Recurrent Neural Network

↳ repeating

MLP [vector] → MLP → y

CNN [image] → CNN → y

Both of them do not take advantage of sequences.

[This smartphone is fast & battery is good]

OR

[Battery good this is phone smart & is the]

} both are same for MLPs using word vectors

① words

② Time Series → Stock market

③ Translation

④ Audio data → is a sequence

⑤ Image → caption

→ just to make it easier to index & search.

TLDR; if output depends on sequences, we need some form of RNN

Left table an example of words data.

$x_i \longrightarrow$ Google will love me in 2021
x_1 x_2 x_3 x_4 x_5 x_6

If we have one hot encoding for 10k words,
each word vector is 10k dim.

So if we have 5 words, we 50k dim if
we preserve order.

↓

The input is variable anyway. how do
you handle that?

Conclusion → This doesn't end up working.

Large word corpuses & large sequences
will always be a challenge for RNNs.

Review Sentiment Analysis

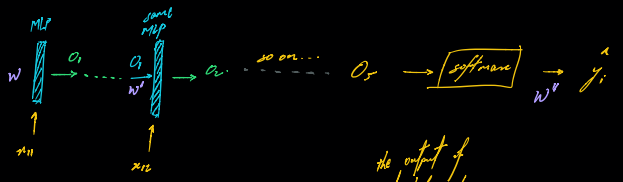
Dataset (D) = $\{x_i, y_i\}$
↳ binary
sentence

↓

will have
crazy words
obviously

Task words → {0, 1}

$x_1 \rightarrow (x_{11}, x_{12}, x_{13}, x_{14}, x_{15})$



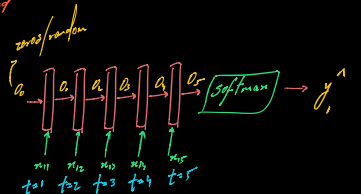
the output of word before it

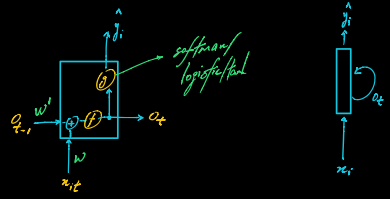
3 set of weights

We want O_2 to depend on x_{12} & O_1

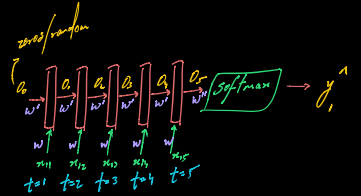
W → current input
W' → previous output
W'' → in the end

Each output depend on → current word + output of previous word.





\# Backprop over Time



$$O_1 = f(W x_{11} + W' O_0)$$
$$O_2 = f(W x_{12} + W' O_1)$$
$$\vdots$$
$$O_5 = f(W x_{15} + W' O_4)$$
$$\hat{y}_i = g(W'' \cdot O_5)$$

$$\frac{\partial L}{\partial W''} = \frac{\partial L}{\partial \hat{y}_i} \cdot \frac{\partial \hat{y}_i}{\partial W''}$$

$$\frac{\partial L}{\partial (W)} = \frac{\partial L}{\partial \hat{y}_i} \cdot \frac{\partial \hat{y}_i}{\partial O_5} \cdot \frac{\partial O_5}{\partial W}$$

W on the last time

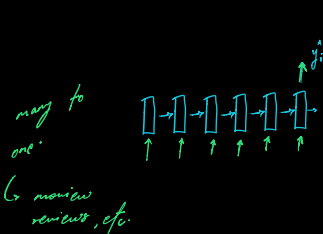
$$\frac{\partial L}{\partial W'} = \frac{\partial L}{\partial \hat{y}_i} \cdot \frac{\partial \hat{y}_i}{\partial O_5} \cdot \frac{\partial O_5}{\partial W'}$$

This backprop happens over time, on the same weights.

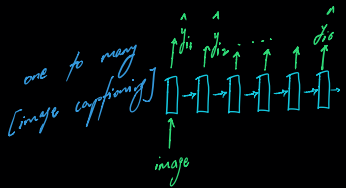
The longer the sequence, the longer the backprop product. ∴ Vanishing/Exploding gradient problem will persist.

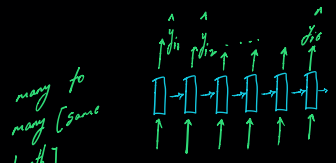
↓

LSTM / GRU avoid these problems through modifications to RNN

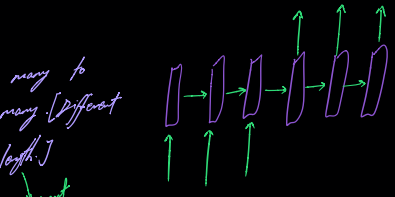


↳ movie reviews, etc.





part of speech detection → flag some particular word as verb or noun or a further flag



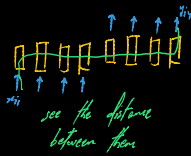
→ # Translation tasks

Simple RNNs suck

① Vanishing gradients ruin both forward & backward passes

↳ if first & last word are related to each other, RNN will not effectively detect that.

RNNs suck at long term dependencies in input sequences

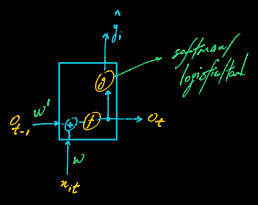


see the distance between them

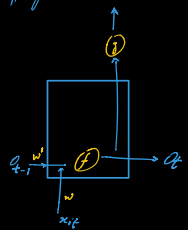
↳ we want the network to have complete input before outputting something

LSTM

KID



Legend



⇐ Vinnue if this is correct

$o_t = f([w', w], [x_{it}, o_{t-1}])$

concatenate (not add (like kids))
& dot product

Colah's blog is great for LSTMs

GRU

→ optimized
→ less params to train
→ faster training
→ Almost same performances as LSTMs. ↗ special cases apply