Performing Sentiment Analysis Using Word Embeddings

Janani Ravi
CO-FOUNDER, LOONYCORN
www.loonycorn.com

Overview

Representing textual data as features for machine learning

Count, frequency, and probability-based embeddings

GloVe and word2vec for pre-trained word embeddings

Performing sentiment analysis using pre-trained word embeddings

Bi-directional RNNs for sentiment analysis

Numeric Representations of Text

```
d = “This is not the worst restaurant in the metropolis,
not by a long way”
```

Document as Word Sequence

Model a document as an ordered sequence of words

```
d = “This is not the worst restaurant in the metropolis,
not by a long way”

(“This”, “is”,”not”,”the”,”worst”,”restaurant”,”in”,”the”,
“metropolis”, “not”,”by”,”a”,”long”,”way”)
```

Document as Word Sequence

Tokenize document into individual words

x_0 Some numeric encoding of word

w_0 Word (text)

("This", "is","not","the","worst","restaurant","in","the", "metropolis", "not","by","a","long","way")

Represent Each Word as a Number

x_1 Some numeric encoding of word

w_1 Word (text)

("This", "is","not","the","worst","restaurant","in","the",
"metropolis", "not","by","a","long","way")

Represent Each Word as a Number

X_n → Some numeric encoding of word

W_n → Word (text)

("This", "is","not","the","worst","restaurant","in","the",
"metropolis", "not","by","a","long","way")

Represent Each Word as a Number

$$d = [x_0,\ x_1,\ \ldots\ x_n]$$

Document as Tensor

Represent each word as numeric data, aggregate into tensor

Numeric Representations of Text

One-hot

Frequency-based

Prediction-based

Numeric Representations of Text

One-hot

Frequency-based

Prediction-based

Represent each word in text by its presence or absence

Numeric Representations of Text

One-hot

Frequency-based

Prediction-based

Frequency-based Embeddings

Count

TF-IDF

Co-occurrence

Frequency-based Embeddings

Count

TF-IDF

Co-occurrence

Capture how often a word occurs in a document i.e. the **counts** or the **frequency**

Frequency-based Embeddings

Count

TF-IDF

Co-occurrence

Captures how often a word occurs in a **document** as well as the **entire corpus**

Tf-Idf

Frequently in a single document

Might be important

Frequently in the corpus

Probably a common word like “a”, “an”, “the”

Frequency-based Embeddings

Count

TF-IDF

Co-occurrence

Similar words will occur together and will have similar context

Context Window

A window centered around a word, which includes a certain number of neighboring words

Co-occurrence

The number of times two words *w1* and *w2* have occurred together in a context window

Word Embeddings

One-hot

Frequency-based

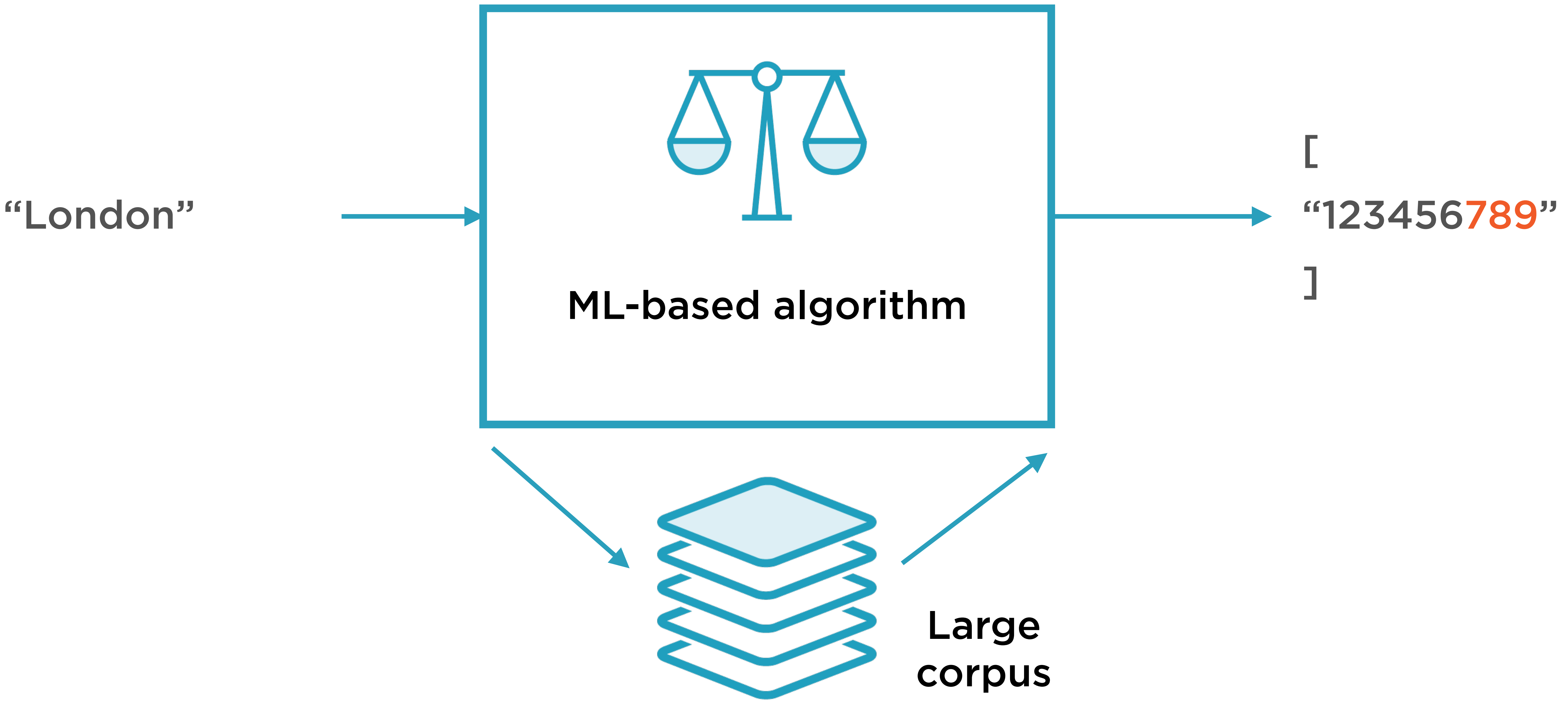
Prediction-based

Predictions-based embeddings

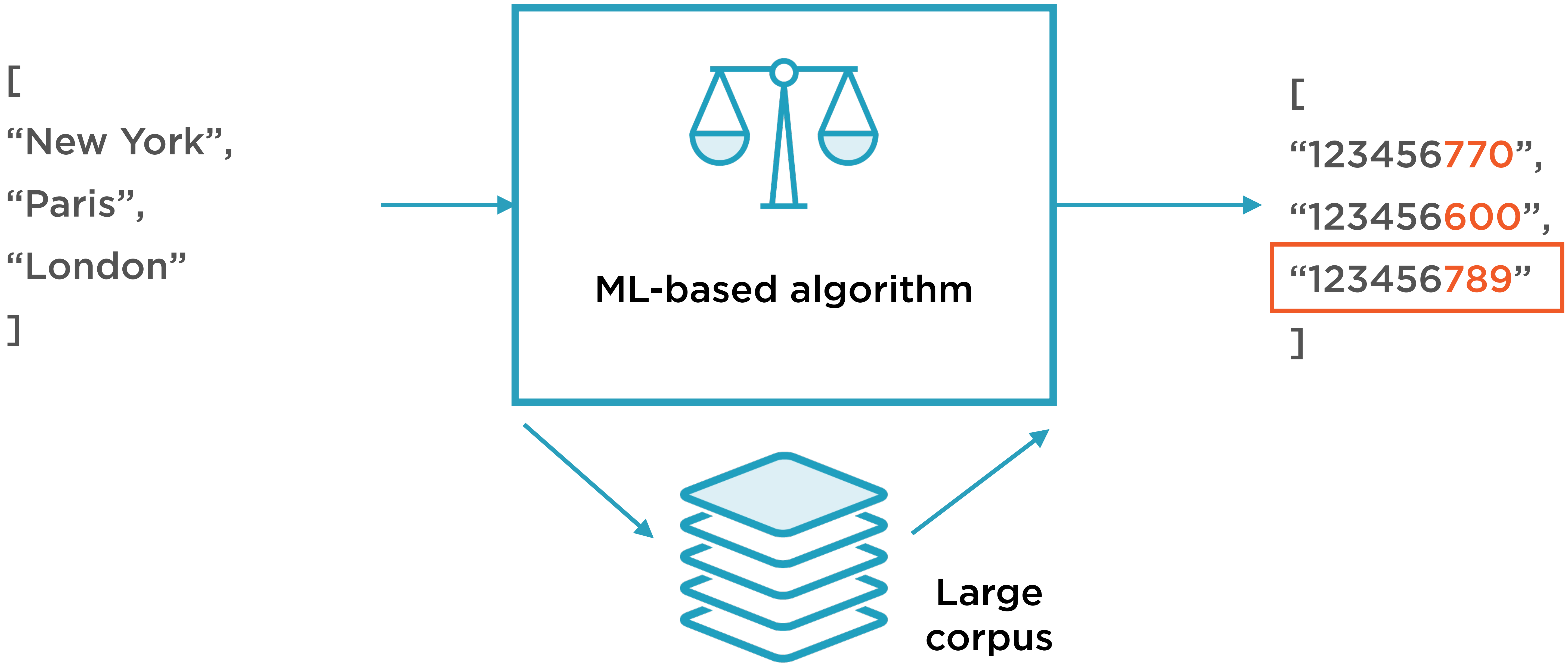
Capture meanings and semantic relationships, generated using ML models

“~~Birds~~ Words of a feather flock together”

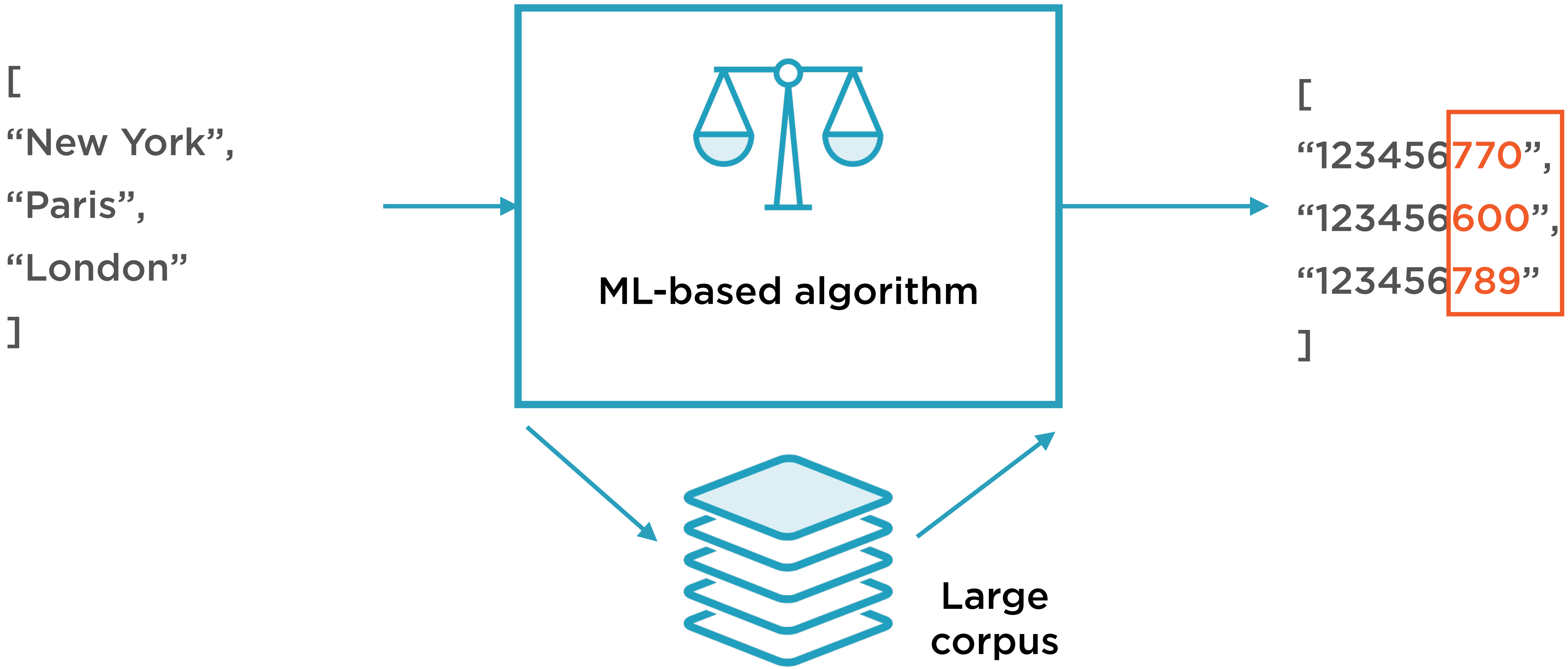
Prediction-based Word Embeddings



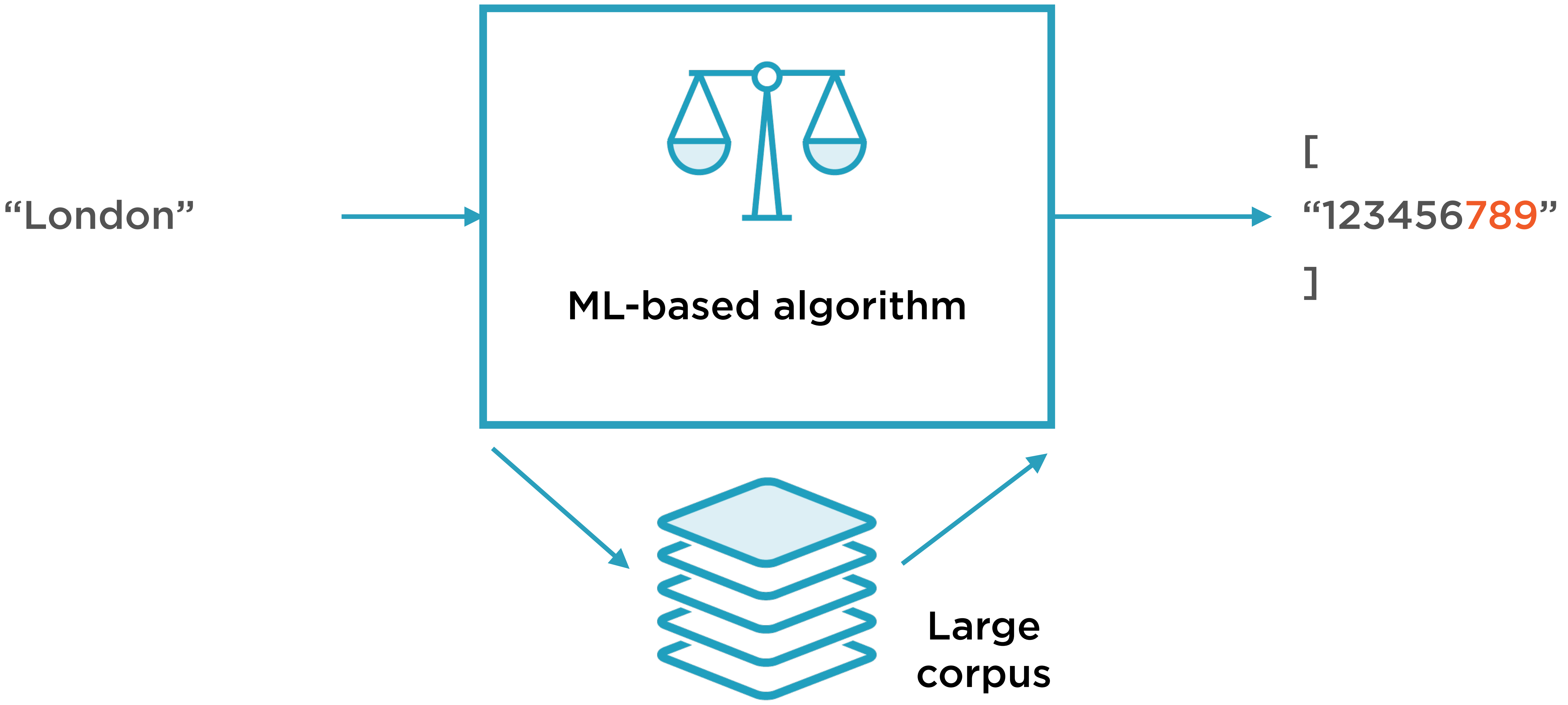
Prediction-based Word Embeddings



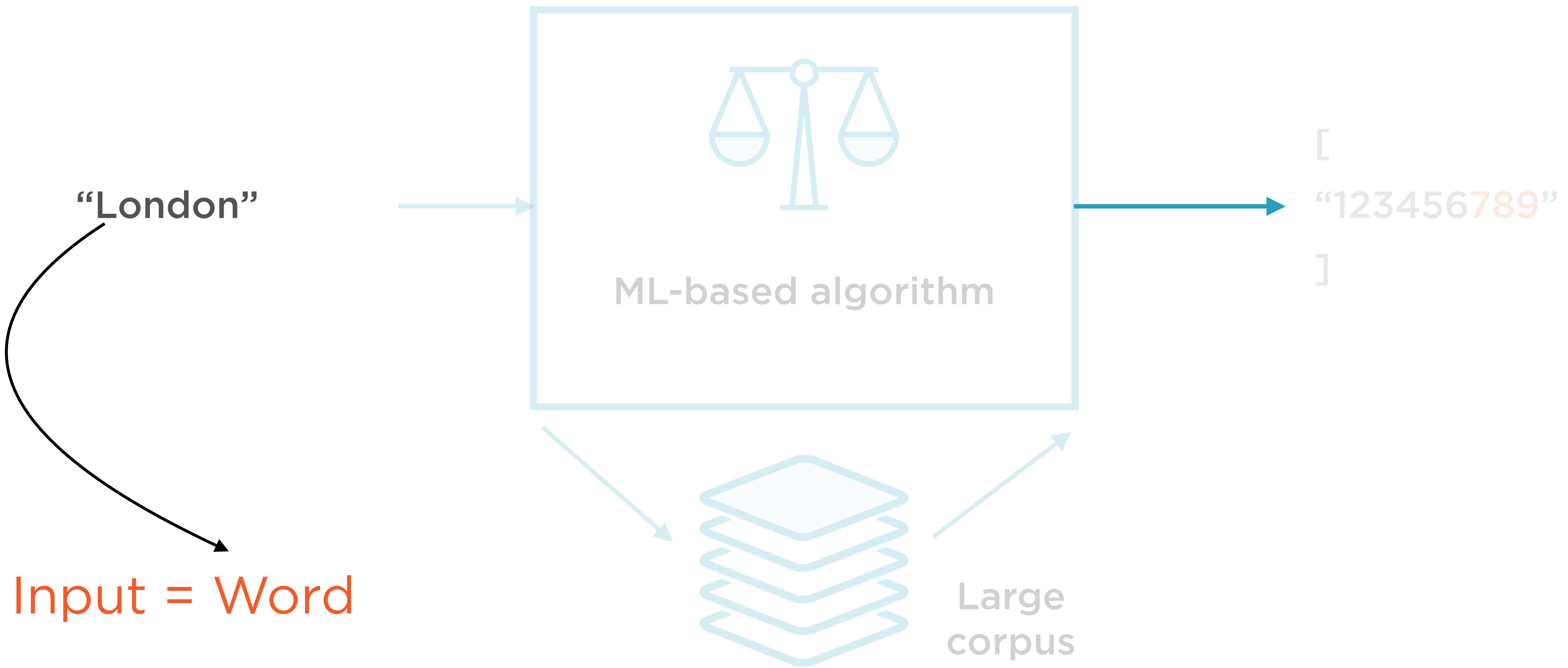
Prediction-based Word Embeddings



Prediction-based Word Embeddings

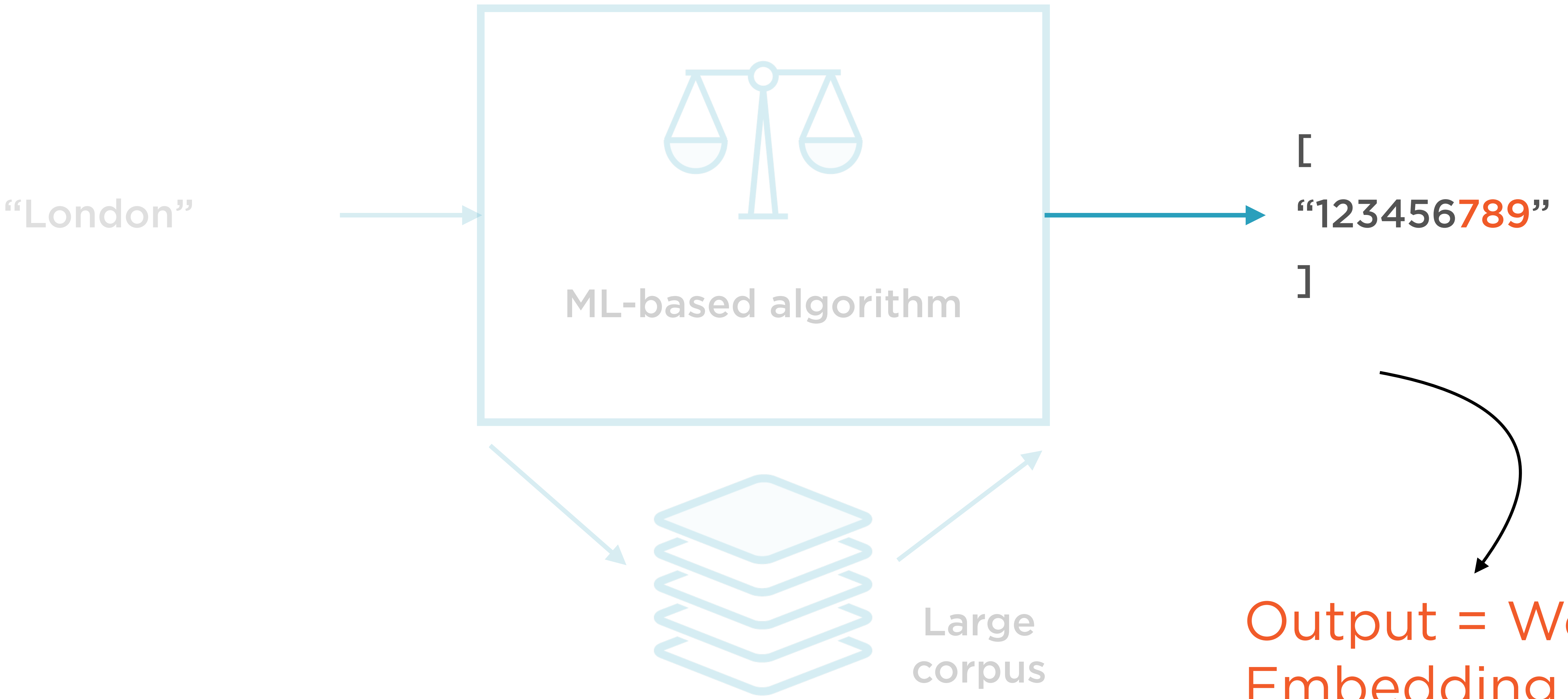


Prediction-based Word Embeddings



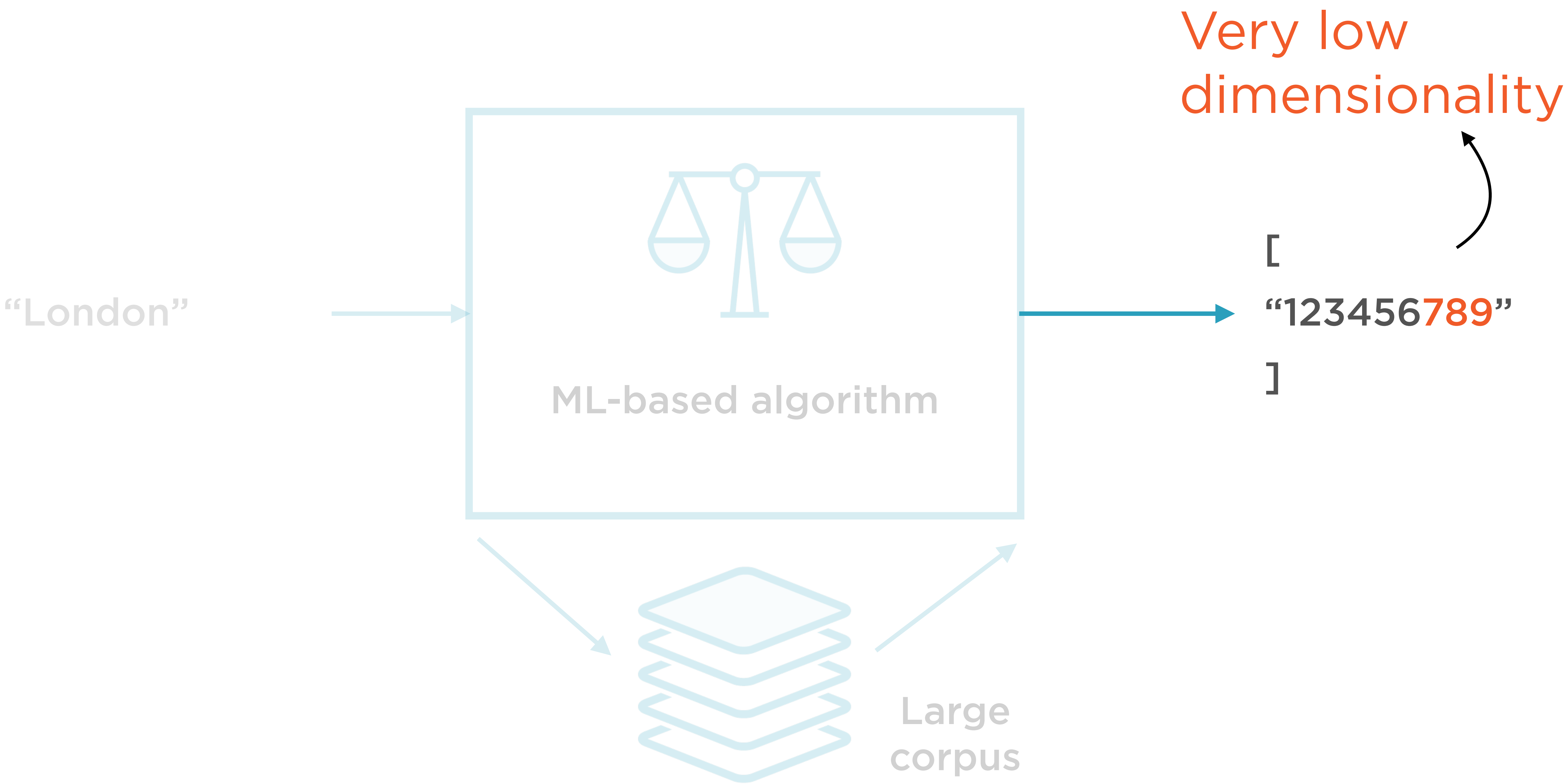
Input = Word

Prediction-based Word Embeddings

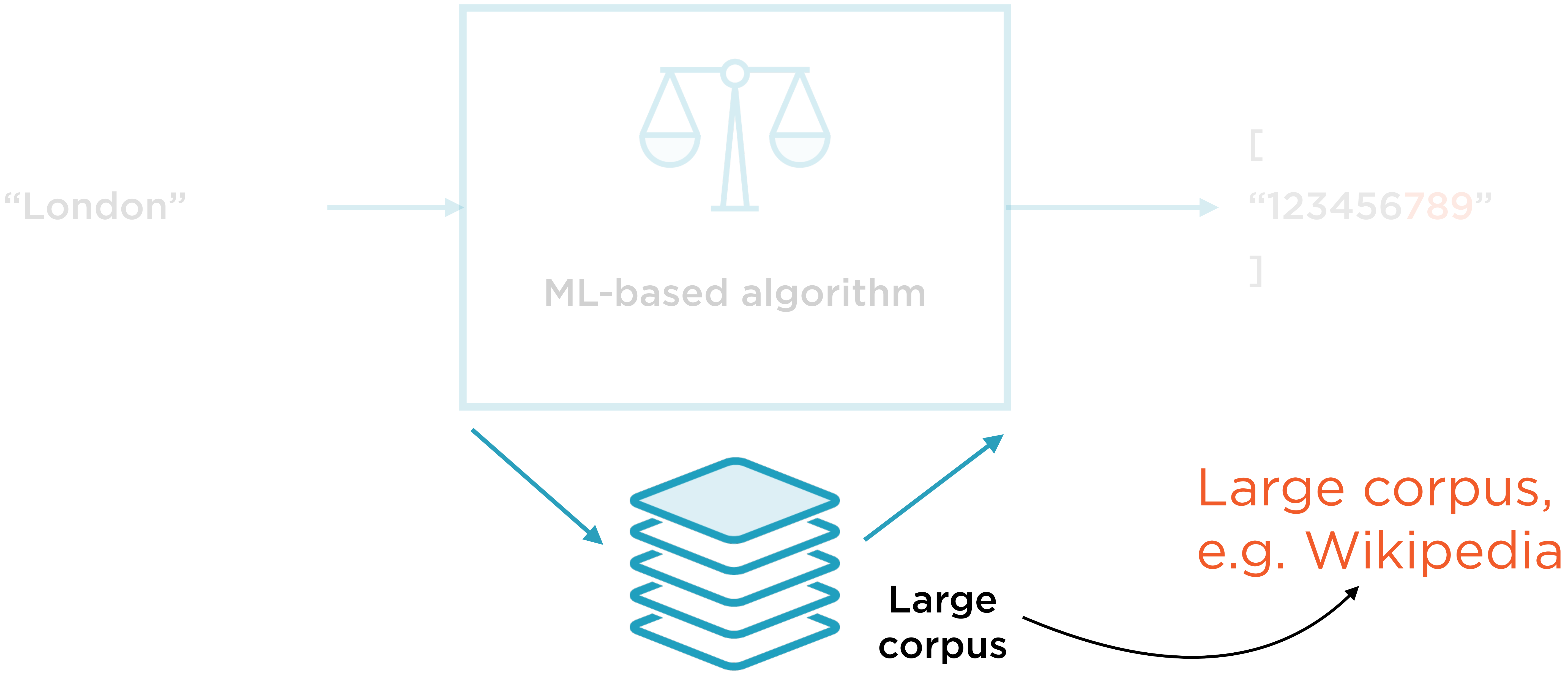


Output = Word Embedding

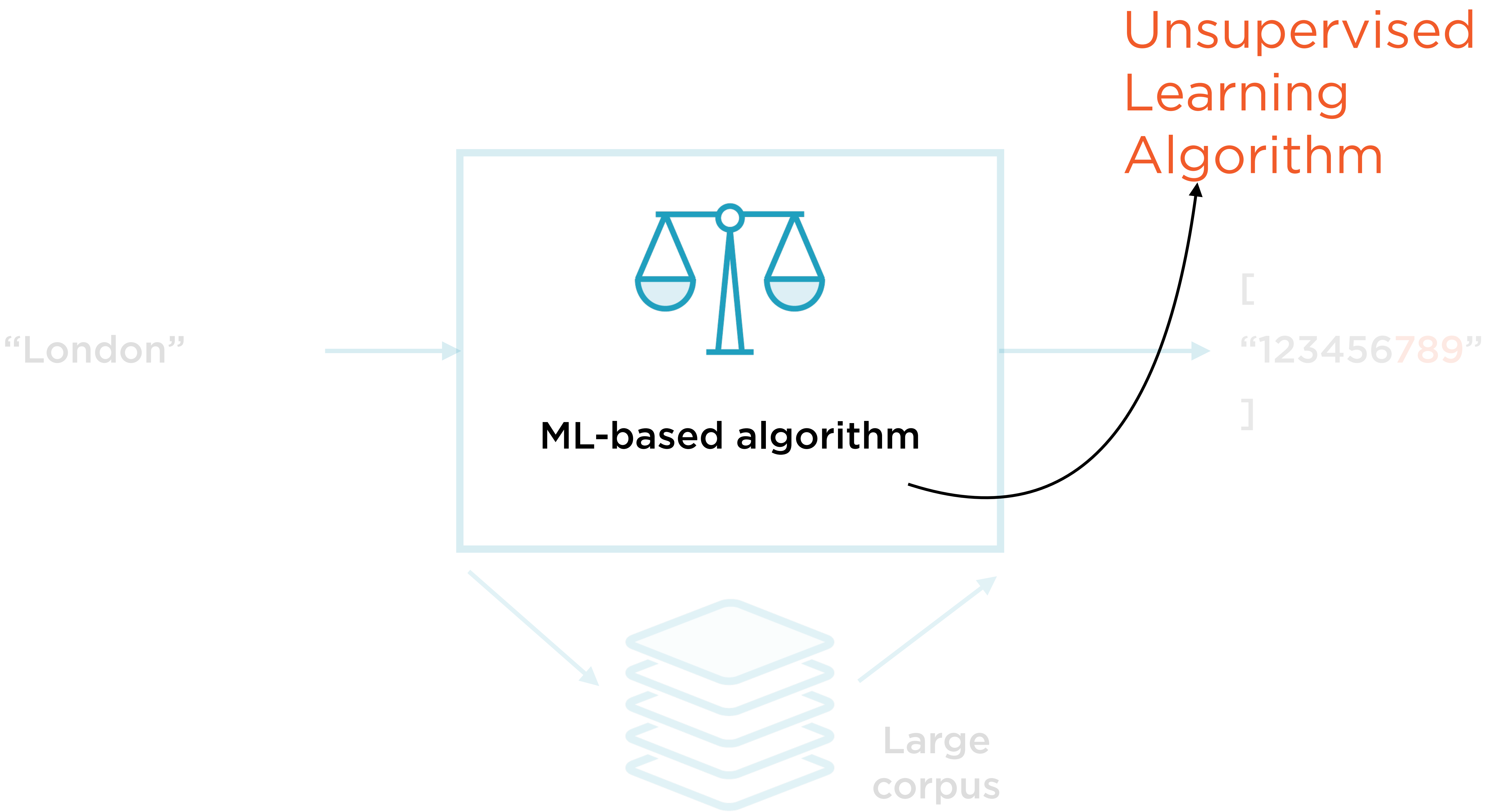
Prediction-based Word Embeddings



Prediction-based Word Embeddings



Prediction-based Word Embeddings



Pre-trained Word Embeddings

Word2Vec

GLoVe

Word2Vec

Most popular word embedding model

Mikolov (Google), 2013

Use simple NN (not deep) to learn embeddings

GLoVe

Global Vectors for Word Representation

Jeffrey Pennington, Richard Socher, Christopher D. Manning, (Stanford) 2014

Uses word-word co-occurrence matrix, nearest-neighbors for word relationships

Given words from its context,
predict the word

Given a word, **predict the words in its context**

Word Embeddings

Encode each word as a vector of other words

What words? A few related ones

Embedding(“true”) = [“false”]

Embedding(“London”) = [“New York”, “Paris”]

Magic

Word embeddings capture meaning

“Queen” ~ “King” == “Woman” ~ “Man”

“Paris” ~ “France” == “London” ~ “England”

Dramatic dimensionality reduction

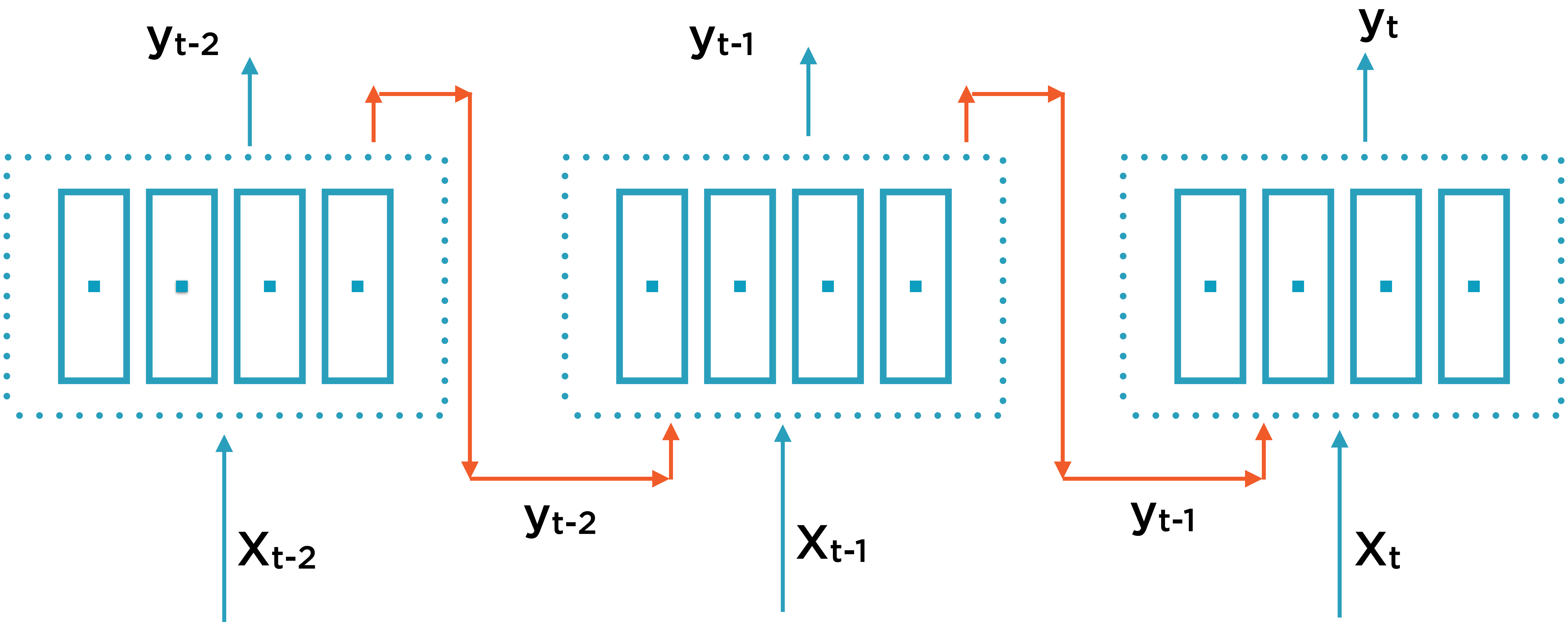
Embeddings are a way to encode words capturing the **context** around them

Demo

Explore pre-trained GloVe word embeddings

Multilayer RNNs

Conventional RNNs

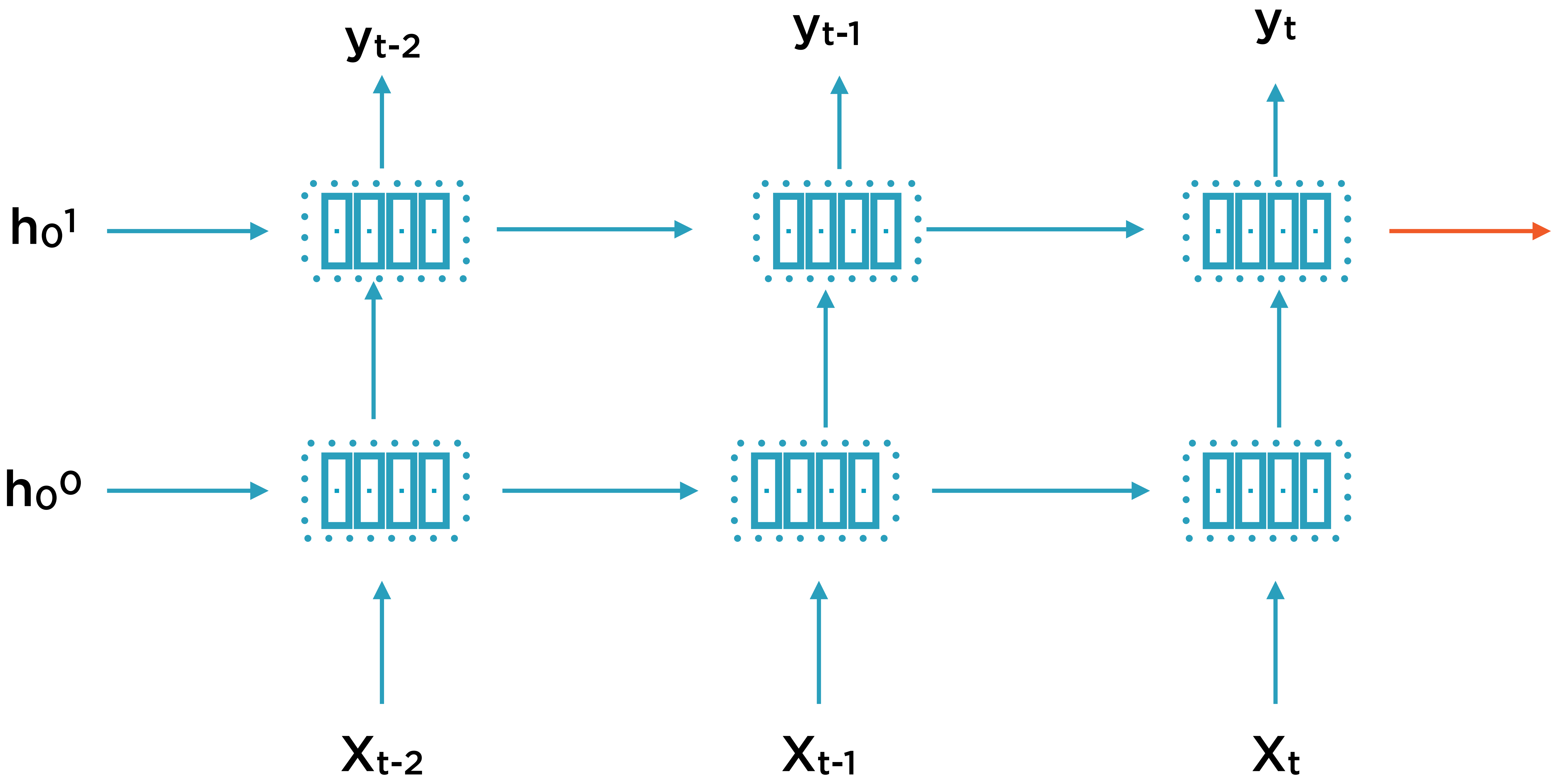


The cell unrolled through time form the layers of the neural network

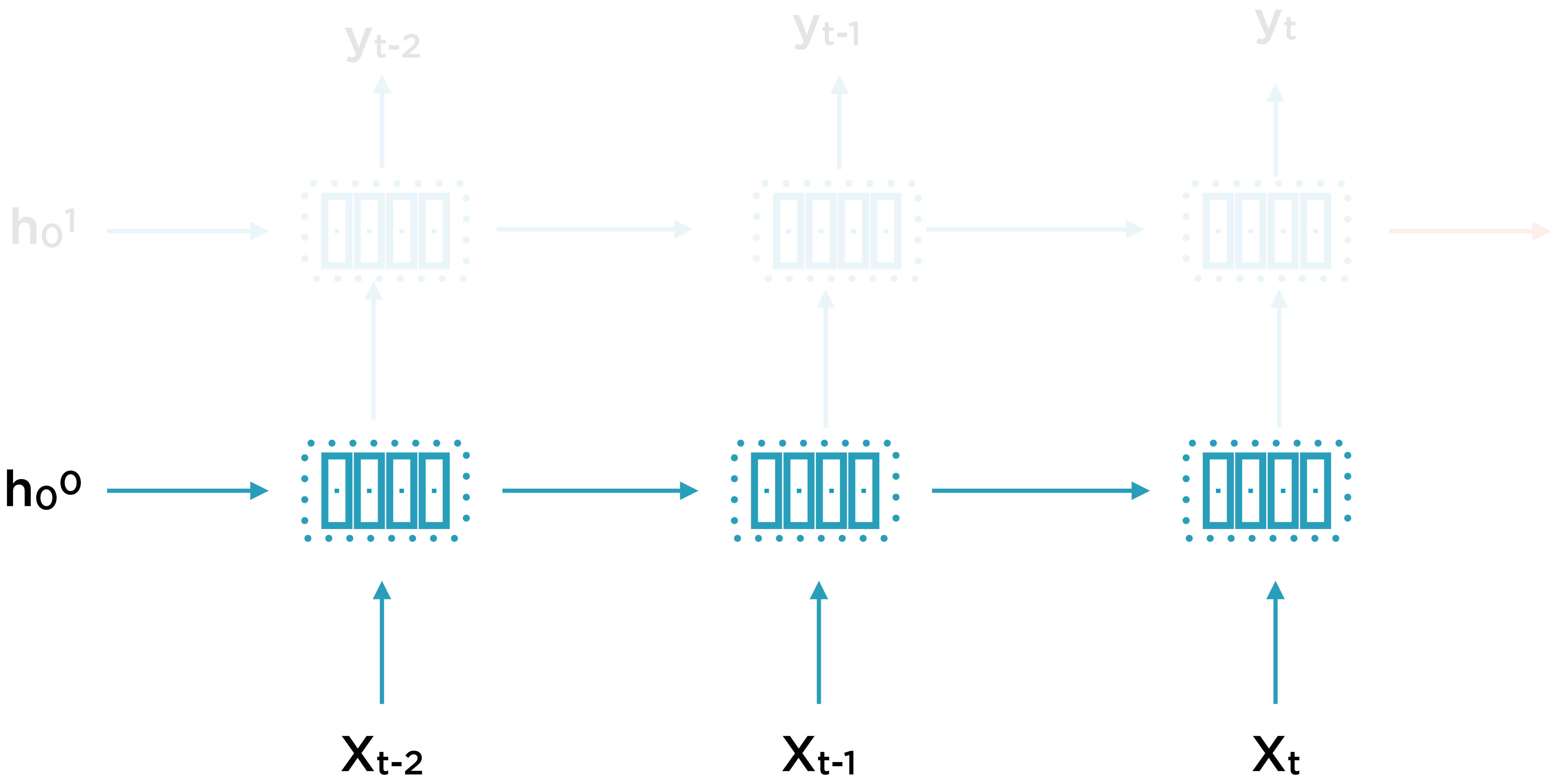
Multilayer RNNs

Add additional RNNs on top of the original RNN, where each RNN added is another layer

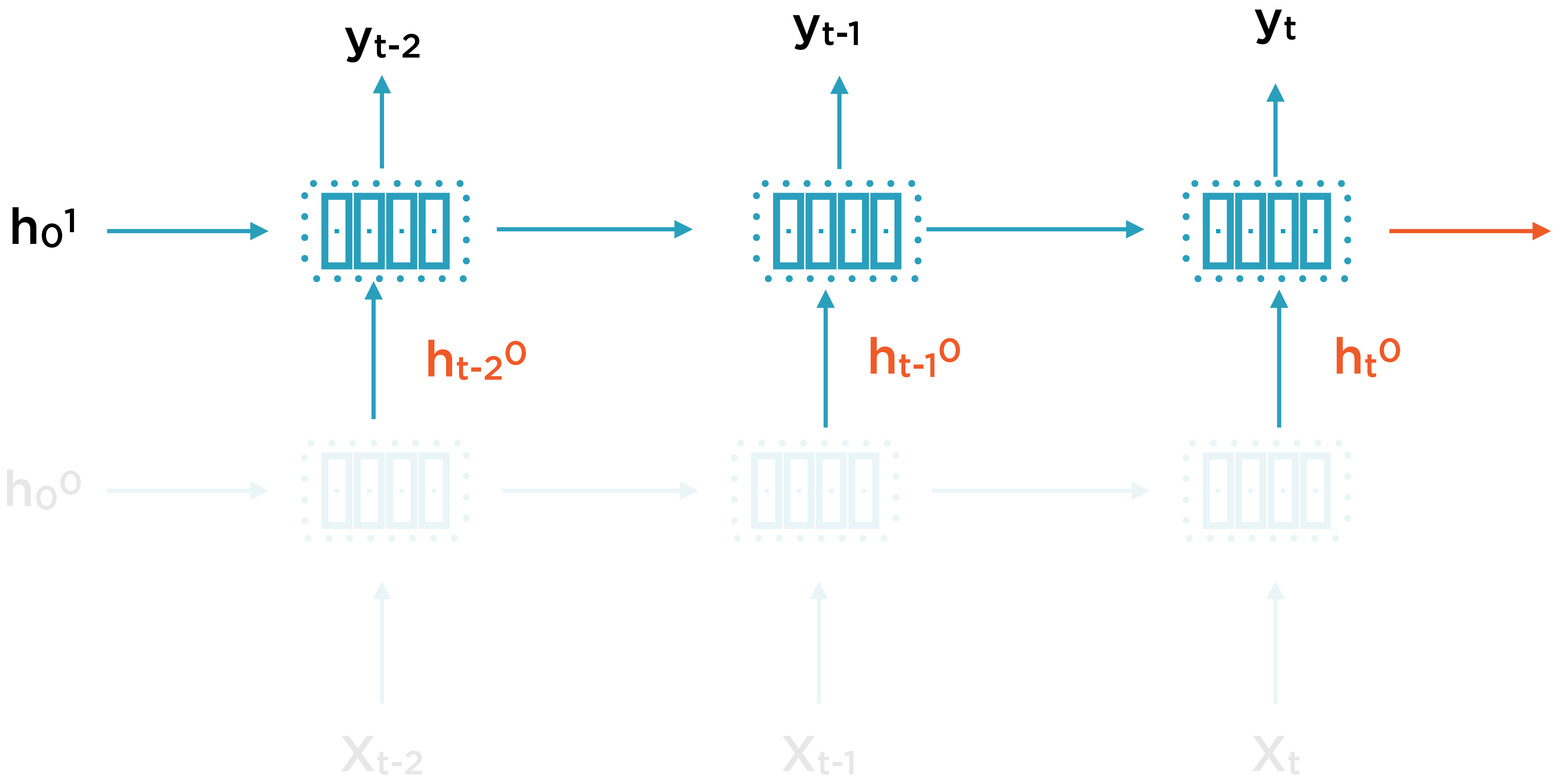
Multilayer RNNs



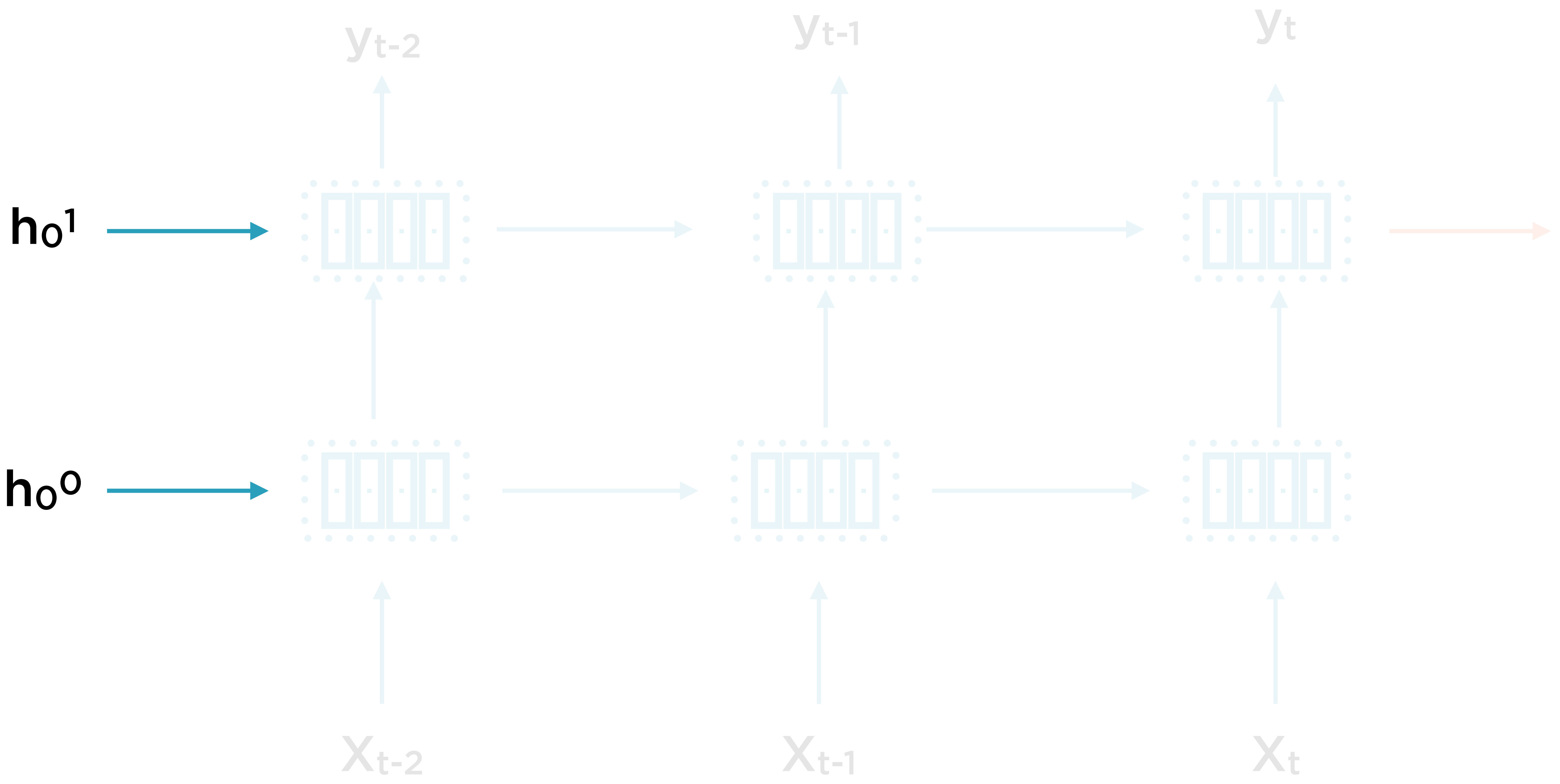
Multilayer RNNs



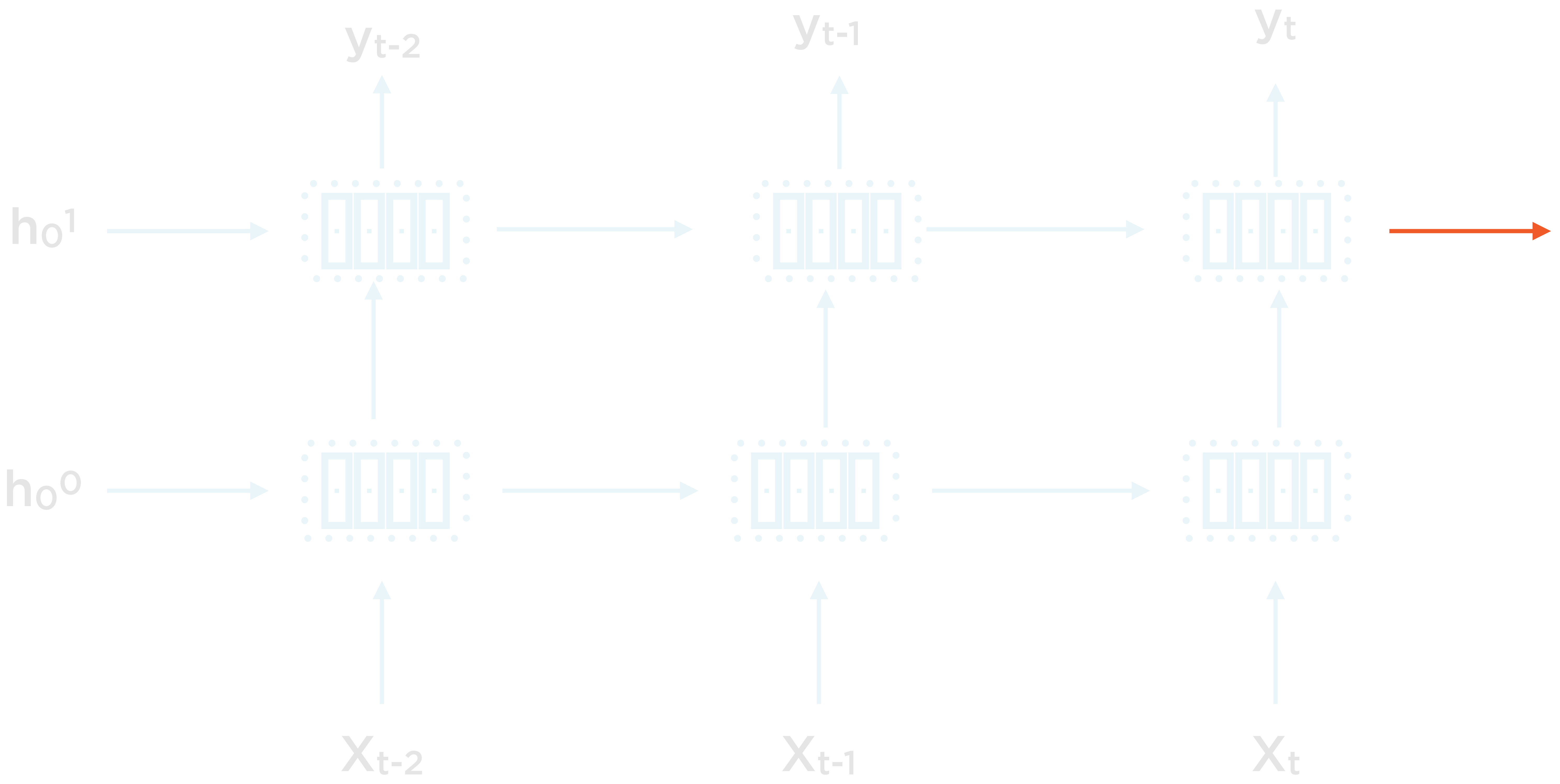
Multilayer RNNs



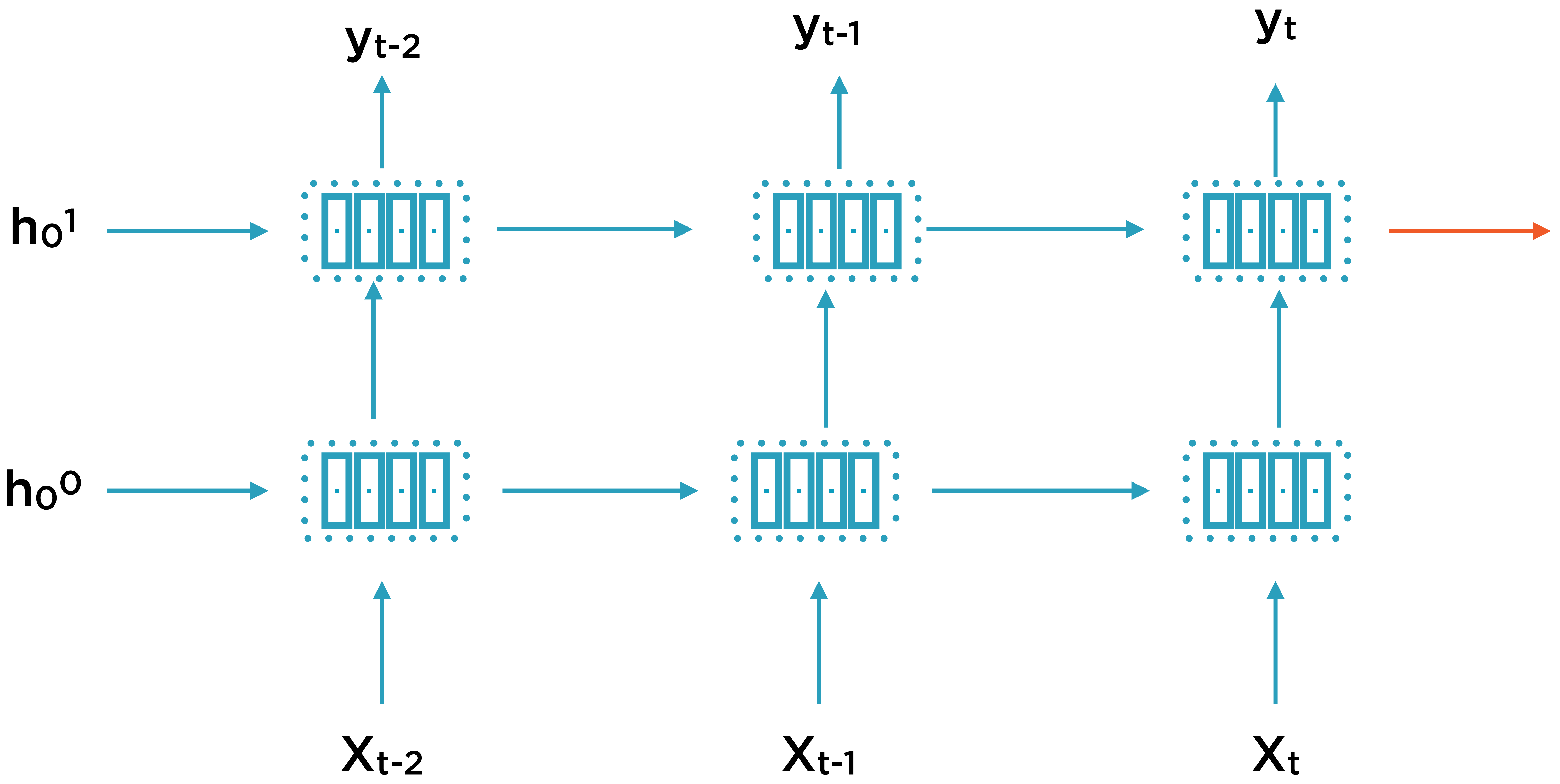
Multilayer RNNs



Multilayer RNNs



Multilayer RNNs



Multi-layer RNNs (also called deep RNNs) are another simple concept. The idea is that we add additional RNNs on top of the initial standard RNN, where each RNN added is another layer. The hidden state output by the first (bottom) RNN at time-step will be the input to the RNN above it at time step . The prediction is then made from the final hidden state of the final (highest) layer.

The image below shows a multi-layer unidirectional RNN, where the layer number is given as a superscript. Also note that each layer needs their own initial hidden state, .

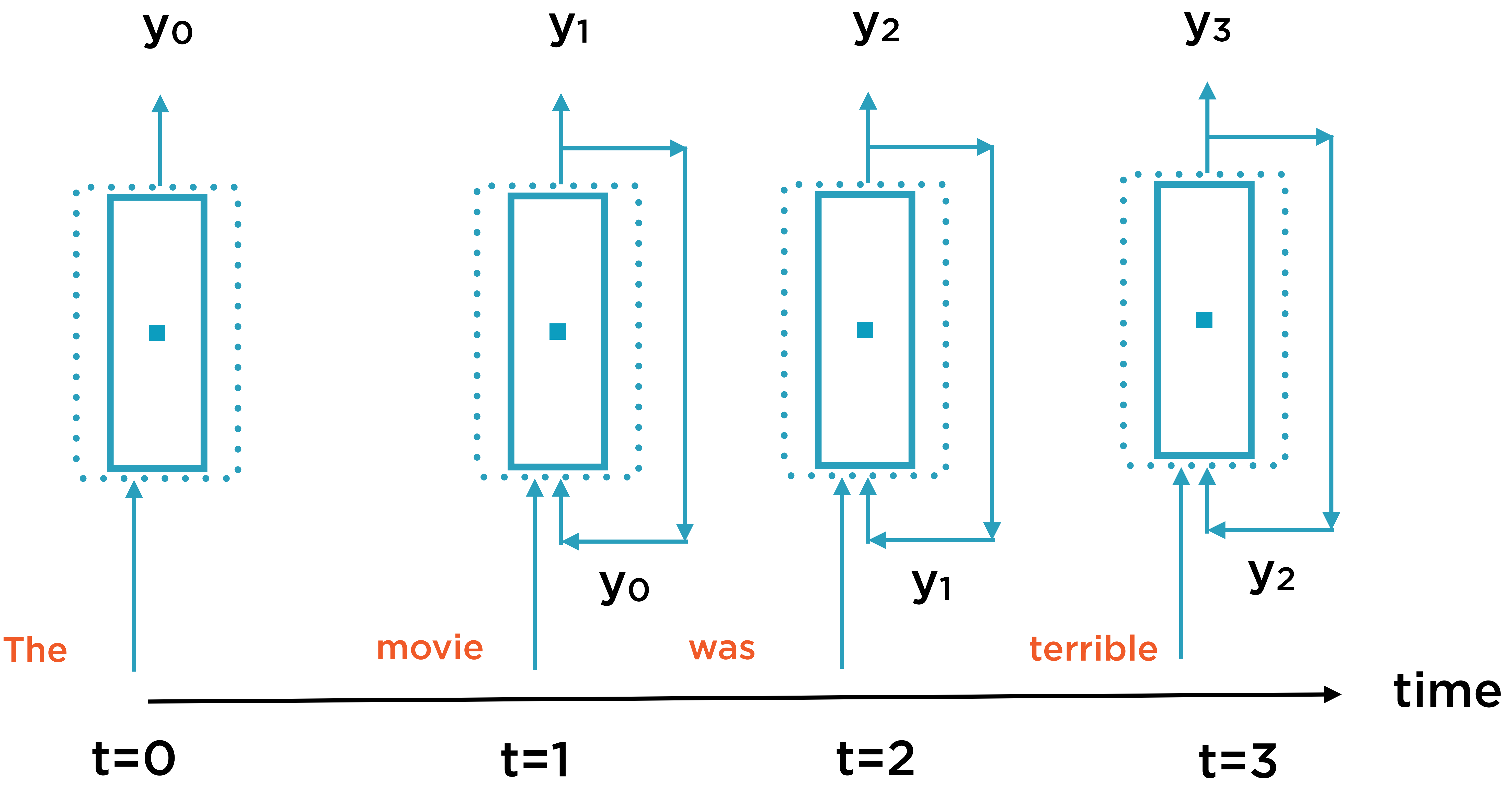
Bidirectional RNNs

```
d = “The movie was terrible. Hated it.”
```

Document as Word Sequence

Model a document as an ordered sequence of words

Conventional RNN



Conventional RNN

The sky is cloudy, it looks like. ______

The blank slot will be filled in using preceding information only

Bidirectional RNN

The sky is $\boxed{\_\_\_\_\_\_,}$ it looks like $\boxed{\text{rain}}$

Bidirectional RNNs will also use following information

```
d = “.it Hated .terrible was movie The”
```

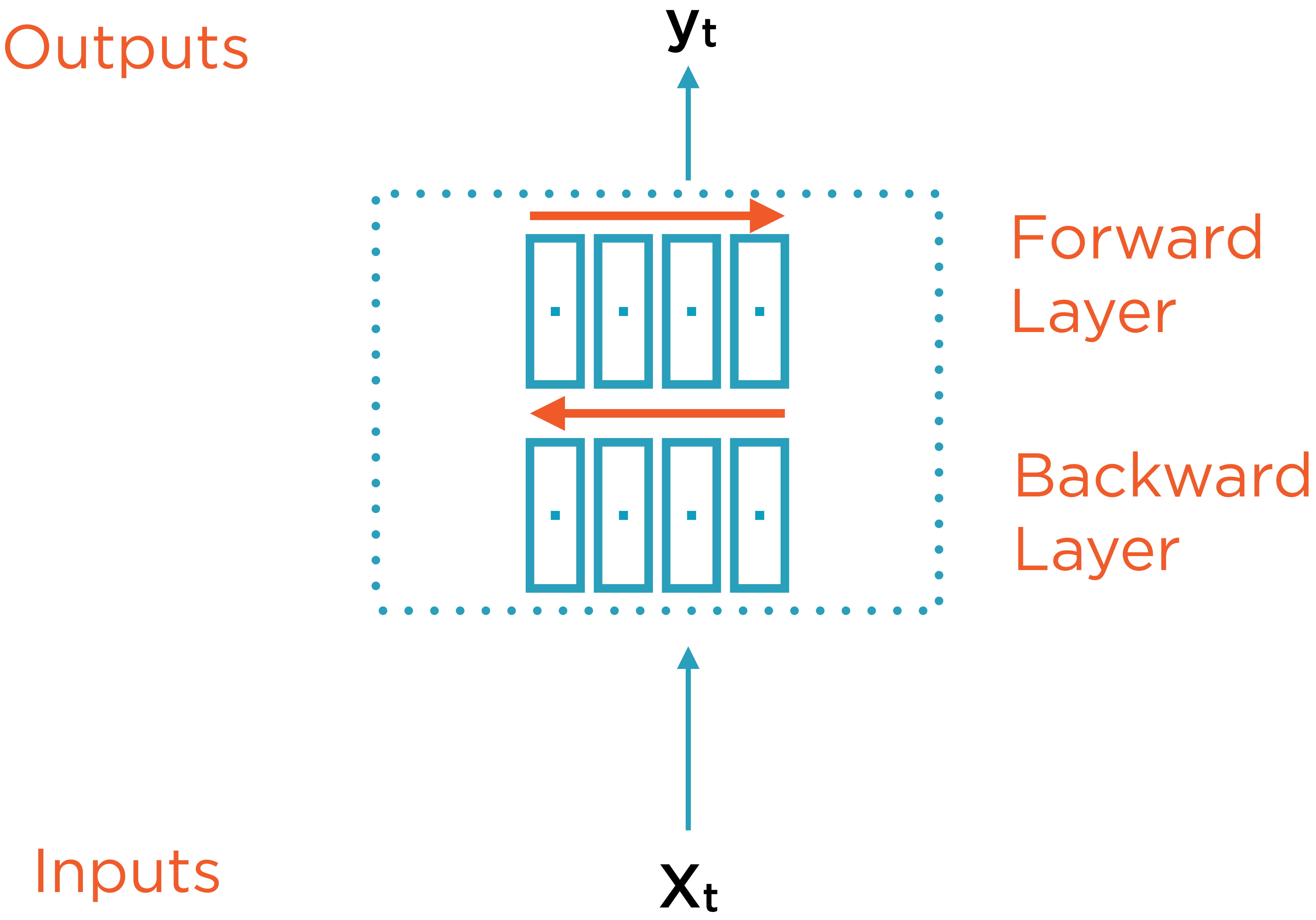
Reverse the Document Word Sequence

Bidirectional RNNs run words both forward and backward

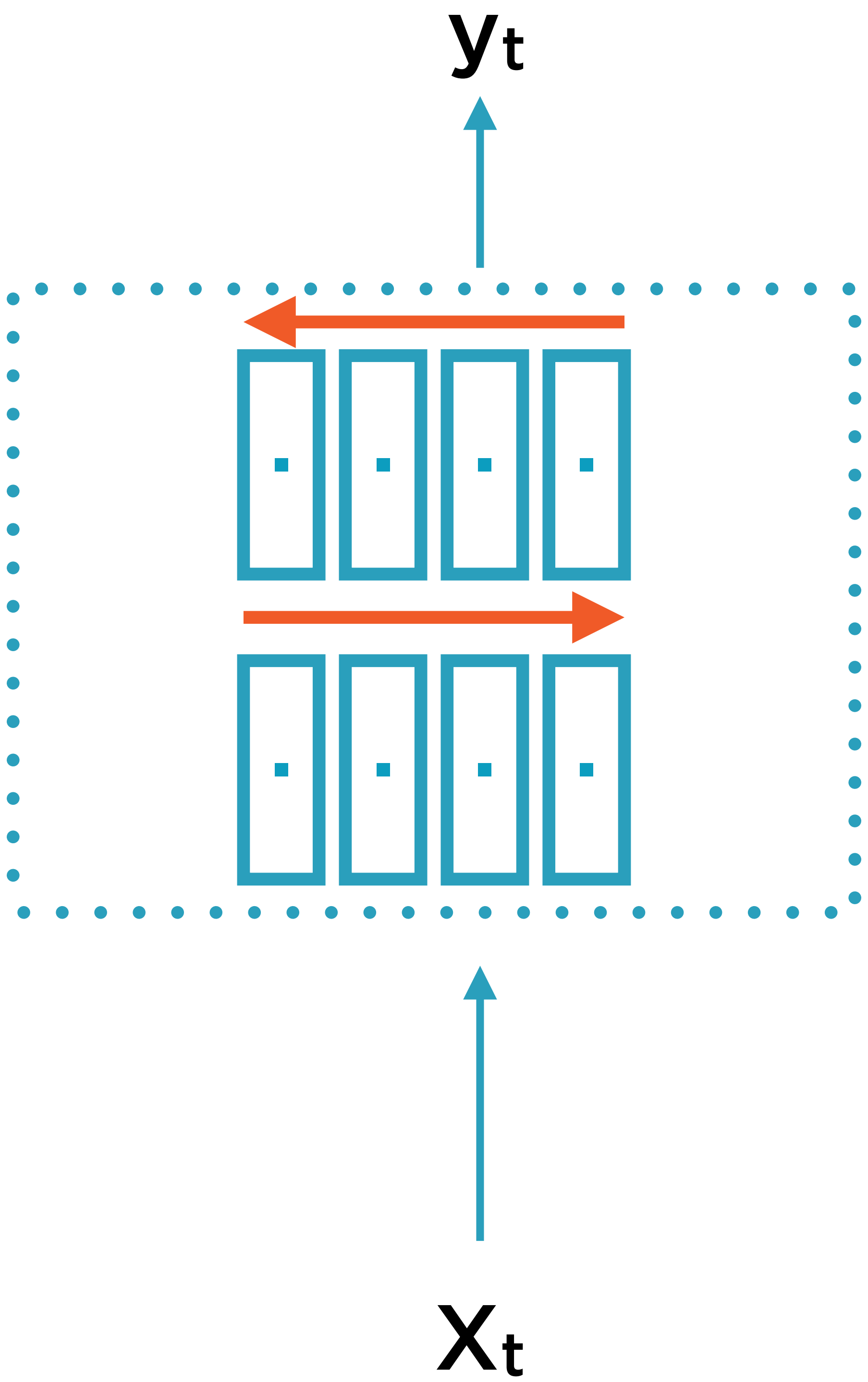
Is the use of a Bidirectional RNN **cheating**?

For streaming applications - yes; for batch applications - **no!**

Bidirectional RNNs



Bidirectional RNNs

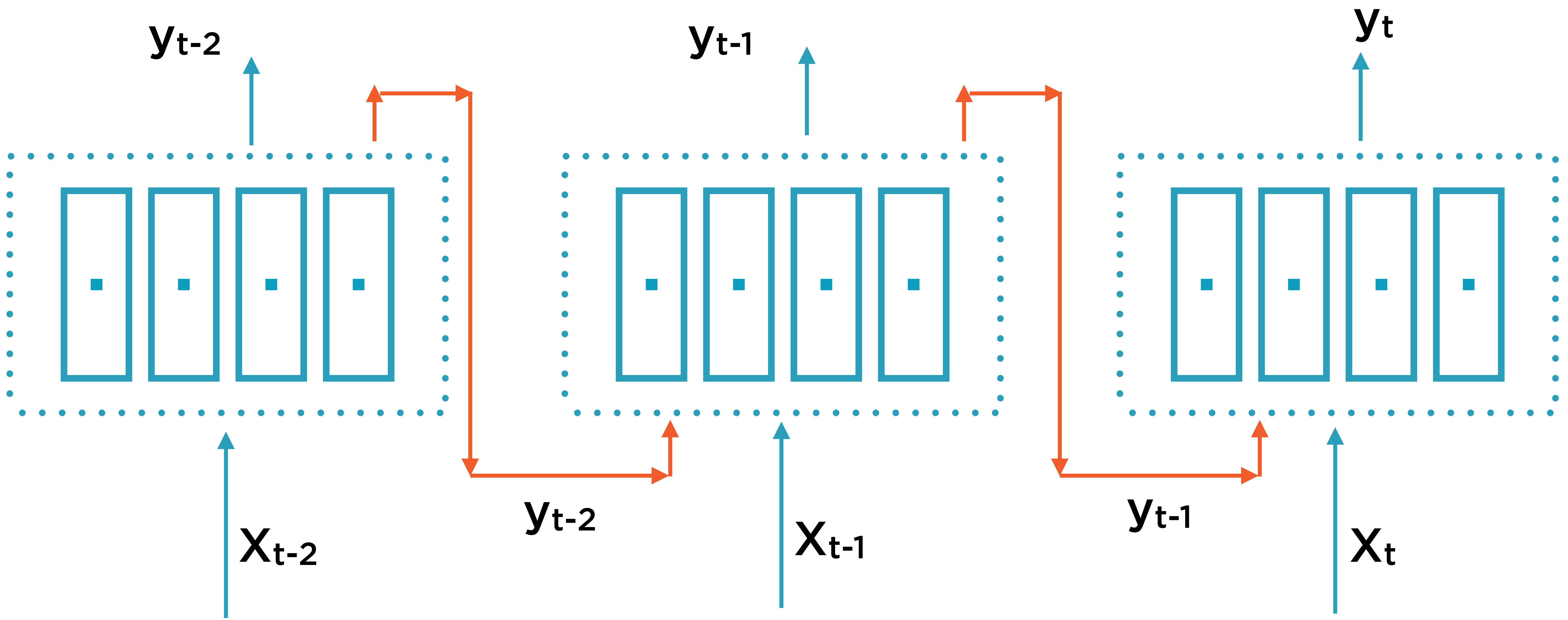


Speech recognition: Utterance = batch of words spoken together

Text auto-complete, auto-correct: Fragment = batch of characters input so far

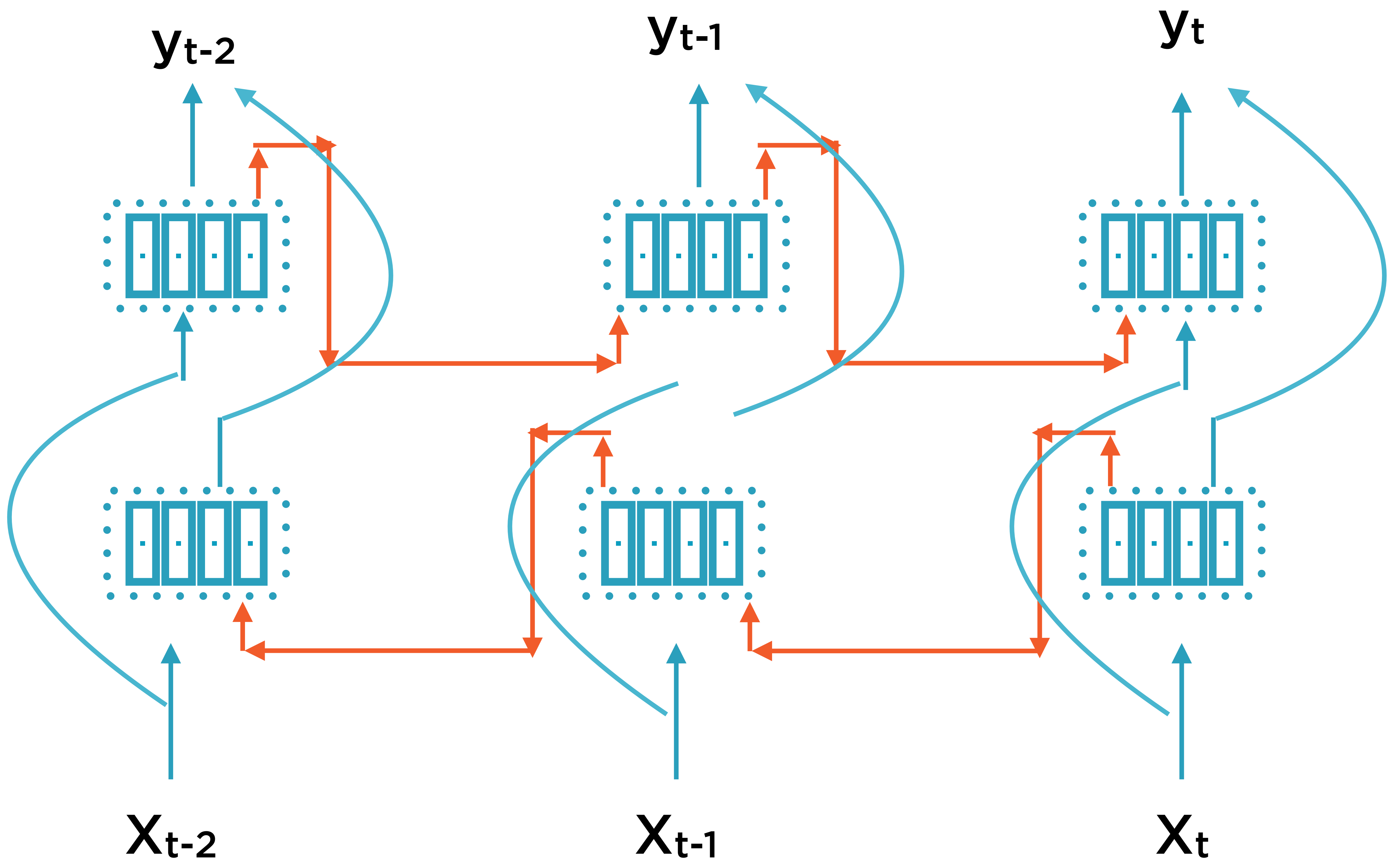
Language Modeling: e.g. Cause-effect relationships

Conventional RNNs

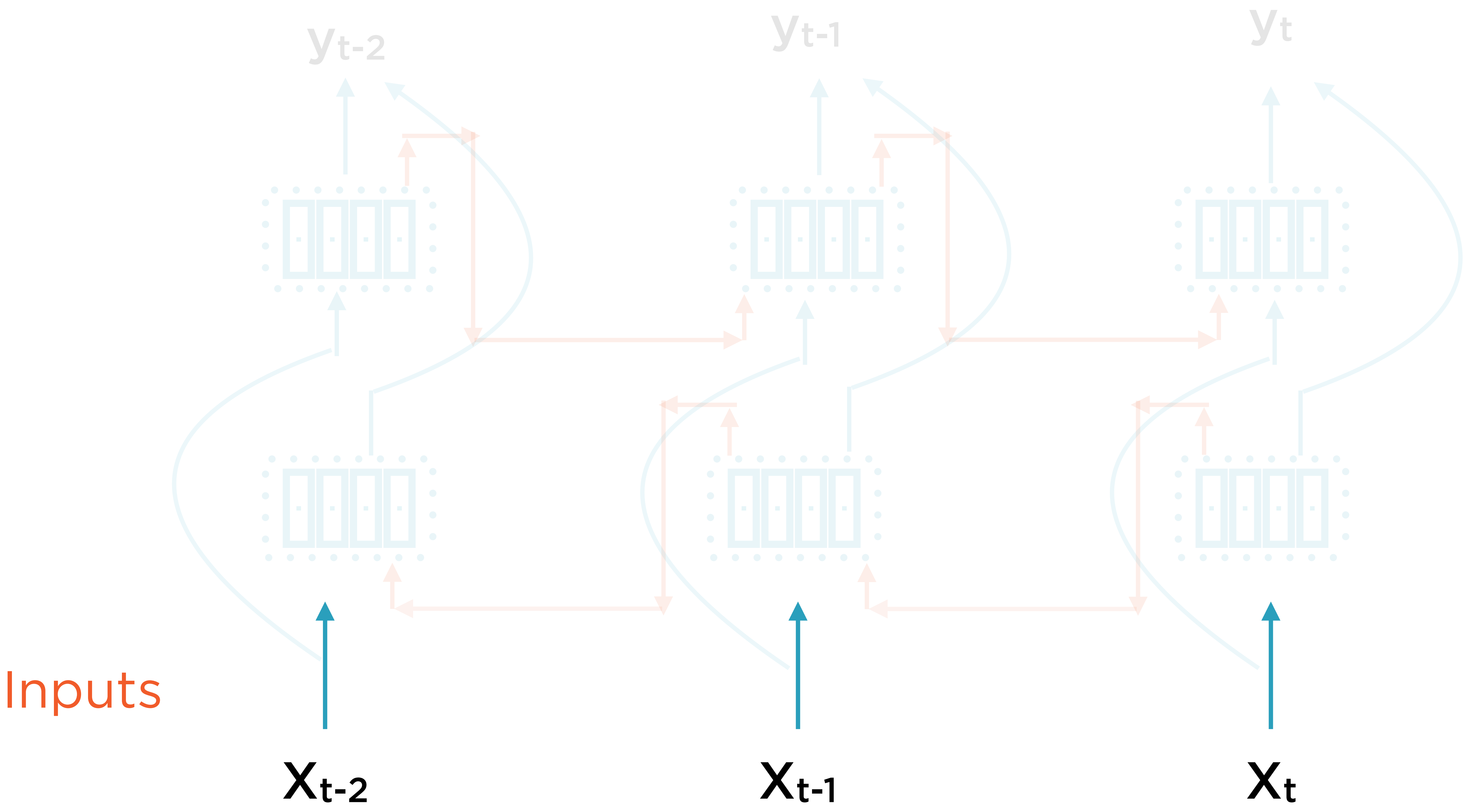


The cell unrolled through time form the layers of the neural network

Bidirectional RNNs

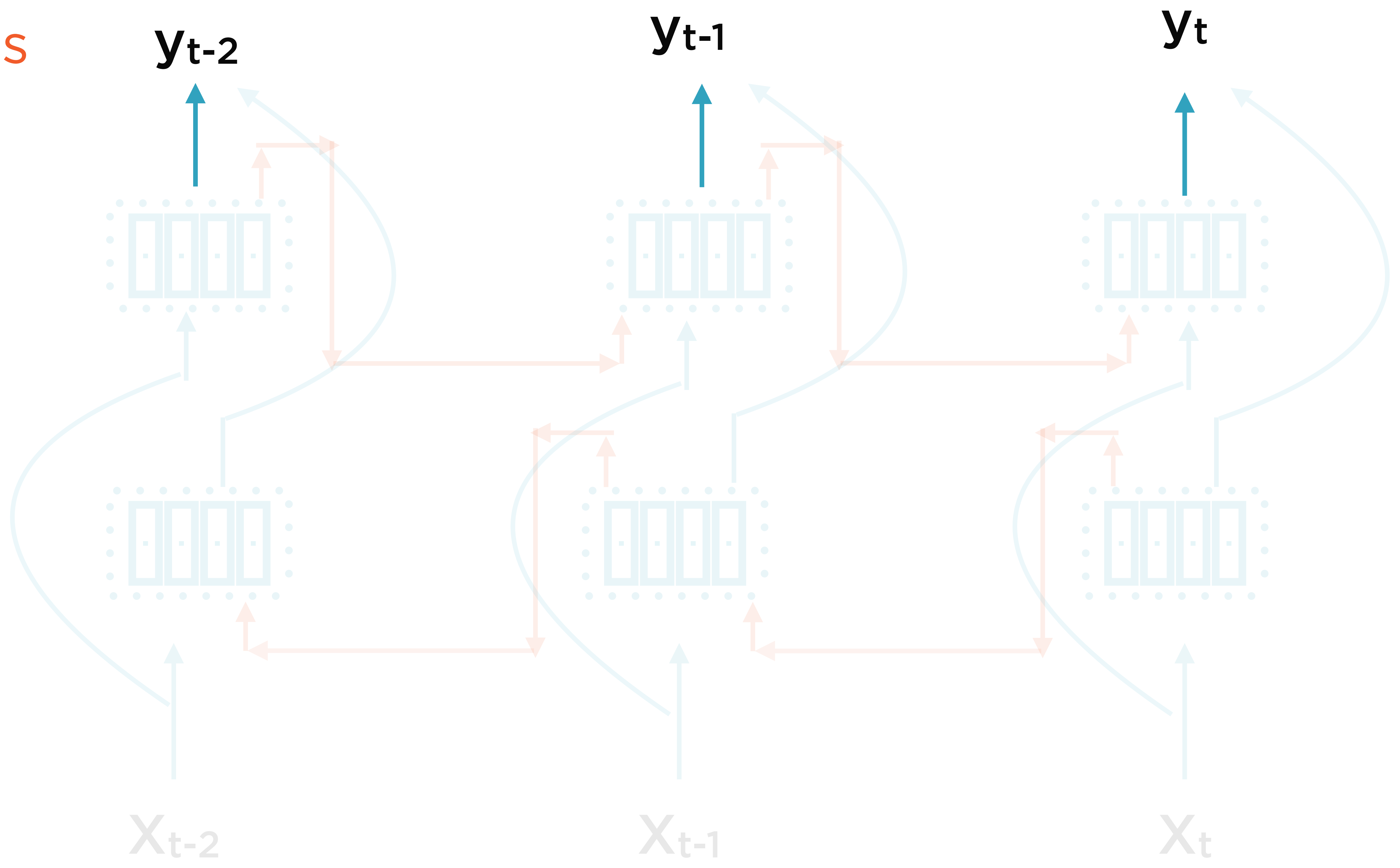


Bidirectional RNNs

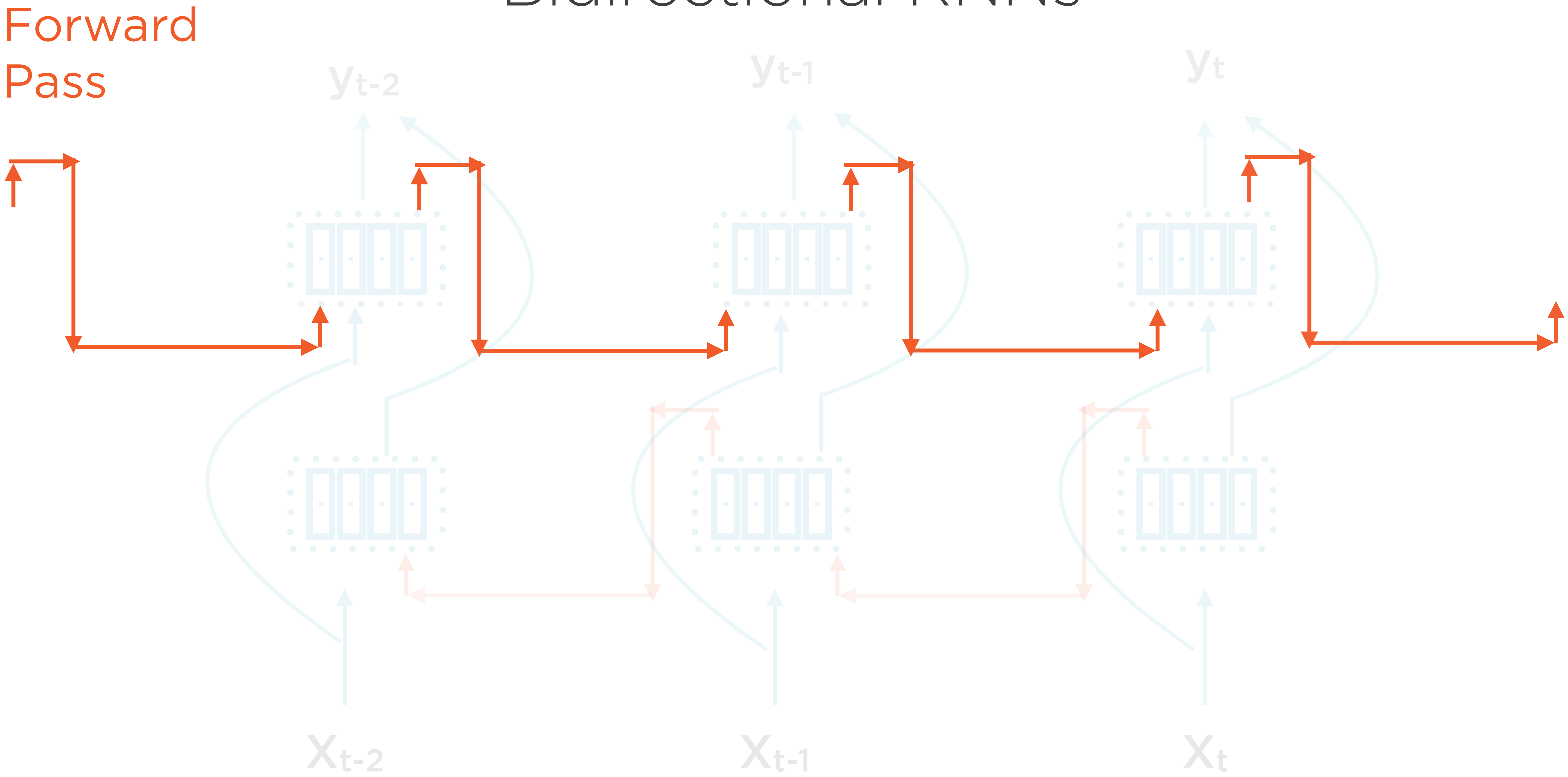


Bidirectional RNNs

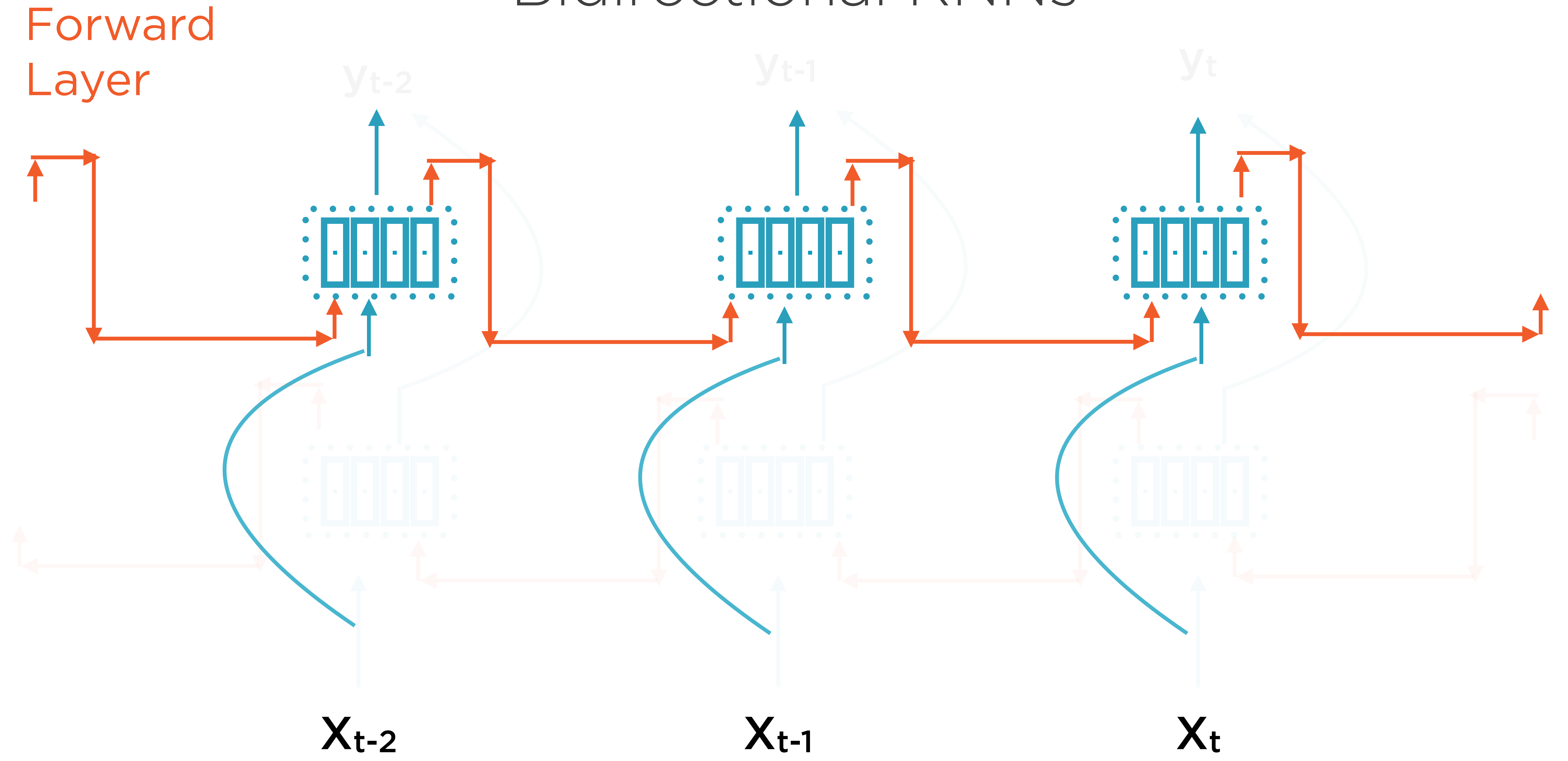
Outputs



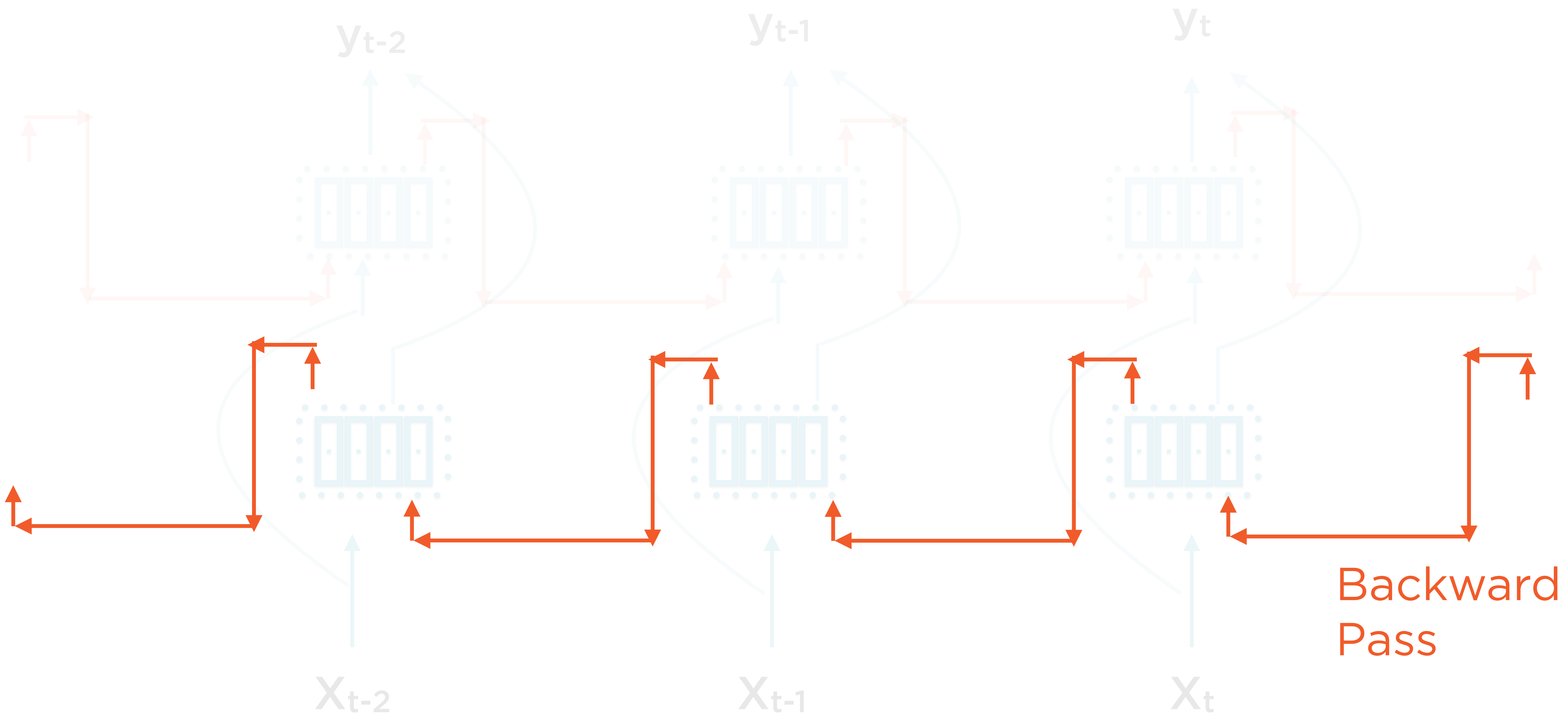
Bidirectional RNNs



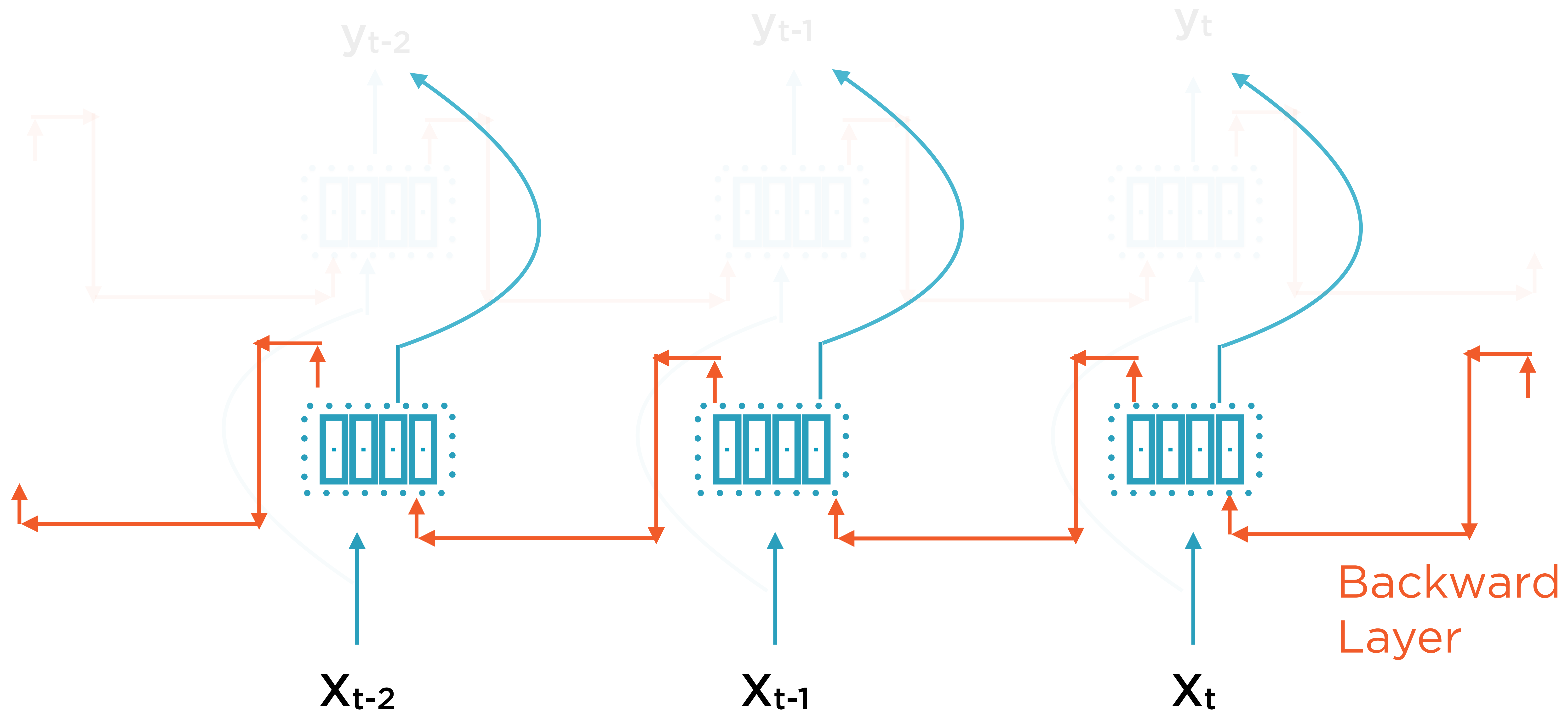
Bidirectional RNNs



Bidirectional RNNs

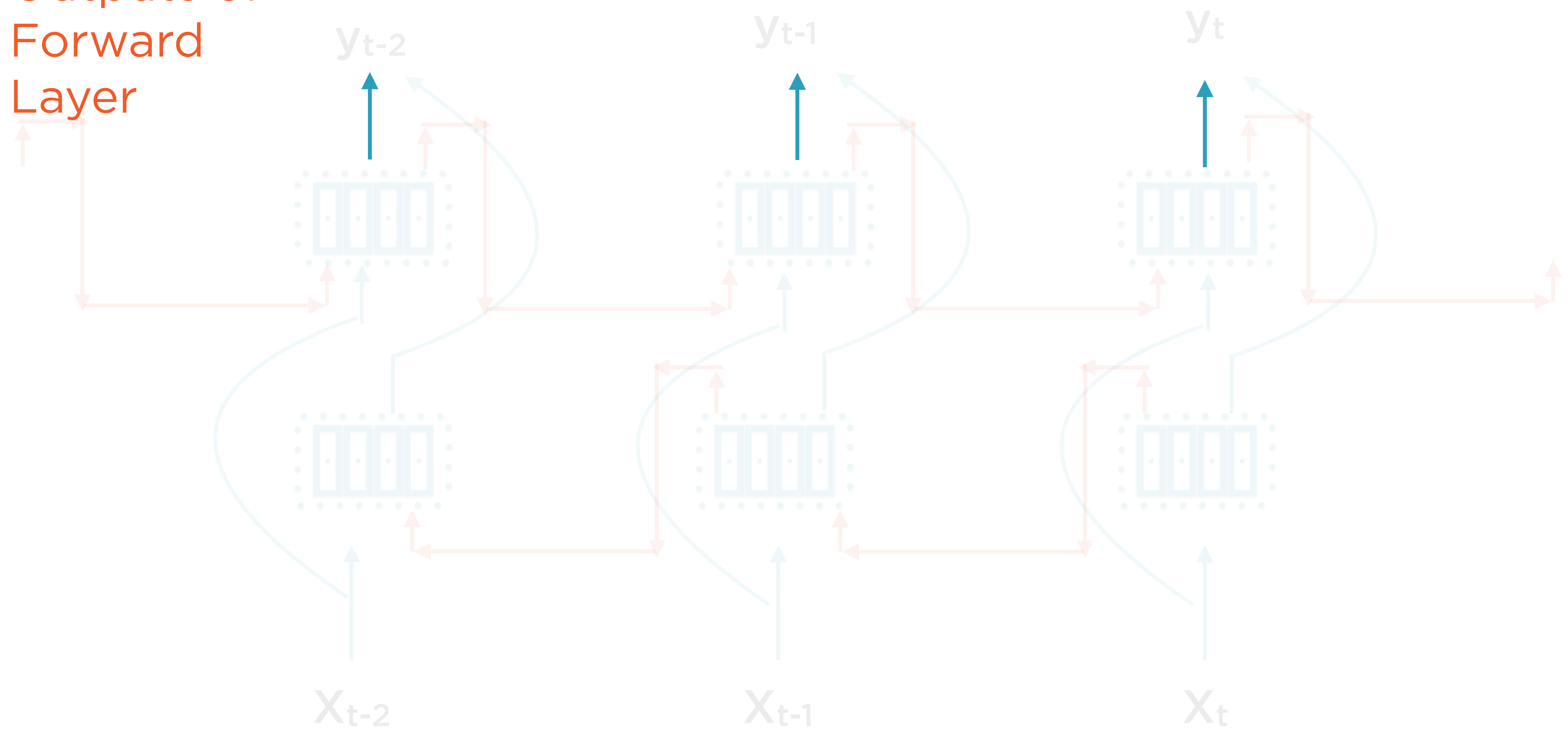


Bidirectional RNNs



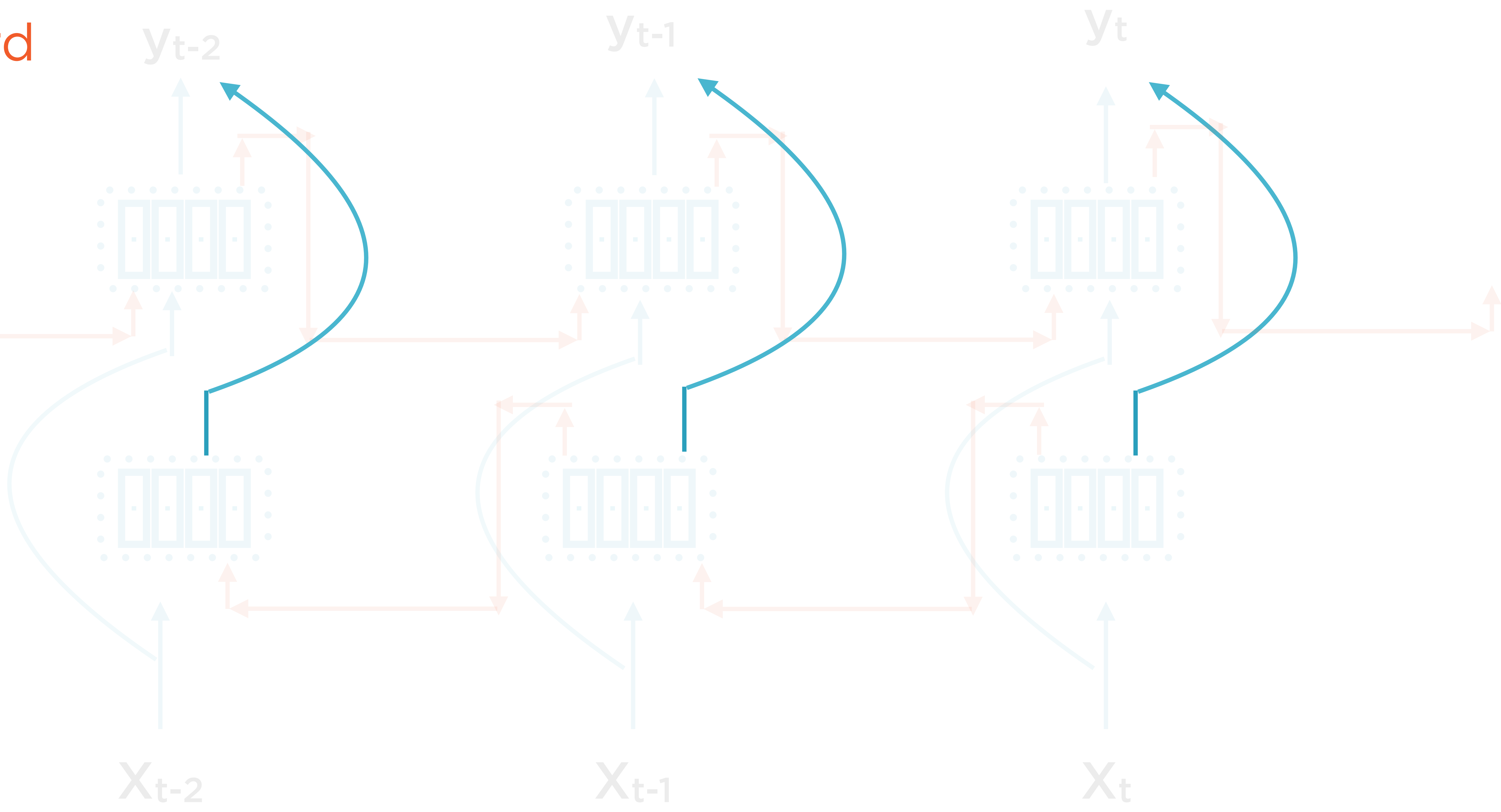
Bidirectional RNNs

Outputs of Forward Layer

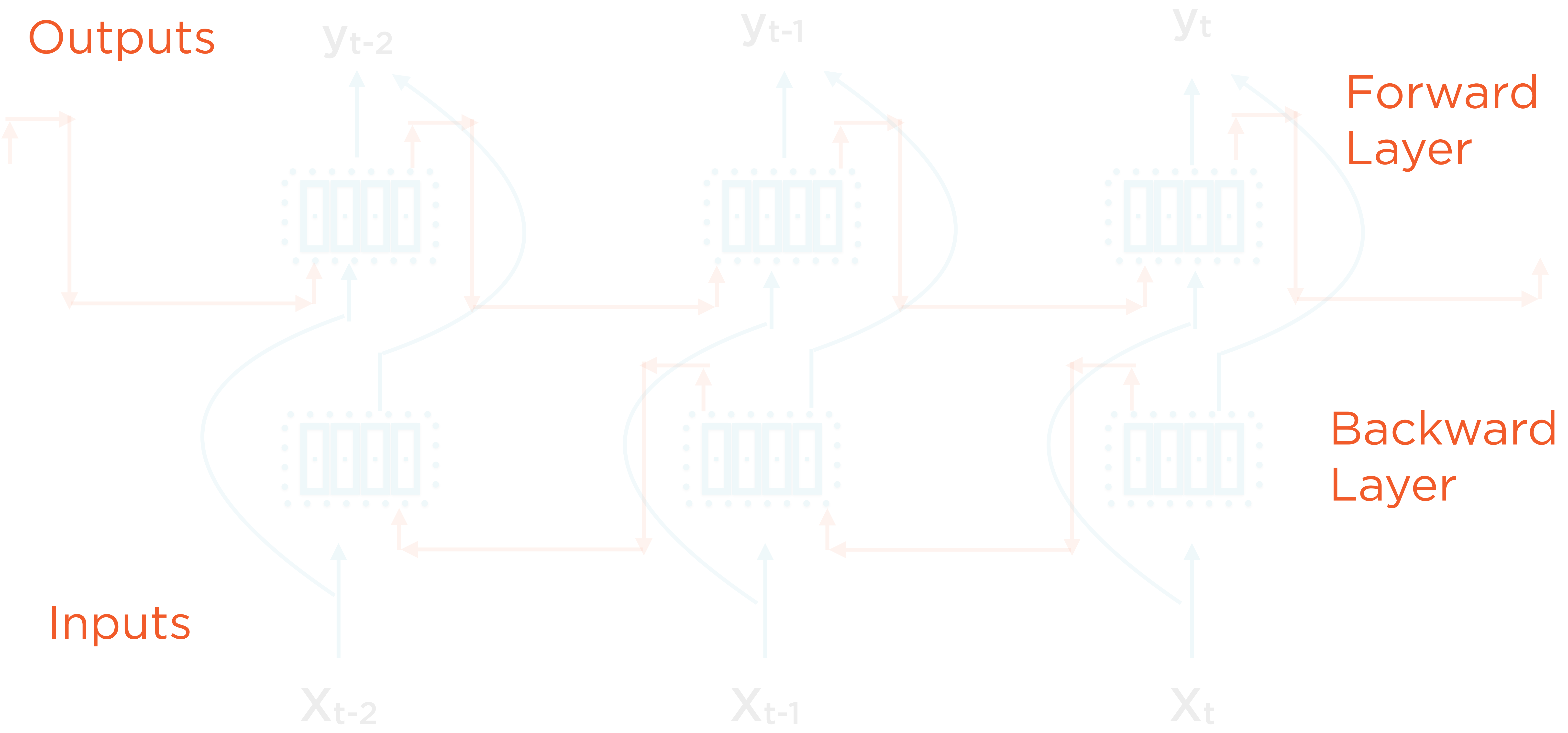


Bidirectional RNNs

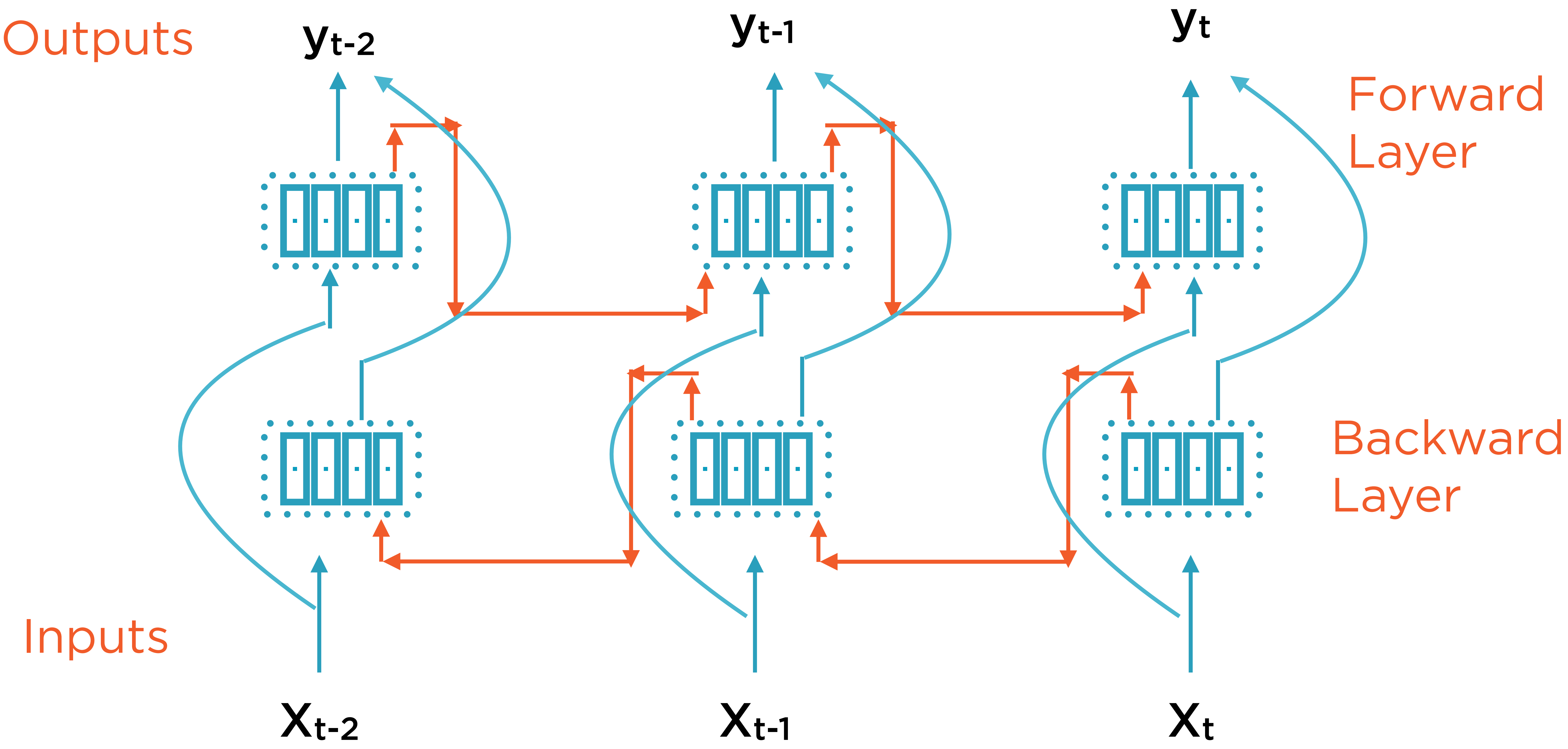
Outputs of Backward Layer



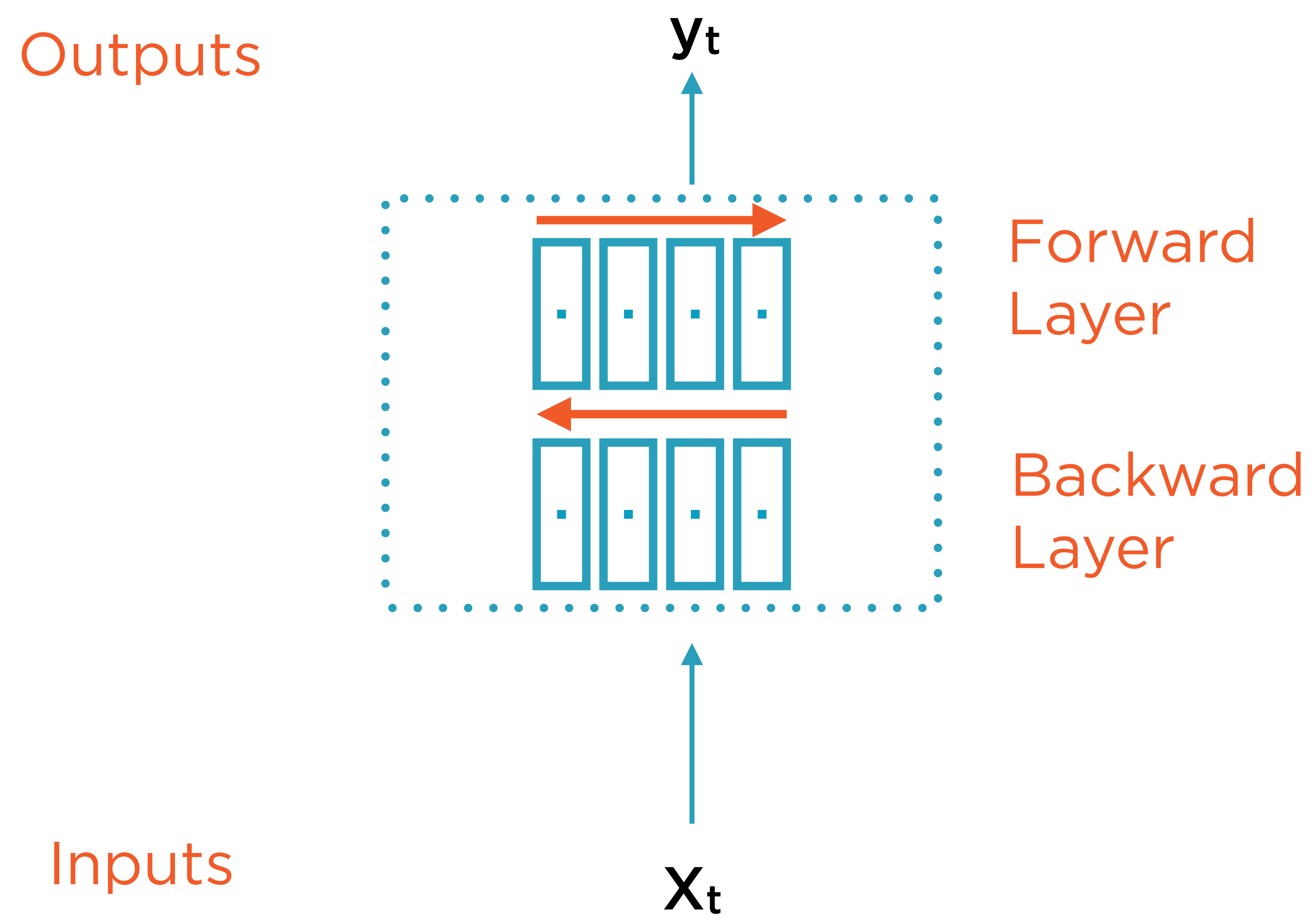
Bidirectional RNNs



Bidirectional RNNs



Bidirectional RNNs



Forward and backward outputs can be **combined** together in some manner

Demo

Sentiment analysis using word embeddings

Summary

Representing textual data as features for machine learning

Count, frequency, and probability-based embeddings

GloVe and word2vec for pre-trained word embeddings

Performing sentiment analysis using pre-trained word embeddings

Bi-directional RNNs for sentiment analysis