Chapter-06 RNN

6.1 Variable length sequence of words:-

processing of images often forces them into a specific input dimensions

- Not obvious how to do this with text
- For example: classifying tweets as positive, negative, neutral
  - Tweets can have variable words
  - What to do?

Ordering of words is important:

Want to do better than "bag of words" implementation

- Ideally, each word is processed or understood in the appropriate context
- words should be handled differently depending on "context".
- Also each word should update the Next Concept

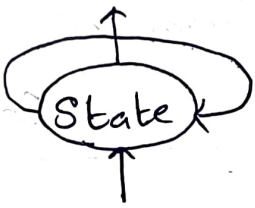
Idea: Use the Notion of Recurrence:-

- Input word by one by one.
- This way, we can handle variable lengths of text
- The response to a word depends on the words that preceeds it

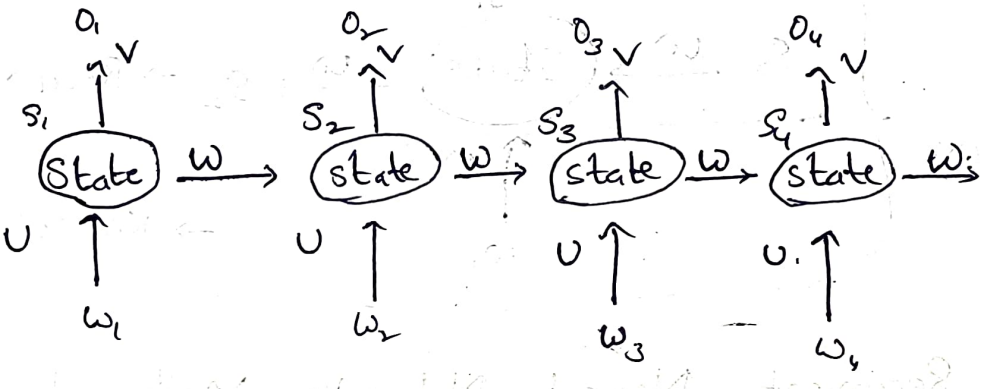
Network outputs of things:-

* Prediction: what would be the prediction if sequence ended with that word
* State: Summary of everything that happened in past

6.2 State and Recurrent Neural Networks



unrolling the RNN:-



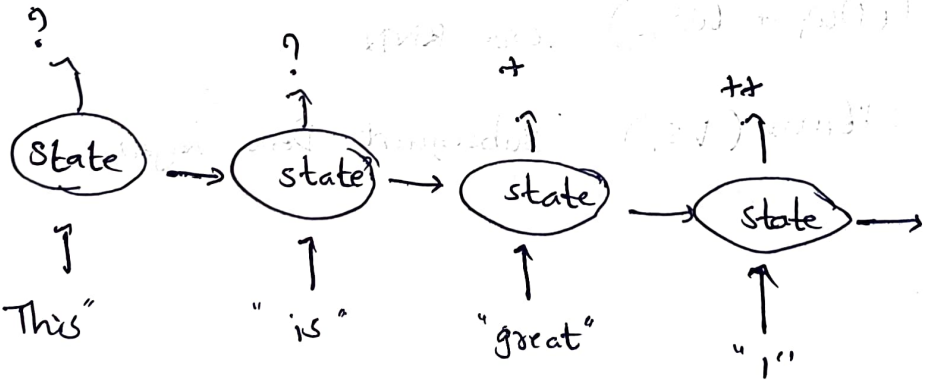
w_1 → inputs U → vector form of words

W → total info from state passing to the next state

$W = S_1 + (U \cdot w_1)$ O_1 → output prediction

V → Activation function

U → linear transformation ↳ vector of words.

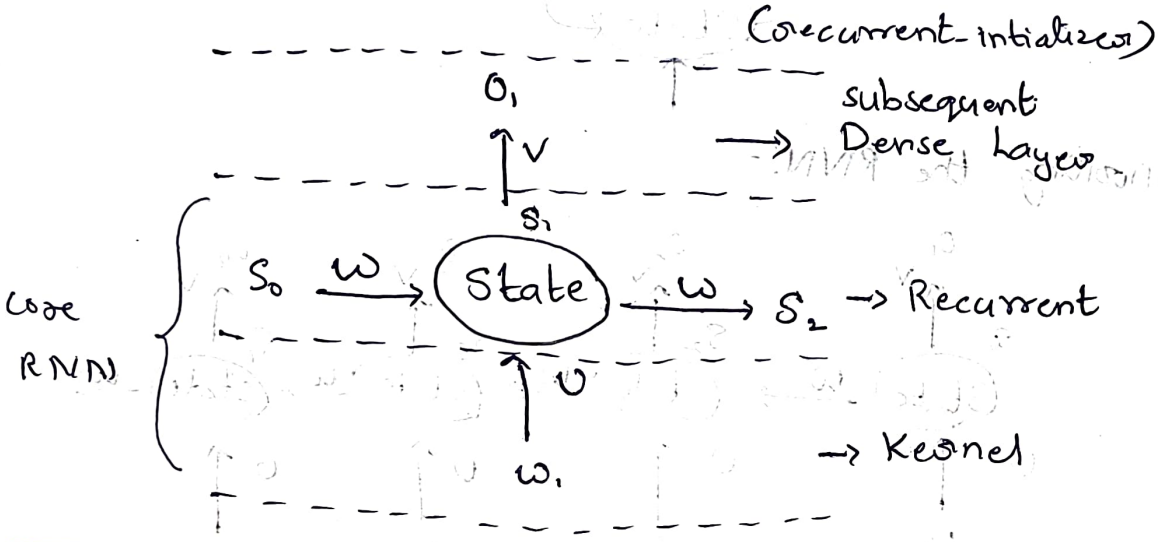


O_1V, O_2V, O_3V, O_4V → In keras this part is accomplished by Dense layer.

The remaining part is the core RNN.

Uw_1, Uw_2, Uw_3, Uw_4 → keras calls this part as "the kernel", (eg. kernel_initializer)

S_1W, S_2W, S_3W, S_4W → keras calls this "recurrent" (recurrent_initializer)



6.3 Recurrent Neural Networks Mathematical Detail

w_i is the word at position i

s_i is the state at position i

o_i is the output at position i

$s_i = f(Uw_i + Ws_{i-1})$ core RNN

$o_i = \text{softmax}(Vs_i)$ subsequent Dense layer

In other words:-

- current_state = function1 (oldstate, current input)
- current output = function 2 (current state)
- we learn function1 & function 2 by training our network.

---

n = dimension of input vector

s = dimension of hidden state

t = dimension of output vector (after dense layer)

U is a sxn matrix

W is a sxs matrix

V is a txs matrix

Practical Details :-

often we train on just the final output & ignore the intermediate ~~the~~ outputs.

- Slight variation called Backpropagation Through Time (BPTT) is used to train RNNs.
- sensitive to length of sequence (due to vanishing gradient prob)

In practice, we still get a maximum length to our sequence,

- if input is shorter than maximum, we "pad" it
- if input is longer than maximum, we truncate.

Other Uses of RNN:-

RNN often used for text application

But RNN's can be used for other sequential data:

- Forecasting: Customer sales, Loss rates, Traffic, etc
- Speech Recognition: Call center automation, voice apps
- Manufacturing sensor data
- Genome sequence

Weakness of RNN:-

Nature of a state transition means it is hard to keep info from distant past in current memory without reinforcement.