# Query Recommendation System

Ghaith Kouki
ghaith.kouki@studenti.unitn.it
University of trento

Jihed Dachraoui
jihed.dachraoui@studenti.unitn.it
University of trento

## 1 INTRODUCTION

Over the past few years, and due to the development of communication technology, a massive amount of information has been widely available to users at any time, which has led to information overload. The question of how to help users find information that satisfies their needs is an important issue, and the recommender system is considered a significant way to solve this problem. Query recommendation systems can be very effective in improving the user experience, as such systems can help users find relevant information both quickly and efficiently by suggesting possible queries as they type. These systems are commonly used in search engines, online libraries, and other information-rich websites to help users find the information they need easily.

Besides improving the user experience, query recommendation systems can also be beneficial to the organization or website that hosts the system. By helping users better reach the information they need, these systems can reduce the workload of customer service and support teams, improving the overall efficiency of the search process.

However, implementing a query recommendation system also presents a number of challenges. One of the main challenges is ensuring that the system is accurate and reliable, as users can become frustrated if the suggested queries do not match their requirements. It is also important to ensure that the system respects users' privacy and does not compromise sensitive information.

When designing and deploying a query recommendation system, it is important to consider a number of key factors. These include the target audience for the system, the type of information users are likely to search for, and the resources available to build and maintain the system. By carefully considering these criteria, we can develop a query recommendation system that meets the needs of its users and makes it easier for them to find the information they need.

There are several strategies for building a query recommender system. These methods allow the system to learn from past user queries and search results to better predict what users are looking for.

In this report, we will explore how to build a query recommender system and discuss the various benefits and challenges of implementing such a system. We will also review some of the key considerations for designing and deploying an effective query recommendation system.

## 2 PROBLEM STATEMENT

The main problem that a query recommendation system aims to solve is the difficulty that users often face in finding relevant information quickly and efficiently. With the vast amount of information available online, it can be challenging for users to identify the most relevant sources, particularly when searching for complex or specialized information. This can lead to frustration and a poor user experience, as users may have to sift through numerous irrelevant search results before finding what they are looking for.

We suppose that we have a set of users and, for each user, a set of queries that they have sent to the database. When a user asks a question to the database, he receives a set of tuples in answer. If he is satisfied with this answer, he gives them a high score, otherwise he gives them a low score. So given these inputs in a csv format:

(1) A relational table representing our database where each column is an attribute
(2) A set of users that is nothing more than a set of ids.
(3) A set of queries where each query has a unique query id and its definition. The definition is a set of conditions on the dataset attributes.
(4) A User-Query utility matrix that contains the scores from 1 to 100 assigned by users to queries. Each row of the matrix represents a user, and each column represents a query. The entries in the matrix are the scores that users have given to items. The matrix has missing values knowing that not every user tried every query and rated it.

Thus, given this utility matrix, our task is to develop a method to fill in the missing values in the matrix. The completed matrix should accurately reflect users' preferences and be able to predict their ratings for queries they have not yet rated so that we can recommend the top-k queries that might be of interest to the user u. We also need to consider the scalability and complexity of the algorithm, as the utility matrix may be large and the recommender system may need to handle a large number of users and queries. In summary, the algorithm presented in this work is a method for filling in the missing values in the utility matrix and thus recommending to the user a set of queries whose answer set contains data that might be of interest to the user. The query recommended to the user has not been requested by him in the database before, and it is believed that if he requests it, he will give it a high score.