Domain-specific data gathering and exploitation

Nabil Moncef Boukhatem, Davide Buscaldi, Leo Liberti

Domain-specific data gathering and exploitation

Nabil Moncef Boukhatem[1,3], Davide Buscaldi[2], and Leo Liberti[3]

[1] LundiMatin, Paris, France moncef-nabil.boukhatem@lundimatin.fr
[2] LIPN CNRS, Université de Paris-Nord, Villetaneuse, France, buscaldi@lipn.univ-paris13.fr
[3] LIX CNRS Ecole Polytechnique, Institut Polytechnique de Paris, 91128 Palaiseau, France, liberti@lix.polytechnique.fr

Abstract. This paper addresses challenges in domain-specific data acquisition for NLP, particularly where real-world data is limited due to privacy or availability constraints. Two primary approaches are explored: synthetic data generation using models like GANs and VAEs, and advanced data cleaning techniques to maximize existing datasets. Additionally, the role of large language models (LLMs) and multimodal models in automating data tagging, filtering, and preprocessing is discussed. Case studies in healthcare, finance, and cybersecurity demonstrate the effectiveness of these methods. The paper concludes by highlighting future directions, including scalability, privacy concerns, and the integration of LLMs in domain-specific applications.

1 Introduction

In specific domains, the scarcity of domain-specific data is a pervasive challenge that has significantly impacted several of our experimental endeavors, as in [5] or in the study of cataphoras. This issue is not confined to academic research; it is also a prevalent problem in industrial applications, one that I have encountered frequently in my professional work. Addressing this challenge has pushed me to look for potential solutions, two of which have shown considerable promise depending on the specific goals and expected outcomes.

The first approach involves the generation of synthetic data [37, 18, 17] to compensate for the deficiencies in available domain-specific datasets. This method allows for the creation of controlled, contextually relevant datasets that can mitigate the limitations posed by the absence of sufficient real-world data.

The second approach focuses on the comprehensive utilization of all accessible data, regardless of its initial quality or completeness. This strategy includes the application of advanced data cleaning techniques, chunking methods with overlapping segments to preserve context, and the use of late-chunking with long-context embeddings. Additionally, this approach integrates the use of metadata such as tags [27], and the deployment of multimodal models for data extraction and processing, complemented by sophisticated information classification techniques to filter out irrelevant data. These techniques, along with the concept of unlearning [10] are crucial for preventing the degradation of model performance due to the inclusion of non-essential or noisy data.

2 Creation, Filtering, and Cleansing of Synthetic Data

introduction

Synthetic data generation [37, 18, 17] involves creating artificial data that closely resembles real-world data, which is particularly useful for training LLMs in domains where labeled data is scarce or sensitive, difficult to acquire or use due to legal, ethical, or logistical challenges, making it a viable solution in situations where obtaining authentic data is problematic. This process is essential for expanding the variety and volume of training data by mirroring the statistical properties of real data, enabling models to learn from a broader range of scenarios.

In fields such as healthcare, finance, or cybersecurity, access to real data is often restricted. For instance, healthcare data is protected by regulations like HIPAA in the U.S. and GDPR in the EU, which limit the sharing of identifiable patient information [1]. Similarly, financial transaction data is proprietary, and cybersecurity threat intelligence is sensitive, making it challenging to share such data for research purposes [44]. In these contexts, synthetic data generation plays a crucial role in creating datasets that replicate the properties of real data while maintaining privacy and security. This enables the training, testing and validation of models in environments where real data is scarce or inaccessible.

However, to ensure the effectiveness of synthetic data, it must be meticulously filtered and cleansed, ensuring that it meets quality and relevance standards. Techniques such as data augmentation, noise injection, and domain adaptation are commonly used to enhance the quality and applicability of synthetic datasets.

The primary purpose of synthetic data generation is to augment training datasets, especially when real-world data is either insufficient, like in our experimentations [5], or unavailable due to privacy concerns. By using techniques like data augmentation, where existing data is modified to create new examples, and generative models like Generative Adversarial Networks (GANs) [11] that can create entirely new data instances for neural models and especially LLMs to be trained more effectively. These methods improve the robustness and specialization capabilities of models, ensuring they perform well across the domain-specific tasks they are trained for.

Effective data exploitation involves strategically selecting, preprocessing, and augmenting large-scale datasets to maximize the performance of models. This process includes fine-tuning models on domain-specific data and leveraging techniques such as transfer learning and domain adaptation to ensure that models can generalize well across various tasks. The goal is to fully utilize available data to train models that are both versatile and robust, capable of handling a wide range of applications.

Techniques for Generating Synthetic Data

Generating synthetic data involves various techniques, each with its own set of advantages and challenges. Below, we explore some of them.

Generative Adversarial Networks Generative Adversarial Networks (GANs) [11], have become one of the most popular methods for generating synthetic data. GANs consist of two competing neural networks: a generator that produces synthetic data and a discriminator that attempts to distinguish between real and synthetic data. Over time, the generator learns to produce data that is increasingly realistic.

In domain-specific contexts, GANs have been used extensively. For example, in healthcare, GANs have been used to generate synthetic medical images for training diagnostic models or health records [43, 6]. In finance, GANs have been applied to create synthetic stock market data, which helps in testing trading algorithms under different market conditions [34]. In cybersecurity, GANs have been utilized to generate synthetic network traffic that mimics both normal and malicious behaviors, aiding in the development of intrusion detection systems [33, 28].

However, GANs have limitations, including the requirement for large amounts of training data and computational power. Additionally, GANs can sometimes suffer from mode collapse, where the generator produces limited diversity in the synthetic data [14].

Variational Autoencoders Variational Autoencoders (VAEs) are another type of generative model, first introduced in [15]. VAEs work by encoding the input data into a latent space and then decoding this representation back into the original data space. The model learns to generate new data by sampling from the latent space, which is assumed to follow a known distribution, typically a Gaussian distribution.

VAEs are particularly effective for generating structured data. In healthcare, they have been used to generate synthetic Electronic Health Records (EHR) [16] that preserve the correlations and relationships between variables, allowing for the testing of predictive models. VAEs have also been applied in other domains, such as generating synthetic sensor data in IoT applications [22].

A notable advantage of VAEs is their ability to generate diverse datasets with relatively small amounts of training data. However, VAEs can struggle with capturing highly complex dependencies, especially in high-dimensional data, which may limit their effectiveness in certain applications.

Synthetic Data Generators Based on Rule-Based Systems Rule-based systems are often employed in domains where the data generation process is well understood and can be described by a set of rules or heuristics [20]. These systems are particularly useful for generating data that must adhere to specific constraints or distributions.

For instance, in finance, synthetic data for algorithmic trading simulations can be generated using rules derived from historical market behaviors, such as price movements, trading volumes, and market events [2].

While rule-based systems offer precise control over the data generation process, they are less flexible than machine learning-based approaches. They require

extensive domain knowledge and may not generalize well to new or unexpected scenarios.

Hybrid Approaches A hybrid approache may combine multiple techniques to leverage their respective strengths. For example, a rule-based system might be used to generate an initial synthetic dataset, which is then refined using a GAN or VAE to introduce variability and complexity.

These hybrid methods might be particularly valuable in domains with both structured and unstructured data. For example, in the healthcare domain, a hybrid approach might generate synthetic patient records by first creating a structured dataset using rules based on medical guidelines, and then enhancing it with a GAN to include realistic variability in patient outcomes.

Hybrid approaches, might be complex to implement and require careful tuning to balance the contributions of each component. Their effectiveness often depends on deep domain-specific knowledge and the careful integration of different models.

Impact of LLMs on Synthetic Data Generation The advent of LLMs, starting with GPT-3 then GPT-4 [25], has introduced new possibilities in the realm of synthetic data generation. LLMs are pre-trained on vast amounts of text data and are capable of generating highly coherent and contextually relevant text. Their impact on synthetic data generation is multi-faceted:

- **Text Data Generation:** LLMs can generate large volumes of synthetic text data that mimic the style, structure, and content of domain-specific texts, such as legal documents, medical records, or financial reports. This is particularly valuable in domains where textual data is prevalent and access to real-world data is restricted due to privacy concerns.
- **Data Augmentation for NLP:** LLMs can be used to augment existing datasets by generating paraphrases, summaries, or expansions of existing data. This can help in creating more diverse and robust datasets for training NLP models.
- **Enhancing Rule-Based Systems:** LLMs can be integrated into rule-based synthetic data generation systems to introduce natural language understanding and generation capabilities. This can improve the realism of synthetic data by allowing the generation of text that better reflects human language patterns.
- **Cross-Domain Applications:** LLMs, due to their general-purpose nature, can be adapted to generate synthetic data across multiple domains, thereby reducing the need for domain-specific models. This is particularly useful in scenarios where developing separate models for each domain would be resource-intensive.
- **Automated Data Generation Pipelines:** LLMs can be used to create automated pipelines for synthetic data generation by understanding instructions or prompts given in natural language. This lowers the barrier to entry for non-experts who need to generate synthetic data for specific use cases.

Reverse Prompting Using LLMs for Dataset Generation

Introduction to Reverse Prompting Reverse prompting is an innovative approach to generating datasets using LLMs. In traditional prompting, the model is given a prompt or question and is expected to generate a corresponding answer. However, in reverse prompting, the process is inverted: given an answer or a statement, the model generates a corresponding question or prompt. This method is particularly useful when working with pre-existing data that are straightforward declarative sentences. For example, if we have a dataset consisting of company data in the form of statements or facts, reverse prompting can be employed to generate a conversation-like format where these statements are converted into a dialogue structure such as *[User: Question, Agent: Answer]*.

Methodology The primary objective of utilizing reverse prompting was to transform the existing straightforward data into a question-answer format that aligns with the discussion style often required in natural language processing tasks. Given the declarative statements from the company's data, the challenge was to formulate relevant questions that would lead to these statements as answers. This format is not only more suitable for training dialogue models but also enhances the dataset's utility across various applications.

To achieve this, a few-shot learning approach was employed. Few-shot learning involves providing the LLM with a limited set of examples to guide its output. The examples provided to the model illustrated the desired input-output format, showing how a straightforward statement (which acts as the answer) could be transformed into a corresponding question. By carefully selecting and crafting these few-shot examples, the model was able to learn the pattern required to generate appropriate questions for the given answers in the demanded format.

Implementation The implementation involved the following steps:

- **Data Preparation:** The initial dataset consisted of declarative sentences from the company's data. These sentences were selected and analyzed to understand their structure and content, which informed the crafting of few-shot examples.
- **Few-Shot Example Creation:** A small set of examples was created manually. Each example consisted of a declarative statement (the answer) paired with a relevant question that would lead to that answer. These examples were designed to capture various types of questions—such as "wh'' questions—to ensure the model could generalize across different scenarios.
- **Model Few Shot prompting:** The few-shot examples were fed into a pre-trained LLM. Using these examples, the model was prompted to generate questions for the remaining statements in the dataset. The model's outputs were then reviewed and adjusted as necessary to ensure they matched the desired format and quality.
- **Data Validation:** The generated question-answer pairs were rigorously tested to ensure their accuracy and relevance. This involved comparing the

generated questions to the original statements to confirm that the questions were logically consistent and contextually appropriate.

Results and Impact on Dataset Generation The use of reverse prompting with an LLM (Mixtral [12] in our case) proved to be highly effective in converting straightforward declarative sentences into question-answer pairs. The generated datasets now align with the desired discussion format, enabling their direct application in training dialogue systems and other conversational AI models.

The results of this approach significantly enhanced the quality and usability of the generated datasets. The reverse-prompted data showed improved contextual relevance and variety in the generated questions, which not only enriched the datasets but also improved the performance of downstream models that utilized this data for training.

Additionally, the implementation of reverse prompting as part of the data preprocessing pipeline demonstrated its utility in large-scale dataset generation. It allowed for the efficient transformation of existing data into more versatile formats, reducing the need for extensive manual curation and enabling quicker iterations in model development.

Overall, reverse prompting using LLMs has proven to be a valuable technique in the preprocessing and generation of high-quality datasets, particularly in scenarios where the data needs to be converted into a dialogue-friendly format. This approach is now integrated into the company's standard preprocessing workflow for dataset generation, underscoring its effectiveness and adaptability in various applications.

Evaluation of Synthetic Data Quality

Evaluating the quality of synthetic data is critical to ensuring that it can effectively replace or complement real-world data in domain-specific applications. Several metrics and methodologies are used to assess the quality of synthetic data.

Statistical Similarity Statistical similarity between synthetic and real data is often assessed using metrics such as means, variances, correlations, and distribution shapes of key variables [30]. Tools like the Kolmogorov-Smirnov (K-S) test [24], Earth Mover's Distance (EMD) [19], and Maximum Mean Discrepancy (MMD) [32] are commonly employed to quantitatively assess the similarity.

For instance, in finance, the statistical properties of synthetic time series data, such as volatility clustering and autocorrelation, are compared to those of real financial data to ensure realism [2]. In healthcare, the distributions of synthetic patient demographics and clinical outcomes are compared with those of real patient data to ensure that synthetic EHR are representative [40].

Utility for Models The utility of synthetic data for models is often evaluated by training models on synthetic data and testing them on real-world data. If the models trained on synthetic data perform comparably to those trained on real data, it suggests that the synthetic data captures the essential features of the domain.

For example, in cybersecurity, synthetic network traffic is used to train Intrusion Detection Systems (IDS). The performance of the IDS on real traffic data provides a measure of the quality of the synthetic data [7]. Similarly, in healthcare, models trained on synthetic medical images are tested on real images to validate the utility of the synthetic data in diagnostic applications [26].

Privacy and Security Considerations In domains where privacy and security are critical, it is essential to ensure that synthetic data does not inadvertently expose sensitive information. Differential privacy is a common framework used to quantify and guarantee privacy in synthetic data generation [13]. It ensures that the inclusion or exclusion of any single individual in the original dataset does not significantly affect the generated synthetic data.

For instance, synthetic health records are evaluated for privacy risks using differential privacy techniques to ensure that no individual's health information can be reconstructed from the synthetic dataset [23]. In finance, synthetic transaction data is assessed to ensure that it does not reveal patterns that could be traced back to real trading strategies [2].

Challenges and Future Directions Despite the advances in synthetic data generation, several challenges remain. One significant challenge is ensuring the diversity and realism of synthetic data across different scenarios within a domain. For example, generating synthetic data that accurately reflects rare events, such as medical complications or financial crises, remains a difficult task.

Another challenge is the computational cost associated with generating high-quality synthetic data. Techniques like GANs and VAEs require substantial computational resources, which can be a barrier in resource-constrained environments [14].

The integration of LLMs into synthetic data generation presents both opportunities and challenges [35]. On the one hand, LLMs can significantly enhance the quality and diversity of synthetic data, particularly in text-heavy domains. On the other hand, the reliance on LLMs may introduce new challenges related to the interpretability and bias of the generated data. Moreover, the high computational demands of LLMs may exacerbate existing resource constraints, particularly for small organizations or research teams.

Future research is likely to focus on improving the efficiency and scalability of synthetic data generation techniques, including those that incorporate LLMs. Advances in unsupervised learning and transfer learning may offer new ways to generate high-quality synthetic data with fewer resources. Additionally, developing more sophisticated methods for evaluating synthetic data quality, partic-

ularly in terms of privacy and utility, will be crucial for the continued adoption of synthetic data in domain-specific applications.

Conclusion The combination of data synthesis, exploitation, and cleansing ensures that models are trained on high-quality, diverse datasets, leading to more robust and reliable models. These processes are fundamental in developing AI systems that can operate effectively in specialized domains, providing more accurate and contextually relevant outputs while reducing the dependency on large quantities of real-world data. Through these techniques, AI models become more adaptable, scalable, and capable of delivering high performance across various domain-specific applications.

3 Data Extraction and Cleansing, An Approach Using Multimodal Models

Data extraction and cleansing are crucial processes in data-driven research, particularly in the era of big data [9], where the quality of data significantly impacts the reliability and validity of analytical outcomes. Traditional data extraction techniques often rely on structured databases and predefined schemas, making them less effective when dealing with unstructured or semi-structured data. Moreover, data cleansing—ensuring that the extracted data is accurate, complete, and consistent—often involves labor-intensive processes that require domain expertise.

Data cleansing [31], the process of ensuring that the extracted data is accurate, complete, and consistent—often involves labor-intensive processes that require domain expertise, as detecting and correcting (or removing) corrupt or inaccurate records from a dataset, is a crucial step in the data preparation pipeline. High-quality data is essential for producing reliable and accurate insights in data-driven research and applications. Traditionally, data cleansing has been handled using a combination of rule-based methods, statistical techniques, and manual interventions.

However, as datasets grow in complexity and heterogeneity, these traditional approaches face significant challenges. With the advent of LLMs and particularly Multimodal Language Models (MLM) [29, 42], new opportunities have arisen to enhance and automate data extraction and cleansing processes, especially in environments where data is derived from various modalities such as text, images, and structured tabular data as these models can integrate and analyze multiple types of data—text, images, audio, and even video.

MLMs can help automate the extraction and cleansing process by identifying and correcting errors, removing noise, and ensuring that the data is totally extracted from documents and is of high quality. This step is crucial for maintaining the reliability and effectiveness of AI models, especially in technical and commercial applications where data accuracy is of high importance

Traditional Data Extraction Techniques

Traditional data extraction techniques have long served as the backbone for retrieving and processing data from structured and semi-structured sources. These methods include the use of Structured Query Language (SQL) for querying relational databases, Extract, Transform, Load (ETL) [39] processes for data warehousing, web scraping for collecting data from websites, and API (Application Programming Interface) for structured data retrieval from online services. Each of these approaches has proven effective in specific contexts, particularly when dealing with well-organized data that fits neatly into predefined schemas. For instance, SQL excels in environments where data is stored in relational databases, offering powerful querying capabilities, while ETL processes are widely used to consolidate data from multiple sources into data warehouses.

However, these traditional methods are not without their limitations. SQL and ETL processes are heavily dependent on structured data, making them less effective for unstructured or semi-structured data sources such as free-text documents, multimedia content, or irregularly formatted datasets. Web scraping, while versatile, often faces legal, ethical, and technical challenges, including issues with data quality and site access restrictions. APIs provide a more structured way to extract data but come with constraints such as rate limits, access restrictions, and the need for continuous adaptation due to versioning and deprecation. These challenges have become more pronounced as the volume and variety of data have increased, highlighting the need for more advanced extraction methods.

As data grows more complex and diverse, encompassing various forms such as text, images, audio, and video, the limitations of traditional data extraction techniques become increasingly evident. These methods often struggle to effectively handle unstructured data or integrate information from different modalities. Furthermore, the scalability of these techniques is a concern, as processing large volumes of data can be computationally intensive and time-consuming. The need for more sophisticated data extraction methods has led to the exploration of new technologies, particularly those involving AI and machine learning.

In response to these challenges, the development and application of LLMs, especially MLMs, have emerged as a promising solution. Unlike traditional techniques, MLMs are capable of processing and integrating data across multiple modalities [4], such as text, images, and audio, within a single framework. This approach not only automates the extraction process but also enhances the accuracy and efficiency of data handling, making it a powerful tool for modern data extraction and cleansing tasks.

Data Extraction Using MLMs

The use of MLMs in data extraction [4] involves several steps.

Data Ingestion MLMs begin by ingesting data from various sources. These sources can include text documents, scanned images of handwritten notes, au-

dio recordings of interviews, or even videos containing both audio and visual data. The LLM processes these different modalities simultaneously, enabling it to extract information that spans across multiple data types.

Feature Extraction For each modality, the LLM identifies and extracts relevant features. For text, this might include keywords, named entities, or sentiment. For images, it could involve object detection, image classification, or even text extraction using Optical Character Recognition (OCR). Audio data may be transcribed into text, followed by speech sentiment analysis, while video data can be dissected into individual frames and analyzed for both visual and auditory features.

Contextual Integration One of the key strengths of MLMs is their ability to integrate context across different data types. For example, when extracting data from a research paper that includes both text and images (such as charts or tables), the LLM can correlate the textual description with the visual data, ensuring a more accurate and contextually relevant extraction.

Data Structuring Once the relevant data has been extracted, the LLM structures it into a format suitable for further analysis. This might involve converting unstructured text into structured datasets, categorizing images, or aligning audio transcripts with corresponding visual data. The structured data can then be stored in databases or directly used for analysis.

Challenges

Despite their advanced capabilities, MLMs face several challenges in data extraction. Accurately integrating data from different modalities—such as text, images, and audio—requires sophisticated algorithms to understand the nuanced relationships between these diverse data types, and any misinterpretation or misalignment can result in incorrect extraction [41]. Additionally, the quality of input data plays a crucial role in the LLM's performance, as noisy data like low-resolution images, poor-quality audio, or incomplete text can significantly hinder the model's ability to accurately extract and interpret information. Furthermore, processing large volumes of multimodal data is computationally intensive, making it a significant challenge to ensure that the LLM can scale efficiently while maintaining high levels of accuracy.

Traditional Data Cleansing Techniques

.

Conventional data cleansing techniques [31] primarily involve several steps, including the detection of missing values, outlier identification, data deduplication, inconsistency resolution, and standardization. These methods often rely

on predefined rules or heuristics that may not generalize well across different datasets. For instance, regular expressions can be employed to validate email addresses, while statistical models might detect outliers in numerical data. However, these approaches are typically limited by their dependence on domain-specific knowledge and the inability to handle unstructured data or multimodal inputs efficiently.

The Role of MLMs in Data Cleansing

MLMs, such as those that combine text, image, and structured data understanding [42], are trained on diverse datasets, enabling them to process and integrate information from various modalities. This capability is particularly valuable in data cleansing, where datasets may include complex and unstructured elements such as textual descriptions, images, and numerical data.

Textual Data Cleansing In the context of textual data, LLMs excel at understanding and generating human-like text, which can be leveraged to identify and correct errors in textual datasets. For example, LLMs can be employed to detect and suggest corrections for spelling and grammatical errors, standardize textual data (e.g., transforming different date formats into a unified format), and even infer missing values based on context. The ability of these models to comprehend context allows them to outperform traditional rule-based methods, particularly in detecting and correcting subtle errors that might be missed by simpler algorithms.

Image Data Cleansing For image data, MLMs can be used to detect anomalies or inconsistencies by comparing the visual content of images against expected patterns or by cross-referencing image data with associated textual descriptions. For instance, in a dataset containing product images and descriptions, an LLM can flag images that do not match their descriptions, thereby identifying potential errors in the dataset. This approach is particularly useful in large-scale datasets where manual inspection would be prohibitively time-consuming.

Structured Data Cleansing When dealing with structured data, such as tables or databases, MLMs can assist in identifying inconsistencies across different data types. For example, an LLM might be used to check for logical inconsistencies in a dataset where numerical, categorical, and textual data are intertwined. By leveraging the model's understanding of context and semantics, it can detect discrepancies that might not be evident through traditional rule-based cleansing techniques.

Integration of MLMs in the Data Cleansing Pipeline Integrating MLMs into the data cleansing pipeline involves several key steps:

- **Data Ingestion:** The raw data from various sources and modalities is ingested into the system. This data can include text, images, structured tables, and more.
- **Data Preprocessing:** The data is preprocessed to a format suitable for the LLM, involving steps like tokenization for text, resizing and normalization for images, and encoding for structured data.
- **Error Detection:** The MLM is used to identify potential errors in the dataset. For textual data, this could involve detecting anomalies in grammar or syntax. For images, it could involve flagging visually inconsistent data. For structured data, it could involve identifying outliers or logical inconsistencies.
- **Error Correction:** Based on the model's output, corrections are either suggested to the user or applied automatically. The corrections are informed by the model's understanding of context and are validated against existing data to ensure accuracy.
- **Post-Processing and Validation:** After the corrections have been made, the data is reprocessed to ensure that no new errors have been introduced. This step may involve additional validation against external datasets or domain-specific rules.

Advantages and Challenges MLMs offer significant advantages in data cleansing by automating the process, thereby reducing the need for time-consuming and error-prone manual interventions. Their ability to understand the context of data enables more accurate detection and correction of errors compared to traditional methods. Moreover, their versatility in handling multiple data modalities simultaneously allows these models to effectively cleanse complex datasets that include a combination of text, images, and structured data.

However, MLMs, while powerful, come with challenges such as high computational requirements, making them resource-intensive and potentially limiting their use in environments with constrained resources. Additionally, the decision-making process of these models is often opaque, complicating the understanding of why specific corrections are made, which can be problematic in domains where transparency is essential. Furthermore, these models can inherit biases from their training data, leading to biased error detection and correction, making it crucial to address these biases to ensure fair and accurate data cleansing.

Conclusion

MLMs represent a transformative approach to data cleansing, offering the ability to automate and enhance the accuracy of the process across diverse datasets. By leveraging the contextual and multimodal understanding of these models, it is possible to address some of the limitations of traditional data cleansing methods. However, the integration of these models into data cleansing workflows must be done thoughtfully, considering the challenges of computational demands, interpretability, and bias. As these technologies continue to evolve, they hold the potential to significantly improve the quality and reliability of data in a wide range of applications.

4 Data tagging and filtering for RAG applications

In domain-specific applications, where the precision and relevance of information retrieval are critical, the implementation of RAG demands meticulous attention to data preprocessing to which we wanted to add the tagging and filtering options to obtain better performances for our domain-specific application. This section provides an in-depth exploration of the methodologies, tools, and frameworks utilized in our project to optimize data extraction, cleaning, and filtering, ensuring that the RAG model performs at its best within the specific context of our domain.

Data Extraction and Cleaning Techniques

The initial stages of our project involved rigorous experimentation with various data extraction and cleaning techniques to prepare the dataset for RAG application. We began by evaluating several tools and frameworks that offered diverse approaches to data preprocessing, each with its strengths and limitations, we aimed to ensure that the data fed into our RAG system was not only relevant but also preserved the contextual integrity necessary for accurate information retrieval and generation.

One of the key challenges we encountered was maintaining the semantic coherence of documents, especially when dealing with lengthy texts that could be fragmented during the chunking process.

Apache Tika We initially used Apache Tika for data extraction from a wide range of file formats, including PDFs, Word documents, and HTML pages. Tika's ability to extract metadata and text content from different formats proved beneficial in standardizing the input data. However, Tika's limitations in handling complex document structures led us to explore more sophisticated extraction methods.

SpaCy and NLTK For NLP tasks, including tokenization, lemmatization, and Named Entity Recognition (NER), we integrated SpaCy [38] and NLTK [3]. These tools were essential in the early stages for cleaning and structuring the text data. However, these tools primarily operate at the token level, which sometimes resulted in fragmented context when dealing with larger text blocks.

LangChain Framework To overcome the challenges posed by traditional chunking methods, we adopted LangChain [36], a powerful framework designed for managing long documents in NLP tasks. LangChain's ability to handle chunking with overlap was particularly valuable in preserving the semantic continuity of the text. By configuring the overlap between chunks, after multiple tests, we mitigated the risk of losing context when concepts were mentioned in one chunk and referred to in another. This overlap ensures that the RAG model receives

a continuous stream of relevant information, improving its ability to generate contextually accurate responses.

- **Chunk Size Optimization:** We experimented with different chunk sizes and overlap parameters using LangChain's built-in tools. This iterative process involved fine-tuning the balance between chunk size and overlap to maximize the retention of context without excessively increasing the input size to the model, which could degrade performance or increase computational costs.

Exploration of Late Chunking with Long-Context Models Recognizing the potential of long-context models like GPT-4 [25] and Claude, we have been exploring late chunking strategies. These models are capable of handling extended sequences of text, allowing for chunking at a later stage in the processing pipeline. Late chunking helps maintain a more extensive context within a single chunk, reducing the need for frequent context switching and potentially improving the model's ability to understand and generate responses that are semantically richer and more coherent. This approach is particularly promising for documents that require a deep understanding of complex relationships between different sections.

Classification Models for Optimizing Data Exploitation for RAG and Finetuning

In the development of our RAG system, we recognized the importance of efficiently managing and filtering input prompts and datasets to ensure the system operates with optimal performance. To achieve this, we trained two distinct classification models: one designed to determine when RAG should be applied, and another aimed at automating the filtering of unusable data from large datasets. This section outlines the methodologies, datasets, and outcomes associated with these models, highlighting their roles in enhancing the overall efficiency and effectiveness of our RAG application.

Detecting the Need for RAG: Small Talk vs. Domain-Specific Questions One of the critical challenges in deploying a RAG system within a domain-specific application is ensuring that the system only engages its computationally intensive retrieval mechanisms when necessary. To address this, we trained a classification model to distinguish between prompts that required RAG and those that did not. Specifically, the model was designed to identify whether a given prompt was a domain-specific question that necessitated RAG or if it was a small talk or out-of-topic query that could be handled with a simpler, more efficient response mechanism.

Dataset and Model Training:

- *Handcrafted Small Talk Dataset:* We curated a custom dataset of "small talk'' examples, which included common conversational phrases, greetings,

and out-of-topic questions. This dataset was essential in training the model to recognize non-domain-specific queries that did not require RAG.
- *Domain-Specific Dataset:* In contrast, we compiled a dataset of domain-specific questions derived from the company's internal knowledge base and user interactions. This dataset represented the type of prompts that would necessitate the use of RAG for accurate and contextually appropriate responses.
- *Model Architecture:* We used a fine-tuned version of a pre-trained transformer model (BERT [8] and RoBERTa [21]) for text classification. The model was trained on a balanced combination of the small talk and domain-specific datasets, allowing it to learn the nuanced differences between conversational queries and those that required in-depth retrieval from the domain-specific knowledge base.
- *Training Process:* The model underwent several iterations of training and validation to optimize performance metrics such as accuracy, precision, and recall. We are planning to employ data augmentation to address class imbalance issues in the dataset.

Outcome and Integration

The resulting classification model demonstrated high accuracy in distinguishing between small talk and domain-specific questions. By integrating this model into the RAG system's query preprocessing pipeline, we were able to significantly reduce unnecessary computational overhead. The model efficiently routed small talk queries to a lightweight response generator while reserving the RAG system's resources for prompts that truly required domain-specific retrieval and generation.

This approach not only improved system efficiency but also enhanced the user experience by delivering faster responses to trivial queries while maintaining high-quality, contextually rich responses for more complex questions.

Automated Data Filtering for Finetuning: Usable vs. Unusable Data

As part of our efforts to optimize the training and fine-tuning processes of models, we identified the need to filter out unusable data from large datasets. Unusable data—such as irrelevant, outdated, or low-quality content—can adversely affect model performance by introducing noise and leading to an unlearning process where the model starts to lose the ability to accurately retrieve and generate relevant information [10].

To address this challenge, we trained a second classification model aimed at automating the detection and removal of unusable data from our datasets.

Dataset and Model Training

- *Large-Scale Company Dataset:* We utilized a vast dataset owned by the company, which included both usable (high-quality, relevant information) and unusable data. The unusable data comprised content that was irrelevant to the domain, contained significant errors, or was otherwise unsuitable for fine-tuning.

- *Annotation and Labeling:* A portion of this dataset was manually labeled to create a ground truth for training. Usable and unusable data were clearly defined, with specific guidelines developed to ensure consistent labeling. This labeled data formed the basis for training the classification model.
- *Model Architecture:* For this task, we employed a BERT [8] model, similar to the one used in the first classification task, but fine-tuned specifically for binary classification of usable versus unusable data. The model was designed to understand the context and quality of text, learning to identify patterns that distinguished high-value content from noise.
- *Training Process:* The model was trained on a subset of the labeled dataset, with validation performed on a separate holdout set to ensure generalization. We applied techniques such as cross-validation, regularization, and early stopping to prevent overfitting and to ensure that the model could accurately classify data across various contexts and content types.

Outcome and Integration

The trained model achieved robust performance in classifying usable and unusable data, with high precision and recall metrics indicating its reliability in filtering out low-quality content. By integrating this model into our data preprocessing pipeline, we automated the process of cleaning large datasets before they were used for fine-tuning purposes.

This automation proved to be a significant time-saver, enabling the team to process and prepare large volumes of data with minimal manual intervention, especially for Proof-of-Concepts (PoC) and prototyping. Moreover, by ensuring that only high-quality, relevant data was used in the fine-tuning process, we mitigated the risks associated with unlearning and hallucinations [10] and maintained the integrity and performance of the models over time.

Automatic Tagging of Documents and Queries for RAG Filters

In the context of modern information retrieval systems, particularly those utilizing RAG models, the effective and efficient tagging of documents and their associated queries is critical. This section outlines the design and implementation of an automatic tagging system that accelerates document access, facilitates precise filtering, and minimizes the loss of contextual meaning during chunk processing.

Overview of the Automatic Tagging System The system's primary function is to automatically generate and assign tags to documents and their corresponding chunks. These tags are derived from the subject matter and key concepts identified within the text. By leveraging advanced natural language processing models, specifically a BERT classifier, the system analyzes each document to determine the most relevant tags. These tags encompass a broad spectrum of attributes, including the document's subject, specific keywords, and any user-defined tags that may be manually added.

These tags, once assigned, are stored in a vector database, allowing for efficient retrieval and filtering processes. The inclusion of both automatically generated and user-specified tags ensures that the system remains flexible and adaptable to various use cases and user preferences. The tags are not only instrumental in categorizing the documents but also play a crucial role during the retrieval phase, where they help to surface the most relevant content based on user queries.

Tagging Process and Vector Database Storage The tagging process begins with the decomposition of the document into smaller, contextually coherent chunks. Each chunk is then analyzed individually to determine its key themes and concepts. The BERT classifier, trained on a diverse corpus, is employed to generate a set of tags for each chunk, reflecting its content and context within the larger document.

In addition to the automatically generated tags, users can manually indicate specific tags that are particularly relevant to their needs or the organizational context. These manually added tags are integrated with the automatically generated tags to form a comprehensive tagging profile for each document and its chunks.

Once the tags are assigned, they are stored in a vector database. The vector representation of these tags allows for efficient similarity comparisons during the retrieval process, enabling the system to quickly identify and rank the most pertinent documents based on the tags and the contextual embeddings generated for both the documents and the queries.

Retrieval Process and Filtering Mechanism When a user initiates a search, the system begins by tagging the user's query using the same BERT classifier that was utilized for the documents. The tagging of the query is informed by the user's profile, which may include predefined tags such as brand, country, or other contextual attributes. This ensures that the query is enriched with relevant metadata, enhancing the precision of the retrieval process.

The system then filters the stored documents by comparing the tags associated with the query to those assigned to the documents and chunks. This filtering mechanism narrows down the search space to the most relevant documents, effectively prioritizing those with the highest similarity scores.

To further refine the retrieval, the system evaluates the similarity between the query and the document chunks using two key methods:

- **Title Similarity:** An LLM is employed to generate a summary title for each chunk, capturing its essence based on the content of the chunk, the overall document, and preceding context. The similarity between this title and the user query is assessed, contributing to the relevance score.
- **Chunk Embeddings:** The system also calculates the similarity between the query and the chunk embeddings. These embeddings capture the semantic content of the chunks, allowing for a nuanced comparison that considers both explicit and implicit meanings within the text.

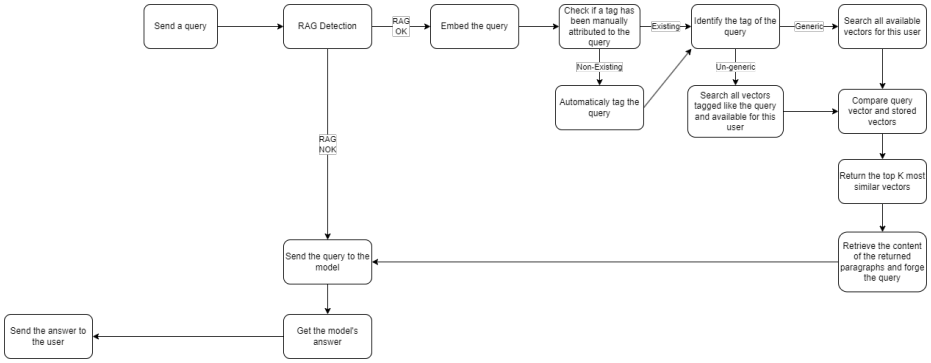

Fig. 1: An illustration of the RAG process including the RAG detector and tag filtering functions

Performance and User Satisfaction The implementation of this automatic tagging system has resulted in significant improvements in both the speed and accuracy of document retrieval. Users have reported high levels of satisfaction, particularly appreciating the system's ability to surface highly pertinent documents quickly. The categorization and tagging of documents have become more streamlined, reducing the manual effort required and ensuring that the most relevant content is always accessible.

Moreover, the integration of automatic and manual tags within a unified system has provided a balanced approach, combining the strengths of algorithmic precision with the flexibility of user input. The reduction in retrieval time, coupled with the enhanced relevance of results, underscores the effectiveness of this approach in managing large document repositories within a RAG framework.

This automatic tagging system not only enhances the user experience but also lays a robust foundation for further advancements in the field of information retrieval, particularly in scenarios involving complex and diverse document sets.

Conclusion

The development and integration of these two classification models into our RAG application pipeline have been instrumental in enhancing the system's overall efficiency and effectiveness. The first model, by accurately identifying when RAG is necessary, reduces unnecessary computational loads and ensures that resources are allocated appropriately. The second model, by automating the filtering of unusable data, safeguards the quality of the training datasets, preventing degradation in model performance and supporting the long-term sustainability of the system while also simplifying prototyping tasks.

These advancements underscore the importance of targeted classification models, instead of using abusively LLMs, in optimizing complex AI systems, particularly in domain-specific applications where precision, efficiency, and data quality are paramount. As we continue to refine these models and explore new

techniques, we anticipate further improvements in the performance and scalability of our RAG system.

References

1. A. Appenzeller, M. Leitner, P. Philipp, E. Krempel, and J. Beyerer, *Privacy and utility of private synthetic data for medical data analyses*, Applied Sciences, 12 (2022).
2. S. A. Assefa, D. Dervovic, M. Mahfouz, R. E. Tillman, P. Reddy, and M. Veloso, *Generating synthetic data in finance: opportunities, challenges and pitfalls*, in Proceedings of the First ACM International Conference on AI in Finance, 2020, pp. 1–8.
3. S. Bird, *Nltk: the natural language toolkit*, in Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, 2006, pp. 69–72.
4. A. Biswas and W. Talukdar, *Robustness of structured data extraction from in-plane rotated documents using multi-modal large language models (llm)*, Journal of Artificial Intelligence Research, (2024).
5. N. M. Boukhatem, D. Buscaldi, and L. Liberti, *Empirical comparison of semantic similarity measures for technical question answering*, in New Trends in Database and Information Systems, S. Chiusano, T. Cerquitelli, R. Wrembel, K. Nørvåg, B. Catania, G. Vargas-Solar, and E. Zumpano, eds., Cham, 2022, Springer International Publishing, pp. 167–177.
6. E. Choi, S. Biswal, B. Malin, J. Duke, W. F. Stewart, and J. Sun, *Generating multi-label discrete patient records using generative adversarial networks*, in Machine learning for healthcare conference, PMLR, 2017, pp. 286–305.
7. C. G. Cordero, E. Vasilomanolakis, A. Wainakh, M. Mühlhäuser, and S. Nadjm-Tehrani, *On generating network traffic datasets with synthetic attacks for intrusion detection*, ACM Transactions on Privacy and Security (TOPS), 24 (2021), pp. 1–39.
8. J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *BERT: Pre-training of deep bidirectional transformers for language understanding*, in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1, Minneapolis, Minnesota, 2019, ACL, pp. 4171–4186.
9. S. García, S. Ramírez-Gallego, J. Luengo, J. M. Benítez, and F. Herrera, *Big data preprocessing: methods and prospects*, Big data analytics, 1 (2016), pp. 1–22.
10. Z. Gekhman, G. Yona, R. Aharoni, M. Eyal, A. Feder, R. Reichart, and J. Herzig, *Does fine-tuning llms on new knowledge encourage hallucinations?*, 2024.
11. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, *Generative adversarial networks*, Communications of the ACM, 63 (2020), pp. 139–144.
12. A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. S. Chaplot, D. de las Casas, E. B. Hanna, F. Bressand, G. Lengyel, G. Bour, G. Lample, L. R. Lavaud, L. Saulnier, M.-A. Lachaux, P. Stock, S. Subramanian, S. Yang, S. Antoniak, T. L. Scao, T. Gervet, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed, *Mixtral of experts*, 2024.

13. J. Jordon, J. Yoon, and M. Van Der Schaar, *Pate-gan: Generating synthetic data with differential privacy guarantees*, in International conference on learning representations, 2018.
14. T. Karras, S. Laine, and T. Aila, *A style-based generator architecture for generative adversarial networks*, in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 4401–4410.
15. D. P. Kingma, M. Welling, et al., *An introduction to variational autoencoders*, Foundations and Trends® in Machine Learning, 12 (2019), pp. 307–392.
16. D. Lee, H. Yu, X. Jiang, D. Rogith, M. Gudala, M. Tejani, Q. Zhang, and L. Xiong, *Generating sequential electronic health records using dual adversarial autoencoder*, Journal of the American Medical Informatics Association, 27 (2020), pp. 1411–1419.
17. H. Li, X. Zhao, D. Guo, H. Gu, Z. Zeng, Y. Han, Y. Song, L. Fan, and Q. Yang, *Federated domain-specific knowledge transfer on large language models using synthetic data*, arXiv preprint arXiv:2405.14212, (2024).
18. Z. Li, H. Zhu, Z. Lu, and M. Yin, *Synthetic data generation with large language models for text classification: Potential and limitations*, arXiv preprint arXiv:2310.07849, (2023).
19. H. Ling and K. Okada, *An efficient earth mover's distance algorithm for robust histogram comparison*, IEEE transactions on pattern analysis and machine intelligence, 29 (2007), pp. 840–853.
20. R. Liu, B. Fang, Y. Y. Tang, and P. P. Chan, *Synthetic data generator for classification rules learning*, in 2016 7th International Conference on Cloud Computing and Big Data (CCBD), IEEE, 2016, pp. 357–361.
21. Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, *Roberta: A robustly optimized BERT pretraining approach*, CoRR, (2019).
22. M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, *Conditional variational autoencoder for prediction and feature recovery applied to intrusion detection in iot*, Sensors, 17 (2017), p. 1967.
23. H. Murtaza, M. Ahmed, N. F. Khan, G. Murtaza, S. Zafar, and A. Bano, *Synthetic data generation: State of the art in health care domain*, Computer Science Review, 48 (2023), p. 100546.
24. T. Necasova and D. Svoboda, *Visual and quantitative comparison of real and simulated biomedical image data*, in Proceedings of the European Conference on Computer Vision (ECCV) Workshops, 2018, pp. 0–0.
25. OpenAI, *Gpt-4 technical report*, 2024.
26. V. C. Pezoulas, D. I. Zaridis, E. Mylona, C. Androutsos, K. Apostolidis, N. S. Tachos, and D. I. Fotiadis, *Synthetic data generation methods in healthcare: A review on open-source tools and methods*, Computational and Structural Biotechnology Journal, (2024).
27. M. Poliakov and N. Shvai, *Multi-meta-rag: Improving rag for multi-hop queries using database filtering with llm-extracted metadata*, 2024.
28. S. Rahman, S. Pal, S. Mittal, T. Chawla, and C. Karmakar, *Syn-gan: A robust intrusion detection system using gan-based synthetic data for iot security*, Internet of Things, 26 (2024), p. 101212.
29. W. Rahman, M. K. Hasan, S. Lee, A. Zadeh, C. Mao, L.-P. Morency, and E. Hoque, *Integrating multimodal information in large pretrained transformers*, in Proceedings of the conference. Association for Computational Linguistics. Meeting, vol. 2020, NIH Public Access, 2020, p. 2359.

30. F. Ramzan, C. Sartori, S. Consoli, and D. Reforgiato Recupero, *Generative adversarial networks for synthetic data generation in finance: Evaluating statistical similarities and quality assessment*, AI, 5 (2024), pp. 667–685.
31. F. Ridzuan and W. M. N. W. Zainon, *A review on data cleansing methods for big data*, Procedia Computer Science, 161 (2019), pp. 731–738.
32. J. Snoke, G. M. Raab, B. Nowok, C. Dibben, and A. Slavkovic, *General and specific utility measures for synthetic data*, Journal of the Royal Statistical Society Series A: Statistics in Society, 181 (2018), pp. 663–688.
33. O. S. Striuk and Y. P. Kondratenko, *Generative adversarial networks in cybersecurity: Analysis and response*, in Artificial Intelligence in Control and Decision-making Systems: Dedicated to Professor Janusz Kacprzyk, Springer, 2023, pp. 373–388.
34. S. Takahashi, Y. Chen, and K. Tanaka-Ishii, *Modeling financial time-series with generative adversarial networks*, Physica A: Statistical Mechanics and its Applications, 527 (2019), p. 121261.
35. R. Tang, X. Han, X. Jiang, and X. Hu, *Does synthetic data generation of llms help clinical text mining?*, arXiv preprint arXiv:2303.04360, (2023).
36. O. Topsakal and T. C. Akinci, *Creating large language model applications utilizing langchain: A primer on developing llm apps fast*, in International Conference on Applied Engineering and Natural Sciences, vol. 1, 2023, pp. 1050–1056.
37. J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield, *Training deep networks with synthetic data: Bridging the reality gap by domain randomization*, in Proceedings of the IEEE conference on computer vision and pattern recognition workshops, 2018, pp. 969–977.
38. Y. Vasiliev, *Natural language processing with Python and spaCy: A practical introduction*, No Starch Press, 2020.
39. P. Vassiliadis, A. Simitsis, and S. Skiadopoulos, *Conceptual modeling for etl processes*, in Proceedings of the 5th ACM international workshop on Data Warehousing and OLAP, 2002, pp. 14–21.
40. J. Walonoski, M. Kramer, J. Nichols, A. Quina, C. Moesel, D. Hall, C. Duffett, K. Dube, T. Gallagher, and S. McLachlan, *Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic electronic health care record*, Journal of the American Medical Informatics Association, 25 (2018), pp. 230–238.
41. S. Wu, H. Fei, X. Li, J. Ji, H. Zhang, T.-S. Chua, and S. Yan, *Towards semantic equivalence of tokenization in multimodal llm*, arXiv preprint arXiv:2406.05127, (2024).
42. S. Wu, H. Fei, L. Qu, W. Ji, and T.-S. Chua, *Next-gpt: Any-to-any multimodal llm*, arXiv preprint arXiv:2309.05519, (2023).
43. X. Yi, E. Walia, and P. Babyn, *Generative adversarial network in medical imaging: A review*, Medical image analysis, 58 (2019), p. 101552.
44. J. Yoon, M. Mizrahi, N. F. Ghalaty, T. Jarvinen, A. S. Ravi, P. Brune, F. Kong, D. Anderson, G. Lee, A. Meir, et al., *Ehr-safe: generating high-fidelity and privacy-preserving synthetic electronic health records*, NPJ Digital Medicine, 6 (2023), p. 141.