Leveraging Knowledge-Graph representation for RAG-based copiloting in the VC context

Student: Matteo Spadaccia - S2748897

Supervisor: Prof. Filippo Menolascina

Tutors: Dr. Shadullah Sartaj,

Dr. Muhammad Zain Bin Aamir

Master of Science

Data Science

School of Informatics

University of Edinburgh

2025

Abstract

In the increasingly competitive startup ecosystem, effective matchmaking between early-stage ventures and investors is critical for securing thoughtful funding and maximizing innovation potential. This project aspires to enhance the current systems that facilitate this combination process, by bolstering the navigability of the partnerships realm on the startups' side. Leveraging Artificial Intelligence (AI) and graph analysis, models with this aim represent historical funding patterns and the involved players' relationships, usually in a network-form able to capture the domain's interconnected nature.

Specifically, the present study is built upon the algorithmic framework of Doriot AI, an emerging company focused on supporting startups' fundraising journeys: experimenting a Knowledge-Graph (KG) representation of the Venture Capital (VC) context, as extrapolated by the firm's core DataBase (DB), the work aims at improving the performance of Doriot AI's Retrieval Augmented Generative (RAG) model for investment environment exploration. The ultimate goals are to convert the underlying information-set to a directly exploitable network format, developing a robust graph-embedding pipeline, and thus to lay the foundations for a higher-level Question-Answering (QA) system, while drawing the directions for future developments. In fact, the so-obtained KG-DB opens great and incremental possibilities to upgrade this service and the other Doriot AI's main one - a recommender model to uncover plausible funders - therefore pushing the boundaries of Machine Learning (ML) methods oriented to empowering startups with better copiloting tools, rather than to assisting investors' decision-making.

Research Ethics Approval

This project was planned in accordance with the Informatics Research Ethics (IRE) policy. It did not involve any aspects that required approval from the IRE committee.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Student: Matteo Spadaccia - S2748897*
Supervisor: Prof. Filippo Menolascina
Tutors: Dr. Shadullah Sartaj,
Dr. Muhammad Zain Bin Aamir)

Acknowledgements

To my friends, especially those who - from near and far - coloured Edinburgh with vivid emotions, daily matching in a whiff of year the recovered taste for a dynamical clique. To my family, for actively nurturing curiosity and depth, while gently allowing the space to evaluate what such a pursuit may ever offer.

Table of Contents

Chapter 1

Introduction

Constituting crucial sources of innovation with extensive societal impact, startups offer significant wealth opportunities, with Venture Capital (VC) investment returns reaching even 10,000% - totally incomparable to typical saving accounts' or standard investments' interest rates [69]. In fact, ever increasing sums have been allocated in this domain during the last 20 years, with a constantly high rate of ventures' foundation globally [28]. However, given the almost steady number of deals concluded in this field, an asymmetry appears evident between the amount of deployable capital and the number of available investment opportunities: a VC firm inevitably receives thousands of pitch decks to review yearly, forcing an immediate refusal in over 90% of the cases [40]. In this setting, it is crucial for startup founding teams to target the right investors, and thoroughly gauge the contextual funding and industry ecosystem, in order to reduce the volume of unworthy interactions and autonomously refine their own business-model, while accurately ensuring that potential investors and launching-partners are reached.

1.1 Promising Benefits

By incrementing the efficiency of investors-ventures matchmaking procedures by targeted funders pre-selection, both startups and VC entities might benefit; considerably lower efforts would indeed be necessary to both parties if the investment relations were mediated through a specific platform and the compatibility assessment aided via automated recommendations and explorative AI-systems. Besides facilitating the job of the actors involved, a powerful suite of copiloting services over the funding framework would indirectly have positive effects in disparate domains, bolstering the allocation of economic and expertise supports to impacting ventures across sectors [15].

1.1.1 Potential Solutions

To effectively operate a wise pre-selection over the different potential investors, it is fundamental to firstly consider why each VC firm has a different probability to fund a specific startup: as always mirrored by high returns, the uncertainty in this setting is very high, such that having a relevant know-how becomes central for the investor to discriminate actually promising ventures. In other words, Venture Capitalists are prone to bet on startups development forecasts that convey them informed and particularly positive expectations, intensively basing the selection on their skills set, and preferring initiatives of which they have full understanding, sometimes offering a sort of financial mentoring itself [109]. This interaction rationale is well depictable by conceiving the already correlated ventures and investors as mutually identifying their characteristics; this is nowadays achievable only by considering a large amount of past funding data as a whole, and leveraging a network representation to train AI recommender systems [3].

On the other hand, beyond algorithms narrowly focused on optimizing investors–startups matches, growing demand is flowing on solutions that enable smart and structured exploration of the same funding patterns and the underlying business-networks of interest. Software-services are on the rise aiming at revealing connections across industry domains, market segments, and even educational or professional backgrounds, providing not only actionable insights, but also a fundamentally explainable perspectives on these interconnected realities. However, these knowledge-driven retrieval and reasoning models stand as complementary tools to recommendation engines, offering transparency and contextual understanding that enhance trust in the decision-making process.

1.1.2 Timeliness

Collaborating with the startups' world through the Venture Builder Incubator and piloted in the Venture Builder Accelerator at Edinburgh Innovations [21], Doriot AI was founded in 2024 with the aim to bridge the gap between ventures and investors. The ideation started in 2022: just a few years ago, startups were observed struggling to firstly connect with investors who truly understood their vision, shared their values and presented complementary strengths [19]. The initiative has been successful in innovatively approaching the underlying asymmetry from the viewpoint of new businesses; in fact, Doriot AI was created to simplify the capital-retrieval and networking process by empowering novel founders to align with supportive partners, rather than to improve the subsequent entrepreneurs' selection.

Around the latter a vast majority of the current academic literature revolves too (section 1.3), as also mirrored by the almost absolute uniqueness of Doriot AI's endeavour. The company is indeed offering AI-powered development tools for startups globally, with the ultimate objective of supporting the entire fundraising journey through a single platform. Analysing extensive data, Doriot AI provides valuable market insights tailored to the specific businesses, and proposes customised founders-investors interactions based on past transactions [19]. However, the company is attempting the composition of a full suite of ventures-oriented services: besides a core investors-to-ventures matchmaking engine, based on state-of-the-art Machine Learning (ML) algorithms and already in production, a Natural-Language Question-Answering system is under development. Being of a Retrieval Augmented Generation kind, this model was brought to embryonic stage by following the recent fervour on this relatively new field. So, willing to fruitfully perpetuate its inherent perspective-subversion effort, Doriot AI is once-again aiming at enhancing the level of copiloting services offered to startups, since similar Question Answering (QA) solutions are currently oriented mainly to investors instead. These are also directly targeted by service prices that could only be offset by an overall high-profit, and thus prohibitive for any early-stage venture.

1.2 Objectives

The principal goal of the present project is exactly producing advancements to the algorithmic framework of Doriot AI, and specifically to its RAG system for Natural-Language scouting of the fundraising environment, which leverage a Large Language Model (LLM) to encapsulate the information conglomeration in a user-friendly Question Answering fashion. At the base of the company digital services, a rich relational database lays, great part of which was acquired through the partnership with Noumena Capital [55]. The latter organization allows access to one of the most comprehensive VC transactions datasets, and provides the underlying investor-side domain knowledge. However, diverse queries are necessary on the composite database to extract information, reaching considerable complexity depending on the insight level, and demonstrating the tabular encode physiologically inefficient in representing network contexts.

1.2.1 Plan

In fact, the focus of the work would be to convert Doriot AI's data foundations into Knowledge-Graph (KG) format, and then to gauge the advancement achievable by basing the aforementioned AI models on this new informational structure. Specifically, the relevant literature would be vastly gauged (section 1.3) to gather the conceptual and technical bases useful for comprehensively framing the main task; then Doriot AI's case would be analysed under this lens, its information-set preliminarily explored and the Knowledge-Graph built following a promising architecture (section 2.1). Successively, the RAG-model would be experimented in a modified - and debugged - version, aligning its informative rationale to the newly-composed graph-DB (section 2.2), progressively leveraging this embedding type to enhance the data-extraction and answering performances under specific metrics (section 2.3).

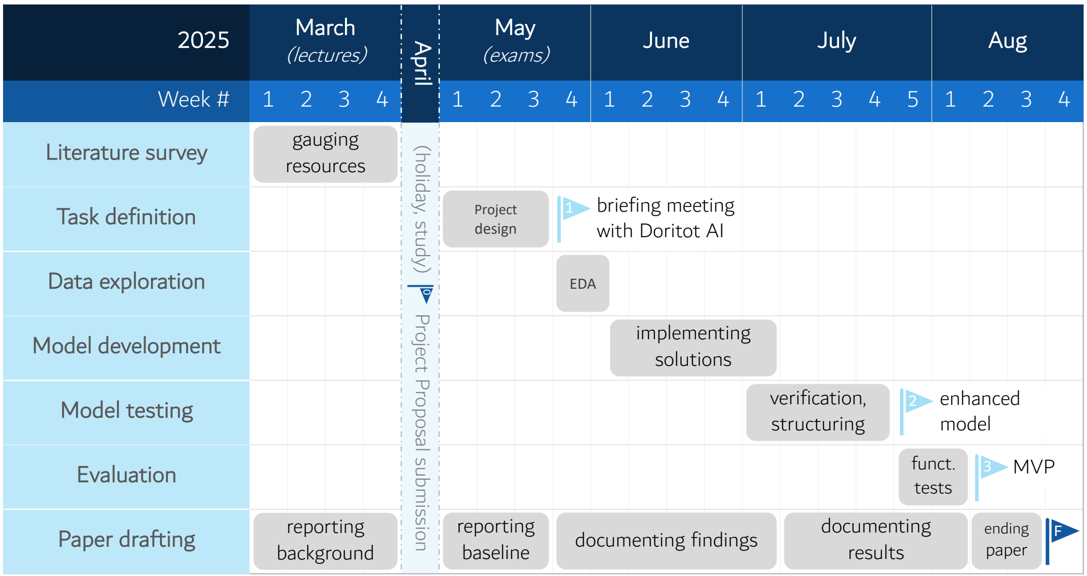

Figure 1.1: Gantt-chart project plan with deliverables

Considering the fundamentally result-oriented fashion of the project, its detailed plan is shown above (fig. 1.1) and the relevant working principles reported hereafter (section 1.2.2). As largely reiterated (section 2.2.1 and chapter 3), the work would indeed not be conceived as a formal experiment to compare the two data-embedding and retreival strategies, but would rather prioritize to pragmatically bring valuable advances to Doriot AI's data-architecture and software-products suites; the extent to which these goals are attained would thus be scientifically quantify through the same proper theoretical framework leveraged to hypothesize and apply the promising developments. Despite the expectation being to obtain a Minimum Viable Product

(MVP) of the desirably enhanced service, a great space would be given to the successive steps recommendation (section 4.1): complex RAG systems of this genre require continuous development to stay performant even at stable production phase [73], and the model was consigned for upgrade in a very incipient state.

1.2.2 Principles

Considering the ethics guidelines for research of the University of Edinburgh (UoE) [95], no procedure in the project at issue had to undergo approval by the relevant committee [93]; however, the conformity of any gradually scheduled step in the same study was promptly assessed, so to keep it in accordance with the aforementioned general directives. In parallel, the specific ethics guidelines of the collaborating companies were enforced too, ensuring that the moral code of each involved party was respected. Specifically, the privacy of the data subjects and the related ownership regulations were directly respected, and any information was securely stored and processed following the associated rulings. It was thus of crucial importance the concious adoption of all the basic cybersecurity[1] measures, prioritising the storage of data and code in the relevant owners' repository; any black-swan event[2] would have had mitigated effects with all the suitable legal directions properly applied. Despite the related computations are considered not to be environmentally impacting[3], the algorithmic solutions were kept efficient under this viewpoint by following literature-valid guidelines [42].

[1]Protection against criminal or unauthorized use of electronic data. [99]
[2]Unforeseen event of serious consequence. [98]
[3]Causing a significative carbon footprint by running on any processor. [35]

1.3 Background

As expressed, the literature widely focuses on framing the recommendation problem from the investors' viewpoint (section 1.3.1), while far less academic work is available on VC suggestion to ventures for funders targeting (section 1.3.2), as reflected by the distinctiveness of Doriot AI's service. The latter leverages Graph Neural Network (GNN) methods layered on top of startup similarity graphs, integrated into a robust, venture-oriented recommendation pipeline (section 1.3.3). On the other hand, the available academic resources on RAG systems and Knowledge-Graph retrieval focus on Question-Answering tasks similar to the one at stake in data complexity and interconnected nature, presenting no proof of any state-of-the-art solution for the specific case of VC-context exploration (section 1.3.4). Overall, the usage of network-formatted DataBases is consistently supported, as all the model-types analysed hereafter are reported to perform better if the underlying information is directly shaped into graph-architectures.

1.3.1 VC investments allocation

Venture Capitalists have increasingly shifted from reliance on traditional heuristics to data-driven approaches for evaluating startups, through profitability forecasting and portfolio optimization [109]. Recent advancements emphasize the role of Machine Learning in capturing latent patterns within extensive financial and economic datasets, to improve predictive accuracy and reduce uncertainty in investment decisions [3]. Commonly used models in this supervised learning[4] context include Random Forests [7], Support Vector Machines [33], Gradient Boosting Machines [24], and Neural Networks [29]. These models are particularly valuable given the inherent 'outlier nature' often characterizing early-stage ventures and the respective information's sparsity. High-quality datasets featuring startup performance indicators are essential to support all these techniques [60]; moreover, feature engineering[5] has emerged as a critical step, especially in quantifying founder team's reliability and market responsiveness statistics.

Especially beneficial in early-stage investments, recommender systems for VC-startup matching employ hybrid approaches that combine content-based filtering with collaborative filtering. These methods integrate startups risk-reward profiling with historical investment patterns of similar VCs [47] to ensure recommendations align

[4]ML paradigm implying training on input objects and respective desired output values. [101]

[5]Preprocessing step in supervised ML modelling which transforms raw data into more effective input features. [100]

with the investor's strategic objectives, risk tolerance, and sector preferences [108]. In this domain, relational data are already widely leveraged in KG form [103], and recommendation systems are often embedded in mobile applications [4].

1.3.2 AI potential for startups

On the startup side, AI and network analysis present significant opportunities for identifying suitable funders. In particular, historical financial interactions and sectoral investment trends allow ventures to target investors with proven domain expertise and relevant portfolios. The matching process is often represented by bipartite graphs, where nodes denote startups and VCs, and edges encode relationships as transactions [26].

As an especially suitable method, network diffusion algorithms simulate indirect interactions and information propagation, enabling dynamic and context-aware investor recommendations tailored to both domain-specific interests and broader graph dynamics [104]. These models account for how startups' characteristics (e.g., business model, market, technology) permeate through the network and influence investors' preferences based on proximity and prior affiliations. Heterogeneous Information Networks (HINs) enhance this by incorporating diverse node types (e.g., technologies, markets) and weighted features that reflect domain complexity and facilitate more nuanced recommendations [102]. Furthermore, these multilayered graph structures integrate various interaction dimensions (e.g., geographical, industrial), enriching the recommendation with additional context. With such a structure, link prediction techniques are indeed employed to forecast the most probable future connections between startups and investors. On the other hand, hybrid filtering approaches are also capable of efficiently leveraging startups' characteristics and VCs' preferences directly, representing startups' profiles (e.g., development stage, technological focus, market position, funding needs) and investors' tendency in terms of industry-specific risk predisposition to yield sometimes refined-enough funding opportunities recommendations [48].

1.3.3 Doriot AI's solution

Framing investor-startup matchmaking as a link prediction problem over graph-modelled investment ecosystem (i.e., a bipartite graph, with nodes representing startups and investors, and edges reflecting funding interactions), Doriot AI firstly explored two complementary models grounded in Graph Neural Networks [70, 1]. The models leverage startup features, historical investment data, and graph-based embedding techniques to produce personalized investor recommendations from the startup's perspective. These GNN-based approaches reflect a broader trend in recommender systems: the use of non-Euclidean data structures (graphs) and relational inductive biases to model high-order dependencies and latent user-item interactions [5]. By embedding both investor and startup nodes into a shared latent space and predicting potential links via dot-product similarity, the models infer likely future funding relationships. Notably, the training objective mimics that of a binary classification task, where positive edges and negative samples are used to train the model to distinguish true matches.

The first model combines a startup-startup similarity graph built via K-Nearest Neighbors (KNN) with a GraphSAGE architecture. The latter is a scalable GNN method that constructs node embeddings by sampling and aggregating feature information from a node's local neighbourhood [30]. Its inductive learning ability allows for effective predictions on unseen nodes without retraining, which is especially valuable in cold-start startup scenarios. The method is particularly suited for recommendation systems due to its ability to process dynamic and large-scale graphs efficiently. The second model builds upon the first by integrating Graph Attention Networks (GATs), which enhance training by assigning learnable importance weights to neighbouring nodes during aggregation [96]. This attention mechanism allows the model to capture finer-grained dependencies, giving more weight to influential or semantically closer nodes during embedding propagation. The resulting startup-investor heterogeneous graph preserves the semantic richness of edge types and node features for use during recommendation.

In synthesis GraphSAGE provides scalability and strong inductive bias, while GAT introduces fine-grained attention mechanisms. Evaluation of both models was performed using recommendation-specific metrics such as Precision, Recall, Hit Rate, and Mean Average Precision (MAP) [71, 12]. KNN+GraphSAGE demonstrated higher MAP - 0.349 across varying startup subsets - reflecting consistent global ranking quality; on the other hand, KNN+GAT achieved higher Hit Rates - up to 35.42% at k=20 - indicating better top-k prediction performance, an important property when founders consider only a shortlist of investors for outreach. The lower F1-scores across both models are acknowledged as a by-product of severe class imbalance typical of investor-startup datasets, where the number of true links is dwarfed by non-links [17].

The underlying graph construction process is another key aspect of these systems: both models use a startup-side KNN graph constructed via cosine similarity of feature vectors, to thicken the graph structure and support more robust neighbourhood aggregation during message passing. This approach is inspired by prior work emphasizing the value of auxiliary graphs and side information in enhancing sparse recommendation environments [41, 10]. The heterogeneous formulation is also consistent with industrial GNN recommender frameworks like AliGraph and Uber Eats' model, which exploit social, temporal, and transactional relationships as multilayered edge types [110, 94].

1.3.4 KG-based RAG models

A Knowledge-Graph (KG) can be broadly defined as a structured representation of different types of entities and their semantic relationships (i.e., nodes and edges respectively), enriched with attributes that capture contextual information about both these vertices and links [34, 22]. The networked nature allows to model complex, heterogeneous domains, such as the Venture Capital ecosystem, where investors, startups, companies, individuals, and other entities might be inherently interlinked [91].

Possibly operating on top of such structures, Retrieval-Augmented Generation (RAG) models provide a hybrid architecture that combines symbolic retrieval with generative reasoning [43]. In a standard configuration (fig. 1.2), an Intent-Coordinator first processes the incoming natural language query to identify the user's purpose and map it to one or more retrieval types. These retrieval requests are then dispatched to a handlers-based retrieval-layer, where specialized modules extract relevant information from the underlying knowledge base according to entity type and query constraints. The retrieved data are finally passed to the LLM-Generator, which integrates them with the original query context to construct a coherent and contextually grounded response. This separation of intent identification, modular retrieval, and Natural-Language generation makes RAG systems particularly suitable for domains requiring both precision and explainability [25]: the KG ensures factual accuracy and relational depth, while the generative layer provides the flexibility of reasoning and answer NL-composition.

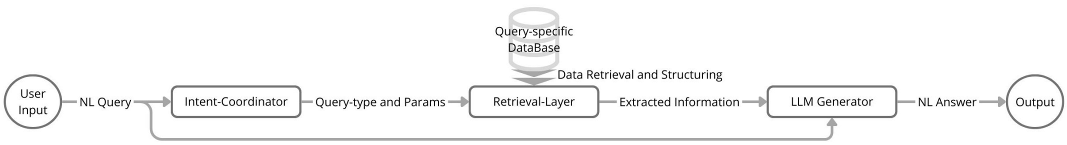

Figure 1.2: typical RAG-system's functioning scheme

Knowledge-Graphs have been demonstrated to have high performance in modelling investment environments and the Venture Capital context in particular [8, 103]; despite this, both Doriot AI's recommendation system and RAG model for data exploration leverage bipartite-graph-conceived tabularly stored information, making desirable a transition to a more complete network. In fact, both the models' precision might enhance by strategically embedding all the datasets into a single Knowledge Graph with a wider set of nodes, edges, and related attributes [44]. Moreover, the same structure is expected to reduce the information retrieval time once certain level of query complexity are reached, which is of substantial importance when the relevant services are grounded on a wide intertwining database and offered in parallel to many customers [45].

The field of graph-bolstered RAG is rising, with performances clearly superior with respect to traditional generative models [58]: effectively utilizing both structured and unstructured data, better results in domain-specific QA are seen over a variety of tasks [2]. However, traditional RAG models have insufficient processing efficiency when facing complex graph structure information, which affects the quality and consistency of the generated results; to solve this problem, Graph Neural Network (GNN) architectures have demonstrated salvific. Thanks to this type of background elaboration, KG-based RAG models can capture the complex relationship between entities, thereby reaching better performances in terms of quality, consistency, and reasoning ability, especially when dealing with tasks that require multi-dimensional reasoning in multiple practical scenarios [18]. Combining language understanding abilities of LLMs with the reasoning abilities of GNNs in a retrieval-augmented style, the leading strategy in the domain of Question Answering over Knowledge-Graphs (KG-QA) has indeed become the GNN-RAG [51], but other tasks in the same domain are under the spotlight too, as the opposite network-construction one [105].

Despite no direct state-of-the-art is proven for the specific task, a series of questions that a QA-copiloting RAG-system would be highly effective at answering are of startups' interest [26]: with the proper underlying information-set, potential investors' profiles could be retrieved among person- an organization-instances, as also competitors' specifics and networking opportunities in terms of similar-interests, compensating-skills, or strategically connecting individuals and events. Specifically, these models' capability to identify the own questioner's venture- or investor-profile would allow a more targeted retrieval of all the above analytical aspects [49].

Chapter 2

Methodology

As per plan (section 1.2.1) and leveraging the presented state-of-the-art (section 1.3), the available data is explored in its original structure, so to design and implement a Knowledge-Graph that efficiently embeds the whole information set (section 2.1). Subsequently, entailing the active interaction with all the relevant development layers, the RAG system is modified in order to largely base the retrieval on the just network-formatted data (section 2.2), and the outputs are evaluated under predefined metrics through a newly built testing-framework (section 2.3).

2.1 Data

A sufficiently large information-set is needed in order to reach satisfactory accuracy and pragmatically correspond to the task of comprehensive fundraising-network understanding; this necessity is specifically addressed by the wide scope of Doriot AI's dataset, vastly extracted from Crunchbase - a platform providing business and funding details about private and public companies [16] - and including information on organizations and persons as interacting in the investment context. However, for experimental purposes, a part of the company's DataBase is assigned to the present project, vastly representing Doriot AI's core information in the same structure as its complete version, but missing some features (i.e., organizations' and peoples' industrial-field allocations, and sets of task-related news-articles) which are actually recorded and preferentially kept in the firm's SQL-instance for this stage of KG-conversion experimentation. Nevertheless, the integration potential and method for these and additional information kinds are subsequently described in detail (section 4.1.1). Notably, the given core-DB is consigned as updated until January 2020.

2.1.1 Knowledge-Graph Building

The first objective is to produce a dataset architecture able to enhance the accuracy and contextual reasoning of Doriot AI's models by capturing rich relationships between entities and concepts that are not easily represented in flat SQL tables. Therefore, a scheme is designed to embedded the available core-DB into a Knowledge-Graph, directly accenting its inherent network-nature (fig. 2.1). To do so, an Exploratory Data Analysis (EDA) `.ipynb`[1] notebook is prepared, easing the preliminary study of the information-set as stored in multiple `.csv`[2] files; this lead to uncover its basic composition and referencing-system, and thus to fix the hypothesis necessary to plan a theoretically-promising graph's framework (e.g., redundancy of specific information across different tables, variability in value of certain features).

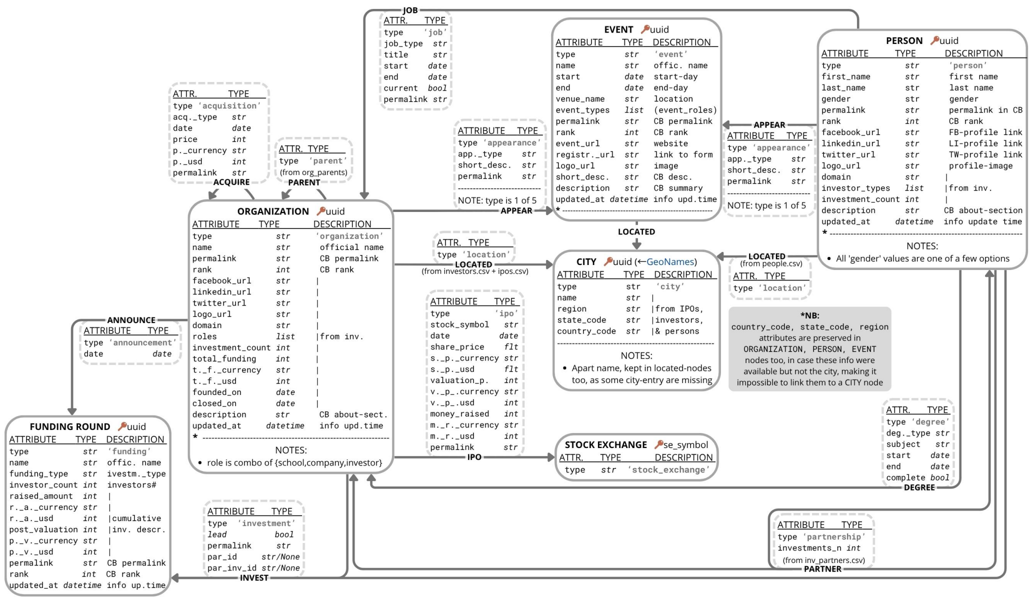

Figure 2.1: Knowledge-Graph structure design for data

In a similarly expandable fashion as the just described code, another `.ipynb` notebook is thus developed to effectively operate a robust KG-conversion process, so to ensure agile eventual additions to the data as translated in the new format (e.g., to enrich it with the aforementioned momentarily set-aside components of the company's data, or to interconnect additional conceptual entities). Specifically, the various nodes- and

[1] Format used by Jupyter Notebook to store code, text, and visualizations in JSON structure for interactive analysis and reproducible workflows. [38]

[2] Plain-text format for storing tabular information where each line represents a data record and fields are separated by commas. [36]

edges-types' instances are inserted into a `NetworkX.MultiDiGraph`[3] object, leveraging isolated functions to clean and extract just the task-significant features (e.g., ignoring the original instance-creation timestamps) from the `Pandas.DataFrame`[4]-loaded tables collection. Specialized sub-pipelines are leveraged to neatly handle people's name attributes - previously stored in mixed first-, last- and full-name versions across the tables - and coherently add location information by producing city-nodes with a GeoNames-operated ID-assignment [27].

All the relevant entries are thus translated with no information-loss into the designed network-framework, making the displayed diagram a meticulous representation of the new DB itself, reporting all the attributes for each type of entity and relation (fig. 2.1). The obtained multi-edge directed graph is then stored in `.pkl`[5] format, but during code execution a series of `.csv`-saved DataFrames are iteratively saved too, displaying for each kind of node and link all the recorded instances with their features, as indicated by a unique entity identifier - for vertices - or a couple of them - for edges. Interestingly, due to the wise architecture choice and the data cleansing operated, the latter tabular representation as a whole weights $0.89GB$ less than its original counterpart (i.e., $1.52GB$ and $2.41GB$ respectively), while the `.pkl`-serialized KG-DB takes slightly more space due to the overhead imposed by the `NetworkX` Python object: $1.63GB$ in total.

2.2 RAG model

As entrusted for the present project, Doriot AI's RAG model presents an Intent-Coordinator trained over huge collections of NL-queries (15960 instances between train-, valid- and test-sets), each associated with a ground-truth set of retrieval-kinds to be triggered - often more than one and up to three per query. The set of handlers corresponding to the latter extraction-types constitutes the retrieval-layer for this system; on this component, the conversion to KG-DB usage is focused (section 2.2.1). The original versions of some of these handlers' scripts and other code parts are then slightly modified too (section 2.2.2), but without changing their logic or debugging them, and with the mere aim to align their functioning with the designed evaluation procedure (section 2.3), for comparing the two model-versions' performances. A brief-description

[3] Data structure from the NetworkX Python library representing a directed graph that allows multiple parallel edges between node pairs. [54]

[4] Two-dimensional labeled data structure from the Pandas library, widely used for tabular data manipulation and analysis in Python. [56]

[5] Format used to serialize Python objects into byte streams via the pickle module for storage or transmission. [59]

of all the 24 query-types follows (table 2.1), with a 0-based indexing helping to identify them coherently throughout the handlers' upgrade and testing phases too.

Handler	**Query Target**
0-Acquisition	acquisition-deals and M&A-activities data
1-CompanyProfile	comprehensive company information and descriptions
2-Competitor	candidate competitor companies by similarity
3-Education	individuals' educational-backgrounds data
4-Event	information on events (e.g., conferences, meetings)
5-FundingNews	news-articles about funding events
6-FundingRound	specific funding-rounds' information
7-Funds	investment-funds entities' information and portfolio data
8-Greeting	conversational greetings based on user-profile
9-InvestmentDetails	specific investment-activities' information
10-InvestmentPartner	investment-partnerships and co-funding relationships
11-Investor	comprehensive investor-profiles and portfolio information
12-InvestorNetwork	investment-network mapping and co-investment patterns
13-Ipo	companies' Initial Public Offering (IPO) data
14-Jobs	individuals' employment-history and career-progression
15-LeadGeneration	potential business-leads and opportunities
16-MarketAnalysis	wide market-trends and sector analysis
17-MarketTrends	comprehensive market-trend insights
18-MyStartup	user-profile's company information
19-NewsBase	news-articles (general DB-interfacing handler)
20-OrganizationRelationship	corporate relationships and hierarchies mapping
21-PeopleProfile	comprehensive people information and descriptions
22-RecommendInvestors	candidate funders (pre-computed recommender's results)
23-TechNews	technology-focused news-articles

Table 2.1: handler's identification and respective query-target description

2.2.1 Retrieval-Layer Conversion

The so-built network-DataBase (section 2.1.1) is thus experimented for data-extraction as underlying knowledge-set for Doriot AI's RAG model, attempting a thorough refinement of its capabilities by switching to a graph-based retrieval-layer (fig. 2.2). This approach would enable more intelligent traversals and relevance scoring, improving both the extracted-information's and the generated-responses' quality [20].

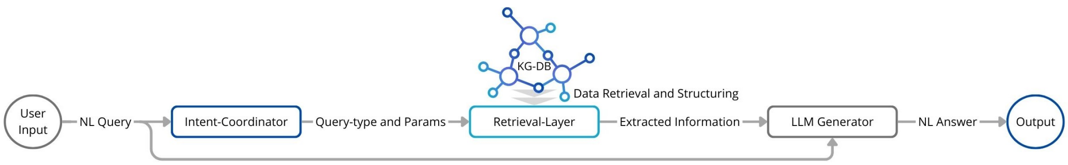

Figure 2.2: KG-based RAG-system's functioning scheme

Each of the previously tabular-based retrieval procedure is progressively refactored to a graph-extracting logic whenever possible; in fact, some features are still extracted by fallback on the original SQL database, since not available in the current Knowledge-Graph instance (e.g., for *2-Competitor*, *11-Investor*, *15-LeadGeneration*, and *16-MarketAnalysis*). In some of these cases (i.e., when the industrial-field or the recommended-investors are to be SQL-retrieved), alternative KG-only extraction strategies are also explored, to gauge the possibility of inferring similar categories from other elements of the graph and its structure (appendix A.2), with scarce results. However, as aforementioned and through the expandable network-building pipeline (section 2.1.1), most of the missing information (e.g., sectors of interest, news-articles) could be successively easily integrated as attributes of already present or additional node- and edge-types, augmenting the current graph framework (section 4.1.1).

Beyond the straightforward translation of retrieval-logic from SQL-tables to graph traversals, several handlers underwent deeper functional re-engineering, either to refine their retrieval rationale, improve accuracy, or extend filtering and ranking strategies (appendix A.1). In all cases, a deterministic multi-level ordering (e.g. temporal, by rank, or based on query-dependent relevance-metrics of some genre) was coherently implemented for intermediate and final results, ensuring a deterministic output of most-relevant instances that was not always granted by the original handlers. Investor-related handlers, crucial to the project's domain, incorporated multiple layers of refinement: the *9-InvestmentDetails* and *10-InvestmentPartner* handlers enriched partnership analysis through dedicated edge semantics, while *12-InvestorNetwork* moved entirely to graph traversal for network discovery, integrating multi-factor similarity scoring and co-investment ranking.

Event- and people-related retrieval also matured substantially: *4-Event* extended its interface to include participant-based filtering, date ranges, and appearance types, with optional delegation to the news subsystem, while the *3-Education*, *14-Jobs*, and *21-PeopleProfile* handlers introduced component-wise name resolution and structured output partitioning (e.g., current vs. past positions, completed vs. ongoing studies), combined with improved geographic resolution via KG location edges. *1-CompanyProfile* went further by orchestrating other retrieval modules into compound profiles, delegating to funding, people, acquisitions, and competitor handlers, while upgrading competitor discovery from raw SQL joins to graph-topology and category-overlap reasoning.

Even the handlers outside the network-DB scope (i.e., *FundingNewsHandler*, *News-BaseHandler*, *TechNewsHandler*, and *FundsHandler*) were upgraded with relevance filtering pipelines, semantic embedding search, timeframe extension strategies, and improved `NaN`-values handling, ensuring balanced and high-quality retrieval even when the underlying content remained purely SQL-based.

Furthermore, the upgraded system introduces a standardized retrieval output formatting (e.g., dates and missing-values displaying) and enhanced fuzzy-matching capabilities for robust entity-names' resolution (appendix A.5). Moreover, the retrieval output format systematically presents fine-grained filtering-parameters' values to provide transparency about the applied search criteria; a crucial aspect for fuzzy-matched entities, where the LLM needs to verify whether retrieved data correspond to the original query or to resolved alternatives. Anyway, each handler logs comprehensive parameter assignment and fuzzy-match resolutions information, enabling full traceability of the retrieval process.

2.2.2 Additional Alignments

On the other hand, since the underlying classification-model seems functional enough, the only modifications to the Intent-Coordinator and other retrieval-overhead scripts are momentarily focused on fixing poor parsing-error handling and plain bugs, besides clearly exposing a unified-searcher to the extraction-layer to allow both access to the KG-DB and connection to the original SQL-instance (appendix A.6). Notably, the changes in assignable-parameters applied to the handlers would not be considered by the classificator, thus nor exploited in full-RAG testing until this ML-component would not be retrained; indications and ideas over this procedure are presented afterwards (section 4.1.2).

Given their necessity for data compensation also for the upgraded retrieval-layer, three repositories - intermediary data access components - are updated too (i.e., *Articles*, *Company*, and *Investor*), as compatibility with the new hybrid-searcher must be enforced and some aspects of the filtering-logic at this level changed, applying a soften but ranked rationale for articles' extraction and categories' matching (appendix A.4). Finally, a functional upgrade is applied to the main NL-instructions set given to the LLM-generator component in the form of initialization-prompt, mainly to enforce stricter adherence with the retrieved-data with respect to general-knowledge, and asking to transparently notify any information reported aside of the underlying dataset's content (appendix A.7).

Importantly, despite numerous bugs are noticed in the original versions of many scripts during this KG-conversion procedure, the development stage of the SQL-based

RAG was accepted as embryonic, and no fixed was applied apart from aligning the original system-version for successive comparative testing (appendix A.3). The ultimate aim is focusing on obtaining a new version of the service as well-performing as possible, keeping the SQL-based counterpart firstly as a reference-point for the product's development-stage, rather than conceiving it as an established baseline with tabular-retrieval.

2.3 Benchmarking

Already during the handlers' upgrade procedure, preliminary tests are leveraged to manually verify the correct behaviour of the new code; the relevant queries are thus gradually added to a collection of query-specific test-suites, that would then be revised and refined for full-testing once the retrieval-layer conversion is complete in all its highly-interdependent components (section 2.3.1). Subsequently, the two RAG versions' performance would be assessed by both strictly quantitative time-metrics and structured quality ones, drawing a comparative benchmarking with the previously existing solution. It is worth noticing that, if the latter is being found in a not-production state and presenting bugs even in the retrieval components themselves, it is also because of the lack of any systematic testing process, apart from the Intent-Coordinator's training, implying the testing of the related classificator model with queries not annotated with filtering-parameters (section 2.2). In fact, the original RAG has been experimented only on a very limited set of queries and with no metric-based evaluation logic, such that the pipeline to verify the performance of the extraction-layer and the complete-system has to be developed from scratch (section 2.3.2).

2.3.1 Testing Framework

A collection of 24 handler-specific test-suites is defined for both retrieval-only and full-RAG processing, each consisting indeed of 10 test-cases with a NL-question specified together with the relevant set of parameters' values; additionally, a last test-suite (i.e., *24-Fuzzy-Matching*) is composed of the same number of queries, but spanning multiple handlers operating fuzzy-resolutions over person, organization or event name-features, exactly to verify the functionality of this inter-handlers component through increasing levels of input-parameters' distortion (i.e., typos, partial-matches and phonetic variations, reported identically in both the NL-question and the respected annotated parameters). This set of 250 queries in total (appendix B.1) is gradually perfected to span all the main intent-parameters combinations - including the activation of the newly implemented ones (section 2.2.1) - and to reflect the plausible focus-dilemmas of an average startup (section 1.3.4), while isolating a single query-type interest as much as possible [39]. Notably, the some handlers are also based on questioner-profile contextual information (e.g., *8-Greeting*, *18-MyStartup*), so same test-suites report these parameters too; in all the remaining cases, the user-descriptors are kept defaulted to Doriot AI's ones.

The two models' execution can therefore be tested in parallel, producing results by feeding the golden-parameters to the extraction-layer directly and checking the retrieved output only, or by assigning the NL-query only to the full-RAG - leveraging the unchanged intent-classifier (section 2.2.2) - and verifying the final LLM-generated response. The original retrieval uses SQLAlchemy to connect to an AWS RDS PostgreSQL server on a *t2.medium* instance, thus the handlers send SQL-queries for execution onto the thereby remotely-hosted DataBase, then the results are returned to the local machine. On the other hand, the network-based extraction is mainly managed on local, with the searcher component loading the Knowledge-Graph object from `.pkl` file and exposing it to all the upgraded handlers, while still initiating the usual SQLAlchemy connection for the aforementioned not-converted aspects of the retrieval process.

2.3.2 Evaluation pipeline

With the above settings and over all the defined test-suites, an evaluation procedure is operated in parallel on the two system versions, testing their retrieval-only and full-RAG capabilities with the just described methods (section 2.3.1), and computing the following metrics to assess the performance on two fronts: latency and outputs accuracy.

Time Metrics [106] The latency is recorded for the extraction-layer testing as its complete execution, and for the complete-system testing as devided in retrieval - in this case including also the Intent-Coordinator's procedures - and generation components:

- ***Retrieval-Only***: latency from golden-parameters' input to the output of extracted information snippets by the relevant handler;
- ***Full-RAG Retrieval***: latency from NL-query's input to the output of all extracted information snippets by the triggered handlers;
- ***Full-RAG Response-Generation***: latency from retrieval-outputs' reception to response's restitution for the LLM-Generator;

- ***Full-RAG Total Time***: end-to-end latency from question's input to final-answer, coinciding with the sum of the above two metrics.

Quality Metrics [107, 23] Being gold-outputs on a predefined query-set not given, and having no baseline model to refer to, plain semantic correctness metrics for QA (e.g., Exact Match, BERTScore, BLEU and ROUGE) would be of no help for the evaluation of the present task. Instead, also considering the fundamental service-enhancement target of the project, the full logs-including (i.e., `INFO/ERROR/WARNING` messages from handlers, searchers, and RAG components too) outputs for both retrieval-only and full-RAG testing over each query of the 25 test suites are saved (appendix B.6), thus the accuracy of the extracted data and of the generated NL-responses are separately annotated one test-case at a time, comparing the two system-versions' results with a structured paradigm (appendix B.3). In particular, the outputs' quality-labelling is firstly operated automatically through an LLM-based script that iteratively delegates through API-calls the predefined evaluation procedure, enforcing transparent reasoning and requiring explanatory notes for all the assigned classes; then the whole results are manually verified and - in rare cases - corrected to be precisely representative. The chosen labels are as follows, with the defined ordinal logic.

Retrieval-outputs:

- *ER*: execution error in retrieval-only logs;
- *NO*: no instance extracted for retrieval-only testing (procedure works, empty result);
- *LOW*: retrieval-only works, but extracted results are imprecise, not respecting some filter, poorly ranked, or weaker than the other model version's ones;
- *OK*: retrieval-only works, and extracted results are precise, coherent, and well-ranked.

Full-RAG responses:

- *ER*: execution error in Intent-Coordination phase of full-RAG testing logs;
- *QF*: Intent-Coordinator runs, but fails to recognize the relevant query-type to trigger (i.e., the handler proper for that test-case, which is annotated together with its parameters, is not among the activated ones, as displayed in the full-RAG logs);
- *FF*: correct handler triggered, but feature values assigned to it are imprecise (i.e., the filtering-features selected for retrieval by the Intent Coordinator, which are clearly shown at full-RAG logs when the relevant handler is called, do not correspond with the ones annotated as correct query's parameters for the relevant test-case);
- *LOW*: the whole Intent-Coordination procedure works and - independently from the retrieval's results which in this case are not explicitly recorded - the final answer is of

low quality (e.g., poor wrt to the expectations set by the NL-query posed, imprecise, seemingly not based on any actually retrieved data but rather leveraging LLM's general-knowledge only, or anyhow less satisfactory in content or structure wrt the response generated by the other full-RAG version for the same test-case);

- *OK*: the whole Intent-Coordination procedure works and the final answer is coherent wrt the expectations set by the NL-query posed, precise, and well-structured.

Quality-change is expressed in unitary steps on a 4 grades scale:

$ER < NO/QF\text{-}FF < LOW < OK$

Trough the coherent test-suite's indexing-logic - from 0 to 23 for handlers-specific ones and 24 for the fuzzy-matching - and numbering each of their 10 queries form 0 to 9, the time-records are merged with the consolidated annotations of test-outputs quality in a complete evaluation table (appendix B.4), summarizing the obtained shift in holistic performance between the previous RAG-version and the upgraded one.

Chapter 3

Results

Overall, the new RAG model reaches significantly higher answers' quality, despite requiring longer processing with respect to its previous counterpart (appendix B.4); which is in line with the expectations, given the original retrieval layer's embryonic nature, operating straightforward SQL queries on plain tabular data. In fact, the fundamental desired breakthrough of the network-DB was to allow entities-aware and relationships-contextual instances extraction, leading to better informative outputs [97]. Nevertheless, as already stated (section 2.2.1), given the non-baseline nature of the SQL-extracting RAG system - kept in a non-production state to focus efforts on the KG-based model's implementation - the present benchmarking aims at measuring the extent of the obtained upgrade in performance for this Doriot AI's product development, rather than comparing directly the two retrieval-methods over this task. In other words, although in specific cases the effects of the extraction-strategy's switch would be evident and coherent with the theoretical expectations [45], rarely the results would reflect overall the change in RAG-techniques as isolated from implementation-level aspects (e.g., efficiency and formal-correctness of the underlying code).

Following this obtained-development analysis-scheme, the improvement is already visible by comparing the quality scores on both the retrieved data and the final NL-responses, while the lower time-efficiency is conceivable as indicative but to be re-scaled: the market-research agent's back-end is fully deployed in its original version on an AWS EC2 *t2.medium* server [68], suitable for light workloads and basic testing only [83]; while just the retrieval-phase for the upgraded RAG system is delegated to the local machine, an Intel Core i7-8750H at evaluation [14]. In other words, the KG-extraction would be even slower if requested to the above server, due to both processing power and network latency [37], but the extent of this difference in elaboration-period would be considerably reduced in final user perception when a proper time processing instance would be used for service-delivery, as *t2.medium* is inherently not even appropriate to handle SQL-query concurrent retrieval at production-level [64]. This aspect would be subsequently further discussed, to draw some strategies to lunch an MVP (section 4.1.4).

Specifically (appendix B.5), with the declared handler-specific experimental setting (section 2.3.1), a nearly-unitary average quality-score (section 2.3.2) improvement was obtained across the 250 test-cases (appendix B.1) for both the output kinds - 0.96 for retrieval and 0.82 for response - highlighting a spread jump of almost one quality-class for the handlers' job. On the other hand, the mean data-extraction times incremented almost of factor 5 for retrieval-only and 7.5 for full-RAG testing, as rebalanced by shorter response-generations to reach $83.71s$ of average total elaboration for the upgraded system, less than $21s$ more than its original counterpart. It is important to keep in mind that these results would only indirectly represent the expected answer-waiting times for a final user, both due to the aforementioned server type's impropriety issue and to the density in the leveraged test-suites of edge-cases (e.g., non-existent instances' searches, pushing the new RAG in particular to resolve them as far as possible through time-consuming fuzzy-matching procedures). Moreover, some test-cases involve filtering-rationales that are out of the capability of the current Intent-Coordinator implementation (e.g., new features or more granular values for them) (section 2.2.2), hindering its classification logic, assigning retrieval-parameters that are partially precise with respect to the NL-query, and ultimately requiring the LLM-generator to compensate the so-extracted data with larger general-knowledge in order to attempt a satisfactory reply [62].

3.1 Time Performance

The distribution of time metrics over the single test-cases (appendix B.2), together with a careful analysis of the complete outputs (appendix B.6), helps to better uncover these patterns. In fact, the highest increments in retrieval-only time for the KG-based switch were recorded - in decreasing order - for the handlers: *9-InvestmentDetails*, *11-Investor*, *10-InvestmentPartner*, *18-MyStartup*, *12-InvestorNetwork*, *20-OrganizationRelationship*, *14-Jobs*, *21-PeopleProfile*, *13-Ipo*, *17-MarketTrends*, *0-Acquisition*, and *1-CompanyProfile*; all actually implying thorough network-traverses and extracting uniformly better information-sets in their upgraded implementations. Interestingly enough, due to the higher significance of the retrieved instances, some of these handlers' new version demonstrated to profitably keep the response-generation task easy enough that they outperformed their unrefined SQL-based counterparts in the matter of full-RAG total answer time [88]; in descending order of cumulative speed-up: *0-Acquisition*, *17-MarketTrends*, *11-Investor*, *13-Ipo*, *21-PeopleProfile*, *1-CompanyProfile*, *14-Jobs*. Mixing diverse handlers with the specific aim of verifying the RAG's ability to deal with imprecise filters definition and typos, the *24-Fuzzy-Matching* testing matched the dual quality- and rapidity-enhancement trend, attaining a slightly lower execution-time for the 10 queries' full-RAG response thanks to the upgraded retrieval-layer, despite the original model's answers were totally unsatisfactory in comparison. Notably, even if still using the same kind of SQL-calls to operate extraction, and thus presenting comparable retrieval times with respect to their orignals, also *7-Funds* and *23-TechNews* handlers' KG-variations allow altogether lightly faster answering, via more precise and complete information-sets delivered to the response-generator.

However, the proportionality is mostly maintained across the other handlers' tests between retrieval-only, full-RAG retrieval and total-response times; for the upgraded model in particular, an almost uniform persistence along time-metrics is visible also in the most-lengthy queries' rankings for each test-suite. Apart from the cases where the Intent-Coordinator largely failed to initiate the relevant extraction-type or to assign significant features' values, the full-RAG retrieval time is indeed commensurate to the the retrieval-only one recorded with pre-determined parameters for the same query; and for the better-flowing upgraded RAG the same is also mirrored by the total-time, as the extracted instances are appropriate not to over-rely on the LLM's work, such that the main variable component of the cumulative execution is always related to the data-retrieval period. By contrast, for the original RAG model, the longest-queries in terms of total response time often do not coincide with the ones that required enduring instances-extraction, due to less complex and fruitful retrieval logics that accumulate less of the execution time but leave much more weight to sparse time-additions needed to compensate lower-quality extracted information [62]. In general, the slowest-running retrievals for each handler-specific test-suite are often outliers due to the voluntary edge-nature of the relevant queries (e.g., the $9th$ test-case for 12 of the suites, corresponding to the one presenting entity-names known absent from the DataBase and triggering the lengthy upgraded-RAG's fuzzy-matching), with slowest records reached by these hampered retrieval-procedures for *11-Investor*, *9-InvestmentDetails* and *12-InvestorNetwork* - more than 450s, about 375s and almost 210s respectively [78]. As aforementioned, this consistently offsets the average execution times, suggesting the use of a smart-stop strategy to limit the excessively tenacious resolution cases and favour the overall QA-service usability; a fitting solution of this kind would be subsequently discussed, together with other priority incremental developments (section 4.1.2).

3.2 Quality Improvement

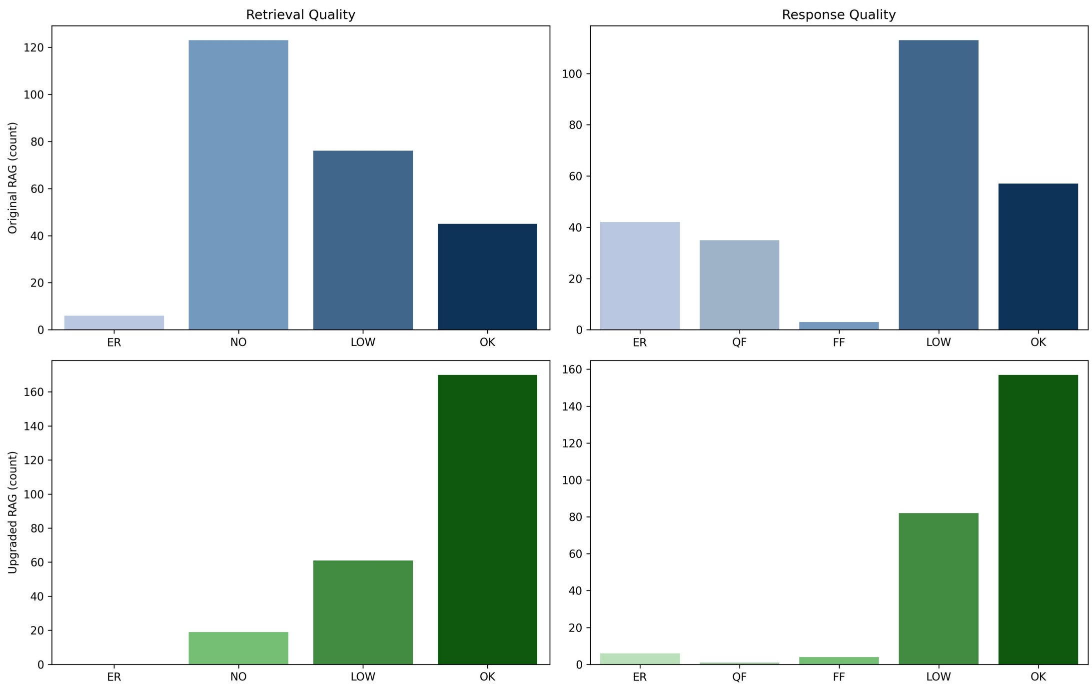

Figure 3.1: retrieval- and response-outputs' distribution by incremental-quality bins

Glancing at the quality-evaluation results (fig. 3.1), it is clear that the two RAG versions are not comparable in terms of retrieval- and response-appropriateness, with the upgraded model attaining indisputably better results over the same wide-spanning test-set compared to its previous counterpart. Specifically, all the handlers that are thoroughly-upgraded in retrieval-logic beyond the plain network-extraction conversion, consistently show higher-level results (appendix B.4); and this is especially true for the investment-related handlers (i.e., *9-InvestmentDetails*, *10-InvestmentPartner*, *12-InvestorNetwork*), which are considerable central for the project's objective.

Furthermore, referring directly to the complete logs-including test-outputs (appendix B.6), it is impossible to oversee some handler-wise concrete improvements: all the additional parameters for *4-Event* are fully functional and make the network-based retrieval far more granular, while the original one always returns no result - as for *18-MyStartup* despite the full features-set alignment - and present a uniform Intent-Coordinator's bug at full-RAG testing too. Moreover, the previous *15-LeadGeneration* handler's outputs include emoticon-described features, which negatively impact the LLM-generator's readability. However, the extracted information is often upgraded with the KG-retrieval also in content: current job positions are always prioritized instead of displaying just random roles of people's career, and disparate previously-malfunctioning filters are flawlessly applied (e.g., the geographical selection for *21-PeopleProfile*, and the automatic yet declared time-period extension for all news-articles retrievers when the default recency requirement fail to obtain any result). The upgraded Intent-Coordinator script, even if leveraging the same exact ML model, solves most of the errors encountered by its original code version thanks to the code-fixes applied (appendix B.7). Apart from runtime-problems, keeping the same exact classification model, the upgraded Intent-Coordinator still manage to avoid many JSON-decoding issues that infest the previous system's parameter-assignment (appendix B.4), thanks to a more robust rationale handling eventual parsing of malformed or empty interpreter's outputs (appendix B.6).

3.3 Time-Efficiency Change

Keeping in mind the clear ineffectiveness of the original system in responding to certain types of questions at all, to preliminarily assess the difference in time-performance per se, a strategy is therefore to isolate the queries that lead to fulfilling results for both the retrieval-layer's implementations [88].Gross of the already discussed hardware changes, the data-extraction itself is thus taking a mean of 2.7*s* more when the two models' extracted information-sets are comparably and finely meeting the relevant query's expectations; similarly, the final-response waiting-time's difference is of 2.4*s* on average when both the NL-answers are satisfactory (fig. 3.2). Considering the single test-cases included in this statistic (appendix B.4), presenting *OK* annotations for the original model' outputs spread-enough over handlers then KG-converted too, the values just mentioned might be indicative of how these time-metrics would change by operating network-retrieval instead of SQL-calls for basic-level queries; however, as also proven by the already higher variability in the upgraded system's distributions, the present snapshot would not stay representative for cases where a higher-level data-extraction is of substantial importance.

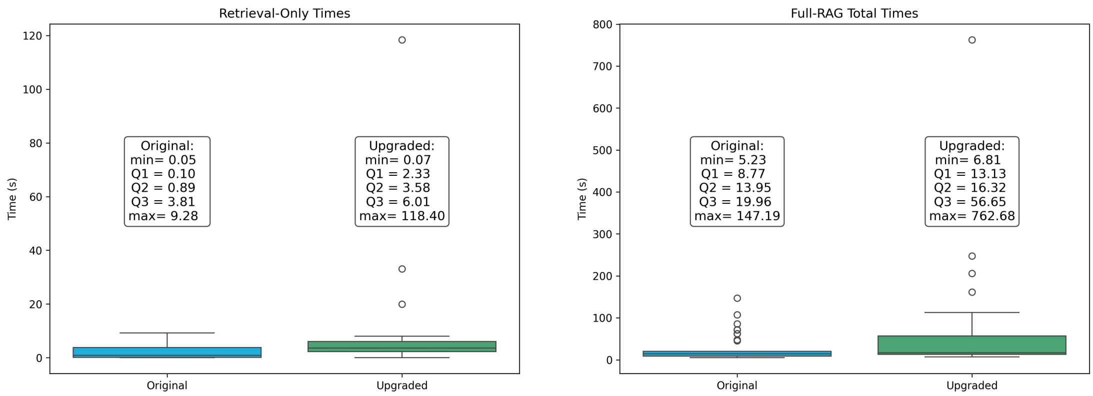

Figure 3.2: retrieval- and total-time distributions for the RAG versions when both produced correct respective outputs (i.e., when 'OK' was annotated for both the extracted information sets in the retrieval case or for both the NL-answers in the full-RAG case)

Gauging the full evaluation-metrics (appendix B.4) and widening-back the viewpoint, the time-difference trend can be observed over the defined quality-score's increment (fig. 3.3): about $11.5s$ on average are needed for KG-retrieval to keep a comparable level of accuracy with respect to its SQL-counterpart - in 80 test-cases - or to step to a slightly superior quality-level - for 86 queries and mainly from from *LOW* to *OK*. With significant variability of the time-value, $24.4s$ more are instead expected when two tiers of retrieval precision are climbed, as this often implies the upgraded handler to obtain a correct result when its original version returns no instances at all; 5 cases bringing errors in SQL-extraction lead instead to fully satisfactory network-drawn information with a small time increment. Across different handler's testing, in a total of 6 occasions the KG-retrieval's results is of lower quality, due to isolate outputs' visualization and elaboration faults spotted by specific test-cases. Referring to the full test-outputs (appendix B.6) the 4 cases where the response is significantly worse for the upgraded full-RAG are caused to an Intent-Coordinator's run-time error that did not correspondingly incurred in the original version of the script, bringing a very-high time difference which is actually accumulated by fallback-retrievals' activations. Also some of the response-quality -1 jumps are due to the same problem, but most others are related to the LLM's general-knowledge slightly surpassing in scope the DB content at original-model's empty or limited retrievals, or to the NL-answer over-commenting fuzzy-matched data for bogus-entities queries despite recognizing the extracted instances' low correlation with the question. Interestingly, while the $+1$ and $+3$ enhancement-levels report variable but averagely consistent time increments, when the response-quality grows of 2 tiers the mean full-RAG execution is $24.4s$ shorter, mainly reflecting shifts from original Intent-Coordinator's failures to non-fallback KG-based retrievals. For all the errors introduced above and also others, some code-fixes would be successively discussed in depth (section 4.1.2).

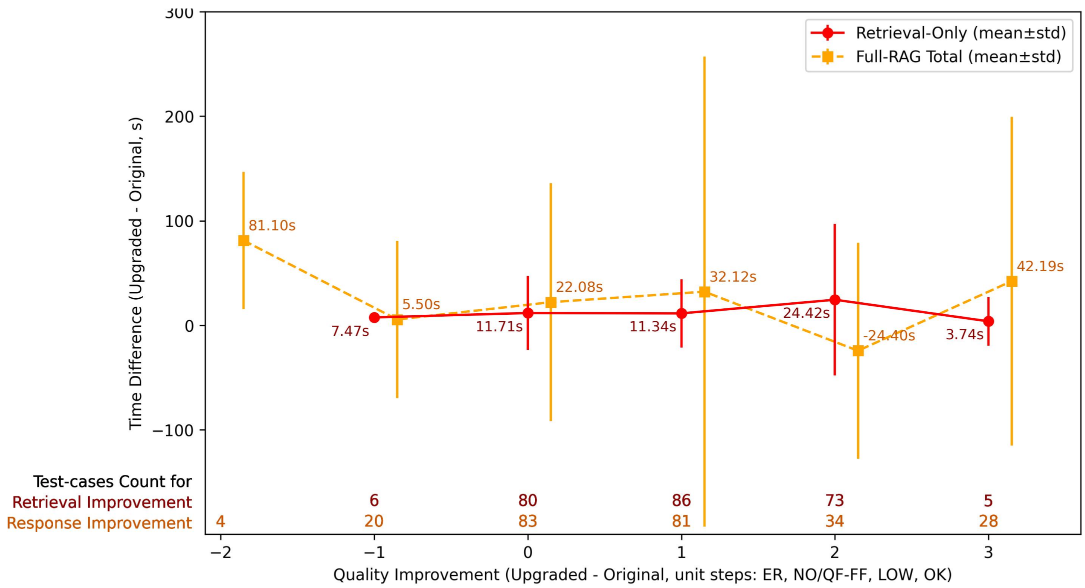

Figure 3.3: average retrieval- and total-times' difference between the two RAG versions by extracted information set's and NL-answer's quality-improvement score respectively

Chapter 4

Conclusions

In conclusion, all the fixed objectives were reached: directly highlighting its interconnection framework as per a prearranged scheme, Doriot AI's core information-set was converted to a Knowledge-Graph form by designing a robust, expandable and reproducible data-cleaning and network-building pipeline, able to fully embed a similarly complex tabular DB (section 2.1); the resulting object was thus experimented as informative-base for the company's Question-Answering service (section 2.2), significantly refining its retrieval-layer and defining from scratch a wide-spanning testing procedure (section 2.3), ultimately reporting quality results adequate for a Minimum Viable Product (chapter 3).

Stemming from the first evaluation process ever run on this software, comparatively assessing both the SQL- and KG-based data-extraction systems and full-RAG models under outputs-accuracy and time-efficiency metrics, some debugging and prototype-launching indications are summarized hereafter, together with suggested areas for further development (section 4.1). Following the latter, Doriot AI would yield a new integrated product with no real competitors on the startup's side of the funding-matchmaking market, as similarly sophisticated investment-environment exploration tools are currently delivered by larger companies with prices mainly affordable by VC firms and business angels [6].

Therefore, activating a RAG-copilot service explicitly targeted to early-stage ventures would give Doriot AI an outpost role in its industry, possibly profitable if wisely advertised [53], and surely engendering the pragmatical verification of the present project's wider goal: contributing to the state-of-the-art of ML methods empowering startups with efficacious decision-making aids (section 1.2).

4.1 Future Work

Given the complexity of the type of software at issue - requiring continuous maintenance and development to stay performant even at stable production phase [73] - and the embryonic state to which it was at the work's start, consistent enhancements were planned to persist as future-work directions (section 1.2.1). These are technically discussed in detail below, including plain error-fixing indications (sections 4.1.1 to 4.1.3), service-launching recommendations (sections 4.1.4 and 4.1.5), and incremental-enhancements suggestions (sections 4.1.6 and 4.1.7).

4.1.1 Data Enhancement

A first consideration must be done on the underlying information-set itself: the DB framework was consigned by Doriot AI in a not updated version (section 2.1); for the fruitful operability of any algorithm based on them, it is crucial that the data in their full version are ensured spanning even recent periods, which are of fundamental importance when the comprehension of such a dynamic context as the startups-funding one is at stake. Additionally, the complete information-set owned by the company as stored in the tabular-DB exposed to the original retrieval-layer, has been verified to include some features that were missing from the core-data entrusted in `.csv` files format and thus KG-embedded, requiring the network-based handlers to actually extract them via SQL-calls as in their original counterparts (section 2.2). Exploiting the scalability of the relevant cleaning and embedding pipeline (section 2.1.1), these instances might instead be reversed into the Knowledge-Graph too, as supplemental attributes or nodes eventually linked by new edge types, to align the whole data to the upgraded rationale and ease retrieval.

For example, each industrial-sector could be added as a vertex with attributes summarizing the related market trends, then connected to all the person-, organization- and event-nodes participating in it [46]; the news articles, instead, could be indexed and inserted as vertices storing as attributes their plain features (i.e., publication date, title, full-text) and a their vector-embedded representation, then directly linked with people, companies, events and markets through an offline similarity-search over their characteristics and descriptions depicted in the same embedding-space [32]. Subsequently, by public-data web-scraping or abstraction from the already-present network's structure and attributes [80], the Knowledge-Graph could be enriched with other interconnected elements spanning additional informational areas (e.g., theoretical-knowledge and inspiring-cause nodes). Subsequently, leveraging the same queries-set and evaluation pipeline (section 2.3.2), a comparison could be carried out between the original RAG and its modified version, eventually also altering the scheme of the current graph-dataset (e.g., exploding some node kinds as person-ones into sub-types, maybe directly based on the investing-role) and of some handlers' code respectively, until an empirically best-efficient framework is settled [20].

4.1.2 Debugging

Differently from its original counterpart, the upgraded RAG's retrieval-layer produces no runtime errors at testing (appendix B.4), but for 6 queries its outputs are less satisfactory then the SQL-extractor's ones (section 3.3), highlighting the necessity to revise some aspects in particular (appendix B.6):

6-FundingRound - as visible in the extracted instances for test-case 3, the investors' information are displayed in the retrieval results despite not requested through the related boolean parameter;

9-InvestmentDetails - while the category-based filtering works - unlike the one of its original version - the handler does not show the relevant industrial-sector attributes of the extracted entities;

13-Ipo - the *MoneyRaised* and *Valuation* features are not correctly displayed when single company's IPO-instances are retrieved;

16-MarketAnalysis - the handler caps the amount of considered companies to 1000, limiting the scope of the resulting analysis when larger amounts of organizations should be involved instead.

On the full-RAG side, instead, just a few test-cases bypassed the Intent-Coordinator's more robust values-handling logic, raising errors that directly indicate some additional debugging steps to be applied to the upgraded script (appendix B.7). However, it is fundamental to re-train the underlying classification model, to exploit the better granularity given by the new and augmented handlers' parameters, which are currently mis-assigned during complete-system testing (section 2.2.1).

Furthermore, the tenacious fuzzy-matching procedure may impede fluent QA-service usability, as the system would take too much time trying to resolve excessively-misspelled entries or instances not present in the underlying DB at all; a straightforward solution could be to implement a time-based smart-stop strategy that breaks this script execution after a predetermined period and eventually regards no matches, otherwise the stop-condition could be applied on the fuzziness-level by premeditatedly deactivating the methods that aims at too-loose matching-extents.

In general, to ensure better autonomous working of retrieval-outputs' quality LLM-based evaluation (section 2.3.2) it is suggested to remove the word 'error' from the standard messages printed when no instances is retrieved with the given filters: this wording sometimes cause the tests-reading agent to mistakenly - then corrected during the manual revision phase - consider *NO* retrievals as *ER* instead (e.g. the last test-case for *3-Education* and *12-InvestorNetwork*), despite the data-extraction got no runtime interruptions (appendix B.6).

4.1.3 System Refinement

Before deploying the MVP, additional testing are highly recommended through the Intent-Coordinator's training data-sets [82]: despite having no annotation of filtering-parameters, these huge lists of NL-queries associated to ground-truth callable handlers - often more than one - can be used to assess the full-RAG model's capacity and speed to answer questions inherently implying multiple retrieval-intents, eventually uncovering additional runtime-errors to fix at any layer of the system. Moreover, over this task, it would be possible to infer the accuracy of the NL-responses thorugh a LLM-operated evaluation similar to the one used in this work (section 2.3.2), and of the extractions too, by assuming of reference any features-set that is coordinated independently from its actual relevance with respect to the query.

Vice-versa, the new set of handler-specific and parameters-annotated test-cases might be used to fine-tune the classification model itself, as precisely favouring examples not only of the relevant intent but also of the related filtering-features' values, which could be very effective in augmenting the Intent-Coordinator's precision both in assigning the parameters and in recognizing the correct query-type itself [86]. For instance, this may bolster more accurate query-type choices for test-suites now consistently misinterpreted (e.g., *22-RecommendInvestors*), solve the confusion between the subtly different 'acquired' and 'acquirer' entries of the *0-Acquisition* retriever, and refrain people titles (e.g., *'Mr.'*, *'Ms.'*) from being included into name-values during *3-Education*, *14-Jobs* and *21-PeopleProfile* features-coordination (appendix B.6).

Furthermore, some handlers might benefit in accuracy from an ancillary parameter taking a value between *'any'* and *'single'* to specify whether to conceive matches over a particular filter as accepting a single best-candidate or multiple ones by importance, whenever the retrieval-logic permits multiple entities to be resolved (i.e., not for analysers as *20-OrganizationRelationship* or *21-PeopleProfile*, which inherently have to individuate a single company- or person-node in order to delve into its attributes and relations in depth). As an high-level example, the *14-Jobs* handler might leverage a $person\_plurality$ parameter taking the value *'any'* for queries like *"is there a Bill working for Apple?"* which would need the extraction of all the individuals with that name who have a professional relation with the specified company, or *'single'* for questions as *"What position did Mr. Gates have in Apple?"* entailing - even more clearly in the eventual context of a conversation on Microsoft [75] - the interest for the specific most-renowned professional, so that the already sensibly-ranked matched-names list would be cut to the first result before filtering, and with no retrieved instances about less-prominent Gates-surnamed employees the response would directly deny any role of Bill Gates at Apple [11].

4.1.4 Server Deployment

As mentioned in chapter 3, the time-performance evaluation of the upgraded RAG system was carried out under non-production conditions, with SQL-retrieval entrusted to an AWS *t2.medium* instance [68] and KG-retrieval executed locally on an *Intel i7-8750H* machine [14]. Both setups are suitable only for development or testing, as neither can sustain production-grade workloads requiring concurrency, scalability, and stability [64]. In practice, RAG-based services are usually deployed on cloud instances tailored to parallel data-intensive operations, ensuring reliable performance even under multi-user conditions [13].

An overview of candidate server from the AWS *m5*, *r6*, and *c6* families [66, 67, 65] can be drawn (table 4.1), where concurrency levels are reported together with the expected average retrieval-only times over, based on declared speed-comparisons [64] under the simplifying assumption of negligible network latency. While exact improvements depend on query complexity, graph size, and implementation-level parallelism [37], the figures indicate that a move to modern compute- or memory-optimized instances would immediately reduce per-query extraction times, while unlocking efficient concurrent processing. In particular, compute-oriented families (e.g., *c6a*) are expected to halve response times relative to local testing, while memory-optimized alternatives (e.g., *r6i*) would ensure sustainable handling of larger KGs without bottlenecks.

In practice, the `.pkl`-saved `NetworkX.MultiDiGraph` object can be loaded on

Instance Type	vCPU	RAM	Speedup	Retrieval Time (s)		Concurrency
				full-test	base-test	
[14]i7-8750H	6	16GB	baseline	18.80	3.58	4–6
[68]t2.medium	2	4GB	0.4–0.7x	27–47	5.1–8.9	2–3
[66]m5.2xlarge	8	32GB	1.5–2x	9.5–13	1.8–2.4	8+
[67]r6i.2xlarge	8	64GB	1.5–2x	9.5–13	1.8–2.4	8+ (large KG)
[65]c6a.2xlarge	8	16GB	1.7–2.2x	8.5–11	1.6–2.1	8+

Table 4.1: performance and concurrency comparison of AWS EC2 instances with respect to local baseline, with estimated retrieval-only times by expected-speedup for full- and basic-test (i.e., queries that reported *OK* extraction outputs for both RAG versions)

the chosen instance and exposed via a lightweight Python service (e.g., FastAPI or Flask), with client queries remotely served in a similar manner to SQL back-ends [81]. Production deployment would not only normalize execution times by eliminating the developmental bias of heterogeneous testing environments, but also verify that the retrieval-layer scales as expected with larger graphs and more concurrent users [87]. Since retrieval over networks is inherently more complex than SQL extraction and relies on graph traversals or vector-based indexing [61], further efficiency gains will require both hardware upgrading and algorithmic optimizations (e.g., approximate search structures) [79]. A crucial next step would be thus to test the upgraded retrieval-layer directly on the chosen production-level server, verifying the full usability of the KG-DB, the compatibility of the whole experimented code, and the scaling of extraction times in realistic service conditions.

4.1.5 Go-to-Market

A crucial step after the technical validation of the upgraded RAG system is the definition of a coherent go-to-market strategy; in the present case, the business scope is clearly oriented towards early-stage ventures, which were already identified as the natural primary users of the service (section 1.2). As startups are typically resource-constrained and require agile and low-cost decision-support tools, it is fundamental to adapt the product specifically to their needs, by focusing on the main query types that these actors would realistically pose in their strategic reasoning [57]: further research should deepen the exploration of domain-relevant questions, such as market-size estimation or pricing-sensitivity analysis, which are often of central importance for entrepreneurs in fundraising and positioning phases [74]. These queries can then be translated into additional handlers or refined retrieval-parameters, so that the system is capable of delivering answers that are not only informative but also directly actionable from a business standpoint [72]. Moreover, for the solution to reach adoption, a competitive pricing scheme needs to be devised, balancing affordability for startups with the sustainability of the service for Doriot AI [85]; such a scheme must be integrated into a precise business plan that delineates the initial entry strategy, the customer-acquisition channels, and the planned differentiation of the product with respect to other intelligence services currently targeting investors and business angels rather than ventures themselves [92]. Only by combining these elements into a unified roadmap - target identification, question-type refinement, handler and data-set extension, and pricing - can the service successfully transition from technical prototype to market-ready MVP [50].

On the usability side, the User Interface (UI) constitutes a decisive factor in shaping the product's perception and adoption [89]. For testing purposes, a parallel UI should be developed as an inspection tool, enabling researchers and developers to closely monitor the retrieval and reasoning processes of the RAG system (section 4.1.3); this experimental interface could allow a preliminary setting of the user context - otherwise defaulting to Doriot AI's standard configuration - and the submission of queries to either the original or the upgraded model. The outputs would be displayed sequentially, with the retrieved instances shown first, followed by the generated NL-response, and complemented by a dedicated logging column reporting real-time execution steps, including the elapsed times for both retrieval and full-RAG completion. Crucially, to safeguard usability, time-limits should be imposed on both stages, so that excessively lengthy runs are gracefully terminated [90]. Additionally, a comparative screen-split mode could be enabled to visualize side-by-side outputs of the SQL- and KG-based versions on the same query, thereby facilitating direct evaluation. By contrast, the consumer-oriented UI would need to abstract from such technical details, embedding only visually appealing progress notifications and high-level reasoning markers to provide transparency without overwhelming the user [89]. The same logs leveraged in testing can thus be repurposed as back-end signals for user-facing procedural feedback, ensuring the right balance between system explainability and seamless interaction.

4.1.6 Recommender-System Alignment

While the recommender system represents the core algorithmic asset of Doriot AI, it was not employed in the present project: the only indirect interaction consisted of the *22-RecommendInvestors* handler leveraging pre-computed lists of recommended investors already stored in the SQL-database - information that might be embedded in the graph by a net of apposite-type links. Nevertheless, the development of the new Knowledge-Graph DataBase (KG-DB) opens promising opportunities to explore how recommendation techniques could be either adapted or re-designed upon this updated infrastructure. In particular, the KG-DB in its finalized version (section 4.1.1) would provide a richer and more flexible representation of the underlying entities and their relations, which may serve as a powerful substrate for recommender pipelines [97]. In further up-to-speed service-delivery stage, hypothesizing a large enough customer base, such recommender-system's alignment could be a prerequisite to shift the role of the RAG system from being a stand-alone querying service to enable novel matchmaking strategies based on users' behaviours and retrieved insights, co-informing the ranking processes jointly with the graph-embedded relations themselves [52].

Preliminarily, the current matchmaking model might be experimented on the complete KG-DB, with a slightly modified rationale and an input format converted to directly match the network's one, testing for baseline performance enhancements; subsequently, additional techniques in the recommending domain could be experimented too (section 1.3.2). The improvements might be benchmarked through held-out validation techniques (e.g., using k-fold[1] or temporal-split[2]), measuring the ability of the Knowledge-Graph to influence recommendations' quality by common ranking metrics such as Recall, Hit Rate, Mean Average Precision (MAP), or Normalized Discounted Cumulative Gain (NDCG) [71].

4.1.7 Multi-Agent Upgrade

Building upon the preceding RAG upgrades and the KG-DB infrastructure (sections 4.1.5 and 4.1.6), the implementation of a multi-agent system represents a compelling future direction, potentially transforming the service from a sequential retrieval-and-generation pipeline into a more autonomous, coordinated ecosystem of specialized components. In this framework, individual agents could be designed to handle distinct aspects of the startup support process, including Natural-Language Question-Answering, structured data retrieval from the KG, predictive modelling for market-trends and pricing-strategies,

[1]Cross-validation method partitioning the dataset into k equal subsets, to iteratively train on $k-1$ of them and validate on the remaining one, robustly estimating model performance. [31]

[2]Validation technique dividing the dataset based on time, to train on earlier information and test on later ones, simulating real-world scenarios with predictions on past data. [9]

and the recommendation of VC partners, with each agent operating semi-independently yet communicating to resolve complex, multi-faceted queries [76]. Such a system would allow concurrent processing of heterogeneous tasks, reduce bottlenecks typical of monolithic pipelines, and enable dynamic adaptation of strategies based on user inputs or evolving database content [77]. Moreover, agents could be equipped with specialized reasoning capabilities, such as context-aware ranking, cross-validation of predictions, or ensemble-based recommendation refinement [84], smoothly leveraging both the KG's structural information and behavioural signals from past user interactions as theorized above (section 4.1.6). From an operational perspective, a multi-agent upgrade would require robust error handling, modular testing, and incremental improvements; as individual agents should be debugged, updated, or retrained without impacting the entire system [63]. Ultimately, such an architecture would not only extend the current RAG system's capabilities but also lay the groundwork for a scalable, adaptive, and intelligent platform, capable of supporting early-stage ventures with a far richer, more interactive, and contextually aware decision-assistance environment [77].

Bibliography

[1] Muhammad Zain Bin Aamir. “Tinder for startups: Building a matchmaking engine for new ventures and Venture Capital firms”. MA thesis. University of Edinburgh, School of Informatics, 2023.

[2] Akram Alkouz et al. “UKRAG: A Unified Knowledge Graph to Enhance Retrieval Augmented Generation Performance”. In: *International Symposium on Intelligent Computing Systems*. Springer. 2024, pp. 1–19. ⇒.

[3] Javier Arroyo et al. “Assessment of machine learning performance for decision support in venture capital investments”. In: *Ieee Access* 7 (2019), pp. 124233–124243. ⇒.

[4] Asfak Asif and Mahfuzulhoq Chowdhury. “A Machine Learning-Based Suitable Startup Recommendation Scheme and Investor Assistance Mobile Application”. In: *Journal of Comprehensive Business Administration Research* (2024). ⇒.

[5] Peter W Battaglia et al. “Relational inductive biases, deep learning, and graph networks”. In: *arXiv preprint arXiv:1806.01261* (2018). ⇒.

[6] Andrea Bellucci, Alexander Borisov, and Andrea Zaccaria. “The funding gap and the role of financial return crowdfunding: some evidence from European projects”. In: *Small Business Economics* 61 (2023), pp. 1613–1636. ⇒.

[7] Leo Breiman. “Random forests”. In: *Machine learning* 45 (2001), pp. 5–32. ⇒.

[8] Chenyang Bu et al. “Which companies are likely to invest: knowledge-graph-based recommendation for investment promotion”. In: *2022 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2022, pp. 11–20. ⇒.

[9] Pedro Campos, Fernando Rubio, and Iván Cantador. “Time-aware recommender systems: A comprehensive survey and analysis of existing evaluation protocols”. In: *User Modeling and User-Adapted Interaction* 24 (2014). ⇒.

[10] Mengru Chen et al. "Heterogeneous graph contrastive learning for recommendation". In: *Proceedings of the sixteenth ACM international conference on web search and data mining*. 2023, pp. 544–552. ⇒.

[11] Mingyue Cheng et al. *A Survey on Knowledge-Oriented Retrieval-Augmented Generation*. 2025. arXiv: `2503.10677 [cs.CL]`. ⇒.

[12] D Manning Christopher, Raghavan Prabhakar, and Schutze Hinrich. *Introduction to information retrieval*. 2008. ⇒.

[13] RAG Research Collective. "Here's how to build a Production-Ready RAG System". In: *RAG About It Technical Journal* (2025). ⇒.

[14] Intel Corporation. *Intel Core i7-8750H Processor (9M Cache, up to 4.10 GHz) Product Specifications*. 2025. ⇒.

[15] Annalisa Croce et al. "Social impact venture capital investing: an explorative study". In: *Venture Capital* 23.4 (2021), pp. 345–369. ⇒.

[16] Crunchbase. *Crunchbase: Business Information Platform for Innovative Companies*. 2025. ⇒.

[17] Jesse Davis and Mark Goadrich. "The relationship between Precision-Recall and ROC curves". In: *Proceedings of the 23rd international conference on Machine learning*. 2006, pp. 233–240. ⇒.

[18] Yuxin Dong et al. "Advanced RAG Models with Graph Structures: Optimizing Complex Knowledge Reasoning and Text Generation". In: *2024 5th International Symposium on Computer Engineering and Intelligent Communications (ISCEIC)*. IEEE. 2024, pp. 626–630. ⇒.

[19] Doriot AI. *Doriot AI: AI-Powered Tools for Startups*. 2024. ⇒.

[20] Jens Dörpinghaus and Andreas Stefan. "Optimization of Retrieval Algorithms on Large Scale Knowledge Graphs". In: *arXiv preprint arXiv:2002.03686* (2020). ⇒.

[21] Edinburgh Innovations. *Edinburgh Innovations: the University of Edinburgh's commercialisation service*. 2025. ⇒.

[22] Lisa Ehrlinger and Wolfram Wöß. "Towards a Definition of Knowledge Graphs". In: Sept. 2016. ⇒.

[23] Shahul Es et al. *Ragas: Automated Evaluation of Retrieval Augmented Generation*. 2025. arXiv: `2309.15217 [cs.CL]`. ⇒.

[24] Jerome H Friedman. “Greedy function approximation: a gradient boosting machine”. In: *Annals of statistics* (2001), pp. 1189–1232. ⇒.

[25] Yunfan Gao et al. *Retrieval-Augmented Generation for Large Language Models: A Survey*. 2024. arXiv: `2312.10997 [cs.CL]`. ⇒.

[26] Zitian Gao and Yihao Xiao. *Enhancing Startup Success Predictions in Venture Capital: A GraphRAG Augmented Multivariate Time Series Method*. 2025. arXiv: `2408.09420 [q-fin.CP]`. ⇒.

[27] GeoNames. *GeoNames: geographical database*. 2025. ⇒.

[28] Paul Gompers and Josh Lerner. “The venture capital revolution”. In: *Journal of economic perspectives* 15.2 (2001), pp. 145–168. ⇒.

[29] Kevin Gurney. *An introduction to neural networks*. CRC press, 2018. ⇒.

[30] Will Hamilton, Zhitao Ying, and Jure Leskovec. “Inductive representation learning on large graphs”. In: *Advances in neural information processing systems* 30 (2017). ⇒.

[31] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Series in Statistics)*. Springer, 2009. ⇒.

[32] Mike Havey and Amazon Neptune Team. “Find and link similar entities in a knowledge graph using Amazon Neptune, Part 2: Vector similarity search”. In: *AWS Database Blog* (2024). ⇒.

[33] Marti A. Hearst et al. “Support vector machines”. In: *IEEE Intelligent Systems and their applications* 13.4 (1998), pp. 18–28. ⇒.

[34] Aidan Hogan et al. “Knowledge Graphs”. In: *ACM Computing Surveys* 54 (July 2021), pp. 1–37. DOI: `10.1145/3447772`. ⇒.

[35] IBM. *Green Coding*. 2023. ⇒.

[36] IETF. *Common Format and MIME Type for Comma-Separated Values (CSV) Files*. 2005. ⇒.

[37] Chao Jin et al. “RAGCache: Efficient Knowledge Caching for Retrieval-Augmented Generation”. In: *arXiv preprint arXiv:2404.12457* (2024). ⇒.

[38] Project Jupyter. *Jupyter Notebook: interactive computing environment*. 2025. ⇒.

[39] Evangelos Kanoulas. "Building Reliable Test and Training Collections in Information Retrieval". PhD thesis. Jan. 2009. ⇒.

[40] Steven N Kaplan and Josh Lerner. "Venture capital data: Opportunities and challenges". In: *Measuring entrepreneurial businesses: Current knowledge and challenges* (2016), pp. 413–431. ⇒.

[41] T Jaya Lakshmi and S Durga Bhavani. "Link prediction approach to recommender systems". In: *Computing* 106.7 (2024), pp. 2157–2183. ⇒.

[42] Loïc Lannelongue et al. "Ten simple rules to make your computing more environmentally sustainable". In: *PLOS Computational Biology* 17.9 (Sept. 2021). Ed. by Russell Schwartz, e1009324. ISSN: 1553–7358. DOI: `10.1371/journal.pcbi.1009324`. ⇒.

[43] Patrick Lewis et al. *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks*. 2021. arXiv: `2005.11401 [cs.CL]`. ⇒.

[44] Donghe Li et al. "Knowledge graph-enhanced large language model for domain-specific question answering systems". In: *Authorea Preprints* (2024). ⇒.

[45] Mufei Li, Siqi Miao, and Pan Li. "Simple is effective: The roles of graphs and large language models in knowledge-graph-based retrieval-augmented generation". In: *arXiv preprint arXiv:2410.20724* (2024). ⇒.

[46] Yunqing Li et al. "Exploiting Knowledge Graphs in Industrial Products and Services: A Survey of Key Aspects, Challenges, and Future Perspectives". In: *Engineering Applications of Artificial Intelligence* 119 (2023), pp. 105–128. ⇒.

[47] Johannes Luef et al. "A recommender system for investing in early-stage enterprises". In: *Proceedings of the 35th Annual ACM Symposium on Applied Computing*. 2020, pp. 1453–1460. ⇒.

[48] Ninda Lutfiani et al. "Artificial intelligence based on recommendation system for startup matchmaking platform". In: *2022 IEEE Creative Communication and Innovative Technology (ICCIT)*. IEEE. 2022, pp. 1–5. ⇒.

[49] Reza Yousefi Maragheh et al. *ARAG: Agentic Retrieval Augmented Generation for Personalized Recommendation*. 2025. arXiv: `2506.21931 [cs.IR]`. ⇒.

[50] Markets and Markets Research. "Knowledge Graph Market Size & Share, Global Forecast, Industry Trends". In: *Markets and Markets Industry Report* (2024). ⇒.

[51] Costas Mavromatis and George Karypis. "Gnn-rag: Graph neural retrieval for large language model reasoning". In: *arXiv preprint arXiv:2405.20139* (2024). ⇒.

[52] Empathy First Media. "Leveraging User Feedback for Better RAG Systems". In: *YouTube Technical Content* (2024). ⇒.

[53] David Molina. "Driving Profit with LLMs and RAG: Practical Applications and Case Studies". In: *LinkedIn Professional Networks* (2024). ⇒.

[54] NetworkX Developers. *NetworkX - Documentation Homepage*. 2024. ⇒.

[55] Noumena Capital. *Noumena Capital: Strategic Investment and Advisory Services*. 2020. ⇒.

[56] Pandas Development Team. *Pandas - Documentation Homepage*. 2024. ⇒.

[57] Noah John Pettit. "Venture capitalists' decision-making under changing resource availability: Exploring the use of evaluative selection criteria". In: *University of Pennsylvania Doctoral Dissertation* (2024). ⇒.

[58] Tyler Thomas Procko and Omar Ochoa. "Graph retrieval-augmented generation for large language models: A survey". In: *2024 Conference on AI, Science, Engineering, and Technology (AIxSET)*. IEEE. 2024, pp. 166–169. ⇒.

[59] Python Software Foundation. *pickle — Python object serialization*. 2025. ⇒.

[60] Mona Razaghzadeh Bidgoli, Iman Raeesi Vanani, and Mehdi Goodarzi. "Predicting the success of startups using a machine learning approach". In: *Journal of Innovation and Entrepreneurship* 13.1 (2024), p. 80. ⇒.

[61] Tencent Cloud Research. "How do graph databases optimize query performance when handling large-scale data". In: *Tencent Cloud Techpedia* (2025). ⇒.

[62] Alireza Salemi and Hamed Zamani. "Evaluating Retrieval Quality in Retrieval-Augmented Generation". In: *arXiv preprint arXiv:2404.13781* (2024). ⇒.

[63] Aniruddha Salve et al. *A Collaborative Multi-Agent Approach to Retrieval-Augmented Generation Across Diverse Data*. 2024. arXiv: `2412.05838 [cs.AI]`. ⇒.

[64] Amazon Web Services. *Amazon EC2 Instance Types Documentation*. 2025. ⇒.

[65] Amazon Web Services. *Amazon EC2 Instance Types: c6a.2xlarge*. 2025. ⇒.

[66] Amazon Web Services. *Amazon EC2 Instance Types: m5.2xlarge*. 2025. ⇒.

[67] Amazon Web Services. *Amazon EC2 Instance Types: r6i.2xlarge*. 2025. ⇒.

[68] Amazon Web Services. *Amazon EC2 Instance Types: t2.medium*. 2025. ⇒.

[69] Agnieszka Skala and Agnieszka Skala. "Characteristics of startups". In: *Digital Startups in Transition Economies: Challenges for Management, Entrepreneurship and Education* (2019), pp. 41–91. ⇒.

[70] Shadullah Sartaj Syed. "A "tinder for startups": building a matchmaking engine for new ventures and Venture Capital firms". MA thesis. University of Edinburgh, School of Informatics, 2023.

[71] Yan-Martin Tamm, Rinchin Damdinov, and Alexey Vasilev. "Quality metrics in recommender systems: Do we calculate metrics consistently?" In: *Proceedings of the 15th ACM conference on recommender systems*. 2021, pp. 708–713. ⇒.

[72] CodeContent Research Team. "Optimizing RAG Performance: Enhancing AI with User Feedback and Multimodal Data". In: *CodeContent AI Blog* (2024). ⇒.

[73] Coralogix Research Team. "RAG in Production: Deployment Strategies and Practical Considerations". In: *AI Engineering Blog* (2025). ⇒.

[74] Data-Mania Research Team. "Top-Down Market Sizing for Tech Startups". In: *Data-Mania Technical Blog* (2025). ⇒.

[75] Databricks Research Team. "Long Context RAG Performance of LLMs". In: *Databricks Engineering Blog* (2024). ⇒.

[76] Empathy First Media Research Team. "Building Multi-Agent RAG Systems: A Step-by-Step Implementation Guide". In: *Empathy First Media Technical Blog* (2025). ⇒.

[77] Inclusion Cloud Research Team. "What Are Multiagent Systems? The Future of AI in 2025". In: *Inclusion Cloud Insights Blog* (2025). ⇒.

[78] Label Studio Research Team. "Seven Ways Your RAG System Could be Failing and How to Fix Them". In: *Label Studio Technical Blog* (2025). ⇒.

[79] Memgraph Research Team. "Optimizing Graph Databases through Denormalization". In: *Memgraph Technical Blog* (2023). ⇒.

[80] Neptune.ai Research Team. "Knowledge Graphs With Machine Learning: Web Scraping and NLP Integration". In: *Neptune.ai Technical Blog* (2025). ⇒.

[81] NetworkX Development Team. *NetworkX Documentation: Deployment Patterns for Production Systems*. 2024. ⇒.

[82] Patronus AI Research Team. “RAG Evaluation Metrics: Best Practices for Evaluating RAG Systems”. In: *Patronus AI Technical Documentation* (2025). ⇒.

[83] Ragie Research Team. “The Architect’s Guide to Production RAG: Navigating Challenges and Building Scalable AI”. In: *Ragie Technical Blog* (2025). ⇒.

[84] Smythos Research Team. “The Future of Multi-Agent Systems: Trends, Challenges, and Opportunities”. In: *Smythos Developer Resources* (2025). ⇒.

[85] Stripe Research Team. “A guide to SaaS pricing models”. In: *Stripe Business Resources* (2023). ⇒.

[86] SuperAnnotate Research Team. “Optimizing RAG systems with fine-tuning techniques”. In: *SuperAnnotate Technical Blog* (2025). ⇒.

[87] Superlinked Research Team. “Scaling RAG for Production”. In: *VectorHub by Superlinked* (2024). ⇒.

[88] Toloka Research Team. “RAG evaluation: a technical guide to measuring retrieval-augmented generation”. In: *Toloka AI Technical Blog* (2025). ⇒.

[89] Vorecol Editorial Team. “The Impact of User Experience (UX) Design on Adoption Rates of Change Management Software”. In: *Vorecol HR Management Systems Blog* (2024). ⇒.

[90] White Prompt Design Team. “The Impact of Seamless UX/UI on Software Adoption Rates”. In: *LinkedIn Professional Networks* (2024). ⇒.

[91] WiseCube AI Research Team. “20 Real-World Industrial Applications of Knowledge Graphs”. In: *WiseCube AI Industry Analysis* (2025). ⇒.

[92] Wrike Research Team. “How to Create a Go-To-Market Strategy: 8 Step Framework”. In: *Wrike Project Management Guide* (2024). ⇒.

[93] The University of Edinburgh. *College Research Ethics and Integrity Committee*. 2025. ⇒.

[94] Uber Blog. *Food Discovery with Uber Eats: Using Graph Learning to Power Recommendations*. 2019. ⇒.

[95] UoE Research Services. *UoE Research - Ethics and Integrity Guidelines*. 2025. ⇒.

[96] Petar Veličković et al. “Graph attention networks”. In: *arXiv preprint arXiv:1710.10903* (2017). ⇒.

[97] Shoujin Wang et al. “Graph Learning based Recommender Systems: A Review”. In: *Proceedings of the International Joint Conference on Artificial Intelligence* (2021), pp. 4644–4652. ⇒.

[98] Wikipedia. *Black swan theory*. 2025. ⇒.

[99] Wikipedia. *Computer security*. 2025. ⇒.

[100] Wikipedia. *Feature Engineering*. 2025. ⇒.

[101] Wikipedia. *Supervised learning*. 2025. ⇒.

[102] Sen Wu et al. “Understanding the Impact of Startups’ Features on Investor Recommendation Task via Weighted Heterogeneous Information Network”. In: *Complexity* 2021.1 (2021), p. 6657191. ⇒.

[103] Ruiyun Xu, Hailiang Chen, and J Leon Zhao. “SocioLink: Leveraging relational information in knowledge graphs for startup recommendations”. In: *Journal of Management Information Systems* 40.2 (2023), pp. 655–682. ⇒.

[104] Shuqi Xu et al. “Recommending investors for new startups by integrating network diffusion and investors’ domain preference”. In: *Information Sciences* 515 (2020), pp. 103–115. ⇒.

[105] Rui Yang et al. “Graphusion: a RAG framework for Knowledge Graph Construction with a global perspective”. In: *arXiv preprint arXiv:2410.17600* (2024). ⇒.

[106] Tiannuo Yang et al. *Demystifying and Enhancing the Efficiency of Large Language Model Based Search Agents*. 2025. arXiv: `2505.12065 [cs.AI]`. ⇒.

[107] Tianyi Zhang et al. *BERTScore: Evaluating Text Generation with BERT*. 2020. arXiv: `1904.09675 [cs.CL]`. ⇒.

[108] Xiaoxue Zhao, Weinan Zhang, and Jun Wang. “Risk-hedged venture capital investment recommendation”. In: *Proceedings of the 9th ACM Conference on Recommender Systems*. 2015, pp. 75–82. ⇒.

[109] Hao Zhong. “Venture capital investment: from rule of thumb to data science”. PhD thesis. Rutgers University-Graduate School-Newark, 2019. ⇒.

[110] Rong Zhu et al. “Aligraph: A comprehensive graph neural network platform”. In: *arXiv preprint arXiv:1902.08730* (2019). ⇒.