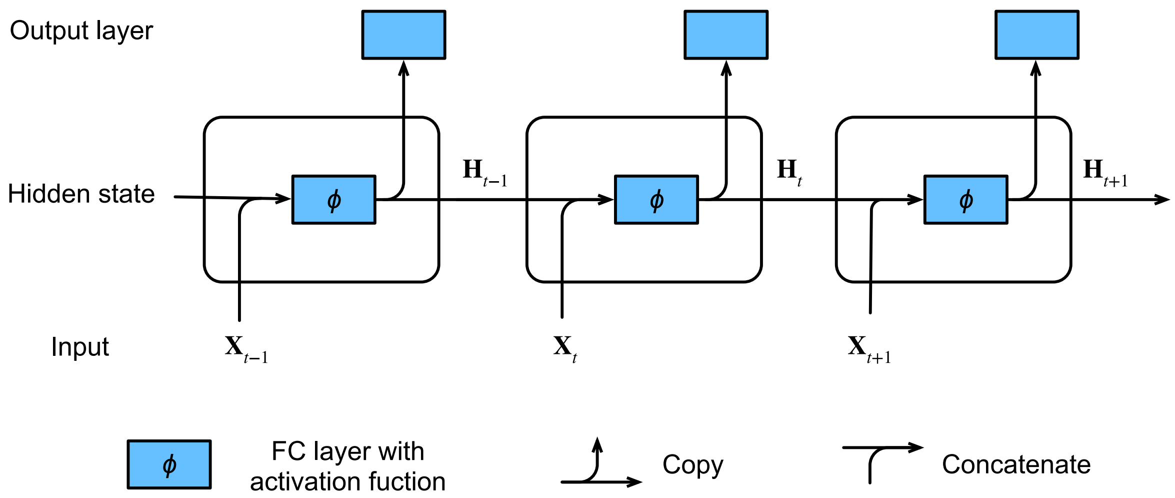
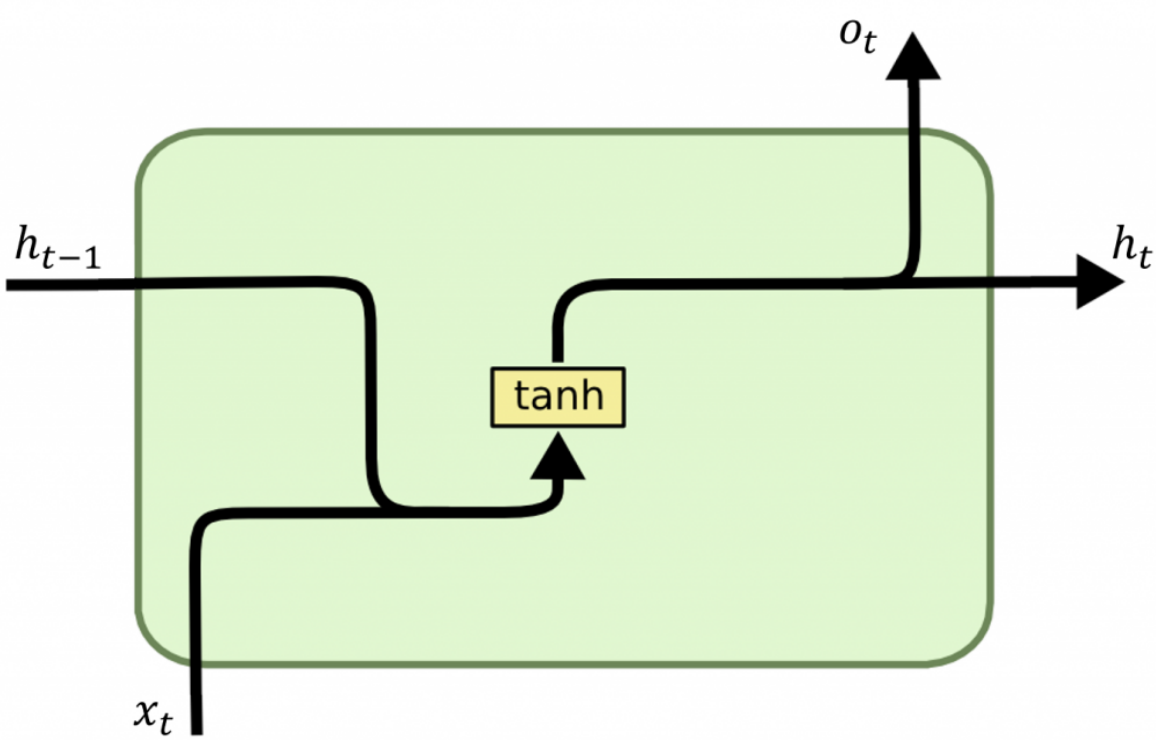
D2L RNN Notes

Recurrent Neural Network

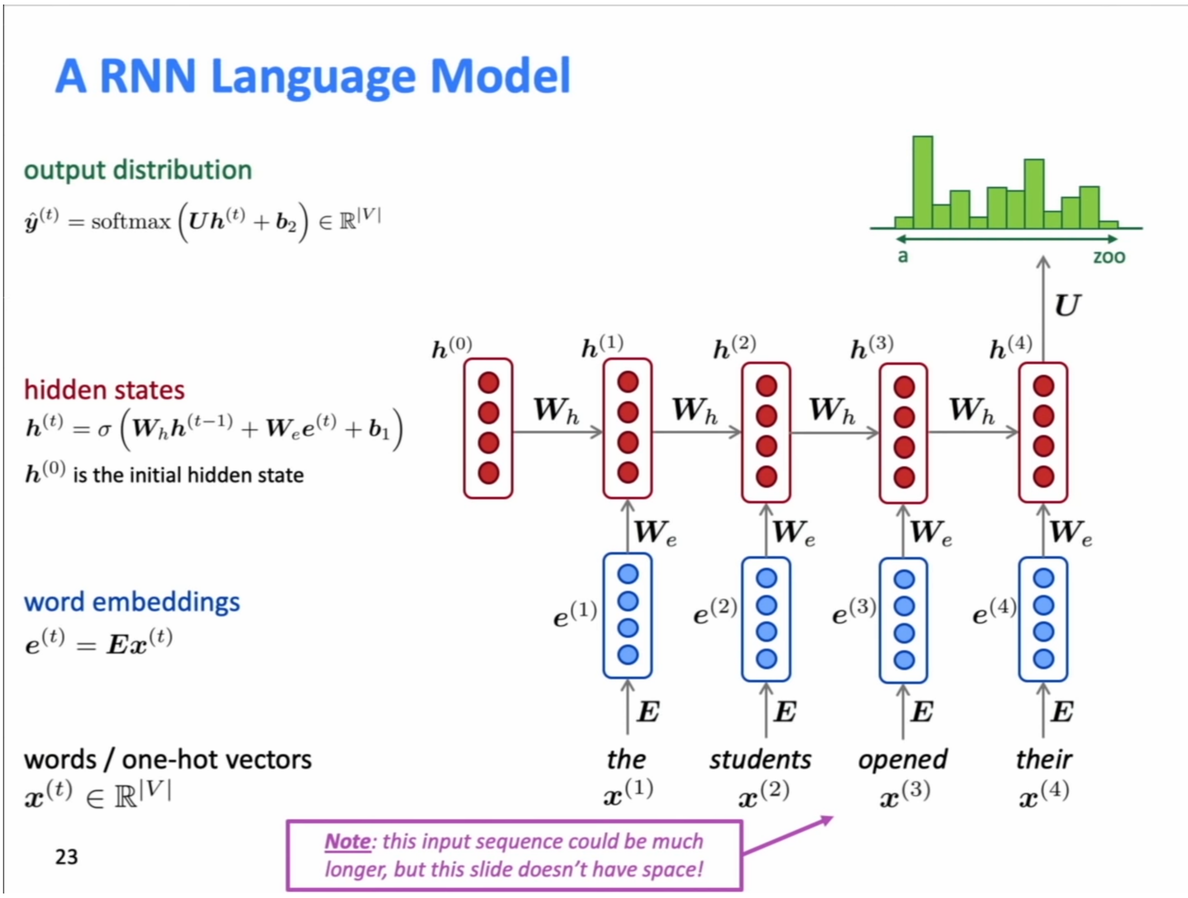


- RNN is a type of model architecture that aims to process **sequential** data (any length of input)
- Application
  - Text Classification
  - Machine Translation
  - Language Model
  - Text Summary
  - Time Series Problem
  - ...

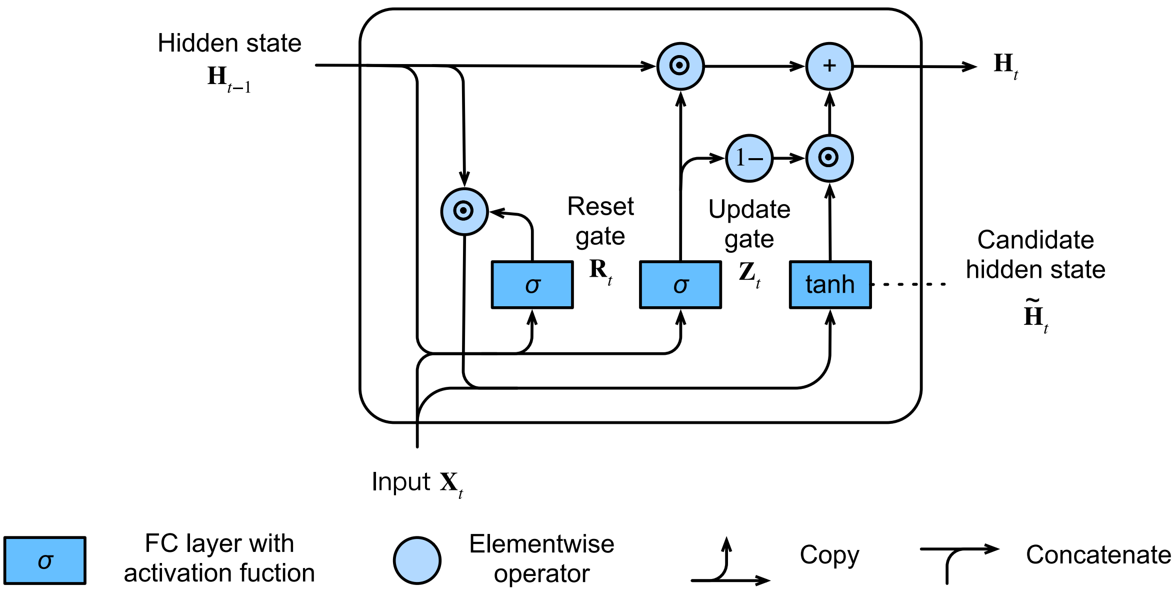
Vanilla RNN



- The classic RNN architecture, may encounter gradient vanishing & gradient explosion

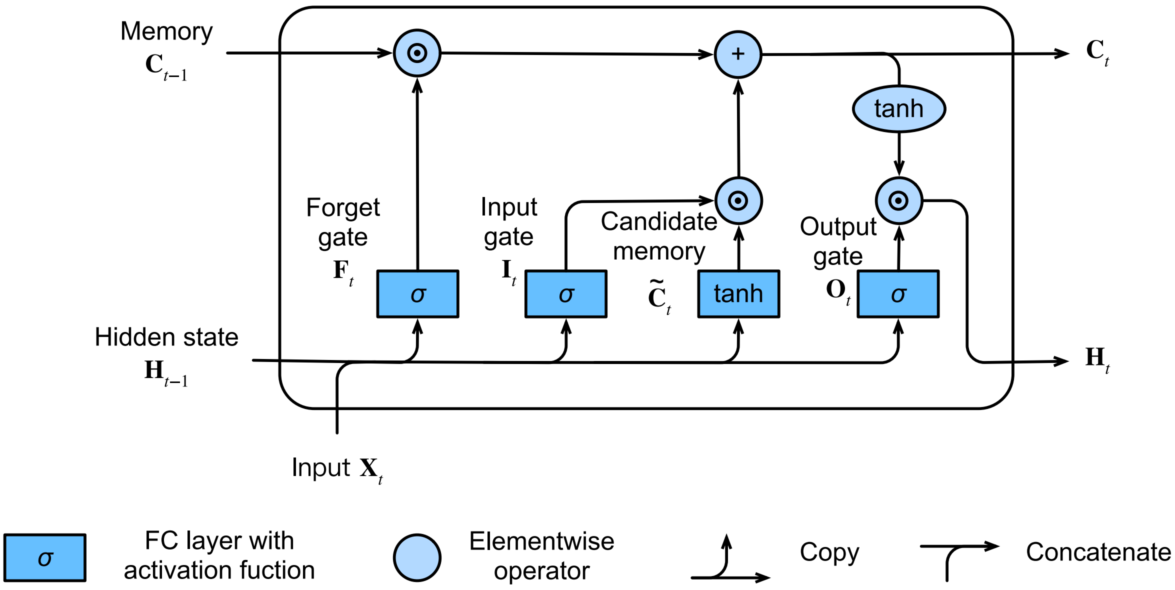


GRU



- Introducing the `Reset Gate` and `Update Gate`
  - Reset: $\mathbf{R}_t = \sigma(\mathbf{X}_t\mathbf{W}_{xr} + \mathbf{H}_{t-1}\mathbf{W}_{hr} + \mathbf{b}_r) \in (0,1)$ (decides what to remember/forget from the last hidden state for current **candidate hidden state**)
  - Update: $\mathbf{Z}_t = \sigma(\mathbf{X}_t\mathbf{W}_{xz} + \mathbf{H}_{t-1}\mathbf{W}_{hz} + \mathbf{b}_z) \in (0,1)$ (decides what to update to **new/current hidden state**)
- Candidate hidden state: $\tilde{\mathbf{H}}_t = \tanh(\mathbf{X}_t\mathbf{W}_{xh} + (\mathbf{R}_t \odot \mathbf{H}_{t-1})\,\mathbf{W}_{hh} + \mathbf{b}_h)$
- Result Hidden State: $\mathbf{H}_t = \mathbf{Z}_t \odot \mathbf{H}_{t-1} + (1 - \mathbf{Z}_t) \odot \tilde{\mathbf{H}}_t$

LSTM



- Similar to GRU, but with 3 gates and separates memory & hidden state
- 3 gates
  - Input Gate: $\mathbf{I}_t = \sigma(\mathbf{X}_t \mathbf{W}_{xi} + \mathbf{H}_{t-1} \mathbf{W}_{hi} + \mathbf{b}_i) \in (0,1)$ (decides what to input/use from **current candidate memory cell**)
  - Forget Gate: $\mathbf{F}_t = \sigma(\mathbf{X}_t \mathbf{W}_{xf} + \mathbf{H}_{t-1} \mathbf{W}_{hf} + \mathbf{b}_f) \in (0,1)$ (decides what to remember/forget from **previous memory cell**)
  - Output Gate: $\mathbf{O}_t = \sigma(\mathbf{X}_t \mathbf{W}_{xo} + \mathbf{H}_{t-1} \mathbf{W}_{ho} + \mathbf{b}_o) \in (0,1)$ (decides what to output to **current hidden state**)
- $\mathbf{H}_t$ vs $\mathbf{C}_t$
  - $\mathbf{H}_t = \mathbf{O}_t \odot \tanh(\mathbf{C}_t)$
  - $\mathbf{H}_t$ is guaranteed to be $\in (-1,1)$, while $\mathbf{C}_t$ is not
  - kind of storing more information inside $\mathbf{C}_t$ to partially prevent **gradient vanishing*
- Candidate Memory Cell: $\tilde{\mathbf{C}}_t = \tanh(\mathbf{X}_t \mathbf{W}_{xc} + \mathbf{H}_{t-1} \mathbf{W}_{hc} + \mathbf{b}_c)$
- Result Memory Cell: $\mathbf{C}_t = \mathbf{F}_t \odot \mathbf{C}_{t-1} + \mathbf{I}_t \odot \tilde{\mathbf{C}}_t$
- Result Hidden State: $\mathbf{H}_t = \mathbf{O}_t \odot \tanh(\mathbf{C}_t)$

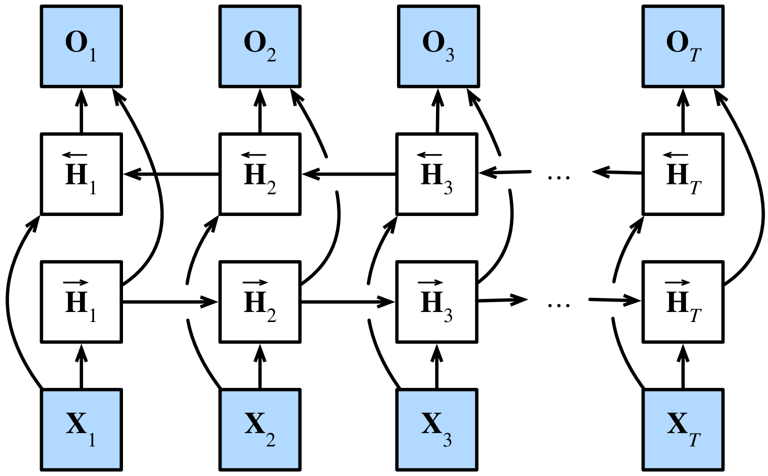
Perplexity

- Loss for language models, describes the loss of a sentence by taking the **exponential of average cross entropy of the tokens** in the sentence.
- $\exp\left(-\frac{1}{n} \sum_{t=1}^{n} \log P(x_t \mid x_{t-1}, \ldots, x_1)\right)$

Gradient Clipping

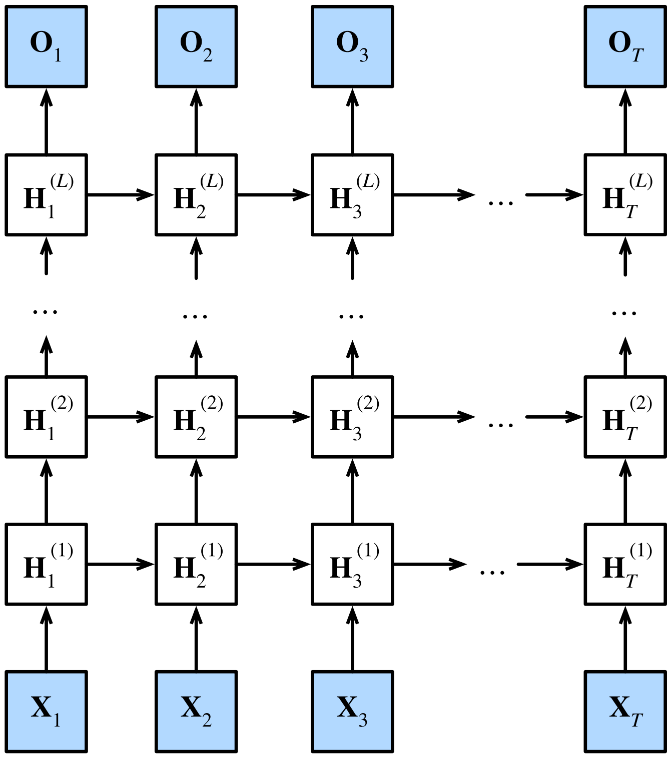
- $\mathbf{g} \leftarrow \min\left(1, \frac{\theta}{\|\mathbf{g}\|}\right) \mathbf{g}$
- Projects $\mathbf{g}$ to a circle with radius of θ, preventing the gradients to be too large and increase the robustness of the parameters

Bidirectional RNN



- Aims to provide RNN the ability to model information from the future (instead of only from previous), commonly used in NMT's encoder

Multilayer RNN



- Aims to make the model more complex and increase its expressivity
- A hidden state $\mathbf{H_i^{(j)}}$ is passed both to the next layer (*up*) and next timestep (*right*)

BackProp Through Time

- TODO