NVIDIA.NCA-GENM.v2026-11-05.q26

Exam Code:	NCA-GENM
Exam Name:	NVIDIA Generative AI Multimodal
Certification Provider:	NVIDIA
Free Question Number:	26
Version:	v2026-11-05
# of views:	114
# of Questions views:	265
https://www.freecram.net/torrent/NVIDIA.NCA-GENM.v2026-11-05.q26.html	

NEW QUESTION: 1
You are developing a ML model for image classification. You have a dataset with 10,000 images of cats, dogs and birds. Which of the following ML models would be the most appropriate choice for this task?
A. Logistic Regression
B. K-Means Clustering
C. Linear Regression
D. Convolutional Neural Network (CNN)
Answer: (SHOW ANSWER)
CNNs are the standard architecture for image classification because their convolutional layers exploit the spatial locality and translation invariance inherent to image data: learned filters detect local patterns (edges, textures, shapes) that compose hierarchically into higher-level features (parts, objects) as depth increases, without requiring the manual feature engineering that traditional models would need to reach comparable accuracy on raw pixel data. Pooling layers further provide a degree of spatial invariance, and parameter sharing across the image keeps the model tractable relative to a fully connected network operating on raw pixels.
Logistic Regression (A) is a linear classifier that operates on flattened feature vectors; applied directly to raw pixels of a 3-class image problem, it cannot capture the non-linear spatial structure needed to separate cats, dogs, and birds reliably, though it could serve as a baseline or as the final classification head atop CNN- extracted features. K-Means (B) is an unsupervised clustering algorithm - inappropriate here because the task is supervised classification with labeled classes. Linear Regression (C) predicts continuous outputs and is not designed for categorical class prediction at all.
For 10,000 labeled images, a CNN (potentially fine-tuned from a pretrained backbone via transfer learning, given the modest dataset size) is the appropriate and industry-standard choice.
Reference: Core Machine Learning and AI Knowledge domain - CNN architecture for image classification.

NEW QUESTION: 2
What are some methods to overcome limited throughput between CPU and GPU?
A. Increase the clock speed of the CPU.
B. Increase the number of CPU cores.
C. Using techniques like memory pooling.
D. Upgrade the GPU to a higher-end model.
Answer: (SHOW ANSWER)
CPU-GPU data transfer over the PCIe (or NVLink) bus is frequently a throughput bottleneck in ML pipelines, particularly when small, frequent transfers dominate rather than large batched ones - each transfer incurs fixed overhead independent of data size, so many small transfers waste a disproportionate amount of time on overhead rather than useful data movement. Memory pooling techniques - pre-allocating and reusing pinned (page-locked) host memory buffers rather than repeatedly allocating and freeing memory for each transfer - reduce this overhead and enable faster, more predictable DMA transfers between host and device. Related software-level techniques include using CUDA streams to overlap data transfer with computation (so the GPU keeps computing while the next batch transfers in the background), and batching transfers to amortize fixed per-transfer overhead across more data.
Options A, B, and D each propose hardware upgrades that address a different bottleneck than the one described: increasing CPU clock speed (A) or core count (B) improves CPU-side compute throughput, not the data-transfer bandwidth or latency between CPU and GPU specifically. Upgrading the GPU (D) increases GPU compute capability but does nothing to address a PCIe/interconnect bandwidth limitation - a faster GPU sitting idle waiting for data across the same bottlenecked bus would not see meaningfully improved end- to-end throughput. The question specifically asks about *throughput between* CPU and GPU, which points to interconnect/transfer-management optimization rather than raw compute upgrades on either side.
Reference: Performance Optimization domain - CPU-GPU data transfer optimization, pinned memory, memory pooling, CUDA streams.

NEW QUESTION: 3
Which technique involves leveraging pre-trained models to achieve efficient results with less data and computation?
A. State management and composition
B. Transfer learning
C. Prompt engineering
D. Neural network integration
Answer: (SHOW ANSWER)
Transfer learning takes a model already trained on a large, general-purpose dataset (e.g., ImageNet for vision, or a large text corpus for language models) and adapts it to a new, typically smaller and more specific target task - either by fine-tuning some or all of the pretrained weights, or by freezing the pretrained backbone and training only new task-specific layers on top. Because

the pretrained model has already learned general- purpose, reusable features (edge and texture detectors in early CNN layers, syntactic and semantic structure in language model layers), the target task requires substantially less labeled data and less compute than training a comparable model from random initialization.
Prompt engineering (C) is a related but distinct technique specific to large language and generative models: it adapts a *frozen* pretrained model's behavior through the design of the input prompt alone, without any weight updates - a lighter-weight technique than transfer learning, applicable only where a sufficiently capable pretrained model already exists. Options A and D are not standard, well-defined ML techniques matching this description; "state management and composition" and "neural network integration" are generic software-engineering-sounding terms without a specific technical meaning in this context, making them straightforward distractors to eliminate.
Reference: Core Machine Learning and AI Knowledge domain - transfer learning, fine-tuning, pretrained backbones.

NEW QUESTION: 4
Which technique is commonly used to speed up AI model training and inference on hardware accelerators?
A. Quantization
B. Data augmentation
C. Model enlargement
D. Dropout
Answer: (SHOW ANSWER)
Quantization reduces the numerical precision used to represent model weights and activations - for example, converting FP32 weights to INT8 - which decreases memory bandwidth requirements and allows hardware accelerators (GPU Tensor Cores, dedicated INT8 inference engines) to execute more operations per cycle, directly speeding up both training (in its mixed-precision form) and, especially, inference. Post-training quantization and quantization-aware training are the two dominant approaches, with the latter simulating quantization effects during training to better preserve accuracy at reduced bit-widths. NVIDIA's TensorRT relies heavily on quantization (alongside layer fusion and kernel auto-tuning) to accelerate deployed inference.
The distractors describe techniques that serve entirely different purposes: data augmentation (B) increases training data diversity to improve generalization, not computational speed - it typically adds preprocessing overhead rather than reducing it. Model enlargement (C) does the opposite of speeding up computation; larger models require more FLOPS and memory, increasing latency. Dropout (D) is a regularization technique applied during training to prevent overfitting by randomly zeroing activations - it has no role in inference- time speed (and is typically disabled at inference) and does not meaningfully accelerate training compute either.
Quantization is frequently paired with pruning and kernel/operator fusion as the three core techniques for hardware-accelerated performance optimization.

Reference: Performance Optimization domain - quantization, TensorRT, inference acceleration techniques.

NEW QUESTION: 5
What is a main application of Triton Inference Server?
A. Triton Server can be used to generate images from pure noise.
B. Triton Server can be used to deploy AI models on the GPU only.
C. Triton Server can be used to execute GPU-accelerated graph analysis with cuGraph.
D. Triton Server can be used to deploy neural networks from various frameworks.
Answer: (SHOW ANSWER)
NVIDIA Triton Inference Server is an open-source model-serving platform designed to standardize production deployment across heterogeneous model formats and hardware backends. Its defining capability is multi-framework support: a single Triton instance can concurrently serve models trained in TensorFlow, PyTorch, ONNX Runtime, TensorRT, OpenVINO, and custom Python/C++ backends, exposing them through unified HTTP/REST and gRPC inference APIs. This eliminates the need for framework-specific serving stacks and lets teams standardize deployment infrastructure independent of how a given model was trained.
Option B is a common but incorrect assumption - Triton explicitly supports both GPU and CPU inference, which matters for cost-sensitive or edge deployments where GPU availability is limited.
Option C confuses Triton with cuGraph, a separate RAPIDS library for GPU-accelerated graph analytics (unrelated to model serving), and option A describes a generative task (denoising diffusion), which is a *model capability*, not a Triton *server* function - Triton can host such a model, but "generating images from noise" is not what the server itself does.
Triton also provides dynamic batching, concurrent model execution, model ensembling (chaining pre/post- processing with inference), and metrics export - features tested elsewhere in the Software Development and Engineering and Performance Optimization domains.
Reference: Software Development and Engineering domain - Triton Inference Server, multi-framework model deployment.

NEW QUESTION: 6
You are conducting an experiment to evaluate the performance of different AI models. What is the purpose of AI model evaluation?
A. To determine the best AI model architecture.
B. To determine the ethical implications of AI model usage.
C. To study the impact of AI models on human behavior.
D. To analyze the cost-effectiveness of AI model development.
Answer: (SHOW ANSWER)
In the context described - comparing the performance of different AI models against each other - the purpose of evaluation is to systematically measure each candidate model's performance on relevant metrics (accuracy, F1, WER, BLEU, latency, or task-specific measures) using held-out data, in order to determine which architecture, configuration, or training approach performs best

for the target task. This is the immediate, operational purpose of the evaluation experiment being described: comparative performance measurement that informs model-selection decisions.
The other options describe legitimate but distinct concerns that belong to different domains within a full AI development lifecycle rather than to the "evaluate performance of different models" activity specifically described in the question: ethical implications (B) fall under Trustworthy AI governance - fairness audits, bias assessments, and impact reviews - conducted alongside, not as a substitute for, performance evaluation.
Studying impact on human behavior (C) belongs to human-computer interaction or longitudinal deployment studies, a separate research activity from a controlled model-comparison experiment.
Cost-effectiveness analysis (D) is a business/engineering consideration weighing performance gains against compute, infrastructure, and development cost - relevant to deployment decisions, but not what "evaluating model performance" itself measures.
Rigorous evaluation in this context requires a held-out test set the models were not trained or tuned on, appropriate metric selection for the task, and often statistical significance testing when comparing close results.
Reference: Experimentation domain - model evaluation as comparative performance measurement.

NEW QUESTION: 7
In the development of Trustworthy AI, what is the significance of 'Certification' as a principle?
A. It requires AI systems to be developed with an ethical consideration for societal impacts.
B. It ensures that AI systems are transparent in their decision-making processes.
C. It mandates that AI models comply with relevant laws and regulations specific to their deployment region and industry.
D. It involves verifying that AI models are fit for their intended purpose according to regional or industry- specific standards.
Answer: (SHOW ANSWER)
Within Trustworthy AI frameworks, "Certification" is best understood as the formal verification process confirming that an AI system meets defined standards of fitness-for-purpose - whether those standards are set by regulatory bodies, industry consortia, or internal governance frameworks - for the specific context in which the system will be deployed. This is distinct from, though related to, the broader Trustworthy AI principles of ethics (option A), transparency (option B), and legal compliance (option C): certification is the
verification mechanism that attests a system satisfies applicable standards, rather than being one of those underlying values itself.
The distinction between C and D is subtle and worth being precise about: C describes compliance as an obligation ("must follow laws and regulations"), while D describes certification as a verification activity ("confirming fitness according to standards") - certification is the audit/attestation process, and compliance is one of the things that process may confirm. A system can be legally compliant without having undergone formal certification, and certification processes often assess criteria broader than legal compliance alone, including performance benchmarks,

robustness testing, and domain-appropriate validation (e.g., clinical validation standards for a medical imaging model).
In practice, certification connects Trustworthy AI to concrete deployment gates: a healthcare AI model, for instance, may require certification against medical device standards before clinical use - the verification step, not merely the legal requirement, is the "Certification" principle's substance.
Reference: Trustworthy AI domain - certification, compliance, and verification as distinct governance mechanisms.

NEW QUESTION: 8
In a Generative Adversarial Network (GAN), what is the role of the discriminator?
A. To generate new data based on the training set.
B. To distinguish between real and generated data.
C. To optimize the training process.
D. To calculate the loss function and update the generator.
Answer: B (LEAVE A REPLY)
A GAN's discriminator is a binary classifier trained to distinguish real samples (drawn from the actual training data) from fake samples (produced by the generator), outputting a probability that a given input is real rather than generated. This is the discriminator's entire function - it never generates data itself. The generator, by contrast, takes random noise as input and learns to produce increasingly realistic synthetic samples, with the explicit goal of fooling the discriminator into classifying its outputs as real.
Training proceeds as a minimax adversarial game: the discriminator's weights are updated to improve its ability to correctly classify real vs. fake, while the generator's weights are updated (using gradients that flow back through the discriminator) to make its outputs harder for the discriminator to detect as fake. As training progresses, both networks improve in tandem, ideally converging to a point where the generator produces samples statistically indistinguishable from real data and the discriminator can no longer reliably tell them apart (outputting close to 50% confidence either way).
Option A describes the generator's role, not the discriminator's - a common point of confusion this question is testing directly. Option C is too vague to describe either network's specific function precisely. Option D conflates the discriminator with the general backpropagation/optimization process; while the discriminator's output does supply the gradient signal used to update the generator, "calculating the loss function and updating the generator" overstates and mischaracterizes the discriminator's role as a classifier.
Reference: Core Machine Learning and AI Knowledge domain - GAN architecture, discriminator vs.
generator roles.

NEW QUESTION: 9
In the transformer architecture, what is the purpose of positional encoding?
A. To encode the semantic meaning of each token in the input sequence.

B. To add information about the order of each token in the input sequence.
C. To remove redundant information from the input sequence.
D. To encode the importance of each token in the input sequence.
Answer: (SHOW ANSWER)
Unlike recurrent architectures, which process tokens sequentially and thereby inherently encode order through the sequence of computation, the transformer's self-attention mechanism processes all tokens in parallel and is permutation-invariant by construction - attention scores between tokens do not inherently depend on their position in the sequence. Positional encoding solves this by injecting explicit information about each token's position into its input representation, typically by adding a positional vector (computed via fixed sinusoidal functions in the original "Attention Is All You Need" formulation, or learned as trainable embeddings in many modern variants) to the token's embedding before it enters the attention layers. Without this, "the cat sat on the mat" and "the mat sat on the cat" would be indistinguishable to the self-attention mechanism, since the same set of token embeddings would be processed identically regardless of order.
Semantic meaning (option A) is the role of the token embeddings themselves, learned separately from positional information - the two are combined (typically summed) but serve distinct purposes. Positional encoding does not remove information (C); it adds it. And while attention weights do effectively encode a learned notion of token importance relative to a query (option D), that importance-weighting mechanism is a separate, downstream function of the attention layers, not the role of positional encoding itself, which only supplies order information as an input feature.
Reference: Core Machine Learning and AI Knowledge domain - transformer architecture, self-attention, positional encoding.

NEW QUESTION: 10
You are developing a GenAI-Multimodal system that uses data from various sources. What is one potential issue you need to consider in relation to bias in data?
Freecram.net
A. The data used to train the AI system may not be representative of the population it is intended to serve.
B. Bias in data is irrelevant as long as the AI system produces accurate predictions.
C. Bias in data can only be addressed after the AI system has been deployed.
D. Bias in data is not a concern for AI systems as they are designed to be neutral and objective.
Answer: (SHOW ANSWER)
Representativeness bias occurs when a training dataset systematically over- or under-samples subpopulations relative to the population the deployed system will actually encounter - for example, a facial recognition dataset skewed toward lighter-skinned faces, or a multimodal medical dataset drawn predominantly from one demographic group. Because models learn statistical patterns from their training distribution, an unrepresentative dataset produces a model whose accuracy, calibration, and fairness properties degrade for underrepresented groups, even when aggregate accuracy metrics look acceptable.

This is precisely why aggregate accuracy is an insufficient safeguard: option B's framing - that bias doesn't matter "as long as predictions are accurate" - conflates overall accuracy with subgroup accuracy, and a model can post strong aggregate numbers while systematically failing specific populations. Option D is factually false; AI systems have no inherent neutrality - they inherit and can amplify whatever patterns (including societal biases) exist in their training data and objective function. Option C is also incorrect:
mitigating representativeness bias is significantly cheaper and more effective when addressed at the data- collection and curation stage - through stratified sampling, bias audits, and diverse data sourcing - than after deployment, when it becomes a retraining and remediation problem, and by then real-world harm may have already occurred.
Reference: Trustworthy AI domain - data representativeness, fairness, bias identification and mitigation.

NEW QUESTION: 11
In large-language models, what is the purpose of the attention mechanism?
A. To measure the importance of the words in the output sequence.
B. To assign weights to each word in the input sequence.
C. To determine the order in which words are generated.
D. To capture the order of the words in the input sequence.
Answer: (SHOW ANSWER)
The attention mechanism computes a set of weights over the tokens in the input (or context) sequence for each step of processing, reflecting how relevant each input token is to the computation currently being performed
- for instance, how relevant each word in a source sentence is to correctly translating a given target word, or how relevant each prior token is to predicting the next one in an autoregressive model. Mechanically, this is computed via query, key, and value projections: a query (representing the current focus) is compared against keys (representing each input token) to produce attention scores, which are normalized (typically via softmax) into weights and used to compute a weighted sum over the corresponding values - allowing the model to dynamically focus more on relevant tokens and less on irrelevant ones, rather than treating all input tokens with equal importance.
Option D describes positional encoding's role (covered directly in an earlier question in this set) - capturing token order - which is a distinct mechanism from attention; attention operates on token *content and relevance*, while positional encoding separately supplies *order* information as an input feature, since self- attention itself is permutation-invariant without it. Option A misdirects the weighting toward the output sequence specifically, when attention weights are computed primarily over the input/context tokens being attended to. Option C describes the decoding/generation procedure (autoregressive sampling), not attention's mechanism.
Reference: Core Machine Learning and AI Knowledge domain - transformer attention mechanism vs.
positional encoding.

NEW QUESTION: 12
For building a zero-shot image classification pipeline, what could be a crucial step in the process?
A. Focusing on enhancing the resolution and quality of images before classification.
B. Manually labeling each image in the dataset for precise classification.
C. Using a model like CLIP for encoding both images and their textual descriptions into a shared representation space for comparison.
D. Designing an algorithm to replace the need for textual descriptions in the classification process.
Answer: (SHOW ANSWER)
Zero-shot image classification, by definition, requires classifying images into categories the model was never explicitly trained to recognize, with no task-specific labeled examples. CLIP-style models enable this by encoding both images and candidate text labels (e.g., "a photo of a {class}") into a shared embedding space; classification then reduces to a similarity comparison - computing cosine similarity between the image embedding and each candidate text embedding and selecting the closest match. This is the crucial architectural step: without a shared embedding space linking visual and textual semantics, there is no mechanism to generalize to unseen classes using only their names or descriptions.
Option B directly contradicts the "zero-shot" premise - manual labeling of the target dataset is precisely what zero-shot classification is designed to avoid; if labels were being collected for the target classes, the task would be standard supervised classification, not zero-shot. Option A (image enhancement) may marginally help downstream accuracy but is not the crucial, defining step. Option D is incoherent with how CLIP-style zero-shot classification actually works - the textual description of each candidate class is the essential input that makes zero-shot generalization possible; eliminating it would remove the mechanism entirely, not improve it.
Reference: Multimodal Data domain - zero-shot classification via shared embedding spaces (CLIP).

NEW QUESTION: 13
In multimodal machine learning, what does 'early fusion' refer to?
A. Integrating different modalities at the beginning of the model pipeline.
B. Ignoring certain modalities and only using one modality for analysis and prediction.
C. Training separate models for each modality and then combining their predictions.
D. Implementing the model in the early stages of development of the ML solution.
Answer: (SHOW ANSWER)
Early fusion concatenates or otherwise combines raw or lightly processed features from each modality before they enter the main model pipeline, producing a single joint input representation that a downstream network learns from jointly. This contrasts with late fusion (option C), where separate unimodal models process each modality independently and their outputs - logits, embeddings, or decisions - are combined only at the end, and with intermediate/hybrid fusion, where combination happens at one or more intermediate feature layers.

Early fusion's advantage is that it allows the model to learn cross-modal correlations from the earliest layers, potentially capturing low-level interactions that later fusion stages would miss. Its disadvantage is sensitivity to modality-specific noise, differing sampling rates, and missing modalities: if one input stream is corrupted or absent, the joint representation degrades more severely than in late fusion, where the surviving modality's model can still function independently. Option B describes unimodal reduction, not fusion at all, and option D confuses a data-processing strategy with a project-management timeline - an easy distractor to eliminate. Exam questions frequently test the ability to distinguish early, late, and hybrid fusion by identifying *where* in the pipeline combination occurs, so anchor your answer to pipeline stage rather than performance characteristics.
Reference: Multimodal Data domain - fusion strategies (early, late, hybrid/intermediate).

NEW QUESTION: 14
What characteristic of autoencoders makes them suitable for anomaly detection?
A. Their capacity to learn a compressed representation of the data.
B. Their ability to classify images with high accuracy.
C. Their function in enhancing the quality of images.
D. Their capability to predict future outcomes based on past data.
Answer: A (LEAVE A REPLY)
An autoencoder learns to compress input data into a lower-dimensional latent (bottleneck) representation via its encoder, then reconstruct the original input from that representation via its decoder, trained by minimizing reconstruction error on normal data. Because the model is optimized specifically to reconstruct patterns it has seen frequently during training, it becomes proficient at compressing and reconstructing "normal" instances but performs poorly - producing high reconstruction error - on inputs that deviate structurally from the training distribution, i.e., anomalies. Thresholding reconstruction error thus provides a natural, unsupervised anomaly score without requiring labeled anomalous examples, which are often scarce or unavailable in real- world settings.
This mechanism is the operative characteristic tested here, not classification accuracy (B, which describes a supervised discriminative task the autoencoder is not directly trained for), image enhancement (C, a description closer to denoising autoencoders' side effect rather than the core anomaly-detection mechanism), or forecasting (D, which describes sequence models like RNNs/LSTMs applied to time series, a different architecture family and objective).
Variants such as variational autoencoders (VAEs) extend this idea probabilistically, and in multimodal settings, cross-modal autoencoders can flag anomalies where reconstruction fails to reconcile one modality given another.
Reference: Core Machine Learning and AI Knowledge domain - autoencoders, latent representations, reconstruction-error-based anomaly detection.

NEW QUESTION: 15
Hyperparameter tuning is used for what purpose in machine learning experimentation?

A. Adjusting the weights and biases of a neural network to optimize its performance.
B. Selecting the best ML algorithm for a given task.
C. Collecting and preprocessing data to improve the accuracy of the model.
D. Selecting the optimal values for non-trainable parameters, such as learning rate or batch size.
Answer: D (LEAVE A REPLY)
Hyperparameters are configuration values set *before* training begins and are not updated by the optimization process itself - learning rate, batch size, number of layers, regularization strength, and number of training epochs are canonical examples. Hyperparameter tuning is the systematic search for the combination of these values that yields the best model performance on a validation set, using strategies such as grid search, random search, or more sample-efficient approaches like Bayesian optimization and population-based training.
This is explicitly distinct from option A, which describes the *training* process itself - weights and biases are trainable parameters, updated automatically via backpropagation and gradient descent, not selected through hyperparameter search. Option B describes algorithm selection, a higher-level modeling decision that may precede hyperparameter tuning but is not what tuning itself accomplishes (you tune hyperparameters
within a chosen algorithm/architecture). Option C describes data engineering work that happens upstream of model training entirely, unrelated to parameter search.
In practice, hyperparameter tuning requires careful experimental design to avoid overfitting to the validation set - techniques like k-fold cross-validation, held-out test sets, and tracking tools (e.g., experiment trackers logging each trial's configuration and resulting metric) are standard practice, connecting this topic directly to the Experimentation domain's broader emphasis on rigorous, reproducible model evaluation.
Reference: Experimentation domain - hyperparameter tuning, search strategies, validation methodology.

NEW QUESTION: 16
You are working with a large dataset and want to visualize the distribution of a continuous variable. Which type of data visualization would be most appropriate?
A. Histogram chart
B. Bar chart
C. Line chart
D. Pie chart
Answer: (SHOW ANSWER)
A histogram bins a continuous variable into contiguous intervals and plots the frequency (or density) of observations falling into each bin, making it the standard tool for visualizing the shape of a continuous distribution - skewness, modality, spread, and outliers are all immediately visible. This distinguishes it from a bar chart (B), which is designed for discrete or categorical variables where bars are separated and ordering is often arbitrary; applying a bar chart to continuous data loses the notion of a numeric scale between categories.

A line chart (C) is appropriate for showing trends of a variable across an ordered sequence, typically time, not for summarizing the overall shape of a value distribution. A pie chart (D) shows proportions of a whole across categorical segments and becomes visually unreadable and statistically meaningless for continuous data with many possible values.
In practice, histogram bin width is a critical hyperparameter: too few bins oversmooth the distribution and hide multimodality, while too many bins introduce noise. Tools like Freedman-Diaconis or Sturges' rule provide principled starting points, and kernel density estimates (KDE) are often overlaid as a smoothed alternative when bin-width sensitivity is a concern.
Reference: Data Analysis and Visualization domain - chart selection by variable type (continuous vs.
categorical vs. temporal).

Valid NCA-GENM Dumps shared by ExamDiscuss.com for Helping Passing NCA-GENM Exam! ExamDiscuss.com now offer the **newest NCA-GENM exam dumps**, the ExamDiscuss.com NCA-GENM exam **questions have been updated** and **answers have been corrected** get the **newest** ExamDiscuss.com NCA-GENM dumps with Test Engine here: https://www.examdiscuss.com/NVIDIA/exam/NCA-GENM/premium/ (**58** Q&As Dumps, **35%OFF Special Discount Code: freecram**)

NEW QUESTION: 17
Which of the following tasks can be performed using the transformer LLM encoder model?
A. Semantic analysis
B. Generating code
C. Image generation
D. Speech recognition
Answer: (SHOW ANSWER)
Encoder-only transformer architectures (BERT and its derivatives being the canonical examples) process an entire input sequence bidirectionally, attending to both left and right context simultaneously, and output contextualized representations for each token - well suited to understanding tasks that require a rich representation of input meaning without generating new sequential output. Semantic analysis - including sentiment classification, named entity recognition, semantic similarity, and text classification more broadly - is a natural fit, typically implemented by taking the encoder's output representation (often a special [CLS] token embedding) and feeding it to a lightweight classification head.
The other options require architectures the encoder-only design is not built for: generating code (B) and open- ended text generation generally require autoregressive, causal (decoder-only or encoder-decoder) architectures that predict tokens sequentially, since encoder-only models produce a fixed set of contextualized representations rather than generating novel token sequences left-to-right. Image generation (C) requires an entirely different model family (diffusion

models, GANs, or autoregressive image models), unrelated to text- encoder architectures.
Speech recognition (D) requires an audio-processing pipeline - typically a CTC-based or encoder-decoder acoustic model operating on spectrogram or waveform input - not a text-only transformer encoder.
Reference: Core Machine Learning and AI Knowledge domain - transformer encoder vs. decoder vs.
encoder-decoder architectures and their respective task fits.

NEW QUESTION: 18
How does the batch size influence VRAM consumption during inference with ML models on GPUs?
A. The batch size has no impact on VRAM consumption during inference.
B. Increasing or decreasing the batch size has the same impact on VRAM consumption.
C. Increasing the batch size reduces VRAM consumption because more data can be processed in parallel.
D. Decreasing the batch size reduces VRAM consumption.
Answer: (SHOW ANSWER)
Batch size has a direct, proportional relationship with VRAM consumption during both training and inference: each sample in a batch requires its own memory allocation for input tensors, intermediate activations at every layer, and output tensors, all of which must reside in GPU memory simultaneously while the batch is being processed. Decreasing the batch size means fewer samples occupy memory concurrently, directly reducing peak VRAM consumption - this is precisely why reducing batch size is one of the first, most common remedies when a model run fails with an out-of-memory (OOM) error on a GPU with limited VRAM.
Option C states the inverse of the correct relationship and is a genuinely important misconception to correct:
increasing batch size increases VRAM consumption, not decreases it - parallelism across the batch means more simultaneous memory occupancy, not less. It's true that larger batches improve GPU compute
utilization and *throughput* (better amortizing fixed kernel-launch overhead and better exploiting parallel hardware) up to the point VRAM allows, but that throughput benefit is a separate effect from, and does not reduce, memory consumption. Options A and B both incorrectly claim batch size is memory-neutral, when it is in fact one of the most direct, easily controlled levers for managing VRAM usage - alongside model precision (quantization, mixed precision) and activation checkpointing, covered in the Performance Optimization domain elsewhere in this set.
Reference: Performance Optimization domain - batch size, VRAM/memory management, inference-time resource tradeoffs.

NEW QUESTION: 19
What is contrastive learning in the context of multimodal deep learning? Pick the 2 correct responses below.

A. Contrastive learning is a technique used to manipulate and analyze multimodal data using Generative AI.
B. In a multimodal context, usually, contrastive learning increases the similarity of representations across modalities for the different objects and decreases the similarity of representations across modalities for same objects.
C. In a multimodal context, usually, contrastive learning decreases the similarity of representations across modalities for the same objects and increases the similarity of representations across modalities for different objects.
D. Contrastive learning is a technique used to train deep learning models by comparing similar and dissimilar inputs and optimizing the model to maximize the similarity between representations of similar inputs and minimize the similarity between representations of dissimilar inputs.
E. In a multimodal context, usually, contrastive learning increases the similarity of representations across modalities for the same objects and decreases the similarity of representations across modalities for different objects.
Answer: D,E (LEAVE A REPLY)
Option D captures the general, task-agnostic definition of contrastive learning: given pairs of inputs labeled as similar (positive pairs) or dissimilar (negative pairs), the training objective pulls positive pairs' representations closer together in embedding space while pushing negative pairs' representations further apart - typically implemented via losses like InfoNCE, triplet loss, or contrastive loss with a margin. This is the mechanism underlying self-supervised representation learning broadly, not only in multimodal settings.
Option E correctly applies this general principle to the multimodal case: for the *same* object described across modalities (e.g., an image of a dog and the caption "a dog"), the model should increase representational similarity, since they refer to the same underlying entity; for *different* objects across modalities (an image of a dog paired with the caption "a cat"), the model should decrease similarity. This is exactly CLIP's training objective, tested elsewhere in this set - matching image-text pairs pulled together, mismatched pairs pushed apart.
Options B and C both invert this relationship - B increases similarity for *different* objects and decreases it for *same* objects, and C similarly reverses the correct direction - describing the opposite of what contrastive learning is designed to achieve, making both clearly incorrect distractors that test careful reading of directionality. Option A is too vague and mischaracterizes contrastive learning as a generative
/manipulation technique rather than a representation-learning objective.
Reference: Multimodal Data domain - contrastive learning objective and its application to cross-modal alignment (CLIP-style training).

NEW QUESTION: 20
What is the significance of A/B testing in ML software engineering?
A. A/B testing is used to measure the impact of changes in the user interface of a ML application.
B. A/B testing helps in optimizing the hyperparameters of a machine learning model.
C. A/B testing is irrelevant in ML software engineering.

D. A/B testing helps in evaluating the performance and effectiveness of different machine learning models.

Answer: (SHOW ANSWER)

A/B testing in an ML engineering context is a controlled experimental methodology where two (or more) model variants - for example, a current production model (A) and a candidate replacement (B) - are deployed simultaneously to randomly assigned, statistically comparable segments of live traffic, and their real- world performance is compared on business or task-relevant metrics (conversion rate, click-through rate, task accuracy, latency-adjusted engagement). This provides causal evidence of which model performs better under actual production conditions, which offline evaluation on static held-out datasets cannot fully capture, since production data distributions shift and downstream user behavior interacts with model outputs in ways offline metrics miss.

Option A narrows A/B testing incorrectly to UI changes alone; while A/B testing originated in and remains common for UI/UX experimentation, its application in ML engineering explicitly extends to comparing model versions, algorithms, and feature sets - not just interface elements. Option B misattributes a distinct methodology (systematic hyperparameter search, covered in the previous question) to A/B testing, which evaluates already-trained model variants rather than searching a hyperparameter space. Option C is simply incorrect - A/B testing is a cornerstone of responsible ML deployment and MLOps practice, gating rollout decisions before full production release.

Reference: Experimentation domain - A/B testing, online evaluation, controlled experimentation in production ML.

NEW QUESTION: 21

You have a dataset containing information about sales performance for different regions in the last ten years.

Which type of data visualization would be most appropriate to compare the sales performance across regions on a year-by-year basis?

A. Bar chart

B. Scatter plot

C. Pie chart

D. Line chart

Answer: (SHOW ANSWER)

NEW QUESTION: 22

Which of the following best describes the role of the Hugging Face model repository in ML software development?

A. A convenient tool for deploying neural networks for production-scale inference similar to Triton Server.

B. A library for customizing large language models like GPT, LLaMA-2, and Falcon using the NeMo framework.

C. A set of NVIDIA SDKs, such as Riva, NeMo, Triton, and ACE, for implementing neural network architectures.

D. A platform for sharing and accessing pre-trained models and transformers for natural language processing.

Answer: D (LEAVE A REPLY)

The Hugging Face Hub is a community-driven platform hosting hundreds of thousands of pretrained models
- spanning NLP, computer vision, audio, and multimodal tasks - along with the accompanying `transformers` library that provides a standardized API to load, fine-tune, and run these models. Its role in the ML development workflow is discovery and access: developers can find a pretrained checkpoint suited to their task, download it with a few lines of code, and fine-tune or deploy it, dramatically lowering the barrier to applying transfer learning without training models from scratch.

This is explicitly distinct from deployment infrastructure: option A describes Triton Server's role (production- scale, multi-framework serving), a different layer of the ML stack than a model repository - Hugging Face models are commonly *exported to* and served *through* Triton in production pipelines, making them complementary rather than equivalent. Option B incorrectly ties Hugging Face specifically to NVIDIA's NeMo framework - Hugging Face is an independent, framework-agnostic ecosystem, not built on or limited to NeMo, though NeMo can import from and export to Hugging Face formats. Option C conflates Hugging Face with the NVIDIA SDK stack (Riva, NeMo, Triton, ACE) entirely - Hugging Face is not an NVIDIA product; it is a separate open-source and commercial company/platform in the ML ecosystem.

Reference: Software Development and Engineering domain - model repositories and their relationship to NVIDIA's deployment/training stack.

NEW QUESTION: 23

What is the purpose of a kernel in a Convolutional Neural Network (CNN)?

A. To perform convolution operations on input data.

B. To calculate the loss function.

C. To classify the data into different categories.

D. To normalize the input data.

Answer: A (LEAVE A REPLY)

A kernel (or filter) in a CNN is a small matrix of learnable weights that slides across the input (an image, feature map, or intermediate activation) computing a dot product at each spatial position - the convolution operation. Each kernel is trained to detect a specific local pattern: early-layer kernels typically learn to detect low-level features like edges and color gradients, while kernels in deeper layers combine these into detectors for more complex, higher-level patterns (textures, object parts, and eventually whole-object representations as receptive fields grow with depth). A convolutional layer typically applies many kernels in parallel, each producing its own output channel, collectively forming the layer's feature map.

The other options describe separate CNN components with distinct responsibilities: the loss function (B) is computed at the network's output based on the difference between predictions and ground truth, entirely separate from the kernel's role in feature extraction. Classification (C) is

typically performed by fully connected (dense) layers - often with a softmax activation - placed after the convolutional feature- extraction stack, not by the kernels themselves. Normalization (D) is handled by dedicated layers such as batch normalization or layer normalization, inserted between convolutional layers to stabilize activations, again a separate mechanism from the convolution operation itself.
Reference: Core Machine Learning and AI Knowledge domain - CNN architecture, kernels/filters, feature extraction.

NEW QUESTION: 24
Which metric is commonly used to evaluate machine-translation models?
A. F1 score
B. Accuracy
C. Mean Absolute Error (MAE)
D. BLEU score
Answer: (SHOW ANSWER)
BLEU (Bilingual Evaluation Understudy) is the standard automatic metric for evaluating machine translation quality. It measures n-gram precision - the overlap of contiguous word sequences (unigrams through typically 4-grams) between the model's translated output and one or more human reference translations - combined with a brevity penalty to discourage overly short translations that could otherwise achieve artificially high precision. BLEU scores range from 0 to 1 (or 0-100 as a percentage), with higher scores indicating closer alignment to reference translations.
The distractors represent metrics standard to other task families: F1 score (A) evaluates classification tasks by balancing precision and recall over discrete positive/negative predictions, ill-suited to open-ended text generation where there is no fixed set of "correct" tokens. Accuracy (B) similarly assumes a discrete correct
/incorrect judgment, inappropriate for translation where multiple valid phrasings can convey the same meaning. Mean Absolute Error (C) is a regression metric measuring average magnitude of numeric prediction error, irrelevant to text output evaluation entirely.
It's worth noting BLEU has known limitations - it correlates imperfectly with human judgments of fluency and can penalize valid paraphrases - which has motivated complementary metrics like METEOR, ROUGE (more common for summarization), and learned metrics like BERTScore, though BLEU remains the benchmark most commonly referenced for translation specifically.
Reference: Core Machine Learning and AI Knowledge / Multimodal Data domains - task-specific evaluation metrics (BLEU for translation, WER for ASR, MOS for TTS).

NEW QUESTION: 25
Which visualization technique is suitable for representing the distribution of performance scores for different multimodal ML models over different modalities?
A. Heatmap
B. Histogram

C. Box plot
D. Pie chart
Answer: (SHOW ANSWER)
A box plot (box-and-whisker plot) summarizes the distribution of a numeric variable - median, interquartile range, and outliers - as a single compact glyph, and critically, multiple box plots can be placed side by side to compare distributions across categorical groupings. This makes it well suited to the scenario described:
comparing the spread and central tendency of performance scores across several models, further faceted by modality, in one readable figure. Box plots make skew, variance, and outlier prevalence immediately comparable across groups in a way a single summary statistic (like mean accuracy) cannot.
A histogram (B) shows the distribution of a single variable well but does not scale cleanly to side-by-side comparison across many model/modality combinations without becoming visually cluttered. A heatmap (A) is excellent for showing a matrix of values (e.g., mean score per model × modality pair) but represents point estimates, not distributions - it cannot convey variance or spread. A pie chart (D) is inappropriate for any continuous performance metric.
In practice, a violin plot - which overlays a kernel density estimate on the box plot's summary statistics - is often preferred when the underlying distribution's shape (e.g., bimodality) matters, but among the given options, the box plot is the correct choice for distributional comparison across groups.
Reference: Data Analysis and Visualization domain - comparative distribution visualization, box plots vs.
heatmaps.

NEW QUESTION: 26
In the context of multimodal machine learning, what does 'data fusion' refer to?
A. Separating different modalities of data into distinct representations.
B. Combining different modalities of data into a single representation.
C. Removing missing or incomplete information from different modalities.
D. Evaluating the quality of diverse data types in multimodal machine learning.
Answer: (SHOW ANSWER)
Data fusion is the process of combining information from multiple modalities into a single, unified representation that a downstream model can act on. As covered in the early-fusion and late-fusion questions elsewhere in this set, fusion can occur at different pipeline stages - raw/feature-level (early), intermediate representation level (hybrid), or decision level (late) - but in every case the defining operation is combination, not separation.
Option A describes the inverse operation and does not correspond to any standard multimodal technique under the name "fusion." Option C describes missing-data handling or imputation, a data-quality concern that is often addressed *before* fusion (a model needs some representation for each modality, even an imputed or masked one, before combining them) but is not fusion

itself. Option D describes evaluation or quality assessment, a distinct concern from the mechanical act of combining modalities into one representation.
Fusion technique choice has real architectural consequences: early fusion assumes tight temporal/spatial correspondence between modalities and is sensitive to missing streams; late fusion is more robust to missing or noisy modalities since each unimodal branch can still contribute independently; hybrid/intermediate fusion, common in modern transformer-based multimodal architectures via cross-attention, aims to capture the benefits of both while mitigating each one's weaknesses.
Reference: Multimodal Data domain - data fusion as the combination operation across early/late/hybrid strategies.

Valid NCA-GENM Dumps shared by ExamDiscuss.com for Helping Passing NCA-GENM Exam! ExamDiscuss.com now offer the **newest NCA-GENM exam dumps**, the ExamDiscuss.com NCA-GENM exam **questions have been updated** and **answers have been corrected** get the **newest** ExamDiscuss.com NCA-GENM dumps with Test Engine here: https://www.examdiscuss.com/NVIDIA/exam/NCA-GENM/premium/ (**58** Q&As Dumps, **35%OFF Special Discount Code: freecram**)