🧠 Evaluation Metrics for LLMs

LLMs are evaluated using automatic (intrinsic) metrics and human (extrinsic) metrics depending on task type — such as text generation, retrieval-augmented generation (RAG), reasoning, or coding.

1. Intrinsic (Automatic) Metrics

These are computed automatically by comparing model outputs against reference data or internal probabilities.

A. Overlap & Matching Metrics

Measure the lexical overlap between the model output ("candidate") and the human-written "reference."

Metric	Focus	What it Measures	Common Use
ROUGE (Recall-Oriented Understudy for Gisting Evaluation)	Recall	How much of the reference content appears in the generated text	Summarization, QA
ROUGE-N	N-gram overlap	Measures recall of N-gram sequences (ROUGE-1 for unigrams, ROUGE-2 for bigrams)	Fluency and content recall
ROUGE-L	Longest Common Subsequence	Captures how well the main ideas are preserved in correct order	Abstractive summarization
BLEU (Bilingual Evaluation Understudy)	Precision	How much of the system output overlaps with the reference	Machine translation, seq2seq models

🧱 Example: BLEU Score Calculation

Reference: "The cat is on the mat."

Candidate: "The cat is on mat."

Compute:

- 1-gram precision: 4/5 = 0.8
- 2-gram precision: 3/4 = 0.75
- Brevity penalty (since candidate is shorter):

BP = $e^{1-\frac{r}{c}} = e^{1-\frac{6}{5}} = e^{-0.2} \approx 0.82$

Final BLEU = $BP \times \exp(\text{avg log precision}) = 0.82 \times \exp(\frac{\ln(0.8)+\ln(0.75)}{2}) = 0.82 \times 0.77 \approx 0.63$

✅ BLEU = 0.63 (moderate translation quality).

B. Embedding & Semantic Similarity Metrics

Instead of word overlap, these use embeddings to measure semantic equivalence.

Metric	Focus	What it Measures	Common Use
BERTScore	Contextual Similarity	Cosine similarity between token embeddings of candidate and reference using pretrained BERT	Paraphrasing, open-ended generation
METEOR	Synonymy	Uses stemming, synonyms, and paraphrases to match meaning beyond surface form	Machine translation, summarization

🧱 Example: BERTScore

Given candidate “A man is playing a guitar” and reference “A person plays the guitar.”

- Token embeddings from BERT are compared using cosine similarity.
- Suppose average similarity = 0.94 → BERTScore = 0.94, much higher than BLEU/RO

C. Perplexity (PPL)

Perplexity is a measure of how well a probability model predicts a sample. In the context of LLMs, it essentially tell you how "surprised" the model is by a piece of text. It is calcualted as the exponentiated average of the negative log-likelihood of the test data (N words):

$$\text{PPL} = 2^{-\frac{1}{N}\sum_{i=1}^{N}\log_2 P(w_i|w_{1\ldots i-1})}.$$

Interpretation:

- Low PPL (Good) → The model is not surprised. It assigns high probability to the words it is seeing, meaning the text is highly probable and fluent according to the model's training.
- High PPL (Bad) → The model is highly surprised. It assigns low probability to the words, meaning the text is incoherent or random according to its learned patterns.

Limitation:

> PPL is primarily a measure of fluency and grammatical correctness, but it does not tell you if the generated text is factually correct or relevant to the user's prompt (it is not a great metric for comparing models trained on different datasets).

🧱 Example

Sentence: "I love machine learning."

Log probabilities = [–0.2, –0.3, –0.1, –0.5]

$$\text{PPL} = 2^{-(-1.1)/4} = 2^{0.275} = 1.21$$

✅ Low perplexity = fluent generation.

D. Diversity and Creativity

Metric	Focus	What it Measures
Distinct-N	Novelty	Ratio of unique N-grams to total N-grams (Distinct-1 for words, Distinct-2 for pairs). Higher = more diverse.
Self-BLEU	Repetition	BLEU of one sample vs. others from the same model; lower = more varied generation.

Generated 3 sentences:

```
1.  "I love dogs."
2.  "I love cats."
3.  "I love animals."
```

Distinct-1 = 6 unique unigrams / 9 total = 0.67

Distinct-2 = 5 unique bigrams / 6 total = 0.83

✅ High diversity.

2. Extrinsic (Human-based) Metrics

Automatic scores can miss context, coherence, and truthfulness. Human evaluation captures subjective and holistic qualities.

Metric	Focus	What it Measures
Factuality/Hallucination Rate	Truthfulness	% of false or unsupported statements
Relevance/Helpfulness	Usefulness	How well the response addresses the query
Preference(A/B Testing)	Quality Comparison	Human raters pick the better output
Groundedness	Source faithfulness	In RAG, % of statements traceable to context docs

LLM-as-a-Judge

Use a stronger LLM (e.g., GPT-4) to score another model's output.

A common trend is to use a strong, proprietary LLM (like GPT-4 or Claude) to evaluate the output of a smaller, target LLM. The judge LLM is given the user's prompt, the reference answer (optional), and the generated answer, then asked to rate the quality based on criteria like coherence, helpfulness, and conciseness.

```
• Scales human-like judgment.
• Correlates strongly with human preferences.
• Common in evaluations like MT-Bench or AlpacaEval.
```

3. Retrieval and Ranking Metrics (for RAG)

Measure retrieval quality before generation.

Metric	Focus	What it Measures
Precision@k	Accuracy	Fraction of retrieved docs that are relevant
Recall@k	Coverage	Fraction of relevant docs retrieved
F1@k	Balance	Harmonic mean of precision and recall
Rank@k	Hit rate	Fraction of queries with a correct item in top-k

🧱 Example

Top-5 retrieved docs: 3 relevant.

Total relevant = 4.

$$\text{Precision@5} = 3/5 = 0.6, \quad \text{Recall@5} = 3/4 = 0.75$$

$$F1@5 = \frac{2 \times 0.6 \times 0.75}{0.6 + 0.75} = 0.67$$

Mean Reciprocal Rank (MRR)

MRR is ideal when there is only one single, correct answer (or a single best answer) for a query.

Focus: First Correct Hit

What it Measures: The reciprocal of the rank of the first correct answer found.

Interpretation: An MRR of 1.0 means the first result was always the correct one. An MRR of 0.5 means the first correct answer was, on average, at rank 2.

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i}$$

If correct answer appears at rank 1, 3, and 2 for three queries:

MRR = $\frac{1}{3}(1 + 1/3 + 1/2) = 0.61$

✅ Higher = better early ranking.

Normalized Discounted Cumulative Gain (NDCG)

NDCG is the most comprehensive ranking metric, often used when relevance is graded (e.g., highly relevant, somewhat relevant, irrelevant).

What it Measures: It sums the Gain (relevance score) of each result, discounts it logarithmically by its position (rank), and normalizes the result against a perfect ranking (Ideal DCG).

Interpretation: This is the gold standard for measuring the utility of the retrieved list. Placing highly relevant items at rank 1 is rewarded much more than placing them at rank 10.

$$\text{DCG@}k = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_2(i+1)}$$

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

🧱 Example

Relevance scores = [3, 2, 0] →

$$DCG = \frac{2^3 - 1}{\log_2 2} + \frac{2^2 - 1}{\log_2 3} = 7 + 1.89 = 8.89$$

If perfect ranking = [3,2,0] → IDCG = 8.89 → NDCG = 1.0 (perfect).

4. Task-Specific Evaluation

A. Question Answering

The goal is to generate an accurate and concise answer based on a prompt, often involving retrieval (Retrieval-Augmented Generation, RAG).

Metric	Focus	Why it's Used
Exact Match (EM)	Accuracy	Exact string match with reference
F1 Score	Overlap	Harmonic mean of precision/recall over words
Groundedness/Attribution	Source faithfulness	% of generated facts traceable to retrieved docs

🧱 Example

Reference: “Paris is the capital of France.”

Model: “The capital of France is Paris.”

EM = 0 (not identical); F1 ≈ 1.0 (all words overlap). ✅ High semantic accuracy.

B. Code Generation

Metric	Focus	What it Measures
Pass@k	Functional correctness	Probability ≥1 of k generated codes passes unit tests
Execution Accuracy	Test success	% of code runs producing correct output

🧱 Example

Out of 10 problems, at least one of 3 code completions works for 7 problems →
$\mathrm{Pass@3}$ = 7/10 = 0.7

C. Dialogue / Conversational Systems

Metric	Focus	What it Measures
Coherence / Flow	Logical quality	Consistency with conversation history
Engagingness	User experience	Human rating of interest level
Distinct-N	Diversity	Avoiding repetitive responses
Safety / Toxicity	Harm control	% of unsafe outputs

5. Model Lifecycle Evaluation

A. Pre-training Stage (Building the Foundation Model)

This stage focuses on scaling up the model's fundamental ability to predict the next token based on a massive corpus of general text data. The goal is to achieve broad knowledge and linguistic fluency.

Metric	Focus	Why it's Used
Perplexity (PPL)	Fluency	Main optimization metric for next-token prediction
Training Loss	Learning progress	Cross-entropy minimized by optimizer
Token/Second	Efficiency	ETraining throughput metric
Zero-Shot/Few-Shot Accuracy	Emergent skills	Early capability probe

B. Fine-tuning Stage (Task/Domain Adaptation)

After pre-training, the model is refined on a smaller, high-quality dataset specific to a task (e.g., question answering, summarization, or dialogue).

Metric	Focus	Why it's Used
Task Accuracy/F1	For classification and QA	
ROUGE/BLEU	Overlap	For summarization or translation
Validation Loss / PPL	Generalization	Prevent overfitting

C. Post-training Stage(Alignment & Safety)

This is the final stage, often involving Reinforcement Learning from Human Feedback (RLHF) or similar techniques to align the LLM with human preferences, safety standards, and

instruction-following abilities.

Metric	Focus	Why it's Used
Human Preference Rate	Alignment	% preferred by raters
Safety Violation Rate	Safety	% toxic / harmful outputs
Factuality/Hallucination Rate	Trustworthiness	% verifiably correct claims
LLM-as-a-Judge Score	Scalable Quality	GPT-4/Claude judge smaller model
Benchmark Scores	Capability Testing	Finalized scores on industry-standard datasets like MMLU (knowledge), GSM8K (reasoning), HumanEval (coding), etc., to compare the model's final, versatile capabilities against competitors.

✅ Summary: Metric Categories by Task

Stage	Key Metric	Interpretation
Pretraining	PPL ↓	Fluency, next-token prediction
Fine-tuning	ROUGE / BLEU / F1	Task-specific overlap
Alignment	Human Preference ↑	Instruction-following, safety
RAG	Recall@k, NDCG@k ↑	Retrieval quality
Generation	BERTScore ↑, Distinct-N ↑	Meaning and creativity
Code	Pass@k ↑	Functional correctness

Questions

Q1. What is the fundamental difference between intrinsic and extrinsic metrics when evaluating LLMs?

Intrinsic metrics are automatic and token-based, such as ROUGE, BLEU, or Perplexity — they measure linguistic similarity or fluency.

Extrinsic metrics rely on human or human-like judgment, such as preference, factuality, or helpfulness.

Intrinsic metrics are objective and fast, while extrinsic ones better capture real-world usefulness and alignment.

Q2. How does ROUGE differ from BLEU in what they emphasize, and when would you prefer one over the other?

BLEU measures precision — how much of the system output matches the reference — and is mainly used in translation.

ROUGE measures recall — how much of the reference is captured by the output — and is more suitable for summarization or abstractive tasks.

So BLEU is about how exact the match is; ROUGE is about how complete the coverage is.

Q3. Why is perplexity (PPL) not always a reliable metric for comparing two large language models trained on different datasets?

Perplexity reflects how well a model predicts the token distribution of a given dataset.

If two models are trained on different corpora or tokenizers, their probability scales differ, making PPL values incomparable.

It’s best used for tracking learning progress within the same model and dataset.

Q4. Suppose an LLM generates fluent but factually incorrect text in a RAG system. Which metrics would capture this issue, and how?

We’d use Groundedness and Factuality rate.

Groundedness measures whether each statement is supported by retrieved documents, and factuality rate quantifies how many claims are true.

These metrics detect hallucinations that fluency-based scores like ROUGE would miss.

Q5. What problem does BERTScore solve compared to ROUGE/BLEU, and how does it compute similarity?

ROUGE and BLEU only count exact word overlap, missing semantic similarity.

BERTScore uses contextual embeddings from models like BERT to compute cosine similarity between tokens, capturing synonyms and paraphrases.

It aligns better with human judgment when wording varies but meaning stays consistent.

Q6. How does NDCG differ from Recall@k in evaluating retrieval quality, and why is it preferred in RAG systems?

Recall@k only checks whether relevant items appear in the top-k results, ignoring their rank.

NDCG weights each result by both relevance and position — highly relevant documents near the top contribute more.

That’s why it better reflects retrieval quality in RAG, where ranking order matters.

Q7. How can human evaluation and LLM-as-a-Judge complement each other in assessing LLM quality?

Human evaluation is the gold standard for nuanced judgments but is expensive and slow.

LLM-as-a-Judge, using a strong model like GPT-4, provides scalable, consistent scoring that often correlates well with human ratings.

Together, they combine scalability with reliability for large-scale evaluation.

Q8. How would you evaluate the diversity and creativity of a generative LLM beyond accuracy metrics?

We use Distinct-N and Self-BLEU.

Distinct-N measures the proportion of unique n-grams, encouraging novelty, while Self-BLEU measures similarity among outputs — lower is better.

Together they quantify whether the model generates varied, non-repetitive responses.

Q9. Why might two LLMs with similar ROUGE scores have very different human preference ratings?

ROUGE captures surface overlap, not deeper qualities like clarity, factuality, or coherence.

Two outputs can use similar words but differ in reasoning quality or tone.

Human or LLM-judge evaluations capture those higher-level aspects that ROUGE ignores.

Q10. In fine-tuning or RLHF, how are evaluation metrics like preference rate and factuality rate used to improve alignment?

Preference rate comes from human or model comparisons and trains a reward model to favor preferred responses.

Factuality or safety rates can act as penalties or constraints in the reinforcement objective.

These guide the model to generate outputs that are not just fluent, but also helpful, truthful, and safe.

Q11. Explain why high BLEU/ROUGE scores do not guarantee high NDCG or MRR in a retrieval-augmented generation pipeline.

BLEU and ROUGE evaluate text generation quality, while NDCG and MRR evaluate retrieval ranking.

A generator can produce fluent answers even if the retriever surfaces poor documents — it may just hallucinate.

So strong generation metrics don’t imply strong retrieval performance; both need separate evaluation.

Q12. Suppose a model has BLEU = 0.45 and ROUGE-L = 0.72. What does this discrepancy tell you?

BLEU measures precision (how much of the generated text appears in the reference), while ROUGE-L emphasizes recall (how much of the reference appears in the generated text).

A higher ROUGE-L but lower BLEU suggests the model captures the main ideas (good coverage) but adds extra or paraphrased words that reduce exact overlap.

To improve BLEU, the model should generate more concise and lexically aligned outputs — for example, by constraining decoding (smaller beam width or lower temperature).

Q13. How would you calculate ROUGE-1 recall and precision for this example?

Reference: “The cat sat on the mat.”

Generated: “A cat is sitting on the mat.”

- Overlapping unigrams: {cat, on, the, mat} → 4 overlaps
- Reference unigrams: 6 → Recall = 4/6 = 0.67
- Generated unigrams: 6 → Precision = 4/6 = 0.67
- F1 = 2 × (0.67 × 0.67)/(0.67 + 0.67) = 0.67

The model matches most key words, so high ROUGE-1 indicates semantic alignment, but mismatched morphology (“is sitting”) slightly hurts ROUGE-2 or BLEU.

Q14.If Perplexity (PPL) increases during validation, what might be happening?

Increasing PPL means the model is assigning lower probability to real tokens → it’s becoming more “surprised.”

This usually signals overfitting (training loss ↓ but validation PPL ↑).

To improve:

```
• Add regularization (dropout, weight decay).
• Use early stopping or a smaller learning rate.
• Increase training data diversity.
```

Q15. Model A has PPL = 10, Model B = 20, but Model B achieves higher human ratings. Why?

Perplexity reflects fluency and predictability, not factual accuracy or helpfulness.

Model A might generate safe, generic text ("I don't know"), leading to low PPL but low usefulness.

Model B may produce diverse, context-specific answers that slightly hurt token likelihood but improve user satisfaction — showing PPL doesn't correlate with preference beyond fluency.

Q16. If a model's Self-BLEU is high (e.g., 0.95), what does it indicate, and how can you fix it?

A high Self-BLEU means different outputs are highly similar → low diversity or repetitive patterns.
To improve:

```
• Increase decoding randomness (temperature, top-p).
• Use nucleus sampling instead of greedy decoding.
• Fine-tune on more diverse data or add penalization for repetition (e.g., repet
```

Q17. Given this retrieval ranking:

Rank	Doc	Relevance Score
1	D1	3
2	D2	2
3	D3	0

Compute DCG@3 and NDCG@3, if ideal ranking is [D1, D2, D3].

$$DCG@3 = \frac{2^3 - 1}{\log_2(2)} + \frac{2^2 - 1}{\log_2(3)} + \frac{2^0 - 1}{\log_2(4)} = 7 + 1.89 + 0 = 8.89$$

$IDCG@3 = 8.89 \Rightarrow NDCG@3 = 8.89/8.89 = 1.0$

If ranks were swapped (e.g., D2 first, D1 second), NDCG would drop to ≈ 0.89 — showing discounted gain punishes lower-ranked relevant items.

Q18. When optimizing retrieval, if Recall@5 increases but Precision@5 decreases, what's happening?

The system retrieves more relevant documents (higher recall) but also more irrelevant ones (lower precision).

This often happens when you raise the number of retrieved items per query (larger k).

To rebalance: tune retrieval threshold or reranker to focus on higher-precision top results.

Q19. In RAG, the model's Factuality improves but ROUGE drops. Why?

Factuality measures truthfulness to sources, while ROUGE measures overlap with reference wording.

The model may now generate factually correct but lexically different sentences (e.g., paraphrases).

This trade-off is acceptable in open-ended generation — you should prefer higher factuality, even with slightly lower ROUGE.

Q20. How can you empirically improve NDCG in a retrieval system?

```
• Train a reranker (e.g., cross-encoder) to rescore top-k retrieved items based
• Use dense embeddings (e.g., from Contriever or E5) instead of sparse BM25.
• Calibrate ranking loss (like pairwise margin or softmax loss) to push relevant
```

Each of these shifts relevant documents toward rank 1, which logarithmically boosts NDCG.

Q21. If Distinct-2 ratio = 0.1, what does that indicate, and how can you improve it?

Distinct-2 = 0.1 means only 10% of bigrams are unique → the model's outputs are repetitive or templated.

To improve:

```
• Use temperature > 1.0 or top-p sampling for generation diversity.
• Add reinforcement learning rewards that encourage novelty.
• Expand fine-tuning corpus to include stylistically varied text.
```

1️⃣ Interpret what high/low X means (precision vs recall trade-off).

2️⃣ Explain what caused it (model bias, decoding, dataset).

3️⃣ Suggest specific, actionable fixes (e.g., change loss, sampling, or data type).

1. Example data used in these worked examples

Reference (R):

the cat is on the the mat → tokens (lowercased):

["the","cat","is","on","the","mat"] → length |R| = 6

Candidate / Generated (C):

the cat is on mat → tokens:

["the","cat","is","on","mat"] → length |C| = 5

(Using these two short sentences keeps calculations small while showing all important steps.)

———

2. ROUGE-1 (unigram): recall, precision, F1

Step A — Count overlapping unigrams (token counts):

Reference counts:
• the: 2, cat:1, is:1, on:1, mat:1

Candidate counts:
• the:1, cat:1, is:1, on:1, mat:1

Overlap count for each token = min(ref_count, cand_count). Sum of overlaps:

```
• the: min(2,1) = 1
• cat: 1
• is: 1
• on: 1
• mat: 1
```

Total overlaps = 5

Compute:

$$\bullet \text{Recall} = \frac{\text{overlap}}{|R|} = \frac{5}{6} \approx 0.8333$$

$$\bullet \text{Precision} = \frac{\text{overlap}}{|C|} = \frac{5}{5} = 1.0$$

$$\bullet F_1 = \frac{2 \cdot P \cdot R}{P + R} = \frac{2 \cdot 1.0 \cdot 0.8333}{1.0 + 0.8333} = \frac{1.6666}{1.8333} \approx 0.9091$$

Interpretation / how to improve:
• High ROUGE-1 recall & perfect precision here means candidate covers most reference words and contains few extras. To raise F1 further you need perfect recall (e.g., include the missing “the”).

———

3. ROUGE-2 (bigram): recall, precision, F1

Reference bigrams (adjacent pairs):

["the cat", "cat is", "is on", "on the", "the mat"] → 5 bigrams

Candidate bigrams:

["the cat", "cat is", "is on", "on mat"] → 4 bigrams

Overlapping bigrams: "the cat", "cat is", "is on" → 3 overlaps

Compute:

$$\bullet \text{Recall}_2 = 3/5 = 0.6$$
$$\bullet \text{Precision}_2 = 3/4 = 0.75$$
$$\bullet F_1^{(2)} = \frac{2 \cdot 0.6 \cdot 0.75}{0.6 + 0.75} = \frac{0.9}{1.35} = 0.6667$$

Interpretation / how to improve:

```
• Lower ROUGE-2 indicates word order/phrase mismatch. Improve by generating phra
```

———

4. ROUGE-L (LCS = Longest Common Subsequence) — recall, precision, F1

Find LCS between token sequences (not necessarily contiguous, but order preserved).

Reference: the cat is on the mat

Candidate: the cat is on mat

One LCS is ["the","cat","is","on","mat"] → length $\text{LCS} = 5$.

Compute:

$$\bullet R_L = \frac{\text{LCS}}{|R|} = 5/6 \approx 0.8333$$
$$\bullet P_L = \frac{\text{LCS}}{|C|} = 5/5 = 1.0$$
$$\bullet ROUGE-LF_1 = \frac{2 R_L P_L}{R_L + P_L} \approx 0.9091$$

Interpretation:

```
• ROUGE-L captures longer ordered matches; here it matches the ROUGE-1 F1 becaus
```

———

5. BLEU (brief worked example for BLEU-2: 1-gram & 2-gram)

Formulas:

• p_n = modified n-gram precision for $n = 1..N$ (counts clipped by reference counts)

• Geometric mean for $N = 2 : \mathrm{GM} = \sqrt{p_1 \cdot p_2}$

• Brevity Penalty (BP): if c = candidate length, r = reference length:
$
BP =
\begin{cases}
1 & c > r\[4pt]
e^{1 - r/c} & c \le r
\end{cases}
$

$$•BLEU = BP \times \exp\Big(\frac{1}{N}\sum_{n=1}^{N}\ln p_n\Big).For N = 2 we use GM above.$$

From earlier counts:

• p_1 = unigram precision = overlap / candidate_unigrams = 5/5 = 1.0

• p_2 = bigram precision = overlap_bigrams / candidate_bigrams = 3/4 = 0.75

• Geometric mean $\sqrt{1.0 \times 0.75} = \sqrt{0.75} \approx 0.8660254$

Brevity penalty:

• $c = 5, r = 6 \Rightarrow c \le r$, so

• $BP = e^{1-r/c} = e^{1-6/5} = e^{-0.2} \approx 0.81873075$

BLEU (N=2):

• $\mathrm{BLEU} = BP \times \mathrm{GM} \approx 0.81873075 \times 0.8660254 \approx 0.71$ (approx.)

Interpretation / improving BLEU:

```
• BLEU is precision-oriented and penalizes brevity. To raise BLEU in this exampl
```

———

6. Mean Reciprocal Rank (MRR)

Definition: for a set of queries Q, and each query i has the rank \text{rank}i of the first correct result,
$
\text{MRR} = \frac{1}{|Q|}\sum{i=1}^{|Q|} \frac{1}{\text{rank}_i}.$

Example: 3 queries, first-correct ranks = 1, 3, 2.

Compute:

$$\bullet \text{reciprocals} : 1/1 = 1.0, 1/3 \approx 0.3333, 1/2 = 0.5.$$
$$\bullet \text{Sum} = 1.0 + 0.3333 + 0.5 = 1.8333.$$
$$\bullet \text{MRR} = 1.8333/3 \approx 0.6111.$$

Interpretation / improving MRR:

```
• MRR focuses on first relevant hit; boosting rank-1 relevant items (via better
```

———

7. Precision@k, Recall@k, F1@k — worked example

Scenario: For a query, top-5 retrieved docs contain 3 relevant docs; there are 4 relevant docs in the whole corpus.

Compute:

$$\bullet \text{Precision@5} = 3/5 = 0.6$$
$$\bullet \text{Recall@5} = 3/4 = 0.75$$
$$\bullet \text{F1@5} = \frac{2 \cdot 0.6 \cdot 0.75}{0.6 + 0.75} = \frac{0.9}{1.35} = 0.6667$$

Interpretation / improving tradeoffs:

```
• Increasing k often raises Recall@k but lowers Precision@k. Use rerankers or st
```

---

8. Normalized Discounted Cumulative Gain (NDCG)

Formulas:

$$\bullet \text{DCG@}k = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_2(i+1)} \text{where } rel_i \text{ is graded relevance at rank} i.$$
$$\bullet \text{NDCG@}k = \frac{\text{DCG@}k}{\text{IDCG@}k} \text{with IDCG being DCG for ideal ranking.}$$

Example 1 (ideal order): relevance at ranks 1..3 = [3, 2, 0]

Compute DCG:

$$\bullet \text{rank1} : (2^3 - 1)/\log_2(2) = 7/1 = 7$$
$$\bullet \text{rank2} : (2^2 - 1)/\log_2(3) = 3/1.58496 \approx 1.8928$$
$$\bullet \text{rank3} : (2^0 - 1)/\log_2(4) = 0$$

Total DCG ≈ 7 + 1.8928 = 8.8928. Since ranking is ideal, IDCG = 8.8928 → NDCG = 1.0.

Example 2 (swapped top two): relevance [2, 3, 0]

Compute DCG:

$$\bullet \text{rank1} : (2^2 - 1)/\log_2(2) = 3$$
$$\bullet \text{rank2} : (2^3 - 1)/\log_2(3) = 7/1.58496 \approx 4.4185$$
$$\bullet \text{sum} \approx 3 + 4.4185 = 7.4185$$
$$\bullet \text{NDCG} = 7.4185/8.8928 \approx 0.8346$$

Interpretation / improving NDCG:

```
• NDCG rewards placing highly relevant docs near the top; improvements come from
```

---

9. Quick checklist: what metric changes imply and typical fixes

• ROUGE up, BLEU down → coverage increased but lexical fidelity decreased (more paraphrasing). Fix: tighten decoding (lower temperature), constrain generation length, or fine-tune with word-level losses.

• PPL down but human preference down → model became safe/generic (low surprisal) but unhelpful; encourage utility via fine-tuning on helpful dialogs or RLHF.

• Recall@k up, Precision@k down → retrieving more relevant docs but adding noise; improve reranker or thresholding.

• High Self-BLEU (low diversity) → increase sampling temperature/top-p, penalize repetition, diversify training data.

• NDCG low → tune for ranking losses that emphasize top ranks (e.g., LambdaRank, pairwise losses).