Introduction Evaluations

Pavlos Protopapas

RAG - Evaluations

There are two areas of interventions:

1. Retrieval
2. Generation

RAG - Evaluations

There are two areas of interventions:

1. **Retrieval**
2. Generation

Retrieval Evaluations

For retrieval evaluation, we want metrics that can accurately quantify the quality of the information retrieved in response to queries.

There can be two types of such evaluation metrics:

1. Non-Rank based
2. Rank based

Retrieval Evaluations – Non-Rank Based

Non-Rank based metrics examine if the results are relevant or irrelevant, regardless of the order they're in.

Some examples of such metrics are:

1. Precision@k
2. Recall@k
3. F1@k

k is the number of results considered.

k operates like a sliding window, allowing us to consider the metric's value at a given position.

Retrieval Evaluations – Non-Rank Based

Precision@k

Precision@k examines how many items in the result set are relevant.

$$Precision@k = \frac{true\ positives@k}{true\ positives@k + false\ positives@k}$$

Precision@k is ideal when the accuracy of each result is more important than finding every relevant document.

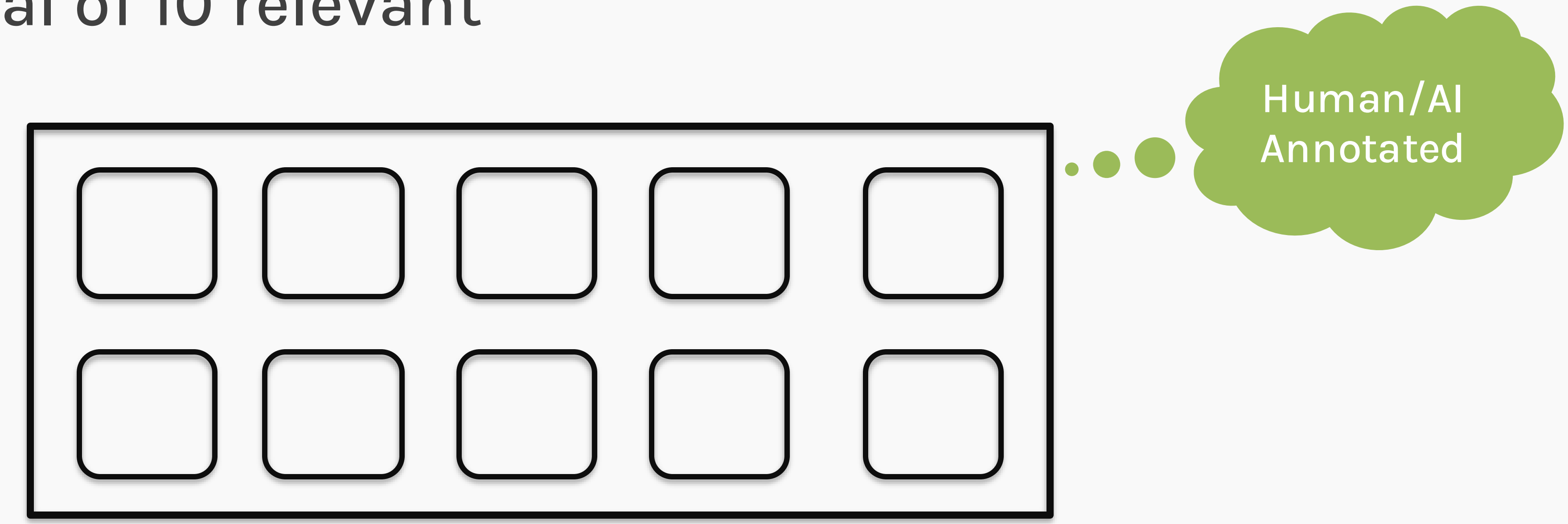
Retrieval Evaluations – Non-Rank Based

Precision@k

$$Precision@k = \frac{true\ positives@k}{true\ positives@k + false\ positives@k}$$

For a particular query, there will always be relevant documents.

Let there be a total of 10 relevant documents

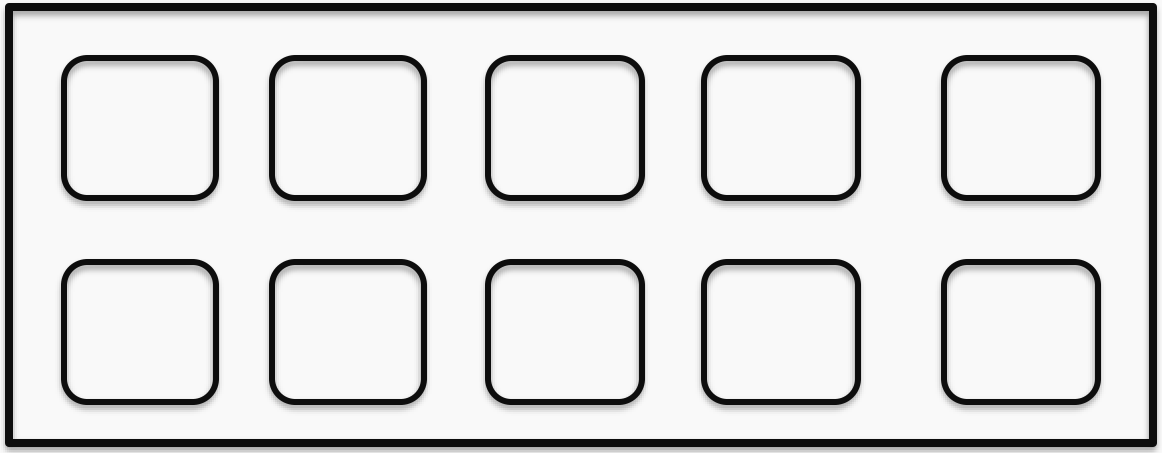


Relevant Documents (RD)

Retrieval Evaluations – Non-Rank Based

Precision@k

$$Precision@k = \frac{true\ positives@k}{true\ positives@k + false\ positives@k}$$



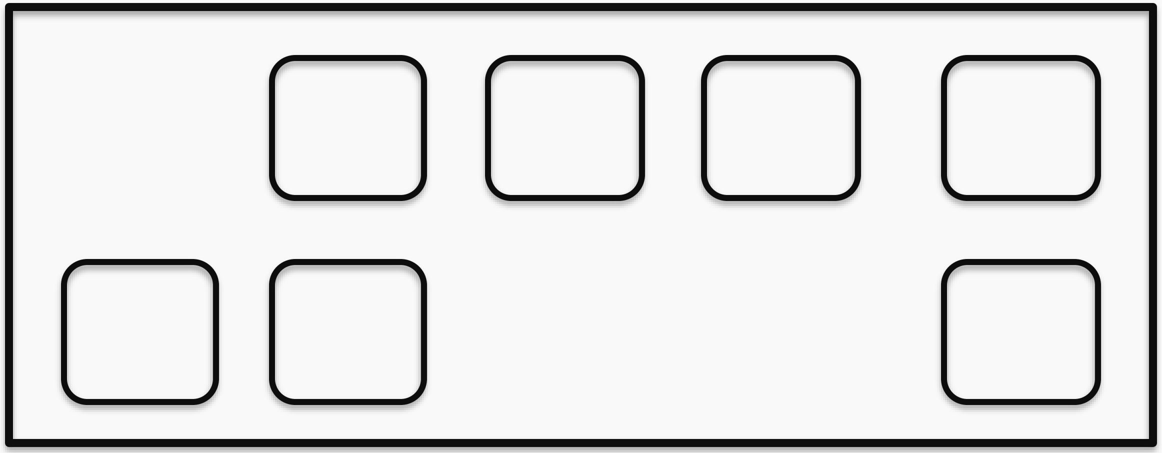
Relevant Documents (RD)

For example, let us assume that our RAG retrieves a total of 5 documents that it thinks is relevant.

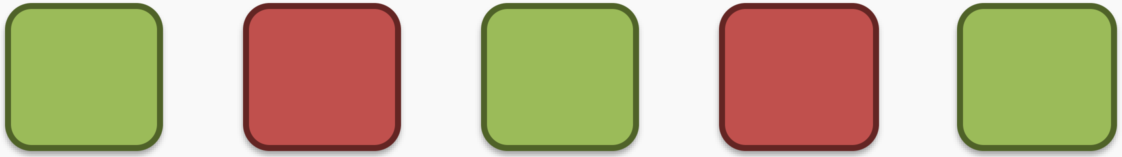
Retrieval Evaluations – Non-Rank Based

Precision@k

$$Precision@k = \frac{true\ positives@k}{true\ positives@k + false\ positives@k}$$



Relevant Documents (RD)



For example, let us assume that our RAG re

K=5 documen

relevant.

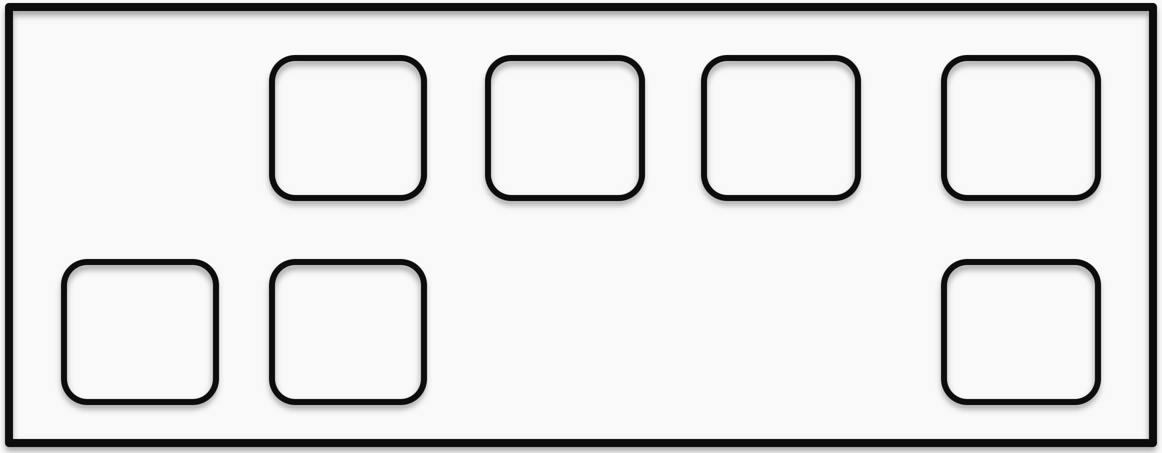
K in the number of results we get from RAG retrieval (top-K). This is different from @k.

This is our top-K (in this case top-5) retrieved documents (Top_{Kd}).

Retrieval Evaluations – Non-Rank Based

Precision@k

$$Precision@k = \frac{true\ positives@k}{true\ positives@k + false\ positives@k}$$



Relevant Documents (RD)

Let’s say we want to get precision at k=2.

$$Precision@2 = \frac{1}{1+1} = 0.5$$

Retrieval Evaluations – Non-Rank Based

∩ is for intersection

Recall@k

Recall@k examines how many relevant results have been retrieved from the total relevant results for the query.

$$Recall@k = \frac{|RD \cap Top_{kd}|}{|RD|}$$

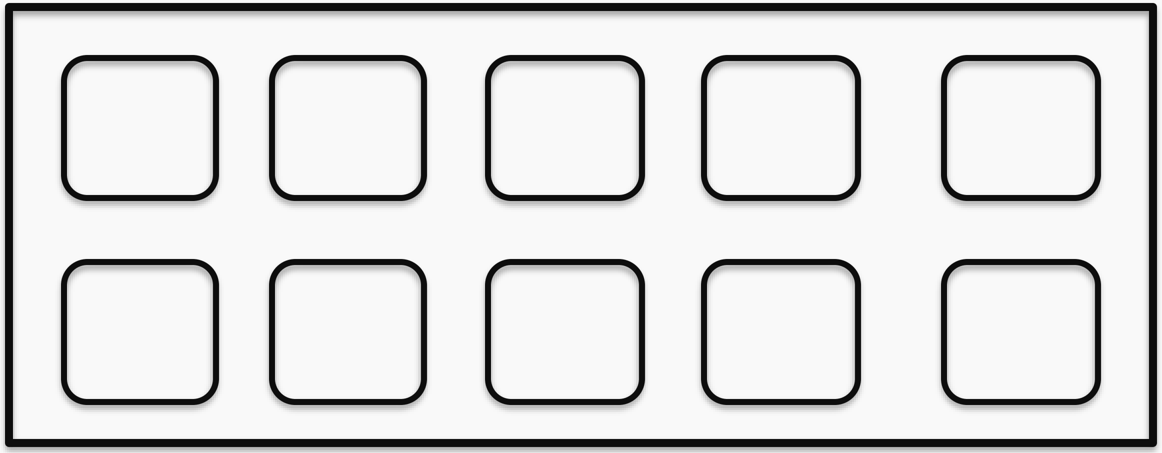
Top_{kd} is the top-k retrieved documents

Recall@k is ideal when capturing all relevant items in the result set is essential, even if this means including irrelevant ones.

Retrieval Evaluations – Non-Rank Based

Recall@k

$$Recall@k = \frac{|RD \cap Top_{kd}|}{|RD|}$$



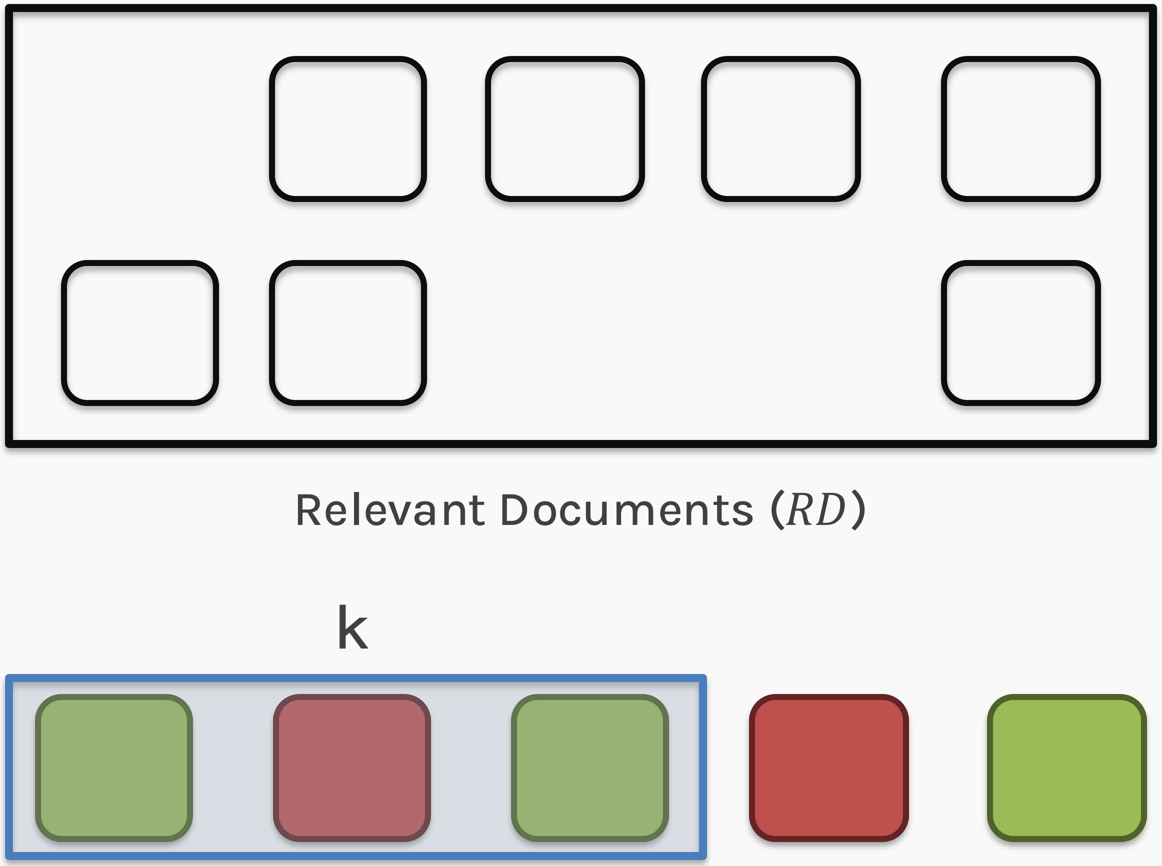
Relevant Documents (RD)

Let’s assume the same example as before.

Retrieval Evaluations – Non-Rank Based

Recall@k

$$Recall@k = \frac{|RD \cap Top_{kd}|}{|RD|}$$



Let's say we want to get the recall at k=3.

$$Recall@3 = \frac{2}{10} = 0.2$$

Retrieval Evaluations – Non-Rank Based

F1@k

F1@k combines both precision and recall into a single metric.

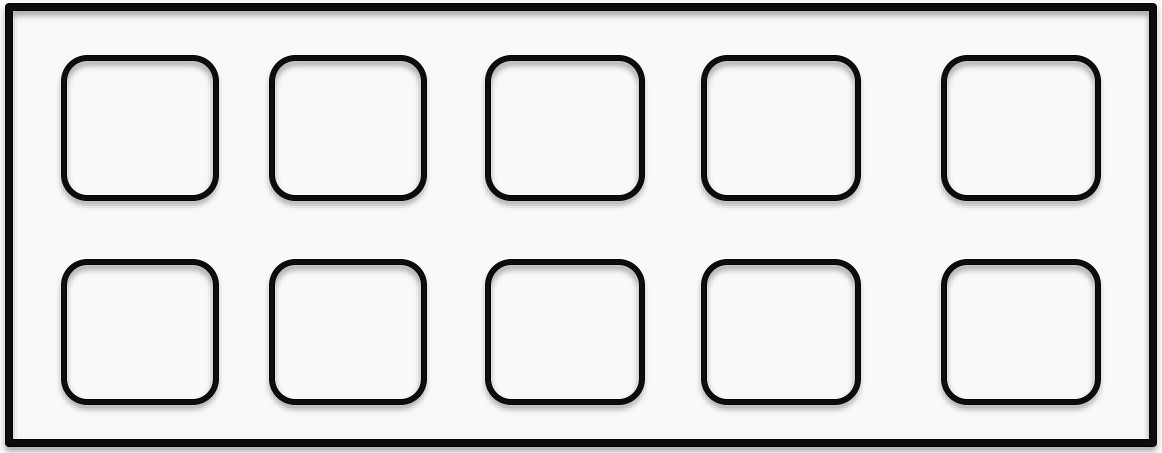
$$F1@k = \frac{2 \ * Precision@k \ * Recall@k}{Precision@k + Recall@k}$$

F1@k is ideal when we want to balance out retrieving all relevant items (recall) and ensuring they are applicable (precision).

Retrieval Evaluations – Non-Rank Based

F1@k

$$F1@k = \frac{2 \ * Precision@k \ * Recall@k}{Precision@k + Recall@k}$$

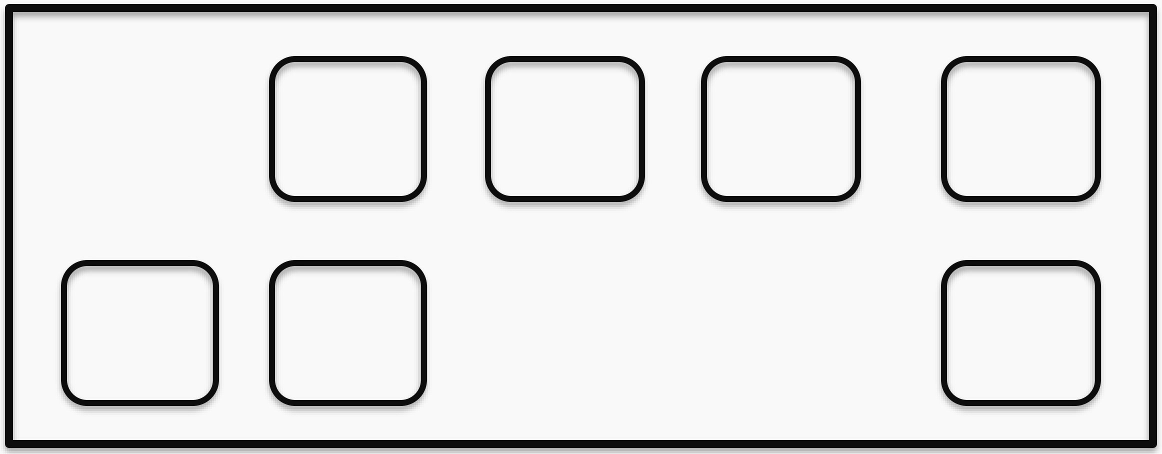


Relevant Documents (RD)

Retrieval Evaluations – Non-Rank Based

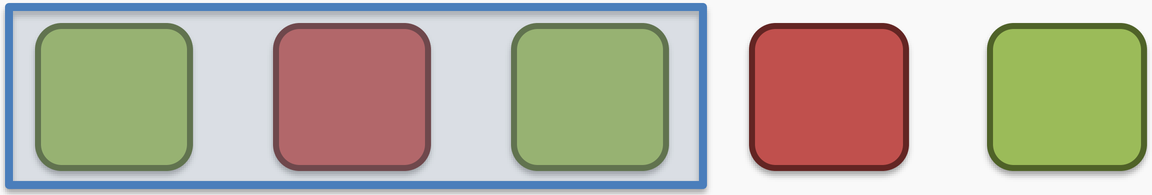
F1@k

$$F1@k = \frac{2 * Precision@k * Recall@k}{Precision@k + Recall@k}$$



Relevant Documents (RD)

k



Let's say we speak of F1 at k=3.

$$Recall@3 = \frac{2}{10} = 0.2$$

$$Precision@3 = \frac{1}{1+1} = 0.5$$

$$F1@3 = \frac{2 * 0.5 * 0.2}{0.5 + 0.2} = 0.29$$

Quick Summary:

Non-Rank based metrics examine if the results are relevant or irrelevant, regardless of the order they're in.

Let's now look at the 2nd type of evaluation metrics:

Rank Based Metrics

Unlike Non-Rank based, the order is considered in Rank based metrics!

Retrieval Evaluations - Rank Based

Rank-Based Metrics assess how well relevant items are ordered, with higher importance given to the positioning of relevant items at the ranking list.

We will be looking at 4 such metrics:

1. Mean Reciprocal Rank (MRR)
2. Mean Average Precision (MAP)
3. Discounted Cumulative Gain (DCG@k)
4. Normal Discounted Cumulative Gain (NDCG@k)

Retrieval Evaluations - Rank Based

Mean Reciprocal Rank (MRR)

MRR is a metric where the relevance of the top-ranked result is more important than the relevance of subsequent results.

|Q| is used to check the overall efficiency of the retriever. We need to check its performance for all the queries as compared to just one.

|Q| is the number of queries

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i}$$

$rank_i$ is the rank position of the 1st relevant document for the i-th query.

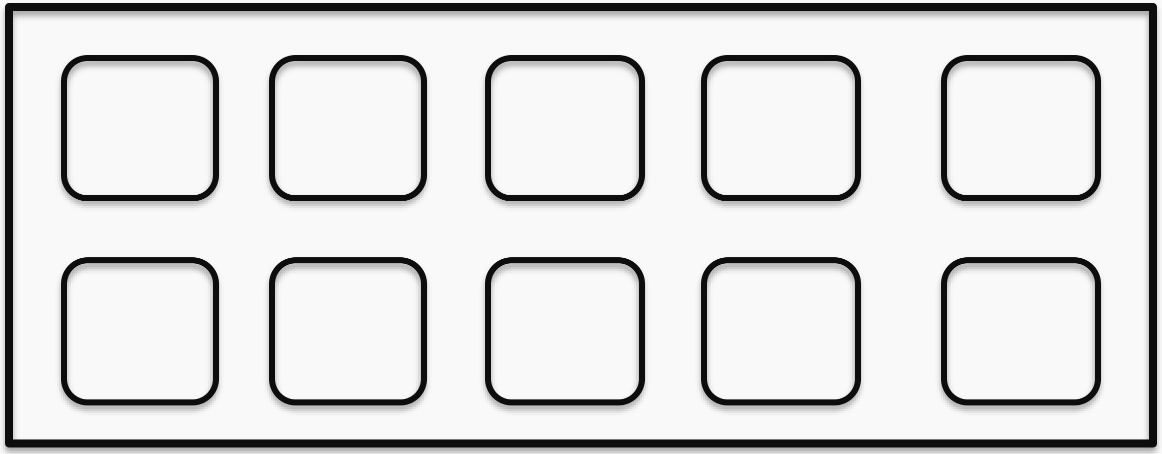
MRR ranges from **0 to 1**, where a higher value indicates better performance.

Retrieval Evaluations - Rank Based

Mean Reciprocal Rank (MRR)

$$MRR = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i}$$

MRR is ideal when the goal is to bring as many relevant items as possible close to the top of the results set.



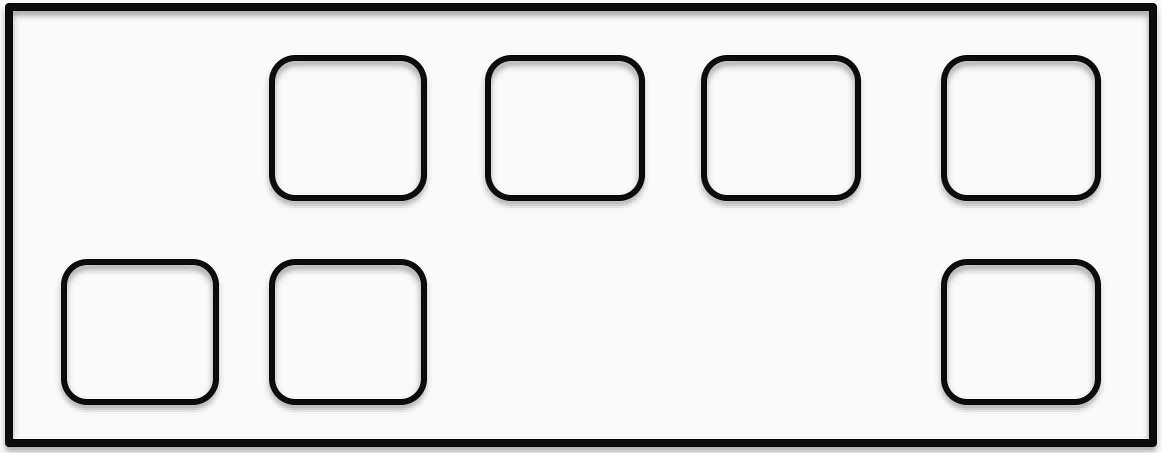
Relevant Documents (RD)

Let's take a look at an example to better understand this!

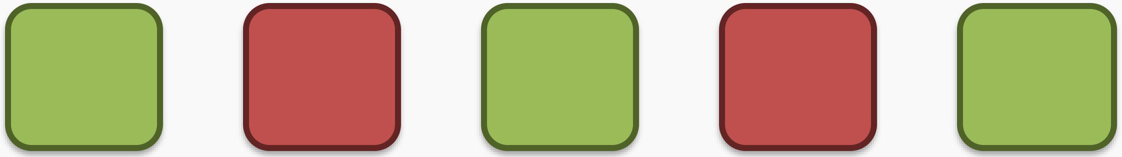
Retrieval Evaluations - Rank Based

Mean Reciprocal Rank (MRR)

$$MRR = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i}$$



Relevant Documents (RD)



For this example, our very first document is relevant.

The rank we will use associated to this query will be 1

In MRR, the number of relevant documents does not matter.

Retrieval Evaluations - Rank Based

$$MRR = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{rank_i}$$

Now, let's say for another query we get the following top-5 results:

The rank we will use associated to this query will be 2

$$MRR = \frac{1}{2}\sum_{i=1}^{|2|}\frac{1}{rank_i} = \frac{1}{2}\left(\frac{1}{1}+\frac{1}{2}\right) = \frac{3}{4}$$

Mean Average Precision (MAP)

MAP is a metric that assesses the quality of the results in a ranking system where order is important.

$$MAP = \frac{\sum_{k=1}^{n} P(k) * rel(k)}{number\ of\ elements\ in\ RD}$$

$P(k)$ is the Precision@k.

n is the number of retrieved documents.

$rel(k)$ is an indicator function equal to 1 if the item at rank k is relevant, 0 otherwise

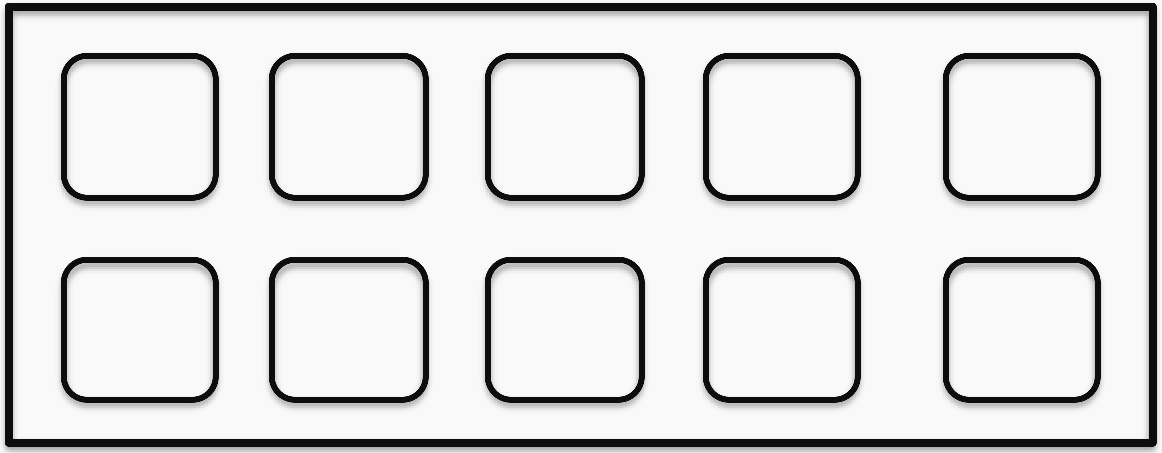
Retrieval Evaluations - Rank Based

Mean Average Precision (MAP)

$$MAP = \frac{\sum_{k=1}^{n} P(k) * rel(k)}{number\ of\ elements\ in\ RD}$$

Unlike MRR, that prioritizes position of the first relevant document, MAP considers all relevant results.

In this example, let's see how we use Precision@k values for each result to calculate the MAP.

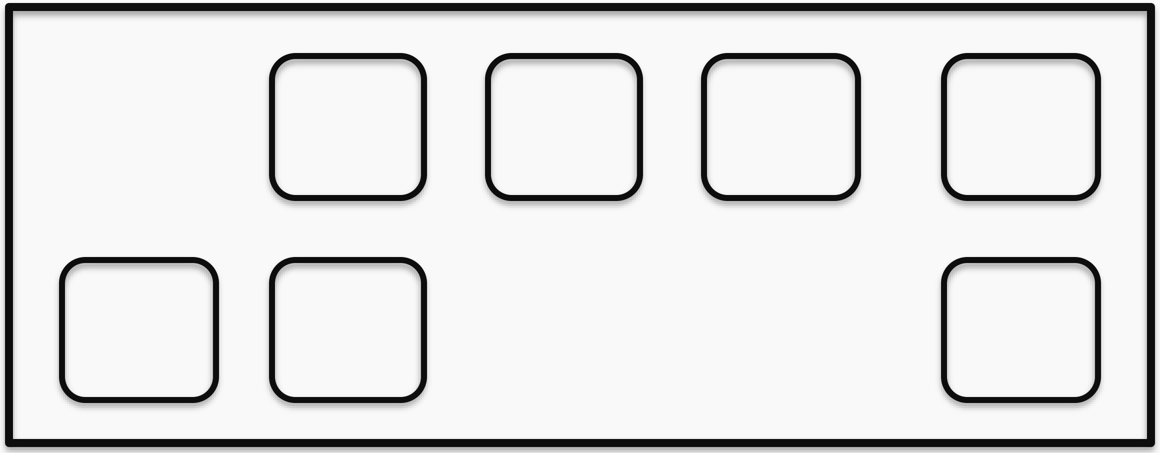


Relevant Documents (RD)

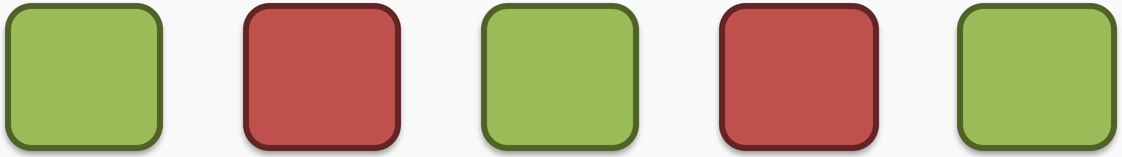
Retrieval Evaluations - Rank Based

Mean Average Precision (MAP)

$$MAP = \frac{\sum_{k=1}^{n} P(k) * rel(k)}{number\ of\ elements\ in\ RD}$$



Relevant Documents (RD)

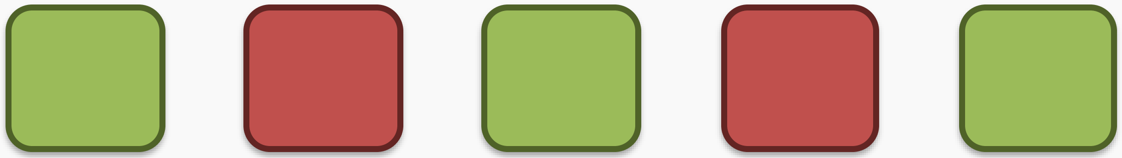


Precision@k $\frac{1}{1}$ $\frac{1}{2}$ $\frac{2}{3}$ $\frac{2}{4}$ $\frac{3}{5}$

Retrieval Evaluations - Rank Based

Mean Average Precision (MAP)

$$MAP = \frac{\sum_{k=1}^{n} P(k) * rel(k)}{number\ of\ elements\ in\ RD}$$



Precision@k $\frac{1}{1}$ $\frac{1}{2}$ $\frac{2}{3}$ $\frac{2}{4}$ $\frac{3}{5}$

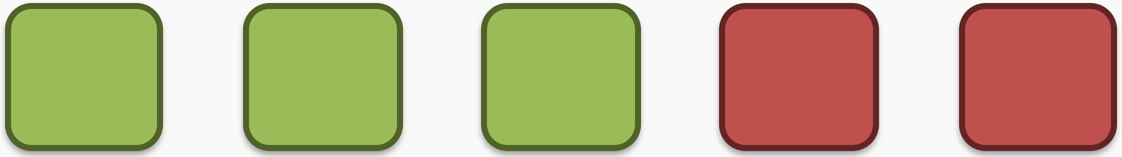
$$MAP = \frac{\sum_{k=1}^{n} P(k) * rel(k)}{number\ of\ elements\ in\ RD} = \frac{1 + \frac{2}{3} + \frac{3}{5}}{10} = 0.226$$

Retrieval Evaluations - Rank Based

Mean Average Precision (MAP)

$$MAP = \frac{\sum_{k=1}^{n} P(k) * rel(k)}{number\ of\ elements\ in\ RD}$$

Let's take another example where the results are higher in order.



Precision@k $\quad \frac{1}{1} \quad \frac{2}{2} \quad \frac{3}{3} \quad \frac{3}{4} \quad \frac{3}{5}$

$$MAP = \frac{\sum_{k=1}^{n} P(k) * rel(k)}{number\ of\ elements\ in\ RD} = \frac{1+1+1}{10} = 0.3$$

Retrieval Evaluations

The metrics that we have talked about deal with binary relevance: a result is either relevant or irrelevant.

What if we want to model shades of relevance? Where one result is extremely relevant, and another is less so?

A given result can be given a value ranging from 0 to 5 which we call relevancy scores, for example

How do we address this idea of relevancy scores?

Retrieval Evaluations

The metrics that we have talked about deal with binary relevance: a result is either relevant or irrelevant.

What if we want to model shades of relevance? Where one result is extremely relevant, and another is less so?

Graded Relevance Metrics address this for us!

A given res … example

1 2 3 4 5

How do we address this spectrum?

Retrieval Evaluations – Rank Based

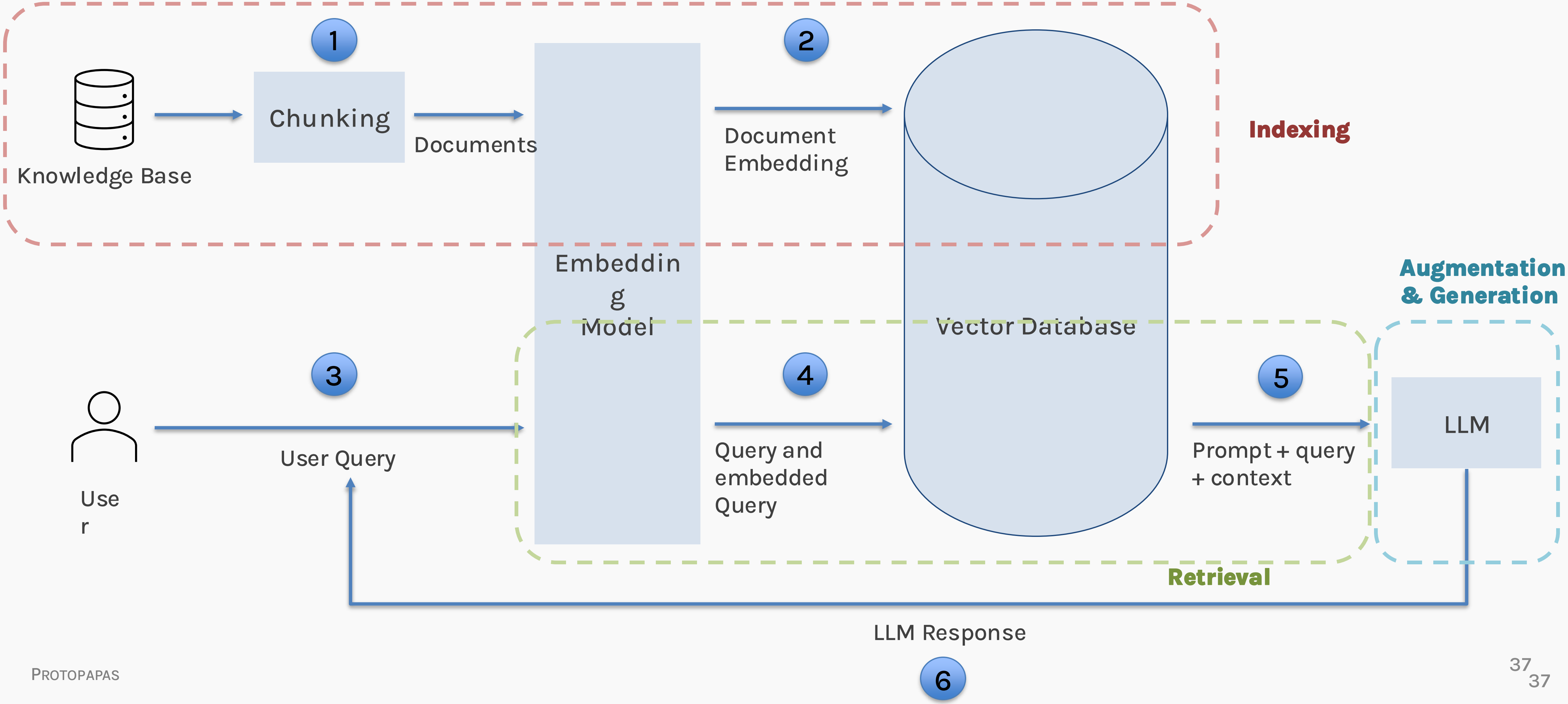
Graded relevance metrics comes under the Rank based evaluation metrics.

There are 2 metrics:

1. Discounted Cumulative Gain (DCG@k)
2. Normal Discounted Cumulative Gain (NDCG@k)

Think of graded relevance metrics as an alternate approach to what we have seen.

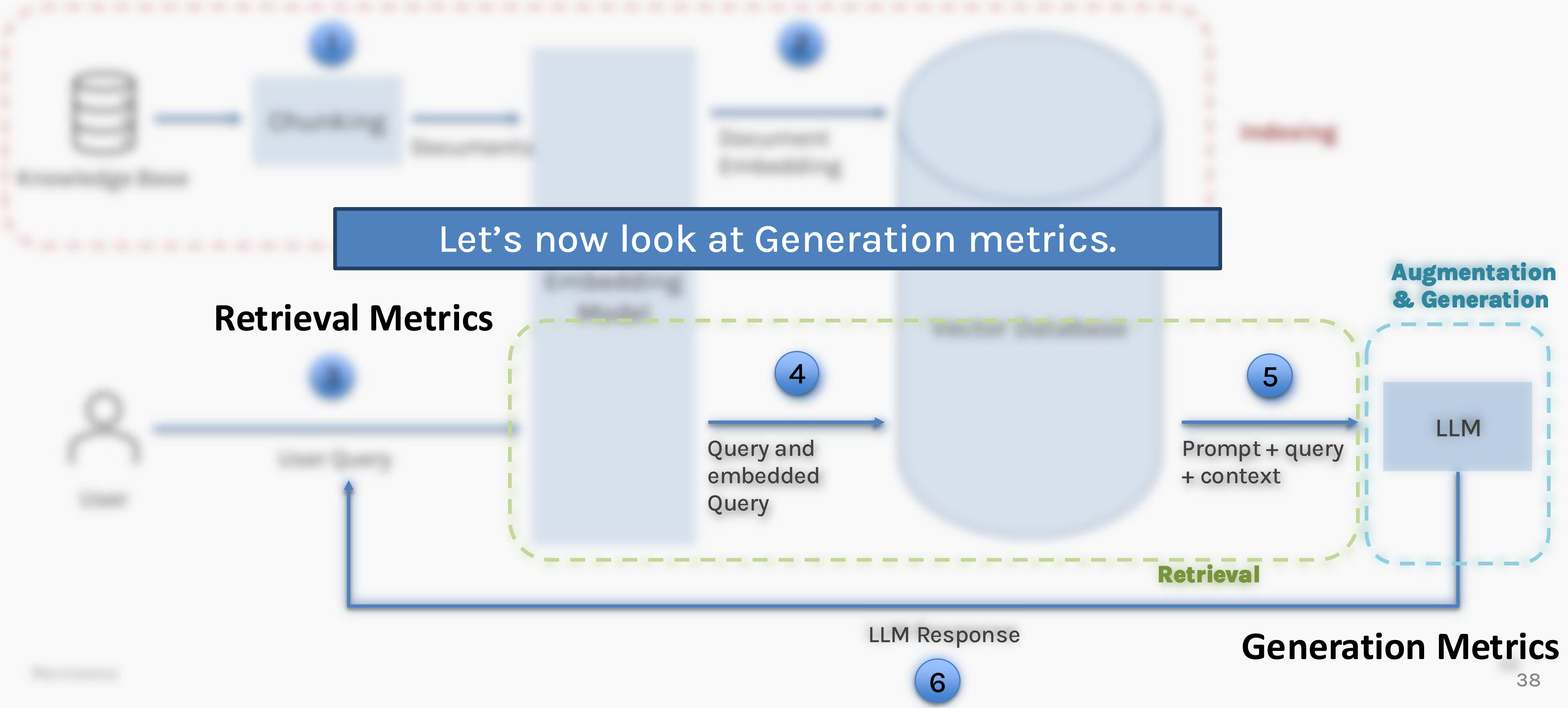
Naïve RAG


Indexing
1
Knowledge Base
Chunking
Documents
Embedding Model
2
Document Embedding
Vector Database
Augmentation & Generation
3
User
User Query
4
Query and embedded Query
5
Prompt + query + context
LLM
Retrieval
LLM Response
6

37

Naive RAG

Let's now look at Generation metrics.



Generation Evaluations

In the realm of generation, evaluation goes beyond accuracy of generated responses.

We need to consider the text's coherence, relevance, fluency, and alignment with human judgment.

Thus, the metrics needs to assess the factual correctness, readability and user satisfaction with the generated response.

Generation Evaluations

Some of the metrics that we can use for Generation Evaluations are:

1. ROUGE
2. BLEU
3. BertScore
4. LLM as a Judge

These metrics are not just used for RAG evaluation but also for standalone LLMs as well!

Generation Evaluations

ROUGE

- ROUGE - Recall-Oriented Understudy for Gisting Evaluation

- ROUGE is a set of metrics designed to evaluate the quality of machine-generated summaries by comparing them to human-generated reference summaries.

- ROUGE can be indicative of content overlap between generated text and the reference text.

Generation Evaluations

ROUGE

There are multiple variants of ROUGE, but here we will talk about just one, **ROUGE-N.**

Let's take a look at an example to understand what it is:

I gave Hunger Games 5 stars because I simply could not put it down! The characters are superb, and the writing keeps you on the edge of your seat. I loved reading the book.

Book Review

I really loved reading the Hunger Games

Machine generated summary

I loved reading the Hunger Games

The Hunger Games is a great read. I loved it.

Human reference summaries

ROUGE

ROUGE-N compares n-grams of the generated text with n-grams of the reference texts.

Let's look at **ROUGE-1,** which is based on unigram precision and recall to measure summarization quality.

I really loved reading the Hunger Games

Machine generated summary

I loved reading the Hunger Games

Human reference summaries

Here there are 6 unigram matches

Generation Evaluations

ROUGE – ROUGE-1

I really loved reading the Hunger Games

Machine generated summary

I loved reading the Hunger Games

Human reference summaries

$$ROUGE - 1\ Recall = \frac{Num\ unigram\ matches}{Num\ unigrams\ in\ reference} = \frac{6}{7}$$

$$ROUGE - 1\ Precision = \frac{Num\ unigram\ matches}{Num\ unigrams\ in\ summary} = \frac{6}{6}$$

We can do the same with bigrams to get ROUGE-2.

Generation Evaluations

ROUGE – ROUGE-2

I really

really loved

loved reading

reading the

the Hunger

Hunger Games

Machine generated summary

I loved

loved reading

reading the

the Hunger

Hunger Games

Human reference summaries

$$ROUGE-2\ Recall = \frac{Num\ bigram\ matches}{Num\ bigrams\ in\ reference} = \frac{4}{5}$$

$$ROUGE-2\ Precision = \frac{Num\ bigram\ matches}{Num\ bigrams\ in\ summary} = \frac{4}{6}$$

Generation Evaluations

BLEU

- BLEU - Bilingual Evaluation Understudy

- BLEU is a metric for evaluating the quality of machine-translated text against one or more reference translations.

- BLEU applies a brevity penalty to discourage overly short translations.

- BLEU has limitations, such as not accounting for the fluency or grammaticality of the generated text.

Generation Evaluations

BLEU

Let's take a look at an example to understand what it is:

Source Text (Spanish)	Machine generated translation	Human reference translations
Tengo trienta y sies años	I have thirty six years	I am thirty six years old
		I am thirty six years

BLEU compares the n-grams of the generation with the references.

Generation Evaluations

BLEU

BLEU-N compares n-grams of the generated translation with n-grams of the reference translations.

In **BLEU-1,** which is based on unigram precision to measure translation quality.

Here there are 4 unigram matches

I	have	thirty	six	years

Machine generated translations

$$BLEU - 1\ Precision = \frac{Num\ unigram\ matches}{Num\ unigrams\ in\ reference} = \frac{4}{5}$$

I	am	thirty	six	years	old

Human reference translations

Generation Evaluations

BLEU-N

A problem arises when a model over-generates ‘reasonable’ words.

six six six six six

Machine generated translations

I am thirty six years old

Human reference translations

Here, all the unigrams match, giving us a perfect score!

$$BLEU-1\ Precision = \frac{Num\ unigram\ matches}{Num\ unigrams\ in\ reference} = \frac{5}{5} = 1$$

What can we do?

BLEU-N

To handle this, BLEU uses a modified precision that clips the number of times to count a word based on the maximum occurrences of that word in the reference translation.

six | six | six | six | six

Machine generated translations

I | am | thirty | six | years | old

Human reference translations

$$BLEU - 1\ Precision = \frac{clip(Num\ unigram\ matches)}{Num\ unigrams\ in\ reference} = \frac{1}{5} = 0.2$$

Generation Evaluations

BLEU-N

Another problem that can come up is that it does not consider the order in which the words appear in the translation!

thirty six years have I

Machine generated translations

I am thirty six years old

Human reference translations

$$BLEU - 1\ Precision = \frac{clip(Num\ unigram\ matches)}{Num\ unigrams\ in\ reference} = \frac{4}{5} = 0.8$$

It still gives a high precision regardless of the order of the words.

BLEU solves this problem by computing the precision of several different n-grams and then averages the results.

Generation Evaluations

BLEU-N

Let’s take a look at an example where we take 4-grams.

thirty six years have I

thirty six years have I

Machine generated translations

I am thirty six years old

I am thirty six years old

I am thirty six years old

Human reference translations

$$BLEU-4\ Precision = \frac{clip(Num\ unigram\ matches)}{Num\ unigrams\ in\ reference} = \frac{0}{5} = 0$$

Generation Evaluations

BertScore

- BertScore uses the contextual embedding from BERT to check semantic similarity between generated and reference text.

- BertScore computes token-level similarity and produces precision, recall, and F1 scores.

- Unlike n-gram-based metrics, BertScore captures the meaning of words in context.

Generation Evaluations

BertScore

This makes BertScore more robust to paraphrasing and more sensitive to semantic equivalence.

Generation Evaluations

LLM as a Judge

- LLMs are used to score the generated text based on coherence, relevance, and fluency.
- The LLM can be fine-tuned on human judgments or used in zero-shot/few-shot settings to evaluate unseen text quality.
- This approach leverages the LLM's understanding of language and context to provide a more nuanced text quality assessment.

Generation Evaluations

LLM as a Judge

- Proving LLM judges with detailed scoring guidelines, such as a scale from 1 to 5, can standardize the evaluation process.
- This methodology encompasses critical aspects of content assessment like:
  - Coherence, relevance, fluency, coverage, diversity, detail.
- These aspects are considered in the context of answer evaluation and query formulation.

Evaluations - Libraries

Like everything in Python, we have libraries for evaluations as well.

	LangChain	LlamaIndex	Deepeval	ragas
Retrieval Evaluation Metric	MRR, Recall, Precision, Context Relevancy			Context Recall, Context Relevancy
Generation Evaluation Metric	LLM as a Judge, Conciseness, Correctness	Correctness, Faithfulness, Guidelines, Pairwise, LLM as a judge	Factual Consistency, Answer Relevancy, LLM as a judge	Faithfulness, Answer Relevancy, LLM as a judge

Thank you

Retrieval Evaluations – Rank Based

Discounted Cumulative Gain (DCG@k)

DCG is an order-aware metric that measures an item's usefulness based on its order in the result set.

It incorporates a logarithmic penalty to diminish the value of items that are lower in the order.

This leads to the intuition that top results are most valuable.

$$DCG@k = \sum_{i=1}^{k} \frac{rel_i}{\log_2(i+1)}$$

rel_i is the relevancy score (1-5).

Retrieval Evaluations – Rank Based

Discounted Cumulative Gain (DCG@k)

$$DCG@k = \sum_{i=1}^{k} \frac{rel_i}{\log_2(i+1)}$$

1 2 3 4 5

Here's what the growing penalty looks like for our above example:

i	Calculation	Penalty
1	$\log_2(1+1)$	1
2	$\log_2(2+1)$	1.584
3	$\log_2(3+1)$	2
4	$\log_2(4+1)$	2.321
5	$\log_2(5+1)$	2.584

The further a relevant result is from the top of the set, higher is the penalty.

Retrieval Evaluations – Rank Based

Discounted Cumulative Gain (DCG@k)

$$DCG@k = \sum_{i=1}^{k} \frac{rel_i}{\log_2(i+1)}$$

But there is a key drawback in DCG@K

i	Calculation	Penalty
1	$\log_2(1+1)$	1
2	$\log_2(2+1)$	1.584
3	$\log_2(3+1)$	2
4	$\log_2(4+1)$	2.321
5	$\log_2(5+1)$	2.584

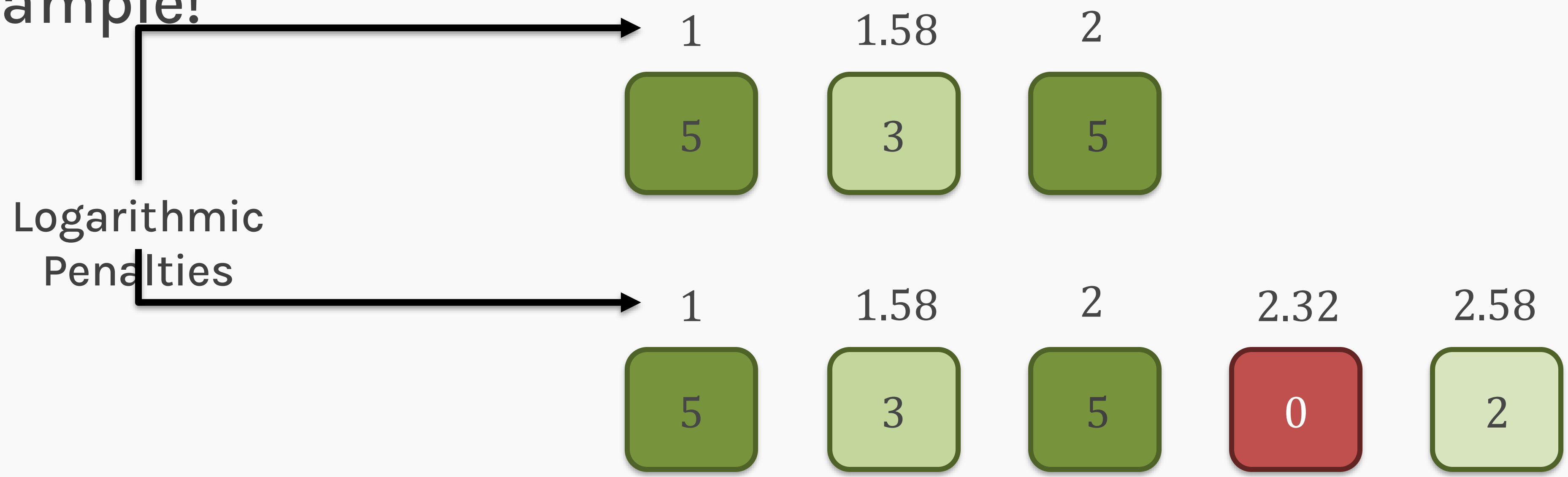
The further a relevant result is from the top of the set, higher is the penalty

Retrieval Evaluations – Rank Based

Discounted Cumulative Gain (DCG@k)

DCG@k does not take into account the varying lengths of result sets and thus naturally favours longer sets!

Let's look at an example!



Retrieval Evaluations – Rank Based

Discounted Cumulative Gain (DCG@k)

1	1.58	2
5	3	5

$$DCG@3 = \frac{5}{1} + \frac{3}{1.58} + \frac{5}{2} = 9.40$$

1	1.58	2	2.32	2.58
5	3	5	0	2

$$DCG@5 = \frac{5}{1} + \frac{3}{1.58} + \frac{5}{2} + \frac{0}{2.32} + \frac{2}{2.58} = 10.17$$

Even though the second set was not significantly more relevant, it received a higher score, due to its length!

Retrieval Evaluations – Rank Based

Discounted Cumulative Gain (DCG@k)

$$DCG@3 = \frac{5}{1} + \frac{3}{1.58} + \frac{5}{2} = 9.40$$

Normalized Discounted Cumulative Gain addresses this limitation.

Even though the second set was not significantly more relevant, it received a higher score, due to its length!

Retrieval Evaluations – Rank Based

Normalized Discounted Cumulative Gain (NDCG@k)

NDCG is the ratio of DCG TO IDCG which thus provides us normalized scores that remove the problem.

$$NDCG@k = \frac{DCG@k}{IDCG@k}$$

The Ideal Discounted Cumulative Gain (IDCG) score is the DCG score that we attain after sorting the order of items based on relevance.

Retrieval Evaluations – Rank Based

Normalized Discounted Cumulative Gain (NDCG@k)

$$NDCG@k = \frac{DCG@k}{IDCG@k}$$

Let's revisit the example that we used in DCG@k:

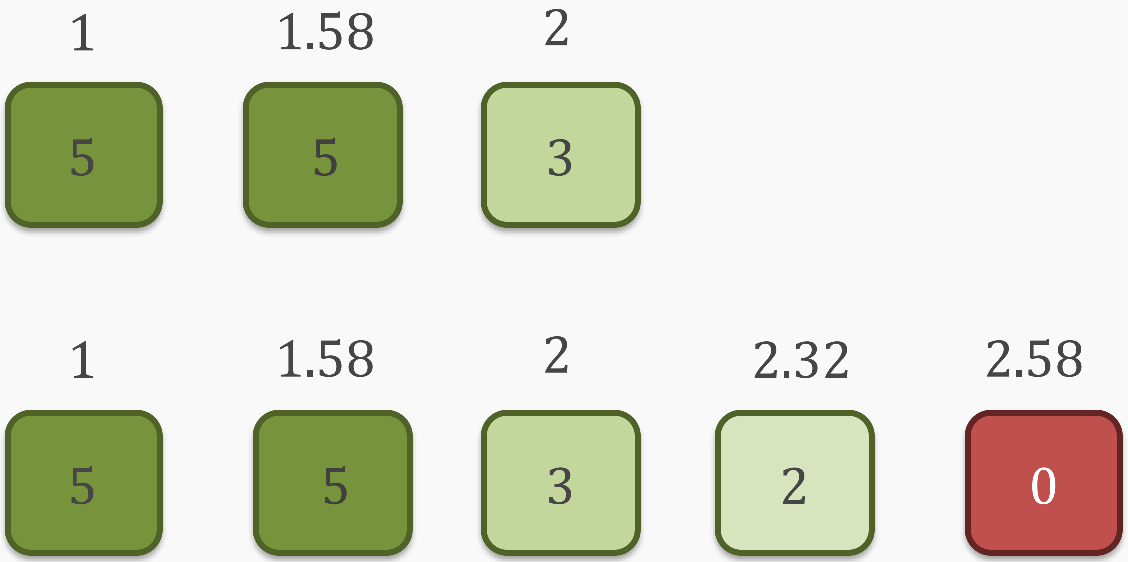
1	1.58	2
5	3	5

1	1.58	2	2.32	2.58
5	3	5	0	2

Before we calculate NDCG@k, we will need to sort by relevance to calculate IDCG@k

Normalized Discounted Cumulative Gain (NDCG@k)

$$NDCG@k = \frac{DCG@k}{IDCG@k}$$



As the first 3 documents are the same for both examples, let us keep the 2nd to make it easier for us to follow.

Retrieval Evaluations – Rank Based

Normalized Discounted Cumulative Gain (NDCG@k)

$$NDCG@k = \frac{DCG@k}{IDCG@k}$$

1	1.58	2	2.32	2.58
5	5	3	2	0

Position	Relevance	$\log_2(i+1)$	$\frac{rel(i)}{\log_2(i+1)}$	IDCG@k
1	5	$\log_2(1+1) = 1$	$\frac{5}{1} = 5$	5
2	5	$\log_2(2+1) = 1.585$	$\frac{5}{1.585} = 3.154$	5 + 3.154 = 8.154
3	3	$\log_2(3+1) = 1.5$	$\frac{3}{2} = 1.5$	5 + 3.154 + 1.5 = 9.654
4	2	$\log_2(4+1) = 0.861$	$\frac{2}{2.3219} = 0.861$	5 + 3.154 + 1.5 + 0.861 = 10.515
5	0	$\log_2(5+1) = 2.585$	$\frac{0}{2.585} = 0$	5 + 3.154 + 1.5 + 0.861 + 0 = 10.515

Retrieval Evaluations – Rank Based

Normalized Discounted Cumulative Gain (NDCG@k)

$$NDCG@k = \frac{DCG@k}{IDCG@k}$$

Position	Relevance	$\log_2(i+1)$	$\frac{rel(i)}{\log_2(i+1)}$	IDCG@k
1	5	$\log_2(1+1) = 1$	$\frac{5}{1} = 5$	5
2	5	$\log_2(2+1) = 1.585$	$\frac{5}{1.585} = 3.154$	5 + 3.154 = 8.154
3	3	$\log_2(3+1) = 1.5$	$\frac{3}{2} = 1.5$	5 + 3.154 + 1.5 = 9.654
4	2	$\log_2(4+1) = 0.861$	$\frac{2}{2.3219} = 0.861$	5 + 3.154 + 1.5 + 0.861 = 10.515
5	0	$\log_2(5+1) = 2.585$	$\frac{0}{2.585} = 0$	5 + 3.154 + 1.5 + 0.861 + 0 = 10.515

Thus, IDGC@3 = 9.654 and IDGC@5 = 10.515

Retrieval Evaluations – Rank Based

Normalized Discounted Cumulative Gain (NDCG@k)

$$NDCG@k = \frac{DCG@k}{IDCG@k}$$

IDGC@3 = 9.654

1	1.58	2
5	5	3

$$DCG@3 \ = 9.40$$

IDGC@5 = 10.515

1	1.58	2	2.32	2.58
5	5	3	2	0

$$DCG@5 \ = 10.17$$

We can now obtain the NDCG@k

$$NDCG@3 \ = \frac{9.40}{9.654} = 0.973$$

$$NDCG@5 \ = \frac{10.17}{10.515} = 0.967$$

Retrieval Evaluations – Rank Based

Normalized Discounted Cumulative Gain (NDCG@k)

$$NDCG@k = \frac{DCG@k}{IDCG@k}$$

1	1.58	2
5	5	3

1	1.58	2	2.32	2.58
5	5	3	2	0

$$NDCG@3 = \frac{9.40}{9.654} = 0.973$$

$$NDCG@5 = \frac{10.17}{10.515} = 0.967$$

Normalized DCG scores range from 0 to 1, allowing fair comparison of query quality regardless of result set length.

Retrieval Evaluations – Rank Based

Normalized Discounted Cumulative Gain (NDCG@k)

$$NDCG@k = \frac{DCG@k}{IDCG@k}$$

$$NDCG@3 = \frac{9.40}{9.654} = 0.973$$

$$NDCG@5 = \frac{10.17}{10.515} = 0.967$$

There are 2 reasons as to why the NDCG@5 is lesser than NDCG@3:

- We have sorted the documents based on relevancy.
- NDCG@5 had more lesser relevant documents and thus was penalized more, reducing its overall NDCG value.

THANK YOU!