21017 - EARS

Environment Adaptive Recommendation System

WP 2 - Use Case Description & Requirements Specification

Deliverable 2.2: Report on SotA & Innovation

Deliverable Type: Document

Classification: Public

Due Date: September 2024

HISTORY

Document Version	Date	Remarks
V1	**17.07.2024**	**First Version**

Table of Contents

1. Executive Summary

1.1. Project Objectives

The goal of the EARS project is to address the challenges faced by current recommender systems by providing a comprehensive and adaptable system with multiple technical outputs. The main goals of the project can be summarized as follows:

- Enabling a federated and cross-domain approach to improving recommender systems: The project aims to create a platform where recommender systems from different domains can collaborate and learn from each other without sharing raw data, improving accuracy and solving problems such as cold start and data sparsity.
- Incorporating explainable capabilities into recommendation engines: EARS aims to develop explainable AI models that provide transparent recommendations, increasing user trust and addressing the problem of opacity in existing recommender systems.
- Preserving data privacy across silos: The project focuses on ensuring that data privacy is not compromised while different domains collaborate, leveraging techniques such as federated learning and homomorphic encryption.
- Benefiting from intellectual assets shared on the core platform: EARS envisions an ecosystem where algorithm developers and solution providers can share and access assets such as algorithms and models across the platform, fostering innovation and accelerated development.

1.2. Work Package Objectives

The main objectives of this Work Package are to identify and specify the project use cases, to conduct an extensive literature, legislation and industry review of existing technologies and approaches. The research will focus on each identified use case domain, allowing a clear identification of the EARS contribution, beyond the state of the art. We will specify the functional and non-functional requirements of EARS.

1.3. Scope of Deliverable

The objectives of this document are to detail the work undertaken during the execution of Task 2.2: SoTA: Description and Innovation, which aims to elaborate a more detailed description of the State of the Art (SoTA) and provide a clear identification of the solutions beyond SoTA that EARS will develop within the scope of this project.

2. Introduction

2.1. Context of the EARS Innovation Project

The global recommendation engine market was valued at US$ 1.77 billion in 2020 and is expected to expand at a compound annual growth rate (CAGR) of 33.0% from 2021 to 2028. The increasing adoption of digital technologies among organizations is also leading to a surge in demand for recommendation engine solutions. The COVID-19 pandemic impacted several industries, leading to a noticeable shift in the way businesses operate and shoppers shop. These changes are expected to endure post-pandemic as well, leaving a lasting impact on businesses and individuals.

Recommendation systems are becoming crucial for modern subscription businesses. Netflix has stated that 80% of streaming provenance time for recommended movies and shows. The EARS project addresses key challenges in the domain such as privacy concerns, data scarcity, cold start problem, and the need for diverse recommendations through a set of innovative solutions. It employs a federated learning-based architecture that improves privacy by allowing data to remain on local nodes, mitigating the risks of centralization. To address cold start and data sparsity issues, the project integrates hybrid recommendation algorithms that combine content-based and collaborative filtering, improving the accuracy and personalization of recommendations. Furthermore, the use of explainable AI models aims to improve transparency, fostering user trust by providing clear explanations of how recommendations are generated.

The federated learning-based recommendation system to be developed aims to provide users with more personalized and trustworthy recommendations while preserving their privacy. Unlike traditional AI/ML-based recommendation systems, the federated learning approach allows users' data to be processed across devices and platforms without sharing it directly, thus ensuring the quality of the recommendation system while protecting users' personal information.

Collaboration in the EARS project

Each country participating in the EARS project brings unique contributions, enriching the project with diverse experiences and perspectives.

Beia from Belgium focuses on improving the customer experience in e-commerce through innovative components such as a powerful search engine, a user-controllable recommendation engine and a user-friendly chatbot. These solutions aim to revolutionize the online shopping experience of Flemish retailers, resulting in increased sales and customer loyalty. In addition, Beia modules will be available for integration into the use cases of other project partners, fostering collaboration and knowledge sharing.

Portugal, represented by a consortium, prioritizes the development of a recommendation engine with explainable capabilities within the EARS framework. Its focus on the retail market for safety and security products aims to empower non-expert customers with personalized product recommendations, thus promoting safety and health in the workplace. The consortium's commitment to scientific and industrial collaboration ensures wide dissemination and utilization of the project results.

Spain's participation is marked by Glintt's expertise in providing secure and explainable recommendations in the healthcare field, especially in cancer treatment. In addition, DEXTRO's innovations in hospital logistics and Square1's innovative recommendation services in the media sectors further enrich the scope of the project. By leveraging advanced data analytics within a secure framework, the Spanish consortium aims to deliver market-leading products and maintain its competitive edge.

The Turkish consortium aims to address the problem of cold start and data scarcity in recommendation engine development by proposing a hybrid platform that is both privacy-protecting and explainable. This solution has broad applicability across multiple domains in the Turkish market, and promises to fill a crucial gap in current systems. ARD Group, ADESSO, Dogus Teknoloji and DonanimHaber each bring unique capabilities to the project, ranging from software integration to media recommendation systems. Their collaborative efforts not only aim to improve existing systems, but also expand into new domains, ensuring widespread impact and commercial success.

The EARS project, realized thanks to contributions from Belgium, Portugal, Spain and Turkey, will be implemented with a versatile approach. This project aims to improve user experience, make recommendation systems more reliable and increase operational efficiency in various sectors. The main outcome of the project will be a framework encompassing user-friendly interfaces, explainable and secure recommendation systems. This solution will have a wide range of applications, including increasing customer loyalty in e-commerce, providing more effective recommendations for cancer treatment in the healthcare sector, identifying in-demand content in the media sector and optimizing operational processes in various industries. The EARS project aims to increase the competitiveness of participating countries while providing societal benefits through international collaboration and the integration of innovative technologies.

2.2. Definition and Key Concepts

The EARS project seeks to create an adaptive recommendation system that addresses the limitations of current systems.

Key Concepts:

● Federated learning: Allows machine learning models to be trained on different distributed data sets without sharing the raw data. In EARS, it is used to enhance data privacy while improving the accuracy of the recommendation system.

● Multi-domain recommendations: Leverage information from multiple domains to provide more accurate and personalized recommendations. EARS uses a black-box approach to access relevant information from other domains without violating privacy restrictions.

● Explainable AI (XAI): Provides transparency in the recommendation process by explaining to users why certain items are recommended. EARS integrates XAI to foster user trust.

● Hybrid recommendation engine: It combines content-based and collaborative filtering techniques to overcome the limitations of each method. EARS seeks to develop hybrid algorithms to address problems such as cold start and data sparsity.

● EARS platform: A robust and scalable system that integrates the key components of the project, such as recommendation algorithms, federated learning frameworks, and XAI models. This platform serves as a hub to connect different systems and facilitate collaboration between partners.

The EARS project is based on the collaboration of partners from Belgium, Portugal, Spain, and Turkey, each of whom brings specific expertise and knowledge.

Impact on different sectors:

The EARS project aims to impact several sectors including:

- **E-commerce and media:** Provide personalized and relevant recommendations to online consumers, taking into account their preferences and the ever-changing landscape of online content.
- **Healthcare:** Improve patient care and operational efficiency by providing accurate and timely recommendations to healthcare professionals.
- **Retail:** Offer a personalized and engaging shopping experience that mimics interaction with a real shopping assistant.
- **Social media:** Address the cold start problem and improve user experience by leveraging information from other domains without compromising data privacy.

The success of the EARS project will be measured using specific, measurable, achievable, relevant and time-bound key performance indicators (KPIs).

In short, the EARS project seeks to revolutionize recommendation systems by addressing the key challenges of privacy, accuracy, transparency and adaptability. Its focus on federated learning, multi-domain recommendations, explainable AI, and hybrid recommendation engines, along with international collaboration, aims to create a more effective and trustworthy recommendation system with significant impact across diverse industries.

3. System Recommendation: Current Status

Recommender systems have become essential tools in multiple industries, from e-commerce to media streaming, due to their ability to personalize user experience and increase engagement and retention. Over the years, these systems have evolved significantly, both in their methodology and application.

1. Evolution of Recommendation Systems

Initially, recommendation systems were mainly based on Collaborative Filtering (CF) and Content-Based Filtering (CBF) techniques. Collaborative Filtering takes advantage of similarities in preferences between users to make recommendations, while Content-Based Filtering uses the characteristics of items (products, movies, etc.) that a user has rated to recommend similar ones.

However, these approaches face notable limitations, such as the data sparsity problem (when there is little information about user-item interactions) and the cold start problem (when a new user or a new product enters the system with no interaction history).

2. Recent Advances and Hybrid Methods

In recent years, recommender systems have incorporated advances in machine learning and deep learning, which has allowed the creation of more sophisticated and accurate models. Hybrid models, which combine collaborative filtering and content-based filtering, have gained popularity because they can overcome some of the limitations of traditional approaches.

For example, Neural Collaborative Filtering (NCF) is a technique that integrates neural models to capture nonlinear interactions between users and products, improving the accuracy of recommendations. In addition, Knowledge Graphs are being used to enrich recommender systems by providing a broader and more structured context of the relationships between users, products, and features.

3. Federated Learning and Privacy

With the growing focus on data privacy, Federated Learning has emerged as an innovative solution. This technique allows recommendation models to be trained on distributed data that remains on local devices, avoiding the need to centralize sensitive user information. This not only protects user privacy, but also allows companies to collaborate on improving their recommendation models without sharing data directly.

4. Explainable Recommendation and Transparency

Another key aspect in the evolution of recommendation systems is explainability. Users and organizations are increasingly demanding systems that not only provide accurate recommendations, but can also explain in a clear and understandable way why a particular recommendation was made. This has led to the development of Explainable AI (XAI) techniques, which allow recommendation systems to generate transparent explanations for their decisions.

5. Current Challenges and Future Trends

Despite advances, recommendation systems still face significant challenges. The problem of adaptability to changing user preferences and the ability to handle extremely sparse or noisy data remain active areas of research. Furthermore, the integration of multimodal data sources (such as text, images, and video) into recommendations represents another frontier to explore.

In the future, recommendation systems are expected to become even more sophisticated, leveraging techniques such as deep learning, generative models, and transfer learning to improve the personalization and effectiveness of real-time recommendations.

3.1. Recommendation Systems Models

3.1.1. Collaborative Filtering

Collaborative filtering (CF) is a popular technique in recommender systems that leverages user interactions (such as ratings, clicks, or purchases) to predict a user's preferences based on the behavior of similar users. This approach is widely used due to its effectiveness in capturing latent patterns in user behavior.

Federated Collaborative Filtering: Addressing privacy concerns, federated learning allows training collaborative filtering models across decentralized datasets without sharing raw user data (Ammad-Ud-Din, 2019). This method enhances user privacy, improves data security, and enables leveraging data from diverse sources. FedNCF (Federated Neural Collaborative Filtering) (Perifanis, 2022) is an approach that incorporates a privacy-preserving aggregation method which satisfies the security requirements and leads to faster convergence compared to the existing methods.

Graph Neural Networks (GNNs): GNNs have emerged as a powerful tool for collaborative filtering by modeling user-item interactions as a graph. Recent papers have explored various GNN architectures like LightGCN (He, 2020) and PinSage (Ying, 2018) for improved performance. The ability to capture high-order relationships, handling sparse data, and integrate side information are the main advantages of this approach.

Neighborhood-based Collaborative Filtering with Attention: Recent research has focused on incorporating attention mechanisms into traditional neighborhood-based methods to dynamically weigh the importance of neighbors based on user preferences and item characteristics. Wang et al., introduces a knowledge-aware attention network combining neighborhood information for recommendation (KCNR) that can discover structural and associative semantic information in the knowledge graph (Wang, 2023). KAT (Liu, 2024) is another knowledge-aware attentive recommendation model that integrates two-terminal neighbor features. This approach simultaneously extracts user and item features that enhances the distinction of adjacent entities, so the embeddings can effectively represent the potential semantics of users and items. It is demonstrated that this method outperforms many recent baselines.

Schafer et al. (2007) provide an extensive survey of collaborative filtering recommender systems, outlining the different types, including user-based, item-based, and model-based methods. Their work discusses the strengths and weaknesses of each approach, particularly in dealing with sparse data and scalability challenges. The authors also highlight the importance of incorporating contextual information to improve the quality of recommendations.

He et al. (2017) introduce Neural Collaborative Filtering (NCF), a deep learning-based approach that combines the strengths of both matrix factorization and deep neural networks. NCF models complex user-item interactions through multiple hidden layers, enabling it to capture non-linear patterns that traditional CF methods may miss. This technique has shown significant improvements in recommendation accuracy, particularly in environments with large datasets and diverse user preferences.

Xue et al. (2017) propose Deep Matrix Factorization models for recommender systems, extending traditional matrix factorization methods by integrating deep neural networks. Their approach aims to enhance the representation learning capability of matrix factorization by allowing the model to learn more complex patterns and interactions between users and items. This method demonstrates improved performance, especially in cases where user-item interaction data is sparse or noisy.

Covington, Adams, and Sargin (2016) discuss the application of deep neural networks in the YouTube recommendation system, showcasing how collaborative filtering techniques can be scaled to massive datasets. Their approach combines user engagement signals with content features to recommend videos, achieving high accuracy by effectively leveraging both explicit and implicit feedback.

Su et al. (2023) explore beyond traditional two-tower models in collaborative filtering by learning sparse retrievable cross-interactions for recommendation. Their innovative approach addresses the limitations of two-tower models by capturing fine-grained interactions between user and item representations, leading to more accurate and diverse recommendations.

Yang et al. (2020) investigate mixed negative sampling for learning two-tower neural networks in recommendation systems, highlighting the importance of selecting appropriate negative samples to improve model performance. Their findings suggest that mixed negative sampling can significantly enhance the learning process of collaborative filtering models, particularly in scenarios with sparse data or imbalanced interaction distributions.

Gosh et al. (2020) present a recommendation system for e-commerce using Alternating Least Squares (ALS) on Apache Spark, demonstrating the effectiveness of ALS in handling large-scale datasets. Their study highlights the scalability and efficiency of ALS in collaborative filtering, making it a suitable choice for real-time recommendation tasks.

Steck et al. (2021) offer a case study on Netflix's use of deep learning for recommender systems, detailing the architecture and methodologies employed to provide personalized content to millions of users. Their approach integrates collaborative filtering with deep learning techniques to achieve state-of-the-art recommendation performance, particularly in dealing with diverse content types and user preferences.

In conclusion, both content-based and collaborative filtering recommender systems have evolved significantly, incorporating advanced machine learning and deep learning techniques to address their respective challenges. While content-based methods excel in leveraging item features to generate recommendations, collaborative filtering is particularly effective in capturing latent patterns in user behavior. The integration of these techniques, along with innovative approaches such as neural collaborative filtering and deep matrix factorization, continues to push the boundaries of recommendation accuracy and diversity.

3.1.2. Content-Based Filtering

Content-based recommender systems primarily rely on the attributes or features of items to recommend similar items to users based on their preferences. These systems operate by creating a profile for each user or item and matching items with users whose profiles indicate a high likelihood of interest.

Multi-Modal Learning for Content Understanding: Combining information from different modalities like text, images, and audio enhances the understanding of item characteristics and user preferences and improves accuracy. With this approach personalized recommendations across different content types becomes possible. To enhance the specific characteristics of each modality, Pre-training Modality-Disentangled Representations Model (PAMD) was introduced by

Han et al. (Han, 2022), which utilizes pretrained VGG19 and Glove embeddings from visual and textual modalities and devises a disentangled encoder for extracting their modality-common characteristics while preserving the modality-specific characteristics. This method results in performance gain against a series of state-of-the-art alternatives. A Multi-Modal Graph Convolutional Network (M2GCN) is proposed in (Liu, 2023) for link prediction in multi-modal networks to predict adverse drug reactions.

Transfer Learning for Content-Based Recommendations: Leveraging pre-trained models from related domains can improve the performance of content-based filtering, especially in cold-start scenarios. Zhao et al., introduced a framework that facilitates knowledge transfer among various systems and addresses the problem of identifying entity-correspondence across different systems (Zhao, 2013). A parameter-efficient transfer learning architecture is introduced by Yuan et al. (Yuan, 2020) that addresses the problem of re-training models for new tasks. This approach can be adapted to various downstream tasks and allows the pre-trained parameters to remain unaltered during fine-tuning.

Lops, De Gemmis, and Semeraro (2011) provide a comprehensive overview of content-based recommender systems, detailing the evolution of methodologies and the integration of machine learning techniques to improve recommendation accuracy. Their review emphasizes the importance of feature extraction and representation in creating effective content-based models, highlighting advancements such as the use of natural language processing (NLP) and semantic analysis to better understand item descriptions and user preferences.

Aggarwal and Aggarwal (2016) expand on the theoretical foundation of content-based recommender systems, offering a detailed explanation of various algorithms and their practical applications. Their work discusses the challenges associated with these systems, such as over-specialization, where users are recommended items too similar to those they have already interacted with, and the inability to suggest novel or diverse items. The textbook also explores hybrid approaches that combine content-based techniques with collaborative filtering to overcome these limitations.

Pazzani and Billsus (2007) focus on the implementation aspects of content-based recommendation systems, illustrating various strategies for representing item content and user profiles. They propose methods for integrating user feedback and continuous learning to enhance recommendation quality over time. The authors also explore how these systems can be adapted to different domains, from movie recommendations to personalized news feeds.

Reddy et al. (2019) discuss a content-based movie recommendation system using genre correlation to improve recommendation accuracy. Their approach involves correlating different genres to suggest movies that align with user preferences, demonstrating the effectiveness of genre-based filtering in content-based systems. This method showcases how leveraging domain-specific content features can significantly enhance user satisfaction by providing more tailored recommendations.

3.1.3. Hybrid Models

Deep Fusion Models: Advanced architectures like deep factorization machines and neural collaborative filtering models are being developed to seamlessly integrate collaborative and content information within a single deep learning framework.

Two-Tower Method: In the realm of deep learning-based recommendation systems, the "two towers" architecture represents a prevalent design pattern which employs two distinct neural network architectures, one dedicated to representing users and the other to representing items. These networks, often likened to "towers," generate embeddings – dense vector representations – that encapsulate the salient features of each user and item.

Subsequently, these embeddings are combined, typically through a similarity measure such as the dot product, to predict the likelihood of a user-item interaction. This separation of user and item representations enables a more granular and comprehensive understanding of individual preferences and item attributes, ultimately contributing to the generation of more accurate and personalized recommendations.

Cross-Domain Hybrid Models: Leveraging data from multiple domains can improve recommendations in a target domain by transferring knowledge and addressing data sparsity issues.

Cross-domain recommendation systems leverage data and knowledge from multiple domains to enhance recommendation accuracy, addressing the cold start problem where new users or items lack sufficient interaction data. This field has seen significant advancements through various innovative approaches.

Fernández-Tobías et al. (2012) provide a foundational survey of cross-domain recommender systems, exploring methodologies and applications of transferring knowledge between different domains to improve recommendation accuracy. Their survey highlights techniques used to bridge domains, discussing challenges and future directions. By utilizing data from multiple domains, these systems can mitigate the cold start problem by leveraging auxiliary information from related domains (Fernández-Tobías et al., 2012). Building on this, Bi et al. (2020) introduce DCDIR, a deep cross-domain recommendation system specifically designed for cold start users in the insurance domain. Their approach employs deep learning techniques to model user preferences and item features across different domains, effectively transferring knowledge to improve recommendations for new users. This method demonstrates significant improvements in recommendation quality by addressing the lack of user interaction data through cross-domain information transfer (Bi et al., 2020).

Furthering the discussion, Zhang et al. (2017) propose a cross-domain recommender system focused on consistent information transfer between domains. Their approach ensures that the knowledge transferred from one domain to another is consistent and relevant, enhancing recommendation accuracy. This method addresses the cold start problem by leveraging reliable information from auxiliary domains, making it particularly effective for new users and items (Zhang et al., 2017). Complementing this, Ma et al. (2018) explore integrating cross-media content information into multi-domain recommendation systems. Their study demonstrates how data from social media, such as tweets, can reveal user preferences and be incorporated into recommendation models. By using cross-media information, the system can provide more accurate recommendations, especially for cold start users with limited interaction data in the primary domain (Ma et al., 2018).

Shapira et al. (2013) investigate the use of Facebook data for single and cross-domain recommendation systems. Their research shows how social media data can be utilized to enhance recommendation accuracy by providing additional context and user preferences. This approach is particularly useful for addressing the cold start problem, as it leverages the rich user interaction data available on social media platforms to improve recommendations in other domains (Shapira et al., 2013). Building on earlier work, Cremonesi et al. (2011) present one of the foundational studies on cross-domain recommender systems, exploring various techniques for transferring knowledge between domains. Their study highlights the potential of cross-domain approaches to improve recommendation accuracy by addressing the cold start problem. By integrating data from multiple domains, the system can provide more robust recommendations even when user interaction data is sparse (Cremonesi et al., 2011).

Expanding on these methodologies, Taneja and Arora (2018) introduce a cross-domain recommendation approach using multidimensional tensor factorization. This method models multiple types of interactions between users and items across different domains, enabling effective knowledge transfer. The approach enhances recommendation accuracy by capturing complex relationships and addressing the cold start problem through comprehensive data integration (Taneja & Arora, 2018).

Context-Aware Hybrid Models: Incorporating contextual information like time, location, and user mood can further enhance the personalization and relevance of recommendations.

3.2. Transfer Learning

Transfer Learning (TL) is a machine learning based approach in which a model developed for one task can be reused to develop a model for a second task (Zhuang et al, 2020, Weiss et al, 2016, Torrey et al, 2010). It is useful when the second task has limited data or computational resources but is still related to the first task. The idea is that the knowledge gained in solving one problem can be applied to a new but related problem, reducing the need to train a new model from scratch. TL improves learning efficiency, reduces training time, and enhances model performance, particularly in domains like computer vision and natural language processing. Some of the key concepts related to TL include the source identification in which the initial model is trained, and the domain where the model can be then applied. Additionally, both source and target tasks are also important, not only to establish the task in the source domain for which the model was originally trained but also the task in the target domain where the model needs to perform. Lastly, the feature space concept is also necessary for the representation of data (input features) in the source and target domains, which may or may not be the same. TL often relies on model reusability using pretrained models from large datasets as the base for transfer learning and then fine-tuned on the target task. Besides that, fine-tuning may also be considered by taking a pretrained model and adjusting its weights with new data from the target task, which helps in adapting the model to specific new data.

Transfer Learning Methods

There are different types of transfer learning methods, depending on the relationship between the source and target domains, tasks, and feature spaces.

- Inductive Transfer Learning (Torrey et al, 2010) - In this method, the target task is different from the source task, but the domains may be the same. The main goal is to transfer knowledge from the source task to improve performance on the target task.
- Fine-tuning (Vrbančič, 2020, Iman et al, 2023) - Adjusts pretrained model weights with new task-specific data.
- Parameter Freezing (Iman et al, 2023) – Holds some layers of the pretrained model and only trains the last few layers with the target task's data.
- Transductive Transfer Learning (Moreo et al, 2021) – This method is applied when the target domain is different from the source domain, but the tasks are the same. The goal is to transfer knowledge from the source domain to help solve the target domain problem.
- Domain Adaptation (Pan&Young, 2009) - A key approach in transductive TL, where techniques like instance re-weighting, feature transformation, and adversarial learning are used to align the feature spaces of the source and target domains.
- Unsupervised Transfer Learning (Niu et al, 2020) - Both the target task and target domain are different from those of the source. This is generally used in unsupervised settings where the target domain has unlabeled data.
- Multi-Task Learning (Pan&Young, 2009) - Instead of transferring knowledge after the learning process is complete, multi-task learning involves learning multiple tasks simultaneously. Tasks are trained in parallel with shared representations.
- Negative Transfer (Zhang et al, 2022) – This method occurs when the knowledge transfer from the source task negatively impacts the performance of the target task. This can happen if the source and target domains are too dissimilar, leading to a situation where the pretrained model's knowledge hinders rather than helps learning in the target domain.

- Neighborhood Enhanced Transfer Learning (Cai et al, 2019) - An advanced method that enhances TL by focusing on local data relationships or neighborhood structures in the target domain. It considers the similarity of data points in the target domain and applies transfer learning selectively, ensuring that the transfer is more context-sensitive and less likely to cause negative transfer.

3.3. Current Challenges

Despite significant advances in recommendation systems, there are still several challenges that need to be addressed to improve their effectiveness and applicability across various domains. These challenges are not only related to recommendation accuracy but also to aspects such as privacy, security, explainability, and cross-domain interoperability.

3.3.1. Data Privacy and Security

One of the most critical challenges in modern recommender systems is data privacy and security. Recommender systems rely heavily on the collection and analysis of large volumes of personal data, which poses significant risks to user privacy. These risks include the possibility of data breaches, re-identification of users from anonymized data, and misuse of personal information.

To mitigate these risks, approaches such as Federated Learning have been developed, which allows training of recommendation models without the need to centralize user data. In this approach, data remains on local devices and only the parameters of the trained models are shared, reducing the possibility of sensitive data exposure. In addition, homomorphic and differentially private encryption techniques are being explored to ensure that processed data cannot be used to extract specific individual information.

However, implementing and maintaining security in distributed systems remains a challenge, especially in terms of balancing model accuracy and data privacy. The adoption of these techniques must be careful to avoid compromising system functionality while protecting user privacy.

3.3.2. Explainability

Another key challenge in recommendation systems is explainability. As these systems become more complex, using advanced deep learning techniques and hybrid models, they also become more opaque, making it difficult to understand how recommendations are generated. This lack of transparency can lead to user distrust and limit the adoption of these systems, especially in sensitive areas such as healthcare and finance.

To address this issue, Explainable Artificial Intelligence (XAI) has been developed, which focuses on creating models that are not only accurate, but also understandable to users. XAI techniques can provide explanations that describe why a specific item was recommended, increasing user trust and facilitating informed decision making.

A common approach in explainability is the use of feature attribution models, such as LIME or SHAP, which help identify the most relevant features that influence a model's decision. Additionally, knowledge graphs are also used to provide explanatory paths between user preferences and the recommendations made.

Despite these advances, challenges remain, such as balancing explainability with model complexity, and how to provide explanations that are useful and understandable to different types of users, from tech experts to end users with limited technical knowledge.

3.3.3. Cross-Domain Recommendations

Traditional recommendation systems typically operate within a single domain, such as movies, music, or e-commerce products. However, users interact with multiple domains, which poses the opportunity and challenge of developing cross-domain recommendations (Cross-Domain Recommendations). This approach seeks to transfer knowledge and preferences from one domain to another to improve the quality of recommendations when little information is available in a specific domain (cold start problem).

The main challenge in cross-domain recommendations is how to manage differences in data and features across domains. Domains can vary significantly in terms of data structure, user distribution, and types of interactions. Furthermore, knowledge transfer across domains must be done in a way that maximizes the relevance and accuracy of recommendations without introducing bias or errors.

An emerging solution is the use of transfer learning and deep learning models that can capture and carry over common patterns across domains. For example, Multimodal Knowledge Graphs (MMKGs) can integrate and relate information from different sources and domains, allowing the system to generate more holistic and contextualized recommendations.

However, the implementation of cross-domain recommendation systems still faces barriers, such as data interoperability, privacy management in data exchange between domains, and the ability to adapt to changing user preferences.

4. Key Innovations of the EARS Project

The EARS project aims to introduce a new platform that provides federated and connected solutions, unlike the existing ones, where users can strengthen the accuracy of their recommendation systems with the feedback they receive from the results of other recommendation systems. The platform will allow different domains to work together in an architecture that allows information sharing. This new platform will address four key technological challenges:

- Enabling a cross-domain and federated approach to improving recommendation systems.
- Incorporating explainable capabilities into recommendation engines.
- Preserving data privacy across silos.
- Leveraging shared intellectual assets on the centralized platform.

These key innovations are detailed below:

- **Federated Learning**

Federated learning (FL) is a method for training AI models without direct access to the underlying data. The goal is to train algorithms across multiple parties without sharing private user data. For example, recommender systems are omnipresent in our online interactions, offering personalized suggestions derived from user data. However, conventional methods pose privacy risks by centralizing user data. FL addresses this by collaboratively training recommender system models on decentralized data stored on project partners' devices, safeguarding sensitive company information.

- **Domain-Specific Recommender System (RS) for End Users**

Recommender systems (RS) are a subdomain of information filtering that employ algorithms to generate personalized suggestions of items (products, services, information, or content) that are tailored to a user's specific interests and preferences. The two main paradigms in recommender system (RS) design are collaborative filtering and content-based filtering.

- Collaborative filtering: This is based on the premise that users who have had similar preferences in the past are likely to share preferences in the future.
- Content-based filtering focuses on item attributes and a user's past interactions with similar items, recommending items based on shared characteristics. This method is beneficial when complete item descriptions are available and user profiles are well-defined.

- **Explainable AI**

Explainable artificial intelligence (XAI) encompasses processes and techniques that enable human users to understand and trust decisions or predictions made by AI systems. It involves using interpretable and transparent methods to explain the reasoning behind AI model outputs. In the context of the EARS project, XAI plays a crucial role in providing insights into how federated recommender systems work and the reasons behind specific recommendations.
In federated learning-based recommender systems (FL-RS), incorporating explanations poses unique challenges due to the distributed nature of data and models. Two main approaches to generating explanations in FL-RS are:

- **Federated Explainable Learning (FedXAI):** This approach involves training local models to generate explanations along with predictions.
- **Explanation Aggregation:** This method involves training local models without explanation capabilities and then aggregating explanations on a central server after federated training, offering more flexibility but raising privacy concerns about intermediate results transmitted to the server.

- **Knowledge Graph-Based Recommender Systems**

In the EARS project, knowledge graph-based recommender systems represent a key technological innovation. These systems use knowledge graphs to capture and represent in a structured way the complex relationships between users, products, and other relevant entities. By integrating multiple data sources and contexts, knowledge graphs enable improved recommendation accuracy and relevance by offering a deeper understanding of user preferences and needs. This ability to connect diverse information and provide transparent explanations makes knowledge graphs critical to addressing problems such as data sparsity and the need for personalized recommendations in the EARS project.

- **Natural Language Processing (NLP)**

Natural Language Processing (NLP) is incorporated as an innovative technology in the EARS project to improve user interaction and understanding in recommendation systems. Through advanced NLP techniques, EARS is able to analyze and extract meaning from large volumes of unstructured text, such as product reviews, user comments, and other user-generated content. This enables the system to not only offer more accurate and contextualized recommendations, but also to generate understandable explanations in natural language for users, which increases transparency and trust in the system. The integration of NLP in EARS facilitates a more intuitive and personalized user experience, better adapting to individual needs and improving overall interaction with the system.

4.1. Federated Learning for Data Privacy

Federated Learning (FL) is a distributed machine learning paradigm that has gained significant attention due to its potential to train models across decentralized devices while preserving data privacy. This is particularly critical in domains like healthcare, finance, and IoT, where data privacy is of utmost importance. Below, we delve deeper into the key aspects of state-of-the-art Federated Learning for data privacy.

1. **Data Privacy and Security Mechanisms**:
    - **Differential Privacy (DP)**: DP is a widely adopted technique in FL to ensure that the inclusion or exclusion of a single data point does not significantly affect the output of a model, thereby protecting individual privacy. The integration of DP into FL allows for the addition of controlled noise to model updates before they are

shared, ensuring that the underlying data remains secure. This is particularly important in scenarios where sensitive information, such as medical records or financial data, is involved.

- **Secure Multi-Party Computation (SMPC)**: SMPC is another critical technique used in FL to perform computations on encrypted data, ensuring that no individual party learns anything about the data held by others. This technique is essential when FL is deployed in environments where multiple stakeholders, such as different healthcare providers, collaborate but do not fully trust each other. SMPC enables secure model training without revealing any raw data.
- **Homomorphic Encryption**: Homomorphic encryption allows computations to be performed on ciphertexts, generating an encrypted result that, when decrypted, matches the result of operations performed on the plaintext. This method is increasingly being combined with FL to enhance privacy while maintaining the functionality of the model training process.

2. **Heterogeneity Challenges**:
   - **Non-IID Data**: In FL, clients often have data that are not Independent and Identically Distributed (non-IID), meaning the data across different clients can vary significantly in terms of distribution and features. This poses a challenge for model training, as standard aggregation methods may not effectively capture the nuances of each client's data. Research has focused on personalized federated learning approaches, which aim to create models that are tailored to individual clients while still benefiting from shared learning.
   - **Robust Aggregation Techniques**: Techniques such as FedProx, which introduces a proximal term to the loss function to handle variability in data distributions, and other robust aggregation methods are being developed to address the challenges posed by non-IID data. These methods help in stabilizing the training process and improving the performance of the global model across diverse clients.
3. **Decentralized Federated Learning**:
   - **Single Point of Failure (SPoF)**: Traditional FL typically relies on a centralized server for aggregating model updates, which introduces a single point of failure and scalability issues. To overcome these challenges, Decentralized Federated Learning (DFL) is gaining traction. DFL distributes the aggregation process across multiple nodes, reducing the risk of a single point of failure and enhancing the system's robustness.
   - **Blockchain Integration**: Blockchain technology is being explored to further decentralize and secure FL systems. By using a distributed ledger, blockchain can manage model updates and aggregations across a decentralized network, enhancing trust and reducing the burden on any single server. This integration also aids in improving transparency and accountability within the FL process.

4. **Communication Efficiency**:
    - **Model Compression and Quantization**: Communication overhead is a significant challenge in FL, especially as the number of clients increases. Techniques such as model compression, where less critical parts of the model are compressed or removed, and quantization, which reduces the precision of the model parameters, are used to reduce the amount of data that needs to be communicated between clients and the server.
    - **Asynchronous Updates**: To address communication delays and inefficiencies, asynchronous update methods are employed in FL. Unlike synchronous updates, where all clients must complete their updates before aggregation, asynchronous methods allow clients to update the model independently. This reduces wait times and improves the overall efficiency of the learning process.

4.2. Domain-specific recommender system

Domain-specific recommender systems have the main goal to provide personalized suggestions within a particular domain or context, such as e-commerce, streaming services, education, healthcare, etc (Khan et all, 2017). These systems offer more relevant and specific recommendations compared to general-purpose recommender systems, and they can be enhanced depending on the user, whether he/she is a consumer, content creator or provider. These kinds of systems use algorithms to filter and suggest items (products, content, services) based on the preferences and behaviors of the user. There are three main types of recommendation techniques that can be implemented within a recommender system (Thorat et al, 2015):

- Collaborative Filtering - This method relies on the interactions between different users and items to find patterns and similarities.
- Content-Based Filtering: - This method combines attributes of items and user preferences to identify the most appropriate recommendations.
- Hybrid Approaches - This method combines collaborative and content-based techniques to improve performance and personalization of the identified recommendations.

Sometimes these kinds of systems can face the issue known as “Cold Start Problem” for which there is either a lack of historical data for new users or the system has limited user interaction data (Lika et all, 2014). To solve this, recommender systems can use popular items as initial suggestions, ask new users to rate a few items, use item attributes to find similar items, combine different recommendation methods, and use demographic or contextual information. These strategies help create better recommendations even with limited initial data.

4.2.1. Recommender Systems for End Users

End-users are typically consumers of content, products, or services (Wischenbart et al, 2021, Vorm&Miller, 2018). The system’s goal is to improve the experience by providing relevant and personalized suggestions according to the user preferences, intentions and necessities. For this the recommender system may consider data that users actively provide, such as ratings, likes, and reviews or that can be inferred from user behavior, such as browsing history, clicks, and purchases. Other relevant information might include the time of day, location, and device used which can be important in domains like streaming or retail.

4.2.2. Recommender Systems for Content Creators and Providers

Content creators and providers (such as artists, teachers, publishers, and retailers) also benefit from recommender systems, which help them optimize their content distribution (Zhan et al, 2021, Ha, 2006). The goal is to ensure that the right content is provided to the right audience, maximizing engagement and user experience. For this one of the aspects that should be considered is to provide recommendations according to the audience in which creators/providers can receive feedback about what type of content works best with specific user segments. Additionally, recommender systems help creators understand how to better engage their audiences through personalized content delivery and marketing strategies. Another strategy is based on content discovery by exposing content to users who may not have found them otherwise, potentially leading to growth in the audience. Systems can also identify trends and patterns in user engagement, and help creators produce more appealing content to users. Recommender systems can also provide creators with data-driven insights about user interaction, such as most liked content or retention rates, to help improve their work.

4.3. Explainable AI:

Introduction to XAI and a taxonomical summary of the conventional methods

Explainable AI (XAI) has become a critical area of research, driven by the necessity to ensure safety, trust, and transparency in automated decision-making systems across various applications, such as autonomous driving, medical diagnosis, and financial services. Explainable AI aims to make the decision-making processes of AI systems understandable to humans, addressing the "how" and "why" behind the actions and predictions of these systems . This field has its roots in the early days of AI with expert systems and has evolved significantly to encompass modern machine learning and neural-symbolic approaches, which integrate symbolic and subsymbolic reasoning (Confalonieri et al, 2020).

There are several techniques in XAI, each with its strengths and applications. Global explanation methods aim to provide an overall understanding of a model's behavior by creating interpretable counterparts of black-box models (Craven&Shaolin, 2017)

Local methods, such as LIME (Local Interpretable Model-agnostic Explanations, Ribeiro et al., 2016) and SHAP (Shapley Additive Explanations; Lundberg & Lee, 2017), focus on explaining individual predictions by approximating the black-box model locally around a specific instance. Introspective methods relate inputs to outputs, often through techniques like saliency maps in deep learning, highlighting which parts of the input data are most influential for the model's decisions (Hendricsk et al, 2016)

Advancing the field of XAI involves addressing several ongoing challenges. First, there is a need to enhance the causal understanding of explanations, ensuring that they not only describe correlations but also clarify causal relationships (Holzinger et al., 2019; Holzinger et al., 2020).

This includes developing methods to measure the causality of explanations effectively. Second, providing counterfactual explanations, which offer "what-if" scenarios to illustrate how changes in input features could alter the output, can make explanations more intuitive and actionable for users (Wachter, Mittelstadt, & Russell, 2018).

These explanations should be both contrastive and direct, addressing user preferences for understanding why one event occurred instead of another. The interactive and social dimensions of XAI are also crucial. Explanations should facilitate a dialogue between the AI system and the user, tailored to the user's background and level of expertise (Hilton, 1990). This involves creating selective explanations that focus on the most relevant information without overwhelming the user since humans psychologically prefer counterfactual or contrastive explanations (Miller, 2019).

Transparency is essential to help users understand the logic behind AI decisions and identify potential errors, but it must be balanced with privacy concerns, especially when explanations could reveal sensitive information about the model or training data (Harder, Bauer, & Park, 2020).

Lastly, by combining these advancements, XAI can push the boundaries of current AI technologies, ensuring they are not only powerful but also transparent, trustworthy, and aligned with ethical standards and societal needs comprehensively. This project will contribute to this goal by developing generalizable XAI solutions and gathering evidence to demonstrate the feasibility of achieving higher Technology Readiness Levels (TRLs).

Global Fed-XAI of a Horizontal, Cross-Silo Federated Learning Scheme

A summary of OpenFL and OpenFL XAI:

OpenFL is an open-source framework created by Intel® Labs in collaboration with the University of Pennsylvania. Primarily written in Python, it is available through pip, Conda, and Docker packages. OpenFL allows multiple participants (i.e., data owners) who may be located at different sites to collaboratively train a machine learning model using their own hardware while keeping their raw data private. Only the parameters of the locally trained models are exchanged via secure remote procedure calls (RPCs), with the option for mutually-authenticated transport layer security (TLS) connections. Although described as 'ML framework agnostic' (supporting

PyTorch, TensorFlow, and Keras), it is primarily designed for deep learning models. However, it is well-suited for our needs due to its modular nature, allowing for extension to support various ML models while leveraging the existing backend for the federated learning process.

OpenFL-XAI, an extension of OpenFL, inherits its core functionalities but expands upon them by incorporating support for rule-based systems (RBSs), which were not directly accommodated in the original version. This work specifically addresses FRBSs, which are a generalization of traditional RBSs. The OpenFL framework was modified to meet two non-functional requirements. First, the modifications aimed to enable federated learning (FL) of explainable artificial intelligence (XAI) models while preserving OpenFL's core functionalities and data structures. Second, the goal was to provide users with a user-friendly, practical solution for training their own (F)RBSs (Fuzzy Rule-Based Systems) in a federated manner. Given that OpenFL is primarily designed for neural networks (NNs), its FL workflow and communication layer focus on exchanging tensors for NN weights. The primary modifications involved adapting these tensors to facilitate the exchange of rules rather than NN weights.

The IBM Federated Learning framework also supports a Federated Decision Tree using the ID3 algorithm. In this approach, a single decision tree is created at a centralized server, while the clients contribute by providing counting information based on their local datasets. During each round, the server generates a list of candidate values for the input features to be split and a list of class labels to request count information from the clients. The server then calculates the information gain and performs the split accordingly. Traditional stopping criteria for decision trees are applied, such as reaching the tree's maximum depth or the absence of split values in all nodes.

A recent study suggests using a federated version of the AdaBoost algorithm. Notably, this approach imposes minimal restrictions on the clients' learning settings, allowing for a federation of models like Decision Trees (DTs) and Support Vector Machines (SVMs) without depending on gradient-based methods.

In another work, the authors explore a vertical federated learning approach for tree-based models, utilizing a privacy-preserving method based on Partially Homomorphic Encryption. In this approach, all clients participate in constructing the decision tree's structure by iteratively providing encrypted statistics to a super client, which is responsible for selecting the split points of the most relevant attributes. During the entire process, no intermediate information is revealed to any client. The authors compared the outcomes of their privacy-preserving FL methods with non-private versions. They observed slight reductions in accuracy when using privacy-based FL approaches.

Another recent work proposes a federated learning (FL) scheme to generate more explainable Takagi-Sugeno-Kang Fuzzy Rule-Based Systems (TSK-FRBSs). They used fuzzy uniform partitions with up to 5 fuzzy sets, avoiding the clustering-based approach that often creates numerous, overlapping sets. This method enhances semantic interpretability. Also, a maximum voting inference strategy instead of the traditional weighted averaging method is introduced. This FL approach is non-iterative: clients generate local TSK fuzzy rules and send them to a central server for aggregation. Conflicts between rules from overlapping attribute regions are resolved by calculating new “consequents” as the weighted average of the original rules. Experiments on benchmark datasets shows that this FL scheme outperforms locally generated models and produces results comparable to three centralized TSK-FRBS learning methods, including the classical clustering-based algorithm.

4.4. Knowledge Graph-Based Recommender Systems

Knowledge Graph-Based Recommender Systems (KG-based RS) utilize interconnected entities and relationships in a knowledge graph to enhance recommendation quality. Unlike traditional systems that rely solely on user-item interactions, KG-based RS integrates rich contextual and semantic data, offering deeper insights into user preferences and content features. They improve recommendation accuracy through reasoning and path-based algorithms, leveraging graph embeddings and multi-hop connections. This approach is highly valuable in IT for creating personalized, context-aware, and explainable recommendations in diverse domains such as e-commerce, media, and education. (Zhang et al, 2024)

KG-based RS are designed to tackle common challenges in traditional recommendation systems, such as sparse interaction data and cold-start problems, by incorporating knowledge graphs (KGs). KGs provide structured, interconnected semantic data about entities (e.g., users, items) and their relationships. This enhances recommendation accuracy and enables richer, more personalized suggestions. (Guo et al, 2022)

KG-based RS operates by embedding KGs into low-dimensional vectors, linking this semantic information to user-item interactions. These systems utilize two-stage, joint, and alternate learning methods, providing recommendations across various domains like news, music, movies, and products.

Major Benefits:

- **Enhanced Contextuality**: KGs capture deeper relationships between users and items, offering more accurate, explainable recommendations.
- **Improved Cold-Start Handling**: By leveraging external knowledge, KG-based systems can recommend items to new users more effectively.
- **Scalability and Explainability**: These systems can scale better and provide explanations by tracing recommendations to specific graph connections.

Classifications:

KG-based recommender systems are categorized based on how KGs are utilized:

- **Embedding-Based Methods**: Map KG entities into vector space to enhance recommendations.
- **Connection-Based Methods**: Utilize graph structures to identify user-item connections.
- **Propagation-Based Methods**: Use KG paths for recommendation through multi-hop connections.

KG-based RS is increasingly applied in domains such as e-commerce, media, and social platforms, where personalization and recommendation diversity are critical for enhancing user experience. These systems not only improve accuracy but also deliver explainable recommendations by providing insights into how specific recommendations are made.

4.5. Natural Language Processing (NLP)

Natural language processing, abbreviated NLP, is a field of computer science, artificial intelligence, and linguistics that studies the interactions between computers and human language, as well as the computational details of natural languages.

Natural Language Processing (NLP) is a core technology in the EARS project, used to improve the interaction between users and recommendation systems. NLP enables systems to interpret and generate natural language text, facilitating more fluid and effective communication with users. This approach is crucial for analyzing large volumes of unstructured text, such as product reviews, social media comments, and other user-generated content, improving the accuracy and relevance of recommendations.

The EARS project uses NLP to extract information from structured and unstructured data sources. This process is essential to building the knowledge structures that feed the system's recommendations.

How NLP is used in the EARS project:

- **Extracting data to create knowledge structures:** In the e-commerce and media sectors, project partners use ML and AI systems to extract data from structured and unstructured sources. This data extraction process is essential to building the knowledge structures that enable the system to make recommendations.
- **Creating multilingual search engines:** In the retail sector, the EARS project seeks to create a combined image and text search engine that is multilingual and capable of finding identical and similar solutions. This functionality relies heavily on NLP to process and understand search queries in multiple languages.

- **Developing conversational chatbots:** The EARS project also aims to create a conversational chatbot that can receive customer requests, propose recommendations, and solicit feedback to improve the recommendation process. This chatbot needs NLP to understand natural language and engage in meaningful interactions with users.

4.5.1. Extracting data to create knowledge structures

NLP is used in the EARS project to extract relevant information from both structured data (such as databases and tables) and unstructured data (such as article text, product reviews, and social media comments). This data extraction is essential for creating knowledge structures, which are organized models of information that allow the recommendation system to understand context and content more deeply.

4.5.1.1. NER (Named Entity Recognition)

Named Entity Recognition (NER) is an NLP technique that automatically identifies and classifies entities mentioned in a text into predefined categories, such as names of people, organizations, locations, dates, and more. This technique is essential for extracting structured data from unstructured textual sources, such as articles, reviews, and social media posts.

NER Process:

1. Preprocessing:

This involves cleaning up the text, removing unnecessary elements (such as HTML tags, if any), and sometimes converting the entire text to a uniform format (usually lowercase) to standardize the input data.

2. Tokenization:

The text is split into sentences and then into words or tokens.

3. Part-of-speech tagging:

Each word or symbol is tagged with a part of speech (noun, verb, adjective, etc.), based on its definition and its context in the sentence.

4. Entity detection:

This can be approached in a number of ways, including rule-based methods, machine learning models, or a combination of both. Rule-based methods use predefined patterns and dictionaries, while machine learning models, especially those based on deep learning, learn from annotated training data.

5. Entity classification:

Once potential entities are detected, they are classified into predefined categories.

6. Post-processing:

This could involve normalization, i.e. standardizing entity representations (e.g. dates) into a common format, and disambiguation, i.e. ensuring that entities are correctly identified in terms of their real-world counterparts.

Example in python: NER with LLM

The steps followed for this test are detailed below:

It consists of locating and classifying the entities found in a text into predefined categories (people, organizations, places, expressions of time and quantities, etc.).

1. **First step: load key from .env file**

```
from dotenv import load_dotenv
import os
load_dotenv(dotenv_path="../../.env")
OPENAI_KEY = os.getenv('OPENAI_KEY')
```

2. **Import classes and declare labels to identify by service**

```
from  ner_llm_service import (
    NERService, NerEntity, Span,
    ClassificationLabels, Label,
    ChatOpenAI,
    NERResult,
    parse_entities
)

ner_service = NERService(llm=ChatOpenAI(
    model_name='gpt-4o'
    , temperature=0.0
    , openai_api_key= OPENAI_KEY
    , model_kwargs={
        "seed": 42  # Add your seed value here
    }
))

NER_LABELS= ClassificationLabels(labels=[
        Label(name= 'person', description='classifies the proper names of people, including fictional characters')
    ,   Label(name= 'fac', description='classifies the proper names ofbuildings, airports, highways, bridges')
    ,   Label(name= 'org', description='organizations, companies, agencies, institutions')
    ,   Label(name= 'gpe', description='geopolitical entities like countries, cities, states')
    ,   Label(name= 'loc', description='non-gpe locations')
    ,   Label(name= 'product', description='vehicles, foods, appareal, appliances, software, toys')
    ,   Label(name= 'event', description='named sports, scientific milestones, historical events')
    ,   Label(name= 'work_of_art', description='titles of books, songs, movies')
    ,   Label(name= 'law', description='named laws, acts, or legislations')
    ,   Label(name= 'language', description='any named language')
    ,   Label(name= 'date', description='absolute or relative dates or periods')
    ,   Label(name= 'time', description='time units smaller than a day')
    ,   Label(name= 'percent', description='percentage (e.g., "twenty percent", "18%")')
    ,   Label(name= 'money', description='monetary values, including unit')
    ,   Label(name= 'quantity', description='measurements, e.g., weight or distance')
])
```

```
examples = [
    (
        "The ocean is vast and blue. It's more than 20,000 feet deep. There are many fish in it.",
        NERResult(entities=[
            NerEntity(span="The ocean", label="loc"),
            NerEntity(span="20,000 feet", label="quantity"),
        ]),
    ),
    (
        "Fiona traveled far from France to Spain.",
        NERResult(entities=[
            NerEntity(span="Fiona", label="person"),
            NerEntity(span="France", label="gpe"),
            NerEntity(span="Spain", label="gpe"),
        ]),
    )
]
```

3. Next is the text used for test

```
text = '''Angel Reese isn't just playing basketball; she's redefining it. The Chicago Sky's star rookie is tearing through the WNBA record books with a force r

The third quarter saw Reese pull off a put-back layup, putting her at 11 points and 11 rebounds for the game. She didn't stop there, though. By the final buzze

Drafted seventh overall in 2024, Reese has been nothing short of spectacular for the Sky. She's not just consistent; she's relentlessly dominant. In 14 of her

Angel Reese's relentless commitment to team success

But here's the kicker: Reese isn't even focused on the stats. After dropping a career-high 27 points and snagging 10 rebounds in the Sky's 88-84 victory over t

"I just go out there and do my job. My job is to rebound, so I'm going to go out there and do my job and rebound. I know that's what my teammates need me to do

Commitment. It's a word thrown around in sports, but for Reese, it's the bedrock of her game. Her relentless pursuit of excellence is not just about personal g

Breaking records in any sport is an extraordinary feat, but doing it as a rookie? That's a whole different ballgame. Reese is carving her path with a ferocity
'''
```

4. Invoke service and parse entities

```
result_async = await ner_service.ainvoke(text, NER_LABELS, examples=examples)
entities = parse_entities(text, result_async.entities)
```

5. Plot detected entities

```
from utils import plot_ner
plot_ner(text, entities)
```

Angel Reese person isn't just playing basketball; she's redefining it. The Chicago Sky org 's star rookie is tearing through the WNBA org record books with a force reminiscent of a young Candace Parker person . In the Sky org 's recent 84-71 loss to the Seattle Storm org , Reese person clinched her 13th consecutive double-double, surpassing Parker person 's record that spanned the 2009 and 2010 seasons. And let's be clear, she's just getting started.

The third quarter saw Reese person pull off a put-back layup, putting her at 11 points and 11 rebounds for the game. She didn't stop there, though. By the final buzzer, Reese person had racked up 17 points and 14 rebounds, cementing her place in WNBA org history.

Drafted seventh overall in 2024 date , Reese person has been nothing short of spectacular for the Sky org . She's not just consistent; she's relentlessly dominant. In 14 of her first 20 games, she has posted double-doubles, leading the league in rebounds with an impressive 11.7 boards per game. Her scoring prowess isn't far behind, with an average of 13.9 points per game, second only to fellow rookie sensation Caitlin Clark person .

How NER is applied in the EARS project:

- Identifying key information: NER enables the recommendation system to identify relevant entities within a text. For example, in the context of e-commerce, NER can extract product names, brands, and specific features mentioned in user reviews.
- Building user and product profiles: By identifying key entities, NER helps build detailed user and product profiles, which are then used to improve the personalization of recommendations.
- Improving search and filtering: NER improves search capabilities by allowing the system to recognize and classify entities in search queries, facilitating accurate information retrieval.

4.5.1.2. Sentiment Analysis

Sentiment analysis is another fundamental NLP technique that is used to determine the attitude or emotion expressed in a text, classifying it into categories such as positive, negative or neutral. This technique is particularly useful in analyzing customer opinions and evaluating comments on social media.

Sentiment analysis is the process of using computational and natural language processing techniques to analyze text or audio data and categorize it based on the identified context as positive, neutral, or negative.

Sentiment analysis, also known as opinion mining, uses natural language processing techniques to identify the emotional tone behind textual data. The goal of sentiment analysis goes beyond simply classifying text data based on its motive. The task is to identify the hidden context of text data and use it to draw meaningful inferences that will help, for example, companies understand their customers' perception of their products or services.

Types of sentiment Analysis

Sentiment analysis has numerous options that can be used to identify a wide variety of sentiments.

1. Emotion Detection

Emotion detection, also known as lexical analysis, is a type of sentiment analysis that focuses on unraveling the underlying sentiment or emotion behind textual data. Based on the identified context of a text, this type of analysis detects different emotions: frustration, happiness, and sadness.

2. Intent Detection

Intent detection is a form of sentiment analysis that analyzes text data to identify the underlying intent of a text. Regardless of the medium, every communication has a message or information that it intends to convey. What intent detection does is uncover such intent behind a given text.

When identified, these intentions can provide valuable insights. Businesses can use intent detection systems to identify the intent of a text message received from a customer: they can identify whether the customer is requesting technical support or is about to stop using the service.

3. Detailed Sentiment Analysis

Fine-grained sentiment analysis classifies textual data beyond positive, negative, and neutral; offers a more detailed and specific analysis of a text by rating its positivity or negativity in moderate and extreme contexts: extremely negative, very negative, neutral, very positive, and extremely positive.

4. Multilingual Sentiment Analysis

As the name suggests, multilingual sentiment analysis involves analyzing multiple languages to identify and extract the emotional tone. This analysis is applied to analyzing social media streams or customer feedback in various dialects.

5. Aspect-based Sentiment Analysis

Instead of analyzing the textual data in its entirety, aspect-based analysis offers detailed analysis based on the specific aspects. It simply identifies and analyzes the sentiment of an aspect in the textual data and classifies it as positive or negative.

Sentiment Analysis Proccess:

1. Text preprocessing

The first step in sentiment analysis is to preprocess the data. Preprocessing simply means cleaning and making the data suitable for sentiment analysis.

Preprocessing tasks in sentiment analysis are lowercase, tokenization, punctuation removal, stopword removal, stemming, and lemmatization.

2. Feature Extraction

Feature extraction is the process of converting data into a feature or numerical format that a sentiment analysis model can decode. There are different types of techniques used to perform feature extraction.

3. Sentiment Analysis Model

Once all the relevant feature extraction and text processing tasks have been performed, the next crucial step is to develop the sentiment analysis model. The sentiment analysis model is a model specifically built to analyze and identify sentiments in text or audio data using natural language processing (NLP) techniques.

Other efficient algorithms are:

- Naive Bayes Classifier
- Support Vector Machines (SVM)
- Recurrent Neural Network (RNN)

- Convolutional Neural Network (CNN)
- Transformer-based models (e.g. BERT and GPT)

4. Model Evaluation

Once the sentiment analysis model is built, an evaluation is needed to ensure that the model is performing as expected. First, we need to split the dataset into two: the training dataset and the test dataset. Then, we will use the training dataset to train the model, and then we will use the test dataset to evaluate its efficiency based on the following evaluation metrics:

- **Accuracy**: The model's accuracy in prediction. That is, how many predictions are accurate out of the total predictions.
- **Precision**: The number of accurate predictions that are actually correct.
- **Recall:** The number of actual positives that are correctly predicted.
- **F1 Score:** The average values of precision and recall.

Example in python: SENTIMENT ANALYSIS with LLM

Below is a detailed description of the example performed with the executed code and the steps to understand how sentiment analysis works.

1. **Load key from .env file**

```
from dotenv import load_dotenv
import os
load_dotenv(dotenv_path=".env")
OPENAI_KEY = os.getenv('OPENAI_KEY')
```

2. **Declare class to be used for sentiment detection using LLM as classificator**

```
from typing import Optional, List
from langchain_core.pydantic_v1 import BaseModel, Field

class SentimentEntity(BaseModel):
    """Information about a Sentiment Recognition."""
    explanation: str = Field(default=None, description="Explanation of why a text is classified with this label and its intensity ")
    label: str = Field(default=None, description="Clasification label of sentiment ")
    intensity: float= Field(default=None, ge=0, le=100, description="Intensity of sentiment from 0 to 100")

class SentimentResult(BaseModel):
    """Information about a Sentiment Recognition detected."""
    sentiments: List[SentimentEntity] = Field(default=None, description="sentiments detected in the text ")

class Label(BaseModel):
    name: str = Field(default=None, description="Name of class ")
    description: str = Field(default=None, description="Description of class ")

SENTIMENT_NERS = [
    Label(name="Vigilance",description='Keeping careful watch for possible danger or difficulties. ')
    , Label(name="Ecstasy",description='Overwhelming feeling of great happiness or joyful excitement. ')
    , Label(name="Admiration",description='Respect and warm approval. ')
    , Label(name="Terror",description='Respect and warm approval. ')
    , Label(name="Amazement",description='Feeling of great surprise or wonder. ')
    , Label(name="Grief",description='An instance or cause of intense sorrow. ')
    , Label(name="Loathing",description='Feel intense dislike or disgust for. ')
    , Label(name="Rage",description='Violent uncontrollable anger. ')
]
```

3. Next is the text used for test

```
text = '''
Angel Reese isn't just playing basketball; she's redefining it. The Chicago Sky's star rookie is tearing through the WNBA record books with a force reminiscent

The third quarter saw Reese pull off a put-back layup, putting her at 11 points and 11 rebounds for the game. She didn't stop there, though. By the final buzze

Drafted seventh overall in 2024, Reese has been nothing short of spectacular for the Sky. She's not just consistent; she's relentlessly dominant. In 14 of her

Angel Reese's relentless commitment to team success

But here's the kicker: Reese isn't even focused on the stats. After dropping a career-high 27 points and snagging 10 rebounds in the Sky's 88-84 victory over t

"I just go out there and do my job. My job is to rebound, so I'm going to go out there and do my job and rebound. I know that's what my teammates need me to dc

Commitment. It's a word thrown around in sports, but for Reese, it's the bedrock of her game. Her relentless pursuit of excellence is not just about personal g

Breaking records in any sport is an extraordinary feat, but doing it as a rookie? That's a whole different ballgame. Reese is carving her path with a ferocity
'''
```

4. This is the prompt used by the LLM as system message

```python
from langchain_core.prompts import ChatPromptTemplate, MessagesPlaceholder

# Define a custom prompt to provide instructions and any additional context.
# 1) You can add examples into the prompt template to improve extraction quality
# 2) Introduce additional parameters to take context into account (e.g., include metadata
#    about the document from which the text was extracted.)

labels = [f'{label.name} -> {label.description}' for label in SENTIMENT_NERS ]

SYSTEM_PROMPT = '''
You are an expert in Natural Language Processing. Your task is to identify sentiments in a given text.
The possible sentiment types are exclusively:
{sentiment_labels}
'''

prompt = ChatPromptTemplate.from_messages(
    [
        (
            "system", SYSTEM_PROMPT,
        ),
        # Please see the how-to about improving performance with
        # reference examples.
        # MessagesPlaceholder('examples'),
        ("human", "{text}"),
    ]
)
```

5. Declare Chain and test with the text

```
from langchain_openai import  ChatOpenAI
from langchain_core.output_parsers import PydanticOutputParser
from langchain_core.runnables import RunnableLambda, RunnableParallel
from langchain.output_parsers import RetryOutputParser

llm = ChatOpenAI(
    model_name="gpt-4o"
    , temperature=0
    , openai_api_key= OPENAI_KEY
    , model_kwargs={
        "seed": 42  # Add your seed value here
    }
)

parser = PydanticOutputParser(pydantic_object=SentimentResult)
retry_parser = RetryOutputParser.from_llm(parser=parser, llm=llm, max_retries=3)
# https://github.com/langchain-ai/langchain/issues/19145
def parse_with_prompt(args):
    completion = args['completion']

    if (type(completion) is SentimentResult):
        args = args.copy()
        del args['completion']
        completion = completion.json(ensure_ascii=False)
        args['completion'] = completion

    return retry_parser.parse_with_prompt(**args)

chain_sentiment = prompt | llm.with_structured_output(
    schema=SentimentResult,
    method="function_calling",
    include_raw=False
)

main_chain = RunnableParallel(
    completion=chain_sentiment,
    prompt_value=prompt
) | RunnableLambda(parse_with_prompt)

results: SentimentResult
results = main_chain.invoke(
    {
        "text": text,
        "sentiment_labels" : labels
    }
)
```

6. Print the result

```
for sent in results.sentiments:
    print(f'Label: {sent.label}')
    print(f'Intensity: {sent.intensity}')
    print(f'Explanation:  \n{sent.explanation}')
    print('-'*80)
```

```
Label: Admiration
Intensity: 90.0
Explanation:
The text highlights Angel Reese's exceptional performance, commitment, and impact on the WNBA, evoking a strong sense of respect and warm approval.
--------------------------------------------------------------------------------
Label: Amazement
Intensity: 85.0
Explanation:
The text describes Reese's record-breaking achievements and her dominant presence on the court, which elicits a feeling of great surprise and wonder.
--------------------------------------------------------------------------------
Label: Ecstasy
Intensity: 70.0
Explanation:
The text conveys a sense of overwhelming happiness and excitement about Reese's accomplishments and potential in the WNBA.
--------------------------------------------------------------------------------
```

How Sentiment Analysis is applied in the EARS project:

- Evaluating user perception: In the EARS project, sentiment analysis is used to evaluate user perception about specific products or services. This allows the system to better understand users' preferences and dislikes.

- Adjusting recommendations: By knowing users' sentiments toward certain products or features, the system can dynamically adjust recommendations to better align with user emotions and expectations.
- Monitoring brand reputation: This technique is also applied to monitor brand reputation in real time, providing project partners with valuable insights on how to improve their products or services.

4.5.1.3. Topic Classification

Topic classification is an NLP technique that automatically groups and categorizes text into different topics or categories. This technique is essential for organizing large volumes of unstructured data and extracting meaningful insights for recommendation systems.

Topic classification is a Natural Language Processing (NLP) technique used to automatically categorize text into different predefined topics or categories. This technique is essential for organizing and structuring large volumes of unstructured data, allowing recommender systems and other NLP applications to better understand the content and extract relevant information.

Topic classification involves assigning topic labels to documents, text fragments, or phrases based on the content and context of the text. The purpose of this technique is to group similar texts into categories, facilitating information retrieval, data analysis, and the generation of recommendations based on user interests or needs.

Topic Classification Proccess
The topic classification process typically follows several key steps:

1. Text Preparation (Preprocessing):

Tokenization: Text is broken down into smaller units, such as words or phrases, called tokens.
Lemmatization and Stemming: Tokens are transformed into their root or lemma forms to normalize the words and reduce the dimensionality of the data (e.g., “running” and “run” are reduced to “run”).
Stopword Removal: Common words that do not contribute significant meaning to the topic (such as “and,” “the,” “a”) are removed from the text to reduce noise.

2. Text Vectorization:

Bag of Words (BoW): A simplified representation of text where the frequency of words in a document is counted. Although easy to implement, this technique does not capture semantics or word order.
TF-IDF (Term Frequency-Inverse Document Frequency): Improves the Bag of Words by weighting the frequency of words in a document compared to their inverse frequency in the full dataset. This highlights words that are important in one document but not common in others.

Word Embeddings: Advanced techniques such as Word2Vec, GloVe or BERT create dense, vector representations of words that capture both syntax and semantics, allowing for more accurate and context-rich classification.

3. Model Training:

A classification model is trained using a pre-labeled dataset with known topics. Common algorithms used for this purpose include Naive Bayes, Support Vector Machines (SVM), Neural Networks and Transformer-based models such as BERT.
The model learns to identify patterns and features that are indicative of each topic, based on the feature vectors generated during vectorization.

4. Text Classification:

Once trained, the model can classify new texts into one of the predefined categories. It evaluates the input text using the same vectorization techniques and applies the model to predict the most likely topic.

5. Evaluation and Improvement:

The accuracy of the model is evaluated using metrics such as precision, recall, and F1-score. Depending on the results, the model can be fine-tuned or refined, either by adjusting hyperparameters or by using different vectorization techniques.

Example in python: TOPIC CLASSIFICATION with LLM

Below is a detailed description of the example performed with the executed code and the steps to understand how topic classification works.

1. Load key from .env file

```
from dotenv import load_dotenv
import os
load_dotenv(dotenv_path="../../.env")
OPENAI_KEY = os.getenv('OPENAI_KEY')
```

2. Declare class to be used for topic classification

```
from typing import Optional, List
from langchain_core.pydantic_v1 import BaseModel, Field


class TopicEntity(BaseModel):
    """Information about a Topic detection."""
    label: str = Field(default=None, description="Clasification label of sentiment ")
    explanation: str = Field(default=None, description="Explanation of why a text is classified with this label ")

class TopicResult(BaseModel):
    """Information about a topics detected."""
    topics: List[TopicEntity] = Field(default=None, description="topics detected in the text ")


TOPIC_LABELS = [
    {"name": "SPORT", "description": "text talks about sport/s"}
    , {"name": "BUSSINES", "description": "text talks about bussine/s"}
    , {"name": "PEOPLE", "description": "text talks about a person or people"}
    , {"name": "LOCATIONS", "description": "text talks about location/s"}
    , {"name": "RELIGION", "description": "text talks about religion/s"}
]
```

3. Next is the text used for test

```
text = '''
Angel Reese isn't just playing basketball; she's redefining it. The Chicago Sky's star rookie is tearing through the WNBA record books with a force reminiscent of a young Candace Parker. In the Sky's recent 84-7

The third quarter saw Reese pull off a put-back layup, putting her at 11 points and 11 rebounds for the game. She didn't stop there, though. By the final buzzer, Reese had racked up 17 points and 14 rebounds, ce

Drafted seventh overall in 2024, Reese has been nothing short of spectacular for the Sky. She's not just consistent; she's relentlessly dominant. In 14 of her first 20 games, she has posted double-doubles, leadi

Angel Reese's relentless commitment to team success

But here's the kicker: Reese isn't even focused on the stats. After dropping a career-high 27 points and snagging 10 rebounds in the Sky's 88-84 victory over the Storm on Friday, she downplayed her achievement.

"I just go out there and do my job. My job is to rebound, so I'm going to go out there and do my job and rebound. I know that's what my teammates need me to do, and I've committed to that," Reese said.

Commitment. It's a word thrown around in sports, but for Reese, it's the bedrock of her game. Her relentless pursuit of excellence is not just about personal glory but about lifting her team every single night.

Breaking records in any sport is an extraordinary feat, but doing it as a rookie? That's a whole different ballgame. Reese is carving her path with a ferocity that leaves defenders bewildered and fans in awe. An
'''
```

4. This is the prompt used by the LLM as system message

```
from langchain_core.prompts import ChatPromptTemplate, MessagesPlaceholder

# Define a custom prompt to provide instructions and any additional context.
# 1) You can add examples into the prompt template to improve extraction quality
# 2) Introduce additional parameters to take context into account (e.g., include metadata
#    about the document from which the text was extracted.)


SYSTEM_PROMPT = '''
You are an expert in Natural Language Processing. Your task is to extract topics in a given text.
The possible main topics are exclusively:
{topics}
'''

prompt = ChatPromptTemplate.from_messages(
    [
        (
            "system", SYSTEM_PROMPT,
        ),
        # Please see the how-to about improving performance with
        # reference examples.
        # MessagesPlaceholder('examples'),
        ("human", "{text}"),
    ]
)
```

5. Declare Chain and test with the text & Print the result

```
from langchain_openai import  ChatOpenAI
from langchain_core.output_parsers import PydanticOutputParser
from langchain_core.runnables import RunnableLambda, RunnableParallel
from langchain.output_parsers import RetryOutputParser

llm = ChatOpenAI(
    model_name="gpt-4o"
    , temperature=0
    , openai_api_key= OPENAI_KEY
    , model_kwargs={
        "seed": 42  # Add your seed value here
    }
)

parser = PydanticOutputParser(pydantic_object=TopicResult)
retry_parser = RetryOutputParser.from_llm(parser=parser, llm=llm, max_retries=3)
# https://github.com/langchain-ai/langchain/issues/19145
def parse_with_prompt(args):
    completion = args['completion']

    if (type(completion) is TopicResult):
        args = args.copy()
        del args['completion']
        completion = completion.json(ensure_ascii=False)
        args['completion'] = completion

    return retry_parser.parse_with_prompt(**args)

chain_topic = prompt | llm.with_structured_output(
    schema=TopicResult,
    method="function_calling",
    include_raw=False
)

main_chain = RunnableParallel(
    completion=chain_topic,
    prompt_value=prompt
) | RunnableLambda(parse_with_prompt)


topics = topics= [f'{topic["name"]} -> {topic["description"]}' for topic in TOPIC_LABELS ]
results: TopicResult
results = main_chain.invoke(
    {
        "text": text,
        "topics": topics
    }
)
```

```
[(topic.label, topic.explanation) for topic in results.topics]
```

```
[('SPORT',
  "The text discusses Angel Reese's performance and achievements in the WNBA, including breaking records and her commitment to her team."),
 ('PEOPLE',
  'The text focuses on Angel Reese, a person, and her impact on the sport of basketball.')]
```

How Topic Classification is applied in the EARS project:

- Content organization: Topic classification allows the EARS project to efficiently organize content from various sources, such as news articles, blogs, and forums, into relevant categories. This makes it easier to search and retrieve specific, relevant information for recommendations.
- Identifying user interests: By categorizing content based on topics, the system can identify patterns in user interests and tailor recommendations accordingly. For example, if a user shows interest in technology-related topics, the system will prioritize similar content in its recommendations.
- Detecting emerging trends: Topic classification also allows for detecting emerging trends in user preferences, which is crucial for anticipating changes in user behavior and adjusting recommendation strategies accordingly.

4.5.2. Creating multilingual search engines

The EARS project aims to develop a search engine that combines text and images and is capable of handling multiple languages. This search engine requires NLP to understand and process search queries in multiple languages, identify key elements, and find similar or identical solutions in the database.

4.5.2.1 Multilingual NLP Capabilities

Natural Language Processing (NLP) is essential for handling multilingual search queries in the EARS project. The ability to accurately and efficiently translate, interpret, and process search queries, regardless of the input language, is critical to delivering high-quality, relevant results to users. Here's how NLP is used in this context:

1. Machine Translation of Query

The first step in handling queries in different languages is machine translation. Using advanced NLP models, such as those based on Transformers (e.g., BERT, GPT-3, or T5), the system can translate queries from one language to another while preserving the original meaning and context.

- Neural Translation Models (NMT): Neural translation models have significantly improved the quality of machine translation. Unlike traditional statistical approaches, NMT models use deep neural networks to learn contextual representations of words and phrases, allowing them to capture nuances and ambiguities in language.
- Multilingual Translation: Models such as M2M-100 or mBART have been trained on multiple language pairs, allowing direct translation between multiple languages without needing to rely on English as an intermediary. This reduces information loss and improves the accuracy of translations.

2. Interpretation of Meaning and Context

After translation, the next challenge is to correctly interpret the meaning and context of the query. NLP uses several techniques to understand human language in its complexity:

- Multilingual Embeddings: Tools like LASER or MUSE create vector representations (embeddings) of words that are consistent across different languages. This means that words with similar meanings are placed close to each other in a common vector space, regardless of language, making it easier to interpret context and semantics across multiple languages.
- Syntactic and Semantic Analysis: NLP also applies syntactic and semantic analysis to understand the grammatical structure of the query and its underlying meaning. This is especially useful for disambiguating polysemous words (those with multiple meanings) and understanding complex phrases or idioms that may vary across languages.

3. Information Processing and Retrieval

Once the query has been interpreted, the system must process and retrieve relevant information. This is where NLP plays a crucial role in ensuring that results are accurate and relevant:

- Multilingual Indexing: Content in the search index is preprocessed and indexed using NLP techniques that support multiple languages. This includes text normalization (e.g. stemming and lemmatization) and the application of multilingual embeddings, which ensures that content in different languages can be retrieved efficiently.
- Expanded Query and Reformulation: To improve the relevance of results, the search system can expand the original query with additional terms that are synonymous or semantically related in different languages. This is achieved using attention neural networks that can identify key terms and generate alternatives that improve information retrieval.

4. Generating Relevant Results and Multilingual Accuracy

Finally, NLP ensures that the results returned to the user are accurate and relevant, regardless of the input language of the query:

- Evaluation and Relevance: Machine learning algorithms are trained to rank search results not only based on keyword matching, but also considering semantic and contextual factors derived from NLP models. This means that the most relevant results are prioritized, even if the exact word match is not high.
- Multilingual Personalization: Using NLP techniques, the system can tailor results to the user's linguistic and cultural preferences, providing a more personalized and meaningful search experience.

4.5.2.2 Text and Image Search Integration:

In the EARS project, the search engine was designed to improve the user experience by integrating Natural Language Processing (NLP) with image-based search capabilities. This integration allows for accurate interpretation of textual queries and combining this information with image analysis, delivering more comprehensive and relevant results that significantly improve system functionality and user satisfaction.

1. Interpreting Textual Queries with NLP

The first step in this process is the interpretation of textual queries entered by users. Here, NLP plays a crucial role by breaking down the query into its meaningful components and understanding its intent.

- Syntactic and Semantic Analysis: NLP uses syntactic analysis techniques to identify the grammatical structure of the query and semantic analysis to understand the meaning of words in the context of the sentence. This is crucial to understanding the user's intent, especially in complex or ambiguous queries.
- Named Entity Recognition (NER): The search engine employs NER techniques to identify key entities within the query, such as product names, brands, locations, or specific features. This helps pinpoint the relevant terms that should be considered in both textual and visual search.
- Query Reformulation and Expansion: Through NLP, the search engine can expand and reformulate the original query to include synonyms, related terms, or linguistic variations that might be relevant. This increases the likelihood of retrieving results that, while not exactly matching the query words, are semantically relevant.

2. Combining Textual and Image-Based Search

Once the textual query has been interpreted and reformulated, the search engine proceeds to combine these textual results with the image-based search. This combination is done in several ways:

- Visual Analysis Based on Semantic Features: The system uses computer vision algorithms that apply convolutional neural networks (CNN) to analyze images and extract meaningful visual features. These features align with the entities identified through NLP, allowing visual searches to be complemented with textual context.
- Multimodal Embeddings: The search engine uses multimodal embeddings, which are vector representations that combine textual and visual information in a common space. These embeddings allow the search engine to understand complex relationships between text and images, making it easier to retrieve relevant images that match the interpreted textual query.
- Joint Semantic Search: Through the use of deep learning techniques, the search engine performs a semantic search that considers both textual and visual content. This allows

images to be retrieved not only for literal matches to the text, but also for their semantic relevance in relation to the interpreted query.

3. Improved User Experience and System Functionality

Combining NLP with image-based search offers multiple benefits that improve user experience and system functionality:

- More Relevant and Contextualized Results: By interpreting both text and image together, the search engine can deliver results that are more accurate and aligned with the user's intent. This is especially useful in cases where visual and textual information needs to be considered together for an optimal recommendation.
- Natural and Fluid Interaction: Users can use both text and images for their searches more naturally and without worrying about formatting limitations. For example, a user could search for a specific product using a textual description and an image, and the search engine would provide results that match both the image and the description.
- Adaptability and Flexibility: The integration of NLP and image analysis allows the search engine to be adaptable to different types of queries and use cases, providing a more flexible and robust platform that can meet a variety of user needs.

Example in python: Combining NLP and Image-Based Search

Suppose we have an application within the EARS project that allows users to search for products using both text and images. The goal is to interpret the user's query, which may include textual descriptions and visual examples, to find relevant products that match both the text and image characteristics.

1. **Preprocessing Text Queries with NLP:**

```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

# Download necessary NLTK resources
nltk.download('punkt')
nltk.download('stopwords')

def preprocess_text(query):
    # Tokenize the text
    words = word_tokenize(query)

    # Convert to lowercase and remove stopwords
    stop_words = set(stopwords.words('english'))
    filtered_words = [word.lower() for word in words if word.isalpha() and word.lower(

    return filtered_words

# Example text query
query_text = "Find a pair of stylish red sneakers for running."
processed_text = preprocess_text(query_text)
print("Processed keywords:", processed_text)
```

2. Extracting Visual Features from the Image:

```
from keras.applications.vgg16 import VGG16, preprocess_input
from keras.preprocessing.image import load_img, img_to_array
import numpy as np

# Load the pre-trained VGG16 model without the top classification layer
model = VGG16(weights='imagenet', include_top=False)

def extract_image_features(image_path):
    # Load and preprocess the image
    image = load_img(image_path, target_size=(224, 224))
    image_array = img_to_array(image)
    image_array = np.expand_dims(image_array, axis=0)
    image_array = preprocess_input(image_array)

    # Extract features using VGG16
    features = model.predict(image_array)
    return features

# Example image path
image_path = 'path/to/example_image.jpg'
image_features = extract_image_features(image_path)
print("Extracted image features shape:", image_features.shape)
```

3. Combining Text and Image Search Results:

```
# Calculate cosine similarity for text features
def text_similarity(query, product_features):
    query_vector = np.mean([nltk.FreqDist(query).get(word, 0) for word in product_features])
    return query_vector

# Calculate visual similarity
def visual_similarity(query_image_features, product_image_features):
    similarities = cosine_similarity(query_image_features.reshape(1, -1), product_image_features.reshape(product_image_features.shape[0], -1))
    return similarities

# Get similarities
text_similarities = [text_similarity(processed_text, features) for features in product_text_features]
visual_similarities = visual_similarity(image_features, product_image_features)

# Combine scores (simple average for this example)
combined_scores = [(0.5 * text_sim + 0.5 * visual_sim) for text_sim, visual_sim in zip(text_similarities, visual_similarities.flatten())]
best_match_index = np.argmax(combined_scores)

print(f"Recommended product (index {best_match_index}): text features {product_text_features[best_match_index]}, combined similarity {combined_scores[best_match_index]}")
```

Explanation of the Code

1. Text Preprocessing: We use NLTK to tokenize and remove stopwords from the user's query. This helps identify the keywords describing what the user is searching for.
2. Image Feature Extraction: We use VGG16, a pre-trained Convolutional Neural Network model, to extract features from the image provided by the user. These features represent the visual content of the image in a format that can be compared with others.
3. Combining Results: We calculate cosine similarity for both the text keywords and image features, then combine these scores to find the most relevant product in a simulated product database.

4.5.3. Developing conversational chatbots

The EARS project also incorporates the development of an advanced conversational chatbot that can interact with users, understand their needs, provide recommendations and collect feedback to improve the recommendation system.

A chatbot is a software application used to conduct an on-line chat conversation via text or text-to-speech, in lieu of providing direct contact with a live human agent.Designed to convincingly simulate the way a human would behave as a conversational partner, chatbot systems typically require continuous tuning and testing, and many in production remain unable to adequately converse or pass the industry standard Turing test.The term ChatterBot was originally coined by Michael Mauldin (creator of the first Verbot) in 1994 to describe these conversational programs.

Chatbots are typically used in dialog systems for various purposes including customer service, request routing, or for information gathering. While some chatbot applications use extensive word-classification processes, Natural Language processors, and sophisticated AI, others simply scan for general keywords and generate responses using common phrases obtained from an associated library or database.

Today, most chatbots are accessed on-line via website popups, or through virtual assistants such as Google Assistant, Amazon Alexa, or messaging apps such as Facebook Messenger or WeChat.Chatbots are typically classified into usage categories that include: commerce (e-commerce via chat), education, entertainment, finance, health, news, and productivity.

Chatbot competitions focus on the Turing test or more specific goals. Two such annual contests are the Loebner Prize and The Chatterbox Challenge (the latter has been offline since 2015, however, materials can still be found from web archives). DBpedia created a chatbot during the GSoC of 2017 and can communicate through Facebook Messenger. DBpedia started in 2007 and allows users to extract structured content from the Wikipedia dataset, along with many other datasets. DBpedia is currently one of the biggest representatives of Linked Open Data (LOD).

Chatbots are also appearing in the healthcare industry. A study suggested that physicians in the United States believed that chatbots would be most beneficial for scheduling doctor appointments, locating health clinics, or providing medication information.

Certain patient groups are still reluctant to use chatbots. A mixed-methods study showed that people are still hesitant to use chatbots for their healthcare due to poor understanding of the technological complexity, the lack of empathy and concerns about cyber-security.The analysis showed that while 6% had heard of a health chatbot and 3% had experience of using it, 67%

perceived themselves as likely to use one within 12 months. The majority of participants would use a health chatbot for seeking general health information (78%), booking a medical appointment (78%) and looking for local health services (80%). However, a health chatbot was perceived as less suitable for seeking results of medical tests and seeking specialist advice such as sexual health. The analysis of attitudinal variables showed that most participants reported their preference for discussing their health with doctors (73%) and having access to reliable and accurate health information (93%). While 80% were curious about new technologies that could improve their health, 66% reported only seeking a doctor when experiencing a health problem and 65% thought that a chatbot was a good idea. Interestingly, 30% reported dislike about talking to computers, 41% felt it would be strange to discuss health matters with a chatbot and about half were unsure if they could trust the advice given by a chatbot. Therefore, perceived trustworthiness, individual attitudes towards bots and dislike for talking to computers are the main barriers to health chatbots.

4.5.3.1 Generating personalized responses and recommendations

In the EARS project, the chatbot is used to interact with users, understand their requests, and provide meaningful and personalized responses. Natural Language Processing (NLP) is central to this process, as it enables the chatbot to correctly interpret human language and generate appropriate and contextually relevant responses, thereby improving the user experience. Below we explain how NLP techniques are used to achieve these goals.

Chatbot Process

1. Understanding User Requests with NLP

The first step in chatbot interaction is to understand user requests. This involves interpreting the meaning and intent behind the words and phrases used in natural language. Several NLP techniques are applied to achieve this:

- **Tokenization and Normalization:** NLP breaks down user text into individual words (tokens) and converts them into a standard form (normalization), removing common words (stopwords) and applying techniques such as lemmatization or stemming to reduce words to their root forms. This helps to identify keywords and relevant content from the user request.
- **Syntactic and Semantic Analysis:** Syntactic analysis (parsing) identifies the grammatical structure of the sentence, while semantic analysis understands the meaning of words and the relationships between them. This allows the chatbot to better understand the context of the user request and disambiguate complex or ambiguous sentences.
- **Intent Recognition:** Chatbots use NLP models to identify the underlying intent of a user request, such as “search for a product,” “check the status of an order,” or “ask for help.” This is done by training classification models that recognize patterns in the user’s text that are associated with different intents.

- **Named Entity Recognition (NER):** The chatbot also uses NER techniques to extract specific entities from the user's request, such as product names, locations, or dates. This information is crucial to providing accurate and relevant responses.

2. Generating Meaningful and Personalized Responses

Once the chatbot has understood the user's request, the next step is to generate meaningful and personalized responses. This is achieved using several NLP techniques:

- **Natural Language Generation (NLG):** NLG is the part of NLP that focuses on creating understandable and coherent text based on data and machine learning models. The chatbot uses NLG to construct responses that are grammatically correct and make sense in the context of the conversation.
- **Context-Based Personalization:** The chatbot uses prior information about the user, such as their interaction history, preferences, and behavior, to personalize responses. For example, if a user has previously shown interest in technology products, the chatbot can prioritize recommendations related to that interest.
- **Reinforcement Learning Techniques:** Some chatbots use reinforcement learning techniques to continually improve their responses. The chatbot receives feedback from users about the quality of the responses and adjusts its models to improve relevance and accuracy.
- **Handling Multi-turn Conversations:** Advanced chatbots handle multi-turn conversations, where the context of the previous interaction is used to understand and respond appropriately to the user's current requests. For example, if a user first asks "Do you have any sneakers?" and then follows up with "And in red?", the chatbot uses context to understand that the second question refers to the sneakers mentioned above.

3. Improved User Experience

Using NLP to understand and respond to user requests significantly improves the user experience in several ways:

- **More Natural and Efficient Interaction:** By using NLP, the chatbot can interpret users' natural language more accurately, allowing for smoother and less frustrating communication. Users can freely express themselves, knowing that the chatbot will understand their requests and respond appropriately.
- **Relevant and Contextual Responses:** Thanks to personalization and context management, the chatbot can provide responses that are highly relevant and specific to each individual user, increasing user satisfaction and perceived quality of service.
- **Continuous Learning and Adaptation:** By integrating machine learning techniques, chatbots can continually improve their responsiveness capabilities, adapting to new trends and patterns in user behavior, ensuring that interactions remain effective and satisfying over time.

In summary, the use of NLP in the EARS project chatbot allows not only to understand user requests accurately, but also to generate personalized and meaningful responses that improve the user experience. By combining advanced natural language understanding and generation techniques, the chatbot can offer a more natural, relevant and efficient interaction, continuously adapting to users' needs and preferences.

5. Conclusion

The "Report on State of the Art (SoTA) & Innovation" deliverable for the EARS project provides a comprehensive analysis of current advancements in recommendation systems, particularly focusing on addressing key challenges in privacy, accuracy, and transparency. It reviews the existing technologies and highlights the innovative solutions proposed within the project.

The EARS project aims to revolutionize the recommendation system landscape through several key innovations:

1. Federated Learning: A key aspect of the project is leveraging federated learning to improve the accuracy of recommendations while maintaining data privacy. By distributing model training across different devices and platforms without sharing raw data, EARS enhances data protection while addressing cold start and data sparsity issues, common in traditional recommendation engines.
2. Explainable AI (XAI): The project also integrates explainable AI techniques to foster user trust. These models provide transparent and comprehensible recommendations, helping users understand the rationale behind the suggestions. This is crucial for building trust, especially in sensitive sectors like healthcare and finance.
3. Hybrid Recommendation Engines: EARS proposes a hybrid approach combining collaborative filtering and content-based methods to overcome limitations like cold start and sparse data. This combination improves recommendation quality and diversity by leveraging the strengths of both techniques.
4. Knowledge Graph-Based Recommender Systems: The use of knowledge graphs allows EARS to map complex relationships between users, products, and contexts, further enhancing recommendation relevance. By incorporating diverse data sources, these systems can generate more accurate and personalized recommendations.
5. Natural Language Processing (NLP): NLP is used to enhance the interaction between users and the system. EARS employs advanced NLP techniques to interpret unstructured data (such as reviews and comments), thereby improving recommendation accuracy and generating explanations in natural language, making the system more user-friendly.

The report outlines the current challenges in the recommendation domain, including issues related to data privacy, cold start problems, and the difficulty in cross-domain recommendations.

It also explores the need for federated collaboration across recommendation systems from multiple domains, without violating data privacy laws, such as GDPR.

The SoTA section of the document critically reviews existing techniques, including collaborative filtering, content-based filtering, and hybrid models. The EARS innovations are positioned as an advancement beyond these traditional methods, particularly through the use of federated learning, which allows systems to learn from other domains without sharing sensitive data directly.

In conclusion, the EARS project seeks to address critical issues in recommendation systems by implementing state-of-the-art technologies like federated learning, explainable AI, and knowledge graphs, ensuring more accurate, personalized, and trustworthy recommendations while preserving data privacy.

6. References

1. Ammad-Ud-Din, M., Ivannikova, E., Khan, S. A., Oyomno, W., Fu, Q., Tan, K. E., & Flanagan, A. (2019). Federated collaborative filtering for privacy-preserving personalized recommendation system. arXiv preprint arXiv:1901.09888.
2. Perifanis, V., & Efraimidis, P. S. (2022). Federated neural collaborative filtering. *Knowledge-Based Systems*, *242*, 108441.
3. He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., & Wang, M. (2020, July). Lightgcn: Simplifying and powering graph convolution network for recommendation. In Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval (pp. 639-648).
4. Ying, R., He, R., Chen, K., Eksombatchai, P., Hamilton, W. L., & Leskovec, J. (2018, July). Graph convolutional neural networks for web-scale recommender systems. In Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining (pp. 974-983).
5. Wang, X., Qin, J., Deng, S., & Zeng, W. (2023). Knowledge-Aware Enhanced Network Combining Neighborhood Information for Recommendations. Applied Sciences, 13(7), 4577.
6. Liu, T., Zhang, X., Wang, W., & Mu, W. (2024). KAT: knowledge-aware attentive recommendation model integrating two-terminal neighbor features. International Journal of Machine Learning and Cybernetics, 1-18.
7. Han, T., Wang, P., Niu, S., & Li, C. (2022, April). Modality matches modality: Pretraining modality-disentangled item representations for recommendation. In Proceedings of the ACM Web Conference 2022 (pp. 2058-2066).
8. Liu, Q., Yao, E., Liu, C., Zhou, X., Li, Y., & Xu, M. (2023). M2GCN: multi-modal graph convolutional network for modeling polypharmacy side effects. Applied Intelligence, 53(6), 6814-6825.
9. Zhao, L., Pan, S., Xiang, E., Zhong, E., Lu, Z., & Yang, Q. (2013, June). Active transfer learning for cross-system recommendation. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 27, No. 1, pp. 1205-1211).
10. Yuan, F., He, X., Karatzoglou, A., & Zhang, L. (2020, July). Parameter-efficient transfer from sequential behaviors for user modeling and recommendation. In Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval (pp. 1469-1478).
11. Ding, X., Zhang, F., & Jin, H. (2019). Data anonymization for big crowdsourcing data. IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 1–6.
12. Duan, M., Liu, D., Chen, X., et al. (2020). Self-balancing federated learning with global imbalanced data in mobile systems. IEEE Transactions on Parallel and Distributed Systems, 32(1), 59–71.

13. Zhu, L., Huang, T., & Li, Z. (2023). Heterogeneous Federated Learning: State-of-the-art and Research Challenges. arXiv preprint arXiv:2307.10616.
14. Hardy, S., Henecka, W., Ivey-Law, H., et al. (2017). Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. arXiv preprint arXiv:1711.10677.
15. Zhao, Z., Zheng, L., & Xu, W. (2023). Decentralized Federated Learning: A Survey and Perspective. arXiv preprint arXiv:2306.01603.
16. Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., He, Q.J.P.o.t.I.: A comprehensive survey on transfer learning. 109, 43-76 (2020)
17. Weiss, K., Khoshgoftaar, T.M., Wang, D.J.J.o.B.d.: A survey of transfer learning. 3, 1-40 (2016)
18. Torrey, L., Shavlik, J.: Transfer learning. Handbook of research on machine learning applications and trends: algorithms, methods, and techniques, pp. 242-264. IGI global (2010)
19. Vrbančič, G., Podgorelec, V.J.I.A.: Transfer learning with adaptive fine-tuning. 8, 196197-196211 (2020)
20. Iman, M., Arabnia, H.R., Rasheed, K.J.T.: A review of deep transfer learning and recent advancements. 11, 40 (2023)
21. Moreo, A., Esuli, A., Sebastiani, F.J.A.T.o.K.D.f.D.: Lost in transduction: Transductive transfer learning in text classification. 16, 1-21 (2021)
22. Pan, S.J., Yang, Q.J.I.T.o.k., engineering, d.: A survey on transfer learning. 22, 1345-1359 (2009)
23. Niu, S., Liu, Y., Wang, J., Song, H.J.I.T.o.A.I.: A decade survey of transfer learning (2010–2020). 1, 151-166 (2020)
24. Zhang, W., Deng, L., Zhang, L., Wu, D.J.I.C.J.o.A.S.: A survey on negative transfer. 10, 305-329 (2022)
25. Cai, W., Zheng, J., Pan, W., Lin, J., Li, L., Chen, L., Peng, X., Ming, Z.J.N.: Neighborhood-enhanced transfer learning for one-class collaborative filtering. 341, 80-87 (2019)
26. Khan, M.M., Ibrahim, R., Ghani, I.J.A.C.S.: Cross domain recommender systems: A systematic literature review. 50, 1-34 (2017)
27. Thorat, P.B., Goudar, R.M., Barve, S.J.I.J.o.C.A.: Survey on collaborative filtering, content-based filtering and hybrid recommendation system. 110, 31-36 (2015)
28. Lika, B., Kolomvatsos, K., Hadjiefthymiades, S.J.E.s.w.a.: Facing the cold start problem in recommender systems. 41, 2065-2073 (2014)
29. Wischenbart, M., Firmenich, S., Rossi, G., Bosetti, G., Kapsammer, E.J.M.T., Applications: Engaging end-user driven recommender systems: personalization through web augmentation. 80, 6785-6809 (2021)
30. Vorm, E.S., Miller, A.D.: Assessing the value of transparency in recommender systems: an end-user perspective. (2018)

31. Zhan, R., Christakopoulou, K., Le, Y., Ooi, J., Mladenov, M., Beutel, A., Boutilier, C., Chi, E., Chen, M.: Towards content provider aware recommender systems: A simulation study on the interplay between user and provider utilities. In: Proceedings of the Web Conference 2021, pp. 3872-3883. (Year)
32. Zhan, R., Christakopoulou, K., Le, Y., Ooi, J., Mladenov, M., Beutel, A., Boutilier, C., Chi, E., Chen, M.: Towards content provider aware recommender systems. In: Proceedings of the Web Conference. (Year)
33. Ha, S.H.J.I.I.S.: Digital content recommender on the internet. 21, 70-77 (2006)
34. Confalonieri, R., Coba, L., Wagner, B., & Besold, T. R. (2020). A historical perspective of explainable artificial intelligence. WIREs Data Mining and Knowledge Discovery, 11(1). https://doi.org/10.1002/widm.1391
35. Foley, Patrick & Sheller, Micah & Edwards, Brandon & Pati, Sarthak & Riviera, Walter & Sharma, Mansi & Moorthy, Prakash & Wang, Hans & Martin, Jason & Mirhaji, Parsa & Shah, Prashant & Bakas, Spyridon. (2022). OpenFL: the open federated learning library. Physics in Medicine & Biology. 67. 10.1088/1361-6560/ac97d9.
36. Daole, M., Schiavo, A., Bárcena, J. L. C., Ducange, P., Marcelloni, F., & Renda, A. (2023). OpenFL-XAI: Federated learning of explainable artificial intelligence models in Python. SoftwareX, 23, 101505.
37. H. Ludwig, N. Baracaldo, G. Thomas, Y. Zhou, A. Anwar, S. Rajamoni, Y. Ong, J. Radhakrishnan, A. Verma, M. Sinn, M. Purcell, A. Rawat, T. Minh, N. Holohan, S. Chakraborty, S. Whitherspoon, D. Steuer, L. Wynter, H. Hassan, S. Laguna, M. Yurochkin, M. Agarwal, E. Chuba, A. Abay, Ibm federated learning: an enterprise framework white paper v0.1, 2020. URL: https://arxiv.org/abs/2007.10987. doi:1 0 . 4 8 5 5 0 / A R X I V . 2 0 0 7 . 1 0 9 8 7.
38. M. Polato, R. Esposito, M. Aldinucci, Boosting the federation: Cross-silo federated learning without gradient descent, 2022. doi:1 0 . 1 1 0 9 / I J C N N 5 5 0 6 4 . 2 0 2 2 . 9 8 9 2 2 8 4.
39. Y. Wu, S. Cai, X. Xiao, G. Chen, B. C. Ooi, Privacy preserving vertical federated learning for tree-based models 13 (2020) 2090–2103.
40. J. L. C. Bárcena, P. Ducange, A. Ercolani, F. Marcelloni, A. Renda, An approach to federated learning of explainable fuzzy regression models, in: 2022 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), IEEE, 2022, pp. 1–8.
41. Zhang, J.-C., Azlan Mohd Zain, Zhou, K.-Q., Chen, X., & Zhang, R.-M. (2024). Review of recommender systems based on knowledge graph embedding. Expert Systems with Applications, 250, 123876–123876. https://doi.org/10.1016/j.eswa.2024.123876
42. Guo, Q., Zhuang, F., Qin, C., Zhu, H., Xie, X., Xiong, H., & He, Q. (2022). A Survey on Knowledge Graph-Based Recommender Systems. IEEE Transactions on Knowledge and Data Engineering, 34(8), 3549–3568. https://doi.org/10.1109/tkde.2020.3028705