Why RNN's are needed?

09 October 2024 15:53

Sequential data

ANN → tabular data CNN → image data

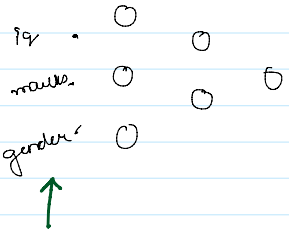
RNN → Recurrent Neural Network

↳ type of sequential model to work on sequential data

Non-sequential data

Eg: Suppose we have data

ig | marks | gender | placement



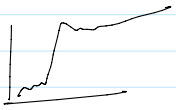
↑ Sequence of input doesn't matter

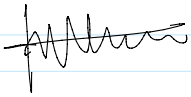
Sequential data

① Text

Hi my name is Sakalya

→ sequence matters for grammatical correctness

② Time Series data 

③ Speech 

④ DNA sequence

- RNNs are mostly used in NLP.

The Why of RNN?

Input	Output
Hi my name is Sakalya	0
I love Campusx	0
India won the match	1

↓ vectorize (OHE)

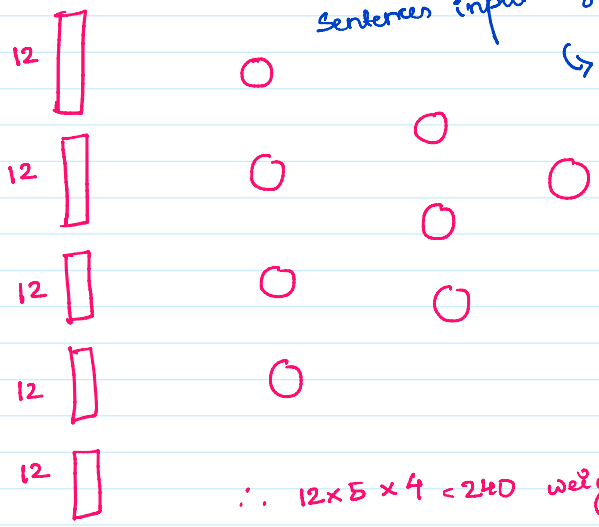
↓ unique words

[1,0,0...] Hi [0,1,...0] my

But now for the next sentences input size is 36848

↳ Input size varies

↓ ANN cannot handle varying input size

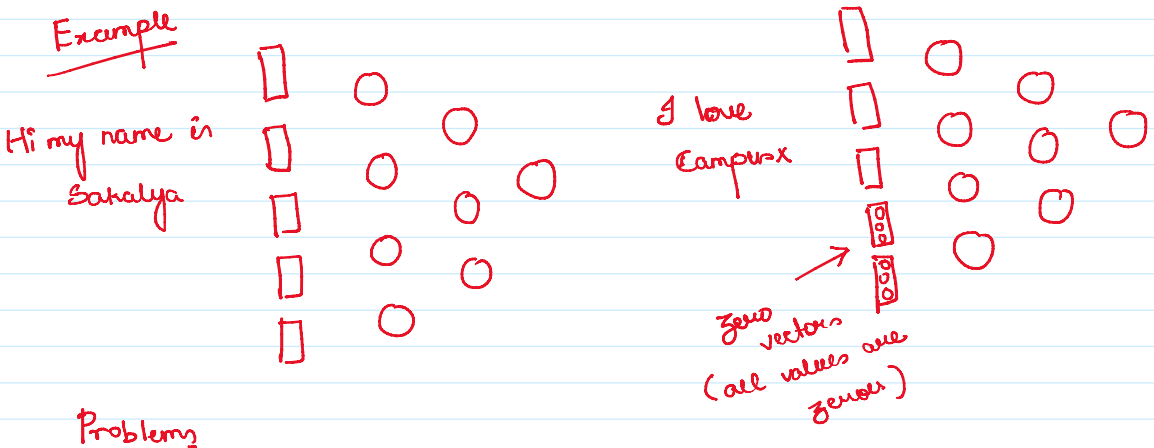


∴ 12 × 5 × 4 = 240 weights

- Textual data is of varying size

One workaround possible → zero padding → Select the sentence having max words
→ For smaller sentences use zero vectors

Example



Problems

- this is not a good solution
- leads to lot of unnecessary computation
- Prediction problem → if predicting sentence has 200 words
- completely discarding semantic meaning → ANN has no memory to remember sequence

Forward Propagation in RNN

09 October 2024 16:21

Data for RNN

(timesteps, input_features)

t=1

review-1

[10000][01000][00100]

(3,5)

no. of features

no. of timesteps

	Review	Sentiment
(3,5)	movie was good	1
(2,5)	movie was bad	0
(4,5)	movie was not good	0

vocab → 6 words

movie was good bad

[10000][01000][00100][00010]

[00001]

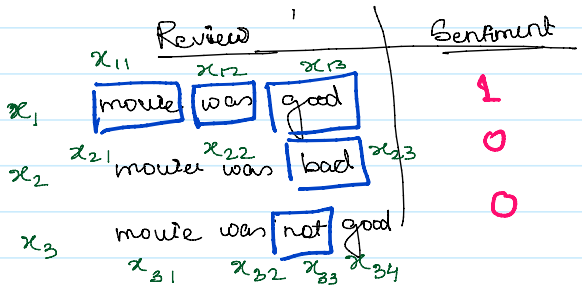
not

Keras → SimpleRNN → batchsize, timesteps, input_features

(3, 4, 5)

↳ longest timestep is taken

How RNN works?

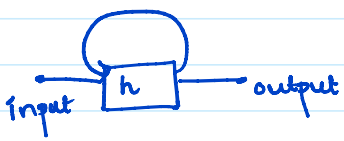


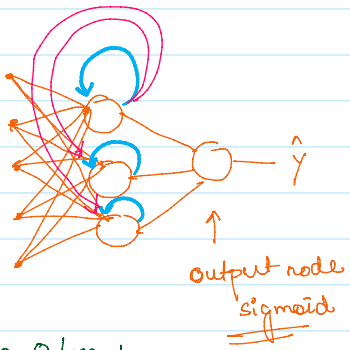
movie was good bad

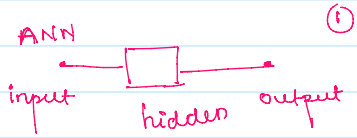
[10000][01000][00100][00010]

[00001]

not

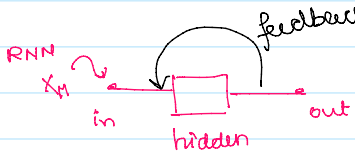






① whole input at a time

② feed forward



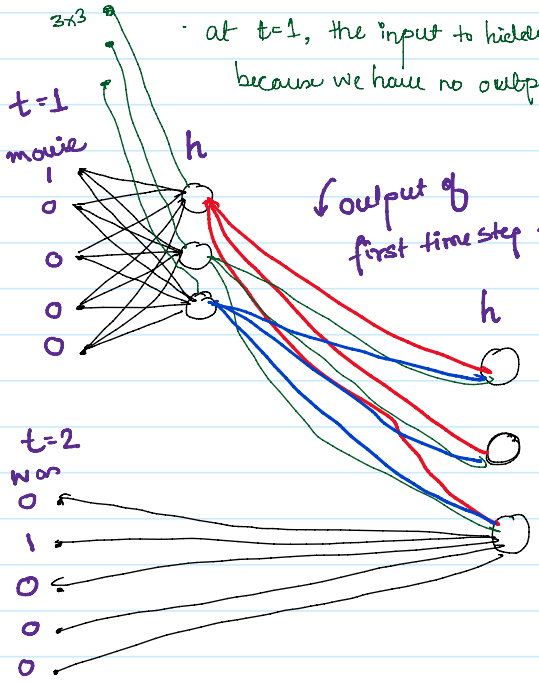
① time based input

- full input is not given at a time
- we input through time step.

② state concept sends feedback

3×3

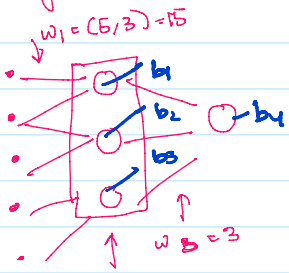
- at t=1, the input to hidden layer is 0/random because we have no output till now



- at t=1, the input to hidden layer is 0/random because we have no output till now

output of first time step fed as input to same hidden layer

Weights and biases



w_2 $3 \times 3 = 9$ ↑ self connection weights

∴ 27 weights + 4 biases

trainable params

Code example

```
from keras import Sequential
from keras.layers import Dense,SimpleRNN

model = Sequential()

model.add(SimpleRNN(3,input_shape=(4,5)))
model.add(Dense(1,activation='sigmoid'))

model.summary()
```

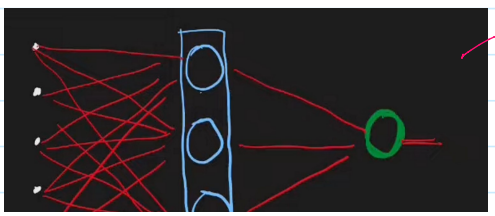
3 hidden layer RNN
max 4 time steps
5 input features
final dense layer for output

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 simple_rnn (SimpleRNN)      (None, 3)                 27

 dense (Dense)               (None, 1)                 4

=================================================================
Total params: 31
Trainable params: 31
Non-trainable params: 0
_________________________________________________________________
```

trainable params are 31 as we calculated (27+4)

Forward Propagation

review	Sentiment
x_{11} x_{12} x_{13}	1
x_{21} x_{22} x_{23}	0
	0



forward propagation
↓
unfolding through time
↓
works as a loop

$X_{11}\ X_{12}\ X_{13}$ | 1

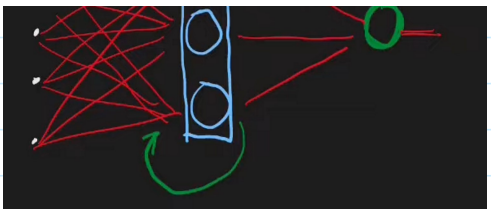
$X_{21}\ X_{22}\ X_{23}$ 0

$X_{31}\ X_{32}\ X_{33}$ 0

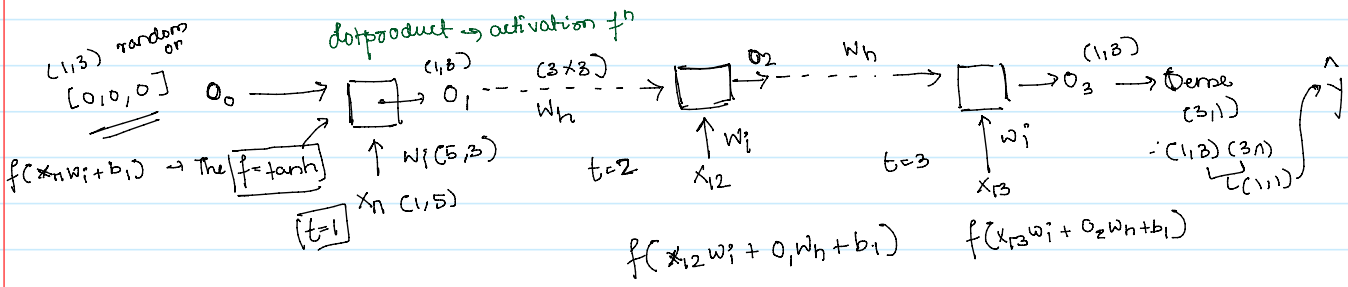
X_{34} |

vectors

5-dimension



works as a loop



X_{12} W_i O_1 W_h

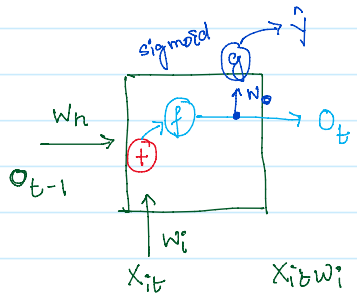
(1×5) (5,3) (1×3) (3,3)

(1,3) + (1,3) → (1,3)

Advantages

1. Parameter / Weight sharing → same weights are used for every input
2. RNN can store sequential information upto 10 time steps

Simplified Representation



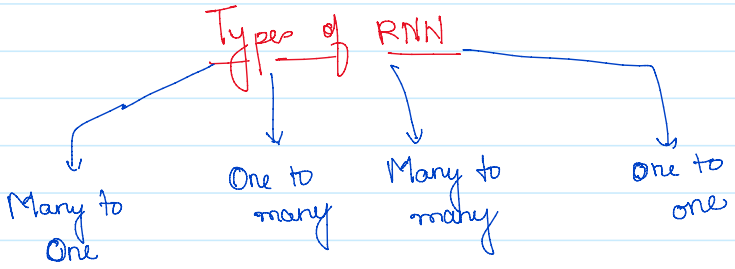
If last time step pass output to dense layer

i = row #

t = time step

Types of RNN

11 October 2024 11:33



Many to One

input → sequence

output → non-sequence → int/number → scalar (1,0)

example → → Sentiment analysis → (1,0)

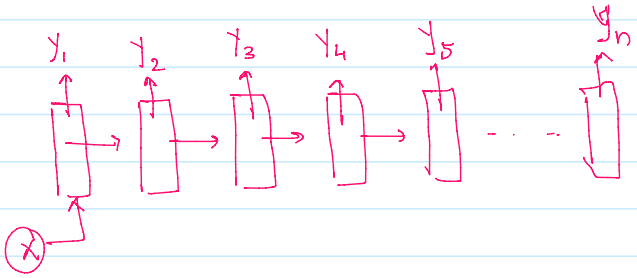
→ review prediction → 1 → 5

multiclass classification

One to many

input → non-sequential → image, number

output → sequence → words etc.



Ex: Image captioning
music generation

Many to many

input → seq

output → seq

seq2seq model

Input length == output length
same length

variable length

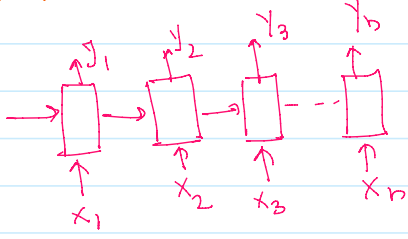
Ex: machine translation

POS tagging

my name is Sakalya

pronoun noun verb noun

NER

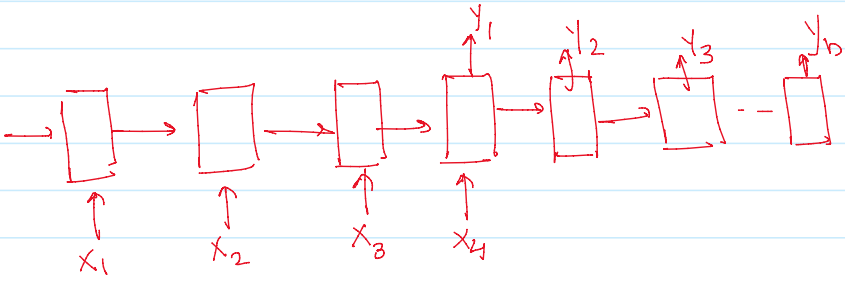


Ex: machine translation

Hi my name is Sakalya → 5

मेरा नाम साकल्य है → 4

Length cannot be determined



first study input and then get output

also called encoder-decoder architecture

One to one (Not typically a RNN because no sequence)

image classification

simple neural network

no feedback

Backpropagation in RNN

11 October 2024 11:50

It is called Backpropagation through time Step

BPTT

Many to One RNN

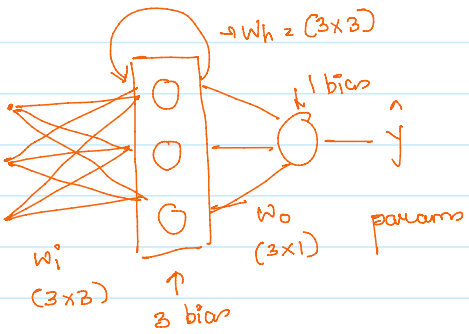
text → 1/0

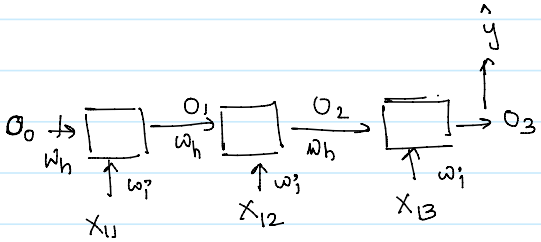
cat mat rat	1
rat mat mat	1
mat mat cat	0

	X			y
x_1	[1 0 0]	[0 1 0]	[0 0 1]	1
x_2	[0 0 1]	[0 1 0]	[0 1 0]	1
x_3	[0 1 0]	[0 1 0]	[1 0 0]	0

vocab = cat mat rat

[1 0 0] [0 1 0] [0 0 1]





$$O_1 = f(X_{11}W_i + O_0W_h)$$

$$O_2 = f(X_{12}W_i + O_1W_h)$$

$$O_3 = f(X_{13}W_i + O_2W_h)$$

$$\hat{y} = \sigma(O_3 \cdot W_o)$$

$$L = -y_i \log(\hat{y}_i) - (1-y_i)\log(1-\hat{y}_i)$$

minimize

Gradient descent

W_h W_o W_i

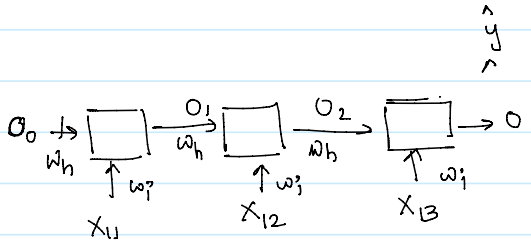
$$W_i = W_i - \eta \frac{\partial L}{\partial W_i} \qquad W_o = \eta - \frac{\partial L}{\partial W_o}$$

$$W_h = W_h - \eta \frac{\partial L}{\partial}$$

$w_h \quad w_o \quad w_i$

$w_i = w_i - \eta \left[\frac{\partial L}{\partial w_i}\right] \qquad w_o = \eta - \left[\frac{\partial L}{\partial w_o}\right]$

$w_h = w_h - \eta \left[\frac{\partial L}{\partial w_h}\right]$



$O_1 = f(x_{11} w_i + O_0 w_h)$

$O_2 = f(x_{12} w_i + O_1 w_h)$

$O_3 = f(x_{13} w_i + O_2 w_h)$

$\hat{y} = \sigma(O_3 \cdot w_o)$

$\frac{\partial L}{\partial w_o}$ — $L \to \hat{y}$ (depends) $\to O_3$, w_o (depends)

$$\frac{\partial L}{\partial w_o} = \frac{\partial L}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial w_o}$$

$L = -y_i \log(\hat{y}_i) - (1-y_i)\log(1-\hat{y}_i)$

$$\frac{\partial L}{\partial \hat{y}} = \frac{-y_i}{\hat{y}_i} - \frac{(1-y_i)}{(1-\hat{y}_i)} \times (-1) = \frac{-y_i}{\hat{y}_i} + \frac{(1-y_i)}{(1-\hat{y}_i)} = \frac{-y_i(1-\hat{y}_i) + (1-y_i)\hat{y}_i}{\hat{y}_i(1-\hat{y}_i)}$$

$$= \frac{-y_i + y_i\hat{y}_i + \hat{y}_i - y_i\hat{y}_i}{\hat{y}_i(1-\hat{y}_i)}$$

$\frac{\partial L}{\partial w_i}$

∴ 3 paths

$$\frac{\partial L}{\partial w_i} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial O_3} \frac{\partial O_3}{\partial w_i} + \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial O_3} \frac{\partial O_3}{\partial O_2} \frac{\partial O_2}{\partial w_i} + \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial O_3} \frac{\partial O_3}{\partial O_2} \frac{\partial O_2}{\partial O_1} \frac{\partial O_1}{\partial w_i}$$

$$\frac{\partial L}{\partial w_i} = \sum_{j=1}^{n} \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial O_j} \frac{\partial O_j}{\partial w_i}$$

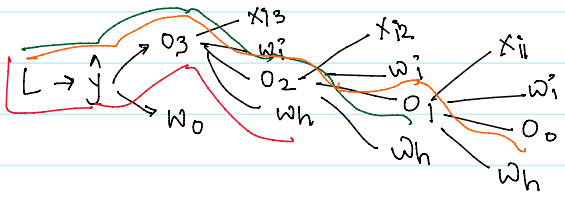
n = timesteps

$j=1 \quad = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial O_1} \frac{\partial O_1}{\partial w_i} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial O_3} \frac{\partial O_3}{\partial O_2} \frac{\partial O_2}{\partial O_1} \frac{\partial O_1}{\partial w_i}$

$j=2 \quad = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial O_2} \frac{\partial O_2}{\partial w_i} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial O_3} \frac{\partial O_3}{\partial O_2} \frac{\partial O_2}{\partial w_i}$ — ⊕ → $\frac{\partial L}{\partial w_i}$

$j=3 \quad = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial O_3} \frac{\partial O_3}{\partial w_i} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial O_3} \frac{\partial O_3}{\partial w_i}$

$\frac{\partial L}{\partial w_h}$



$$\frac{\partial L}{\partial w_h} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial o_3} \frac{\partial o_3}{\partial w_h} + \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial o_3} \frac{\partial o_3}{\partial o_2} \frac{\partial o_2}{\partial w_h} + \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial o_3} \frac{\partial o_3}{\partial o_2} \frac{\partial o_2}{\partial o_1} \frac{\partial o_1}{\partial w_h}$$

$$\frac{\partial L}{\partial w_h} = \sum_{j=1}^{n} \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial o_j} \frac{\partial o_j}{\partial w_h}$$

n = timesteps

Problems with RNN

15 October 2024 21:07

2 major problems: → lead to need for LSTMs

- Problem of Long Term dependencies
- Unstable Training

caused by → Unstable gradients

① Long term dependency problem

If time steps are very large, it forgets previous data.

Next word prediction

Marathi is spoken in Maharashtra → short term dependency

Maharashtra is a beautiful place. I went there last year, but I could not enjoy because I do not understand Marathi.

→ Long term dependency → RNN fails here

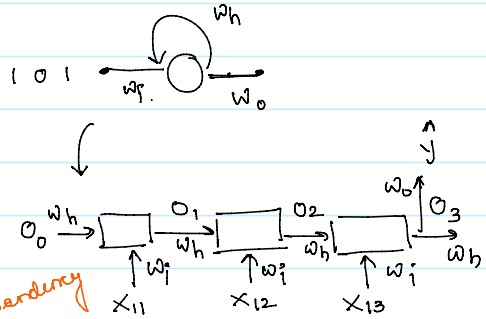
many time steps

↓

arises due to vanishing gradients

Example

Inp	Output
1 0 1	1
0 0 1	0
0 0 0	0
1 1 1	1



short term dependency

$$w_{in} = w_{in} - \eta \frac{\partial L}{\partial w_{in}}$$

$$w_0 = w_0 - \eta \frac{\partial L}{\partial w_0}$$

$$w_h = w_h - \eta \frac{\partial L}{\partial w_h}$$

$$\frac{\partial L}{\partial w_{in}} = \underbrace{\frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial O_3} \frac{\partial O_3}{\partial w_{in}}}_{①} + \underbrace{\frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial O_3} \frac{\partial O_3}{\partial O_2} \frac{\partial O_2}{\partial w_{in}}}_{②}$$

Four 3 time steps

$$+ \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial O_3} \frac{\partial O_3}{\partial O_2} \frac{\partial O_2}{\partial O_1}$$ ← long term dependency

For 3 time steps

$$\frac{\partial L}{\partial w_{in}} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial o_3} \frac{\partial o_3}{\partial o_2} \frac{\partial o_2}{\partial o_1} \frac{\partial o_1}{\partial w_{in}} + \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial o_3} \frac{\partial o_3}{\partial o_2} \frac{\partial o_2}{\partial o_1} \frac{\partial o_1}{\partial w_{in}}$$

← long term dependency

(3)

For 100 time steps

$$\frac{\partial L}{\partial w_{in}} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial o_{100}} \frac{\partial o_{100}}{\partial o_{99}} \cdots \frac{\partial o_1}{\partial w_{in}}$$

The long term contribution reduces

compressed

$$\frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial o_{100}} \prod_{t=2}^{100} \left(\frac{\partial o_t}{\partial o_{t-1}} \right) \frac{\partial o_1}{\partial w_{in}}$$

$$o_t = \tanh\left(x_{it} w_{in} + o_{t-1} w_h\right)$$

$$\frac{\partial o_t}{\partial o_{t-1}} = \tanh'\left(x_{it} w_{in} + o_{t-1} w_h\right) \cdot w_h$$

$[0,1]$

$$= \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial o_{100}} \prod_{t=2}^{100} \left(\tanh'(\,) \cdot w_h \right) \frac{\partial o_1}{\partial w_{in}}$$

$\tanh'(\,)$: 0-1, w_h: 0-1

When we multiply 100 or more numbers in the range 0-1 it is close to 0.

∴ Contribution by Long term dependency is almost negligible

(Vanishing gradient problem)

Solutions

- different activation functions (relu, leaky relu)
- better weight initialization
- Skip RNNs
- LSTMs

(2) Unstable Training

$$\frac{\partial o_t}{\partial o_{t-1}} = \tanh'\left(x_{it} w_{in} + o_{t-1} w_h\right) \cdot w_h$$

Suppose we use

Suppose we use ReLU → gradient increases → multiplication increases gradient → exploding gradient

Solutions

- gradient clipping – don't let gradient to increase beyond a value
- controlled learning rate
- LSTMs