_j} = g_{StructNet}(e_i, r_k, e_j) = \sigma(\vec{v}_h^s \ tanh(\mathbf{V}_{h,d}^s \ (w_{e_i} + w_{r_k} + w_{e_j}) + b_h^s) \tag{5.2}$$

where $\mathbf{V}_{h,d}^s \in R^{h\times d}$ and $\vec{v}_h^s \in R^h$ are parameters of StructNet and b_h^s is the scalar bias of the hidden layer, h being the size of the hidden layer. We use logistic loss as loss function for this binary classification task. It is important to note that unlike ER-MLP [35], in StructNet we compute the sum of $w_{e_i}, w_{r_k}, w_{e_j}$ embeddings instead of concatenating them, as it has shown better performance in the experiments.

Numerical Attribute Learning (AttrNet): The second learning task of KGMTL4Rec is a regression task, where the objective is to predict the correct numerical value of an entity attribute (e.g. the price of an air ticket). AttrNet takes as input the attributes a_i and a_j linked to e_i and e_j entities. The embedding $w_{a_i} \in R^m$ is concatenated with w_{e_i} and $w_{a_j} \in R^m$ with w_{e_j}, then the concatenated vectors are passed into a tanh nonlinear hidden layer and finally passed into a sigmoid linear layer to compute the estimated numerical values $v_i^{'}$ and $v_j^{'}$. More formally, the estimated value $v_i^{'}$ is computed as follows:

$$v_i^{'} = g_{AttrNet}(e_i, a_i) = \sigma(\vec{v}_h^a \ tanh(\mathbf{V}_{h,md}^a \ [w_{e_i}; w_{a_i}] + b_h^a) \tag{5.3}$$

where $\mathbf{V}_{h,md}^a \in R^{h\times(m+d)}$ and $\vec{v}_h^a \in R^h$ are parameters of AttrNet and b_h^a is the scalar bias of the hidden layer.

Mean squared error (MSE) is used as a loss function for AttrNet. Unlike what was done in MT-KGNN [141], we use only one single AttrNet regardless if an attribute is linked to the tail or the head entity of a triplet.

Text description Learning (DescNet): The third learning task of KGMTL4Rec is a multi-label classification task, where the objective is to predict the correct entities e_i and e_j described by the input text descriptions d_i and d_j. The first part of DescNet is a convolutional neural network (CNN) composed of one convolutional layer and a max-pooling layer used to reduce the dimension of input word vectors. Similarly to what is done in [29], we assign to each word of the text description d_i and d_j a weighted tf-idf pre-trained word vector from fasttext [46]. the CNN is then fed with $w_{d_i} \in R^{|d_i| \times k}$ and $w_{d_j} \in R^{|d_j| \times k}$, vector representations of d_i and d_j, where $|d_i|$ and $|d_j|$ represent the length of the text descriptions d_i and d_j and k the dimension of word vectors. Finally, the output vectors of the CNN ($w_{d_i}^{CNN}, w_{d_j}^{CNN}$) are passed into a tanh nonlinear hidden layer, then passed into a Softmax linear layer to compute the estimated vectors $s_i^{'}$ and $s_j^{'} \in R^{|D|}$, D being the set of travel destinations. More formally, $s_i^{'}$ is computed as follows:

$$s_i^{'} = g_{DescNet}(d_i) = Softmax(\vec{v}_h^d \ \ tanh(\mathbf{V}_{h,k}^d \ \ w_{d_i}^{CNN}) + b_h^d) \tag{5.4}$$

where $\mathbf{V}_{h,k}^d \in R^{h \times k}$ and $\vec{v}_h^d \in R^h$ are parameters of DescNet and b_h^d is the scalar bias of the hidden layer.

Note that the learning task is performed twice for the head and the tail entity of the input triplet (e_i, r_k, e_j) for each of the learning tasks in AttrNet and DescNet.

Multi-task learning algorithm: We adopt an alternating learning strategy for the five learning tasks. More formally, for each epoch, we run the following:

- Sample mini-batch of positive and negative triples (e_i, r_k, e_j) from the knowledge graph, train StructNet and update KGMTL4Rec parameters by back-propagation according to Eq 5.2.
- Sample mini-batch of numerical attributes a_i and a_j and their corresponding numerical values v_i and v_j, train AttrNet and update KGMTL4Rec parameters by back-propagation according to Eq 5.3.
- Sample mini-batch of textual descriptions d_i and d_j of e_i and e_j entities, train DescNet and update KGMTL4Rec parameters by back-propagation according to Eq 5.4.

In the experiments, we compare the alternating learning strategy with the weighting loss strategy [20, 159] where the different losses of the sub-networks are summed so that the sum of the losses is back-propagated through KGMTL4Rec.

Recommendation scoring function As mentioned in section 5.3.2, the task of recommending destinations to travelers is formulated as a link prediction task in the knowledge graph. Therefore, in order to compute the probability of recommending a destination e_d to a traveler e_t, we use StructNet sub-network and compute the score of the triplet (e_t, 'travelto', e_d) comprising the traveler e_t, the destination e_d, and the property 'travelto'. The recommendation scoring function is defined as follows:

$$f_{recommendation}(e_t, e_d) = g_{StructNet}(e_t, travelto, e_d) \tag{5.5}$$

5.3.5 Experimental Setup

In this section, we present the dataset used to conduct our experiments, then we present the baseline models implemented to compare our model with and the settings of the experiments. Finally, we present and discuss the results obtained in the experiments.

Dataset For the experiments, we use the private dataset described in Section 5.3.3. It is important to note that due to the specificity of our recommendation task 'recommending **new** travel destinations for **leisure** purpose', the amount of data used in the experiments is significantly reduced. Indeed, The original dataset used to build the knowledge graph comes from a major partner airline and counts more than 10 million bookings in one calendar year. In this work, we focus only on leisure trips, which corresponds to approximately 56% of the bookings similarly to what has been done in [29]. Furthermore, the dataset that is used to train the recommender system is reduced as we only consider travelers who have made at least two bookings (for evaluation purposes), resulting in 486.807 travel interactions. The characteristics of the dataset are summarized in table 5.9.

Table 5.9 – Statistics of the experimental dataset.

#travels	#travelers	#destinations	Sparsity ρ
486 807	40 965	136	91.26%

In figure 5.6, we plot an histogram that represents the number of visits per travel destination as a percentage of the total number of visits (#travels) for the top-10 most visited destinations. This histogram shows the high popularity of certain travel destinations which is accounted for in the experiments by comparing the performance of our model with the system currently in production which some airline partners use and that recommends this top-10 list of popular destinations regardless of the traveler. In figure 5.7, we plot an histogram representing the number of travelers (as a percentage of total number of travelers) per historical travels. In the experiments, we compare the performance of our model with respect to the number of historical travels per traveler.

Evaluation protocol Widely used in the literature [55, 122], and more importantly adopted in [29], the *leave-one-out* protocol suggests to select the latest interaction as the test set and use the remaining data in the training/validation set. We use this protocol to evaluate the performance of KGMTL4Rec and also to compare it with the different baseline models.

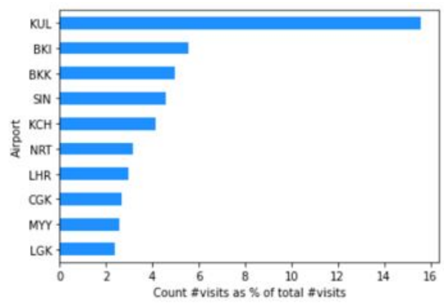

Figure 5.6 – Top-10 Most visited travel destinations (airports).
Each Airport its IATA Code.

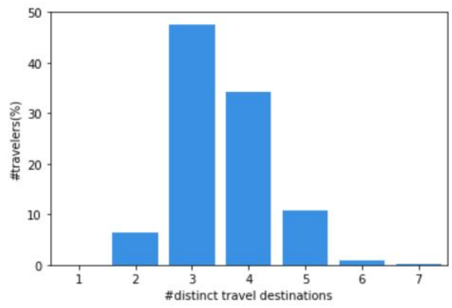

Figure 5.7 – Histogram showing the number of travelers per number of distinct historical travel destinations.

Our dataset is temporally sorted so that the latest travel corresponds to the most recent destination visited by a traveler, in order to represent the notion of recommending the 'next' travel destination during evaluation. For each traveler, we rank all destinations except the ones that are already visited by the traveler and truncate the list at 10, as 10 destinations are included in the email sent to the travelers. To validate our model, we apply a cross-fold validation to the training dataset (k=5, a split of 80% for training and 20% for validation). The split between training and validation set is performed randomly on travels in order to avoid a seasonality effect which is usually occurring in the travel industry.

Baseline models and parameter settings

We implement a wide list of baseline models to compare our model KGMTL4Rec with. More specifically, the baseline models include CF, *context-aware*, *hybrid* and knowledge graph-based recommender systems. Following the experimental work conducted in [29], this represents the state-of-the-art recommender systems for travel destination recommendation. We describe the main baseline models implemented:

- **BPRMF [122]:** *BPRMF* is a *Matrix Factorization* method tailored for implicit feedback where the authors propose to minimize a pairwise ranking loss rather than minimizing a mean squared error between the predicted and the observed 'rating' as usually done in Matrix Factorization algorithm.
- **NCF [55]:** *Neural Collaborative Filtering* is a state-of-the-art CF method. It combines the (user, item) interaction as input of a multi-layer perceptron and a single layer perceptron that models the matrix factorization method.
- **FM [123]:** *Factorization Machines* was proposed to incorporate contextual information in the recommender system. The author propose a method that computes not only users' and items' latent vectors but also contextual features latent vectors.
- **WDL [21]:** *Wide & Deep Learning* model is a hybrid recommender system. It is a deep learning based recommender system that combines a deep component (feed forward

neural network) plus a wide component that can be seen as a linear model that computes cross products between input features.

- **DKFM [29]:** *Deep Knowledge Factorization Machines* combines Factorization Machines in order to represent contextual information and WDL that takes as input user-item interactions and metadata information about the items and users.
- **NTN [131]:** *Neural Tensor Network* is a neural network based method for representation learning in knowledge graphs [149]. Given a fact (h, r, t), it first projects entities to their vector embeddings in the input layer and then predicts the existence of this fact in the knowledge graph. Similarly to StructNet (see section 5.3.4), we rank destinations based on NTN output score.
- **TransE [12]:** *TransE* is the most used translational distance model [149]. Given a fact (h, r, t), the relation is interpreted as a translation vector r so that the embedded entities h and t can be connected by r with low error, i.e., $h + r \approx t$ when (h, r, t) holds. Similarly to [115], we use TransE scoring function $f_r(h, t) = -||h + r - t||$ to produce the ranked list of destinations.
- **CKE [168]** *Collaborative Knowledge base Embedding* is a two stages approach that consists in first computing the embeddings coming from a knowledge base composed of structural knowledge, image and text representing the items, then use the generated embeddings as input of a CF algorithm. In this work, we implement the structural and textual modules in addition to the CF algorithm.

We implement our model KGMTL4Rec using Pytorch[14] as it provides us more easiness for the implementation of new neural network architectures and use Pykg2vec[15] library for knowledge graph-based models, finally we use Tensorflow[16] to implement the neural network baseline models. We use Xavier uniform initializer to randomly initialize the models parameters and we use a mini-batch optimization technique based on Adam [75] optimizer to train all the models. To tune the hyper-parameters of our model and the baseline models, we use the validation set mentioned above. We apply grid-search algorithm on the implemented models using the following values: the entity embedding size $d \in \{16,32,64,128,256\}$, the batch size $\in \{128,256,512,1024\}$, the number of epochs $\in \{10,20,50,100,200\}$, the learning rate $\lambda \in \{0.00001,0.0001,0.0003,0.001,0.003,0.01,0.1\}$ and negative samples Ns $\in [2,10]$.

5.3.6 Results

In table 5.10, we present the recommendation performance of KGMTL4Rec and the baseline models with respect to HR@10 and MRR@10. The results reported in table 5.10 correspond to

[14] https://pytorch.org/
[15] https://pykg2vec.readthedocs.io/
[16] https://www.tensorflow.org/

the performance of the different models based on the best performing hyper-parameters. We report the mean and standard deviation of HR@10 and MRR@10 over 5 different seeds due to the random initialization of neural networks parameters.

Table 5.10 – Experimental results.

(a) Recommendation performance of CF, hybrid and CA recommender systems.

(b) Recommendation performance of KG recommender systems.

Model	HR@10	MRR@10	Model	HR@10	MRR@10
Item-pop	0.5168	0.2634	NTN [131]	0.3096 ± 0.002	0.1511 ± 0.001
IKNN [128]	0.3223	0.1367	SME [11]	0.3746 ± 0.001	0.1992 ± 0.0004
BPRMF [122]	0.5698 ± 0.002	0.3036 ± 0.0004	TransE [12]	0.4548 ± 0.0005	0.2268 ± 0.0001
NCF [55]	0.5132 ± 0.008	0.2994 ± 0.0010	TransR [86]	0.4031 ± 0.0009	0.1883 ± 0.0001
FM [123]	0.5986 ± 0.003	0.3401 ± 0.0001	ER-MLP [35]	0.6218 ± 0.002	0.3559 ± 0.0028
WDL [21]	0.6301 ± 0.005	0.3472 ± 0.0003	CKE [168]	0.6493 ± 0.003	0.3865 ± 0.001
DKFM [29]	**0.6619** ± 0.007	**0.3901** ± 0.0006	**KGMTL4Rec**	**0.7109** ± 0.013	**0.4254** ± 0.0083

It is important to note that not all recommender systems use the same input information. In fact, recommender systems which use not only traveler history but also other types of information as input such as DKFM or WDL tend to perform better than simple Collaborative Filtering models such as ImplicitMF, NCF or IKNN as shown in sub-table (a). Similarly to DKFM, knowledge graph-based recommender systems represented in sub-table (b) make use of all the information mentioned in section 5.3.2. It is therefore legitimate to compare KGMTL4Rec with DKFM, where we clearly observe that KGMTL4Rec performs better with respect to HR@10 and MRR@10. KGMTL4Rec is not only outperforming DKFM model but also the other knowledge graph-based recommender systems represented in sub-table (b). The major difference between KGMTL4Rec and the other knowledge graph-based recommender systems, is that KGMTL4Rec uses each type of information optimally in one of the sub-networks defined in section 5.3.4, while models like TransE, NTN or even CKE (that uses TransE to generate structural embeddings) consider numerical values as a separate entity, which not only increases considerably the cardinality of entities set considered in this type of method, but also considers equal numerical values as the same entity: it is not correct to consider 12 'years old' and 12 'days' as the same entity.

In what follows, we take an excerpt (~20%) from the original knowledge graph described in table 5.9 in order to conduct additional experiments. All the results that follow are based on this excerpt (Table 5.11).

In table 5.12, we report the performance of our model compared to the best performing models when we use different types of input information, so the knowledge graph is reduced to keep only the information needed in each experiment to be fairly comparable to other

Table 5.11 – Statistics of the sample knowledge graph.

#Nodes	#Edges	#Properties	#Trip Reservations
125 610	~ 2.7 M	48	35698

models:

Table 5.12 – Performance of KGMTL4Rec compared to best performing models on specific type of input data. *: All Information mentioned in section 5.3.2

Input Data	Collaborative Information		Content & Collaborative Information		All Information*	
Model	BPRMF	**KGMTL4Rec**	WDL	**KGMTL4Rec**	DKFM	**KGMTL4Rec**
HR@10	0.5462	**0.5623**	0.6001	**0.6508**	0.6464	**0.6907**
MRR@10	0.3020	**0.3153**	0.3472	**0.4061**	0.3856	**0.4189**

Figure 5.8 shows the performance of the models represented in table 5.12 with respect to the number of iterations used to train the models. We use the same learning rate for all the models ($lr = 0.00003$) presented in figure 5.8. We observe, that the most effective updates are occurred in the first 3 iterations for all the models except for DKFM where the convergence requires more iterations. In addition, we notice that there is a significant difference of HR@10 and MRR@10 in the first iteration (iteration 0) for the different models. Moreover, it is important to note that for KGMTL4Rec, we do not get the best value of MRR@10 and HR@10 in the same iteration.

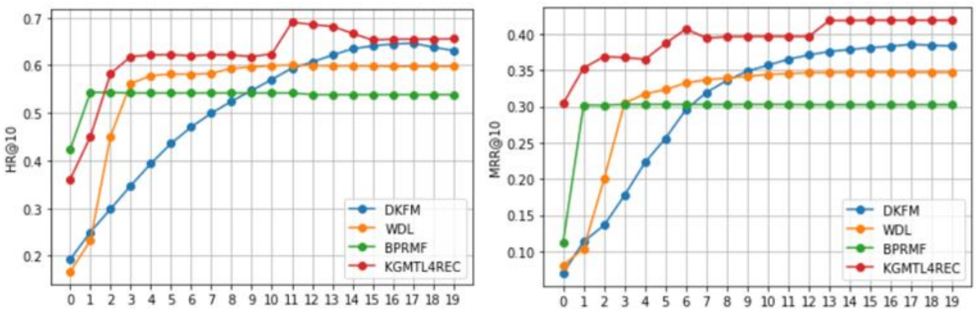

Figure 5.8 – Performance of the 4 main models (presented in table 5.12) with respect to the number of iterations.

In the remaining of this section, we discuss the analysis of KGMTL4Rec on different aspects. We first start by performing an ablation study which consists in removing some input information from the knowledge graph and using only some sub-networks of KGMTL4Rec. Then, we study the influence of the travel history of travelers (number of historical travels) on the performance of KGMTL4Rec. We observe the convergence time of KGMTL4Rec with respect to two different MTL strategies. Finally, we perform a qualitative analysis of KGMTL4Rec recommendations and investigate the impact of KGMTL4Rec hyper-parameters on the performance of the model.

Ablation Study:

Table 5.13 shows the performance of KGMTL4Rec with respect to the information included in the knowledge graph. In the first row of the table, we present the results of KGMTL4Rec when we consider neither the STD knowledge graph nor the textual information from Wikipedia, nor the numerical literals included in the Airline Travel KG (e.g., the number of passengers in a reservation, the ticket price, etc.), hence in this case, we use only the sub-network StructNet to train the model. Then, for each of the rows that follow, we add incrementally one of the preceding removed information. We observe that the results are the best when we use the most possible information in the KG, and notice that the large gap between the results is reduced when we consider the use of numerical literals.

Table 5.13 – Performance of KGMTL4Rec model based on the information contained in the knowledge graph.

Numerical literals	STD KG	Wikipedia	Sub-networks	HR@10	MRR@10
No	No	No	StructNet	0.5884	0.3264
Yes	No	No	StructNet, AttrNet	0.6508	0.4061
Yes	Yes	No	StructNet, AttrNet	0.6781	0.4119
Yes	Yes	Yes	StructNet, AttrNet, DescNet	**0.6907**	**0.4189**

Influence of travel history:

CF algorithms which rely only on users' past interactions perform naturally better when we have more history about the users. We study the performance of KGMTL4Rec model and DKFM model presented in table 5.12 with respect to the number of historical travels per traveler. More formally, we compute HR@10 and MRR@10 for travelers which traveled in N_{hist} different destinations in their past ($N_{hist} \in [1,5]$). We observe in figure 5.9 more variation of HR@10 and MRR@10, when we vary the number of historical travels for DKFM than for KGMTL4Rec. Indeed, the standard deviation of the different values of HR@10 for DKFM is equal to 2×10^{-2}, while for KGMTL4Rec it is equal to 5×10^{-3}. For MRR@10, the standard deviation is equal to 2.5×10^{-2} for DKFM, while for KGMTL4Rec it is equal to 6×10^{-3}. These results demonstrate that our model KGMTL4Rec is more resilient to variation of the traveler history than DKFM.

Multi-task learning strategy:

While in most multi task learning algorithms the back-propagation is performed based on the sum of the losses of the different tasks [20, 159], we decide to use another strategy which is to perform a back-propagation to update the weights of our model for each learning task, as we do not think judicious to share the same loss across the different sub-networks of KGMTL4Rec when updating the model parameters. We demonstrate in figure 5.10 that our learning strategy

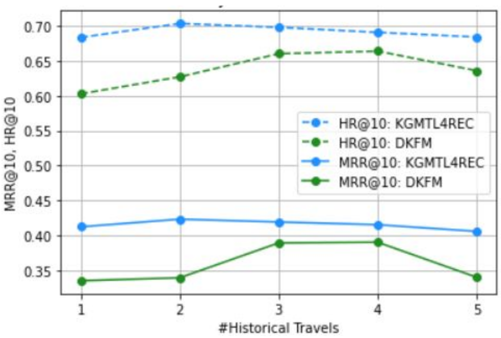

Figure 5.9 – Performance of KGMTL4Rec with respect to the number of Historical travels per traveler.

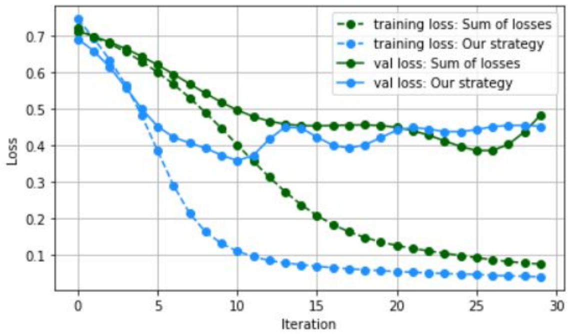

Figure 5.10 – Training and Validation loss with respect to the number of iterations for two different MTL strategies.

converges faster than the strategy used in [20, 159]. Indeed, we observe that the 'sum of losses' strategy needs 12 more iterations than our strategy for the training loss to be equal to 0.1. Moreover, for the validation loss, our strategy needs 10 iterations to converge to a value of 0.38 while for the 'sum of losses' strategy 13 more iterations are needed.

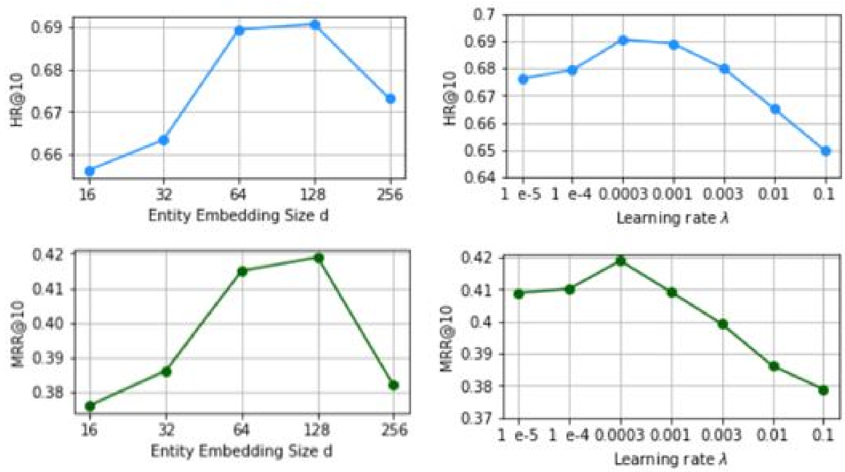

Figure 5.11 – Performance of KGMTL4Rec with respect to the entity embedding size and the learning rate λ.

Hyper-parameters Sensitivity:

We investigate the influence of some hyper-parameters on the performance of KGMTL4Rec. In figure 5.11, we report the score of HR@10 and MRR@10 when we vary the learning rate λ and the Entity Embedding size d as specified in section 5.3.5. We observe in figure 5.11 that when increasing d, the performance is initially improved because embeddings with larger size can encode more useful information, but drops after d = 128 due to possible overfitting. The same pattern is observed when varying λ, indeed the HR@10 and MRR@10 scores increase until $\lambda = 0.0003$ as the use of a higher λ does not allow to find the optimal loss.

5.3.7 Summary

In this section, we revisited the use-case of 'Next Trip Recommendation' by proposing a model that incorporates heterogeneous information from a multi-typed knowledge graph namely **KGMTL4Rec**, a multi-task learning algorithm designed to consider not only knowledge graph entities but also numerical and text literals in order to recommend personalized travel destinations to airlines' customers through email marketing campaigns. KGMTL4Rec is based on a neural network architecture which can incorporate different types of information available in the knowledge graph. We conducted several experiments to address the research question ***RQ3.2***: Our model is capable of predicting the missing links 'travelTo' in the knowledge graph with a HR@10 of ~ 0.69. Additionally, we demonstrated, through an in-depth comparison between KGMTL4Rec and DKFM (see section 4.1), the valuable contribution of using the knowledge graph as a unique structure to represent the heterogeneous information used for travel destination recommendation.

The results confirm the significant contribution of using knowledge graphs as a means of representing the heterogeneous information used for the recommendation task, as well as the benefit of using a multi-task learning model in terms of recommendation performance and training time.

In this section, we demonstrate the methodology of building recommendation using a knowledge graph to represent heterogeneous information, and a multi-task learning algorithm to make the most of this heterogeneous information through the multiple learning tasks (regression, binary-classification, multi-class classification). The outcome of the results demonstrate that even with such sparse data, adding qualitative data through the enrichment of the travel interactions can lead to better travel destination recommendations than using traditional hybrid recommender systems.

Chapter 6

Conclusion

Inspired by the new offer distribution flow (NDC) introduced by the IATA organization, we addressed a set of research challenges related to recommender systems applied to the airline travel industry in this thesis. The aim of NDC is to facilitate the ability of airlines to sell their products more easily by creating more personalized offers through complete control over the offer distribution flow which allows the personalization and contextualization of airline offers, thus creating a better travel experience for their customers.

However, the particular characteristics of the airline industry compared to other industries that are very mature in the use of recommender systems such as entertainment or e-commerce industries make the development of recommender systems in this field a real challenge and the reasons for that are manifold: First of all data is very sparse in the travel domain (see chapter 3), secondly due to the way the airline reservation system works (several different reservation platforms are possible without any user identification required) we have a lot of new travelers in the travel domain which is the *cold start* problem, thirdly the lack of ML application in the airline domain and especially the very small number of recommender systems developed in this domain make the collection of useful and necessary data in the development of recommender systems a difficulty in itself.

On the one hand, this leads us to look for a way to enrich and populate our data through other sources to overcome these two problems of sparsity and cold start. Semantic data are used for this purpose in order to semantically enrich our data and to bring a certain classification and a well-defined coherent structure on the logical and semantic level through the definition of an ontology. Incorporating semantic data and airlines' data (e.g. travel interactions, travel context, etc.) into one single data structure (Knowledge graph) have proven to be very valuable as a source of data for recommender system algorithms algorithms to perform prediction as we show in Chapter 5.

On the other hand, knowledge graph-based recommender systems have shown to be effective to deal with data sparsity and cold start problem as we show in Chapter 5. More specifically, KG benefit the recommendation from three main aspects: (1) KG incorporates heterogeneous information coming from different sources of data through the use of relations with various types, therefore improving data integration and data augmentation for machine learning usage and avoiding the heavy task of feature engineering necessary for improving recommender systems accuracy (see section 5.2); (2) KG introduces semantic relatedness among items, which can help find their latent connections and improve the precision of recommended items [146] (see section 5.3); (3) KG contains information about the entire traveler journey, from inspiration to flight departure, making it a unified resource to serve as input all the recommendation use-cases that span the entire traveler journey.(see section 5.1).

In the following we summarize the content of this thesis, reporting the main contributions from the obtained results. We will discuss the implications of the development of recommender systems on the personalization of the traveler experience. Finally, we will conclude by recapitulating the limitations of this work and suggesting some perspective for further research on these topics.

6.1 Summary

In this section, we will go over the research questions listed in the introduction of this thesis and provide some answers to them based on the results obtained in the previous sections.

In a first stage, this thesis contributes particularly to the personalization of airlines' offers covering the traveler journey:

- ***RQ1***: How can we propose personalized items (travel destinations, ancillary services, third party content) to travelers using recommender systems? (Chapter 4)

To address this research question, we developed hybrid recommender systems that makes use of numerous data ranging from travelers' interactions with the airline catalog to the purchase context of a product. In the remaining, we summarize the outcome of the work conducted to tackle the three recommendation use-cases presented in chapter 4.

Next Trip Recommendation

We developed DKFM[1], a neural network based algorithm designed to incorporate heterogeneous information to recommend travel destinations to past travelers. The results obtained allowed us to confirm the relevance of using ML algorithms to provide personalized recommendations of travel destinations. This use-case considered by many airlines in a large

[1]DKFM: https://gitlab.eurecom.fr/amadeus/DKFM-recommendation

number of their product is an important source of inspiration and attractiveness to travelers. The key ingredient of DKFM is the use of numerous information coming from different sources that are useful for the recommender system. However, this model has proven to perform less well on sparse data and cannot, by design, account for new users which imply a cold start user problem.

Advertised Ancillary Services

For this use-case, we developed a recommender system based on a simple ML classifier that computes the probability of recommending an ancillary service that an airline wants to offer to travelers who have booked an airline ticket during a given period. The results of the experiments showed the relevance of using ML algorithms instead of rule-based algorithms to better target passengers for ancillary recommendation. Targeting customers through marketing campaigns has been a very important technique used by airlines for a long time, despite its rudimentary use. We have shown through our study the profits that airlines can generate by implementing ML in email marketing campaigns. However, the use of handcrafted features is very demanding as we pointed out in section 4.2, which leads us to use other types of recommender systems to replace the feature engineering work by less heavier in chapter 5.

Hotel Recommendation

For this particular use-case of hotel recommendation that belongs to the third-party recommendation use-case, we use a publicly available dataset provided by Trivago as part of the 2019 RecSys challenge. To address this use-case, we developed a multi-architecture neural network consisting of a recurrent neural network that considers sequential navigation sessions and a multi-layer perceptron that considers session contextual data and hotel content information. The results showed that using this model improves Trivago recommender system. The limited set of features and anonymization of hotels identifiers represent the main limitations of this work as we were not able to enrich our database with more content and useful contextual features such as the duration stay or/and the check-in dates necessary to improve the recommendation accuracy.

- ***RQ2***: How can we build a comprehensive knowledge graph intended for the airline domain? (Section 5.1)

Airline Travel Knowledge Graph

To build a knowledge graph belonging to the airline domain, we made an inventory of the data collected by the airlines but also of the data available on the web in order to cover the entire journey of a traveler. The objective is that the knowledge graph contains all the useful information to reconstruct the traveler's journey from the moment he/she was searching for a flight to the moment he/she boarded the plane. Several data sources have been collected to build this knowledge graph such as those containing information on reservations, transactional information, but also descriptive information on entities (e.g. destinations, ancillaries, etc.) thanks to the cross-referencing in the web and the use of properties such as "owl:sameAs" which allowed us to integrate external data into the knowledge graph. We have built a very large knowledge graph able to be an input source for any recommendation use-case covering the traveler's journey. In addition to being able to standardize the data source to have one useful for any use-case, we have shown the benefit of using a knowledge graph as a data structure in the sections of chapter 5.

- ***RQ3***: How can we leverage knowledge graphs to improve the predictions for each of the recommendation use-cases addressed in this thesis and overcome the standard recommender system limitations? (Chapter 5)

Advertised Ancillary Services

By addressing this use-case through the use of knowledge graphs, the goal was to reduce the time spent building handcrafted features and replace it with another method that produces features that would be able to perform at least as well as handcrafted features. We develop TKE4Rec[2], a framework capable to incorporate latent features (knowledge graph embeddings) coming from the airline travel knowledge graph into a machine learning classifier for ancillary recommendation. Experiments show that using graph embeddings outperformed the use of handcrafted features as input to an XGBoost classifier trained to predict which audience to target for recommending a given ancillary through an email marketing campaign. In addition to being more efficient, the use of embeddings computed through algorithms such as TransE which incorporates triples from the knowledge graph allows us to avoid spending a lot of time on feature engineering.

Next Trip Recommendation

In chapter 5, we revisit the use-case of 'Next Trip Recommendation' by developing KGMTL4Rec[3] a neural network-based model designed to be trained on several tasks (Regression, Binary classification, etc.) in order in incorporate heterogeneous information coming from the airline travel knowledge in order to recommend travel destinations. The results obtained confirm the significant contribution of the use of knowledge graphs as a way to represent the heterogeneous information used for the recommendation task, thus alleviating the problem of data

[2]TKE4Rec: https://gitlab.eurecom.fr/amadeus/tke4rec
[3]KGMTL4Rec: https://gitlab.eurecom.fr/amadeus/KGMTL4Rec

sparsity as the data is densified through the multitudes of links between the different entities and their meaningful use through KGMTL4Rec, hence applying filters in order to keep only travelers who have a fair number of travel interactions is no longer necessary as specified in section 4.1.3. In addition, new users are also taken into account by KGMTL4Rec, as we can train parts of the neural network to obtain embeddings of new users, which was not possible for the DKFM model. Thus, this allows us to overcome not only the data sparsity problem, but also the problem of cold start user.

6.2 Future Work

In this section, we will go over the limitations of the work carried out in this thesis and discuss some of the gaps and opportunities for future work.

We first start by suggesting future work relatively to each recommendation use-case tackled in this thesis:

- **Next Trip Recommendation:** In future, we suggest to explore new data sources such as images that would help to enrich destinations characteristics and could be added in KGMTL4Rec as another learning task in the model. From a more general point of view, we have addressed the task of recommending travel destinations through email campaigns. This task concern only travelers who already traveled with the airline. However, there are several channels where other travelers can be approached, from the airline's website to social networks or online travel agencies. In these channels the available data is different and therefore other contextual data driven recommendation systems or session-based recommendation systems need to be developed for this purpose. We see this as an indispensable asset for the airline to reach a larger audience of consumers in the inspiration phase of the traveler journey.
- **Advertised ancillary services:** Travelers and trip reservations embeddings are computed (see section 5.2.3) in order to replace the handcrafted features, several KGE algorithms have been tested. However, some recent KGE algorithms could be beneficial to improve the model accuracy if implemented. Moreover, we suggest addressing the task of ranking personalized ancillaries in email marketing campaigns. Specifically, the goal would be to recommend a list of ancillaries instead of just a single ancillary. In addition to addressing and optimizing what to recommend to a traveler, it would be interesting to optimize the timing of the notification as this is an important decision factor, especially in the air travel industry [25].
- **Hotel Recommendation:** Particularly in the travel realm, context is very important; as an example the traveling season but also the number of people who travel (e.g. alone, in groups or families) can change the decision a user make to choose an accommodation,

it is important to take this into account and incorporate this information into the recommender system. In addition, enriching the accommodations with external data would be very beneficial to improve the recommendation precision. Indeed, this could be very useful to improve the recommendation performance as shown in chapter 5, due to the nature of the data used in SB recommender systems (highly sparse and large number of new users).

The work presented in this thesis could be extended or improved in many ways. In the remainder of this last part of the thesis, we give an overall picture of the thesis, summarize the limitations and try to indicate some research challenges for the future.

At first, this thesis aims to show the benefits of the implementation of recommender systems in the personalization of the offers proposed by the airlines in the context of the new offer management system as part of the new distribution capability (see section 3.2.2). Secondly, we show through the enrichment of initial interaction data by semantic data the value of using knowledge graphs as a data source for the recommender systems developed to address the use-cases (see chapter 5).

Below, we go through the limitations of our work:

- **Feedback Loop**: Recommender system algorithms influence decisions made by users, thus their preferences, which in turn affect the data (generated from user interactions) used to train the recommender system creating a feedback loop as represented in figure 6.1. Most of recommender system algorithms are correlational and belong to the category of supervised learning algorithms. In fact, the use of historical data is necessary to train and build a model that will predict users' preferences. Despite the efficiency of these algorithms, it drives most of the data resulting in a feedback loop (see figure 6.1). Therefore, recommender systems make decisions that affect the preferences of users. In other words, a user will certainly interact with one of the products that has been offered. Hopefully, the advent of reinforcement learning [136] demonstrated its effectiveness on some recommender system use-cases [126], and is able to overtake the problem of user feedback loop. Indeed, this family of algorithms are causal models, which help the algorithm to reason about the impact of recommendation, and thus consider the user feedback loop in its mechanism.
- **Explainability**: One of the main challenges for the AI community is to bring explainability to decision-making algorithms. Indeed, it is crucial to understand why an algorithm has recommended a specific item. One popular method of explainability arises from Neighborhood Methods that can state, for example, that "a customer that bought this item, also bought these items". KG recommender systems are also ideally suited for this purpose, as this algorithm constructs an explainable path within the knowledge graph that lead to the item recommendation [132]. Moreover, performing an ablation study on

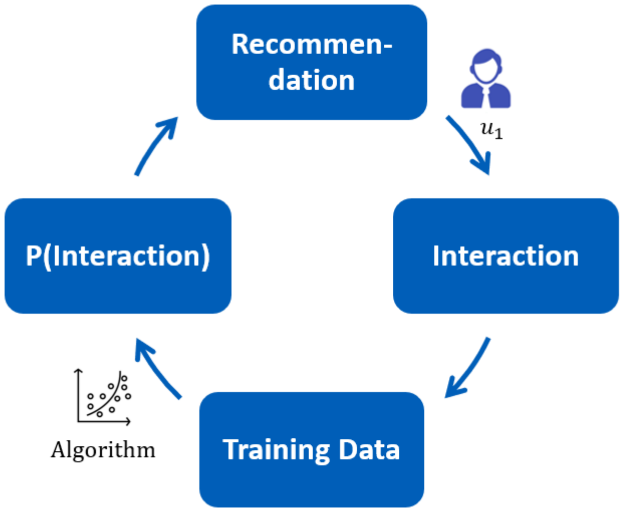

Figure 6.1 – User feedback loop

algorithm inputs, where an input of a model is removed to assess the effect on algorithm performance, would allow us to understand what input data are the most beneficial for an accurate prediction.

- **Offer price**: In this thesis, we focused on the recommendation of products (hotels, ancillary, flights) regardless of their price. We believe, however, that the price of the product is a decisive factor in the customer's final decision to purchase a product, especially in the airline industry where the price of the same product often varies according to supply and demand as presented in section 1.1 (see figure 1.1). Several ideas can be exploited to take into account the price of the product in the recommendation: Consider each pair of product associated with a price as a product in its own, consider the price of the product as an input feature of the model, optimize the price of the product by using a pricing model once the desired product is selected by the recommender system (as shown in figure 3.5).

Finally, if we consider the benefits of NDC listed in section 3.2.2, in this thesis, we have mainly focused on the aspect of personalization and contextualization of offers, showing the usefulness of recommendation systems to reach this objective. However some other aspects of the offer personalization and contextualization were not addressed and we do not demonstrate the value of NDC on achieving the other benefits listed in section 3.2.2. We believe that an improved version of the work around recommender systems in the airline industry could benefit from the following ideas and strategies:

- **Retailing**: Airlines can start customizing the way products are offered to their customers, for example: personalize (or contextualize) product description through enriched multimedia content using personalized visual elements, such as infographics or photos (an airline may be able to show a passenger two different images of Barcelona depending on his preferences). The airline may also want to optimize the time it offers the products by addressing the question: What is the optimal time to offer a certain product? But also optimize the media used to reach a customer. More concretely, if we take the use-case of advertised ancillary services, we could think about optimizing the optimal time to send an email to a traveler depending on the ancillary offered. Indeed a traveler is in a different mind-set depending the time before the flight departure, for example, he/she may think about buying a baggage few days before departure, while an ancillary such as fast-track or lounge maybe desired and though of only few hours before the flight departure. Airline may also want consider optimizing the delivery media used to reach the traveler (email, sms, WhatsApp message, etc.) depending on the preferred device a traveler uses to buy products.

- **Offer Packaging/Fare Families**: Offering a set of products along with airline tickets is the most traditional sales method that airlines have been using for a long time, however the fare families that are offered are built in a rather rudimentary way and this is seriously lacking in personalization and contextualization. Recommender systems can help personalize the way fare families are offered by building a customized set of products that will be suggested with a flight, which would constitute a complete travel solution. Airlines can then customize the search ranking of these travel solutions by offering a different ranking based on customer search criteria, flight context, but also traveler history if identified.
- **Dynamic Bundling**: Once a user has chosen the travel solution that suits him/her, he/she enters his/her personal information and then he/she is redirected to the page where he/she is offered ancillaries to add to his/her flight ticket. In the case where the traveler chooses to add a first product to his cart (baggage for example), the airline can intervene by proposing to add one or more services in addition to the first service added by proposing what is called a bundle of products (different from packs). This bundle, created in a dynamic and contextualized way (and/or personalized) according to the traveler's shopping session, is a way for the airline to sell its products more easily by proposing a discount on the whole offer when the traveler agrees to buy it. Again, recommender systems are the appropriate technology to be able to create this offer in a contextualized (and personalized) way for each traveler, thus providing a personalized traveler experience.
- **Dynamic Pricing**: When a customer makes a shopping request, airlines can intervene to dynamically price the product the customer is looking for. The optimal price is computed to optimize the expected revenue from the shopping session by estimating the customer preferences. The price is optimized taking into consideration the market conditions and the airline's capacity constraints as calculated by the airlines' RMS. More concretely, the goad would be to detect and take advantage of situations where a con-

trolled price would greatly benefice to improve the booking probability. Recommender systems can be combined with dynamic pricing models to take advantage of the personalization capabilities of recommender systems and dynamic pricing models for price optimization.

List of Publications

The research carried out during this PhD thesis has lead to the following scientific publications:

Journal

1. Amine Dadoun, Raphaël Troncy, Michael Defoin Platel, Riccardo Petitti, Gerardo Ayala Solano (2021). Optimizing email marketing campaigns in the airline industry using knowledge graph embeddings. *IEEE Data Engineering Bulletin Journal. KMEcommerce'21 Workshop, held in conjunction with WWW'21.*
2. Amine Dadoun, Michael Defoin-Platel, Thomas Fiig, Corinne Landra, Raphaël Troncy (2021). How recommender systems can transform airline offer construction and retailing. *Journal of Revenue and Pricing Management.*

Conferences and Workshops

1. Amine Dadoun, Raphaël Troncy, Michael Defoin Platel, Riccardo Petitti, Gerardo Ayala Solano (2021). Predicting your next trip: A knowledge graph-based multi-task learning approach for travel destination recommendation–submitted. In RecTour'21 held in conjunction with Recsys '21: Fourteenth acm conference on recommender systems, Amsterdam: Association for Computing Machinery.
2. Amine Dadoun, Raphaël Troncy, Olivier Ratier, Riccardo Petitti (2019). Location embeddings for next trip recommendation. In Companion proceedings of the 2019 world wide web conference. locweb'19 (pp. 896–903). doi:10.1145/3308560.3316535

Posters and Demos

1. Amine Dadoun, Raphaël Troncy, Olivier Ratier, Riccardo Petitti (2018). Semantic Data Driven Approach for Merchandizing Optimization. In Statlearn'18 conference. Nice.
2. Amine Dadoun (2019). Semantic Data Driven Approach for Merchandizing Optimization. In ISWS'19. Bertinoro.

Preprint Articles

1. Nacira Abbas, Kholoud Alghamdi, Mortaza Alinam, Francesca Alloatti, Glenda Amaral, Claudia d'Amato, Luigi Asprino, Martin Beno, Felix Bensmann, Russa Biswas, Ling Cai, Riley Capshaw, Valentina Anita Carriero, Irene Celino, Amine Dadoun, Stefano De Giorgis, Harm Delva, John Domingue, Michel Dumontier, Vincent Emonet, Marieke van Erp, Paola Espinoza Arias, Omaima Fallatah, Sebastián Ferrada, Marc Gallofré Ocaña, Michalis Georgiou, Genet Asefa Gesese, Frances Gillis-Webber, Francesca Giovannetti, Marìa Granados Buey, Ismail Harrando, Ivan Heibi, Vitor Horta, Laurine Huber, Federico Igne, Mohamad Yaser Jaradeh, Neha Keshan, Aneta Koleva, Bilal Koteich, Kabul Kurniawan, Mengya Liu, Chuangtao Ma, Lientje Maas, Martin Mansfield, Fabio Mariani, Eleonora Marzi, Sepideh Mesbah, Maheshkumar Mistry, Alba Catalina Morales Tirado, Anna Nguyen, Viet Bach Nguyen, Allard Oelen, Valentina Pasqual, Heiko Paulheim, Axel Polleres, Margherita Porena, Jan Portisch, Valentina Presutti, Kader Pustu-Iren, Ariam Rivas Mendez, Soheil Roshankish, Sebastian Rudolph, Harald Sack, Ahmad Sakor, Jaime Salas, Thomas Schleider, Meilin Shi, Gianmarco Spinaci, Chang Sun, Tabea Tietz, Molka Tounsi Dhouib, Alessandro Umbrico, Wouter van den Berg, Weiqin Xu (2020). Knowledge graphs evolution and preservation – a technical report from isws 2019. arXiv: 2012.11936 [cs.AI]
2. Amine Dadoun, Ismail Harrando, Pasquale Lisena, Alison Reboud, Raphael Troncy (2020). Two stages approach for tweet engagement prediction. arXiv: 2008.10419 [cs.LG]
3. Amine Dadoun, Raphael Troncy (2020). Many-to-one recurrent neural network for session-based recommendation. arXiv: 2008.11136 [cs.LG]

Bibliography

[1] Redouan Abakouy, El Mokhtar En-Naimi, and Anass El Haddadi. Classification and prediction based data mining algorithms to predict email marketing campaigns. In *Proceedings of the 2nd International Conference on Computing and Wireless Communication Systems*, ICCWCS'17, New York, NY, USA, 2017. Association for Computing Machinery.

[2] R. Acuna Agost and G. Chiari. The importance of understanding travelers' motivation. 2018.

[3] Gediminas Adomavicius and Alexander Tuzhilin. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Transaction on Knowledge and Data Engineering*, 17(6):734–749, 2005.

[4] Gediminas Adomavicius and Alexander Tuzhilin. *Context-Aware Recommender Systems*, pages 191–226. Springer US, 2015.

[5] Rakesh Agrawal, Tomasz Imieliundefinedski, and Arun Swami. Mining Association Rules between Sets of Items in Large Databases. In *ACM SIGMOD International Conference on Management of Data (ICDM)*, pages 207–216, Washington, D.C., USA, 1993.

[6] Vito Walter Anelli, Alejandro Bellogín, Antonio Ferrara, Daniele Malitesta, Felice Antonio Merra, Claudio Pomo, Francesco M. Donini, and Tommaso Di Noia. Elliot: a comprehensive and rigorous framework for reproducible recommender systems evaluation. *CoRR*, abs/2103.02590, 2021.

[7] Diego Antognini and Boi Faltings. Hotelrec: a novel very large-scale hotel recommendation dataset. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 4917–4923, Marseille, France, May 2020. European Language Resources Association.

[8] Trapit Bansal, David Belanger, and Andrew McCallum. Ask the gru: Multi-task learning for deep text recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, page 107–114, New York, NY, USA, 2016. Association for Computing Machinery.

[9] Joeran Beel, Marcel Genzmehr, Stefan Langer, Andreas Nürnberger, and Bela Gipp. A comparative analysis of offline and online evaluations and discussion of research paper recommender system evaluation. In *Proceedings of the International Workshop on Reproducibility and Replication in Recommender Systems Evaluation*, RepSys '13, page 7–14, New York, NY, USA, 2013. Association for Computing Machinery.

[10] R. Boin, A. Cosmas, and N. Wittkamp. Airline retailing: The value at stake. 2019.

[11] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 94(2):233–259, Feb 2014.

[12] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26, Lake Tahoe, Nevada, United States, 2013. Curran Associates, Inc.

[13] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.

[14] Leo Breiman, Jerome Friedman, Charles J. Stone, and R.A. Olshen. *Classification and Regression Trees*. Chapman and Hall/CRC, 1984.

[15] Dan Brickley and R. V. Guha. RDF Schema 1.1, W3C Recommendation, 2014.

[16] Diogo V. Carvalho, Eduardo M. Pereira, and Jaime S. Cardoso. Machine learning interpretability: A survey on methods and metrics. *Electronics*, 8(8), 2019.

[17] Allison J. B. Chaney, Brandon M. Stewart, and Barbara E. Engelhardt. How Algorithmic Confounding in Recommendation Systems Increases Homogeneity and Decreases Utility. In *12^h ACM Conference on Recommender Systems (RecSys)*, pages 224—232, Vancouver, British Columbia, Canada, 2018.

[18] Spencer Chang. Scaling knowledge access and retrieval at airbnb. 2018.

[19] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *22^{nd} ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 785–794, San Francisco, California, USA, 2016. Association for Computing Machinery.

[20] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 794–803, Stockholm, Sweden, 10–15 Jul 2018. PMLR.

[21] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, DLRS 2016, pages 7–10, New York, NY, USA, 2016. ACM.

[22] L. Chittenden and R. Rettie. An evaluation of e-mail marketing and factors affecting response. *Journal of Targeting, Measurement and Analysis for Marketing*, 11:203–217, 2003.

[23] Michael Chui, James Manyika, Mehdi Miremadi, Nicolaus Henke, Rita Chung, Pieter Nel, and Sankalp Malhotra. Notes from the ai frontier insights from hundreds of use-cases. 2018.

[24] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

[25] Amine Dadoun, Michael Defoin-Platel, Thomas Fiig, Corinne Landra, and Raphaël Troncy. How recommender systems can transform airline offer construction and retailing. *Journal of Revenue and Pricing Management*, Mar 2021.

[26] Amine Dadoun, Ismail Harrando, Pasquale Lisena, Alison Reboud, and Raphael Troncy. Two stages approach for tweet engagement prediction, 2020.

[27] Amine Dadoun, Raphaël Troncy, Michael Defoin-Platel, and Gerardo Ayala Solano. Predicting your next trip: A knowledge graph-based multi-task learning approach for travel destination recommendation-Submitted. In *RecSys '21: Fourteenth ACM Conference on Recommender Systems*, RecSys '21, New York, NY, USA, 2021. Association for Computing Machinery.

[28] Amine Dadoun, Raphaël Troncy, Michael Defoin-Platel, Riccardo Petitti, and Gerardo Ayala Solano. Optimizing email marketing campaigns in the airline industry using knowledge graph embeddings. In *Companion Proceedings of The 2021 World Wide Web Conference*, WWW '21, New York, NY, USA, 2021. Association for Computing Machinery.

[29] Amine Dadoun, Raphaël Troncy, Olivier Ratier, and Riccardo Petitti. Location embeddings for next trip recommendation. In *Companion Proceedings of The 2019 World Wide Web Conference-(LocWeb'19)*, WWW '19, page 896–903, New York, NY, USA, 2019. Association for Computing Machinery.

[30] Yuanfei Dai, Shiping Wang, Neal N. Xiong, and Wenzhong Guo. A survey on knowledge graph embedding: Approaches, applications and benchmarks. *Electronics*, 9(5), 2020.

[31] Marco de Gemmis, Pasquale Lops, Giovanni Semeraro, and Cataldo Musto. An investigation on the serendipity problem in recommender systems. *Inf. Process. Manage.*, 51(5):695–717, September 2015.

[32] Wim De Mulder, Steven Bethard, and Marie-Francine Moens. A survey on the application of recurrent neural networks to statistical language modeling. *Computer Speech & Language*, 30(1):61–98, 2015.

[33] Thierry Delahaye, Rodrigo Acuna-Agost, Nicolas Bondoux, Anh-Quan Nguyen, and Mourad Boudia. Data-driven models for itinerary preferences of air travelers and application for dynamic pricing optimization. *Journal of Revenue and Pricing Management*, 16(6):621–639, Dec 2017.

[34] Alexandros Deligiannis, Charalampos Argyriou, and Dimitrios Kourtesis. Predicting the optimal date and time to send personalized marketing messages to repeat buyers. *International Journal of Advanced Computer Science and Applications*, 11(4), 2020.

[35] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 601–610, New York, NY, USA, 2014. Association for Computing Machinery.

[36] Gintare Karolina Dziugaite and Daniel M. Roy. Neural network matrix factorization. *CoRR*, abs/1511.06443, 2015.

[37] Ali Mamdouh Elkahky, Yang Song, and Xiaodong He. A multi-view deep learning approach for cross domain user modeling in recommendation systems. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, page 278–288, Republic and Canton of Geneva, CHE, 2015. International World Wide Web Conferences Steering Committee.

[38] Expedia. Expedia: Millenial traveler report. why millennials will shape the next 20 years of travel. 2016.

[39] A. Felfernig, S. Gordea, D. Jannach, E. Teppan, and M. Zanker. A short survey of recommendation technologies in travel and tourism. 2006.

[40] Thomas Fiig, Remy Guen, and Mathilde Gauchet. Dynamic pricing of airline offers. *Journal of Revenue and Pricing Management*, 17, 04 2018.

[41] I. Franko. Airline retailing: The value at stake. 2019.

[42] Zeno Gantner, Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. MyMediaLite: A free recommender system library. In *Proceedings of the 5th ACM Conference on Recommender Systems (RecSys 2011)*, 2011.

[43] Mouzhi Ge, Carla Delgado-Battenfeld, and Dietmar Jannach. Beyond accuracy: Evaluating recommender systems by coverage and serendipity. In *Proceedings of the Fourth ACM Conference on Recommender Systems*, RecSys '10, page 257–260, New York, NY, USA, 2010. Association for Computing Machinery.

[44] Genet Asefa Gesese, Russa Biswas, Mehwish Alam, and Harald Sack. A survey on knowledge graph embeddings with literals: Which model links better literal-ly? *Semantic Web*, Preprint:1–31, 2020.

[45] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using Collaborative Filtering to Weave an Information Tapestry. *Communication of the ACM*, 35(12):61–70, 1992.

[46] Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. Learning word vectors for 157 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA).

[47] Mihajlo Grbovic and Haibin Cheng. Real-time Personalization using Embeddings for Search Ranking at Airbnb. In *24^{th} ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2018.

[48] Aditya Grover and Jure Leskovec. Node2vec: Scalable feature learning for networks. In *22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 855–864, New York, NY, USA, 2016. ACM.

[49] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. Deepfm: A factorization-machine based neural network for CTR prediction. In *Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1725–1731, 2017.

[50] Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1, 2020.

[51] Guy Hadash, Oren Sar Shalom, and Rita Osadchy. Rank and rate: Multi-task learning for recommender systems. In *Proceedings of the 12th ACM Conference on Recommender Systems*, RecSys '18, page 451–454, New York, NY, USA, 2018. Association for Computing Machinery.

[52] Henry H. Harteveldt. Notes from the ai frontier insights from hundreds of use-case. the future of airline distribution, 2016-2021. 2016.

[53] Xiangnan He and Tat-Seng Chua. Neural factorization machines for sparse predictive analytics. *CoRR*, abs/1708.05027, 2017.

[54] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural Collaborative Filtering. In *26^{th} International Conference on World Wide Web (WWW)*, pages 173—182, 2017.

[55] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, page 173–182, Republic and Canton of Geneva, CHE, 2017. International World Wide Web Conferences Steering Committee.

[56] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua. Fast Matrix Factorization for Online Recommendation with Implicit Feedback. In *39^{th} International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 549–558, 2016.

[57] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. 22(1):5–53, January 2004.

[58] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*, 2015.

[59] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. arXiv 1511.06939, 2016.

[60] Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutiérrez, José Emilio Labra Gayo, Sabrina Kirrane, Sebastian Neumaier, Axel Polleres, Roberto Navigli, Axel-Cyrille Ngonga Ngomo, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan F. Sequeda, Steffen Staab, and Antoine Zimmermann. Knowledge graphs. *CoRR*, abs/2003.02320, 2020.

[61] Yanik Hoyles. New Distribution Capability (NDC) - Together, Let's Build Airline Retailing, 2015.

[62] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S. Yu. Leveraging meta-path based context for top- n recommendation with a neural co-attention model. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '18, page 1531–1540, New York, NY, USA, 2018. Association for Computing Machinery.

[63] Y. Hu, Y. Koren, and C. Volinsky. Collaborative Filtering for Implicit Feedback Datasets. In *8^{th} IEEE International Conference on Data Mining (ICDM)*, pages 263–272, 2008.

[64] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, page 263–272, USA, 2008. IEEE Computer Society.

[65] IATA. Together, let's build airline retailing ndc program update january 2020, 2020.

[66] Folasade Isinkaye, Yetunde Folajimi, and Bolanle Ojokoh. Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal*, 16, 2015.

[67] Anthony Jameson, Martijn C. Willemsen, Alexander Felfernig, Marco de Gemmis, Pasquale Lops, Giovanni Semeraro, and Li Chen. *Human Decision Making and Recommender Systems*, pages 611–648. Springer US, Boston, MA, 2015.

[68] Paweł Jankiewicz, Liudmyla Kyrashchuk, Paweł Sienkowski, and Magdalena Wójcik. Boosting algorithms for a session-based, context-aware recommender system in an online travel domain. In *Proceedings of the Workshop on ACM Recommender Systems Challenge*, RecSys Challenge '19, New York, NY, USA, 2019. Association for Computing Machinery.

[69] Dietmar Jannach and Michael Jugovac. Measuring the Business Value of Recommender Systems. *ACM Transactions on Management Information Systems*, 10(4), 2019.

[70] Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. *Recommender Systems: An Introduction.* Cambridge University Press, USA, 1st edition, 2010.

[71] Iman Kamehkhosh, Dietmar Jannach, and Malte Ludewig. A comparison of frequent pattern techniques and a deep learning method for session-based recommendation. In *RecTemp@RecSys*, 2017.

[72] Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver. Multiverse Recommendation: N-Dimensional Tensor Factorization for Context-Aware Collaborative Filtering. In *4^{th} ACM Conference on Recommender Systems (RecSys)*, pages 79—86, Barcelona, Spain, 2010.

[73] Elias Kärle, Umutcan Simsek, Oleksandra Panasiuk, and Dieter Fensel. Building an ecosystem for the tyrolean tourism knowledge graph. *CoRR*, abs/1805.05744, 2018.

[74] Houda Khrouf and Raphaël Troncy. Hybrid Event Recommendation Using Linked Data and User Diversity. In *7th ACM Conference on Recommender Systems*, pages 185–192, Hong Kong, China, 2013. ACM.

[75] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[76] Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Skip-thought vectors. In *NIPS*, 2015.

[77] Julia Kiseleva, Melanie J.I. Mueller, Lucas Bernardi, Chad Davis, Ivan Kovacek, Mats Stafseng Einarsen, Jaap Kamps, Alexander Tuzhilin, and Djoerd Hiemstra. Where to go on your next trip? optimizing travel destinations based on user preferences. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, page 1097–1100, New York, NY, USA, 2015. Association for Computing Machinery.

[78] Benjamin Klotz, Raphaël Troncy, Daniel Wilms, and Christian Bonnet. Generating semantic trajectories using a car signal ontology. In *WWW (Companion Volume)*, pages 135–138. ACM, 2018.

[79] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, August 2009.

[80] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, August 2009.

[81] Arun Krishnan. Making search easier. 2018.

[82] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[83] Chenyi Lei, Dong Liu, Weiping Li, Zheng-Jun Zha, and Houqiang Li. Comparative deep learning of hybrid representations for image recommendations. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2016.

[84] Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. A Contextual-Bandit Approach to Personalized News Article Recommendation. In *19^{th} International Conference on World Wide Web*, pages 661—670, Raleigh, North Carolina, USA, 2010.

[85] Henry Lieberman. Letizia: An Agent That Assists Web Browsing. In *14^{th} International Joint Conference on Artificial Intelligence (IJCAI)*, 1995.

[86] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 2181–2187, Austin, Texas, 2015. AAAI Press.

[87] Greg Linden, Brent Smith, and Jeremy York. Amazon.Com Recommendations: Item-to-Item Collaborative Filtering. *IEEE Internet Computing*, 7(1):76—80, 2003.

[88] Lajanugen Logeswaran and Honglak Lee. An efficient framework for learning sentence representations. In *International Conference on Learning Representations*, 2018.

[89] Fabiana Lorenzi, Stanley Loh, and Mara Abel. Personaltour: A recommender system for travel packages. In *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Volume 02*, WI-IAT '11, page 333–336, USA, 2011. IEEE Computer Society.

[90] Chun Lu, Philippe Laublet, and Milan Stankovic. Travel attractions recommendation with knowledge graphs. In *20th International Conference on Knowledge Engineering and Knowledge Management - Volume 10024*, EKAW 2016, page 416–431, Berlin, Heidelberg, 2016. Springer-Verlag.

[91] Chun Lu, Philippe Laublet, and Milan Stankovic. Travel attractions recommendation with knowledge graphs. In Eva Blomqvist, Paolo Ciancarini, Francesco Poggi, and Fabio Vitali, editors, *Knowledge Engineering and Knowledge Management*, pages 416–431, Cham, 2016. Springer International Publishing.

[92] Yichao Lu, Ruihai Dong, and Barry Smyth. Why i like it: Multi-task learning for recommendation and explanation. In *Proceedings of the 12th ACM Conference on Recommender Systems*, RecSys '18, page 4–12, New York, NY, USA, 2018. Association for Computing Machinery.

[93] Malte Ludewig and Dietmar Jannach. Evaluation of Session-Based Recommendation Algorithms. *User Modeling and User-Adapted Interaction*, 28(4–5):331–390, 2018.

[94] Lawrence Lundy. Building a more rewarding journey for future traveller tribes, 2015.

[95] Xiao Ma, Liqin Zhao, Guan Huang, Zhi Wang, Zelin Hu, Xiaoqiang Zhu, and Kun Gai. Entire space multi-task model: An effective approach for estimating post-click conversion rate. In *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '18, page 1137–1140, New York, NY, USA, 2018. Association for Computing Machinery.

[96] Augusto Q. Macedo, Leandro B. Marinho, and Rodrygo L.T. Santos. Context-Aware Event Recommendation in Event-based Social Networks. In *9th ACM Conference on Recommender Systems*, pages 123–130, New York, NY, USA, 2015. ACM.

[97] Ricardo Alonso Maturana, Elena Alvarado-Cortes, Susana López-Sola, María Ortega Martínez-Losa, and Pablo Hermoso-González. La rioja turismo: The construction and exploitation of a queryable tourism knowledge graph. In Cesare Pautasso, Fernando Sánchez-Figueroa, Kari Systä, and Juan Manuel Murillo Rodriguez, editors, *Current Trends in Web Engineering - ICWE 2018 International Workshops, MATWEP, EnWot, KD-WEB, WEOD, TourismKG, Cáceres, Spain, June 5, 2018, Revised Selected Papers*, volume 11153 of *Lecture Notes in Computer Science*, pages 213–220. Springer, 2018.

[98] Prem Melville, Raymond Mooney, and Ramadass Nagarajan. Content-Boosted Collaborative Filtering for Improved Recommendations. In *18^{th} National Conference on Artificial Intelligence (AAAI)*, pages 187–192, 2002.

[99] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013.

[100] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[101] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, page 3111–3119, Red Hook, NY, USA, 2013. Curran Associates Inc.

[102] Diego Monti, Enrico Palumbo, Giuseppe Rizzo, and Maurizio Morisio. Sequeval: An offline evaluation framework for sequence-based recommender systems. *Information 2019, 10(5), 174, Special Issue Modern Recommender Systems: Approaches, Challenges and Applications*, 2019. MDPI.

[103] Diego Monti, Enrico Palumbo, Giuseppe Rizzo, Raphaël Troncy, and Maurizio Morisio. Semantic Trails of City Explorations: How Do We Live a City. arXiv 1812.04367, 2018.

[104] Alejandro Mottini, Alix Lheritier, and Rodrigo Acuna-Agost. Airline passenger name record generation using generative adversarial networks, 2018.

[105] Maxim Naumov, Dheevatsa Mudigere, Hao-Jun Michael Shi, Jianyu Huang, Narayanan Sundaraman, Jongsoo Park, Xiaodong Wang, Udit Gupta, Carole-Jean Wu, Alisson G. Azzolini, Dmytro Dzhulgakov, Andrey Mallevich, Ilia Cherniavskii, Yinghai Lu, Raghuraman Krishnamoorthi, Ansha Yu, Volodymyr Kondratenko, Stephanie Pereira, Xianjie Chen, Wenlin Chen, Vijay Rao, Bill Jia, Liang Xiong, and Misha Smelyanskiy. Deep learning recommendation model for personalization and recommendation systems. *CoRR*, abs/1906.00091, 2019.

[106] Carlos Gomez-Uribe Netflix. Challenges and limitations in the offline and online evaluation of recommender systems : A netflix case study. 2012.

[107] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 809–816, Madison, WI, USA, 2011. Omnipress.

[108] Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[109] J. R. Norris. *Markov Chains*. Cambridge University Press, Cambridge, UK, 1997.

[110] Vito Claudio Ostuni, Tommaso Di Noia, Eugenio Di Sciascio, and Roberto Mirizzi. Top-n recommendations from implicit feedback leveraging linked open data. In *Proceedings of the 7th ACM Conference on Recommender Systems*, RecSys '13, page 85–92, New York, NY, USA, 2013. Association for Computing Machinery.

[111] E. Palumbo, G. Rizzo, and R. Troncy. Predicting your next stop-over from location-based social network data with recurrent neural networks. In *ACM RecSys Workshop on Recommenders in Tourism (RecTour)*, Como, Italy, 08/2017 2017.

[112] Enrico Palumbo. *Knowledge graph embeddings for recommender systems*. PhD thesis, 2020. EURECOM. Personal use of this material is permitted. The definitive version of this paper was published in Thesis and is available at :.

[113] Enrico Palumbo, Diego Monti, Giuseppe Rizzo, Raphaël Troncy, and Elena Baralis. entity2rec: Property-specific knowledge graph embeddings for item recommendation. *Expert Syst. Appl.*, 151:113235, 2020.

[114] Enrico Palumbo, Giuseppe Rizzo, and Raphaël Troncy. Entity2rec: Learning user-item relatedness from knowledge graphs for top-n item recommendation. In *Eleventh ACM Conference on Recommender Systems*, pages 32–36, New York, NY, USA, 2017. ACM.

[115] Enrico Palumbo, Giuseppe Rizzo, Raphaël Troncy, Elena Baralis, Michele Osella, and Enrico Ferro. Translational models for item recommendation. In *The Semantic Web: ESWC 2018 Satellite Events*, pages 478–490, Cham, 2018. Springer International Publishing.

[116] Umberto Panniello, Alexander Tuzhilin, Michele Gorgoglione, Cosimo Palmisano, and Anto Pedone. Experimental Comparison of Pre- vs. Post-Filtering Approaches in Context-Aware Recommender Systems. In *3^{rd} ACM Conference on Recommender Systems (RecSys)*, pages 265—268, 2009.

[117] Heiko Paulheim. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web*, 8(3):489–508, 2017.

[118] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

[119] Yves Raimond, Tristan Ferne, Michael Smethurst, and Gareth Adams. The bbc world service archive prototype. *Journal of Web Semantics*, 27-28:2–9, 2014. Semantic Web Challenge 2013.

[120] Steffen Rendle. Factorization Machines. In *IEEE International Conference on Data Mining (ICDM)*, pages 995–1000, 2010.

[121] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *25^{th} Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 452–461, 2009.

[122] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 452–461, Arlington, Virginia, United States, 2009. AUAI Press.

[123] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing Personalized Markov Chains for Next-Basket Recommendation. In *19^{th} International Conference on World Wide Web*, pages 811—820, Raleigh, North Carolina, USA, 2010.

[124] Paul Resnick and Hal R. Varian. Recommender Systems. *Communication of the ACM*, 40(3):56–58, 1997.

[125] Francesco Ricci. Travel recommender systems. *IEEE Intelligent Systems*, 17(6):55–57, 2002.

[126] David Rohde, Stephen Bonner, Travis Dunlop, Flavian Vasile, and Alexandros Karatzoglou. RecoGym: A Reinforcement Learning Environment for the problem of Product Recommendation in Online Advertising. In *International Workshop on Offline Evaluation for Recommender Systems (REVEAL)*, 2018.

[127] Navdeep S. Sahni, Dan Zou, and Pradeep K. Chintagunta. Do targeted discount offers serve as advertising? evidence from 70 field experiments. *Manage. Sci.*, 63(8):2688–2705, August 2017.

[128] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International*

Conference on World Wide Web, WWW '01, page 285–295, New York, NY, USA, 2001. Association for Computing Machinery.

[129] Koceski Saso and Petrevska Biljana. Empirical Evidence of Contribution to E-Tourism by Application of Personalized Tourism Recommendation System. *Scientific Annals of Economics and Business*, 59(1):363–374, July 2013.

[130] Benedikt Schifferer, Gilberto Titericz, Chris Deotte, Christof Henkel, Kazuki Onodera, Jiwei Liu, Bojan Tunguz, Even Oldridge, Gabriel De Souza Pereira Moreira, and Ahmet Erdem. Gpu accelerated feature engineering and training for recommender systems. In *Proceedings of the Recommender Systems Challenge 2020*, RecSysChallenge '20, pages 16–23, New York, NY, USA, 2020. Association for Computing Machinery.

[131] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26, Lake Tahoe, Nevada, United States, 2013. Curran Associates, Inc.

[132] Weiping Song, Zhijian Duan, Ziqing Yang, H. Zhu, M. Zhang, and J. Tang. Explainable knowledge graph-based recommendation via deep reinforcement learning. *ArXiv*, abs/1906.09506, 2019.

[133] Claus Stadler, Jens Lehmann, Konrad Höffner, and Sören Auer. Linkedgeodata: A core for a web of spatial open data. *Semant. Web*, 3(4):333–354, October 2012.

[134] Zhu Sun, Jie Yang, Jie Zhang, Alessandro Bozzon, Long-Kai Huang, and Chi Xu. Recurrent knowledge graph embedding for effective recommendation. In *Proceedings of the 12th ACM Conference on Recommender Systems*, RecSys '18, page 297–305, New York, NY, USA, 2018. Association for Computing Machinery.

[135] Ilya Sutskever, James Martens, and Geoffrey Hinton. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 1017–1024, USA, 2011. Omnipress.

[136] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, 2018.

[137] Nima Taghipour and Ahmad Kardan. A Hybrid Web Recommender System Based on Q-Learning. In *ACM Symposium on Applied Computing (SAC)*, pages 1164—1168, Fortaleza, Ceara, Brazil, 2008.

[138] Kai Sheng Tai, Richard Socher, and Christopher D. Manning. Improved semantic representations from tree-structured long short-term memory networks. *CoRR*, abs/1503.00075, 2015.

[139] Kar Yan Tam and Shuk Ying Ho. Web personalization as a persuasion strategy: An elaboration likelihood model perspective. *Info. Sys. Research*, 16(3):271–291, September 2005.

[140] Xiaoli Tang, Tengyun Wang, Haizhi Yang, and Hengjie Song. Akupm: Attention-enhanced knowledge-aware user preference model for recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '19, page 1891–1899, New York, NY, USA, 2019. Association for Computing Machinery.

[141] Yi Tay, Luu Anh Tuan, Minh C. Phan, and Siu Cheung Hui. Multi-task neural network for non-discrete attribute prediction in knowledge graphs. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, page 1029–1038, New York, NY, USA, 2017. Association for Computing Machinery.

[142] Raphaël Troncy, Giuseppe Rizzo, Anthony Jameson, Oscar Corcho, Julien Plu, Enrico Palumbo, Juan Carlos Ballesteros Hermida, Adrian Spirescu, Kai-Dominik Kuhn, Catalin Barbu, Matteo Rossi, Irene Celino, Rachit Agarwal, Christian Scanu, Massimo Valla, and Timber Haaker. 3cixty: Building comprehensive knowledge bases for city exploration. *Journal of Web Semantics*, 2017.

[143] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 2071–2080, New York, New York, USA, 20–22 Jun 2016. PMLR.

[144] Saúl Vargas and Pablo Castells. Rank and relevance in novelty and diversity metrics for recommender systems. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, page 109–116, New York, NY, USA, 2011. Association for Computing Machinery.

[145] Denny Vrandečić and Markus Krötzsch. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85, September 2014.

[146] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, page 417–426, New York, NY, USA, 2018. Association for Computing Machinery.

[147] Hongwei Wang, Fuzheng Zhang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. Multi-task feature learning for knowledge graph enhanced recommendation. In *The World*

Wide Web Conference, WWW '19, page 2000–2010, New York, NY, USA, 2019. Association for Computing Machinery.

[148] Nan Wang, Hongning Wang, Yiling Jia, and Yue Yin. Explainable recommendation via multi-task learning in opinionated text data. In *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '18, page 165–174, New York, NY, USA, 2018. Association for Computing Machinery.

[149] Q. Wang, Z. Mao, B. Wang, and L. Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743, Dec 2017.

[150] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. Explainable reasoning over knowledge graphs for recommendation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):5329–5336, Jul. 2019.

[151] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, pages 1112–1119, Québec City, Québec, Canada, 2014. AAAI Press.

[152] Wolfgang Wörndl, Alexander Hefele, and Daniel Herzog. Recommending a sequence of interesting places for tourist trips. *Information Technology & Tourism*, 17(1):31–54, 2017.

[153] R. E Wright. Logistic regression. In *Reading and understanding multivariate statistics*, page 217–244. G. Grimm and P. R. Yarnold (Eds.). American Psychological Association., 1995.

[154] Yikun Xian, Zuohui Fu, S. Muthukrishnan, Gerard de Melo, and Yongfeng Zhang. Reinforcement knowledge graph reasoning for explainable recommendation. SIGIR'19, page 285–294, New York, NY, USA, 2019. Association for Computing Machinery.

[155] Dinghe Xiao, Nannan Wang, Jiangang Yu, Chunhong Zhang, and Jiaqi Wu. A practice of tourism knowledge graph construction based on heterogeneous information. In *Proceedings of the 19th Chinese National Conference on Computational Linguistics*, pages 939–949, Haikou, China, October 2020. Chinese Information Processing Society of China.

[156] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. Attentional Factorization Machines: Learning the Weight of Feature Interactions via Attention Networks. In *26^{th} International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3119–3125, 2017.

[157] Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. Representation learning of knowledge graphs with entity descriptions. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 2659–2665, Phoenix, Arizona, 2016. AAAI Press.

[158] Xin Xin, Xiangnan He, Yongfeng Zhang, Yongdong Zhang, and Joemon Jose. Relational collaborative filtering: Modeling multiple item relations for recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, page 125–134, New York, NY, USA, 2019. Association for Computing Machinery.

[159] Yichong Xu, Xiaodong Liu, Yelong Shen, Jingjing Liu, and Jianfeng Gao. Multi-task learning with sample re-weighting for machine reading comprehension. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2644–2655, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[160] Bishan Yang, Scott Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the International Conference on Learning Representations (ICLR) 2015*, May 2015.

[161] Carl Yang, Lanxiao Bai, Chao Zhang, Quan Yuan, and Jiawei Han. Bridging collaborative filtering and semi-supervised learning: A neural approach for poi recommendation. In *23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1245–1254, New York, NY, USA, 2017. ACM.

[162] Carl Yang, Lanxiao Bai, Chao Zhang, Quan Yuan, and Jiawei Han. Bridging Collaborative Filtering and Semi-Supervised Learning: A Neural Approach for POI Recommendation. In *23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1245–1254, New York, NY, USA, 2017. ACM.

[163] Kiseol Yang, Ji Hye Min, and Krishna Garza-Baker. Post-stay email marketing implications for the hotel industry: Role of email features, attitude, revisit intention and leisure involvement level. *Journal of Vacation Marketing*, 25(4):405–417, 2019.

[164] Weilong Yao, Jing He, Guangyan Huang, Jie Cao, and Yanchun Zhang. A Graph-based model for context-aware recommendation using implicit feedback data. *World Wide Web*, 18(5):1351–1371, 2015.

[165] Shih-Yuan Yu, Sujit Rokka Chhetri, Arquimedes Canedo, Palash Goyal, and Mohammad Abdullah Al Faruque. Pykg2vec: A python library for knowledge graph embedding. *Journal of Machine Learning Research*, 22(16):1–6, 2021.

[166] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. Personalized entity recommendation: A heterogeneous information network approach. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, WSDM '14, page 283–292, New York, NY, USA, 2014. Association for Computing Machinery.

[167] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. Personalized entity recommendation: A heterogeneous information network approach. In *7th ACM International Conference on Web Search and Data Mining*, pages 283–292, New York, NY, USA, 2014. ACM.

[168] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. Collaborative Knowledge Base Embedding for Recommender Systems. In *22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 353–362, New York, NY, USA, 2016. ACM.

[169] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. Deep learning based recommender system: A survey and new perspectives. *ACM Comput. Surv.*, 52(1), February 2019.

[170] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. Deep Learning Based Recommender System: A Survey and New Perspectives. *ACM Computing Surveys*, 52(1), 2019.

[171] Weizhen Zhang, Han Cao, Fei Hao, Lu Yang, Muhib Ahmad, and Yifei Li. The chinese knowledge graph on domain-tourism. In James J. Park, Laurence T. Yang, Young-Sik Jeong, and Fei Hao, editors, *Advanced Multimedia and Ubiquitous Engineering*, pages 20–27, Singapore, 2020. Springer Singapore.

[172] Yongfeng Zhang, Qingyao Ai, Xu Chen, and Pengfei Wang. Learning over knowledge-base embeddings for recommendation. *CoRR*, abs/1803.06540, 2018.

[173] Huan Zhao, Quanming Yao, Jianda Li, Yangqiu Song, and Dik Lun Lee. Meta-graph based recommendation fusion over heterogeneous information networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, page 635–644, New York, NY, USA, 2017. Association for Computing Machinery.

[174] Xiangyu Zhao, Long Xia, Liang Zhang, Zhuoye Ding, Dawei Yin, and Jiliang Tang. Deep Reinforcement Learning for Page-Wise Recommendations. In *12th ACM Conference on Recommender Systems (RecSys)*, pages 95—-103, Vancouver, British Columbia, Canada, 2018.

Résumé en français

A.1 Introduction

L'industrie du voyage se concentre généralement sur la vente de produits individuels, même lorsque ces produits sont interdépendants. La nature hétérogène et complexe de cette industrie ne permet pas d'offrir de manière évidente une expérience de voyage complète et flexible dans lesquelles tous les produits nécessaires au voyageur seraient regroupés dans une offre personnalisée représentant l'intégralité d'un voyage. Afin de créer une telle offre, il est nécessaire de comprendre les motivations du voyageur, ses préférences et la manière dont il prend ses décisions.

L'industrie du voyage doit être capable de combler cet écart entre les motivations des voyageurs et la manière dont les services sont proposés, en s'inspirant d'autres secteurs tels que l'ecommerce ou le divertissement.

Si l'on se concentre sur l'industrie du transport aérien, les compagnies aériennes ont commencé et suivi la déréglementation qui a eu lieu dans l'industrie du transport aérien à partir des années 70, elles ont fortement investi dans des systèmes de gestion des revenus (RMS). Pour les compagnies aériennes, ces systèmes sont chargés de définir le prix auquel les sièges des avions doivent être vendus, en tenant compte à la fois de la demande et de l'offre.

Entre-temps, les compagnies aériennes ont connu des changements importants dans la manière de structurer leur offre. Ne vendant au départ que des billets d'avion, les compagnies aériennes vendent désormais des volumes importants de services auxiliaries (ancillary [4]), allant des options de flexibilité au confort supplémentaire à bord. Les compagnies aériennes sont allées plus loin en distribuant également, notamment sur leur site web, du contenu vendu par des fournisseurs tiers (voitures de location, hôtels, excursions, activités, etc.), afin que leur offre couvre l'ensemble du voyage. En vendant maintenant un ensemble d'offres beaucoup plus diversifié, les compagnies aériennes, afin de maximiser leurs revenus, doivent

[4]Ancillary : Les services annexes sont tous les produits proposés par la compagnie aérienne au-delà des billets d'avion. Il peut s'agir de services liés au vol (par exemple, bagages supplémentaires, siège préféré, etc.) ou indépendants (par exemple, accès aux salons)

non seulement décider du **prix** des billets d'avion, mais aussi décider **quoi** offrir, à **quel** client, **quand** l'offrir, à quel prix, et enfin **comment** cette offre doit être présentée au client et sur quel canal de distribution.

D'autres secteurs disposant d'un stock important et d'une large pénétration numérique, tels que les plateformes d'ecommerce, ont déployé des techniques de vente avancées, souvent fondées sur des données et faisant donc largement appel à des méthodes d'apprentissage automatique telles que les systèmes de recommandation, ce qui leur permet de choisir la bonne offre pour le bon client et ainsi d'augmenter leurs revenus ainsi que la satisfaction de leurs clients.

Un système de recommandation peut être considéré comme un algorithme permettant de calculer la probabilité qu'un utilisateur (client) souhaite interagir avec un élément (produit ou service). Ces systèmes ont été introduits à l'origine pour surmonter le problème de la surcharge d'informations auquel les clients sont confrontés lorsqu'ils sont exposés à un large catalogue de produits ou de services. En fournissant aux clients des recommandations contextualisées et personnalisées, les systèmes de recommandation visent à réduire la recherche à un sous-ensemble gérable de produits pertinents pour le client.

Les systèmes de recommandation se sont avérés populaires à la fois pour les clients et les vendeurs, en particulier pour la vente au détail en ligne [124]. L'exemple le plus représentatif est celui d'Amazon, qui est devenu l'un des plus grands vendeur en ligne au monde parce que, parmi d'autres éléments importants tels qu'un grand choix de produits et une chaîne de livraison rapide et fiable, il offre une expérience client optimale grâce à une utilisation intensive des systèmes de recommandation.

Les systèmes de recommandation permettent une expérience d'achat plus personnalisée, donnant aux clients le sentiment d'être compris et reconnus, ce qui contribue à renforcer la confiance et à maintenir la fidélité. Du point de vue du vendeur, les systèmes de recommandation offrent la possibilité de contrôler et d'augmenter l'exposition de son catalogue en conduisant les clients vers des produits manquant de visibilité.
Les systèmes de recommandation sont aussi notoirement bons pour diminuer le taux d'echecs et augmenter le temps moyen passé sur une page web pour la vente en ligne [137]. Enfin, les systèmes de recommandation se sont également avérés très efficaces hors ligne dans les campagnes de marketing par courriel, permettant aux vendeurs de mener à grande échelle ce que l'on appelle le "marketing personnalisé" [69].

Cependant, malgré l'application réussie des systèmes de recommandation dans de nombreux secteurs, la construction et la vente au détail des offres des compagnies aériennes restent assez rudimentaires, avec peu ou pas de différenciation dans la façon dont les produits et services sont sélectionnés, vendus au détail ou tarifés selon les clients.

Nous pensons que l'approche actuelle est inadéquate et que la clé de la rentabilité consiste à gérer les offres de manière cohérente dans un système intégré de gestion des offres (OMS) au service du client tout au long de son voyage, de l'inspiration à l'après-voyage.

Les systèmes de recommandation dans l'industrie du transport aérien souffrent généralement du problème de démarrage à froid et de la rareté des données [29], ainsi l'établissement de profils d'utilisateurs peut être une tâche difficile car les activités individuelles de planification de voyage sont typiquement beaucoup moins fréquentes comme, par exemple, l'achat de livres ou le visionnage de vidéos ; ainsi, les techniques de recommandation sophistiquées telles que largement utilisées par Amazon par exemple ne peuvent pas être directement appliquées au domaine du transport aérien [39].

En effet, en comparaison avec le commerce électronique ou l'industrie du divertissement (Netflix, YouTube, etc.) où les interactions des utilisateurs sont assez nombreuses : Le spectateur moyen de YouTube regarde 5 heures de vidéos par mois[5]. les membres privilégiés d'Amazon passent 24 commandes par an[6] (13 commandes pour les membres non privilégiés) alors que dans le secteur aérien, par exemple, les voyageurs britanniques prennent en moyenne 6,5 vols par an[7] et moins de 5% des voyageurs achètent un service auxiliaire pour un vol donné sur le marché européen. Le manque d'interactions des voyageurs avec le catalogue de produits des compagnies aériennes confirme la rareté de l'ensemble de données et l'utilisation des seules réservations historiques des voyageurs comme informations d'entrée du système de recommandation peut ne pas être suffisante pour suggérer des recommandations précises.

Par conséquent, l'incorporation d'informations supplémentaires telles que le contexte du voyage, les données démographiques des voyageurs ou les métadonnées de la destination dans le système de recommandation pourrait être utile pour résoudre les problèmes mentionnés ci-dessus. Pour intégrer ces informations hétérogènes dans une structure de données unique, le graphe de connaissances est un candidat approprié à considérer. En effet, des travaux récents ont illustré l'efficacité de l'utilisation de l'intégration de graphes de connaissances pour la recommandation d'articles [50].

Une autre raison d'utiliser le graphe de connaissances comme structure pour rassembler toutes les informations nécessaires au développement d'un système de recommandation et donc d'en être une entrée est la multiplicité des cas d'utilisation de la recommandation qui peuvent être traités comme le montre la figure 2. En effet, avoir le graphe de connaissances comme structure de données commune et comme entrée commune à tous les cas d'utilisation est un gain de temps précieux pour les chercheurs et les data scientists lorsqu'ils veulent adresser à chaque fois un nouveau cas d'utilisation.

[5] https://www.comscore.com/
[6] https://www.statista.com/
[7] https://www.news24.com/

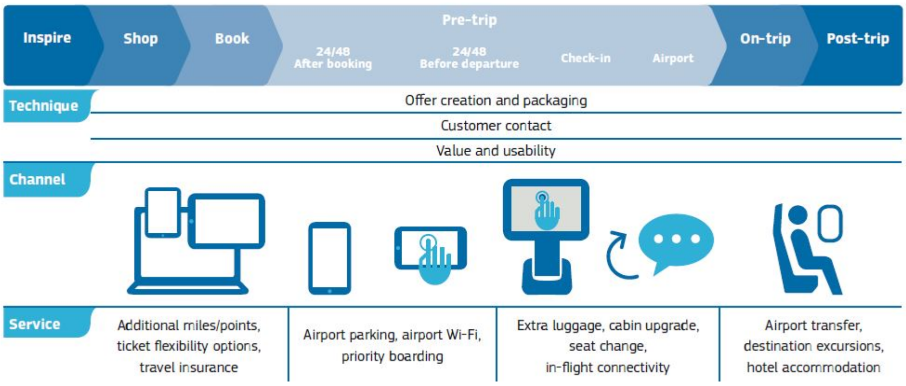

Figure 2 – La figure présente les opportunités de merchandising offertes aux compagnies aériennes tout au long du parcours du voyageur. Source: https://amadeus.com/documents/en/blog/pdf/2014/12/report-thinking-like-a-retailer-airline-merchandising.pdf

Les graphes de connaissances sont devenus une orientation de recherche de plus en plus populaire vers la cognition et l'intelligence de niveau humain, et sont maintenant utilisés dans de nombreuses applications d'IA telles que la recherche sémantique ou la détection automatique des fraudes.

Ces dernières années, les graphe de connaissances ont également été introduits dans les systèmes de recommandation basés sur les graphes de connaissances [50] afin d'enrichir le graphe des interactions utilisateur-article avec des informations plus complexes et structurées sur les utilisateurs, les articles et les interactions elles-mêmes.

L'un des défis de recherche de cette thèse est de pouvoir construire un graphe de connaissances complet qui représente d'abord le voyage complet d'un voyageur depuis le moment où il entre sur la page Web de la compagnie aérienne jusqu'à son embarquement dans l'avion. Le graphe de connaissances doit contenir des informations telles que le contexte du voyage, les données démographiques des voyageurs ou les métadonnées de la destination, mais aussi des descriptions d'événements et d'activités, de lieux et de points d'intérêts, de moyens de transport ainsi que d'activités sociales pertinentes pour une destination. Ces ensembles de données sont collectés auprès de nombreux fournisseurs de données locales et mondiales statiques, en temps réel ou quasi réel, dans le domaine du tourisme. Les entités de ces graphes de connaissances sont automatiquement dédupliquées, interconnectées et enrichies à l'aide des technologies du web sémantique.

Cette thèse se situe à l'intersection entre les domaines de recherche des systèmes de recommandation et des graphes de connaissances avec une application dans l'industrie aérienne, montrant comment les systèmes de recommandation peuvent être mis en place dans cette industrie et transformer la façon dont les compagnies aériennes construisent et vendent leurs produits. La rareté des données collectées dans l'industrie du transport aérien, à l'opposé de la richesse des données disponibles sur le Web, nous amène à utiliser les graphes de connaissance comme structure pour incorporer les données provenant non seulement des bases de données des compagnies aériennes mais aussi du Web. Cette approche permet de tirer parti des avancées des systèmes de recommandation en utilisant des algorithmes basés sur les graphes de connaissances pour améliorer les recommandations proposées aux voyageurs tout au long de leur voyage.

Plusieurs défis et questions de recherche se posent, qui constitueront le point central du travail de recherche de la thèse. Nous décrivons les défis de recherche et les contributions de la thèse à cet objectif de recherche global.

Nous formulons les questions de recherche suivante que nous adressons au fil de la thèse:

- ***RQ1*** : Comment proposer des contenus personnalisés (destinations de voyage, services auxiliaires, contenus tiers) aux voyageurs à l'aide de systèmes de recommandation ?
- ***RQ2*** : Comment construire un graphe de connaissances complet destiné au domaine du transport aérien ?
- ***RQ3*** : Comment pouvons-nous tirer parti des graphes de connaissances pour améliorer les prédictions pour chacun des cas d'utilisation de la recommandation abordés dans cette thèse et surmonter les limites des systèmes de recommandation standard ?

A.2 Vers une nouvelle capacité de distribution des offres de compagnies aériennes

La figure 3 montre comment la demande d'itinéraire d'un client passe d'une plateforme de vente au détail (plateforme Airline Retailing ou autres plateformes de vente au détail), éventuellement par l'intermédiaire d'un distributeur, au système d'inventaire de la compagnie aérienne pour évaluation, en utilisant le modèle de distribution en place aujourd'hui. Pour le canal direct (Direct Connect), la compagnie aérienne contrôle entièrement le flux d'achat et de tarification. Cependant, pour les canaux indirects, le paradigme de distribution actuel repose sur un processus en deux étapes. Tout d'abord, la compagnie aérienne dépose les tarifs auprès de distributeurs de données tels qu'ATPCO ou SITA. Ces tarifs déposés déterminent la construction et la tarification des produits qui peuvent être offerts aux clients. Ensuite, le calcul de la disponibilité dans le système d'inventaire de la compagnie aérienne (Flight Execution) détermine quels sont les tarifs déposés qui sont disponibles à la vente. Les compagnies aériennes contrôlent le calcul de la disponibilité via leur système de gestion des revenus (RMS), qui peut essentiellement être effectué à l'aide de l'optimisation hors ligne (Airline planning).

D'autres plateformes de vente au détail peuvent interagir directement avec la couche d'exécution

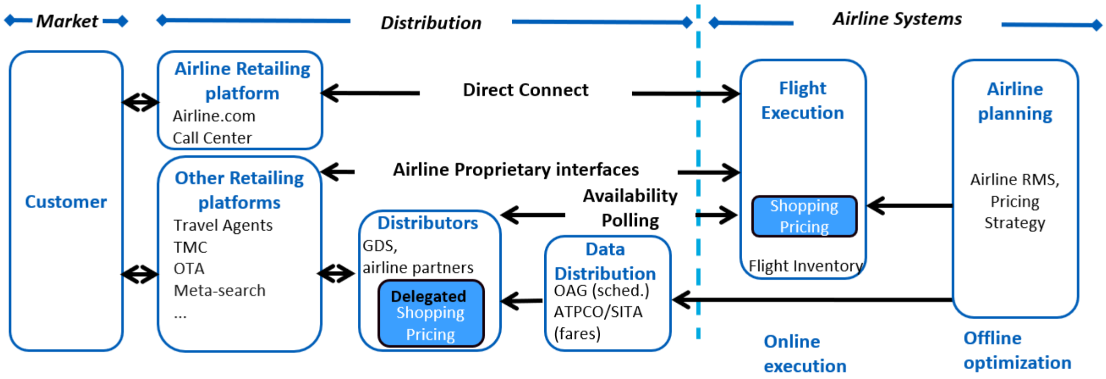

Figure 3 – Modèle de distribution traditionnel

des vols de la compagnie aérienne. Les distributeurs tels que les GDS acquièrent le contenu des tarifs déposés et sont autorisés à créer des offres pour le compte des compagnies aériennes (Delegated Shopping & Pricing). Les distributeurs interrogent ensuite la disponibilité de la compagnie aérienne pour déterminer quels produits tarifaires sont disponibles à la vente. La cohérence entre les canaux indirects est rendue possible par un contenu hautement standardisé et une logique de traitement associée que les SMD adoptent et mettent en oeuvre lorsqu'ils acceptent le contenu des compagnies aériennes et développent leurs moteurs de shopping et de tarification. Cela signifie que la possibilité d'utiliser des informations spécifiques au client dans le canal de distribution indirect est limitée. En principe, même si les compagnies aériennes pouvaient créer des offres contextualisées et personnalisées dans le canal direct, cela créerait une incohérence qui ne peut être résolue entre les canaux de distribution.

Nouvelle capacité de distribution (NDC) La nouvelle capacité de distribution des compagnies aériennes consiste en un ensemble de nouvelles normes de technique lancé il y a près de dix ans par l'Association internationale du transport aérien (IATA). L'objectif de la NDC est de moderniser la distribution des offres des compagnies aériennes et de leur permettre de mieux contrôler leurs offres et leur vente au détail. Nous énumérons ci-dessous les avantages les plus importants pour les compagnies aériennes qui adoptent NDC, qui sont particulièrement pertinents pour cette thèse. Pour de plus amples informations sur les objectifs et les avantages de la NDC, nous renvoyons le lecteur à [61].

- **Offres personnalisées et contextualisées**. Les compagnies aériennes auront accès aux informations du client et aux informations contextuelles dans une demande d'achat ou de réservation, ce qui permettra de proposer des offres personnalisées et contextualisées.
- **Offres dynamiques**. Les compagnies aériennes seront en mesure de créer, de distribuer et d'exécuter des offres dynamiques, comme décrit dans la section suivante.
- **Prix dynamique**. Les compagnies aériennes peuvent employer la tarification dynamique

en utilisant un prix continu.

- **Vente au détail**. Les compagnies aériennes peuvent fournir aux plateformes de vente au détail une description des produits qui englobe les préférences et les informations relatives à la vente au détail. Par exemple, un contenu multimédia enrichi qui complète leurs offres en utilisant des éléments visuels, tels que des infographies, des photos, des vidéos, etc.
- **Merchandising**. Les compagnies aériennes seront capables d'utiliser des techniques de merchandising pour influencer le comportement d'achat des clients.

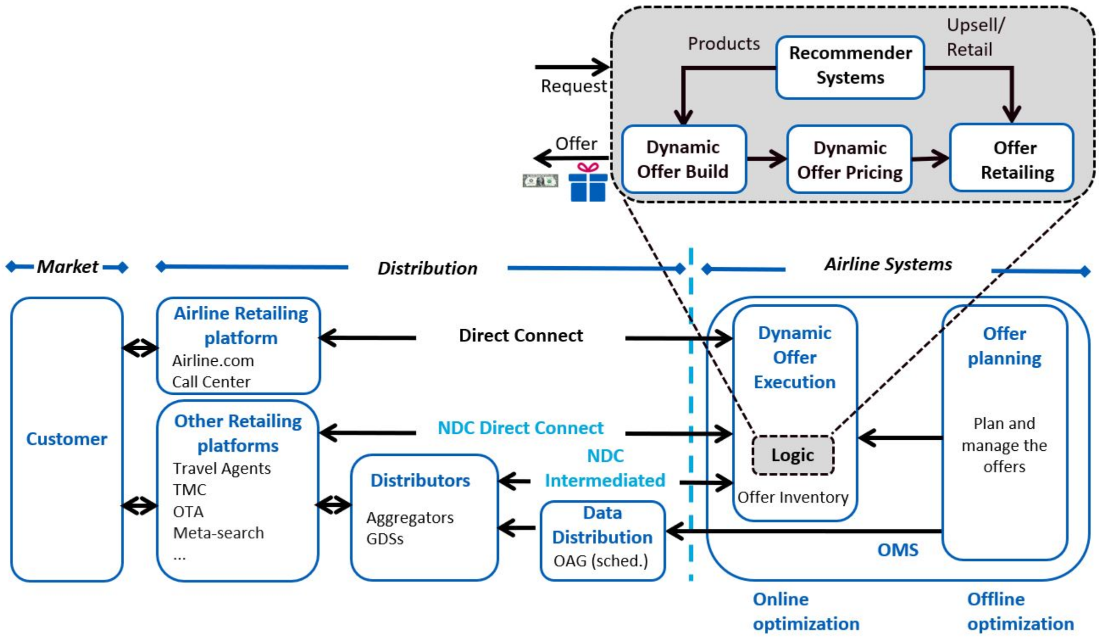

Figure 4 – Modèle de distribution utilisant le NDC

La figure 4 montre comment les compagnies aériennes aspirent à prendre le contrôle de la création de l'offre, à leur propre échelle mais aussi sur tous les canaux de distribution.

Dans l'environnement NDC, les compagnies aériennes prennent toujours la décision de distribuer via des canaux directs et/ou via des canaux indirects avec l'intermédiation de tiers. Cependant, la délégation de la création de l'offre aux intermédiaires n'existe plus. Au lieu de cela, chaque demande d'achat d'un client dans le système de front-office d'un agent est transmise au SGD de la compagnie aérienne, soit directement dans le cas de la distribution par connexion directe NDC, soit par l'intermédiaire d'un agrégateur dans le cas de la distribution par intermédiation NDC. Notez que les flèches "Airline Proprietary Interfaces" et "Availability Polling" de la figure 3 ont été remplacées par les flèches "NDC Direct Connect" et "NDC Intermediated" de la figure 4, permettant un déploiement rentable à l'échelle pour les acteurs du réseau de distribution. L'OMS de la compagnie aérienne crée un ensemble d'une ou plusieurs offres qui sont renvoyées au client. Chaque offre est étiquetée individuellement avec un ID d'offre qui peut être utilisé dans toute demande ultérieure sur cette offre. Si le client accepte une offre, celle-ci est convertie en commande et le contrat avec le client est établi.

A.3 Systèmes de recommandation : cas pratiques au cours du voyage du voyageur

Le parcours du voyageur est un élément clé pour comprendre les besoins et les intentions du client (figure 2). Les recherches menées par Frost and Sullivan [94] indiquent qu'il y a "certains moments où le client est dans un état d'esprit d'achat et pense à son voyage et à ce dont il aura besoin". Par exemple, au moment de la réservation, le client est dans un état d'esprit de 'planification'. À ce stade, la compagnie aérienne peut proposer au client des offres plus 'coûteuses', comme un surclassement en cabine ou des options de flexibilité. À l'approche du départ (48h/24h), le client a un état d'esprit différent - il effectue les derniers préparatifs de son voyage. À ce moment-là, les compagnies aériennes peuvent proposer au client des bagages supplémentaires, un transfert de l'aéroport, un parking, un enregistrement prioritaire ou un accès rapide. Dans cette section, nous détaillons quelques cas d'utilisation des systèmes de recommandation au cours des différentes phases du voyageur.

Afin de fournir une discussion plus approfondie, nous nous concentrons sur les systèmes de recommandation qui sont sous le contrôle des compagnies aériennes. Ces cas d'utilisation concernent les clients qui recherchent et réservent activement des produits de voyage par le biais des canaux de distribution standard rendus possibles par le NDC - canaux directs et indirects. Ainsi, les cas d'utilisation des systèmes de recommandation concernant l'acquisition de clients par le biais des interfaces web des géants de l'Internet, des médias sociaux et des moteurs de recherche ne seront pas couverts, car dans ces cas, les systèmes de recommandation ne sont pas sous le contrôle de la compagnie aérienne.

Recommendation de destination de voyage

La phase d'inspiration est une occasion clé pour influencer le processus de décision du client. Nous distinguons 'l'inspiration passive' de 'l'inspiration interactive'. La première représente le cas où un client (typiquement anonyme) atterrit sur une page web et reçoit une inspiration de voyage simplement parce que certains itinéraires sont populaires en général, tandis que la seconde correspond au cas où le client interagit avec le système de recommandation en fournissant des critères de recherche personnalisés. Dans ce qui suit, pour être concret, nous partons de l'hypothèse que le client reste anonyme et s'engage dans une inspiration interactive, ce qui donne plus de poids au système de recommandation.

Les outils de shopping par affinité peuvent être utilisés pour créer une expérience d'achat personnalisée. Plutôt que de sélectionner les critères traditionnels d'origine/destination et les dates du calendrier, ces outils permettent une inspiration basée sur des critères personnalisés, tels que le budget et les intérêts des clients (événements ou type de destination, comme la plage, la ville, etc.) Un système de recommandation ayant accès à des informations sur les événements à venir (festivals de jazz, événements sportifs, expositions, etc.) et à des informations en temps réel sur les prix des vols et les tarifs promotionnels (campagnes) pourrait être utilisé pour recommander les destinations et les dates les plus appropriées qui correspondent aux critères des clients.) et des informations en temps réel sur les prix des vols et les tarifs promotionnels (campagnes) pourraient être utilisées pour recommander les destinations et les dates les plus appropriées aux critères des clients. Par exemple, un voyage pendant l'été à Nice Côte d'Azur en France, devrait avoir une présentation très différente selon que le client est intéressé par la plage, la vie nocturne ou une expérience culinaire.

Personnalisation du programme de fidélisation des voyageurs

Le modèle économique du programme de fidélisation (FFP) repose sur le fait que les membres du FFP doivent être suffisamment motivés pour gagner et dépenser leurs points. Cependant, dans la réalité, cela peut ne pas être aussi facile. Les membres de niveau supérieur ayant un solde de points important peuvent ne pas être en mesure de trouver de la disponibilité sur des vols attrayants ou des classes supérieures en raison de coupures ou d'un manque de disponibilité des primes, tandis que les membres de niveau inférieur ayant un solde de points faible ne peuvent souvent pas se permettre un billet de remboursement et ne voient pas l'intérêt du programme.

Les systèmes de recommandation sont bien placés pour augmenter le nombre de points brûlés en utilisant des informations concernant à la fois le solde de points des membres et la disponibilité des billets primes. Par exemple, on peut proposer à un membre de niveau supérieur de brûler des points pour obtenir des surclassements pour sa famille lors de son voyage de vacances annuel (afin d'atténuer le risque de dilution du billet prime qui se substitue à un billet commercial) ou des contenus non aériens qui ne sont pas facilement accessibles à l'achat sur le marché libre (par exemple, des laissez-passer pour les coulisses de concerts, de jeux, etc.) Pour les membres de rang inférieur, les systèmes de recommandation pourraient offrir une "remise" sur le prix d'un billet commercial.

Plusieurs autres cas d'utilisation des systèmes de recommandation peuvent également être identifiés, tels que l'incitation des membres à gagner des points pour atteindre le niveau supérieur ou à brûler des points proches de l'expiration. Dans tous ces cas, le système peut être en mesure d'augmenter la valeur du programme en envoyant des courriels personnalisés aux membres avec la bonne offre au bon moment.

Filtrage et classement des résultats de recherches

Pour un client qui effectue ses recherches en comparant les prix, la réservation d'un voyage en avion peut être une expérience décourageante. Il doit établir un ordre de priorité parmi des centaines d'itinéraires potentiels, avec des prix et des caractéristiques de produits différents parmi les multiples compagnies aériennes partenaires. En conséquence, il devient presque impossible pour le client de prendre une décision d'achat. Aujourd'hui, la plupart des algorithmes de recherche visent à trouver les tarifs les plus bas mais, ce faisant, ils créent des itinéraires non pertinents ou peu attrayants qui distraient ou submergent le client.

Un système de recommandation peut filtrer l'ensemble des choix en un nombre gérable d'alternatives et les classer par ordre de pertinence sur la base d'une compréhension des critères énoncés par le client. De cette façon, le système de recommandation guide le client dans son processus de décision et profite à la compagnie aérienne en améliorant les taux de conversion. Nous pouvons également ajouter de nouveaux critères personnalisés au-delà des critères habituels d'origine-destination, de plage de dates, de temps de vol, de temps au sol et de nuitée, afin d'incorporer des attributs de produit tels que la cabine, la flexibilité du billet, la réservation de siège et la franchise de bagages, qui ne sont généralement pas pris en compte dans les demandes de comparaison des prix aujourd'hui.

Vente incitative, vente croisée et contenu tiers

Lorsque le client a choisi son itinéraire préféré, il entre dans la phase de réservation. Pendant la phase de réservation, le système de recommandation dispose d'informations idéales sur le client et son groupe de voyageurs - non seulement la destination actuelle du voyage, sa durée et les services auxiliaires déjà sélectionnés, mais aussi le profil du client et l'historique de ses achats. Au moment de la réservation, le client est dans un état d'esprit de planification et c'est l'occasion idéale d'augmenter les recettes accessoires des compagnies aériennes et d'offrir une expérience d'achat unique qui couvre l'ensemble du voyage du client.

Parmi les produits qui peuvent être recommandés à ce stade, citons les offres de vente incitative telles que les surclassements de cabine ou les options de flexibilité des billets, ainsi que les offres de vente croisée telles que les bagages, les réservations anticipées de sièges ou les services en vol (par exemple, les repas). En outre, la compagnie aérienne peut également proposer un contenu tiers. En fonction des besoins du client, de la relation commerciale avec les tiers, des prix et des disponibilités des ressources pertinentes, le système de recommandation peut proposer des produits simples tels que des assurances, des transferts d'aéroport, etc. ou même des voyages groupés plus complexes tels que des forfaits vacances comprenant des hôtels et des voitures de location.

Advertised Services

Pendant la période de post-achat, la compagnie aérienne a la possibilité de proposer des offres aux clients par le biais de courriers non sollicités ou de notifications sur un appareil mobile. Cette période est une phase critique pour les décisions de dernière minute des clients et les préparatifs de leur voyage. Les clients peuvent être approchés pour leur proposer des services auxiliaires tels que des bagages supplémentaires, un parking à l'aéroport, la sélection d'un siège, un enregistrement prioritaire, etc. et être informés de la disponibilité de surclassements en cabine correspondant à leurs préférences. Là encore, l'offre et la communication seraient très différentes selon qu'il s'agit d'une famille de quatre personnes voyageant sur un long courrier de Francfort à New York en classe économique pour des vacances de deux semaines, ou d'un client à vocation professionnelle voyageant sur le même itinéraire et dans la même cabine, mais ne restant que deux jours. Un système de recommandation proposerait non seulement les offres les plus pertinentes, mais aussi le canal et le moment les plus appropriés pour diffuser ces offres, ce qui permettrait d'augmenter les taux d'adoption et la satisfaction des clients.

Expérience aéroport

Pendant l'enregistrement, les clients interagissent activement avec la compagnie aérienne par l'intermédiaire des employés au comptoir d'enregistrement, du kiosque ou sur des appareils mobiles. Pendant cette phase, le client se concentre sur les aspects pratiques avant le décollage. Cela peut concerner la logistique de la navigation dans l'aéroport, mais le client peut aussi souhaiter se faire plaisir avec des restaurants, un accès au salon ou des surclassements en cabine, qui peuvent être payés par exemple avec des points FFP.

Si l'on considère les personnages mentionnés précédemment, la famille de quatre personnes revenant de vacances à New York peut avoir un excédent de bagages, tandis que le client à vocation professionnelle revenant de New York sur un vol de nuit peut rechercher un surclassement en cabine affaires. Ces exemples illustrent le fait que les besoins des clients peuvent varier considérablement et que la compagnie aérienne a la possibilité d'approcher les clients avec des offres pertinentes basées sur une compréhension approfondie de leurs besoins, préférences et intentions.

A.4 Personnalisation de l'offre de destinations de voyage à travers l'utilisation de graphe de connaissances

Les solutions actuelles des compagnies aériennes qui fournissent des recommandations sur les destinations de voyage manquent de contextualisation et, surtout, de personnalisation. Elles utilisent soit une solution qui suggère les destinations les plus populaires à tous les voyageurs, soit un outil d'inspiration interactif qui fait correspondre les critères des voyageurs (budget, intérêts, etc.) avec les destinations de voyage. Dans notre étude, nous nous concentrons sur un scénario d'inspiration passive, dans lequel les destinations de voyage sont envoyées aux voyageurs par des campagnes de marketing par courriel des compagnies aériennes, dans le but de faciliter le processus de recherche pour les voyageurs.

L'utilisation de méthodes de *filtrage collaboratif* (CF) pour la recommandation de destinations de voyage souffre du problème du démarrage à froid et de la rareté des données [29]. En effet, utiliser uniquement l'historique des réservations des voyageurs comme information d'entrée du système de recommandation peut ne pas être suffisant. Par conséquent, l'incorporation d'informations supplémentaires telles que le contexte du voyage, les données démographiques du voyageur ou les métadonnées de la destination dans le système de recommandation pourrait être utile pour résoudre les problèmes mentionnés ci-dessus. Pour intégrer ces informations hétérogènes dans une structure de données unique, le graphe de connaissances est un candidat approprié à considérer. En effet, des travaux récents ont illustré l'efficacité de l'utilisation de de plongements (embeddings) issus de graphes de connaissances pour la recommandation d'articles. Cependant, comme le souligne [44], tous les algorithmes de plongement de graphes de connaissances ne sont pas efficaces pour combiner différents types de littéraux et la plupart d'entre eux ne disposent pas d'un mécanisme approprié pour gérer les littéraux à valeurs multiples (texte, image, valeur numérique, etc.). Inspirés par le travail proposé dans [141], où les auteurs proposent une approche pour l'apprentissage relationnel et la prédiction d'attributs non discrets sur des graphes de connaissances, nous proposons **K**nowledge **G**raph-based **M**ulti **Task**. **L**earning For **Rec**ommendation (KGMTL4Rec[8]), un algorithme d'apprentissage multitâche basé sur un réseau neuronal pour la recommandation de destinations de voyage qui exploite les informations du graphe de connaissances[9]. Nous présentons l'architecture du modèle dans la figure 5.

Les contributions du travail proposé peuvent être résumées comme suit : (a) nous construisons un graphe de connaissances englobant les réservations historiques des voyageurs (informations collaboratives) ainsi que les contextes de réservation, les métadonnées des voyageurs et des destinations grâce au Linked Open Data ; (b) nous proposons un modèle d'apprentissage multi-tâches pour apprendre les représentations vectorielles des entités du graphe de connaissances ; (c) nous utilisons KGMTL4Rec pour calculer les scores de recommandation de destination de voyage entre les voyageurs et les destinations ; (d) nous menons des expériences approfondies pour comparer KGMTL4Rec avec le système de recommandation actuellement en production et les systèmes de recommandation de destination de voyage de pointe.

[8] https://gitlab.eurecom.fr/amadeus/KGMTL4Rec

[9] https://gitlab.eurecom.fr/amadeus/KGMTL4Rec/ontology

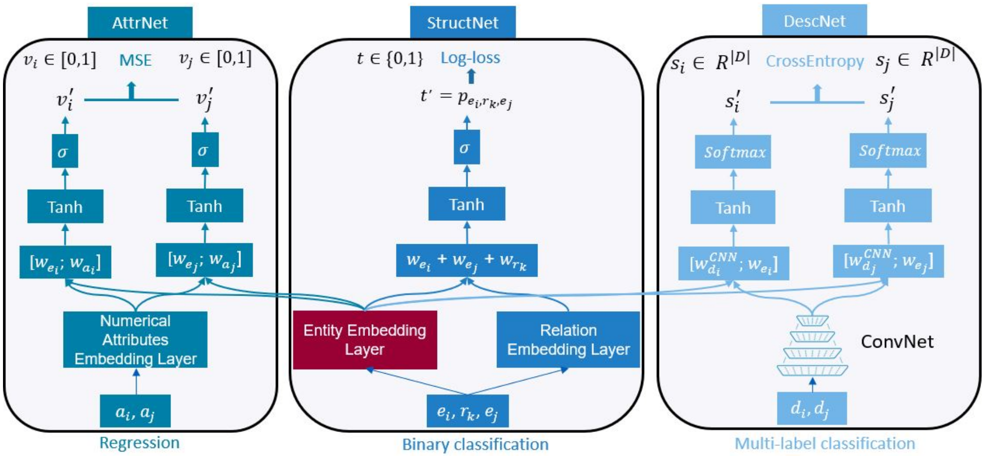

Figure 5 – Architecture de KGMTL4Rec : Un réseau neuronal composé de trois sous-réseaux, chaque sous-réseau étant spécialisé dans une tâche d'apprentissage. La même couleur est utilisée pour les différents éléments d'un sous-réseau (par exemple, la couleur turquoise pour AttrNet). La couleur rouge est attribuée à la couche d'intégration des entités, car ses poids sont partagés par les différents sous-réseaux.

A.4.1 Problématique et Questions de recherche

L'objectif de notre travail est de construire un système de recommandation qui suggère une liste classée de destinations où les voyageurs aimeraient se rendre, comme le montre la figure 6.

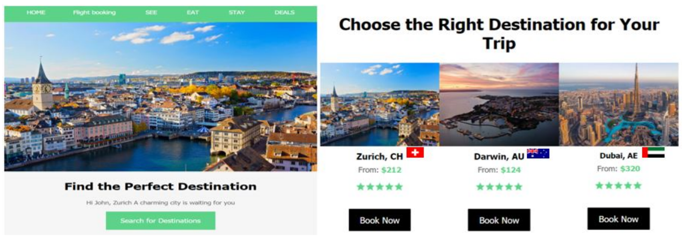

Figure 6 – Recommandations de destinations de voyage incluses dans un e-mail de marketing.

Plus précisément, notre objectif est de proposer aux voyageurs touristes des destinations de voyage qu'ils n'ont encore jamais visitées. Nous considérons les réservations passées des voyageurs, les contextes de réservation et les métadonnées des voyageurs et des destinations comme des informations à utiliser dans notre système de recommandation. Ces informations sont collectées et stockées dans le graphe de connaissances décrit dans la section A.4.2. La tâche consistant à recommander la prochaine destination à un voyageur est formulée comme une tâche de prédiction de liens dans un graphe de connaissances. Dans ce travail, nous abordons les questions de recherche suivantes :

1. Quel est l'avantage d'utiliser un graphe de connaissances comme structure de données unique contenant toutes les informations d'entrée du système de recommandation ?

2. Compte tenu de la nature hétérogène des informations incluses dans le graphe de connaissances (valeurs numériques, dates, textes, etc.), quelle est l'approche la plus performante pour la recommandation de destinations de voyage ?

A.4.2 Construction du Graphe de connaissance

Nous travaillons sur un ensemble de données de production réelles de réservations provenant de la base de données T-DNA [10]. Chaque réservation contient un ou plusieurs achats de billets d'avion, et est stockée à l'aide des informations du 'Passenger Name Record' (PNR). Le PNR est créé au moment de la réservation par le système de réservation de la compagnie aérienne et contient des informations sur le billet d'avion acheté (par exemple, l'itinéraire de voyage, les informations relatives au paiement, etc.), les données démographiques du voyageur et les services supplémentaires (par exemple, un siège préféré, un sac supplémentaire) s'ils ont été achetés. L'ensemble de données considéré contient environ 36 mille réservations de novembre 2018 à décembre 2019, environ 9271 voyageurs uniques et 136 destinations différentes. Notre graphe de connaissance englobe 5 types d'entités, à savoir :

- **Traveler** : Un voyageur est identifié de façon unique par un identifiant T-DNA. Un voyageur possède un historique de réservation d'achats (par exemple, des billets d'avion). Une instance de voyageur est un `schema:Person`[11].
- **Réservation de voyage** : Une réservation de voyage (PNR) représente la réservation de tous les voyageurs contenus dans le PNR. Elle contient des informations telles que le nombre de voyageurs, la destination, etc.
- **Journey** : Un voyage est lié à une réservation de voyage. Chaque voyage a une durée de séjour, un aéroport de départ et un aéroport d'arrivée.
- **Billet d'avion** : Un billet d'avion est contenu dans un PNR et contient des informations sur le vol et la transaction.
- **Airport** : Il représente l'aéroport vers lequel le voyageur se rend. Un aéroport dessert une ou plusieurs villes.

Tout au long de ce travail, nous utilisons l'ontologie qui est définie et disponible au format Turtle[12]. Une destination vers laquelle un voyageur s'est rendu est décrite par une propriété que nous nommons 'travelto' non définie par l'ontologie. L'objectif du système de recommandation est de prédire les liens corrects étiquetés par la propriété 'travelto' entre les voyageurs et les destinations.

[10] T-DNA : Traveler DNA est une base de données qui contient les réservations des voyageurs d'une douzaine de compagnies aériennes. L'ensemble de données utilisé dans les expériences est conforme aux régulations régis par le RGPD et ne comprend aucune information personnelle identifiable.

[11] Le préfixe `schema` est utilisé pour les concepts définis par https://schema.org

[12] http://bit.ly/kg-ontology

En plus du graphe de connaissance construit à travers les données collectés par les compagnies aériennes (T-DNA), nous exploitons la propriété 'owl:sameas' pour utiliser des données disponibles dans le web à travers le 'Linked Open Data' afin d'enrichir le graphe de connaissances avec des métadonnées sur les destinations. De manière plus formelle, nous utilisons le graphe de connaissances de wikidata[13], le graphe de connaissances de Semantic Trails Dataset [103] et la description textuelle de Wikipedia des destinations de voyage pour alimenter notre graphe de connaissance original de voyage. Au final, le graphe de connaissance utilisé pour notre tâche de recommandation contient 48 propriétés différentes, 2,7 millions d'arêtes et 125 000 noeuds.

Dans la figure 7, un extrait du graphe de connaissance est représenté, où un voyageur singapourien, né le 27 Mars 1994 a réservé un vol aller simple de Kuala Lumpur à Melbourne (la propriété 'travelto' provenant du voyageur pointe sur l'aéroport de Melbourne).

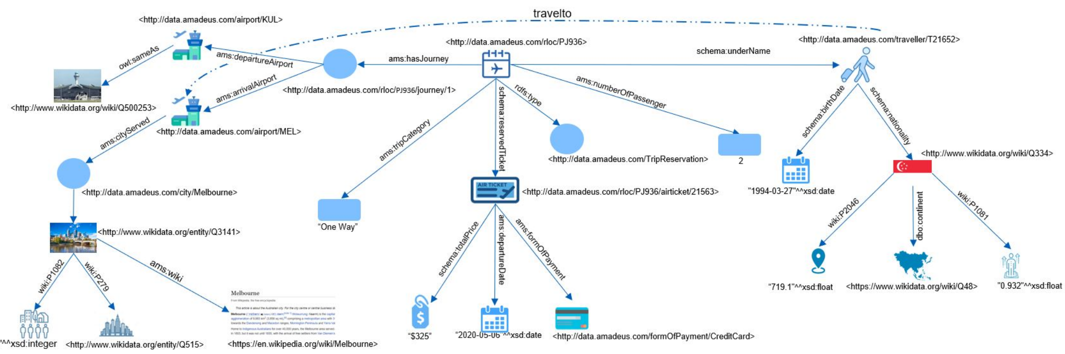

Figure 7 – Extrait du graphe de connaissances représentant un voyageur inclus dans une réservation de Voyage à travers la propriété `schema:underName`, ainsi que d'autres propriétés et relations avec d'autres entités. Les littéraux sont représentés dans un rectangle bleu, tandis que les autres entités sont représentées dans un cercle bleu. Dans cette représentation, certaines propriétés qui relient les voyageurs, les réservations de voyage, les billets d'avion, les destinations de voyage sont représentées à titre d'exemple, mais d'autres propriétés sont incluses dans le graphique.

A.4.3 Étude empirique du modèle KGMTL4Rec

Largement utilisé dans la littérature [55, 122], et surtout adopté dans [29], le protocole *leave-one-out* suggère de sélectionner la dernière interaction comme ensemble de test et d'utiliser les données restantes dans l'ensemble d'apprentissage et de validation. Nous utilisons ce protocole pour évaluer la performance de KGMTL4Rec et aussi pour le comparer avec les différents modèles de base (figurant sur le tableau 1). Notre ensemble de données est trié temporellement de sorte que le dernier voyage corresponde à la destination la plus récente visitée par un voyageur, afin de représenter la notion de recommandation de la 'prochaine'

[13] https://www.wikidata.org/

destination de voyage pendant l'évaluation. Pour chaque voyageur, nous classons toutes les destinations sauf celles qui ont déjà été visitées par le voyageur et nous tronquons la liste à 10, puisque 10 destinations sont incluses dans l'email envoyé aux voyageurs. Pour valider notre modèle, nous appliquons une validation croisée à l'ensemble de données de formation (k=5, une répartition de 80 % pour la formation et 20 % pour la validation). La répartition entre l'ensemble de formation et l'ensemble de validation est effectuée de manière aléatoire sur les voyages afin d'éviter un effet de saisonnalité qui se produit habituellement dans l'industrie du voyage.

La sortie du système de recommandation est une liste classée de 10 destinations, où au mieux, un élément des 10 destinations recommandées est pertinent et correspond à la "prochaine" destination du voyageur. Compte tenu de cela, nous pensons qu'il est judicieux d'utiliser la métrique du taux de réussite (hit rate) pour mesurer si la destination pertinente est ou non dans la liste des 10 premières destinations et d'utiliser la métrique du rang réciproque moyen (Mean reciprocal rank) pour capturer à la qualité du ranking retourné par le système de recommandation. Les deux métriques sont définies comme suit :

- **HR@K**:

$$HR@K = \frac{1}{n}\sum_{t=1}^{n}\sum_{j=1}^{K} hit(t, d_j) \tag{1}$$

- **MRR@K**:

$$MRR@K = \frac{1}{n}\sum_{t=1}^{n}\sum_{j=1}^{K} \frac{1}{rank(rel_t)} \tag{2}$$

où n représente le nombre de voyageurs, K la longueur de la liste classée et $hit(t, d_j)$ est égal à 1 si le voyageur t a voyagé vers la destination d_j. Dans l'équation 2, $rank(rel_t)$ est le rang de la destination pertinente où le voyageur t s'est rendu. Le rang n'est pris en compte que si la destination concernée figure dans la liste top-K.

Nous implémentons notre modèle KGMTL4Rec en utilisant Pytorch[14] puisqu'il nous offre plus de flexibilité dans l'implémentation de nouvelles architectures de réseaux neuronaux et nous utilisons la bibliothèque Pykg2vec[15] pour les modèles basés sur les graphes de connaissances, enfin nous utilisons Tensorflow[16] pour implémenter les modèles de base des réseaux neuronaux. Nous utilisons l'initialisateur uniforme de Xavier pour initialiser de manière aléatoire les paramètres des modèles et nous utilisons une technique d'optimisation par mini-batchs basée sur l'optimisateur Adam [75] pour entraîner tous les modèles. Pour ajuster

[14]https://pytorch.org/
[15]https://pykg2vec.readthedocs.io/
[16]https://www.tensorflow.org/

les hyperparamètres de notre modèle et des modèles de base, nous utilisons l'ensemble de validation mentionné ci-dessous. Nous appliquons l'algorithme de recherche par grille sur les modèles implémentés en utilisant les valeurs suivantes : la taille des plongements des entités $d \in \{16,32,64,128,256\}$, la taille du batch $\in \{128,256,512,1024\}$, le nombre d'époques $\in \{10,20,50,100,200\}$, le taux d'apprentissage $\lambda \in \{0.\ 00001,0.0001,0.0003,0.001,0.003,0.01,0.1\}$ et les échantillons négatifs Ns $\in [2,10]$.

Dans le tableau 1, nous présentons les performances de recommandation de KGMTL4Rec et des modèles de base par rapport à HR@10 et MRR@10. Les résultats rapportés dans le tableau 1 correspondent aux performances des différents modèles basés sur les hyperparamètres les plus performants. Nous rapportons la moyenne et l'écart type de HR@10 et MRR@10 sur 5 graines différentes en raison de l'initialisation aléatoire des paramètres des réseaux neuronaux.

Table 1 – Experimental results.

(a) Performance de recommendation des algorithmes recommandation collaboratifs, hybrid et basés sur le contexte.

Model	HR@10	MRR@10
Item-pop	0.5372	0.3021
IKNN [128]	0.3265	0.1412
BPRMF [122]	0.5462 ± 0.001	0.2993 ± 0.0005
NCF [55]	0.5097 ± 0.013	0.2966 ± 0.0010
FM [123]	0.5806 ± 0.006	0.3260 ± 0.0002
WDL [21]	0.6001 ± 0.015	0.3472 ± 0.0007
DKFM [29]	**0.6464** ± 0.018	**0.3856** ± 0.0012

(b) Performance de la recommandation des algorithmes de recommandation à base de graphe de connaissance.

Model	HR@10	MRR@10
NTN [131]	0.3060 ± 0.002	0.1463 ± 0.0012
SME [11]	0.3628 ± 0.001	0.1959 ± 0.0003
TransE [12]	0.4148 ± 0.003	0.2100 ± 0.0002
TransH [151]	0.3813 ± 0.004	0.1713 ± 0.0005
TransR [86]	0.3908 ± 0.002	0.1808 ± 0.0007
ER-MLP [35]	0.5896 ± 0.016	0.3359 ± 0.0053
KGMTL4Rec	**0.6907** ± 0.023	**0.4189** ± 0.0193

Il est important de noter que tous les systèmes de recommandation n'utilisent pas les mêmes informations en entrée. En effet, les systèmes de recommandation qui utilisent non seulement l'historique du voyageur mais aussi d'autres types d'informations en entrée, comme DKFM ou WDL, ont tendance à être plus performants que les systèmes de recommandation collaboratifs simples comme ImplicitMF, NCF ou Item-KNN, comme le montre le sous-tableau (a). Comme DKFM, les systèmes de recommandation basés sur les graphes de connaissances représentés dans le sous-tableau (b) utilisent toutes les informations mentionnées dans la section A.4.1. Il est donc légitime de comparer KGMTL4Rec à DKFM, où nous observons clairement que KGMTL4Rec est plus performant en ce qui concerne HR@10 et MRR@10. KGMTL4Rec surpasse non seulement le modèle DKFM mais aussi les autres systèmes de recommandation basés sur les graphes de connaissances représentés dans le sous-tableau (b). La principale différence entre KGMTL4Rec et les autres systèmes de recommandation basés sur les graphes de connaissances est que KGMTL4Rec utilise chaque type d'information de manière optimale dans l'un des sous-réseaux montrés dans la figure 5, alors que des modèles comme TransE, NTN ou même ER-MLP considèrent les valeurs numériques comme une entité séparée, ce qui non seulement augmente considérablement la cardinalité de l'ensemble des entités considérées dans ce type de méthode, mais aussi considère des valeurs numériques égales comme la même entité (Il n'est pas correct de considérer 12 'anciens' et 12 'jours' comme la même entité.).

A.4.4 Conclusion

Dans cette section, nous avons étudié le cas d'utilisation de la "recommandation du prochain voyage" en proposant un modèle qui incorpore des informations hétérogènes provenant d'un graphe de connaissances multitypes, à savoir : **KGMTL4Rec**, un algorithme d'apprentissage multitâche conçu pour prendre en compte non seulement les entités du graphe de connaissances, mais aussi les littéraux numériques et textuels, afin de recommander des destinations de voyage personnalisées aux clients des compagnies aériennes dans le cadre de campagnes de marketing par courrier électronique. KGMTL4Rec est basé sur une architecture de réseau neuronal qui peut incorporer différents types d'informations disponibles dans le graphe de connaissances. Nous avons menons plusieurs expériences pour répondre aux questions de recherche mentionnées dans la section A.1: Notre modèle est capable de prédire les liens manquants 'travelto' dans le graphe de connaissances avec un HR@10 de ~ 0.69. De plus, nous démontrons, par une comparaison approfondie entre KGMTL4Rec et DKFM (voir article [27]), la contribution précieuse de l'utilisation du graphe de connaissances comme structure unique pour représenter les informations hétérogènes utilisées pour la recommandation de destinations de voyage.

Les résultats confirment la contribution significative de l'utilisation des graphes de connaissances comme moyen de représenter les informations hétérogènes utilisées pour la tâche de recommandation, ainsi que l'avantage de l'utilisation d'un modèle d'apprentissage multitâches en termes de performance de recommandation et de temps de formation.

À travers ce travail, nous avons démontrons la méthodologie de construction de la recommandation en utilisant les graphes de connaissances pour représenter les informations hétérogènes, et l'apprentissage multi-tâches qui prend en compte ces informations hétérogènes à travers les tâches d'apprentissage multiples (régression, classification binaire, classification multi-classes). Les résultats montrent que, même avec des données aussi éparses, l'ajout de données qualitatives par l'enrichissement des interactions de voyage peut conduire à de meilleures recommandations de destinations de voyage qu'avec les systèmes de recommandation hybrides traditionnels.

A.5 Optimisation des campagnes de marketing à travers l'utilisation de graphe de connaissance

Afin d'adopter les techniques d'ecommerce et d'augmenter leurs revenus, certaines compagnies aériennes utilisent le système de notification 'Amadeus Anytime Merchandizing' (AAM[17]), une solution informatique qui permet aux spécialistes du marketing des compagnies aériennes de définir, déployer, contrôler et ajuster efficacement les campagnes de marketing par courrier électronique envoyées aux voyageurs en temps réel. Des notifications personnalisées peuvent être définies et envoyées aux voyageurs, après la réservation d'un vol, pour leur suggérer d'acheter des services supplémentaires (par exemple, des bagages supplémentaires, un repas spécifique, un siège préféré). La solution fait office de passerelle entre les points de contact de vente au détail des voyageurs et le système de service et de livraison de la compagnie aérienne. Comme le montre la partie gauche de la figure 8, lorsqu'il utilise cette solution, le responsable marketing de la compagnie aérienne peut choisir le moment approprié (**quand**) pour envoyer la notification (par exemple, 5 jours avant le départ), **quel** produit recommander (par exemple, un siège pour les jambes), **comment** envoyer l'offre (par exemple, via un e-mail) et à **qui** cette offre doit être envoyée (en fonction des critères de ciblage).

Malgré toutes les fonctionnalités incluses dans le système de notification AAM, il est difficile pour une compagnie aérienne de trouver le public coorect à cibler pour une offre donnée. Nous avons effectué une analyse des ventes historiques déclenchées par certaines campagnes de notification au cours de la période du 14 mai 2019 au 17 déc 2019 menées par l'une de nos compagnies aériennes partenaires et nous avons observé une faible conversion des offres de notification (c.f. [28]). Cela s'explique en partie par le processus décisionnel difficile auquel est confronté un spécialiste du marketing d'une compagnie aérienne lorsqu'il s'agit de décider quelles valeurs (appartenant à de larges intervalles) sont appropriées pour les critères à utiliser (par exemple, l'heure d'envoi, les itinéraires de vol, le point de départ du vol, etc.) Le ciblage des clients par des notifications non sollicitées peut être contre-productif et avoir des effets négatifs sur la fidélité des clients s'il est mal fait. Il est donc essentiel d'identifier les clients dont on attend une réaction positive à un service annoncé, afin d'éviter de les spammer avec des e-mails non personnalisés. Ce problème peut être considéré comme un scénario de recommandation inverse, c'est-à-dire la recommandation d'un utilisateur à un article.

Inspirés par des travaux récents qui ont illustré l'efficacité de l'utilisation de plongements issus de graphes de connaissances [113, 115, 134] et d'algorithmes de boosting de gradient [68, 130] pour la recommandation, nous proposons '**T**ravel **K**nowledge Graph **E**mbeddings for email marketing campaigns' (TKE [18]) pour mieux cibler l'audience d'un service que la compagnie aérienne souhaite recommander par le biais de campagnes d'email marketing (figure 8). Dans

[17]AAM:https://amadeus.com/en/portfolio/airlines/anytime-merchandising
[18]TKE:https://gitlab.eurecom.fr/amadeus/tke4rec

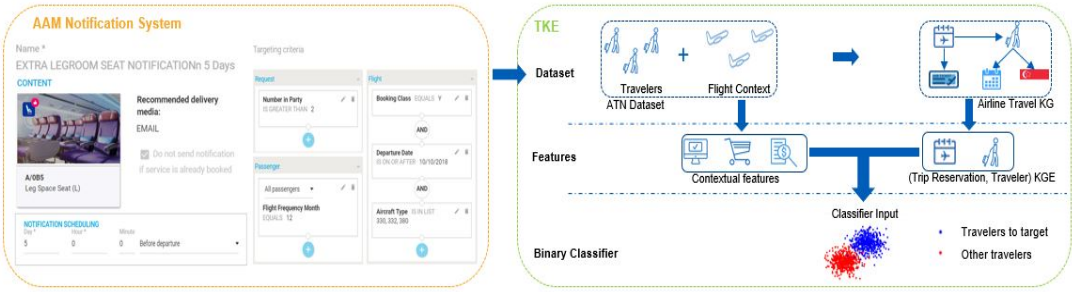

Figure 8 – Sur le côté gauche : Système de notification AAM. Sur le côté droit : Organigramme du modèle TKE que nous proposons. L'ensemble de données de notification utilisé dans cette étude est généré par le système de notification AAM. Les caractéristiques contextuelles comprennent le contexte de réservation (par exemple, le nombre de passagers, la date de départ, etc.), les informations de notification (par exemple, le média utilisé pour envoyer la notification, l'heure de la notification, etc.

ce travail de recherche, nous avons apporté les contributions suivantes :

1. Nous concevons et développons un graphe de connaissances en utilisant les technologies du Web sémantique pour représenter les voyages passés des voyageurs ainsi que pour enrichir sémantiquement les produits des compagnies aériennes.
2. Nous apprenons les représentations vectorielles des entités de voyage via des algorithmes de plongement de graphe de connaissances et nous tirons parti des algorithmes de boosting de gradient pour calculer les scores de prédiction afin de mieux cibler le public dans les campagnes de marketing par e-mail.
3. Nous effectuons une comparaison empirique de notre approche avec le système actuel basé sur des règles en production ainsi qu'avec une approche hypothétique d'apprentissage automatique classique utilisant des caractéristiques fabriquées à la main sur un ensemble de données de production du monde réel.

A.5.1 Problématique

Étant donné une campagne de notification destinée à un large public de voyageurs ayant déjà réservé un vol dans un contexte donné, nous cherchons à cibler les voyageurs pertinents parmi tous les voyageurs que les notifications vont atteindre. Plus précisément, nous répondons aux questions de recherche suivantes :

1. Comment extraire l'échantillon pertinent de voyageurs à cibler pour une campagne de notification donnée ? (Figure 9).
2. Quelles sont les performances d'une approche d'apprentissage automatique supervisée par rapport à une approche basée sur des règles pour cibler le public pertinent pour une campagne de notification ?

3. Comment l'utilisation de graphe de connaissanceembeddings se compare-t-elle à l'utilisation de caractéristiques artisanales comme entrée d'un modèle d'apprentissage automatique supervisé entraîné à cibler le public pertinent pour une campagne de notification ?

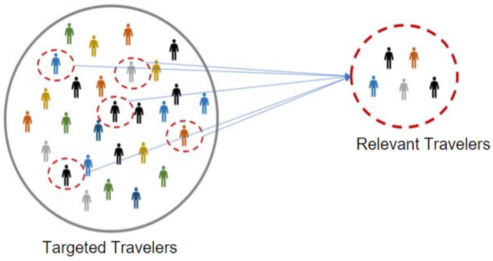

Figure 9 – La tâche consiste à extraire les voyageurs pertinents parmi l'ensemble des voyageurs qui ont été initialement ciblés par la campagne de notification via le système de notification AAM.

Dans notre travail, nous nous concentrons sur l'optimisation du taux de conversion :

Definition A.5.1 () *Nous définissons le taux de conversion d'une campagne de notification comme suit :*

$$CR = \frac{1}{N_o} \sum_{i=1}^{N_o} hit(N_i) \tag{3}$$

où N_o est le nombre de notifications envoyées par la campagne de notification, et $hit(N_i)$ est égal à 1 si la notification N_i déclenche un achat.

A.5.2 Jeu de données et construction du graphe de connaissance

Nous avons mené des expériences sur le même ensemble de données utilisés pour le cas d'usage de recommandation de destinations présenté lors de la section précédente: T-DNAT-DNA : Traveler DNA est une base de données qui contient les réservations de voyageurs sur une douzaine de compagnies aériennes. Cependant, nous avons utilisé un plus grand nombre de données pour convrir l'ensemble de la période d'envoi des notifications : L'ensemble de données considéré contient environ 2,33 millions de réservations pour environ 2,85 millions de voyageurs uniques.

L'ensemble de données sur les notifications de voyage des compagnies aériennes (ATN) est produit en joignant l'ensemble de données sur les notifications et l'ensemble de données historiques sur les réservations de T-DNA. Cet ensemble de données contient des informations sur le contexte des achats et des réservations (par exemple, la date de recherche, le nombre de passagers, la date de départ, etc.) et des informations sur les voyageurs (par exemple, des données démographiques et des informations sur l'adhésion à un programme de fidélité). Au total, le jeu de données contient 42 colonnes et ~ 8,2 millions de lignes. Pour nos expériences, le jeu de données a été divisé en trois sous-jeux de données différents correspondant aux trois campagnes de notification (Tableau 4.8).

Airline Travel Knowledge Graph Le graphe de connaissance est construit à partir de la base de données T-DNA. Nous développons une ontologie qui est disponible au format Turtle sur https://gitlab.eurecom.fr/amadeus/tke4rec/-/blob/master/ontology/ams_ontology.ttl. Pour concevoir le KG, nous avons défini 7 classes correspondant à des entités de haut niveau et basées sur les différentes tables disponibles dans la base de données T-DNA :

- **Traveler** : Un voyageur est identifié par un identifiant T-DNA. Un voyageur a un historique de réservation (PNRs) qui contient un historique d'achat (billets d'avion, billets EMD). Une instance de voyageur est une `schema:Person`[19].
- **Trip Reservation** : Une réservation de voyage (PNR) représente la réservation de tous les voyageurs contenus dans le PNR. Elle contient des informations telles que le nombre de voyageurs, la destination, etc.
- **Journey** : Un voyage est lié à une réservation de voyage. Chaque voyage a une durée de séjour, des aéroports de départ et d'arrivée.
- **Air Ticket** : Un billet d'avion est contenu dans un PNR et contient des informations sur le vol et les transactions. Un PNR peut avoir plusieurs billets d'avion en raison des différentes étapes du vol (par exemple, Nice-Paris, Paris-New York) ou/et du nombre de passagers.
- **EMD Ticket** : Un billet EMD (Electronic Miscellaneous Document) est lié à un billet d'avion. Il contient des informations sur les prestations accessoires achetées par le voyageur (par exemple, le type de prestations accessoires, le prix des prestations accessoires, etc.)
- **Ancillary** : Un 'ancillary' est un service acheté par un voyageur (associé à un vol) en plus du billet d'avion. Il est identifié par un sous-code (RFISC), étiqueté par un nom commercial, défini par ATPCO[20]. Il appartient à un groupe d'ancillaires (Groupe, RFIC). Nous proposons de modéliser les différents ancillaires comme des concepts SKOS et nous créons un thésaurus d'ancillaires représenté comme un schéma de concepts.
- **Airport** : Il représente l'aéroport où le voyageur se rend. Un aéroport dessert une ou plusieurs villes.

Le graphe de connaissance utilisé pour aborder notre cas d'utilisation contient 41 propriétés différentes (c.f. figure 10), ~ 80 millions d'arêtes et ~ 9 millions de noeuds.

Nous présentons dans le tableau 2 quelques statistiques sur le graphe de connaissance utilisés

[19]Le préfixe `schema` est utilisé pour les concepts définis par https://schema.org
[20]ATPCO Ancillary description : https://www.atpco.net/resource/optional-services-industry-sub-codes

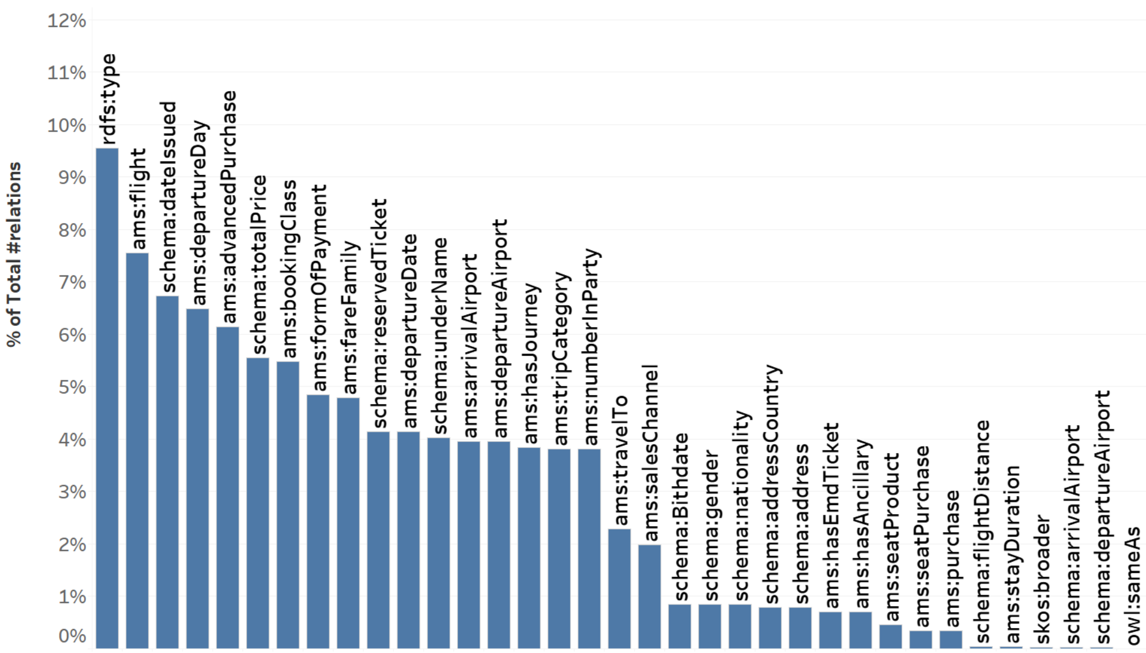

Figure 10 – Distribution du nombre de relations des propriétés dans le graphe de connaissances construit. Tous les préfixes se trouvent dans la définition de l'ontologie.

pour chaque cas d'usage.

Table 2 – Statistics of subgraphs

Subgraph	#Edges	#Nodes	#travelers	#PNRs
Extra leg room seat	7M	800K	67K	205K
Prepaid baggage	64M	7.6M	572K	2.2M
Lounge	6.7M	789K	42K	203K

Dans la Figure 11, un extrait du graphe de connaissanceest représenté, où un voyageur malaisien identifié par T21354, né le "1988-05-05" a réservé un vol aller simple pour deux personnes de Kuala Lumpur à Melbourne. Le billet EMD identifié par 23143 et lié au billet d'avion 21563 représente l'achat d'un accessoire (un siège).

A.5.3 Étude empirique du modèle TKE4Rec

L'objectif des expériences est de comparer l'utilisation de données calculés à l'aide de méthodes de 'features engineering' (a) avec l'utilisation de plongements issus de graphe de connaissance (b). (a) aide à interpréter les résultats et les prédictions obtenus par l'algorithme, tandis

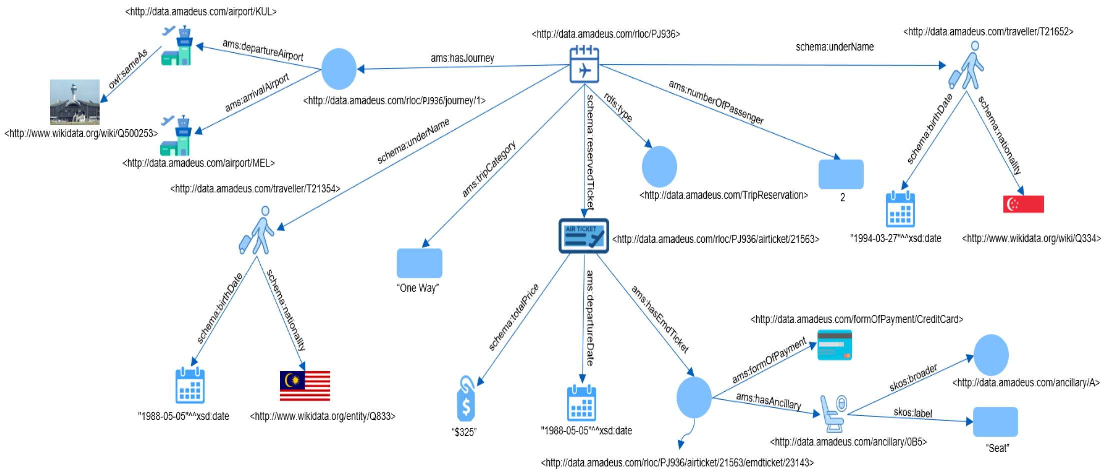

Figure 11 – Extrait du graphe de connaissances représentant les voyageurs inclus dans une réservation de Voyage à travers la propriété `schema:underName`, ainsi que d'autres propriétés et relations avec d'autres entités. Les littéraux sont représentés dans un rectangle bleu, tandis que les autres entités sont représentées dans un cercle bleu. Dans cette représentation, certaines propriétés qui relient les voyageurs, les réservations de voyage, les billets d'avion et les billets d'avion sont représentées à titre d'exemple, mais d'autres propriétés sont incluses dans le graphique.

que (b) manque d'interprétation (caractéristiques latentes), mais est plus facile à calculer et à maintenir. Nous publions notre code en source ouverte afin de faciliter la reproductibilité[21].

Données d'apprentissage et de test: Les trois ensembles de données correspondant aux trois campagnes de notification sont divisés selon la même stratégie. Chaque ensemble de données est trié dans le temps, et 80 % des premières lignes de chaque ensemble de données sont utilisées comme ensembles d'apprentissage/validation. Nous utilisons une validation croisée pour former et valider tous les modèles (k=5, une répartition de 80 % pour la formation et 20 % pour la validation). Les 20 % restants sont utilisés comme ensemble de test pour évaluer le modèle. La répartition entre l'ensemble d'apprentissage et l'ensemble de validation est effectuée de manière aléatoire afin d'éviter un effet de saisonnalité qui se produit généralement dans le secteur du voyage. Les algorithmes d'incorporation graphe de connaissance sont souvent conçus pour résoudre une tâche de prédiction de liens. Nous considérons qu'il est approprié de diviser le graphe de connaissance en supprimant certains liens qui sont inclus dans l'ensemble des propriétés qui relient les voyageurs aux services achetés et de les considérer comme des ensembles de test, afin d'évaluer la qualité des plongements obtenus.

Métriques d'évaluations: Le résultat de notre approche est la probabilité d'acheter le service

[21] https://gitlab.eurecom.fr/amadeus/tke4rec

Table 3 – Résultats de l'évaluation des différentes approches. (a) représente les résultats de XGBoost [19] pour différentes entrées ; (b) représente les résultats de l'approche TKE pour différents algorithmes d'incorporation KG. L'écart type moyen (en faisant varier la graine lors du fractionnement de l'ensemble de données) de chaque métrique est le suivant : $AUC-ROC$: $\pm 0,02$, TPR : $\pm 3\%$, TNR : $\pm 2\%$, CR : $\pm 0,1\%$.

Model	Extra leg room seat				Prepaid baggage				Lounge			
	AUC-ROC	TPR	TNR	CR	AUC-ROC	TPR	TNR	CR	AUC-ROC	TPR	TNR	CR
Rule-based	-	-	-	0.8%	-	-	-	0.15%	-	-	-	0.03%
(a) ADS	0.75	78%	58%	2.2%	0.83	80%	71%	0.38%	0.76	80%	62%	0.18%
(a) HDS	0.79	81%	60%	2.37%	0.85	82%	74%	0.4%	0.84	86%	67%	0.22%
(a) AHDS	0.83	85%	65%	2.8%	0.88	86%	**74**%	0.56%	0.89	88%	65%	0.36%
(b) TransE	0.85	86%	69%	3.1%	0.91	92%	65%	0.6%	0.90	89%	78%	0.35%
(b) TransH	0.84	85%	67%	3%	0.90	91%	65%	0.59%	**0.95**	**96%**	**85%**	**0.59**%
(b) TransR	0.84	85%	67%	2.9%	0.90	91%	65%	0.6%	0.92	92%	80%	0.52%
(b) MLP	**0.87**	**88%**	**69%**	**3.2%**	**0.92**	**94%**	65%	**0.62%**	0.91	90%	81%	0.56%

recommandé a inclus dans la notification N :

$$P(achat = a|N) = P(achat|Contexte, TE, RE) \tag{4}$$

où, TE et RE sont les plongements de réservation du voyageur et de la réservation.

Pour évaluer et comparer, les différentes approches mises en oeuvre, nous avons utilisé le taux de conversion défini à la définition A.5.1 et les trois métriques définies comme suit :

- _textbfTPR : Le taux de vrais positifs est le pourcentage de prédictions positives correctes. Il représente le ratio de voyageurs que l'algorithme suggère d'envoyer la notification et d'acheter effectivement le service. Le TPR est défini comme suit :

$$TPR = \frac{TP}{(TP + FN)} \tag{5}$$

- **TNR** : Le taux de vrais négatifs est le pourcentage de prédictions négatives correctes. Il représente le ratio de voyageurs à qui l'algorithme suggère de ne pas envoyer la notification et qui n'achètent effectivement pas le service. Le TNR est défini comme suit :

$$TNR = \frac{TN}{(TN + FP)} \tag{6}$$

- **ROC-AUC** : L'aire sous la courbe ROC (FPR, TPR) aide à choisir le seuil de probabilité optimal qui maximise le CR et le TPR et est définie comme suit :

$$ROC\text{-}AUC = \int_0^1 TPR\,;d(FPR) \tag{7}$$

  où, $FPR = 1 - TPR$ est le taux de faux positifs

Il est à noter que le taux de conversion a été mesuré hors ligne ainsi que toutes les métriques basées sur l'ensemble de test. Selon l'équation 3, N_o représente le nombre de prédictions positives et chaque hit hit_i correspond à une prédiction positive vraie.

Résultats et Discussion

Dans cette section, nous discutons des résultats obtenus lors des expériences. Les résultats des expériences menées sont présentés dans le tableau 3. Les mesures TPR, TNR et ROC-AUC ne sont pas fournies pour l'approche basée sur les règles mise en oeuvre dans le système de notification AAM. La raison en est que l'ensemble de données utilisé dans les expériences est généré par le système de notification AAM, ce qui est différent de l'ensemble de données original qui contient tous les voyageurs utilisés pour l'approche à base de règles afin d'identifier les voyageurs correspondant aux critères de ciblage.

Étude d'ablation: Le tableau 3 montre que l'utilisation des caractéristiques de l'ensemble de données ATN en plus des caractéristiques créées par les voyageurs (AHDS : ATN + Handcrafted features) en tant qu'entrée de XGBoost donne de meilleurs résultats que l'utilisation d'une seule d'entre elles (ADS : ATN features ou HDS : Handcrafted features) en tant qu'entrée pour toutes les campagnes de notification. Nous observons également que l'utilisation des caractéristiques artisanales des voyageurs comme information d'entrée de XGBoost donne de meilleurs résultats que l'utilisation de l'ensemble des données ATN pour toutes les campagnes de notification. Nous calculons les caractéristiques les plus importantes du modèle (a) AHDS pour chaque campagne de notification et nous présentons ci-dessous les trois plus importantes avec leur gain d'information respectif :

- Extra Leg Room Seat: {*Preferred Seat Characteristic*: 0.31, *Preferred ancillary*: 0.12, *Ticket amount*: 0.08}.
- Prepaid Baggage: {*Preferred destination*: 0.21, *Destination*: 0.12, *Prepaid Baggage sales Frequency*: 0.10}.
- Lounge: {*Average Flight Revenue*: 0.22, *Destination*: 0.20, *Age*: 0.15}.

Plongements graphe de connaissance: Nous observons dans le Tableau 3 que l'utilisation d'embeddings graphe de connaissance(concaténation d'embeddings graphe de connaissancede voyageurs et de réservations) avec des caractéristiques contextuelles en entrée de

XGBoost est plus performante que l'utilisation de caractéristiques artisanales de voyageurs, quel que soit l'algorithme utilisé pour calculer les embeddings ou la campagne de notification. De plus, les incorporations graphe de connaissancecalculées à partir du modèle **MLP** s'avèrent plus performantes que les incorporations graphe de connaissancecalculées à partir des modèles de distance translationnelle, sauf pour la campagne de notification des salons, où l'utilisation des incorporations graphe de connaissancecalculées à partir du modèle **TransH** donne les meilleurs résultats.

A.5.4 Conclusion

Dans ce travail, nous avons présenté une approche en deux étapes pour aborder le problème du ciblage de l'audience pour les campagnes de marketing par courriel : premièrement, nous calculons les plongements des voyageurs et des réservations issus de graphe de connaissance ; deuxièmement, nous utilisons ces encastrements en plus des caractéristiques contextuelles comme entrée d'un classificateur XGBoost pour apprendre quelle est l'audience pertinente à cibler pour une campagne de notification donnée. Nous avons mené plusieurs expériences pour répondre à nos questions de recherche :

1. Les résultats des expériences présentés dans le tableau 3 montrent que l'extraction de l'audience pertinente pour une campagne de notification donnée n'est pas une tâche facile. En effet, malgré le fait que le taux de conversion augmente significativement avec notre approche, il reste relativement faible. Cependant, grâce à notre approche, les campagnes de notification sont mieux ciblées et nous parvenons à éviter de recommander un service inadapté à au moins 65% des passagers.
2. Les expériences ont montré que l'approche d'apprentissage automatique supervisé (a) basée sur des caractéristiques artisanales donne de meilleurs résultats que l'approche basée sur des règles. En effet, dans le tableau 3, nous pouvons observer que le taux de conversion est multiplié par plus de 3 pour le siège avec espace pour les jambes, par presque 4 pour les bagages prépayés et par 12 pour le salon. Ainsi, nous prouvons l'avantage d'utiliser l'apprentissage automatique supervisé par rapport à une approche plus simple basée sur des règles, alors qu'il s'agit du mécanisme actuellement adopté par les spécialistes du marketing des compagnies aériennes. Il convient de noter que la liste des critères possibles disponibles dans le système de notification AAM (Figure 8) est la même que la liste des caractéristiques utilisées dans l'approche d'apprentissage automatique supervisé.
3. Les expériences montrent que, quel que soit l'algorithme d'incorporation graphe de connaissancetesté, l'approche d'incorporation graphe de connaissanceest meilleure que l'approche des caractéristiques artisanales. Ceci est très intéressant d'un point de vue scientifique, car cela montre la valeur ajoutée d'avoir un graphe de connaissancedans le domaine du voyage qui pourrait être utilisé non seulement pour la tâche de recomman-

dation auxiliaire, mais aussi pour d'autres tâches de recommandation (par exemple, la recommandation de voyage) car les mêmes graphe de connaissanceembeddings pourraient être utilisés. Il est intéressant de noter que lorsqu'il s'agit d'un problème de démarrage à froid (nouvel utilisateur ou nouvel élément) pour la convivialité en ligne, une approche basée sur des règles est plus appropriée.

Enfin, dans le cadre de travaux futurs, nous prévoyons de nous attaquer à la tâche de classement auxiliaire personnalisé dans les campagnes de marketing par courriel. Plus précisément, l'objectif de nos travaux futurs sera de répondre à la question de savoir quel est le service le plus approprié à recommander dans une campagne de notification. En plus d'aborder et d'optimiser ce qu'il faut recommander à un voyageur, il serait intéressant d'optimiser le moment où il faut envoyer la notification car c'est un facteur de décision important, surtout dans l'industrie du voyage aérien [25].

A.6 Conclusion

Inspirés par le nouveau flux de distribution des offres (NDC) introduit par l'organisation IATA, nous avons abordé dans cette thèse un ensemble de défis de recherche liés aux systèmes de recommandation appliqués à l'industrie du voyage aérien. L'objectif du NDC est de permettre aux compagnies aériennes de vendre leurs produits plus facilement en créant des offres plus personnalisées grâce à un contrôle complet du flux de distribution des offres qui permet la personnalisation et la contextualisation des offres des compagnies aériennes, créant ainsi une meilleure expérience de voyage pour leurs clients.

Cependant, les caractéristiques particulières de l'industrie aérienne comparées à celles d'autres industries très matures dans l'utilisation des systèmes de recommandation, telles que les industries du divertissement ou du commerce électronique, font du développement des systèmes de recommandation dans ce domaine un véritable défi et les raisons en sont multiples : Premièrement, les données sont très rares dans le domaine du voyage en général, deuxièmement, en raison de la façon dont le système de réservation des compagnies aériennes fonctionne (plusieurs plates-formes de réservation différentes sont possibles sans qu'aucune identification de l'utilisateur ne soit requise), nous avons beaucoup de nouveaux voyageurs dans le domaine du voyage, ce qui constitue un problème de démarrage à froid, troisièmement, le manque d'application de la science des données dans le domaine des compagnies aériennes et surtout le très petit nombre de systèmes de recommandation développés dans ce domaine font de la collecte de données utiles et nécessaires au développement de systèmes de recommandation une difficulté en soi.

D'une part, cela nous amène à chercher un moyen d'enrichir et de peupler nos données par d'autres sources pour surmonter ces deux problèmes de sparsité et de démarrage à froid. Les données sémantiques sont utilisées à cet effet afin d'enrichir sémantiquement nos données et d'apporter une certaine classification et une structure cohérente bien définie au niveau logique et sémantique à travers la définition d'une ontologie. L'incorporation des données sémantiques et des données des compagnies aériennes (par exemple, les interactions de voyage, le contexte de voyage, etc.) dans une seule structure de données (graphe de connaissances) s'est avérée très précieuse comme source de données pour les algorithmes des systèmes de recommandation afin d'effectuer des prédictions, comme nous le montrons dans les sections précédentes (c.f. section A.4) et A.5.

D'autre part, les systèmes de recommandation basés sur les graphes de connaissances se sont avérés efficaces pour gérer la sparsité des données et le problème du démarrage à froid comme montré dans [27]. Plus précisément, les systèmes de recommandation basés sur les graphes de connaissances présentent trois avantages principaux : (1) graphe de connaissanceincorpore des informations hétérogènes provenant de différentes sources de données grâce à l'utilisation de relations de différents types, améliorant ainsi l'intégration et l'augmentation des données pour l'utilisation de l'apprentissage automatique et évitant la lourde tâche d'ingénierie des caractéristiques nécessaire pour améliorer la précision des systèmes de recommandation ; (2) graphe de connaissanceintroduit une relation sémantique entre les éléments, ce qui peut aider à trouver leurs connexions latentes et à améliorer la précision des éléments recommandés ; (3) graphe de connaissancecontient des informations sur l'ensemble du voyage du voyageur, de l'inspiration au départ du vol, ce qui en fait une ressource unifiée pouvant servir d'entrée à tous les cas d'utilisation de la recommandation qui couvrent l'ensemble du voyage du voyageur.