30.10.23

OPENCAMPUS.sh

Deep Dive into LLMs

TOKEN & EMBEDDINGS REVISITED

QUIZ

https://forms.office.com/e/CgWRRGSrTL

PROJECT MILESTONES

- 30.10 Form Groups
- 06.11 Literature Review I
- 13.11 Literature Review II
- 20.11 TBD
- 27.11 TBD
- 04.12 TBD
- 11.12 TBD
- 18.12 TBD
- 08.01 Project presentations

LEARNING GOALS

1. **Understand the Evolution of Language Models**: Gain a broad understanding of how language models have evolved, from early methods like n-grams to advanced transformer architectures.
2. **Grasp Core Concepts of Word Embeddings and RNNs**: Understand the significance and limitations of word embeddings and recurrent neural networks, including LSTMs.
3. **Master the Basics of Attention and Transformers**: Comprehend the fundamentals of attention mechanisms and the architecture of Transformer models, including key components like Multi-Head Attention and Positional Encoding.
4. **Familiarize with Advanced Models**: Be aware of advancements beyond basic Transformers, including models like BERT and GPT.
5. **Synthesize and Reflect**: Summarize key takeaways from the session and formulate thoughts or questions for further exploration.

EARLY LANGUAGE MODELS

LANGUAGE MODEL

Language Modeling Task:

- Input: sequence of words $x^1, x^2, \dots, x^n$
- Output: prob dist of the next word $P\left(x^{(n+1)} \middle| x^t, \dots, x^1\right)$

TYPICAL TASKS

- Text generation
- Information Retrieval
- Text Classification
- Grammer checking
- Word Prediction
- Named Entity Recognition (NER)

STATISTICAL METHODS

N-GRAMS

- Sentence: “I love cake”
- Unigram: “I”, “love”, “cake”
- Bigram: “I love”, “love cake”
- Trigram: “I love cake”

N-grams can break large texts into manageable pieces

STATISTICAL METHODS

$$P(cake|\mathrm{I}, \mathrm{love}, \mathrm{to}, \mathrm{eat})$$

- If “I love to eat” occurs 100 times in the corpus and “I love to eat cake” occurs 90 times, then:

$$P(cake|\mathrm{I}, \mathrm{love}, \mathrm{to}, \mathrm{eat}) = \frac{90}{100} = 0.9$$

These methods are simple but have many limitations

NEURAL METHODS

- Solution: Neural Networks
- Problem: Neural Networks work with numbers not text

Word Embeddings to the rescue

WORD EMBEDDINGS

STATIC VECTORS

- Feature-based (sparse)
- Count-based (sparse)
- Classical dimensionality reduction (dense)
- Learned dimensionality reduction (dense)

PROBLEMS

- In web search if user searches for *“New York motel”,* we would like to match results containing *“New York hotel”*

Motel = [0 0 0 0 0 0 0 0 1 0 0]
Hotel = [0 1 0 0 0 0 0 0 0 0 0]

IMPORTANCE OF WORD EMBEDDINGS

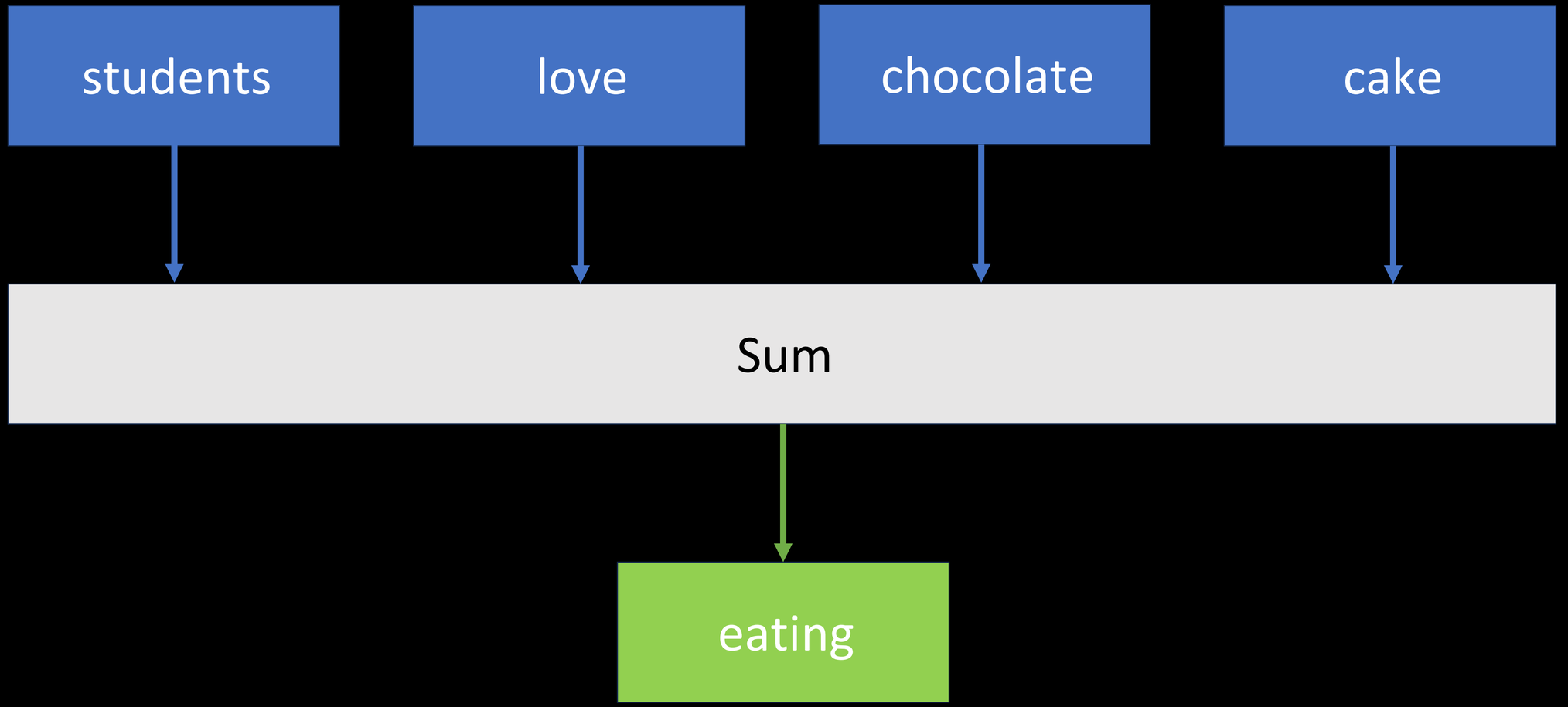
- Semantic meaning
- Contextual relationships
- Dimensionality reduction
- Improved model performance
- Handling OOV tokens

WORD2VEC IDEA

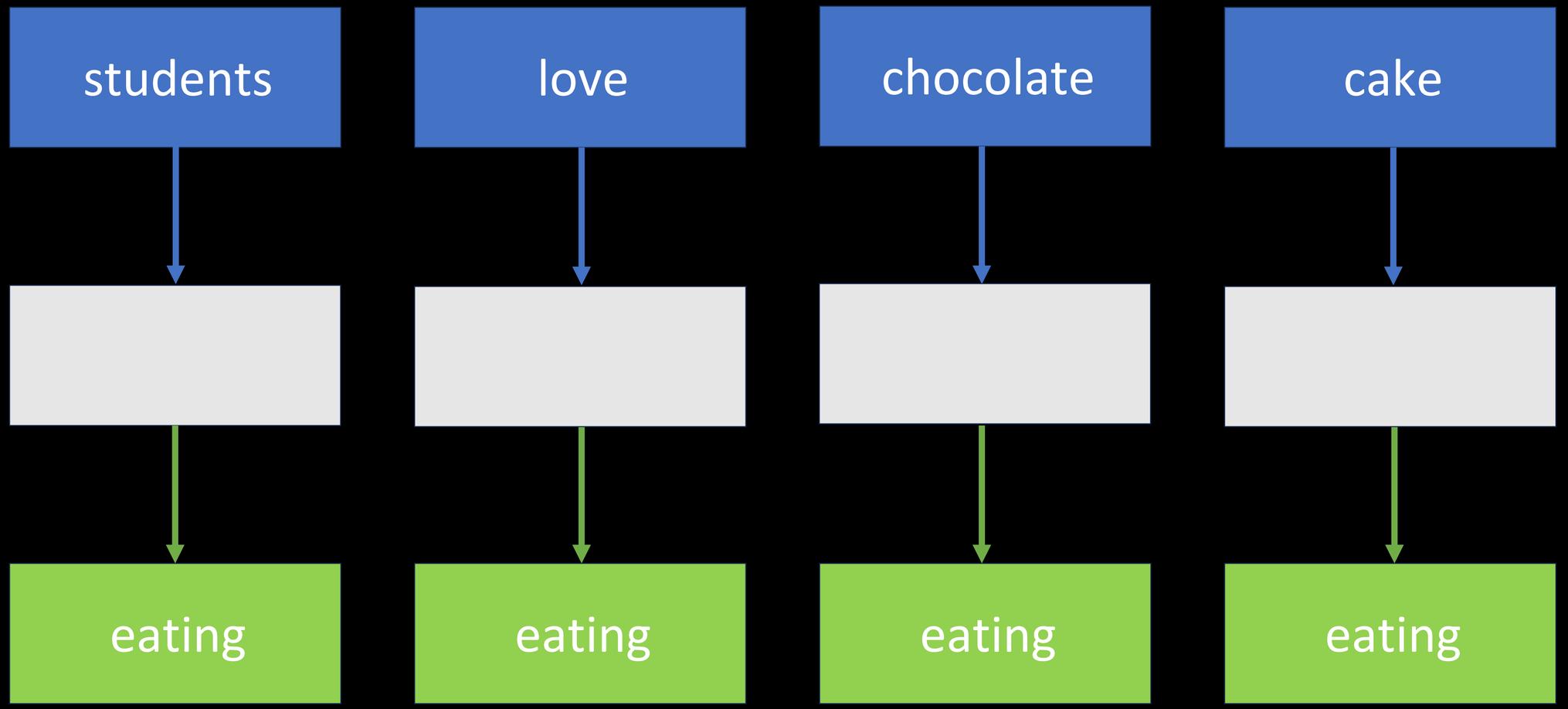
- group of related models that produce **word embeddings** by utilizing shallow neural networks
- Context-based learning
- train a neural network with a single hidden layer to predict a target word based on its context (neighboring words)
- In the end we use the word embeddings, not the network itself

WORD2VEC IDEA

CBOW



SKIP-GRAM



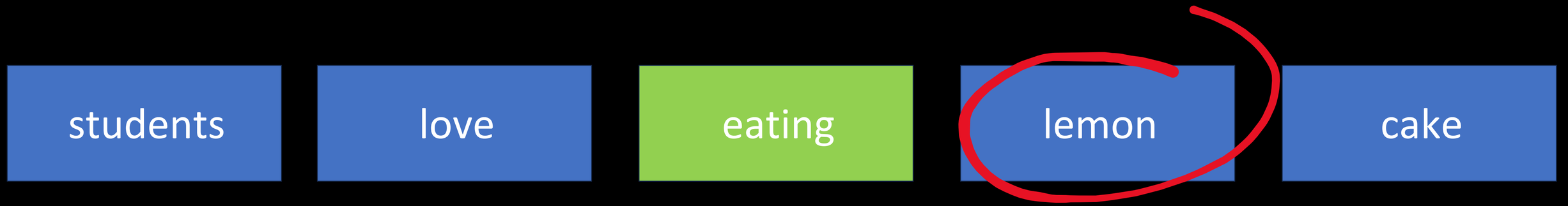
WORD2VEC VISUALIZATION

https://projector.tensorflow.org

GLOVE

- GloVe combines global statistical information (from the entire corpus) with local information (from specific word pairs) to create the word embeddings

PROBLEM



What to do with Out Of Vocabulary words?

FAST TEXT

- builds on the ideas of Word2Vec but introduces two major tweaks to improve the quality of the embeddings:

1. **Subword Modeling**
2. **Efficient Learning**

NEURAL METHODS

SIMPLE NEURAL NETWORKS

Output distribution

$$\hat{y} = \text{softmax}(\text{Uh} + \text{b}_2) \in \mathbb{R}^{|V|}$$

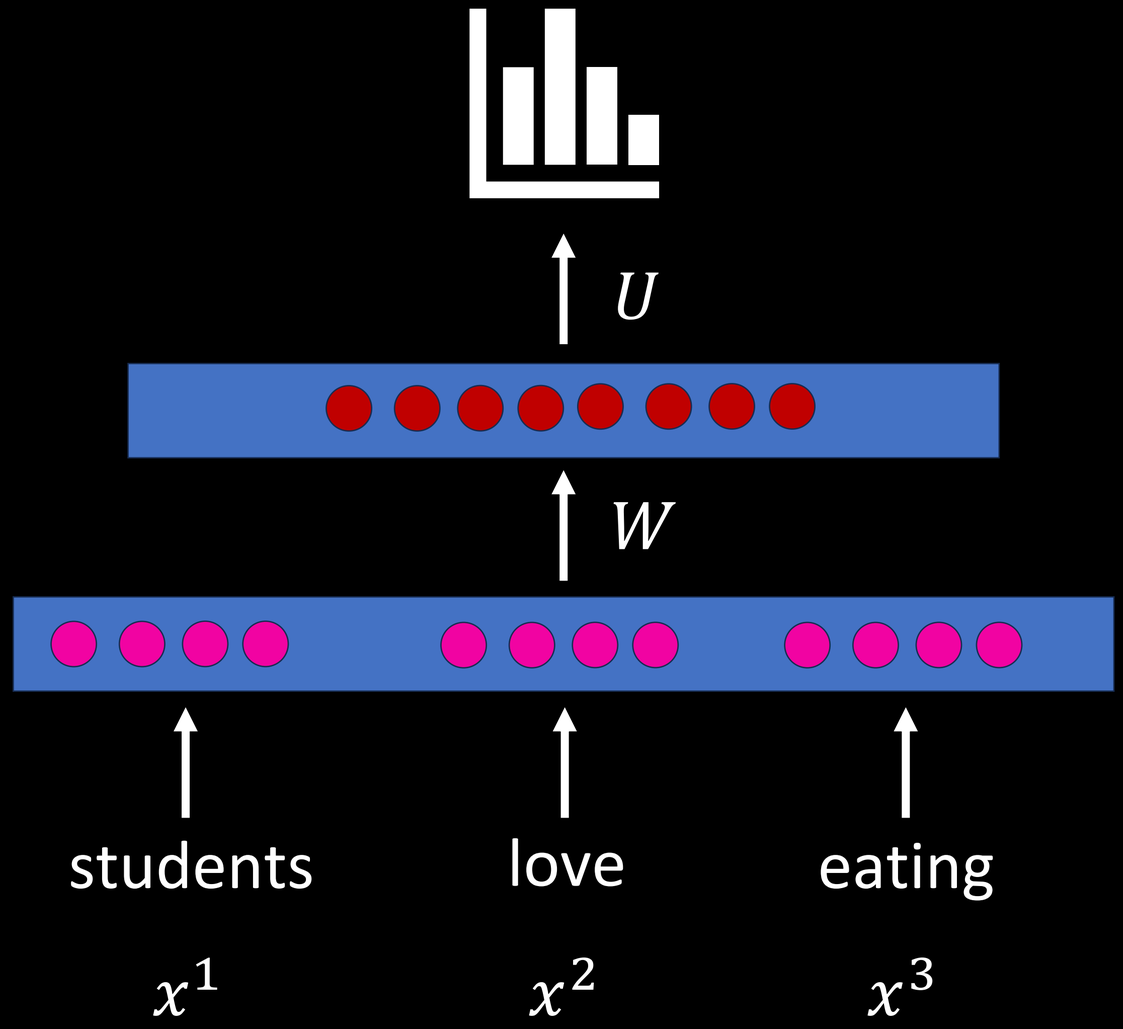
Hidden layer

$$h = f(We + b_1)$$

Concatenated word embeddings

$$e = [e^1; e^2; e^3]$$

Words / one-hot vectors



LEARNINGS

Improvements over *n-gram* LM:

- No sparsity problem
- Don´t need to store all observed *n-grams*

Remaining problems:

- Fixed window is too small
- Enlarging window enlarges W

→ We need an architecture that can process any length input

SIMPLE RNN LANGUAGE MODEL

Output distribution

$$\widehat{y^t} = \text{softmax}(\text{Uh}^t + \text{b}_2) \in \mathbb{R}^{|V|}$$

Hidden states

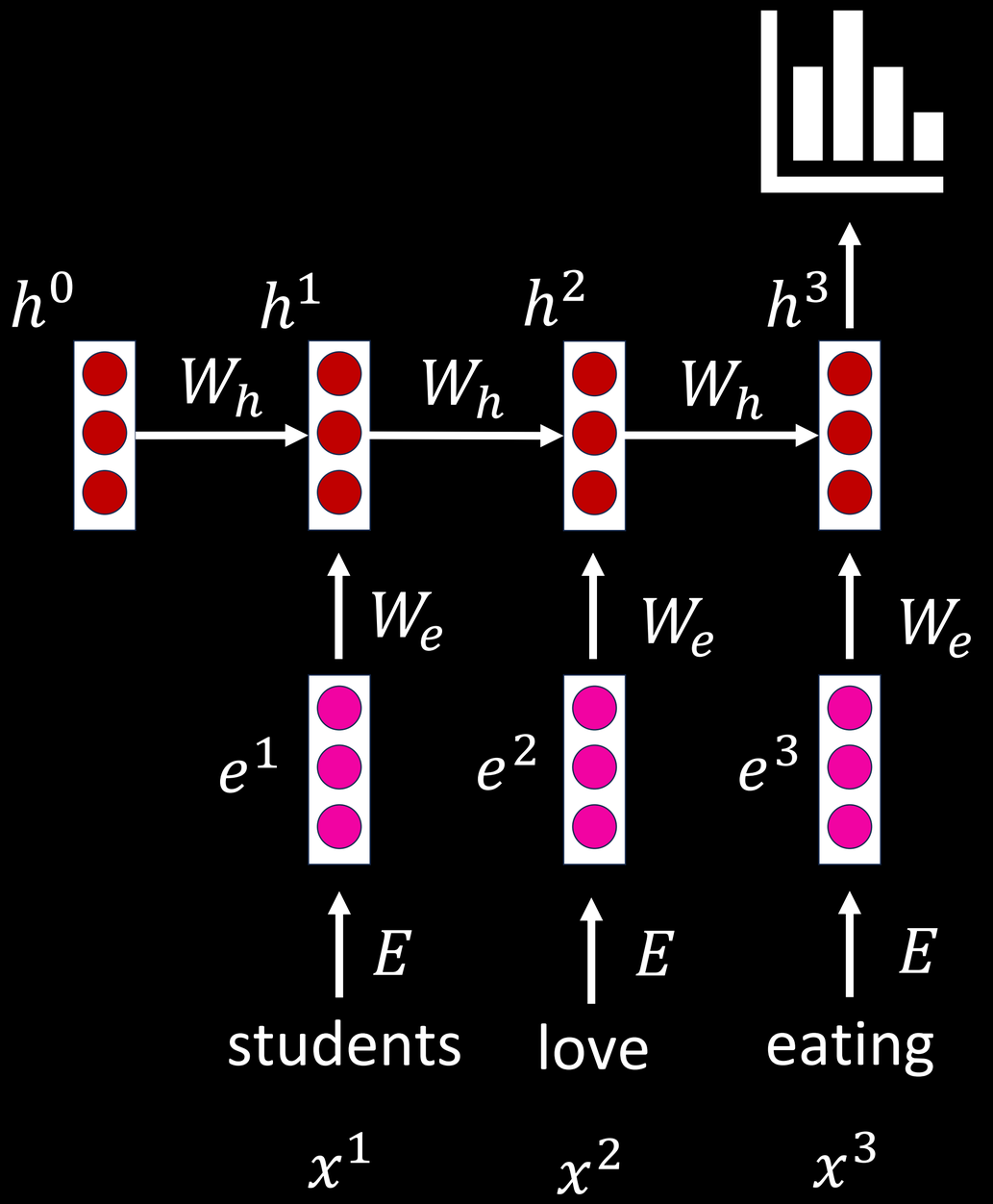
$$h^t = \sigma\left(W_h h^{t-1} + W_e e^t + b_1\right)$$

h^0 is the initial hidden state

Word embeddings

$$e = Ex^t$$

Words / one-hot vectors



PROBLEMS OF RNNS

Exploding Gradients

→ Model becomes unstable

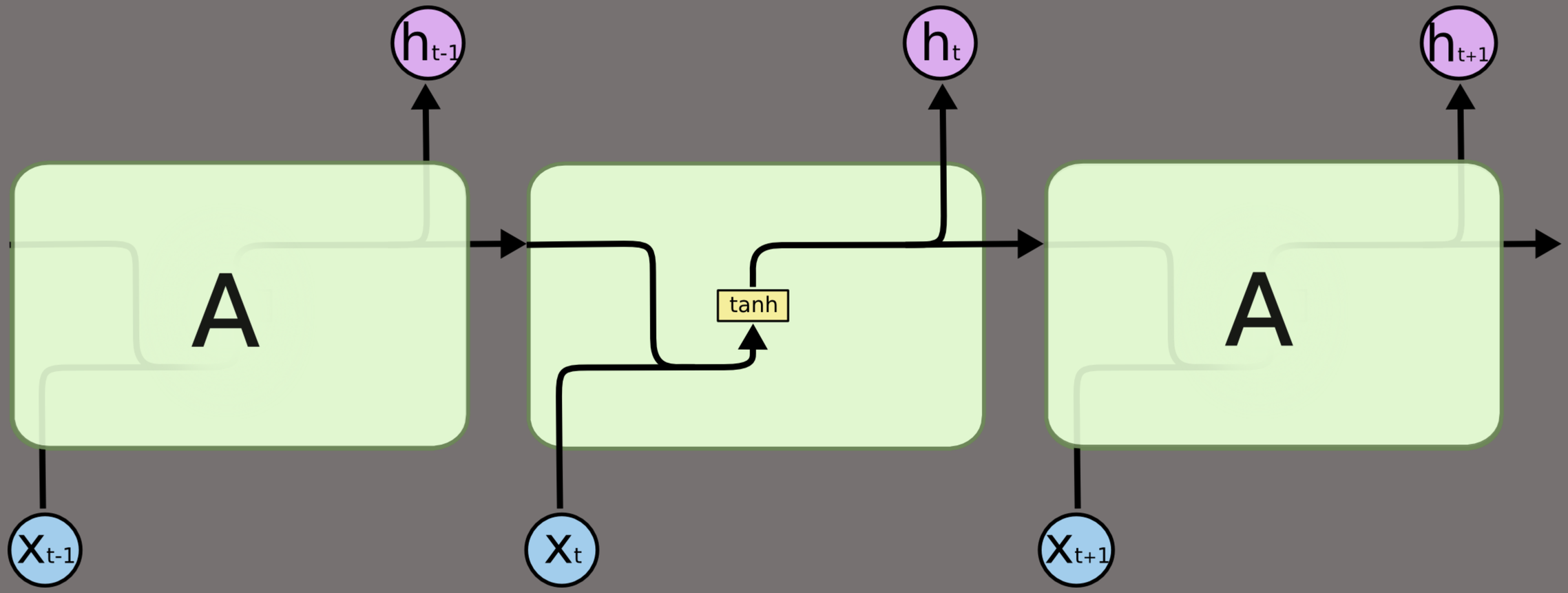
Vanishing Gradients

→ The model doesn´t learn anything

PROBLEMATIC EXAMPLE

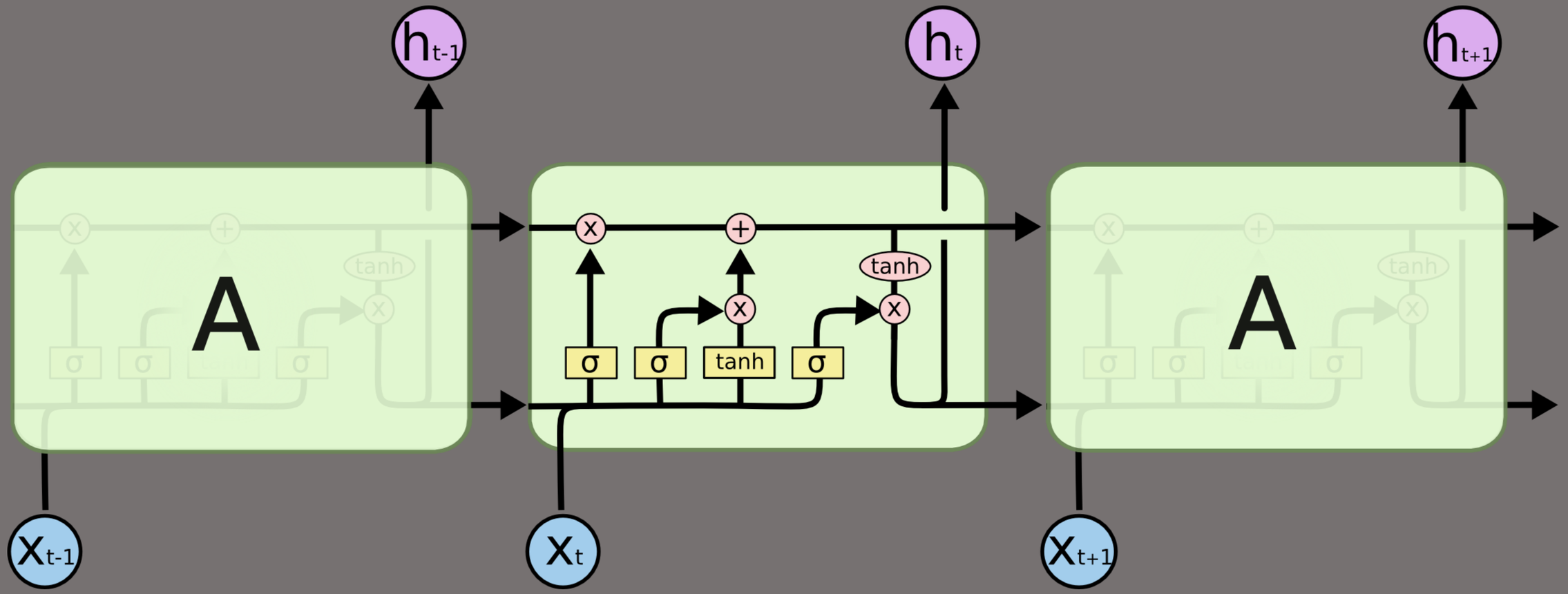
- "David had a big presentation scheduled for next week. He spent days preparing the slides, rehearsing his speech, and researching additional material to address potential questions. The night before the big day, he made sure to get a full night's sleep. The next morning, he dressed in his best suit and drove to the office. After one final run-through, he stood before his colleagues and began his _____ “

RNN



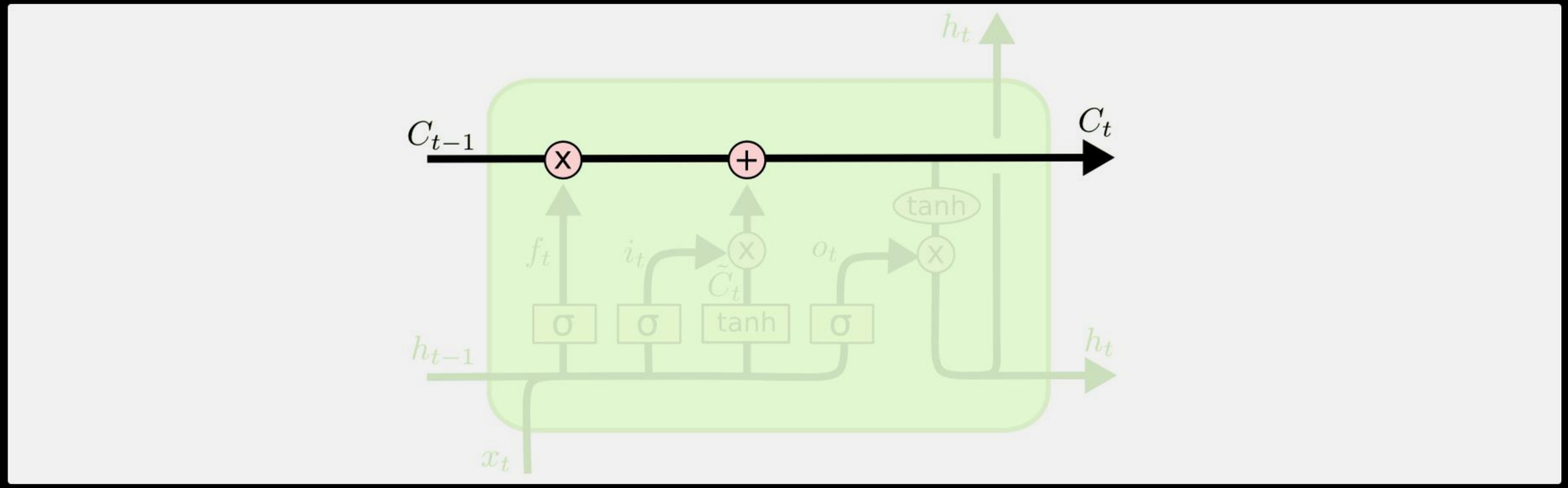
https://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM



https://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM



https://colah.github.io/posts/2015-08-Understanding-LSTMs/

BIDIRECTIONAL RNN

- So far only processing input from left to right

→ Process input from left to right

→ Process input from right to left

→ Concatenate the outputs

Problem:

- We need access to the entire input sequence

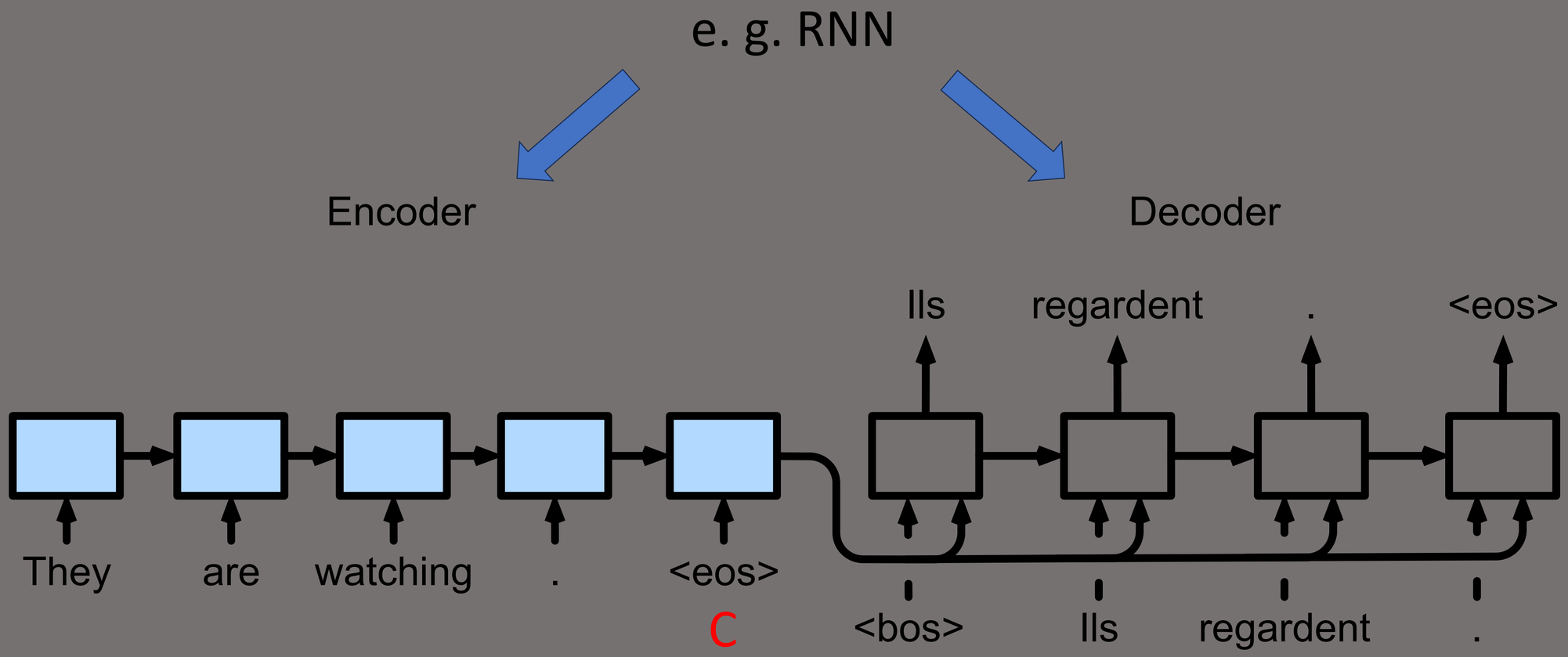
→ Bidirectional RNNs are not applicable to LM

SEQ2SEQ

- While RNNs can process input of any length they have a fixed length output

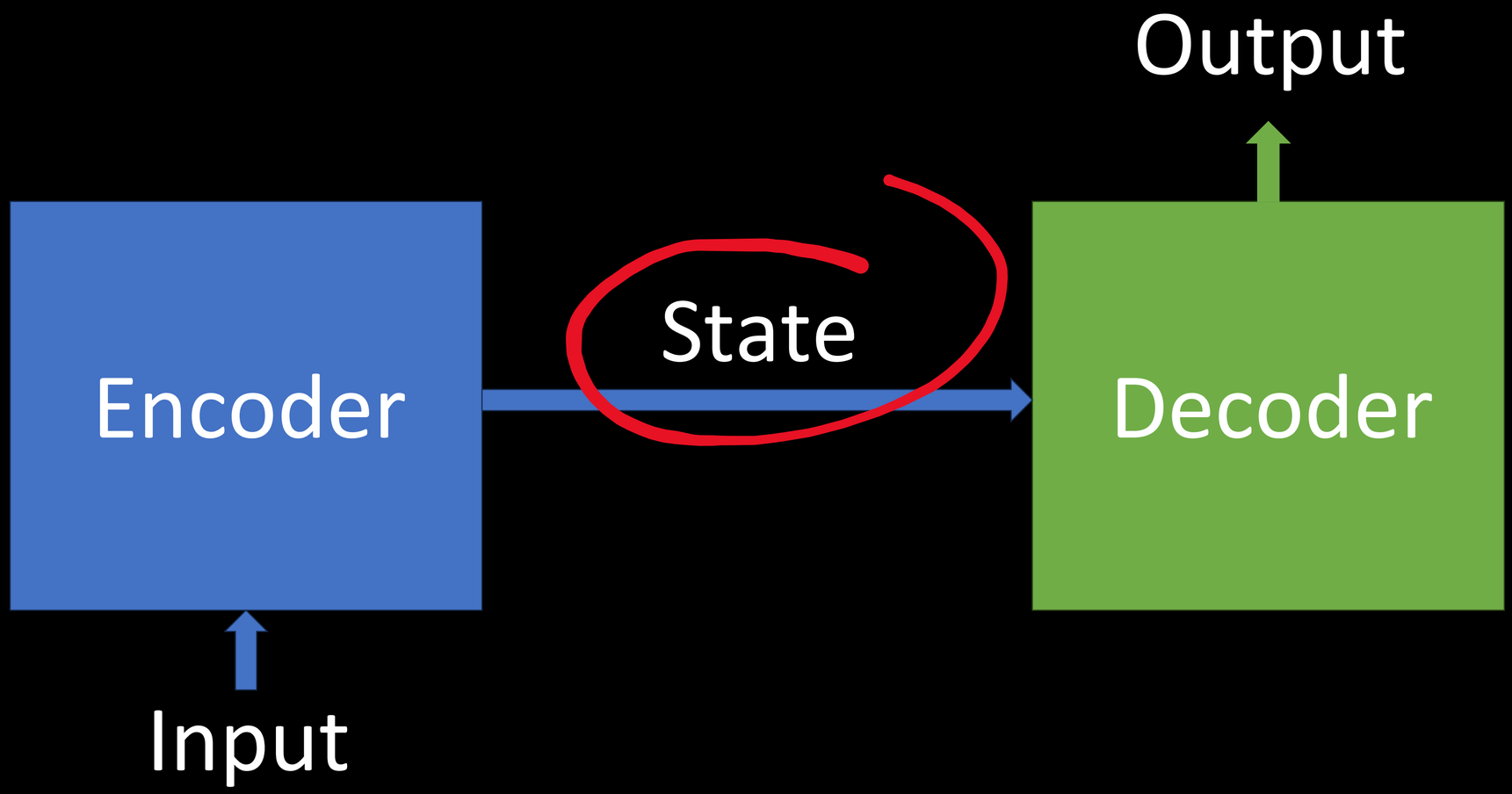
What if we want to do machine translation where the length of the output changes?

SEQ2SEQ



https://d2l.ai/chapter_recurrent-modern/seq2seq.html

PROBLEM



ATTENTION IDEA

- At each step of the decoding process, the focus of the network is only on a particular part of the input sequence
- Use a direct connection to the encoder
- Which part to focus on?

→ Use dot product to measure similarity with the input

→ Compare the hidden state of the decoder with every hidden state of the encoder

TASKS UNTIL NEXT WEEK

- Watch the last ten minutes of this Stanford lecture on Attention
- Watch the first 25 Minutes of this Stanford lecture on Attention
- Watch all 4 Videos of the Rasa Attention Series
- Complete the Notebook and play around with different embeddings
- Optional: Watch this Stanford lecture on Transformers