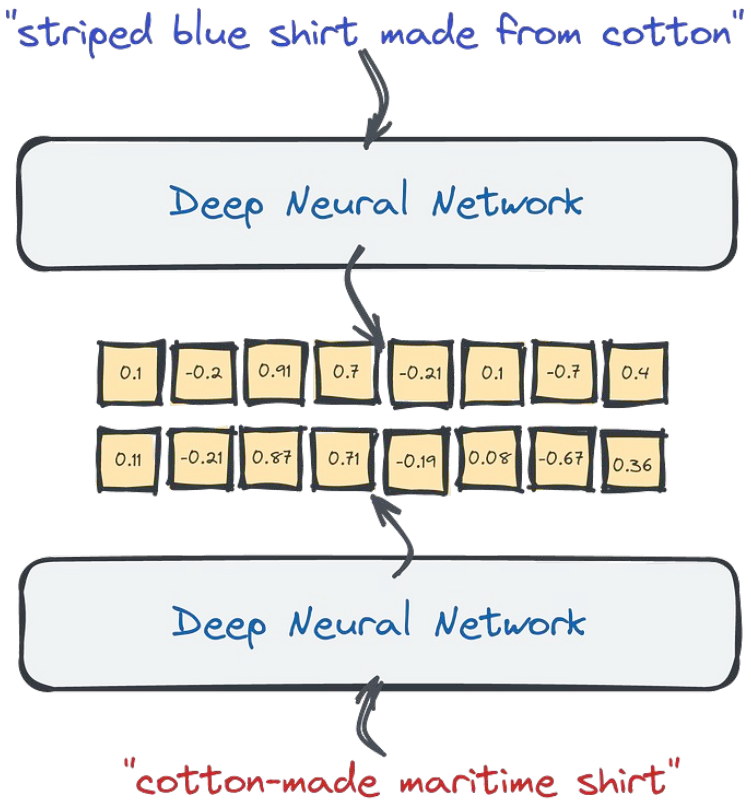
Beyond the known: exploratory and diversity search with vector embeddings

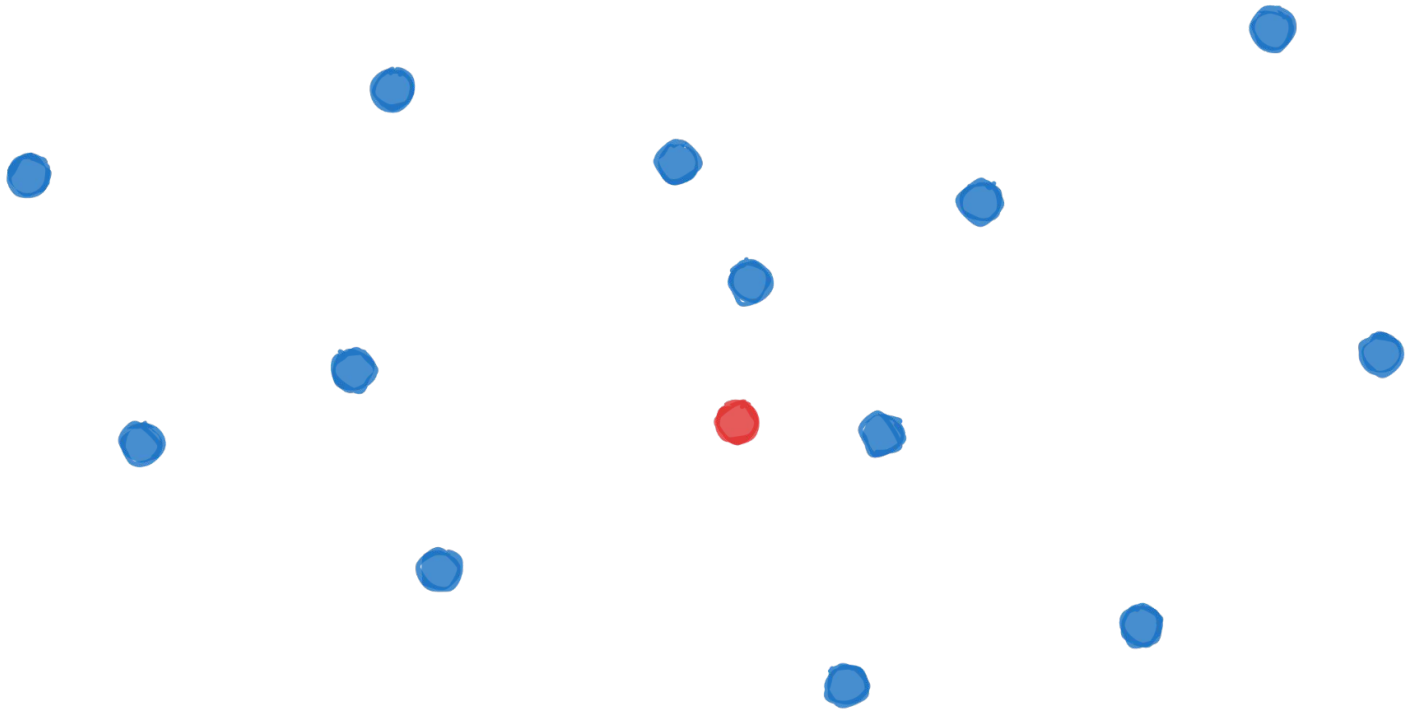
Kacper Łukawski, Developer Advocate, Qdrant

Vector search

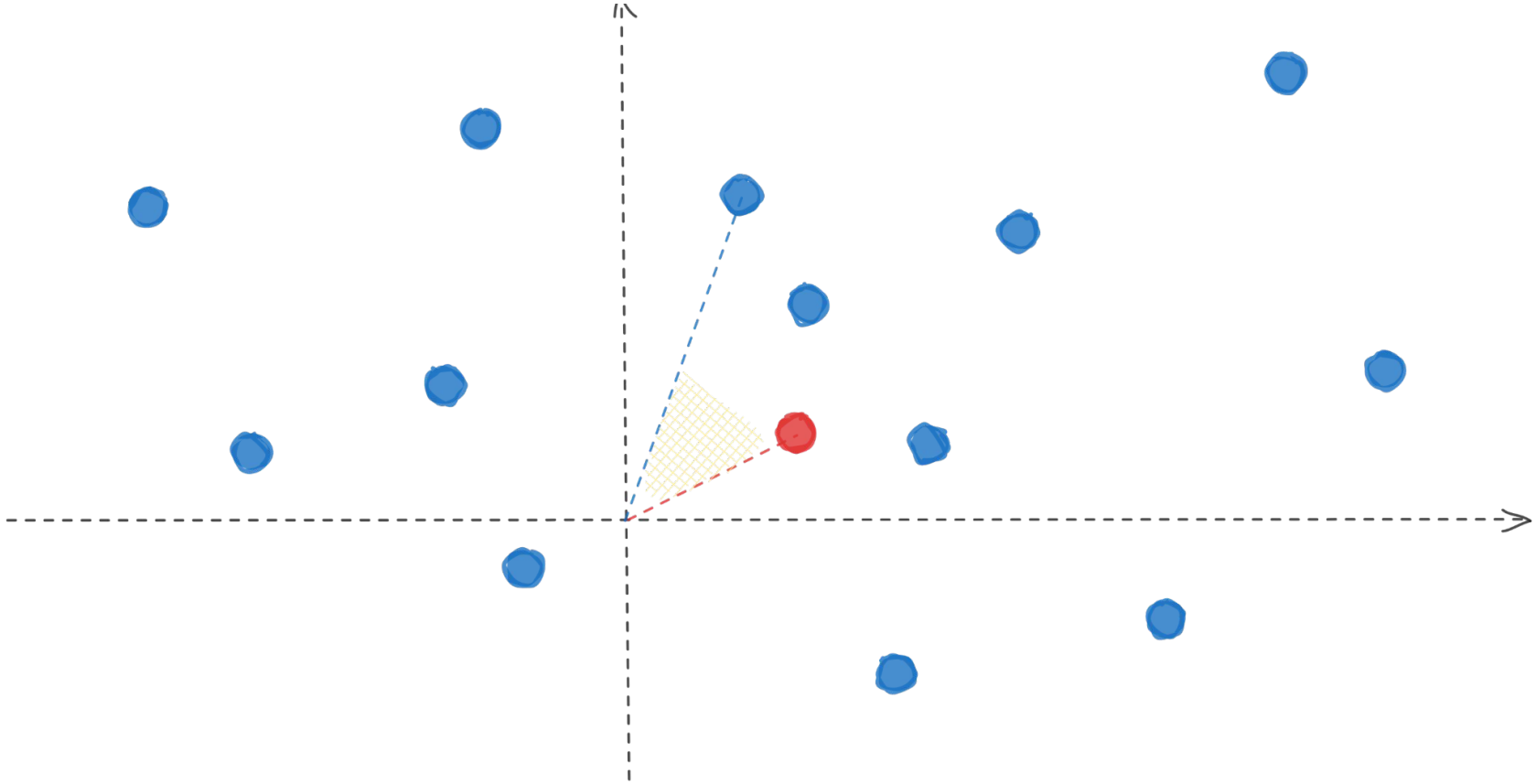
Basics of vector search

Documents are being modelled as fixed-dimensional vectors, created through some neural encoders.

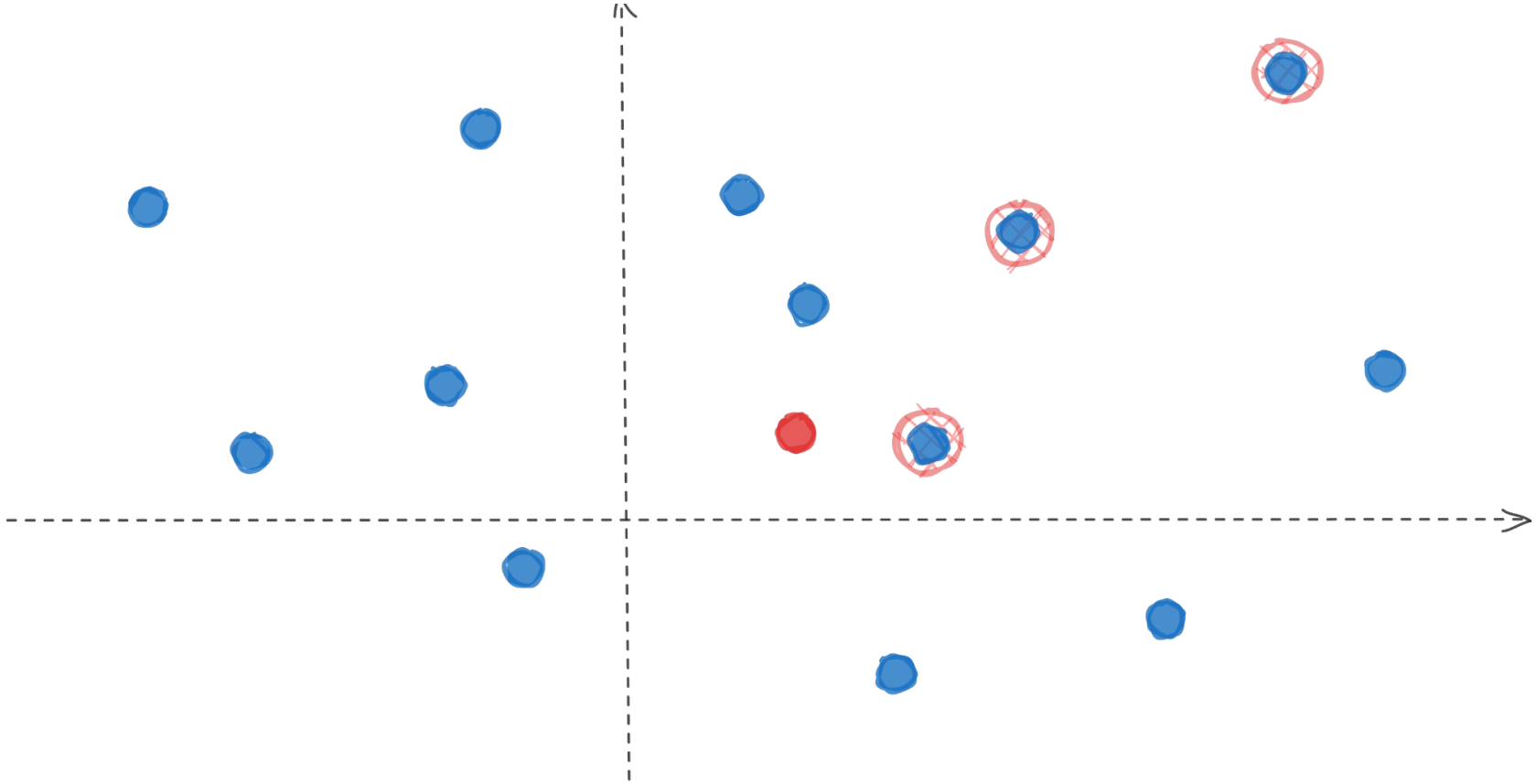




A distribution of document vectors and query vector in 2d space.



Calculating the cosine similarity is based on the angle between two vectors.

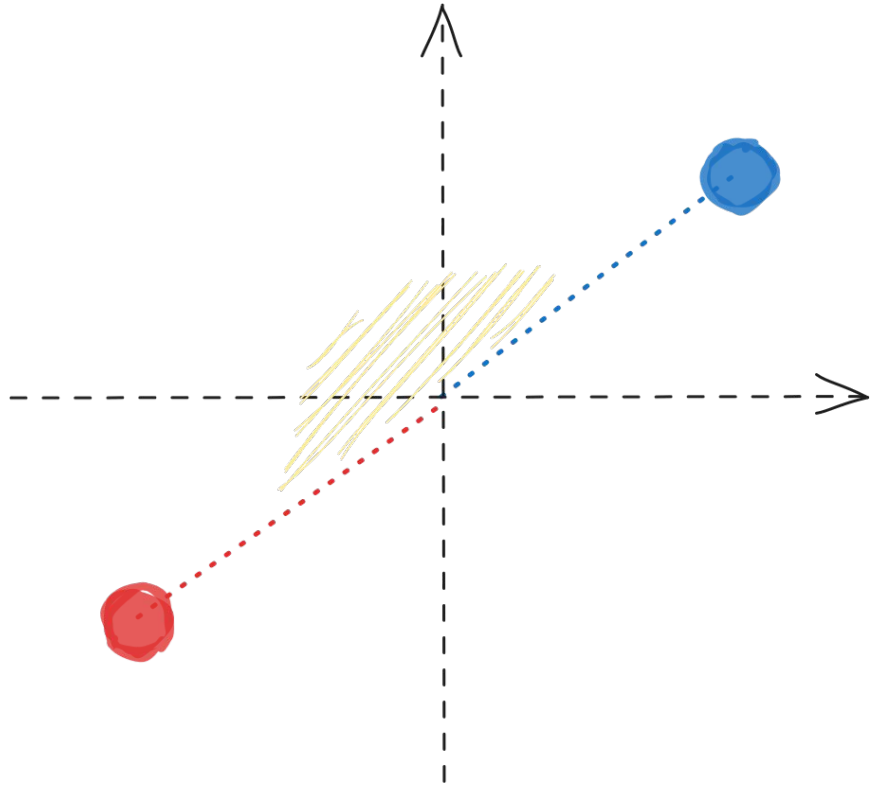


The three nearest neighbors of the query vector. The closest points will have the highest cosine similarity score (a value of 1 means a perfect match, while -1 is the opposite direction vector).

Approximate Furthest Neighbours

The cosine similarity of a vector with its negation is always -1.

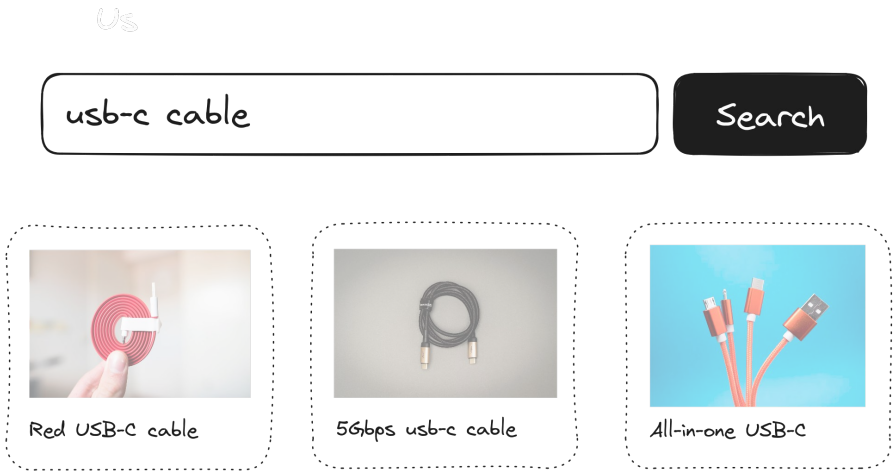
That works only with the cosine similarity, not for the other distance metrics.



Known-item search

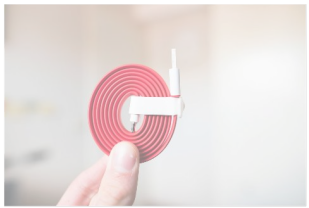
Known-item search

Whenever a user has a particular item in find, and can express it with a **query**, usually textual



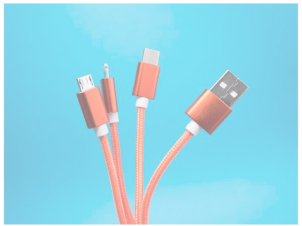
a set of wires used to transmit signals between different devices, also used to charge them, popular for mobile devices

Search



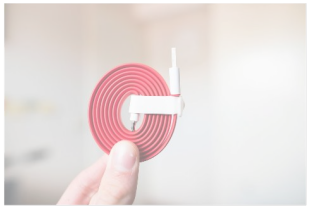
Red USB-C cable

5Gbps usb-c cable



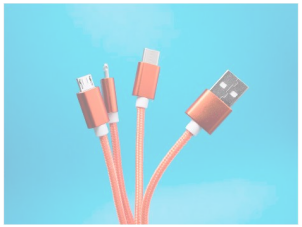
All-in-one USB-C

Search



Red USB-C cable

5Gbps usb-c cable

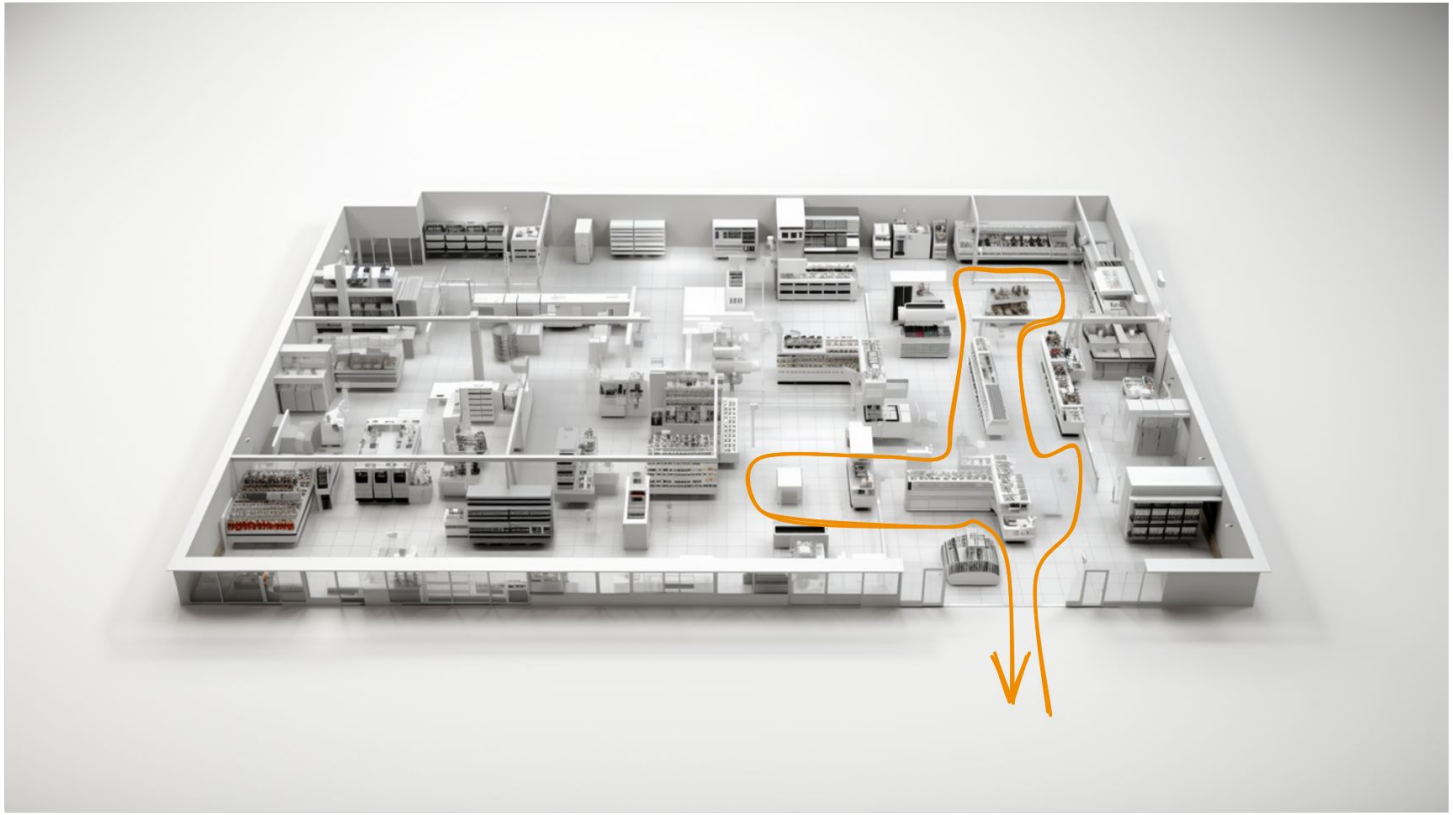


All-in-one USB-C

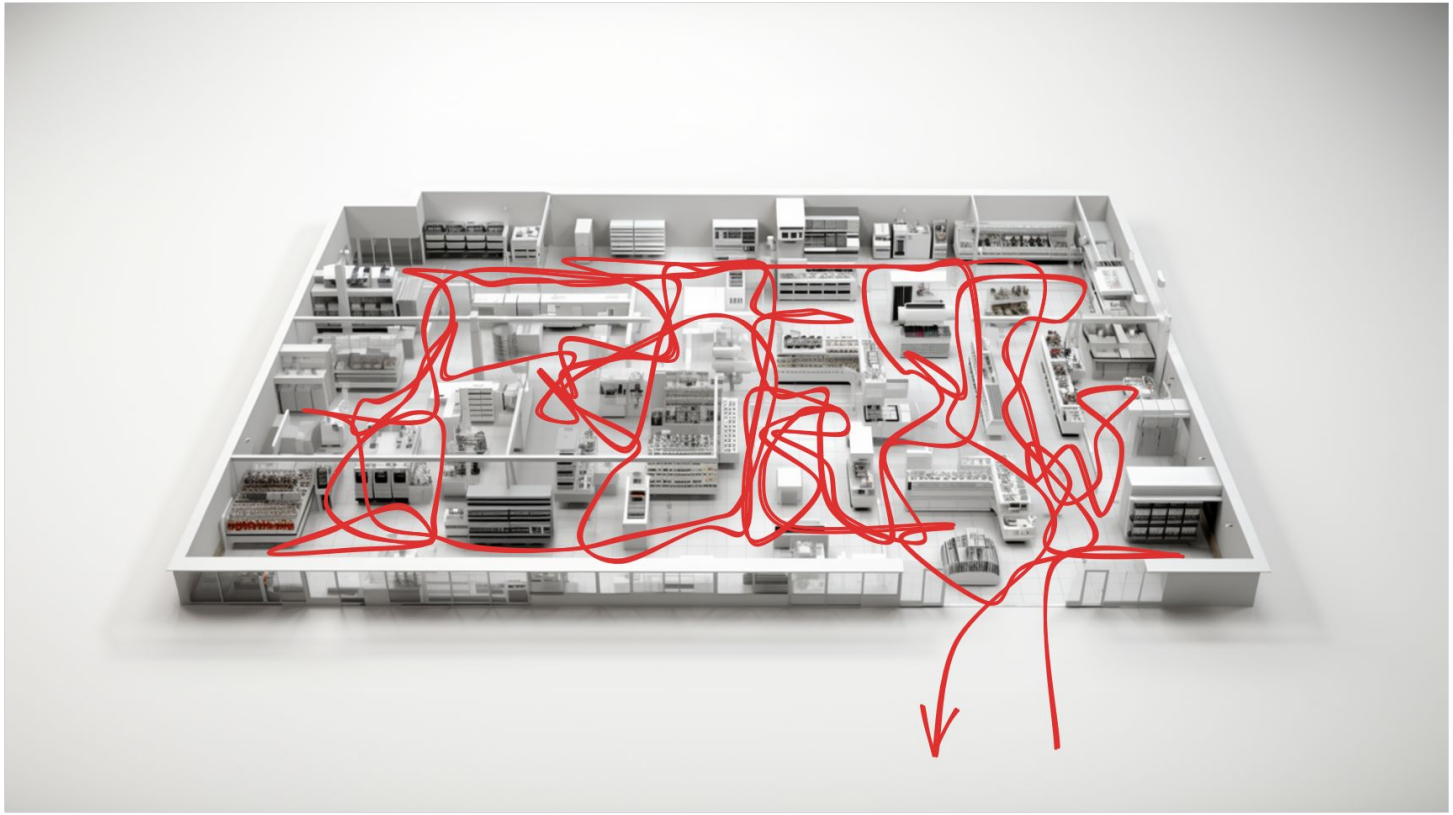
Measuring the quality of known-items search

There are various ways of verifying how satisfactory are the results of search for a particular query.

- Precision, Recall, F1
- DCG, nDCG
- Business metrics
- …



Known-items search is like visiting the favourite supermarket with a shopping list



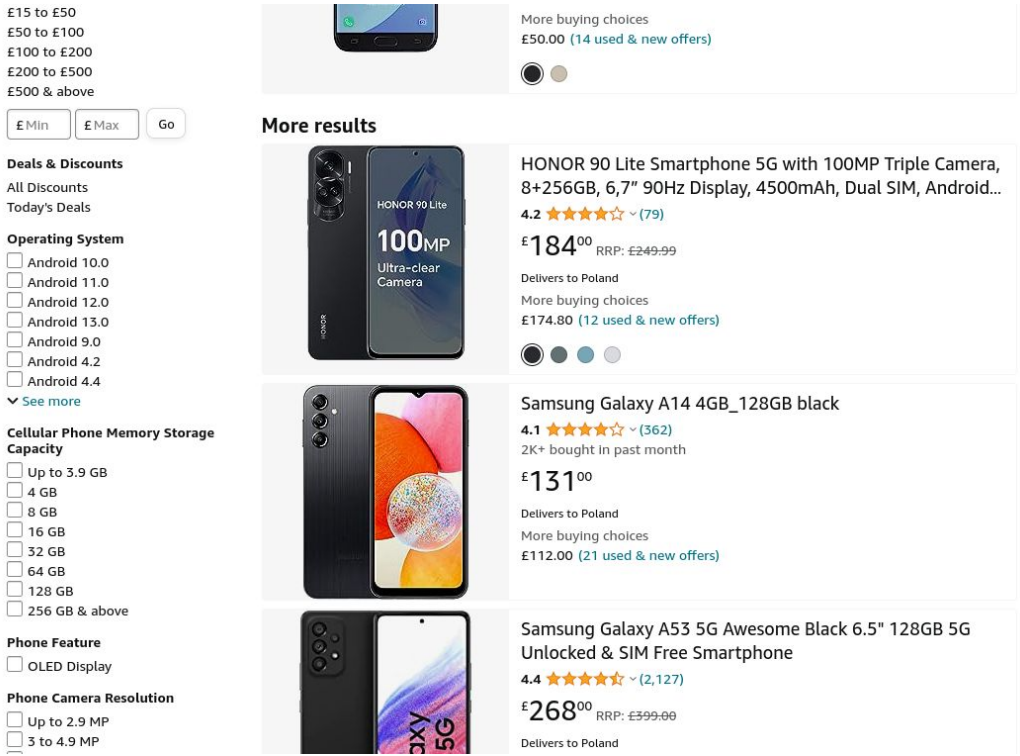
But without a specific need the journey may look differently

Exploratory search

A different kind of information exploration, usually conducted by people unfamiliar with a specific domain, or simply unsure about their goals.

Faceted search

One of the most popular ways to simplify exploring the datasets is to display **facets**. They might be used to narrow down the set of results to eventually find an item, by applying the constraints. Faceted search is also commonly known as **faceted navigation**.




Exploratory search is not a new concept

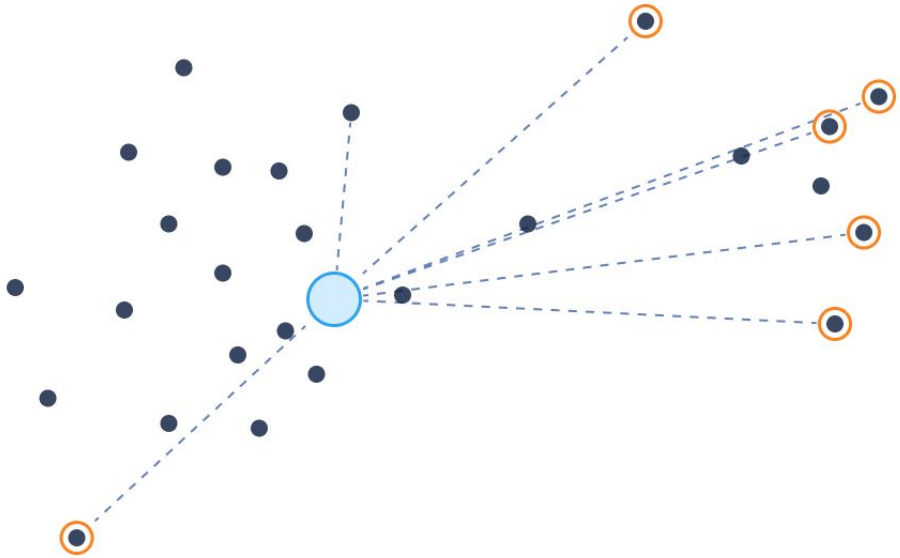
Exploratory search quality

Since user intentions were kept unexpressed, we cannot use the same metrics as we used to calculate for known-items search.

- Perhaps some business metrics

Dissimilarity search

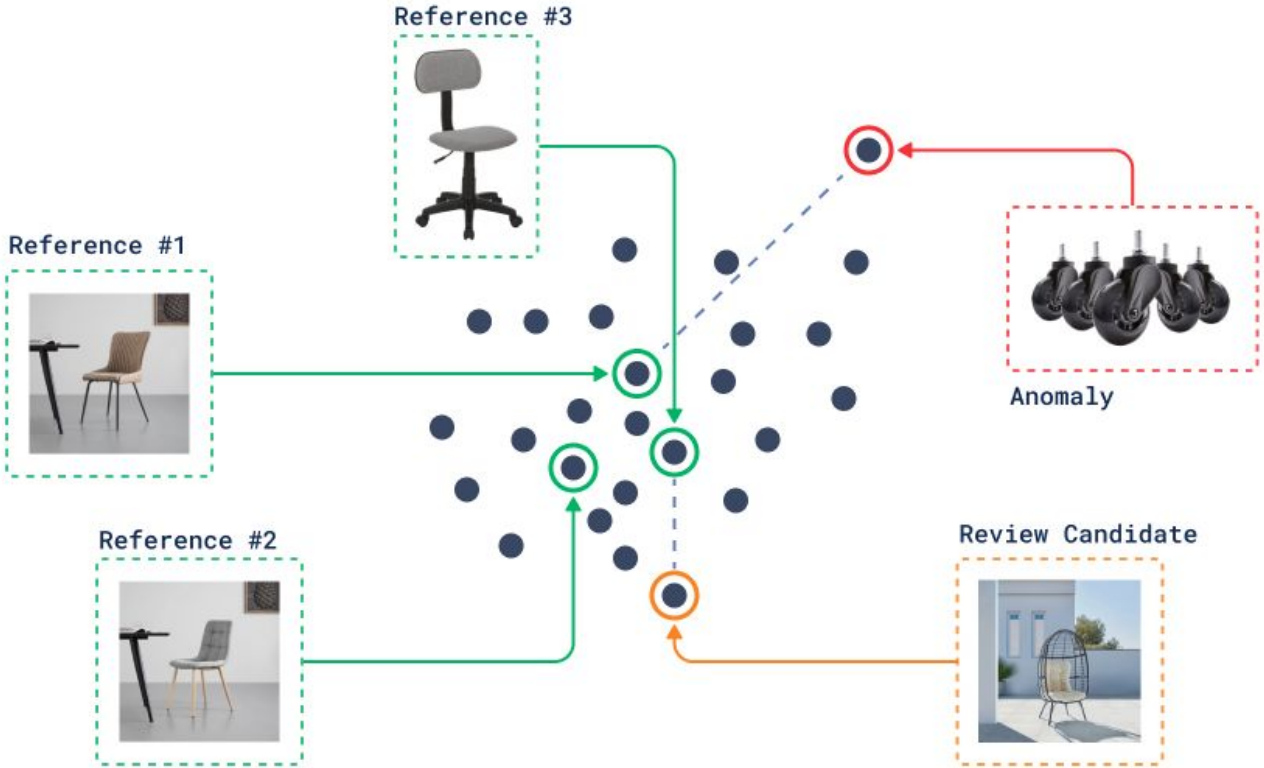
Finding the furthest points from the given query vector



Data quality

Mislabeling detection

Query:
Chair



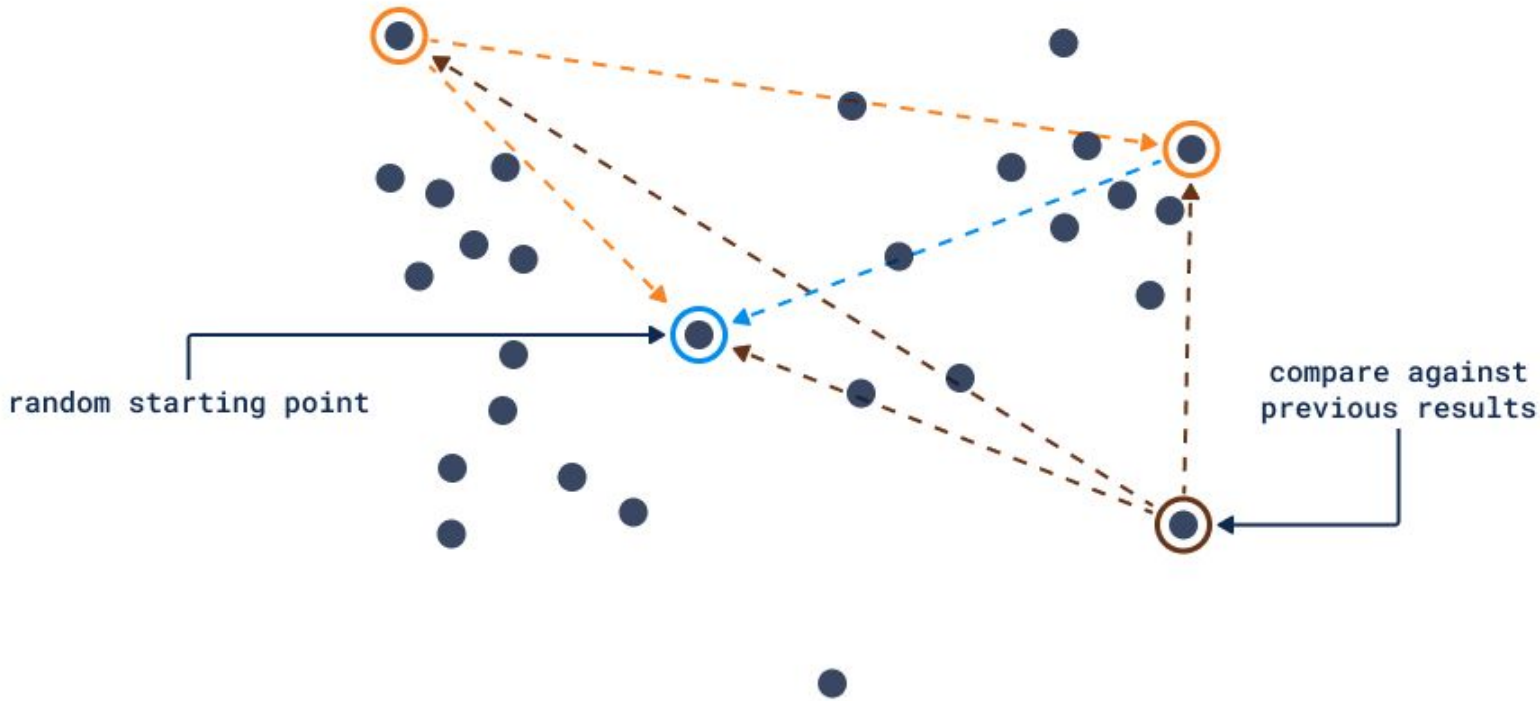
Outlier detection without any specific labels

Diversity search

Random vs similarity-based sampling

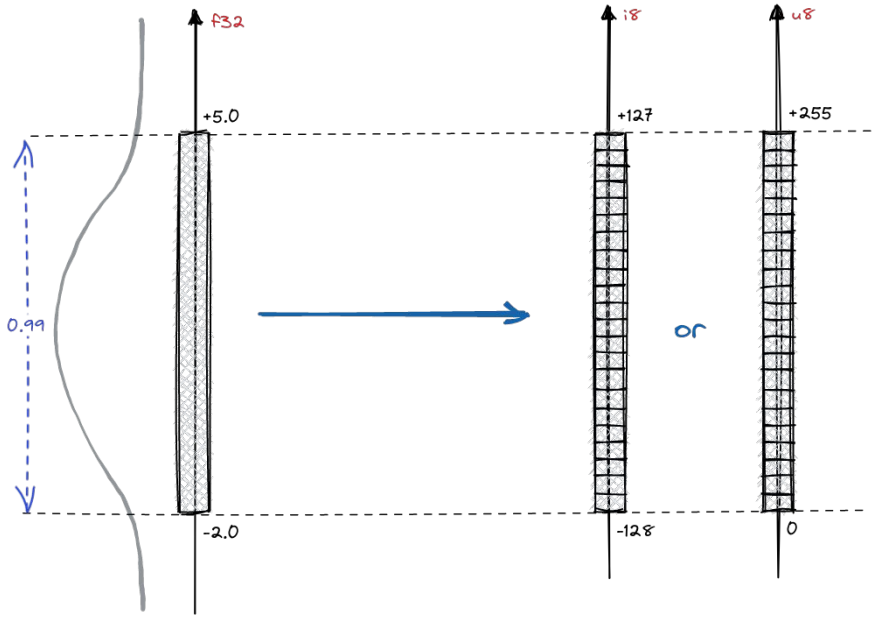
Random sampling might be biased toward more frequent types of documents.

Similarity-based sampling incorporates previously selected vectors and iteratively returns dissimilar items.



Diversity search as an iterative process

Speeding up vector search: Scalar Quantization



Binary Quantization

Pushing the Scalar Quantization to the limits.

Binary Quantization is a completely new feature introduced in Qdrant 1.5.0. Not all the embeddings work properly once binary quantized, but if so, dot product might be replaced with simple XOR operation.

- OpenAI `text-embedding-ada-002`
- Cohere AI `embed-english-v2.0`

Diversification

Search Result Diversification

*"The problem of search result diversification is **NP-hard**. Therefore, approximation algorithms have to exploit inherent structural properties of the solution space to achieve adequate system response times"*

Current Approaches to Search Result Diversification

Enrico Minack, Gianluca Demartini, and Wolfgang Nejdl

L3S Research Center, Leibniz Universität Hannover,
30167 Hannover, Germany, {lastname}@L3S.de

Abstract With the growth of the Web and the variety of search engine users, Web search effectiveness and user satisfaction can be improved by diversification. This paper surveys recent approaches to search result diversification in both full-text and structured content search. We identify commonalities in the proposed methods describing an overall framework for result diversification. We discuss different diversity dimensions and measures as well as possible ways of considering the relevance / diversity trade-off. We also summarise existing efforts evaluating diversity in search. Moreover, for each of these steps, we point out aspects which are missing in current approaches as possible directions for future work.

1 Introduction

In the last years, the Web has become the largest and most consulted public source of information, and Web search emerged as the primary technique for finding relevant information on the Web. Search engines usually provide a long list of results that contains thousands of entries, where the most relevant results tend to be quite similar [1]. In particular for informational queries [2], users reading through a list of relevant but redundant pages quickly stop as they do not expect to learn more. The phenomenon of *saturated user satisfaction*

Maximal Marginal Relevance

Maximal Marginal Relevance

MMR is a method proposed back in 1998, in the paper "*The Use of MMR, Diversity-Based Reranking for Reordering Documents and Producing Summaries*". The main goal is to reduce redundancy and increase diversity in the search results.

1. Prefetch **significantly more** documents, than the requested number of results, using a standard vector search procedure.
2. Iteratively go through the list of the prefetched results and calculate the MMR for all the documents that were not selected yet:

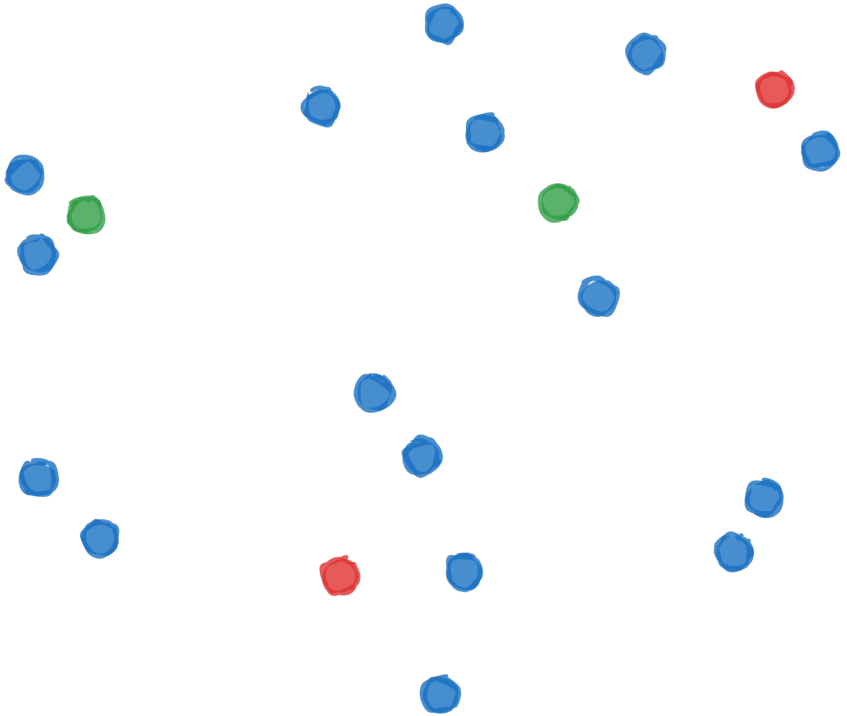
$$MMR \stackrel{\text{def}}{=} Arg \max_{D_i \in R \setminus S} \left[\lambda (Sim_1(D_i, Q) - (1-\lambda) \max_{D_j \in S} Sim_2(D_i, D_j)) \right]$$

   Each iteration adds a single previously unselected item with the maximum value of MMR.
3. Repeat until reach the number of requested results.

Short-term recommendations

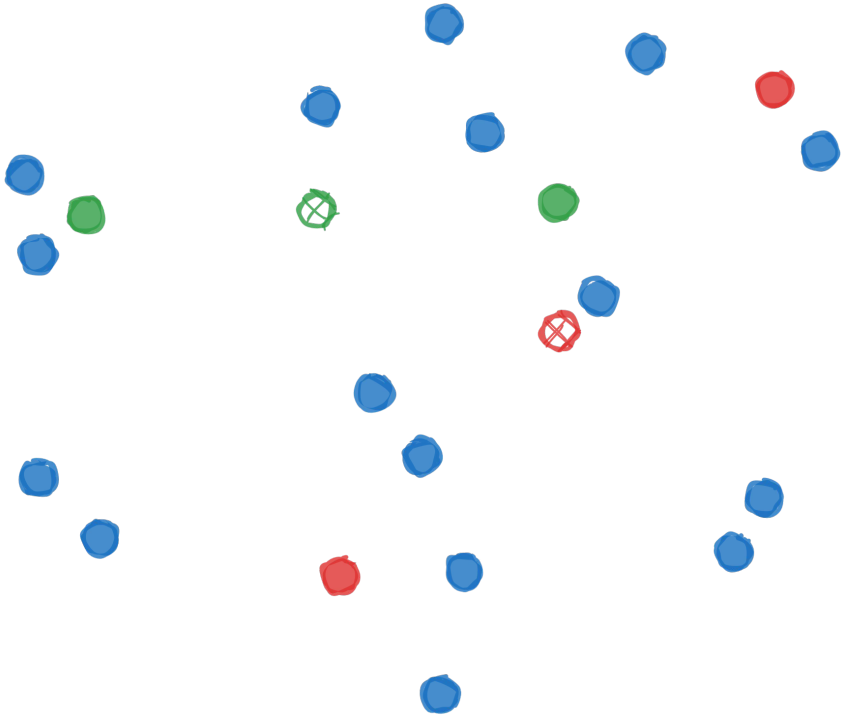
Recommendations

In a short term, user preferences might be described by the items they liked/disliked. If we have the items vectorized, averaging them makes sense.



Recommendations

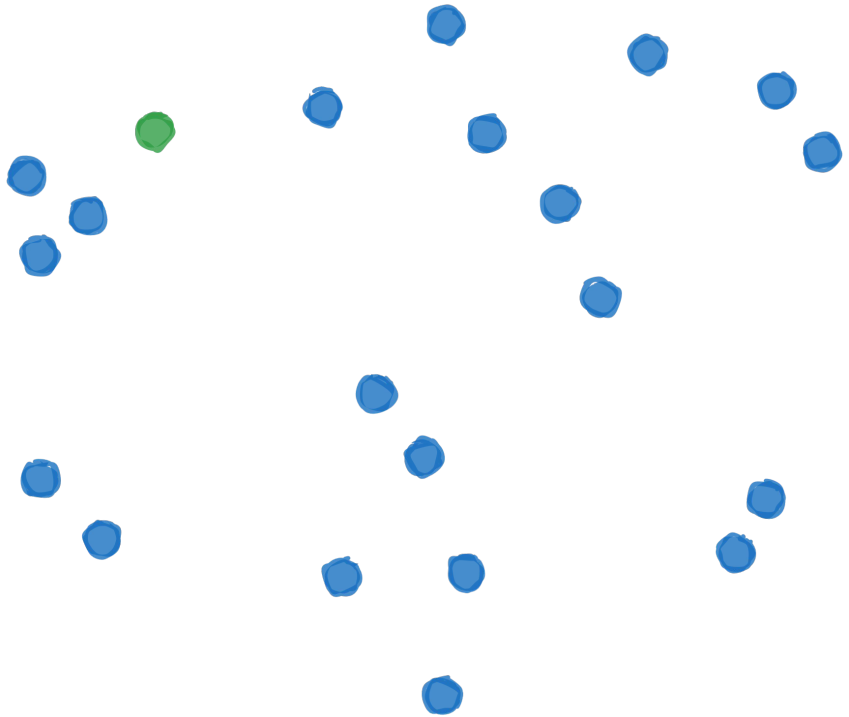
Both liked and disliked items might be averaged to create centroids.



Recommendations

Their combination is then used as a query, so we perform a standard vector search using a single query vector.

Fact: We operate using existing vectors only, so creating new embeddings is unnecessary.



Recommendation API

> ⚠ Negative vectors is an experimental functionality that is not guaranteed to work with all kind of embeddings.

In addition to the regular search, Qdrant also allows you to search based on multiple vectors already stored in the collection. This API uses vector search without involving the neural network encoder for already encoded objects.

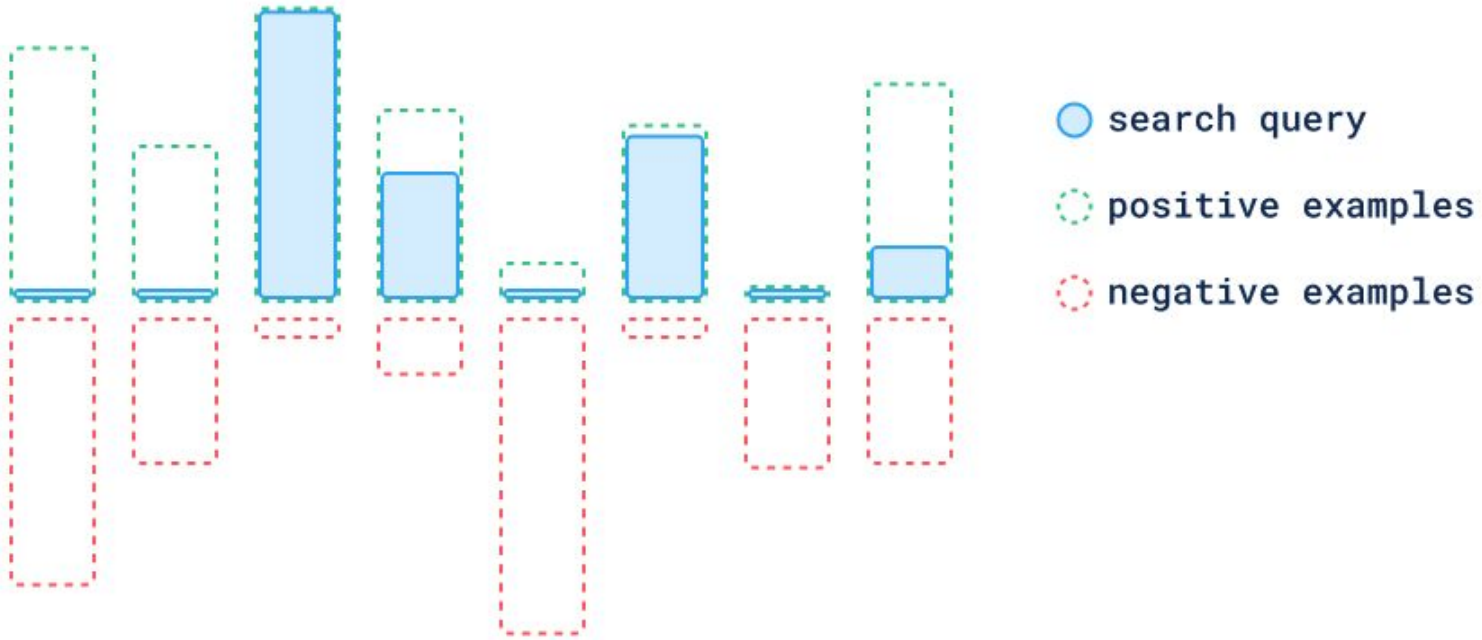
The recommendation API allows specifying several positive and negative vector IDs, which the service will combine into a certain average vector.

```
average_vector = avg(positive_vectors) + ( avg(positive_vectors) - avg(negative_vectors) )
```

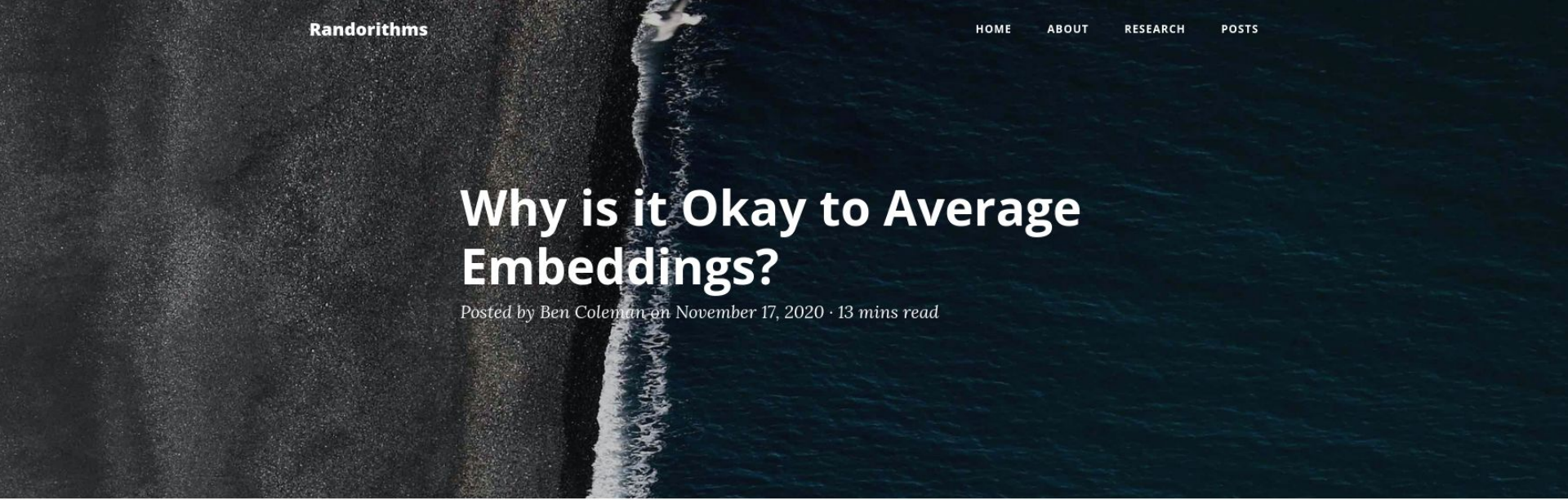
If there is only one positive ID provided - this request is equivalent to the regular search with vector of that point.

Vector components that have a greater value in a negative vector are penalized, and those that have a greater value in a positive vector, on the contrary, are amplified. This average vector will be used to find the most similar vectors in the collection.

Qdrant Recommendation API



Qdrant approach emphasizes positive examples more than negative ones



Why is it Okay to Average Embeddings?

Posted by Ben Coleman on November 17, 2020 · 13 mins read

People often summarize a "bag of items" by adding together the embeddings for each individual item. For example, graph neural networks summarize a section of the graph by averaging the embeddings of each node [1]. In NLP, one way to create a sentence embedding is to use a (weighted) average of word embeddings [2]. It is also common to use the average as an input to a classifier or for other downstream tasks.

I have heard the argument that the average is a good representation because it includes information from all of the individual components. Each component "pulls" the vector in a new direction, so the overall summary has a unique direction that is based on all of the components. But these arguments bother me because addition is not one-to-one: there are an unlimited number of ways to pick embeddings with

Powered by drant

SEMANTIC SEARCH
Food Discovery

Tuna Salad

Chef Salad

Recommendation results

Select food items you like and dislike to improve your search results.

2. Çoban Salata

Sallad, tomat, gurka, peppar, lök, citron, o...

Ceviche

Argentinske rejer m. mango, avocado, rad...

Small Salad Bowl

Ceviche

Argentinske rejer m. mango, avocado, rad...

Grilled Veg Salad

Grilled peppers, aubergines and artichoke...

Small Mixed Salad

97. Insalata Mare

gemischter Salat, Tomaten, Mais, Gurken ...

Small Mixed Salad

Green salad

Green salad

Tomato Salad

2. Farmer's Salad

Iceberg lettuce, feta cheese, ham, tomato...

Αγγουροντομάτα

Relative distance recommendations

Future plans

Questions?

Kacper Łukawski
Developer Advocate
Qdrant

https://www.linkedin.com/in/kacperlukawski/
https://twitter.com/LukawskiKacper
https://github.com/kacperlukawski

A free forever 1GB cluster included for trying out. No credit card required.