

Bảng 1: So sánh các phương pháp tiếp cận tính toán nghiêng về các vị trí khác nhau sử dụng Swin-T. Dấu (\*) là để đề cập tới độ chính xác top-1 trên tập ImageNet-1k được huấn luyện từ đầu. Các mô hình trong cột \* sẽ được sử dụng để đánh giá trên các nhiệm vụ phân loại ảnh của tập ImageNet-1K sử dụng độ phân giải của cửa sổ/ảnh lớn hơn, đánh dấu bởi dấu +. Với các kết quả này, chúng tôi ghi lại cả các kết quả có và không có fine-tuning. Các mô hình này cũng được sử dụng cho tinh chỉnh trên các bài toán phát hiện vật thể trên tập COCO và bài toán phân chia ngữ nghĩa (semantic segmentation) ADE20K.

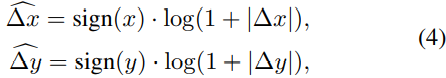
**Thiên lệch vị trí tương đối liên tục (continuous relative position bias)**: Thay vì trực tiếp tối ưu các thiên lệch được tham số hóa, phương pháp tiếp cận thiên lệch vị trí liên tục sử dụng một mạng nhỏ trên các tọa độ tương đối:



Với G là một mạng nhỏ, ví dụ một mạng MLP 2 lớp với một hàm kích hoạt ReLU mặc định ở giữa.

Mạng G sinh ra các giá trị bias cho các tọa độ tương đối tùy ý, và vì vậy có thể được chuyến tiếp một các tự nhiên để tinh chỉnh các nhiệm vụ với kích thước cửa sổ thay đổi tùy ý. Trong khi suy diễn, các giá trị bias ở mỗi vị trí tương đối có thể được tính toán trước và lưu lại như các tham số của mô hình, như vậy suy diễn cũng giống như cách tiếp cận bias được tham số hóa ban đầu.

**Các tọa độ không gian logarit**: Khi chuyển đổi theo các kích thước cửa sổ thay đổi lớn, một phần lớn của khoảng tọa độ tương đối cần được ngoại suy. Để giảm nhẹ vấn đề này, chúng tôi đề xuất sử dụng các tọa độ không gian logarit thay vì không gian tuyến tính như lúc đầu:



Với deltax, deltay và deltaxhat ,deltayhat là các tọa độ tương ứng ở không gian log và được scale tuyến tính.

Bằng việc sử dụng các tọa độ không gian log (log-spaced), khi ta chuyển đổi các bias vị trí tương đối theo các độ phân giải của cửa sổ, tỉ lệ ngoại suy yêu cầu sẽ nhỏ hơn bias mà sử dụng tọa độ không gian tuyến tính ban đầu. Lấy ví dụ chuyển đổi từ kích thước cửa sổ 8x8 được huấn luyện trước sang một kích thước cửa sổ 16x16 đã được tinh chỉnh, sử dụng các tọa độ gốc ban đầu, khoảng tọa độ đầu vào sẽ là từ [-7,7]x[-7,7] tới [-15,15]x[-15,15]. Tỉ lệ ngoại suy là khoảng ban đầu. Sử dụng các tọa độ không gian log, khoảng đầu vào sẽ từ tới . Tỉ lệ ngoại suy là khoảng ban đầu, tức là nhỏ hơn khoảng 4 lần tỉ lệ ngoại suy sử dụng tọa độ không gian tuyến tính ban đầu.

Bảng 1 so sánh hiệu năng chuyển đổi của các hướng tính toán khác nhau thiên về vị trí. Có thể thấy rằng hướng tiếp cận thiên vị trí liên tục không gian log (log-spaced CPB) cho hiệu năng tốt nhất, đặc biệt khi được chuyển đổi sang các kích thước cửa sổ lớn hơn.

3.4. Huấn luyện trước tự giám sát (self-supervised pre-training):

Các mô hình lớn hơn thì cũng cần nhiều dữ liệu hơn. Để giải quyết vấn đề cần nhiều dữ liệu, các mô hình thị giác lớn trước đây điển hình khai thác các tập dữ liệu có nhãn lớn nhưn JFT-3B. Trong bài báo này, chúng tôi khai thác phương pháp huấn luyện trước tự giám sát, SimMIM, để giải quyết yêu cầu cần dữ liệu có nhãn. Bằng cách này, chúng tôi đã huấn luyện thành công một mô hình Swin Transformer rất mạnh với 3 tỷ tham số và đạt được kết quả SOTA trên 4 tập đánh giá đại diện, với chỉ 70 triệu ảnh có nhãn (bằng 1/40 tập JFT-3B).

3.5. Cách triển khai để tiết kiệm bộ nhớ GPU:

Một vấn đề khác nằm ở việc sử dụng bộ nhớ GPU khá tốn kém với cách triển khai thông thường khi cả năng lực và độ phân giải đều lớn. Để giải quyết vấn đề về bộ nhớ, chúng tôi áp dụng các cách triển khai sau:

* Hàm tối ưu không độ trễ (Zero-redundancy Optimizer)(ZeRO). Trong cách triển khai các hàm tối ưu song song hóa dữ liệu, các tham số mô hình và trạng thái tối ưu được truyền trực tiếp tới mỗi GPU. Cách triển khai này rất ko phù hợp trong việc sử dụng bộ nhớ GPU, ví dụ, một mô hình với 3 tỉ tham số sẽ tiêu tốn 46GB bộ nhớ GPU khi một hàm tối ưu AdamW và tham số/trạng thái fp32 được sử dụng. Với một hàm tối ưu ZeRO, các tham số của mô hình và các trạng thái tối ưu tương ứng sẽ được phân chia và phân bố tới nhiều GPUs đồng thời, điều nãy sẽ giảm đáng kể việc sử dụng bộ nhớ GPU. Chúng tôi áp dụng DeepSpeed framework và sử dụng tùy chọn ZeRO stage-1 trong các thí nghiệm. Việc tối ưu này ít có ảnh hưởng tới tốc độ huấn luyện.
* Điểm kiểm tra sự kích hoạt (Activation check-pointing). Feature maps trong các lớp của Transformer cũng tiêu tốn khá nhiều bộ nhớ GPU, điều này có thể gây ra nghẽn cổ chai khi ảnh và độ phân giải của cửa sổ đều lớn. Kỹ thuật kiểm tra sự kích hoạt có thể giảm đáng kể tiêu tốn bộ nhớ, trong khi tốc độ huấn luyện có thể chậm hơn khoảng 30%.
* Tính toán tự tập trung tuần tự (sequential self-attention computation). Để huấn luyện các mô hình lớn trên độ phân giải rất lớn, ví dụ, một bức ảnh độ phân giải với kích thước cửa sổ , với GPU A100 100GB bộ nhớ cũng vẫn không đủ, thâm chi với cả 2 kỹ thuật tối ưu vừa trình bày ở trên. Chúng tôi nhận thấy rằng trong trường hợp này, mô đun tự tập trung (self-attention) gây ra sự nghẽn cổ chai. Để giảm nhẹ vấn đề này, chúng tôi triển khai tính toán tự tập trung một cách tuần tự, thay vì sử dụng cách tiếp cận tính toán theo tệp (batch) như trước đây. Cách tối ưu này được áp dụng cho các lớp trong 2 tầng đầu tiên và ít có ảnh hưởng tới tốc độ huấn luyện.

Với các cách triển khai này, chúng tôi có thể huấn luyện một mô hình 3 tỷ tham số sử dụng GPU A100-40GB của NVIDIA cho bài toán phát hiện vật thể trên tập COCO với độ phân giải đầu vào là và bài toán phân loại hành động Kinetics-400 với độ phân giải đầu vào là .

3.6. Các cấu hình cho mô hình:

Chúng tôi giữ nguyên thiết lập các tầng, khối và các kênh của mô hình Swin Transformer gốc với 4 thiết lập của Swin Transformer V2:

* Swin V2-T: C = 96, #. Khối = {2,2,6,2}
* Swin V2-S/B/L: C=96/128/192, #. Khối = {2,2,18,2}

Với C là số kênh trong tầng đầu tiên.

Chúng tôi tiếp tục tăng kích thước của Swin Transformer V2 lên cỡ lớn và cỡ rất lớn, tương ứng với 658 triệu tham số và 3 tỷ tham số:

* SwinV2-H: C = 352, #. Khối = {2, 2, 18,2}
* SwinV2-G: C = 512, #. Khối = {2,2,42,4}

Với SwinV2-H và SwinV2-G, chúng tôi thêm một lớp được chuẩn hóa trên nhánh chính sau mỗi 6 lớp. Để tiết kiệm thời gian thí nghiệm, chúng tôi chi tiến hành với SwinV2-G cho các thí nghiệm kích cỡ lớn. SwinV2-H được tiến hành cho các nghiên cứu song song về học tự giám sát.

4. Các thí nghiệm:

4.1. Các bài toán và tập dữ liệu:

Chúng tôi tiến hành các thí nghiệm trên tập phân loại ảnh ImageNet-1K (V1 và V2), tập phát hiện vật thể COCO, và tập phân loại ngữ nghĩa ADE20K. Với các thí nghiệm của mô hình 3 tỷ tham số, chugns tôi cũng thống kê về độ chính xác trên tập nhận diện hành động trên video Kinetics-400.

* Bài toán phân loại ảnh. Tập dữ liệu ImageNet-1K V1 và V2 được sử dụng để đánh giá. ImageNet-22K gồm có 14 triệu ảnh và 22 nghìn loại đối tượng được tùy chọn tiến hành cho huấn luyện trước. Với mô hình SwinV2-G lớn nhất được huấn luyện trước, một tập dữ liệu mở rộng ImageNet-22K được thu thập riêng với hơn 70 triệu ảnh đã được sử dụng. Với tập dữ liệu này, một quy trình loại bỏ ảnh trùng lặp dã được tiến hành để loại bỏ các bức ảnh chồng lấn nhau trên tập validation ImageNet-1K V1 và V2.
* Bài toán phát hiện vật thể. COCO được sử dụng để đánh giá. Với các thí nghiệm cho mô hình lớn nhất, chúng tôi tiến hành một bước huấn luyện trước việc phát hiện vật thể sử dụng tập dữ liệu Object 365 v2, ở giữa bước huấn luyện trước cho phân loại hình ảnh và bước tinh chỉnh trên tập COCO.
* Semantic segmentation. Tập dữ liệu ADE20K được sử dụng.
* Phân loại hành động từ video. Tập Kinetics-400 (K400) được sử dụng để đánh giá.

Các thiết lập về tinh chỉnh và huấn luyện trước sẽ được đề cập chi tiết trong phần phụ lục.

4.2. Mở rộng các thí nghiệm:

Chúng tôi trước hết trình bày về các kết quả trên các đánh giá trực quan bằng việc tăng kích thước của mô hình tới 3 tỷ tham số và độ phân giải cao cho cửa sổ/ảnh.

Các thiết lập cho thí nghiệm với mô hình SwinV2-G. Chúng tôi sử dụng độ phân giải nhỏ hơn cho ảnh với 192x192 pixel trong bước huấn luyện trước để tiết kiệm chi phí huấn luyện. Chúng tôi tiến hành bước huấn luyện trước theo 2 bước. Đầu tiên, mô hình được huấn luyện trước sử dụng một phương pháp tự giám sát trên tập dữ liệu mở rộng ImageNet-22K-ext với 20 epochs. Sau đó, mô hình được huấn luyện trước với 30 epochs sử dụng bài toán phân loại hình ảnh trên tập dữ liệu này. Các thiết lập về tinh chỉnh và huấn luyện trước sẽ được mô tả trong phần phụ lục.

Trong các phần tiếp theo, chúng tôi trình bày về độ chính xác của SwinV2-G trên đánh giá thị giác đại diện (representative vision benchmarks). Lưu ý rằng vì mục đích chính của chúng tôi là để tìm ra làm thế nào để dễ dàng mở rộng năng lực của mô hình và độ phân giải cửa sổ, và liệu các bài toán về thị giác có được lợi ích từ năng lực lớn hơn này không, chúng tôi không so sánh riêng về độ phức tạp hoặc dữ liệu huấn luyện trước ở đây.

Các kết quả của bài toán phân loại hình ảnh trên tập ImageNet-1K. Bảng 2 so sánh SwinV2-G với các mô hình thị giác tốt nhất trước đây trên ImageNet-1K V1 và V2. SwinV2-G là mô hình thị giác lớn nhất trong số này. Nó đạt độ chính xác top-1 84% trên đánh giá ImageNet V2, 0.7% cao hơn mô hình tốt nhất trước đây (83.3%).