# Applications of RNNs

From now on, RNN refers to LSTM

# Applications of RNNs

Task Variations

- Sequence classification
- Per timestep classification
- Sequence generation

Architecture Variations

- Bidirectional RNN
- Averaging vs. Summary vector
- Multilayer RNN

# Applications of RNNs

Task Variations

- Sequence classification
- Per timestep classification
- Sequence generation

Architecture Variations

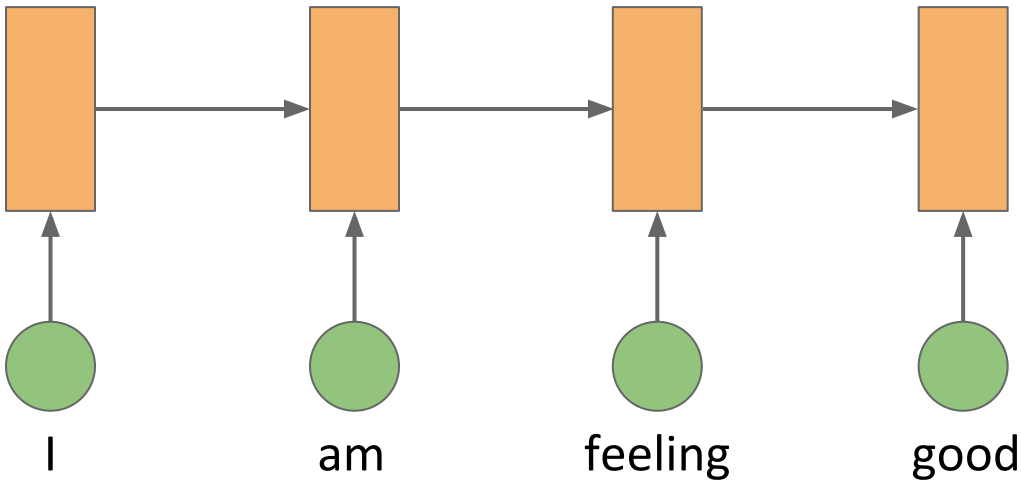
- Bidirectional RNN
- Averaging vs. Summary vector
- Multilayer RNN

# Sequence Classification

- In the code examples, we saw **sentiment detection**
- RNN models are a great way to achieve this!

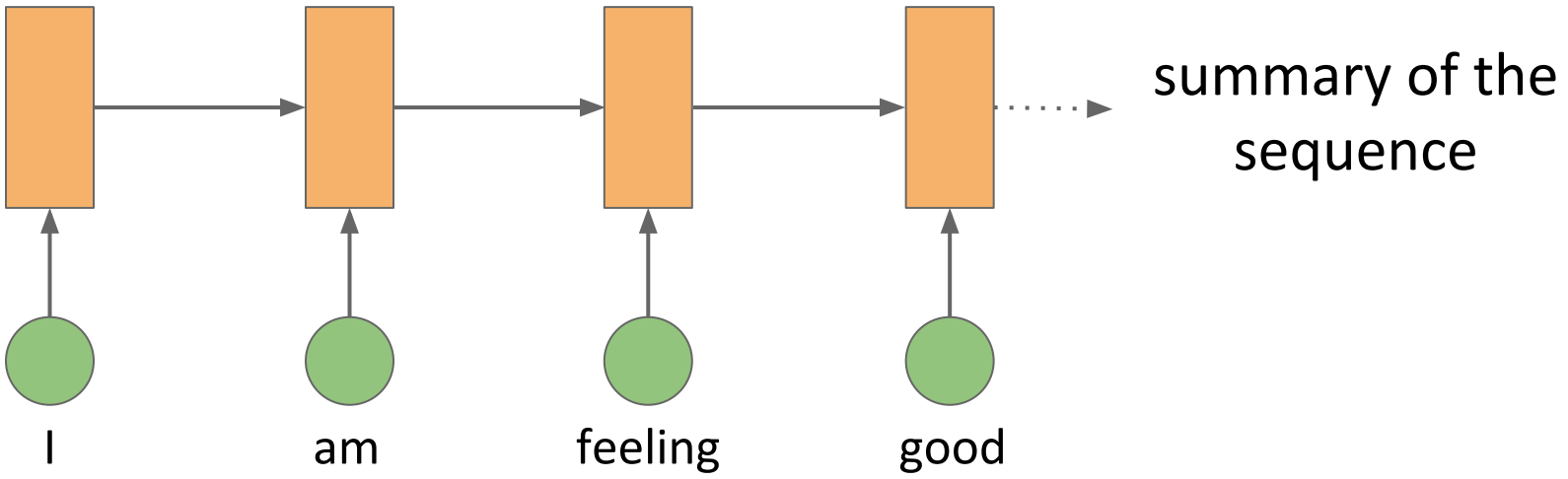
# Sequence Classification

- Read the sequence step by step



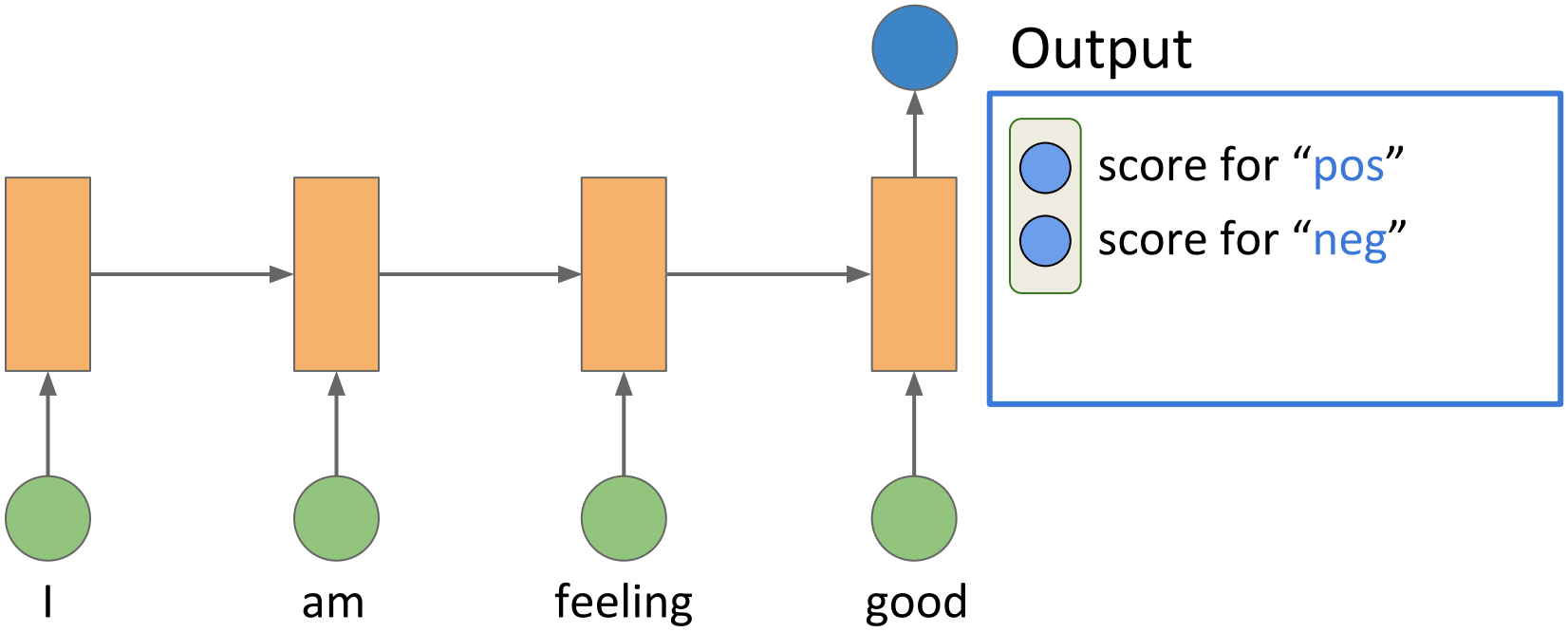
# Sequence Classification

- Read the sequence step by step
- Last step has a summary of the complete sequence



# Sequence Classification

- Read the sequence step by step
- Last step has a summary of the complete sequence
- Apply softmax



# Applications of RNNs

Task Variations

- Sequence classification
- Per timestep classification
- Sequence generation

Architecture Variations

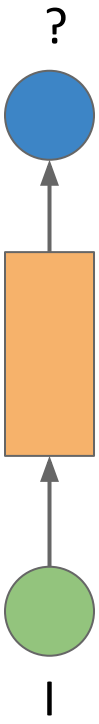
- Bidirectional RNN
- Averaging vs. Summary vector
- Multilayer RNN

# Per timestep Classification

- Given a sequence, we want to make a prediction for every time step
- Here, prediction of a word is dependent on the input word and its context only
- For example, in POS tagging, to predict the tag of the current word, we look at the current input word and it’s previous word

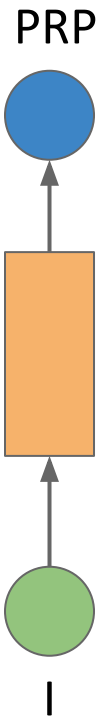
# Per timestep Classification

Predict the POS tag of the current word “I”. Here there is no context



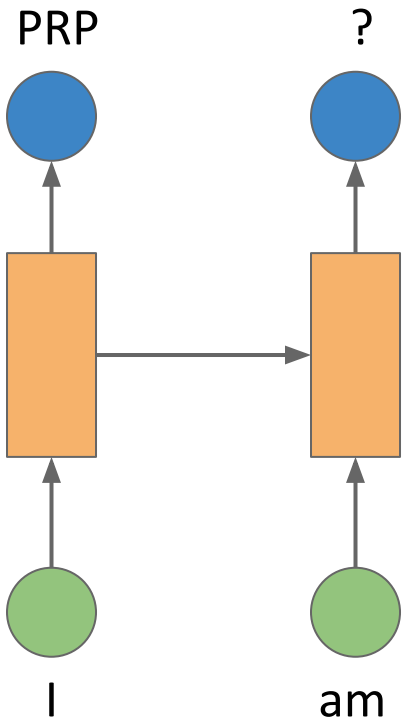
# Per timestep Classification

Predict the POS tag of the current word “I”. Here there is no context



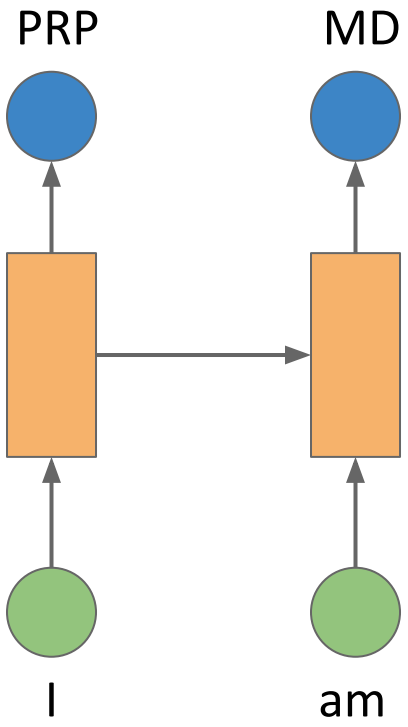
# Per timestep Classification

Predict the POS tag of the current word “am” given the context “I”



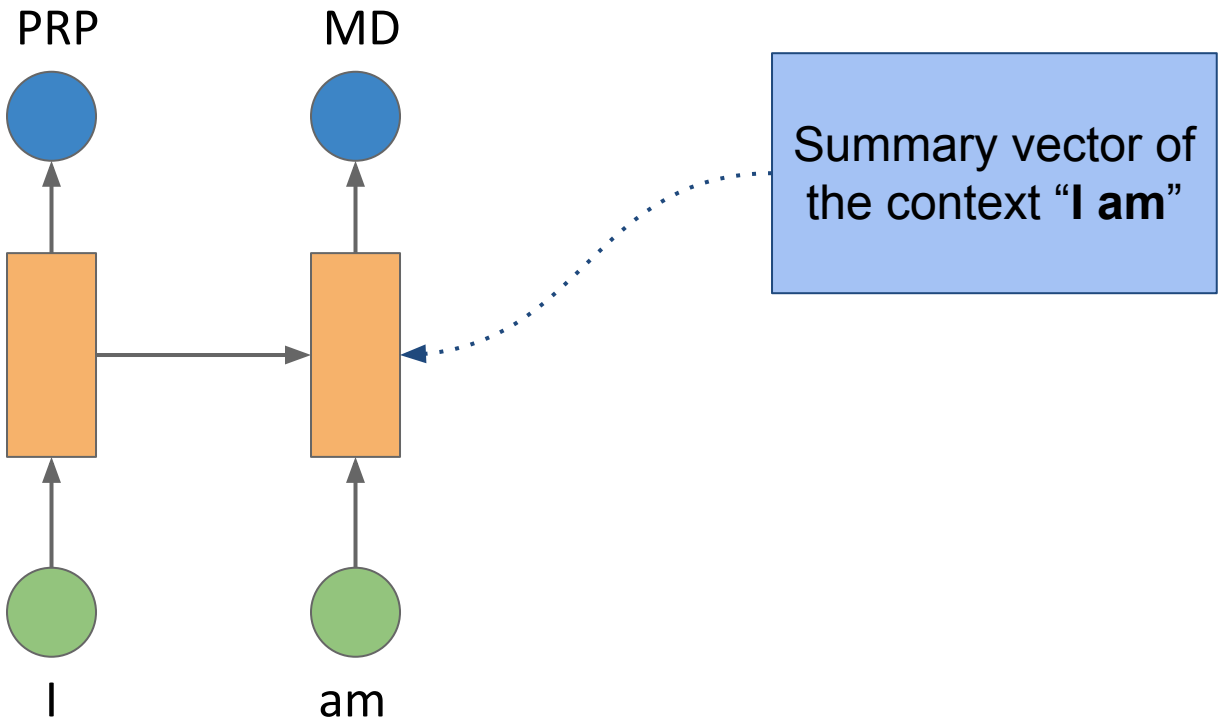
# Per timestep Classification

Predict the POS tag of the current word “am” given the context “I”



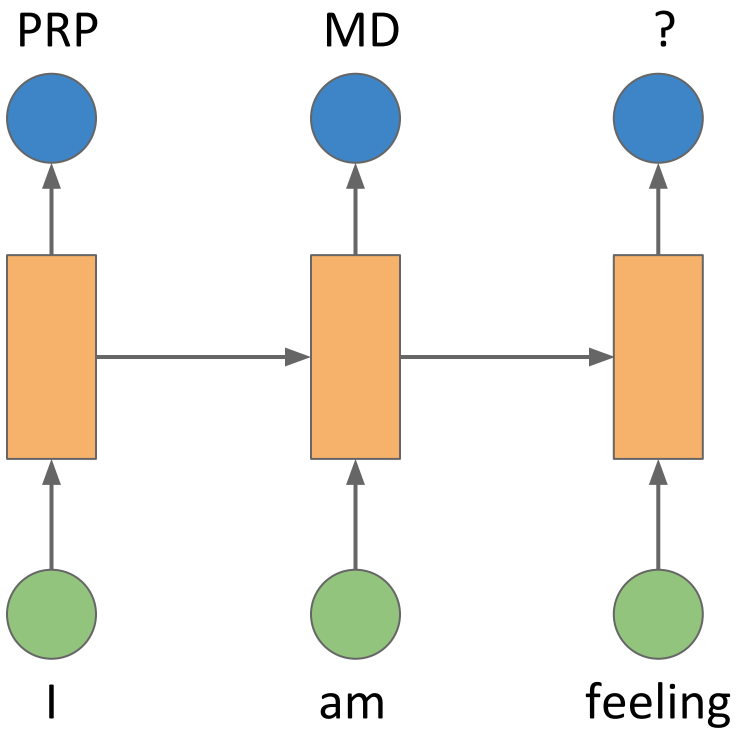
# Per timestep Classification

Predict the POS tag of the current word “am” given the context “I”



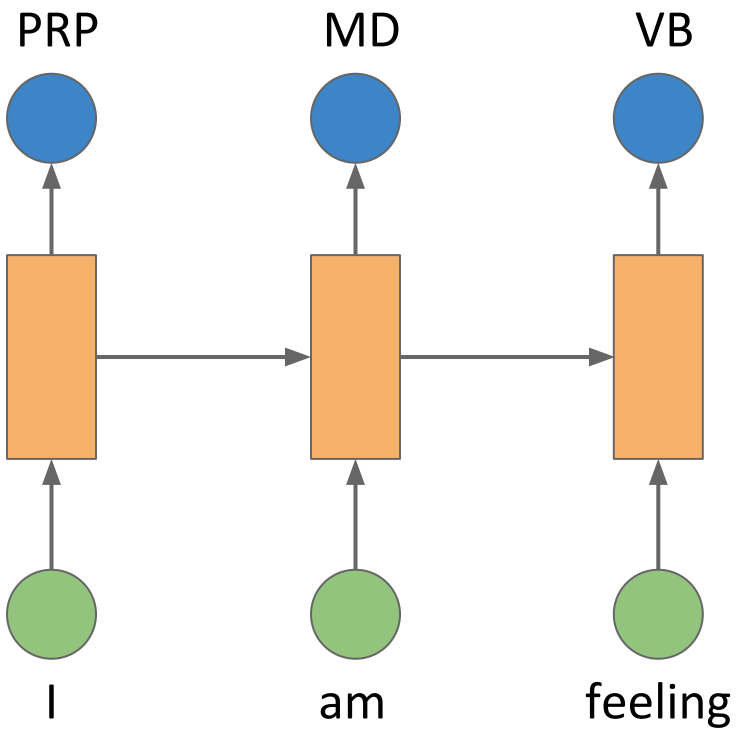
# Per timestep Classification

Predict the POS tag of the current word “feeling” given the context “I am”



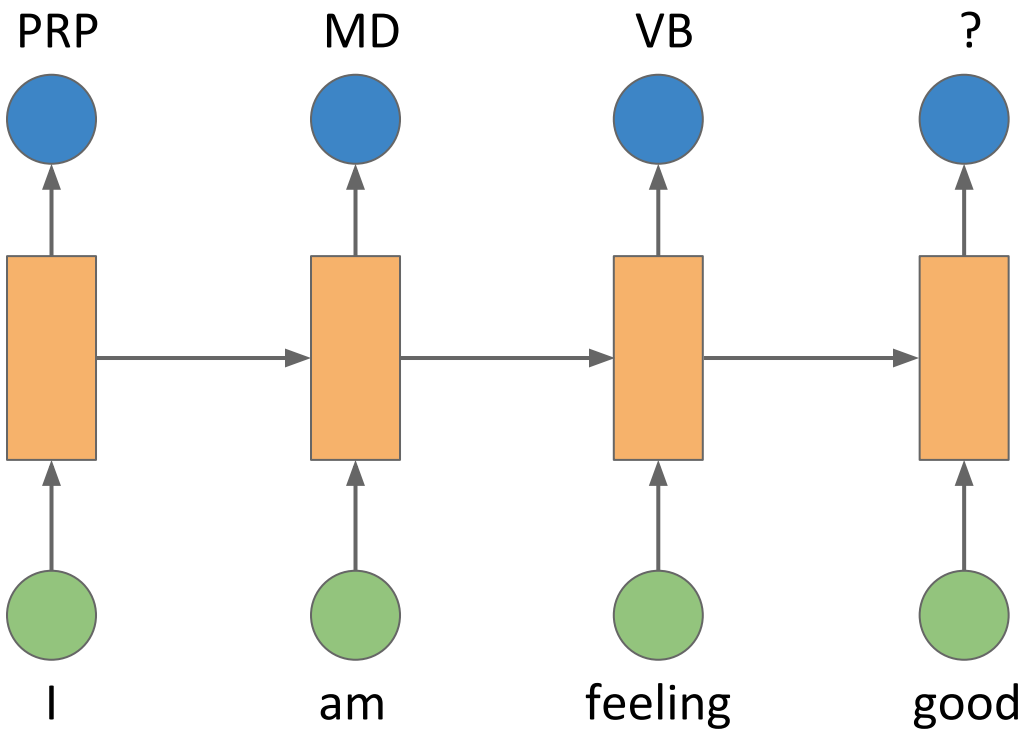
# Per timestep Classification

Predict the POS tag of a current word given the context



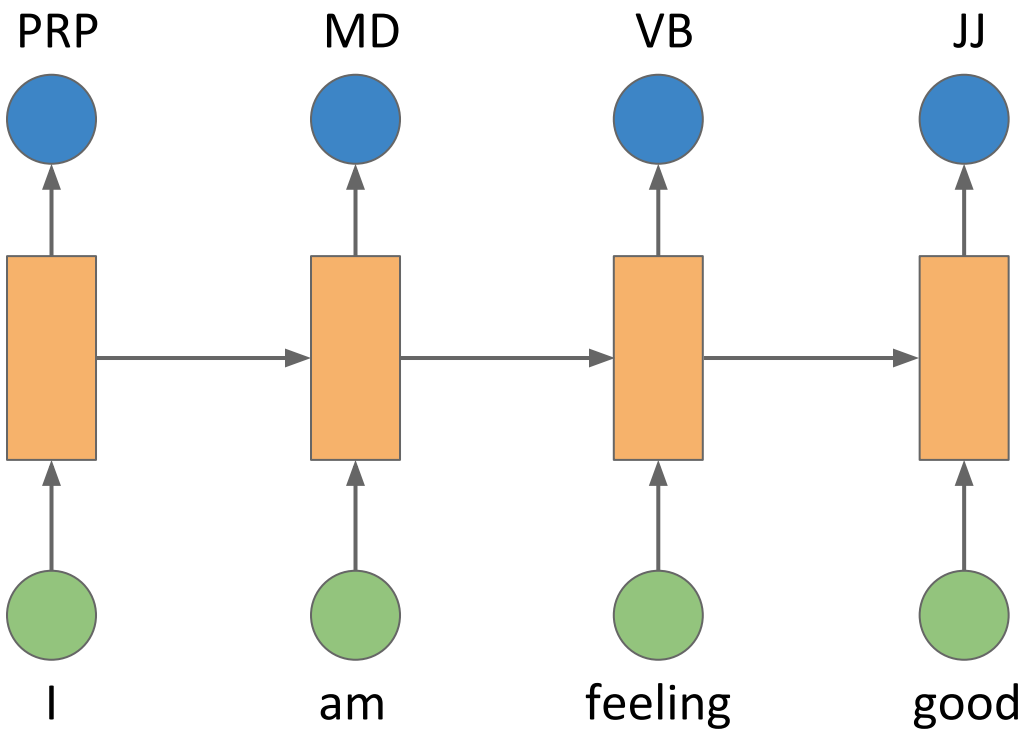
# Per timestep Classification

Predict the POS tag of a current word given the context



# Per timestep Classification

Predict the POS tag of a current word given the context



# Per timestep Classification

- Since we are predicting at every timestep, loss for one input sequence will be the sum of individual losses at every step

$$L_i = -\log(softmax(f)_c)$$

# Applications of RNNs

Task Variations

- Sequence classification
- Per timestep classification
- Sequence generation

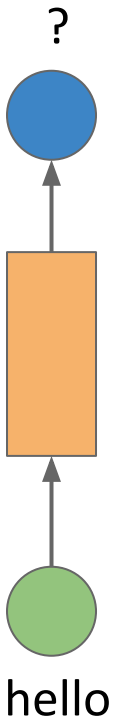
Architecture Variations

- Bidirectional RNN
- Averaging vs. Summary vector
- Multilayer RNN

# Sequence Generation

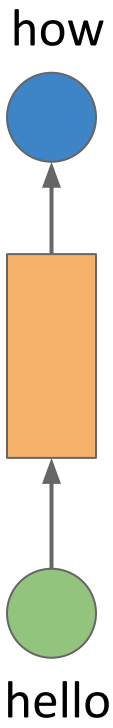
- Given a context, predict the next word

# Sequence Generation



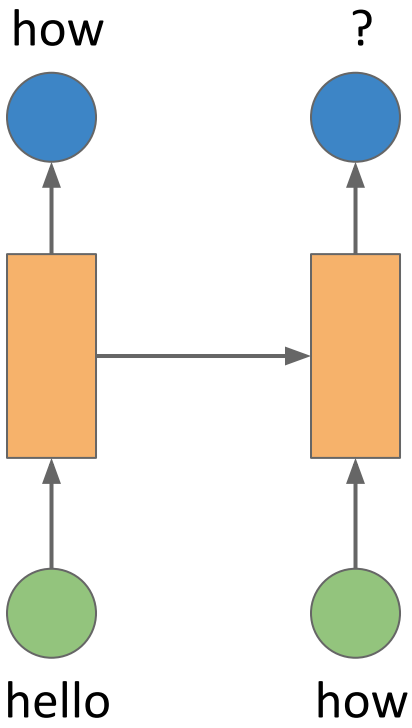

Given an input word, what is the next word?

# Sequence Generation



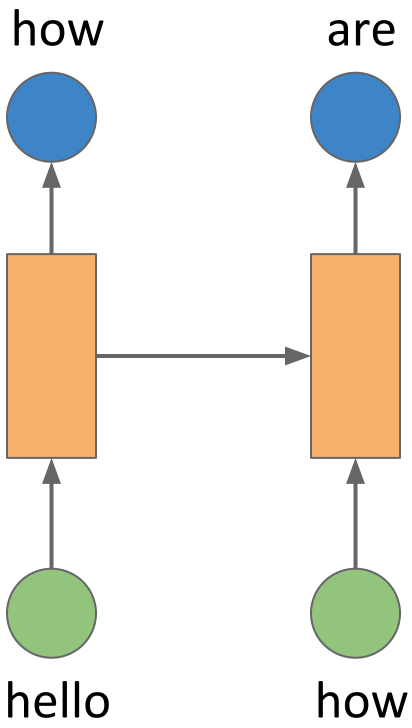

Given an input word, what is the next word?

# Sequence Generation



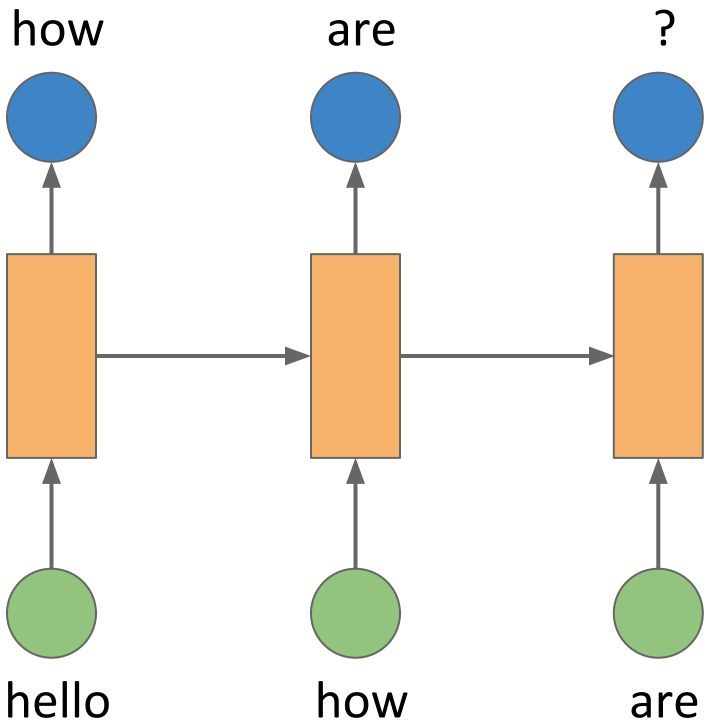

Given an input word and previous context, what is the next word?

# Sequence Generation



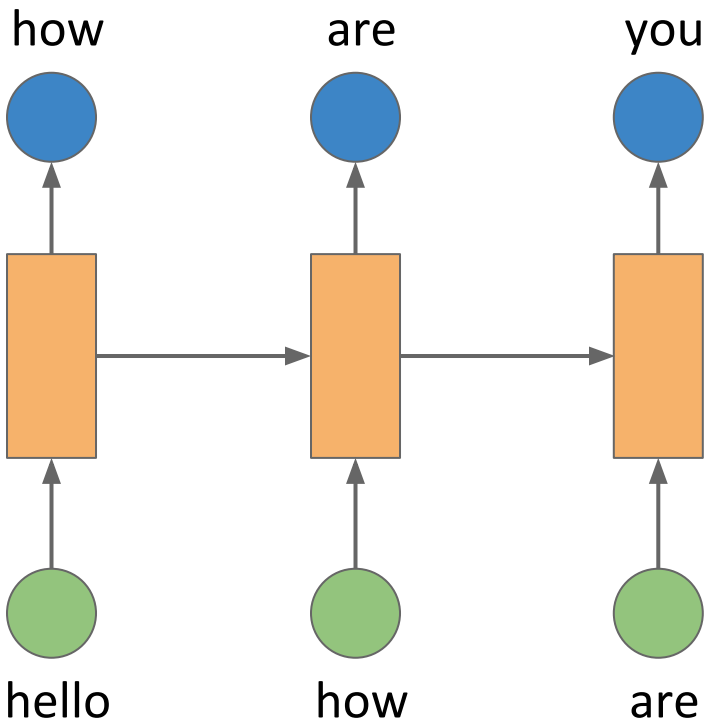

Given an input word and previous context, what is the next word?

# Sequence Generation



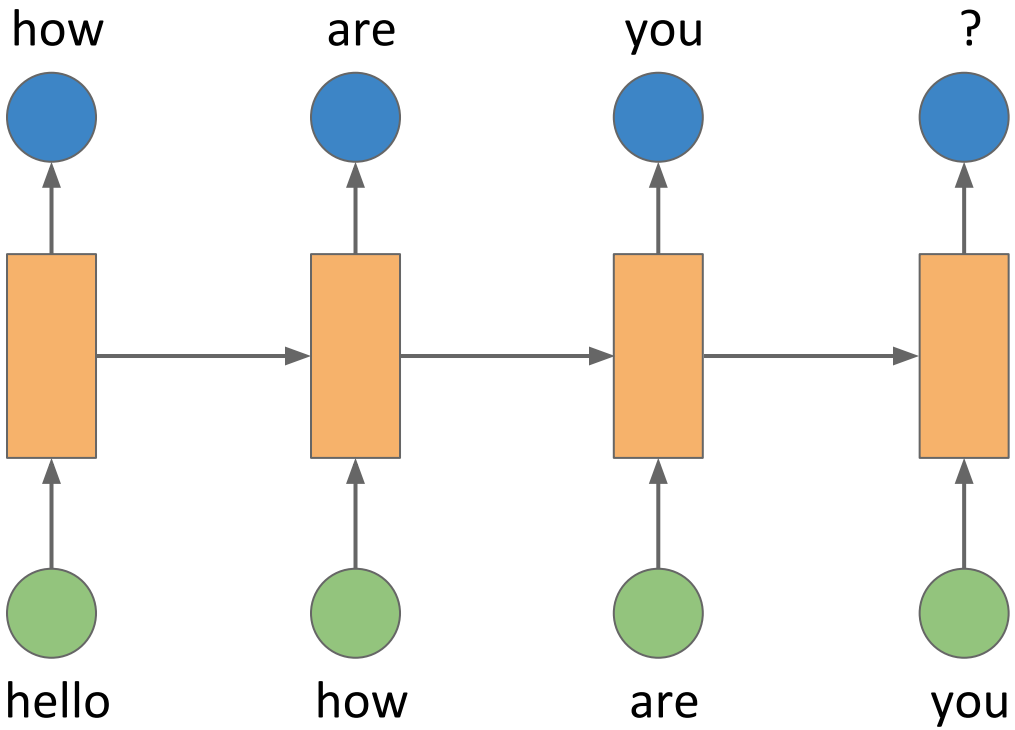

Process continues till an end of sentence is generated.

# Sequence Generation




Process continues till an end of sentence is generated.

# Sequence Generation



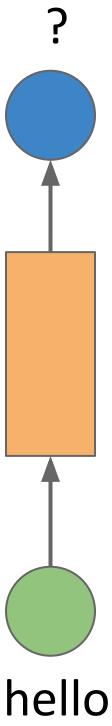

Process continues till an end of sentence is generated.

# Sequence Generation

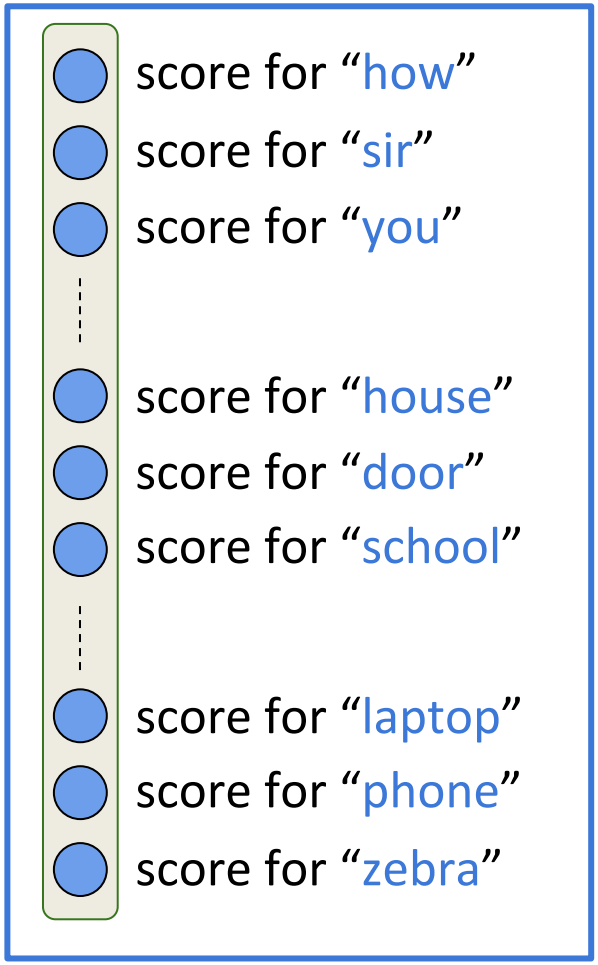
Also referred as time series problems where each input can be considered as a step in time

At every step, we score all possible next words and choose the most probable word

# Sequence Generation




Output




Choose the most probable next word

# Sequence Generation

Training vs Test

During training time, next word/label to predict is available *because we have the entire gold target sequence*

# Sequence Generation

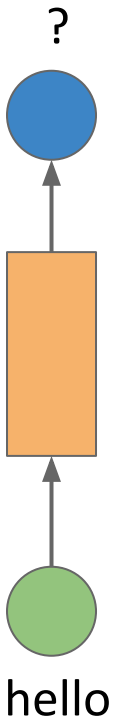
Training vs Test

During training time, next word/label to predict is available *because we have the entire gold target sequence*

At test time, we do not have gold labels, and our history is made up of predicted labels in the past

# Sequence Generation

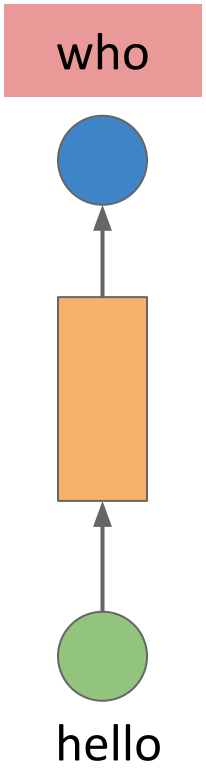
**Training** (full sequence is available)




hello | how | are | you | ?

# Sequence Generation

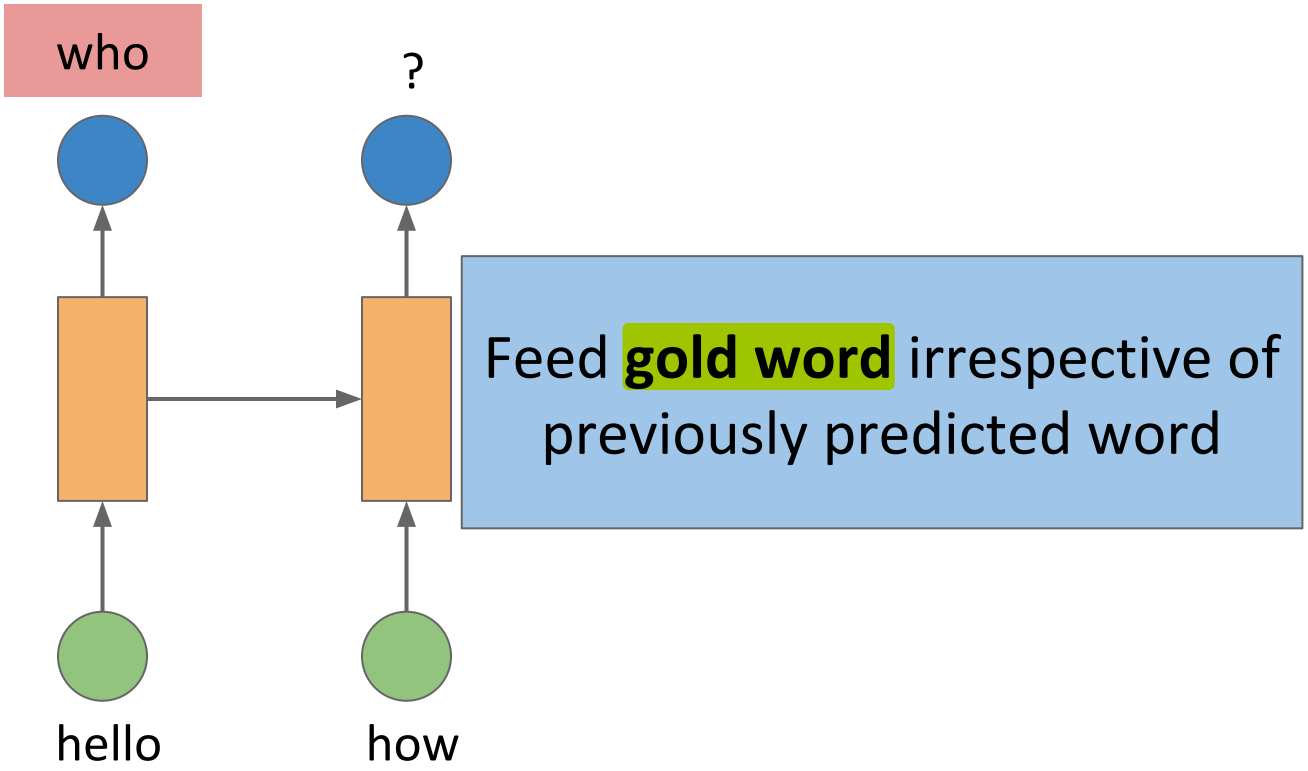
**Training** (full sequence is available)




hello | how | are | you | ?

# Sequence Generation

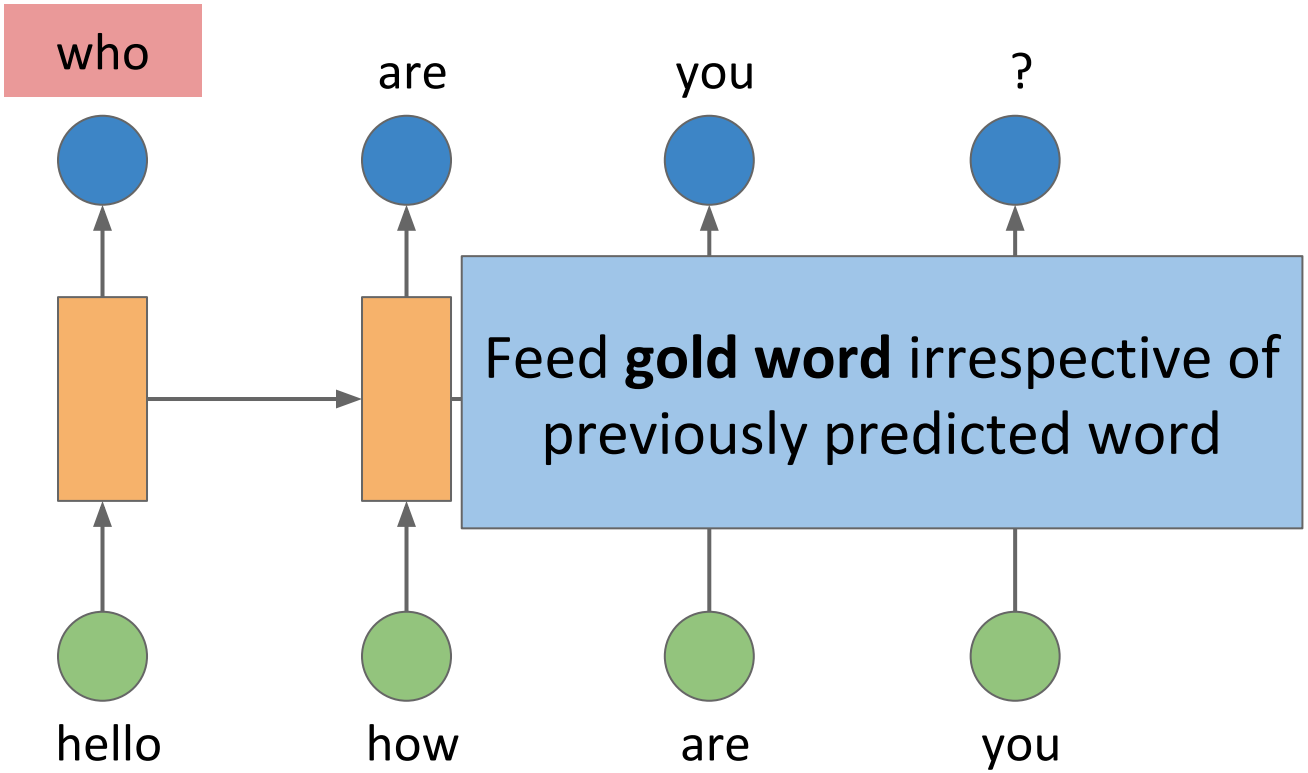
**Training** (full sequence is available)




hello | how | are | you | ?

# Sequence Generation

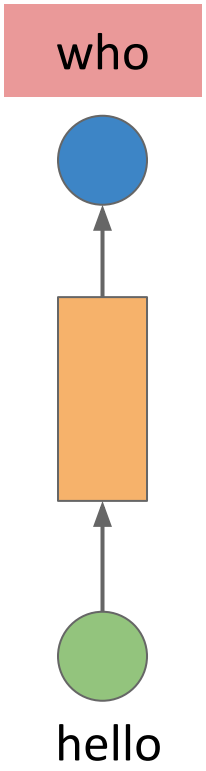
**Training** (full sequence is available)




hello | how | are | you | ?

# Sequence Generation

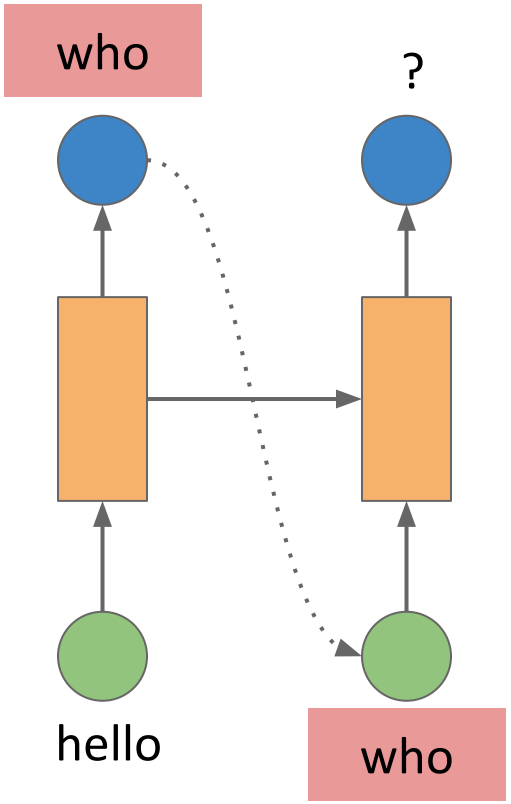
**Testing** (sequence is *not* available)




At **test** time, feed **previously** predicted word

# Sequence Generation

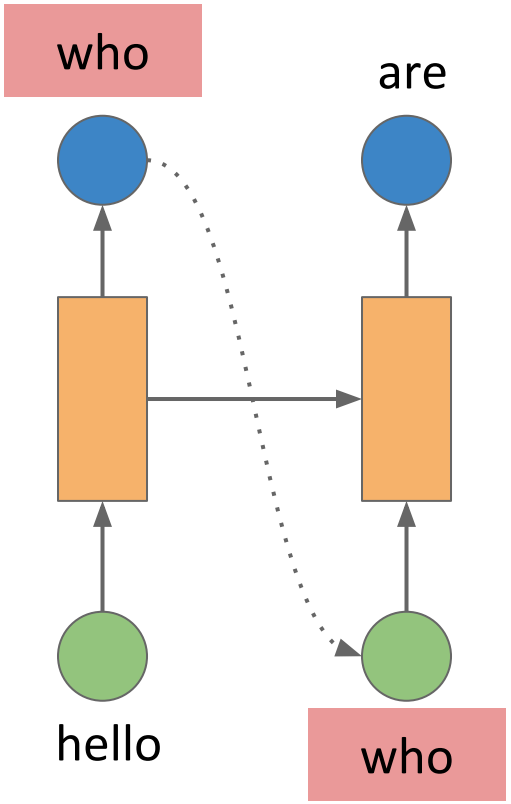
**Testing** (sequence is *not* available)




At **test** time, feed **previously** predicted word

# Sequence Generation

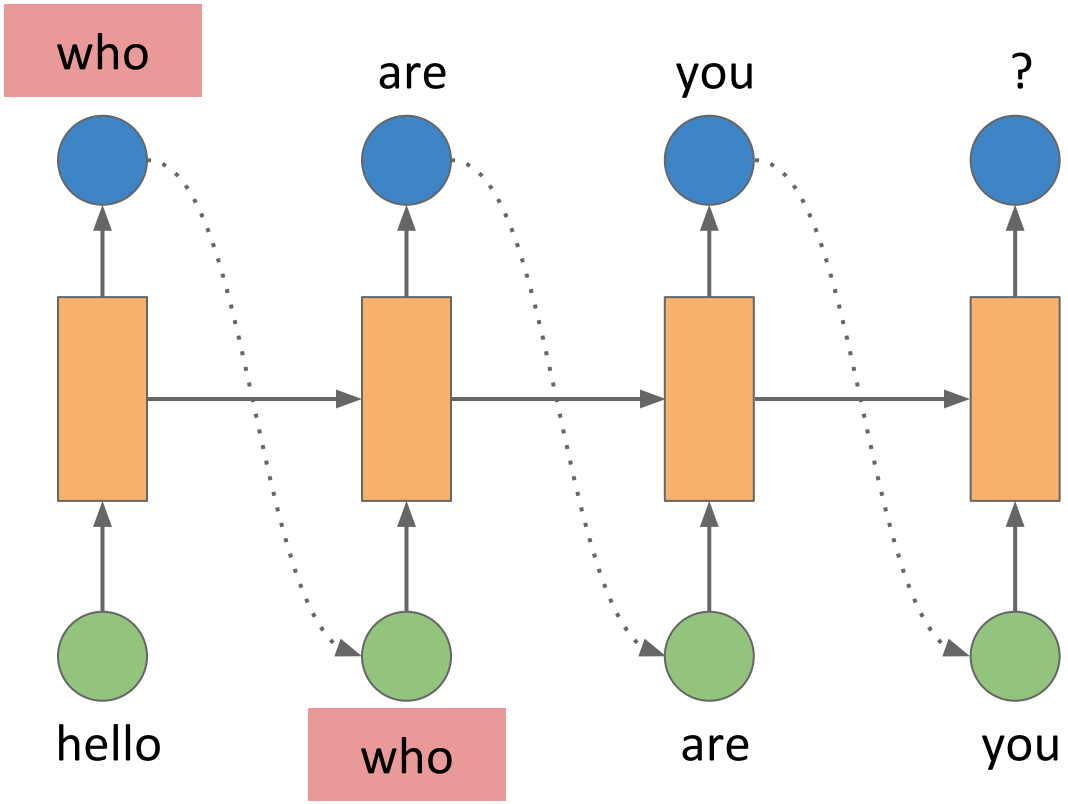
**Testing** (sequence is *not* available)




At **test** time, feed **previously** predicted word

# Sequence Generation

**Testing** (sequence is *not* available)



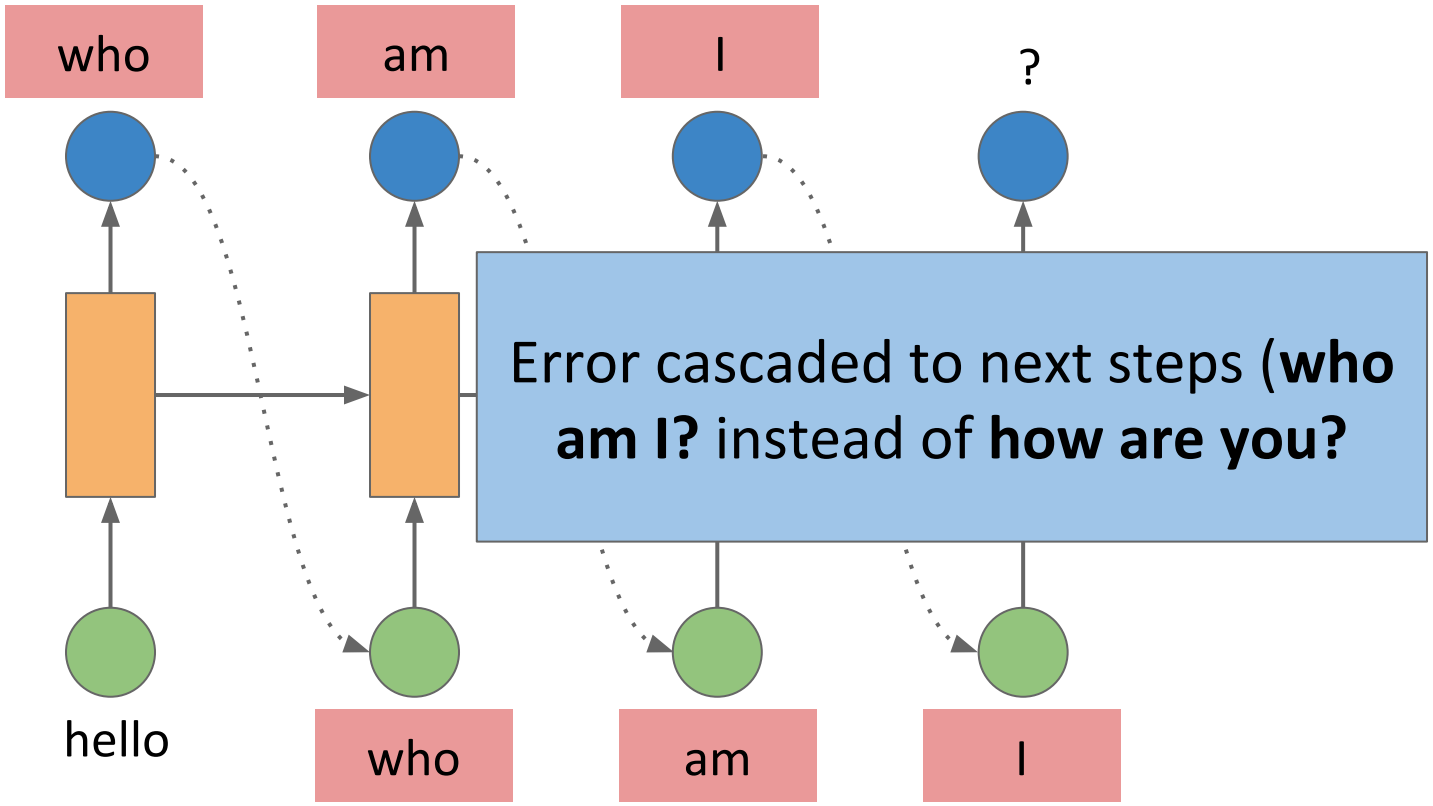

At **test** time, feed **previously** predicted word

# Sequence Generation

Training vs. Test

- At training time, gold words are used as input irrespective of the previously predicted word
- At test time, word predicted at previous step is used as input in the next step
  - A prediction error at one step can affect next predictions

# Sequence Generation




Testing

# Applications of RNNs

Task Variations

- Sequence classification
- Per timestep classification
- Sequence generation

Architecture Variations

- Bidirectional RNN
- Averaging vs. Summary vector
- Multilayer RNN

# Bidirectional RNN

- In sequence classification problems, we learn a summary vector while reading a sequence from left to right
  - considering left context only
- But, a word can be dependent on both the left context and the right context

# Bidirectional RNN

- In sequence classification problems, we learn a summary vector while reading a sequence from left to right
  - considering left context only
- But, a word can be dependent on both the left context and the right context

**Solution**

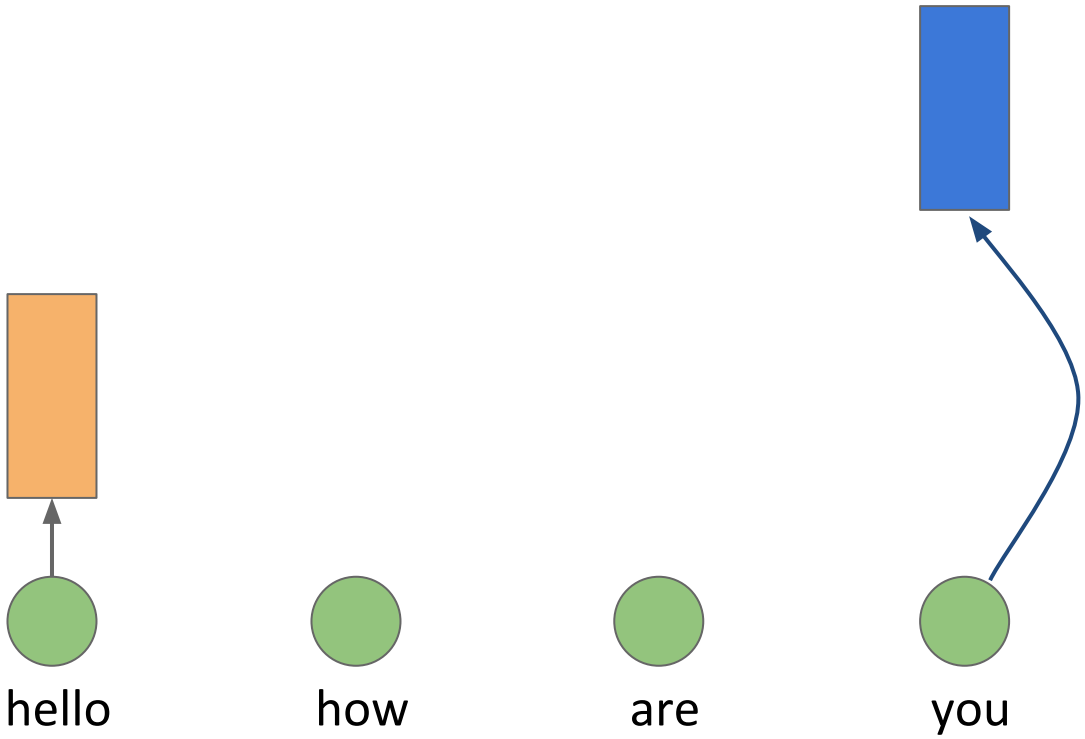
Read the sequence from left to right and from right to left

# Bidirectional RNN

- Given a sequence as input

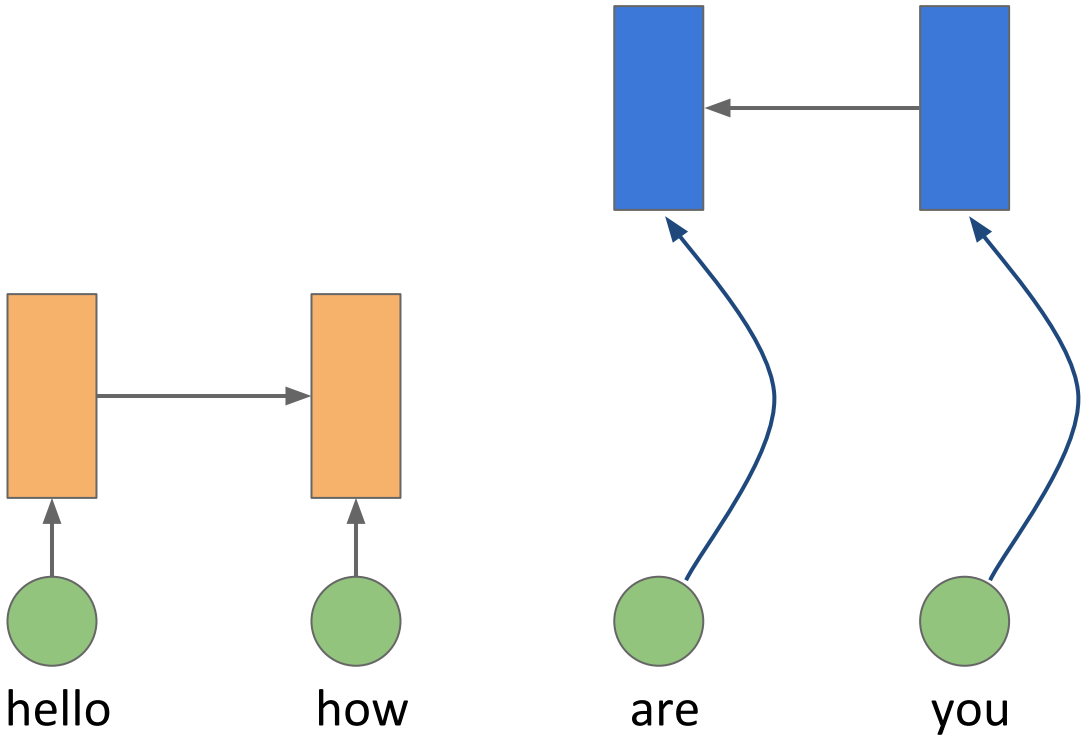
# Bidirectional RNN

- Use two RNNs - one to read sequence from left to right and other from right to left



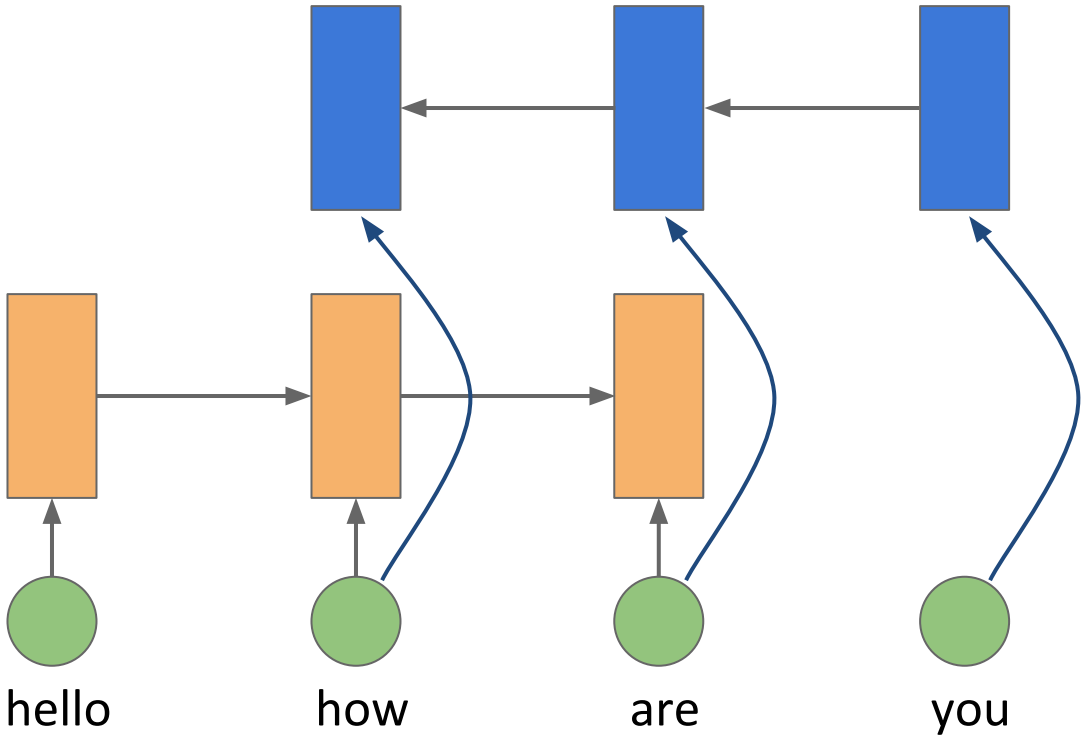
# Bidirectional RNN

- Use two RNNs - one to read sequence from left to right and other from right to left



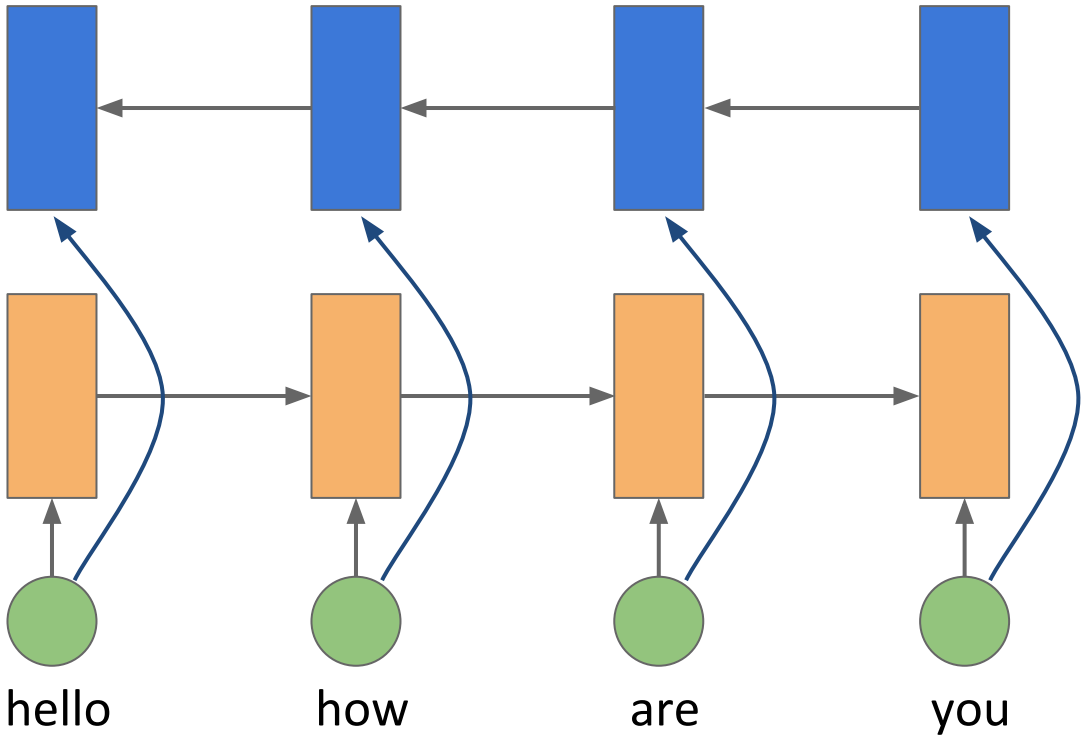
# Bidirectional RNN

- Use two RNNs - one to read sequence from left to right and other from right to left



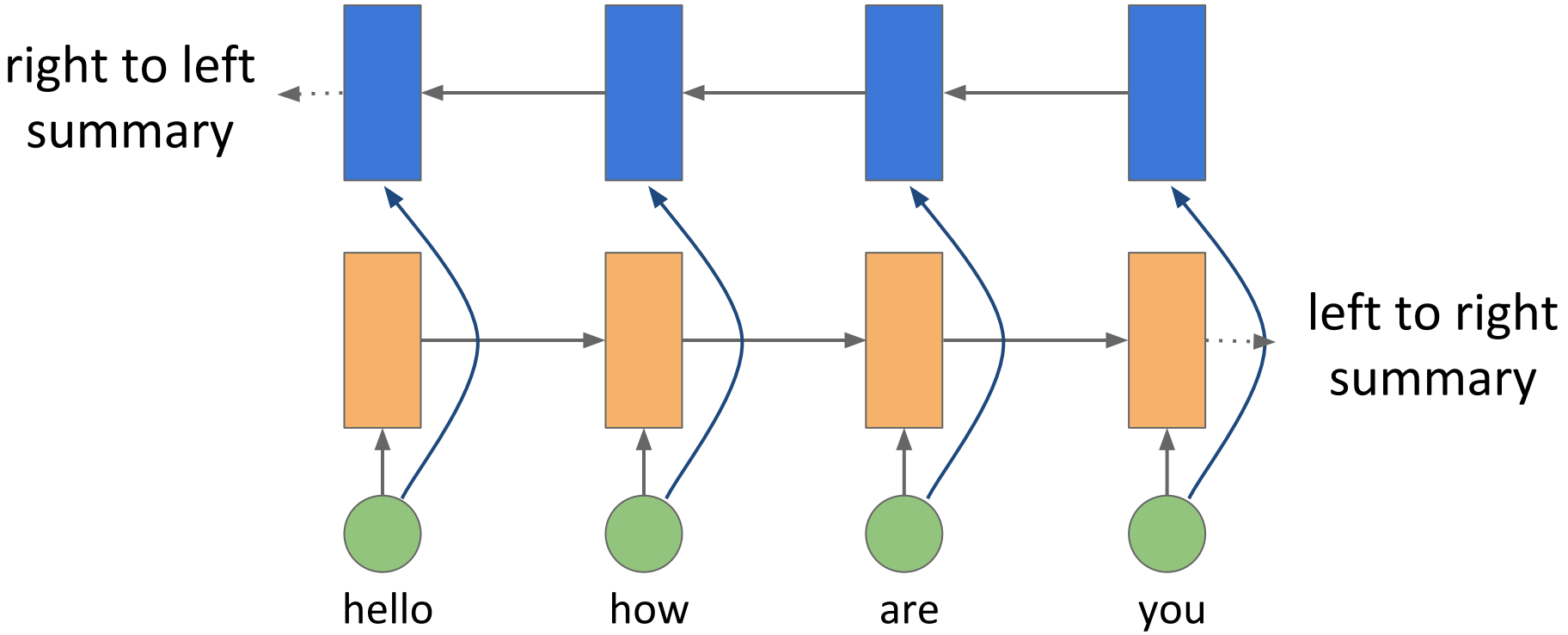
# Bidirectional RNN

- Use two RNNs - one to read sequence from left to right and other from right to left

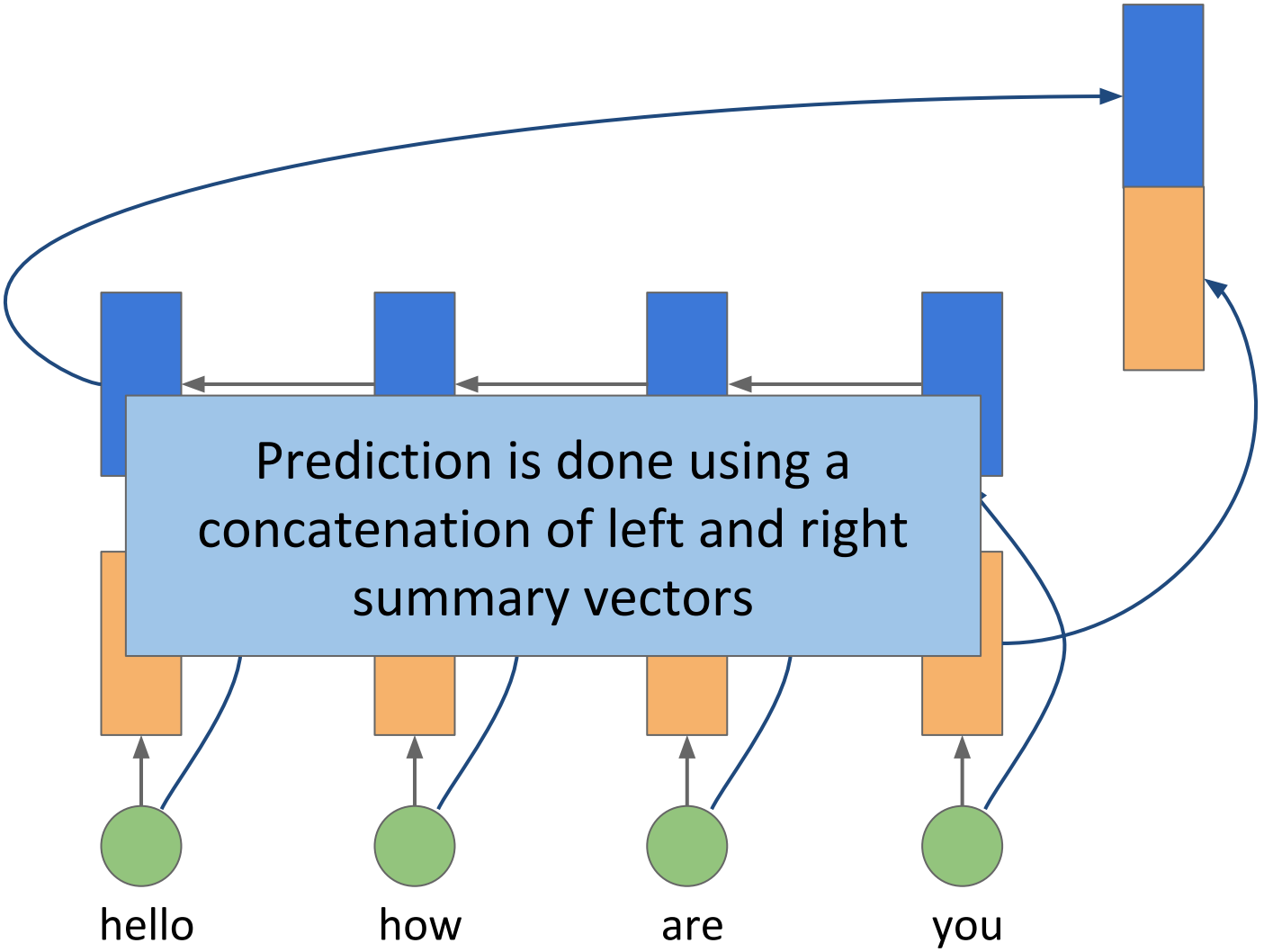


# Bidirectional RNN

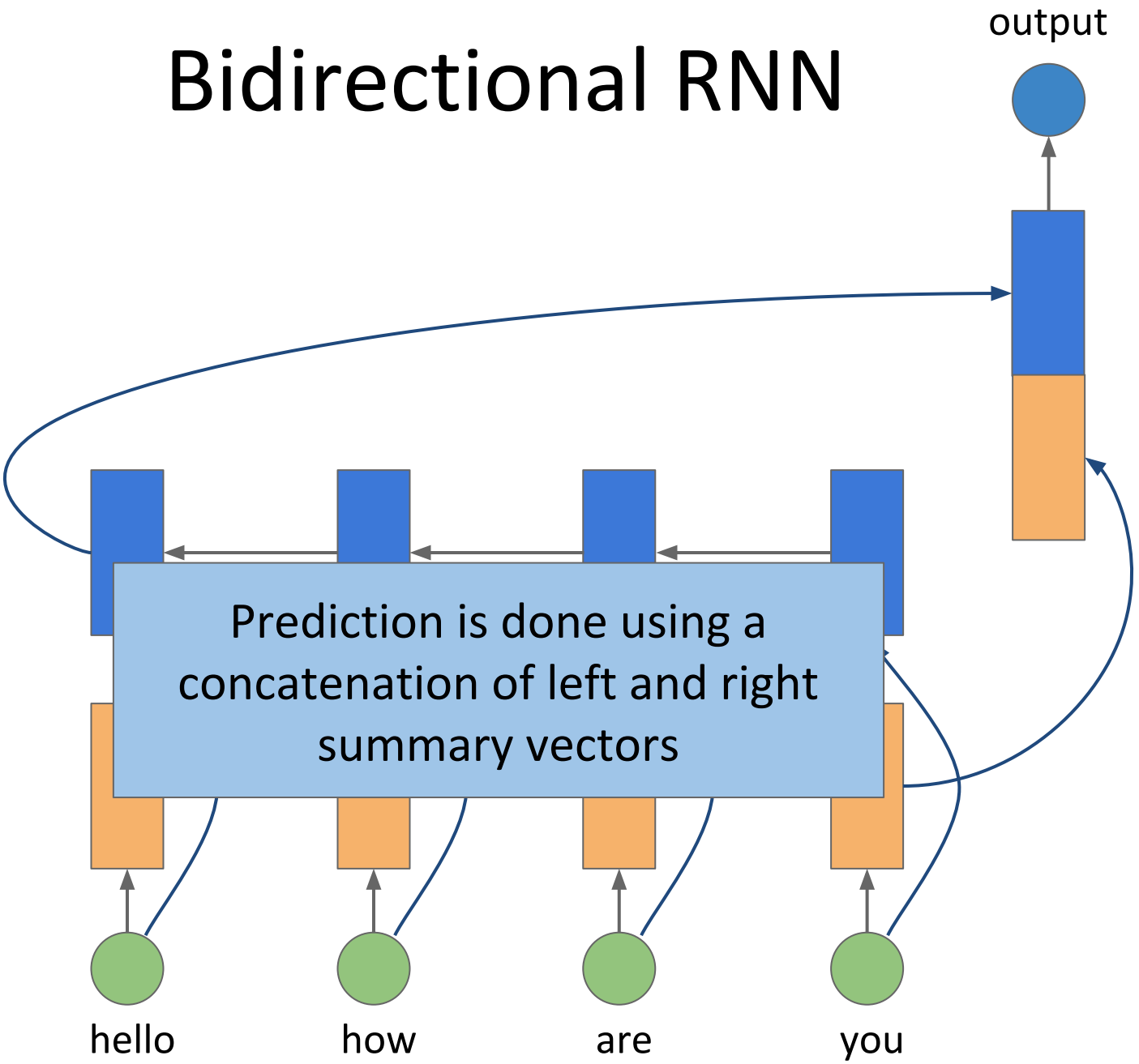
- Use two RNNs - one to read sequence from left to right and other from right to left



# Bidirectional RNN



# Bidirectional RNN



# Bidirectional RNN

Advantages

- Consider both left and right contextual information
- Performs better than unidirectional RNN

Practical consideration

- Number of parameters is doubled
- Training time is increased

# Applications of RNNs

Task Variations

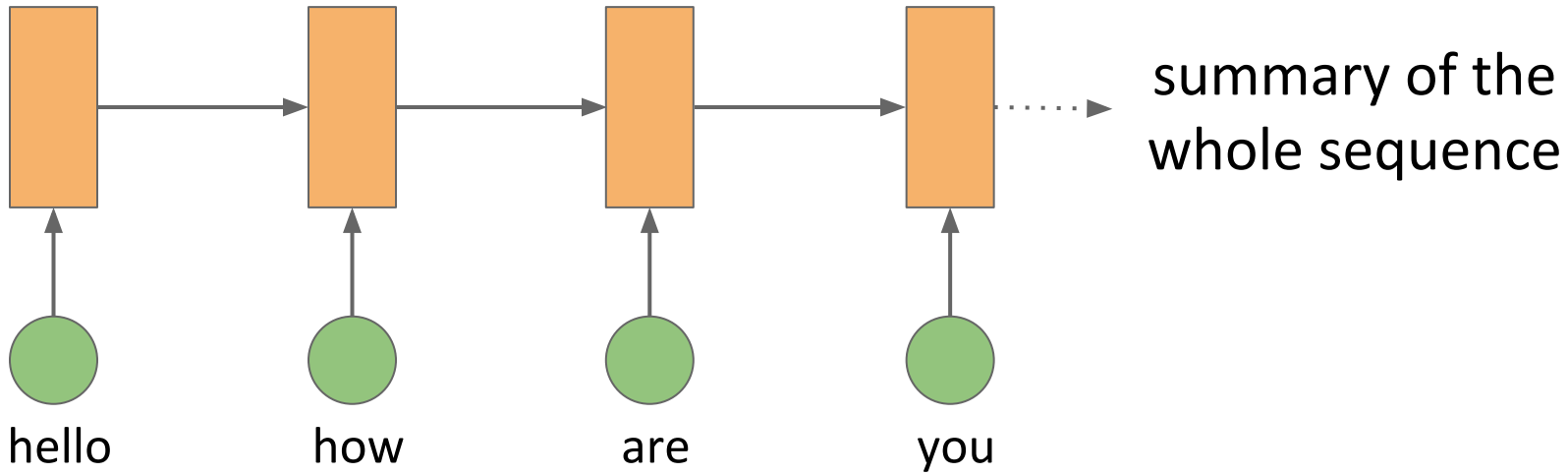
- Sequence classification
- Per timestep classification
- Sequence generation

Architecture Variations

- Bidirectional RNN
- Averaging vs. Summary vector
- Multilayer RNN

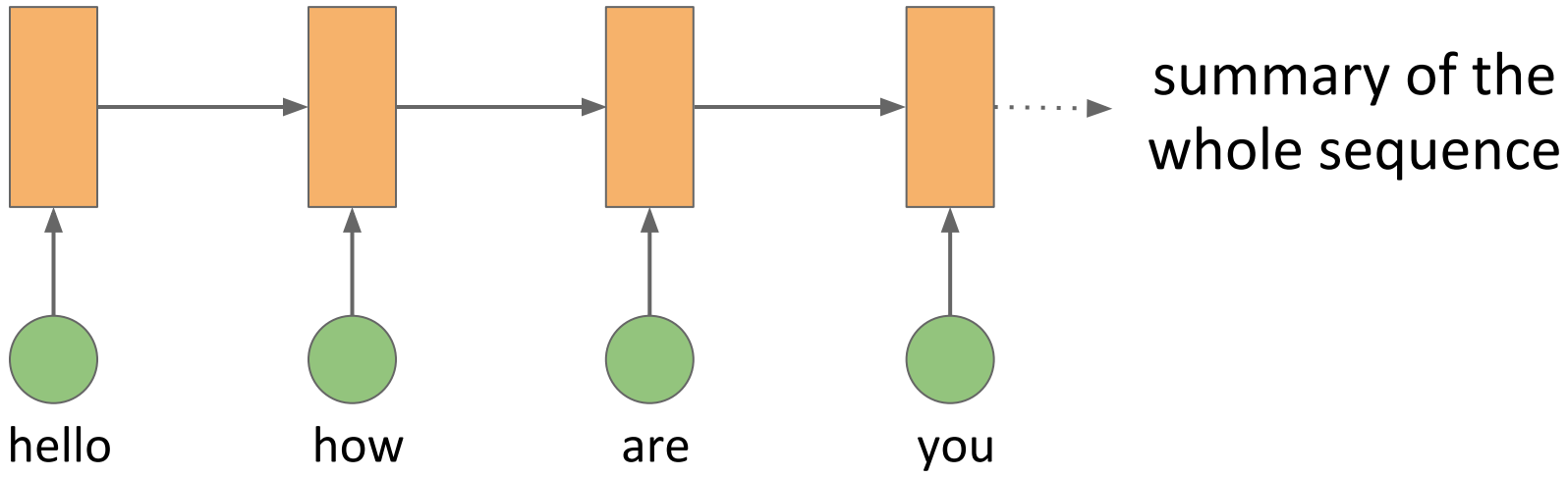
# Average vs. Single Summary Vector

- A summary vector may have forgotten the information seen at the very beginning of the sequence



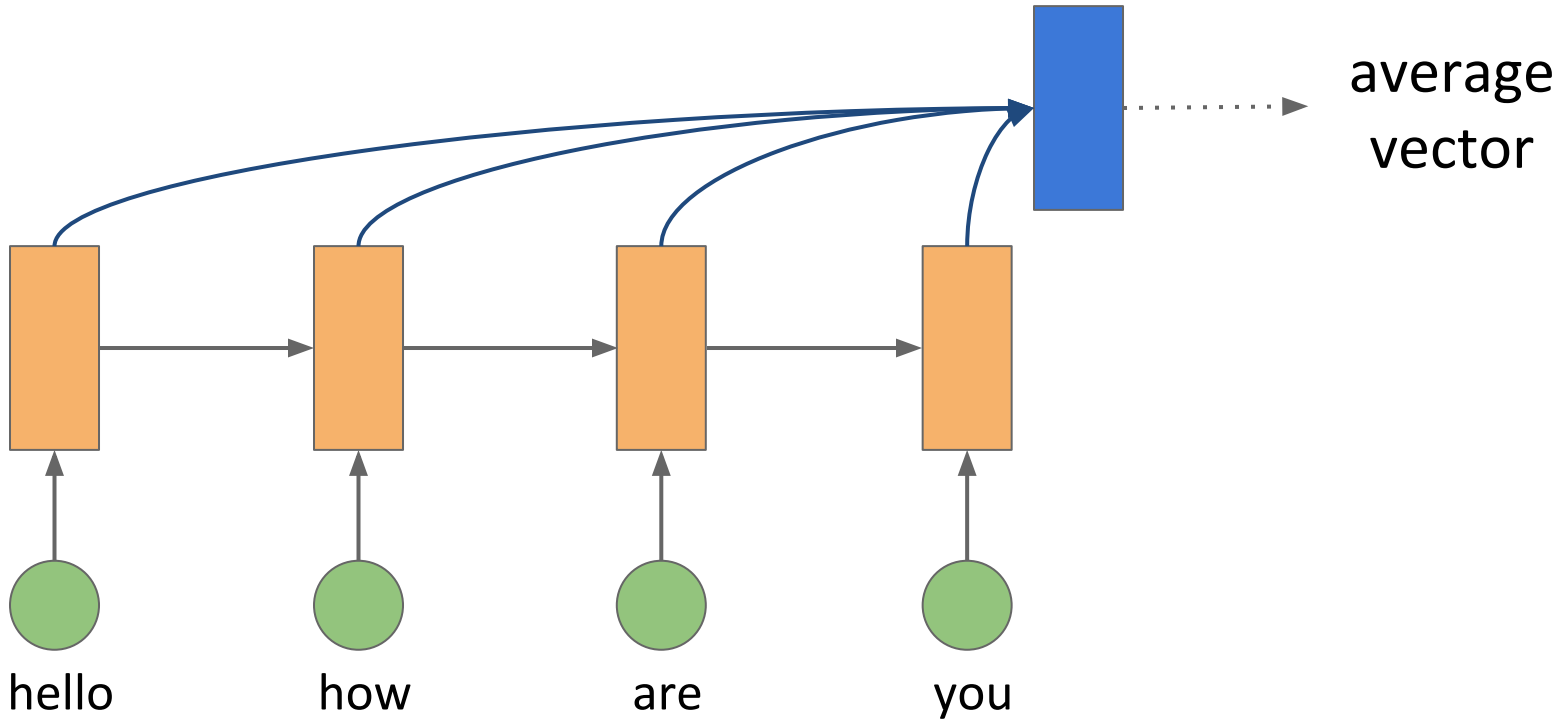
# Average vs. Single Summary Vector

- An alternate is to take an average of vectors at every time step

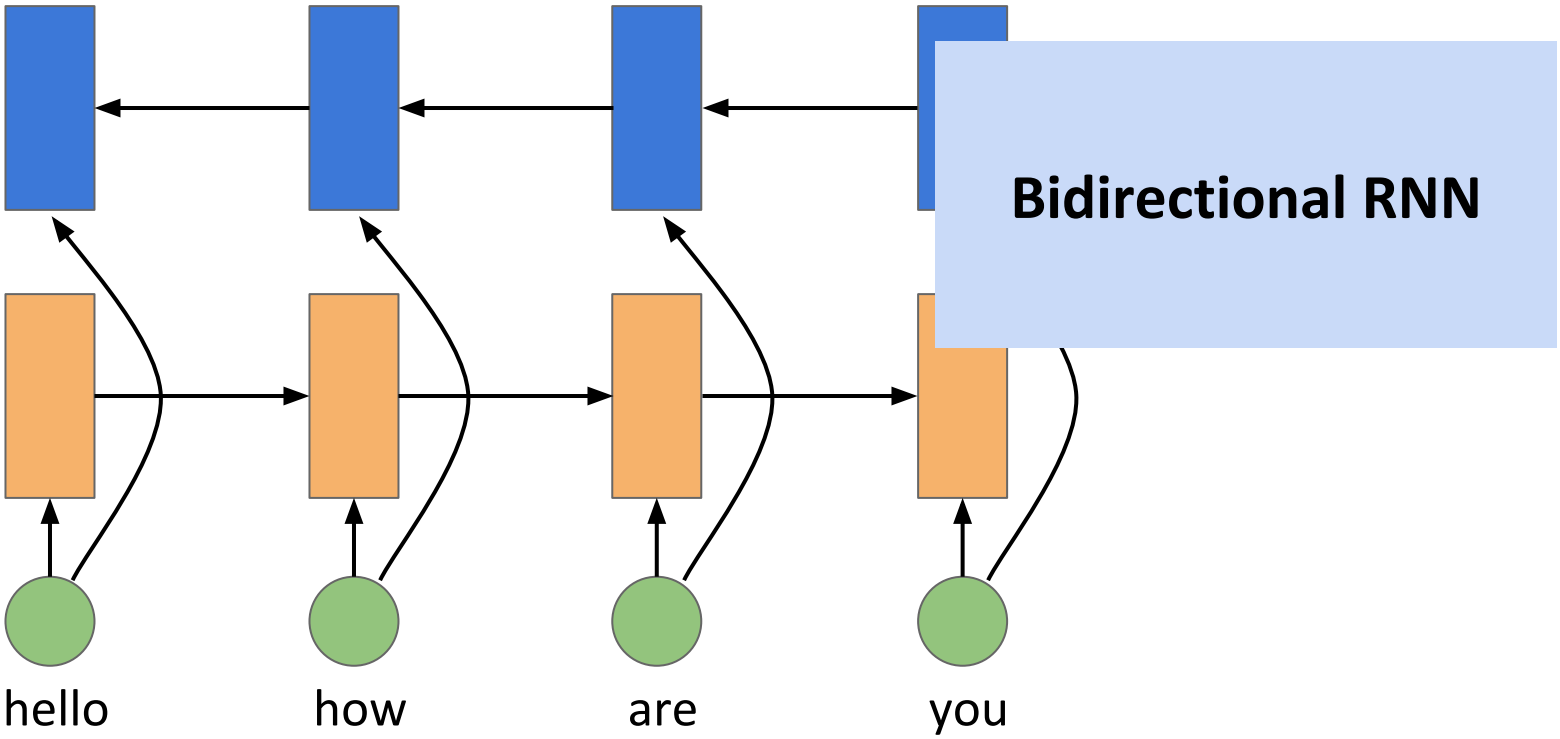


# Average vs. Single Summary Vector

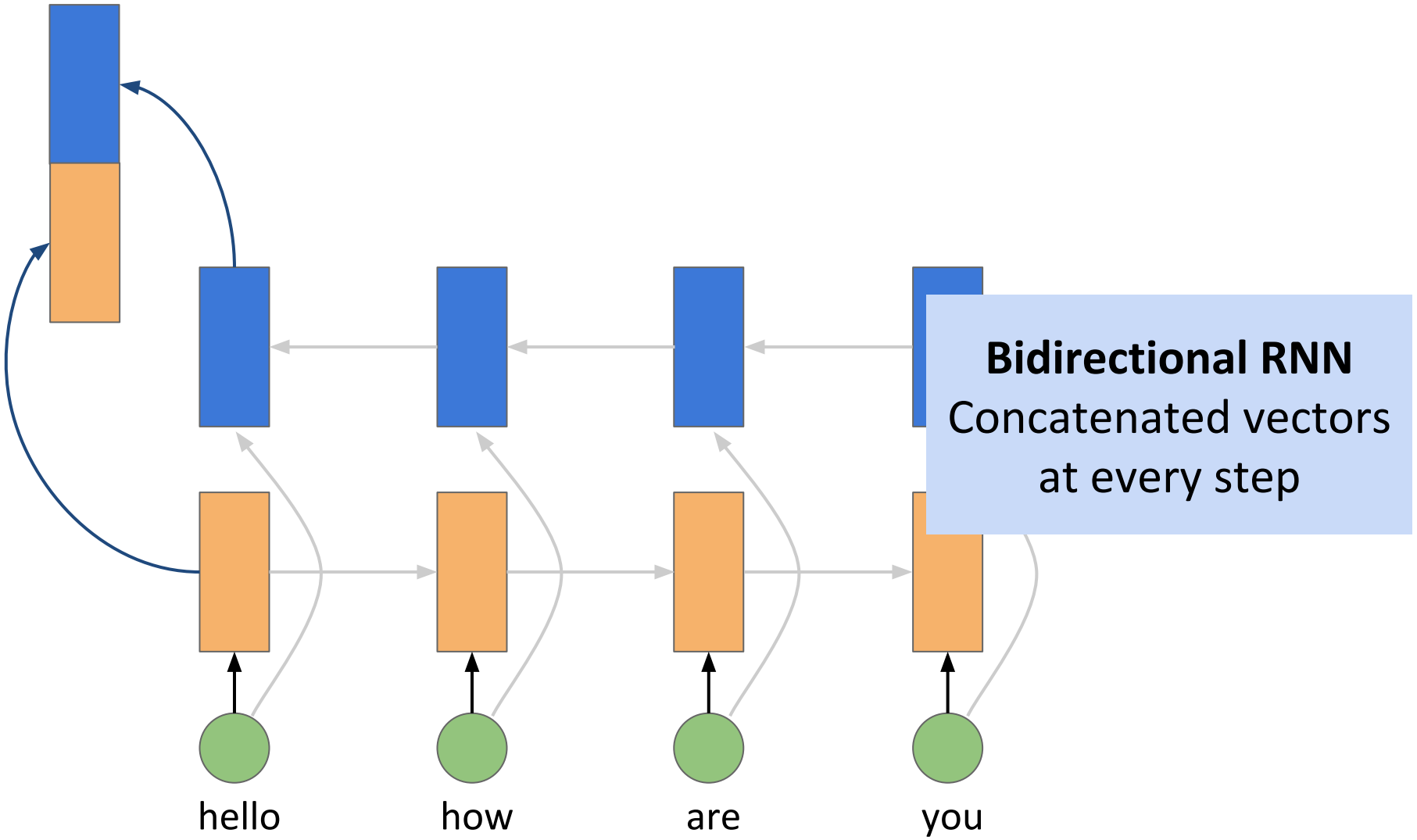
- An alternate is to take an average of vectors at every time step



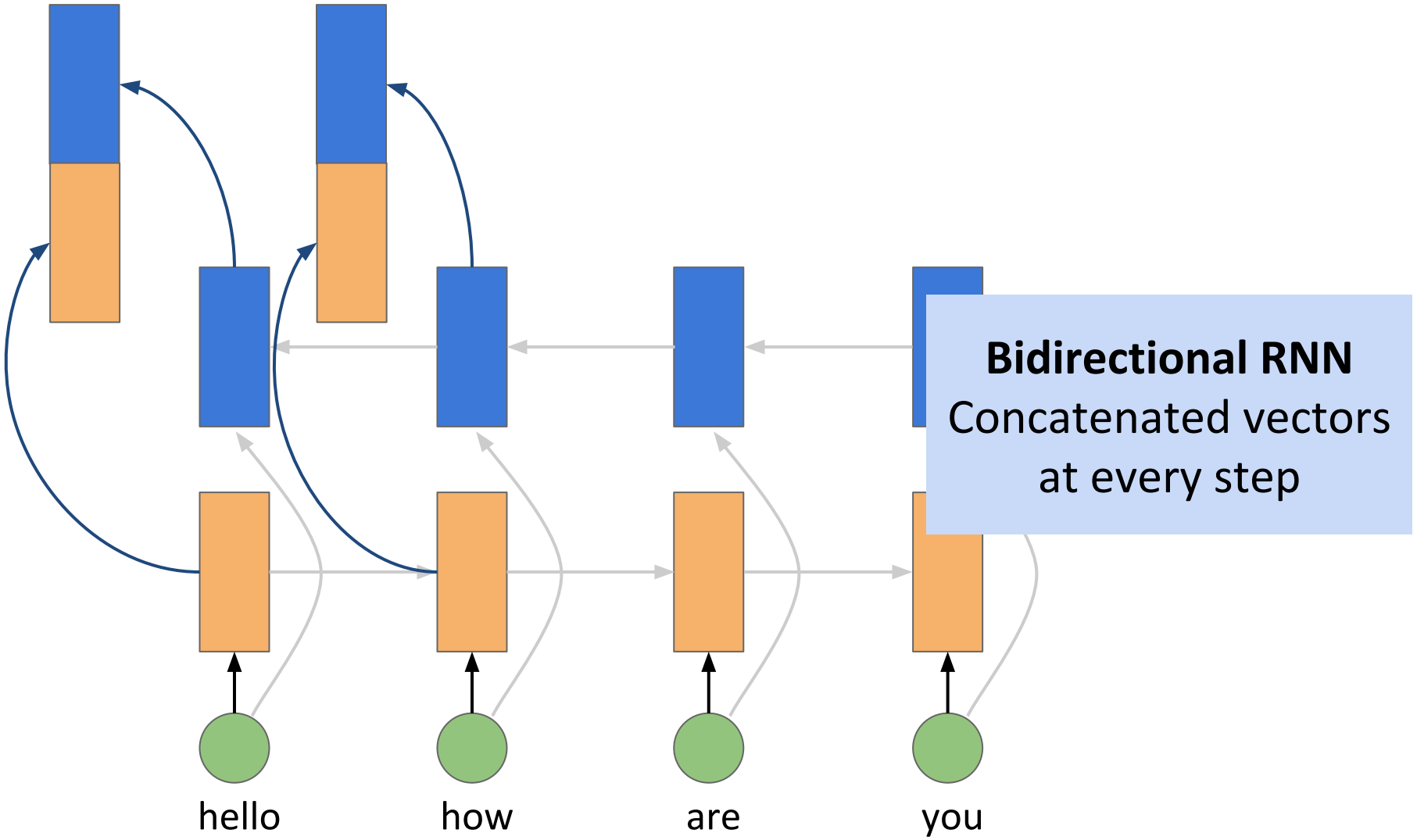
# Average vs. Single Summary Vector



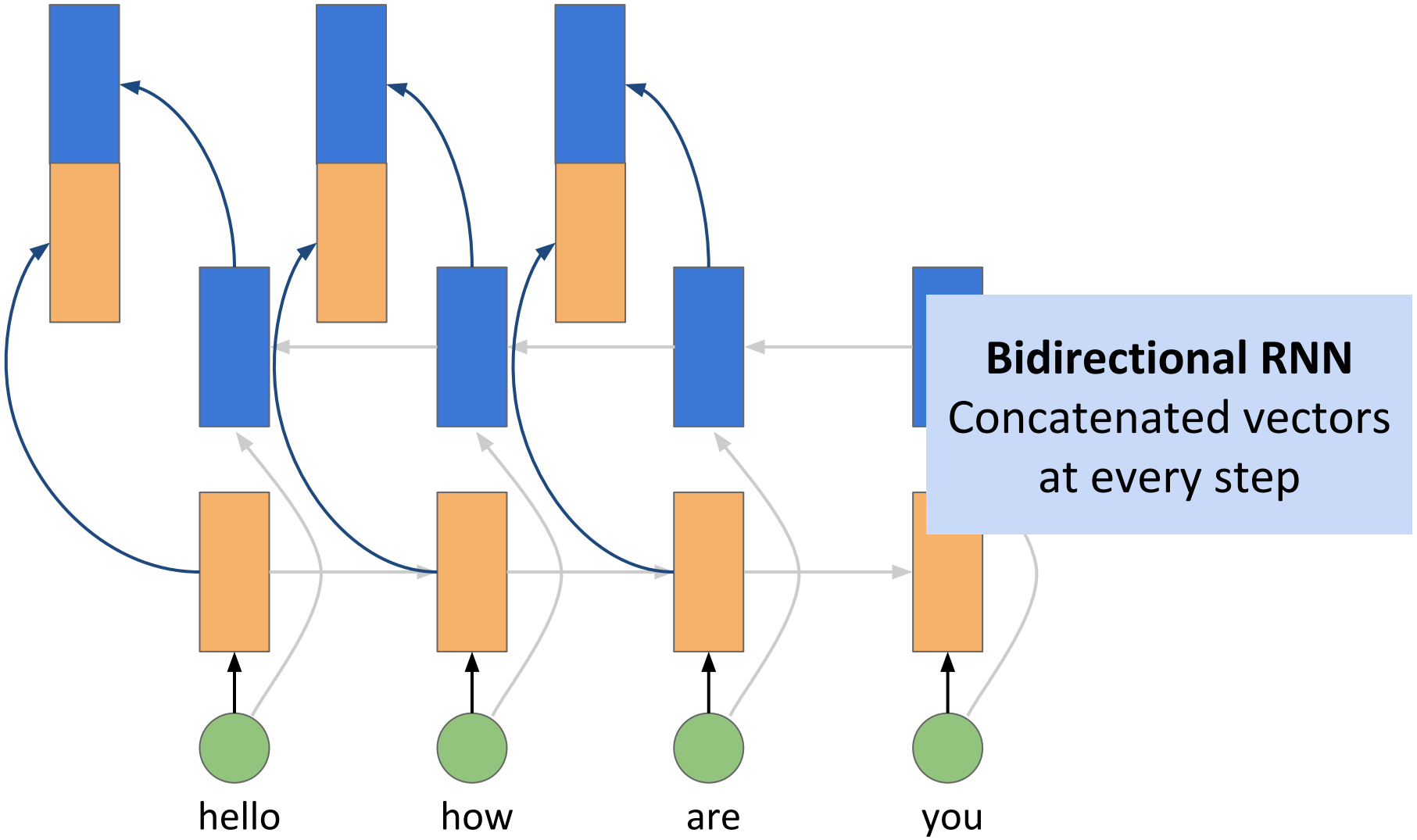
# Average vs. Single Summary Vector



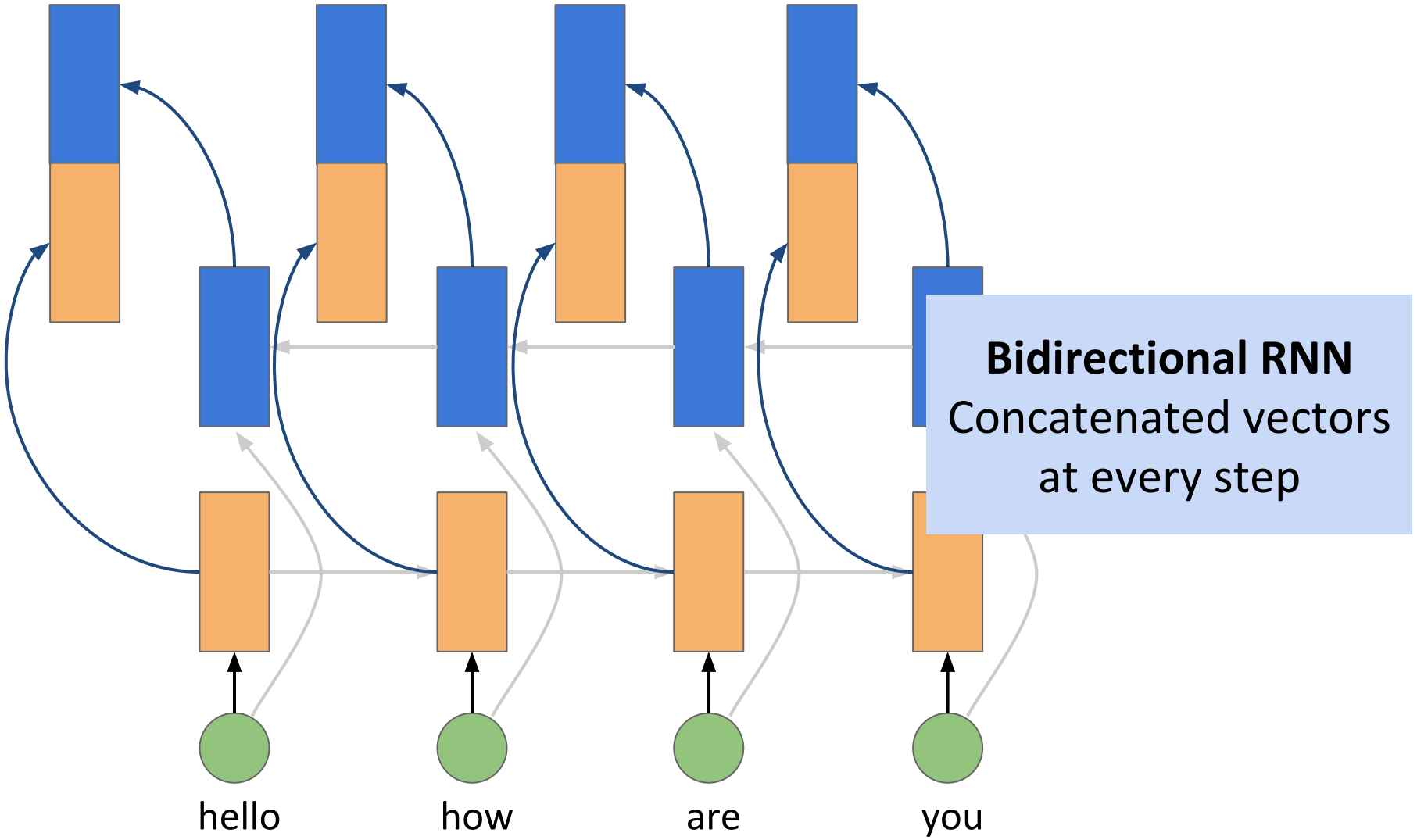
# Average vs. Single Summary Vector



# Average vs. Single Summary Vector

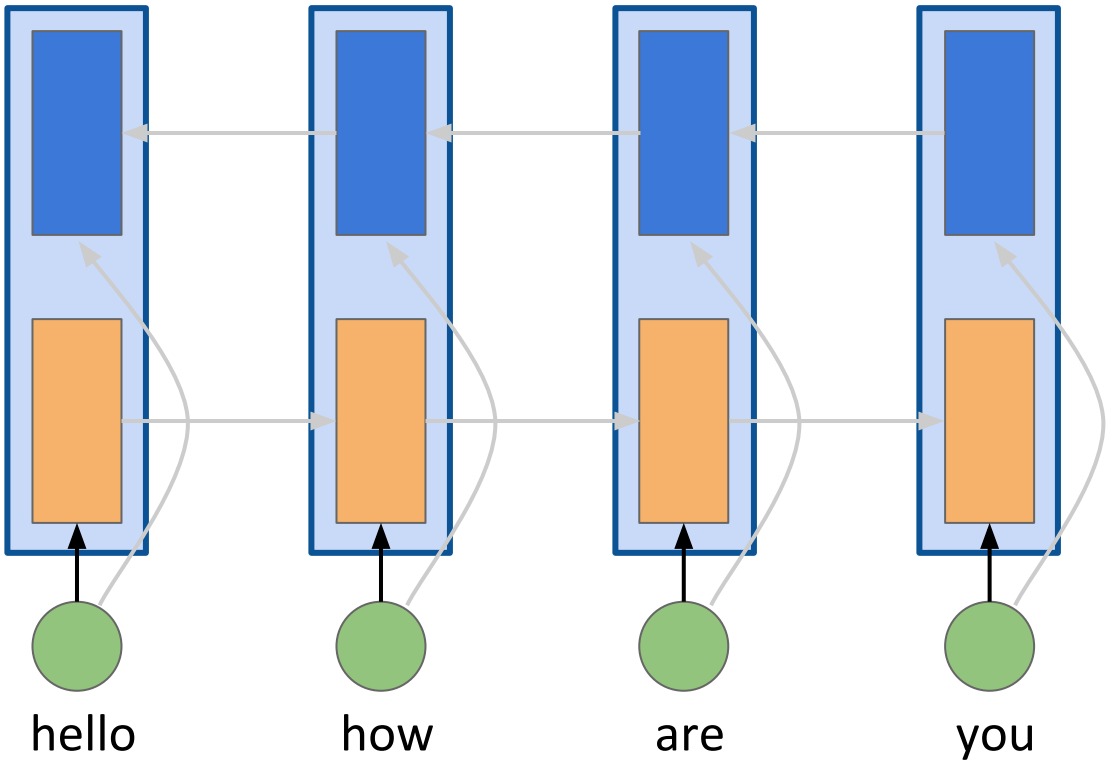


# Average vs. Single Summary Vector

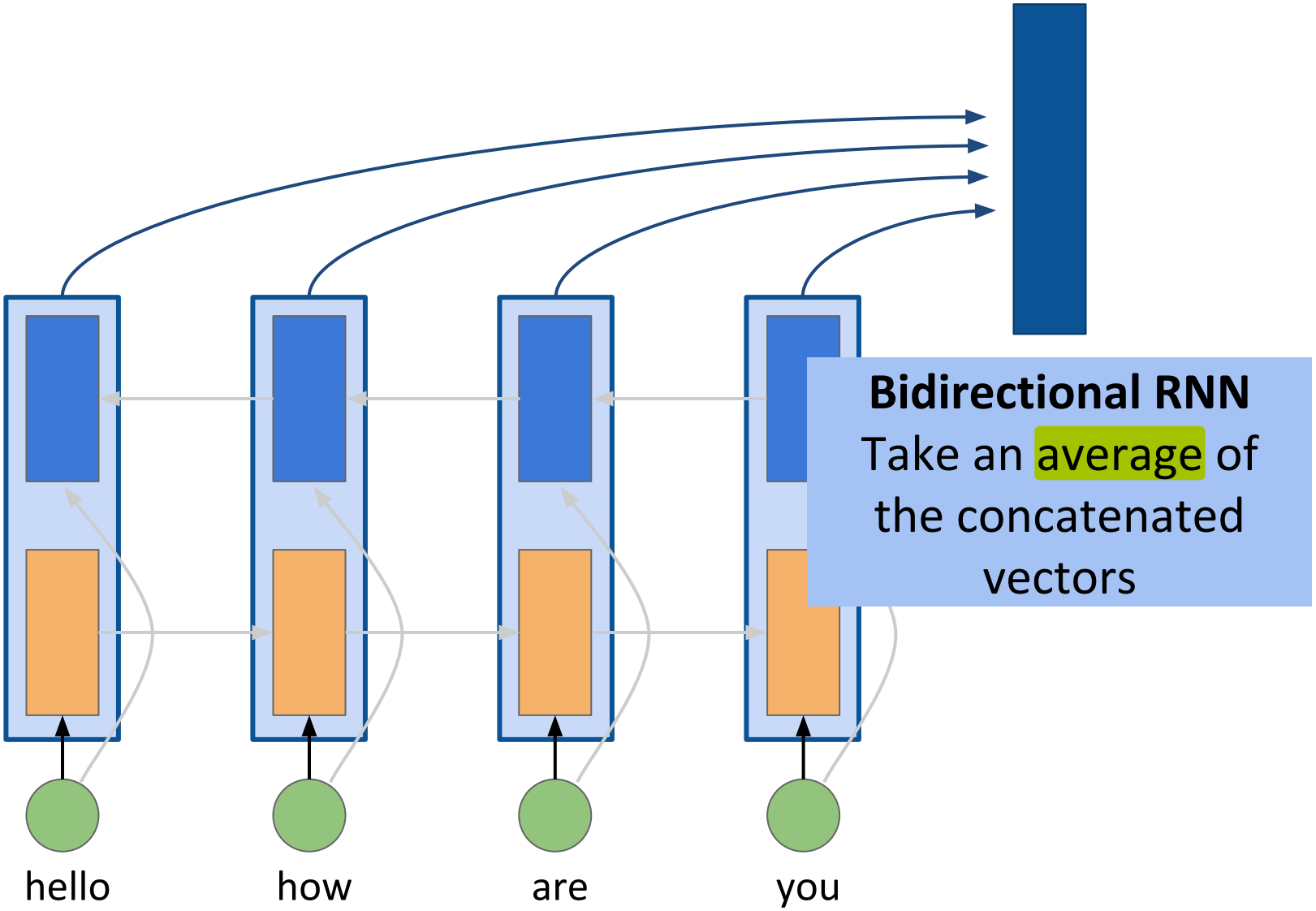


# Average vs. Single Summary Vector

**Bidirectional RNN**
Concatenated vectors at every step



# Average vs. Single Summary Vector



# Average vs. Summary Vector

- Average vector has an advantage of explicitly considering the information available at every timestep

# Applications of RNNs

Task Variations

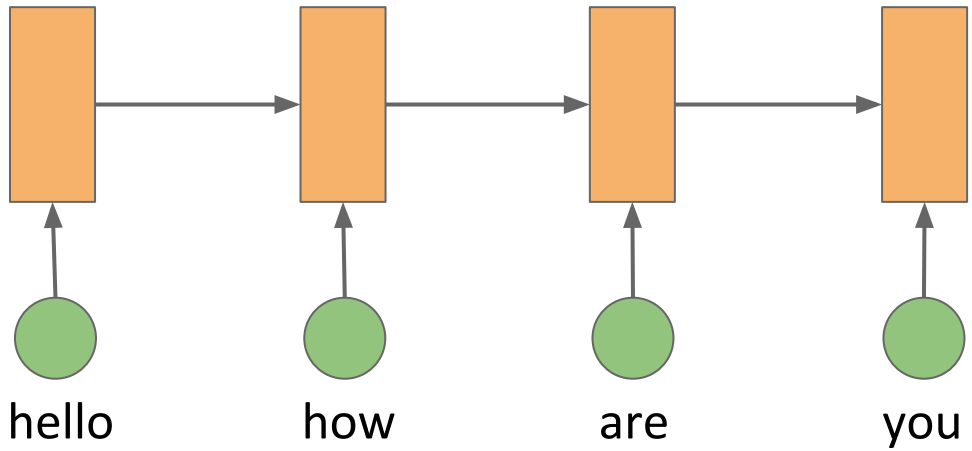
- Sequence classification
- Per timestep classification
- Sequence generation

Architecture Variations

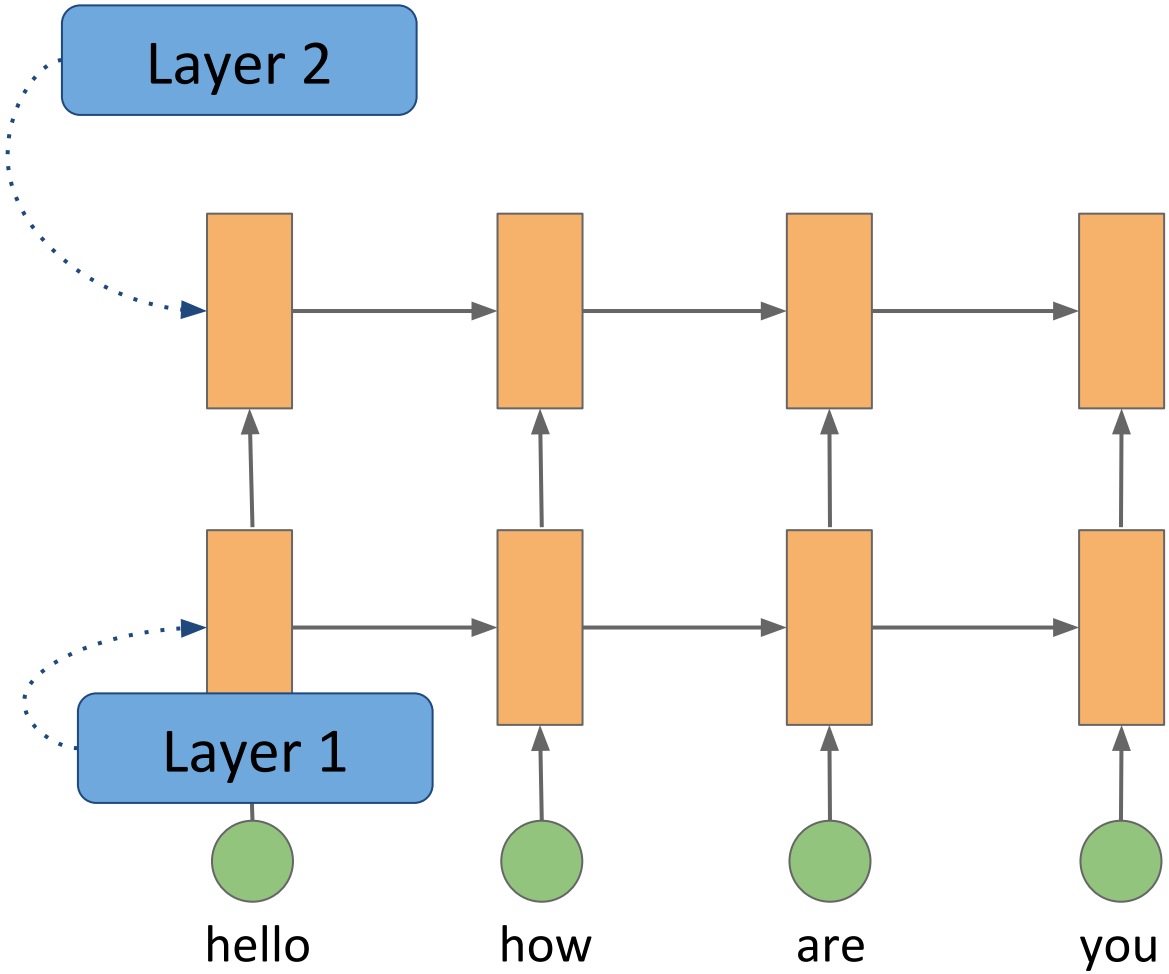
- Bidirectional RNN
- Averaging vs. Summary vector
- Multilayer RNN

# Multilayer RNN

- Add more layers to the model

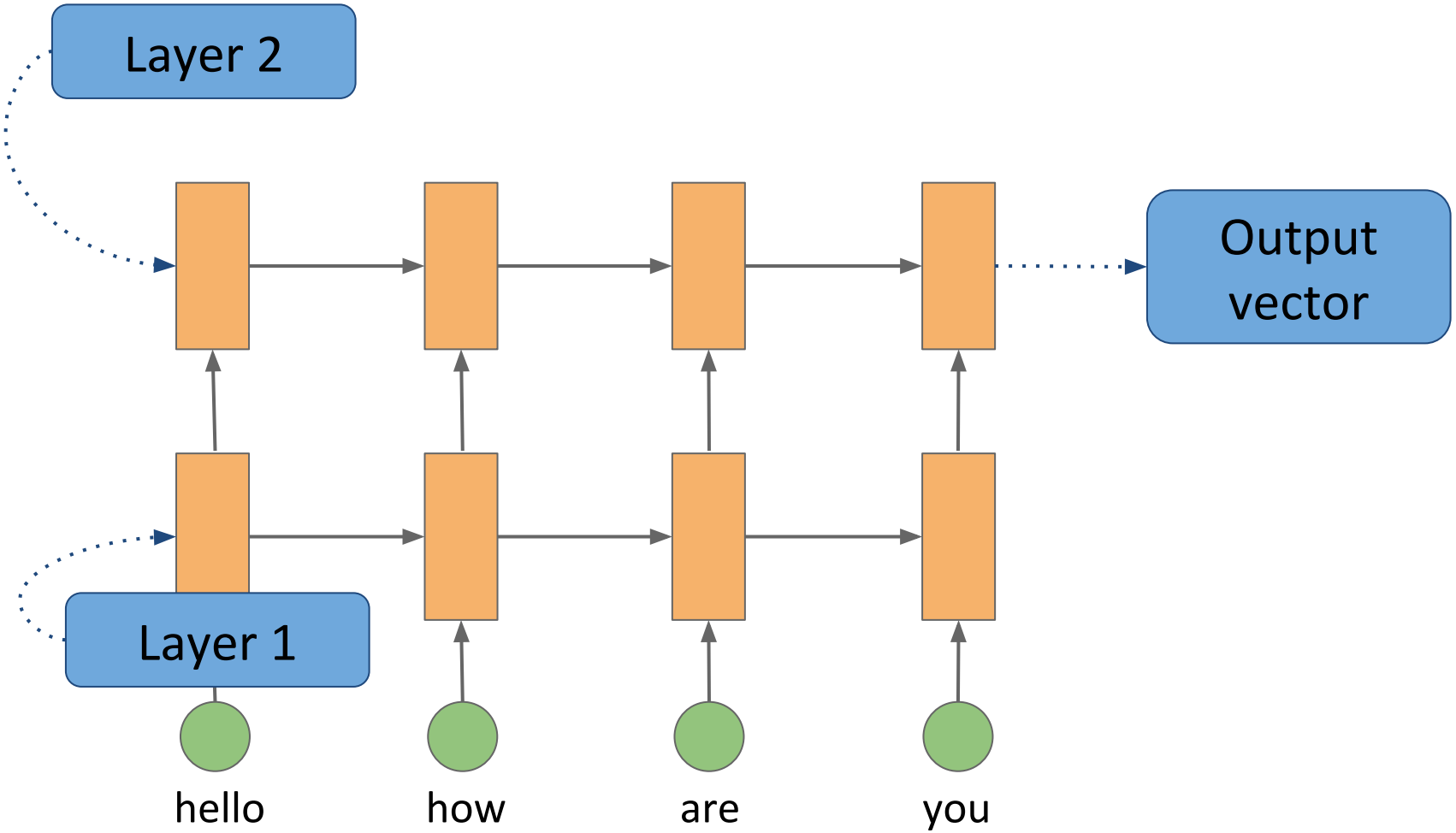


# Multilayer RNN

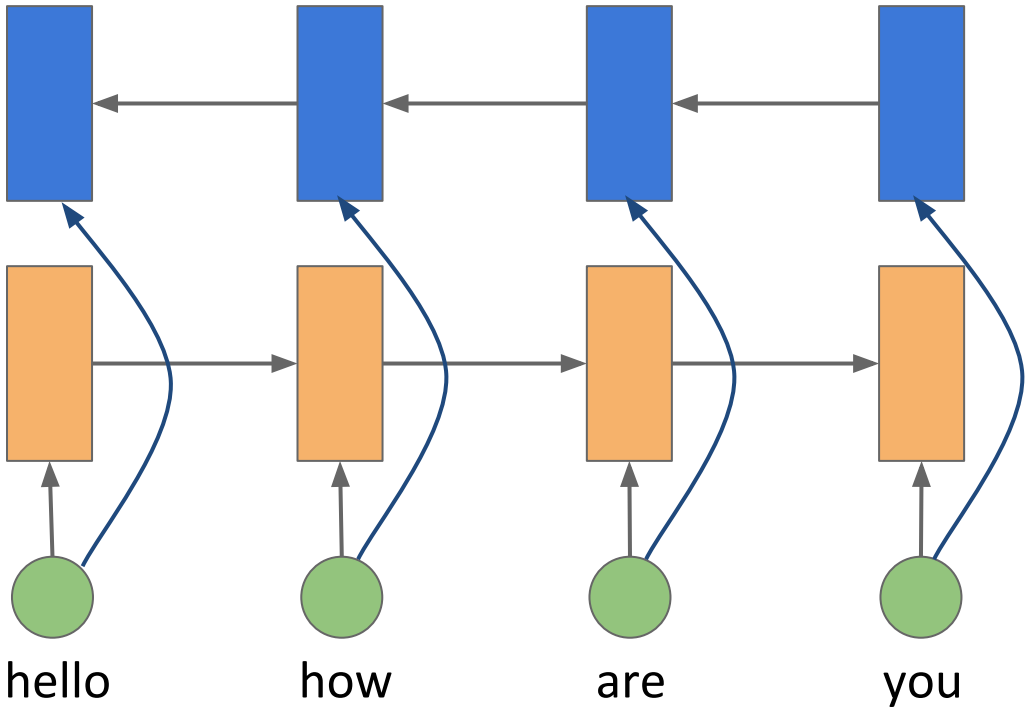



Each time step of layer 1 serves as input to layer 2

# Multilayer RNN

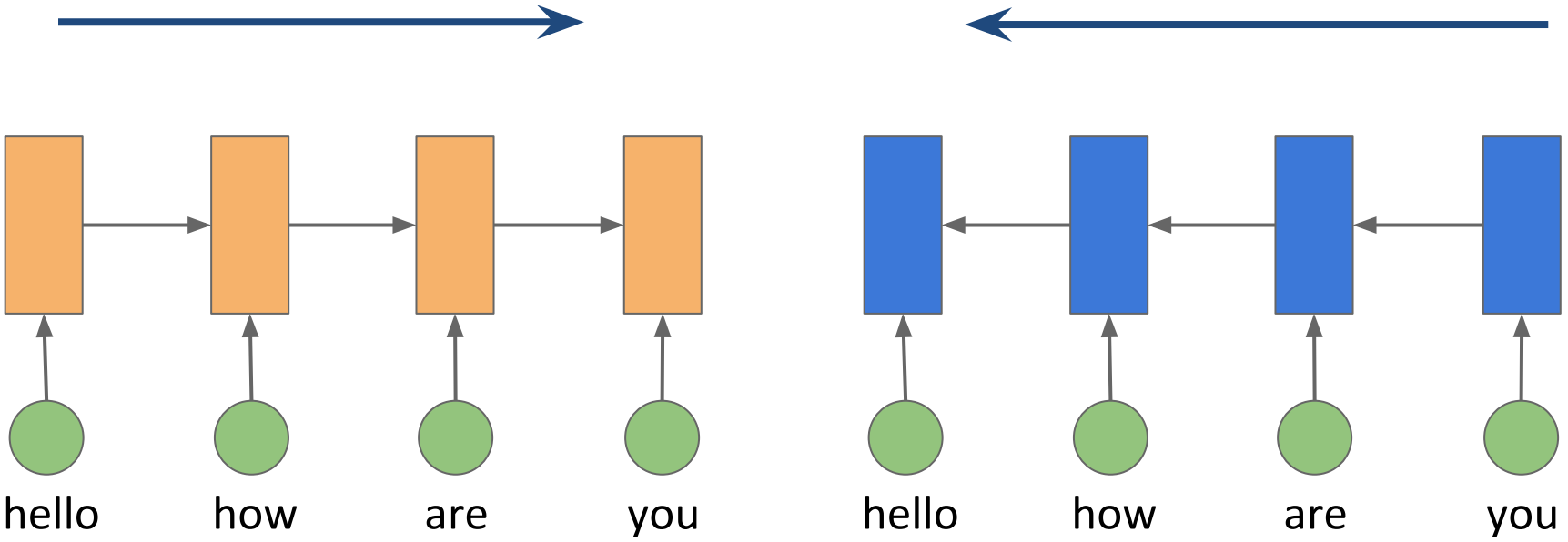


# Multilayer Bidirectional RNN



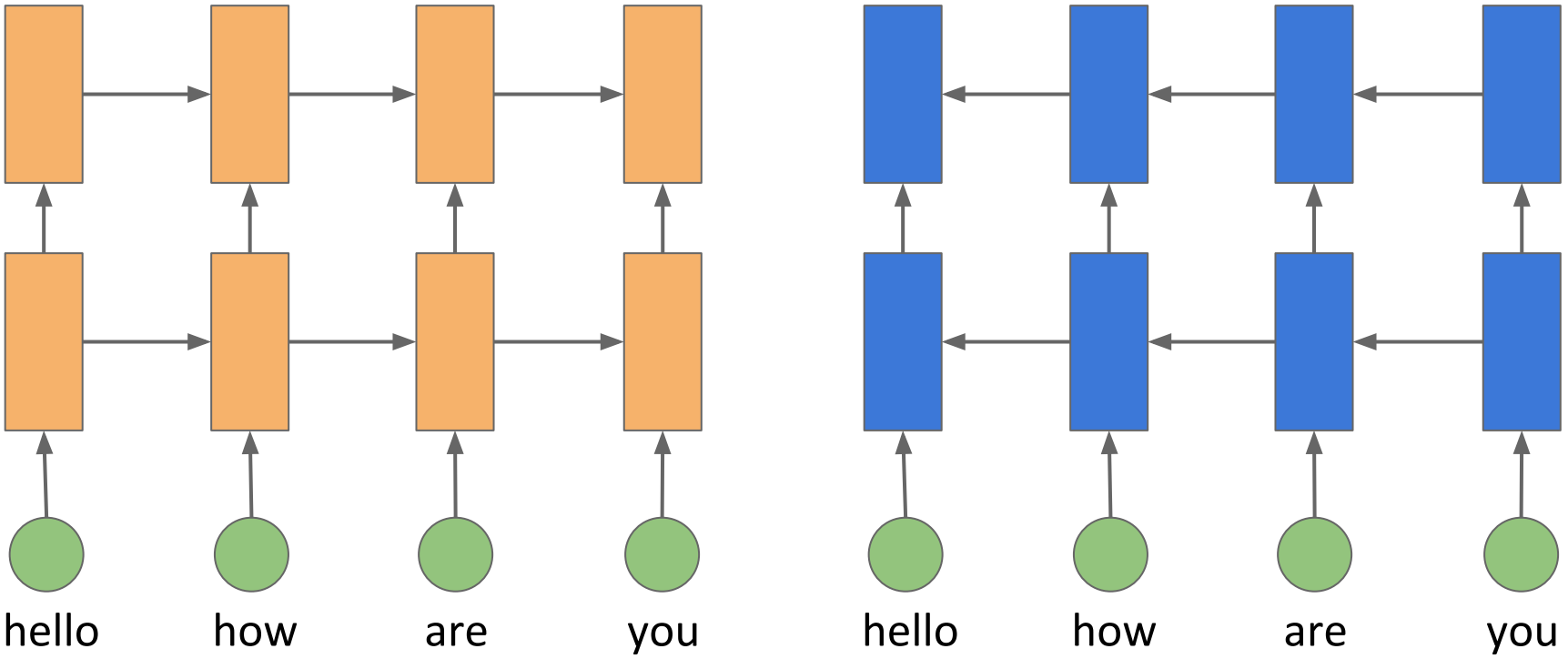
# Multilayer Bidirectional RNN

Here, consider left to right and right to left directions as separate networks

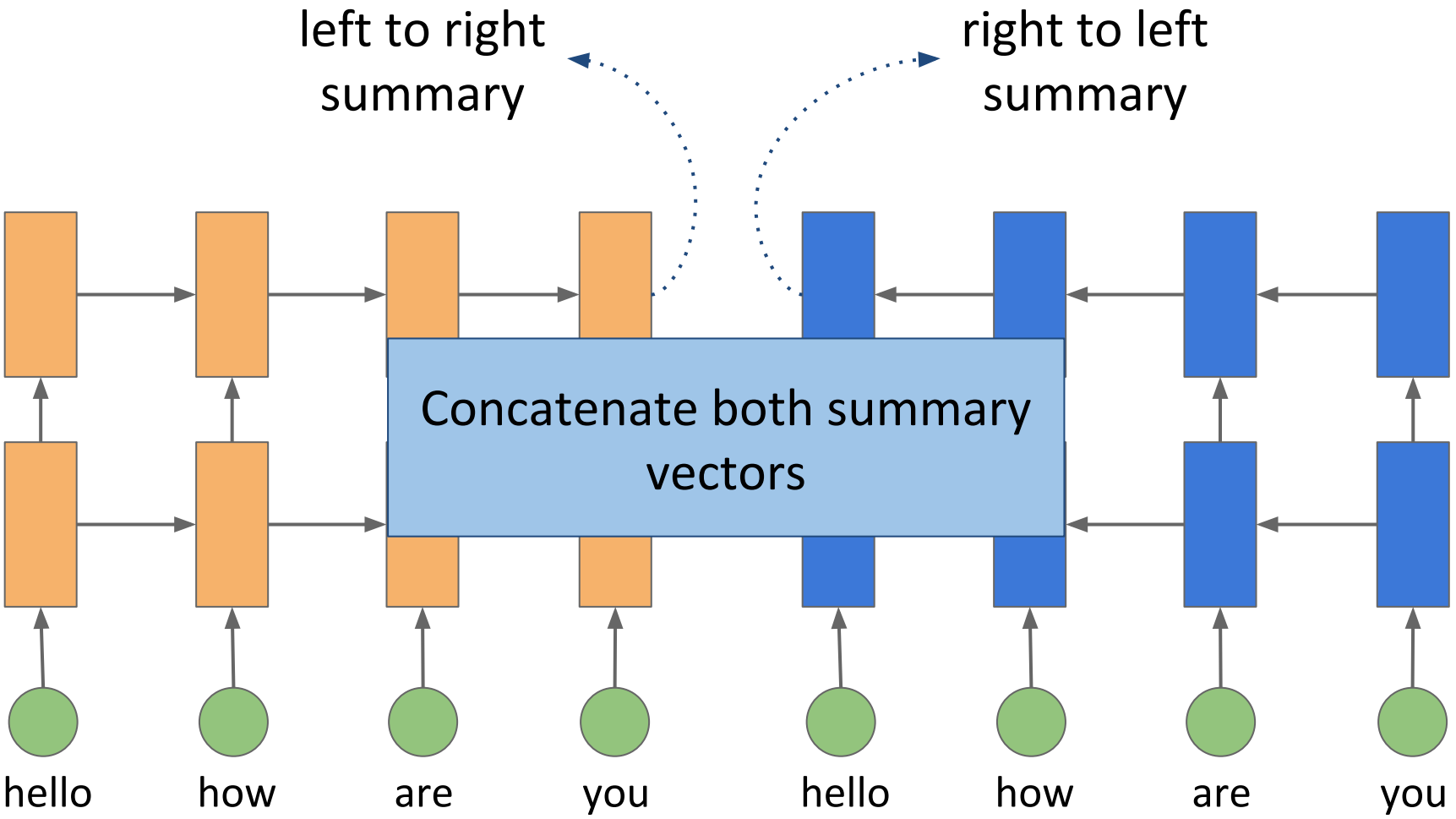


# Multilayer Bidirectional RNN

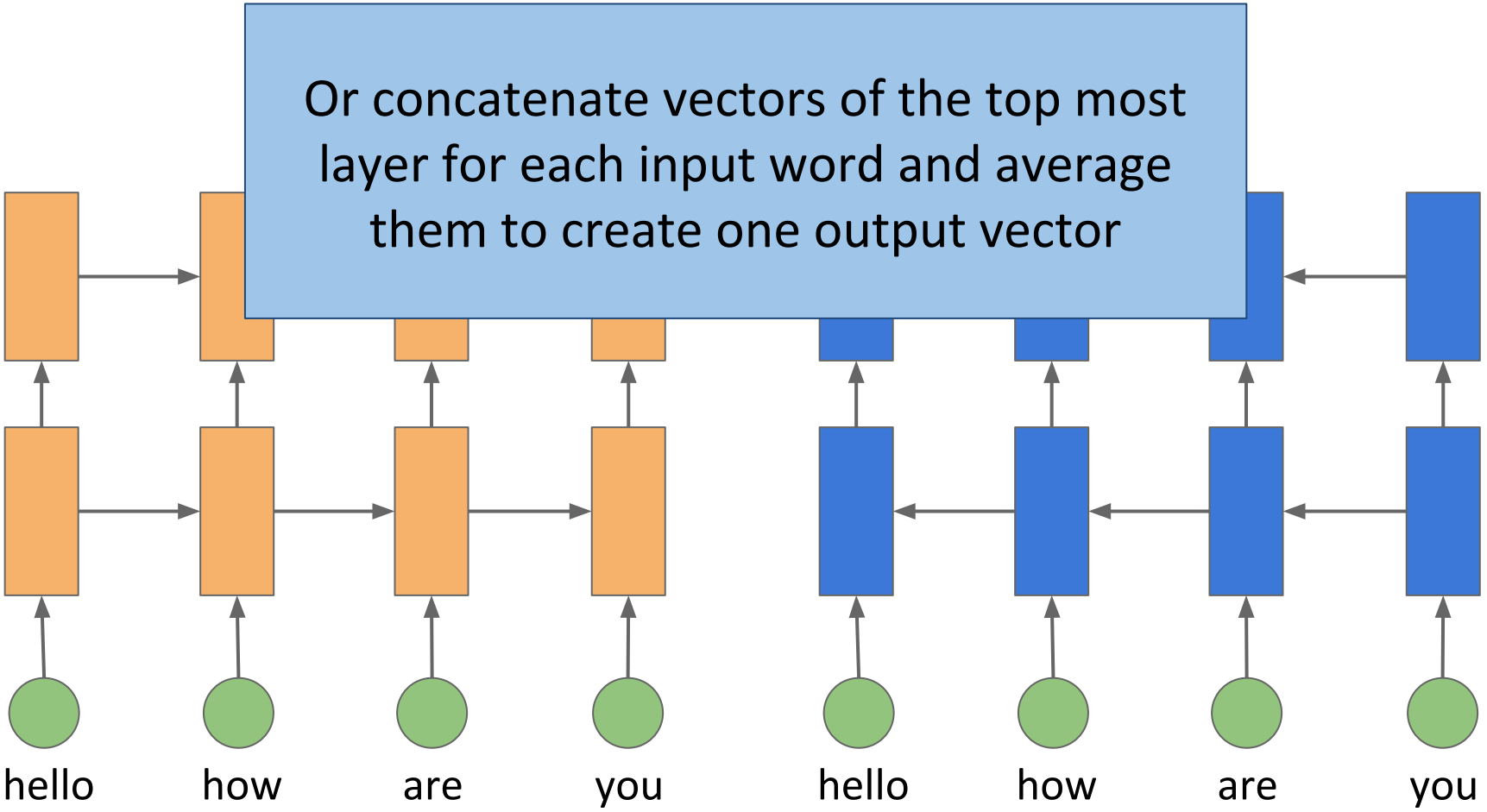
Add layers for each direction



# Multilayer Bidirectional RNN



# Multilayer Bidirectional RNN



# Applications of RNNs

Task Variations

- Sequence classification
- Per timestep classification
- Sequence generation

Architecture Variations

- Bidirectional RNN
- Averaging vs. Summary vector
- Multilayer RNN