Gated RNN: The Gated Recurrent Unit (GRU) RNN

Background :-

- Have shown success in several applications involving sequential / temporal data.

  Ex :- In speech recognition, music synthesis, NLP, machine translation etc.

- Just like LSTM, GRUs perform well in long sequence applications.

- Introduced by Kyunghyun Cho in 2014 and offers an improvement over LSTM and Vanilla RNN.

- Its success can be attributed to the "gate signaling" that controls how the present input-block and previous memory are fused to update the present activation and produce present state.

- Only flipside: Increase in parameterization through the added gate-networks.

- Unlike LSTM, GRU employs 2 gate-networks (couples two LSTM gates).

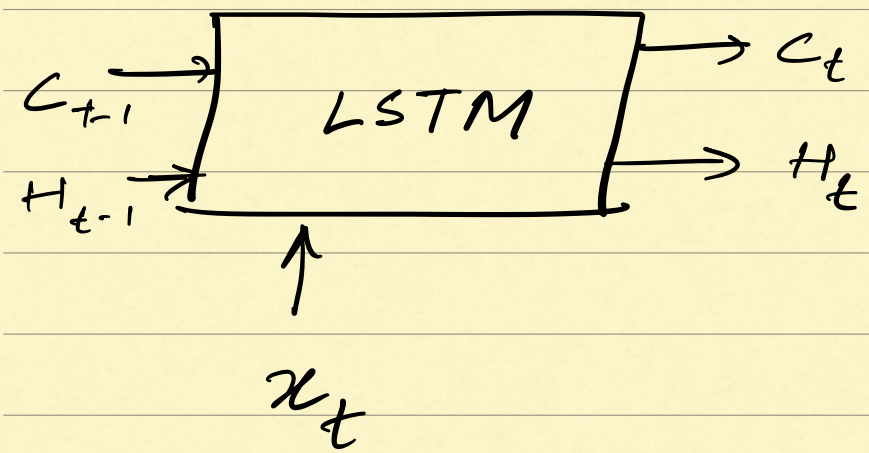
↓

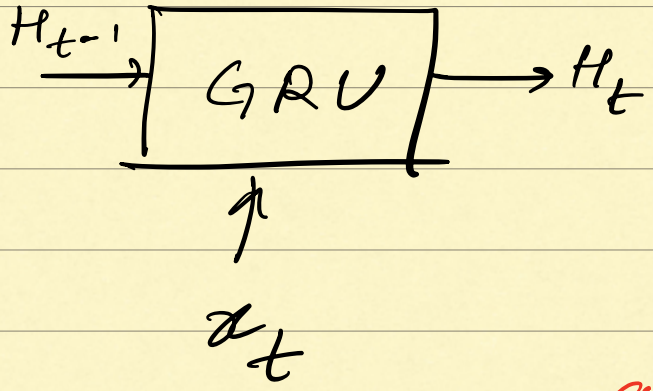
How?

- GRV couples the input gate and the forget gate by a direct sum

— GRV also removes one non-linearity in the network "signal path" by feeding back the memory cell state.

The Standard GRV RNN :-





Doesn't have a separate cell state (C_t).

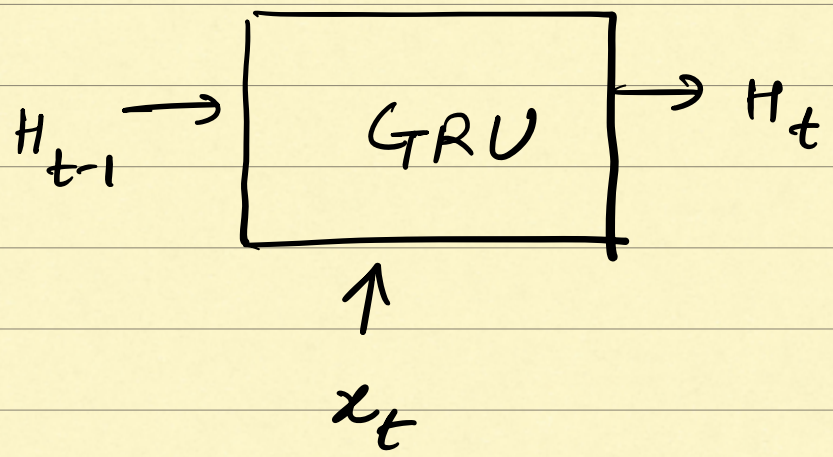
Due to simpler architecture, GRUs are faster to train.

Architecture of GRU :-

GRU reduces the gating signals to two from three in the LSTM architecture.

The two gates are

- Reset gate (r_t)
- Update gate (u_t / z_t)



- At each timestamp t, it takes an input x_t and the prev. hidden state H_{t-1}, and subsequently outputs the new hidden state H_t.

Reset gate (Short Term Memory) :-

$$r_t = \sigma(x_t \cdot U_r + H_{t-1} \cdot W_r + b_r)$$

U_r, W_r → weight matrices

b_r → 0 (for simplicity)

Range: $0 < r_t < 1$

Update gate (Long Term Memory):-

$$u_t / z_t = \sigma(x_t \cdot U_u + H_{t-1} \cdot W_u + b_u)$$

0

* How GRU works?

— Two step process:-

Step 1: Generate Candidate Hidden State

$$\hat{H}_t = \tanh\left(x_t \cdot U_g + (r_t \cdot H_{t-1}) \cdot W_g \right)$$

Tells us how much influence the previous hidden state can have on the candidate state (by adjusting r_t).

If $r_t \to 0$ $\Rightarrow$ Info. from the previous hidden state is ignored.

If $r_t \to 1$ $\Rightarrow$ Entire info. from the previous hidden state is

considered.

Step 2: Calculate the current hidden state (H_t).

— Instead of using a separate gate (like in LSTM), in GRU we use a single update gate to control both the historical info. (H_{t-1}) as well as the new info ($\hat{H}_{t-1}$).

— **Eqn:** $$H_t = u_t \cdot H_{t-1} + (1-u_t) \cdot \hat{H}_t$$

" Hence, the current hidden memory

state is a linear interpolation b/w previous hidden memory states and the candidate new state."

If $u_t \to 0$: New hidden state won't have much info from prev. hidden state

$$H_t \sim \hat{H}_t$$

If $u_t \to 1$: Previous step states for units are copied over to the current step.

$$H_t \sim H_{t-1}$$

Extra :-

The GRU model as presented in the original paper :- (Chung etc. 2014)

$$\tilde{h}_t = g \cdot (U_h \cdot (r_t \odot h_{t-1}) + W_h \cdot s_t + b_h)$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

* The two gates were presented as :-

$$z_t = \sigma (W_z h_{t-1} + W_z s_t + b_z)$$

$$r_t = \sigma(U_r h_{t-1} + W_r s_t + b_r)$$

$g(\cdot)$ $\Rightarrow$ Activation non-linearity

Typically tanh (can also be implemented as a reLU).

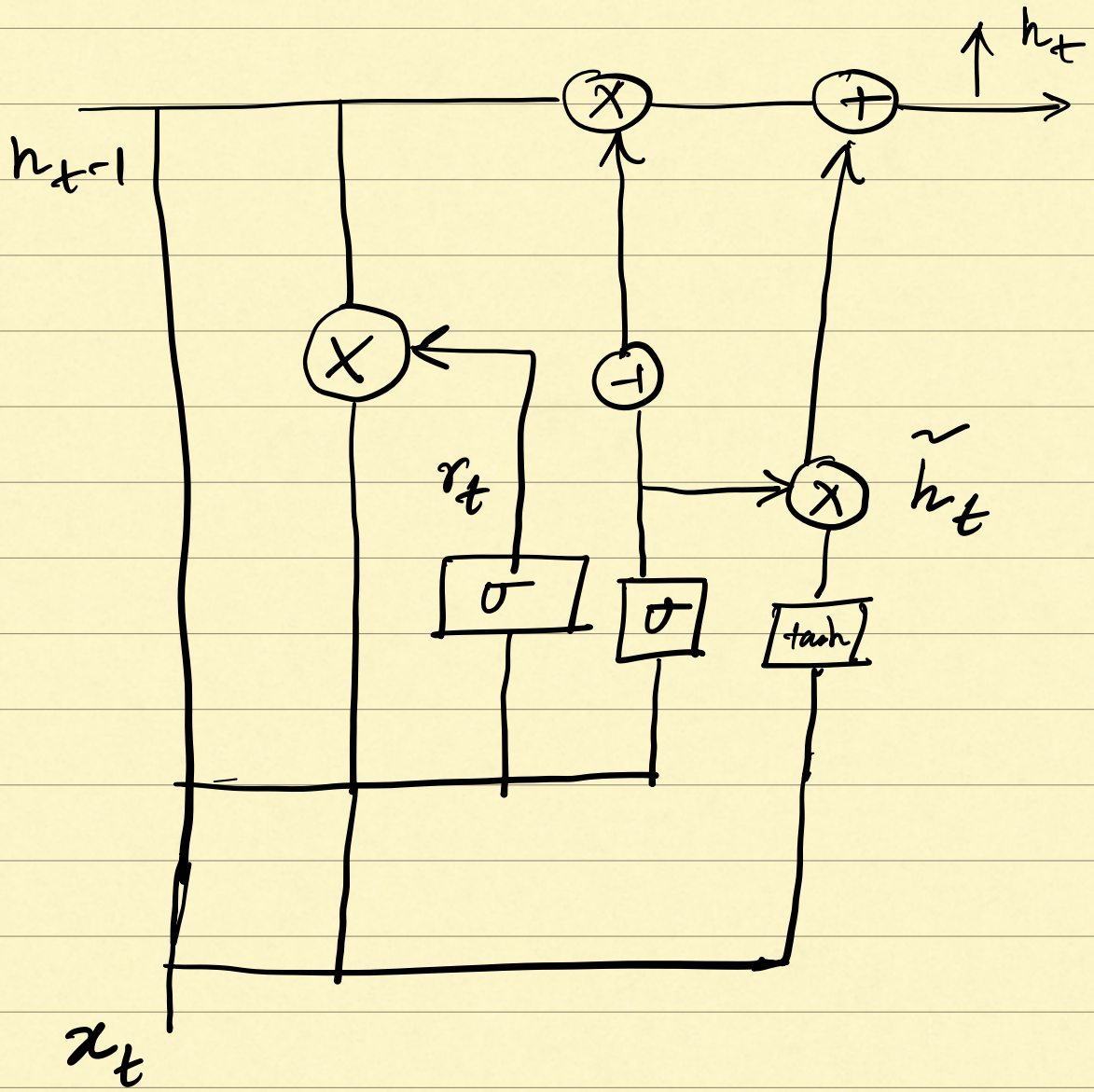
$\odot$ $\Rightarrow$ Hadamard multiplication

Weighted sum is implemented via element wise H.M. to gating signals.

Gates intuitively resemble "analog switches".

- GRU is similar to LSTM, however with one less external gating signal in the coupling interpolation.

- This eliminates one gating signal and the associated parameters.



⊛ How GRU overcomes the problem of vanishing / exploding gradients during backpropagation ?

⇒ In RNN BP, the component that may lead to vanishing or exploding gradients in the component $\frac{\partial h_t^{(i)}}{\partial h_k^{(i)}}$, which backpropagation the error at sequence step t to sequence step k so that the model learns distant dependencies or correlations.

$$\frac{\partial h_t^{(i)}}{\partial h_k^{(i)}} = (u_{ii})^{t-k} \prod_{g=k+1}^{t} \sigma'(z_g^{(i)})$$

When $(t-k)$ is large, a vanishing gradient condition will arise when the gradients of the activations in the hidden state and / or the weights are less than 1 since the product of $(t-k)$ in them would force the overall product to near zero.

Sigmoid and tanh gradients are often less than 1 and saturate fast where they have near-zero gradients, thus making the problem of v.g. more severe.

Similarly, exploding grodients can happen when the weight-connection w_{ii} b/w the ii^{th} hidden to the i^{th} hidden unit

is greater than 1 since the product of $(t-k)$ in them would make the term $(u_{ii})^{t-k}$ exponentially large for large values of $(t-k)$.

When $u_t \to 1 \Rightarrow h_t^{(i)} \simeq h_{t-1}^{(i)}$

$\forall\, i \in k$

k : set of all hidden units for which $z_t^{(i)} \simeq 0$

Now, $\dfrac{\partial h_t^{(i)}}{\partial h_{t-1}^{(i)}} \simeq 1$

This will ensure that the notorious term $\dfrac{\partial h_t^{(i)}}{\partial h_k^{(i)}}$ is also 1

Why ?

$$\Rightarrow \frac{\partial h_t^{(i)}}{\partial h_k^{(i)}} = \prod_{q=t'+1}^{t} \frac{\partial h_q^{(i)}}{\partial h_{q-1}^{(i)}}$$

$$= \underbrace{1 \cdot 1 \cdot \cdots \cdot 1}_{(t-k) \text{ times}}$$

$$= 1$$

This allows the hidden states to be copied over many sequence steps w/o activation and hence the chances of a v.g. diminish and the model is able to learn temporally long-distance correlation b/w words.

GRU vis LSTM :-

Correspondence among two gated RNNs :

$$i_t \longleftrightarrow z_t$$

$$f_t \longleftrightarrow (1 - z_t)$$

$$c_t \longleftrightarrow h_t$$

$$z_t \longleftrightarrow \tilde{h}_t$$

Using LSTM notations, we can express GRU eqns :-

$$\tilde{c}_t = g\left(U_c h_{t-1} + W_c s_t + b_c\right)$$

$$c_t = (1 - i_t) \odot c_{t-1} + i_t \odot \tilde{c}_t$$

$$h_{t-1} = o_t \odot c_{t-1}$$

(*) **Summary :-**

Hochreiter & Schmidhuber (1997)

Bengio (1994)

$\Rightarrow$ It is difficult to capture long-term dependencies using simple RNNs because the gradients tend to vanish or explode with long sequences.

- Two particular architectures :-

① LSTM (1997)

② GRU (2014)