Contents

Chapter 2 — Intuitive MCQ Bank (Foundation Models / LLMs)

75 multiple-choice questions, intuitive/conceptual level, covering the main chapter AND the extended supplement. One correct option each. Cover the answer (blockquote) while testing yourself. Bilingual explanations (English + বাংলা).

Exam style: "Which statement best describes…", exactly one correct option; use full technical terms, no abbreviations.

বাংলা ব্যাখ্যা: এই ব্যাংকে অধ্যায় ২ (মূল + extended)-এর ৭৫টি ধারণামূলক MCQ আছে প্রতিটি প্রশ্নের নিচে উত্তর ও সংক্ষিপ্ত ব্যাখ্যা (ইংরেজি + বাংলা) এই প্রশ্নগুলো extended ফাইলের ৪৭টি প্রশ্ন থেকে আলাদা — একই টপিক হলেও ভিন্ন কোণ ও ভিন্ন শব্দে লেখা

Topic: Training Data and Bias

Q1. A team trains an English-only model on Wikipedia plus Reddit, then is surprised it performs poorly on Bengali medical questions. Which statement best explains this outcome?

- A. Capability tracks the training distribution, and a language absent from the corpus is never learned.
- B. The transformer architecture is incompatible with Bengali script.
- C. The tokenizer automatically deletes non-English characters during inference.
- D. The model needs a larger context window to handle Bengali.

> **Answer: A.** A model encodes only the statistical regularities present in its data ("no Thai data, no Thai capability"); the gap is a data artefact, not an architecture or tokenizer one. The other options misattribute a data-coverage limitation to architecture, tokenization, or context length. **বাংলা:** মডেল যা খায় তাই শেখে — কর্পাসে যে ভাষা নেই, সেই ভাষায় দুর্বল; এটা স্থাপত্য বা টোকেনাইজারের দোষ নয়

Q2. Why does training tend to *amplify* a bias rather than merely mirror it?

- A. A dedicated "bias parameter" stores the asymmetry and scales it up.
- B. The optimizer rewards confident, low-loss predictions, so it pushes probability past the raw data frequency of the majority pattern.
- C. Tokenization doubles the frequency of majority tokens.
- D. Larger vocabularies inject extra bias.

> **Answer: B.** The next-token objective rewards confident predictions, so the model generalizes a mild majority correlation and over-shoots the observed frequency. The other options invent mechanisms (a "bias parameter", tokenizer doubling, vocabulary) that the lecture never describes. **বাংলা:** optimizer নিশ্চিত-low-loss prediction-কে পুরস্কার দেয়, তাই সংখ্যাগরিষ্ঠ ঝোঁককে ডেটা-ফ্রিকোয়েন্সির চেয়েও বেশি বাড়িয়ে দেয়

Q3. A startup believes adding 10x more raw Common Crawl data will always improve their model. Which statement best challenges this belief?

- A. Raw web data cannot be tokenized.
- B. Adding data removes the need for alignment.
- C. Quality matters: a smaller classifier-filtered corpus (FineWeb-Edu) can beat a larger raw one, the "garbage in, garbage out" principle.
- D. More tokens always lower loss regardless of quality.

> **Answer: C.** The lecture stresses that data quality drives performance; filtered FineWeb-Edu outperforms larger raw Common Crawl. The "more tokens always help" option is the myth being refuted; raw web data is in fact tokenized; and adding data does not replace alignment. **বাংলা:** বেশি ডেটা সবসময় ভালো নয় — মান খারাপ হলে ফল খারাপ; ছোট কিন্তু ফিল্টার করা FineWeb-Edu বড় কাঁচা কর্পাসকে হারায়

Q4. Removing documents that appear hundreds of times in the corpus (deduplication) improves generalization mainly because:

- A. it shrinks the vocabulary so sequences get shorter.
- B. it converts near-duplicates into exact duplicates.
- C. it increases the irreducible loss floor.
- D. duplicated documents receive multiplied gradient weight, causing verbatim memorization instead of learning general patterns.

Answer: D. A document seen 100 times gets 100 times the gradient, so the model memorizes it rather than generalizing; deduplication frees capacity for diverse text. The other options confuse deduplication with tokenization, reverse the goal, or name an unrelated quantity. **বাংলা:** ১০০ বার দেখা ডকুমেন্ট ১০০ গুণ গ্রেডিয়েন্ট পায় → মডেল মুখস্থ করে, সাধারণ প্যাটার্ন শেখে না; dedup এই অপচয় থামায়

Topic: Tokenization and API Cost

Q5. Modern large language models use byte-level Byte-Pair Encoding chiefly because it:

- A. can represent any string as bytes (no out-of-vocabulary) while giving frequent strings short codes.
- B. guarantees one token per word in every language.
- C. produces the linguistically optimal split into morphemes.
- D. uses a fixed dictionary that never changes after release.

Answer: A. Starting from the 256 byte values, every string is representable, eliminating out-of-vocabulary failures, and learned merges shorten frequent strings. The other options are false: rare words still split, Byte-Pair Encoding is a greedy frequency heuristic rather than morphological, and a fixed dictionary is irrelevant to the core advantage. **বাংলা:** byte থেকে শুরু বলে যেকোনো স্ট্রিং লেখা যায় (OOV নেই), আর ঘনঘন স্ট্রিং ছোট কোড পায় — এটাই byte-level BPE-র মূল সুবিধা

Q6. Two providers charge the same price *per token* yet bill different amounts for the identical sentence. Why?

- A. One provider secretly charges per character.
- B. They use different tokenizers, so the same text maps to a different token count.
- C. The context window differs between them.
- D. One uses cosine similarity for billing.

Answer: B. Different tokenizers segment identical text into different numbers of tokens, so the effective cost per word differs even at an equal price-per-token. The other options are not the billing mechanism described. **বাংলা:** ভিন্ন টোকেনাইজার একই বাক্যকে ভিন্ন সংখ্যক টোকেনে ভাঙে, তাই প্রতি-টোকেন দাম এক হলেও মোট খরচ আলাদা

Q7. A smaller vocabulary, all else equal, tends to *increase* which cost?

- A. Embedding-matrix memory.
- B. The entropy of natural language.
- C. Per-sequence compute, because each sentence becomes more tokens (longer sequences).
- D. The size of the tokenizer file on disk.

Answer: C. Fewer token types mean more tokens per sentence, lengthening sequences, and attention cost grows with sequence length. Embedding-matrix memory moves the opposite way (a larger vocabulary costs more), while the entropy of language and the tokenizer file size are unrelated. **বাংলা:** ছোট ভোকাব → প্রতি বাক্যে বেশি টোকেন → লম্বা সিকোয়েন্স → বেশি (quadratic) attention খরচ

Q8. Why can you not swap a model's tokenizer for a better one after training?

- A. The license prohibits it.
- B. Tokenizers are stored encrypted.
- C. It would change the learning-rate schedule.
- D. Every embedding row and the output projection are keyed to specific token identifiers from the trained vocabulary.

Answer: D. Input embeddings and the output layer are bound to specific token identifiers, so a new tokenizer would mismatch every parameter that touches token identity. The other options are not the structural reason. **বাংলা:** input embedding ও output layer নির্দিষ্ট token ID-র সাথে বাঁধা, তাই টোকেনাইজার বদলালে প্রতিটি token-সম্পর্কিত প্যারামিটার মিলবে না

Q9. In a Byte-Pair Encoding merge process, why does the word "lowest" tokenize into only two pieces (low + est) even though "lowest" never appeared in the training corpus?

- A. Because merges learned from frequent substrings (low, est) compose to cover unseen words.
- B. Because the tokenizer memorized "lowest" as a special token.
- C. Because byte-level encoding stores whole words.
- D. Because the end-of-word marker was ignored.

Answer: A. Sub-word merges built from frequent substrings recombine to represent novel words compactly, the core power of Byte-Pair Encoding. Memorizing whole words or storing whole words by byte-level encoding contradicts how it works, and ignoring the end-of-word marker would change the counts rather than enable composition. **বাংলা:** ঘনঘন আসা টুকরো (low, est) থেকে শেখা merge জোড়া লেগে অদেখা শব্দও অল্প টোকেনে গড়ে — এটাই sub-word টোকেনাইজেশনের শক্তি

Q10. Which statement best describes Byte-Pair Encoding as an algorithm?

- A. It finds the provably optimal vocabulary for a corpus.
- B. It greedily merges the most frequent adjacent symbol pair, repeatedly, until a target vocabulary size is reached.
- C. It assigns each whole word a random integer identifier.
- D. It splits text purely on whitespace.

Answer: B. Byte-Pair Encoding iteratively merges the most frequent adjacent pair, a greedy heuristic rather than an optimizer. Calling it optimal overstates it, while random word identifiers and whitespace splitting describe word-level or whitespace schemes. **বাংলা:** BPE বারবার সবচেয়ে ঘনঘন আসা পাশাপাশি জোড়াকে merge করে — এটা greedy পদ্ধতি, optimal নয়

Topic: Input/Output Pipeline and Context Window

Q11. Training a transformer is parallel across all positions but generation is inherently sequential. Why?

- A. Generation uses a different architecture than training.
- B. Training disables the causal mask.
- C. During generation each new token depends on the previously sampled token, so positions cannot be computed all at once.
- D. Generation cannot use the embedding matrix.

Answer: C. Teacher forcing supplies all true prefixes in training (parallel), but at inference each token must be sampled before the next can be conditioned on it. The other options are false, since the same architecture and components are used. **বাংলা:** ট্রেনিং-এ teacher forcing সব আসল prefix একসাথে দেয় (parallel), কিন্তু জেনারেশনে আগের টোকেন স্যাম্পল না হলে পরেরটা হিসাব করা যায় না

Q12. What is "teacher forcing" and why does it matter for transformer pretraining?

- A. Forcing the model to use a teacher network at inference.
- B. Feeding the model's own outputs back during training to add noise.
- C. A regularization penalty on large weights.
- D. Conditioning every position on the *true* prefix during training, which lets all positions be computed in parallel.

Answer: D. Using ground-truth prefixes (not the model's outputs) lets every position's loss be computed simultaneously, enabling efficient parallel training. The other options mischaracterize it. **বাংলা:** teacher forcing মানে ট্রেনিং-এ প্রতিটি অবস্থানকে আসল prefix দেওয়া — তাই সব অবস্থান একসাথে সমান্তরালে হিসাব করা যায়

Q13. The context window of a model is best described as:

- A. the maximum number of tokens (input plus generated) it can process at once.
- B. the model's long-term knowledge stored in its weights.
- C. the number of transformer layers.
- D. the vocabulary size.

Answer: A. It is short-term memory bounding the combined input and output token count, fixed at design time. It is not parametric knowledge, layer count, or vocabulary size. **বাংলা:** কনটেক্সট উইন্ডো = একসাথে প্রসেস করা সর্বোচ্চ টোকেন সংখ্যা (ইনপুট + আউটপুট); এটা স্বল্পমেয়াদি স্মৃতি, weight-এ রাখা জ্ঞান নয়

Q14. A chatbot must "remember" a 300-page manual that far exceeds its context window. Which approach is *not* among the standard overflow strategies from the lecture?

- A. Drop the oldest tokens.
- B. Enlarge the context window at runtime to fit everything.
- C. Replace older tokens with a running summary.
- D. Move older content into a database and retrieve relevant chunks (retrieval-augmented generation).

Answer: B. The window is a fixed architectural property set at training time; you manage overflow by dropping, summarizing, or retrieving, not by enlarging it on the fly. The other three are the lecture's strategies. **বাংলা:** উইন্ডো স্থাপত্যগত, রানটাইমে বাড়ানো যায় না; overflow সামলানো হয় বাদ দিয়ে / সারাংশ করে / RAG দিয়ে

Q15. Why does doubling the context length more than double the per-step attention compute?

- A. Longer contexts use a larger vocabulary.
- B. The embedding dimension doubles with context.
- C. Self-attention compares every token with every other, so cost grows with the square of sequence length.
- D. Doubling context halves the precision.

Answer: C. Attention is quadratic in sequence length, so twice the tokens means about four times the attention compute. The other options invent dependencies that do not exist. **বাংলা:** attention প্রতিটি টোকেনকে প্রতিটি টোকেনের সাথে তুলনা করে (O(n^2)), তাই দৈর্ঘ্য ২ গুণ হলে খরচ প্রায় ৪ গুণ

Topic: Embeddings and Word2Vec

Q16. Why can one-hot vectors not express that "bank" and "loan" are related?

- A. One-hot vectors are too high-dimensional to compute with.
- B. One-hot vectors change during inference.
- C. One-hot vectors require a graphics processing unit.
- D. Each token sits on its own axis, so all one-hot vectors are mutually orthogonal and every pair is equidistant.

Answer: D. Orthogonality means every pair has identical (zero) similarity, so no relation can be encoded, the core representational limit. High dimensionality is a secondary efficiency issue rather than why similarity fails, and one-hot vectors neither change at inference nor require special hardware. **বাংলা:** one-hot ভেক্টর সব আলাদা অক্ষে $\rightarrow$ পরস্পর orthogonal $\rightarrow$ সব জোড়ার মিল সমান (শূন্য), তাই কোনো সম্পর্ক ধরা যায় না

Q17. Looking up row i of the embedding matrix for token i is mathematically equivalent to:

- A. multiplying the embedding matrix by the one-hot vector of token i.
- B. applying a non-linear activation to the token identifier.
- C. running the token through the tokenizer again.
- D. normalizing the token's frequency.

Answer: A. A one-hot vector times the embedding matrix selects exactly that token's row; the embedding layer is a linear projection from the vocabulary size to the embedding dimension. The

other options describe different operations. **বাংলা:** one-hot x embedding matrix মানেই সেই টোকেনের সারি বেছে নেওয়া — embedding layer হলো V→d একটা linear projection

Q18. In a trained embedding space, a token's vector *length (norm)* most closely reflects its:

- A. exact meaning.
- B. training frequency / generality.
- C. position in the sentence.
- D. part of speech.

Answer: B. Frequent words accumulate more gradient updates and get larger norms (a dense cluster of generic usage); direction, not length, carries meaning. The other options confuse length with meaning, position, or grammar. **বাংলা:** ঘনঘন শব্দ বেশি গ্রেডিয়েন্ট পেয়ে বড় norm পায় (সাধারণ ব্যবহার); অর্থ থাকে দিকে, দৈর্ঘ্যে নয়

Q19. Why is cosine similarity, not Euclidean distance, the standard tool for comparing word meanings?

- A. Cosine is always faster to compute.
- B. Cosine never returns negative values.
- C. Cosine compares direction and ignores magnitude, so synonyms of different frequency still match.
- D. Cosine needs no vectors.

Answer: C. Meaning lives in direction; cosine ignores the norm, so "car" and "automobile" match regardless of how often each occurs. Cosine is not generally faster, it ranges from minus one to one (so it can be negative), and it of course requires vectors. **বাংলা:** অর্থ থাকে দিকে; cosine দৈর্ঘ্য উপেক্ষা করে, তাই ভিন্ন ফ্রিকোয়েন্সির সমার্থক শব্দও মিলে যায়

Q20. Word2Vec's skip-gram objective is to:

- A. predict the center word from its surrounding context.
- B. translate words between two languages.
- C. compress the vocabulary into fewer tokens.
- D. predict context words from a center word.

Answer: D. Skip-gram maximizes the probability of the context given the center word. Predicting the center from context is the continuous-bag-of-words variant, while translation and compression are unrelated tasks. **বাংলা:** skip-gram কেন্দ্র-শব্দ থেকে context শব্দ predict করে (P(context|center)); উল্টোটা (context থেকে কেন্দ্র) হলো CBOW

Q21. What is the key difference between Word2Vec embeddings and the embeddings inside a transformer?

- A. Word2Vec vectors are static (one fixed vector per word), whereas transformer embeddings become context-dependent through attention.
- B. Word2Vec vectors are larger.
- C. Transformer embeddings are hand-designed.
- D. Word2Vec uses cosine and transformers use Euclidean.

Answer: A. Word2Vec gives one vector per word with no context; in a transformer the input embeddings are made context-dependent by self-attention. Size is not the point, transformer embeddings are learned rather than hand-designed, and the metric claim is invented. **বাংলা:** Word2Vec-এর ভেক্টর স্থির (প্রতি শব্দে একটা), কিন্তু transformer-এ attention দিয়ে সেগুলো context-নির্ভর হয়ে যায়

Topic: Attention and the Transformer

Q22. In the Query-Key-Value formulation, which intuition is correct?

- A. The Query says "what content I hand over if selected."
- B. The Query says "what I am looking for," the Key "what I advertise," and the Value "what content I pass on."
- C. The Value says "what I am looking for."
- D. All three vectors are identical for a given token.

Answer: B. Query is what I seek, Key is what I advertise, and Value is the content handed over; a query-key dot product scores relevance and the weights mix the values. Two options swap these roles, and they are not identical, since they come from three different projections. **বাংলা:** Query = কী খুঁজছি, Key = নিজের সম্পর্কে কী বিজ্ঞাপন, Value = নির্বাচিত হলে কী বিষয়বস্তু দিই — তিনটি আলাদা projection

Q23. Why are the query-key dot products divided by the square root of the key dimension before the softmax?

- A. To normalize the attention weights so each row sums to one.
- B. To reduce attention's complexity from quadratic to linear.
- C. To keep the dot-product variance near one, preventing softmax saturation and vanishing gradients.
- D. To apply the causal mask.

Answer: C. A dot product of d_k components has variance about d_k; dividing by its square root restores variance near one so the softmax stays sensitive and gradients flow. Summing each row to one is the softmax's own job, the cost stays quadratic, and masking is a separate step. **বাংলা:** d_k মাত্রার ডট-প্রোডাক্টের variance $\approx$ d_k; $\sqrt{\text{d\_k}}$ দিয়ে ভাগ করলে variance ≈ 1 থাকে $\rightarrow$ softmax saturate করে না, gradient বাঁচে

Q24. In decoder-only models, the causal mask sets future-position scores to negative infinity *before* the softmax. What would go wrong if you instead zeroed those weights *after* the softmax?

- A. Nothing, the result is identical.
- B. The model would become bidirectional and faster.
- C. The embedding matrix would need retraining.
- D. The rows would no longer sum to one, distorting the weighted average of values.

Answer: D. Masking must happen before the softmax so each row still normalizes to one; zeroing afterward breaks the probability distribution over keys. The other options are incorrect consequences. **বাংলা:** মাস্ক softmax-এর আগে দিতে হয় যাতে প্রতিটি সারির যোগফল ১ থাকে; পরে শূন্য করলে বণ্টন ভেঙে যায়

Q25. Why does multi-head attention often outperform a single attention head of the same total width?

- A. Each head can specialize in a different relation (syntax, coreference, positional patterns).
- B. It reduces total compute below one head.
- C. It removes the need for positional encoding.
- D. It eliminates the residual connection.

Answer: A. Splitting into several heads of smaller size lets each head learn a distinct pattern at the same total cost. Compute is matched rather than reduced, and positional encoding and residual connections are unrelated to this. **বাংলা:** h-টা মাথায় ভাগ করলে মোট খরচ একই থাকে কিন্তু প্রতিটি মাথা আলাদা সম্পর্ক (syntax, coreference) শিখতে পারে

Q26. What does the residual (skip) connection y = x + Sublayer(x) primarily enable?

- A. It shrinks the vocabulary.
- B. It gives gradients an identity path so very deep stacks can train.
- C. It removes the need for attention.
- D. It makes the model permutation-invariant.

Answer: B. The identity path lets gradients flow back unimpeded, enabling deep stacks. The other options are unrelated to what residual connections do. **বাংলা:** residual সংযোগ gradient-কে একটা identity পথ দেয়, তাই বহু স্তর গভীর মডেলও ট্রেন করা যায়

Q27. Self-attention by itself is permutation-equivariant. What practical problem does this create, and what fixes it?

- A. It makes the model too slow; weight tying fixes it.
- B. The vocabulary explodes; sub-word tokenization fixes it.
- C. Word order is invisible, so "dog bites man" and "man bites dog" look identical; positional encoding fixes it.
- D. Gradients vanish; RMSNorm fixes it.

Answer: C. Permutation equivariance means shuffling tokens just shuffles outputs, so order is lost; positional encoding injects position. The other options pair the wrong problem with the wrong fix. **বাংলা:** permutation-equivariant হওয়ায় শব্দের ক্রম অদৃশ্য ("কুকুর কামড়াল মানুষ" = "মানুষ কামড়াল কুকুর"); positional encoding এই ক্রম যোগ করে

Q28. The feed-forward network inside each transformer block typically expands the dimension about fourfold and back. Which statement best describes its role?

- A. It mixes information across tokens, replacing attention.
- B. It computes the positional encoding.
- C. It normalizes activations across the batch.
- D. It operates per position (per token), holds most of the model's parameters, and often acts like key-value memory for facts.

Answer: D. The feed-forward network is applied independently at each position, contains most parameters, and is often interpreted as factual memory. Attention (not the feed-forward network) mixes tokens, and positional encoding and batch normalization are other components. **বাংলা:** FFN প্রতিটি অবস্থানে আলাদাভাবে কাজ করে, বেশিরভাগ প্যারামিটার ধরে রাখে, প্রায়ই তথ্যের key-value মেমরির মতো আচরণ করে

Q29. When reading a decoder-only attention map, which observation confirms the causal mask is working?

- A. The upper triangle (key position later than query position) is empty.
- B. Every column sums to one.
- C. The matrix is symmetric.
- D. The diagonal is all zeros.

Answer: A. A causal mask zeroes attention to future tokens, so the upper-right triangle is empty and each row (a token's attention budget) sums to one. The column-sum claim names the wrong axis, and symmetry and a zero diagonal are false for causal attention. **বাংলা:** causal মাস্ক ভবিষ্যৎ টোকেনে ওজন শূন্য করে, তাই উপরের-ডান ত্রিভুজ ফাঁকা; প্রতিটি সারির যোগফল ১

Q30. Which statement best captures *why* attention was an advance over recurrent sequence-to-sequence models?

- A. Attention removes positional information that confused recurrent networks.
- B. Attention replaces a single fixed-length context vector with a dynamic, content-dependent representation, dissolving the information bottleneck.
- C. Attention makes the vocabulary smaller.
- D. Attention eliminates the need for embeddings.

Answer: B. Recurrent sequence-to-sequence models compressed the whole source into one vector (the bottleneck); attention lets each output token look at all inputs dynamically. The other options are false advantages. **বাংলা:** RNN seq2seq পুরো বাক্য এক ভেক্টরে চাপত (bottleneck); attention প্রতিটি আউটপুটকে সব ইনপুটে গতিশীলভাবে তাকাতে দেয়

Topic: Modern Transformer Variants

Q31. During generation, why does the key-value cache exist, and what does it save?

- A. It stores the vocabulary so detokenization is faster.
- B. It stores the optimizer states between steps.
- C. It stores past tokens' keys and values so each new token needs one forward pass over a single position instead of re-encoding the whole prefix.
- D. It caches the positional encodings only.

Answer: C. Recomputing all past keys and values each step would be quadratic per token; caching them makes each step process just the new position. The other options misidentify what is cached. **বাংলা:** প্রতিবার সব আগের K/V আবার হিসাব করলে O(n^2) খরচ; KV ক্যাশ সেগুলো জমিয়ে রাখে, তাই প্রতি ধাপে শুধু নতুন অবস্থান হিসাব হয়

Q32. Grouped-query attention lets several query heads share one key/value head. What is its *main* benefit, and a common misconception to avoid?

- A. It greatly reduces the parameter count; the misconception is that it saves memory.
- B. It removes the need for the causal mask; the misconception is that it changes the vocabulary.
- C. It doubles the context window; the misconception is that it slows inference.
- D. It saves key-value cache memory and bandwidth at inference; the misconception is that it mainly reduces parameter count.

Answer: D. Grouped-query attention's win is a smaller key-value cache (memory and bandwidth) at inference, not a large parameter reduction. The other options invert the truth or are simply false. **বাংলা:** GQA-র মূল লাভ inference-এ ছোট KV ক্যাশ (মেমরি/ব্যান্ডউইথ), প্যারামিটার কমানো নয় — এই ভুল ধারণা এড়াতে হবে

Q33. A model with 32 layers and 32 query heads switches from multi-head attention to grouped-query attention with 8 key/value heads. The key-value cache per token changes by roughly:

- A. a fourfold reduction.
- B. no change.
- C. a thirty-twofold increase.
- D. a twofold increase.

Answer: A. The cache scales with the number of key/value heads; 32 to 8 heads is a $32/8 =$ fourfold reduction. The other options contradict the linear dependence on the number of key/value heads. **বাংলা:** ক্যাশ K/V head সংখ্যার সমানুপাতিক; ৩২ $\rightarrow$ ৮ মানে ৩২/৮ = ৪ গুণ কম

Q34. Rotary Position Embedding encodes position by *rotating* query and key vectors. Why is this elegant compared with *adding* a position vector?

- A. Rotation increases the vector length proportionally to position.
- B. Rotations preserve length, and the dot product of two rotated vectors depends only on their position *difference*, so attention scores encode relative distance.
- C. Rotation is applied to the values, freeing the queries.
- D. Rotation removes the need for any frequency parameters.

Answer: B. Because rotation is length-preserving and the rotated dot product depends on the position difference, attention scores naturally capture relative position. Rotation does not change length, it is applied to queries and keys (not values), and per-pair frequencies remain. **বাংলা:** ঘোরালে দৈর্ঘ্য অপরিবর্তিত থাকে, আর দুই ঘোরানো ভেক্টরের ডট-প্রোডাক্ট নির্ভর করে অবস্থানের পার্থক্যের ওপর $\rightarrow$ attention আপেক্ষিক দূরত্ব ধরে

Q35. RMSNorm differs from LayerNorm mainly in that it:

- A. normalizes across the batch instead of across features.
- B. adds a second bias term.
- C. drops the mean-subtraction (and bias), normalizing only by the root-mean-square, with empirically equal quality.
- D. is applied only at inference.

Answer: C. RMSNorm omits mean-centering and the bias, scaling by the root-mean-square, fewer operations with similar quality. Both methods normalize per token across features, it removes rather than adds a bias term, and it is used in both training and inference. **বাংলা:** RMSNorm mean-বিয়োগ ও bias বাদ দিয়ে শুধু root-mean-square দিয়ে normalize করে — কম অপারেশন, একই মানের ফল

Q36. Where you place LayerNorm changes whether a deep transformer trains at all. Why does Pre-LayerNorm allow far deeper networks than Post-LayerNorm?

- A. Pre-LayerNorm uses fewer parameters.
- B. Post-LayerNorm removes attention.
- C. Pre-LayerNorm enlarges the vocabulary.
- D. Pre-LayerNorm keeps the residual path a clean identity, so gradients flow back unmodified instead of being rotated or scaled at every block.

Answer: D. Post-LayerNorm sits on the residual path and distorts gradients (unstable past roughly 24 layers); Pre-LayerNorm normalizes inside the block, leaving a clean identity path for 100 or more layers. The other options are unrelated. **বাংলা:** Post-LN residual পথের উপর বসে gradient বিকৃত করে; Pre-LN ব্লকের ভেতরে normalize করে, residual পথ পরিষ্কার identity রাখে → ১০০+ স্তর স্থিরভাবে ট্রেন হয়

Topic: Decoding and Sampling

Q37. As the sampling temperature approaches zero, decoding becomes:

- A. equivalent to greedy decoding (always the highest-probability token).
- B. uniform random over the vocabulary.
- C. equivalent to pure sampling.
- D. undefined.

Answer: A. Dividing logits by a shrinking temperature amplifies every gap, so the maximum-probability token's probability tends to one, greedy decoding as a limit. Uniform behavior is the high-temperature limit, and the remaining options are wrong. **বাংলা:** $T \to 0$ হলে logit-পার্থক্য বিশাল হয়ে যায়, সর্বোচ্চ টোকেনের সম্ভাবনা ১.০০ → greedy decoding

Q38. Raising the temperature above one has what effect on the next-token distribution?

- A. It sharpens the distribution toward the top token.
- B. It flattens the distribution toward uniform, increasing diversity and incoherence risk.
- C. It makes decoding deterministic.
- D. It removes the long tail entirely.

Answer: B. A temperature above one shrinks logit differences, flattening probabilities (more diverse, riskier). Sharpening is what a temperature below one does, the zero limit is deterministic, and removing the tail is truncation (top-k or top-p) rather than temperature. **বাংলা:** $T > 1$ logit-পার্থক্য চেপে দেয় → বণ্টন সমতল, বৈচিত্র্য বেশি কিন্তু অসংগতির ঝুঁকিও বেশি

Q39. What is the essential difference between top-k and top-p (nucleus) sampling?

- A. Top-k renormalizes, top-p does not.
- B. Top-p is deterministic, top-k is stochastic.
- C. Top-k keeps a fixed number of tokens, while top-p keeps the smallest set whose cumulative probability reaches the threshold, so its size adapts to the distribution.
- D. Top-k keeps a variable number, top-p keeps a fixed number.

Answer: C. Top-k keeps a fixed count regardless of shape; top-p keeps an adaptive set (narrow when confident, wide when uncertain). Both renormalize and both are stochastic, and the last option reverses the roles. **বাংলা:** top-k fixed সংখ্যা রাখে; top-p জমা সম্ভাবনা threshold ছোঁয়া পর্যন্ত adaptive সেট রাখে (নিশ্চিত হলে ছোট, দ্বিধায় বড়)

Q40. After truncating a distribution with top-k or top-p, why must you renormalize the surviving probabilities?

- A. To convert them to logits.
- B. To apply the causal mask.
- C. To increase the temperature.
- D. Because removing tokens leaves the remaining probabilities summing to less than one, so they must be rescaled to form a valid distribution.

Answer: D. Dropping the tail leaves the mass below one; dividing by the kept mass restores a valid probability distribution to sample from. The other options are unrelated steps. **বাংলা:** লেজ বাদ দিলে বাকি সম্ভাবনার যোগফল ১-এর কম হয়; রাখা মাস দিয়ে ভাগ করে আবার বৈধ বণ্টন বানাতে হয়

Q41. For factual question-answering and code extraction, which decoding setting does the lecture recommend, and why?

- A. Greedy or a temperature near zero to about 0.3, for reproducibility and the fewest hallucinated tokens.
- B. A temperature near 1.3 with top-p, for maximum creativity.

- C. Pure sampling, to explore the long tail.
- D. Top-k of 1000, to keep all tokens.

Answer: A. A low or zero temperature gives reproducible, low-hallucination output suited to factual and code tasks. High-temperature options suit creative writing, and a very large top-k effectively disables truncation. **বাংলা:** factual QA ও কোডে greedy বা T ≈ ০–০.৩ — পুনরুৎপাদনযোগ্য, কম hallucination; সৃজনশীল লেখায় বেশি T লাগে

Topic: Pretraining Systems (FLOPs, Precision, GPU, Distributed)

Q42. The 6ND rule estimates training compute. Where does the factor of 6 come from?

- A. Six graphics processing units are required.
- B. Approximately 2 floating-point operations per parameter for the forward pass plus about 4 for the backward pass, per token.
- C. Six bytes are stored per parameter.
- D. Six transformer layers.

Answer: B. The forward pass is about 2 operations per parameter and the backward pass about 4, summing to roughly 6 per parameter per token. The other options misread the 6 as hardware, storage, or depth. **বাংলা:** forward-এ ~২ আর backward-এ ~৪ অপারেশন প্রতি প্যারামিটারে প্রতি টোকেনে — মোট ~৬

Q43. Why is making a model *deeper* generally cheaper in floating-point operations than making it *wider* (larger hidden size)?

- A. Depth has no compute cost.
- B. Width scales linearly and depth quadratically.
- C. Depth scales the operation count linearly while width scales it quadratically.
- D. Both scale logarithmically.

Answer: C. Adding layers grows cost linearly, but widening grows matrix dimensions on both axes, so cost grows with the square of width. One option falsely claims depth is free, another reverses the scaling, and the last is wrong. **বাংলা:** গভীরতা (depth) FLOPs linear বাড়ায়, কিন্তু width quadratic বাড়ায় — তাই গভীর করা সস্তা

Q44. Why has nearly every large language model trained since 2021 used BF16 rather than FP16?

- A. BF16 uses fewer total bits.
- B. BF16 has more mantissa (precision) bits.
- C. BF16 is an integer format used for inference.
- D. BF16 keeps FP32's exponent range (8 bits), so it trains stably without the loss-scaling hacks FP16 needs.

Answer: D. BF16 trades mantissa precision but keeps FP32's wide exponent range, avoiding underflow and overflow and the need for loss scaling. Both formats use 16 bits, BF16 has fewer mantissa bits, and it is not an integer format. **বাংলা:** BF16 FP32-এর exponent range (৮ বিট) রাখে, তাই loss-scaling ছাড়াই স্থিরভাবে ট্রেন হয়; FP16-এর range ছোট বলে loss scaling লাগে

Q45. Why is large-scale training of a large language model practically infeasible on central processing units?

- A. Central processing units are sequential-optimized with low parallelism and bandwidth and no Tensor Cores, whereas training is memory-bound and needs massive parallel matrix multiplication.
- B. Central processing units lack floating-point units entirely.
- C. Central processing units cannot store model weights.
- D. Central processing units only support BF16.

Answer: A. Training needs the parallel matrix-multiply throughput and very high memory bandwidth that graphics processing units (with Tensor Cores) provide and central processing units lack. The other options are false. **বাংলা:** ট্রেনিং memory-bound ও বিশাল parallel matmul দরকার — CPU-তে যথেষ্ট parallelism/bandwidth/Tensor Core নেই, তাই অবাস্তব

Q46. In *tensor parallelism*, what is split across graphics processing units?

- A. The training data into different micro-batches.
- B. Individual weight matrices, sharded within a layer.
- C. Whole layers grouped into pipeline stages.
- D. Only the optimizer states.

Answer: B. Tensor parallelism shards weight matrices inside a layer (each device computes a partial result later combined). Splitting data is data parallelism, splitting whole layers is pipeline parallelism, and sharding only optimizer states is the Zero Redundancy Optimizer's job. **বাংলা:** tensor parallelism একটি স্তরের ভেতরের ওজন-ম্যাট্রিক্স ভাগ করে; ডেটা ভাগ = DP, স্তর ভাগ = PP

Q47. A 70-billion-parameter model needs about 140 gigabytes just for its BF16 weights, before activations and optimizer states. Why does this force distributed training?

- A. Because BF16 cannot be used on one device.
- B. Because the tokenizer is too large.
- C. Because 140 gigabytes exceeds a single graphics processing unit's memory, so the model and/or data must be split across many devices.
- D. Because the context window must shrink.

Answer: C. 140 gigabytes (plus activations and optimizer states two to four times the model) far exceeds one 80-gigabyte device, requiring data, tensor, or pipeline parallelism. The other options are false reasons. **বাংলা:** ১৪০ GB (+ activation ও optimizer state) এক GPU-র মেমরির অনেক বেশি, তাই DP/TP/PP দিয়ে ভাগ করতে হয়

Q48. In pure data parallelism, the optimizer states, gradients, and parameters are replicated on every device. Which technique removes this redundancy, and at what cost?

- A. Tensor parallelism, at the cost of a smaller vocabulary.
- B. Gradient accumulation, at the cost of more floating-point operations.
- C. Mixed precision, at the cost of accuracy.
- D. The Zero Redundancy Optimizer, which shards the training states across devices at the cost of more communication.

Answer: D. The Zero Redundancy Optimizer shards (rather than replicates) optimizer states, gradients, and parameters, freeing memory but adding communication. The other options address different problems. **বাংলা:** ZeRO training state replicate না করে shard করে → মেমরি বাঁচে, কিন্তু communication বাড়ে

Q49. Gradient accumulation sums gradients over several micro-batches before one optimizer step. What does it actually save, and what does it *not* save?

- A. It saves memory (fitting a large effective batch on small devices) but not compute, the total operation count is unchanged.
- B. It saves floating-point operations but not memory.
- C. It saves both memory and compute.
- D. It saves neither.

Answer: A. Accumulation simulates a large batch within limited memory; the number of operations is identical to one big batch. One option reverses this, and the others are wrong. **বাংলা:** gradient accumulation ছোট GPU-তে বড় effective batch-এর প্রভাব দেয় (মেমরি বাঁচায়), কিন্তু FLOPs একই থাকে

Q50. Training a 7-billion-parameter model with the Adam optimizer needs roughly 16 bytes per parameter (about 112 gigabytes), while *inference* in FP16 needs only about 14 gigabytes. Which statement best explains the gap?

- A. Inference uses a different architecture.
- B. Training stores working weights, gradients, an FP32 master copy, and Adam's two moment estimates per parameter, whereas inference stores only the weights.
- C. Inference uses a larger context window.
- D. Training uses INT4.

Answer: B. The roughly eightfold gap comes from gradients, the FP32 master copy, and Adam's first and second moments on top of the weights. The other options are false. **বাংলা:** ট্রেনিং-এ ওজন + গ্রেডিয়েন্ট + FP32 master + Adam-এর দুই moment লাগে; inference-এ শুধু ওজন — তাই ~৮ গুণ ফারাক

Q51. Why does the learning-rate schedule begin with a linear warmup?

- A. To save memory at the start.
- B. To increase the vocabulary gradually.
- C. Because the Adam optimizer's moment estimates and the loss surface are unreliable at step zero, so a large early learning rate could destabilize or diverge the run.
- D. To apply the causal mask slowly.

Answer: C. Early on, the optimizer's statistics are noisy; ramping the learning rate up avoids destabilizing the run. The other options are unrelated. **বাংলা:** শুরুতে Adam-এর moment ও loss-পৃষ্ঠ অনির্ভরযোগ্য, তাই বড় learning rate দিলে ট্রেনিং অস্থির হয়; warmup ধীরে বাড়িয়ে এটা এড়ায়

Topic: In-Context Learning

Q52. Few-shot prompting lets a model perform a new task from examples in the prompt without any weight update. The lecture frames this mechanism as:

- A. gradient descent performed silently at inference.
- B. permanent fine-tuning of the embedding matrix.
- C. retrieval from an external database.
- D. pattern recognition, the examples cue a task pattern the model implicitly learned during pretraining, which it then continues.

Answer: D. In-context learning is pattern completion, not gradient descent; the prompt examples help the model recognize and continue a known pattern. The other options mischaracterize it. **বাংলা:** in-context learning গ্রেডিয়েন্ট-শেখা নয়, প্যাটার্ন-চেনা — উদাহরণ দেখে মডেল pretraining-এ শেখা প্যাটার্ন চিনে চালিয়ে যায়

Q53. A team must support many ad-hoc tasks with fast iteration and only a few examples each. Compared with fine-tuning, in-context learning is preferable because:

- A. it has zero training setup and adapts per prompt, though examples consume context budget at every call.
- B. it permanently changes the model weights.
- C. it allows unlimited training examples.
- D. it has no latency or cost.

Answer: A. In-context learning needs no training infrastructure and adapts per call, with the trade-off that examples eat context tokens (cost and latency). Permanently changing weights describes fine-tuning, examples are limited by the window, and it certainly has latency and cost. **বাংলা:** in-context learning-এ ট্রেনিং সেটআপ লাগে না, প্রতি prompt-এ মানিয়ে নেয়; তবে উদাহরণ context-বাজেট খরচ করে (খরচ ও latency)

Q54. Why is the benefit of few-shot examples described as an *emergent* property of scale?

- A. Because only the tokenizer changes with scale.
- B. Because small models barely benefit from examples, while large models gain strikingly.
- C. Because few-shot learning updates weights only in large models.
- D. Because emergence reduces the irreducible loss.

Answer: B. The lecture notes few-shot gains appear strongly only as models grow large, an emergent capability. The other options mischaracterize the phenomenon. **বাংলা:** ছোট মডেল উদাহরণ থেকে সামান্যই লাভ করে, বড় মডেল অনেক বেশি — তাই few-shot ক্ষমতা scale-এর emergent বৈশিষ্ট্য

Topic: Alignment (Supervised Fine-Tuning, RLHF, DPO)

Q55. A pretrained base model, asked "What is the capital of France?", may continue with *more exam questions* rather than answering. Which statement best explains this?

- A. The base model has a corrupted tokenizer.
- B. The base model lacks an embedding matrix.
- C. A base model is a fluent completion engine trained to predict likely continuations, not yet aligned to follow instructions helpfully.
- D. The context window is too small for the question.

Answer: C. Pretraining yields a completion engine; instruction-following helpfulness comes from later alignment (supervised fine-tuning, then preference optimization). The other options are false. **বাংলা:** base model শুধু সম্ভাব্য ধারাবাহিকতা predict করে (completion engine); নির্দেশ মানা ভদ্রতা পরে alignment-এ শেখানো হয়

Q56. Why can supervised fine-tuning alone not stop a model from confidently hallucinating?

- A. It updates no parameters.
- B. It requires a reward model that blocks hallucinations.
- C. It uses Euclidean distance instead of cross-entropy.
- D. Its loss rewards fluent (likely) continuations, not factual correctness, and it treats every demonstration as correct, so false-but-fluent answers are never penalized.

Answer: D. Supervised fine-tuning is imitation with a next-token loss; without a signal for "worse" it cannot punish fluent falsehoods. It does update weights, needs no reward model, and uses cross-entropy. **বাংলা:** SFT next-token loss fluency-কে পুরস্কার দেয়, accuracy নয়; সব demonstration "ঠিক" ধরে নেয়, তাই fluent মিথ্যা শাস্তি পায় না

Q57. What do preference-based methods (reinforcement learning from human feedback, and direct preference optimization) add that supervised fine-tuning structurally lacks?

- A. A "better-versus-worse" ranking signal learned from comparisons, letting the model be pushed *away* from bad outputs.
- B. A larger vocabulary.
- C. Positional encoding.
- D. A new tokenizer.

Answer: A. Training on preferred-versus-rejected pairs supplies the ranking signal supervised fine-tuning cannot express. The other options are unrelated. **বাংলা:** preference পদ্ধতি তুলনা থেকে "ভালো বনাম খারাপ" সংকেত দেয় — SFT যা পারে না — তাই মডেলকে খারাপ আউটপুট থেকে দূরে ঠেলা যায়

Q58. In reinforcement learning from human feedback, why is the Kullback-Leibler penalty to a frozen reference model essential?

- A. It speeds up tokenization.
- B. Because the reward model is an imperfect proxy; without the leash the policy finds adversarial high-reward but degenerate outputs (reward hacking) and loses diversity (mode collapse).
- C. It increases the vocabulary during training.
- D. It replaces the need for preferences.

Answer: B. The reward model is only a proxy, so the Kullback-Leibler term anchors the policy to fluent reference behavior, preventing reward hacking and mode collapse. The other options are false. **বাংলা:** reward model অসম্পূর্ণ proxy; KL-দড়ি না থাকলে policy reward-hacking ও mode collapse করে — তাই reference-এর সাথে বেঁধে রাখা জরুরি

Q59. A common claim is that "Direct Preference Optimization has no Kullback-Leibler control." Why is this wrong?

- A. Because direct preference optimization uses a reward model after all.
- B. Because it trains on demonstrations, not preferences.

- C. Because the parameter beta is the Kullback-Leibler knob and the frozen reference policy appears explicitly in the loss.
- D. Because it removes the reference model entirely.

Answer: C. In direct preference optimization, beta controls how far the policy may drift and the reference policy appears directly in the loss, so Kullback-Leibler control is built in. It has no explicit reward model, it trains on preferences, and it keeps the reference model. **বাংলা:** DPO-তে beta-ই KL নিয়ন্ত্রক আর reference policy লসে সরাসরি থাকে — তাই "KL নিয়ন্ত্রণ নেই" কথাটা ভুল

Q60. Which statement best distinguishes the *training signal* of reinforcement learning from human feedback (with Proximal Policy Optimization) from direct preference optimization?

- A. Reinforcement learning from human feedback trains on ideal demonstrations; direct preference optimization trains a reward model.
- B. Both train on demonstrations only.
- C. Direct preference optimization requires a value/critic network and reinforcement learning from human feedback does not.
- D. Reinforcement learning from human feedback optimizes the policy with a reinforcement-learning loop against a learned reward model, while direct preference optimization optimizes preference pairs directly with a closed-form supervised loss, with no separate reward model and no reinforcement-learning loop.

Answer: D. Reinforcement learning from human feedback uses a learned reward model plus an unstable reinforcement-learning loop; direct preference optimization turns the same preference objective into a stable supervised loss with only a reference model. One option confuses both with supervised fine-tuning, another is false, and the last reverses which method needs the critic. **বাংলা:** RLHF শেখা reward model-এর বিরুদ্ধে RL-loop চালায়; DPO একই preference লক্ষ্যকে সরাসরি supervised লসে পরিণত করে — আলাদা reward model বা RL-loop লাগে না

Topic: Scaling Laws (Kaplan, Chinchilla, Deploy-Optimal)

Q61. The irreducible loss term in scaling laws represents:

- A. the entropy of natural language, a floor that no model can beat.
- B. a bug in the training code.
- C. the loss at random initialization.
- D. the optimizer's learning rate.

Answer: A. Even a perfect model faces inherent unpredictability in language, so loss cannot drop below this entropy. The other options name unrelated quantities. **বাংলা:** irreducible loss = ভাষার অন্তর্নিহিত entropy — কোনো মডেলই এর নিচে নামতে পারে না

Q62. Chinchilla's central correction to the earlier Kaplan scaling laws was that:

- A. data does not matter for large models.
- B. parameters and training tokens should scale roughly equally (about 20 training tokens per parameter).
- C. compute is irrelevant to loss.
- D. the largest possible model is always best.

Answer: B. Chinchilla showed balanced scaling (about 20 tokens per parameter), correcting Kaplan's over-emphasis on model size. Two options are simply false, and the "largest model is best" view is the one Chinchilla overturned. **বাংলা:** Chinchilla দেখাল প্যারামিটার ও টোকেন প্রায় সমানভাবে বাড়াতে হয় (~২০ টোকেন/প্যারামিটার) — Kaplan-এর "শুধু বড় মডেল" ভুল সংশোধন

Q63. Why is GPT-3 (175 billion parameters, about 300 billion tokens) cited as an example of an *undertrained* model?

- A. It saw far more tokens than Chinchilla prescribes.
- B. It used too small a vocabulary.
- C. By the Chinchilla ratio it should have seen roughly 3 to 4 trillion tokens, so it received too few tokens for its size.

- D. It was trained with FP16 instead of BF16.

  Answer: C. At about 1.7 tokens per parameter versus the roughly 20 Chinchilla prescribes, GPT-3 was undertrained for its size. One option reverses this, and the others are irrelevant. **বাংলা:** Chinchilla অনুযায়ী GPT-3-এর ৩–৪ trillion টোকেন দরকার ছিল, কিন্তু পেয়েছে ~৩০০B — তাই আকারের তুলনায় undertrained

Q64. Why do modern production models (LLaMA 3, Mistral, and others) deliberately train *past* the Chinchilla compute-optimal point, sometimes more than 100 tokens per parameter?

- A. Because Chinchilla recommends it.
- B. Because larger models are always cheaper to serve.
- C. Because it reduces the irreducible loss.
- D. Because inference cost is paid on every request forever while training is a one-time cost, so overtraining a *smaller* model makes it more capable at the same inference cost.

  Answer: D. Compute-optimal is not deploy-optimal: a small, heavily-trained model minimizes the lifetime inference cost that dominates a deployed model. One option is false, another reverses the logic, and reducing the irreducible loss is impossible. **বাংলা:** inference খরচ বারবার লাগে, training একবার — তাই ছোট মডেলকে বেশি ট্রেন করলে একই inference খরচে বেশি সক্ষম হয় (compute-optimal $\neq$ deploy-optimal)

Q65. Why did Kaplan's laws *overestimate* the optimal model size?

- A. They assumed fixed data and training settings, which underestimated how much data large models need.
- B. They ignored the number of parameters.
- C. They used Chinchilla's 400-model sweep.
- D. They measured capabilities instead of loss.

  Answer: A. Holding data and schedule fixed credited model size too much and tokens too little. The other options are false. **বাংলা:** Kaplan স্থির ডেটা/সেটিং ধরে নিয়েছিল, তাই বড় মডেলের ডেটা-চাহিদা কম আঁচ করে আকার বেশি বলেছিল

Topic: Evaluation (Perplexity, BLEU/ROUGE, Pass@k, Benchmarks, Safety)

Q66. Perplexity is best described as:

- A. accuracy on the MMLU benchmark.
- B. exponentiated cross-entropy, the model's average uncertainty, or "effective branching factor," per token.
- C. a tool-call success rate.
- D. a safety metric for jailbreak resistance.

  Answer: B. Perplexity equals the exponential of cross-entropy; it measures next-token predictive uncertainty (lower is better). The other options name different metrics. **বাংলা:** perplexity = exp(cross-entropy) — প্রতি টোকেনে গড় অনিশ্চয়তা ("effective branching factor"); কম = ভালো

Q67. Why is it meaningless to compare the perplexities of two models that use *different* tokenizers?

- A. Because perplexity ignores the model entirely.
- B. Because perplexity requires BF16.
- C. Because perplexity is computed per token, and different tokenizers split the same text into different token counts, so the per-token uncertainty is not comparable.
- D. Because one model must be larger.

  Answer: C. Perplexity is tokenizer-dependent; differing token counts make per-token numbers incomparable across vocabularies. The other options are false. **বাংলা:** perplexity প্রতি-টোকেন হিসাব, আর ভিন্ন টোকেনাইজার একই টেক্সটকে ভিন্ন সংখ্যক টোকেনে ভাঙে — তাই তুলনা অর্থহীন

Q68. Which statement best captures *why* a model with very low perplexity can still be a poor assistant?

- A. Low perplexity forces high latency.
- B. Low perplexity means the vocabulary is too small.
- C. Perplexity directly measures user satisfaction.

- D. Perplexity only measures next-token prediction and is blind to reasoning, instruction-following, tool use, factuality, and safety.

  Answer: D. Predicting tokens well does not guarantee good multi-turn, tool-using, or factual behavior. The latency and vocabulary claims are false, and the user-satisfaction claim contradicts the point. **বাংলা:** perplexity শুধু next-token prediction মাপে — reasoning, instruction-following, tool-use, factuality, safety মাপে না, তাই কম perplexity-তেও সহকারী দুর্বল হতে পারে

Q69. BLEU and ROUGE differ primarily in that:

- A. BLEU measures n-gram precision (machine translation) while ROUGE measures n-gram recall (summarization).
- B. BLEU is for images and ROUGE for code.
- C. BLEU needs no reference text.
- D. ROUGE ignores n-grams entirely.

  Answer: A. BLEU divides matches by what was generated (precision); ROUGE divides by what the reference contains (recall). The other options are false. **বাংলা:** BLEU = precision (যা বানালাম তার কতটা ঠিক, অনুবাদ); ROUGE = recall (reference-এর কতটা ধরলাম, সারাংশ)

Q70. The same reference "the quick brown fox" (4 unigrams) and candidate "the quick fox" (3 matched unigrams) give a BLEU-style precision of $3/3 = 1.00$ but a ROUGE-1 recall of $3/4 = 0.75$. Why do the two scores differ?

- A. Because BLEU and ROUGE use different references.
- B. Because BLEU divides by the number of generated n-grams while ROUGE divides by the number of reference n-grams.
- C. Because ROUGE counts characters and BLEU counts words.
- D. Because BLEU applies no matching at all.

  Answer: B. Precision divides by what you said (3), recall by what you should have said (4), the same matches with different denominators. The other options are false. **বাংলা:** BLEU ভাগ করে generated সংখ্যা (৩) দিয়ে, ROUGE ভাগ করে reference সংখ্যা (৪) দিয়ে — একই মিল, ভিন্ন হর

Q71. Pass@k is the metric of choice for which task, and what does it measure?

- A. Translation; n-gram overlap with a reference.
- B. Summarization; recall of reference words.
- C. Code generation; the probability that at least one of k sampled programs passes hidden unit tests.
- D. Toxicity; the fraction of harmful outputs.

  Answer: C. Pass@k measures whether at least one of k code samples passes unit tests (for example, HumanEval). The other options name different tasks and metrics. **বাংলা:** Pass@k কোড জেনারেশনের জন্য — k-টা নমুনার অন্তত একটা hidden unit test পাস করার সম্ভাবনা

Q72. If a model solves a task 30 percent of the time on a single attempt (Pass@1 = 0.30), why is Pass@5 much higher (about 0.92)?

- A. Because the model learns between attempts.
- B. Because Pass@5 ignores failed attempts.
- C. Because Pass@5 changes the unit tests.
- D. Because retrying independently raises the chance that at least one of five samples succeeds.

  Answer: D. Independent retries compound the chance of at least one success, so Pass@5 far exceeds Pass@1. There is no weight update between samples, and the remaining options misdescribe the metric. **বাংলা:** স্বাধীনভাবে বারবার চেষ্টা করলে অন্তত একবার সফল হওয়ার সম্ভাবনা বাড়ে — তাই Pass@5 $\gg$ Pass@1

Q73. A model scores very high on MMLU but fails repeatedly when your product asks it to call application programming interfaces in the correct order. Which benchmark targets *this* failure, and what does a low score indicate?

- A. The tau-squared benchmark (Tau-2); unreliable multi-step tool calling, which is the dominant real-world failure mode rather than a reasoning deficit.
- B. BLEU; poor translation quality.

- C. Perplexity; high next-token uncertainty.
- D. ROUGE; low summarization recall.

Answer: A. The tau-squared benchmark evaluates agentic, multi-step tool use across domains such as retail, airline, and telecom; a low score flags tool-call unreliability. The other options measure unrelated abilities. **বাংলা:** τ^2-bench এজেন্ট-সুলভ multi-step tool-use মাপে; কম স্কোর মানে tool-call অনির্ভরযোগ্য — reasoning-এর দোষ নয়

Q74. Benchmark "contamination" inflates scores. Whose defect is it, and what is the fix?

- A. A model architecture defect; fix it by adding layers.
- B. A training-data defect (test data leaked into training data); fix it by decontaminating or deduplicating the corpus against evaluation sets.
- C. A tokenizer defect; fix it by retraining the tokenizer.
- D. A sampling defect; fix it by lowering temperature.

Answer: B. Contamination is a data-side problem, evaluation items in the training corpus, fixed by decontamination and deduplication. The other options misplace the cause. **বাংলা:** contamination ডেটা-সমস্যা (পরীক্ষার প্রশ্ন ট্রেনিং-ডেটায়), মডেলের দোষ নয়; সমাধান কর্পাস decontaminate/dedup করা

Q75. Why does the lecture conclude that "no single number summarizes large-language-model quality"?

- A. Because perplexity alone is sufficient.
- B. Because all benchmarks measure the same thing.
- C. Because different metrics (intrinsic, task-level, capability, safety) capture weakly-correlated aspects, so evaluation must be multi-dimensional and tested in the real workflow.
- D. Because evaluation should be done only once before release.

Answer: C. Perplexity, MMLU, tool-use, and safety scores correlate weakly, so quality requires multi-dimensional evaluation in the actual domain. The other options contradict the lecture. **বাংলা:** intrinsic, task-level, capability, safety — মেট্রিকগুলো দুর্বলভাবে সম্পর্কিত, তাই বহুমাত্রিক ও বাস্তব-workflow-এ মূল্যায়ন লাগে; একটা সংখ্যা যথেষ্ট নয়

Answer Key

Q	Ans	Q	Ans	Q	Ans	Q	Ans	Q	Ans
1	A	16	D	31	C	46	B	61	A
2	B	17	A	32	D	47	C	62	B
3	C	18	B	33	A	48	D	63	C
4	D	19	C	34	B	49	A	64	D
5	A	20	D	35	C	50	B	65	A
6	B	21	A	36	D	51	C	66	B
7	C	22	B	37	A	52	D	67	C
8	D	23	C	38	B	53	A	68	D
9	A	24	D	39	C	54	B	69	A
10	B	25	A	40	D	55	C	70	B
11	C	26	B	41	A	56	D	71	C
12	D	27	C	42	B	57	A	72	D
13	A	28	D	43	C	58	B	73	A
14	B	29	A	44	D	59	C	74	B
15	C	30	B	45	A	60	D	75	C

Answer Distribution

Letter	Count
A	19
B	19
C	19
D	18

Total = 75. A/B/C/D = 19/19/19/18 — balanced across all four letters (no clustering).

বাংলা ব্যাখ্যা: উত্তরগুলো A/B/C/D-তে প্রায় সমানভাবে ছড়ানো (≈২৫% করে) — কোনো একটি অক্ষরে গুচ্ছ নেই