ML, DL, and GPT

1. Machine Learning (ML) is a field of computer science that enables systems to learn from data and improve their performance without explicit programming.
2. Deep Learning (DL) is a subfield of ML that utilizes multi-layered neural networks to model complex patterns in data.
3. Neural Networks are computational models inspired by the human brain, composed of interconnected nodes (neurons) that process information in layers.
4. Training Data is the labeled or unlabeled dataset used to teach a machine learning model how to perform a task.
5. Supervised Learning involves training a model on labeled data, where each example has an input and an associated correct output.
6. Unsupervised Learning uses unlabeled data, allowing the model to discover hidden patterns or groupings without explicit instruction.
7. Reinforcement Learning (RL) trains agents to make decisions by rewarding desired behaviors and penalizing undesirable ones.
8. Generative Models learn to produce new data similar to their training examples (e.g., text, images).
9. Discriminative Models focus on classifying inputs into categories or predicting specific outcomes.
10. Transfer Learning allows a model trained on one task to be reused or fine-tuned on a related task.
11. GPT (Generative Pre-trained Transformer) is a family of large language models developed by OpenAI that can generate human-like text.
12. ChatGPT is an interactive variant of GPT, fine-tuned for conversation and instruction-following tasks.
13. Transformer Architecture was introduced in the paper “Attention Is All You Need,”revolutionizing natural language processing by relying on attention mechanisms.
14. Self-Attention mechanisms let the model weigh different parts of the input sequence when constructing an output representation.
15. Positional Encoding in Transformers helps the model identify the order of tokens in a sequence.
16. Pre-training is the initial phase where a model learns general features from large-scale data before being fine-tuned on specific tasks.
17. Fine-tuning is the process of taking a pre-trained model and adapting it to a narrower task using a smaller, task-specific dataset.
18. Language Modeling is the task of predicting the next token (word or subword) in a sequence, foundational to GPT-like models.

19. Zero-shot Learning allows a model to handle tasks without explicit training examples, relying on learned general knowledge.

20. Few-shot Learning leverages a limited number of task-specific examples to guide model predictions or behaviors.

21. RLHF (Reinforcement Learning from Human Feedback) is used to align model outputs with human preferences and values.

22. Human Feedback can include rankings or labels that guide the model's generation toward more desired responses.

23. Prompt Engineering is the art of crafting input queries or instructions to guide large language models effectively.

24. Context Window refers to the maximum amount of text the model can process at once; GPT models have a limited context length.

25. Inference is the stage where a trained model makes predictions or generates outputs given new inputs.

26. Parameter Count is a key factor in model capacity; larger models can capture more complex patterns but require more computation.

27. Model Compression techniques (e.g., pruning, quantization) reduce a model's size and speed up inference with minimal accuracy loss.

28. Attention Heads in Transformers process different aspects of the input in parallel, improving representational power.

29. Masked Language Modeling (e.g., in BERT) involves predicting missing tokens in a sentence, helping the model learn context.

30. Causal Language Modeling (e.g., in GPT) involves predicting the next token based on all previous tokens.

31. Encoder-Decoder Architecture (e.g., T5) uses one network to encode the input and another to decode it into a target sequence.

32. Convolutional Neural Networks (CNNs) excel at processing grid-like data (e.g., images) via convolutional layers.

33. Recurrent Neural Networks (RNNs) process sequential data by passing hidden states along time steps, though they can struggle with long-term dependencies.

34. Long Short-Term Memory (LSTM) and GRU are RNN variants designed to better capture long-range dependencies.

35. Batch Normalization helps stabilize training by normalizing intermediate layer outputs.

36. Dropout is a regularization technique that randomly “drops”neurons during training to prevent overfitting.

37. Optimizer Algorithms like Stochastic Gradient Descent (SGD), Adam, and RMSProp update model parameters based on gradients.

38. Learning Rate is a hyperparameter that determines how drastically weights are updated during training.

39. Hyperparameters (e.g., batch size, number of layers) are configuration settings chosen before training to control how learning unfolds.

40. Model Overfitting occurs when a model learns training data too well, failing to generalize to new data.

41. Regularization Techniques (e.g., L2 weight decay, dropout) help reduce overfitting and improve generalization.

42. Validation Set is used to tune hyperparameters, while the Test Set evaluates the final performance of the model.

43. Cross-validation splits data into multiple subsets, systematically training and validating to get a more robust performance estimate.

44. Gradient Exploding and Vanishing problems occur in deep networks, making training unstable or ineffective.

45. Residual Connections (skip connections) in networks like ResNet help mitigate vanishing gradients by shortcutting data paths.

46. Scaling Laws suggest that increasing model size and data generally leads to better performance.

47. Compute Efficiency is critical; training large models requires optimized hardware (GPUs, TPUs) and algorithms.

48. Ethical Considerations include bias, fairness, and potential harm—ML models must be carefully tested and monitored.

49. Data Augmentation artificially expands training datasets to improve model robustness (especially in image and speech tasks).

50. Data Preprocessing (e.g., tokenization, normalization) is essential for effective model training.

51. Tokenization splits text into tokens (words or subwords), the fundamental units processed by language models.

52. Vector Embeddings represent tokens or concepts as numerical vectors, preserving semantic relationships.

53. Positional Embeddings add information about the position of each token to help a Transformer understand sequence order.

54. Attention Weights reveal how a model distributes focus across different parts of the input.

55. Beam Search is a decoding strategy in language models that keeps multiple candidate outputs at each step to find the best overall sequence.

56. Greedy Search picks the most probable token at each step, but can lead to suboptimal final outputs.

57. Temperature in sampling adjusts the creativity of language generation: higher temperature = more randomness.

58. Top-k and Top-p (Nucleus) sampling methods restrict the candidate tokens to the k most likely or a cumulative probability p, balancing diversity and coherence.

59. Perplexity measures how well a probability model predicts a sample; lower perplexity indicates better predictive performance.

60. Precision and Recall are metrics for classification tasks, focusing on correctness and completeness, respectively.

61. F1 Score is the harmonic mean of precision and recall, balancing both metrics into a single value.

62. Accuracy is the fraction of correct predictions, but it can be misleading in imbalanced datasets.

63. Area Under the ROC Curve (AUC) measures a classifier's performance across various thresholds.

64. Confusion Matrix shows the counts of true positives, false positives, false negatives, and true negatives.

65. Uncertainty Estimation methods (e.g., Monte Carlo Dropout) gauge how confident a model is in its predictions.

66. Active Learning involves querying new data examples that the model is least confident about, improving data efficiency.

67. Online Learning updates the model incrementally as new data arrives, rather than retraining from scratch.

68. Evolutionary Algorithms and Genetic Algorithms optimize models or hyperparameters using bio-inspired mutation and selection.

69. Bayesian Methods incorporate prior knowledge and update beliefs with incoming data, useful for uncertainty quantification.

70. Ensemble Methods (e.g., Random Forest, Gradient Boosting) combine multiple models to improve performance and stability.

71. Bagging (Bootstrap Aggregating) trains multiple models on different subsets of the data, then averages their predictions.

72. Boosting iteratively trains new models to correct errors made by previously trained models.

73. Gradient Boosted Decision Trees (GBDTs) are powerful for structured data, often outperforming simple neural networks.
74. Autoregressive Models predict the next value (or token) based on previous outputs in a sequence.
75. Autoencoder is a neural network designed to encode data into a latent representation and then decode it back, learning compressed data representations.
76. Variational Autoencoder (VAE) introduces a probabilistic twist to generate new data that resembles the training set.
77. Generative Adversarial Network (GAN) pits a generator against a discriminator, producing realistic images, text, or other data.
78. Self-Supervised Learning leverages large amounts of unlabeled data by creating artificial training tasks (e.g., predicting missing parts).
79. Foundation Models are large pre-trained models that can be adapted to a wide range of downstream tasks.
80. Multimodal Learning integrates data from multiple sources (e.g., text, images, audio) to create richer representations.
81. Data Labeling is often the most time-consuming part of ML, requiring careful annotation for accuracy.
82. Edge Computing brings ML inference closer to the data source, reducing latency and bandwidth usage.
83. Federated Learning trains models across decentralized devices or servers holding local data samples, without exchanging them.
84. Privacy-Preserving ML includes techniques like differential privacy and homomorphic encryption to protect sensitive data.
85. Explainable AI (XAI) aims to make the decisions of complex models more interpretable to humans.
86. Bias and Fairness in ML need careful oversight, as models can inadvertently learn and amplify societal biases.
87. Concept Drift occurs when the statistical properties of the target variable change over time, impacting model performance.
88. AB Testing compares two or more versions of a model to see which performs better in a real-world environment.
89. GPU Acceleration exploits parallel computing on graphics cards to drastically speed up ML training.
90. TPUs (Tensor Processing Units) are specialized hardware accelerators by Google for efficient deep learning workloads.
91. Open-Source Frameworks (e.g., TensorFlow, PyTorch) provide building blocks and tools for ML model development.

92. Model Serving is the practice of deploying trained models so they can handle real-time or batch predictions.

93. Scalability is crucial for handling large datasets or heavy traffic, requiring distributed training and inference strategies.

94. MLOps combines ML development with operations practices, focusing on reproducibility, testing, and continuous integration.

95. Version Control for data and models ensures consistent experiment tracking and collaboration.

96. Deployment Strategies (e.g., containers, microservices) organize how models are packaged and served at scale.

97. Monitoring tracks model performance post-deployment, watching for degradations or anomalies.

98. Retraining and Model Updates keep models current as new data and changing conditions arise.

99. Time Complexity (O-notation) measures how an algorithm's runtime scales with input size; O(1) denotes constant time.

100. Future of ML promises increasingly sophisticated and general models, but must address ethical, social, and environmental considerations.