Deep Learning

Multiple-Choice Questions

Lern-Story: Multiple Choice Fragen zum Thema Deep Learning
Lernziel: Evaluation von Wissen mithilfe von Multiple Choice Fragen.
Input:

- Alle Deep Learning Themen

Output:

- Wissenskontrolle

Lernerfolg: Bestimmen Sie ihren Lernerfolg.

Aufgabe 1: Single Choice Frage

1. Welcher Satz beschreibt den Hauptunterschied zwischen Dense- und Convolutional-Schichten?

   ☐ Dense-Schichten lernen lokale Muster
   ☐ Dense-Schichten lernen globale Muster, Convolutional-Schichten lokale Muster
   ☐ Beide lernen nur globale Muster
   ☐ Beide lernen nur lokale Muster

2. Welches Prinzip ermöglicht es einem CNN, ein gelerntes Muster an jeder Bildposition zu erkennen?

   ☐ Datenaugmentierung
   ☐ Translation Invariance
   ☐ Überanpassung
   ☐ Dropout

3. Wie gross ist die Tiefenachse eines RGB-Bildes?

   ☐ 1
   ☐ 2
   ☐ 3
   ☐ 4

4. Welche zwei Hyperparameter definieren eine Faltung massgeblich?

   ☐ Aktivierungsfunktion und Lernrate
   ☐ Patch-Grösse und Ausgabetiefe
   ☐ Optimierer und Batch-Grösse
   ☐ Anzahl der Epochen und Padding

5. Ein 3×3-Fenster wird auf eine 5×5-Feature-Map ohne Padding angewendet. Um wie viele Tiles schrumpft jede räumliche Dimension?

   ☐ 1
   ☐ 2
   ☐ 3
   ☐ 5

6. Wie heisst die Technik, bei der zusätzliche Zeilen und Spalten hinzugefügt werden, um Eingabe- und Ausgabegrösse gleichzuhalten?

   ☐ Normalisierung
   ☐ Padding
   ☐ Striding
   ☐ Pooling

7. Welche Auswirkung hat Stride=2 bei einer Faltung?

   ☐ Keine Veränderung der Grösse
   ☐ Vergrössert Höhe und Breite
   ☐ Verkleinert Höhe und Breite um Faktor 2
   ☐ Verdoppelt die Anzahl der Filter

8. Was ist das primäre Ziel von Max-Pooling?

   ☐ Gewichte aktualisieren
   ☐ Feature-Maps aggressiv verkleinern
   ☐ Modell regulieren
   ☐ Filterzahl erhöhen

9. Eine Feature-Map besitzt immer Achsen für ...

   ☐ Breite, Höhe, Zeit
   ☐ Höhe, Tiefe, Batch
   ☐ Höhe, Breite, Tiefe
   ☐ Filter, Verlust, Genauigkeit

10. Was lernen tiefere Convolutional-Schichten typischerweise?

    ☐ Einfache Kanten
    ☐ Rauschen
    ☐ Grössere, abstraktere Muster aus früheren Features
    ☐ Gewichtsinitialisierung

11. Average Pooling berechnet als Ausgabe eines Patches

- ☐ das Maximum jeder Tiefe
- ☐ den Median jeder Tiefe
- ☐ den Durchschnitt jeder Tiefe
- ☐ das Minimum jeder Tiefe

12. Wofür ist Max-Pooling nicht hauptsächlich zuständig?

- ☐ Raumhierarchien zu lernen
- ☐ Parameterzahl reduzieren
- ☐ Aktivierungen normalisieren
- ☐ Downsampling durchführen

13. Wenn Sie eine Feature-Map um Faktor 2 verkleinern möchten, wählen Sie am wahrscheinlichsten

- ☐ Max Pooling mit Stride 2
- ☐ Padding
- ☐ Faltung mit Stride 1
- ☐ Average Pooling mit Stride 1

14. Welche Art von Mustern lernt typischerweise die erste Convolution Schicht?

- ☐ Komplexe Objekte wie Gesichter
- ☐ Kleine lokale Merkmale (z.B. Kanten)
- ☐ Globale Semantik
- ☐ Farbverteilungen

15. Nach der ersten Convolution repräsentiert die Tiefenachse der Feature Map ...

- ☐ Farbkanäle
- ☐ Einzelne Pixel
- ☐ Filter / Features
- ☐ Zeitschritte

16. Hauptzweck des Paddings ist ...

- ☐ Training zu beschleunigen
- ☐ Overfitting zu verhindern
- ☐ Räumliche Dimensionen des Inputs zu erhalten
- ☐ Die Output-Depth zu reduzieren

17. Flattening in einem CNN bedeutet ...

- ☐ Eine 1×1-Convolution anwenden
- ☐ Die 3D-Feature-Map vor Dense-Layern in einen 1D-Vektor umwandeln
- ☐ Feature-Maps Zero-Padden
- ☐ Über Kanäle mitteln

18. Convolution-Layer lernen _____ Muster, während Dense-Layer _____ Muster lernen.

- ☐ globale; lokale
- ☐ sequentielle; räumliche
- ☐ lokale; globale
- ☐ zeitliche; kategoriale

19. Ein Filter in einem CNN ist am besten beschrieben als ...

- ☐ Einzelner Pixelwert
- ☐ Gelernte Gewichtsmatrix, die ein spezifisches Merkmal detektiert
- ☐ Lernratenplan
- ☐ Stride der Convolution

20. Wird kein Padding verwendet und der Kernel ist 3×3, wie viele Pixel gehen an jedem Rand verloren?

- ☐ 0
- ☐ 1
- ☐ 2
- ☐ 3

21. Welche Pooling-Operation liefert den Maximalwert jedes Fensters?

- ☐ Min-Pooling
- ☐ Average-Pooling
- ☐ Sum-Pooling
- ☐ Max-Pooling

22. Hierarchisches Lernen von Features in CNNs bedeutet, dass ...

- ☐ Alle Layer dieselbe Abstraktionsebene lernen
- ☐ Tiefere Layer Kombinationen einfacher Features früherer Layer lernen
- ☐ Frühe Layer auf globalen Kontext fokussieren
- ☐ Pooling-Layer Filter lernen

23. Ein CNN benötigt weniger Trainingsdaten als ein vergleichbares Dense-Netz hauptsächlich wegen ...

- ☐ Datenaugmentation
- ☐ Translationsinvarianz & Weight Sharing
- ☐ Hoher Lernrate
- ☐ Dropout-Regularisierung

24. Was ist das Hauptziel der Regularisierung

- ☐ Die Lernrate zu erhöhen
- ☐ Die Trainingszeit zu verkürzen
- ☐ Overfitting zu vermeiden
- ☐ Die Datenmenge zu reduzieren

25. Wie wird die Regularisierung technisch in der Optimierung des Modells umgesetzt?

- ☐ Durch Erhöhung der Lernrate
- ☐ Durch Hinzufügen eines Strafterms zur Verlustfunktion
- ☐ Durch Reduktion der Batchgrösse
- ☐ Durch Datenaugmentation

26. Was bewirkt der Strafterm in der Regularisierung?

- ☐ Er erhöht die Gewichte des Modells
- ☐ Er macht das Modell langsamer
- ☐ Er verhindert, dass das Modell die Trainingsdaten zu genau lernt
- ☐ Er reduziert die Anzahl der Epochen

27. Was ist ein typischer Effekt von L1-Regularisierung?

- ☐ Erhöhte Anzahl an Epochen
- ☐ Gleichmässige Verteilung der Gewichte
- ☐ Setzt Gewichte auf Null
- ☐ Schnelleres Training

28. Welche Norm wird in der L2-Regularisierung verwendet?

- ☐ Absolutwert
- ☐ Quadratwurzel der quadrierten Gewichte
- ☐ Kubikwurzel
- ☐ Maximumswert

29. Was bewirkt die L2-Regularisierung im Modell?

- ☐ Sie reduziert die Datenmenge
- ☐ Sie bevorzugt grössere Gewichte
- ☐ Sie schränkt die Grösse der Gewichte ein
- ☐ Sie entfernt irrelevante Features

30. Welche Aussage trifft auf L1-Regularisierung zu?

- ☐ Führt zu gleichmässiger Gewichtung
- ☐ Führt zu Feature Selektion
- ☐ Erhöht die Modellkomplexität
- ☐ Führt zu längerer Trainingszeit

31. Was stellt der Regularisierungsparameter dar?

- ☐ Die Lernrate des Modells
- ☐ Die maximale Anzahl an Epochen
- ☐ Das Gewicht des Strafterms in der Verlustfunktion
- ☐ Die Anzahl der Features

32. Was passiert bei zu starker Regularisierung?

- ☐ Das Modell overfittet schneller
- ☐ Die Gewichte werden zu klein und das Modell underfittet
- ☐ Es entsteht kein Unterschied zur normalen Verlustfunktion
- ☐ Die Batchgröße steigt

33. Was bewirkt Dropout?

- ☐ Es reduziert die Anzahl der Trainingsdaten
- ☐ Es entfernt dauerhaft Neuronen
- ☐ Es deaktiviert zufällig Neuronen während des Trainings
- ☐ Es erhöht die Netzwerkkomplexität

34. Wozu dient Early Stopping?

- ☐ Um das Modell schneller zu machen
- ☐ Um die Anzahl der Neuronen zu reduzieren
- ☐ Um das Training zu stoppen, wenn sich der Validierungsfehler nicht mehr verbessert
- ☐ Um alle Epochen vollständig zu nutzen

35. Was beschreibt das Bagging-Verfahren?

- ☐ Lernen mit nur einem Modell
- ☐ Kombination von Modellen durch Mittelung ihrer Vorhersagen
- ☐ Reduktion der Datenmenge
- ☐ Entfernen von Ausreissern

36. Welche Technik erweitert künstlich den Datensatz?

- ☐ Dropout
- ☐ Batch Normalization
- ☐ Data Augmentation
- ☐ Early Stopping

37. Welche Transformation ist kein typisches Beispiel für Data Augmentation?

- ☐ Rotation
- ☐ Cropping
- ☐ Normalisierung
- ☐ Farbänderung

38. Was ist eine Einschränkung von Data Augmentation?

- ☐ Funktioniert nur mit numerischen Daten
- ☐ Kann nicht für Klassifikationen verwendet werden
- ☐ Ist bei stark korrelierten neuen Daten begrenzt wirksam
- ☐ Verändert die Netzwerkarchitektur

39. Was wird bei der Batch Normalization normalisiert?

- ☐ Die Daten vor dem Training
- ☐ Die Eingabeschicht
- ☐ Die Gewichte der Output-Schicht
- ☐ Die Aktivierungen innerhalb eines Layers

40. Was ist das Ziel von Batch Normalization?

- ☐ Overfitting zu verstärken
- ☐ Nur die Trainingsdaten zu normalisieren
- ☐ Die Verteilung der Aktivierungen zu stabilisieren
- ☐ Die Gewichtsmatrix zu vergrößern

41. Wann wird Batch Normalization durchgeführt?

- ☐ Einmalig vor dem Training
- ☐ Nach jeder Epoche
- ☐ Nach jedem Batch
- ☐ Nur bei großen Modellen

42. Was ist das Hauptziel von Data Augmentation?

- ☐ Die Genauigkeit des Validierungsdatensatzes zu reduzieren
- ☐ Den Trainingsprozess zu verlangsamen
- ☐ Mehr Trainingsdaten aus bestehenden Bildern zu generieren
- ☐ Bilder zufällig zu löschen

43. Warum sieht ein Modell mit Data Augmentation nie das gleiche Bild zweimal?

- ☐ Weil das Bildformat geändert wird
- ☐ Weil immer neue Informationen generiert werden
- ☐ Weil durch Zufallstransformationen jede Version leicht unterschiedlich ist
- ☐ Weil alte Bilder gelöscht werden

44. Was ist ein Vorteil vortrainierter Modelle?

- ☐ Sie löschen irrelevante Daten automatisch
- ☐ Sie lernen ohne Labels
- ☐ Sie können gelernte Features auf andere Datensätze überführen
- ☐ Sie ignorieren visuelle Merkmale

45. Warum wird der Dense Classifier eines pretrained CNNs nicht wiederverwendet?

- ☐ Er ist zu komplex
- ☐ Er ist zufällig
- ☐ Er ist spezifisch für die alten Klassen
- ☐ Er ist zu langsam

46. Was bedeutet "Freezing" in Bezug auf ein Modell?

- ☐ Die GPU wird deaktiviert
- ☐ Trainingsdaten werden gesperrt
- ☐ Layers und deren Gewichte werden eingefroren
- ☐ Alle Bilder werden eingefroren gespeichert

47. Wann darf man die oberen Schichten einer CNN zum Fine-Tuning freigeben?

- ☐ Sofort nach Modellinitialisierung
- ☐ Bevor der Classifier trainiert wurde
- ☐ Nachdem der Classifier trainiert wurde
- ☐ Vor dem Einfrieren

48. Welche Eigenschaft trifft auf die unteren Schichten eines CNN zu?

- ☐ Sie erkennen komplexe Formen
- ☐ Sie erkennen spezifische Objekte
- ☐ Sie extrahieren generische Merkmale
- ☐ Sie haben keine Funktion

49. Was passiert beim "Fine Tuning"?

- ☐ Neue Bilder werden erzeugt
- ☐ Der Convolutional Base wird vollständig neu trainiert
- ☐ Nur obere Schichten des Bases werden weitertrainiert
- ☐ Dropout wird deaktiviert

50. Warum ist ein kleines Learning Rate beim Fine Tuning wichtig?

- ☐ Um schneller zu trainieren
- ☐ Um grosse Änderungen an gelernten Repräsentationen zu vermeiden
- ☐ Um das Modell zurückzusetzen
- ☐ Um Data Augmentation zu verbessern

51. Worin besteht ein Nachteil, zu viele Schichten feinzujustieren?

- ☐ Modelle werden unbrauchbar
- ☐ Man kann keine Merkmale mehr extrahieren
- ☐ Erhöhtes Risiko für Overfitting
- ☐ Speicherplatz wird knapp

52. Warum ist die Wiederverwendung von Features über verschiedene Probleme hinweg möglich?

- ☐ Weil alle Bilder gleich sind
- ☐ Weil CNNs Bilddaten ignorieren
- ☐ Weil Feature Maps generisch sind
- ☐ Weil Labels nicht benötigt werden

53. Was macht Dropout beim Training mit Data Augmentation?

- ☐ Fügt Rauschen hinzu
- ☐ Verhindert das Speichern von Bildern
- ☐ Ergänzt die Wirkung von Data Augmentation zur Reduzierung von Overfitting
- ☐ Stoppt den Trainingsprozess

54. Was bedeutet es, wenn Eingaben “hoch korrelieren” sind?

- ☐ Sie sind komplett zufällig
- ☐ Sie enthalten keine Informationen
- ☐ Sie stammen aus derselben Quelle und ähneln sich
- ☐ Sie sind unbrauchbar

55. Wann ist Feature Extraction besonders hilfreich?

- ☐ Wenn kein Computer verfügbar ist
- ☐ Bei extrem grossen Datensätzen
- ☐ Bei kleinen Datensätzen mit ähnlichen Aufgaben
- ☐ Wenn keine GPU vorhanden ist

56. Was beschreibt ein Feed Forward Neural Network am besten?

- ☐ Ein neuronales Netz mit Rückkopplung
- ☐ Ein Netz, in dem Informationen nur in eine Richtung – von Input zu Output – fliessen
- ☐ Ein Netz, das ausschliesslich für Textverarbeitung verwendet wird
- ☐ Ein Netz mit unendlich vielen Schichten

57. Welche Aussage über Aktivierungsfunktionen ist korrekt?

- ☐ Sie sind nur in der Ausgabeschicht notwendig
- ☐ Sie bestimmen die Lernrate
- ☐ Sie helfen, Nichtlinearitäten im Modell abzubilden
- ☐ Sie reduzieren die Anzahl der benötigten Datenpunkte

58. Wozu dient eine Verlustfunktion (Loss Function) in neuronalen Netzen?

- ☐ Zur Verbesserung der Aktivierungsfunktion
- ☐ Zur Bewertung der Vorhersagequalität durch Vergleich mit den echten Labels
- ☐ Zum Generieren neuer Trainingsdaten
- ☐ Zum Speichern der Gewichtswerte

59. Was macht ein Optimizer im Training eines neuronalen Netzes?

- ☐ Fügt neue Daten hinzu
- ☐ Passt die Aktivierungsfunktionen an
- ☐ Ändert die Struktur des Netzes
- ☐ Aktualisiert Gewichte und Biases basierend auf der Verlustfunktion

60. Was ist das Ziel beim Trainieren eines Deep-Learning-Modells?

- ☐ Möglichst viele Schichten und Neuronen zu verwenden
- ☐ Zufällige Gewichtswerte beizubehalten
- ☐ Die Vorhersagen möglichst nahe an den echten Labels auszurichten
- ☐ Immer eine binäre Klassifikation zu verwenden

61. Was ist ein typisches Ziel bei Regressionsproblemen in neuronalen Netzen?

- ☐ Die Erkennung von Objekten
- ☐ Die Vorhersage diskreter Kategorien
- ☐ Die Reduktion der Trainingszeit
- ☐ Die Vorhersage kontinuierlicher Werte

62. Warum ist die Initialisierung der Gewichte wichtig?

- ☐ Sie beschleunigt das Speichern der Modelle
- ☐ Sie beeinflusst die Trainingsdatenmenge
- ☐ Sie kann den Lernprozess positiv oder negativ beeinflussen
- ☐ Sie bestimmt die Anzahl der Ausgabeklassen

63. Welche Aussage zur Wahl der Anzahl versteckter Schichten (Hidden Layers) ist korrekt?

- ☐ Mehr Schichten sind immer besser
- ☐ Ein einfaches Problem erfordert viele versteckte Schichten
- ☐ Die Wahl hängt von der Komplexität des Problems ab
- ☐ Versteckte Schichten sind optional

64. Welcher Optimierer passt die Lernrate für jeden Parameter an?

- ☐ SGD
- ☐ Adam
- ☐ RMSProp
- ☐ Momentum

65. Was ist das Hauptziel der Regularisierung in neuronalen Netzwerken?

- ☐ Die Trainingszeit zu verkürzen
- ☐ Überanpassung zu verhindern
- ☐ Die Lernrate zu erhöhen
- ☐ Die Netzwerkgrösse zu vergrössern

66. Was passiert beim Dropout während des Trainings?

- ☐ Die Gewichte werden eingefroren
- ☐ Einige Neuronen werden zufällig deaktiviert
- ☐ Der Lernrate wird reduziert
- ☐ Der Optimierer wird gewechselt

67. Was sagt die Lernrate aus?

- ☐ Die Lernrate bestimmt, wie viele Schichten ein neuronales Netz haben darf.
- ☐ Die Lernrate gibt an, mit welcher Geschwindigkeit das Modell aus Informationen lernt
- ☐ Die Lernrate legt fest, wie gross das endgültige Modell sein wird.
- ☐ Die Lernrate gibt an, wie viele Datenpunkte pro Sekunde verarbeitet werden.

68. Was ist die Gefahr bei einer zu niedrigen Lernrate?

- ☐ Das Modell überspringt ständig das Minimum der Fehlerfunktion.
- ☐ Das Training wird sofort abgebrochen.
- ☐ Kann in einem lokalen Minimum stecken bleiben
- ☐ Eine zu niedrige Lernrate führt zu Überanpassung (Overfitting).

69. Was ist die Gefahr bei einer zu hohen Lernrate?

- ☐ Das Modell erreicht das Optimum schneller und genauer.
- ☐ Die Lösung kann divergieren
- ☐ Overfitting
- ☐ Das Training funktioniert nur bei linearen Modellen nicht mehr.

70. Was macht die Lernrate-Strategie Step decay?

- ☐ Sie erhöht die Lernrate exponentiell mit jeder Epoche.
- ☐ Macht die Lernrate um fixen Faktor kleiner nach einer gewünschten Anzahl Epochen
- ☐ Sie passt die Lernrate zufällig während des Trainings an.
- ☐ Sie setzt die Lernrate nach jeder Epoche auf null und startet neu.

71. Was ist eine Epoche?

- ☐ Eine Epoche ist ein einzelner Durchlauf durch nur einen Datenpunkt des Trainingssatzes.
- ☐ Eine Epoche ist ein Durchlauf bei dem alle Trainingsdaten einmal verwendet werden.
- ☐ Eine Epoche beschreibt die Anzahl der Schichten in einem neuronalen Netzwerk.
- ☐ Eine Epoche ist ein Durchlauf mit einem Anteil der Trainingsdaten

72. Was ist die Gefahr, wenn mit zu weniger Epochen trainiert wird?

- ☐ Das Modell wird übertrainiert und passt sich zu stark den Trainingsdaten an.
- ☐ Das Modell konvergiert allenfalls nicht zu einer guten Lösung
- ☐ Das Modell wird zu stark regularisiert und erzielt daher keine guten Ergebnisse.
- ☐ Das Modell benötigt keine Feinabstimmung mehr und ist sofort einsatzbereit

73. Wann wird die Modellleistung evaluiert?

- ☐ Am Ende der letzten Epoche
- ☐ Nach jeder Epoche
- ☐ Nach jedem gelerntem Datensatz
- ☐ Vor dem Modell Training

74. Was ist ein Batch?

- ☐ Ein Batch ist die Anzahl der Schichten in einem neuronalen Netzwerk.
- ☐ Ein Batch ist eine Teilmenge des Trainingsdatensatzes, die in einem Schritt verarbeitet wird.
- ☐ Ein Batch ist eine einzelne Datenprobe, die dem Modell während des Trainings präsentiert wird.
- ☐ Ein Batch ist eine spezielle Art von Modell, das auf den Trainingsdaten angewendet wird.

75. Was für eine Gefahr gibt es, wenn man die Anzahl Epochen zu gross wählt?

- ☐ Das Modell wird zu stark regularisiert und lernt keine sinnvollen Muster.
- ☐ Das Modell wir zu instabil und verlangsamt das Training
- ☐ Das Modell wird zu langsam und benötigt mehr Epochen, um Ergebnisse zu liefern.
- ☐ Das Modell wird zu stark auf den Trainingsdaten angepasst und zeigt keine Fehler mehr.

76. Was für eine Gefahr gibt es, wenn man die Anzahl Epochen zu gross wählt?

- ☐ Das Modell wird immer besser auf den Trainingsdaten, aber schlechter auf den Testdaten.
- ☐ Das Modell benötigt umso mehr Zeit um mit den Trainingsdaten zu trainieren.
- ☐ Das Modell könnte überanpassen (Overfitting), indem es sich zu sehr an die Trainingsdaten anpasst.
- ☐ Das Modell wird automatisch in der Lage sein, auf alle unbekannten Daten sehr gut zu generalisieren.

77. Welche grundlegende Idee steckt hinter dem Gradient Descent-Verfahren?

- ☐ Es findet das Maximum der Verlustfunktion durch zufällige Gewichtsanpassung.
- ☐ Es passt die Gewichte so an, dass die Rechenzeit minimiert wird.
- ☐ Es bewegt sich entlang der Gradientenrichtung, um lokale Maxima zu erreichen.
- ☐ Es passt die Modellparameter schrittweise in Richtung des negativsten Gradienten an, um die Verlustfunktion zu minimieren.

78. Welche Aufgabe hat ein Optimierer im Training eines neuronalen Netzwerks?

- ☐ Er bestimmt, wie viele Daten für das Modell gespeichert werden sollen.
- ☐ Er reduziert den Speicherverbrauch, indem er Parameter entfernt.
- ☐ Er passt die Gewichte des Modells so an, dass die Verlustfunktion minimiert wird.
- ☐ Er erhöht die Modellkomplexität automatisch, um Overfitting zu vermeiden.

79. Wofür wird eine Loss Function im Training verwendet?

- ☐ Um den Unterschied zwischen Vorhersage und echtem Wert zu messen
- ☐ Um die Trainingsdaten zu normalisieren
- ☐ Um die Anzahl der Layer zu bestimmen
- ☐ Um das Modell schneller zu machen

80. Warum muss eine Loss Function differenzierbar sein?

- ☐ Damit Gradient Descent sie minimieren kann
- ☐ Damit sie auf alle Probleme passt
- ☐ Damit man keine Aktivierungsfunktion braucht
- ☐ Damit sie ohne Optimizer funktioniert

81. Was ist der Hauptunterschied zwischen einer Loss Function und einer Metrik?

- ☐ Die Loss Function wird zur Optimierung genutzt, die Metrik zur Bewertung
- ☐ Beide machen dasselbe
- ☐ Metriken funktionieren nur bei Regression
- ☐ Die Loss Function wird nur nach dem Training verwendet

82. Welche Aussage zu MSE ist korrekt?

- ☐ Grosse Fehler werden stärker bestraft als kleine
- ☐ Alle Fehler werden gleich bewertet
- ☐ MSE funktioniert nur bei Klassifikation
- ☐ MSE ist nur für binäre Aufgaben geeignet

83. Wofür ist die Categorical Cross-Entropy geeignet?

- ☐ Für Klassifikation mit mehr als zwei Klassen
- ☐ Für Regression mit kontinuierlichen Werten
- ☐ Für binäre Entscheidungen
- ☐ Für Bilder mit nur einem Farbkanal

84. Warum kann man bei Multi-Klassifikation keine MSE verwenden?

- ☐ Weil MSE keine Wahrscheinlichkeitsverteilungen berücksichtigt
- ☐ Weil MSE zu schnell konvergiert
- ☐ Weil MSE keine Label erkennt
- ☐ Weil MSE nur bei Sigmoid funktioniert

85. Was passiert mit der Loss Function, wenn das Modell besser wird?

- ☐ Sie wird kleiner
- ☐ Sie bleibt konstant
- ☐ Sie wird negativ
- ☐ Sie steigt exponentiell

86. Was misst die Loss Function bei einem Klassifikationsmodell mit Softmax-Ausgabe?

- ☐ Wie gut die Wahrscheinlichkeitsverteilung zur wahren Klasse passt
- ☐ Ob die Summe aller Gewichte 1 ist
- ☐ Wie gross die Lernrate ist
- ☐ Ob die Metrik Accuracy ¿ 90% ist

87. Was ist der Hauptunterschied zwischen Binary und Categorical Cross-Entropy?

- ☐ Binary ist für 2 Klassen, Categorical für viele Klassen
- ☐ Binary ist für Regression, Categorical für Bilder
- ☐ Binary ist schneller, Categorical langsamer
- ☐ Categorical benötigt keine One-Hot-Encoding

88. Welche Loss Function würdest du für Hauspreisvorhersage nehmen?

- ☐ MSE
- ☐ Binary Cross-Entropy
- ☐ Categorical Cross-Entropy
- ☐ Hinge Loss

89. Wozu dient ein Optimizer im Training eines neuronalen Netzes?

- ☐ Er passt die Gewichte an, um die Loss Function zu minimieren
- ☐ Er misst die Testgenauigkeit
- ☐ Er kontrolliert die Batch-Grösse
- ☐ Er bestimmt die Anzahl der Epochen

90. Welche Information benötigt ein Optimizer bei jedem Schritt?

- ☐ Den Gradienten der Loss Function
- ☐ Die Zielmetrik (z. B. Accuracy)
- ☐ Die Anzahl der Layer
- ☐ Den Dateinamen des Datasets

91. Was ist der wichtigste Hyperparameter beim Optimizer?

- ☐ Die Lernrate
- ☐ Die Anzahl der Layer
- ☐ Die Grösse des Testsets
- ☐ Die Aktivierungsfunktion

92. Was passiert bei einer zu hohen Lernrate?

- ☐ Das Modell konvergiert möglicherweise nicht oder divergiert
- ☐ Das Modell lernt sehr stabil
- ☐ Es wird kein Backpropagation benötigt
- ☐ Die Loss Function wird ignoriert

93. Was passiert bei einer zu niedrigen Lernrate?

- ☐ Das Modell lernt extrem langsam
- ☐ Die Genauigkeit steigt sofort auf 100%
- ☐ Die Loss Function wird unbrauchbar
- ☐ Das Modell vergisst alle vorherigen Schritte

94. Was ist “Learning Rate Decay”?

- ☐ Eine Technik, bei der die Lernrate schrittweise gesenkt wird
- ☐ Eine Methode, um die Lernrate zu erhöhen
- ☐ Eine spezielle Art von Loss Function
- ☐ Ein Aktivierungstyp in der Ausgangsschicht

95. Was zeichnet Adam gegenüber SGD aus?

- ☐ Adam passt die Lernrate für jeden Parameter dynamisch an
- ☐ Adam ignoriert Gradienten
- ☐ Adam benötigt keine Loss Function
- ☐ Adam ist deterministisch

96. Was ist das Ziel des Gradient Descent Algorithmus?

- ☐ Das Finden eines Minimums der Loss Function
- ☐ Die Erstellung neuer Testdaten
- ☐ Das Erhöhen der Batchgrösse
- ☐ Die Änderung der Modellarchitektur

97. Warum sind rekurrente neuronale Netze (RNNs) besonders geeignet für Sequenzdaten?

- ☐ Weil sie keine Erinnerung haben.
- ☐ Weil sie Abhängigkeiten zwischen aufeinanderfolgenden Datenpunkten erfassen können.
- ☐ Weil sie schneller trainiert werden können.
- ☐ Weil sie weniger Parameter benötigen.

98. Welche Art von Daten sind Beispiele für Sequenzdaten?

- ☐ Bilder
- ☐ Text und Zeitreihen
- ☐ Tabellen und Datenbanken
- ☐ Unabhängige Datenpunkte

99. Warum ist die Reihenfolge in Sequenzdaten wichtig?

- ☐ Weil sie die Trainingszeit verkürzt.
- ☐ Weil der Vorgänger den nachfolgenden Wert beeinflusst.
- ☐ Weil sie die Anzahl der Parameter reduziert.
- ☐ Weil sie die Klassifikationsgenauigkeit erhöht.

100. Wie wird der state in einem RNN aktualisiert?

- ☐ Durch Multiplikation vom Input mit dem vorherigen State
- ☐ Durch Anwendung einer nichtlinearen Funktion auf Input und den vorherigen State
- ☐ Input und der vorangehende State werden skaliert und anschliessend addiert
- ☐ Der skalierte Input wird mit dem bereits skalierten Output-State multipliziert

101. Wie verarbeiten RNNs Sequenzen?

- ☐ Durch parallele Verarbeitung aller Elemente gleichzeitig.
- ☐ Durch Iteration über die Sequenzelemente.
- ☐ Durch Zufallsauswahl der Elemente.
- ☐ Durch Verarbeitung aller Elemente in einem einzigen Schritt.

102. Was bewahrt ein RNN während der Verarbeitung einer Sequenz?

- ☐ Einen festen Wert.
- ☐ Einen Zustand, der Informationen über die bisher gesehenen Elemente enthält.
- ☐ Eine zufällige Zahl.
- ☐ Eine Konstante.

103. Wie wird der Zustand eines RNNs zwischen zwei unabhängigen Sequenzen behandelt?

- ☐ Er wird beibehalten.
- ☐ Er wird zurückgesetzt.
- ☐ Er wird zufällig initialisiert.
- ☐ Er wird verdoppelt.

104. Welche Form hat die Eingabe eines RNNs?

- ☐ Ein 1D-Tensor.
- ☐ Ein 2D-Tensor.
- ☐ Ein 3D-Tensor.
- ☐ Ein skalarer Wert.

105. Wofür steht die Abkürzung LSTM?

- ☐ Long Short-Term Memory
- ☐ Large Scale Training Model
- ☐ Linear Sequential Training Model
- ☐ Layered Sequential Training Model

106. Welches Problem löst LSTM?

- ☐ Das Problem der Überanpassung.
- ☐ Das Problem des verschwindenden Gradienten.
- ☐ Das Problem der zu grossen Parameteranzahl.
- ☐ Das Problem der langsamen Trainingszeit.

107. Was ist die Funktion des Carry states c_t in einem LSTM?

- ☐ Er speichert zufällige Werte.
- ☐ Er trägt Informationen über Zeitschritte hinweg.
- ☐ Er initialisiert die Gewichte.
- ☐ Er berechnet die Ausgabe direkt.

108. Wie wird der nächste Carry state c_{t+1} in einem LSTM berechnet?

- ☐ Durch Addition der Eingabe und des vorherigen Zustands.
- ☐ Durch Addition von gewichteten neuen und gewichteten bestehenden irrelevanten Informationen
- ☐ Durch Multiplikation mit einer Konstanten.
- ☐ Durch Anwendung einer Aktivierungsfunktion.

109. Welche Komponenten (Gates) hat eine LSTM-Zelle?

- ☐ Eingabe-Gate, Ausgabe-Gate, Vergessens-Gate
- ☐ Aktivierungsfunktion, Verlustfunktion, Optimierer
- ☐ Hidden Layer, Output Layer, Input Layer
- ☐ Bias, Gewichte, Aktivierungsfunktion

110. Was ist die Hauptaufgabe des Carry-States in einem LSTM?

- ☐ Die Aktualisierung des versteckten Zustands
- ☐ Die Steuerung, welche Informationen beibehalten oder vergessen werden
- ☐ Die Berechnung der Ausgabe
- ☐ Die Anwendung der Aktivierungsfunktion

111. Was ist rekurrentes Dropout?

- ☐ Eine Methode zur Beschleunigung des Trainings, durch zufälliges Auslassen von Datenpunkten.
- ☐ Eine Methode zur Reduktion der Parameteranzahl durch Auslassen jedes n-ten Werts.
- ☐ Eine Methode zur Bekämpfung von Overfitting durch Anwendung einer konstanten Dropout-Maske über die Zeit.
- ☐ Eine Methode zur Erhöhung der Klassifikationsgenauigkeit durch Dropout irrelevanter Parameter.

112. Warum ist das Stapeln von rekurrenten Schichten nützlich?

- ☐ Es reduziert die Trainingszeit.
- ☐ Es erhöht die Repräsentationskraft des Netzwerks.
- ☐ Es verringert die Anzahl der Parameter.
- ☐ Es beschleunigt die Vorwärtspropagation.

113. Wie funktioniert ein bidirektionales RNN?

- ☐ Es verarbeitet die Eingabesequenz in einer zufälligen Reihenfolge.
- ☐ Es verarbeitet die Eingabesequenz in beiden Richtungen (chronologisch und antichronologisch).
- ☐ Es verarbeitet die Eingabesequenz nur in umgekehrter Reihenfolge.
- ☐ Es verarbeitet die Eingabesequenz parallel.

114. Warum können bidirektionale RNNs die Leistung verbessern?

- ☐ Weil sie die Trainingszeit verkürzen.
- ☐ Weil sie unterschiedliche Repräsentationen der Daten nutzen.
- ☐ Weil sie die Anzahl der Parameter reduzieren.
- ☐ Weil sie die Klassifikationsgenauigkeit direkt erhöhen.

115. Warum ist es wichtig, zuerst einfache Modelle auszuprobieren?

- ☐ Weil sie schneller trainiert werden können.
- ☐ Weil sie eine Basis für die Erklärung der Nutzung eines komplexeren Modells bieten.
- ☐ Weil sie weniger Parameter haben.
- ☐ Weil sie immer die besten Ergebnisse liefern.

116. Welche Art von Dropout sollte in RNNs angewendet werden?

- ☐ Zufälliges Dropout.
- ☐ Zeitlich konstantes Dropout.
- ☐ Kein Dropout.
- ☐ Nur Dropout auf die Eingabeschicht.

117. Wann sind bidirektionale RNNs möglicherweise nicht geeignet?

- ☐ Wenn die jüngste Vergangenheit viel informativer ist als der Beginn der Sequenz.
- ☐ Wenn man Voraussagen treffen möchte, basierend auf die Vergangenheit.
- ☐ Wenn es sich um Textdaten handelt.
- ☐ Wenn die Sequenz in umgekehrter Reihenfolge verarbeitet werden soll.

118. Was ist ein Tensor im Kontext von Deep Learning?

- ☐ Eine spezielle Aktivierungsfunktion
- ☐ Ein Container für numerische Daten
- ☐ Ein Optimierungsverfahren
- ☐ Ein neuronales Netz

119. Wie nennt man einen Tensor mit nur einer Zahl?

- ☐ Vektor
- ☐ Matrix
- ☐ Skalar
- ☐ Tabelle

120. Woraus besteht ein 2D-Tensor typischerweise?

- ☐ Aus einer Liste von Skalaren
- ☐ Aus einer Liste von Matrizen
- ☐ Aus einem Array von Vektoren
- ☐ Aus einem Array von Bildern

121. Welcher Begriff beschreibt die Anzahl der Achsen eines Tensors?

- ☐ Tiefe
- ☐ Breite
- ☐ Rang (Rank)
- ☐ Grösse

122. Welcher dieser Tensors hat die Form eines Würfels aus Zahlen?

- ☐ 1D-Tensor
- ☐ 2D-Tensor
- ☐ 3D-Tensor
- ☐ 0D-Tensor

123. Welches Datenformat wird typischerweise für Zeitreihendaten (Timeseries) verwendet?

- ☐ 2D-Tensor mit (samples, features)
- ☐ 3D-Tensor mit (samples, timesteps, features)
- ☐ 4D-Tensor mit (samples, height, width, channels)
- ☐ 5D-Tensor mit (samples, channels, timesteps, features)

124. Wie ist die typische Struktur eines 4D-Tensors für Bilddaten?

- ☐ (samples, features, labels, channels)
- ☐ (samples, channels, height, width)
- ☐ (samples, timesteps, channels)
- ☐ (features, samples, height, width)

125. Welche Aussage über Vektordaten ist korrekt?

- ☐ Vektordaten werden immer als 1D-Tensoren gespeichert
- ☐ Vektordaten bestehen meist aus 3D-Tensoren
- ☐ Vektordaten sind in der Regel als 2D-Tensoren organisiert, mit (samples, features)
- ☐ Vektordaten enthalten nur Zeitangaben

126. Was ist ein Hauptvorteil von 1D Convolutional Neural Networks gegenüber RNNs?

- ☐ Höhere Genauigkeit bei Sprachverarbeitung
- ☐ Geringerer Rechenaufwand
- ☐ Berücksichtigen Langzeitabhängigkeiten besser
- ☐ Können keine Sequenzen verarbeiten

127. Welche Aufgabe erfüllt ein 1D Convolutional Layer?

- ☐ Klassifiziert ganze Bilder
- ☐ Analysiert lokale Muster in Sequenzen
- ☐ Berechnet statistische Kennzahlen
- ☐ D. Sortiert Daten nach Zeit

128. Was beschreibt "Translation Invariance" bei CNNs?

- ☐ Fähigkeit, Muster unabhängig von ihrer Position zu erkennen
- ☐ Fähigkeit, Wörter zwischen Sprachen zu übersetzen
- ☐ Anpassung der Lernrate über Zeit
- ☐ Reduktion der Trainingszeit

129. Was bewirkt eine Pooling-Schicht im Zusammenhang mit 1D CNNs?

- ☐ Verstärkt Signale
- ☐ Fügt Rauschen hinzu
- ☐ Reduziert die Sequenzlänge
- ☐ Wandelt Text in Zahlen um

130. Was ist ein Nachteil von 1D CNNs im Vergleich zu RNNs?

- ☐ Höherer Speicherverbrauch
- ☐ Unfähigkeit, Reihenfolge global zu berücksichtigen
- ☐ Nur für Bilder geeignet
- ☐ Sehr langsam in der Verarbeitung

131. Welche Layer werden oft am Ende eines 1D CNN verwendet, um Klassifikation zu ermöglichen?

- ☐ Recurrent Layers
- ☐ Dense Layers
- ☐ Embedding Layers
- ☐ Batch-Normalization-Layers

132. Was beschreibt ein "Window" bei einer 1D Convolution?

- ☐ Ein Modellparameter
- ☐ Ein Trainingsdatensatz
- ☐ Ein lokaler Abschnitt der Eingabesequenz
- ☐ Ein Visualisierungstool

133. Was passiert bei Max-Pooling in einer 1D Sequenz?

- ☐ Mittelwertbildung über alle Werte
- ☐ Auswahl des höchsten Werts in einem Patch
- ☐ Duplizieren von Sequenzteilen
- ☐ Addition aller Patches

134. Welche Eigenschaft unterscheidet RNNs von 1D CNNs?

- ☐ RNNs sind translational invariant
- ☐ RNNs verarbeiten Daten ohne Reihenfolge
- ☐ RNNs sind auf die Reihenfolge der Eingaben sensitiv
- ☐ RNNs verwenden keine Gewichtungen

135. Was kann eine Kombination aus CNN und RNN ermöglichen?

- ☐ Schnelleres Training ohne Qualitätsverlust
- ☐ Gleichzeitige Text- und Bildverarbeitung
- ☐ Berücksichtigung von Reihenfolge bei langen Sequenzen
- ☐ Automatische Hyperparameterwahl

136. Welche Art von Daten eignet sich typischerweise NICHT für 1D CNNs?

- ☐ Tonaufnahmen
- ☐ Zeitreihen
- ☐ Texte
- ☐ Farbbilder

137. Was ist eine wichtige Designentscheidung bei 1D CNNs?

- ☐ Anzahl der LSTM-Zellen
- ☐ Fenstergröße (Kernel size)
- ☐ Reihenfolge der Wörter
- ☐ Verwendung von Backpropagation

138. Warum nutzt man in der Praxis oft mehrere gestapelte Convolution- und Pooling-Schichten?

- ☐ Zur Visualisierung von Eingabedaten
- ☐ Um Speicherplatz zu sparen
- ☐ Um komplexere und längerfristige Muster zu erkennen
- ☐ Um die Lernrate konstant zu halten

139. Was ist das Grundprinzip eines Multi-Input Modells?

- ☐ Nutzung eines einzelnen Datentyps
- ☐ Kombination verschiedener Eingangsdaten in einem Modell
- ☐ Verwendung eines einzigen Neurons
- ☐ Reduktion des Speichers

140. Welche Technik wird verwendet, um unterschiedliche Input-Datenströme zusammenzuführen?

- ☐ Dropout
- ☐ Add oder Concatenate Layer
- ☐ Batch Normalization
- ☐ Activation Layer

141. Was ermöglicht ein Multi-Input Modell?

- ☐ Einsatz von nur einem Datentyp
- ☐ Nutzung mehrerer unabhängiger Datenquellen
- ☐ Reduktion der Modellgrösse
- ☐ Entfernen von Bias

142. Wie funktioniert die naive Methode zur Kombination multimodaler Inputs?

- ☐ Ignorieren einzelner Datenquellen
- ☐ Training separater Modelle mit anschliessender Mittelung der Vorhersagen
- ☐ Reduktion der Anzahl der Inputs
- ☐ Verstärkung einzelner Features

143. Warum gilt das naive Verfahren bei Multi-Input Modellen als nachteilig?

- ☐ Es benötigt zu viele Daten
- ☐ Es erkennt keine Korrelationen zwischen den Eingaben
- ☐ Es verhindert Training
- ☐ Es nutzt zu viele Layer

144. Was zeichnet Multi-Output Modelle aus?

- ☐ Sie erzeugen nur eine einzelne Vorhersage
- ☐ Sie können mehrere Zielattribute gleichzeitig vorhersagen
- ☐ Sie arbeiten nur mit Bildern
- ☐ Sie sind auf Audio beschränkt

145. Warum kann ein Multi-Output Modell Vorteile gegenüber mehreren Einzelmodellen haben?

- ☐ Geringere Trainingsdaten
- ☐ Gemeinsames Lernen von Zusammenhängen zwischen den Zielattributen
- ☐ Komplettes Vermeiden von Dropout
- ☐ Reduzierung der Anzahl an GPUs

10. Wie werden Verluste in einem Multi-Output Modell behandelt?

- ☐ Sie werden einzeln pro Zielattribut berechnet
- ☐ Es wird ein Gesamtverlust über alle Ausgaben gebildet
- ☐ Es wird eine gewichtete Gesamtverlust als Summe über alle Verlustfunktionen der einzelnen Zielattribute
- ☐ Nur der grösste Verlust zählt

146. Welche Herausforderung besteht bei Multi-Input Modellen?

- Die Daten müssen die gleiche Form haben
- Unterschiedliche Eingabestrukturen müssen sinnvoll integriert werden
- Nur Bilddaten sind erlaubt
- Keine

147. Was ist ein Vorteil von Multi-Input Modellen im Vergleich zu klassischen Modellen?

- ☐ Sie reduzieren die Anzahl der Epochen
- ☐ Sie ermöglichen die gleichzeitige Verarbeitung verschiedener Datenformate
- ☐ Sie verhindern alle Fehler
- ☐ Sie benötigen keine Labels

148. Welche Aussage ist korrekt für ein Multi-Output Modell?

- ☐ Alle Ausgaben sind unabhängig
- ☐ Korrelationen zwischen Zielattributen können genutzt werden
- ☐ Nur ein Zielattribut wird gleichzeitig vorhergesagt
- ☐ Modelle sind auf Textdaten beschränkt

149. Wofür stehen die Begriffe "Multiple Inputs" und "Multiple Outputs"?

- ☐ Für sequentielle Daten
- ☐ Für kombinierte Nutzung mehrerer Eingaben und Vorhersage mehrerer Ziele
- ☐ Für Zufallsergebnisse
- ☐ Für Layer-Normalisierung

150. Was passiert, wenn mehrere Modelle in der naiven Multi-Input Methode trainiert werden?

- ☐ Jedes Modell wird separat trainiert und das Ergebnis gemittelt
- ☐ Es wird nur ein Modell trainiert
- ☐ Die Modelle verhindern sich gegenseitig
- ☐ Das Training wird abgebrochen

151. Welche Aussage trifft auf Multi-Input Modelle zu?

- ☐ Sie nutzen immer nur eine einzige Datenquelle
- ☐ Sie erfordern zwingend Bilddaten
- ☐ Sie ermöglichen die Verarbeitung verschiedener Datenquellen im gleichen Modell
- ☐ Sie ersetzen vollständig CNNs

152. Eine Residualverbindung besteht darin, frühere Darstellungen in den nachgelagerten Datenfluss wieder einzuspeisen, sodass die Ausgabe...

- ☐ ... nur die ursprüngliche Eingabe enthält
- ☐ ... nur die Transformation durch die Schicht enthält
- ☐ ... sowohl die ursprüngliche Eingabe als auch die Transformation durch die Schicht enthält
- ☐ ... entweder die ursprüngliche Eingabe oder die Transformation durch die Schicht enthält

153. Was ist die Hauptidee hinter dem Attention-Mechanismus?

- ☐ Alle Input-Teile gleichmässig zu gewichten.
- ■ Bestimmten Teilen des Inputs mehr Bedeutung beizumessen als anderen.
- ☐ Die Input Sequence zu verkürzen.
- ☐ Die Dimensionalität der Input-Daten zu erhöhen.

154. Was ermöglicht der Attention-Mechanismus einem Modell in Bezug auf die Input Features?

- ☐ Nur das erste Feature der Sequence zu berücksichtigen.
- ■ Features context-aware zu interpretieren.
- ☐ Die Anzahl der Features zu reduzieren.
- ☐ Alle Features zufällig zu gewichten.

155. Wozu dient der erste Schritt im Self-Attention Mechanismus, wenn man beispielsweise das Wort "station" in einem Satz betrachtet?

- ☐ Die grammatikalische Rolle von "station" zu bestimmen.
- ■ Die Relevancy Scores zwischen "station" und jedem anderen Wort im Satz zu berechnen.
- ☐ Die häufigsten Wörter neben "station" zu finden.
- ☐ "station" durch ein Synonym zu ersetzen.

156. Was repräsentiert der resultierende Vektor nach Anwendung von Self-Attention auf ein Wort?

- ☐ Eine isolierte Darstellung des Wortes.
- ■ Eine kontextualisierte Darstellung des Wortes unter Berücksichtigung seines Surrounding Context.
- ☐ Die semantische Ähnlichkeit zu einem festen Ankerwort.
- ☐ Die Frequenz des Wortes im gesamten Dataset.

157. Was bedeutet der Begriff "Multi-Head" im Kontext von Multi-Head Attention?

- ☐ Das Modell hat mehrere Output Layers.
- ■ Der Self-Attention Layer operiert auf mehreren unabhängigen Repräsentationen (Sub-Spaces) gleichzeitig.
- ☐ Es werden mehrere verschiedene Attention-Mechanismen kombiniert.
- ☐ Das Modell kann mehrere Sprachen gleichzeitig verarbeiten.

158. Welchen Nutzen hat die Verwendung von unabhängigen "Heads" in Multi-Head Attention?

- ☐ Jeder Head lernt, die gleiche Art von Features zu erkennen, um die Robustheit zu erhöhen.
- ■ Es hilft dem Layer, verschiedene Gruppen von Features oder Beziehungen für jedes Token zu lernen.
- ☐ Es reduziert den Speicherbedarf des Modells.
- ☐ Es vereinfacht die mathematische Formulierung der Attention.

159. Was ermöglicht es einem Transformer, die Beziehung zwischen Wörtern zu verstehen, die weit voneinander entfernt in einem Satz stehen?

- ☐ Die rekursive Struktur des Modells.
- ■ Der Self-Attention Mechanismus, der alle Wortpaare direkt vergleicht.
- ☐ Ein fester Context Window Ansatz.
- ☐ Die Verwendung von Convolutional Layers.

160. Wenn ein Modell lernt, "mehr Aufmerksamkeit" auf bestimmte Features zu richten, was bedeutet das für die internen Weights des Attention-Mechanismus?

- ■ Die Weights für diese Features werden tendenziell grösser.
- ☐ Die Weights für diese Features werden tendenziell kleiner.
- ☐ Die Weights bleiben unverändert, nur die Activations ändern sich.
- ☐ Die Weights werden randomisiert.

161. Welchen Vorteil bietet Multi-Head Attention gegenüber Single-Head Attention?

- ☐ Es ist immer schneller in der Berechnung.
- ■ Es ermöglicht dem Modell, verschiedene Aspekte oder Subtypen von Beziehungen gleichzeitig in unterschiedlichen Sub-Spaces zu erfassen.
- ☐ Es benötigt weniger Parameter.
- ☐ Es ist einfacher zu implementieren.

162. Was ist der Hauptzweck des Transformer-Encoders?

- ☐ Text in Sprache umzuwandeln
- ■ Eine Eingabesequenz in eine kontextbewusste Repräsentation zu überführen
- ☐ Zufällige Textgenerierung
- ☐ Bilderkennung

163. Was ist eine Residual Connection im Transformer-Encoder?

- [ ] Eine Technik zur Optimierung von Verlustfunktionen
- [ ] Eine Methode zur Rechenzeitverkürzung
- [x] Eine Verbindung, die den ursprünglichen Input beibehält und addiert
- [ ] Eine Art von Dropout-Verfahren

164. Was macht die Multi-Head Attention im Transformer-Encoder?

- [ ] Rechnet nur den Mittelwert von Eingabewerten
- [ ] Erzeugt eine einfache gewichtete Summe
- [x] Führt mehrere parallele Selbstaufmerksamkeiten durch
- [ ] Führt eine lineare Transformation durch

165. Wozu dienen Dense Layers im Transformer-Encoder?

- [ ] Zum Filtern irrelevanter Wörter
- [x] Zum Lernen von Repräsentationen
- [ ] Zum Erhöhen der Eingabesequenz
- [ ] Zum Maskieren von Tokens

166. Was ist das Ergebnis des Transformer-Encoders?

- [ ] Eine einzelne Zahl
- [ ] Eine fixierte Textausgabe
- [x] Eine Sequenz von kontextabhängigen Embeddings
- [ ] Ein Vektor mit Zufallswerten

167. Was ermöglicht der Einsatz mehrerer Köpfe (Heads) in der Multi-Head Attention?

- [ ] Geringeren Speicherverbrauch
- [ ] Höhere Trainingsgeschwindigkeit
- [x] Lernen verschiedener Aspekte der Eingabe
- [ ] Verringerung der Sequenzlänge

168. Wozu dient Positional Encoding in einem Transformer?

- [ ] Um die Länge einer Sequenz zu erhöhen
- [ ] Um Wortbedeutungen zu normalisieren
- [x] Um Positionsinformationen in die Eingabedaten zu integrieren
- [ ] Um Stoppwörter zu entfernen

169. Was enthält ein Embedding eines Tokens im Transformer?

- ☐ Nur die Wortbedeutung
- ■ Wortvektor + Positionsvektor
- ☐ Nur der Positionsvektor
- ☐ Zufällige Initialisierung

170. Was passiert, wenn Positional Encoding weggelassen wird?

- ■ Das Modell kann Reihenfolgeinformationen nicht berücksichtigen
- ☐ Das Modell funktioniert besser
- ☐ Der Speicherbedarf sinkt
- ☐ Der Output ist immer zufällig

171. Wofür wurden Transformer ursprünglich entwickelt?

- ☐ Textklassifikation
- ☐ Bildverarbeitung
- ■ Maschinelle Übersetzung
- ☐ Clustering

172. Welche beiden Hauptkomponenten hat ein Transformer-Modell?

- ☐ Attention-Modul und CNN
- ☐ Input-Schicht und LSTM
- ■ Encoder und Decoder
- ☐ Klassifikator und Regulator

173. Was geschieht während der Inferenzphase eines Seq2Seq-Modells?

- ☐ Die Ausgabe wird direkt aus dem Zieltext gelesen
- ☐ Die Zielsequenz wird komplett vorausgeladen
- ■ Die Ausgabe wird Schritt für Schritt generiert
- ☐ Die Decoder-Schicht wird übersprungen

174. Welche Rolle spielt der Decoder im Transformer-Modell?

- ☐ Kodiert den Quelltext
- ☐ Normalisiert die Positionsembeddings
- ■ Generiert neue Tokens auf Basis von Eingabesequenz und bisherigen Tokens
- ☐ Extrahiert Schlüsselwörter

175. Was ist der Hauptzweck des Transformer-Encoders?

- [x] Kontextabhängige Repräsentationen von Eingabetokens erzeugen
- [ ] Text generieren
- [ ] Zielsequenzen in Quellsequenzen übersetzen
- [ ] Das nächste Token in einer Zielsequenz vorhersagen

176. Welcher Mechanismus verwendet der Transformer-Encoder um den Kontext zu berechnen?

- [ ] Rekurrentes Gedächtnis
- [x] Self-Attention
- [ ] Convolutional-Filter
- [ ] Pooling

177. Was versucht der Decoder vorherzusagen?

- [ ] Alle Tokens gleichzeitig
- [ ] Vorheriges Token
- [x] Token N+1 basierend auf Tokens 0 bis N
- [ ] Ein zufälliges Token

178. Was wäre die Folge, wenn der Decoder während des Trainings vollen Zugriff auf die gesamte Zielsequenz hätte?

- [ ] Schnellere Konvergenz
- [ ] Overfitting
- [x] Perfekte Trainingsgenauigkeit, aber nutzlose Inferenz
- [ ] Underfitting