

# Lecture 12: Evaluation

Ryan Jackson and Jack Song

January 2026

## 1 Perplexity

**Definition 1.1 (Perplexity)** Recall that a language model is a probability distribution  $p(\mathbf{x})$  over sequences of tokens. The perplexity of a model on a database  $D$  is defined to be

$$\text{PPL}(D) = p(D)^{-\frac{1}{|D|}},$$

where  $p(D)$  is the probability assigned by the model to the dataset.

Perplexity is a negatively-oriented measure (i.e. lower perplexity values are better). There are two main perspectives from which one can understand perplexity. First, one can readily see from its definition that perplexity is equal to the inverse of the geometric mean probability of the correct token, where the probabilities are conditional on the previous tokens. This follows from the fact that for  $D = x_1 x_2, \dots, x_{|D|}$

$$p(D) = \prod_i p(x_i | x_{<i}),$$

and so

$$\text{PPL}(D)^{-1} = p(D)^{\frac{1}{|D|}} = \left( \prod_i p(x_i | x_{<i}) \right)^{\frac{1}{|D|}},$$

which is the geometric mean of the conditional probabilities  $p(x_i | x_{<i})$  from the model. Perplexity can also be understood as the exponentiation of cross-entropy loss. This formulation provides a more stable approach to computation by working with the sums of log probabilities. Before deriving this equivalence, recall that the cross-entropy loss of a model is negative log-likelihood of the true tokens computed from its distribution:

$$\mathcal{L}_{\text{CE}}(D) := -\frac{1}{|D|} \sum_i \log p(x_i | x_{<i}).$$

Then

$$\begin{aligned}\exp(\mathcal{L}_{\text{CE}}(D)) &= \exp\left(-\frac{1}{|D|} \sum_i \log p(x_i | x_{<i})\right) = \exp\left(\log\left(\left[\prod_i p(x_i | x_{<i})\right]^{-\frac{1}{|D|}}\right)\right) \\ &= \exp\left(\log\left(p(D)^{-\frac{1}{|D|}}\right)\right) \\ &= p(D)^{-\frac{1}{|D|}} = \text{PPL}(D).\end{aligned}$$

**Interpretation** Because perplexity is simply a monotonic transformation of the cross-entropy loss, it is natural to ask why we use perplexity at all rather than working directly with cross-entropy. The key advantage of perplexity is interpretability: its units admit a much more intuitive understanding of model performance. Suppose the perplexity of a dataset  $D$  is  $\text{PPL}(D) = K$ . By definition,

$$K = p(D)^{-\frac{1}{|D|}} = \left(\prod_i p(x_i | x_{<i})\right)^{-\frac{1}{|D|}} \implies \prod_i p(x_i | x_{<i}) = \left(\frac{1}{K}\right)^{|D|}.$$

Thus, a perplexity of  $K$  means that the probability the model assigns to correctly predicting all tokens in  $D$  is exactly what we would obtain if, at each step, the model assigned probability  $1/K$  to the correct token (conditioned on the previous tokens). In particular, when  $K$  is an integer, perplexity admits a simple interpretation: the model behaves as though it were selecting the correct next token uniformly at random from a set of  $K$  possible choices.

### Advantages of Perplexity

- **Theoretically well-founded:** Perplexity is directly derived from cross-entropy and log-likelihood, making it a principled statistical measure.
- **Probability-sensitive:** It accounts for model confidence, strongly penalizing confident incorrect predictions rather than only measuring correctness.
- **Dense evaluation signal:** Perplexity provides a token-level loss over the entire dataset, allowing stable and low-variance comparisons during training.
- **Task-agnostic:** It does not require task-specific labels and can be computed for any probabilistic language model.
- **Smoothness:** Smoother than downstream task accuracy, making it easier to fit scaling laws.

## Disadvantages of Perplexity

- **Weak correlation with downstream performance:** Improvements in perplexity do not reliably translate to gains in reasoning, factual accuracy, or task completion.
- **Dominated by frequent tokens:** Function words and local syntactic patterns disproportionately influence perplexity, while rare but semantically important tokens matter less.
- **Highly sensitive to tokenization:** Changes in vocabulary size or tokenization scheme can significantly alter perplexity, making cross-model comparisons unreliable.
- **Inadequate for behavioral evaluation:** Perplexity cannot assess instruction-following, safety behavior, or user preference alignment.
- **Encourages conservative outputs:** Optimizing perplexity favors high-probability, generic continuations that may be less informative or helpful.

## 2 Zero-Shot, One-Shot, and Few-Shot Evaluation

Large language models can be evaluated under different prompting regimes depending on how much task-specific information is provided at inference time. The most common settings are zero-shot, one-shot, and few-shot evaluation. These settings probe different model capabilities and lead to different trade-offs in realism, stability, and interpretability.

### 2.1 Zero-Shot Evaluation

**Definition 2.1 (Zero-Shot)** In zero-shot evaluation, the model is given only a task instruction and a test input, with no example demonstrations. The model must rely entirely on its pretrained and fine-tuned knowledge to produce an output.

**Motivation** Zero-shot evaluation aims to measure the model’s intrinsic capabilities without assistance from in-context examples. It most closely reflects real-world usage, where users typically do not provide labeled examples.

#### Advantages

- Provides a clean signal of generalization and raw capability.
- Simple to standardize and reproduce.
- Avoids dependence on prompt curation.
- Reduces the risk of benchmark leakage through demonstrations.

### Disadvantages

- Often underestimates the model's maximum achievable performance.
- Sensitive to instruction phrasing.
- Some tasks are unnatural or ambiguous without examples.

**Typical use cases:** Comparing instruction-tuned models, measuring broad capabilities, and evaluating real-world readiness.

## 2.2 One-Shot Evaluation

**Definition 2.2 (One-Shot)** In one-shot evaluation, the model is provided with exactly one example input-output pair before being asked to solve the test instance.

**Motivation** One-shot evaluation tests whether the model can infer the structure of a task from a minimal demonstration, serving as a compromise between zero-shot and few-shot settings.

### Advantages

- Reduces ambiguity in task format.
- Requires minimal additional context.
- Useful for diagnosing in-context learning behavior.

### Disadvantages

- Highly sensitive to the choice of example.
- Performance exhibits high variance.
- Rarely representative of real deployment scenarios.

**Typical use cases:** Quick diagnostic evaluations and simple structured tasks with strict formatting requirements.

## 2.3 Few-Shot Evaluation

**Definition 2.3 (Few-Shot)** In few-shot evaluation, the model is given multiple example input-output pairs (typically between 2 and 10) prior to the test query.

**Motivation** Few-shot evaluation leverages in-context learning, allowing the model to adapt to a task at inference time without parameter updates. This setting was especially important for early large language models.

### **Advantages**

- Often yields substantially higher performance.
- Reduces task ambiguity.
- Demonstrates the model’s ability to learn from context.

### **Disadvantages**

- Performance depends strongly on example selection and ordering.
- Increases token usage and inference cost.
- Less realistic for most users.
- Can inflate benchmark scores and obscure true generalization.

**Typical use cases:** Studying in-context learning, evaluating non-instruction-tuned models, and legacy benchmarks designed around few-shot prompting.

## **2.4 Discussion and Best Practices**

Zero-shot, one-shot, and few-shot evaluations measure distinct aspects of model behavior rather than a single notion of performance. Zero-shot evaluation emphasizes generalization and usability, while few-shot evaluation emphasizes adaptability through in-context learning.

Early LLMs required few-shot prompting to perform well. However, modern best practice favors zero-shot evaluation for standardized benchmarking and deployment decisions, with few-shot results reported separately when in-context learning is of interest. This shift away from few-shot evaluation is driven by the strong zero-shot performance of instruction-tuned models, as well as concerns about score inflation, confounding prompt quality with model quality, and the risk of train–test leakage. In all cases, the prompting protocol should be clearly specified to ensure interpretability and fair comparison.

## **3 Data Contamination and validity**

### **3.1 Validity: how do we know an evaluation is measuring what we think?**

- **Train–test overlap (contamination):** Evaluation cases and benchmark items appear in the training set, becomes closer to a measurement of memorization rather than generalization.
- **Dataset quality:** benchmarks can contain mislabeled answers, contradictions, ambiguous questions, or ill-posed items (e.g., missing information), which can dominate observed error rates.

## 3.2 The data contamination problem

- In LLM training context, it's hard to verify if the testing set is truly "held-out"
  - LLMs are typically trained on large corpuses (i.e. webscraped internet), and it could be that the LLMs have already seen benchmark evaluation questions previously.
  - Especially the case for benchmarks such as MMLU that is sourced directly from online questions.

## 3.3 Strategies for reducing contamination

### 3.3.1 If you control/know the training dataset (And testing set as well)

- Remove exact matches or near exact content between the datasets
- Commonly heuristic is n-gram overlap filtering (Typically  $n$  is large number such as  $n = 13$  to prevent over filtering and many false positives)
  - For a benchmark item and a training corpus, tokenize into  $n$ -grams.
  - If there's any (Any-match rule) or the size of the intersected n-gram set is greater than some threshold  $\tau$  (Threshold rule), then flag it
- Semantic / embedding similarity filters
- If in doubt, remove, false negatives have more impact than false positives as they invalidate eval claims.

### 3.3.2 If training data is unknown

- **Infer overlap via exchangeability / ordering tests.** If benchmark items are truly unseen, the model should not strongly prefer one arbitrary ordering of the same items over another. A simple test is to compare sequence log-probabilities under different orderings:

- **Canonical order:** take a list of benchmark items  $(x_1, x_2, \dots, x_n)$  in the dataset's original order and compute

$$\log p(x_1, x_2, \dots, x_n).$$

- **Shuffled order:** randomly permute the same items via a permutation  $\pi$  and compute

$$\log p(x_{\pi(1)}, x_{\pi(2)}, \dots, x_{\pi(n)}).$$

- **Contamination signal:** if the canonical ordering has *consistently* higher log-probability than shuffled orderings (large gaps across many permutations), this suggests the model may have encountered the benchmark (or its structure/ordering) during training.

This is a *heuristic signal*, not a proof.

- **Encourage reporting norms.** Because external verification is difficult, model providers should report overlap checks and decontamination procedures when publishing benchmark numbers, analogous to reporting uncertainty estimates (e.g., confidence intervals) in empirical results.
  - Use benchmarks with confirmed "held-out" testing datasets
  - Use fresh benchmarks with newly added cases or refresh of benchmark items.
  - Use multiple benchmarks in a similar domain and take note of model robustness

## 4 Knowledge Benchmarks

### 4.1 MMLU and MMLU-Pro

- **Description:** Multiple-choice questions across 57 academic subjects. MMLU-Pro removes noisy questions and increases answer choices.
- **Motivation:** Measure broad academic and factual knowledge.
- **Advantages:**
  - Wide domain coverage
  - Easy to score and compare
  - Popular standardized baseline
- **Disadvantages:**
  - Emphasizes memorization over reasoning
  - Vulnerable to train-test contamination
  - Multiple-choice limits expressiveness

### 4.2 GPQA

- **Description:** Expert-written graduate-level science questions designed to be hard even with search tools.
- **Motivation:** Test deep, expert-level reasoning.
- **Advantages:**
  - High difficulty
  - Written and validated by PhDs
  - Lower contamination risk

- **Disadvantages:**

- Narrow domain focus
- Expensive to scale

### 4.3 Humanity's Last Exam

- **Description:** Large, multimodal, high-quality exam with expert incentives and multi-stage filtering.

- **Motivation:** Create a frontier benchmark resistant to saturation.

- **Advantages:**

- High quality and difficulty
- Broad subject and modality coverage

- **Disadvantages:**

- Very expensive
- Still quiz-style evaluation

## 5 Instruction-Following Benchmarks

### 5.1 Chatbot Arena

- **Description:** Humans compare anonymized model responses; rankings computed via Elo scores.

- **Motivation:** Measure real-world user preference.

- **Advantages:**

- High realism
- Live, evolving prompts

- **Disadvantages:**

- Noisy and hard to reproduce
- Expensive human evaluation

## 5.2 Instruction-Following Eval (IFEval)

- **Description:** IFEval evaluates instruction following by adding synthetic, automatically verifiable constraints (e.g., formatting or length requirements) to prompts.
- **Motivation:** The goal is to enable scalable and reproducible evaluation without human judges.
- **Advantages:**
  - Fully automatic evaluation.
  - Cheap and easy to scale.
  - Clear pass/fail criteria.
- **Disadvantages:**
  - Constraints are artificial and shallow.
  - Does not evaluate semantic correctness or usefulness.
  - Limited coverage of real instruction-following behavior.

## 5.3 AlpacaEval

- **Description:** AlpacaEval consists of 805 instruction-following prompts sourced from multiple datasets. Model outputs are compared against a reference model (typically GPT-4) and judged by GPT-4 itself.
- **Motivation:** The benchmark aims to approximate human preference judgments at lower cost than human evaluation.
- **Advantages:**
  - Scalable and inexpensive.
  - Easy to run for many models.
  - Produces a simple win-rate metric.
- **Disadvantages:**
  - Strong dependence on the chosen reference model.
  - Potential bias from using an LLM as a judge.
  - Less robust to stylistic differences.

## 5.4 WildBench

- **Description:** WildBench is constructed from real human–chatbot conversations. Responses are evaluated using GPT-4-based judges with structured checklists to guide scoring.
- **Motivation:** The benchmark aims to combine realism with scalable automatic evaluation while maintaining high correlation with human judgments.
- **Advantages:**
  - Derived from real-world usage.
  - Strong correlation with Chatbot Arena results.
  - More structured judging than simple preference models.
- **Disadvantages:**
  - Still relies on LLM-based judges.
  - Sensitive to judge prompting and rubric design.
  - Not fully transparent or reproducible.

# 6 Agent Benchmarks

## 6.1 SWE-Bench

- **Description:** Real software engineering tasks evaluated with unit tests.
- **Motivation:** Measure tool use and multi-step reasoning.
- **Advantages:** Realistic and objectively scored.
- **Disadvantages:** Strong dependence on scaffolding and tooling.

## 6.2 CyBench

- **Description:** CyBench consists of cybersecurity Capture-the-Flag (CTF) tasks, with difficulty measured by first-solve time.
- **Motivation:** The benchmark evaluates advanced reasoning and tool use in adversarial cybersecurity environments.
- **Advantages:**
  - Very challenging tasks.
  - Clear notion of task difficulty.
  - Relevant for both security research and defense.

- **Disadvantages:**

- Strong dual-use concerns.
- Sensitive to tooling and environment setup.
- Small benchmark size.

### 6.3 MLE-Bench

- **Description:** MLE-Bench evaluates agents on full machine learning workflows using 75 Kaggle-style competitions.
- **Motivation:** The goal is to assess long-horizon reasoning, planning, and execution in realistic ML engineering tasks.

- **Advantages:**

- Highly realistic end-to-end evaluation.
- Tests planning, iteration, and tool usage.

- **Disadvantages:**

- Expensive and slow to evaluate.
- Hard to standardize and reproduce.
- Conflates model ability with agent scaffolding.

## 7 Pure Reasoning Benchmarks

### ARC-AGI

- **Description:** Abstract pattern-recognition puzzles with minimal reliance on prior knowledge.
- **Motivation:** Isolate reasoning from memorization.
- **Advantages:** Strong test of generalization.
- **Disadvantages:** Small dataset and narrow task format.

## 8 Safety Benchmarks

### 8.1 HarmBench

- **Description:** HarmBench evaluates language models on 510 categories of harmful behaviors that violate legal, ethical, or social norms.
- **Motivation:** The benchmark aims to measure a model’s tendency to generate explicitly harmful content when prompted.

- **Advantages:**

- Systematic coverage of many harmful behaviors.
- Useful for stress-testing refusal behavior.
- Relevant for deployment safety checks.

- **Disadvantages:**

- Over-emphasizes refusal as a notion of safety.
- Highly dependent on prompt phrasing.
- Does not capture contextual or downstream harms.

## 8.2 AIR-Bench

- **Description:** AIR-Bench is a large safety benchmark grounded in regulatory frameworks and company policies, consisting of 5,694 prompts across 314 risk categories.

- **Motivation:** The goal is to align safety evaluation with real regulatory and compliance concerns.

- **Advantages:**

- Explicitly policy-driven.
- Broad and fine-grained taxonomy of risks.
- Suitable for governance and auditing contexts.

- **Disadvantages:**

- Safety definitions are context- and jurisdiction-dependent.
- Hard to reduce to a single interpretable score.
- Less informative about real-world utility trade-offs.

## 8.3 Jailbreaking Evaluations

- **Description:** Adversarial prompts designed to bypass safeguards.

- **Motivation:** Stress-test robustness of safety mechanisms.

- **Advantages:** Reveals real vulnerabilities.

- **Disadvantages:** Unstable and difficult to interpret quantitatively.

## 9 Realism-Oriented Benchmarks

### 9.1 Clio

- **Description:** Clio uses language models to analyze and summarize patterns in real user queries, without releasing raw user data.
- **Motivation:** The benchmark aims to understand how language models are used in practice and what users actually ask for.
- **Advantages:**
  - High realism.
  - Grounded in real-world usage.
  - Avoids direct release of sensitive data.
- **Disadvantages:**
  - Limited reproducibility.
  - Indirect evaluation of model outputs.
  - Less suitable for direct model ranking.

### 9.2 MedHELM

- **Description:** MedHELM evaluates language models on 121 clinical tasks sourced from practicing clinicians, covering realistic medical workflows.
- **Motivation:** The benchmark addresses the gap between standardized medical exams and real clinical decision-making.
- **Advantages:**
  - High clinical realism.
  - Expert-authored tasks.
  - Better proxy for real medical deployment.
- **Disadvantages:**
  - Privacy constraints limit data release.
  - Smaller scale than academic benchmarks.
  - Harder to compare across models.