UNIVERSITY OF TECHNOLOGY
IN THE EUROPEAN CAPITAL OF CULTURE
CHEMNITZ

Neurocomputing

Natural Language Processing

Julien Vitay

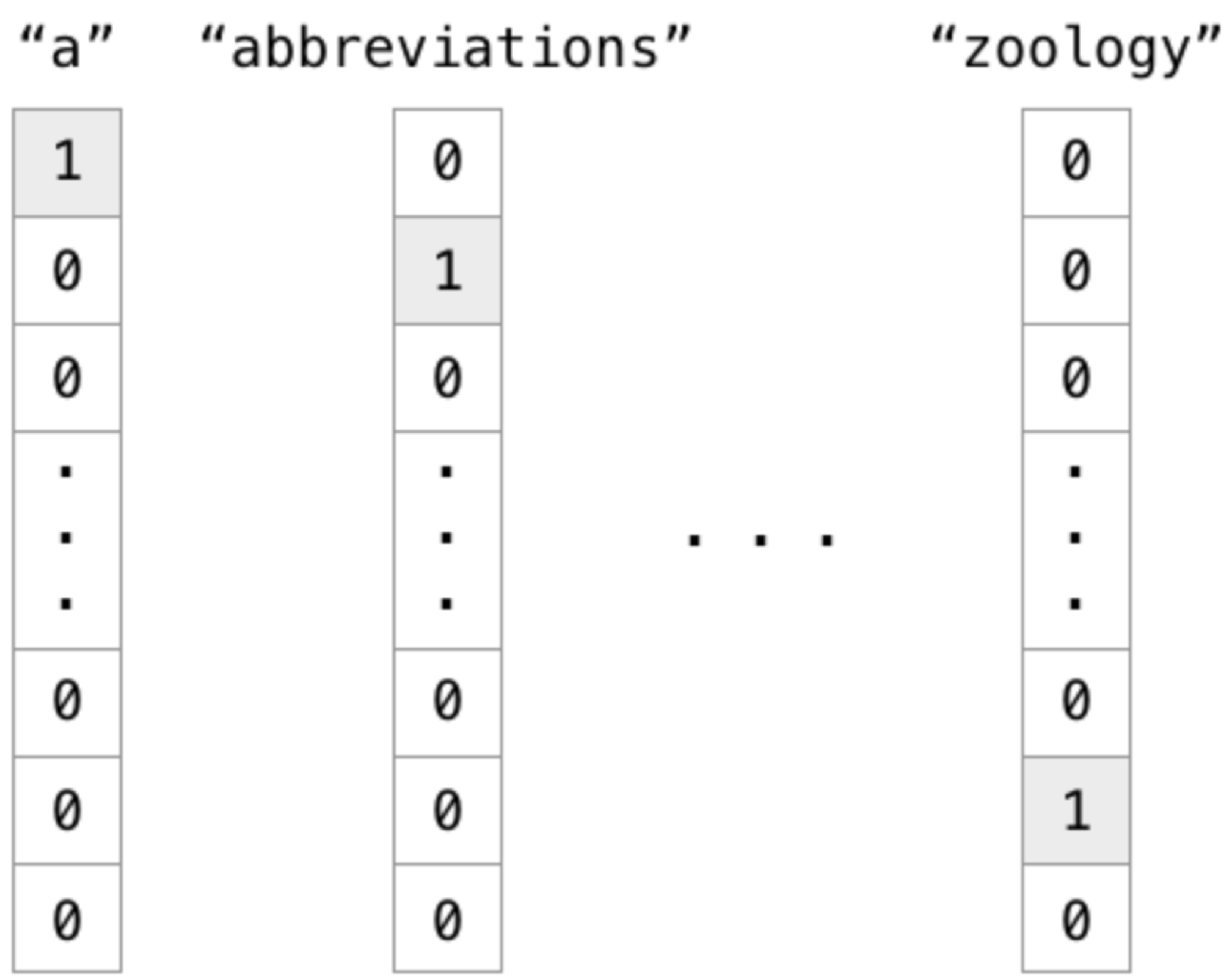
Professur für Künstliche Intelligenz - Fakultät für Informatik

https://tu-chemnitz.de/informatik/KI/edu/neurocomputing

1 - word2vec

Representing words

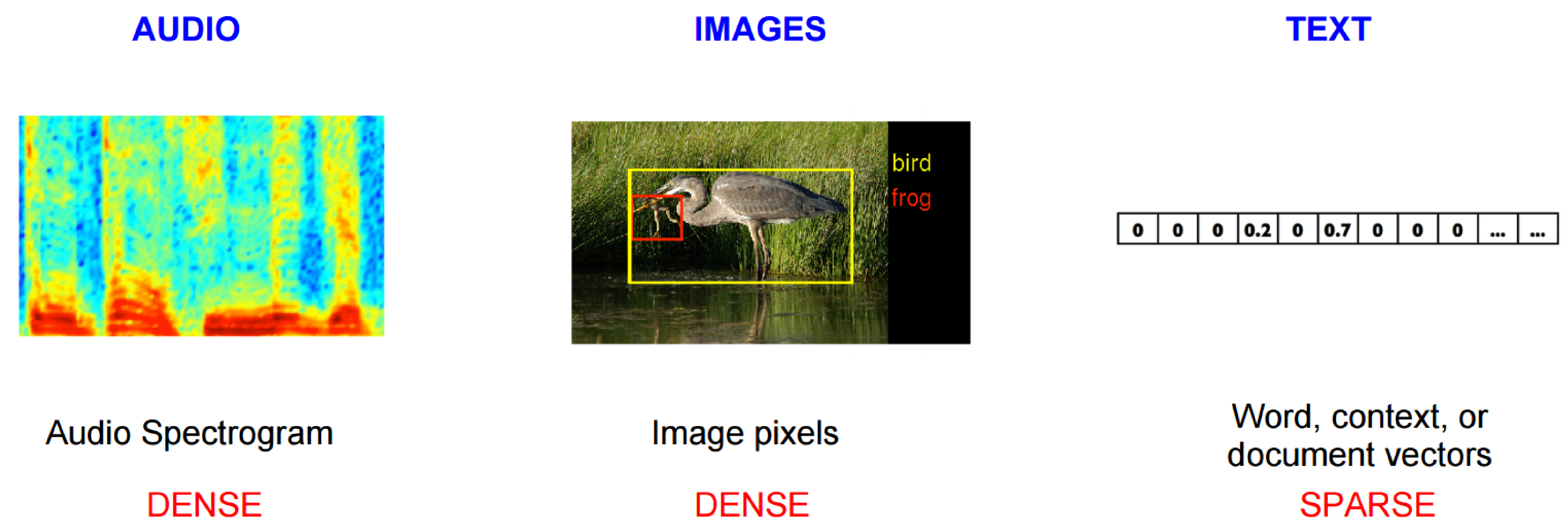
- The most famous application of RNNs is **Natural Language Processing** (NLP): text understanding, translation, etc…
- Each word of a sentence has to be represented as a vector $\mathbf{x}_t$ in order to be fed to a LSTM.
- Which representation should we use?
- The naive solution is to use **one-hot encoding**, one element of the vector corresponding to one word of the dictionary.



Source: https://cdn-images-1.medium.com/max/1600/1*ULfyiWPKgWceCqyZeDTl0g.png

Representing words

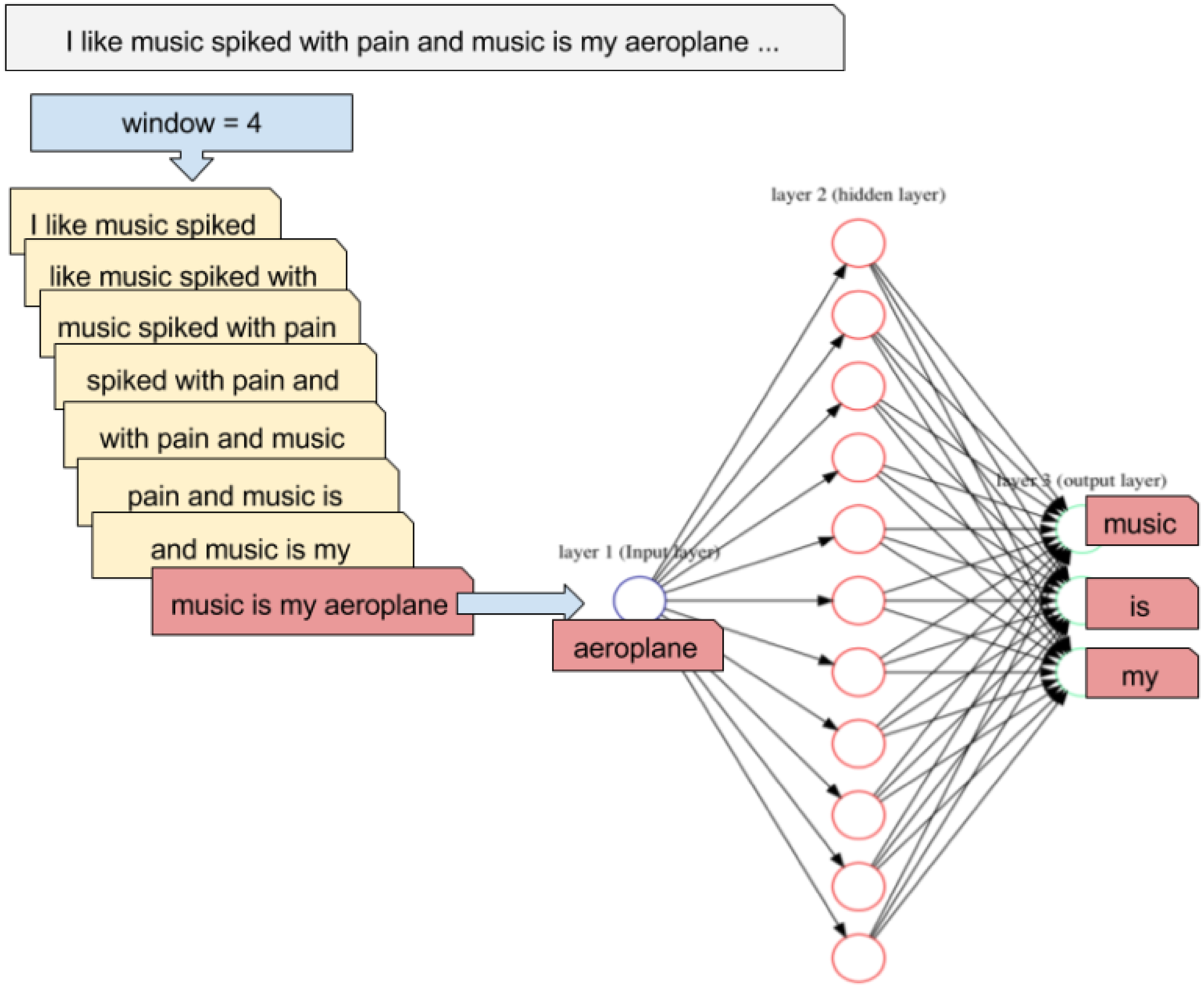
- One-hot encoding is not a good representation for words:
    - The vector size will depend on the number of words of the language:
        - English: 171,476 (Oxford English Dictionary), 470,000 (Merriam-Webster)... 20,000 in practice.
        - French: 270,000 (TILF).
        - German: 200,000 (Duden).
        - Chinese: 370,000 (Hanyu Da Cidian).
        - Korean: 1,100,373 (Woori Mal Saem)
    - Semantically related words have completely different representations (“endure” and “tolerate”).
    - The representation is extremely **sparse** (a lot of useless zeros).



Source: https://www.tensorflow.org/tutorials/representation/word2vec

word2vec

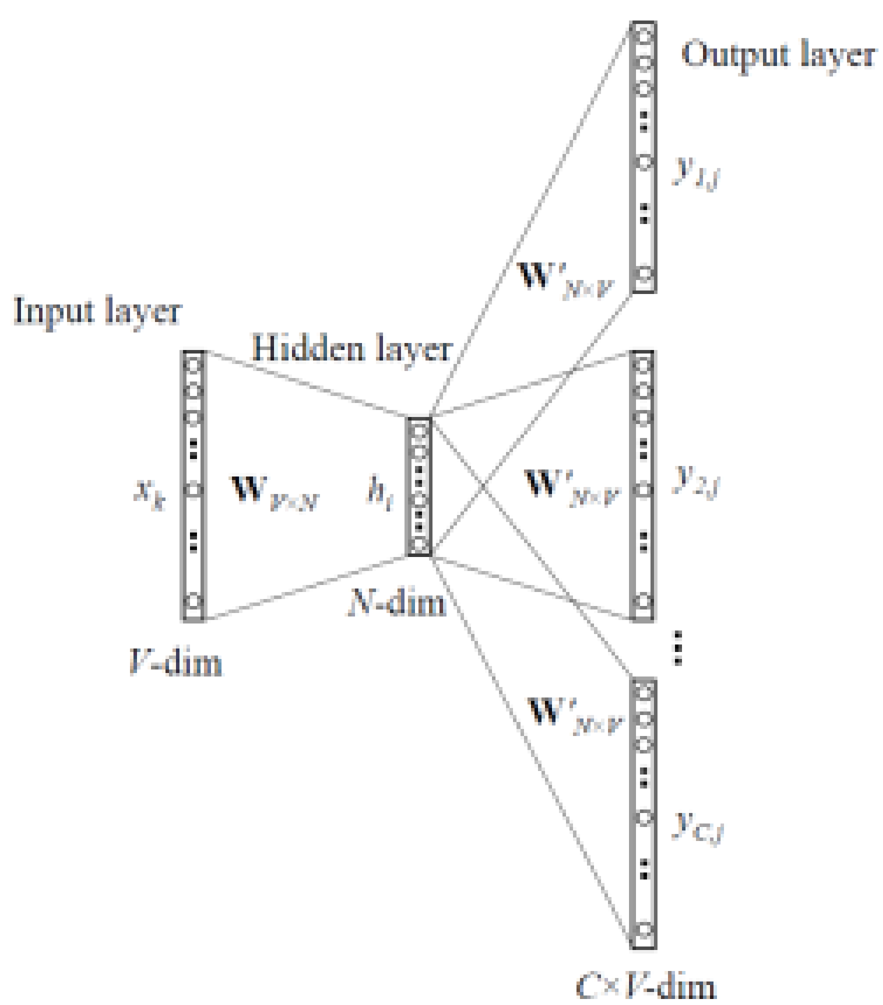
- **word2vec** learns word **embeddings** by trying to predict the current word based on the context (CBOW, continuous bag-of-words) or the context based on the current word (skip-gram).
- It uses a three-layer autoencoder-like NN, where the hidden layer (latent space) will learn to represent the one-hot encoded words in a dense manner.

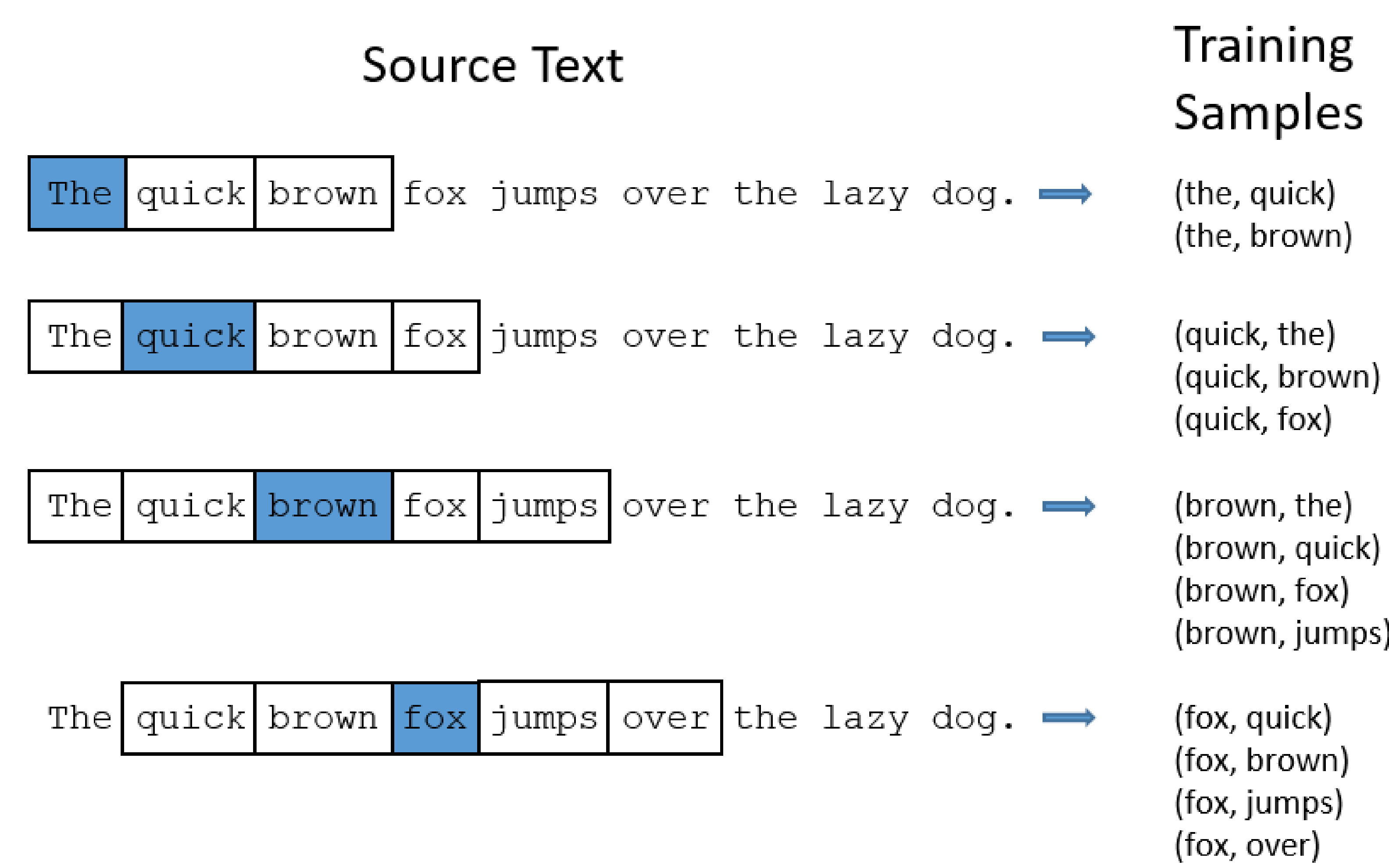


Source: https://jaxenter.com/deep-learning-search-word2vec-147782.html

word2vec

- **word2vec** has three parameters:
  - the **vocabulary size**: number of words in the dictionary.
  - the **embedding size**: number of neurons in the hidden layer.
  - the **context size**: number of surrounding words to predict.
- It is trained on huge datasets of sentences (e.g. Wikipedia).





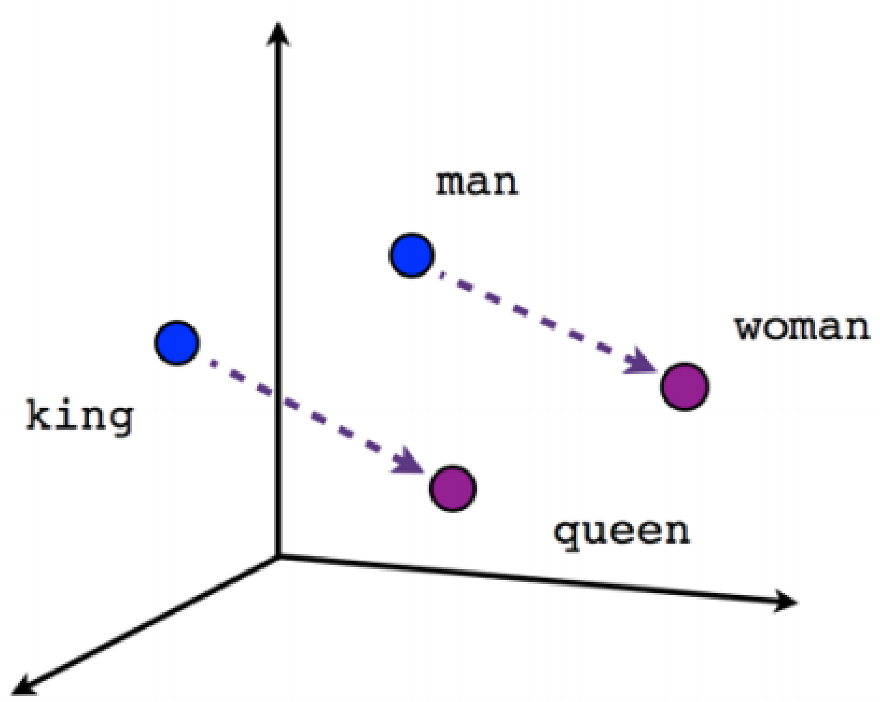
Source: https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/

Mikolov et al. (2013). Distributed Representations of Words and Phrases and their Compositionality. NIPS. https://code.google.com/archive/p/word2vec/

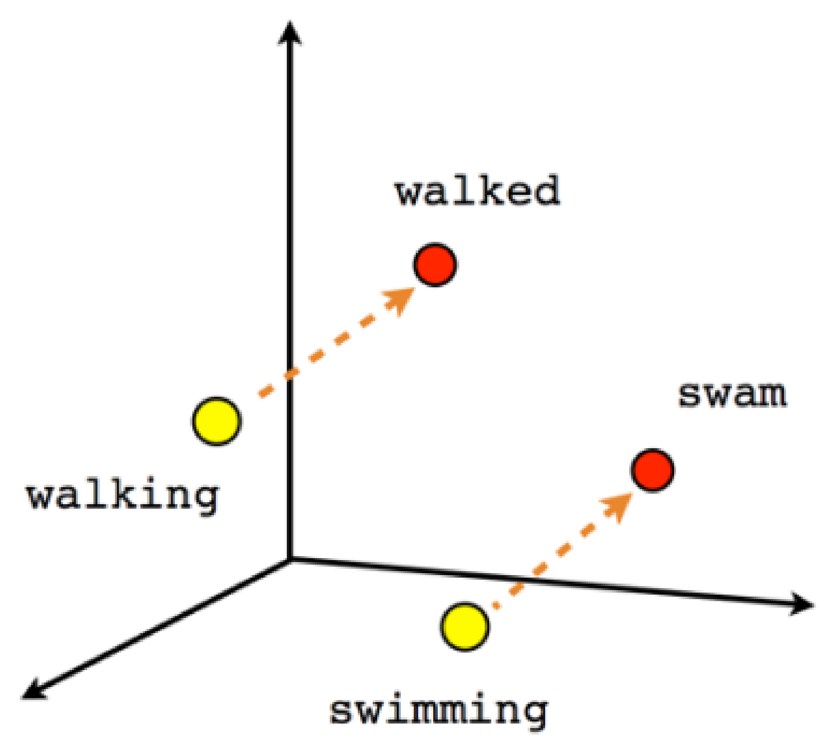
word2vec

- After learning, the hidden layer represents an **embedding vector**, which is a dense and compressed representation of each possible word (dimensionality reduction).
- Semantically close words (“endure” and “tolerate”) tend to appear in similar contexts, so their embedded representations will be close (Euclidian distance).
- One can even perform arithmetic operations on these vectors!

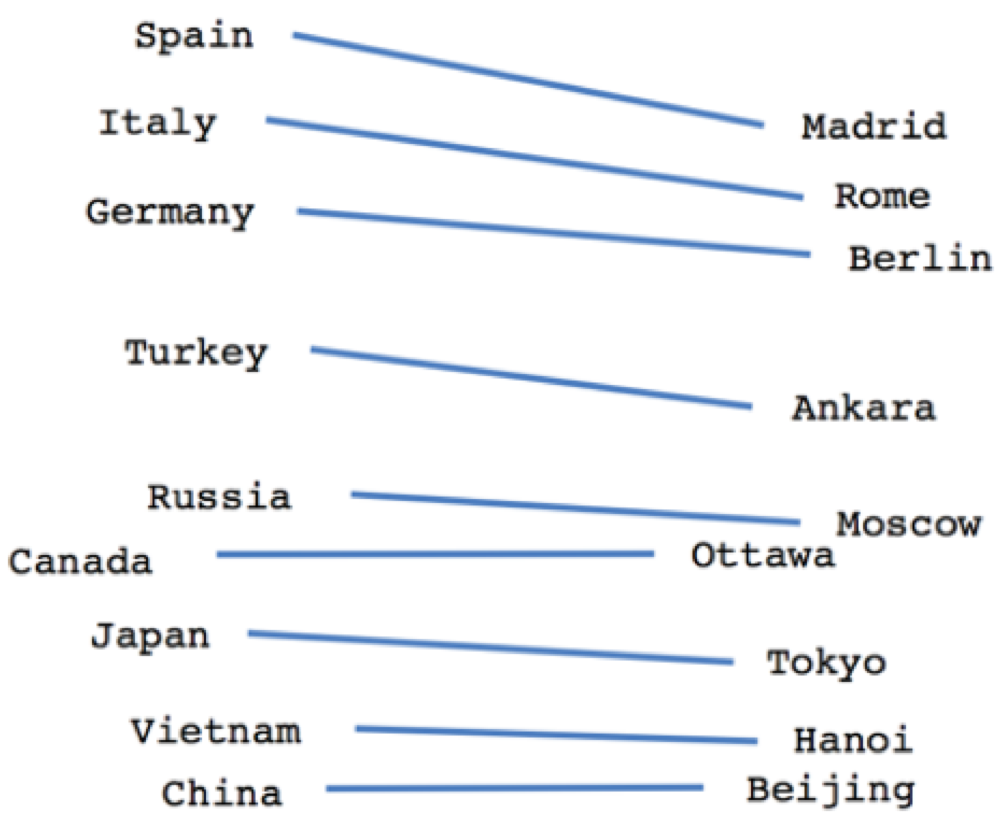
> queen = king + woman - man



Male-Female



Verb tense

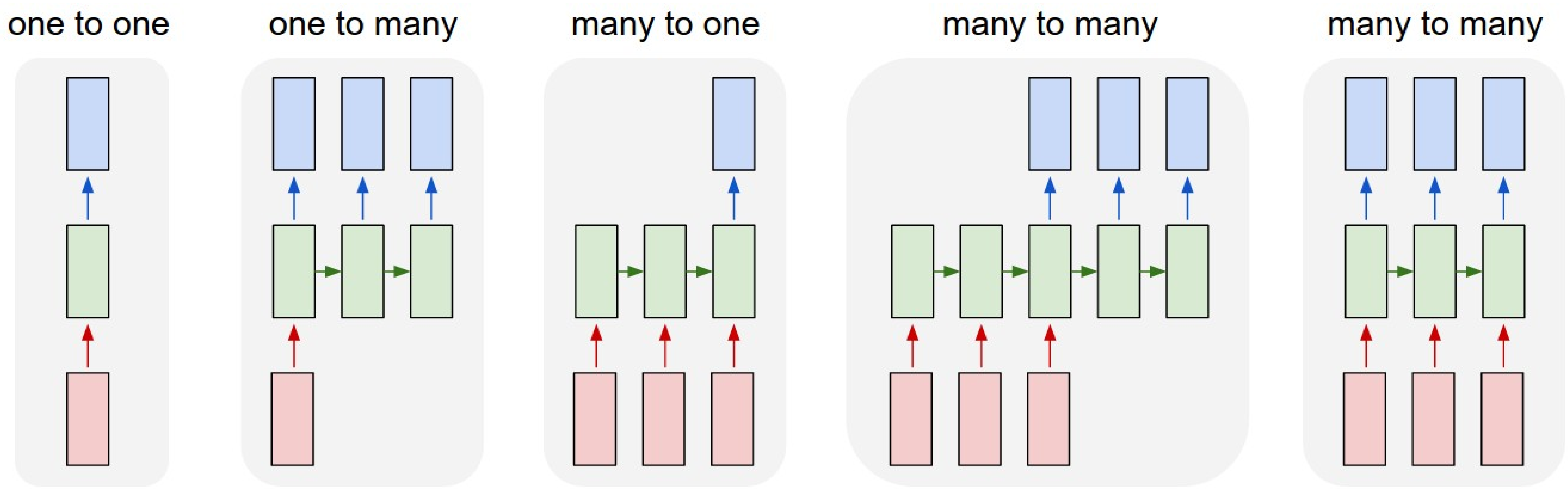


Country-Capital

Source : https://www.tensorflow.org/tutorials/representation/word2vec

2 - Applications of RNNs

Classification of LSTM architectures



Source: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

- **One to One**: classical feedforward network.

  Image $\longrightarrow$ Label.
- **One to Many**: single input, many outputs.

  Image $\longrightarrow$ Text.
- **Many to One**: sequence of inputs, single output.

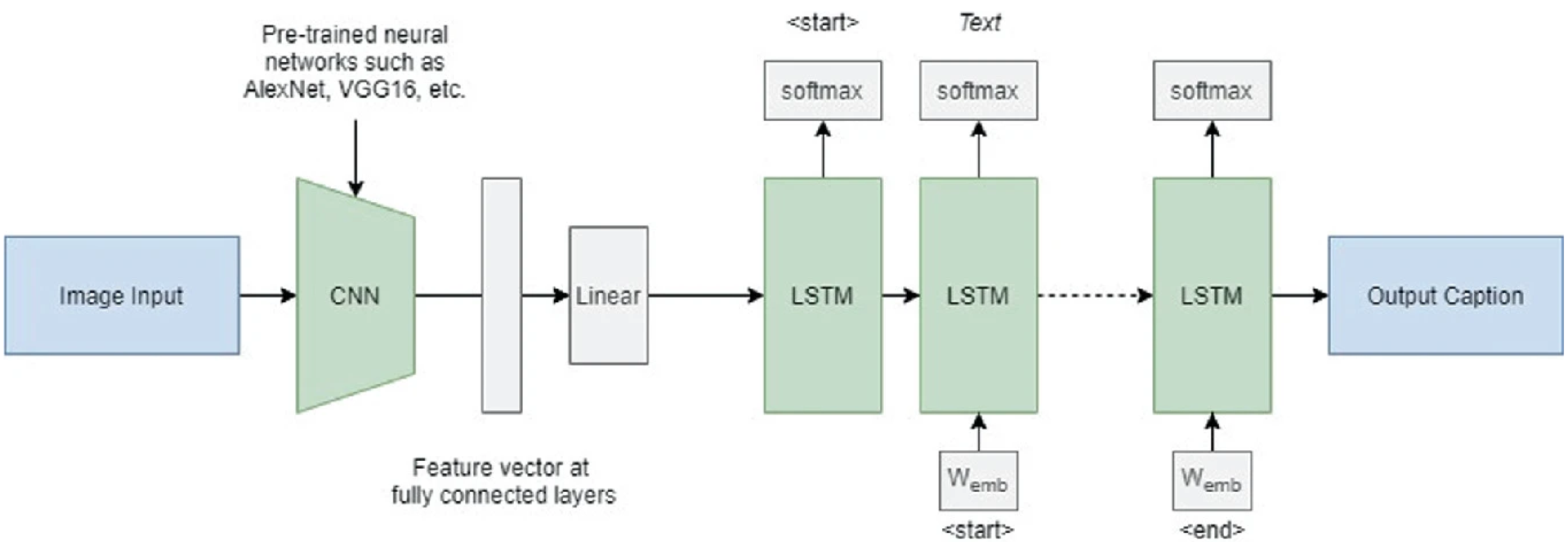
  Video / Text $\longrightarrow$ Label.
- **Many to Many**: sequence to sequence.

  Text $\longrightarrow$ Text.

  Video $\longrightarrow$ Text.

One to Many: image caption generation

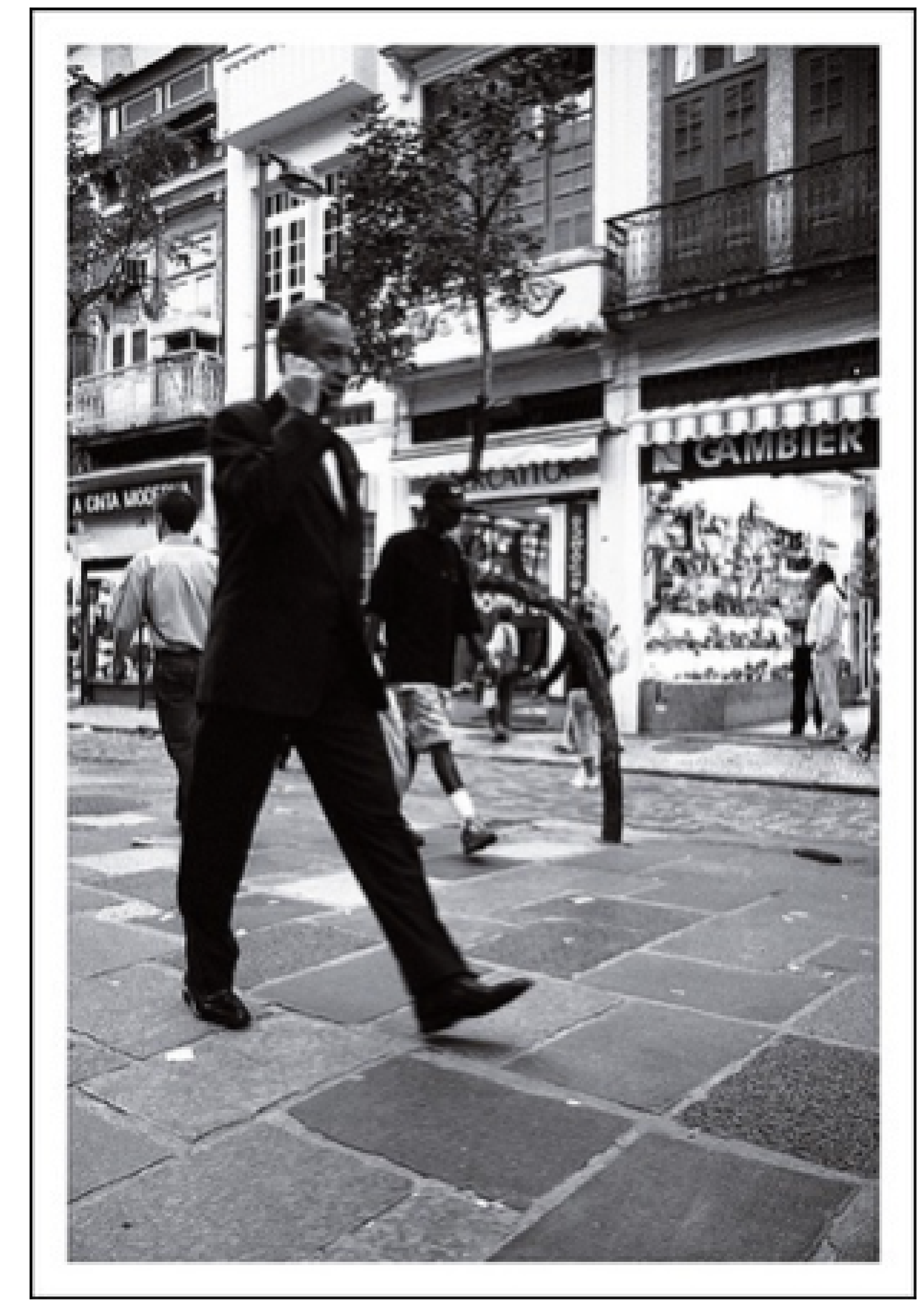
- **Show and Tell** uses the last FC layer of a CNN to feed a LSTM layer and generate words.
- The pretrained CNN (VGG16, ResNet50) is used as a **feature extractor**.



Source: Sathe et al. (2022). Overview of Image Caption Generators and Its Applications. ICCSA. https://doi.org/10.1007/978-981-19-0863-7_8

- Each word of the sentence is encoded/decoded using word2vec.
- The output of the LSTM at time t becomes its new input at time $t+1$.

One to Many: image caption generation



↑ a living room with a couch and a television

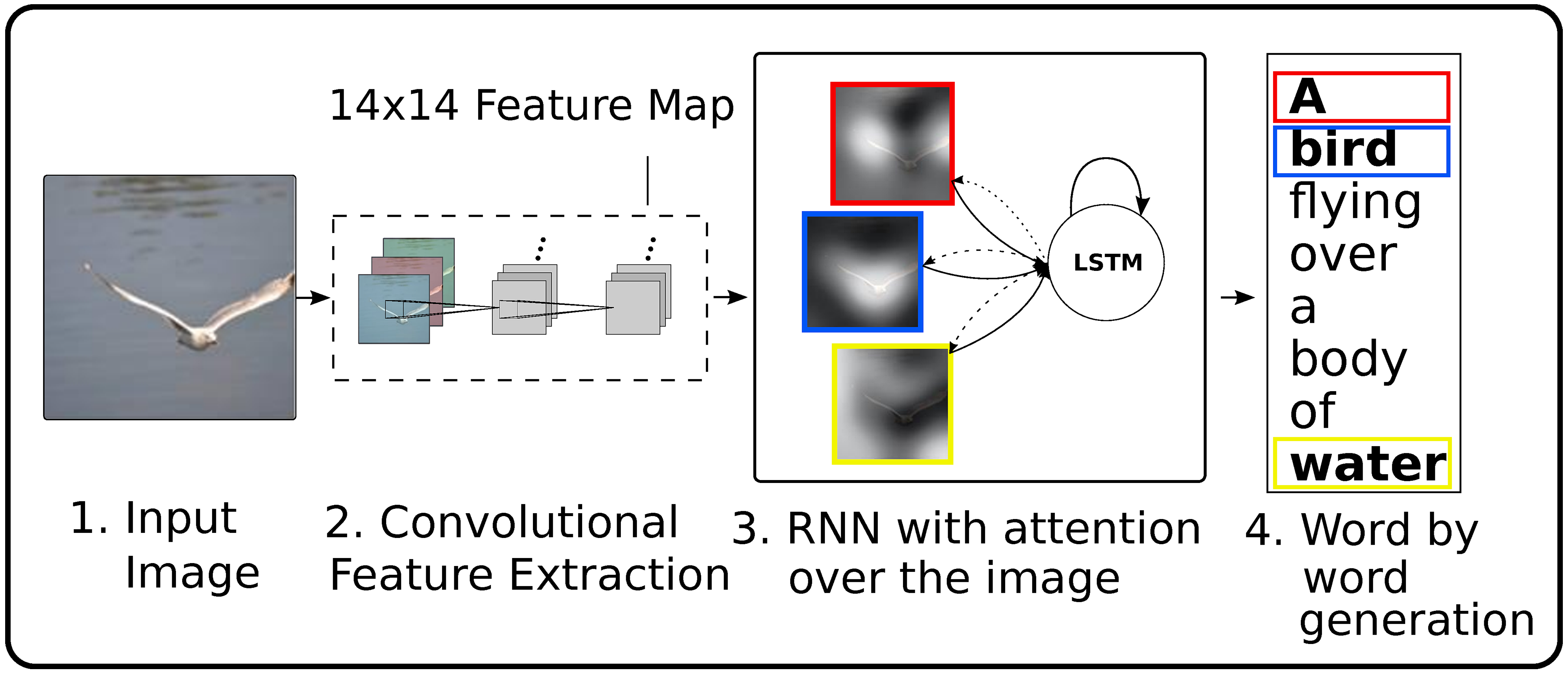
↑ a man riding a bike on a beach

a man is walking down the street with a suitcase ↗

Xu et al. (2015). Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. NIPS.

One to Many: image caption generation

- **Show, attend and tell** uses attention to focus on specific parts of the image when generating the sentence.



Source: http://kelvinxu.github.io/projects/capgen.html

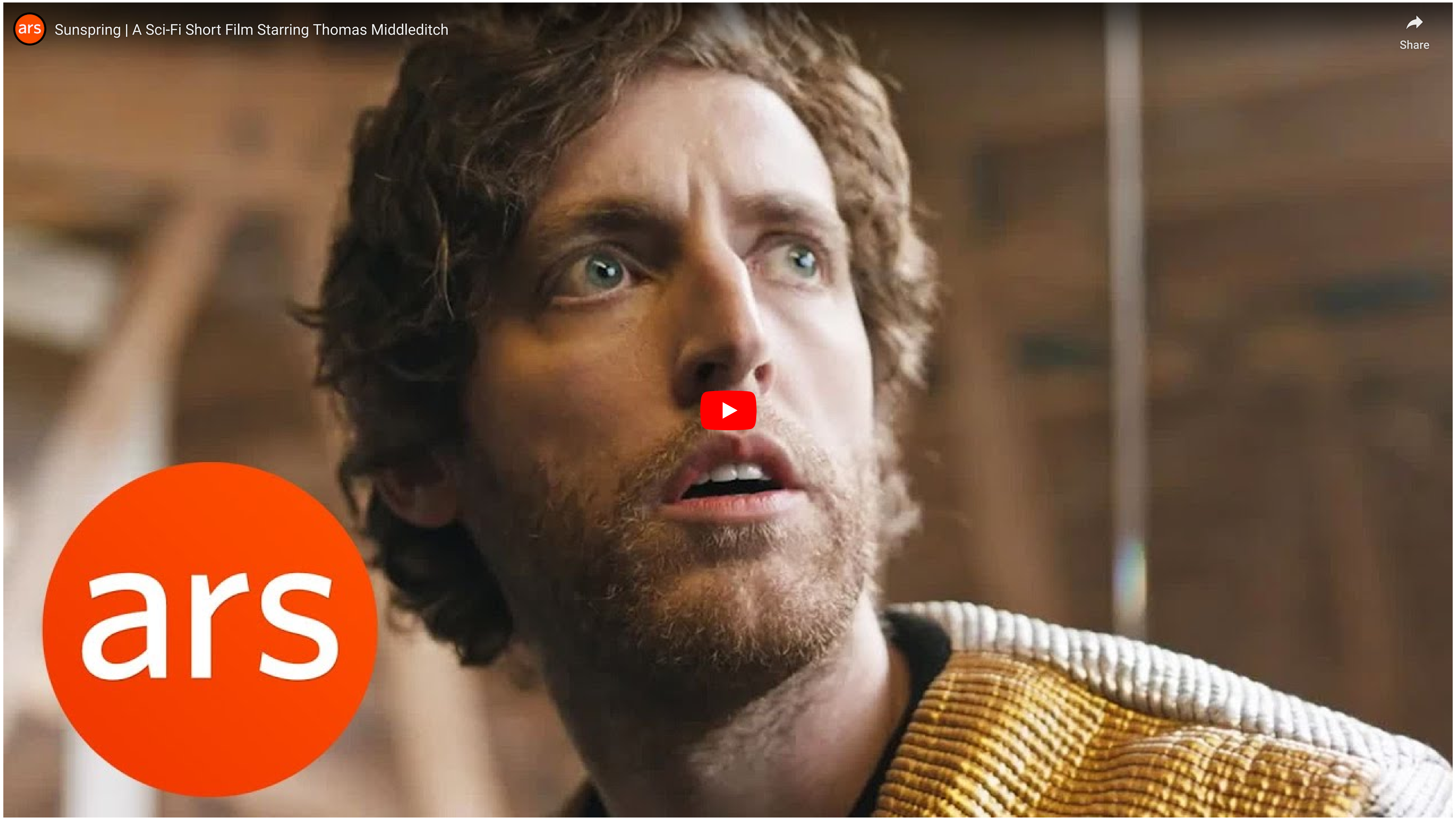
Many to One: next character/word prediction

```
PANDARUS:
Alas, I think he shall be come approached and the day
When little srain would be attain'd into being never
fed,
And who is but a chain and subjects of his death,
I should not sleep.

Second Senator:
They are away this miseries, produced upon my soul,
Breaking and strongly should be buried, when I perish
The earth and thoughts of many states.

DUKE VINCENTIO:
Well, your wit is in the care of side and that.

Second Lord:
They would be ruled after this chamber, and
my fair nues begun out of the fact, to be conveyed,
Whose noble souls I'll have the heart of the wars.

Clown:
Come, sir, I will make did behold your worship.
```

- Characters or words are fed one by one into a LSTM.
- The desired output is the next character or word in the text.
- Example:
    - Inputs: **To, be, or, not, to**
    - Output: **be**
- The text on the left was generated by a LSTM having read the entire writings of William Shakespeare.
- Each generated word is used as the next input.

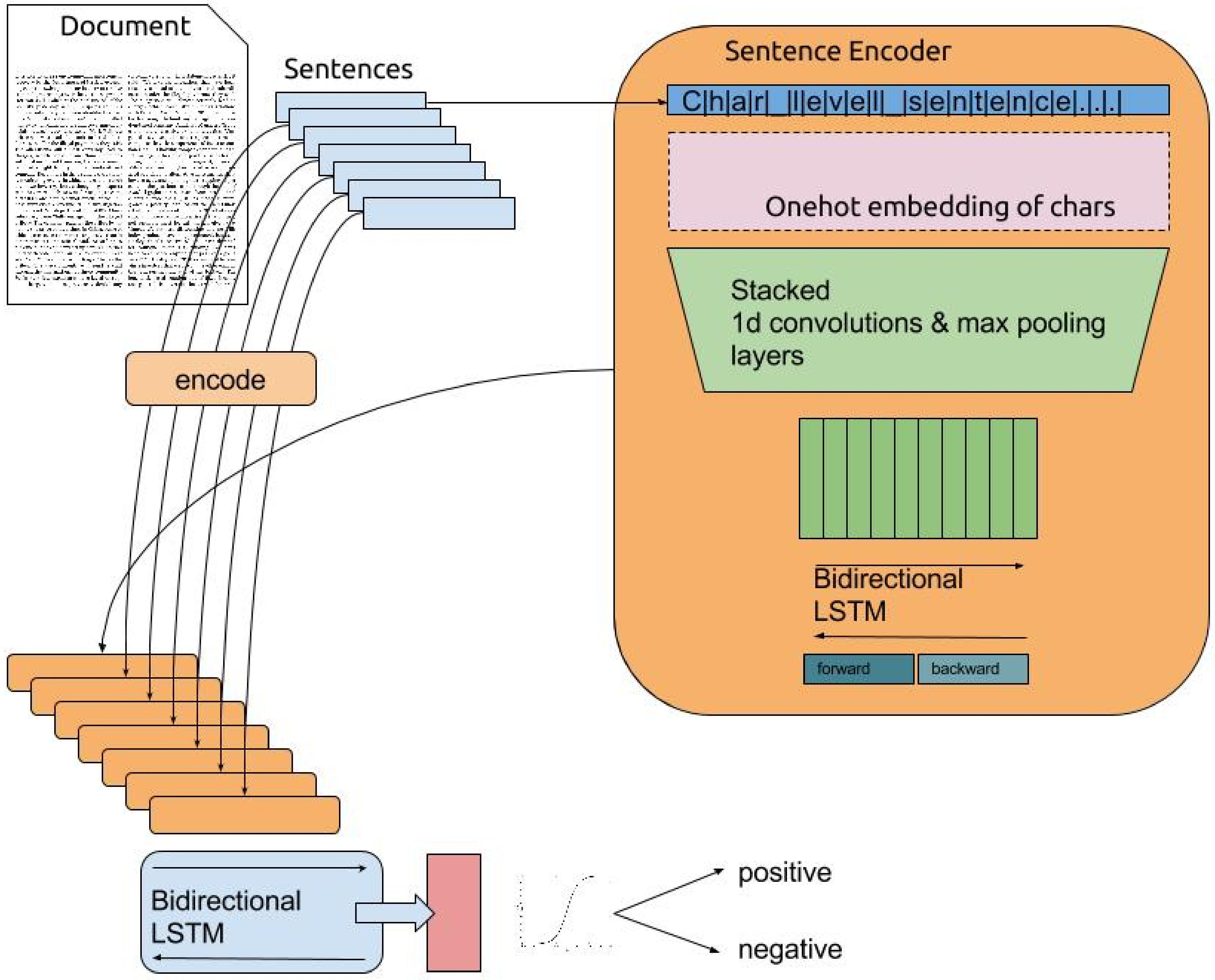
Source: http://karpathy.github.io/2015/05/21/rnn-effectiveness/

Many to one: Sunspring SciFi movie



More info: http://www.thereforefilms.com/sunspring.html

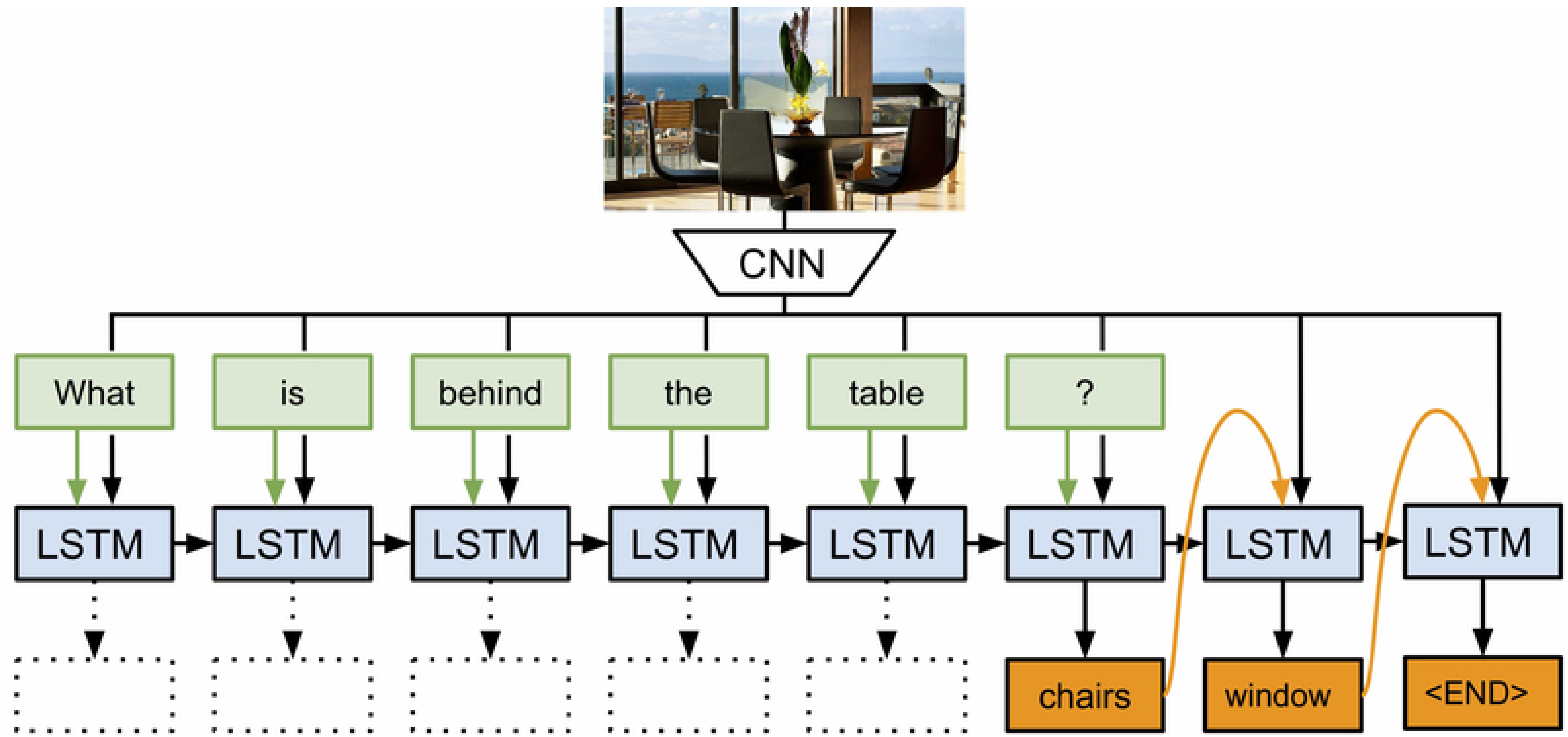
Many to One: sentiment analysis



- To obtain a vector from a sentence, **one-hot encoding** is used (alternative: word2vec).
- A 1D convolutional layers "slides" over the text.
- The bidirectional LSTM computes a state vector for the complete text.
- A classifier (fully connected layer) learns to predict the sentiment of the text (positive/negative).

Source: https://offbit.github.io/how-to-read/

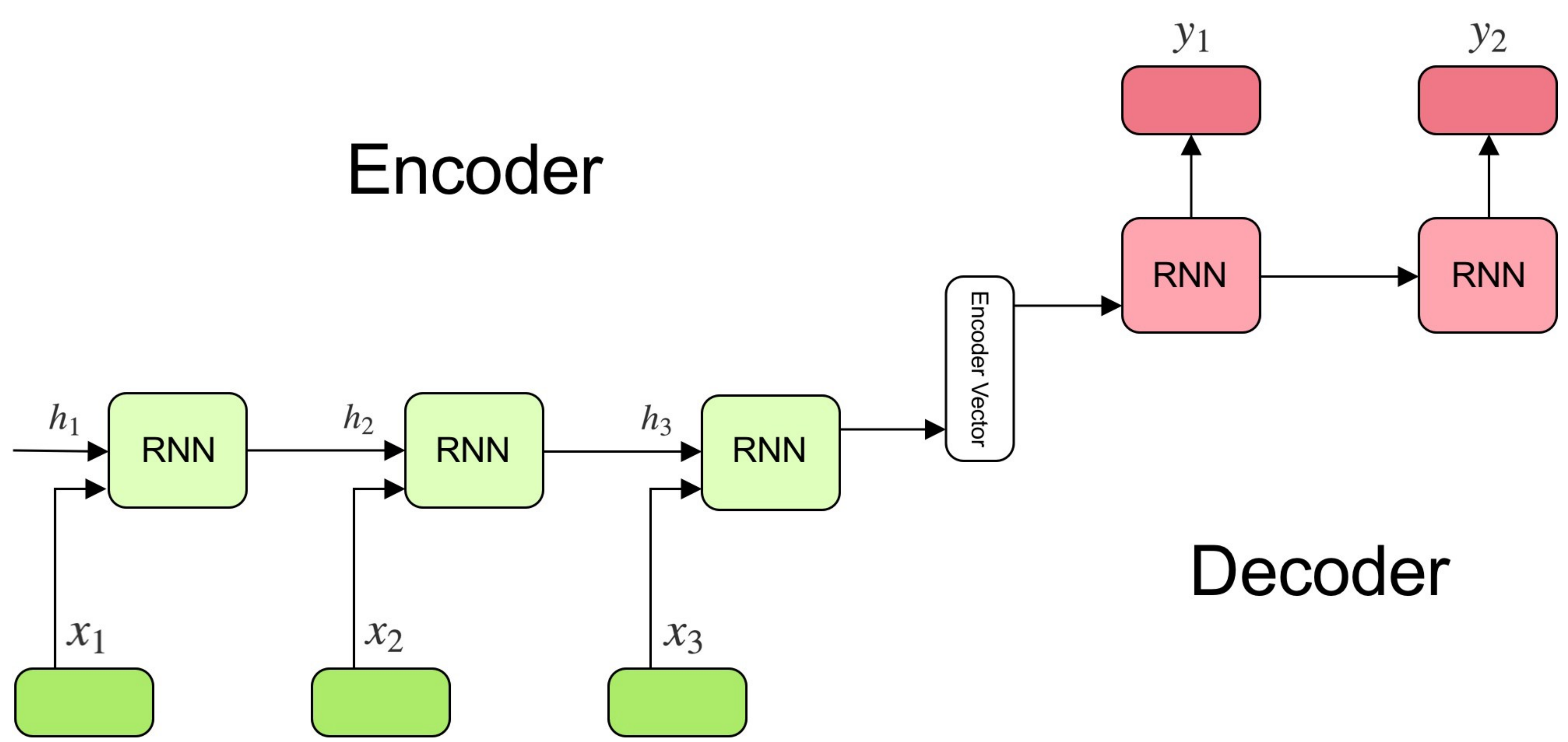
Many to Many: Question answering / Scene understanding



- A LSTM can learn to associate an image (static) plus a question (sequence) with the answer (sequence).
- The image is abstracted by a CNN trained for object recognition.

Many to Many: seq2seq

- The **state vector** obtained at the end of a sequence can be reused as an initial state for another LSTM.
- The goal of the **encoder** is to find a compressed representation of a sequence of inputs.
- The goal of the **decoder** is to generate a sequence from that representation.
- **Sequence-to-sequence** (seq2seq) models are recurrent autoencoders.



Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. arXiv:1409.3215

seq2seq for language translation

- The **encoder** learns for example to encode each word of a sentence in French.
- The **decoder** learns to associate the **final state vector** to the corresponding English sentence.
- seq2seq allows automatic text translation between many languages given enough data.
- Modern translation tools are based on seq2seq, but with attention.

https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/