

VIP Cheatsheet: Mẹo và thủ thuật



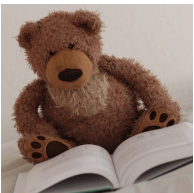

Afshine AMIDI và Shervine AMIDI



Ngày 17 tháng 5 năm 2020

Dịch bởi Hoàng Minh Tuấn, Trần Tuấn Anh và Đàm Minh Tiến

Xử lý dữ liệu

□ **Data augmentation** – Các mô hình học sâu thường cần rất nhiều dữ liệu để có thể được huấn luyện đúng cách. Việc sử dụng các kỹ thuật Data augmentation là khá hữu ích để có thêm nhiều dữ liệu hơn từ tập dữ liệu hiện thời. Những kỹ thuật chính được tóm tắt trong bảng dưới đây. Chính xác hơn, với hình ảnh đầu vào sau đây, đây là những kỹ thuật mà chúng ta có thể áp dụng:

Hình gốc	Lật	Xoay	Cắt ngẫu nhiên
			
- Hình ảnh không có bất kỳ sửa đổi nào	- Lật đối với một trục mà ý nghĩa của hình ảnh được giữ nguyên	- Xoay với một góc nhỏ - Mô phỏng hiệu chỉnh đường chân trời không chính xác	- Lấy nét ngẫu nhiên trên một phần của hình ảnh - Một số cách cắt ngẫu nhiên có thể được thực hiện trên một hàng

Dịch chuyển màu	Thêm nhiễu	Mất mát thông tin	Thay đổi độ tương phản
			
- Các sắc thái của RGB bị thay đổi một chút - Captures noise có thể xảy ra khi tiếp xúc với ánh sáng nhẹ	- Bổ sung nhiễu - Chịu được sự thay đổi chất lượng của các yếu tố đầu vào	- Các phần của hình ảnh bị bỏ qua - Mô phỏng khả năng mất của các phần trong hình ảnh	- Thay đổi độ sáng - Kiểm soát sự khác biệt do phơi sáng theo thời gian trong ngày

□ **Chuẩn hóa batch** – Đây là một bước của hyperparameter γ, β chuẩn hóa tập dữ liệu $\{x_i\}$. Bằng việc kí hiệu μ_B, σ_B^2 là trung bình và phương sai của tập dữ liệu ta muốn chuẩn hóa, nó được thực hiện như sau:

$$x_i \leftarrow \gamma \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta$$

Thường hoàn thành sau một lớp fully connected/nhân chập và trước lớp phi tuyến tính và mục đích cho phép tốc độ học cao hơn và giảm thiểu sự phụ thuộc vào khởi tạo

Huấn luyện mạng neural

□ **Epoch** – Trong ngữ cảnh huấn luyện mô hình, epoch là một thuật ngữ chỉ một vòng lặp mà mô hình sẽ duyệt toàn bộ tập dữ liệu huấn luyện để cập nhật trọng số của nó.

□ **Mini-batch gradient descent** – Trong quá trình huấn luyện, việc cập nhật trọng số thường không dựa trên toàn bộ tập huấn luyện cùng một lúc do độ phức tạp tính toán hoặc một điểm dữ liệu nhiễu. Thay vào đó, bước cập nhật được thực hiện trên các lô nhỏ (mini-batch), trong đó số lượng điểm dữ liệu trong một lô (batch) là một siêu tham số (hyperparameter) mà chúng ta có thể điều chỉnh.

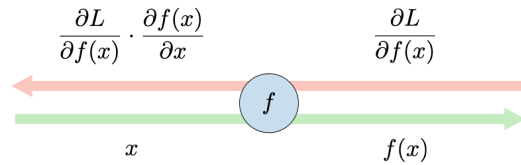
□ **Hàm mất mát** – Để định lượng cách thức một mô hình nhất định thực hiện, hàm mất mát L thường được sử dụng để đánh giá mức độ đầu ra thực tế y được dự đoán chính xác bởi đầu ra của mô hình là z .

□ **Cross-entropy loss** – Khi áp dụng phân loại nhị phân (binary classification) trong các mạng neural, cross-entropy loss $L(z, y)$ thường được sử dụng và được định nghĩa như sau:

$$L(z, y) = - \left[y \log(z) + (1 - y) \log(1 - z) \right]$$

□ **Lan truyền ngược** – Lan truyền ngược (backpropagation) là một phương thức để cập nhật

các trọng số trong mạng neural bằng cách tính toán đầu ra thực tế và đầu ra mong muốn. Đạo hàm tương ứng với từng trọng số w được tính bằng quy tắc chuỗi.

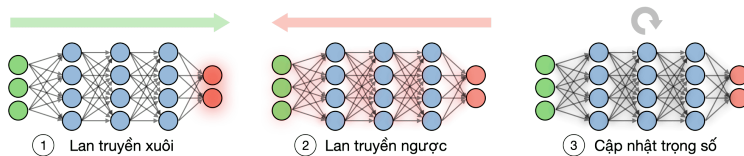


Sử dụng phương thức này, mỗi trọng số được cập nhật theo quy luật:

$$w \leftarrow w - \alpha \frac{\partial L(z, y)}{\partial w}$$

□ **Cập nhật trọng số** – Trong một mạng neural, các trọng số được cập nhật như sau:

- Bước 1: Lấy một loạt dữ liệu huấn luyện và thực hiện lan truyền xuôi (forward propagation) để tính toán mất mát
- Bước 2: Lan truyền ngược mất mát để có được độ dốc (gradient) của mất mát theo từng trọng số
- Bước 3: Sử dụng độ dốc để cập nhật trọng số của mạng.



Tinh chỉnh tham số

□ **Khởi tạo Xavier** – Thay vì khởi tạo trọng số một cách ngẫu nhiên, khởi tạo Xavier cho chúng ta một cách khởi tạo trọng số dựa trên một đặc tính độc nhất của kiến trúc mô hình.

□ **Transfer learning** – Huấn luyện một mô hình deep learning đòi hỏi nhiều dữ liệu và quan trọng hơn là rất nhiều thời gian. Sẽ rất hữu ích để tận dụng các trọng số đã được huấn luyện trước trên các bộ dữ liệu rất lớn mất vài ngày / tuần để huấn luyện và tận dụng nó cho trường hợp của chúng ta. Tùy thuộc vào lượng dữ liệu chúng ta có trong tay, đây là các cách khác nhau để tận dụng điều này:

Kích thước tập huấn luyện	Mô phỏng	Giải thích
Nhỏ		Cố định các tầng
Trung bình		huấn luyện trọng số trên hàm softmax
Lớn		Cố định hầu hết các tầng

□ **Tốc độ học** – Tốc độ học (learning rate), thường được kí hiệu là α hoặc đôi khi là η , cho biết mức độ thay đổi của các trọng số sau mỗi lần được cập nhật. Nó có thể được cố định hoặc thay đổi thích ứng. Phương thức phổ biến nhất hiện nay là Adam, đây là phương thức thích nghi với tốc độ học.

□ **Tốc độ học thích nghi** – Để cho tốc độ học thay đổi khi huấn luyện một mô hình có thể giảm thời gian huấn luyện và cải thiện giải pháp tối ưu số. Trong khi tối ưu hóa Adam (Adam optimizer) là kỹ thuật được sử dụng phổ biến nhất, nhưng những phương pháp khác cũng có thể hữu ích. Chúng được tổng kết trong bảng dưới đây:

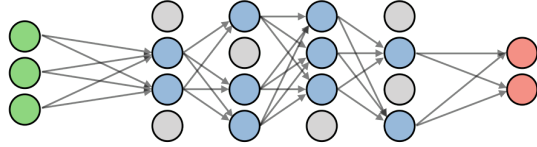
Phương thức	Giải thích	Cập nhật của w	Cập nhật của b
Momentum	- Làm giảm dao động - Cải thiện SGD - 2 tham số để tinh chỉnh	$w - \alpha v_{dw}$	$b - \alpha v_{db}$
RMSprop	- lan truyền Root Mean Square - Thuật toán tăng tốc độ học bằng kiểm soát dao động	$w - \alpha \frac{dw}{\sqrt{s_{dw}}}$	$b \leftarrow b - \alpha \frac{db}{\sqrt{s_{db}}}$
Adam	- Ước lượng Adaptive Moment - Các phương pháp phổ biến - 4 tham số để tinh chỉnh	$w - \alpha \frac{v_{dw}}{\sqrt{s_{dw}} + \epsilon}$	$b \leftarrow b - \alpha \frac{v_{db}}{\sqrt{s_{db}} + \epsilon}$

Chú ý: những phương pháp khác bao gồm Adadelta, Adagrad và SGD.

Chính quy

□ **Dropout** – Dropout là một kỹ thuật được sử dụng trong các mạng neural để tránh overfitting trên tập huấn luyện bằng cách loại bỏ các nơ-ron (neural) với xác suất $p > 0$. Nó giúp mô hình

không bị phụ thuộc quá nhiều vào một tập thuộc tính nào đó.

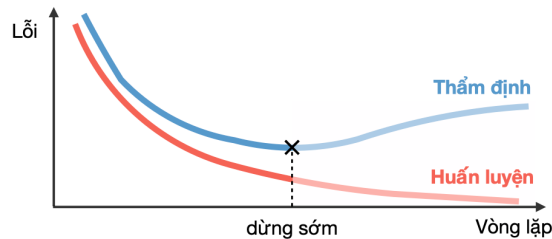


Ghi chú: hầu hết các frameworks deep learning đều có thiết lập dropout thông qua biến tham số 'keep' $1 - p$.

□ **Weight regularization** – Để đảm bảo rằng các trọng số không quá lớn và mô hình không bị overfitting trên tập huấn luyện, các kỹ thuật chính quy (regularization) thường được thực hiện trên các trọng số của mô hình. Những kĩ thuật chính được tổng kết trong bảng dưới đây:

LASSO	Ridge	Elastic Net
<ul style="list-style-type: none"> - Giảm hệ số về 0 - Tốt cho việc lựa chọn biến 	Làm cho hệ số nhỏ hơn	Đánh đổi giữa việc lựa chọn biến và hệ số nhỏ
<p>$\theta _1 \leq 1$</p>	<p>$\theta _2 \leq 1$</p>	<p>$(1 - \alpha) \theta _1 + \alpha \theta _2^2 \leq 1$</p>
$\dots + \lambda \theta _1$ $\lambda \in \mathbb{R}$	$\dots + \lambda \theta _2^2$ $\lambda \in \mathbb{R}$	$\dots + \lambda \left[(1 - \alpha) \theta _1 + \alpha \theta _2^2 \right]$ $\lambda \in \mathbb{R}, \alpha \in [0, 1]$

□ **Dừng sớm** – Kỹ thuật chính quy này sẽ dừng quá trình huấn luyện một khi mất mát trên tập kiểm định (validation) đạt đến một ngưỡng nào đó hoặc bắt đầu tăng.



Thói quen tốt

□ **Overfitting batch nhỏ** – Khi gỡ lỗi một mô hình, khá hữu ích khi thực hiện các kiểm tra nhanh để xem liệu có bất kỳ vấn đề lớn nào với kiến trúc của mô hình đó không. Đặc biệt, để

đảm bảo rằng mô hình có thể được huấn luyện đúng cách, một batch nhỏ (mini-batch) được truyền vào bên trong mạng để xem liệu nó có thể overfit không. Nếu không, điều đó có nghĩa là mô hình quá phức tạp hoặc không đủ phức tạp để thậm chí overfit trên batch nhỏ (mini-batch), chứ đừng nói đến một tập huấn luyện có kích thước bình thường.

□ **Kiểm tra gradient** – Kiểm tra gradient là một phương thức được sử dụng trong quá trình thực hiện lan truyền ngược của mạng neural. Nó so sánh giá trị của gradient phân tích (analytical gradient) với gradient số (numerical gradient) tại các điểm đã cho và đóng vai trò kiểm tra độ chính xác.

	Gradient số	Gradient phân tích
Công thức	$\frac{df}{dx}(x) \approx \frac{f(x+h) - f(x-h)}{2h}$	$\frac{df}{dx}(x) = f'(x)$
Bình luận	<ul style="list-style-type: none"> - Đắt, Mất mát phải được tính hai lần cho mỗi chiều - Được sử dụng để xác minh tính chính xác của việc triển khai phân tích - Đánh đổi trong việc chọn h không quá nhỏ (mất ổn định số) cũng không quá lớn (xấp xỉ độ dốc kém) 	<ul style="list-style-type: none"> - Kết quả 'Chính xác' - Tính toán trực tiếp - Được sử dụng trong quá trình triển khai cuối cùng

★ ★ ★