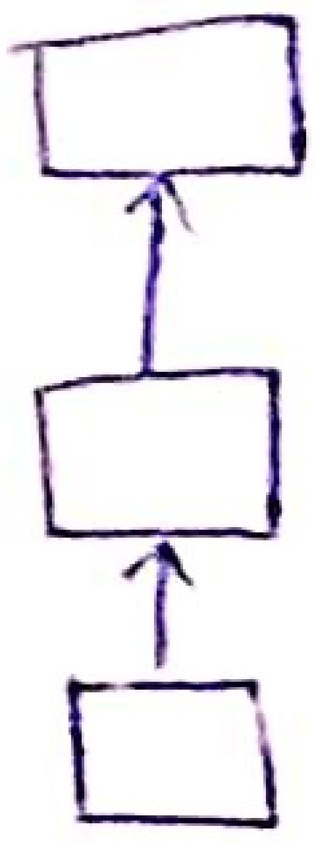
(10)

Recurrent Neural Network

One to one



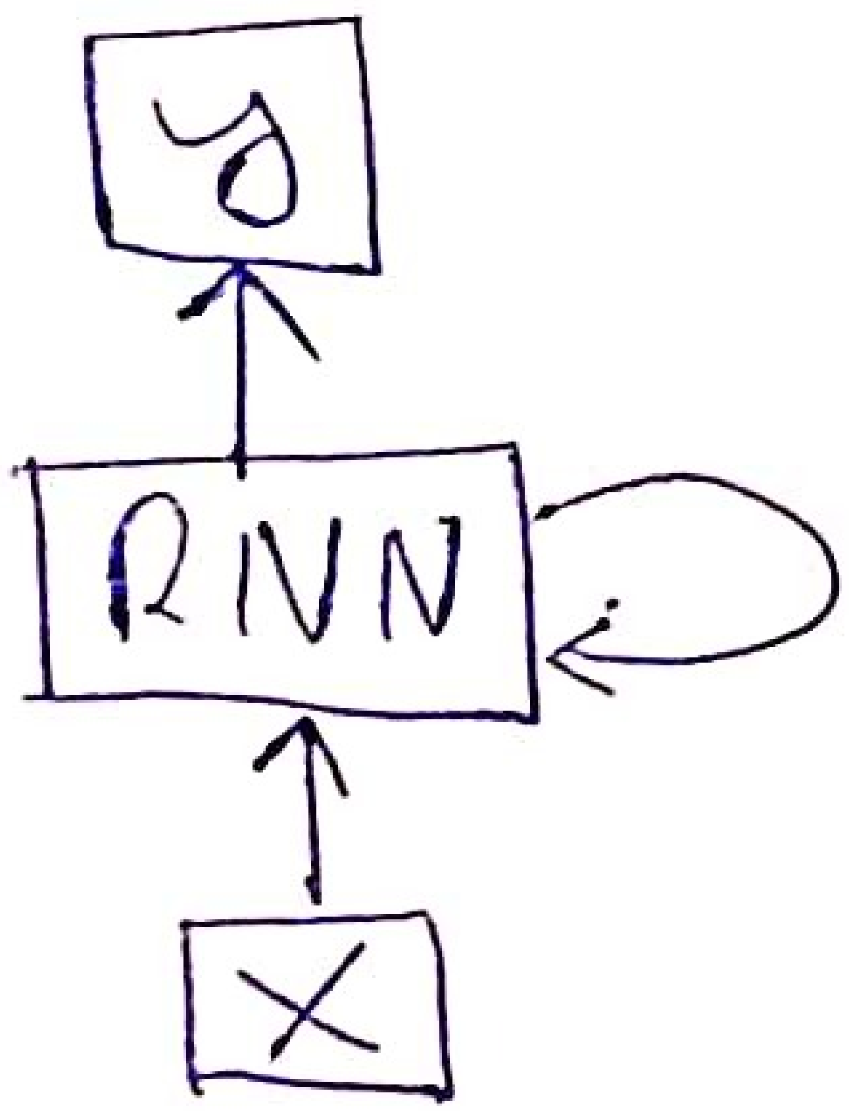
{Vanilla Neural Network}

→ One to many {Image Captioning}

→ many to one {Sentiment Classification}

→ many to many {Machine Translation}

{Video classification on frame level}



⇒ We can process a sequence of vectors x by applying a recurrence formula at every time step:

$$\underbrace{h_t}_{\text{New State}} = \underbrace{f_W}_{\text{Same function with Parameter } W}(\underbrace{h_{t-1}}_{\text{Old State}}, \underbrace{x_t}_{\text{Input vector at some time step}})$$

⇒ Notice: The same function & the same set of parameters are used at every time step.

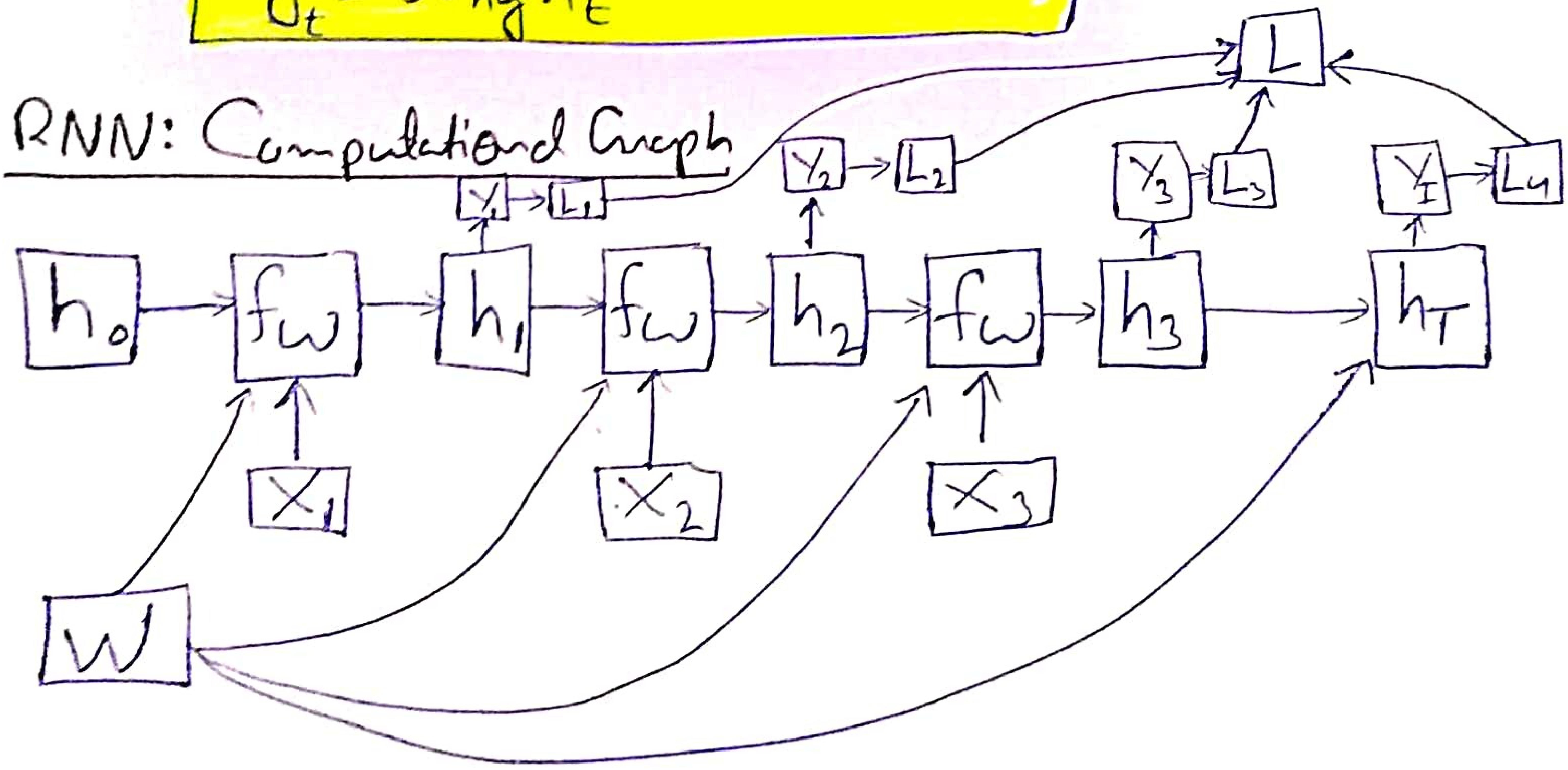
★ (Vanilla) Recurrent Neural Network

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

↓

$$\boxed{\begin{aligned} h_t &= \tanh(W_{hh} h_{t-1} + W_{xh} x_t) \\ y_t &= W_{hy} h_t \end{aligned}}$$

★ RNN: Computational Graph



⇒ Re-use the same weight matrix at every time-step.

{Encode input Sequence in a Single Vector} → {Produce output Sequence from Single input Vector}

Truncated backward propagation through time

- ↳ Run forward & backward through chunks of the Sequence instead of whole Sequence.
- ↳ Carry hidden states forward in time forever, but only backpropagate for some smaller number of steps.