AI/ML Accelerator

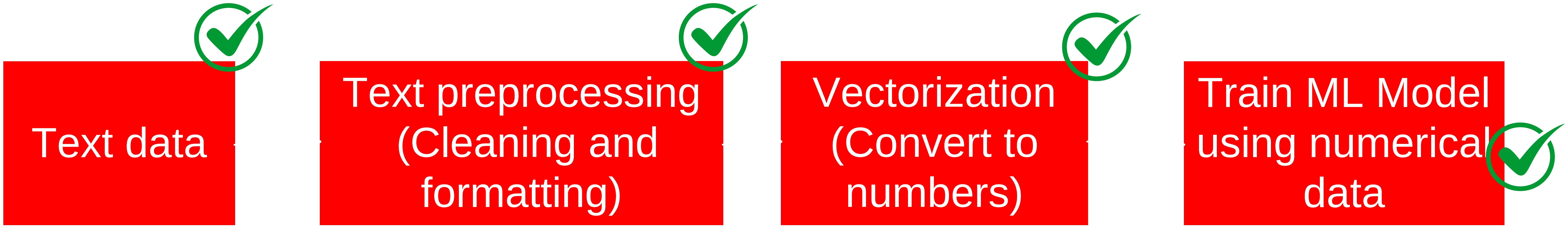
Natural Language Processing

Session 4 - Week 2

Learning Outcomes

- Practical knowledge of natural language processing (NLP) specific model training and applications
- Be comfortable talking with NLP Terminology
- Understand Transformer Architecture
- Basic ML Model

Recap of Week 1



Text data ✓

Text preprocessing (Cleaning and formatting) ✓

Stop words removal, Stemming, Lemmatization

Vectorization (Convert to numbers) ✓

Bag of Words

Train ML Model using numerical data ✓

K Nearest Neighbors (KNN), Neural Network, etc.

Problem set from Kaggle in office Hours

Use Case Summary

For this course, we’ll apply sentiment analysis to product reviews from a travel accessories retailer, Oceanwave15 Retails. By analyzing customer reviews, we aim to identify patterns in customer satisfaction, highlight popular products, and detect potential issues with product quality or usability. This analysis can guide strategic decisions in marketing, product improvement, and customer support.

Review	Sentiment
1. "This water bottle keeps drinks cold for hours and is perfect for long hikes."	Positive
2. "The suitcase is a bit heavy and not very easy to carry, especially when full."	Negative
3. "The seat cushion was comfortable but didn't offer much back support over long trips."	Neutral

NLP Sessions Overview - Week 2

Session 2

- Parts of Speech (POS) Tagging and Demo
- Grammar, Syntax, and Parsing Techniques
- Parsing Demo: Dependency Parsing with Spacy
- Introduction to Encoder-Decoder Models
- Hands-On: Building a Simple Encoder-Decoder

Office Hours 2

- Recap and Discussion on Parsing and Encoder-Decoder Models
- Review Exercises (Hugging Face, Kaggle)

Question

In NLP, tokenization is primarily used to:

a Remove stop words from a text

c Convert a text into smaller units like words or subwords.

b Identify and remove punctuation marks.

d Summarize the main content of a text

Question

What distance metric is most commonly used in K-Nearest Neighbors (KNN) for calculating the distance between points?

a Cosine similarity

b Euclidean distance

c Manhattan distance

d Hamming distance

POS Tagging, Grammar, Syntax, and Parsing

Technique	Details	Use Case	When to Use
Parts of Speech (POS) Tagging	Identifies and assigns parts of speech (noun, verb, adjective, etc.) to each word in a sentence. It helps in understanding sentence structure.	Text classification, information extraction, sentiment analysis, and speech recognition.	Use when needing to understand the role of each word in context.
Grammar	Refers to the rules governing the structure of sentences (syntax, morphology, etc.). Helps identify correct sentence structure.	Grammar checking, text correction, and text generation tasks.	Use when building systems that require language understanding and generation.
Syntax	Describes the arrangement of words and phrases to create well-formed sentences. It includes dependencies and phrase structures.	Language translation, dialogue systems, question answering, and summarization.	Use in syntactic analysis tasks like parsing or language generation.
Parsing	The process of analyzing a sentence's structure according to grammar rules (e.g., constituency parsing, dependency parsing).	Sentence parsing, dependency parsing for syntactic analysis, machine translation.	Use when you need to understand the syntactic structure of a sentence.

Usecase and Example

- **Sentence Parsing:** Helps break down the sentence structure for syntactic analysis.
- **Named Entity Recognition (NER):** Identifying names, organizations, locations, etc., which often depend on POS tags.
- **Machine Translation:** Mapping POS tags helps in translating sentence structures from one language to another.
- **Sentiment Analysis:** Adjectives, verbs, and nouns help determine the sentiment of a sentence (positive, negative, neutral).

Sentence: "Apple Inc. was founded by Steve Jobs in Cupertino in 1976."

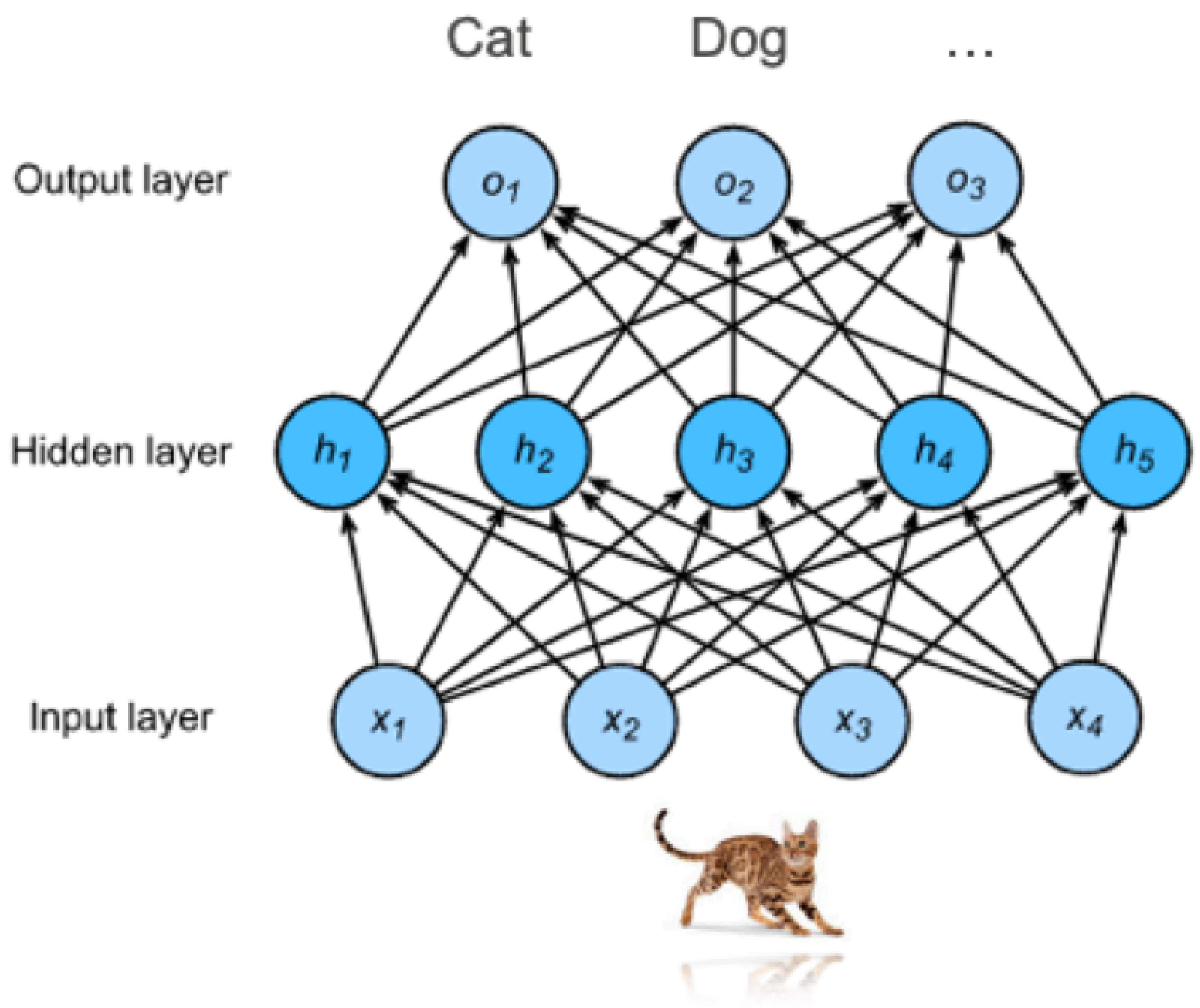
Word	POS Tag	Entity
Apple	Noun (NNP)	Organization
Inc.	Noun (NNP)	Organization
was	Verb (VBD)	-
founded	Verb (VBD)	-
by	Preposition (IN)	-
Steve	Noun (NNP)	Person
Jobs	Noun (NNP)	Person
in	Preposition (IN)	-
Cupertino	Noun (NNP)	Location
in	Preposition (IN)	-
1976	Noun (CD)	Date

Quick Demo -1

- POS Tagging

Neural Network

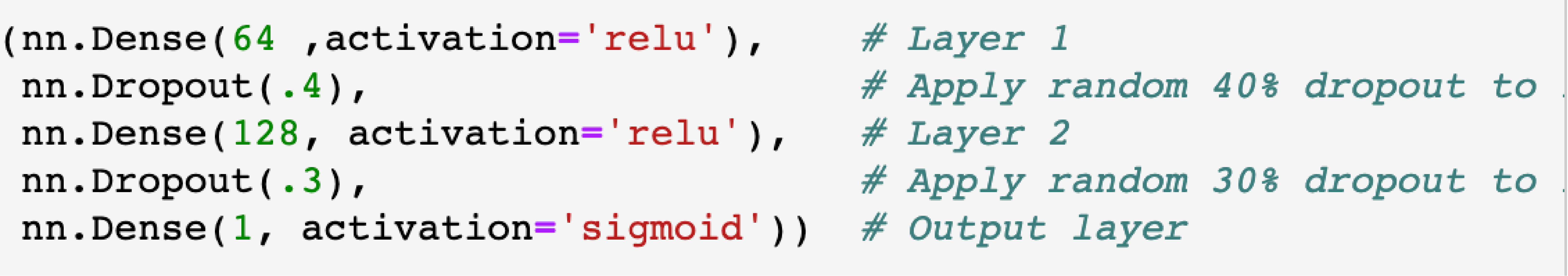
- Automatically **extract useful features** from input data.
- In recent years, deep learning has achieved **state-of-the art results** in many machine learning areas.
- Three pillars of deep learning:
    - Data
    - Compute
    - Algorithms



Neural Network

Build and Train Neural Network

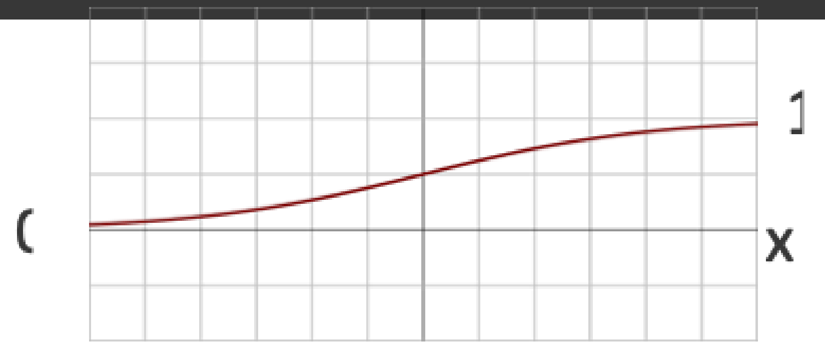
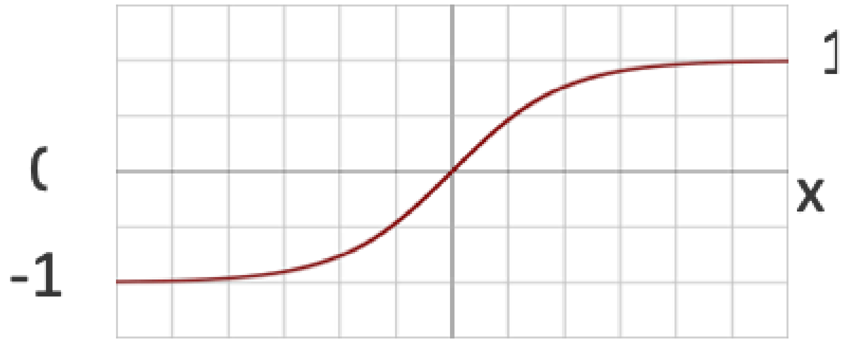
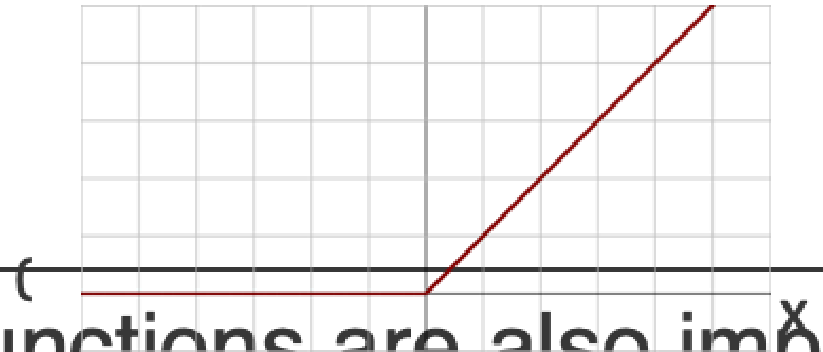
How to build and use these ML models?
Can it be this simple?



```
(nn.Dense(64 ,activation='relu'),     # Layer 1
 nn.Dropout(.4),                      # Apply random 40% dropout to
 nn.Dense(128, activation='relu'),    # Layer 2
 nn.Dropout(.3),                      # Apply random 30% dropout to
 nn.Dense(1, activation='sigmoid'))   # Output layer
```

What is Activation, Dense ?

Activation Function

Name	Plot	Function	Description
Logistic (sigmoid)		$f(x) = \dfrac{1}{1+e^{-x}}$	The most common activation function. Squashes input to (0,1).
Hyperbolic tangent (tanh)		$f(x) = \dfrac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$	Squashes input to (-1, 1).
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0, & \text{if } x < 0 \\ x, & \text{if } x \geq 0 \end{cases}$	Popular activation function. Anything less than 0, results in zero activation.

Derivatives of these functions are also important (gradient descent).

Output Activations / Cost Functions

Problem	Decription	Name	Cost Functions
Binary classification	• Output probability for each class, in (0,1) • Logistic regression of output of last layer	Sigmoid	Cross Entropy for Logistic
Multi-class classification	• Output probability for each class, in (0,1) • Sum of outputs to be 1 (probability distribution) • Training drives target class values up, others down	Softmax	Cross Entropy for SoftMax
Regression		Linear/ ReLU	Mean Squared Error

Training Neural Networks

- Cost function is selected according to problem: **Binary, Multi-class Classification or Regression**.
- Update network weights by applying **the gradient descent method** and **backpropagation**. More details

- Weight update formula:

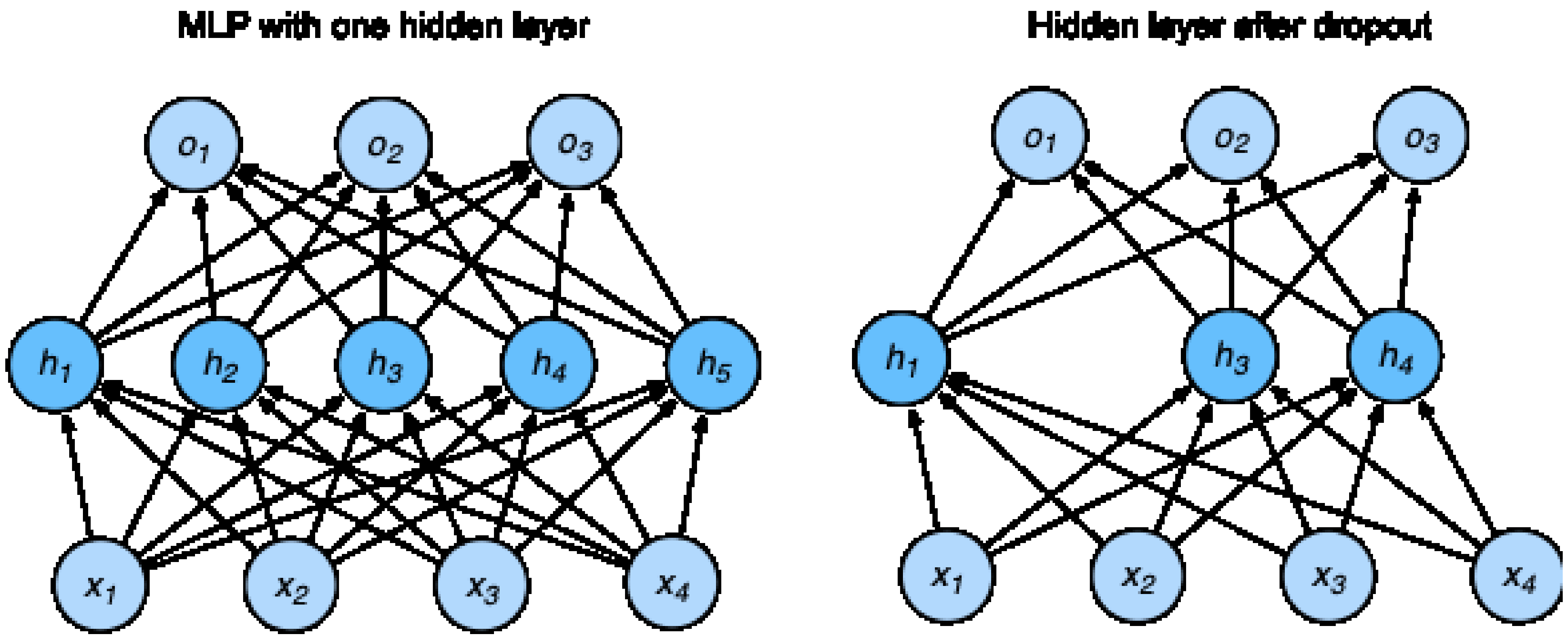
$$w_{new} = w_{old} - learning\_rate * \boxed{\frac{\partial C}{\partial w}}$$

C: Cost

Gradient with respect to w

DropOut

- Regularization technique to **prevent overfitting**.
- Randomly removes some nodes with a fixed probability during the training.

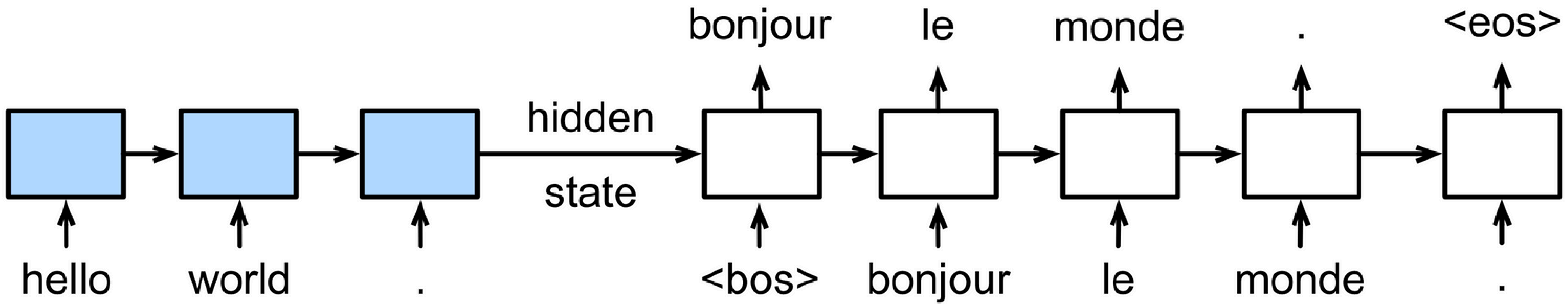


MLP with one hidden layer

Hidden layer after dropout

Neural Network for Sequential Data ?

Text data has sequential information of words.

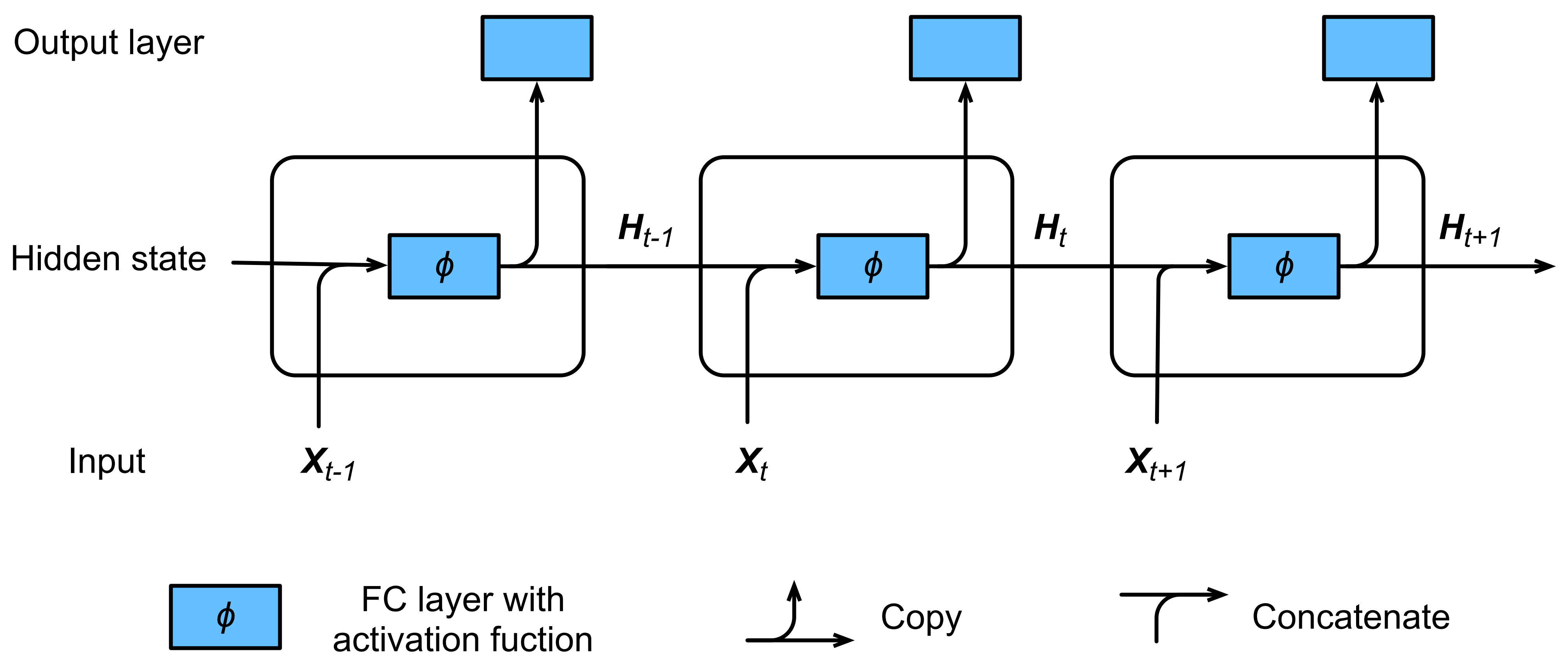


Example: Language translation

Other sequential inputs (time series, music notes, video, etc.)

Recurrent Neural Networks (RNN)

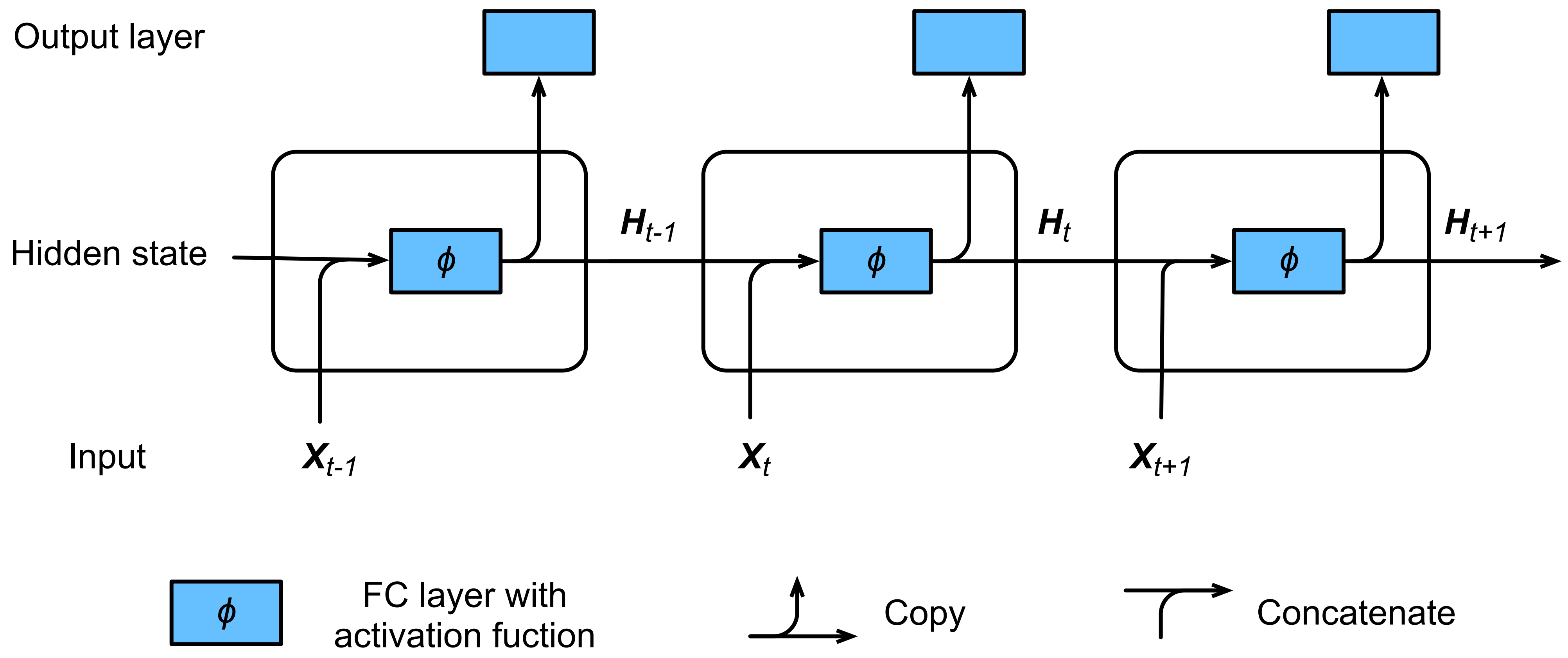
RNN uses an **internal state** to preserve the sequential information between input elements.



X: Input
H: Hidden state
t: Timestep

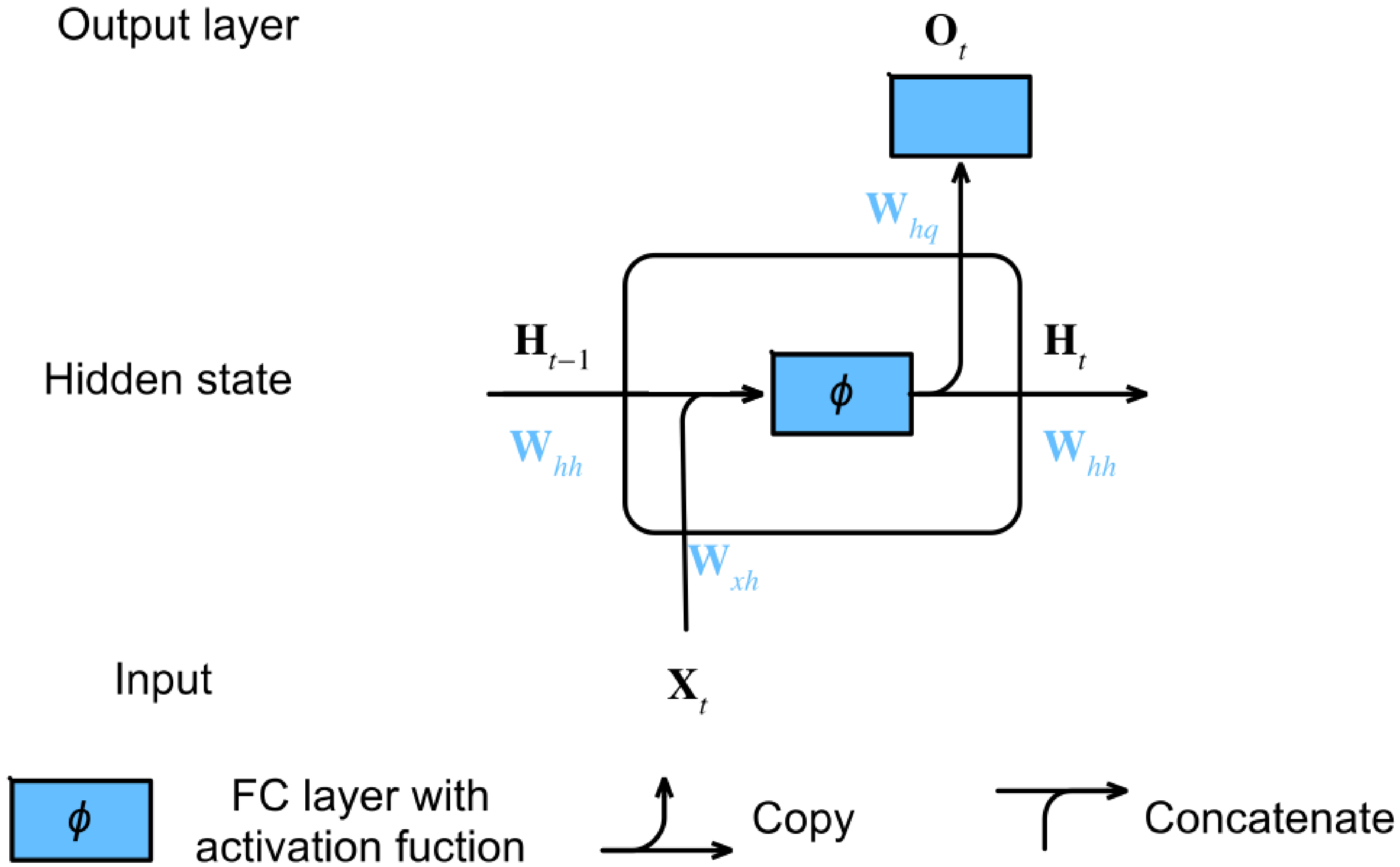
Recurrent Neural Networks (RNN)

RNN uses an **internal state** to preserve the sequential information between input elements.

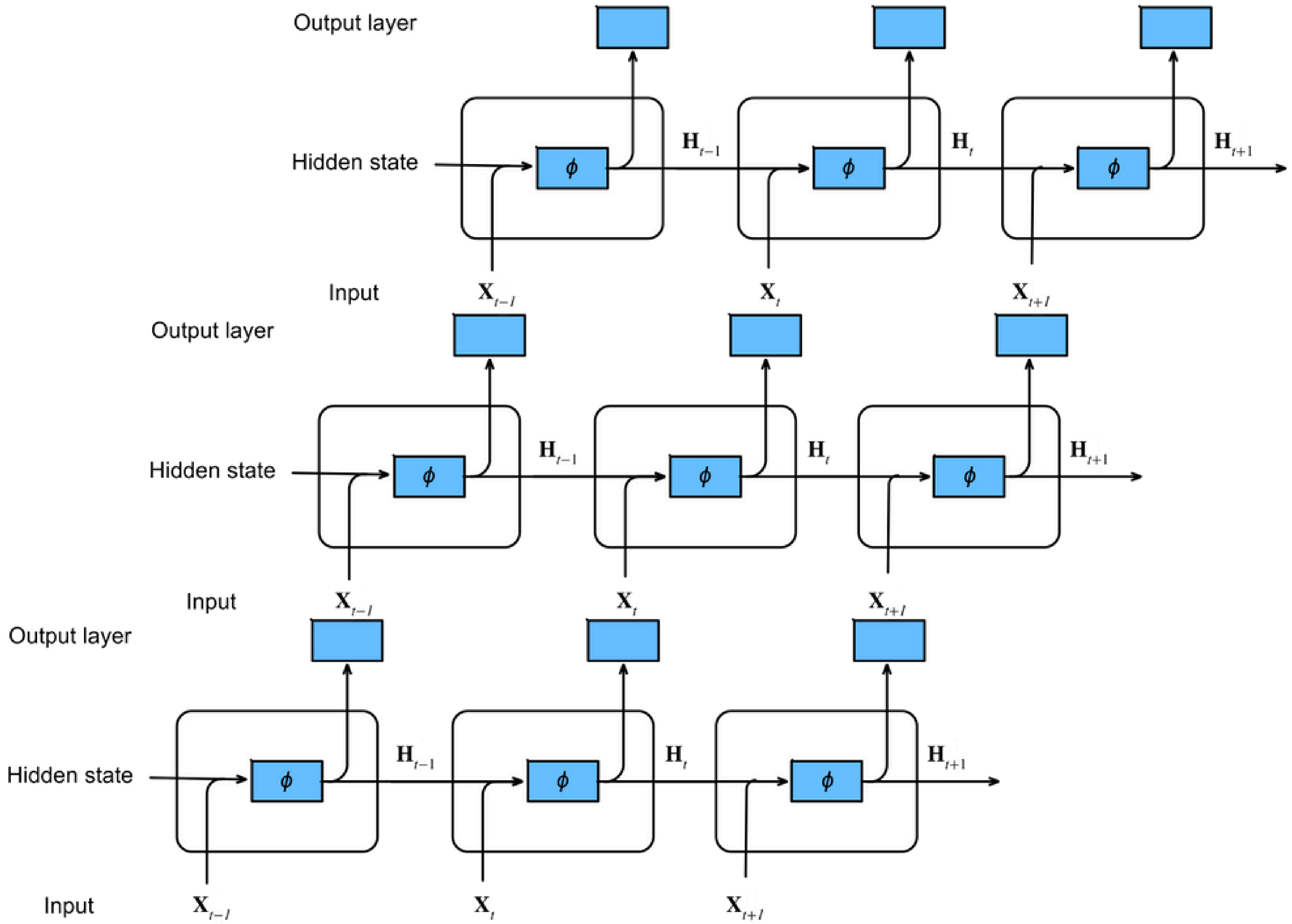


X: Input
H: Hidden state
t: Timestep

RNN



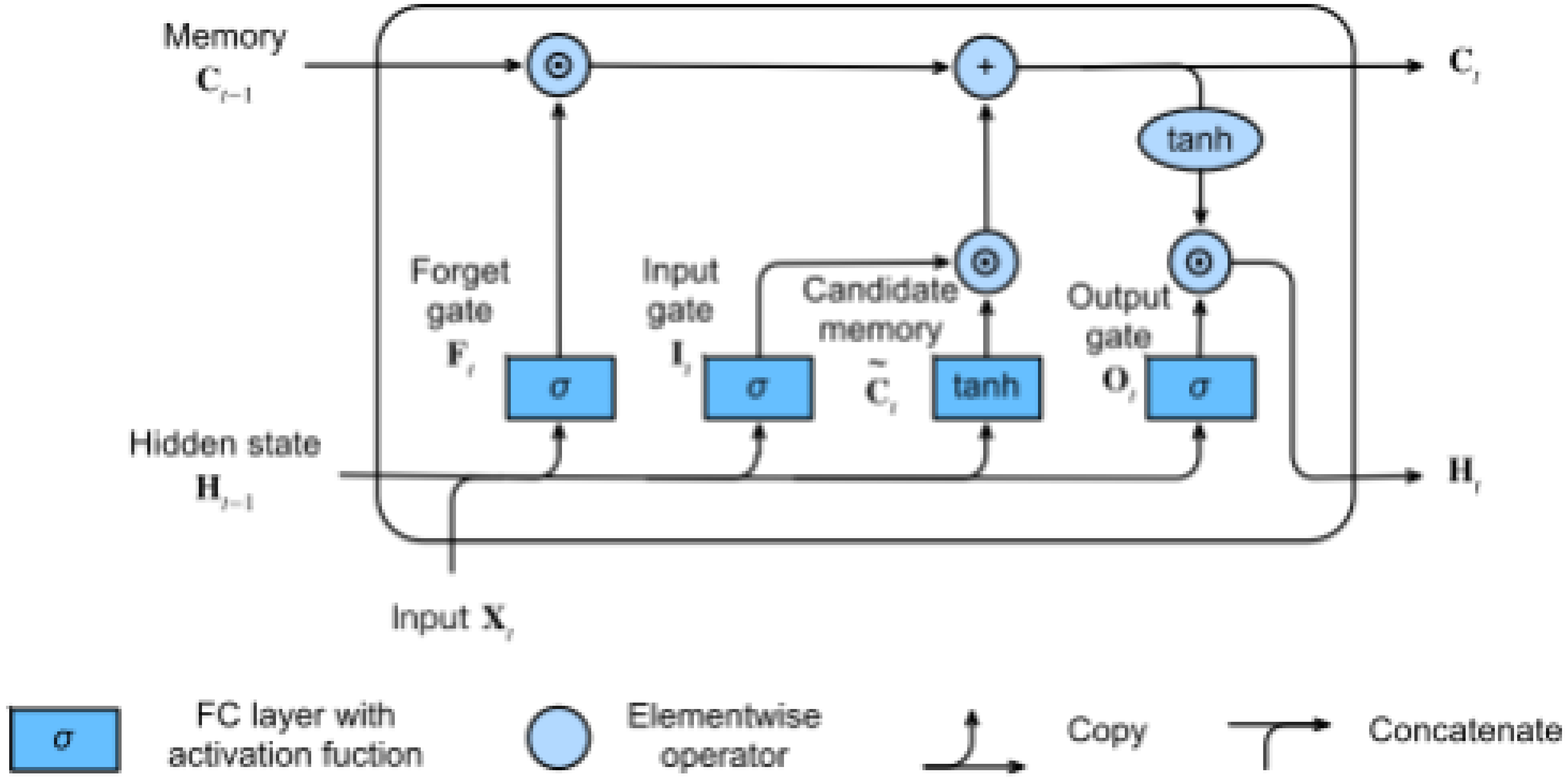
Stacked RNN



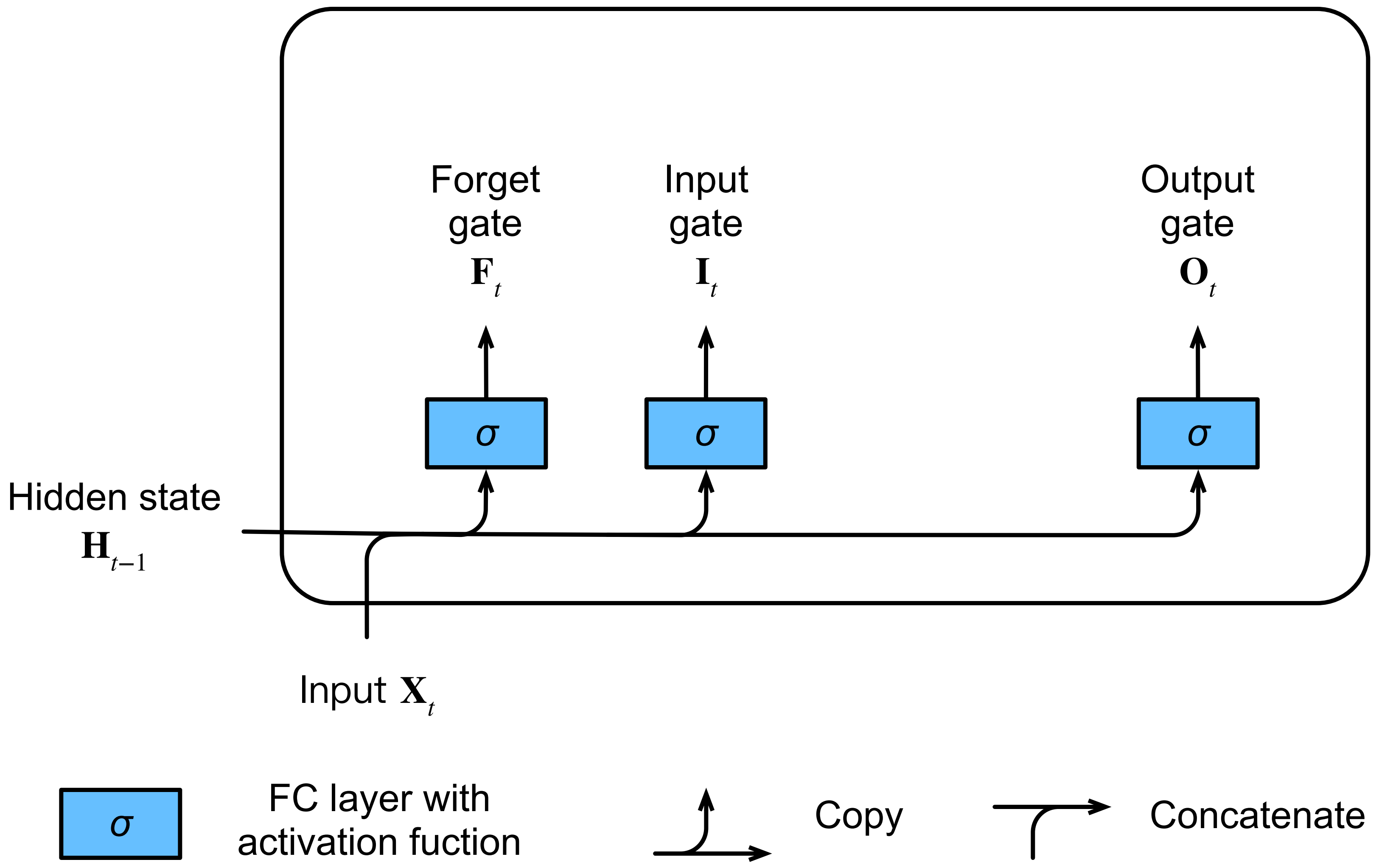
Long Short-term Memory Networks (LSTM)

Long Short-Term Memory (LSTM) networks are special RNNs, with different gates and memory cells:

- **Gates**:
    - Input gate
    - Forget gate
    - Output gate
- **Memory cells:**
    - Candidate memory cell
    - Memory cell
- **Hidden state**



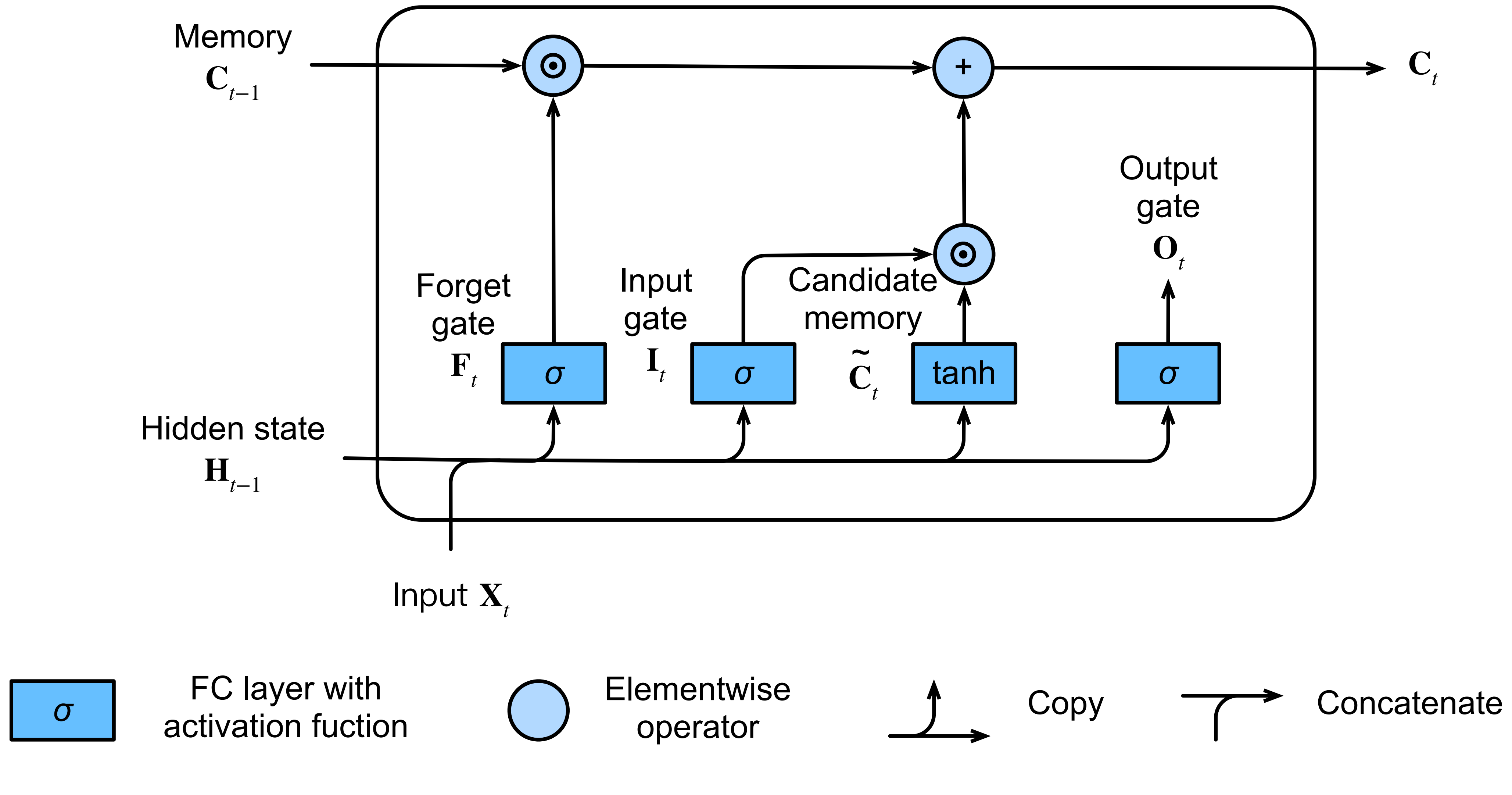
Input, Forget and Output Gates


Forget gate $\mathbf{F}_t$
Input gate $\mathbf{I}_t$
Output gate $\mathbf{O}_t$
σ
σ
σ
Hidden state $\mathbf{H}_{t-1}$
Input $\mathbf{X}_t$
σ
FC layer with activation fuction
Copy
Concatenate

- Input is (number of examples: n, number of inputs: d)
- Hidden state of last timestep (number of hidden states: *h*).

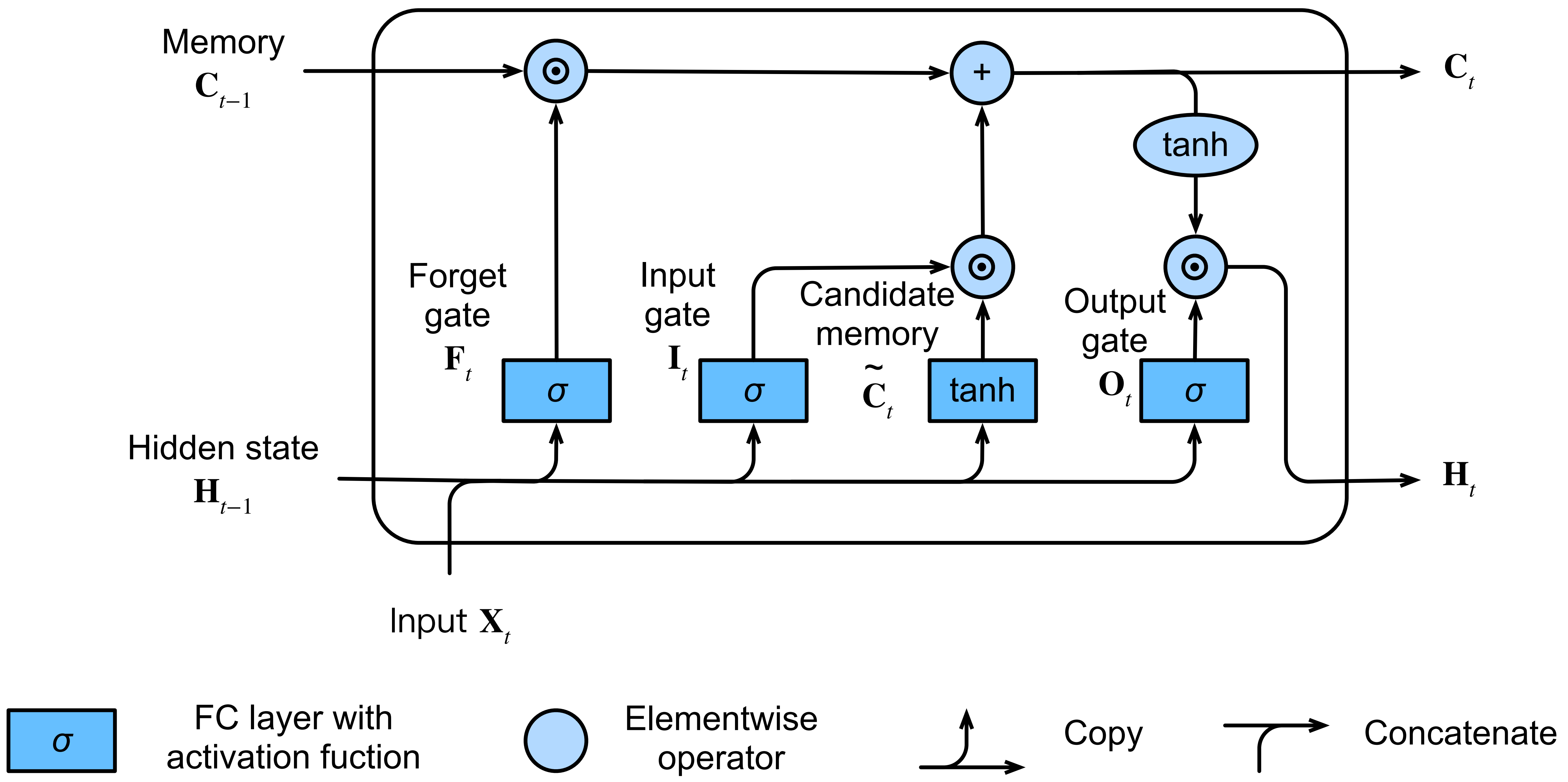
- All gates use **sigmoid activation function**.

Memory Cell



: How much of the old memory will stay and : How much new data will be added

LSTM Architecture



Memory $\mathbf{C}_{t-1}$

Hidden state $\mathbf{H}_{t-1}$

Forget gate $\mathbf{F}_t$

Input gate $\mathbf{I}_t$

Candidate memory $\tilde{\mathbf{C}}_t$

Output gate $\mathbf{O}_t$

σ

σ

tanh

σ

tanh

+

$\mathbf{C}_t$

$\mathbf{H}_t$

Input $\mathbf{X}_t$

σ FC layer with activation fuction

Elementwise operator

Copy

Concatenate

LSTM vs RNN

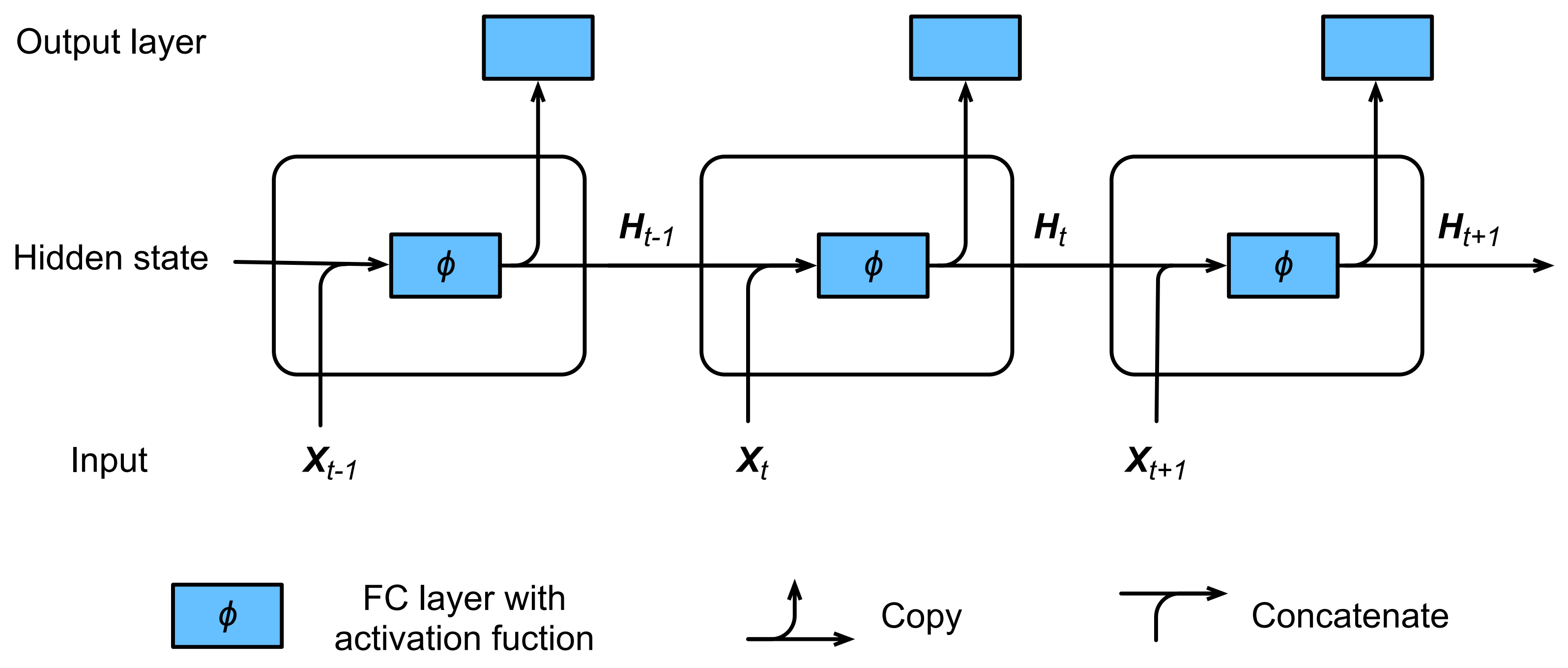
Feature	RNN	LSTM
Architecture	Simple looping units	Complex units with memory cells and gates
Memory Capability	Limited short-term memory	Long-term memory through cell state
Gates	None	Forget, Input, and Output gates
Vanishing Gradient Problem	Prone to vanishing gradient	Mitigates vanishing gradient through gated structure
Ideal Use Cases	Short sequences, limited dependency tasks	Long sequences, tasks with long-term dependencie

Demo 2 - LSTM and RNN

Transformer

Transformer - Why We needed

- RNNs are naturally sequential -> Cannot be trained in parallel



- RNNs (or LSTMs) still need “attention” mechanism to deal with long range dependencies between states
  - If attention gives us access to any state, can we simply utilize the **attention** and ignore the RNN?

Attention Matters

- **Attention** is a mechanism that forces the model to focus on specific parts of the input sequence.
- Can process sequential data parallelly.

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

Aidan N. Gomez* †
University of Toronto
aidan@cs.toronto.edu

Łukasz Kaiser*
Google Brain
lukaszkaiser@google.com

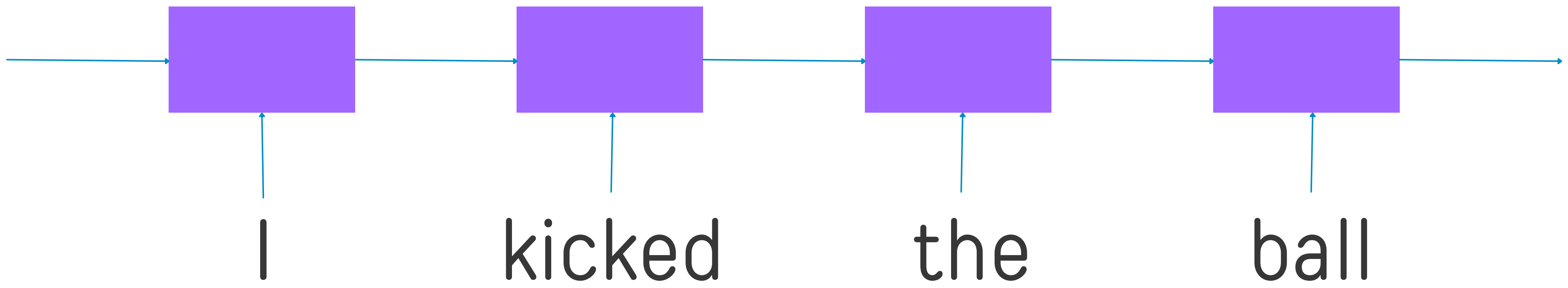
Illia Polosukhin* ‡
illia.polosukhin@gmail.com

Abstract

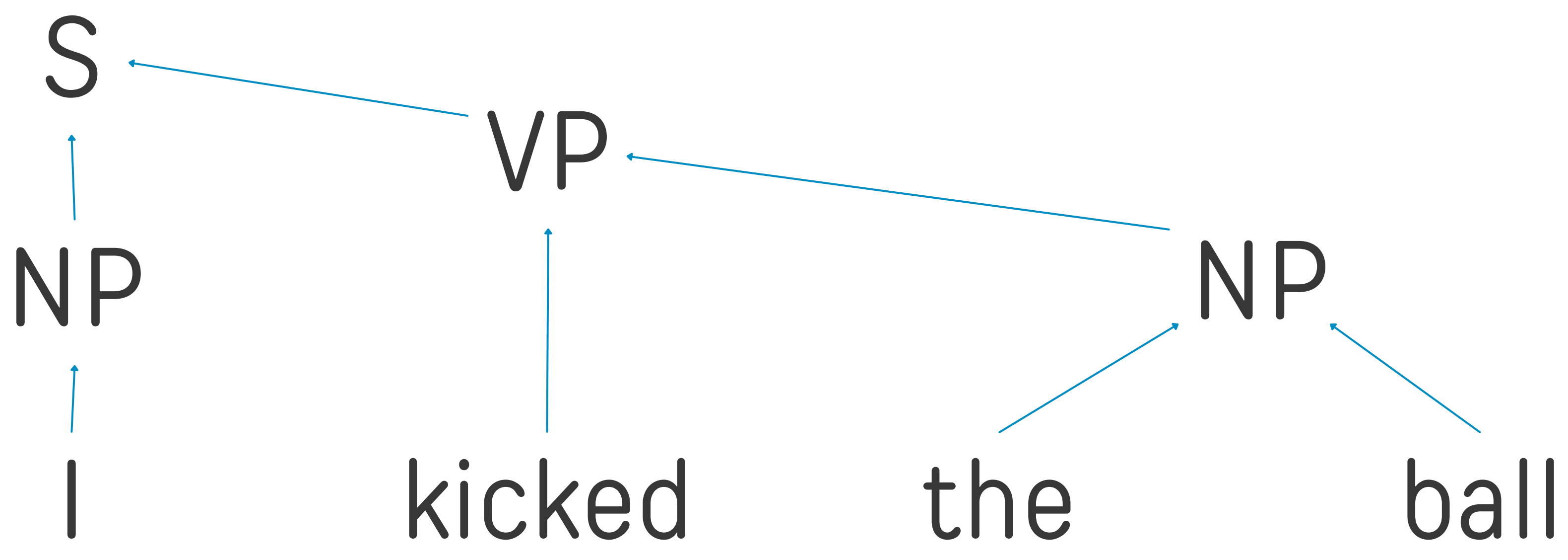
The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

Lingustics Need Context

- Recurrent Neural networks process one token at a time



- In linguistics, people believe that instead sentences are best understood by combining into higher level concepts

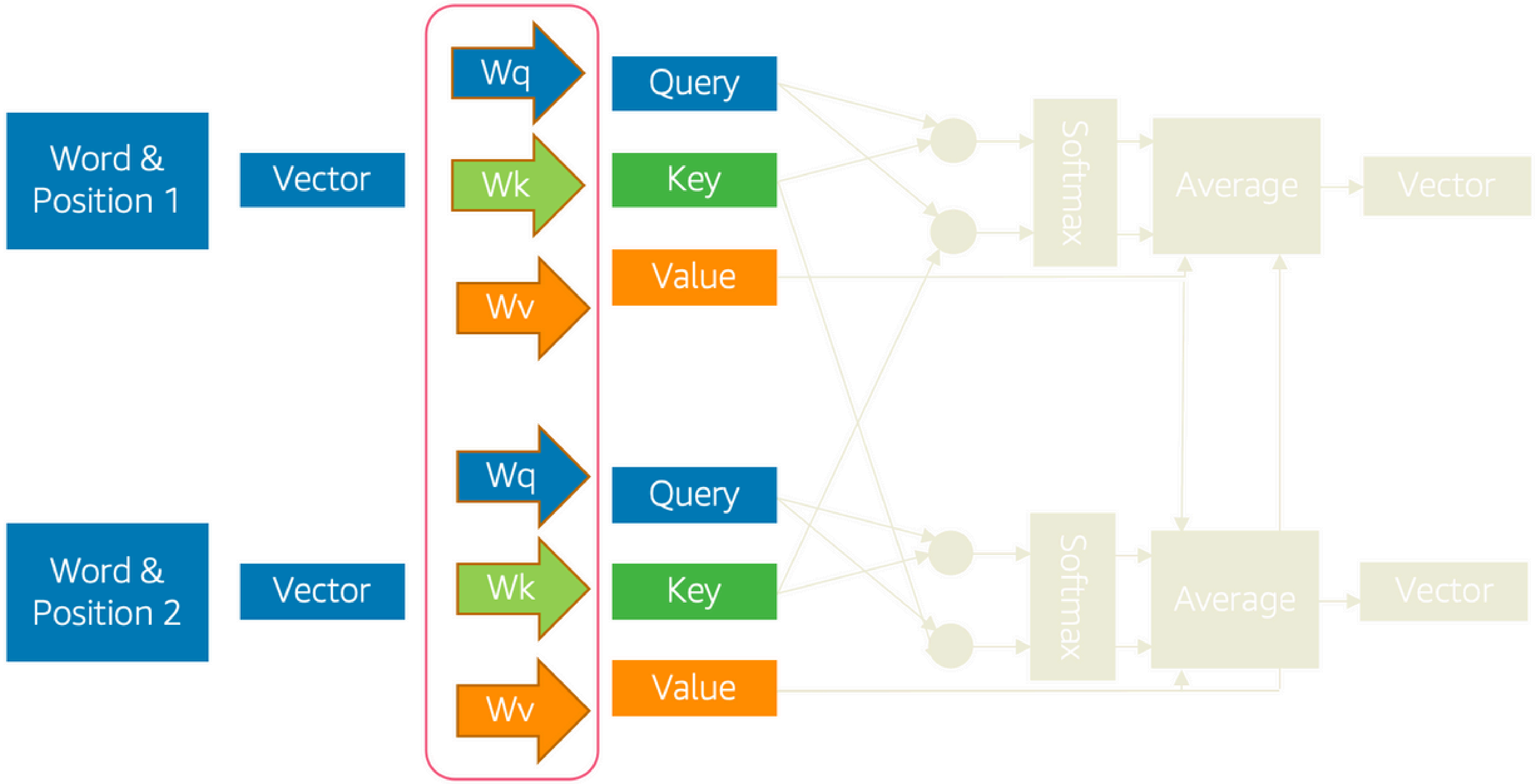


S: Sentence
VP: Verb Phrase
NP: Noun Phrase

Can we make a model that mirrors this philosophy?

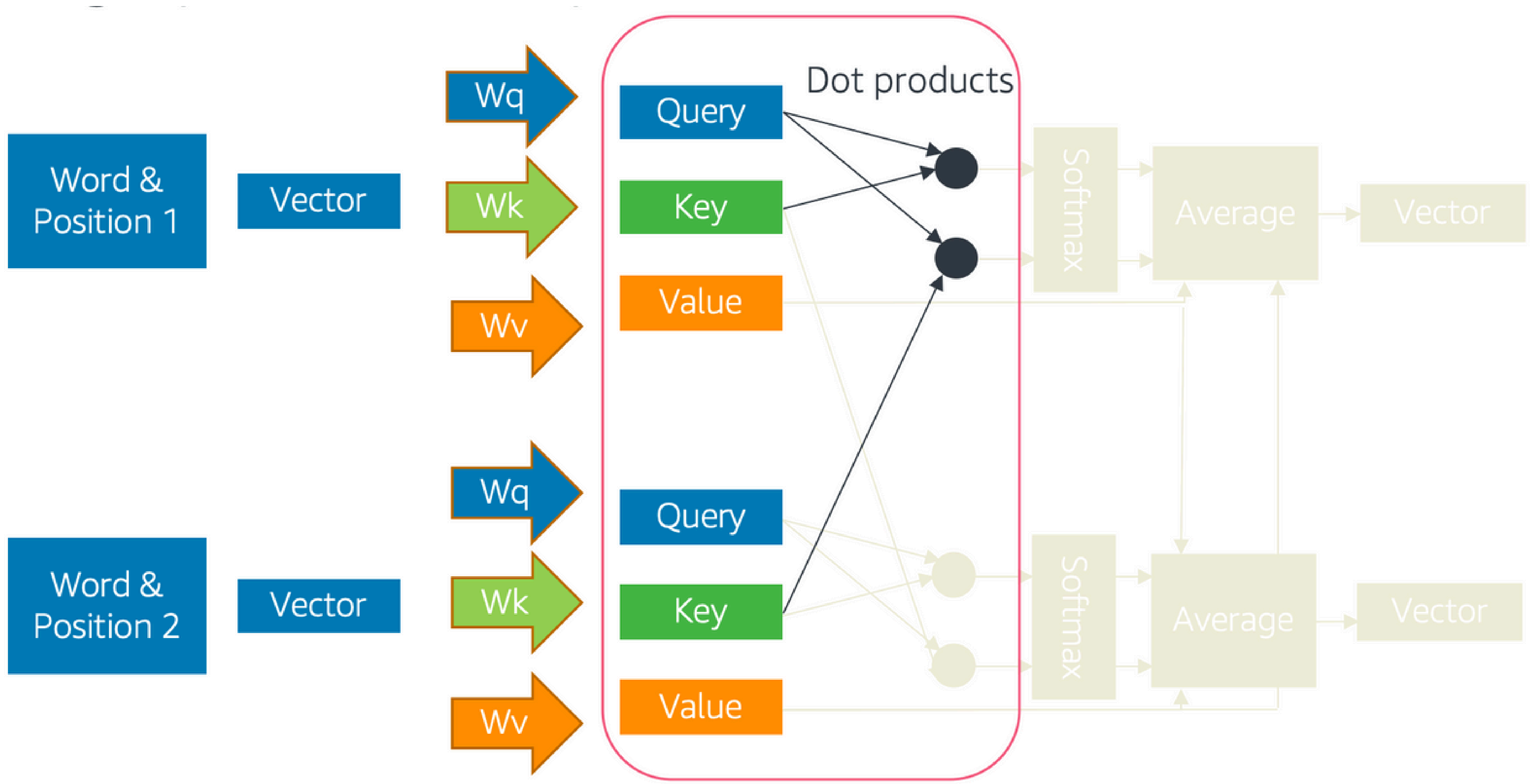
Single Headed Attention

The **key**, **query**, and **value** will all be vectors of numbers.



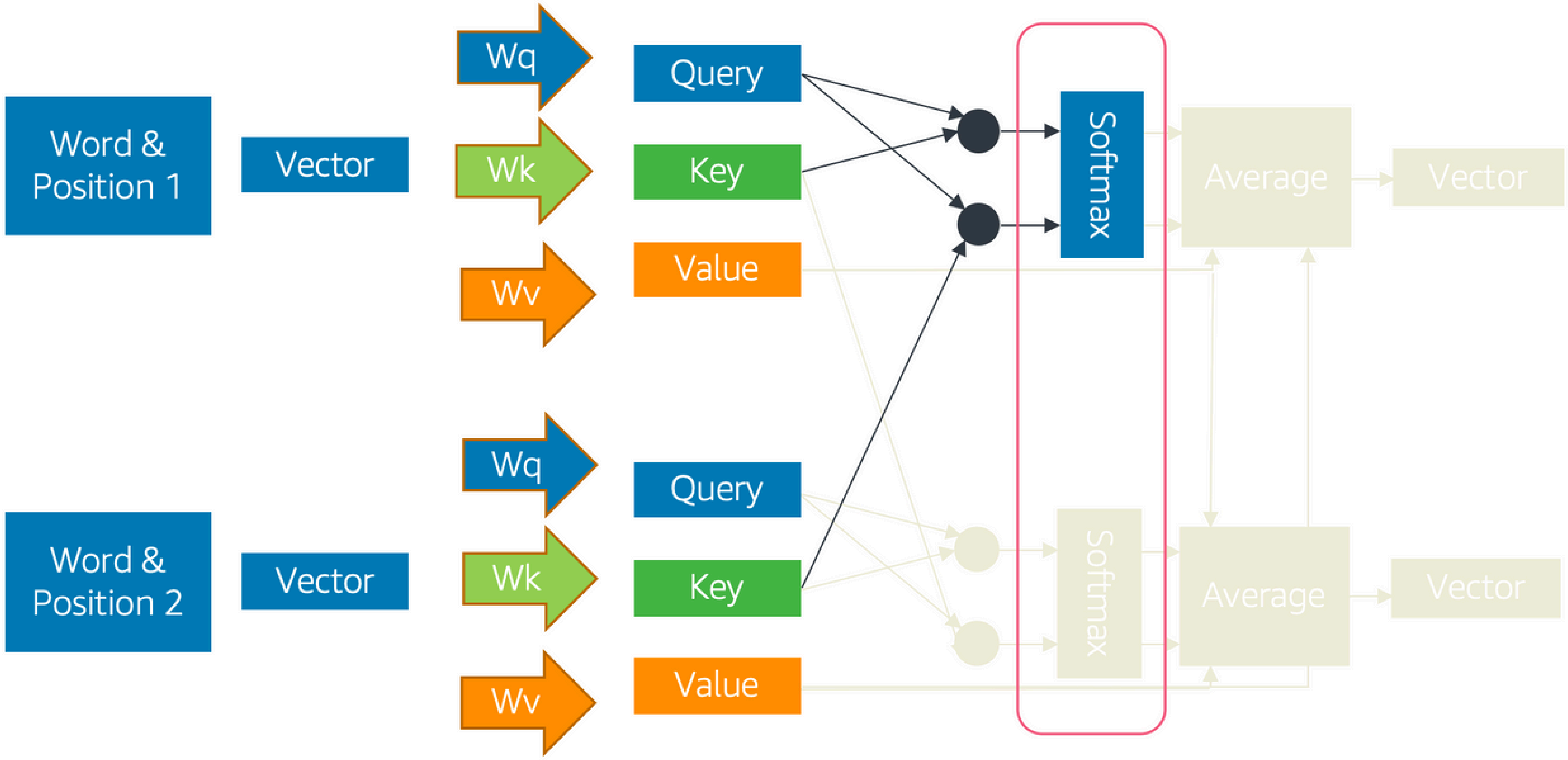
Single Headed Attention

Similarity is commonly given by the **dot product** .
Large positive dot products are similar.



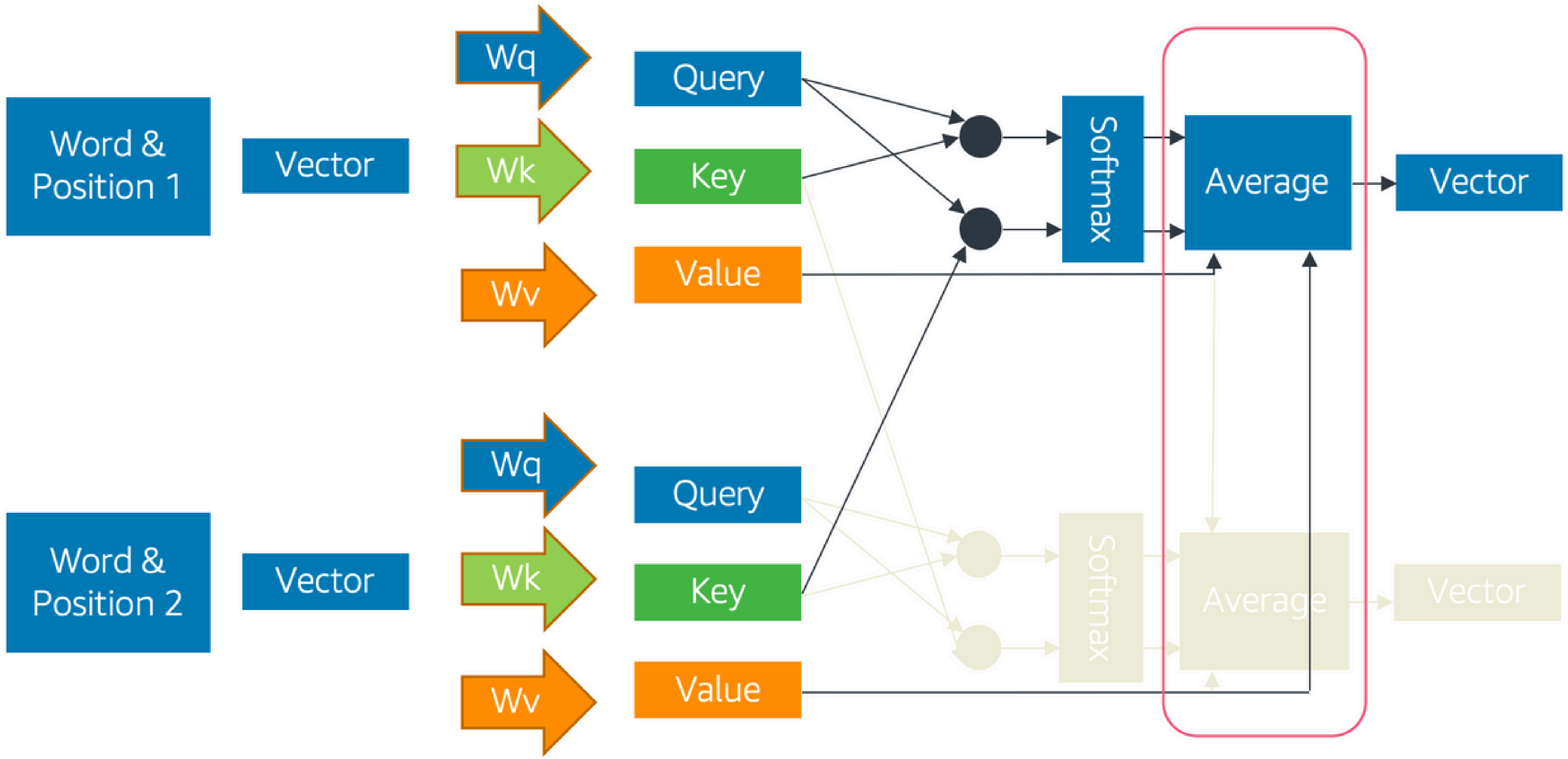
Single Headed Attention

At each token, compute the **softmax** of the dot products to get a collection of weights that sum to one
The larger are corresponding to the **larger dot products**.



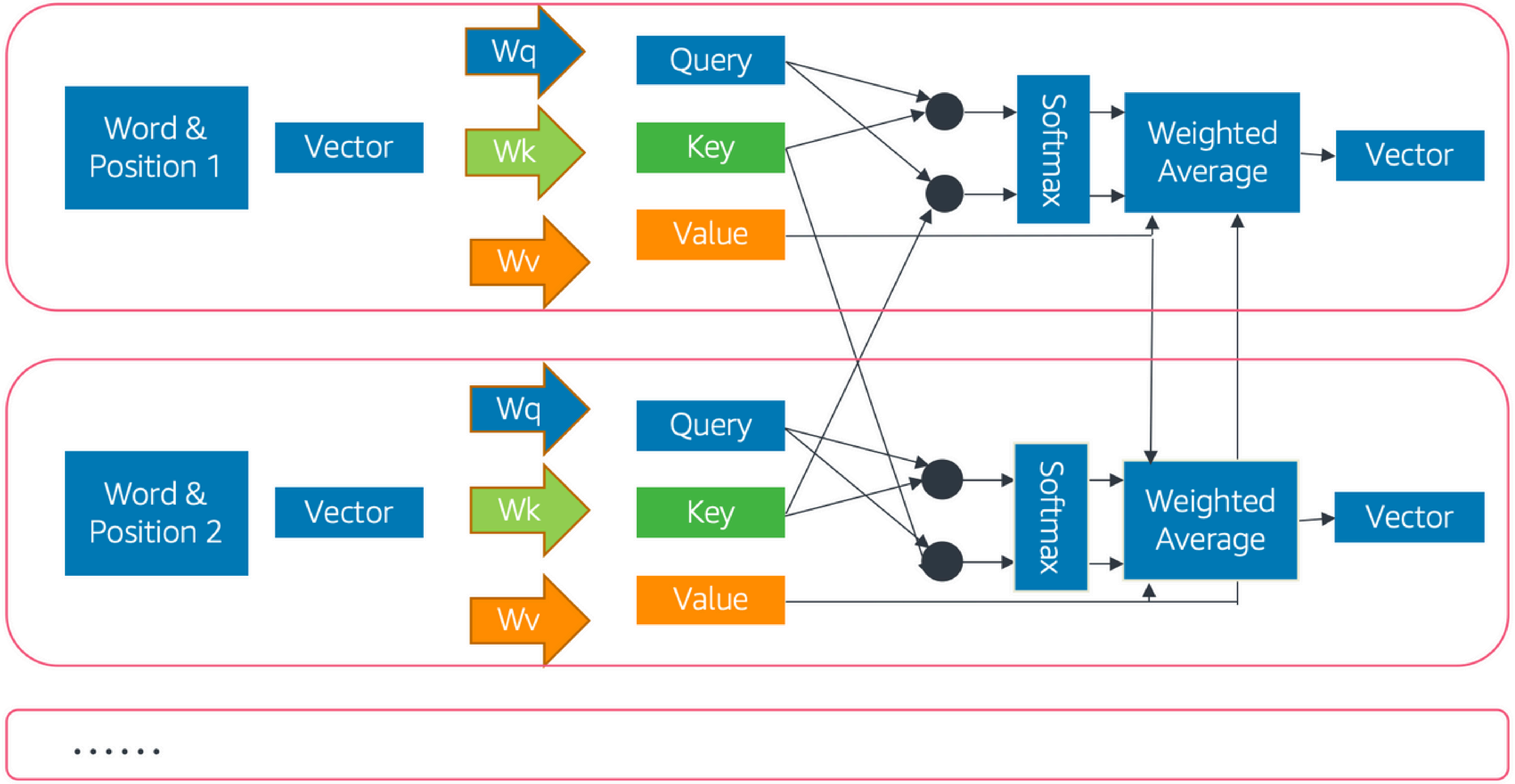
Single Headed Attention

The final lookup is obtained by **weighted averaging** the values:



Single Headed Attention

This process is repeated for every token in the network.



Single Headed Attention

At each token:

- The **key**, **query**, and **value** are represented by vectors of numbers.
- The query and (other token's) key **similarity** is commonly given by the **dot product** . Large positive dot products are similar.
- The query and all (other token's) key **similarities are normalized by** the **softmax** of the dot products to get the weights
- The output value of the query is the **weighted average of** the (other token's) values:

Single Headed Attention - Challenges

Issue of Polysemy

Consider some of the meanings of the word ”tie”

verb: to fasten or attach
“I tied the bag closed.”

verb: to establish in relationship
“We tied the criminal to the scene of the crime.”

noun: railway supports
“She hammered the rail to the tie.”

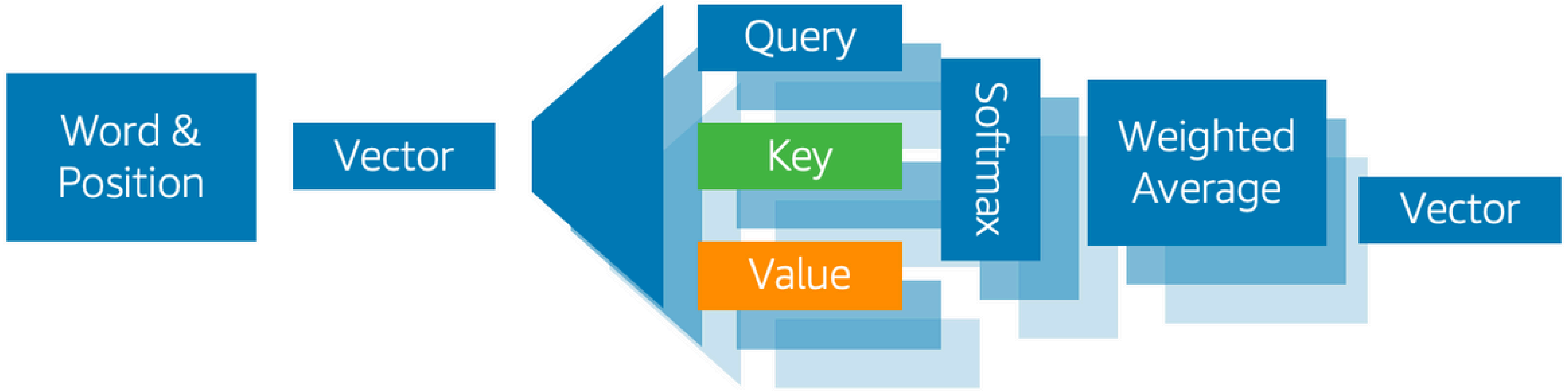
noun: equality in a contest
“The game ended in a tie.”

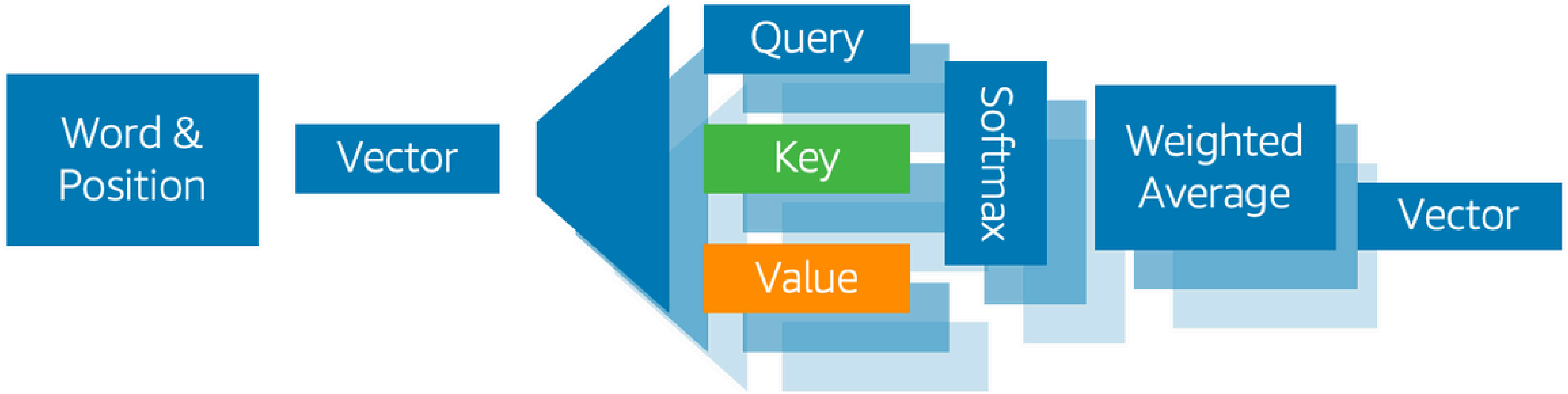
noun: sustained tone in music:
“The tie holds the note into the next measure.”

noun: Something knotted when worn
“He put on his favorite tie for the job interview

Multi Headed Attention

To solve the issue of polysemy, every token (and every subsequent layer) will emit **multiple keys, multiple values, and multiple queries**! This is called **multi-headed attention**.



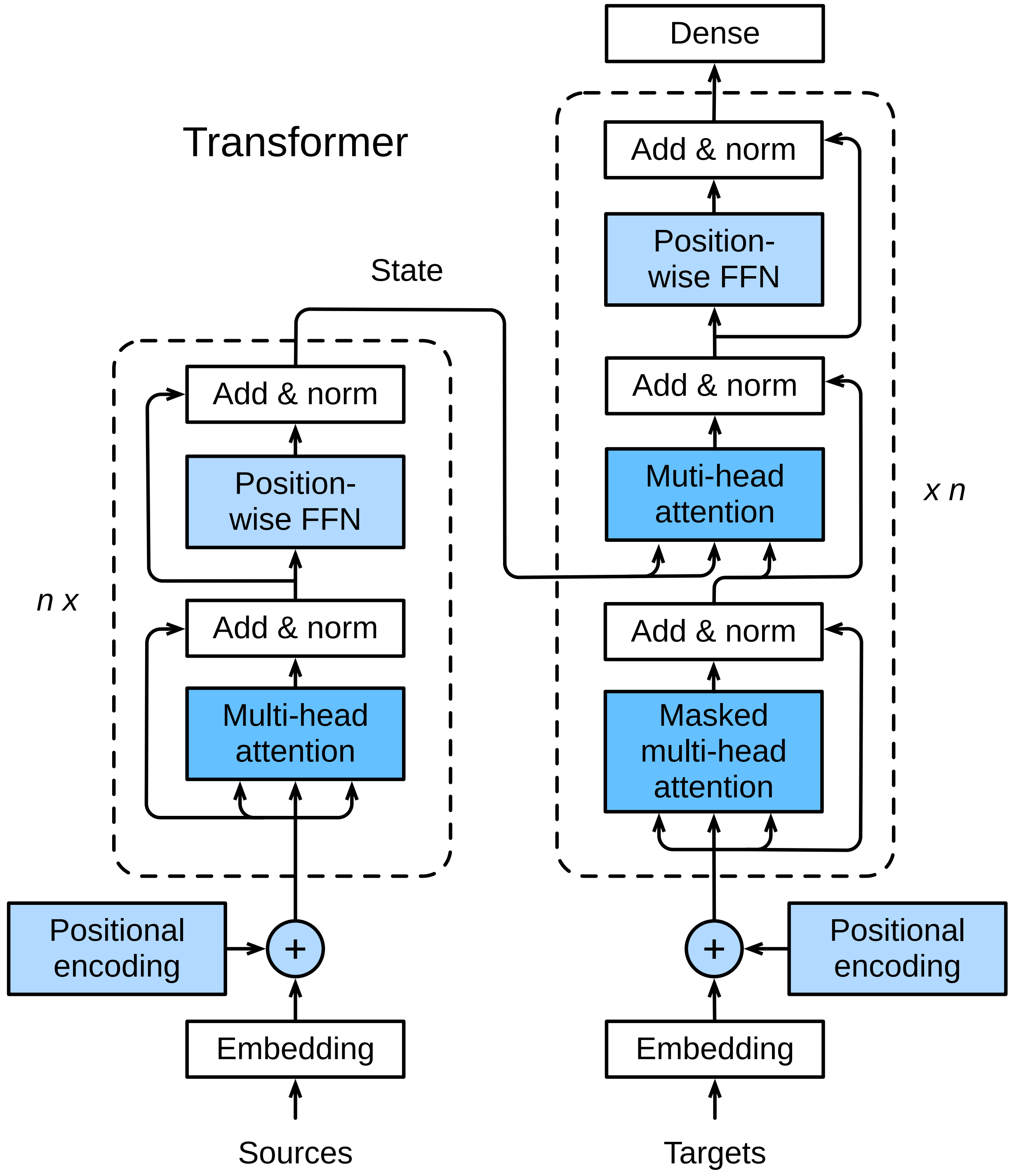


Transformers

The full architecture contains:

- Transformer block
- Add and norm
- Position encoding

More details of transformer.



Using Pre-trained Model

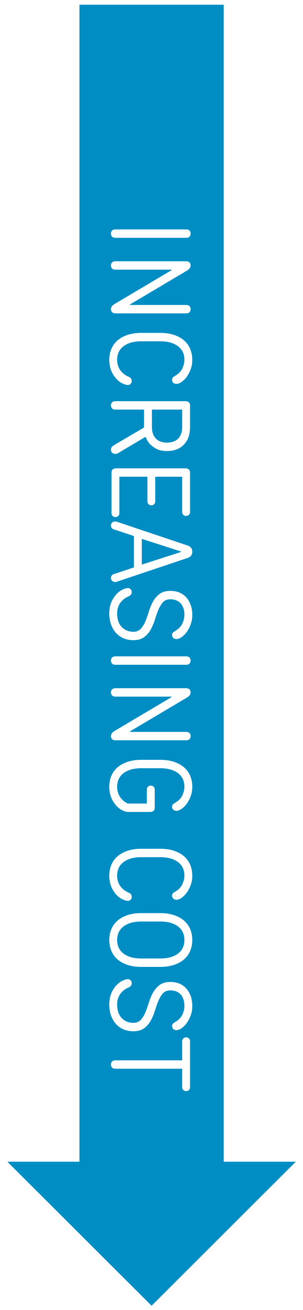
- Transformers take a long time to train
  - GPT (240 GPU days)
  - BERT (256 TPU days)
  - GPT-2 (2048 TPU days)

- Directly use pre-trained models
  - The Hugging Face contains a large number of pretrained transformer models (BERT, RoBERTa, BioBERT, ClinicalBERT, etc., LLAMA) on a varied of corpus.

Using Pre-Trained Models

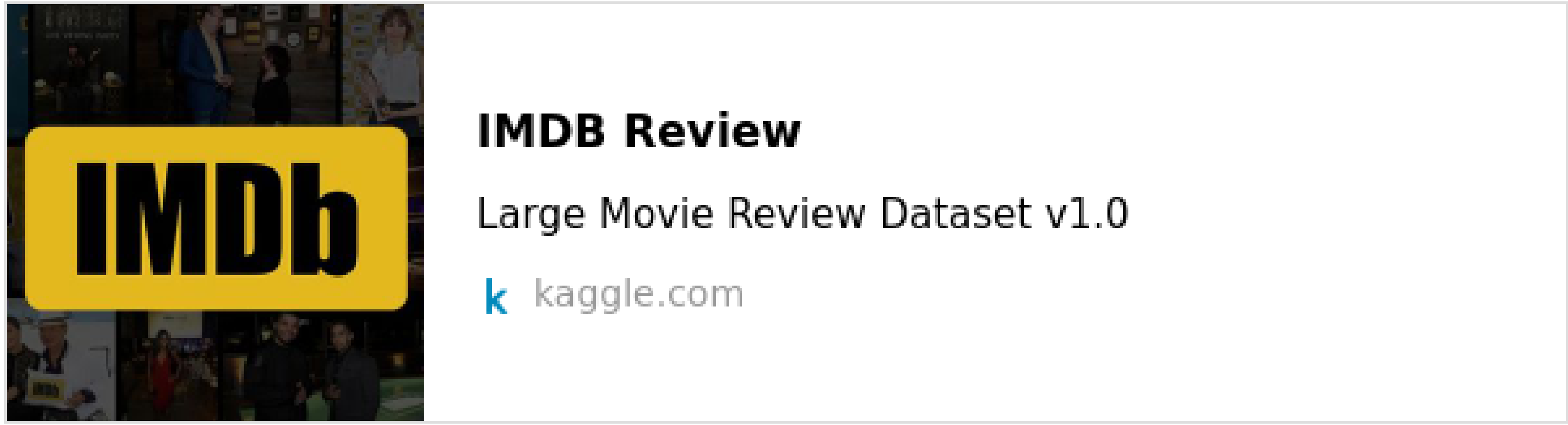
There are multiple ways to use these pre-trained models:

- Use it as a **fixed embedding** (no training cost). We investigate this in a notebook example.
- The model can be **fine-tuned end-to-end** as a classifier.
- The model can be **fine-tuned as a language model** on your specific dataset, then used as an encoder or fine-tuned on the classification task.
- The model can be **trained from scratch**. This is very intensive, and should only be done with **50GB+ of text**.



Demo 3 - BERT Model

Office Hours Week2
Discussion - Exercise - Day 2



IMDB Review

Large Movie Review Dataset v1.0

k kaggle.com

https://www.kaggle.com/datasets/pawankumargunjan/imdb-review

Google Play Store Reviews

App Reviews collected from Google Play Store for the task of sentiment analysis

k kaggle.com

https://www.kaggle.com/datasets/prakharrathi25/google-play-store-reviews