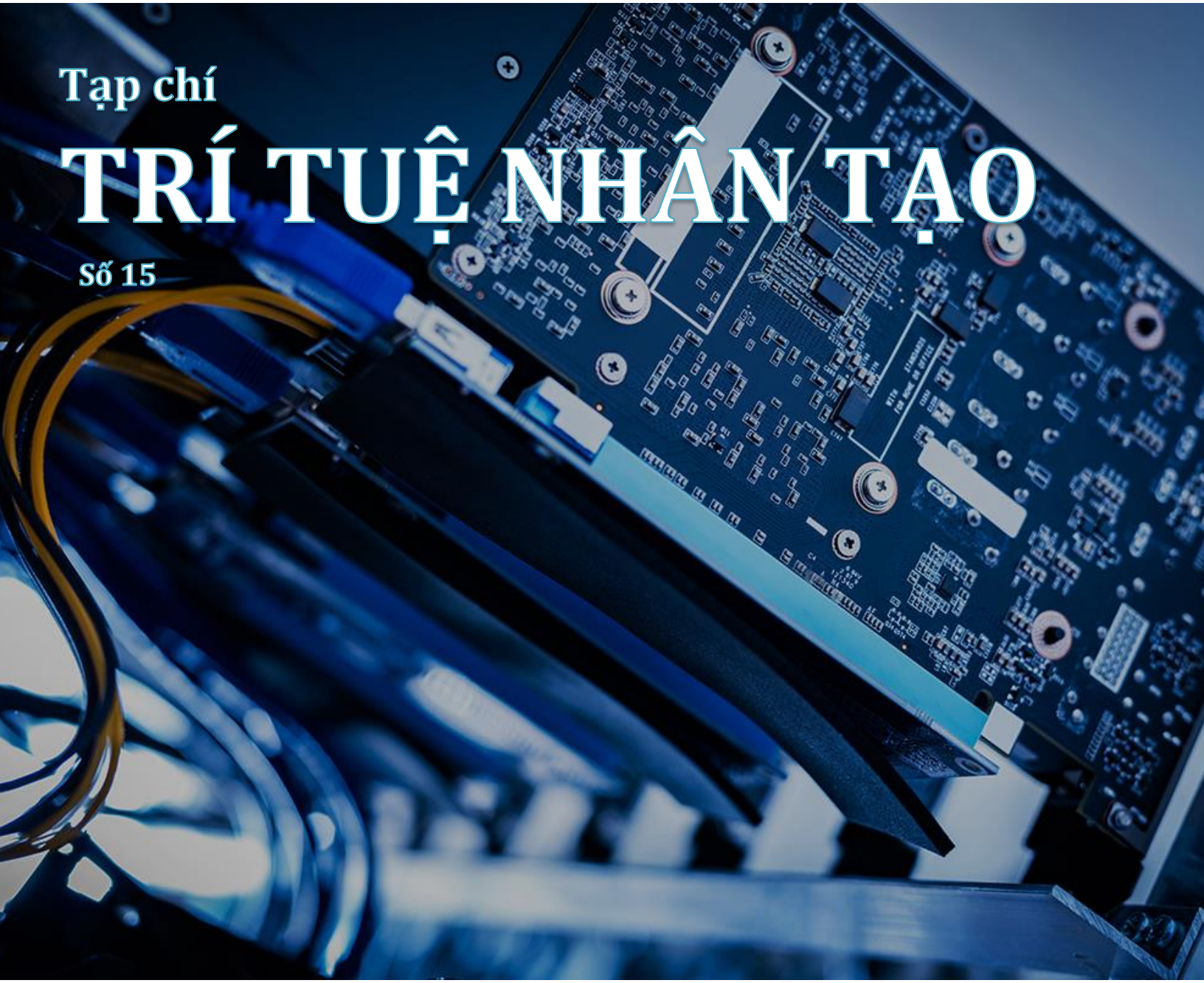


Tạp chí

TRÍ TUỆ NHÂN TẠO

Số 15

Tạp chí Trí tuệ Nhân tạo

Số 15, phát hành ngày 11/03/2019

Hà Nội

Website: www.tapchiai.com

Email: tapchiai@gmail.com

Facebook: @tapchiai

Twitter: @tapchiai

Mọi đóng góp về chất lượng, nội dung tạp chí xin gửi về địa chỉ email tapchiai@gmail.com hoặc qua các mạng xã hội nêu trên.

Ban Biên Tập xin chân thành cảm ơn!!!

Contributors

Mr. Hà Đỗ

Ms. Lưu Bích Hồng

Mr. Chiến Trương

Mục lục

Góc nhìn chuyên gia

Những cỗ máy thông minh

Dịch từ link, của tác giả Karen Hao, ngày 25/01/2019.

Chúng tôi đã phân tích 16,625 bài báo để tìm hiểu xem AI sẽ đi tới đâu.

Nghiên cứu của chúng tôi về 25 năm nghiên cứu trí tuệ nhân tạo gợi ý rằng kỷ nguyên của Deep Learning có thể đang đi tới hồi kết.

Hầu như tất cả những điều các bạn nghe thấy gần đây về trí tuệ nhân tạo là nhờ vào Deep Learning (DL). Nhóm thuật toán này hoạt động bằng cách sử dụng thống kê để tìm các mảnh ghép có trong dữ liệu và nó đã chứng tỏ sức mạnh vô cùng lớn của mình trong việc bắt chước những kỹ năng của con người như khả năng nghe nhìn. Trong một cấp độ nhỏ hẹp hơn, nó thậm chí có thể mô phỏng khả năng suy luận nữa. Những khả năng này mang lại cho Google sức mạnh tìm kiếm, Facebook new feed và bộ máy gợi ý kết quả của Netflix, cũng như đang chuyển đổi các ngành công nghiệp của chúng ta như chăm sóc y tế và giáo dục.

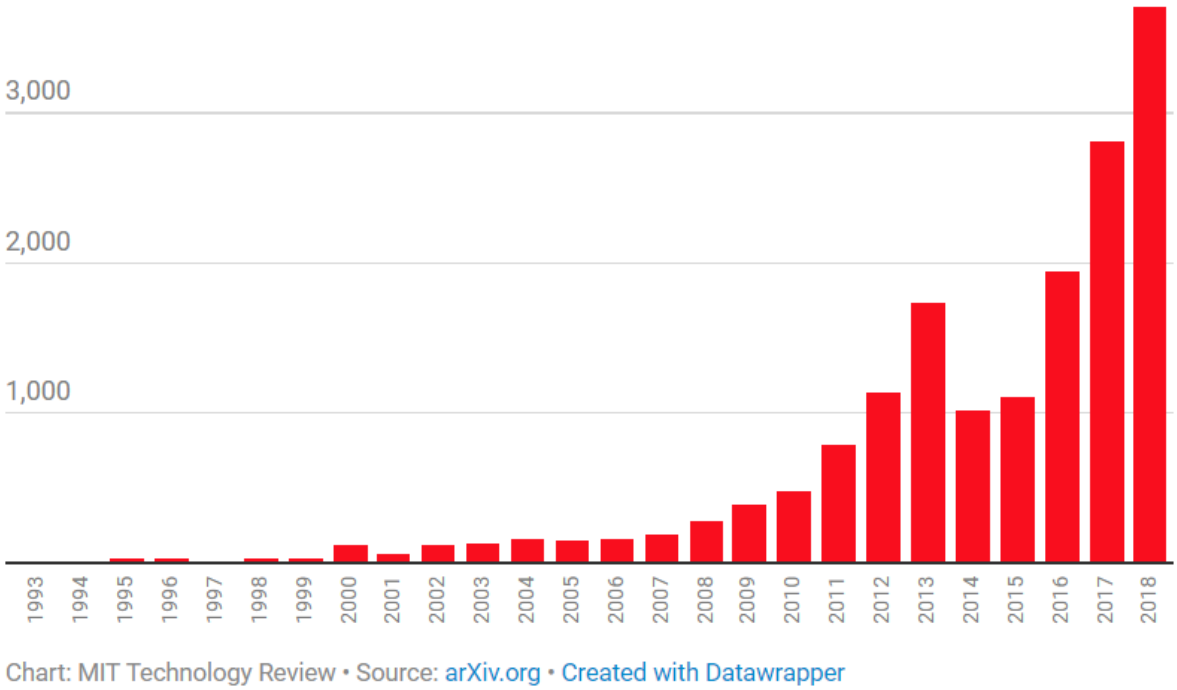
Tuy Deep Learning đã đẩy AI vào sự chú ý ở công chúng, nó chỉ là một cú hích nhỏ trong lịch sử nhân loại trong nhiệm vụ trí tạo trí thông minh của chính chúng ta. DL đi đầu trong nỗ lực đó trong khoảng thời gian chưa tới 10 năm. Nếu như bạn nhìn lại toàn bộ lịch sử trong lĩnh vực này, có thể dễ dàng thấy được rằng nó sớm muộn gì cũng sẽ xuất hiện mà thôi.

« Nếu như ai đó đã viết vào năm 2011 rằng những thứ này sẽ xuất hiện trên các trang nhất của tờ báo và tạp chí trong các năm tới, chúng ta sẽ phản ứng kiểu như : Wow, chúng ta đang thực sự có thứ gì đó mạnh mẽ » Pedro Domingos ; giáo sư khoa học máy tính tại đại học Washington và tác giả của cuốn the Master Algorithm.

Các trào lưu về các kỹ thuật khác nhau dâng lên và hạ xuống một cách đột ngột là một đặc trưng của các nghiên cứu AI trong một thời gian dài, ông nói. Mỗi thập kỷ chúng ta lại chúng kiến những cuộc cạnh tranh nóng bỏng của các ý tưởng khác nhau. Sau đó, thỉnh thoảng thì tất cả mọi người lại hội tụ về cùng 1 kỹ thuật cụ thể nào đó.

Tại MIT Technololy Review, chúng tôi muốn trực quan hóa những điều này, vì vậy chúng tôi đã sử dụng cơ sở dữ liệu lớn nhất về nghiên cứu khoa học arXiv (đọc như là « archieve »). Chúng tôi đã lấy về abstract của 16625 bài báo có trong mục « Trí tuệ nhân tạo » vào ngày 18 tháng 11 năm 2018 và theo dõi các từ được đề cập hàng năm để xem lĩnh vực này đã phát triển như thế nào.

Số bài báo mà chúng tôi đã lấy về từ arxiv



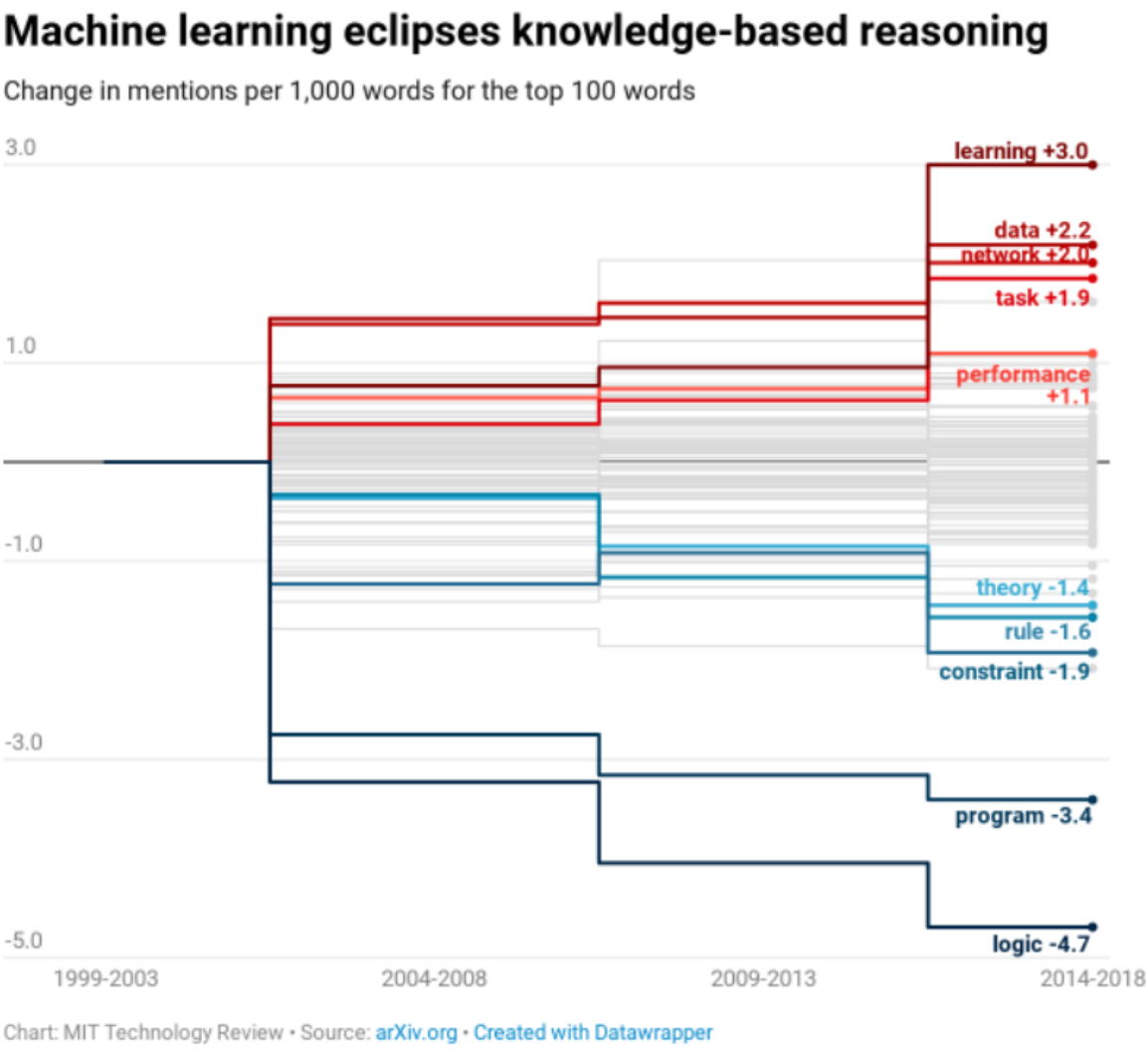
Thông qua phân tích, chúng tôi thấy rõ có 3 xu hướng nghiên cứu chính : **một sự chuyển dịch về Machine Learning từ cuối những năm 1990s và đầu 2000s, một sự dâng lên của neural network vào đầu những năm 2010s và sự tăng trưởng của reinforment learning vào những năm qua.**

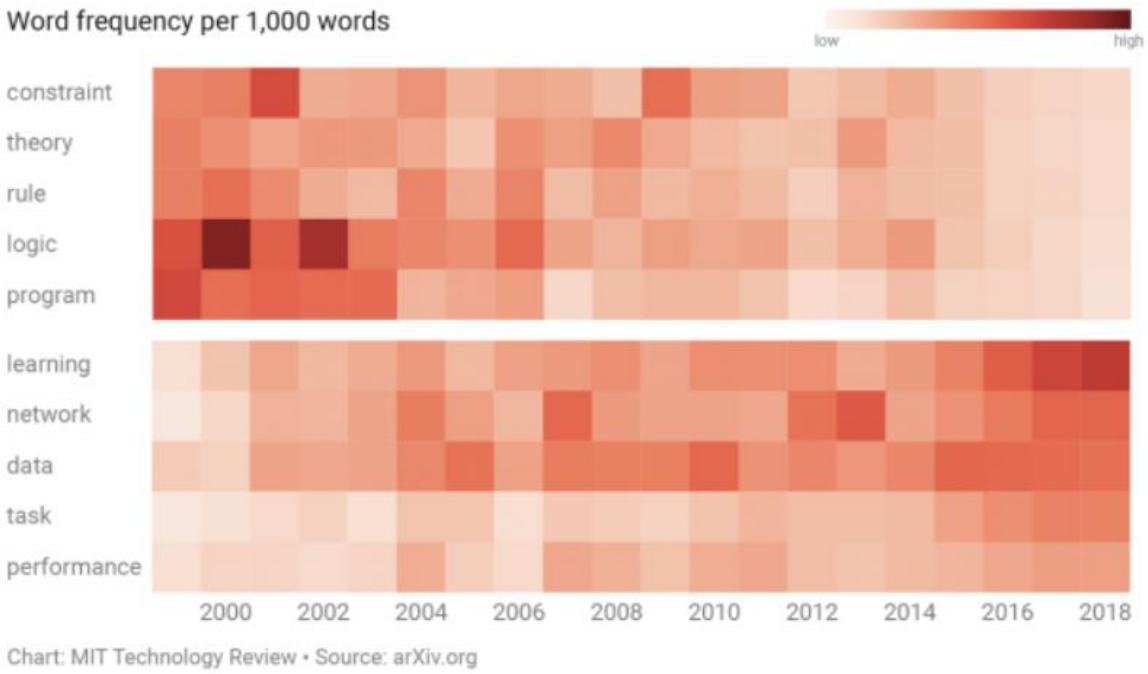
Có một vài chú ý cho phân tích này. Đầu tiên arXiv AI chỉ có lưu trữ tới năm 1993 trong khi thuật ngữ « AI » đã được sử dụng từ những năm 1950s, do vậy cơ sở dữ liệu này chỉ đại diện cho những chương mới nhất của ngành. Thứ hai, những nghiên cứu thêm vào cơ sở dũ liệu này hàng năm chỉ đại diện cho một phần nhỏ những nghiên cứu trong ngành mà thôi. Tuy nhiên arXiv là một nguồn dữ liệu tuyệt vời để thu nhặt một số xu hướng nghiên cứu lớn hơn và để thấy rõ được sự thúc đẩy các ý tưởng khác nhau.

Một thế giới của Machine Learning

Sự chuyển dịch lớn nhất mà chúng tôi tìm thấy là sự chuyển đổi khỏi các hệ thống dựa trên tri thức (knowledge-based system) vào đầu những năm 2000s. Nhưng chương trình máy tính này dựa trên ý tưởng rằng bạn có thể sử dụng các quy tắc để mã hóa toàn bộ kiến thức của con người. Ở vị trí của mình, các nhà nghiên cứu tinh chỉnh các Machine Learning – nhóm kỹ thuật bao gồm cả Deep Learning.

Trong số 100 từ hàng đầu được đề cập tới, những từ liên quan tới các hệ thống tri thức như « logic », « ràng buộc » (constraints) hay « luật lệ » (rule) là những từ có suy giảm lớn nhất. Những từ liên quan tới machine learning như « dữ liệu » (data), « mạng » (network) và « hiệu suất » (performance) thì có sự tăng trưởng lớn nhất.





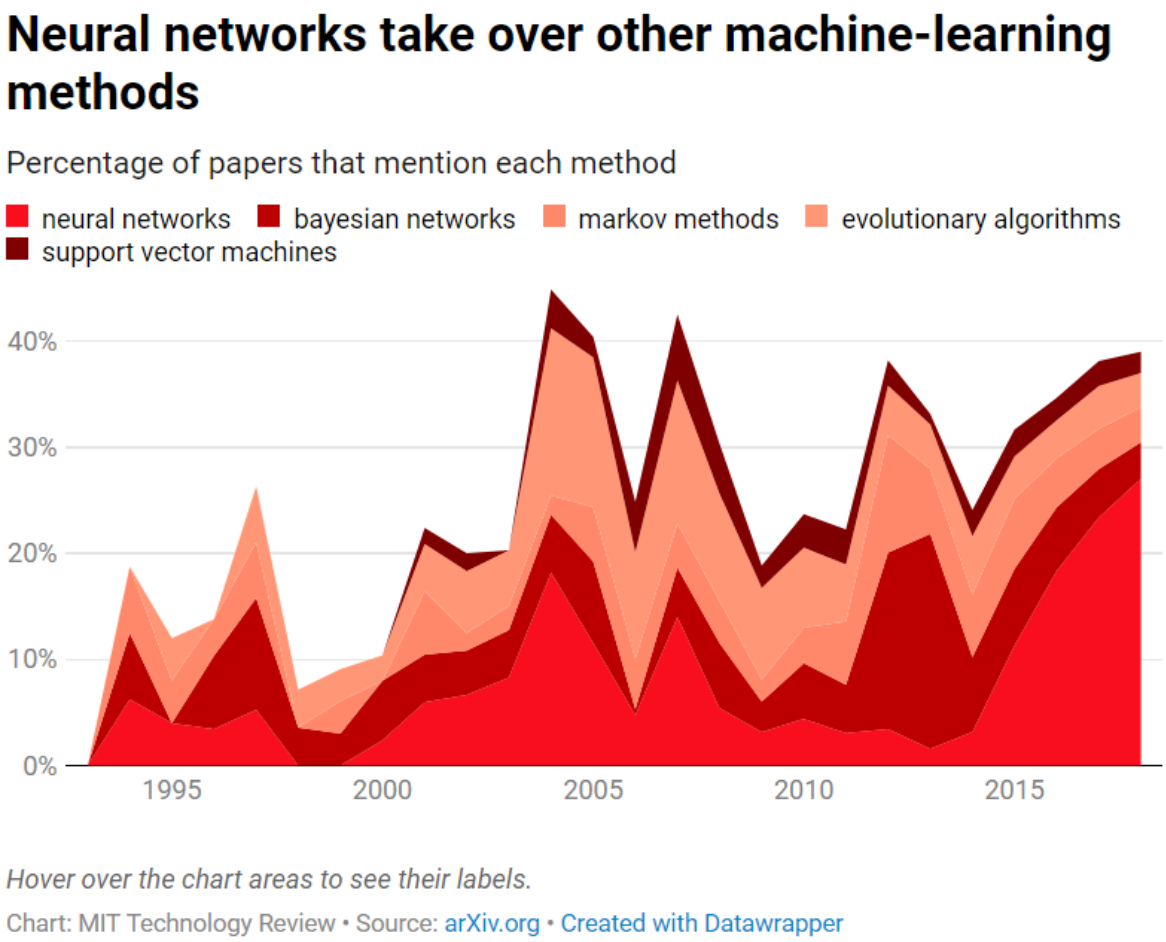
Lý do cho cuộc bể dâu này khá đơn giản. Trong những năm 80s, các hệ thống tri thức đã thu hút được sự theo dõi từ quần chúng nhờ sự phấn khích của những dự án đầy tham vọng, cố gắng tái tạo ý thức chung trong máy móc. Nhưng khi các dự án được bắt đầu, các nhà nghiên cứu gặp một vấn đề lớn : đơn giản là có quá nhiều quy tắc cần được mã hóa để có thể làm được một cái gì hữu ích. Điều này làm tăng chi phí và làm chậm đáng kể các nỗ lực.

Machine Learning trở thành câu trả lời cho vấn đề đó ; thay vì yêu cầu mã hóa thủ công hàng ngàn quy tắc , nó sẽ tự động trích xuất các quy tắc đó từ dữ liệu có trước. Cứ như vậy, lĩnh vực này từ bỏ các hệ thống tri thức và chuyển sang tinh chỉnh Machine Learning

Sự bùng nổ của Neural Networks

Trong thế giới của Machine Learning, việc chuyển sang Deep Learning không được bắt đầu ngay lập tức. Thay vào đó, như trong phân tích của chúng tôi; các nhà nghiên cứu đã thử nghiệm nhiều phương pháp khác nhau ngoài mạng Neural Net, bộ máy cốt lõi của Deep Learning. Các kỹ thuật phổ biến bao gồm các mạng Bayesian; Support Vector Machine hay Evolutionary Algorithm (thuật toán tiến hóa), tất cả những kỹ thuật đó thử nghiệm những cách tiếp cận khác nhau trong tìm kiếm thông tin trong dữ liệu.

Và chúng ta có thể thấy là Neural Net vượt qua các kỹ thuật Machine Learning khác



Vào những năm 1990s và 2000s, có một sự cạnh tranh khá ổn định giữa những kỹ thuật này, sau đó năm 2012, một bước đột phá quan trọng đã dẫn tới một cuộc bể dâu khác. Trong cuộc thi ImageNet hàng năm, cuộc thi nhằm thúc đẩy sự nghiên cứu trong Computer Vision, Geoffrey Hilton và các đồng nghiệp của mình tại đại học Torronto đã đạt được độ chính xác cao nhất trong nhận dạng ảnh với tỉ lệ đáng kinh ngạc: hơn 10% vượt qua vị trí thứ hai.

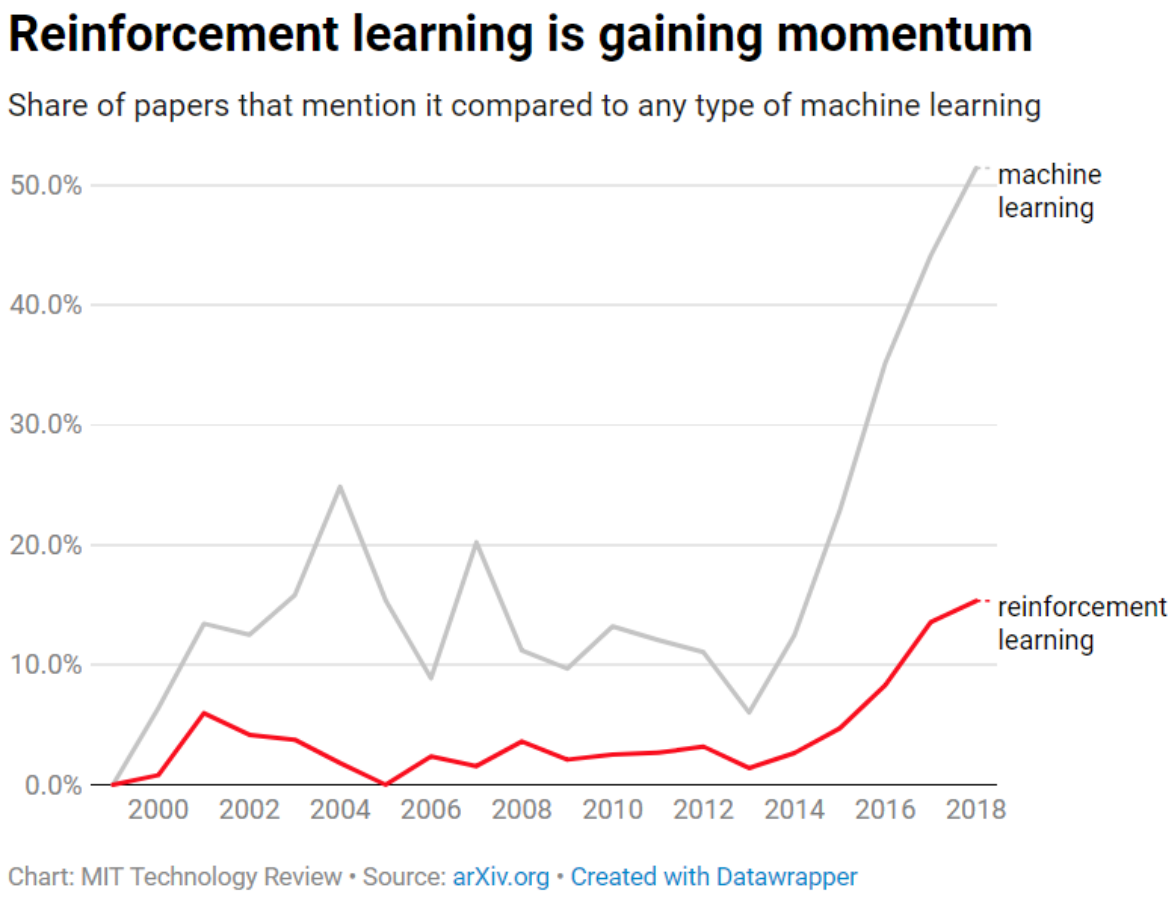
Kỹ thuật mà ông ta sử dụng, Deep Learning, đã tạo ra một làn sóng nghiên cứu mới – bắt đầu từ cộng đồng Computer Vision và sau đó mở rộng hơn nữa. Càng ngày càng có nhiều các nhà nghiên cứu đạt được các kết quả ấn tượng; sự phổ biến của nó cùng mạng neural net đã bùng nổ.

Sự nổi lên của Reinforcement Learning

Trong vài năm kể từ sự nổi lên của Deep Learning; phân tích của chúng tôi cho thấy; làn sóng thứ ba và cuối cùng của sự dịch chuyển đang diễn ra trong nghiên cứu AI.

Có ba kỹ thuật trong machine learning: supervised, unsupervised và reinforcement learning. Supervised learning liên quan tới việc cung cấp cho máy các dữ liệu gắn nhãn là kỹ thuật được sử dụng nhiều nhất trong ứng dụng thực tiễn hiện giờ. Trong một vài năm gần đây, reinforcement

learning, kỹ thuật bắt chước lại khả năng huấn luyện động vật thông qua trao thưởng và trừng phạt đã có một sự gia tăng nhanh chóng trong các đề cập trong abstract của các bài báo.



Ý tưởng thì không mới, nhưng qua hàng thập kỷ thì nó không hiệu quả. "Cộng đồng làm supervised learning thì giễu cợt những người làm reinforcement learning", Domingos nói. Nhưng cũng như với Deep Learning; một khoảnh khắc lịch sử quan trọng bất giờ xuất hiện.

Đó vào vào tháng 10 năm 2015 khi mà DeepMind AlphaGo, được huấn luận với reinforcement learning đã đánh bại nhà vô địch thế giới trong trò chơi cổ xưa Go. Và nó có hiệu quả ngay lập tức với cộng đồng các nhà nghiên cứu.

Thập kỷ tiếp theo

Nghiên cứu của chúng tôi chỉ cung cấp những lát cắt ảnh lịch sử về sự cạnh tranh giữa các ý tưởng đặc trưng cho sự nghiên cứu AI. Nhưng nó cũng minh họa sự thay đổi của nhiệm vụ nhân đôi trí tuệ. "Điều quan trọng là nhận ra rằng không ai biết làm thế nào để giải quyết bài toán này". Domingos nói.

Nhiều kỹ thuật đã được sử dụng trong vòng 25 năm qua được bắt nguồn từ cùng một thời điểm: vào những năm 1950s và đã rơi ra ngoài cũng như quay trở lại với những thách thức và thành công của mỗi thập kỷ. Ví dụ như Neural Net, nó đã đạt đỉnh vào năm 60s và ngắn ngủi kéo trong những năm 80s và gần như chết sau đó cho tới khi lấy lại được sự phổ biến của mình thông qua Deep Learning.

Nói cách khác thì mỗi thập kỷ, cơ bản đã chứng kiến một sự thống trị của các kỹ thuật khác nhau: neural net vào những năm 50s-60s, các cách tiếp cận mang tính biểu tượng vào những năm 70s, hệ thống trí thức vào những năm 80s, mạng bayes vào những năm 90s, Support Vector Machine vào những năm 2000s và sự trở lại của Neural Net vào những năm 2010s.

Những năm 2020s có lẽ sẽ không khác gì cả, Domingos nói, nghĩa là thời đại của Deep Learning sẽ sớm chấm dứt, nhưng một cách đặc trưng, cộng đồng nghiên cứu đã đang cạnh tranh nhau bằng những ý tưởng sắp tới – đó có thể là một kỹ thuật nào đó lấy lại được sự ưu ái hay tạo ra một triều đại hoàn toàn mới.

“Nếu bạn trả lời câu hỏi đó”, Domingos nói; “tôi muốn bảo hộ cho câu trả lời của bạn”.

GPU(s) nào dùng cho Deep Learning: Kinh nghiệm và lời khuyên của tôi cho việc sử dụng GPU(s) trong Deep Learning[1]

Dịch từ link, của tác giả Tim Dettmers, ngày 05/11/2018.

Deep learning (DL) là một lĩnh vực với yêu cầu tính toán mạnh mẽ và việc chọn GPU của bạn sẽ xác định một cách cơ bản kinh nghiệm về Deep Learning. Không có GPU, chúng ta có thể sẽ phải chờ đợi hàng tháng trời để một thí nghiệm có thể hoàn thành hoặc chạy một thí nghiệm thêm nhiều ngày nữa chỉ để quan sát sự thay đổi của mô hình và các tham số được chọn tương ứng.

Với một GPU tốt, tin cậy được, chúng ta có thể dễ dàng duyệt qua các tổ hợp thiết kế và tham số của deep networks và thời gian chạy thí nghiệm chỉ tính bằng ngày thay vì hàng tháng, thậm chí từng phút, thay vì hàng giờ hay hàng ngày. Từ đó đó có thể thấy là lựa chọn đúng GPU là một điều tối quan trọng. Làm thế nào chúng ta có thể xác định GPU nào là phù hợp với mình. Blog post này sẽ cố gắng làm sáng tỏ điều đó và sẽ cho bạn một chút lời khuyên mà tôi nghĩ có thể sẽ giúp bạn.

Có một cái GPU nhanh là một khía cạnh rất quan trọng khi chúng ta bắt đầu học DL vì nó cho phép chúng ta có được các kinh nghiệm thực tế mà từ đó chúng ta có thể dùng để áp dụng vào các thử thách mới. Nếu không có sự phản hồi nhanh chóng từ thí nghiệm, sẽ mất rất nhiều thời gian để ta có thể học từ lỗi lầm của mình và điều này làm nản chí cũng bực bội để tiếp tục với DL. Với nhiều GPU, tôi đã nhanh chóng học được cách áp dụng DL lên rất nhiều cuộc thi Kaggle và đã kiếm cho mình một giải nhì trong cuộc thi Partly Sunny with a chance of Hashtag (nhiệm vụ của cuộc thi này là dự báo chỉ số thời tiết cho một tweet). Trong cuộc thi này, tôi sử dụng mạng deep neural network 2 lớp với ReLu và dropout cho regularization và mạng này chỉ fit vừa đủ với 6GB GPU. Những chiếc GPUs GTX Titan chính là nhân tố chính để tôi có được vị trí thứ 2 trong cuộc thi này.

Tôi có nên có nhiều GPUs không

Với sự hân hoan về những thứ mà DL có thể làm với GPUs, tôi đã đắm mình vào thế giới của multi-GPU bằng cách tạo ra những cluster GPU nhỏ với InfiniBand 40Gbit/s interconnect. Tôi đã rất hồi hộp để xem liệu có thể thu được kết quả tốt hơn với nhiều GPU hay không.

Tôi nhanh chóng tìm ra rằng không chỉ rất khó để chạy song song (parallelization) các mạng neural net trên GPUs một cách hiệu quả mà thậm chí việc tăng tốc có hiệu quả rất ít với các mạng neural

[1] lược dịch – chúng tôi cắt bỏ một số đoạn tác giả diễn đạt rườm rà. Những phần bôi đậm hay nghiêng là những kết luận mà chúng tôi cho rằng là quan trọng. Ngoài ra chúng tôi cũng có thể thêm vào một số bình luận với một số tin tức cập nhật có được sau bài blog này hoặc kinh nghiệm riêng của chúng tôi, phần này sẽ được chú thích rõ ràng. Ngoài ra, bài blog này cũng thường xuyên được cập nhật, vì vậy nếu bạn truy cập trong tương lai có thể tác giả đã update các phiên bản GPU mới. Bạn có thể xem các mốc cập nhật ở cuối của bài viết này.
Bài dịch của chúng tôi được thực hiện vào đầu năm 2019, sau sự xuất hiện của các Card Nvidia GTX. Chúng tôi cũng đã theo dõi bài viết từ lâu, vào khoảng cuối năm 2016 và thấy các nhận xét của tác giả qua từng thời kỳ khá chính xác.
Một số vấn đề chi tiết về kỹ thuật được bàn luận trong phần comment giữa tác giả và các người đọc khác cũng khá thú vị, chúng tôi mời bạn đọc tham khảo trong link gốc.

net nhiều lớp. Những mạng neural net nhỏ có thể được parallized một cách hiệu quả hơn bằng cách sử dụng data parallelism nhưng các mạng lớn như cái mà tôi đã dùng trong cuộc khi Kaggle kể trên gần như không có sự cải thiện tốc độ nào cả.

Tôi đã phân tích sâu về sự parallelization (song song hóa), phát triển một kỹ thuật để tăng tốc GPUs clusters từ 23x tới 50x cho một hệ thống gồm 96 GPUs và công bố kết quả nghiên cứu của mình ở ICLR 2016. Trong phân tích này của mình, tôi cũng tìm ra rằng convolution và recurrent networks thì dễ hơn để parallelize, đặc biệt là nếu chúng ta chỉ sử dụng một computer hay 4 GPUs. Do vậy ngay cả khi các công cụ hiện đại chưa được tối ưu hóa cao cho parallelism, bạn vẫn có thể có được một sự tăng tốc đáng kể.

Hình 1: đây là thiết lập trong máy tính của tôi. Các bạn có thể thấy 3 GPUs và 1 InfiniBan Card. Liệu đây có phải là một thiết lập tốt để làm DL?

Trải nghiệm của người dùng sử dụng các kỹ thuật parallization trong các framework phổ biến hiện tại khá tốt so với 3 năm trước. Các thuật toán của họ khá giản đơn và không thể scale lên GPU Cluster được nhưng họ có hiệu suất khá tốt cho tới 4 GPUs. **Với mạng convolution**, ***bạn có thể có được sự tăng tốc lên tới 1.9x/2.8x/3.5x lần lượt cho 2/3/4 GPUs***; **với mạng recurrent**, sequence length là một trong các tham số quan trọng nhất và cho các vấn đề NLP[2] phổ biến, chúng ta có thể mong đợi **một sự tăng tốc tương tự hoặc kém hơn một chút so với convolutional network.** Một mạng kết nối hoàn toàn (fully connect network) thường có hiệu suất kém hơn với data parallelism và các thuật toàn nâng cao hơn là cần thiết để tăng tốc các phần của mạng này.

[2] Natural Language Processing: Xử lý ngôn ngữ tự nhiên

Do vậy ngày nay, việc sử dụng multi-GPUs có thể làm cho việc training thuận tiện hơn nhiều nhờ sự tăng tốc và nếu bạn có tiền cho chuyện này, multi-GPUs là một sự lựa chọn.

Sử dụng Multi-GPU mà không dùng Parallelism

Một lợi thế khác của việc sử dụng multi-GPU là ngay cả khi bạn không sử dụng các thuật toán song song, bạn có thể chạy nhiều thuật toán hay nhiều thí nghiệm một cách tách rời nhau trên các GPU này. Bạn sẽ **không có được sự tăng tốc, tuy nhiên bạn có nhiều thông tin hơn về kết quả bằng cách sử dụng nhiều thuật toán hay nhiều tham số cùng một lúc**. Điều này đặc biện hữu ích nếu như mục đích của bạn là có nhiều kinh nghiệm với DL nhanh nhất có thể, và nó cũng rất hữu dụng cho các nhà nghiên cứu muốn thử nghiệm vào phiên bản của một thuật toán mới cùng một lúc.

Điều này rất quan trọng nếu bạn muốn học DL. Thời gian càng ngắn để thực hiện một nhiệm vụ và nhận được feedback cho nhiệm vụ này thì việc não bộ có thể tổng hợp được các mảnh thông tin liên quan vào thành một bức tranh phù hợp càng tốt hơn. Nếu bạn train 2 convolutional nets trên 2 GPUS riêng biệt với các datasets nhỏ, bạn sẽ nhanh chóng cảm nhận được điều gì là quan trọng để có được kết quả tốt; bạn sẽ sẵn sàng hơn để phát hiện ra các thông tin trong error của xác thực chéo (cross-validation error) và sẽ hiểu nó đúng đắn hơn. Bạn cũng sẽ phát hiện được các thông tin nào đang đưa ra cho bạn các gợi ý về các tham số hay các lớp cần phải thêm vào hay xóa đi hay điều chỉnh.

Cá nhân tôi cho rằng multi-GPU theo cách này thì hữu ích hơn khi mà chúng ta có thể nhanh chóng tìm ra một tổ hợp thuật toán tốt. Khi mà chúng ta tìm thấy các vùng tham số hay kiến trúc mạng tốt, chúng ta sau đó có thể sử dụng parallelism với multi-GPU để train mạng cuối cùng.

Do vậy, một cách tổng quan, ta có thể thấy một GPU có thể đủ cho hầu hết các nhiệm vụ nhưng muti-GPU đang trở nên ngày một quan trọng hơn để tăng tốc mô hình DL của bạn. **Nhiều GPUs giá rẻ sẽ là tuyệt vời nếu bạn muốn học DL nhanh hơn.** *Cá nhân tôi có nhiều GPUs nhỏ hơn là một GPU lớn*, ngay cả trong các thí nghiệm nghiên cứu của tôi.

NVIDIA vs. AMD vs. Intel vs. Google vs. Amazon

NVIDIA: người dẫn đầu

Thư viện tiêu chuẩn của NVIDIA khiến việc tạo ra một thư viện DL đầu tiên trong CUDA rất dễ dàng trong khi không có một thư viện chuẩn mạnh mẽ như vậy với AMD OpenCL. Lợi thế sớm này kết hợp với sự hỗ trợ cộng đồng mạnh mẽ từ NVDIA làm cộng đồng CUDA tăng trưởng mạnh mẽ. Điều này có nghĩa là nếu bạn sử dụng NVIDIA GPUs, bạn sẽ dễ dàng có được sự hỗ trợ và lời khuyên nếu như có điều gì sai xảy ra cho bạn khi bạn lập trình CUDA, và **bạn sẽ thấy rằng hầu hết các thư viện DL đều hỗ trợ tốt nhất cho NVIDIA GPUS**. Đây là một điểm rất mạnh cho NVDIA GPUs.

Mặt khác, NVDIA bây giờ có chính sách là việc sử dụng CUDA trong "trung tâm dữ liệu" (data center) chỉ cho phép với Tesla GPUs mà không phải GTX hay RTX card. Không có sự rõ ràng trong định nghĩa thế nào là "trung tâm dữ liệu" nhưng điều này thường có nghĩa là các tổ chức hay

cơ sở nghiên cứu, trường đại học thường bị ép phải mua những chiếc Tesla GPU đắt đỏ và không hiệu quả về chi phí chỉ vì sợ hãi vấn đề pháp lý này. Tuy nhiên **Tesla Card không có bất kỳ lợi ích thực nào so với GTX hay RTX mà nó còn đắt gấp 10 lần.**

NVDIA chỉ có thể làm điều này khi mà không có bất kỳ trở ngại lớn nào, điều đó cho thấy sức mạnh độc quyền của họ.

AMD: mạnh mẽ nhưng thiếu sự hỗ trợ

Thư viện HIP via ROCm thống nhất NVDIA và AMD GPU dưới một ngôn ngữ lập trình chung mà biên dịch vào ngôn ngữ GPU tương ứng trước khi nó được biên dịch thành GPU assembly. Nếu chúng ta có thể có tất cả GPU code trong HIP, điều này sẽ là một dấu mốc lớn, tuy nhiên điều này cũng khá là khó khăn vì rất khó để chuyển Tensorflow và PyTorch code bases. **Tensorflow có một số hỗ trợ cho AMD GPUs và tất cả mạng của họ đều có thể chạy trên AMD GPUs**, nhưng thiếu vắng sự phát triển một mạng mới với các chi tiết riêng. Điều này có thể khiến bạn không thể viết ra những gì bạn muốn. Cộng đồng ROCm thì chưa đủ lớn do vậy rõ ràng là chưa thể xử lý vấn đề này một cách nhanh chóng hơn. Ngoài ra thì có vẻ như không có nhiều tiền để phát triển và hỗ trợ cho DL từ phía AMD và điều này khiến cho đà phát triển bị chậm lại.

Tuy nhiên AMD GPUs cho thấy hiệu suất mạnh mẽ so với NVIDIA GPUs và các GPU AMD Vega20 sẽ là một nhà máy tính toán khi nó có các unit tính toán giống như Tensor-Core.

(Update – ND: tại CES 2019 AMD ra mắt Radeon VII card đồ họa trên tiến trình 7nm đầu tiên trên thế giới. trong khi card GTX mới của NVIDIA vẫn dựa trên tiến trình 12nm và phải đợi tới 2020 mới có tiến trình 7nm[3] . Trong khi CEO của NVIDIA lớn tiếng chê sức mạnh của dòng card mới này[4] của AMD thì điểm benchmark lại cho thấy sự hiệu quả của Radeon VII, đặc biệt là khi tính thêm khía cạnh chi phí[5].)

Xét một cách tổng thể thì *tôi vẫn không thể có một khuyến khích rõ ràng nào cho GPU của AMD cho những người dùng bình thường* mà chỉ muốn GPUs của họ chạy một cách mượt mà. Những người dùng kinh nghiệm sẽ có ít vấn đề hơn và với sự hỗ trợ của AMD GPUs và các nhà phát triển ROCm/HIP, họ đóng góp vào trận chiến chống lại sự độc quyền từ NVDIA và điều này sẽ có lợi hơn cho tất cả trong dài hạn. *Nếu bạn là một nhà phát triển GPU và muốn đóng góp cho cộng động GPU computing, AMD GPU sẽ là cách tốt nhất để có ảnh hưởng tốt trong dài hạn. Còn với tất cả những người khác, NVIDIA GPUs là một lựa chọn an toàn hơn.*

Intel: Cố gắng đáng kể

[3] http://gamek.vn/ces-2019-amd-ra-mat-radeon-vii-card-do-hoa-7nm-dau-tien-tren-the-gioi-20190111170315247.chn
[4] https://tinhte.vn/threads/ceo-nvidia-noi-amd-radeon-vii-khong-co-gi-thu-vi-chi-manh-ngang-rtx-2080.2904450/

[5] http://gamek.vn/diem-benchmark-cua-vga-moi-amd-radeon-vii-7nm-manh-me-va-rat-hop-tui-tien-20190131001939812.chn

Kinh nghiệm cá nhân của tôi với Intel Xeon Phi rất là đáng thất vọng và tôi không cho rằng họ (Intel) là đối thủ thực sự của NVIDIA hay AMD và vì vậy nên tôi nói ngắn thôi: *nếu bạn muốn sử dụng Xeon Phi thì hay chú ý rằng bạn sẽ có thể gặp những sự hỗ trợ nghèo nàn, các vấn đề tính toán khiến khu vực code chậm hơn CPUs và rất khó để viết các code tối ưu, không có hỗ trợ cho các tính năng của C++ 11, và một vài GPU design parteern quan trọng không hỗ trợ bởi trình phiên dịch, sự tương thích nghèo nàn với các thư viện khác mà dựa trên BLAS routines (như NumPy hay SciPy) and có thể nhiều sự thất vọng khác mà tôi cũng chưa gặp phải.*

Tôi đã từng rất mong chờ Nervana neural network (NNP) của Intel vì cấu hình của nó cực kỳ mạnh mẽ trong tay của một nhà phát triển GPU và nó sẽ cho phép phát triển các thuật toán mới nhất mà có lẽ sẽ định nghĩa lại cách neural network được sử dụng. Tuy nhiên nó đã bị trì hoãn vô thời hạn và có những tin đồn rằng có phần lớn các phần đã được phát triển đã bị bỏ đi. NNP có kế hoạch được ra mắt vào Q3/Q4 2019. *Nếu bạn muốn chờ đợi tới lúc đó thì hay nhớ rằng phần cứng tốt không phải là tất cả (như chúng ta đã thấy với AMD và Intel Xeon Phi). Có lẽ sẽ phải tới 2020 để NNP có thể đc phát triển chín hơn để sử dụng.*

Google: Rẻ hơn nếu sử dụng theo nhu cầu?

TPU[6] của Google được phát triển thành một sản phẩm đám mây khá tốt, nó khá hiệu quả về mặt chi phí. Cách dễ nhất để hiểu TPU là nhìn nó giống như multi-GPU được đóng gói lại cùng nhau. Nếu chúng ta căn cứ vào các thang đo hiệu suất của Tensor-Core-enabled V100 vs TPUv2 thì chúng ta thấy rằng hai hệ thống gần như có hiệu suất như nhau cho ResNet50. *Tuy nhiên TPU của Google thì có chi phí rẻ hơn.*

Ồ, vậy thì TPU là một giải pháp đám mây có chi phí hiệu quả ư? Câu trả lời là **Có và Không.** Trên giấy tờ và với mức sử dụng thông thường thì nó hiệu quả hơn về mặt chi phí. Tuy nhiên nếu như bạn sử dụng kinh nghiệm thực hành và hướng dẫn như fastai team và fastai library thì *bạn có thể có được sự hội tụ thuật toán nhanh hơn với chi phí rẻ hơn – ít nhất là cho convolutional network cho vấn đề nhận dạng vật thể.*

Với cùng một phần mềm, TPU có thể có sự hiệu quả chi phí hơn nữa, nhưng nó cũng có các vấn đề sau đây: (1) TPU không có sẵn cho thư viện fastai, nó là PyTorch, (2) TPU algorithm dự vào nhóm phát triển nội bộ của Google, (3) không có sự thống nhất của thư viện ở mức độ cao, điều mà tạo nên một tiêu chuẩn tốt cho TensorFlow.

3 điểm nói trên đánh vào sự sử dụng TPU khi mà nó yêu cầu các phần mềm khác nhau phải được cập nhật để theo kịp với sự phát triển các họ thuật toán DL mới. Tôi chắc chắn rằng những việc lặt vặt này được Google thực hiện nhưng không rõ ràng là sự support đó tốt đến thế nào với các mô hình khác nhau. Repo chính chỉ có một mô hình đơn lẻ cho NLP và tất cả những mô hình còn lại là Computer Vision. Tất cả các mô hình sử dụng convolution và không có cái nào trong chúng sử dụng

[6] Tensor Processing Unit

recurrent neural net cả. Có một báo cáo cũ từ tận tháng 2 rằng TPUv2 không hội tụ khi chạy LSTM[7] và tôi không thể tìm được một nguồn nào khẳng định rằng vấn đề này đã được sửa chữa hay chưa. Mặt khác, một cột mốc lớn trong NLP là BERT[8] là một kiến trúc lớn biến đổi 2 chiều, nó có thể được tuned để đạt được hiệu suất cao nhất từ trước tới giờ (state-of-the-art performance) cho rất nhiều vấn đề NLP. TPUs là thành tố quan trọng để train BERT trên rất nhiều dữ liệu. Tổng cộng cần tới 256 giờ TPU để train một mô hình dựa trên BERT. Làm sao có thể so sánh điều này với GPUs? *Tôi đã viết một phân tích chi tiết về điều này và thấy rằng GPU RTX mới ra cũng rất quan trọng cho hiệu suất của BERT và chúng ta có thể hi vọng thời gian chạy là 400 giờ GPU. Điều này cho thấy TPU là hoạt động khá tốt cho nhiệm vụ này và nó có lợi thế lớn khi so với GPU.*

Để kết luận thì hiện tại TPUs dường như là tốt nhất cho convolutional network hoặc large transformer và nó cần phải được bổ sung với các tài nguyên tính toán khác thay vì trở thành một nguồn DL chính.

Amazon: khá tin cậy nhưng đắt đỏ

Rất nhiều GPUs mới đã được thêm vào AWS[9] kể từ lần cuối update bài viết này. Tuy nhiên giá cả thì vẫn hơi cao. AWS GPU instance có thể là một giải pháp hữu ích nếu như các tính toán bổ sung bất chợt xuất hiện, như khi tất cả GPU đều đang được sử dụng như thường thấy trước các hạn nộp nghiên cứu báo cáo.

Tuy nghiên, nếu để đảm bảo về chi phí thì ta cần phải chắc chắn rằng ta đang chạy một vài mạng thôi và chúng ta biết trước vài tham số tốt gần như tối ưu đã được chọn. Nếu không thì sẽ rất là đắt đỏ và GPU chuyên dụng thì phù hợp hơn. Ngay cả một AWS GPU nhanh cỡ GTX 1070 trở lên sẽ có thể cung cấp các hiệu suất tính toán tốt trong một hoặc hai năm mà không tốn quá nhiều chi phí.

Kết luận thì AWS GPU instance có thể hữu dụng nếu ta sử dụng nó khôn ngoan và thận trọng để tiếp kiệm chi phí. Thảo luận hơn nữa về điện toán đám mây thì mời các bạn theo dõi thêm ở dưới.

Điều gì khiến cho một GPU chạy nhanh hơn một GPU khác?

Câu hỏi đầu tiên của các bạn có lẽ là cái gì quan trọng hơn cho hiệu suất tính toán của GPU cho DL: số nhân CUDA? Xung nhịp hay kích cỡ RAM?

Một lời khuyên đơn giản sẽ ngày trước có thể là: "hãy chú ý tới băng thông bộ nhớ". Tôi không còn khuyên bạn như thế nữa bởi vì phần cứng của GPU và phần mềm đã phát triển nhiều năm rồi và băng thông không còn là proxy tốt cho hiệu suất nữa. Việc giới thiệu Tensor Core trong GPU ở mức

[7] LSTM: ***Long Short Term Memory*** networks
[8] BERT: Bidirectional Encoder Representations from Transformers
[9] AWS: Amazon Web Service

độ người tiêu dùng bình thường làm thức tạp vấn đề thêm. Bây giờ thì chỉ số tốt cho hiệu suất GPU là một sự kết hợp giữa băng thông, FLOPS[10] và Tensor Cores.

Một điều giúp tăng cường sự hiểu biết của chúng ta để lựa chọn sáng suốt là nên tìm hiểu một chút về những phần nào khiến cho GPUs chạy nhanh cho các phép tính trên tensor: nhân ma trận và tích chập (matrix multiplication and convolution).

Một cách đơn giản và hiệu quả để nghĩ về phép nhân ma trận là nghĩ về giới hạn băng thông. Băng thông bộ nhớ là rất quan trọng nếu bạn muốn dùng LSTM hay các mạng recurrent khác mà sử dụng cực nhiều nhân ma trận. Tương tự như vậy thì convolution bị giới hạn bởi tốc độ tính toán, vì vậy mà TFLOPs của GPU là chỉ số tốt nhất cho hiệu suất của ResNet hay các kiến trúc convolution.

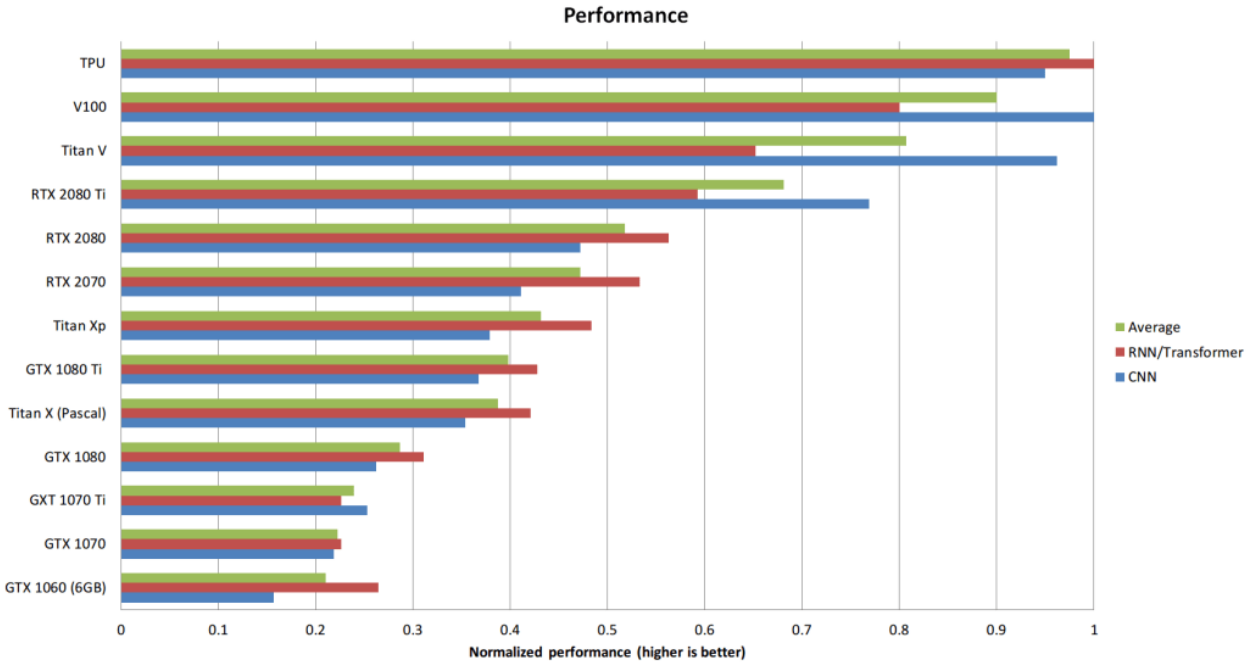
Tensore Core thay đổi phương trình trên một chút. Nó là một kiến trúc tính toán đặc biệt để tăng tốc tính toán chứ không phải tăng băng thông và từ đó lợi ích của nó có thể là tăng tốc khoảng từ 30% tới 100% cho mạng convolutional net.

Khi mà Tensore Core chỉ làm cho tính toán nhanh hơn thì nó cũng cái thiện tính toán sử dụng 16-bit numbers. Đây là một lợi thế lớn cho nhân ma trận vì nếu chỉ sử dụng 16-bit number thay vì 32-bit number thì chúng ta có được gấp đôi số number trong ma trận với cùng số băng thông bộ nhớ. Tôi đã viết chỉ tiết về cách chuyển 32-bit về 16-bit ảnh hưởng như nào tới hiệu suất nhân ma trận. Nói chung thì chúng ta có thể hi vọng một sự tăng tốc từ 100-300% khi sử dụng sự chuyển dịch này và từ 20% tới 60% cho LSTM sử dụng Tensor Cores.

Có một sự tăng trưởng lớn về hiệu suất và 16-bit nên trở thành tiêu chuẩn với các RTX card – đừng bao giờ sử dụng 32-bit nhé. Nếu bạn gặp vấn đề với 16-bit thì bạn nên sử dụng loss scaling: (1) nhân loss của bạn với một số lớn, (2) tính gradient, (3) chia lại cho số lớn đó, và (4) update hệ số. Thông thường thì train 16-bit sẽ ổn thôi, nhưng nếu có vấn đề thì hãy dùng kỹ thuật trên nhé.

Rồi, tóm lại thì kinh nghiệm tốt nhất (best rule of thumb) là: nếu dùng recurrent thì xem băng thông, nếu dùng Convolution thì xem FLOPS và mua Tensor Cores nếu có thể chi trả được (đừng có mua Tesla nhé, trừ khi bắt buộc).

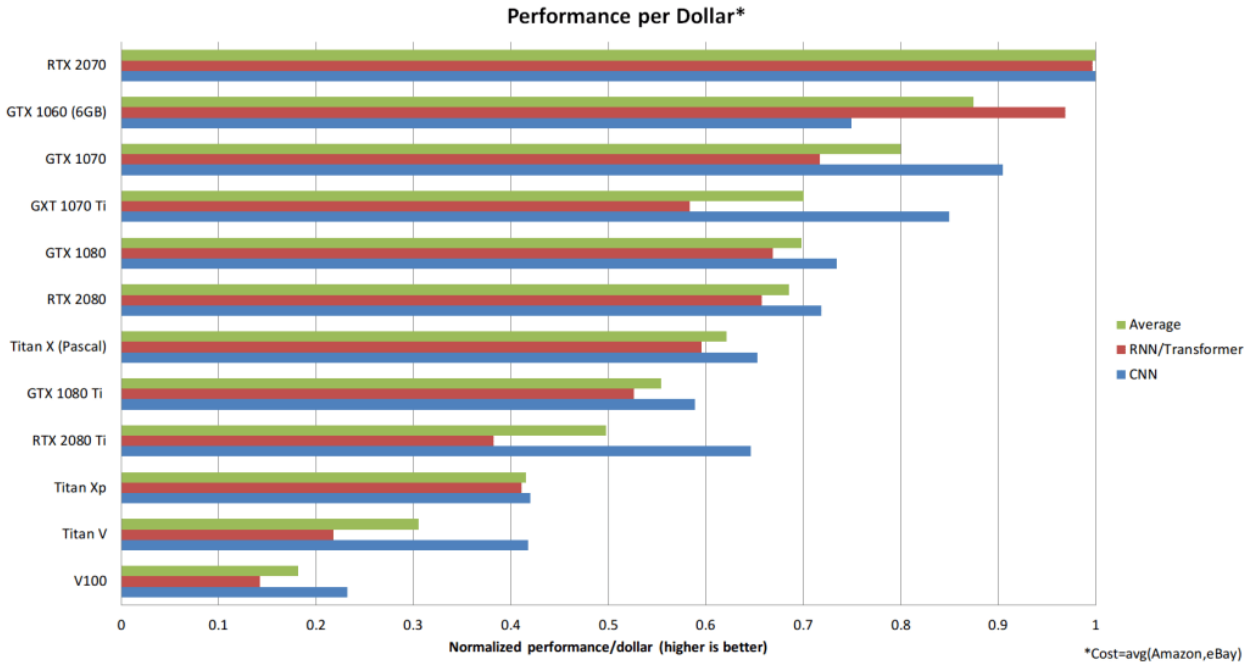
[10] TFLOPs: là viết tắt của Teraflop là cách đo lường sức mạnh máy tính dựa trên khả năng toán học của nó thay vì xung nhịp. Một TFLOPS đề cập tới khả năng tính toán một nghìn tỉ phép tính dấu phẩy động một giây. Một cái gì đó với 6 TFLOPs nghĩa là nó tính được tới trung bình 6 nghìn tỉ phép tính (dấu phẩy động) mỗi giây.



Hình 2: HIệu suất thô của GPU và TPU. Cao hơn tốt hơn. RTX2080 TI mới mạnh gấp đôi GTX 1080: 0.77 vs. 0.4

Phân tích hiệu quả chi phí

Hiệu quả chi phí của GPU có lẽ là tiêu chí quan trọng nhất để chọn GPU. Tôi đã phân tích chi phí kết hợp với băng thông, TFLOP và Tensor Cores. Giá bán được sử dụng là giá ebay và amazon với tỉ lệ 50:50. Sau đó tôi xem xét các chỉ số hiệu suất cho LSTM và CNN với có và không sử dụng Tensor Cores. Cuối cùng các chỉ số được tính là các hiệu suất trung bình tính thêm các yếu tố trên: Dưới đây là kết quả



Hình 3: Hiệu suất- chi phí chuẩn hóa cho convolution CNN, recurrent RNN và Transformer. Cao hơn tốt hơn. Một RTX 2070 có hiệu suất chi phí tốt gấp 5 lần Tesla V100

Từ các dữ liệu này[11], ta có thể thấy rằng RTX 2070 hiệu quả hơn hẳn so với RTX 2080 và hay RTX 2080 Ti. Vì sao lại như vậy ? **Khả năng tính toán 16-bit với Tensor Cores có giá trị hơn nhiều là chỉ sử dụng card lớn hơn với nhiều Tensor Cores hơn. Với RTX 2070, bạn sẽ có những tính năng này với giá rẻ nhất.**

Tuy nhiên chú ý rằng, phân tích trên này thiên lệch về các thông số sau:

- Thay đổi của giá cả : Hiện tại giá của GTX 1080 Ti, RTX 2080 và RTX 2080 Ti vẫn bị định giá cao và nó có thể phù hợp hơn trong tương lai.

- Phân tích này ưu tiên card nhỏ hơn. Nó cũng không bao gồm bao nhiêu bộ nhớ bạn cần cho mạng của mình hay bao nhiêu GPU có thể lắp trong máy của bạn. Một máy với 4 GPUs nhanh sẽ hiệu quả hơn nhiều về chi phí với 2 máy tính với các card hiệu quả về chi phí.

Cảnh báo, vấn đề nhiệt của Multi-GPU RTX

Có vấn đề với RTX 2080 Ti và các RTX GPu khác với bản tiêu chuẩn gồm 2 quạt tản nhiệt. Điều này đặc biệt đúng cho multiple RTX 2080 Ti trong một máy tính tuy nhiên RTX 2080 và RTX 2070 cũng có thể bị ảnh hưởng. Quạt tản nhiệt cho RTX là thiết kế của NVIDIA dành cho game thủ mà họ chỉ chạy GPU đơn (chạy êm ái và ít nhiệt cho GPU). Tuy nhiên thiết kế này là tồi tệ nếu bạn sử dụng multi-GPU khi nó có 2 quạt tản nhiệt mở. Nếu bạn đặt nhiều RTX cards cạnh nhau (trong các PCIe slot cạnh nhau) thì bạn nên lấy quạt đơn thiết kế dạng thổi. **Điều này đặc biệt đúng cho RTX 2080 TI. Trên thị trường thì ASUS và PNY có các mẫu RTX 2080 TI với thiết kế dạng quạt thổi này**. Nếu dùng RTX 2070 thì sẽ ổn với bất cứ thiết kế quạt nào nhưng cá nhân tôi thì sẽ vẫn lấy quạt thổi nếu như chạy 2 cái RTX 2070 cạnh nhau.

Bộ nhớ cần thiết và 16-bit training

Bộ nhớ trên GPU là điều tối quan trọng cho một số ứng ụng như Computer Vision, Machine Translation và một số ứng dụng NLP khác và bạn có thể nghĩ rằng RTX 2070 là hiệu quả chi phí nhưng bộ nhớ 8Gb thì khá nhỏ. Tuy nhiên nếu dùng 16-bit training bạn sẽ có 16Gb memory ảo và bất kỳ mô hình tiêu chuẩn nào cũng có thể fit vào RTX 2070 dễ dàng nếu bạn dùng 16-bit. Tương tự cho RTX 2080 và RTX 2080 Ti.

Khuyến nghị chung về GPU

Hiện tại thì khuyến nghị của tôi là dùng RTX 2070 và 16-bit training. Tôi không bao giờ khuyến khích mua XP Titan, Titan V hay bất kỳ card Quadro[12] nào, hay bất kỳ Founder Edition (FE) nào. Dưới đây là các GPU cho các mục đích của nó

[11] Xem trong bài gốc.

[12] ND - Card Quadro không nên dùng cho tính toán khoa học vì không tối ưu và chi phí cực kỳ đắt đỏ.

- Với nhiều tiền, RTX 2080 Ti.

- Cho hiệu suất cao hơn : a) RTX 2080 Ti hay b) RTX 2070 bây giờ và bán lại sau đó nâng cấp lên RTX Titan vào 2019 Q1/Q2

- Ít tiền hơn : Titan X (Pascal) tren eBay hay GTX 1060 (6Gb)[13]. Nếu vẫn quá đắt GTX 1050 Ti. Nếu vẫn quá đắt thì hay xem ở Colab[14]

- Nếu bạn chỉ muốn thử bắt đầu với DL : GTX 1050Ti (4Gb)[15] là một lựa chọn tốt.

- Nếu bạn có thể chờ đợi : GTX 1080 Ti hay RTX 2080 Ti thật sự tuyệt nhưng giá của nó thì hơi điên rồ. Hi vọng giá có thể ổn định lại trong tương lai.

- Nếu bạn có sẵn GTX 1080 Ti hay GTX Titan (Pascal) thì bạn sẽ muốn chờ tới RTX Titan. GPU của bạn vẫn ok.

Cá nhân tôi muốn một cái RTX 2080 Ti. Nhưng RTX 2070 quá hiệu quả về chi phí với 16-bit memory ảo tương đương với 16GB và tôi sẽ có thể chạy bất kỳ mô hình nào với nó.

Deep Learning trên mây

Cả GPU instances trên AWS [16] và TPUs trên Google Cloud thì đều là các phương án khả thi cho DL. Trong khi TPU thì rẻ hơn một chút nhưng nó lại thiếu đi sự linh hoạt của GPU AWS. TPUs có thể là một lựa chọn tốt cho training pipeline nhận dạng vật thể. Cho các dạng thức công việc khác thì AWS GPU là một lựa chọn an toàn hơn – điều tốt đẹp của các ứng dụng đám mây này là bạn có thể chuyển đổi dễ dàng giữa GPU và TPU bất kỳ lúc nào hay thậm chí dùng chúng cùng một lúc.

Tuy nhiên, hay để ý tới chi phí cơ hội ở đây : nếu bạn học các kỹ năng để có một work-flow mượt mà hơn với AWZ, bạn mất thời gian mà bạn có thể dùng nó để làm việc với GPU của cá nhân bạn. Và bạn cũng sẽ phải học các kỹ năng sử dụng TPU nữa. Nếu dùng GPU cá nhân, bạn sẽ không phát triển đc kỹ năng này trên nhiều GPUs và TPUs hơn trên cloud. Nếu sử dụng TPUs bạn sẽ bị kẹt với TensorFloz và nó cũng không đơn giản để chuyển thẳng sang AWS. Do vậy học cách vận hành cloud một cách mượt mà khá là tốn kém và bạn nên tính toán chi phí này vào khi bạn cân nhắc quyết định sử dụng TPUs và AWS

Một câu hỏi nữa cũng liên quan tới việc khi nào sử dụng đám mây. Nếu bạn muốn học DL hay bạn muốn tạo nguyên mẫu thì một GPU cá nhân có lẽ là lựa chọn tốt nhất vì cloud có thể sẽ khá đắt đỏ. Tuy nhiên nếu bạn đã tìm ra một cấu hình mạng deep network tốt và bạn chỉ muốn train nó sử dụng cấu trúc dữ liệu song song trên mây thì đây là một cách tiếp cận khá chắc chắn. Điều này có nghĩa là

[13] Chú ý là có cả bản 3Gb
[14] Google Colab for free GPU
[15] Chú ý là có cả bản 2Gb
[16] AWS: Amazon Web Service (dịch vụ điện toán đám mây của Amazon)

một GPU nhỏ hơn sẽ đủ cho làm nguyên mẫu và chúng ta có thể dựa trên sức mạnh đám mây để mở rộng tới các thử nghiệm lớn hơn.

Nếu bạn thiết hụt tiền bạc thì điện toán đám mây cũng có thể là giải pháp tốt, nhưng vấn đề là đôi khi bạn phải mua rất nhiều giờ tính toán trong khi bạn chỉ cần một chút cho làm nguyên mẫu. Trong trường hợp này bạn chỉ nên làm nguyên mẫu trên CPU rồi hãy chuyển nó sang GPU/TPU để chạy nhanh hơn. Đây cũng không phải là work-flow tốt nhất vì làm nguyên mẫu trên CPU thì rất là khốn khổ nhưng nó là một giải pháp hiệu quả về chi phí.

Kết luận

Với các thông tin kể trên thì hi vọng bạn có thể lựa chọn được GPU phù hợp với mình. Nói chung với tôi thì 2 chiến thuật sau đây là hợp lý : lấy một RTX 20 series để có nâng cấp nhanh nhất hoặc lấy một GTX 10 Series rẻ và nâng cấp lên RTX Titan khi nó xuất hiện. Nếu như bạn không quá nghiêm trọng về hiệu suất hay bạn không cần nó, ví dụ như dùng trong Kaggle, Startup, làm nguyên mẫu (prototyping) hay học Deep Learning, bạn có thể tận dụng các GTX 10 Series giá rẻ. Chú ý rằng các mẫu này cần có bộ nhớ đáp ứng được công việc của bạn.

Lời khuyên sau cùng

- **GPU chung tốt nhất:** RTX 2070
- **GPU cần tránh:** Tesla Card, Quadro Card, Founder Edition Card, Titan V, Titan XP. (ND: Chúng tôi bổ sung thêm khuyến nghị là tránh các GPU ép xung (overclocked). Các GPU này rất đắt đỏ về chi phí nhưng gần như không mang lại hiệu quả đáng kể nào cho DL. Điều này cũng được chính tác giả bài viết này công nhận trong một comment ở bài viết gốc, hoặc các bạn cũng có thể dễ dàng tìm được trên google[17])
- **Hiệu quả chi phí nhưng đắt**: RTX 2070
- **Hiệu quả chi phí và rẻ:** GTX Titan (Pascal) trên ebay, GTX 1060 (6GB), GTX 1050 Ti (4Gb)
- **Tôi có ít tiền:** GTX Titan (Pascal) trên ebay, GTX 1060 (6GB), GTX 1050 Ti (4Gb)
- **Tôi gần như không có tiền:** GTX 1050 Ti (4GB); CPU (làm nguyên mẫu) + AWS/TPU (training); or Colab.

[17] Ví dụ như trong các link:

- Bài viết gốc: Tác giả: OC GPUs are good for gaming, but they hardly make a difference for deep learning. You are better of buying a GPU with other features such as better cooling. When I tested overclocking on my GPUs it was difficult to measure any improvement. Maybe you will get something in the range of 1-3% improved performance for OC GPUs — so not so much worth it if you need to pay extra for OC.
- https://medium.com/@timyee90/does-overclocking-cpu-gpu-improve-deep-learning-training-speed-2488f9cf4dbd
-

- **Tôi thi Kaggle:** RTX 2070. Nếu tôi không đủ tiền thì hãy dùng GTX 1060(6GB) hay GTX Titan (Pascal) trên ebay cho làm nguyên mẫu và AWS cho training. Sử dụng fastai nữa nhé.
- **Tôi là một người làm Computer Vision hay nghiên cứu Machine Translation** : GTX 2080 Ti với quạt thổi và nâng cấp lên RTX Titan vào 2019
- **Tôi nghiên cứu NLP**: RTX 2070 dùng 16-bit
- **Tôi muốn tạo ra GPU Cluster**: phức tạp đấy, xem vài ý tưởng ở đây nhé
- **Tôi mới bắt đầu DL và tôi nghiêm túc về nó**: Bắt đầu với RTX 2070 sau đó mua thêm RTX 2070 sau 6-9 tháng nếu bạn vẫn muốn đầu tư thời gian vào DL. Dựa trên lĩnh vực của bạn (startup, kaggle, nghiên cứu, ứng dụng DL) bán GPU và nâng cấp lên một thứ phù hợp hơn sau khoảng hai năm.
- **Tôi muốn thử DL và không quá nghiêm túc về nó**: GTX 1050 Ti (4 hay 2Gb). Thường thì nó phù hợp với desktop tiêu chuẩn. Nếu được thì đừng có mua máy tính mới đấy nhé.

Các phiên bản của bài viết:

- Update 2018-11-26: Thêm vào bàn luận về vấn đề quá tải nhiệt của RTX 2080
- Update 2018-11-05: Thêm vào RTX 2070 và sửa lại phần khuyến nghị cùng với các đồ thị. Sửa lại phần TPU
- Update 2018-08-21: Thêm vào RTX 2080 và RTX 2080 Ti, làm lại phần phân tích hiệu suất
- Update 2017-04-09: Thêm vào phân tích chi phí, thêm vào khuyến nghị cho NVDIA Titan Xp
- Update 2017-03-19: Thêm vào GTX 1080 Ti.
- Update 2016-07-23: Thêm vào Titan X Pascal và GTX 1060; update khuyến nghị
- Lược bỏ các update cũ hơn

Titan RTX Deep Learning Benchmarks

ND: Dưới đây là một bài viết khác của lambda lab, một công ty bán các giải pháp phần cứng. Đánh giá này chỉ bao gồm các GPU đầu bảng của NVIDIA.

Link: https://lambdalabs.com/blog/titan-rtx-tensorflow-benchmarks/

Hardware Setup: Lambda Dual - 2x Titan RTX Deep Learning Workstation.

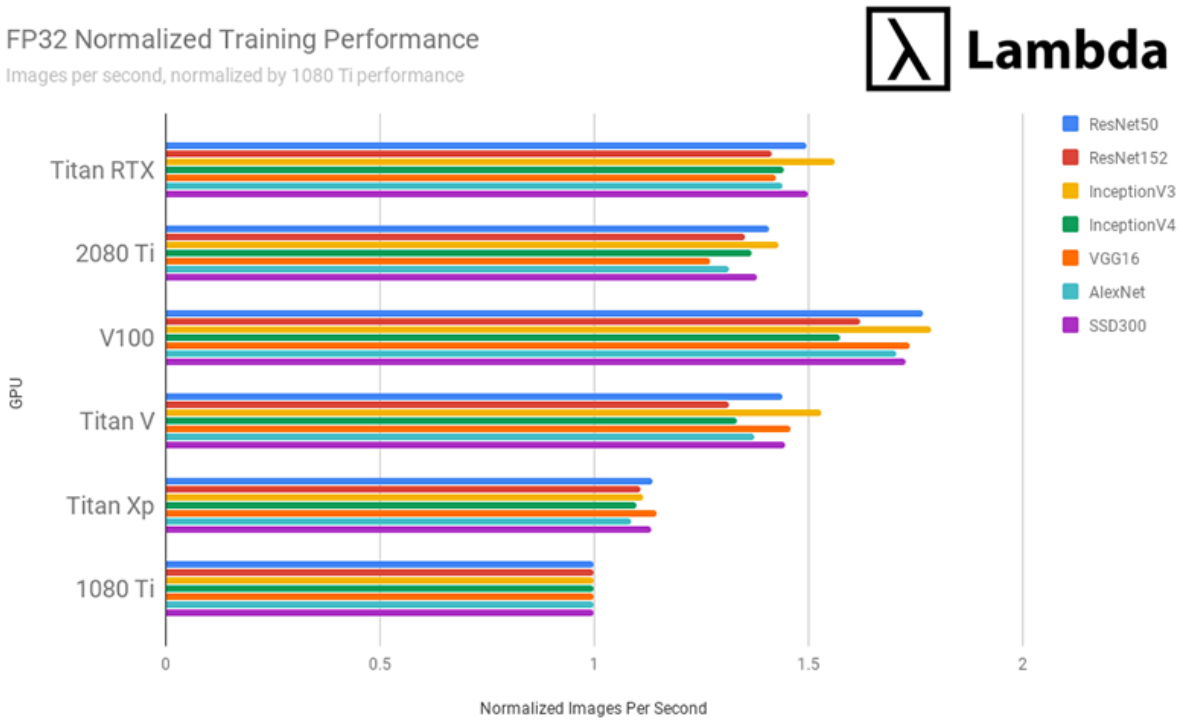
Kết quả

Titan RTX's FP32 hiệu suất:

- 8% nhanh hơn RTX 2080 Ti
- 47% nhanh hơn GTX 1080 Ti
- 31% nhanh hơn Titan Xp
- 4% nhanh hơn Titan V
- 14% nhanh hơn Tesla V100 (32 GB)

The Titan RTX's FP16 hiệu suất

- 21% nhanh hơn RTX 2080 Ti
- 110% nhanh hơn GTX 1080 Ti
- 92% nhanh hơnTitan Xp
- 2% nhanh hơn Titan V



Hình 4: Benchmark của Lambda Lab trên nhiều mạng khác nhau

Kết luận

- *RTX 2080 Ti là GPU tốt nhất cho DL* với 11GB bộ nhớ. Nó đủ dùng cho hầu hết các nhu cầu. 2080 Ti có hiệu suất cao nhất trong số các Card sau đây: Titan RTX, Tesla V100, Tesla V, GTX 1080 Ti và Titan Xp

- *Titan RTX là GPU tốt nhất cho DL nếu như bộ nhớ 11GB không đủ dùng cho nhu cầu của bạn.* Tuy nhiên trước khi kết luận thì hãy nhớ sử dụng 16-bit training. Điều này sẽ gấp đôi số GPU memory của bạn lên. Nếu như ngay cả với điều này mà vẫn không đủ cho nhu cầu của bạn, hãy chọn Titan RTX. Nếu không thì RTX 2080 Ti là đủ rồi. Với 16-bit training Titan RTX cho bạn tới 48 GB bộ nhớ.

- *Tesla V100 là GPU tốt nhất nếu như tiền không phải là vấn đề.*

Phương pháp[18]

- GPU hiệu suất được tính toán tránh vấn đề thắt cổ chai với CPU
- Với mỗi GPU model thì 10 thí nghiệm được thực hiện và lấy trung bình
- Hiệu suất huấn luyện tiêu chuẩn của một GPU được tính bằng cách chia hiệu suất hình trên giây của từng mô hình với hiệu suất hình trên giây của 1080 Ti trên cùng mô hình đó.
- The Titan RTX, 2080 Ti, Titan V, and V100 benchmarks sử dụng Tensor Cores.

Batch-sizes

Model	**Batch Size**
ResNet-50	64
ResNet-152	32
InceptionV3	64
InceptionV4	16
VGG16	64
AlexNet	512
SSD	32

Software

- Ubuntu 18.04
- TensorFlow: v1.11.0

[18] Có thể thực hiện lại benchmark này, hướng dẫn thực hiện có trong bài viết gốc

- CUDA: 10.0.130
- cuDNN: 7.4.1
- NVIDIA Driver: 415.25

Kết quả thô

FP32 - Number of images processed per second

Model / GPU	**Titan RTX**	**1080 Ti**	**Titan Xp**	**Titan V**	**2080 Ti**	**V100**
ResNet50	312	208	237	300	294	369
ResNet152	115	81	90	107	110	132
InceptionV3	212	136	151	208	194	243
InceptionV4	83	58	63	77	79	91
VGG16	191	134	154	195	170	233
AlexNet	3980	2762	3004	3796	3627	4708
SSD300	162	108	123	156	149	187

FP16 - Number of images processed per second

Model / GPU	**Titan RTX**	**1080 Ti**	**Titan Xp**	**Titan V**	**2080 Ti**
ResNet50	540	263	289	539	466
ResNet152	188	96	104	181	167
InceptionV3	342	156	169	352	286
InceptionV4	121	61	67	116	106
VGG16	343	149	166	383	255
AlexNet	6312	2891	3104	6746	4988
SSD300	248	122.49	136	245	195

Một hướng dẫn về phần cứng đầy đủ cho Deep Learning

Dịch từ link, của tác giả Tim Dettmers, ngày 16/12/2018

Tiếp nối với bài hướng dẫn chọn GPU cho Deep Learning, chúng tôi xin giới thiệu với các bạn một bài hướng dẫn của cùng tác giả về việc chọn lựa các thành phần khác của máy tính để có thể lắp ráp được một hệ thống hoàn chỉnh. Ở đây tác giả chú trọng vào hệ thống cho vận hành Deep Learning, do vậy chúng tôi cũng có một số khuyến khị khác nếu như bạn có nhu cầu xây dựng một hệ thống phục vụ cho các dự án data science khác.

Deep learning (DL) rất là nặng về tính toán, vì vậy nên bạn sẽ cần một CPU thật nhanh với nhiều nhân phải không? Hay là việc mua một CPU nhanh chỉ là một sự lãng phí? Một trong những điều tệ hại nhất khi xây dựng một hệ thống Deep Learning là lãng phí tiền của vào phần cứng mà nó hoàn toàn vô dụng. Tôi sẽ hướng dẫn các bạn dưới đây từng bước một qua những phần cứng mà bạn cần để có một hệ thống đạt hiệu năng cao mà lại rẻ.

Trong những năm qua, tôi đã lắp ráp 7 workstation khác nhau cho Deep Learning và mặc dù đã có những nghiên cứu và suy luận cẩn thận, tôi vẫn lặp lại những sai lầm khi lựa chọn các phần cứng khác nhau. Trong hướng dẫn này, tôi muốn chia sẻ lại kinh nghiệm mà tôi đã học được trong những năm qua để các bạn không mắc phải những sai lầm như tôi trước đây nữa.

GPU

Giả sử rằng các bạn sẽ sử dụng GPU cho DL. Nếu bạn đang lắp ráp hoặc nâng cấp hệ thống của bạn cho DL, không thể bỏ qua GPU được. GPU là trái tim của ứng dụng DL – việc tăng tốc độ khủng khiếp của sự tính toán là không thể bỏ qua được.

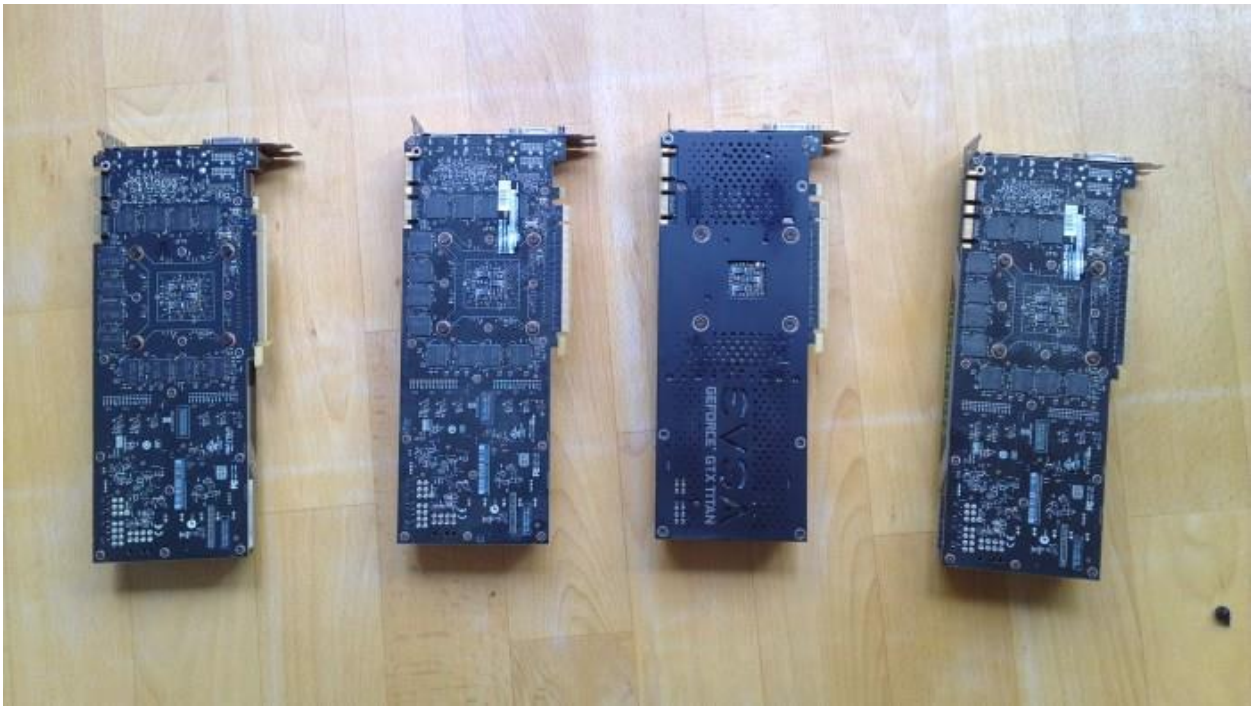
Các bạn có thể xem bài hướng dẫn dài về việc lựa chọn GPU ở phần trên, và việc lựa chọn GPU có lẽ là sự lựa chọn quan trọng nhất cho toàn bộ hệ thống. Có ba lỗi lầm chính khi chọn GPU: (1): không phù hợp về chi phí/ hiệu năng, (2) không đủ bộ nhớ, (3) kém về tản nhiệt.

Về sự phù hợp chi phí/ hiệu năng, tôi khuyến nghị sử dụng RTX 2070 hay RTX 2080 Ti. Nếu dùng card này thì các bạn hãy chạy 16-bit model nhé (ND: xem bài viết trên để rõ ràng hơn). Ngoài ra GTX 1070, GTX 1080, GTX 1070 Ti và GTX 1080 Ti từ ebay cũng là sự lựa chọn khá hợp lý và bạn có thể dùng chúng với 32-bit (không phải 16-bit).

Nói chung thì khuyến nghị cho bộ nhớ là như sau:

- Nghiên cứu để tìm ra điểm số state-of-the-art: >= 11 GB
- Nghiên cứu để tìm ra các kiến trúc: >= 8 GB
- Các nghiên cứu khác: 8 GB
- Kaggle: 4 – 8 GB
- Startups: 8 GB, nhưng hãy nhớ kiểm tra kích cỡ model cho lĩnh vực của bạn nhé.
- Công ty: 8 GB cho làm nguyên mẫu (prototyping), => 11 GB để training.

Một vấn đề nữa cần lưu tâm, đặc biệt là các hệ thống Multi-GPU RTX là tản nhiệt. Nếu bạn muốn gắn GPUs vào PCIe slot cạnh nhau thì bạn cần phải chắc rằng mình lấy các mẫu GPU có thiết kế quạt thổi. Nếu không bạn sẽ gặp vấn đề quá nhiệt và GPUs của bạn không những sẽ chậm hơn mà còn chết nhanh hơn.



Hình 5: Bạn có thể xác định phần cứng nào gây ra hiệu năng tệ hại không? Một trong những chiếc GPU này? Hay là CPU?

RAM

Một trong những sai lầm là mua RAM có clock rate (xung nhịp) quá cao. Sai lầm thứ hai là không mua đủ RAM để có một trải nghiệm làm nguyên mẫu mượt mà.

Clock Rate cần thiết cho RAM

Xung nhịp của RAM chỉ là một chiêu trò marketing của các công ty RAM nhằm hút bạn mua những thanh RAM nhanh hơn mà trong thực tế nó không mang lại quá nhiều hiệu quả. Điều này được giải thích trong bài viết sau đây: Liệu tốc độ của RAM có thực sự liên quan? Bởi RAM von Linus Tech Tips.

Hơn nữa, cần phải biết rằng tốc độ của RAM không liên quan tới sự truyền tải CPU RAM tới GPU RAM. Điều này là bởi vì (1) nếu bạn sử dụng pinned memory, mini-batches sẽ được chuyển trực tiếp tới GPU mà không thông qua CPU và (2) nếu bạn không sử dụng pinned memory thì việc sử dụng RAM nhanh và chậm chỉ có khác biệt khoảng 0 tới 3%. Vậy hãy để tiền của bạn vào việc khác!

Kích cỡ RAM

Kích cỡ RAM thì không giúp gì cho hiệu năng của DL cả. Tuy nhiên, nó có thể cản trở việc thực thi mã GPU của bạn một cách thoải mái (mà không thực hiện swapping to disk). Bạn nên có đủ RAM để làm việc thoải mái với GPU, điều này có nghĩa là bạn **nên có số RAM ít nhất là phù hợp** RAM trong GPU lớn nhất của bạn. Ví dụ như nếu bạn có Titan RTX với bộ nhớ 24G, bạn nên có ít nhất 24GB RAM. Tuy nhiên nếu bạn có nhiều GPU hơn, bạn không nhất thiết phải cần thêm RAM.

Vấn đề với cách làm này là đôi khi bạn có thể sẽ vẫn bị thiếu RAM nếu như bạn làm việc với các bộ dữ liệu lớn (large datasets). Chiến lược tốt nhất ở đây là dùng RAM phù hợp với GPU của bạn sau đó bổ sung sau nếu như bạn cảm thấy không đủ dùng.

Một chiến lược khác là chiến lược ảnh hưởng từ tâm lý học: Tâm lý học nói rằng sự tập trung là một nguồn tài nguyên bị cạn kiệt theo thời gian. RAM là một trong số ít các phần cứng cho phép bạn tiết kiệm tài nguyên cho các vấn đề lập trình khó khăn hơn. Vì vậy thay vì dành nhiều thời gian cho vấn đề tuần hoàn sự tắc cổ chai của RAM thì bạn nên tập trung vào các vấn đề cấp bách hơn nếu bạn có

nhiều RAM hơn. Với rất nhiều RAM, bạn tránh được vấn đề thắt cổ chai, tiết kiệm thời gian và tăng năng suất cho các các vấn đề cấp bách. Đặc biệt là trong các cuộc thi Kaggle, tôi thấy rằng việc có thêm RAM rất hữu ích cho feature engineering. Nếu bạn có tiền và phải làm nhiều preprocess thì việc có thêm RAM là một sự lựa chọn tốt. Với chiến thuật này thì bạn sẽ muốn có nhiều RAM giá rẻ hơn vào hiện tại.[19]

CPU

Sai lầm lớn nhất mà người ta hay mắc phải là chú trọng quá nhiều vào số PCIe lanes của CPU. Bạn không nên quan tâm nhiều về số PCIe lanes này. Thay vào đó, hay xem xem tổ hợp CPU và bo mạch chủ (motherboard) có hỗ trợ số GPU mà bạn muốn chạy không. Sai lầm cơ bản thứ hai là dùng một cái CPU quá mạnh mẽ[20].

CPU và PCI-Express

Mọi người phát điên về PCIe lanes! Tuy nhiên điều này gần như chả có tác dụng gì cho hiệu năng của DL cả. Nếu như bạn có một GPU, PCIe lanes chỉ dùng để chuyển data từ CPU RAM tới GPU RAM nhanh hơn. Tuy nhiên, một mạng ImageNet batch của 32 ảnh (32x225x225x3) và 32-bit cần 1.1 mili-giây với 16 lanes, 2.3 mili-giây với 8 lanes và 4.5 mili-giây với 4 lanes. Đây là con số lý thuyết và trong thực tế bạn sẽ thấy rằng PCIe sẽ chậm hơn gấp đôi – nhưng nó vẫn là nhanh như chớp vậy! PCIe lanes thường có độ trễ trong khoảng nano-giây và độ trễ này có thể bỏ qua được.

Kết hợp vào với nhau thì chúng ta có một ImageNet mini-batch 32 ảnh và một ResNet-152 với timing như sau:

- Forward và backward pass: 216 mili-giây
- 16 PCI lanes CPU -> GPU transfer: khoảng 2 mili-giây (lý thuyết 1.1 mili-giây)
- 8 PCI lanes CPU -> GPU transfer: khoảng 5 mili-giây (lý thuyết 2.3 mili-giây)
- 4 PCI lanes CPU -> GPU transfer: khoảng 9 mili-giây (lý thuyết 4.5 mili-giây)

Do vậy từ 4 lên 16 PCIe lanes chỉ làm tăng hiệu năng lên khoảng 3.2%. Tuy nhiên, nếu bạn sử dụng PyTorch's data loader với pinned memory thì bạn sẽ làm tăng hiệu năng lên chính xác là 0%. Vì vậy đừng có phí tiền vào PCIe lanes nếu bạn chỉ sử dụng GPU đơn.

Nếu bạn lựa chọn CPU PCIe lanes và motherboard PCIe lanes thì hãy nhớ chọn tổ hợp hỗ trợ số GPUS của bạn. Nếu như bạn mua motherboard hỗ trợ 2 GPUs và nếu bạn chỉ muốn có 2 GPUs, hãy chắc rằng bạn chỉ mua một CPU hỗ trợ 2 GPUs, chứ không nhất thiết xem tới số PCIe lanes nhé.

PCIe Lanes và Multi-GPU Parallelism

[19] ND: khuyến nghị từ kinh nghiệm của chúng tôi là nếu hệ thống còn được sử dụng cho các dự án Data Science nói chung thì kích cỡ RAM nên ít nhất là gấp đôi kích cỡ data mà bạn thường xuyên làm việc cho dự án, nếu có thể, để thuận tiện cho việc data processing/ feature engineering. Khi thực hiện data processing, một số thuật toán đơn giản là nhân đôi (duplicate) data của bạn trong RAM và thực hiện tính toán. Hoặc từ một vài feature ban đầu, có thể các bạn sẽ tạo ra x5-x10 feature phái sinh, điều này sẽ làm phình lớn data của bạn đáng kể.
Xu hướng giá RAM nói chung là hạ nên việc sử hữu 16-24-32 GB RAM cũng không còn là điều gì quá khó khăn, đặc biệt là khi chúng ta không cần sử dụng RAM có tốc độ xung nhịp cao.
Dĩ nhiên nếu như kích cỡ data quá lớn thì nên sử dụng các giải pháp như SQL hay HDF5.

[20] Khuyến nghị này của tác giả là ám chỉ vào hệ thống DL. Với một dự án Data Science nói chung, một CPU đa nhân sẽ cho phép tăng tốc độ bước preprocessing hoặc algo không phải DL của bạn với parallelism. Loạt chip Ryzen mới ra mắt những năm gần đây của AMD cho phép sở hữu CPU với nhiều nhân và luồng hơn với mức giá hợp lý. Vì vậy *chúng tôi khuyến nghị sử dụng Ryzen để có tỉ lệ chi phí/hiệu năng tốt nhất*. Theo kinh nghiệm của chúng tôi, *không cần thiết phải sử dụng CPU có xung nhịp cao hơn hay thực hiện ép xung (overclocking) cho CPU*. Điều này sẽ tiết kiệm khá nhiều chi phí cho bạn, đặc biệt là chi phí cho tản nhiệt nếu ép xung.

Vậy thì số PCIe lanes có quan trọng không nếu bạn train neural networks trên multi-GPU với data parallelism ? Tôi đã công bố một nghiên cứu về điều này ở ICLR2016 và tôi có thể nói với bạn nếu bạn có là 96 GPUs thì số PCIe lanes là cực kỳ quan trọng. Tuy nhiên nếu bạn chỉ có 4 hay ít hơn GPUs thì điều này không ảnh hưởng lắm đâu. Nếu như bạn song song hóa 2-3 GPUs thì tôi còn chả quan tâm tới số PCIe lanes nữa cơ. Với 4 GPUs thì tôi sẽ muốn chắc chắn rằng tôi có thể có hỗ trợ 8 PCIe lanes trên một GPU (và 32 cho tổng số). Vì gần như chả có ai chạy một hệ thống với nhiều hơn 4 GPUs nên luật bất thành văn là : Đừng có xài thêm tiền để có thêm PCIe lanes trên một GPU – chúng chả liên quan gì tới nhau đâu.

Số CPU cores cần thiết

Để có thể có lựa chọn phù hợp thì chúng ta cần phải hiểu CPU và cách nó liên quan tới DL. CPU làm được gì cho DL nhỉ ? CPU chỉ tính toán một chút xíu nếu bạn chạy DL trên GPU. Hầu như nó chỉ (1) khởi tạo lệnh gọi các hàm GPU, và (2) thực hiện các hàm trên CPU.

Ứng dụng lớn nhất cho CPU của các bạn là data preprocessing. Có 2 chiến thuật phổ biến về điều này và chúng có nhu cầu về CPU khác nhau.

Thứ nhất là preprocessing khi bạn train :

Vòng lặp như sau :

1. Load mini-batch
2. Preprocess mini-batch
3. Train trên mini-batch

Thứ hai là preprocessing trước và sau training :

1. Preprocess data
2. Vòng lặp :
    a. Load preprocessed mini-batch
    b. Train trên mini-batch

Cho chiến thuật một thì một CPU với nhiều nhân có thể sẽ tăng hiệu năng lên đáng kể. Với chiến thuật thứ hai thì bạn không cần có một CPU quá tốt đâu. Với chiến thuật một thì tôi khuyến nghị tối thiểu 4 luồng cho một GPU, tức là khoảng 2 nhân cho một GPU. Tôi chưa kiểm tra điều này kỹ càng nhưng bạn sẽ tăng được khoảng 0-5% cho mỗi nhân tăng thêm cho một GPU.

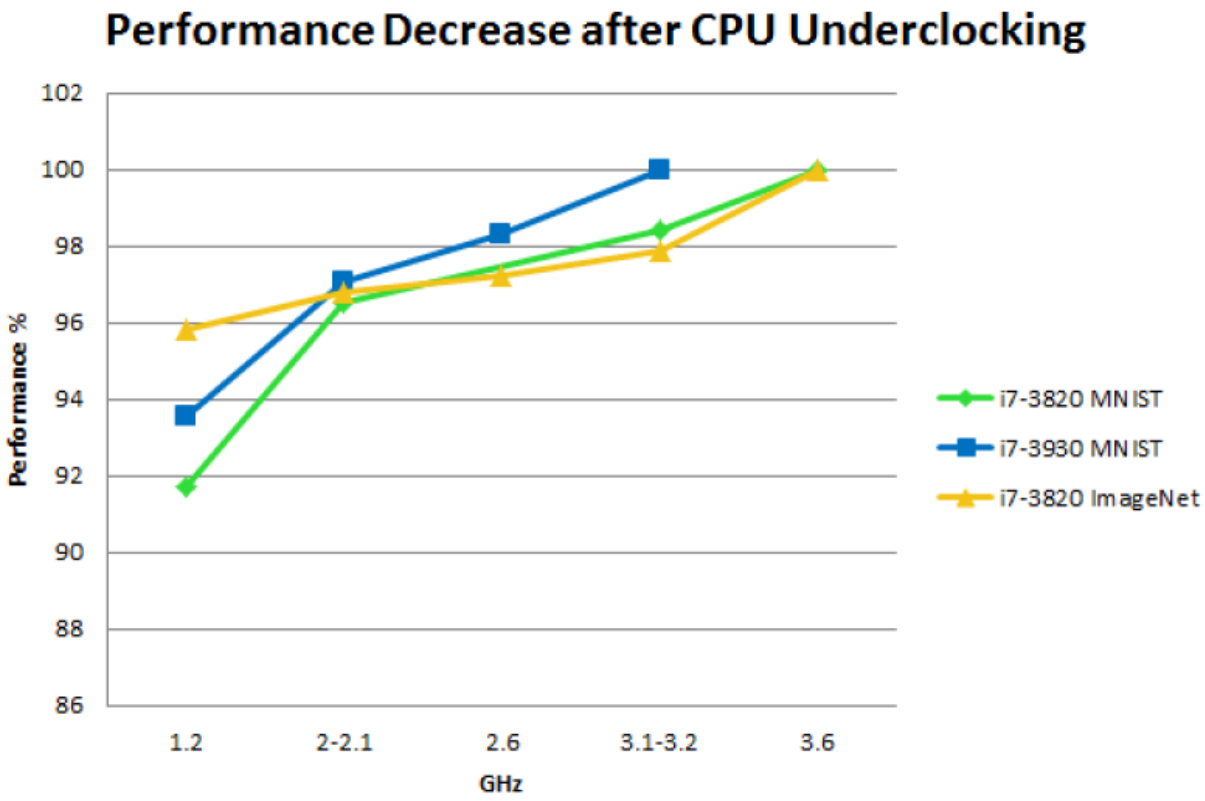
Với chiến thuật thứ hai thì tôi khuyến nghị tối thiểu 2 luồng cho mỗi GPU, tức là khoảng một nhân cho một GPU. Bạn sẽ chẳng thấy được sự tăng tốc nào về hiệu năng đáng kể nếu như bạn có nhiều nhân hơn khi bạn sử dụng chiến thuật này đâu.

Xung nhịp CPU (tần suất)

Khi người ta nghĩ về CPUs nhanh thì người ta thường nghĩ tới xung nhịp của nó. 4GHz thì tốt hơn là 3.5 GHz đúng không ? Điều này nói chung là đúng khi so sánh processor (vi xử lý) với cùng kiến trúc, ví dụ như Ivy Bridge, nhưng nó thường không tốt khi so sánh giữa các processor với nhau. Nó không hẳn lúc nào cũng là một chỉ số tốt cho hiệu năng.

Trong trường hợp của DL thì gần như chỉ có một chút xíu tính toán được thực hiện bởi CPU : tăng vài biến ở đây, đánh giá vài phép tính Boolean ở kia, gọi vài hàm trên GPU hay trong chương trình. Và tất cả nhưng điều này phụ thuộc vào xung nhịp CPU.

Trong khi lý luận này khá là hợp lý, thì có một thực tế là khi CPU bị sử dụng 100% khi tôi chạy DL. Vậy vấn đề ở đây là gì ? Tôi đã thực hiện vài thử nghiệm hạ xung CPU để xem :



*Hình 6: Hạ xung trên MNIST và ImageNet: Hiệu năng được đo bằng thời gian trên 200 epochs MNIST hoặc một phần tư epoch trên ImageNet với các xung nhịp khác nhau của CPU, với xung nhịp cao nhất được dùng làm tiêu chuẩn so sánh. Để so sánh thì **nâng cấp từ GTX 680 lên GTX Titan tăng 15% hiệu năng, GTX Titan lên GTX 980 được 20% nữa. GPU overclocking chỉ tăng khoảng 5% hiệu năng cho tất cả GPU**.*

Chú ý rằng thí nghiệm trên được thực hiện cho các phần cứng đã lỗi thời, tuy nhiên thì kết quả sẽ là tương tự cho các CPU/GPU hiện đại hơn.

Hard drive/SSD[21]

Hard drive thường không phải là lý do nghẽn cổ chai trong DL. Tuy nhiên, nếu như bạn làm điều ngu ngốc thì nó làm tổn hại bạn đấy: nếu như bạn đọc dữ liệu từ đĩa khi nó đang bị sử dụng (block wait) thì 100 MB/s hard drive sẽ tốn của bạn 185 mili-giây cho một ImageNet mini-batch size 32 – ái chà! Tuy nhiên, nếu bạn fetch dữ liệu không đồng bộ trước khi nó được sử dụng (ví dụ như torch vision loaders), thì bạn sẽ load mini-batch trong 185 mili-giây trong khi thời gian tính toán cho hầu hết các mạng DL trên ImageNet là khoảng 200 mili-giây. Do đó bạn sẽ không bị trừng phạt chút nào về hiệu năng cả nếu như bạn load mini-batch kế tiếp trong khi đang thực hiện tính toán.

Tuy nhiên thì **tôi khuyên dùng sử dụng SSD cho thoải mái và có hiệu năng cao hơn**: các phần mềm sẽ khởi động và phản hồi nhanh hơn, và preprocessing với các file lớn hơn thì cũng nhanh hơn một chút nữa. Nếu **bạn mua một SSD NVMe thì bạn sẽ có trải nghiệm mượt mà hơn cả một SSD thông thường.**

Do vậy, thiết lập lý tưởng là một ổ cứng lớn và chậm cho lưu trữ dữ liệu và SSD cho năng suất và sự thoải mái.

Bộ cấp nguồn (Power Supply Unit – PSU)

[21] https://tinhte.vn/threads/tim-hieu-ssd-chuan-sata-m-2-va-nvme.2866611/

Thông thường thì bạn muốn một PSU có khả năng cung cấp đủ cho tất cả future GPUs của bạn. GPUS thường thì có hiệu quả về năng lượng theo thời gian; vì vậy trong khi các thành phần khác cần được thay thế thì một PSU nên được giữ lại trong một thời gian dài, do vậy một PSU tốt là một khoản đầu tư có lợi.

Bạn có thể tính toán số watts cần thiết bằng cách cộng số watt của CPU và GPUS với tầm 10% số watts cho các thành phần khác nữa và làm bộ đệm cho các đột biến về năng lượng. Ví dụ nếu bạn có 4 GPUs với 250 watts TDP và một CPU với 150 watts TDP thì bạn cần CPU với tối thiểu 4*250+150+100 = 1250 watts. Tôi thường sẽ thêm vào 10% để đảm bảo mọi thứ đều ổn, do vậy nên tôi sẽ cần tổng cộng khoảng 1375 watts. Tôi sẽ làm tròn và lấy một cái PSU 1400 watts[22].

Một phần quan trọng cần lưu ý là ngay cả PSU có số watts cần thiết, có thể không có đủ số đầu nối PCIe 8 pin hay 6 pin. Hãy chắc chắn rằng bạn có đủ số kết nối trên PSU để hỗ trợ tất cả GPU của bạn!

Một điều quan trọng nữa là mua PSU có xếp hạng hiệu năng sử dụng điện cao – đặc biệt là khi bạn có nhiều GPUs và bạn chạy nó cho thời gian dài.

Chạy một hệ thống 4 GPU trên toàn bộ công suất (1000-1500 watts) để train một mạng convolution cho 2 tuần sẽ tốn khoảng 300-500 kWh, mà ở Đức với chi phí điện khá cao (20 cents/ kWh) thì sẽ rơi vào khoảng 60-100€ (66-111$). Nếu mức giá này cho 100% hiệu quả thì train một mạng như vậy với nguồn chỉ cấp 80% sẽ làm tăng thêm 18-26 – oái! Điều này sẽ ít hơn nếu sử dụng một GPU, nhưng nó vẫn đúng - hãy chi tiền nhiều hơn một chút cho một bộ nguồn có hiệu năng tốt rõ ràng là hợp lý hơn[23].

Tản nhiệt cho CPU và GPU

Tản nhiệt quan trọng và có thể là một nút thắt cổ chai làm giảm hiệu năng nhiều hơn là sự lựa chọn tồi các mảnh phần cứng. Bạn sẽ ổn nếu chỉ sử dụng một bộ giảm nhiệt tiêu chuật hoặc một bộ tản nhiệt nước all-in-one (AIO) cho GPU của mình, tuy nhiên tản nhiệt cho GPU mới là thứ khiến bạn phải suy nghĩ đấy.

Tản nhiệt khí cho GPUs

Tản nhiệt khí thì an toàn và ổn cho một GPU, hoặc với nhiều GPU mà có khoảng cách giữa chúng, (ví dụ 2-3 GPUs trong 3-4 GPU case). Tuy nhiên một trong những sai lầm lớn nhất là khi bạn cố gắng làm mát 3-4GPUs và bạn cần cẩn thận về cách lựa chọn của mình trong trường hợp này.

Các GPUs hiện đại sẽ có tốc độ cao hơn - do vậy sẽ tiêu thụ nhiều điện hơn - tới mức tối đa khi chúng chạy thuật toán. Nhưng ngay cả khi GPU chạm tới rào cản của nó – thường vào khoảng 80 độ C – GPU sẽ giảm tốc để không vượt quá ngưỡng này. Vậy nên đừng để GPU của bạn quá nóng sẽ cho hiệu năng tốt nhất.

Tuy nhiên, thông thường thì các tốc độ quạt được lập trình sẵn được thiết kế khá tồi cho chương trình DL, do vậy ngưỡng nhiệt độ này dễ chạm phải chỉ vài giây sau khi chạy chương trình DL. Kết

---

[22] Thay vì tính toán như tác giả, chúng tôi *sử dụng công cụ tính toán PSU sau* http://www.coolermaster.com/power-supply-calculator/ .Công cụ này cho phép nhập toàn bộ linh kiện của máy tính và để tính ra công suất tiêu thụ chính xác nhất, cũng như giá điện dự kiến bạn phải trả. Để an toàn thì chúng tôi *cũng sử dụng cách tính phụ trội 10% và làm tròn như tác giả.*

[23] Nếu có thể thì chúng tôi khuyến nghị sử dụng PSU cấp đồng (bronze) trở lên, hoặc nếu có điều kiện thì cấp vàng (gold). Chi tiết các bạn có thể tham khảo ở đây: https://www.tomshardware.com/news/what-80-plus-levels-mean,36721.html
Một chia sẻ từ kinh nghiệm của chúng tôi là với một máy có công suất 800 watts – CPU 8 core, một GTX 1700 và PSU gold chạy liên tục trong 3 tuần max công suất, tiền điện chúng tôi phải trả là 120€ (chi phí điện tại Pháp thấp hơn một chút so với tại Đức như trong bài báo: 14.72 cents vs 20 cents/ kWh). *Do vậy các bạn hãy chuẩn bị tinh thần trước là nếu làm nhiều DL sẽ tốn khá nhiều tiền điện nhé.* Nếu như thời gian training ngắn, chúng ta có thể chạy máy vào giờ có giá điện thấp như buổi đêm.

quả là hiệu suất giảm (0-10%), khá nhiều, và đặc biệt là đối với GPUs (10-25%) khi các GPU tỏa nhiệt lẫn nhau.

Vì GPU NVIDIA lqf GPU chơi game nên nó được tối ưu cho windows. Bạn có thể thay đổi tốc độ quạt chỉ với vài cú click chuột trong Win nhưng trong Linux thì không phải như vậy, và hầu hết các thư viện DL được viết cho Linux thì điều này trở thành một vấn đề.

Tùy chọn duy nhất trong Linux là sử dụng cấu hình cho Xorg Server (Ubuntu), bạn có thể đặt tùy chọn "coolbits". Nó hoạt động khá tốt cho một GPU nhưng nếu bạn có nhiều GPU và vài cái không được gắn màn hình thì bạn sẽ phải giả lập màn hình (điều này khá là khó). Tôi đã thử nó trong một thời gian dài và có hàng giờ bực bội với một live boot CD của tôi để khôi phục lại cài đặt đồ họa của mình. Có lẽ tôi sẽ chẳng bao giờ làm nó chạy được với mấy cái GPUs không gắn màn này mất.

Điểm quan trọng nhất cần lưu tâm là nếu bạn chạy 3-4 GPUs với tản khí thì hãy chú ý tới thiết kế thổi. Thiết kế thổi sẽ đẩy khí ra khỏi lưng vỏ máy và khí mát sẽ được đẩy vào trong GPU. Quạt không thổi sẽ hút khí trong GPU và làm mát nó. Nhưng nếu bạn có nhiều GPU cạnh nhau thì sẽ không có khí mát xung quanh GPU và quạt không thổi sẽ làm chúng ngày một nóng lên cho tới khi chúng bị throttle[24] để có thể trở nên mát hơn. **Tránh thiết kế quạt không thổi với 3-4 GPUs bằng mọi giá nhé.**

Tản nhiệt nước cho nhiều GPU

Một giải pháp tốn kém và thủ công hơn là dùng tản nhiệt nước. Tôi thì không khuyến nghị sử dụng tản nước lắm khi bạn chỉ có một GPU hoặc bạn có khoảng cách giữa nhiều GPUs. Tuy nhiên, tản nước sẽ đảm bảo rằng ngày cả GPU mạnh nhất cũng được mát trong thiết lập 4 GPU. Điều này là không thể nếu bạn chỉ dùng tản khí. Một lợi ích khác là tản nước sẽ êm ái hơn, ít tiếng ồn hơn đấy. Đây là một điểm cộng rất lớn nếu bạn chạy nhiều GPUs ở nơi mà có những người khác làm việc[25]. Tản nước sẽ làm tốn của bạn khoảng 100$ cho mỗi GPU và một vài chi phí ban đầu khác (khoảng 50$) Tản nước cũng cần vài nỗ lực lắp ráp máy tính, nhưng nói chung thì có khá nhiều hướng dẫn cụ thể mà bạn chỉ mất thêm vài giờ với nó mà thôi. Bảo trì cũng không quá phức tạp và phiền phức đâu.

Case lớn cho làm mát?

Tôi đã mua các case lớn dạng tower cho cụm DL của mình vì chúng có thêm quạt cho khu vực GPU, nhưng tôi thấy rằng chuyện này chẳng ảnh hưởng gì cả: chỉ giảm khoảng 2-5 độ C thôi, **không đáng cho sự đầu tư**. Phần quan trọng nhất của vấn đề tản nhiệt này là **tản nhiệt trực tiếp trên GPU** – đừng có tốn tiền cho các Case đắt đỏ chỉ để hỗ trợ tản nhiệt GPU. Hãy mua giá rẻ thôi. Cái case chỉ cần chứa đủ GPUs là đủ!

Kết luận cho vấn đề tản nhiệt

Tóm lại thì chỉ đơn giản thế này thôi: Với một GPU thì tản nhiệt khí, với thiết kế quạt thổi và chấp nhận mất một chút hiệu năng (10-15%), hoặc bạn trả thêm chút tiền cho tản nước mà nó khó hơn để lắp ráp đúng và bạn sẽ không mất chút hiệu năng nào cả. Tản khí và nước là lựa chọn phù hợp trong trường hợp này. Tôi thì khuyến nghị bạn nên dùng tản khí vì sự đơn giản của nó. Với multi GPU thì cũng lấy tản khí loại thổi hoặc nếu muốn dùng tản nước thì dùng loại all-in-one (AIO) nhé.

Bo mạch chủ (Motherboard)

Bo mạch chủ của bạn nên đủ số cổng PCIe để hỗ trợ số lượng GPUs mà bạn muốn (thường là 4, ngay cả khi có nhiều số PCIe hơn); **nhớ rằng hầu hết GPU có bề rộng bằng hai PCIe slot nên bạn nhớ**

[24] ND:Throttling: hiện tượng tự giảm xung nhịp, nhiệt độ khi đạt ngưỡng nhiệt độ tối đa để tự bảo vệ thiết bị

[25] *Nếu như bạn sử dụng hệ thống tại nhà với tản khí thì nếu có thể hãy để hệ thống tránh xa phòng ngủ nhé.* Tiếng ồn quạt khi chạy thuật toán là rất khủng khiếp, đặc biệt là trong đêm.

mua bo mạch chủ có đủ khoảng cách giữa các PCIe slots nếu bạn muốn dùng multi-GPU. Đảm bảo rằng bo mạch chủ không chỉ có PCIe slot mà còn hỗ trợ thiết lập GPU mà bạn muốn chạy. Bạn có thể tìm thấy bo mạch chủ trên newegg hoặc phần PCIe trên trang thông số kỹ thuật.

Màn hình

Lúc đầu thì tôi nghĩ thật ngớ ngẩn nếu viết về màn hình, nhưng chúng tạo ra sự khác biệt lớn và quan trọng đến mức tôi phải viết vài dòng về chúng.

Số tiền mà tôi đã chi cho 3 màn 27 inch có lẽ là số tiền tốt nhất tôi từng chi. Năng suất làm việc tăng rất nhiều khi dùng đa màn hình. Tôi đã cảm thấy tê liệt nếu tôi phải làm việc với một màn duy nhất. Đừng thay đổi bản thân về vấn đề này. Điều gì là tốt nếu như bạn không thể vận hành một hệ thống DL nhanh một cách hiệu quả?

Vài lời về việc lắp ráp PC

Vài người sợ hãi việc ráp máy tính. Các thành phần phần cứng thì đắt đỏ và bạn không muốn điều gì sai xảy ra. Nhưng thực tế đơn giản là cái gì không đi được với nhau sẽ không ráp được với nhau, vậy thôi. Bản hướng dẫn của bo mạch chủ thường khá cụ thể về việc lắp các thứ vào với nhau. Và cũng có đầy các hướng dẫn hay video chi tiết từng bước một có thể hướng dẫn bạn qua quá trình này nếu như bạn không có kinh nghiệm gì cả.

Điều tuyệt nhất về việc lắp ráp máy tính là bạn biết mọi thứ cần thiết. Và sau khi bạn làm xong một lần, bởi vì các máy tính được lắp ráp khá giống nhau, nó trở thành một kỹ năng sống mà bạn có thể sử dụng lại nhiều lần. Chả có lý do gì để do dự cả!

Kết luận

Cấu hình khuyến nghị

- GPU: RTX 2070 hoặc RTX 20810 Ti, GTX 1070, GTX 1080, GTX 1070 Ti và GTX 1080 Ti từ eBay.
- CPU: 1-2 nhân cho mỗi GPU phụ thuộc vào cách bạn xử lý dữ liệu. > 2GHz. CPU nên hỗ trợ số GPU mà bạn chạy. PCIe lanes không liên quan
- RAM
    - Xung không quan trọng, mua RAM rẻ nhất
    - Dung lượng ít nhất là bằng với RAM của GPU lớn nhất
    - Mua nhiều RAM hơn nếu có thể
    - Nhiều RAM sẽ hữu ích nếu làm việc với dữ liệu lớn (ND: large data set – not big data)
- Har drive/ SSD
    - Hard Drive cho dữ liệu (>= 3TB)
    - SSD cho sự tiện dụng và xử lý data nhỏ hơn.
- PSU
    - Cộng số watts GPU và CPU sau đó tính thêm 10% cho số điện thế cần dùng
    - Lấy PSU có chỉ số hiệu quả cao hơn nếu dùng nhieuf GPU
    - Chú ý đảm bảm PSU có đủ có đủ số đầu nối PCIe (6-8 pin)
- Tản nhiệt
    - CPU: lấy một cái tản tiêu chuẩn hoặc tản nước AIO
    - GPU:
        - Dùng tản khí
        - Dùng GPUs có tản khí dạng thổi nếu dùng multiple GPUs
        - Thiết lập coolbit trong Xorg để kiểm soát tốc độ quạt
- Bo mạch chủ

    - Lấy nhiều PCIe slot cho số GPU tương lai cần thiết (một GPU lấy 2 slot); max 4 GPU cho một hệ thống
- Màn hình
    - Một cái màn thêm vào sẽ khiến bạn có năng suất cao hơn một cái GPU thêm vào đấy

Update 2018-12-14: Làm lại toàn bộ blog với các khuyến nghị được cập nhật

Update 2015-04-22: Xóa khuyến nghị cho GTX 580

Bình luận của chúng tôi:

Dưới đây là một số cách build của một số tác giả khác và các thiết lập của họ, chú ý rằng các bài viết dưới đây không được cập nhật với các phần cứng mới nhất, tuy nhiên chúng vẫn có giá trị tham khảo tốt nếu như chúng ta hiểu và biết cách so sánh các thành phần phần cứng. Một chú ý khác là một vài tác giả build hệ thống hướng tới DL trong khi một vài người khác build hệ thống để làm dữ liệu nói chung.

- https://medium.com/yanda/building-your-own-deep-learning-dream-machine-4f02ccdb0460
- https://blog.slavv.com/the-1700-great-deep-learning-box-assembly-setup-and-benchmarks-148c5ebe6415
- https://www.oreilly.com/learning/build-a-super-fast-deep-learning-machine-for-under-1000
- https://medium.com/@nicksharvey/a-powerful-affordable-machine-learning-rig-for-2k-c96ce4bf16b8
- https://medium.com/datadriveninvestor/amd-ryzen-based-deep-learning-server-build-8fcd8f2139d7
- https://medium.com/the-mission/how-to-build-the-perfect-deep-learning-computer-and-save-thousands-of-dollars-9ec3b2eb4ce2
- https://medium.com/data-design/interview-amd-ryzen-as-a-workstation-4d409eec25e2

Dưới đây là cấu hình của chúng tôi, lắp ráp vào tháng 05/2018 với mục đích làm các dự án Data Science nói chung (rất ít Computer Vision)

- CPU: Ryzen 2700x (xem xét update lên Ryzen 3000)
- GPU: Asus GTX 1080 (xem xét update lên như tác giả bài viết khuyến nghị). Asus có các mẫu tản nhiệt khí dạng thổi.
- Motherboard: Asus Z470 hỗ trợ CPU và GPU
- PSU: Corsair enthusiast series RM 750x 80 plus gold 750 watt, eu version
- Ram: 32gb
- HDD: 2TB + SDD 500 GB