9.3. Deep Recurrent Neural Networks

Lecture based on "Dive into Deep Learning" **http://D2L.AI** (Zhang et al., 2020)

Prof. Dr. Christoph Lippert

Digital Health & Machine Learning

In the case of the perceptron we increased *flexibility* of learned functions by adding *more layers*. Within RNNs, we first need to decide *how* and *where* to add extra non-linearity.

- We could add extra non-linearity to the gating mechanisms.

  That is, instead of using a single perceptron we could use multiple layers. This leaves the *mechanism* of the LSTM unchanged.

  Instead it makes it more sophisticated.

  This would make sense if we were led to believe that the LSTM mechanism describes some form of universal truth of how latent variable auto-regressive models work.

- We could stack multiple layers of LSTMs on top of each other.

  This results in a mechanism that is more flexible, due to the combination of several simple layers.

  In particular, data might be relevant at different levels of the stack.

  For instance, we might want to keep high-level data about financial market conditions (bear or bull market) available at a high level, whereas at a lower level we only record shorter-term temporal dynamics.

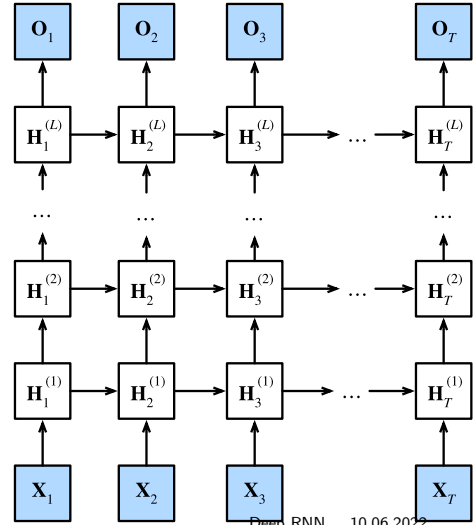
Deep recurrent neural network with L hidden layers. Each hidden state is continuously passed to the next time step of the current layer and the next layer of the current time step.

- At step t we have a mini-batch $\mathbf{X}_t \in \mathbb{R}^{n \times d}$
  (examples: n, inputs: d).
- The hidden state of hidden layer ℓ $(\ell = 1, \ldots, T)$ is $\mathbf{H}_t^{(\ell)} \in \mathbb{R}^{n \times h}$
  (hidden units: h),
- the output layer is $\mathbf{O}_t \in \mathbb{R}^{n \times q}$
  (outputs: q)
- a hidden layer activation function f_l for layer l.
- hidden state of layer 1 uses $\mathbf{X}_t$ as input.

$$\mathbf{H}_t^{(1)} = f_1\left(\mathbf{X}_t, \mathbf{H}_{t-1}^{(1)}\right)$$

- subsequent layers use the hidden state of the previous layer as input.

$$\mathbf{H}_t^{(l)} = f_l\left(\mathbf{H}_t^{(l-1)}, \mathbf{H}_{t-1}^{(l)}\right)$$

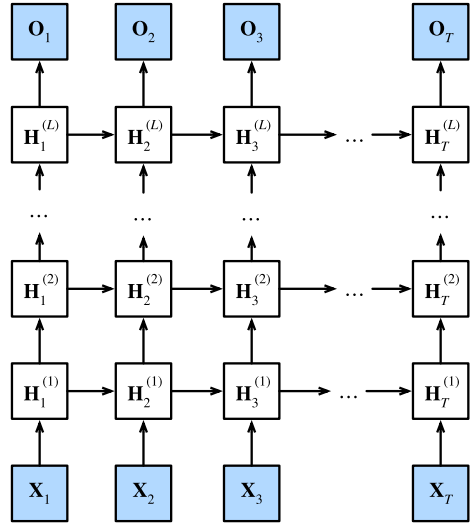


Deep recurrent neural network with L hidden layers. Each hidden state is continuously passed to the next time step of the current layer and the next layer of the current time step.

The **output function** g is only based on the hidden state of hidden layer L.

$$\mathbf{O}_t = g\left(\mathbf{H}_t^{(L)}\right)$$

- Hyper parameters:
  - number of hidden layers L
  - number of hidden units h
- Components can be
  - a regular RNN
  - a GRU
  - an LSTM



Summary

- In deep recurrent neural networks, hidden state information is passed to the next time step of the current layer and the next layer of the current time step.
- There exist many different flavors of deep RNNs, such as LSTMs, GRUs or regular RNNs.
- Initialization of the models requires care.
- Overall, deep RNNs require considerable amount of work (learning rate, clipping, etc) to ensure proper convergence.