Zhang - Recurrent Neural Nets

Prev =

MLP = tabular data
CNN = vector/matrix data

but many tasks require sequential data

such as image caps, speech, time series

RNNs are deep learning models that capture the dynamics of sequences via recurrent connections

recurrent RNNs are unrolled accross time steps

9.1 working w/ sequences

Some datasets consist of a single long seq

we might randomly subsample the dataset into some pre-determined length

often data arrives as collection of seqs

Previously, we assumed w/ individual inputs we sampled from the sample under-lying distribution independently

$P(x)$

with sequences, we still assume they are indep but cannot assume that data @ each time step is indep of each other

e.g. words @ end of doc are heavily dependant on words @ the start

9.2 Converting Raw text into Sequence data

tokenisation

Skipped as not relevant for task

9.3 language models

9.4 Recurrent Neural Networks

we want to predict the next point using many previous points

but modelling $P(x_t \mid x_{t-1}, \ldots, x_{t-n+1})$ soon becomes infeasible

So it is better to use a latent variable model

$P(x_t \mid h_{t-1})$

h_{t-1} is a hidden state that stores the sequence of info upto step $t-1$

@ any time step h_t can be comped using the input x_t & hidden state h_{t-1}

$$h_t = f(x_t, h_{t-1})$$

hidden states are tennicaly inputs

RNNs are NNs w/ hidden states

$$H_t = \phi(X_t W_{xh} + H_{t-1} W_{hh} + b_h)$$

the calc of the hidden layer output of the current time step is determined by the input of the current time step + the hidden layer output of the prev time ~~step~~

hidden layer output is called hidden state

Params of RNN include
- weight of current input
- weights of hidden state

+ bias of current input

At all time steps of sequence, RNNs use the same model params

Params cost of RNN does not grow w/ number of time steps

At any time step, the computation of the new hidden state can be seen as a concat of the input x_t & the hidden state $t-1$

this concat feeds into a fully con layer

the output of the fully connected layer is H_t

H_t is later used to calc H_{t+1}

9.5 RNN from stratch

No notes

9.6 RNN concise implementation

No notes

9.7. Backprop through time

Applying backprop in RNN is called through time

Exploding gradients comes from backprop accroso long sequences

BP through time reqs unrolling the comp graph of a RNN one step @ a time

An unrolled ~~the~~ RNN is essentially a feedforward network

↳ w/ a special property that the same parameters repeat throughout the network

Chain rule can then be applied to the unrolled network to Backprop

Complications arise because sequences can be rather long

1. Computational memory issue

2. Optimization & numerical instability