Ôn tập cuối kỳ
Môn: Deep Learning

Prof: Đào Thành Mạnh, daothanhmanh12@gmail.com

Câu hỏi trắc nghiệm

Problem 1: Câu 1

Điều nào sau đây là **SAI** về Deep Learning và Machine Learning?

(A) Thuật toán Deep Learning dễ hiểu hơn so với Thuật toán Machine Learning

(B) Không có phương án nào đúng

(C) Data augmentation dễ thực hiện hơn trong Deep Learning so với Machine Learning

(D) Deep Learning giải quyết hiệu quả các bài toán thị giác máy tính

Problem 2: Câu 2

Đâu là một loại mạng neural?

(A) Tất cả các phương án trên

(B) Autoencoders

(C) Capsule Neural Networks

(D) CNN (Convolutional Neural Network)

Problem 3: Câu 3

Điều nào sau đây là **SAI** về mạng neural?

(A) Trong quá trình lan truyền ngược, chúng ta cập nhật trọng số bằng thuật toán gradient descent

(B) Chúng ta có thể sử dụng các hàm kích hoạt khác nhau trong các lớp khác nhau

(C) Chúng ta có thể sử dụng các thuật toán gradient descent khác nhau trong các epoch khác nhau

(D) Không có phương án nào đúng

Problem 4: Câu 4

Điều nào sau đây là **SAI** về hàm kích hoạt bước (step activation function)?

(A) Nó có tính chất tuyến tính

(B) Nó chỉ xuất ra 0 hoặc 1

(C) Không có phương án nào đúng

(D) Nó còn được gọi là hàm kích hoạt ngưỡng (Threshold activation function)

Problem 5: Câu 5

Điều nào sau đây là **SAI** về hàm kích hoạt?

(A) Hàm kích hoạt giúp đạt được tính phi tuyến trong mạng neural sâu

(B) Các hàm kích hoạt thường được sử dụng là step, sigmoid, tanh, ReLU và softmax

(C) Hàm kích hoạt giúp giảm vấn đề overfitting

(D) Chúng còn được gọi là hàm nén (squashing functions) vì chúng nén đầu ra trong một phạm vi nhất định

Problem 6: Câu 6

Đầu ra của hàm kích hoạt bước (threshold activation function) nằm trong khoảng:

(A) 0 đến 1

(B) -1 đến 1

(C) Chỉ 0 hoặc 1

(D) Chỉ -1 hoặc 1

Problem 7: Câu 7

Điều nào sau đây là **SAI** về hàm kích hoạt sigmoid và tanh?

(A) Không có phương án nào đúng

(B) Cả hai đều là hàm kích hoạt phi tuyến

(C) Đầu ra của cả sigmoid và tanh đều mượt, liên tục và có thể đạo hàm được

(D) Đầu ra của sigmoid nằm trong khoảng -1 đến 1 trong khi đầu ra của tanh nằm trong khoảng 0 đến 1

Problem 8: Câu 8

Điều nào sau đây là **SAI** về Dropout?

(A) Dropout có thể được sử dụng trong các lớp đầu vào, ẩn và đầu ra

(B) Dropout là một siêu tham số

(C) Không có phương án nào đúng

(D) Dropout được triển khai trên từng lớp trong mạng

Problem 9: Câu 9

Điều nào sau đây là **SAI** về Dropout?

(A) Dropout là một tham số có thể học được trong mạng

(B) Dropout giới thiệu sự thưa thớt trong mạng

(C) Dropout tăng độ chính xác và hiệu suất của mô hình

(D) Dropout làm cho quá trình huấn luyện trở nên nhiễu

Problem 10: Câu 10

Điều nào sau đây là **ĐÚNG** về Dropout?

(A) Dropout có thể so sánh với kỹ thuật boosting trong machine learning

(B) Dropout chỉ nên được triển khai trong giai đoạn huấn luyện, không phải trong giai đoạn kiểm tra

(C) Dropout phức tạp hơn về mặt tính toán so với các phương pháp regularization L1 và L2

(D) Dropout nên được triển khai trong cả giai đoạn huấn luyện và kiểm tra

Problem 11: Câu 11

Điều nào sau đây là **ĐÚNG** về cực tiểu địa phương và cực tiểu toàn cục?

(A) Tinh chỉnh siêu tham số đóng vai trò quan trọng trong việc tránh cực tiểu toàn cục

(B) Tất cả các phương án trên

(C) Lý tưởng, SGD nên đạt đến cực tiểu địa phương và không bị kẹt ở cực tiểu toàn cục

(D) Đôi khi cực tiểu địa phương cũng tốt như cực tiểu toàn cục

Problem 12: Câu 12

Đâu là cách để tránh cực tiểu địa phương?

(A) Tất cả các phương án trên

(B) Sử dụng momentum và học tập thích ứng

(C) Tăng tốc độ học

(D) Thêm một chút nhiễu khi cập nhật trọng số

Problem 13: Câu 13

Biến thể SGD nào sau đây KHÔNG dựa trên học tập thích ứng?

(A) Adagrad

(B) Nesterov

(C) AdaDelta

(D) RMSprop

Problem 14: Câu 14

Điều nào sau đây là **ĐÚNG** về khởi tạo trọng số?

(A) Nếu trọng số quá cao, nó có thể dẫn đến vanishing gradient

(B) Nếu trọng số quá thấp, nó có thể dẫn đến exploding gradient

(C) Tất cả các phương án trên

(D) Mô hình có thể không bao giờ hội tụ do khởi tạo trọng số sai

Problem 15: Câu 15

Điều nào sau đây là **ĐÚNG** về Momentum?

(A) Tất cả các phương án trên

(B) Nó giúp hội tụ nhanh hơn

(C) Nó giúp tăng tốc SGD theo hướng phù hợp

(D) Nó giúp SGD tránh cực tiểu địa phương

Problem 16: Câu 16

Điều nào sau đây là **SAI** về lớp Pooling trong CNN?

(A) Nó thực hiện down-sampling một hình ảnh, giảm kích thước nhưng vẫn giữ lại thông tin quan trọng

(B) Nó thực hiện trích xuất đặc trưng và phát hiện các thành phần của hình ảnh như cạnh, góc, v.v.

(C) Đầu ra của lớp convolutional là đầu vào của lớp pooling

(D) Lớp pooling phải được thêm vào sau mỗi lớp convolutional

Problem 17: Câu 17

Đâu là lý do hợp lý để không sử dụng mạng kết nối đầy đủ (fully connected networks) cho nhận dạng hình ảnh?

(A) Nó tạo ra nhiều tham số hơn so với CNN

(B) CNN hiệu quả hơn về mặt hiệu suất và độ chính xác cho nhận dạng hình ảnh

(C) Tất cả các phương án trên

(D) Nó dễ bị overfit hơn so với CNN

Problem 18: Câu 18

Điều nào sau đây là **SAI** về Padding trong CNN?

(A) Có hai loại padding: Zero Padding và Valid Padding (không padding)

(B) Trong zero padding, chúng ta đệm hình ảnh bằng số 0 để không mất thông tin về cạnh

(C) Padding được sử dụng để ngăn chặn mất thông tin về cạnh và góc trong quá trình convolution

(D) Không có sự giảm kích thước khi sử dụng valid padding

Problem 19: Câu 19

Điều nào sau đây là **SAI** về Kernel trong CNN?

(A) Kernel trích xuất các đặc trưng đơn giản trong các lớp đầu và các đặc trưng phức tạp trong các lớp sâu hơn

(B) Kernel có thể được sử dụng trong cả lớp convolutional và lớp pooling

(C) Kernel liên tục trượt qua hình ảnh để trích xuất các thành phần hoặc mẫu khác nhau của hình ảnh

(D) Không có phương án nào đúng

Problem 20: Câu 20

Đâu KHÔNG phải là một siêu tham số trong CNN?

(A) Số lượng lớp convolutional

(B) Số lượng và kích thước của kernel trong một lớp convolutional

(C) Padding trong một lớp convolutional (zero hoặc valid padding)

(D) Kích thước mã nén (code size for compression)

Problem 21: Câu 21

Điều nào sau đây là **SAI** về LSTM?

(A) Không có phương án nào đúng

(B) LSTM là một phần mở rộng của RNN, mở rộng bộ nhớ của nó

(C) LSTM cho phép RNN học các phụ thuộc dài hạn

(D) LSTM giải quyết vấn đề exploding gradients trong RNN

Problem 22: Câu 22

Đâu KHÔNG phải là ứng dụng của RNN?

(A) Giao dịch thuật toán (Algorithmic trading)

(B) Tạo chú thích hình ảnh (Image captioning)

(C) Nén hình ảnh (Image compression)

(D) Hiểu chuỗi DNA (Understanding DNA sequence)

Problem 23: Câu 23

Đâu KHÔNG phải là ứng dụng của RNN?

(A) Phát hiện bất thường (Anomaly detection)

(B) Dự báo thời tiết (Weather prediction)

(C) Dự đoán thị trường chứng khoán (Stock market prediction)

(D) Dự đoán chuỗi thời gian (Time series prediction)

Problem 24: Câu 24

GAN có thể được chia thành bao nhiêu phần?

(A) 3

(B) 2

(C) 1

(D) 4

Problem 25: Câu 25

Từ nào sau đây được sử dụng để làm quen với các mô hình sinh (generative models) và giải thích cách dữ liệu được tạo ra bằng các mô hình xác suất?

(A) Generative

(B) Discriminator

(C) Adversarial

(D) Networks

Problem 26: Câu 26

Đâu KHÔNG phải là một ví dụ về mô hình sinh (generative model)?

(A) Naive Bayes

(B) Discriminator models

(C) GAN models

(D) PixelRNN/PixelCNN

Problem 27: Câu 27

Dạng chuẩn của YOLO là gì?

(A) Không có phương án nào đúng

(B) You Once Look Only

(C) YOU Look Once

(D) You Only Look Once

Problem 28: Câu 28

Các thành phần của hệ thống nhận dạng đối tượng là gì?

(a) Cơ sở dữ liệu mô hình (Model database)

(b) Bộ giả thuyết (Hypothesizer)

(c) Bộ phát hiện đặc trưng (Feature detector)

(d) Bộ kiểm tra giả thuyết (Hypothesis verifier)

(A) (a) và (c)

(B) (a) và (b)

(C) Không có phương án nào đúng

(D) (c) và (d)

(E) Tất cả (a), (b), (c) và (d)

Problem 29: Câu 29

Hệ thống nhận dạng khuôn mặt được sử dụng trong:

(a) Nhận dạng sinh trắc học (Biometric identification)

(b) Giao diện giữa người và máy tính (Human and computer interface)

(A) (a)

(B) Không có phương án nào đúng

(C) Cả (a) và (b)

(D) (b)

Problem 30: Câu 30

Điều nào sau đây là **ĐÚNG** về các loại Vectorization trong NLP?

(A) Count vectorization xem xét số lượng và trọng số của mỗi từ trong văn bản

(B) Tất cả các phương án trên

(C) N-gram vectorization xem xét ngữ cảnh của từ tùy thuộc vào giá trị của N

(D) TF-IDF vectorization xem xét cả số lượng và ngữ cảnh của các từ trong văn bản

Problem 31: Câu 31

Đâu là ứng dụng của NLP?

(A) Google Assistant

(B) Chatbots

(C) Google Translate

(D) Tất cả các phương án trên

Problem 32: Câu 32

Điều nào sau đây là **ĐÚNG** về NLP?

(A) NLP có thể được sử dụng để lọc thư rác, phân tích cảm xúc và dịch máy

(B) Tất cả các phương án trên

(C) Chúng ta phải quan tâm đến Cú pháp (Syntax), Ngữ nghĩa (Semantics) và Ngữ dụng (Pragmatics) trong NLP

(D) Các tác vụ tiền xử lý bao gồm Tokenization, Stemming, Lemmatization và Vectorization

Problem 33: Câu 33

Kỹ thuật nào sau đây có thể được sử dụng để giảm overfitting của mô hình?

(a) Data augmentation

(b) Dropout

(c) Batch Normalization

(d) Sử dụng Adam thay vì SGD

(A) Tất cả các phương án (a), (b), (c) và (d) đều đúng

(B) (a) và (b)

(C) (a), (b) và (c)

(D) (a)

(E) (b)

Problem 34: Câu 34

Điều nào sau đây là **ĐÚNG** về Dropout?

(A) Dropout dẫn đến sự thưa thớt trong các trọng số được huấn luyện

(B) Trong giai đoạn kiểm tra, dropout được áp dụng với xác suất giữ ngược

(C) Xác suất giữ càng lớn của một lớp, thì regularization của các trọng số trong lớp đó càng mạnh

(D) Tất cả các phương án (a), (b) và (c) đều đúng

Problem 35: Câu 35

Bạn đang huấn luyện một mạng GAN để tạo ra hình ảnh của các loài bò sát. Tuy nhiên, bạn nghĩ rằng Generator của bạn có thể đang gặp vấn đề mode collapse. Đâu là các dấu hiệu

của vấn đề này?

(a) Generator chỉ tạo ra hình ảnh của rồng Komodo

(b) Generator loss dao động

(c) Generator loss vẫn thấp trong khi discriminator loss cao

(d) Discriminator có độ chính xác cao trên hình ảnh thật nhưng độ chính xác thấp trên hình ảnh giả

(A) (a) và (b)

(B) (c)

(C) Tất cả các phương án (a), (b), (c) và (d) đều đúng

(D) (c) và (d)

(E) (d)

Problem 36: Câu 36

Đâu KHÔNG phải là một kỹ thuật tiền xử lý trong NLP?

(A) Chuyển đổi thành chữ thường

(B) Stemming và Lemmatization

(C) Loại bỏ stop words

(D) Phân tích cảm xúc (Sentiment analysis)

(E) Loại bỏ dấu câu

Problem 37: Câu 37

Bạn đang huấn luyện một mạng GAN để tạo ra hình ảnh của các loài bò sát. Tuy nhiên, bạn nghĩ rằng Generator của bạn có thể đang gặp vấn đề mode collapse. Đâu là các dấu hiệu của vấn đề này?

(a) Generator chỉ tạo ra hình ảnh của rồng Komodo

(b) Generator loss không ổn định

(c) Generator loss vẫn thấp trong khi discriminator loss cao

(d) Discriminator có độ chính xác cao trên hình ảnh thật nhưng độ chính xác thấp trên hình ảnh giả

(A) (a) và (b)

(B) (a), (b) và (c)

(C) (a)

(D) (b)

(E) Tất cả các phương án trên

Problem 38: Câu 38

Đâu là các mệnh đề **ĐÚNG** về lớp CONV? (Chọn tất cả các phương án đúng)

(a) Số lượng trọng số phụ thuộc vào độ sâu của khối đầu vào

(b) Số lượng bias bằng số lượng bộ lọc

(c) Tổng số tham số phụ thuộc vào stride

(d) Tổng số tham số phụ thuộc vào padding

(A) (b) và (c)

(B) (c) và (d)

(C) (a) và (b)

(D) Tất cả các phương án đều đúng

(E) (a) và (d)

Câu hỏi tự luận

Problem 39: Câu 39

Bạn được giao nhiệm vụ xây dựng một bộ phân loại nhận đầu vào là một hình ảnh poster phim và phân loại nó vào một trong bốn thể loại: hài, kinh dị, hành động và lãng mạn. Bạn được cung cấp một tập dữ liệu lớn các poster phim, trong đó mỗi poster tương ứng với một phim thuộc chính xác một trong các thể loại này. Mô hình của bạn hiện có độ chính xác 100% trên tập huấn luyện và 96% trên tập kiểm tra! Bây giờ bạn quyết định mở rộng mô hình để phân loại các poster phim thuộc nhiều thể loại. Mỗi poster có thể thuộc nhiều thể loại; ví dụ, poster của phim "Lát mặt: Nhà có khách" thuộc cả hai thể loại "hài" và "hành động". Hãy đề xuất một cách để gán nhãn cho các poster mới, trong đó mỗi ví dụ có thể đồng thời thuộc nhiều lớp? Để tránh thêm công việc, bạn quyết định huấn luyện lại một mô hình mới với cùng kiến trúc (hàm kích hoạt đầu ra softmax với hàm mất mát cross-entropy). Giải thích tại sao điều này là có vấn đề?

Giải:

Để gán nhãn cho các poster phim thuộc nhiều thể loại, chúng ta có thể sử dụng phương pháp **multi-label classification**. Thay vì sử dụng hàm kích hoạt softmax (vốn chỉ phù hợp cho bài toán phân loại đơn nhãn), chúng ta có thể sử dụng hàm kích hoạt **sigmoid** cho mỗi lớp đầu ra. Mỗi đầu ra sẽ là một giá trị xác suất độc lập, biểu thị khả năng poster thuộc về một thể loại cụ thể. Hàm mất mát phù hợp cho bài toán này là **binary cross-entropy loss**,

được tính riêng cho mỗi lớp và sau đó cộng lại.
Lý do tại sao việc sử dụng softmax và cross-entropy là có vấn đề:

- **Softmax** giả định rằng mỗi mẫu chỉ thuộc về một lớp duy nhất. Nó chuẩn hóa đầu ra sao cho tổng xác suất của tất cả các lớp bằng 1. Điều này không phù hợp với bài toán multi-label classification, nơi một mẫu có thể thuộc về nhiều lớp cùng lúc.
- **Cross-entropy loss** được thiết kế cho bài toán phân loại đơn nhãn. Khi áp dụng cho bài toán multi-label classification, nó sẽ không thể xử lý chính xác các trường hợp một mẫu thuộc nhiều lớp, dẫn đến việc huấn luyện mô hình không hiệu quả và kết quả không chính xác.

Problem 40: Câu 40

Giải thích sự khác biệt giữa gradient descent, stochastic gradient descent (SGD) và mini-batch gradient descent.
Gradient Descent (GD):

- Gradient Descent là một thuật toán tối ưu hóa được sử dụng để cực tiểu hóa hàm mất mát bằng cách cập nhật các tham số của mô hình dựa trên gradient của hàm mất mát.
- Trong mỗi bước lặp, gradient của hàm mất mát được tính toán trên toàn bộ tập dữ liệu huấn luyện. Sau đó, các tham số được cập nhật theo hướng ngược lại của gradient.
- Công thức cập nhật tham số:
$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta)$$
trong đó:
  - θ là vector tham số của mô hình.
  - η là tốc độ học (learning rate).
  - $\nabla_\theta J(\theta)$ là gradient của hàm mất mát $J(\theta)$ đối với θ.
- Ưu điểm: Hội tụ ổn định và chính xác.
- Nhược điểm: Chậm khi tập dữ liệu lớn vì phải tính toán gradient trên toàn bộ dữ liệu.

Stochastic Gradient Descent (SGD):

- Stochastic Gradient Descent là một biến thể của Gradient Descent, trong đó gradient được tính toán trên một mẫu dữ liệu ngẫu nhiên (thay vì toàn bộ tập dữ liệu) trong mỗi bước lặp.
- Công thức cập nhật tham số:
$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta; x^{(i)}, y^{(i)})$$
trong đó:
  - $(x^{(i)}, y^{(i)})$ là một mẫu dữ liệu ngẫu nhiên từ tập huấn luyện.
- Ưu điểm: Nhanh hơn GD vì chỉ cần tính toán gradient trên một mẫu dữ liệu.

- Nhược điểm: Có thể dao động nhiều và không ổn định do gradient được tính trên một mẫu ngẫu nhiên.

Mini-batch Gradient Descent:

- Mini-batch Gradient Descent là sự kết hợp giữa GD và SGD. Trong mỗi bước lặp, gradient được tính toán trên một tập con nhỏ (mini-batch) của tập dữ liệu thay vì toàn bộ tập dữ liệu hoặc một mẫu duy nhất.
- Công thức cập nhật tham số:

$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta; x^{(i:i+n)}, y^{(i:i+n)})$$

  trong đó:

  - $(x^{(i:i+n)}, y^{(i:i+n)})$ là một mini-batch gồm n mẫu dữ liệu từ tập huấn luyện.
- Ưu điểm:
  - Nhanh hơn GD vì chỉ cần tính toán gradient trên một tập con nhỏ.
  - Ổn định hơn SGD vì gradient được tính trên nhiều mẫu dữ liệu hơn.
- Nhược điểm: Cần chọn kích thước mini-batch phù hợp để cân bằng giữa tốc độ và độ ổn định.

Do đó:

- **Gradient Descent (GD)**: Tính gradient trên toàn bộ tập dữ liệu, hội tụ ổn định nhưng chậm.
- **Stochastic Gradient Descent (SGD)**: Tính gradient trên một mẫu dữ liệu ngẫu nhiên, nhanh nhưng dao động nhiều.
- **Mini-batch Gradient Descent**: Tính gradient trên một tập con nhỏ (mini-batch), cân bằng giữa tốc độ và độ ổn định.

Problem 41: Câu 41

Xem xét mạng neural tích chập được định nghĩa bởi các lớp trong cột bên trái dưới đây. Điền vào hình dạng của khối đầu ra và số lượng tham số ở mỗi lớp. Bạn có thể viết hình dạng kích hoạt theo định dạng (H, W, C), trong đó H, W, C lần lượt là chiều cao, chiều rộng và số kênh. Trừ khi được chỉ định, giả sử padding 1, stride 1 nếu cần.
Ký hiệu:

- **CONVx-N** biểu thị một lớp tích chập với N bộ lọc có chiều cao và chiều rộng bằng x.
- **POOL-n** biểu thị một lớp max-pooling $n \times n$ với stride bằng n và padding 0.
- **FLATTEN** làm phẳng đầu vào của nó, tương tự như torch.nn.flatten / tf.layers.flatten.
- **FC-N** biểu thị một lớp kết nối đầy đủ với N neuron.

Lớp	Hình dạng khối kích hoạt	Số lượng tham số
Input	$32 \times 32 \times 3$	0
CONV3-8	$32 \times 32 \times 8$	$(3 \times 3 \times 3 + 1) \times 8 = 80$
Leaky ReLU	$32 \times 32 \times 8$	$32 \times 32 \times 8 \times 1 = 8196$
POOL-2	$16 \times 16 \times 8$	0
BATCHNORM	$16 \times 16 \times 8$	$16 \times 16 \times 8 \times 2 = 4096$
CONV3-16	$16 \times 16 \times 16$	$(3 \times 3 \times 8 + 1) \times 16 = 1168$
Leaky ReLU	$16 \times 16 \times 16$	$16 \times 16 \times 16 \times 1 = 4096$
POOL-2	$8 \times 8 \times 16$	0
FLATTEN	1024	0
FC-10	10	10250

Problem 42: Câu 42

Đưa ra một phương pháp để chống lại vanishing gradient trong mạng neural kết nối đầy đủ. Giả sử chúng ta đang sử dụng mạng với hàm kích hoạt Sigmoid được huấn luyện bằng SGD.
Giải:
Vấn đề vanishing gradient xảy ra khi gradient của hàm mất mát trở nên rất nhỏ trong quá trình lan truyền ngược, đặc biệt là trong các mạng sâu. Điều này làm cho việc cập nhật trọng số trở nên khó khăn, dẫn đến việc huấn luyện mô hình chậm hoặc không hiệu quả. Dưới đây là một số phương pháp để chống lại vanishing gradient:

1. **Sử dụng hàm kích hoạt khác:**
    - Thay vì sử dụng hàm kích hoạt Sigmoid, chúng ta có thể sử dụng các hàm kích hoạt khác như **ReLU (Rectified Linear Unit)** hoặc **Leaky ReLU**. Các hàm này không bị bão hòa ở các giá trị đầu vào lớn, giúp gradient không bị giảm quá mức.
    - ReLU có công thức: $f(x) = \max(0, x)$, giúp gradient luôn bằng 1 khi $x > 0$, tránh được vấn đề vanishing gradient.
2. **Khởi tạo trọng số phù hợp:**
    - Sử dụng các phương pháp khởi tạo trọng số thông minh như **He initialization** hoặc **Xavier initialization**. Các phương pháp này giúp trọng số được khởi tạo trong một phạm vi phù hợp, tránh việc gradient quá nhỏ ngay từ đầu.
    - Ví dụ, He initialization khởi tạo trọng số với phương sai $\frac{2}{n}$, trong đó n là số lượng neuron ở lớp trước đó.
3. **Sử dụng Batch Normalization:**
    - Batch Normalization giúp chuẩn hóa đầu vào của mỗi lớp bằng cách điều chỉnh và chia tỷ lệ các giá trị kích hoạt. Điều này giúp giữ cho gradient ổn định và ngăn chặn vanishing gradient.
    - Batch Normalization cũng giúp tăng tốc độ huấn luyện và cải thiện hiệu suất của mô hình.
4. **Sử dụng các kiến trúc mạng phù hợp:**

- Trong các mạng sâu, việc sử dụng các kiến trúc như **ResNet (Residual Networks)** có thể giúp giảm thiểu vanishing gradient. ResNet sử dụng các kết nối tắt (skip connections) để truyền gradient trực tiếp qua các lớp, giúp duy trì gradient ở mức ổn định.

5. **Điều chỉnh tốc độ học (Learning Rate):**

   - Sử dụng các phương pháp điều chỉnh tốc độ học thích ứng như **Adam**, **RMSprop**, hoặc **Adagrad**. Các phương pháp này tự động điều chỉnh tốc độ học dựa trên gradient, giúp tránh được vấn đề vanishing gradient.

Problem 43: Câu 43

Làm thế nào để chúng ta huấn luyện mạng sâu?
Quá trình huấn luyện mạng sâu: Huấn luyện mạng sâu là quá trình tối ưu hóa các tham số của mạng (trọng số và bias) để cực tiểu hóa hàm mất mát (loss function). Quá trình này bao gồm các bước chính sau:

1. **Khởi tạo tham số:**

   - Các tham số của mạng (trọng số và bias) được khởi tạo ngẫu nhiên hoặc sử dụng các phương pháp khởi tạo thông minh như **Xavier initialization** hoặc **He initialization**.
   - Việc khởi tạo tham số phù hợp giúp tránh các vấn đề như vanishing gradient hoặc exploding gradient.

2. **Truyền xuôi (Forward Propagation):**

   - Dữ liệu đầu vào được truyền qua các lớp của mạng để tính toán đầu ra dự đoán.
   - Mỗi lớp thực hiện phép biến đổi tuyến tính (linear transformation) và áp dụng hàm kích hoạt (activation function) để tạo ra đầu ra của lớp đó.
   - Cuối cùng, đầu ra của mạng được so sánh với nhãn thực tế để tính toán hàm mất mát.

3. **Tính toán hàm mất mát (Loss Function):**

   - Hàm mất mát đo lường sự khác biệt giữa đầu ra dự đoán của mạng và nhãn thực tế.
   - Các hàm mất mát phổ biến bao gồm:
     - **Mean Squared Error (MSE)**: Cho bài toán hồi quy.
     - **Cross-Entropy Loss**: Cho bài toán phân loại.

4. **Truyền ngược (Backpropagation):**

   - Gradient của hàm mất mát đối với các tham số của mạng được tính toán bằng cách sử dụng quy tắc chuỗi (chain rule).
   - Gradient được truyền ngược từ lớp đầu ra về các lớp đầu vào, giúp xác định mức độ ảnh hưởng của từng tham số đến hàm mất mát.

5. **Cập nhật tham số (Parameter Update):**

   - Các tham số của mạng được cập nhật dựa trên gradient của hàm mất mát.
   - Công thức cập nhật tham số sử dụng thuật toán tối ưu hóa như Gradient Descent, Stochastic Gradient Descent (SGD), hoặc các biến thể như Adam, RMSprop:

     $$\theta = \theta - \eta \cdot \nabla_\theta J(\theta)$$

     trong đó:
     - θ là vector tham số của mạng.
     - η là tốc độ học (learning rate).
     - $\nabla_\theta J(\theta)$ là gradient của hàm mất mát $J(\theta)$ đối với θ.

6. **Lặp lại quá trình:**

   - Quá trình truyền xuôi, tính toán hàm mất mát, truyền ngược và cập nhật tham số được lặp lại trong nhiều epoch cho đến khi mô hình hội tụ (tức là hàm mất mát giảm đến mức chấp nhận được hoặc đạt được độ chính xác mong muốn).

7. **Đánh giá mô hình:**

   - Sau khi huấn luyện, mô hình được đánh giá trên tập kiểm tra (test set) để kiểm tra khả năng tổng quát hóa (generalization) của mô hình.
   - Các chỉ số đánh giá phổ biến bao gồm độ chính xác (accuracy), precision, recall, F1-score, v.v.

Problem 44: Câu 44

Giải thích sự khác biệt giữa hàm kích hoạt sigmoid và tanh.
Hàm kích hoạt Sigmoid:

- Công thức của hàm sigmoid:

  $$\sigma(x) = \frac{1}{1 + e^{-x}}$$

- Đầu ra của hàm sigmoid nằm trong khoảng từ 0 đến 1, tức là $\sigma(x) \in (0, 1)$.
- Hàm sigmoid thường được sử dụng trong các bài toán phân loại nhị phân (binary classification), nơi đầu ra cần được diễn giải như một xác suất.
- Nhược điểm của hàm sigmoid:
  - **Vanishing gradient**: Khi đầu vào x có giá trị lớn (dương hoặc âm), gradient của hàm sigmoid trở nên rất nhỏ, dẫn đến việc cập nhật tham số chậm hoặc không hiệu quả.
  - **Đầu ra không đối xứng**: Đầu ra của hàm sigmoid luôn dương, điều này có thể gây ra vấn đề trong việc tối ưu hóa mô hình.

Hàm kích hoạt Tanh (Hyperbolic Tangent):

- Công thức của hàm tanh:
$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$
- Đầu ra của hàm tanh nằm trong khoảng từ -1 đến 1, tức là $\tanh(x) \in (-1, 1)$.
- Hàm tanh thường được sử dụng trong các mạng neural vì nó có tính chất đối xứng quanh gốc tọa độ, giúp cải thiện tốc độ hội tụ so với hàm sigmoid.
- Nhược điểm của hàm tanh:
  - **Vanishing gradient**: Tương tự như sigmoid, hàm tanh cũng gặp phải vấn đề vanishing gradient khi đầu vào x có giá trị lớn.

So sánh giữa Sigmoid và Tanh:

- **Phạm vi đầu ra**:
  - Sigmoid: $\sigma(x) \in (0, 1)$.
  - Tanh: $\tanh(x) \in (-1, 1)$.
- **Tính đối xứng**:
  - Sigmoid: Không đối xứng, đầu ra luôn dương.
  - Tanh: Đối xứng quanh gốc tọa độ, đầu ra có thể âm hoặc dương.
- **Vanishing gradient**:
  - Cả hai hàm đều gặp phải vấn đề vanishing gradient khi đầu vào có giá trị lớn.
- **Ứng dụng**:
  - Sigmoid: Thường được sử dụng trong các bài toán phân loại nhị phân.
  - Tanh: Thường được sử dụng trong các mạng neural vì tính đối xứng và tốc độ hội tụ nhanh hơn.

Problem 45: Câu 45

Ma trận Jacobian là gì?
Định nghĩa: Ma trận Jacobian là một ma trận chứa các đạo hàm riêng bậc nhất của một hàm vector đối với các biến đầu vào của nó. Nó được sử dụng để mô tả sự thay đổi của các biến đầu ra khi các biến đầu vào thay đổi.
Công thức: Cho một hàm vector $\mathbf{F} : \mathbb{R}^n \to \mathbb{R}^m$, với:
$$\mathbf{F}(\mathbf{x}) = \begin{bmatrix} f_1(x_1, x_2, \ldots, x_n) \\ f_2(x_1, x_2, \ldots, x_n) \\ \vdots \\ f_m(x_1, x_2, \ldots, x_n) \end{bmatrix}$$
Ma trận Jacobian $\mathbf{J}$ của hàm $\mathbf{F}$ là một ma trận có kích thước $m \times n$, được định nghĩa như

sau:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

Trong đó, $\frac{\partial f_i}{\partial x_j}$ là đạo hàm riêng của hàm f_i đối với biến x_j.

Ý nghĩa:

- Ma trận Jacobian mô tả tốc độ thay đổi của các biến đầu ra $\mathbf{F}$ khi các biến đầu vào $\mathbf{x}$ thay đổi.
- Nó được sử dụng rộng rãi trong các bài toán tối ưu hóa, học máy, và các lĩnh vực khoa học kỹ thuật khác.

Ví dụ: Xét hàm vector $\mathbf{F} : \mathbb{R}^2 \to \mathbb{R}^2$ được định nghĩa như sau:

$$\mathbf{F}(x, y) = \begin{bmatrix} f_1(x, y) \\ f_2(x, y) \end{bmatrix} = \begin{bmatrix} x^2 + y^2 \\ x \cdot y \end{bmatrix}$$

Ma trận Jacobian của $\mathbf{F}$ là:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{bmatrix} = \begin{bmatrix} 2x & 2y \\ y & x \end{bmatrix}$$

Ứng dụng trong học sâu:

- Trong học sâu, ma trận Jacobian được sử dụng trong quá trình lan truyền ngược (backpropagation) để tính toán gradient của hàm mất mát đối với các tham số của mạng.
- Nó cũng được sử dụng trong các mạng neural phức tạp như mạng đối kháng sinh (GAN) và các mô hình biến đổi (transformers).

Problem 46: Câu 46

Giải thích về mạng đối kháng sinh (Generative Adversarial Network - GAN).
Định nghĩa: Mạng đối kháng sinh (GAN) là một kiến trúc mạng neural gồm hai mạng con: một mạng sinh (Generator) và một mạng phân biệt (Discriminator). Hai mạng này được huấn luyện đồng thời trong một quá trình đối kháng, nơi mạng sinh cố gắng tạo ra dữ liệu giống thật, trong khi mạng phân biệt cố gắng phân biệt giữa dữ liệu thật và dữ liệu giả.
Cấu trúc của GAN:

- **Generator (Mạng sinh):**

- Mạng sinh nhận đầu vào là một vector nhiễu ngẫu nhiên (thường được lấy từ phân phối chuẩn hoặc phân phối đều) và tạo ra dữ liệu giả (ví dụ: hình ảnh, văn bản, âm thanh).
- Mục tiêu của mạng sinh là tạo ra dữ liệu giả sao cho mạng phân biệt không thể phân biệt được với dữ liệu thật.

- **Discriminator (Mạng phân biệt):**
  - Mạng phân biệt nhận đầu vào là dữ liệu (có thể là dữ liệu thật hoặc dữ liệu giả) và cố gắng phân loại xem dữ liệu đó là thật hay giả.
  - Mục tiêu của mạng phân biệt là phân biệt chính xác giữa dữ liệu thật và dữ liệu giả.

Quá trình huấn luyện GAN:

- **Bước 1: Huấn luyện Discriminator:**
  - Mạng phân biệt được huấn luyện trên cả dữ liệu thật và dữ liệu giả do mạng sinh tạo ra.
  - Hàm mất mát của mạng phân biệt là:
    $$L_D = -\mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x)] - \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$
    trong đó:
    - $D(x)$ là xác suất mà mạng phân biệt dự đoán dữ liệu x là thật.
    - $G(z)$ là dữ liệu giả được tạo ra bởi mạng sinh từ vector nhiễu z.
- **Bước 2: Huấn luyện Generator:**
  - Mạng sinh được huấn luyện để tạo ra dữ liệu giả sao cho mạng phân biệt không thể phân biệt được với dữ liệu thật.
  - Hàm mất mát của mạng sinh là:
    $$L_G = -\mathbb{E}_{z \sim p_z(z)}[\log D(G(z))]$$
  - Mục tiêu của mạng sinh là cực đại hóa xác suất mà mạng phân biệt dự đoán dữ liệu giả là thật.
- **Lặp lại quá trình:**
  - Quá trình huấn luyện được lặp lại nhiều lần cho đến khi mạng sinh tạo ra dữ liệu giả chất lượng cao và mạng phân biệt không thể phân biệt được giữa dữ liệu thật và giả.

Ứng dụng của GAN:

- **Tạo hình ảnh:** GAN được sử dụng để tạo ra hình ảnh giống thật, ví dụ như hình ảnh khuôn mặt, cảnh quan, v.v.
- **Tạo video:** GAN có thể được sử dụng để tạo ra các video giả mạo chất lượng cao.
- **Tạo dữ liệu tổng hợp:** GAN được sử dụng để tạo ra dữ liệu tổng hợp phục vụ cho việc huấn luyện các mô hình học máy.
- **Nghệ thuật:** GAN được sử dụng để tạo ra các tác phẩm nghệ thuật kỹ thuật số.

Problem 47: Câu 47

LSTM (Long Short-Term Memory) khác biệt như thế nào so với RNN (Recurrent Neural Network)?
Sự khác biệt chính giữa LSTM và RNN:

- **Cấu trúc bên trong:**
  - **RNN**: Chỉ có một trạng thái ẩn đơn giản, được cập nhật tại mỗi bước thời gian dựa trên đầu vào hiện tại và trạng thái ẩn trước đó.
  - **LSTM**: Có cấu trúc phức tạp hơn với ba cổng (gates) chính:
    - **Cổng quên (Forget Gate)**: Quyết định thông tin nào từ trạng thái trước đó cần được giữ lại hoặc loại bỏ.
    - **Cổng đầu vào (Input Gate)**: Quyết định thông tin mới nào từ đầu vào hiện tại sẽ được thêm vào trạng thái.
    - **Cổng đầu ra (Output Gate)**: Quyết định thông tin nào từ trạng thái hiện tại sẽ được truyền đến đầu ra.
  - LSTM cũng có một bộ nhớ tế bào (cell state) giúp lưu trữ thông tin dài hạn.
- **Khả năng học các phụ thuộc dài hạn:**
  - **RNN**: Gặp khó khăn trong việc học các phụ thuộc dài hạn do vấn đề vanishing gradient, khi gradient trở nên rất nhỏ sau nhiều bước thời gian.
  - **LSTM**: Được thiết kế để giải quyết vấn đề này bằng cách sử dụng các cổng và bộ nhớ tế bào, giúp duy trì thông tin qua nhiều bước thời gian mà không bị mất mát đáng kể.
- **Hiệu suất trên các bài toán phức tạp:**
  - **RNN**: Thường hoạt động kém hiệu quả trên các bài toán yêu cầu học các phụ thuộc dài hạn, chẳng hạn như dịch máy, tạo văn bản, hoặc phân tích chuỗi thời gian dài.
  - **LSTM**: Thường đạt hiệu suất tốt hơn trên các bài toán này nhờ khả năng học và duy trì thông tin dài hạn.
- **Độ phức tạp tính toán:**
  - **RNN**: Có cấu trúc đơn giản hơn, do đó ít tốn tài nguyên tính toán hơn so với LSTM.
  - **LSTM**: Có cấu trúc phức tạp hơn với nhiều tham số hơn, dẫn đến chi phí tính toán cao hơn.

Problem 48: Câu 48

Sự khác biệt giữa same padding và valid padding là gì?
Same Padding:

- **Định nghĩa:** Same padding là kỹ thuật thêm các giá trị (thường là 0) xung quanh biên của đầu vào sao cho kích thước của đầu ra sau phép tích chập bằng với kích thước

của đầu vào.

- **Công thức tính toán:**
$$\text{Output Size} = \left\lceil \frac{\text{Input Size}}{\text{Stride}} \right\rceil$$
Trong đó, Input Size là kích thước của đầu vào, Stride là bước nhảy của bộ lọc (filter).
- **Ưu điểm:** Same padding giúp duy trì kích thước của đầu ra, điều này hữu ích khi muốn giữ nguyên kích thước của dữ liệu qua các lớp tích chập.
- **Nhược điểm:** Việc thêm padding có thể làm tăng số lượng tham số và chi phí tính toán.

Valid Padding:

- **Định nghĩa:** Valid padding là kỹ thuật không thêm bất kỳ giá trị nào xung quanh biên của đầu vào. Kích thước của đầu ra sau phép tích chập sẽ nhỏ hơn kích thước của đầu vào.
- **Công thức tính toán:**
$$\text{Output Size} = \left\lfloor \frac{\text{Input Size} - \text{Filter Size} + 1}{\text{Stride}} \right\rfloor$$
Trong đó, Filter Size là kích thước của bộ lọc.
- **Ưu điểm:** Valid padding không làm tăng kích thước của đầu vào, giúp giảm số lượng tham số và chi phí tính toán.
- **Nhược điểm:** Kích thước của đầu ra sẽ giảm dần qua các lớp tích chập, điều này có thể dẫn đến mất thông tin ở các lớp sâu hơn.

So sánh giữa Same Padding và Valid Padding:

- **Kích thước đầu ra:**
  - Same Padding: Kích thước đầu ra bằng kích thước đầu vào.
  - Valid Padding: Kích thước đầu ra nhỏ hơn kích thước đầu vào.
- **Thêm giá trị padding:**
  - Same Padding: Thêm giá trị padding xung quanh biên của đầu vào.
  - Valid Padding: Không thêm giá trị padding.
- **Chi phí tính toán:**
  - Same Padding: Tăng chi phí tính toán do thêm padding.
  - Valid Padding: Giảm chi phí tính toán do không thêm padding.
- **Ứng dụng:**
  - Same Padding: Thường được sử dụng khi muốn duy trì kích thước của đầu ra qua các lớp tích chập.
  - Valid Padding: Thường được sử dụng khi muốn giảm kích thước của đầu ra và tiết kiệm tài nguyên tính toán.

Problem 49: Câu 49

IoU (Intersection over Union) là gì?
Định nghĩa: IoU (Intersection over Union) là một chỉ số được sử dụng để đo lường mức độ chồng lấp giữa hai vùng (thường là hai hình chữ nhật hoặc hai hình dạng khác). Nó được sử dụng phổ biến trong các bài toán nhận diện đối tượng (object detection) và phân đoạn ảnh (image segmentation).
Công thức tính IoU: IoU được tính bằng tỷ lệ giữa diện tích phần giao nhau (intersection) và diện tích phần hợp nhất (union) của hai vùng:

$$\text{IoU} = \frac{\text{Intersection Area}}{\text{Union Area}}$$

Trong đó:

- Intersection Area: Diện tích phần giao nhau giữa hai vùng.
- Union Area: Diện tích phần hợp nhất của hai vùng, được tính bằng tổng diện tích của hai vùng trừ đi diện tích phần giao nhau.

Ví dụ: Giả sử chúng ta có hai hình chữ nhật A và B:

- Diện tích của hình chữ nhật A là Area_A.
- Diện tích của hình chữ nhật B là Area_B.
- Diện tích phần giao nhau giữa A và B là Intersection Area.

Khi đó, IoU được tính như sau:

$$\text{IoU} = \frac{\text{Intersection Area}}{\text{Area}_A + \text{Area}_B - \text{Intersection Area}}$$

Ý nghĩa của IoU:

- IoU có giá trị nằm trong khoảng từ 0 đến 1.
- IoU = 0: Hai vùng không có sự chồng lấp.
- IoU = 1: Hai vùng hoàn toàn trùng khớp.
- Trong các bài toán nhận diện đối tượng, IoU thường được sử dụng để đánh giá độ chính xác của các hộp giới hạn (bounding boxes) dự đoán so với hộp giới hạn thực tế.

Ứng dụng của IoU:

- **Nhận diện đối tượng:** IoU được sử dụng để đánh giá chất lượng của các hộp giới hạn dự đoán. Một hộp giới hạn được coi là chính xác nếu IoU lớn hơn một ngưỡng nhất định (thường là 0.5 hoặc 0.7).
- **Phân đoạn ảnh:** IoU được sử dụng để đo lường sự chồng lấp giữa các vùng phân đoạn dự đoán và vùng phân đoạn thực tế.
- **Non-Maximum Suppression (NMS):** IoU được sử dụng trong thuật toán NMS để loại bỏ các hộp giới hạn trùng lặp hoặc có sự chồng lấp lớn.

Problem 50: Câu 50

Trong NLP (Natural Language Processing), các kỹ thuật word embedding giúp thiết lập khoảng cách giữa 2 token như thế nào?
Cách word embedding thiết lập khoảng cách giữa các token:

- **Biểu diễn vector:**
  - Mỗi từ được biểu diễn bằng một vector trong không gian nhiều chiều (thường là 50, 100, 200, hoặc 300 chiều).
  - Các vector này được học từ dữ liệu văn bản thông qua các mô hình như Word2Vec, GloVe, hoặc FastText.
- **Khoảng cách giữa các vector:**
  - Khoảng cách giữa hai từ có thể được đo lường bằng các độ đo như khoảng cách Euclid, cosine similarity, hoặc các độ đo khác.
  - **Cosine similarity** là một độ đo phổ biến, được tính bằng công thức:

$$\text{cosine similarity}(A, B) = \frac{A \cdot B}{\|A\|\|B\|}$$

    trong đó:
    - A và B là hai vector biểu diễn hai từ.
    - $A \cdot B$ là tích vô hướng của hai vector.
    - $\|A\|$ và $\|B\|$ là độ dài (norm) của hai vector.
  - Cosine similarity có giá trị nằm trong khoảng từ -1 đến 1, trong đó 1 biểu thị hai vector hoàn toàn giống nhau, -1 biểu thị hai vector hoàn toàn ngược nhau, và 0 biểu thị hai vector không có mối quan hệ.
- **Mối quan hệ ngữ nghĩa:**
  - Các từ có ý nghĩa tương tự nhau sẽ có vector gần nhau trong không gian vector, tức là có cosine similarity cao.
  - Ví dụ, từ "king" và "queen" sẽ có vector gần nhau hơn so với từ "king" và "apple".
  - Các mối quan hệ ngữ nghĩa như đồng nghĩa, trái nghĩa, hoặc các mối quan hệ ngữ pháp (ví dụ: "king" - "man" + "woman" = "queen") cũng được phản ánh thông qua các phép toán vector.