Turning Process

tính đến hiện tại thì có khá nhiều tham số có thể ảnh hưởng tới model

alpha – learning rate: trọng số quan trọng nhất

beta1 – momentum: trọng số “gia tốc”, “lực”

weight decay, learning rate decay, …

thì việc chọn hyper parameters giờ cũng là một vấn đề lớn

Calendar, scatter chart

Description automatically generated with medium confidence

phương pháp chọn tại sao lại nên là random?

vì nếu là một vòng lặp for thay đổi 1 giữ tất cả còn lại

thì đến cuối cùng một tham số thử đc rất ít các giá trị (vì đã bị giữ lại)

với random thì sẽ thử đc rất nhiều giá trị của toàn bộ tham số (tỉ lệ có ny sẽ cao hơn)

Chart, scatter chart

Description automatically generated

trong quá trình tìm có thể chơi kiểu

nếu như tìm đc 1 vài “mối ngon” thì sẽ tập chung tìm xung quanh mối đó

lặp lại chắc chắn sẽ tìm đc ny (vì đã rất gần với ny rồi)

Chart

Description automatically generated with low confidence

cách pick learning rate nằm trong một khoảng dị vd: 0.000001 -> 1

thay vì bò vs khoảng cách cố định

bỏ bằng hàm mũ 10 ^ random (-4,0) thì vừa cover đc toàn bộ range, vừa dễ triển khai

Diagram

Description automatically generated with medium confidence

dị hơn là range từ 0.9 -> 0.999999 thì chỉ cần lấy 1 trừ đi rồi làm như hình trước

chú ý tham số Beta này rất nhạy cảm khi càng gần 1 như trên hình

thay đổi một tí có thể loop qua cả nhìn example mới :V

A picture containing diagram

Description automatically generated

một vài chiến thuật train model

panda: tập chung vào tuning 1 model tại 1 thời điểm (máy yếu)

caviar: “đẻ” nhiều model tương ứng mỗi tuning (máy khỏe)

Normalizing activations in a network

thì là bởi vì việc nomalize activation, kể cả input (cũng tính là 1 activation)

có thể khiến việc train nhanh hơn (các số k to quá, nhỏ quá)

đồng thời việc kiểm soát cũng dễ hơn (có visual dễ nhìn)

Diagram

Description automatically generated

idea là norm đc vs các layer thì càng tốt một trong những cách đó là batch norm

bàng việc norm vs Z = W.A + b, (tranh cãi norm A hay Z)

giải thích rõ hơn về việc norm có thể làm nhanh, tốt model hơn là vì

activate function thường là sigmoid, tanh là các hàm mà khi giá trị càng lớn, (càng nhỏ) thì biến thiên càng ít

norm sẽ “kéo” range giá trị về trong khoảng VD: -0.5 -> 0.5 làm tăng tốc có thể tránh “bò” trên đồng bằng

ngoài ra norm sẽ làm input feature về cùng range, cái này giúp gradient decent

vì nếu một thuộc tính range từ 0 -> 100 có thể sẽ “choáng” hết phần grad của 0 -> 1

A picture containing text, whiteboard

Description automatically generated

giờ sẽ lại có thêm các biến beta và gama, có thể là hyper nếu tuning

cái IF của ông andrew nếu xảy ra thì giá trị của norm chả khác gì trước khi norm cả :V

Diagram, schematic

Description automatically generated

triển khai vs mini batch

A picture containing text, person

Description automatically generated

? mỗi mini batch sẽ có một tham số norm riêng không liên quan gì tới nhau cả

và một chú ý nhỏ là việc thêm vào norm thì tham số beta sẽ không cần nữa

vì ở cùng một lớp thì b giống nhau cùng cộng b rồi lại trừ đi mean

do vậy nên tham số b có thể bỏ qua và tập trung vào Beta đc đóng khung đỏ

Text, letter

Description automatically generated

? triển khai vô gradient decent

Why does batch norm work

vấn đề nếu không dùng norm chính là việc dữ liệu đầu vào nằm ngoài dự đoán

chính dự đoán rượu, mình train k norm, thì nếu như ng dùng nhập vào 1 tỉ

thì nếu may mắn thì ăn còn nếu k ăn mà 1 tỉ đó là b thường thì thuật toán trở thành sai

dẫn tới việc phải train lại, hoặc tệ hơn là đập đi xây lại thuật toán mới

Diagram

Description automatically generated

cụ thể hơn

Diagram

Description automatically generated

tưởng tượng nếu k có norm thì range giá trị của các activation thay đổi liên tục

do input thay đổi thì các activation layer trước đổi liên xoành xoạch thì lan ra toàn bộ mạng

làm mất tình ổn đinh của mạng cái này chắc con mẹ nó chắn là làm khó gradecent

do vậy norm đc thêm vào khiến cho dù range như nào thì cũng sẽ đc kéo về

cái này vừa làm ổn định mạng, vừa k làm mất đặc tính của mỗi thằng

vừa làm loosely coupling giữa các layer với nhau

nhưng nó cũng có một vài note nhỏ

vì áp dụng vào batch bản thân nó đã có nhiễu nhẹ

? nên việc scale lại này cũng có thể tăng nhiễu

giải thích cụ thể của ông Andrew ở hình dưới

Graphical user interface, text, application

Description automatically generated

? đoạn này không hiểu lắm

Diagram, text, letter

Description automatically generated

? không chắc lắm cơ mà test time thì chỉ có 1 example 1 thời điểm nên tính norm trở nên vô nghĩa

do vậy nên có thể dùng norm của tập train hoặc của tập nào đó ?

Softmax Regression: hiểu sơ sơ là thuật toán phân loại có nhiều class (VD: phân loại chó, mèo, gà, …)

phía dưới là hình minh họa

bằnh cách thêm một layer trước khi output để coi như việc predict vào các nhãn

cũng là một layer trong mạng neuron

? do việc dùng activation ở các layer trước đó có thể làm giá trị của class (output)

sẽ lệch nhau (chó range 100, gà range 10) thì kiểu bị auto là chó

do vậy cần một layer normalize lại những output này -> softmax layer before output

Diagram

Description automatically generated with low confidence

fun fact nho nhỏ, trong lúc train winepredict

mình dùng max của toàn bộ để lấy ra nhãn yêu cầu, đây chính là “hard max”

còn hiện tại là softmax tức các class khác vẫn còn giá trị, not = 0 như trong hard max

Diagram

Description automatically generated

hình trên giải thích rõ hơn về softmax cũng như so sánh vs hard max

một chú ý nhỏ là nếu số lượng nhãn bằng 2 thì tương đương logistic vì output có 2 giá trị mà tổng chúng = 1

Loss function

A white board with writing

Description automatically generated with low confidence

hàm loss function có vẻ mình đã implement đúng

output mong muốn sẽ dạng là 1 với nhãn đúng, 0 với các nhãn không mon muốn

và kết quả nếu dùng hàm loss đã biết từ trước thì sẽ chuẩn luôn :V (VD: muốn làm y^2 lớn)

như trên hình của ông Andrew