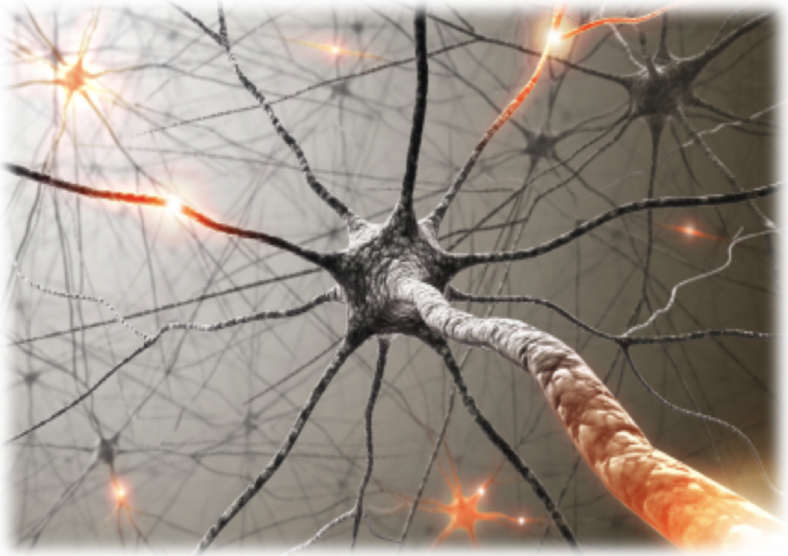
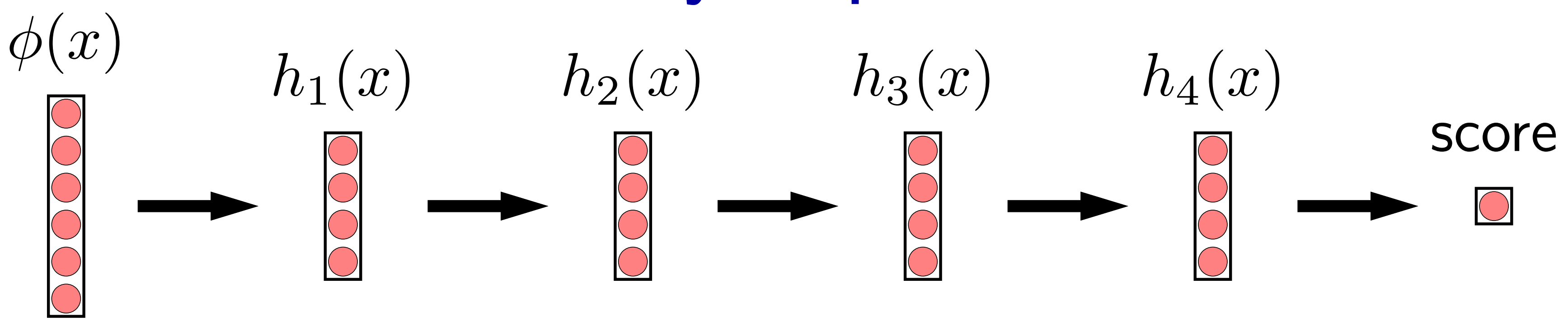
Machine learning: differentiable programming



Why depth?

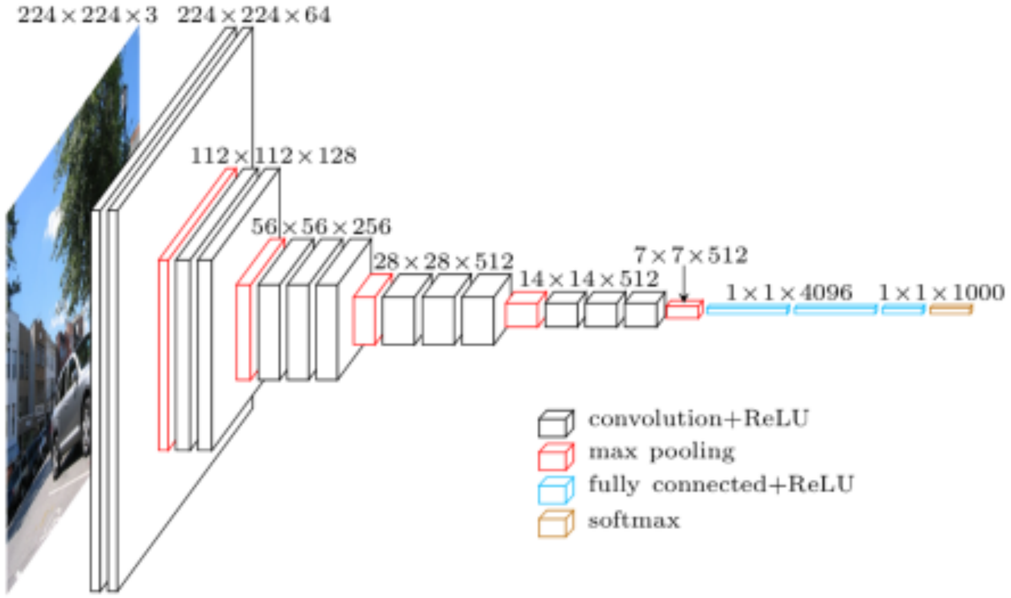


Intuitions:

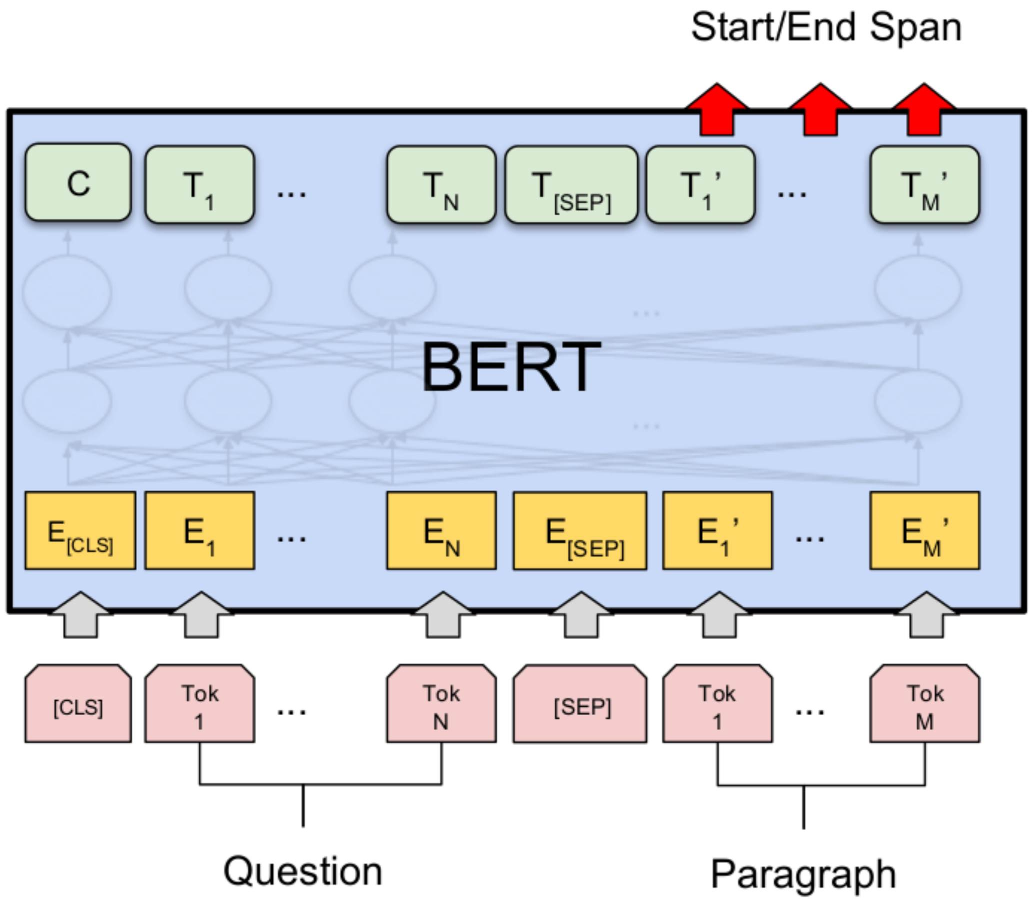
- Multiple levels of abstraction
- Multiple steps of computation
- Empirically works well
- Theory is still incomplete

Deep learning models

VGGNet [Simoyan/Zisserman, 2014]



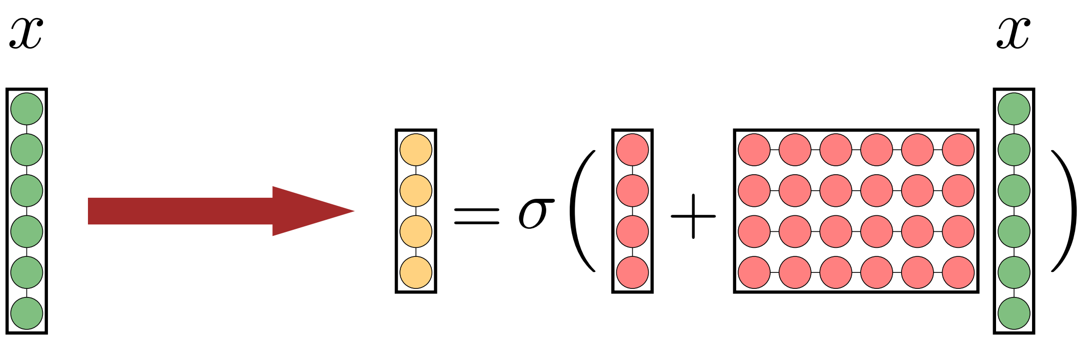
BERT [Devlin+, 2018]



Feedforward neural networks

$$\text{score} = b_3 + \mathbf{V}_3 \, \sigma\left(b_2 + \mathbf{V}_2 \, \sigma\left(b_1 + \mathbf{V}_1 \, \phi(x) \right)\right)$$

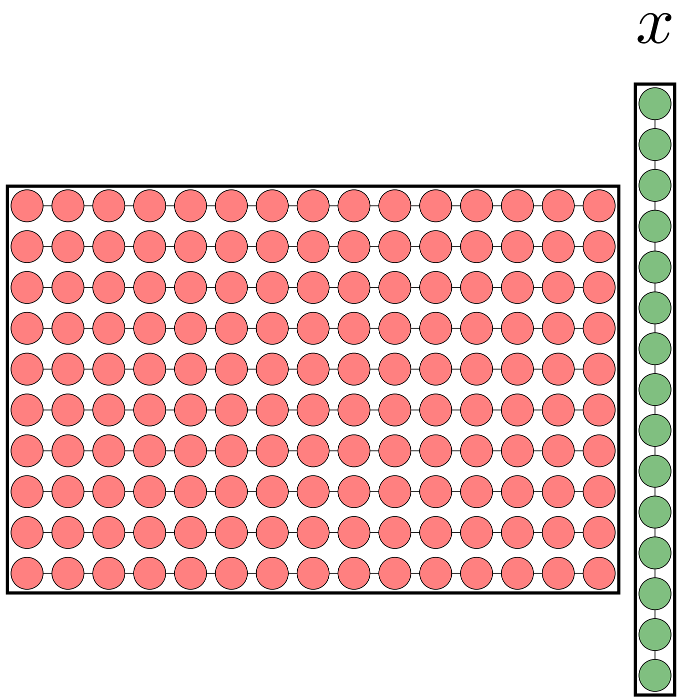
FeedForward



Performs one step of processing.

$$\text{score} = \textbf{FeedForward}(\textbf{FeedForward}(\textbf{FeedForward}(\phi(x)))) = \textbf{FeedForward}^3(\phi(x))$$

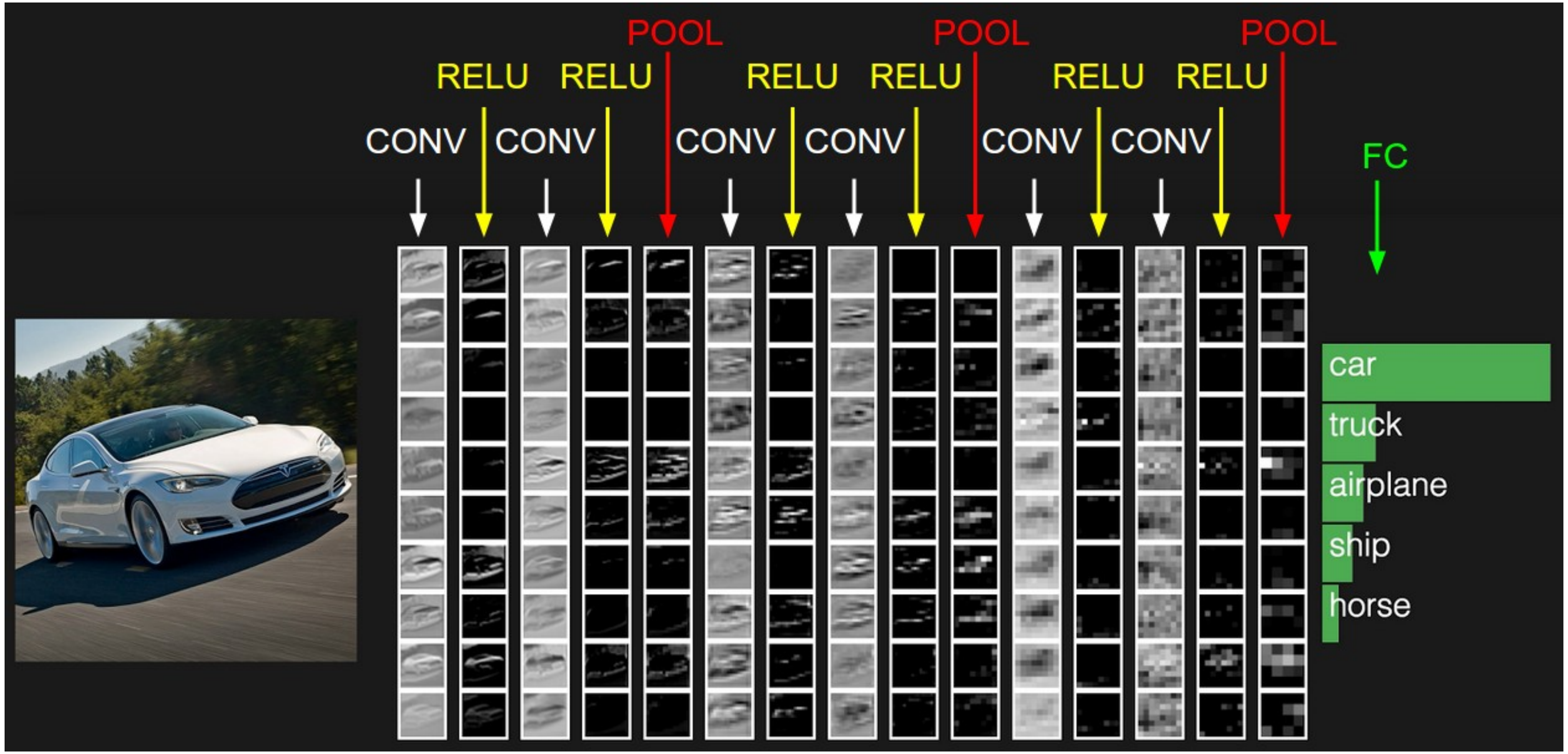
Representing images



Problems:

- Matrix is huge (depending on resolution of image)
- Does not capture the spatial structure (locality) of images

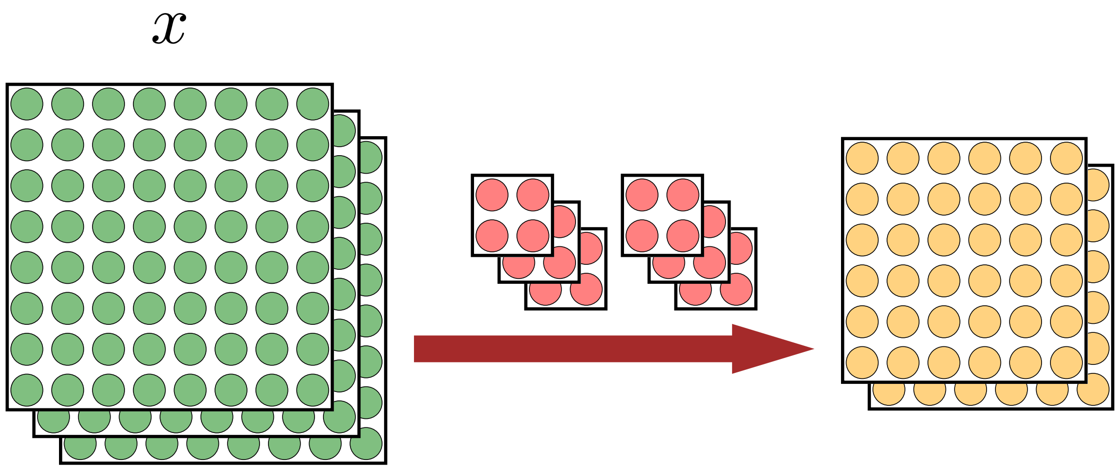
Convolutional neural networks



[Andrej Karpathy's demo]

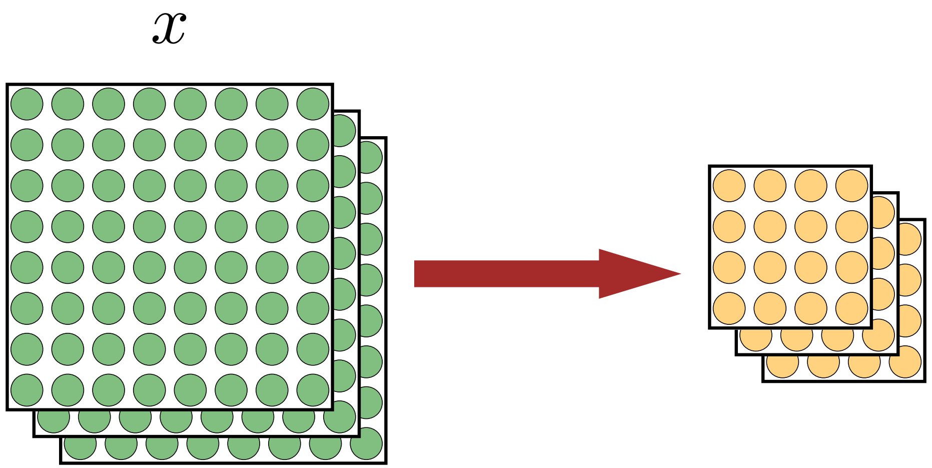
Convolutional neural networks

Conv



Detect local patterns with filters.

MaxPool



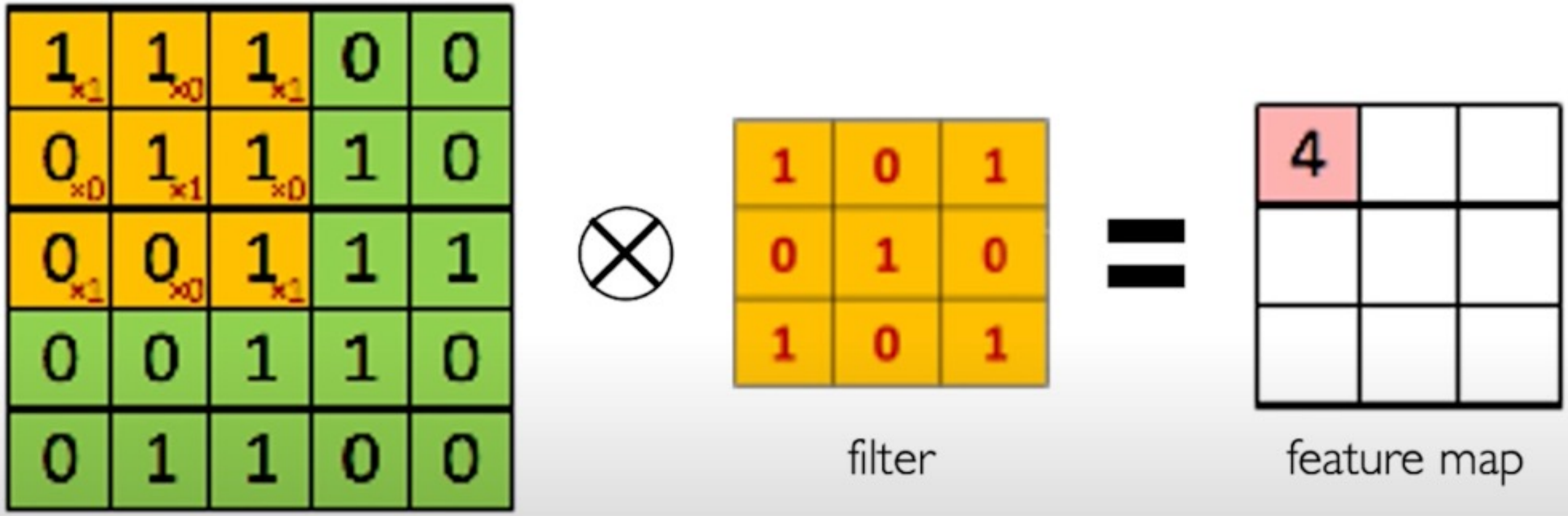
Take the max over each 2x2 region.

[Andrej Karpathy's demo]

$$\textbf{AlexNet}(x) = \textbf{FeedForward}^3(\textbf{MaxPool}(\textbf{Conv}^3(\textbf{MaxPool}(\textbf{Conv}(\textbf{MaxPool}(\textbf{Conv}(x)))))))$$

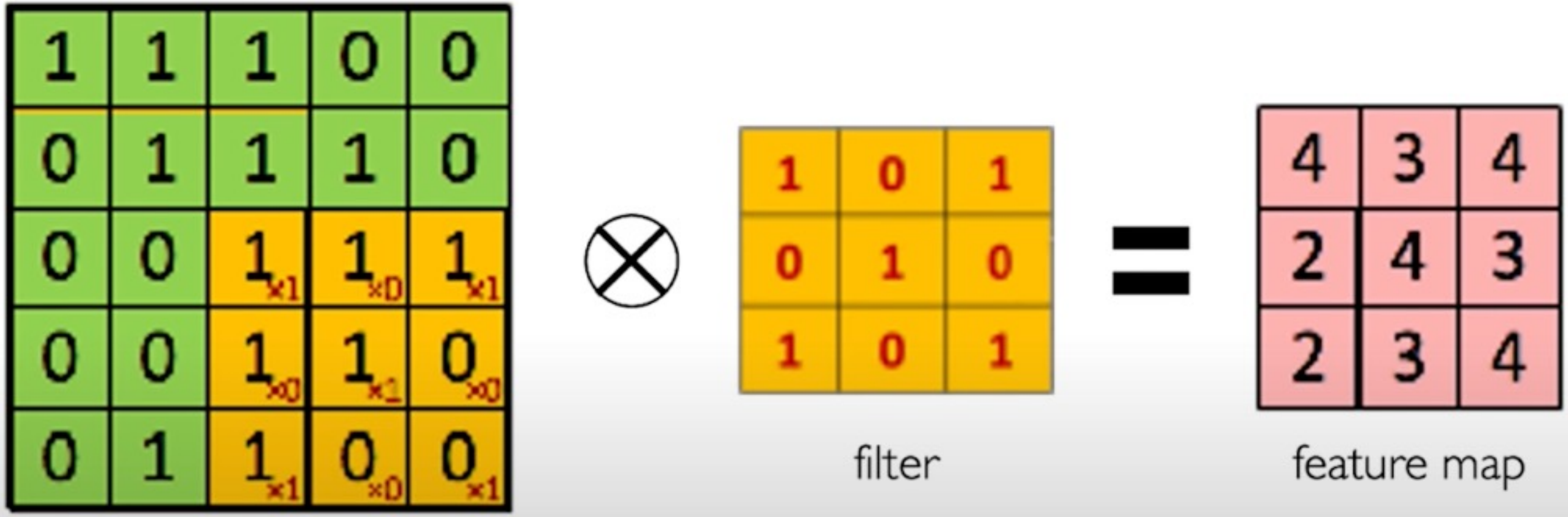
The Convolution Operation

We slide the 3x3 filter over the input image, element-wise multiply, and add the outputs:



The Convolution Operation

We slide the 3x3 filter over the input image, element-wise multiply, and add the outputs:



Producing Feature Maps

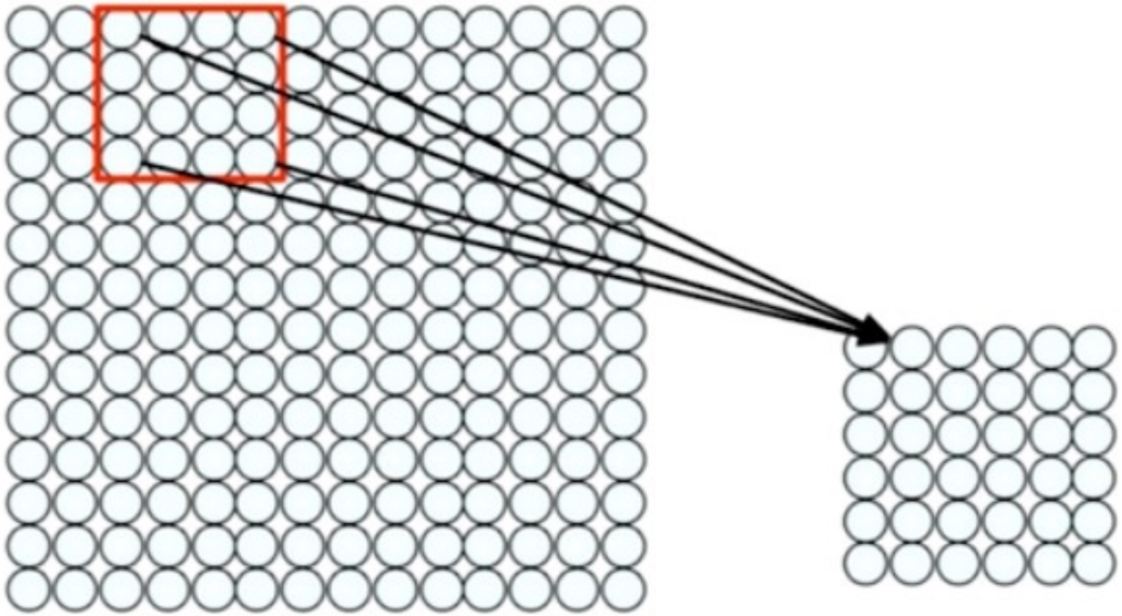
Original

Sharpen

Edge Detect

"Strong" Edge Detect

Feature Extraction with Convolution

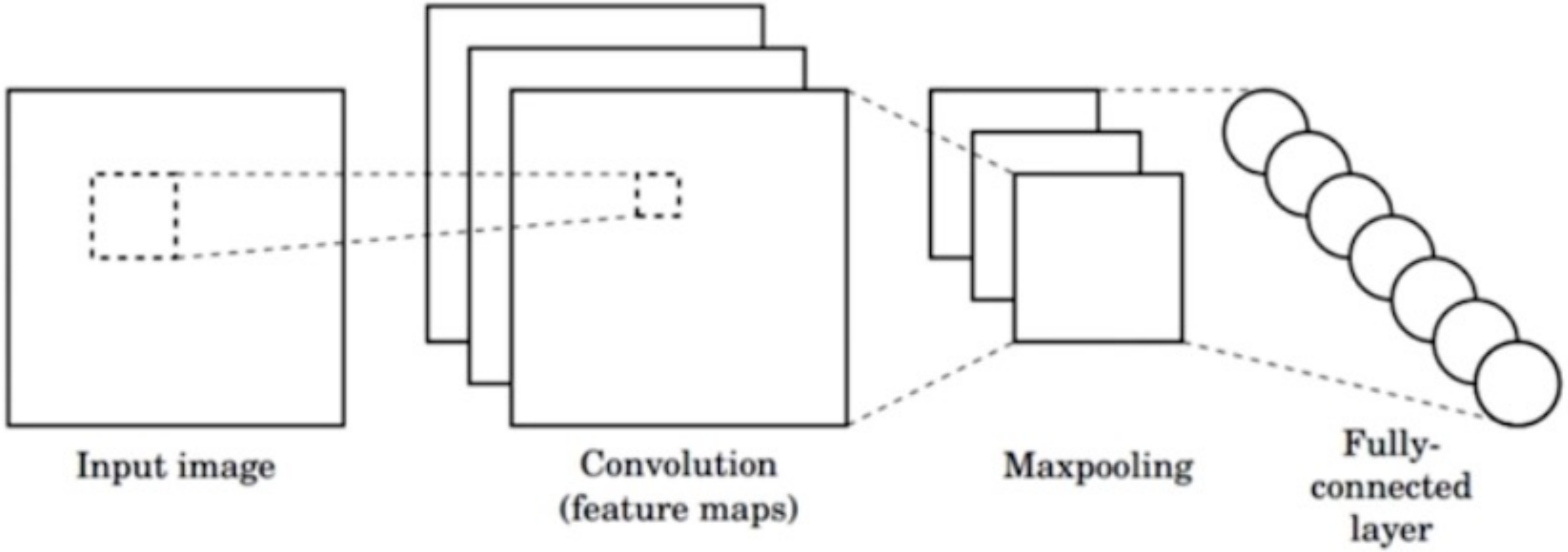


1) Apply a set of weights – a filter – to extract **local features**

2) Use **multiple filters** to extract different features

3) **Spatially share** parameters of each filter

CNNs for Classification



1. **Convolution**: Apply filters to generate feature maps.
2. **Non-linearity**: Often ReLU.
3. **Pooling**: Downsampling operation on each feature map.

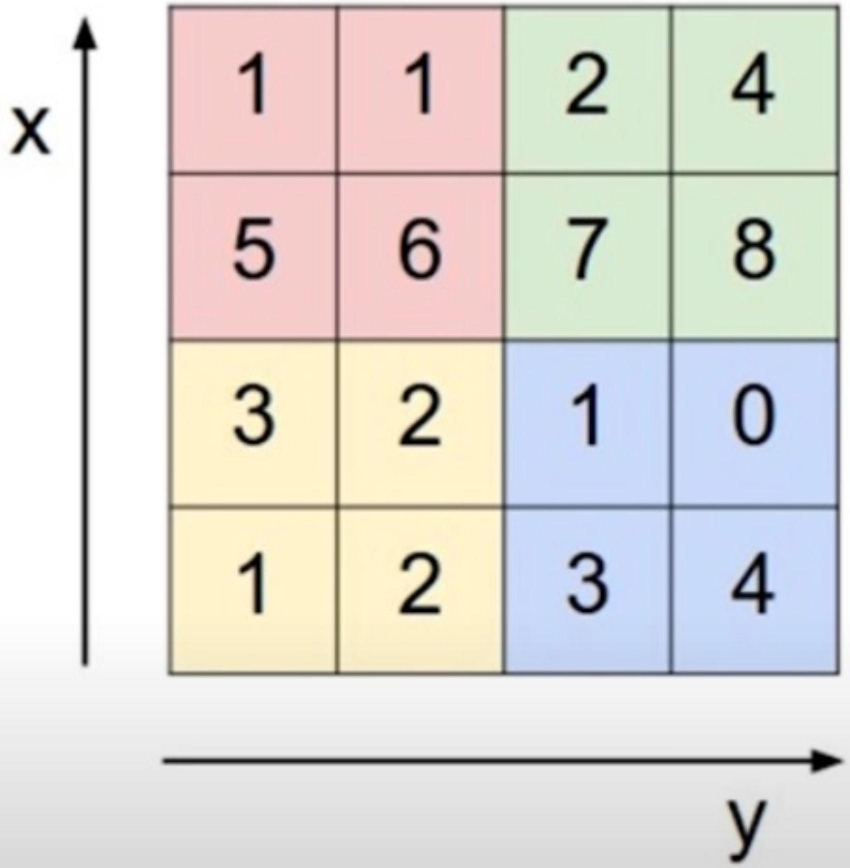
```
tf.keras.layers.Conv2D
```

```
tf.keras.activations.*
```

```
tf.keras.layers.MaxPool2D
```

Train model with image data.
Learn weights of filters in convolutional layers.

Pooling



max pool with 2x2 filters
and stride 2

```
tf.keras.layers.MaxPool2D(
    pool_size=(2,2),
    strides=2
)
```

1) Reduced dimensionality
2) Spatial invariance

Representing natural language

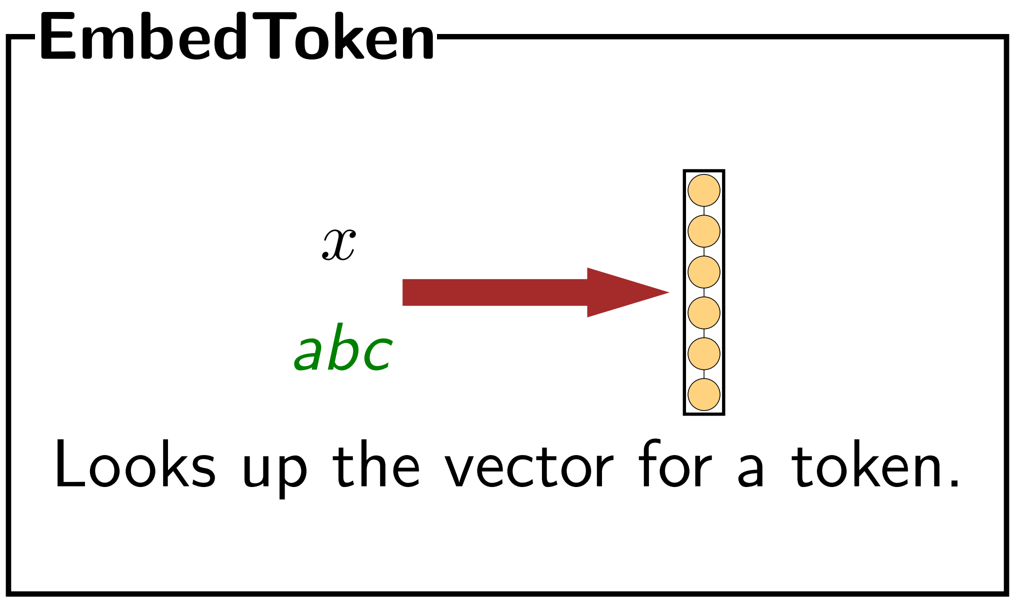
In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under **gravity**. The main forms of precipitation include drizzle, rain, sleet, snow, **graupel** and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals **within a cloud**. Short, intense periods of rain in scattered locations are called "showers".

What causes precipitation to fall?
gravity

What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?
graupel

Where do water droplets collide with ice crystals to form precipitation?
within a cloud

Embedding tokens

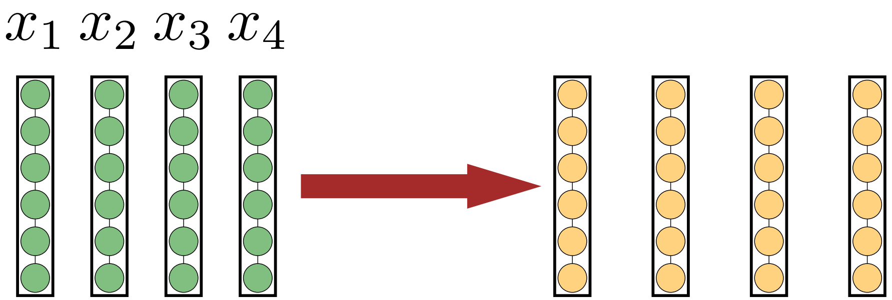


In meterology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravity.

Meaning of words/tokens depends on context...

Representing sequences

SequenceModel



Process each element of a sequence with respect to other elements.

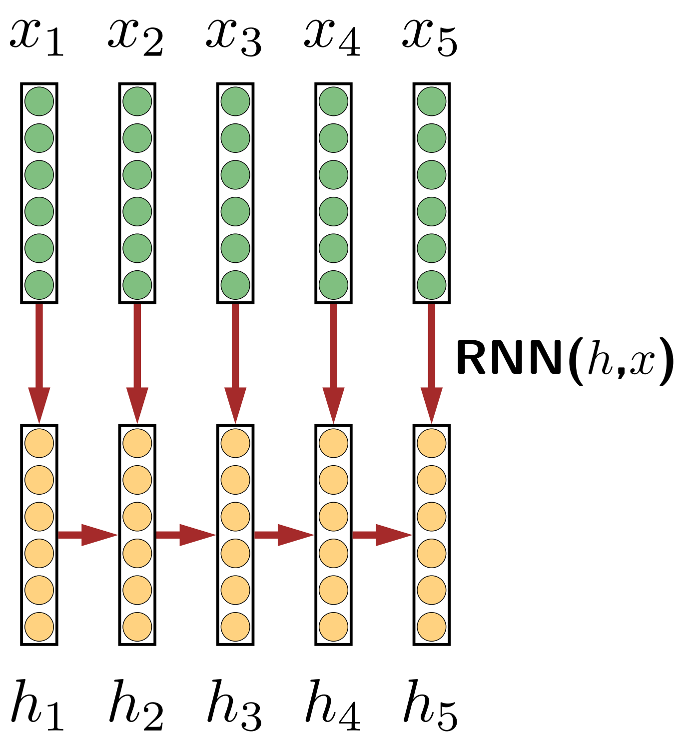
Two implementations:

- Recurrent neural networks
- Transformers

Recurrent neural networks

In meterology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravity.

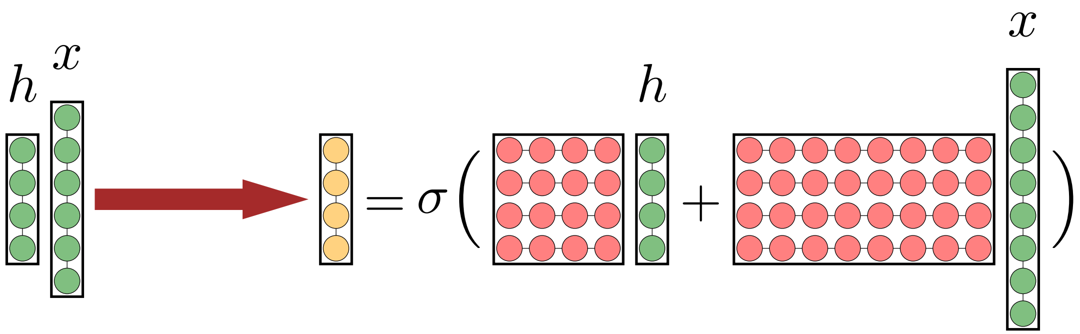
SequenceRNN : SequenceModel



Read left-to-right, updating hidden state.

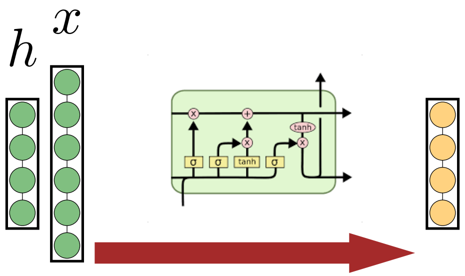
Recurrent neural networks

SimpleRNN : RNN



Update hidden state given a new input.

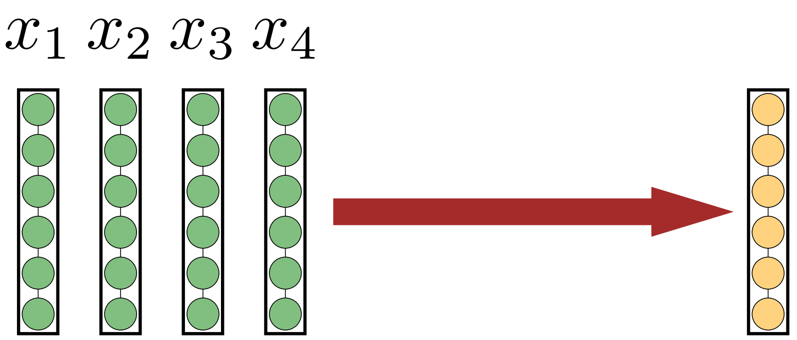
LSTM : RNN



Update hidden state given a new input without forgetting the past.

Collapsing to a single vector

Collapse



Summarize using one vector (first, last, or average).

Example text classification model:

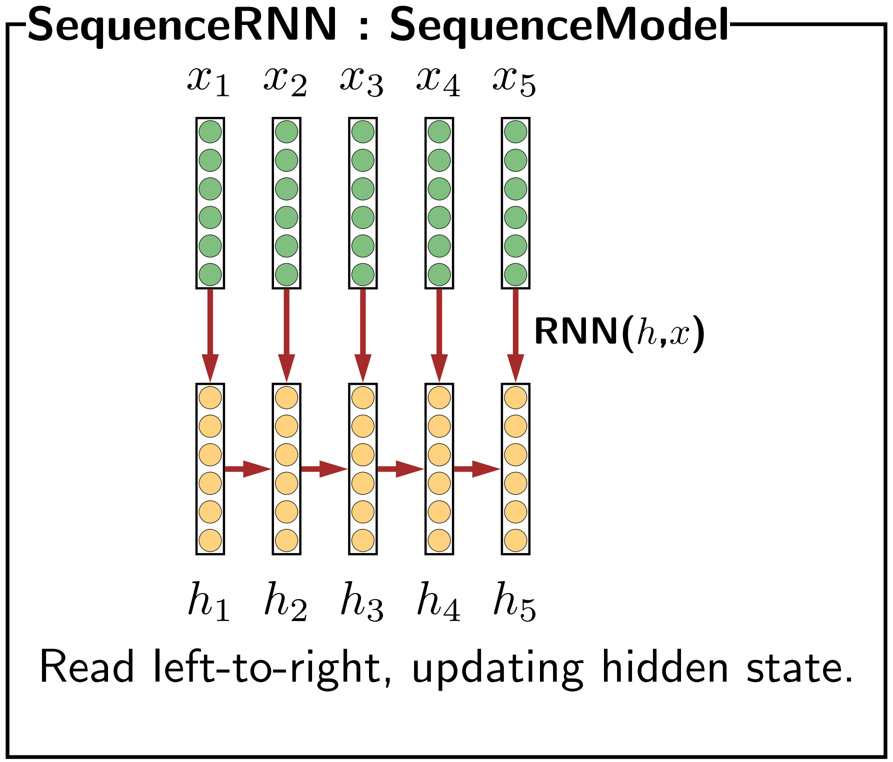
In meterology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravity.

$$\text{score} = \mathbf{w} \cdot \textbf{Collapse}(\textbf{SequenceModel}^3(\textbf{EmbedToken}(\mathbf{x})))$$

Long-range dependencies

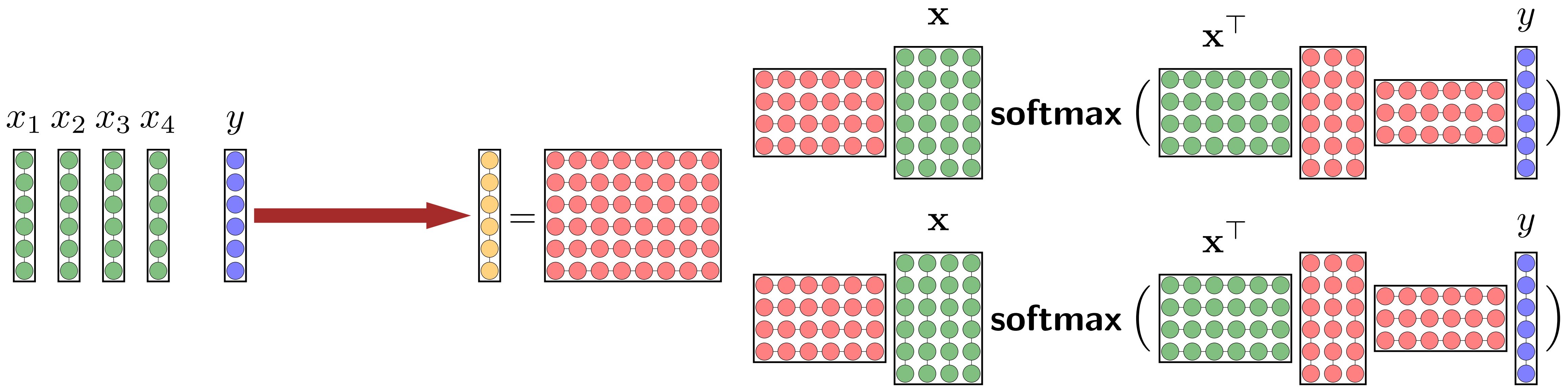
[CLS] What causes precipitation to fall? [SEP] In meterology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravity.

Problem: RNN (and ConvNets) are very local



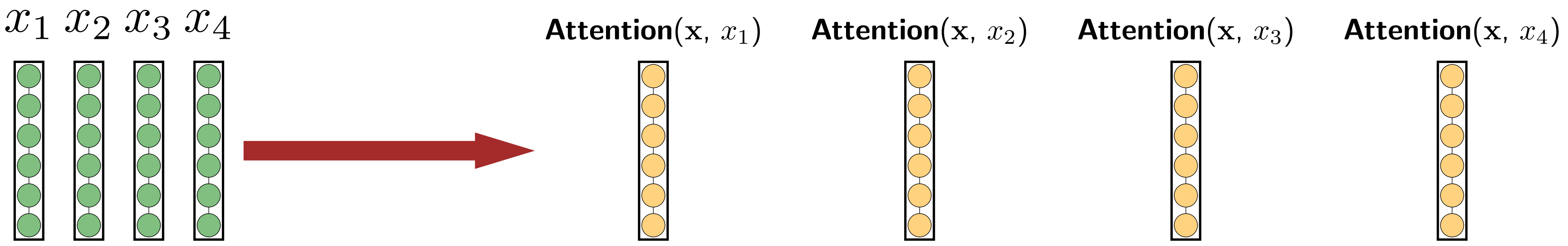
Attention mechanism

Attention



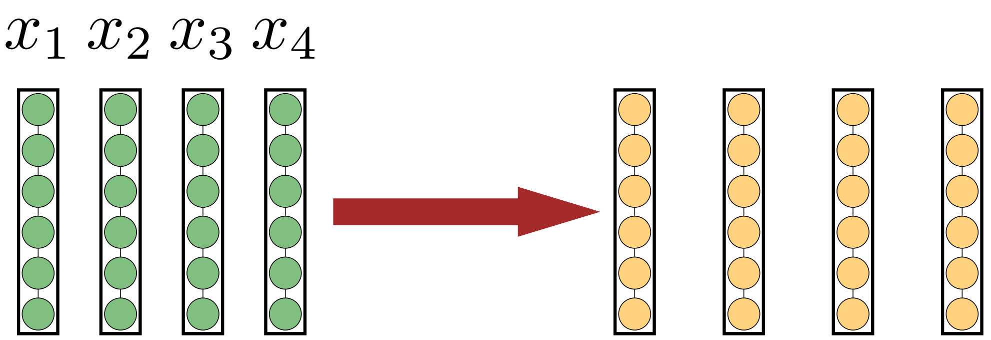
Process y by comparing it to each x_i.

Attention : SequenceModel



Layer normalization and residual connections

AddNorm(f) : SequenceModel

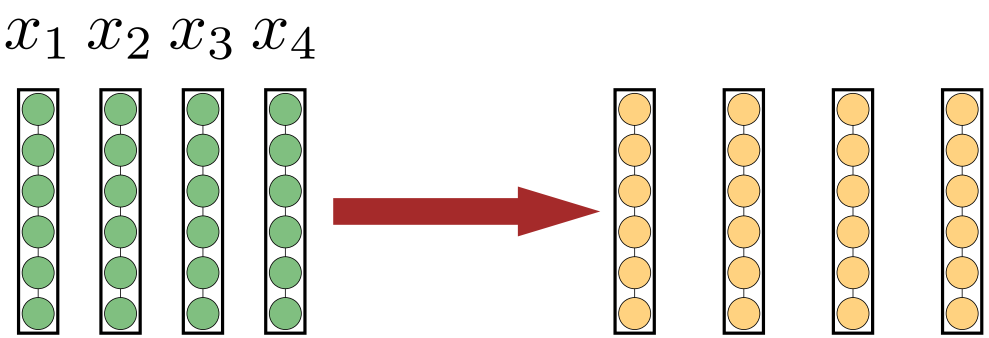


Applies f to $\mathbf{x}$ safely.

AddNorm(f, $\mathbf{x}$) $=$ **LayerNorm**($\mathbf{x} + f(\mathbf{x})$)

Transformer

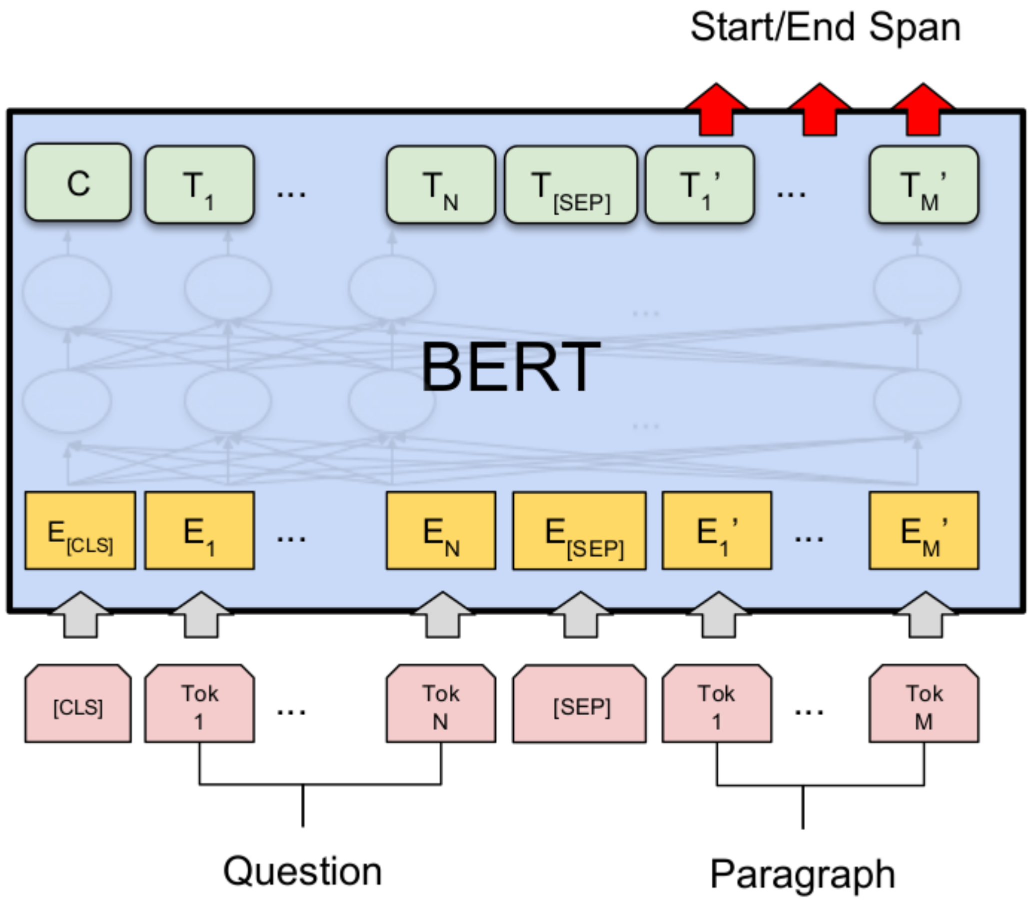
TransformerBlock : SequenceModel



Processes each object x_i in context.

$\textbf{TransformerBlock}(\mathrm{x}) = \textbf{AddNorm}(\textbf{FeedForward}, \textbf{AddNorm}(\textbf{Attention}, \mathrm{x}))$

BERT

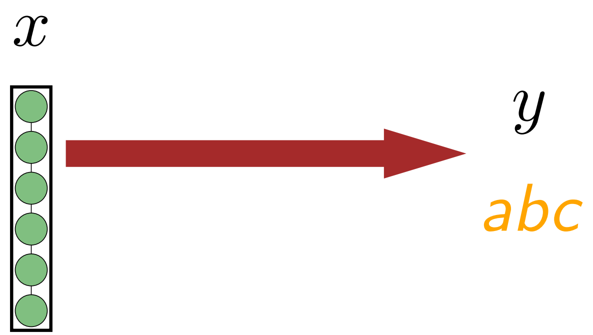


[CLS] What causes precipitation to fall? [SEP] In meterology, precipitation is any product of the condensation of atmospheric water vapor that falls under gravity.

$$\textbf{BERT}(\mathrm{x}) = \textbf{TransformerBlock}^{24}(\textbf{EmbedToken}(\mathrm{x}))$$

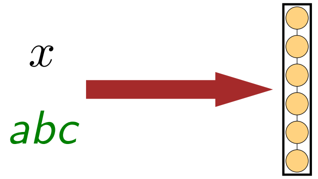
Generating tokens

GenerateToken



Generate token y based on $x\cdot$ **EmbedToken**(y).

EmbedToken



Looks up the vector for a token.

Generating sequences

LanguageModel

$\mathbf{x}$

the quick brown $\longrightarrow$ *fox*

Generate next token in the sequence.

$$\textbf{LanguageModel}(\mathbf{x}) = \textbf{GenerateToken}(\textbf{Collapse}(\textbf{SequenceModel}(\textbf{EmbedToken}(\mathbf{x}))))$$

Sequence-to-sequence models

SequenceToSequence

Generate a sequence from another sequence.

Applications:

- Machine translation: sentence to translation
- Document summarization: document to summary
- Semantic parsing: sentence to code

Summary

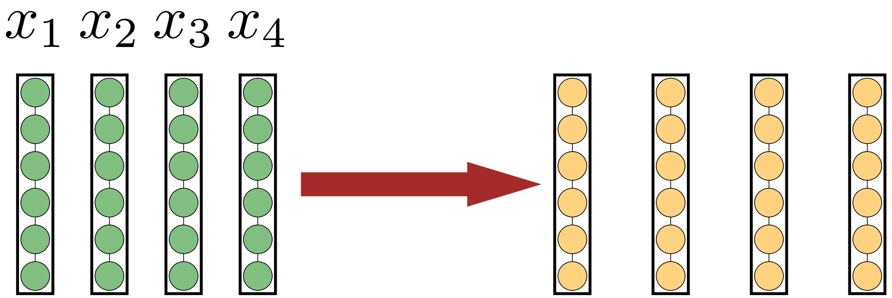
FeedForward Conv MaxPool

EmbedToken SequenceRNN SimpleRNN LSTM

Attention AddNorm TransformerBlock BERT

Collapse GenerateToken LanguageModel SequenceToSequence

SequenceModel



Process each element of a sequence with respect to other elements.