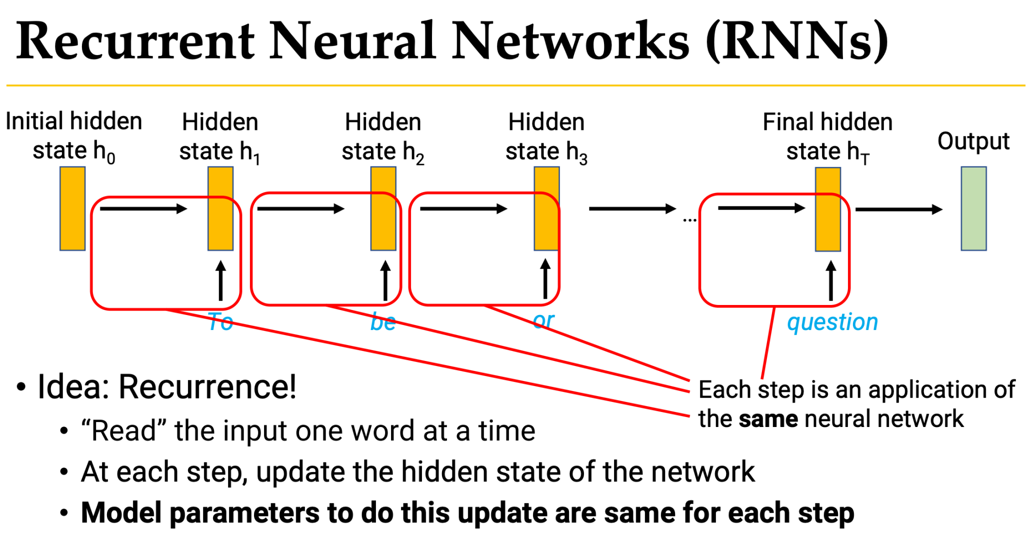
Recurrent NN for sequential data

Handling textual data
- images: inputs are fixed dimensional
  - Can crop/rescale as needed
- text: inputs are variable-sized
- How can we use the same set of model parameters to handle inputs of any size?



Word embedding
- learn a vector that represents each word
  - Each word w in vocabulary V has vector Vw of size d
  - |V|*d parameters needed
- Similar words get similar vectors

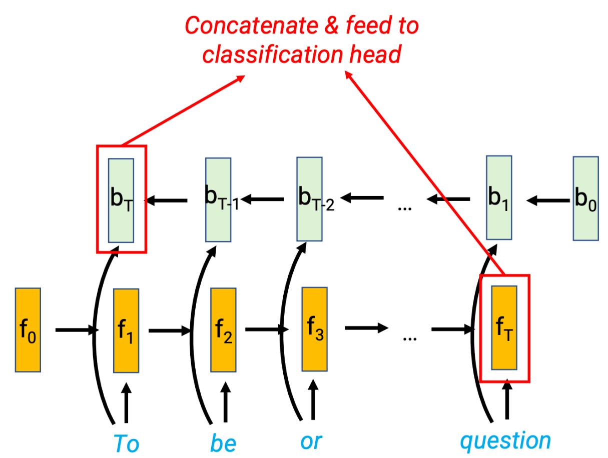
Recurrence vs Depth
- Deep networks (i.e., adding more layers)
  - Computation graph becomes longer
  - Number of parameters also grows; each step uses new parameters
- Recurrent neural networks
  - Computation graph becomes longer
  - Number of parameters fixed; each step uses same parameters

Language Modeling
- At each step, predict the next word given current hidden state
  - Essentially a softmax regression "head"—takes in hidden state, outputs distribution over Vocabulary + [END]
- Start with special [BEGIN] token (so the first word model generates is first real word)
- One step's output becomes next step's input ("autoregressive")
- To mark end of sequence, model should predict the [END] token
- Called a "Decoder" because it looks at the hidden state and "decodes" the next word

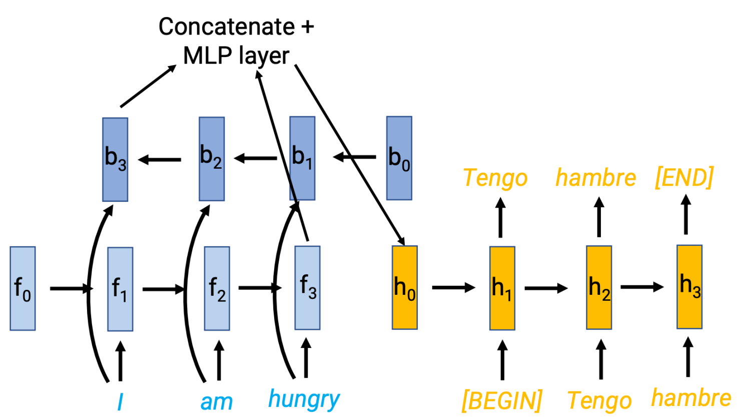
Long-Range Dependencies
- Every step, you update the hidden state with the current word
- Over time, information from many words ago can easily get lost!
- This means RNNs can struggle to model long-range dependencies

Advanced RNNs
- “Gated” RNNs (GRUs, LSTMs)
- Better at holding on to long-range state
- These are usually preferable to the RNN variant I showed today
- They work the same way, but the recurrence relationship between previous hidden state and next hidden state is more complicated…

Bi-directional encoders



Sequence-to-Sequence



Attention