DEEP TENSOR NEURAL NETWORKS

<https://github.com/atomistic-machine-learning/dtnn>

(ví dụ mô hình DTNN)

<https://github.com/khumairraj/Neural-Tensor-Network/blob/master/Neural%20Tensor%20Layer.ipynb> (code ví dụ training model)

<https://nlp.stanford.edu/~socherr/SocherChenManningNg_NIPS2013.pdf> (tensor network của stanford)

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6848846/> (nhận dạng 2 loại táo qua quang phổ)

PHÂN TÍCH VÍ DỤ NHẬN DẠNG HAI LOẠI TÁO QUA QUANG PHỔ

Quang phổ phản chiếu khoảng 450 nm đến 1000 nm

* Để cải thiện model này,2 phương pháp successive projection algorithm (SPA) và x-loading weights (x-LW) cho hiệu quả cao nhất đối với bước sóng.
* PLS: 91, 31 đến 96, 41 %
* RBNN (radial basic neural networks): 97.8 %
* MLP: 99.9 %

Giới thiệu:

* Ở Trung Quốc có hai loại táo phổ biến là “Fuji” và “Qinguan”
* “Fuji” chiếm 70% diện tích trồng táo trên toàn Trung Quốc
* “Fuji thì giòn hơn “Qinguan”

Mục đích của bài làm:

* Xác định bước sóng, từ đó xác định độ giòn của táo

Trong nghiên cứu này, chúng ta sẽ phát triển sâu hơn về Deep neural Network (DNN). Gọi là deep tensor neural network (DTNN)

* DTNN tăng độ thuật lợi bằng cách lặp lại một hay nhiều layers với double-projection (DP) layer, mỗi vector ngõ vào được chiếu thành 2 không gian con không tuyến tính và tensor layer. 2 subspaces đó tác động qua lại lẫn nhau và cùng nhau dự đoán được layer tiếp theo trong deep architecture. Thêm vào đó chúng ta còn mô tả phương pháp lập ra tensor layers thanh conventional sigmoid layers. Vì vậy những cái trước đó có thể được treated và trained bằng cách giống nhau sau cùng. Với cách sắp xếp này ta có thể xem DTNN như là DNN gia tăng thêm DP layers vì vậy nó không chỉ là thuật toán BP learning của DTNNs có thể thu nhận được mà còn là DTNN mới có thể dể dàng phát triển. Ước lượng trẹn Switchboard chỉ ra rằng DTNN có thể perform cao hơn DNN khoảng 4-5%.
* DNN link:

<https://nordiccoder.com/blog/deep-neural-network/>

<https://towardsdatascience.com/the-basics-of-deep-neural-networks-4dc39bff2c96>

<https://github.com/bsplku/dnnwsp/blob/master/Tensorflow_code/.ipynb_checkpoints/ch02_dnnwsp_hsp_tensorflow-checkpoint.ipynb> (dnn model tensorflow code)

<https://github.com/kunalghosh/Deep-Learning-Spectroscopy/blob/master/deep_tensor/deeptensor.py> (deep tensor dùng theano)

DNN là một artificial neural network (ANN) với nhiều layers giữa input và output layers. DNN tìm ra sự vận dụng toán học chính xác để turn từ input into output, có thể là mối quan hệ tuyến tính hoặc không tuyến tính. Network di chuyển qua các layers để tính toán xác xuất mỗi output. Ví dụ DNN train để nhận ra tiếng giống chó. Người dung có thể xem lại và chọn xác xuất nên hiển thị và trả về label đề xuất. Mỗi sự vận dụng tính toán như là xem xét layer, DNN phức tạp có thể có nhiều layers.

* DNN có thể mô hình hóa những mối quan hệ không tuyến tính. Cấu trúc DNN cấu thành những models, nơi mà những đối tượng có thể được xem là nguyên thủy.
* Cấu trúc Deep bao gồm nhiều phương pháp cơ bản. Mỗi cấu trúc có thể tìm thành công những domains cụ thể.
* DNN là một loại feedforward networks với luồng dữ liệu từ ngõ ra cho đến ngõ vào mà không looping về.
* Đầu tiên DNN tạo ra một map với neural ảo và sau đó ấn định những con số ngẫu nhiên hoặc “weights” để kết nối chúng với nhau
* Weight và input là multiples và trả về ngõ ra giữa 0 và 1.
* Nếu network không nhận ra được mẫu cụ thể nào, thuật toán sẽ điều chỉnh lại weights
* Bắng cách đó thuật toán có thể làm cho những con parameters cố định trở nên influential. Cho đến khi xác định chính xác phương pháp toán có thể hoàn thành quá trình

DTNN:

* Mở rộng DNN với một phương pháp mới lạ là DTNN với một hay nhiều double-projection (DP) và tensor layers

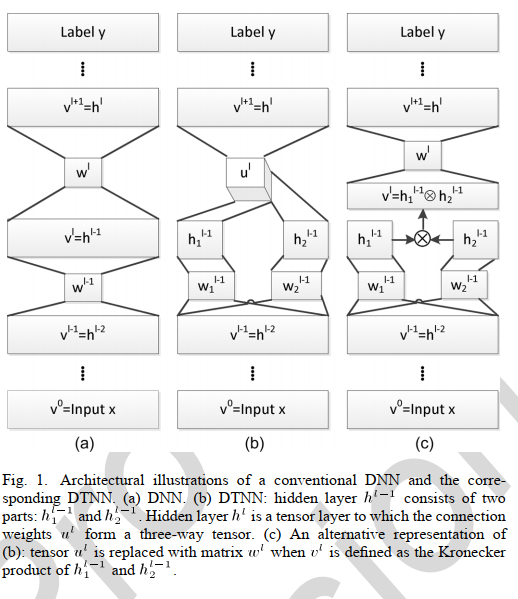
1. Giới thiệu:

* Trong những năm gần đây, chúng ta phát triển matrix thành những tensor với mục đích là tạo ra model 3-way interactions và cải thiện sức mạnh modeling của neura networks.
* Tensor-based bị giới hạn bởi Boltzmann machine (RBM) yêu cầu model 3-way interactions trên từng pixels với model pixels kỳ vọng và phương sai.
* RBM khác biệt ở hai đặc điểm

Đầu tiên là pure generative model, mặc dù nó có thể bị phân biệt bởi fine-tuned. Trong khi DTNN phân biệt model trong tự nhiên

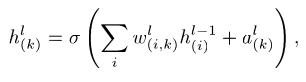
Thứ hai là RBM chỉ có một hidden layer và cơ cấu trên cấu trúc thiết kế của nó mà không được cho phép sử dụng tensor weights nhiều hơn 1 layer. Ngược lại DTNN thì được thiết kế với mục đích là nhúng các tensor weights tự nhiên vào trong nhiều stached hidden layers.

* RBM sau đó được mở rộng thành RNN
* Trong thực tế là bất kỳ sigmoid layer nào in conventional DNN co thể bị thay thề bởi tensor layer.

1. General architecture of DTNN:

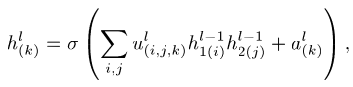
So sánh với DNN:

* Conventional DNN: input denote là x, với I x 1 vector, output là y với C x 1 vector. l là layer index. Mỗi hidden layer hl-1 kết nối với layer tiếp theo h1 với weight matrix wl và bias al

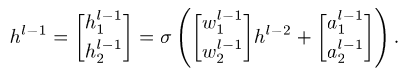


i, k là những indexes trong hidden units của hl-1 và hl

* DTNN với hidden layer được chia ra thành h1l-1 và h2l-1. Những phần đó kết nối với h1 thông qua 3-way tensor ul

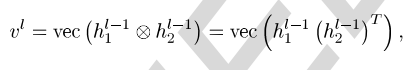


* Chúng ta gọi hidden layer hl-1 là double projection (DP) với thông tin nhận được từ layer trước đó là hl-2 chiếu lên 2 subspaces là h1l-1 và h2l-1



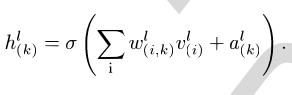
* Hidden layer hhhhl gọi là tensor layer. DP layer trước đó là hl-1 liên kết với hl thông qua weight tensor là ul

Ở hình 1c, là một cách nhìn khác về DTNN, giống như hình b, bằng cách xác định vl, input to the layer l



Ở đó ⊗ là tích của Knonecker, và vec(.) là column vectorized representation của ma trận.

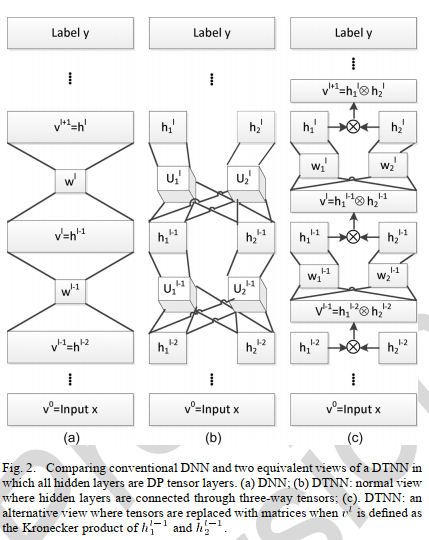
Chúng ta có thể tổ chức là viết lại tensor ul như là wl giống như hình chữ nhật trong hình 1c



Bằng cách biết lại trên cho phép ta giảm và đổi tensor layer thành conventional matrix layers bằng cách định nghĩa cùng loại gigo diện để mô tả hai loại layers.

Ví dụ: trong hình 1c ta có thể thấy hidden layer hl có thể được xem xét như là một conventional layer như trong hình 1a, và có thể học được bằng cách sử dụng thuật toán conventional backpropagation (BP).

Cánh viết trên còn có thể chỉ ra rằng tensor layer có thể được xem như là conventional layer mà ngõ vào gồm có cross product của những giá trị đi qua layer trước đó.



* Hidden layer là hl-1 tuy nhiên DP layer vẫn còn chứa 2 phần là h1l-1 và h2l-1, mà làn lượt phải xác định 2 weight là w1l-1 và w2l-1 bằng cách như nhau như trong hình 1b và 1c matrices
* DTNN ở hình một chỉ chứa một DP layer. Tuy nhiên không thể ngăn layer khác từ DP layer.
* Hình 2b là một ví dụ về DTNN mà ở đó tất cả các hidden layers là DP layers. Ví dụ hidden layer hl-2 chia ra 2 thành phần h1l-2 và h2l-2 được kết nối với h1l-1 và h2l-1 trong qua tensor u1l-1 và u2l-1.

1. Detailed component