# CSCE 689: Special Topics in Advanced NLP

## Lecture 3: Language Modeling, Convolutional and Recurrent Neural Network

Kuan-Hao Huang

Fall 2026

(Some slides adapted from Chris Manning, Dan Jurafsky, Danqi Chen, Karthik Narasimhan, Graham Neubig, and Greg Durrett)

# LaTeX Assignment

- LaTeX Assignment (1%) [Due: 9/10]
- No late submission

**CSCE 689: LaTeX Assignment**

**Your Name**
Your UID and email

### Overview

This assignment is designed to give you practice with LaTeX, which you are expected to use for your literature review, project proposal, and final report in this course.

### Instructions

For this assignment, you will create a PDF containing your answers using LaTeX. If this is your first time working with LaTeX, we recommend starting with this short tutorial, which covers the basic features you will need for this course. Please use the Association for Computational Linguistics LaTeX template (link), a template widely used in major NLP conferences. We suggest using Overleaf as your online editor, since it automatically manages packages for you.

By default, the template is set to *review mode*. To switch to *final mode*, change:

Review Mode (Default)

```
\usepackage[review]{acl}
```

to:

Final Mode

```
\usepackage[final]{acl}
```

This allows you to display the author information. Be sure to include your name, UIN, and email.

The following sections contain questions on some commonly used LaTeX commands. There are a total of 100 points for this assignment. Please answer each question in a separate *section*, and submit the final PDF generated using LaTeX.

### 1 Including Equations [20pts]

Typeset the following expression using LaTeX:

$$\frac{\partial \mathcal{L}_{\text{total}}}{\partial \mathbf{w}_j} = -\frac{1}{m}\sum_{i=1}^{m}\left(y_i - \sigma(z_i)\right)\cdot \mathbf{x}_{i,j}$$

### 2 Including Images [20pts]

Select a picture of a cat and include it with a caption. The figure below is provided as an example.

Figure 1: This is a cute cat!

### 3 Including Tables [20pts]

Create a table that displays your name, UIN, and email. You can follow the example below as a template.

| Name | Kuan-Hao Huang |
|---|---|
| UIN | 123456789 |
| Email | khhuang@tamu.edu |

Table 1: Example table.

### 4 Including Lists [20pts]

Create a list that displays your name, UIN, and email. You can follow the example below as a template.

- Name: Kuan-Hao Huang
- UIN: 123456789
- Email: khhuang@tamu.edu

### 5 Including Citations [20pts]

Use *BibTex* to include the following paper: Paper 1 (Vaswani et al., 2017) and Paper 2 (Devlin et al., 2019). You can learn more about *BibTex* here.

# Topic Study

- Topic Study (30%) (a team of 2 people)
  - Literature Review (15%) [Due: 10/1]
  - Topic Presentation (15%)
    - Starting from Week 5

# Topic Sign-Up

- 30 students
- 15 topics
- Each topic has 2 people

| Topic ID | Date | Topic |
|---|---|---|
| 1 | 9/21 | Alignment, Post-Training |
| 2 | 9/23 | Test-Time Scaling, Reasoning Models |
| 3 | 9/28 | Model Efficiency |
| 4 | 9/30 | Bias Detection and Mitigation |
| 5 | 10/14 | Adversarial Attacks and Jailbreaking |
| 6 | 10/19 | AI-Generated Text Detection |
| 7 | 10/21 | Hallucinations |
| 8 | 10/26 | Model Interpretability, Model Steering |
| 9 | 10/28 | Model Editing, Model Unlearning |
| 10 | 11/2 | Vision-Language Alignment |
| 11 | 11/4 | Multimodal Reasoning |
| 12 | 11/9 | Agents, Model Memory |
| 13 | 11/11 | Long-Context Understanding |
| 14 | 11/16 | Multilingual Models |
| 15 | 11/18 | Non-Autoregressive Generation |

# Topic Sign-Up

- https://docs.google.com/spreadsheets/d/1dX1P9DLlgav3Obu64b4VMN99Yg1C1S5THhcVu3FApoI/edit?usp=sharing
- Log in with TAMU account
- Due: Sep 2 before lecture

| Put Preference with Topic IDs | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Team** | **Member 1** | **Member 2 (Optional)** | **P1** | **P2** | **P3** | **P4** | **P5** | **P6** | **P7** | **P8** | **P9** | **P10** | **P11** | **P12** | **P13** | **P14** | **P15** |
| Example | First_name Last_name | First_name Last_name | 4 | 15 | 5 | 13 | 1 | 14 | 3 | 11 | 9 | 12 | 2 | 8 | 7 | 10 | 6 |
| 1 | | | | | | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | | |
| 8 | | | | | | | | | | | | | | | | | |
| 9 | | | | | | | | | | | | | | | | | |
| 10 | | | | | | | | | | | | | | | | | |
| 11 | | | | | | | | | | | | | | | | | |
| 12 | | | | | | | | | | | | | | | | | |

# Topic Sign-Up

- https://docs.google.com/spreadsheets/d/1dX1P9DLlgav3Obu64b4VMN99Yg1C1S5THhcVu3FApoI/edit?usp=sharing
- Log in with TAMU account
- Due: Sep 2 before lecture

| If you are looking for teamates | | |
|---|---|---|
| | **Name** | **Email** |
| Example | Kuan-Hao Huang | khhuang@tamu.edu |

# Topic Sign-Up: Algorithm

- We will finalize the topic assignment in class on Sep 2
- Decision Process:
  - Preferences will be handled in order (Preference 1 → Preference 2 → …)
  - If none of the team members attends the class, they lose the chance
  - If one team's preference still has a slot, they get it
  - If multiple teams choose the same topic, we will draw a lottery
  - Move on to the next preference order
- So be strategic when choosing your preferences

# Topic Sign-Up: Example 1

| Member 1 | Member 2 | P1 | P2 | P3 | P4 |
|---|---|---|---|---|---|
| Student 1 | -- | 1 | 2 | 3 | 4 |
| Student 2 | -- | 1 | 2 | 3 | 4 |
| Student 3 | -- | 4 | 2 | 3 | 1 |
| Student 4 | -- | 1 | 2 | 3 | 4 |
| Student 5 | -- | 2 | 4 | 1 | 3 |
| Student 6 | -- | 2 | 1 | 3 | 4 |
| Student 7 | -- | 1 | 3 | 4 | 2 |
| Student 8 | -- | 1 | 3 | 4 | 2 |

# Topic Sign-Up: Example 1

| Member 1 | Member 2 | P1 | P2 | P3 | P4 |
|---|---|---|---|---|---|
| Student 1 | -- | 1 | 2 | 3 | 4 |
| Student 2 | -- | 1 | 2 | 3 | 4 |
| Student 3 | -- | 4 | 2 | 3 | 1 |
| Student 4 | -- | 1 | 2 | 3 | 4 |
| Student 5 | -- | 2 | 4 | 1 | 3 |
| Student 6 | -- | 2 | 1 | 3 | 4 |
| Student 7 | -- | 1 | 3 | 4 | 2 |
| Student 8 | -- | 1 | 3 | 4 | 2 |

# Topic Sign-Up: Example 1

| Member 1 | Member 2 | P1 | P2 | P3 | P4 |
|---|---|---|---|---|---|
| Student 1 | -- | 1 | 2 | 3 | 4 |
| Student 2 | -- | 1 | 2 | 3 | 4 |
| Student 3 | -- | 4 | 2 | 3 | 1 |
| Student 4 | -- | 1 | 2 | 3 | 4 |
| Student 5 | -- | 2 | 4 | 1 | 3 |
| Student 6 | -- | 2 | 1 | 3 | 4 |
| Student 7 | -- | 1 | 3 | 4 | 2 |
| Student 8 | -- | 1 | 3 | 4 | 2 |

# Topic Sign-Up: Example 1

| Member 1 | Member 2 | P1 | P2 | P3 | P4 |
|---|---|---|---|---|---|
| Student 1 | -- | 1 | 2 | 3 | 4 |
| Student 2 | -- | 1 | 2 | 3 | 4 |
| Student 3 | -- | 4 | 2 | 3 | 1 |
| Student 4 | -- | 1 | 2 | 3 | 4 |
| Student 5 | -- | 2 | 4 | 1 | 3 |
| Student 6 | -- | 2 | 1 | 3 | 4 |
| Student 7 | -- | 1 | 3 | 4 | 2 |
| Student 8 | -- | 1 | 3 | 4 | 2 |

# Topic Sign-Up: Example 1

| Member 1 | Member 2 | P1 | P2 | P3 | P4 |
|---|---|---|---|---|---|
| Student 1 | -- | 1 | 2 | 3 | 4 |
| Student 2 | -- | 1 | 2 | 3 | 4 |
| Student 3 | -- | 4 | 2 | 3 | 1 |
| Student 4 | -- | 1 | 2 | 3 | 4 |
| Student 5 | -- | 2 | 4 | 1 | 3 |
| Student 6 | -- | 2 | 1 | 3 | 4 |
| Student 7 | -- | 1 | 3 | 4 | 2 |
| Student 8 | -- | 1 | 3 | 4 | 2 |

# Topic Sign-Up: Example 1

| Member 1 | Member 2 | P1 | P2 | P3 | P4 |
|---|---|---|---|---|---|
| Student 1 | -- | 1 | 2 | 3 | 4 |
| Student 2 | -- | 1 | 2 | 3 | 4 |
| Student 3 | -- | 4 | 2 | 3 | 1 |
| Student 4 | -- | 1 | 2 | 3 | 4 |
| Student 5 | -- | 2 | 4 | 1 | 3 |
| Student 6 | -- | 2 | 1 | 3 | 4 |
| Student 7 | -- | 1 | 3 | 4 | 2 |
| Student 8 | -- | 1 | 3 | 4 | 2 |

# Topic Sign-Up: Example 2

| Member 1 | Member 2 | P1 | P2 | P3 | P4 |
|---|---|---|---|---|---|
| Student 1 | Student 2 | 1 | 2 | 3 | 4 |
| Student 3 | -- | 4 | 2 | 3 | 1 |
| Student 4 | -- | 1 | 2 | 3 | 4 |
| Student 5 | -- | 2 | 4 | 1 | 3 |
| Student 6 | -- | 2 | 1 | 3 | 4 |
| Student 7 | Student 8 | 1 | 3 | 4 | 2 |

# Topic Sign-Up: Example 2

| Member 1 | Member 2 | P1 | P2 | P3 | P4 |
|---|---|---|---|---|---|
| Student 1 | Student 2 | 1 | 2 | 3 | 4 |
| Student 3 | -- | 4 | 2 | 3 | 1 |
| Student 4 | -- | 1 | 2 | 3 | 4 |
| Student 5 | -- | 2 | 4 | 1 | 3 |
| Student 6 | -- | 2 | 1 | 3 | 4 |
| Student 7 | Student 8 | 1 | 3 | 4 | 2 |

# Topic Sign-Up: Example 2

| Member 1 | Member 2 | P1 | P2 | P3 | P4 |
|---|---|---|---|---|---|
| Student 1 | Student 2 | 1 | 2 | 3 | 4 |
| Student 3 | -- | 4 | 2 | 3 | 1 |
| Student 4 | -- | 1 | 2 | 3 | 4 |
| Student 5 | -- | 2 | 4 | 1 | 3 |
| Student 6 | -- | 2 | 1 | 3 | 4 |
| Student 7 | Student 8 | 1 | 3 | 4 | 2 |

# Topic Sign-Up: Example 2

| Member 1 | Member 2 | P1 | P2 | P3 | P4 |
|---|---|---|---|---|---|
| Student 1 | Student 2 | 1 | 2 | 3 | 4 |
| Student 3 | -- | 4 | 2 | 3 | 1 |
| Student 4 | -- | 1 | 2 | 3 | 4 |
| Student 5 | -- | 2 | 4 | 1 | 3 |
| Student 6 | -- | 2 | 1 | 3 | 4 |
| Student 7 | Student 8 | 1 | 3 | 4 | 2 |

# Topic Sign-Up: Example 2

| Member 1 | Member 2 | P1 | P2 | P3 | P4 |
|---|---|---|---|---|---|
| Student 1 | Student 2 | 1 | 2 | 3 | 4 |
| Student 3 | -- | 4 | 2 | 3 | 1 |
| Student 4 | -- | 1 | 2 | 3 | 4 |
| Student 5 | -- | 2 | 4 | 1 | 3 |
| Student 6 | -- | 2 | 1 | 3 | 4 |
| Student 7 | Student 8 | 1 | 3 | 4 | 2 |

# Topic Study

- Topic Study (30%)
  - Literature Review (15%) [Due: 10/1]
    - Conduct a literature review on the selected topic
    - 4 suggested papers + at least 10 additional papers published after 2024
    - Page limit: 4 – 5 pages

| | | | |
|---|---|---|---|
| W9 | 10/19 | AI-Generated Text Detection | DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature, ICML 2023<br>Fast-DetectGPT: Efficient Zero-Shot Detection of Machine-Generated Text via Conditional Probability Curvature, ICLR 2024<br>A Watermark for Large Language Models, ICML 2023<br>Watermark Stealing in Large Language Models, ICML 2024 [Present] |
| | 10/21 | Hallucinations | SelfCheckGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models, EMNLP 2023<br>How Language Model Hallucinations Can Snowball, ICML 2024<br>Lookback Lens: Detecting and Mitigating Contextual Hallucinations in Large Language Models Using Only Attention Maps, EMNLP 2024<br>Chain-of-Verification Reduces Hallucination in Large Language Models, ACL-Findings 2024 [Present] |

# Topic Study

- Topic Study (30%)
  - Literature Review (15%) [Due: 10/1]
    - Categorize papers, discuss connections between papers, compare pros and cons
    - ~~Paper 1 summary, Paper 2 summary, …, Paper N summary~~ → Don't do that
      - A Survey of Large Language Model-Based Search Agents
      - It's High Time: A Survey of Temporal Question Answering
      - A Survey of Inductive Reasoning for Large Language Models
      - Table Question Answering in the Era of Large Language Models: A Comprehensive Survey of Tasks, Methods, and Evaluation
      - Scaling Beyond Context: A Survey of Multimodal Retrieval-Augmented Generation for Document Understanding
      - A Comprehensive Survey of Process Reward Models: Data Generation, Model Construction, and Usage

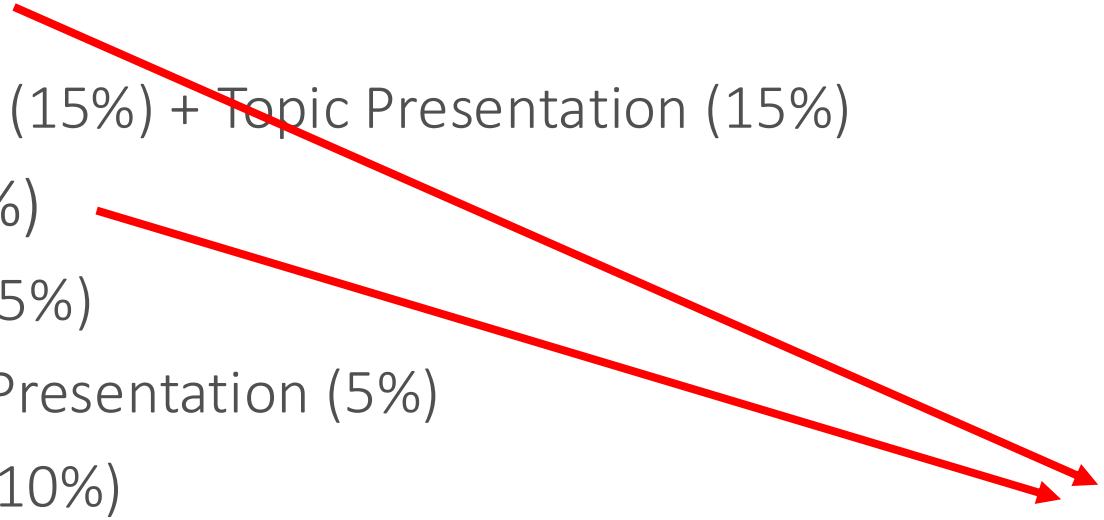
# Topic Study

- Topic Study (30%)
  - Topic Presentation (15%)
    - 30 min of presentation + 10 min of discussion
    - 1 assigned + at least 2 self-selected papers published after 2024
    - Email your slides to the instructor at least 2 days before your presentation
      - Monday presentation: by Saturday 11:59pm
      - Wednesday presentation: by Monday 11:59pm

| | | | |
|---|---|---|---|
| W9 | 10/19 | AI-Generated Text Detection | DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature, ICML 2023<br>Fast-DetectGPT: Efficient Zero-Shot Detection of Machine-Generated Text via Conditional Probability Curvature, ICLR 2024<br>A Watermark for Large Language Models, ICML 2023<br>Watermark Stealing in Large Language Models, ICML 2024 [Present] |
| | 10/21 | Hallucinations | SelfCheckGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models, EMNLP 2023<br>How Language Model Hallucinations Can Snowball, ICML 2024<br>Lookback Lens: Detecting and Mitigating Contextual Hallucinations in Large Language Models Using Only Attention Maps, EMNLP 2024<br>Chain-of-Verification Reduces Hallucination in Large Language Models, ACL-Findings 2024 [Present] |

# About Teams

- Topic Study (30%)
  - Literature Review (15%) + Topic Presentation (15%)
- Course Project (49%)
  - Project Proposal (5%)
  - Project Highlight Presentation (5%)
  - Midterm Report (10%)
  - Final Presentation (12%)
  - Final Report (17%)



Not necessary to be the same team

# Questions?

# What are Language Models?

- A probabilistic model of a sequence of words
  - Evaluate the probability of whether a text is acceptable
- How likely are the following sentences?

*The dog is barking at the stranger in the yard.*

*Yesterday, I went to the park and saw a group of children playing soccer.*

*The sky colorful because painted an artist.*

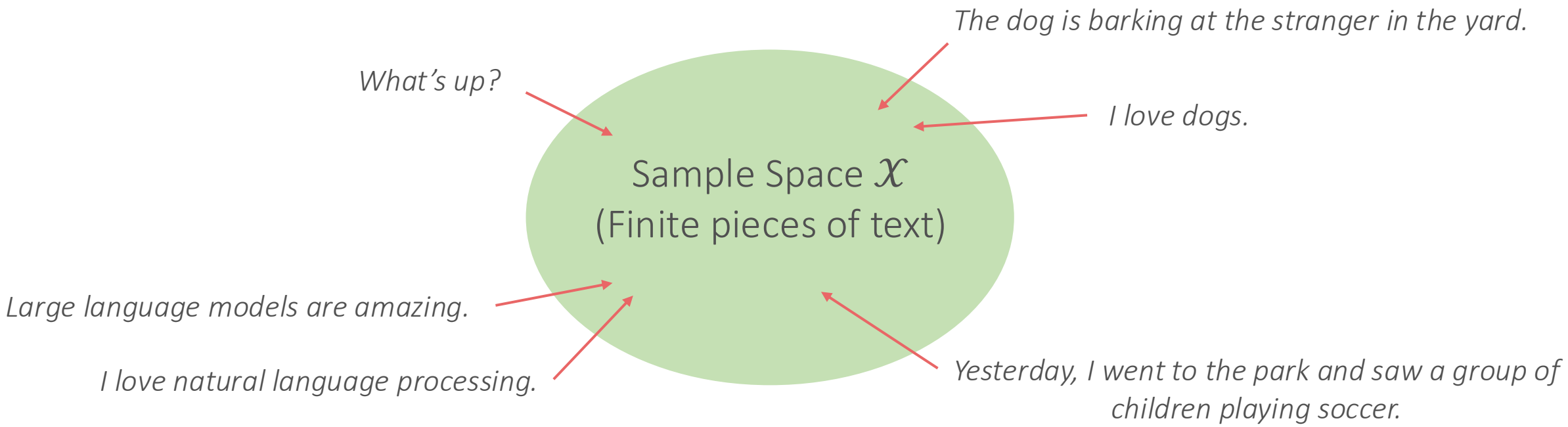
*Cats upon they chairs sleeping their dreams fall.*

*Plorp zix flanned the quibble through treemunk.*

# Language Models

- Learn the probability distribution over texts $x = [w_1, w_2, \ldots, w_l] \in \mathcal{X}$

$$P(x) = P(w_1, w_2, \ldots, w_l)$$



# What Can Language Models Do?

- Score texts

  $P(\textit{The dog is barking at the stranger in the yard.})$ → High

  $P(\textit{Cats upon they chairs sleeping their dreams fall.})$ → Low

- Generate texts

  $$\tilde{x} \sim P(\mathcal{X})$$

# Auto-Regressive Language Models

$$P(w_1, w_2, w_3, \dots, w_l) = P(w_1)P(w_2, w_3, \dots, w_l|w_1)$$
$$= P(w_1)P(w_2|w_1)(w_3, \dots, w_l|w_1, w_2)$$
$$= P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)(w_4, \dots, w_l|w_1, w_2, w_3)$$
$$= \prod_{i=1}^{l} P(w_i|w_1, w_2, \dots, w_{i-1})$$

$P(\textit{She likes to go hiking}) = P(\textit{She}) \cdot P(\text{likes}|\text{She}) \cdot P(\text{to}|\text{She likes})$
$\cdot P(\text{go}|\text{She likes to}) \cdot P(\text{hiking}|\text{She likes to go})$

# Auto-Regressive Language Models

$$P(w_1, w_2, w_3, \ldots, w_l) = \prod_{i=1}^{l} P(w_i | w_1, w_2, \ldots, w_{i-1})$$

Next Token

Context

Next token prediction problem based on context

Challenge: How to predict $P(w_i | w_1, w_2, \ldots, w_{i-1})$?

# Unigram Language Models

Assumption: $P(w_i|w_1, w_2, \ldots, w_{i-1}) \approx P(w_i)$

$$P(w_1, w_2, w_3, \ldots, w_l) \approx P(w_1)P(w_2)P(w_3) \ldots P(w_l)$$

Similar to the concept of bag-of-words!

How to calculate $P(w_i)$?

A count-based solution: Collect training corpus and count

$$P(w_i) = \frac{C_{train}(w_i)}{\sum_t C_{train}(w_t)}$$

# Bigram Language Models

Assumption: $P(w_i|w_1, w_2, \ldots, w_{i-1}) \approx P(w_i|w_{i-1})$

$$P(w_1, w_2, w_3, \ldots, w_l) \approx P(w_1)P(w_2|w_1)P(w_3|w_2)P(w_4|w_3) \ldots P(w_l|w_{l-1})$$

The prediction of the next token depends only on the previous token

A count-based solution: Collect training corpus and count

$$P(w_i|w_{i-1}) = \frac{C_{train}(w_{i-1}, w_i)}{C_{train}(w_{i-1})}$$

# Trigram Language Models

Assumption: $P(w_i|w_1, w_2, \dots, w_{i-1}) \approx P(w_i|w_{i-2}, w_{i-1})$

$$P(w_1, w_2, w_3, \dots, w_l) \approx P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)P(w_4|w_2, w_3) \dots P(w_l|w_{l-2}, w_{l-1})$$

The prediction of the next token depends on the previous **two** tokens

A count-based solution: Collect training corpus and count

$$P(w_i|w_{i-1}, w_{i-2}) = \frac{C_{train}(w_{i-2}, w_{i-1}, w_i)}{C_{train}(w_{i-2}, w_{i-1})}$$

# N-Gram Language Models

Assumption: $P(w_i|w_1, w_2, \ldots, w_{i-1}) \approx P(w_i|w_{i-n+1}, \ldots, w_{i-1})$

$$P(w_1, w_2, w_3, \ldots, w_l) \approx \prod_i P(w_i|w_{i-n+1}, \ldots, w_{i-1})$$

The prediction of the next token depends on the previous **n** tokens

A count-based solution: Collect training corpus and count

$$P(w_i|w_{i-n+1}, \ldots, w_{i-1}) = \frac{C_{train}(w_{i-n+1}, \ldots, w_{i-1}, w_i)}{C_{train}(w_{i-n+1}, \ldots, w_{i-1})}$$

# How to Evaluate Language Models?

- Train a language model on a suitable training corpus
  - Assumption: observed sentences ≈ good sentences
- Test on different, unseen corpus
  - The higher probability that the language model assigns to the test set, the better (why?)
- Evaluation metric: Perplexity

# Perplexity (PPL)

For a corpus $\mathcal{X}$ with sentences $\{x_1, x_2, \ldots, x_n\}$

Likelihood $$P(\mathcal{X}) = \prod_{i=1}^{n} P(x_i)$$ The higher, the better

Log-Likelihood $$\log P(\mathcal{X}) = \sum_{i=1}^{n} \log P(x_i)$$ The higher, the better

Per-Word Log-Likelihood $$WLL(\mathcal{X}) = \frac{1}{W} \sum_{i=1}^{n} \log P(x_i)$$ The higher, the better

The number of words in the test corpus

# Perplexity (PPL)

For a corpus $\mathcal{X}$ with sentences $\{x_1, x_2, \dots, x_n\}$

Perplexity $e^{-WLL(\mathcal{X})}$

The lower, the better

$$WLL(\mathcal{X}) = \frac{1}{W}\sum_{i=1}^{n} \log P(x_i)$$

Computed by language model

Unigram Language Model

$$P(w_j | w_1, w_2, \dots, w_{j-1}) \approx P(w_j)$$

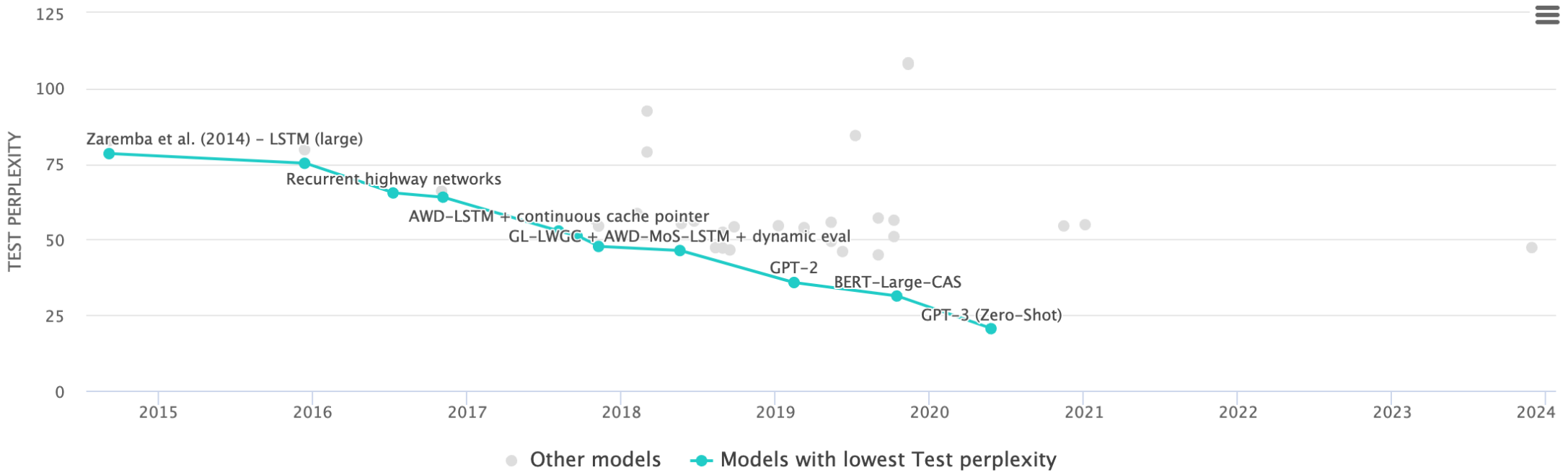
$$P(x) = \prod_j P(w_j)$$

Unigram Language Model

$$WLL(\mathcal{X}) = \frac{1}{W}\sum_i \sum_j \log P(w_{i,j})$$

Minimizing perplexity → maximizing probability of corpus

# Perplexity (PPL)






https://paperswithcode.com/sota/language-modelling-on-penn-treebank-word

# Text Generation with Language Models

Trigram Language Model

$$P(w_1, w_2, w_3, \dots, w_l) \approx \prod_i P(w_i | w_{i-2}, w_{i-1})$$

- Generate the first word $w_1 \sim P(w)$
- Generate the second word $w_2 \sim P(w|w_1)$
- Generate the third word $w_3 \sim P(w|w_1, w_2)$
- Generate the fourth word $w_4 \sim P(w|w_2, w_3)$
- Generate the fifth word $w_5 \sim P(w|w_3, w_4)$
- …
- Until the end of the sentence <eos>

# Generation Examples

**Unigram**

*release millions See ABC accurate President of Donald Will cheat them a CNN megynkelly experience @ these word out- the*

**Bigram**

*Thank you believe that @ ABC news, Mississippi tonight and the false editorial I think the great people Bill Clinton . ''*

**Trigram**

*We are going to MAKE AMERICA GREAT AGAIN! #MakeAmericaGreatAgain https: //t.co/DjkdAzT3WV*

Typical LMs are not sufficient to handle long-range dependencies

"Alice/Bob could not go to work that day because she/he had a doctor's appointment"

# Different Decoding Strategies

Random Sampling

$$w_3 \sim P(w|w_1, w_2)$$

Greedy Decoding

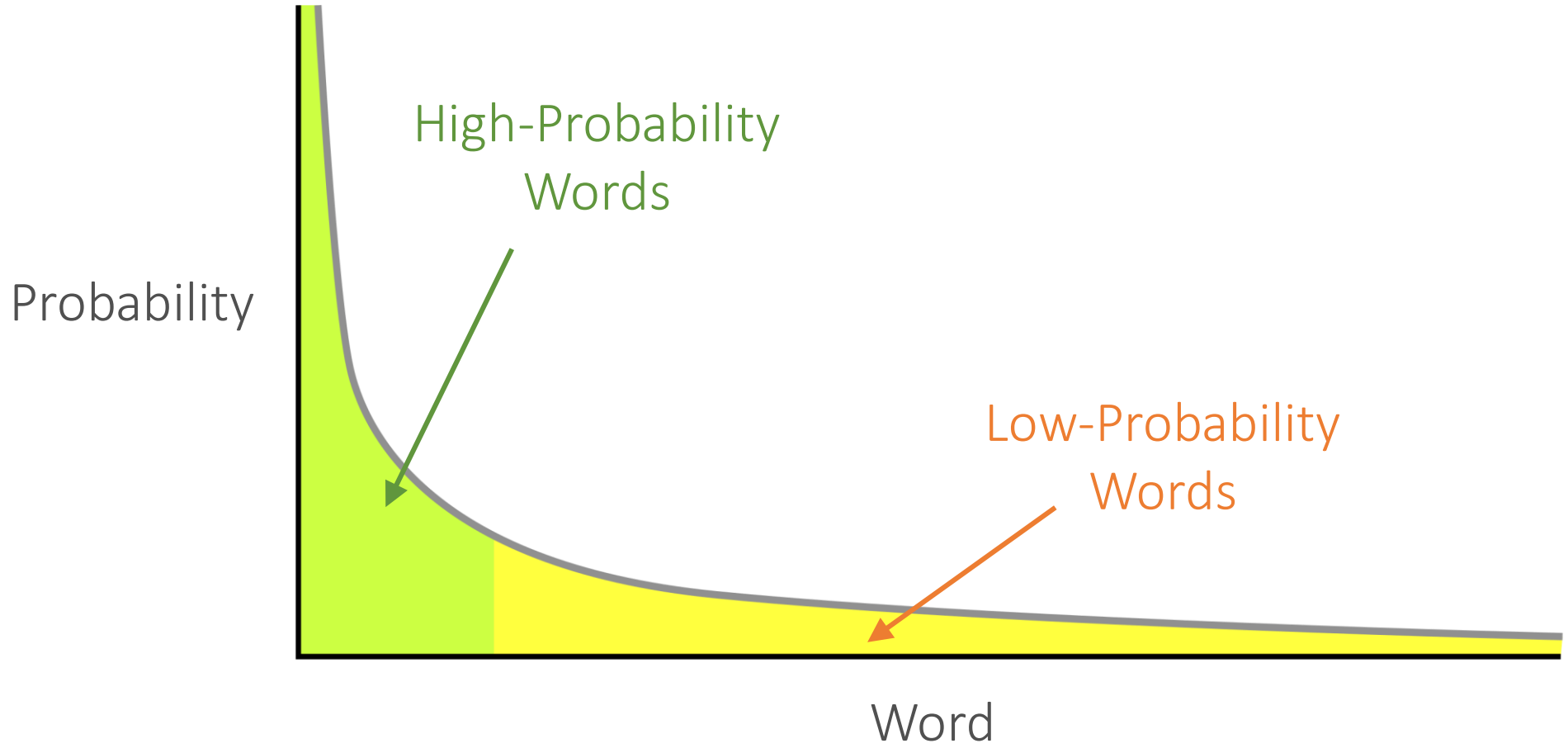
$$w_3 = \arg\max_{w \in \mathcal{V}} P(w|w_1, w_2)$$

Temperature Scaling

$$P(w_i) = \text{softmax}(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

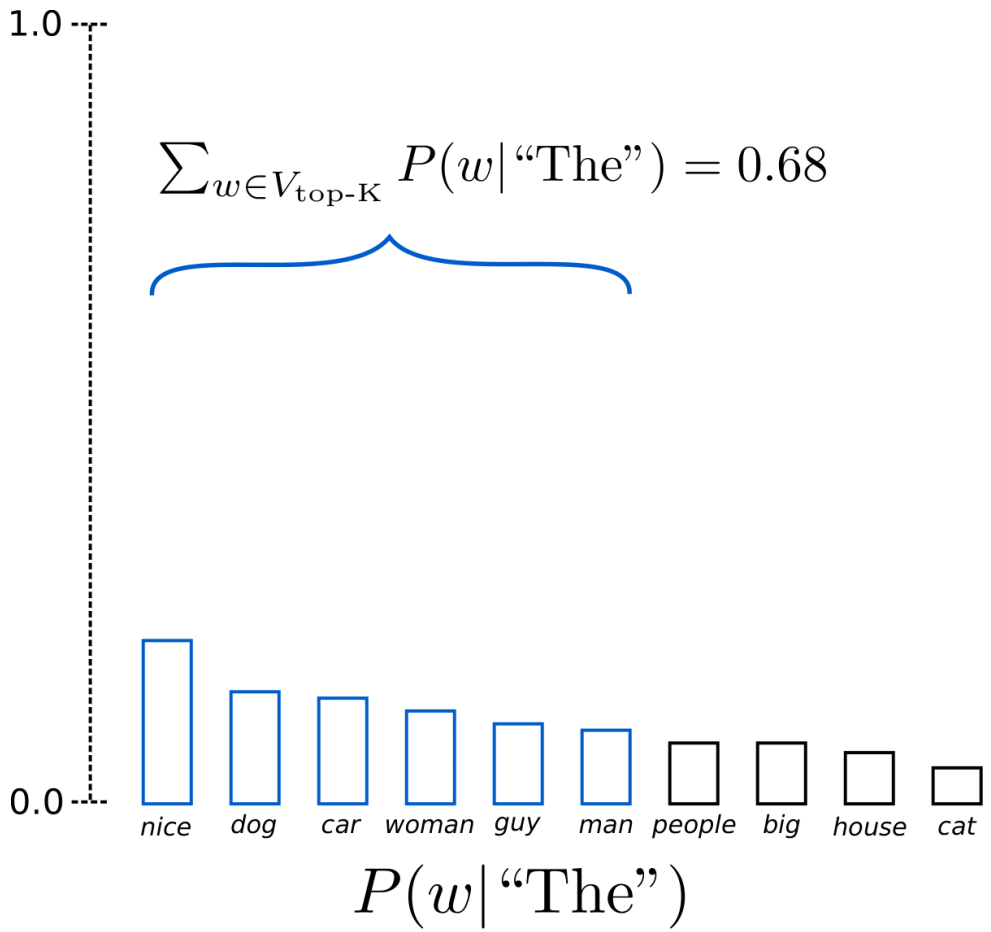
$$P(w_i) = \text{softmax}(z_i) = \frac{e^{\frac{z_i}{T}}}{\sum_j e^{\frac{z_j}{T}}}$$

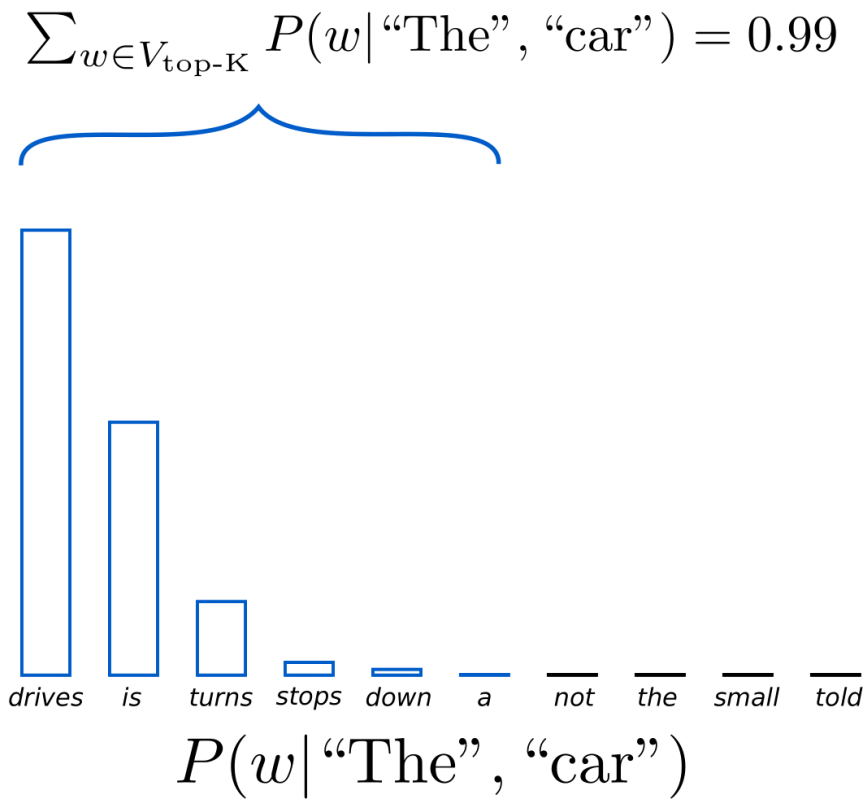
# Long-Tail Word Distribution




https://en.wikipedia.org/wiki/Long_tail

# Top-K Sampling

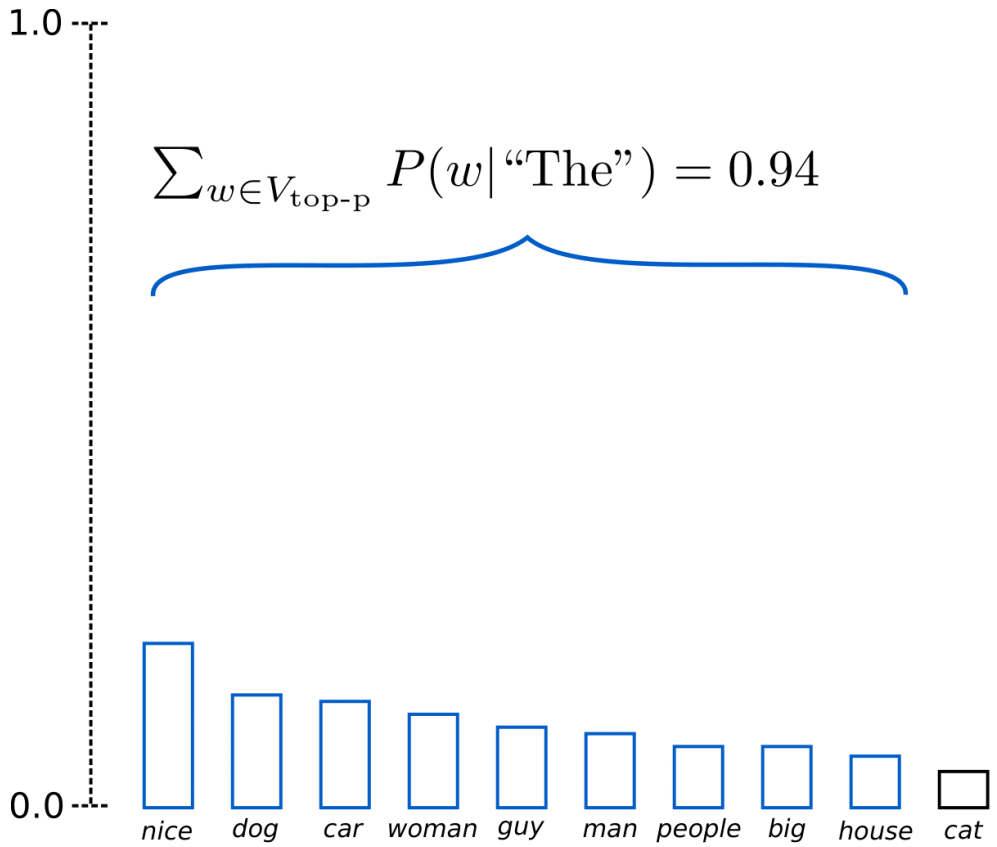


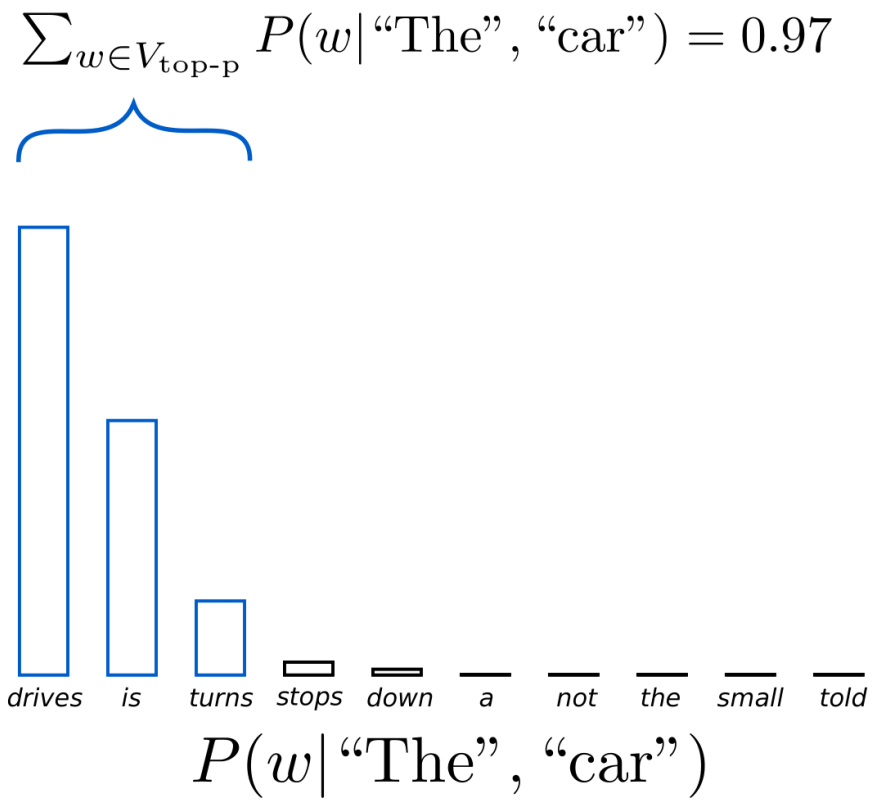






https://huggingface.co/blog/how-to-generate

# Top-P Sampling

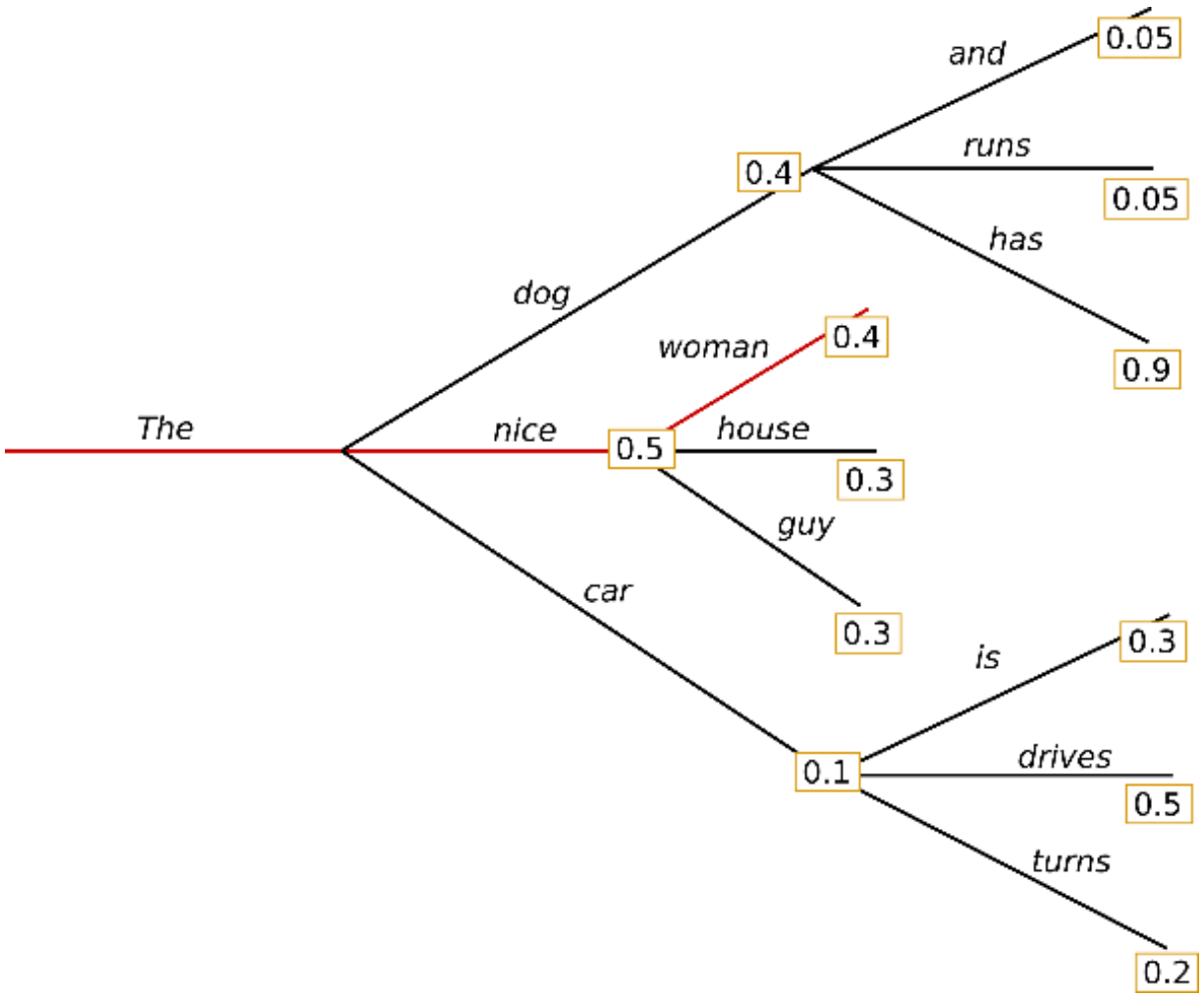





https://huggingface.co/blog/how-to-generate

# Beam Search

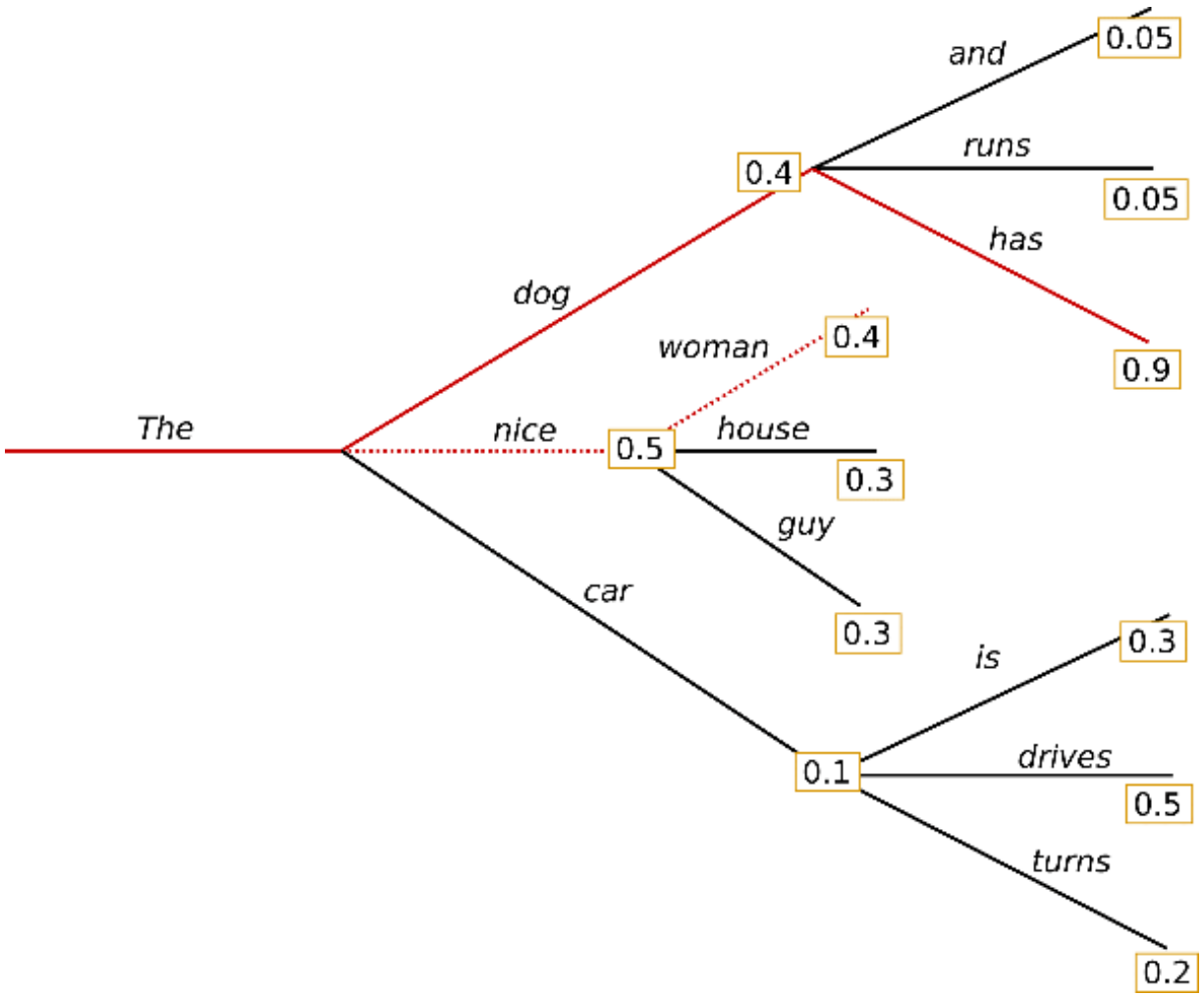



Sequential sampling does not always lead to the optimal sequence



https://huggingface.co/blog/how-to-generate

# Beam Search



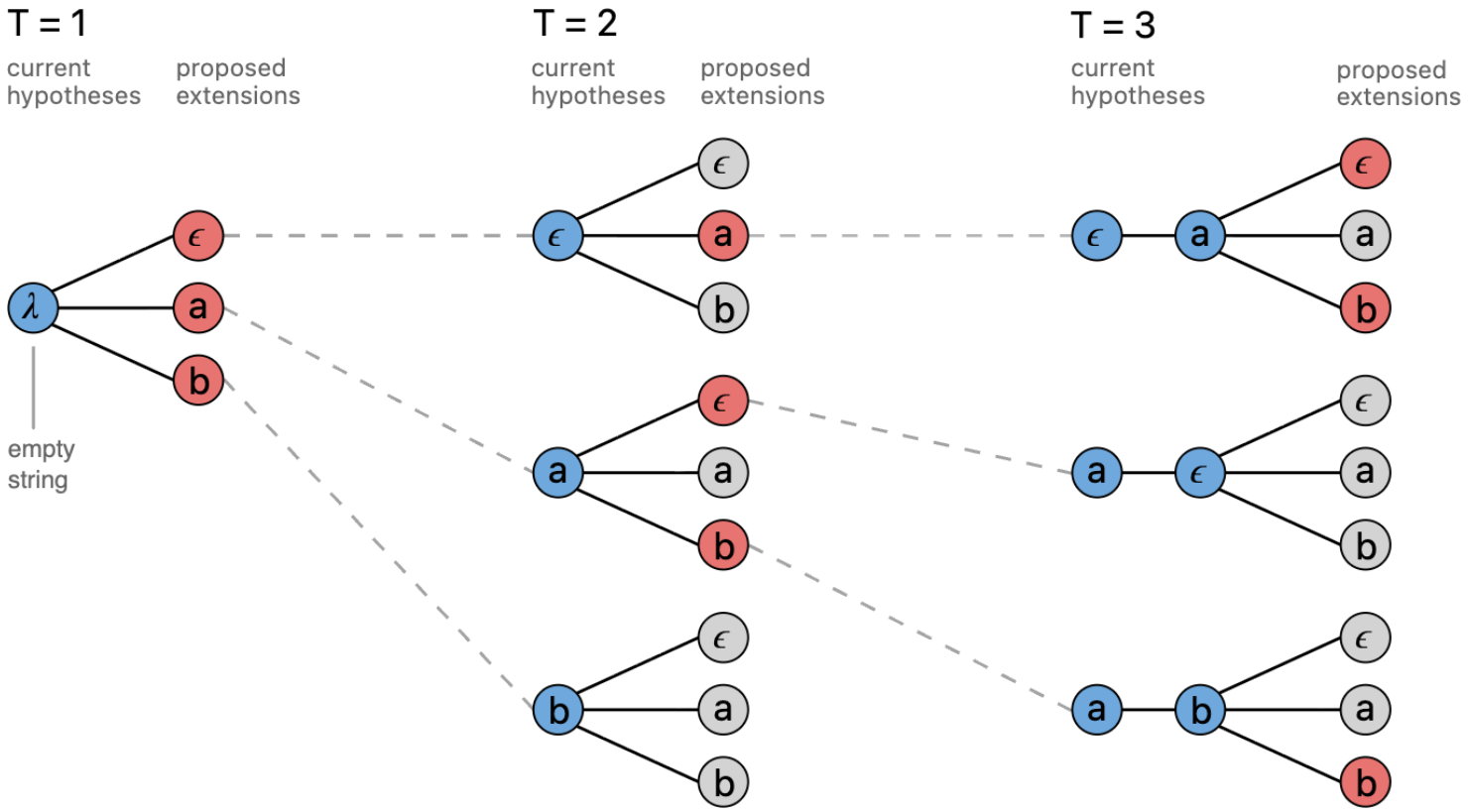

Sequential sampling does not always lead to the optimal sequence



https://huggingface.co/blog/how-to-generate

# Beam Search

- Beam size $m$: Keep $m$ best paths in the queue






https://distill.pub/2017/ctc/

# What Can Language Models Do?

- Score texts

$P(\textit{The dog is barking at the stranger in the yard.}) \rightarrow$ High

$P(\textit{Cats upon they chairs sleeping their dreams fall.}) \rightarrow$ Low

Compute perplexity

- Generate texts

$$\tilde{x} \sim P(\mathcal{X})$$

Sample from word distribution

# N-Gram Language Models

Assumption: $P(w_i|w_1, w_2, \dots, w_{i-1}) \approx P(w_i|w_{i-n+1}, \dots, w_{i-1})$

$$P(w_1, w_2, w_3, \dots, w_l) \approx \prod_i P(w_i|w_{i-n+1}, \dots, w_{i-1})$$

The prediction of the next token depends on the previous **n** tokens

A count-based solution: Collect training corpus and count

$$P(w_i|w_{i-n+1}, \dots, w_{i-1}) = \frac{C_{train}(w_{i-n+1}, \dots, w_{i-1}, w_i)}{C_{train}(w_{i-n+1}, \dots, w_{i-1})}$$

Any Problems?

# Unseen Patterns in Training Corpus

- Not all n-grams in the test set will be observed in training corpus
- Training corpus
  - I like apples
  - I love bananas
- Test set
  - I like bananas
  - I love apples
- This problem becomes severe when n is large

# N-Gram Language Models

Assumption: $P(w_i|w_1, w_2, \dots, w_{i-1}) \approx P(w_i|w_{i-n+1}, \dots, w_{i-1})$

$$P(w_1, w_2, w_3, \dots, w_l) \approx \prod_i P(w_i|w_{i-n+1}, \dots, w_{i-1})$$

The prediction of the next token depends on the previous **n** tokens

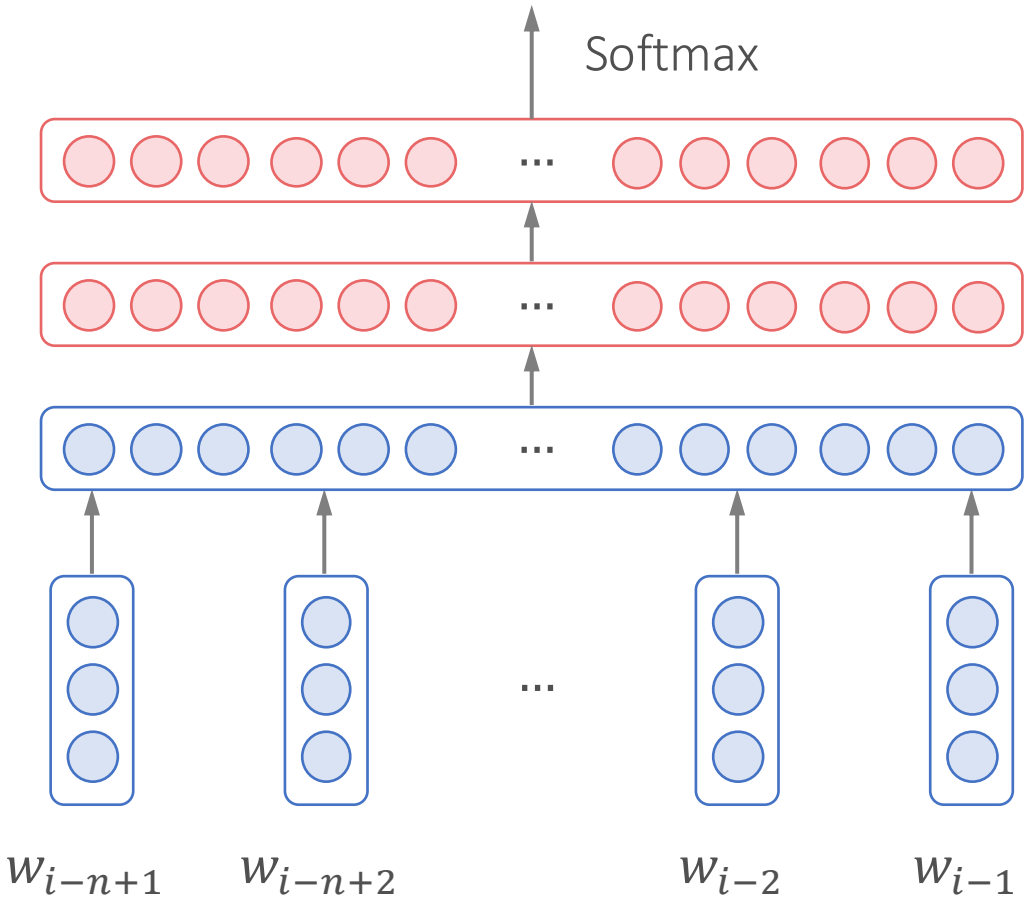
A count-based solution: Collect training corpus and count

$$P(w_i|w_{i-n+1}, \dots, w_{i-1}) = \frac{C_{train}(w_{i-n+1}, \dots, w_{i-1}, w_i)}{C_{train}(w_{i-n+1}, \dots, w_{i-1})}$$

Can we compute the probability in a different way?

# Neural Language Models

$$P(w_i|w_{i-n+1}, \cdots, w_{i-1})$$



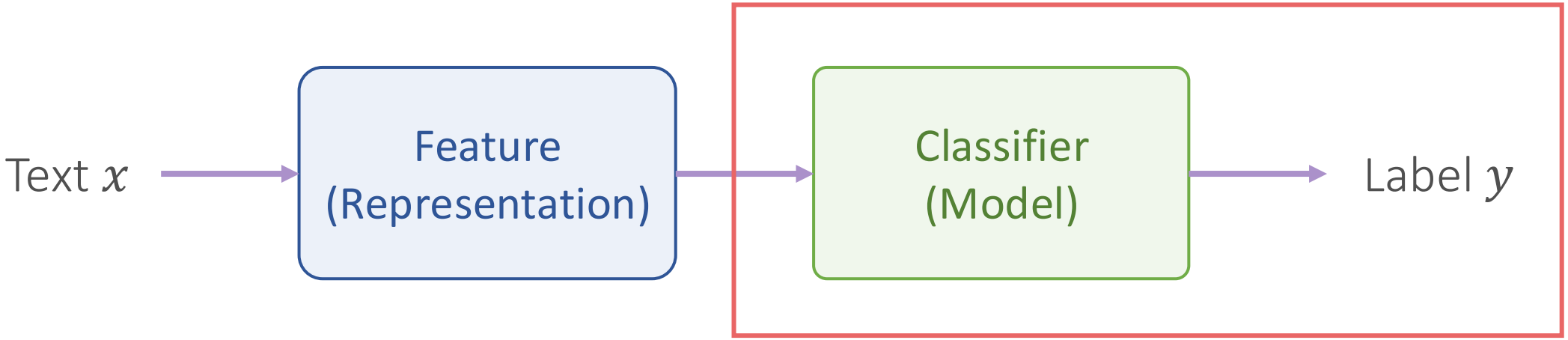

Training corpus

- I like apples
- I love bananas

Test set

- I love apples
- I like bananas

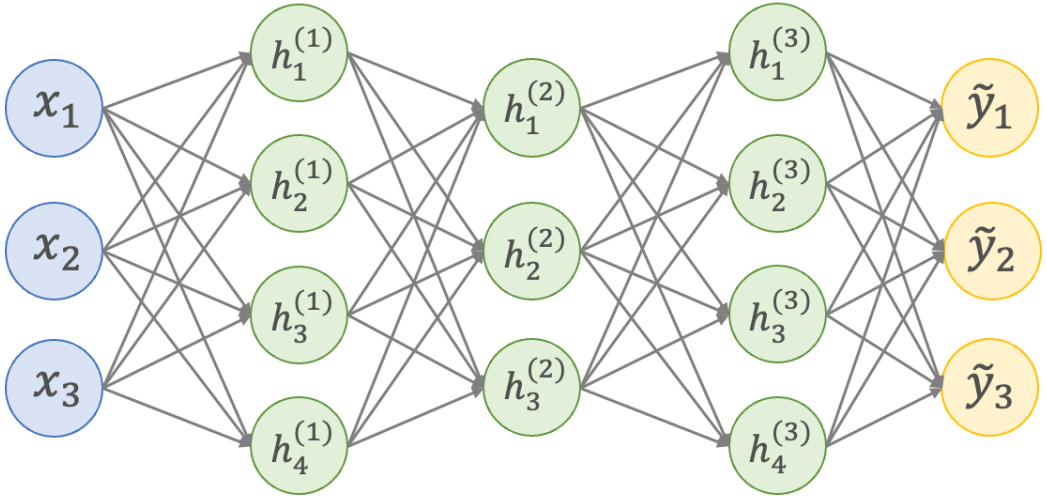
# Recap: A General Framework for Text Classification




- Teach the model how to make prediction $y$
- Logistic regression, neural networks, CNN, RNN, LSTM, Transformers

# A Simple Approach: Averaged Embeddings + DNN

| | Alice | treats | Bob | well | |
|---|---|---|---|---|---|
| Dimension 1 | 0.7 | 2.7 | -0.1 | -5.7 | -0.6 |
| Dimension 2 | 8.6 | -3.9 | 6.7 | -9.8 | 0.4 |
| Dimension 3 | -2.4 | -5.6 | 1.5 | -1.6 | -1.6 |
| Dimension 4 | 2.3 | 1.1 | 2.0 | -1.0 | 1.1 |




$$\mathcal{L}_{CE}(y, \tilde{y}) = -\sum_{c=0}^{C} y_c \log P(y = c|\ \mathbf{x})$$

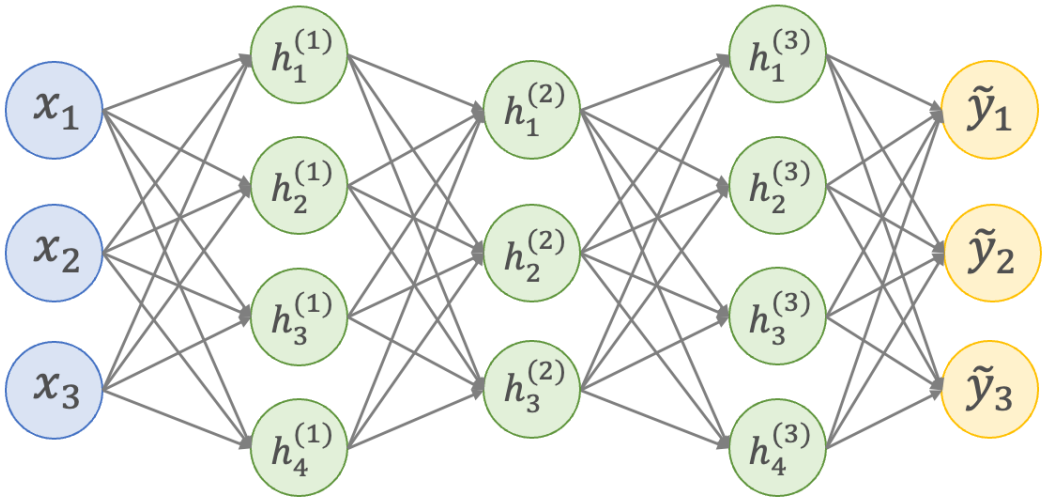
Any problems?

# A Simple Approach: Concatenated Embeddings + DNN

| | Alice | treats | Bob | well |
|---|---|---|---|---|
| Dimension 1 | 0.7 | 2.7 | -0.1 | -5.7 |
| Dimension 2 | 8.6 | -3.9 | 6.7 | -9.8 |
| Dimension 3 | -2.4 | -5.6 | 1.5 | -1.6 |
| Dimension 4 | 2.3 | 1.1 | 2.0 | -1.0 |

0.7
8.6
-2.4
2.3
2.7
-3.9
-5.6
1.1
...
-5.7
-9.8
-1.6
-1.0

Any problems?




$$\mathcal{L}_{CE}(y, \tilde{y}) = -\sum_{c=0}^{C} y_c \log P(y = c|\ \mathbf{x})$$

# Challenges

- Averaged Embeddings
  - Lose order information
- Concatenated Embeddings
  - Cannot handle various lengths

# Solution 1: Capture Local Order Information

*Bob likes Alice very much*

| | |
|---|---|
| **Unigram** | *{Bob, likes, Alice, very, much}* |
| **Bigram** | *{Bob likes, likes Alice, Alice very, very much}* |
| **Trigram** | *{Bob likes Alice, likes Alice very, Alice very much}* |
| **4-gram** | *{Bob likes Alice very, likes Alice very much}* |

We can infer global order information from local order information

# Convolutional Neural Network (CNN)

- Capture local features (N-grams)
  - Filters (Kernels)
- Hierarchical feature learning
  - Multiple layers

# Convolutional Neural Network (CNN)

Learnable Weight (Filter)
Filter Size = 3

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \\ \dots & \dots & \dots \\ w_{4,1} & w_{4,2} & w_{4,3} \end{bmatrix}$$

# Convolutional Neural Network (CNN)

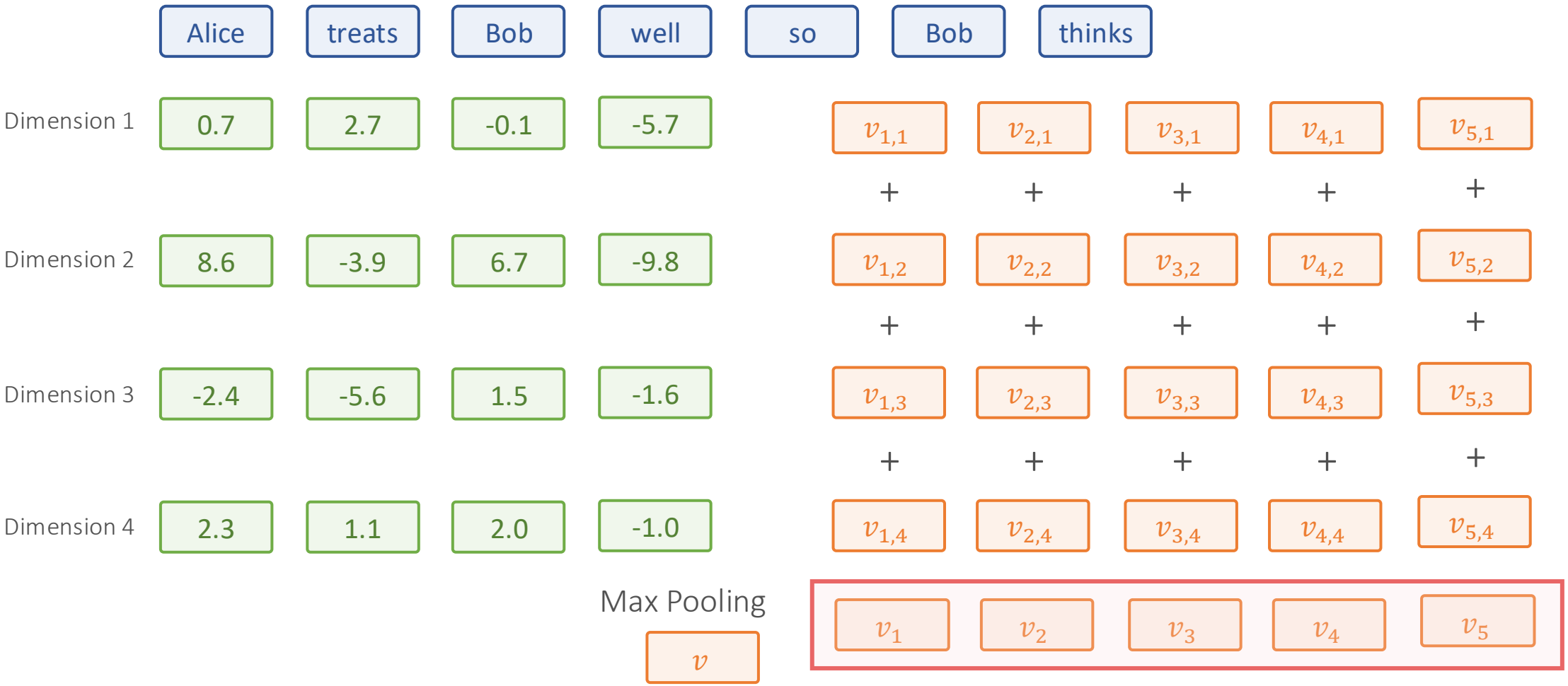
Learnable Weight (Filter)
Filter Size = 3

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \\ \cdots & \cdots & \cdots \\ w_{4,1} & w_{4,2} & w_{4,3} \end{bmatrix}$$

# Convolutional Neural Network (CNN)

Learnable Weight (Filter)
Filter Size = 3

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \\ \dots & \dots & \dots \\ w_{4,1} & w_{4,2} & w_{4,3} \end{bmatrix}$$



# Convolutional Neural Network (CNN)

Learnable Weight (Filter)
Filter Size = 3

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \\ \dots & \dots & \dots \\ w_{4,1} & w_{4,2} & w_{4,3} \end{bmatrix} \quad \mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \\ \dots & \dots & \dots \\ w_{4,1} & w_{4,2} & w_{4,3} \end{bmatrix} \quad \mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \\ \dots & \dots & \dots \\ w_{4,1} & w_{4,2} & w_{4,3} \end{bmatrix}$$

| | Alice | treats | Bob | well |
|---|---|---|---|---|
| Dimension 1 | 0.7 | 2.7 | -0.1 | -5.7 |
| Dimension 2 | 8.6 | -3.9 | 6.7 | -9.8 |
| Dimension 3 | -2.4 | -5.6 | 1.5 | -1.6 |
| Dimension 4 | 2.3 | 1.1 | 2.0 | -1.0 |

$v$ $v$ $v$

# Convolutional Neural Network (CNN)

Filter Size = 3 $\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \\ \cdots & \cdots & \cdots \\ w_{4,1} & w_{4,2} & w_{4,3} \end{bmatrix}$ Filter Size = 2 $\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} \\ \cdots & \cdots \\ w_{4,1} & w_{4,2} \end{bmatrix}$ Filter Size = 4 $\mathbf{W} = \begin{bmatrix} w_{1,1} & \cdots & w_{1,4} \\ \cdots & \cdots & \cdots \\ w_{4,1} & \cdots & w_{4,4} \end{bmatrix}$

| | Alice | treats | Bob | well |
|---|---|---|---|---|
| Dimension 1 | 0.7 | 2.7 | -0.1 | -5.7 |
| Dimension 2 | 8.6 | -3.9 | 6.7 | -9.8 |
| Dimension 3 | -2.4 | -5.6 | 1.5 | -1.6 |
| Dimension 4 | 2.3 | 1.1 | 2.0 | -1.0 |

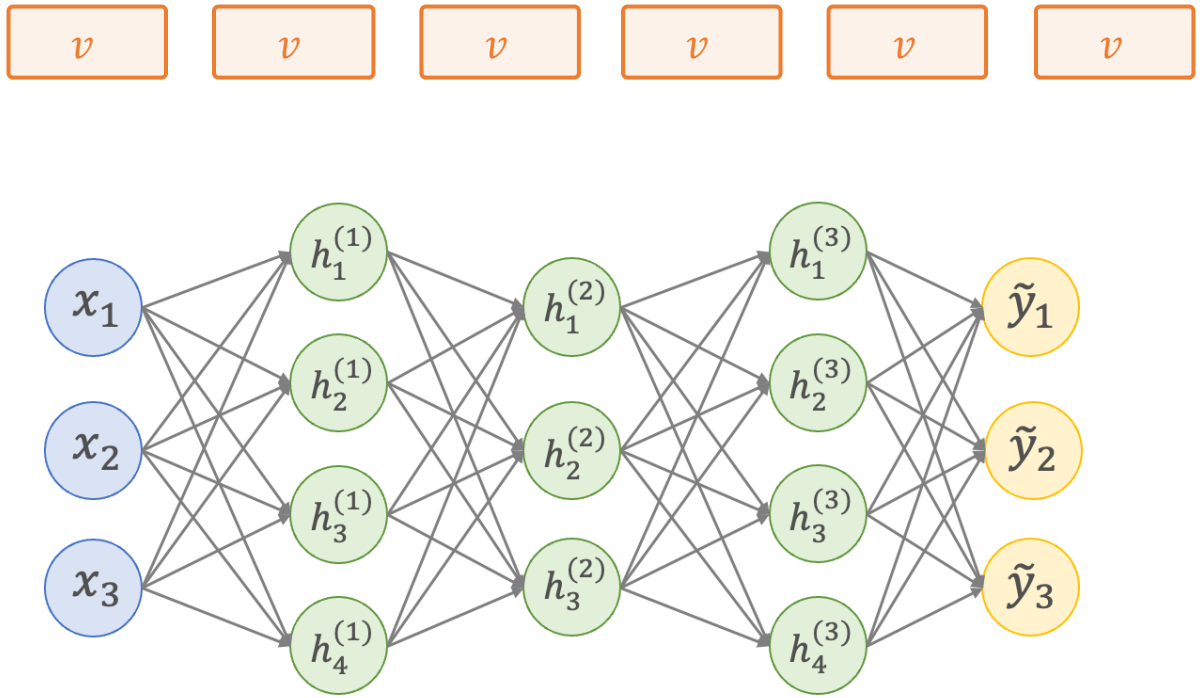
$w_{1,1}$*0.7 + $w_{1,2}$*2.7

$w_{2,1}$*8.6 + $w_{2,2}$*-3.9 + $w_{2,3}$*6.7 + $w_{2,4}$*-9.8

$v$ $v$ $v$

# Convolutional Neural Network (CNN)

| | Alice | treats | Bob | well |
|---|---|---|---|---|
| Dimension 1 | 0.7 | 2.7 | -0.1 | -5.7 |
| Dimension 2 | 8.6 | -3.9 | 6.7 | -9.8 |
| Dimension 3 | -2.4 | -5.6 | 1.5 | -1.6 |
| Dimension 4 | 2.3 | 1.1 | 2.0 | -1.0 |




$$\mathcal{L}_{CE}(y, \tilde{y}) = -\sum_{c=0}^{C} y_c \log P(y = c|\,\mathbf{x})$$

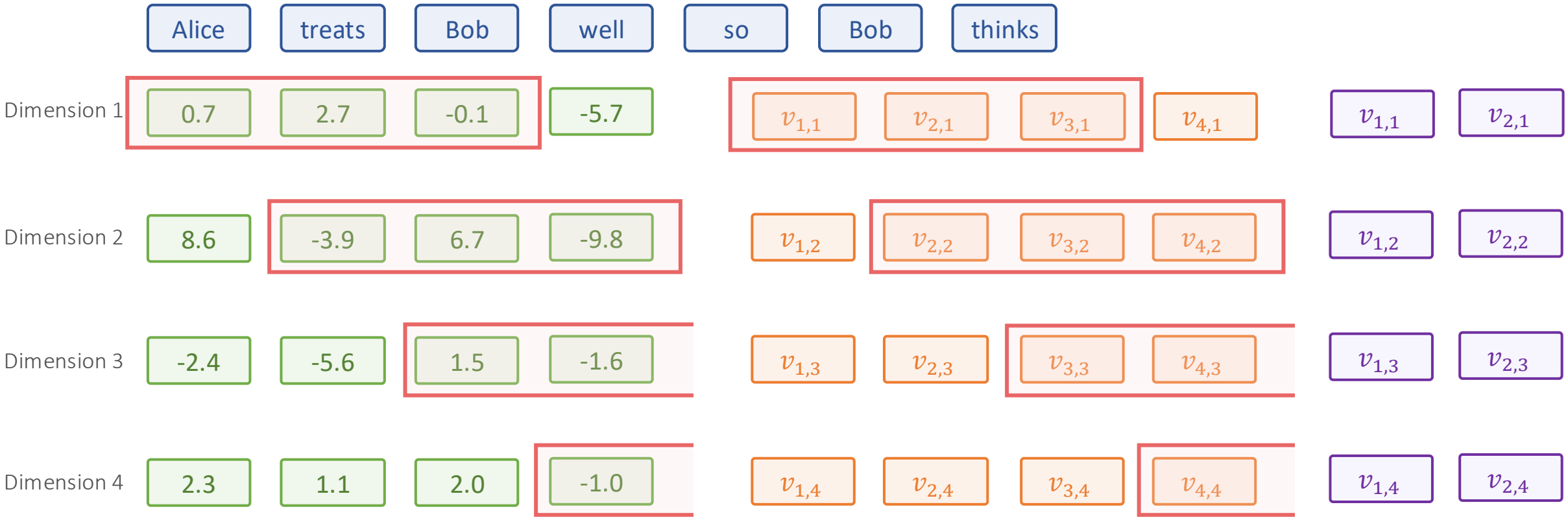
# From Single Layer to Multiple Layers

Learnable Weight (Layer 1)
Filter Size = 3

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \\ \cdots & \cdots & \cdots \\ w_{4,1} & w_{4,2} & w_{4,3} \end{bmatrix}$$

Learnable Weight (Layer 2)
Filter Size = 3

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \\ \cdots & \cdots & \cdots \\ w_{4,1} & w_{4,2} & w_{4,3} \end{bmatrix}$$



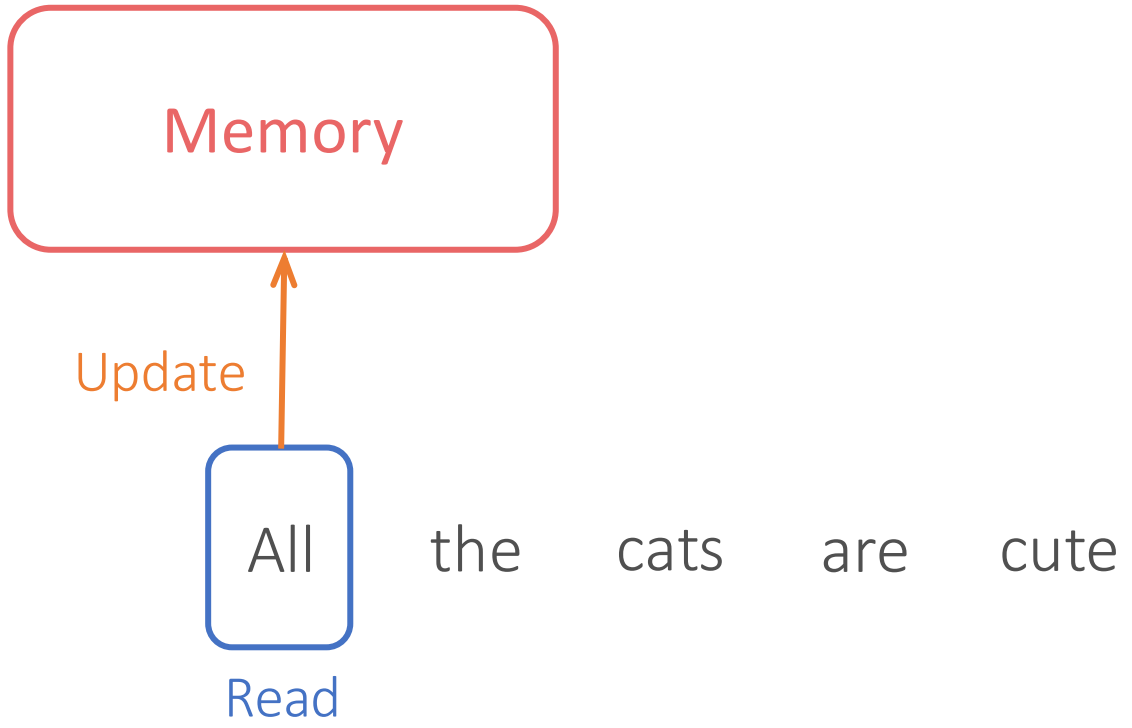

Capture high-order or hierarchical information

# Convolutional Neural Network (CNN)

- Capture local features (N-grams)
  - Filters (Kernels)
- Hierarchical feature learning
  - Multiple layers
- The whole process is still not similar to how human read texts
- Can we model reading texts in a sequential way?

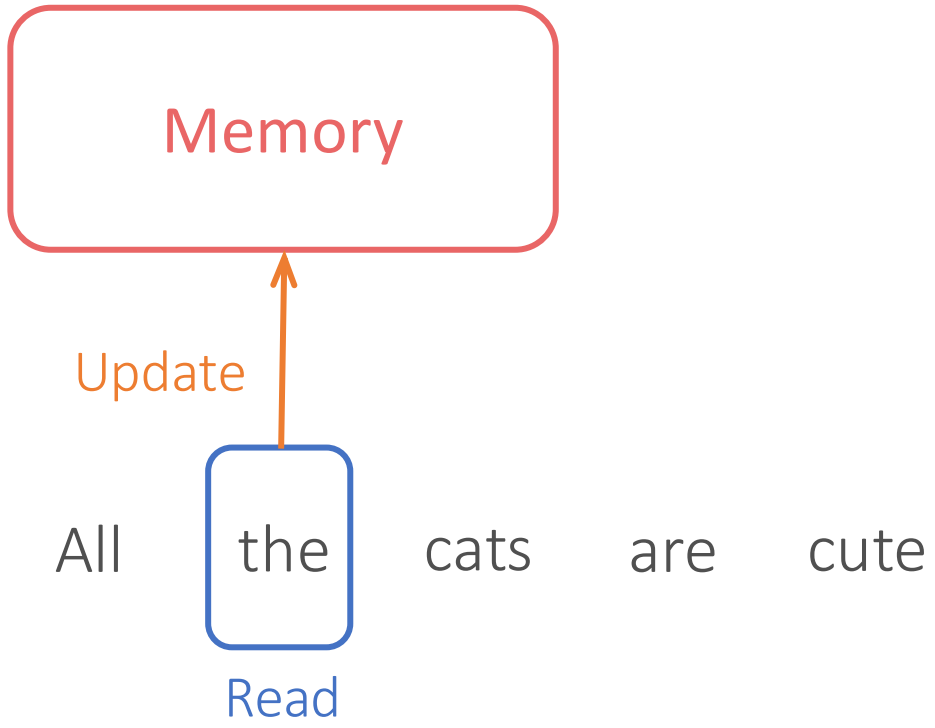
# Recurrent Neural Network (RNN)

- Read texts sequentially like a human with memory
  - Read → update memory



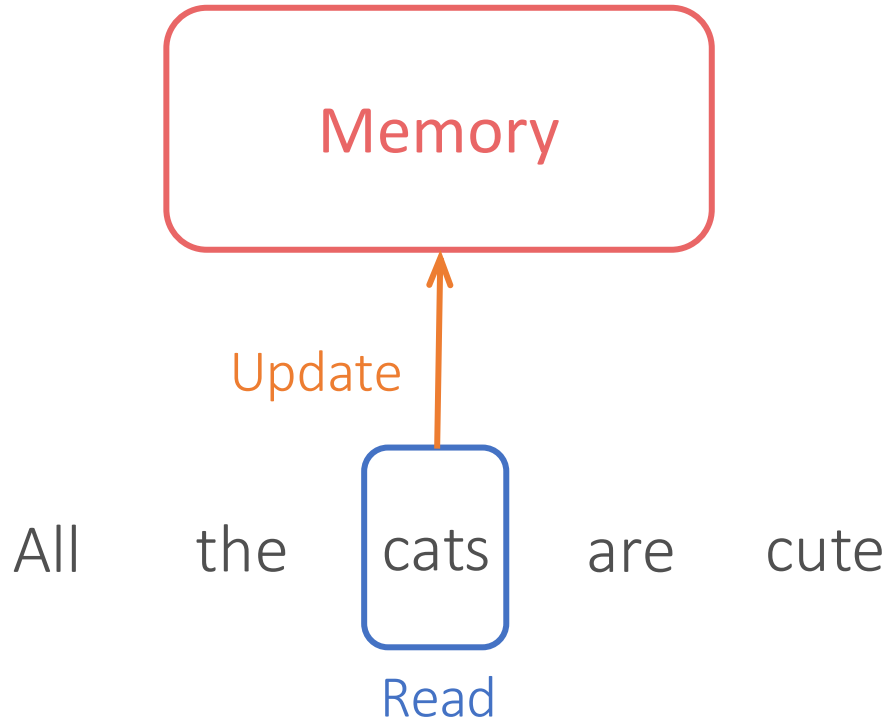
# Recurrent Neural Network (RNN)

- Read texts sequentially like a human with memory
  - Read → update memory



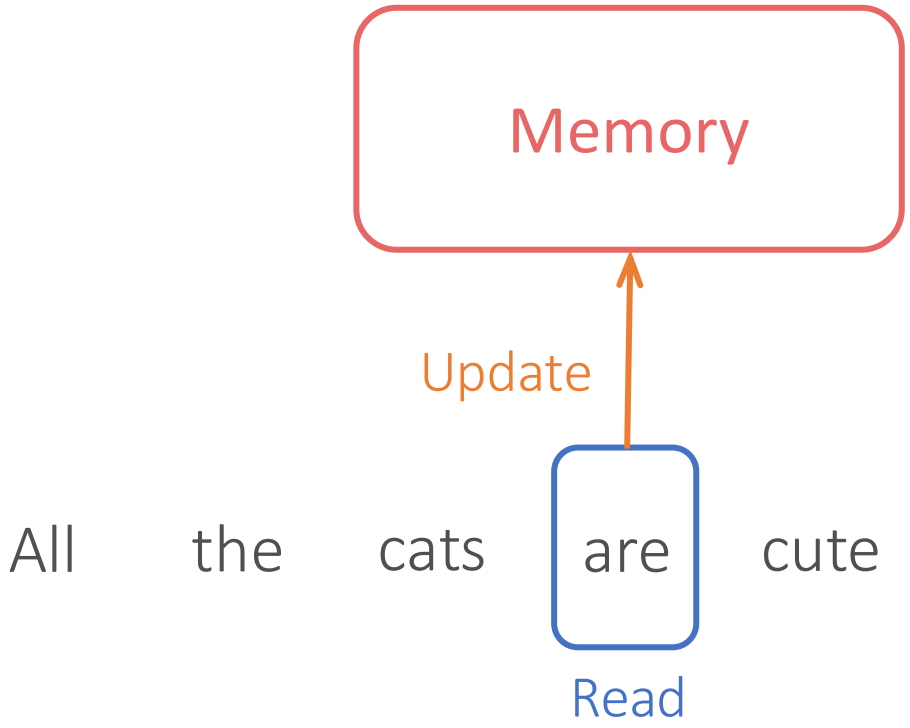
# Recurrent Neural Network (RNN)

- Read texts sequentially like a human with memory
  - Read → update memory



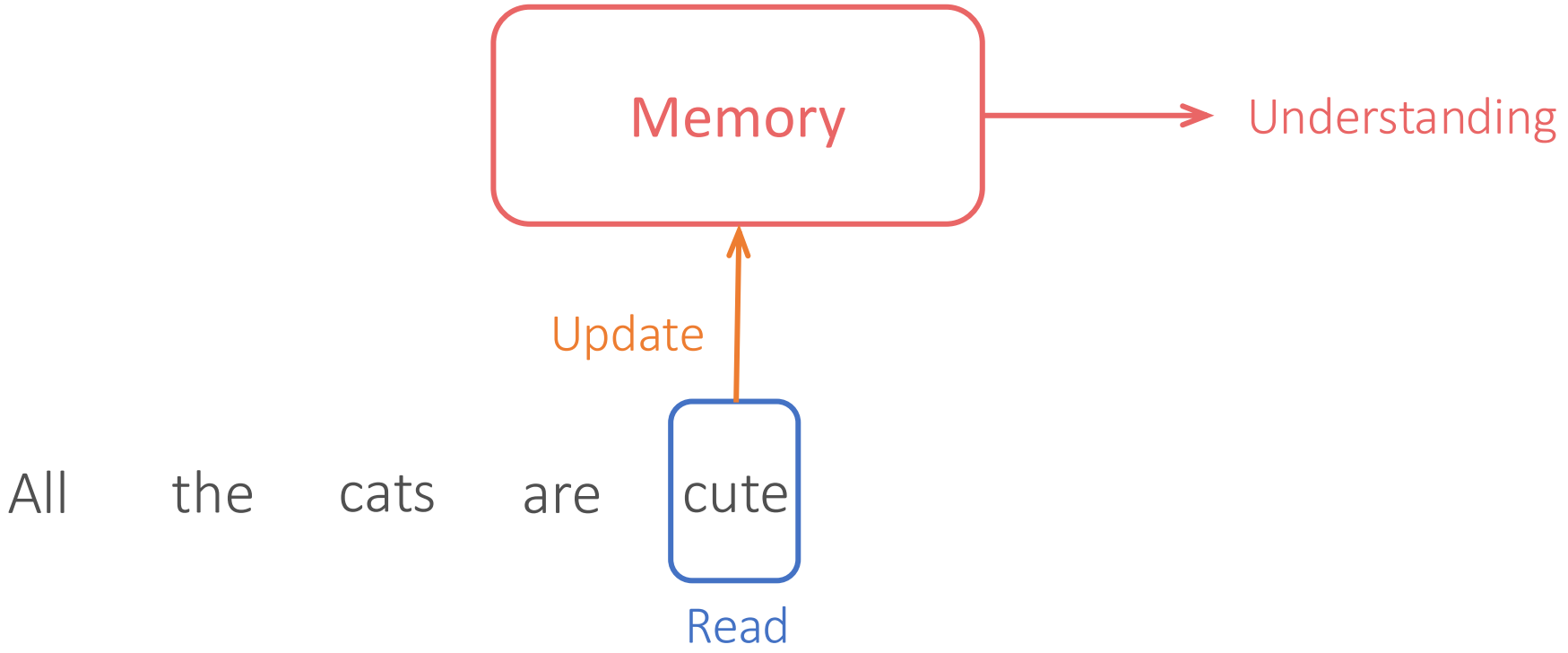
# Recurrent Neural Network (RNN)

- Read texts sequentially like a human with memory
  - Read → update memory



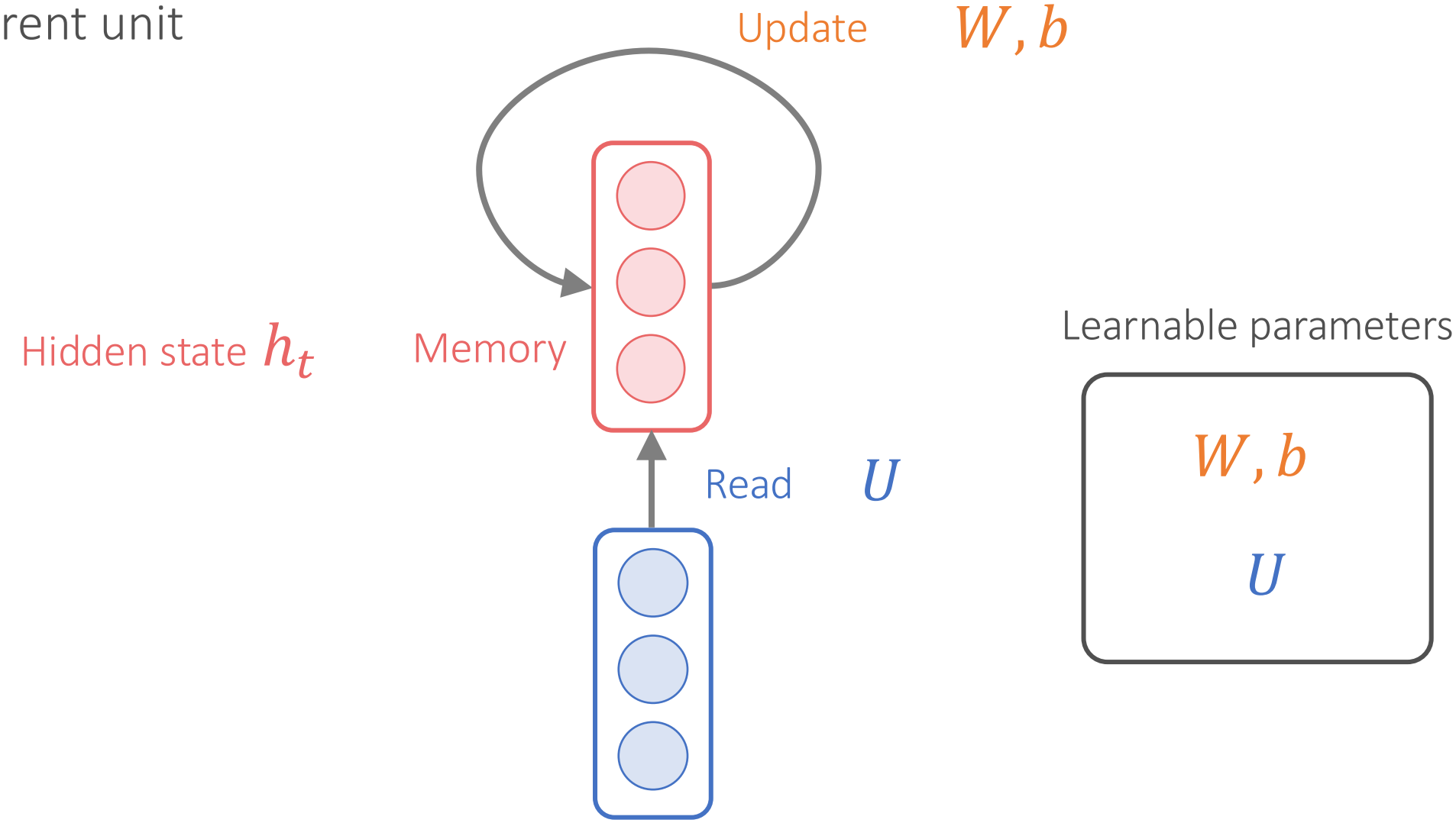
# Recurrent Neural Network (RNN)

- Read texts sequentially like a human with memory
  - Read → update memory

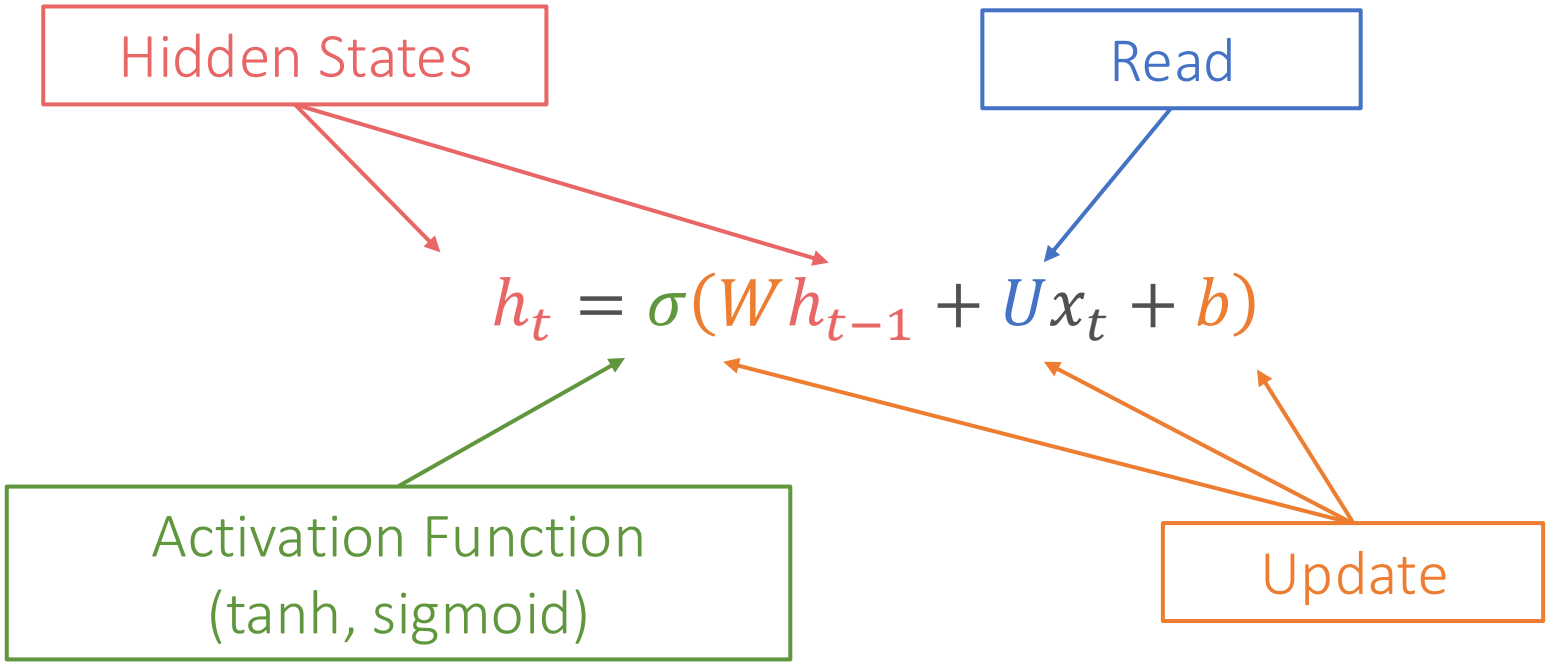


# Recurrent Neural Network (RNN)

- Recurrent unit

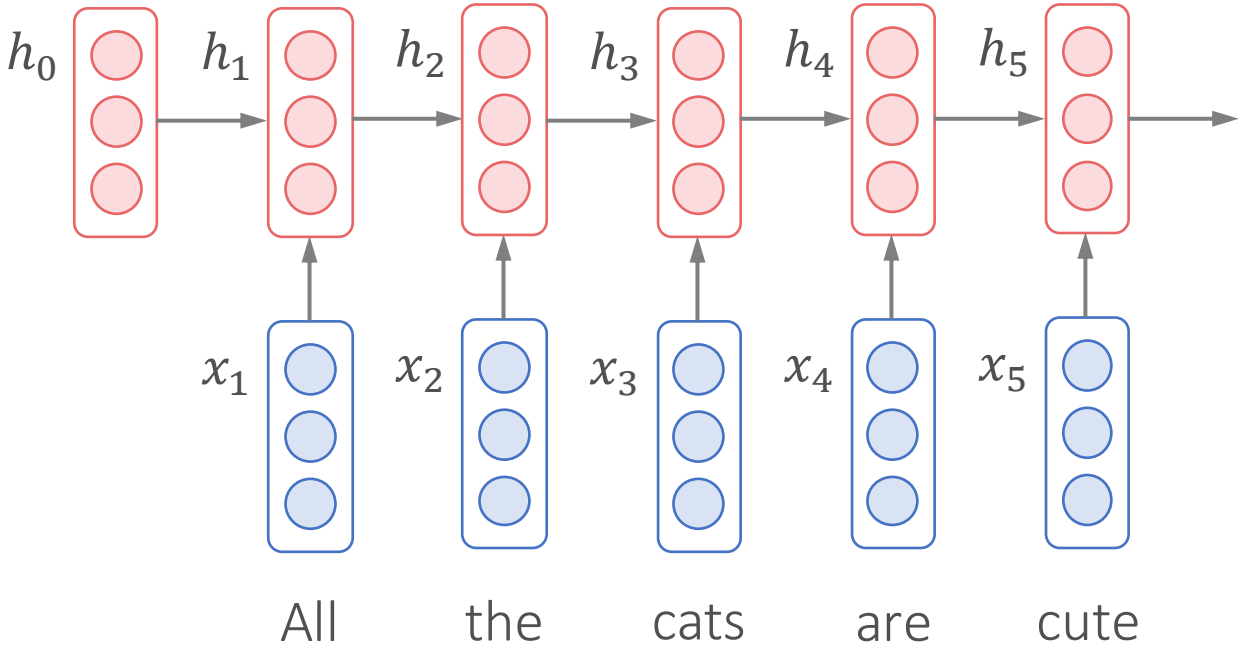


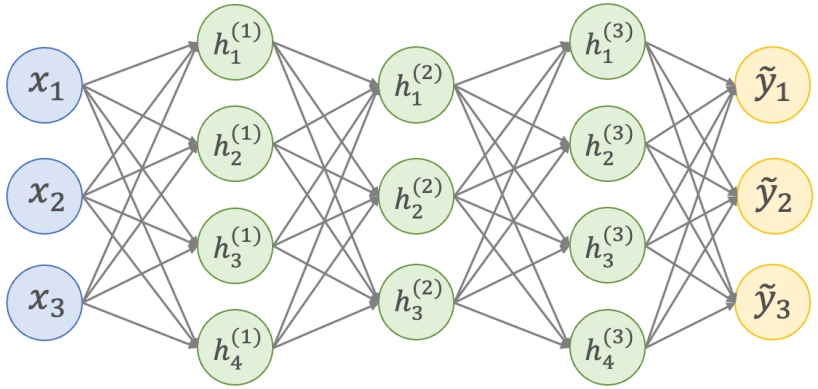
# Recurrent Neural Network (RNN)



# Recurrent Neural Network (RNN)

$$h_t = \sigma(W h_{t-1} + U x_t + b)$$



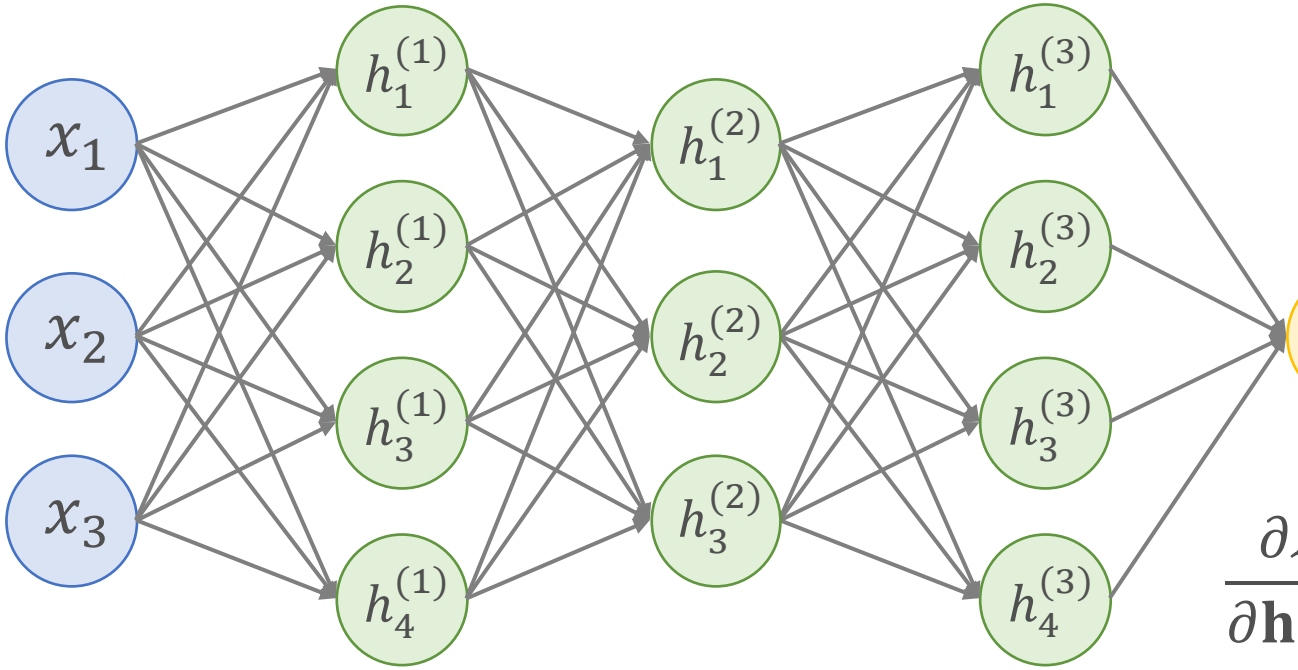




$$\mathcal{L}_{CE}(y, \tilde{y}) = -\sum_{c=0}^{C} y_c \log P(y = c|\, \mathbf{x})$$

# Recurrent Neural Network (RNN)

- Advantages
  - Can process any length input
  - Model size doesn't increase for longer input context
  - Computation for step t can (in theory) use information from many steps back
- Disadvantages
  - Recurrent computation is slow
  - In practice, difficult to access information from many steps back
  - Vanishing gradient

# Back-Propagation




$$\frac{\partial \mathcal{L}}{\partial \tilde{y}}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(3)}} = \frac{\partial \mathcal{L}}{\partial \tilde{y}} \cdot \frac{\partial \tilde{y}}{\partial \mathbf{h}^{(3)}}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{(1)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(1)}} \cdot \frac{\partial \mathbf{h}^{(1)}}{\partial \mathbf{W}^{(1)}}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(2)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(3)}} \cdot \frac{\partial \mathbf{h}^{(3)}}{\partial \mathbf{h}^{(2)}}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{b}^{(1)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(1)}} \cdot \frac{\partial \mathbf{h}^{(1)}}{\partial \mathbf{b}^{(1)}}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(1)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{h}^{(2)}} \cdot \frac{\partial \mathbf{h}^{(2)}}{\partial \mathbf{h}^{(1)}}$$

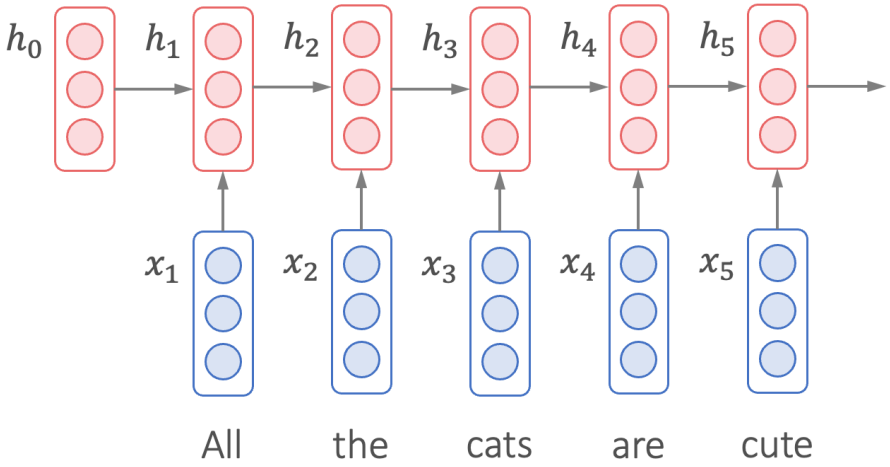
# Vanishing Gradient Problem

$$h_2 = \sigma(Wh_1 + Ux_2 + b)$$

$$h_3 = \sigma(Wh_2 + Ux_3 + b)$$

$$h_4 = \sigma(Wh_3 + Ux_4 + b)$$

$$h_5 = \sigma(Wh_4 + Ux_3 + b)$$




$$\frac{\partial \mathcal{L}}{\partial W} = \frac{\partial \mathcal{L}}{\partial h_5} \cdot \frac{\partial h_5}{\partial W} + \frac{\partial \mathcal{L}}{\partial h_5} \cdot \boxed{\frac{\partial h_5}{\partial h_4}} \cdot \frac{\partial h_4}{\partial W} + \frac{\partial \mathcal{L}}{\partial h_5} \cdot \boxed{\frac{\partial h_5}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3}} \cdot \frac{\partial h_3}{\partial W}$$

$$+ \frac{\partial \mathcal{L}}{\partial h_5} \cdot \boxed{\frac{\partial h_5}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2}} \cdot \frac{\partial h_2}{\partial W} + \frac{\partial \mathcal{L}}{\partial h_5} \cdot \boxed{\frac{\partial h_5}{\partial h_4} \cdot \frac{\partial h_4}{\partial h_3} \cdot \frac{\partial h_3}{\partial h_2} \cdot \frac{\partial h_2}{\partial h_1}} \cdot \frac{\partial h_1}{\partial W}$$

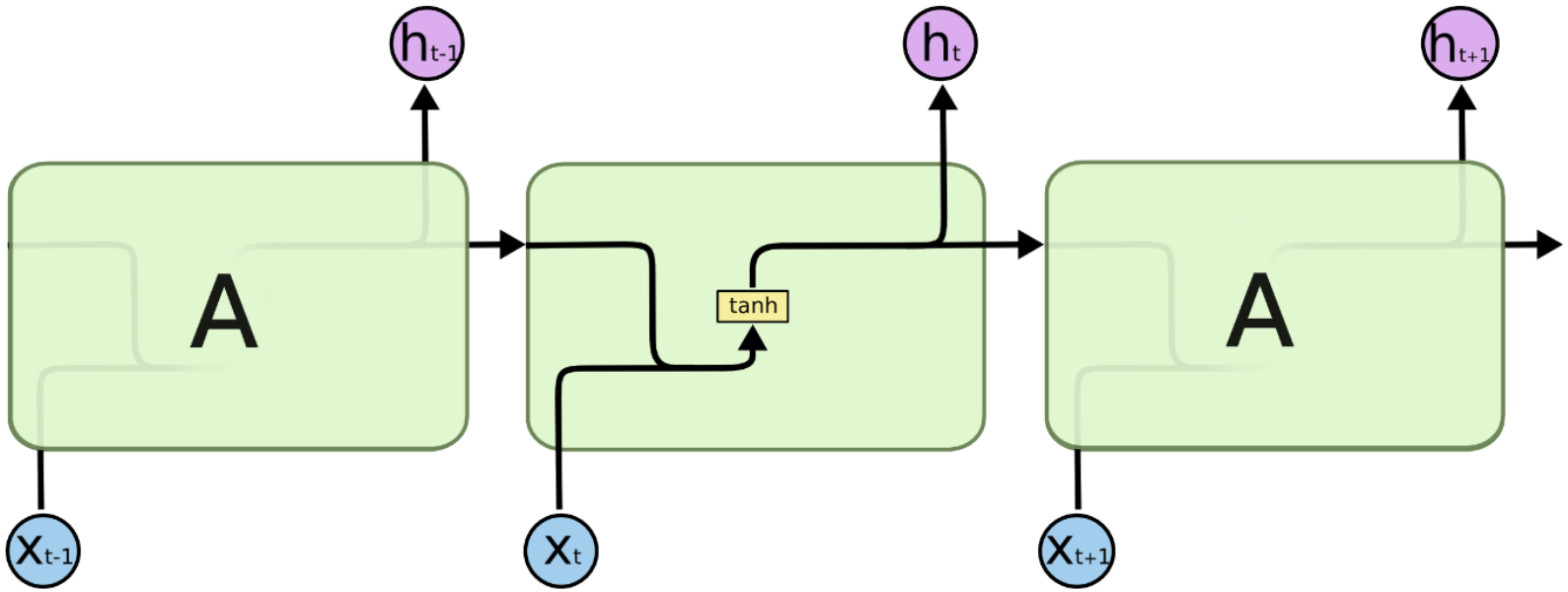
When these are small, the gradient signal gets smaller and smaller as it back-propagates further

Model weights are updated only with respect to short-term effect rather than long-term effect

# Long Short-Term Memory (LSTM)

- Short-term memory: hidden state $h_t$
- Long-term memory: cell state $c_t$
- Key idea
  - Turn multiplication into addition (partially reduce gradient vanishing)
  - use gates to control how much information to add/erase

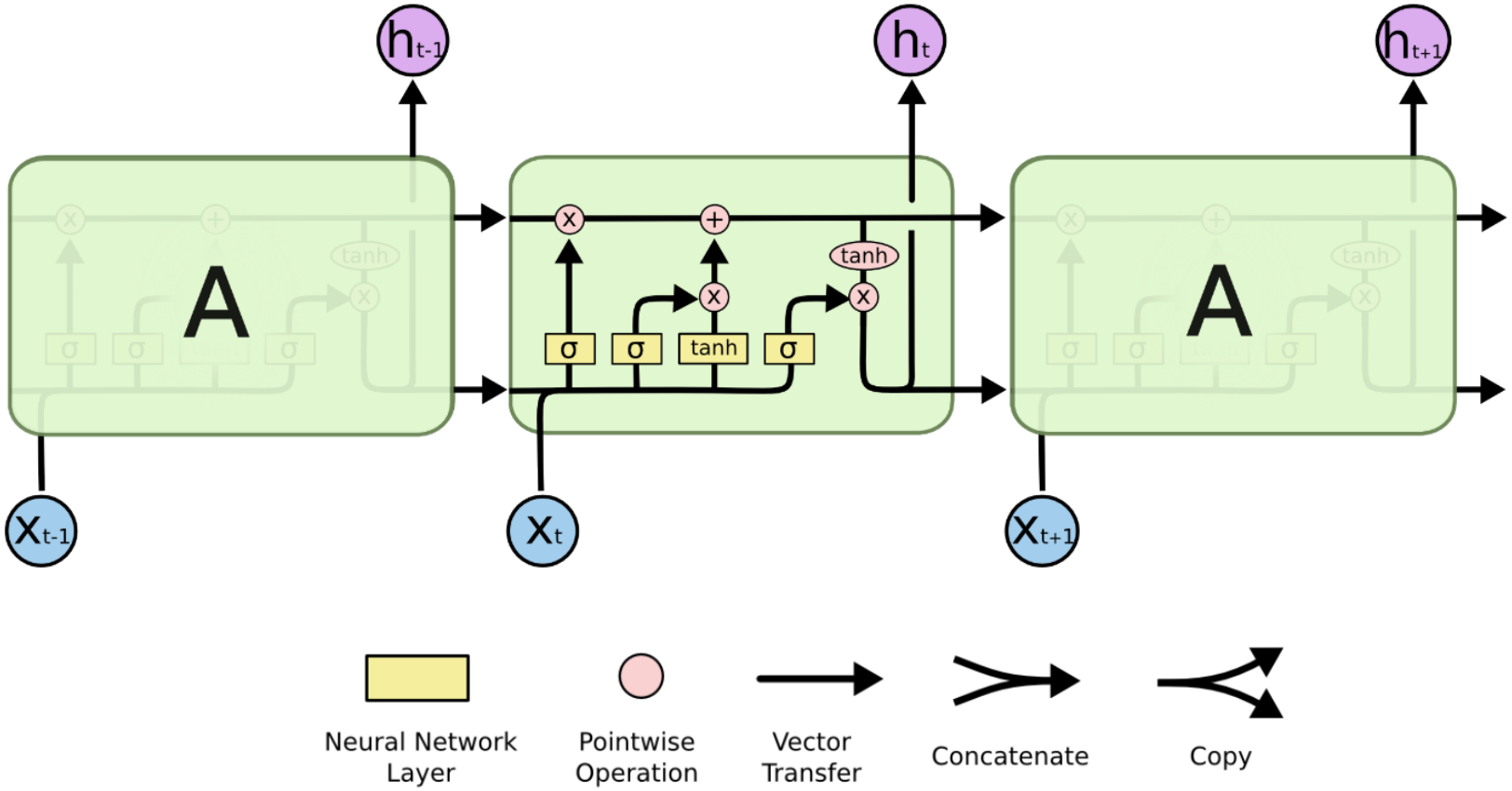
# Recurrent Neural Network (RNN)




$$h_t = \sigma(W h_{t-1} + U x_t + b)$$

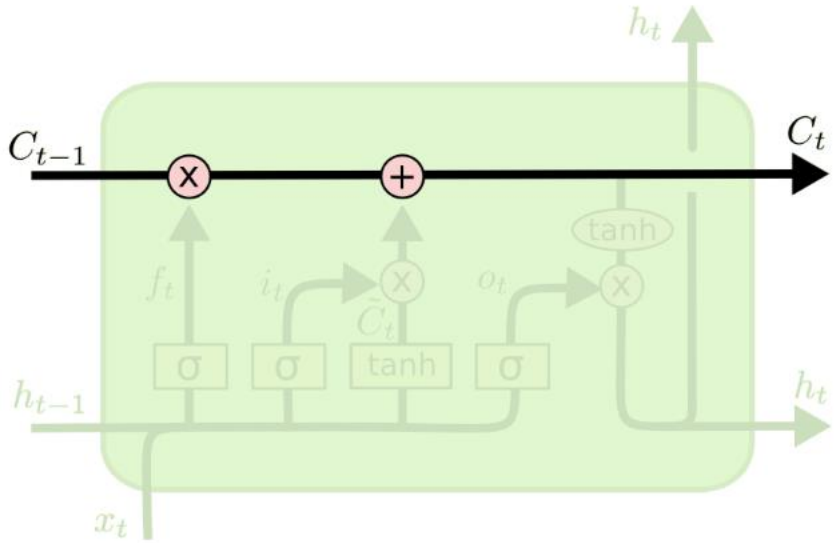
https://colah.github.io/posts/2015-08-Understanding-LSTMs/

# Long Short-Term Memory (LSTM)




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

# Long Short-Term Memory (LSTM)

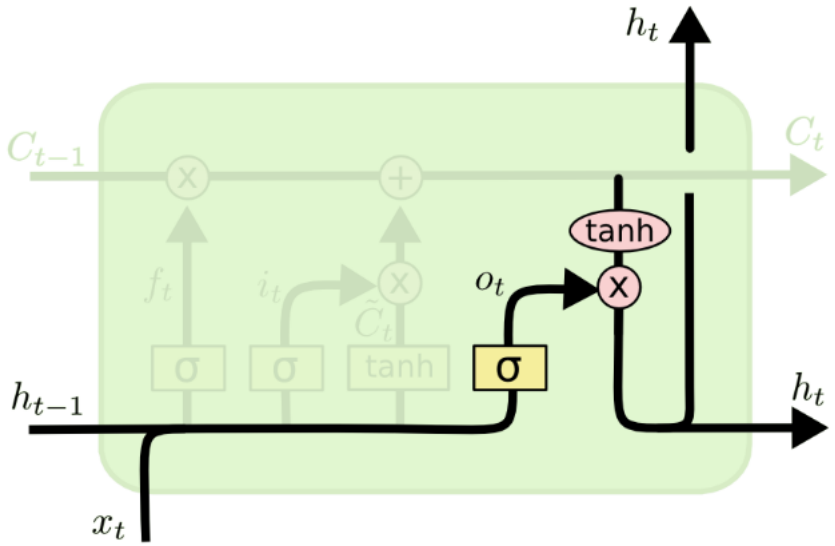



The cell state stores long-term information



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

# Long Short-Term Memory (LSTM)

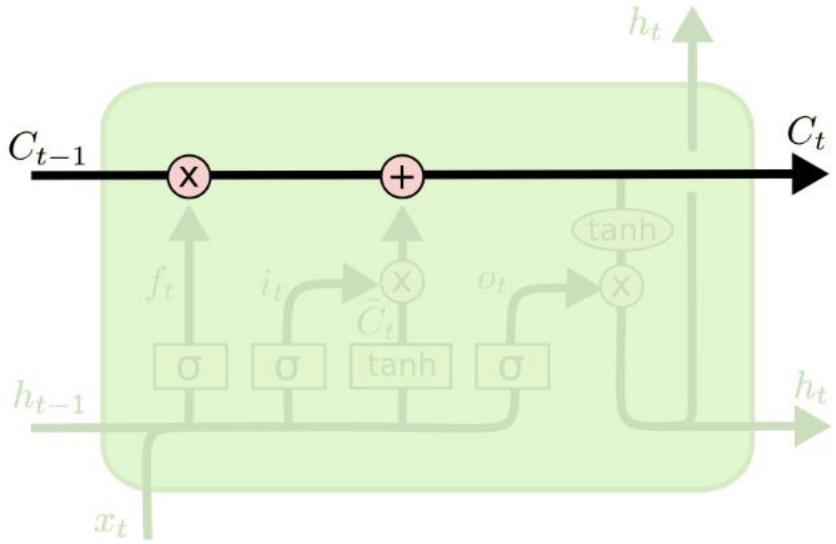



The hidden state stores short-term information



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

# Long Short-Term Memory (LSTM)



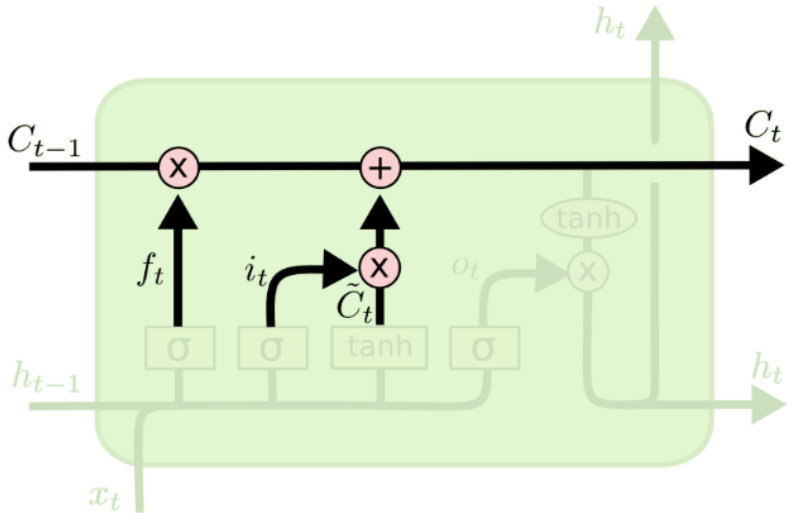

The cell state stores long-term information

Whenever reading a word, we will write/forget information to the cell state



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

# Long Short-Term Memory (LSTM)




What we should write

Update cell state $C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$

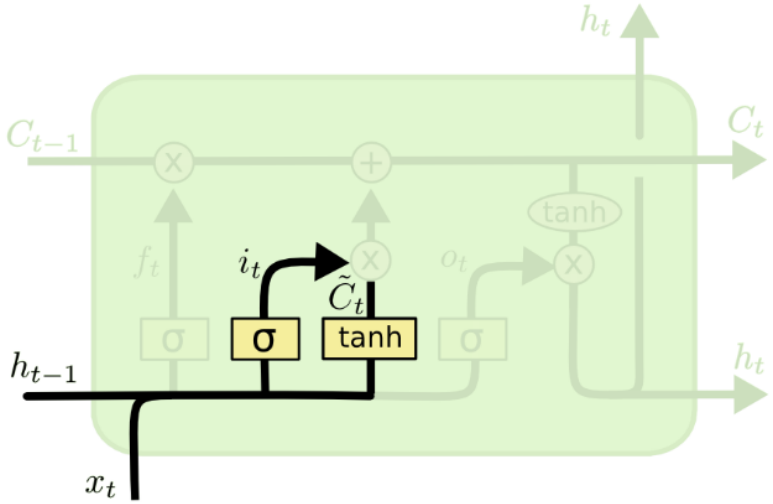
How much we should erase

How much we should write



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

# Long Short-Term Memory (LSTM)




New information $\tilde{C}_t = \tanh\left(W^{(C)}h_{t-1} + U^{(C)}x_t + b^{(C)}\right)$ What we should write

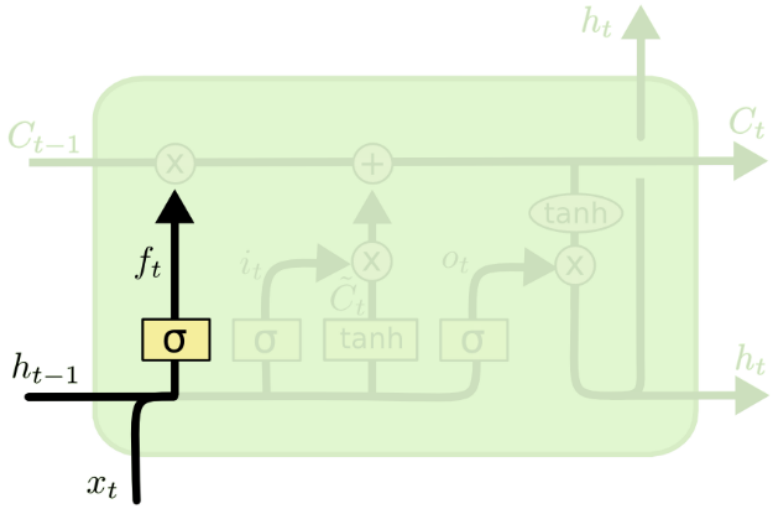
Input gate $i_t = \sigma\left(W^{(i)}h_{t-1} + U^{(i)}x_t + b^{(i)}\right)$ How much we should write

Sigmoid function: gate values are between 0 and 1



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

# Long Short-Term Memory (LSTM)



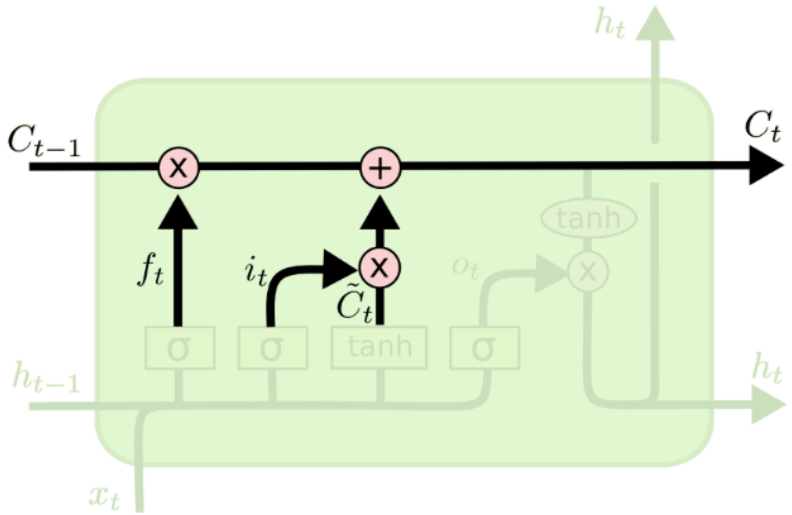

Forget gate $f_t = \sigma\left(W^{(f)} h_{t-1} + U^{(f)} x_t + b^{(f)}\right)$ How much we should erase

Sigmoid function: gate values are between 0 and 1



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

# Long Short-Term Memory (LSTM)




What we should write

Update cell state $C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$

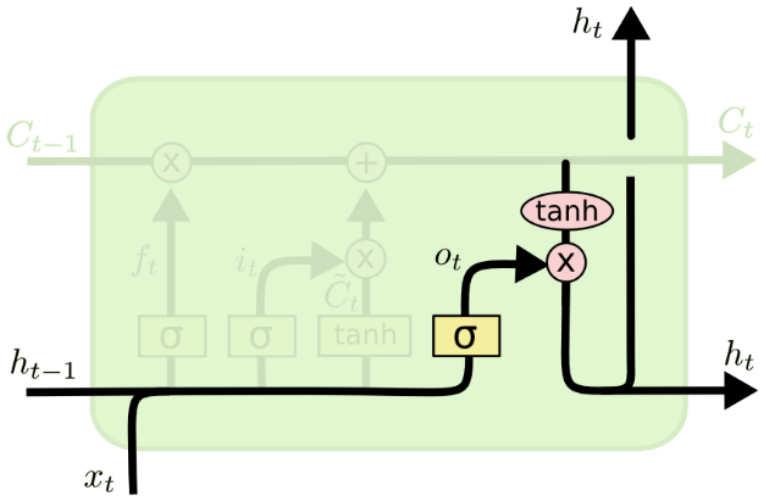
How much we should erase

How much we should write



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

# Long Short-Term Memory (LSTM)




$$o_t = \sigma\left(W^{(o)}h_{t-1} + U^{(o)}x_t + b^{(o)}\right)$$

Update hidden state

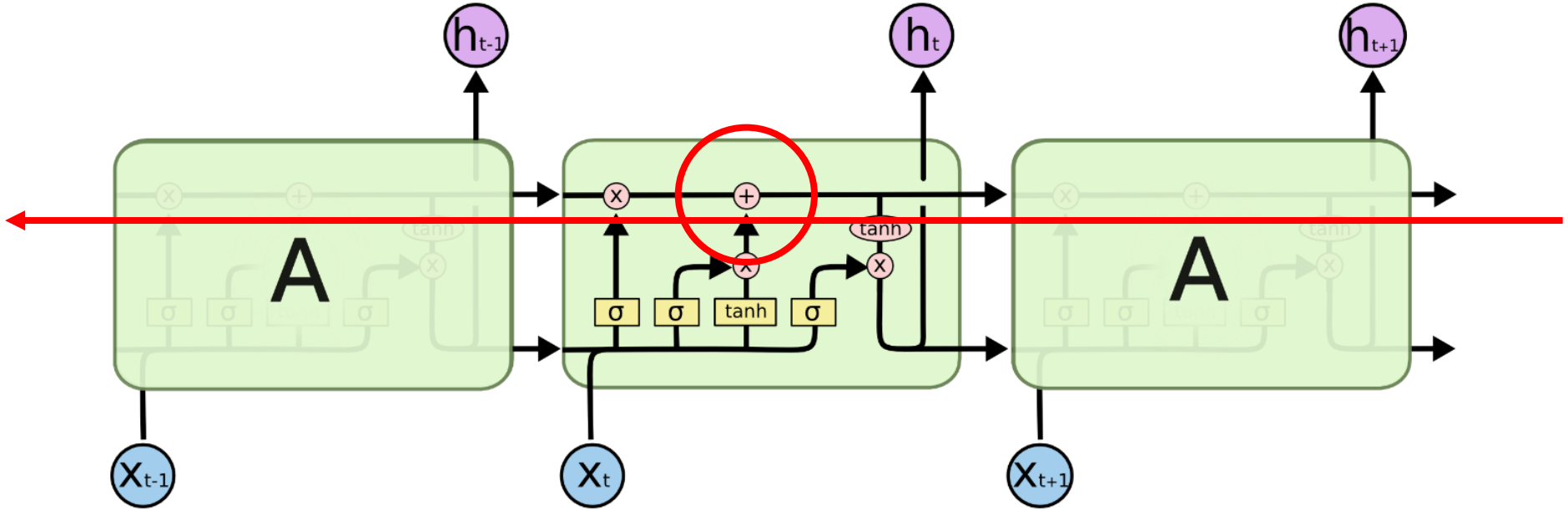
$$h_t = o_t * \tanh(C_t)$$



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

# Long Short-Term Memory (LSTM)

Uninterrupted gradient flow




The addition is the key

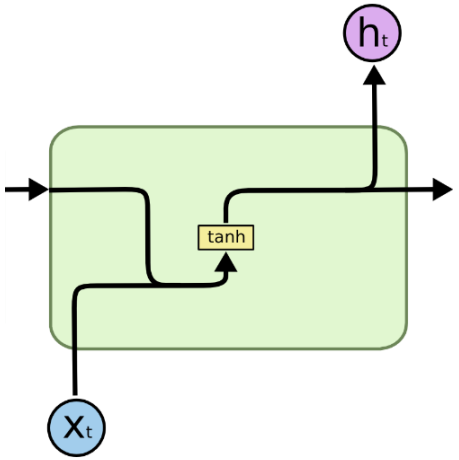
LSTM does not guarantee that there is no vanishing gradient
but it does provide an easier way to learn long-distance dependencies



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

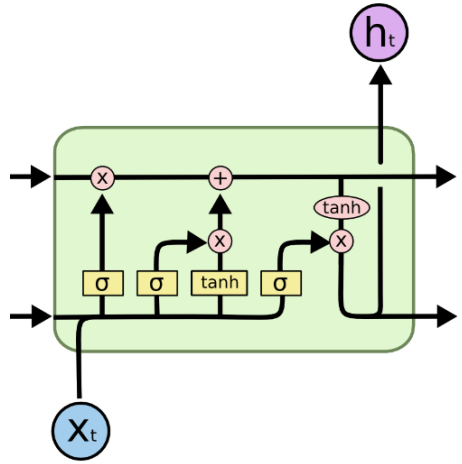
# RNN vs. LSTM

RNN




$$h_t = \sigma(W h_{t-1} + U x_t + b)$$

LSTM




$$i_t = \sigma\left(W^{(i)} h_{t-1} + U^{(i)} x_t + b^{(i)}\right)$$

$$f_t = \sigma\left(W^{(f)} h_{t-1} + U^{(f)} x_t + b^{(f)}\right)$$

$$o_t = \sigma\left(W^{(o)} h_{t-1} + U^{(o)} x_t + b^{(o)}\right)$$

$$\tilde{C}_t = \tanh\left(W^{(C)} h_{t-1} + U^{(C)} x_t + b^{(C)}\right)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

$$h_t = o_t * \tanh(C_t)$$

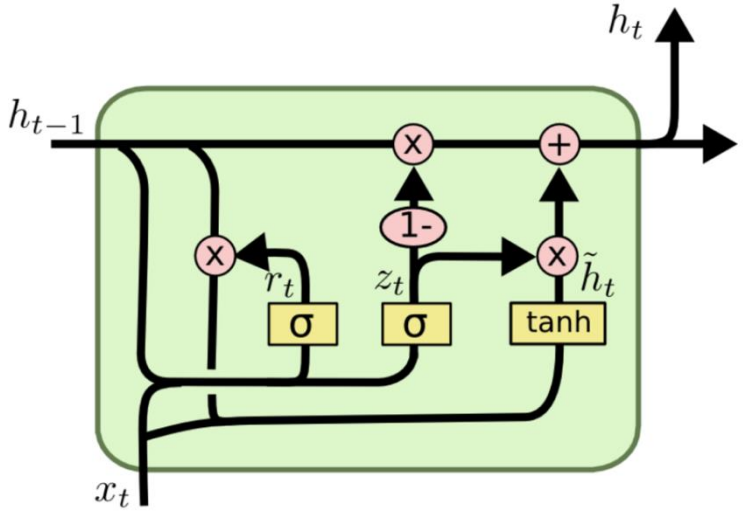


https://colah.github.io/posts/2015-08-Understanding-LSTMs/

# Gated Recurrent Units (GRU)

- Simplify 3 gates to 2 gates
  - Reset gate and update gate
- No explicit cell state
- More training-efficient

# Gated Recurrent Units (GRU)




Reset gate $r_t = \sigma\left(W^{(r)}h_{t-1} + U^{(r)}x_t + b^{(r)}\right)$

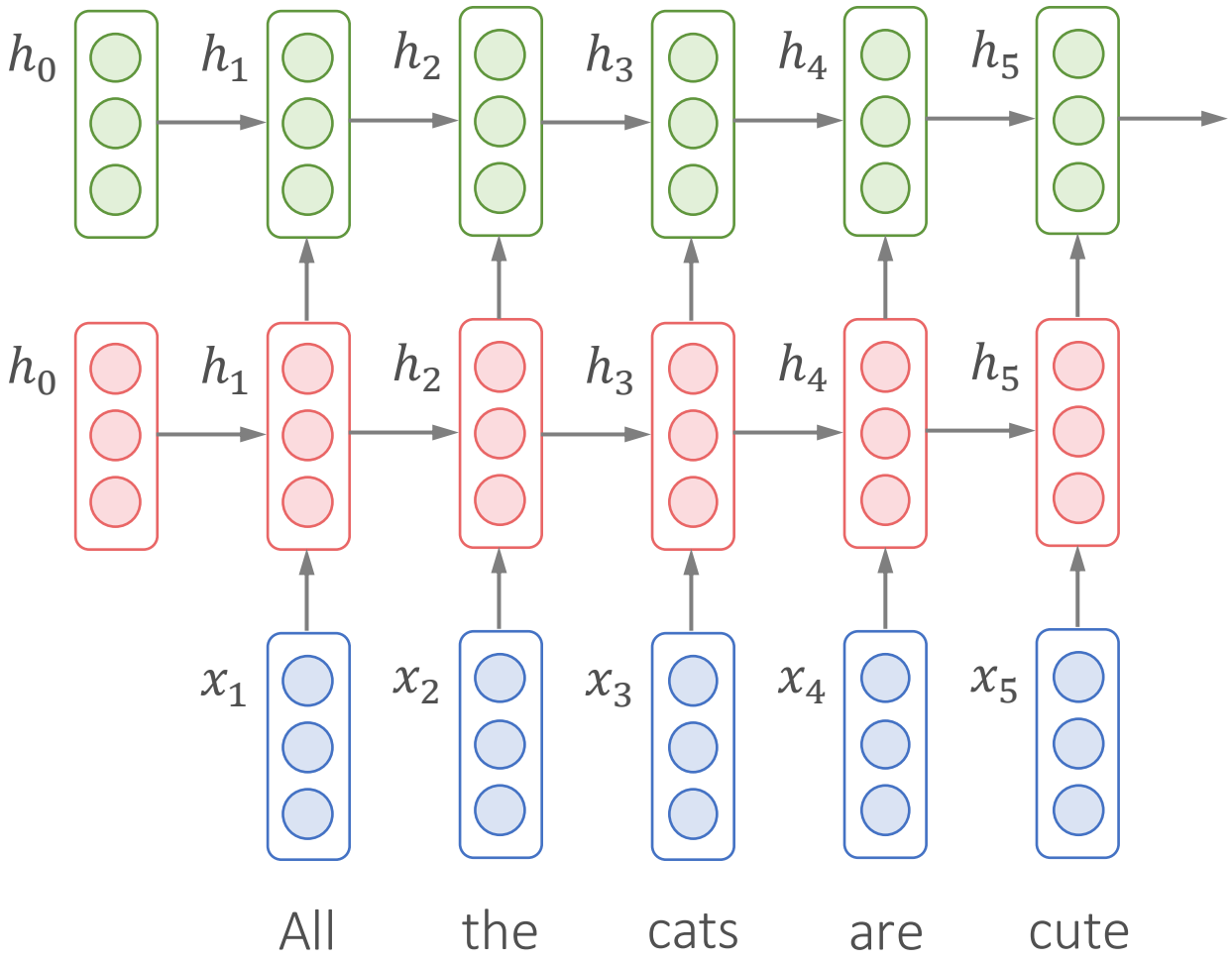
Update gate $z_t = \tanh\left(W^{(z)}h_{t-1} + U^{(z)}x_t + b^{(z)}\right)$

New hidden state $\tilde{h}_t = \tanh(W(r_t * h_{t-1}) + Ux_t + b)$

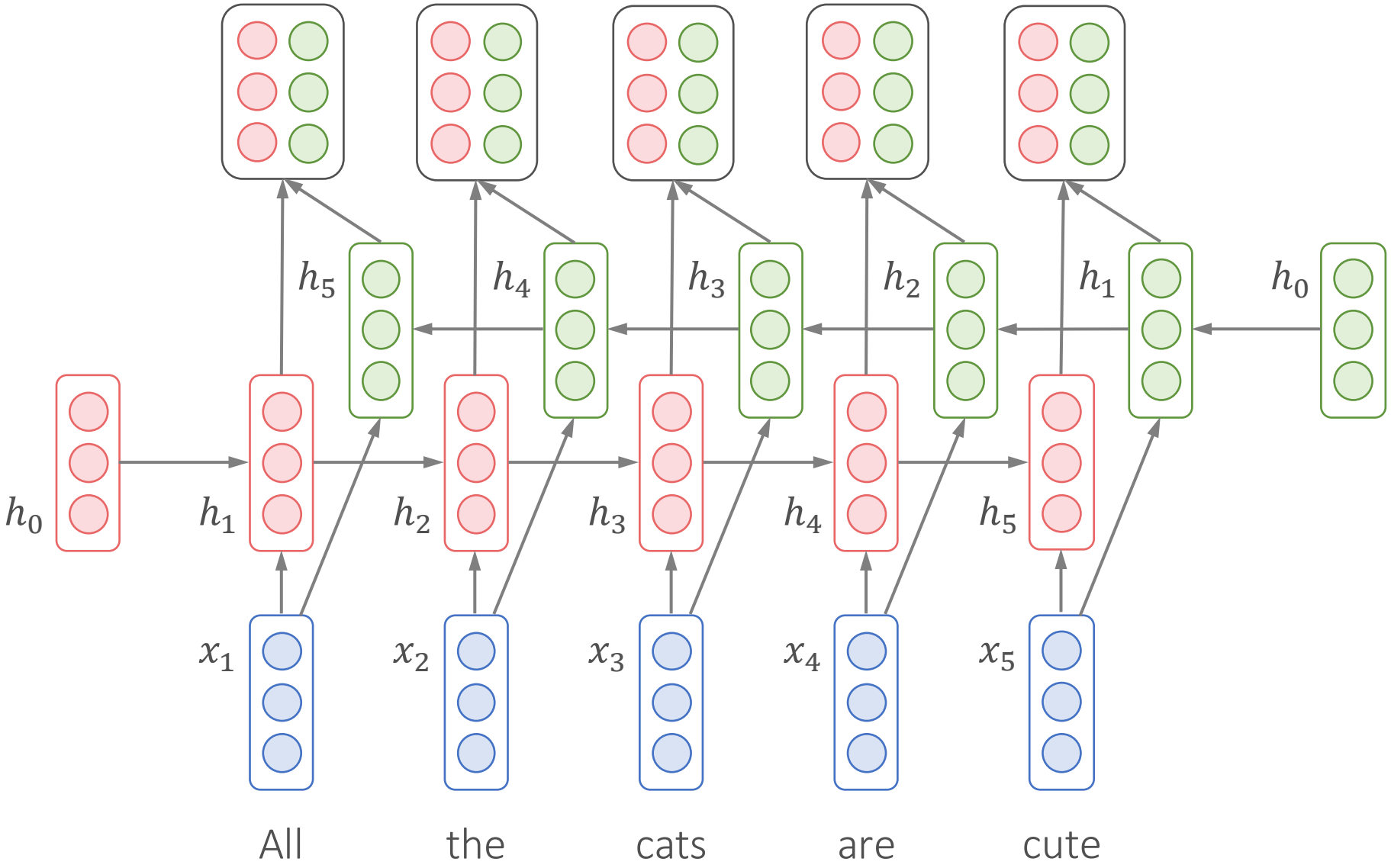
$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$

Merge input and forget gate

# Multi-Layer RNN (Stacked RNN)



# Bidirectional RNN



# Questions?