A simple neural classifier

Antoine Bosselut

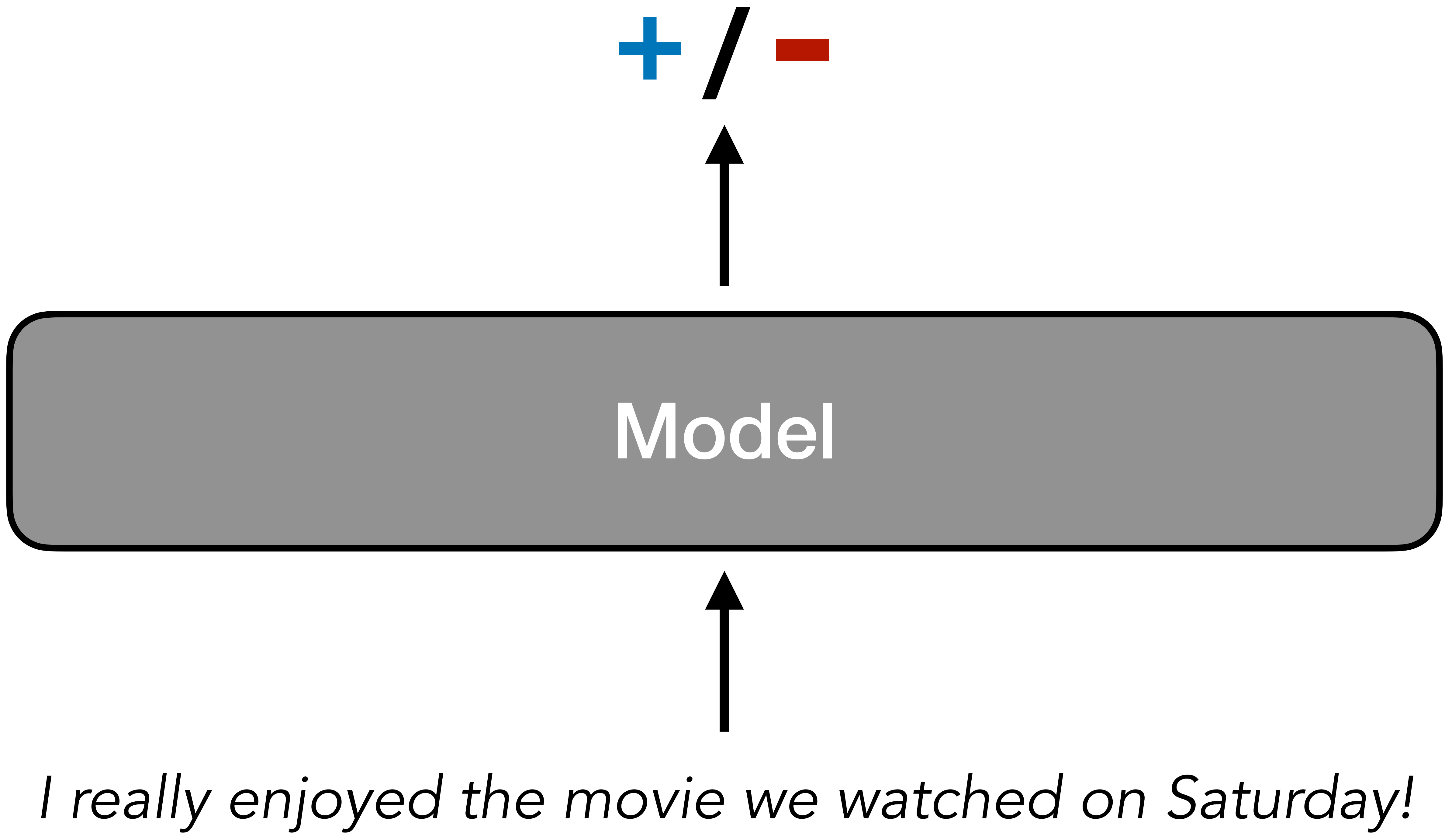
EPFL

Section Outline

- Setting up an NLP problem
- **Embeddings:** how do we represent sequences of discrete words ?
- **Model:** how do we compose our embeddings into higher-level representations?
- **Prediction:** how do we map our model's representation of the task to a prediction?

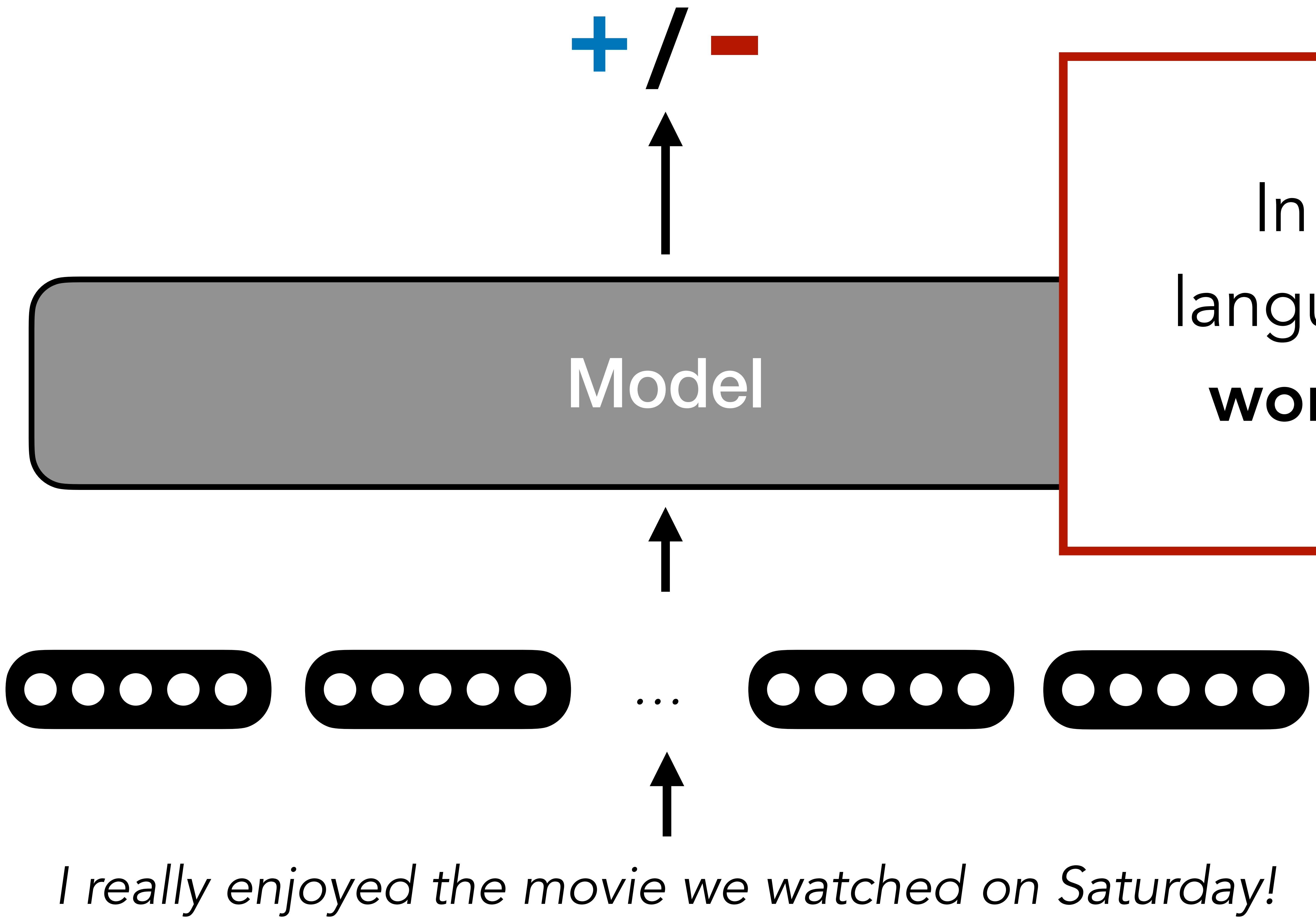
A simple NLP model

- **Sentiment analysis example**: Convert a sentence describing a movie review to a sentiment



A simple NLP model

- **Sentiment analysis example**: Convert a sentence describing a movie review to a sentiment



In neural natural language processing, **words** are **vectors**!

Question

What words should we model as vectors?

Choosing a vocabulary

- Language contains many words (e.g., ~600,000 in English)
  - **What about other tokens**: Capitalisation? Accents ? Typos!? Words in other languages!? In other scripts!? Emojis !? Unicode !?
  - **Millions of potential unique tokens!** Most rarely appear in our training data (Zipfian distribution)
  - Model has limited capacity

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

Choosing a vocabulary

- Language contains many words (e.g., ~600,000 in English)
  - **What about other tokens**: Capitalisation? Accents ? Typos!? Words in other languages!? In other scripts!? Emojis !? Unicode !?
  - **Millions of potential unique tokens!** Most rarely appear in our training data (Zipfian distribution)
  - Model has limited capacity

- How should we select which tokens we want our model to process?
  - Week 3 - tokenisation!
  - For now, initialize a vocabulary *V* of tokens that we can represent as a vector
  - Any token not in this vocabulary *V* is mapped to a special <UNK> token (i.e., “unknown”).

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

Question

How should we model a word as a vector?

One upon a time: **sparse word representations**

$$x_i \in \{0,1\}^V$$

- Define a vocabulary *V*
- Each word in the vocabulary is represented by a sparse *vector*
- Dimensionality of sparse vector is size of vocabulary (e.g., thousands, possibly millions)

I	⟶	[0 … 0 0 0 1 … 0 0]
really	⟶	[0 … 1 … 0 0 0 0 0]
enjoyed	⟶	[0 … 0 0 0 1 0 … 0]
the	⟶	[0 … 0 1 0 0 0 … 0]
movie	⟶	[0 … 0 0 0 0 0 … 1]
!	⟶	[1 … 0 0 0 0 0 0 0 0]

Word Vector Composition

- To represent sequences, beyond single words, define a composition function over sparse vectors

I really enjoyed the movie ! → [1 … 1 1 0 1 … 0 1] **Simple Counts**

I really enjoyed the movie ! → [0.01 … 0.1 0.1 0 0.001 … 0 0.5] **Weighted by Corpus Statistics (e.g., TF-IDF)**

Many others…

Problem

With sparse vectors, similarity is a function of common words!

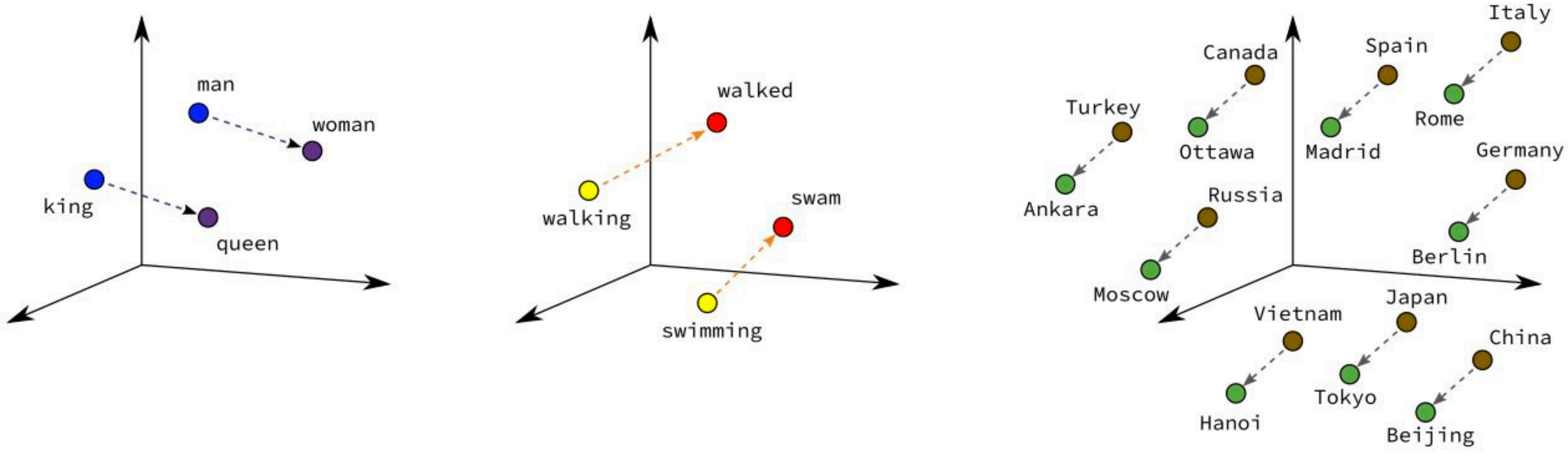
How do you learn similarity between words?

enjoyed $\longrightarrow$ [0 … 0 0 0 1 … 0 0]

loved $\longrightarrow$ [0 … 1 … 0 0 0 0 0]

sim(*enjoyed*, *loved*) = **0**

Embeddings Goal



Male-Female | Verb Tense | Country-Capital

How do we train semantics-encoding embeddings of words?

Image Credit: https://towardsdatascience.com/legal-applications-of-neural-word-embeddings-556b7515012f

Dense Word Vectors

- Represent each word as a high-dimensional*, **real-valued** vector
  - *Low-dimensional compared to V-dimension sparse representations, but still usually O(10^2 - 10^3)

I	→	[0.113 -0.782 1.893 0.984 6.349 ...]
really	→	[0.906 0.661 -0.214 -0.894 -0.880 ...]
enjoyed	→	[-0.842 0.647 -0.882 0.045 0.029 ...]
the	→	[0.100 0.765 -0.333 -0.538 -0.150 ...]
movie	→	[0.104 -0.054 -0.268 -0.877 0.005 ...]
!	→	[0.439 -0.577 -0.727 0.261 0.699 ...]

word vectors

word embeddings

neural embeddings

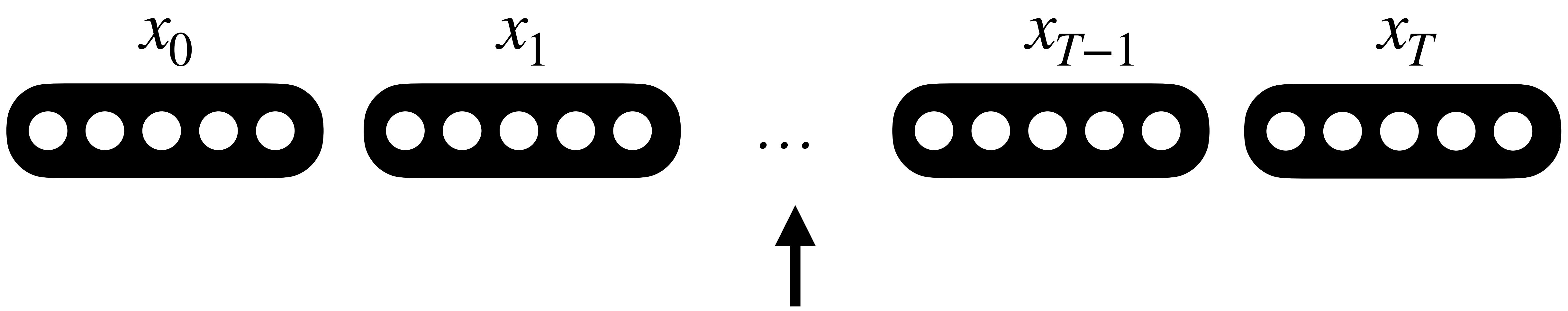
dense embeddings

others...

- Similarity of vectors represents similarity of meaning for particular words

A simple NLP model

- For each sequence S, we have a corresponding sequence of embeddings X

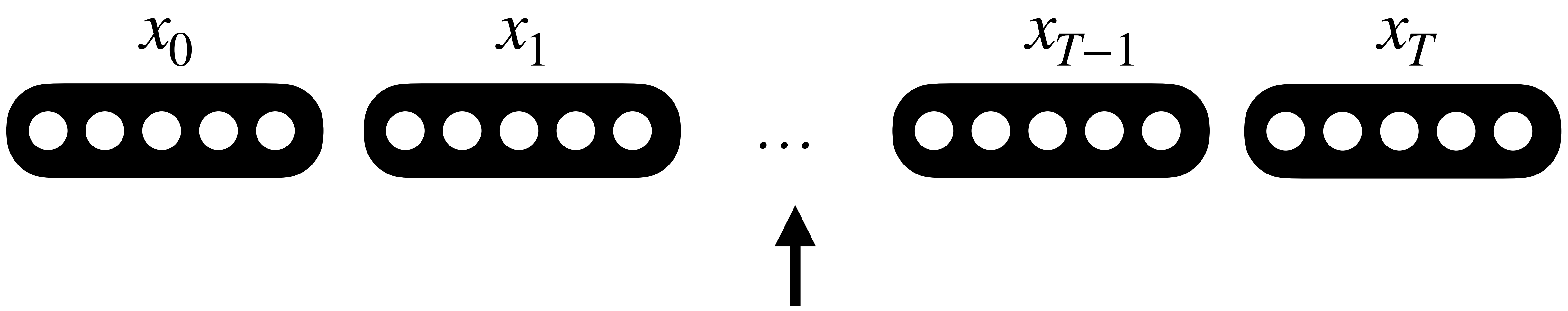


$$X = \{x_0\,, x_1\,, \ldots\,, x_T\}$$

$S =$ *I really enjoyed the movie we watched on Saturday!*

A simple NLP model

- For each sequence S, we have a corresponding sequence of embeddings X



$X = \{x_0, x_1, \dots, x_T\}$

$S_1 =$ *I **really** enjoyed the movie **we** watched on Saturday **!***

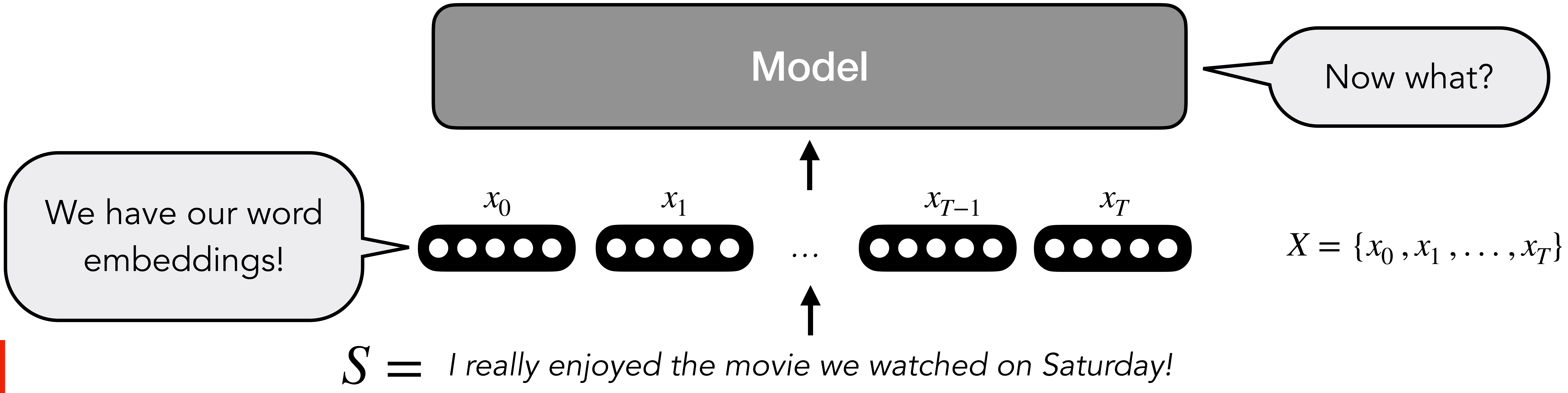
- Embeddings $x_t \in X$ are indexed from shared embedding dictionary $\mathbb{E}$ for all items in vocabulary V

$S_2 =$ ***we** really loved a film **we** saw last Sunday **!***

Bolded words would index the same embedding in $\mathbb{E}$

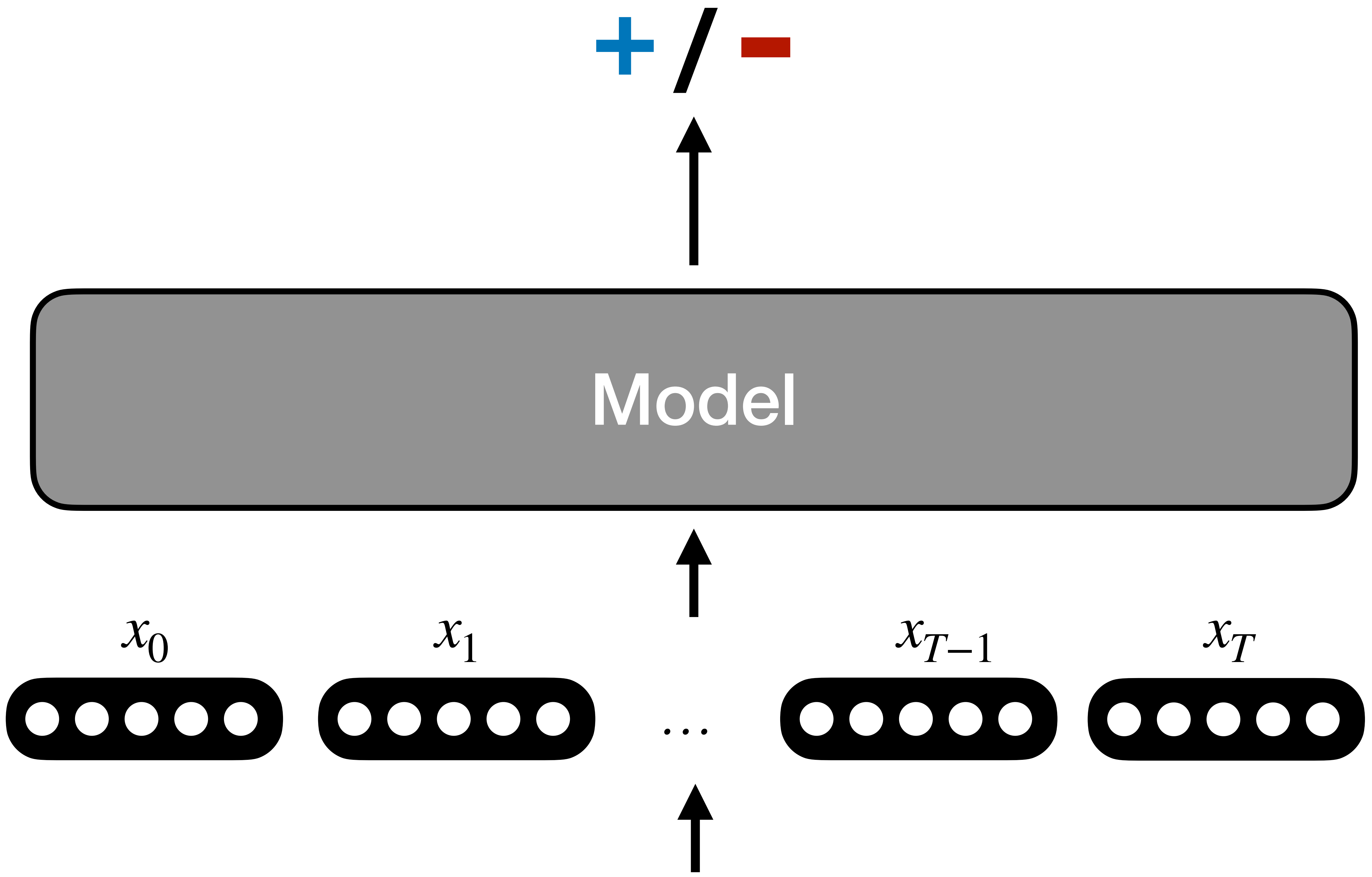
A simple NLP model

- For each sequence S, we have a corresponding sequence of embeddings X



A simple NLP model

- Our model modifies and / or composes these word embeddings to formulate a representation that allows it to predict the correct label



$$X = \{x_0\,, x_1\,, \ldots, x_T\}$$

$S =$ *I really enjoyed the movie we watched on Saturday!*

Question

What should we use as a model?

A simple NLP model

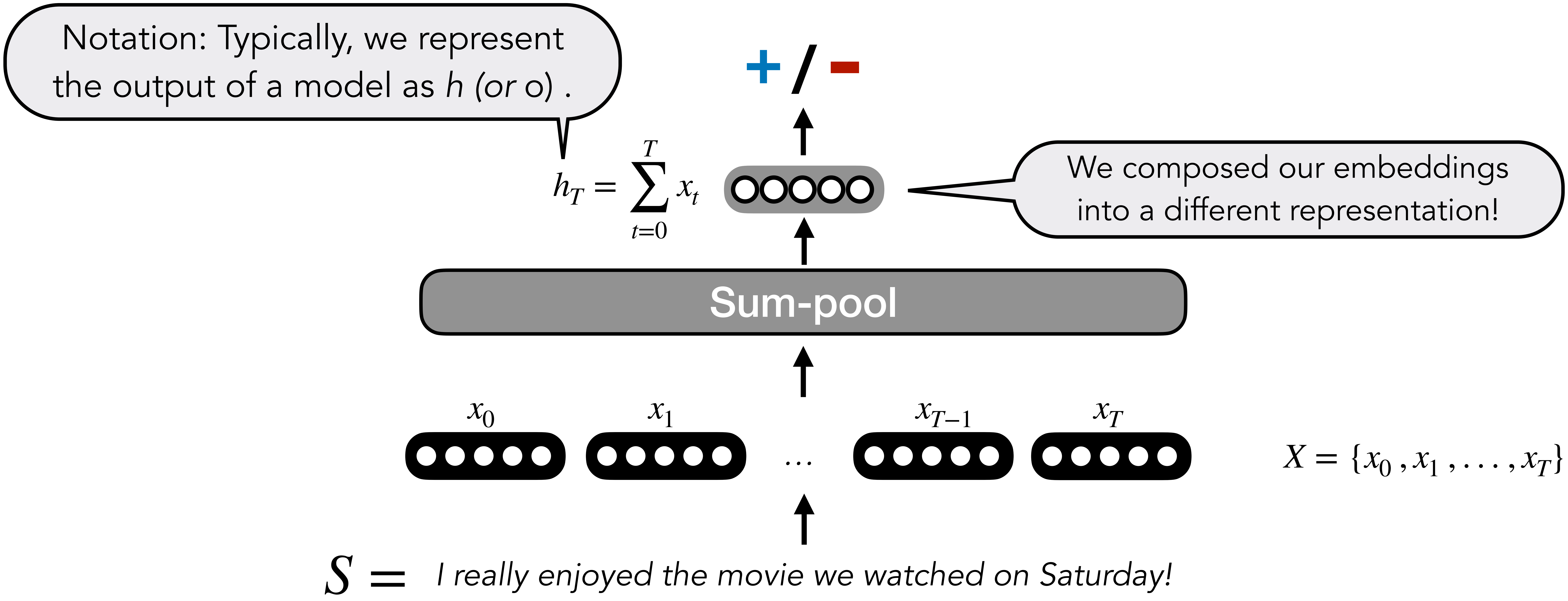
- Our model modifies and / or composes these word embeddings to formulate a representation that allows it to predict the correct label
  - Recurrent neural networks (RNNs) and variants (LSTM, GRU) - Week 2
  - Self-attention & Transformer - Week 3
  - State-space Models (not covered in this course)
  - Multiple of the above ?

A simple NLP model

- Our model modifies and / or composes these word embeddings to formulate a representation that allows it to predict the correct label
  - Recurrent neural networks (RNNs) and variants (LSTM, GRU) - Week 2
  - Self-attention & Transformer - Week 3
  - State-space Models (not covered in this course)
  - Multiple of the above ?
  - Or perhaps something super simple: **Sum-pool, Avg-pool, Max-pool**?

(Shen et al., 2018)

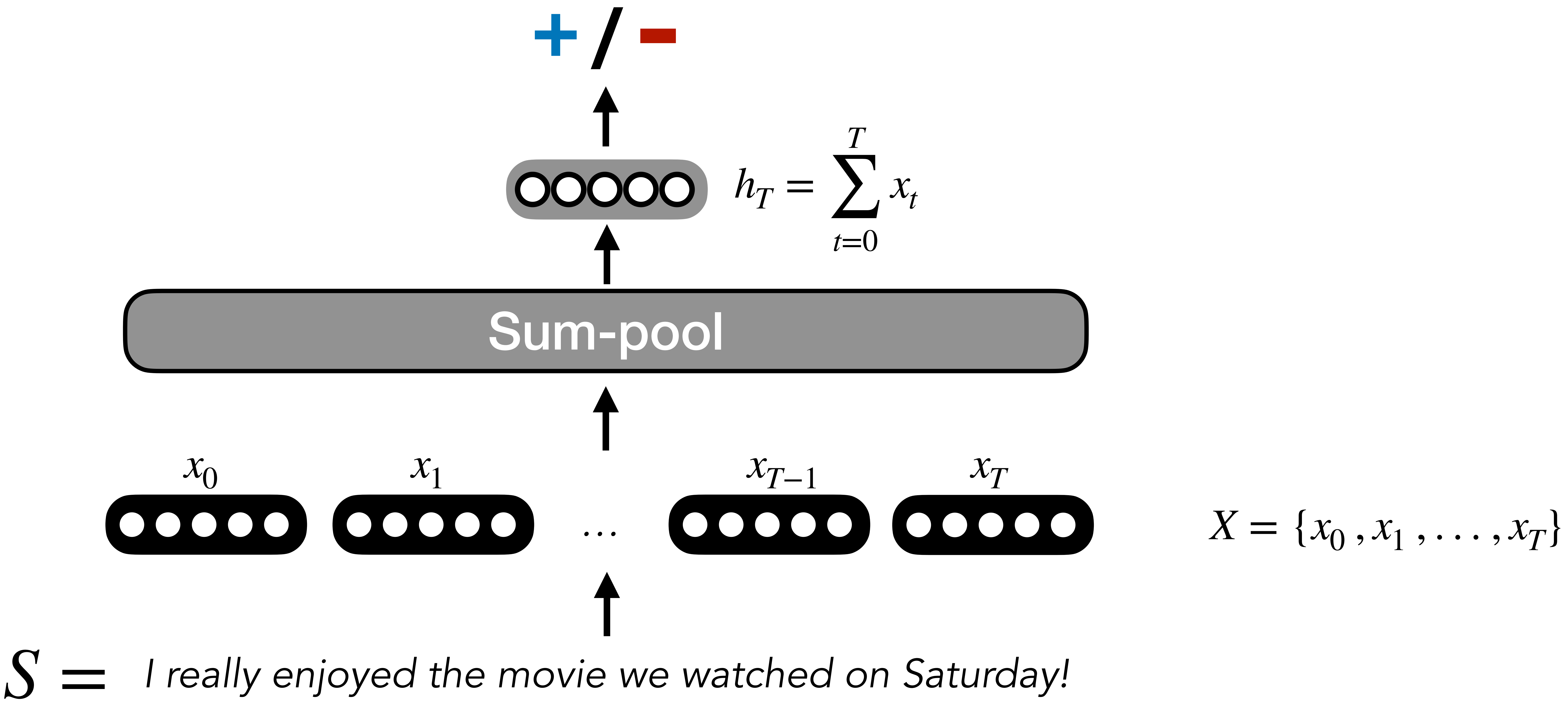
A simple NLP model



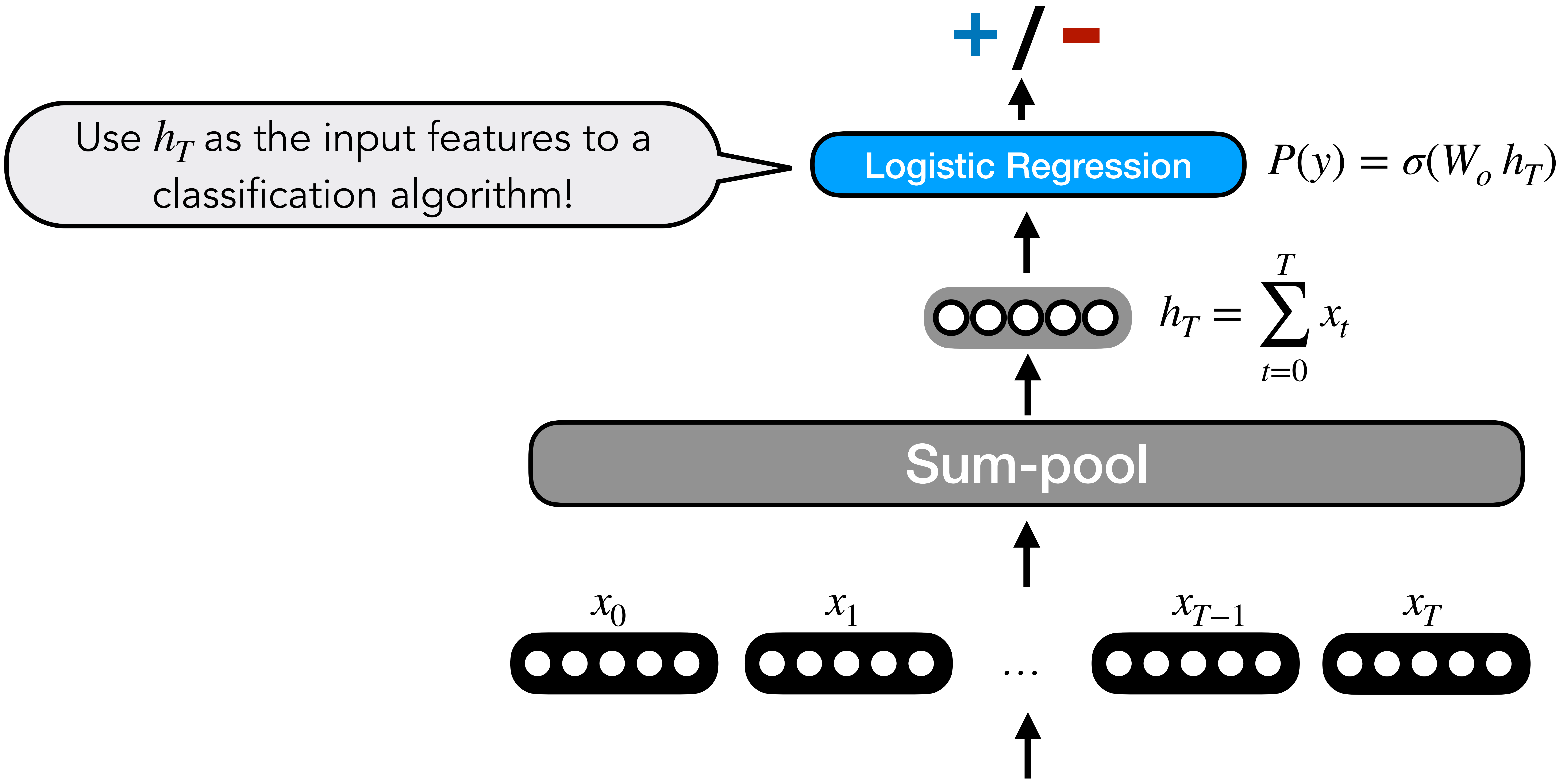
Question

How do we convert the output of our model to a prediction?

Predicting the label



Predicting the label



$S =$ *I really enjoyed the movie we watched on Saturday!*

Learn using **backpropagation:** compute gradients of loss with respect to initial embeddings X

Learn embeddings that allow you to do the task successfully!

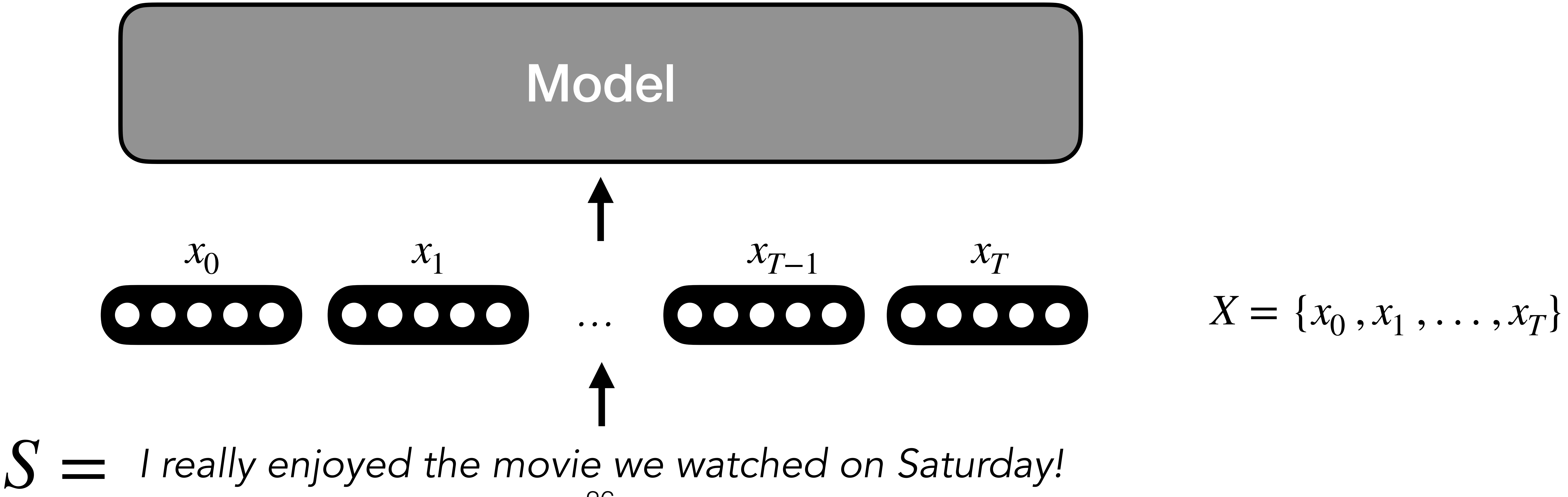
$X = \{x_0\,, x_1\,, \ldots, x_T\}$

Question

How could we use our model for tasks beyond classification?

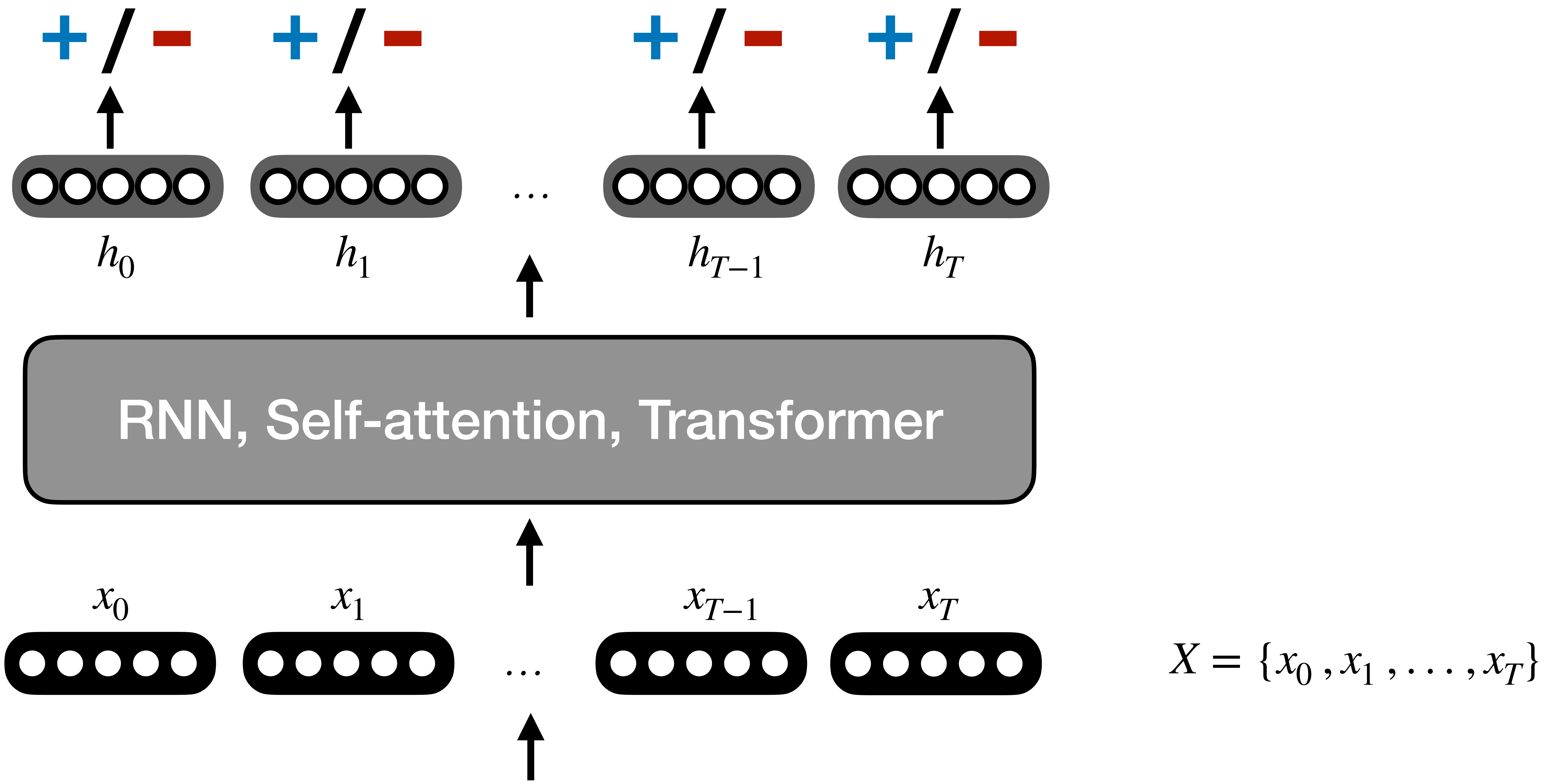
Sequence Labeling

- **Example:** Identify which words correspond to sentimental words



Sequence Labeling

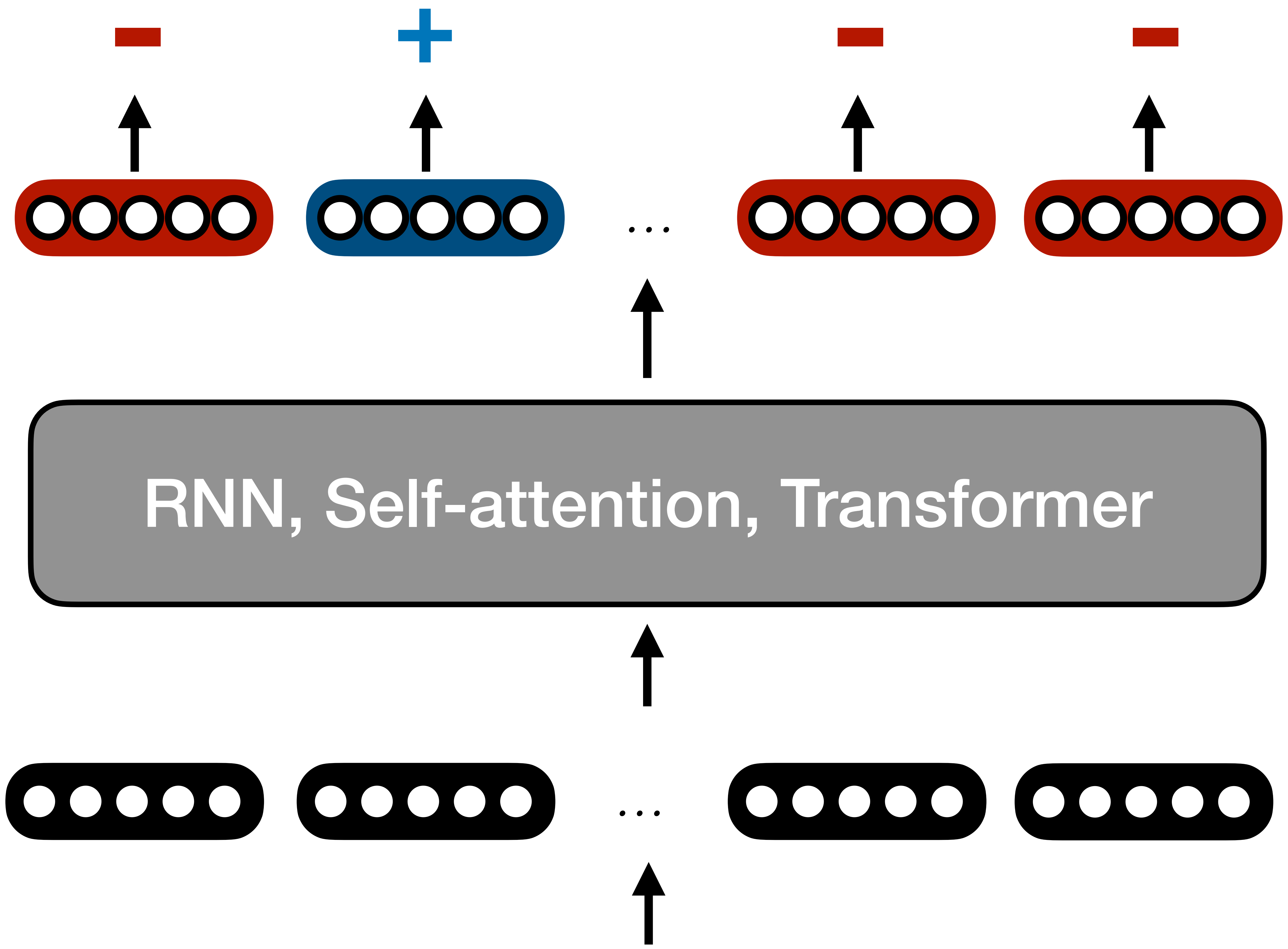
- **Example:** Identify which words correspond to sentimental words



$S =$ *I really enjoyed the movie we watched on Saturday!*

Sequence Labeling

- **Example:** Identify which words correspond to sentimental words

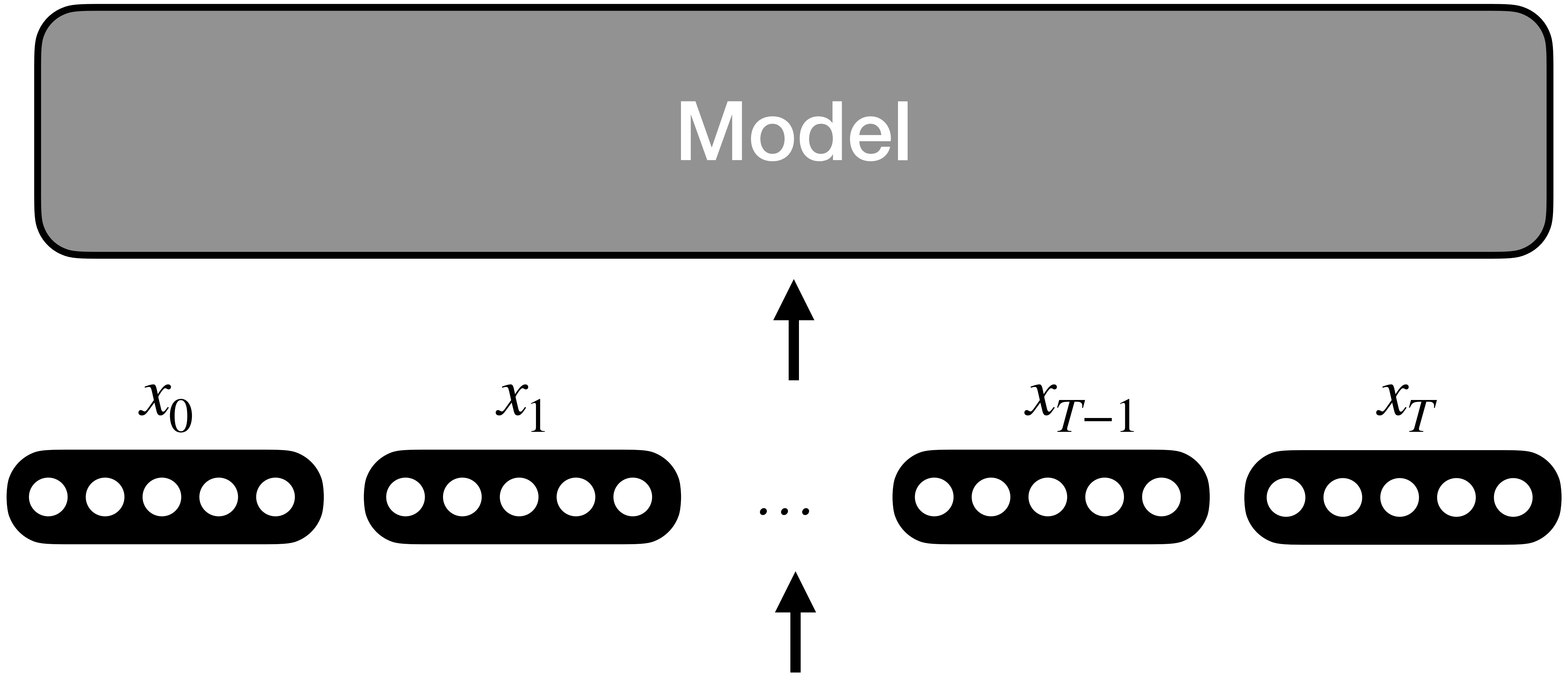


$S =$ *I really enjoyed the movie we watched on Saturday!*

Text Generation

- **Example:** Generate the next sentence in the review.

How might we do this given the ingredients we've seen so far ?

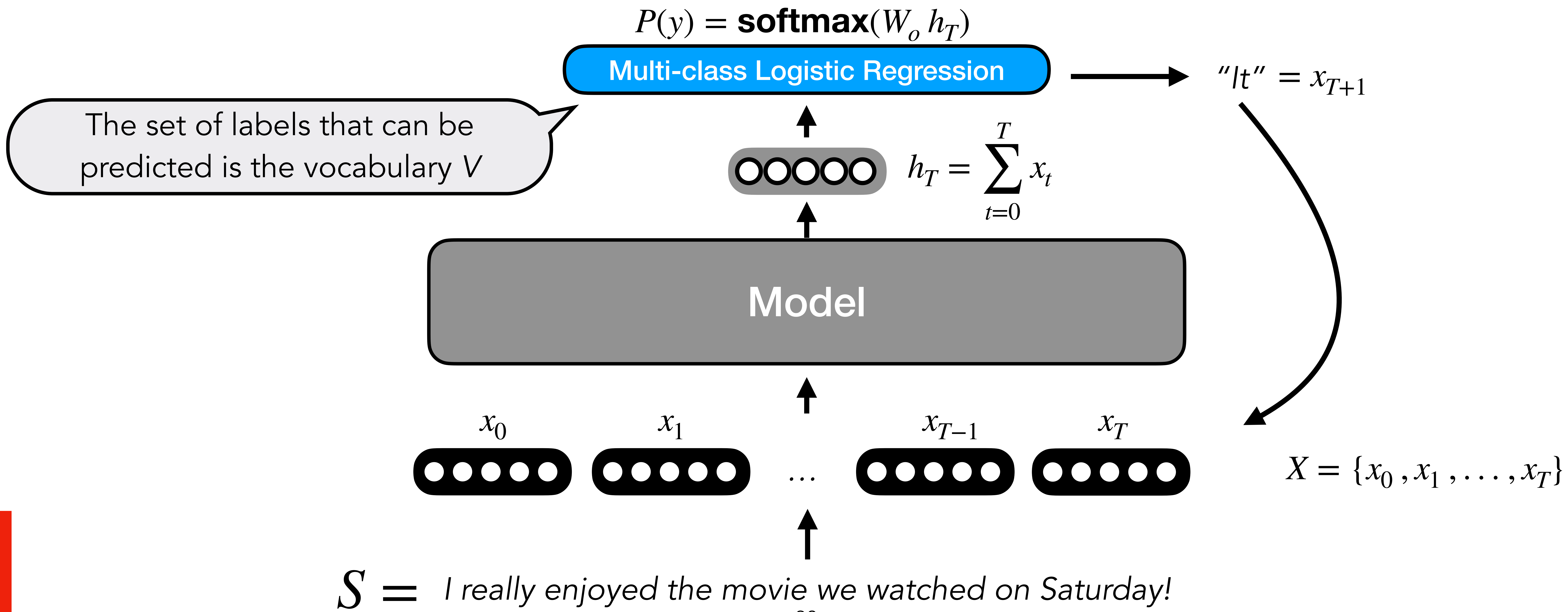


$X = \{x_0, x_1, \ldots, x_T\}$

$S =$ *I really enjoyed the movie we watched on Saturday!*

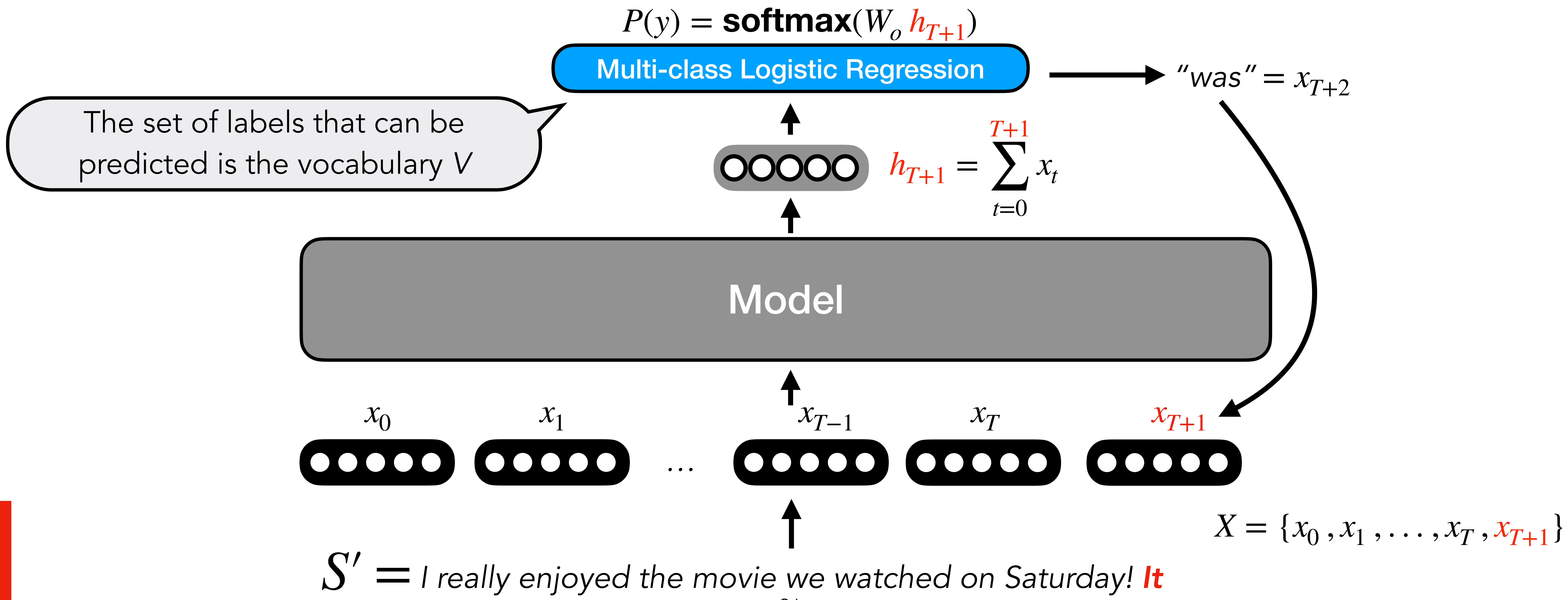
Text Generation

- **Example:** Generate the next sentence in the review. **Word-by-word!**



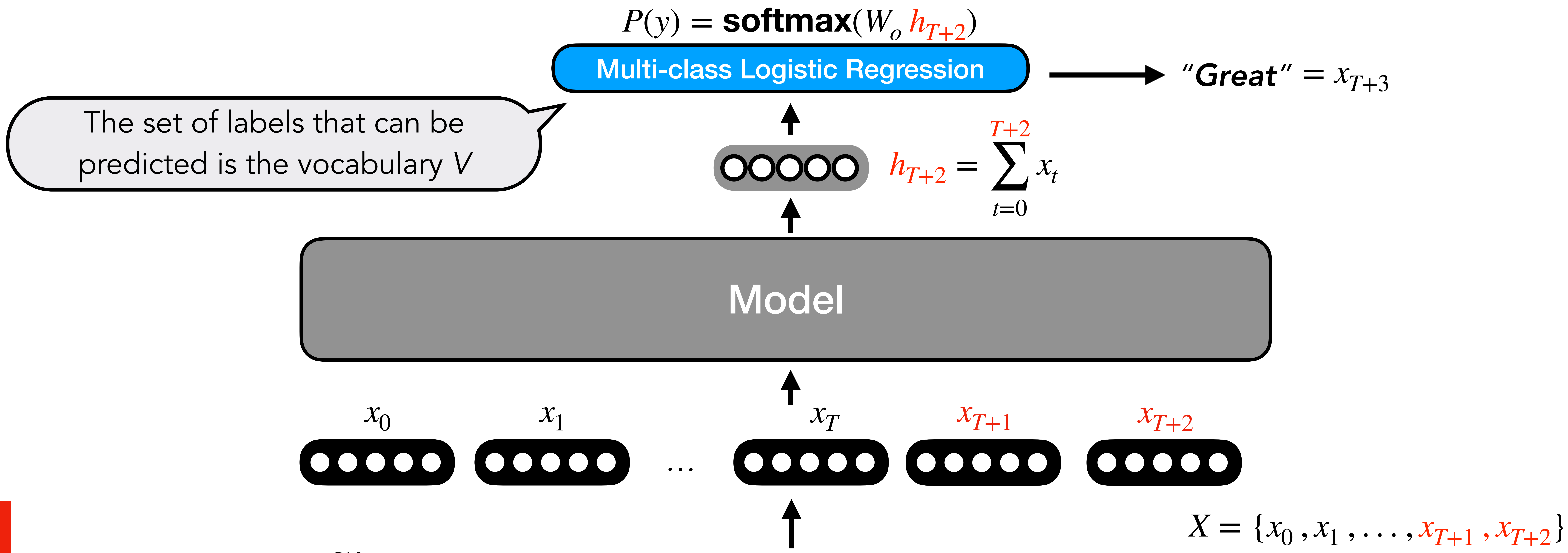
Text Generation

- **Example:** Generate the next sentence in the review. **Word-by-word!**



Text Generation

- **Example:** Generate the next sentence in the review. **Word-by-word!**



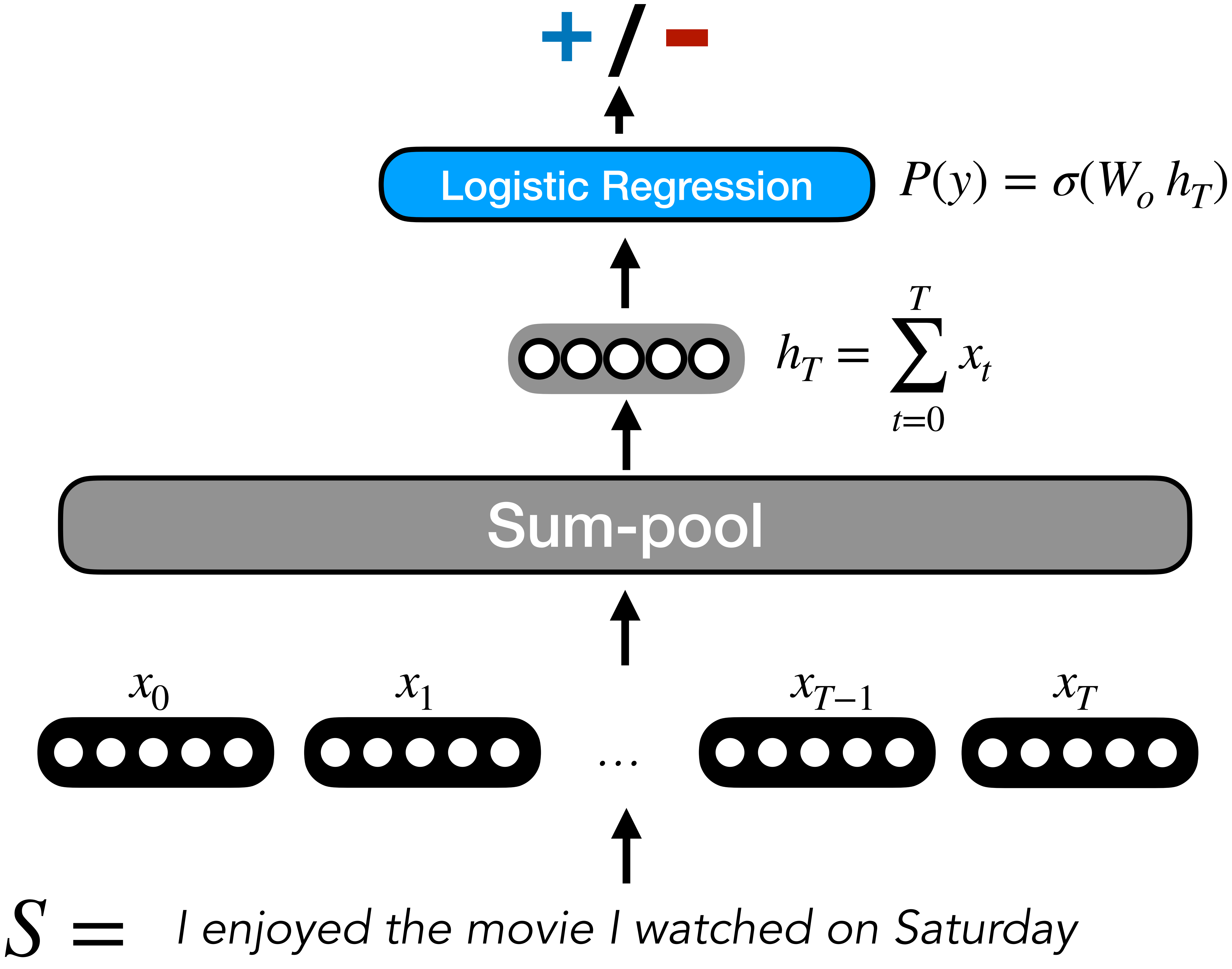
$S' =$ *I really enjoyed the movie we watched on Saturday!* ***It was***

Comprehension Questions

- What are the learnable parameters in our system?

Embeddings $\mathbb{E}$

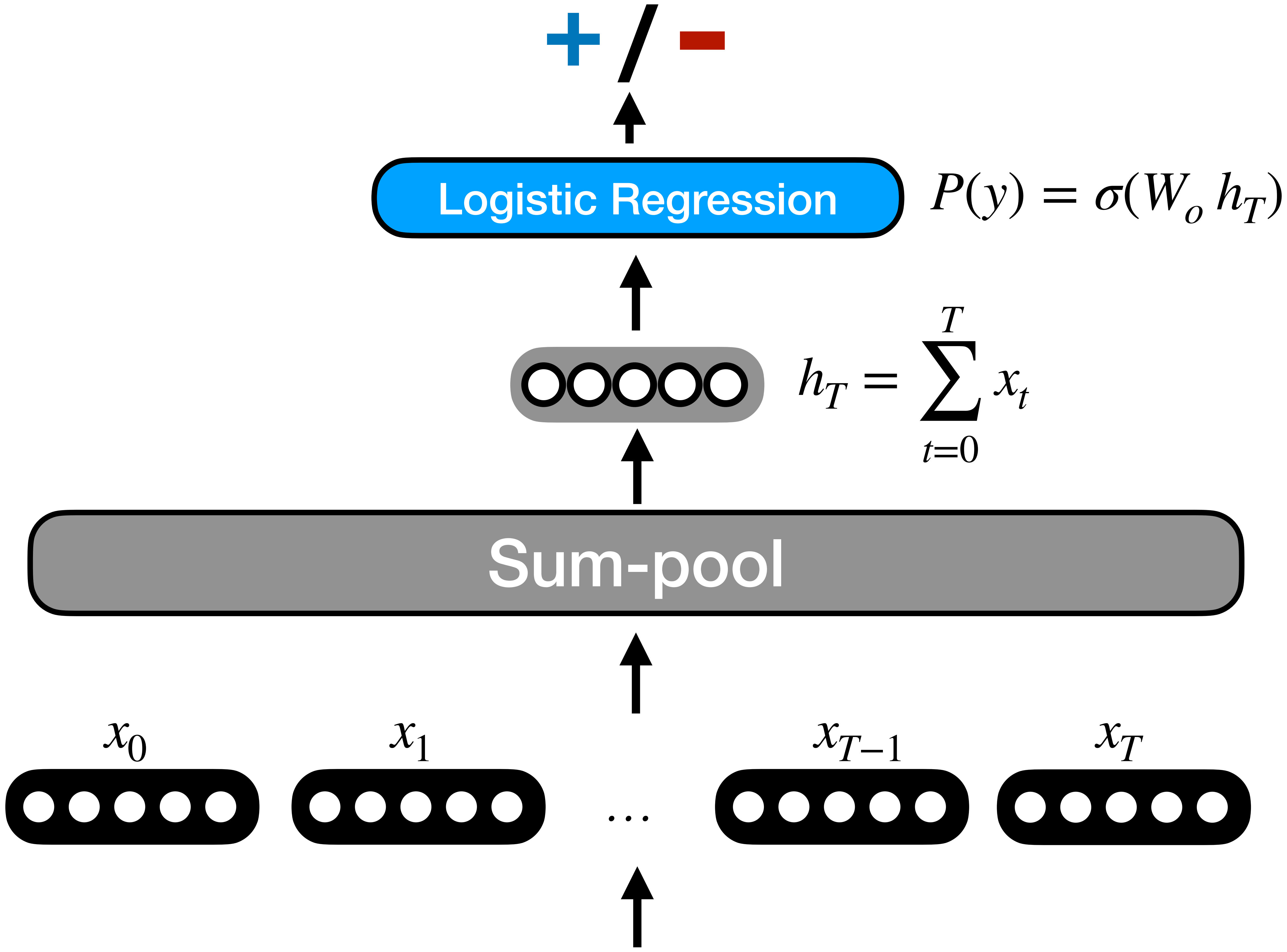
Logistic Regression matrix W_o



Comprehension Questions

- What are the learnable parameters in our system?
- How many **unique** embeddings are in *X* for this example sentence *S*?

7

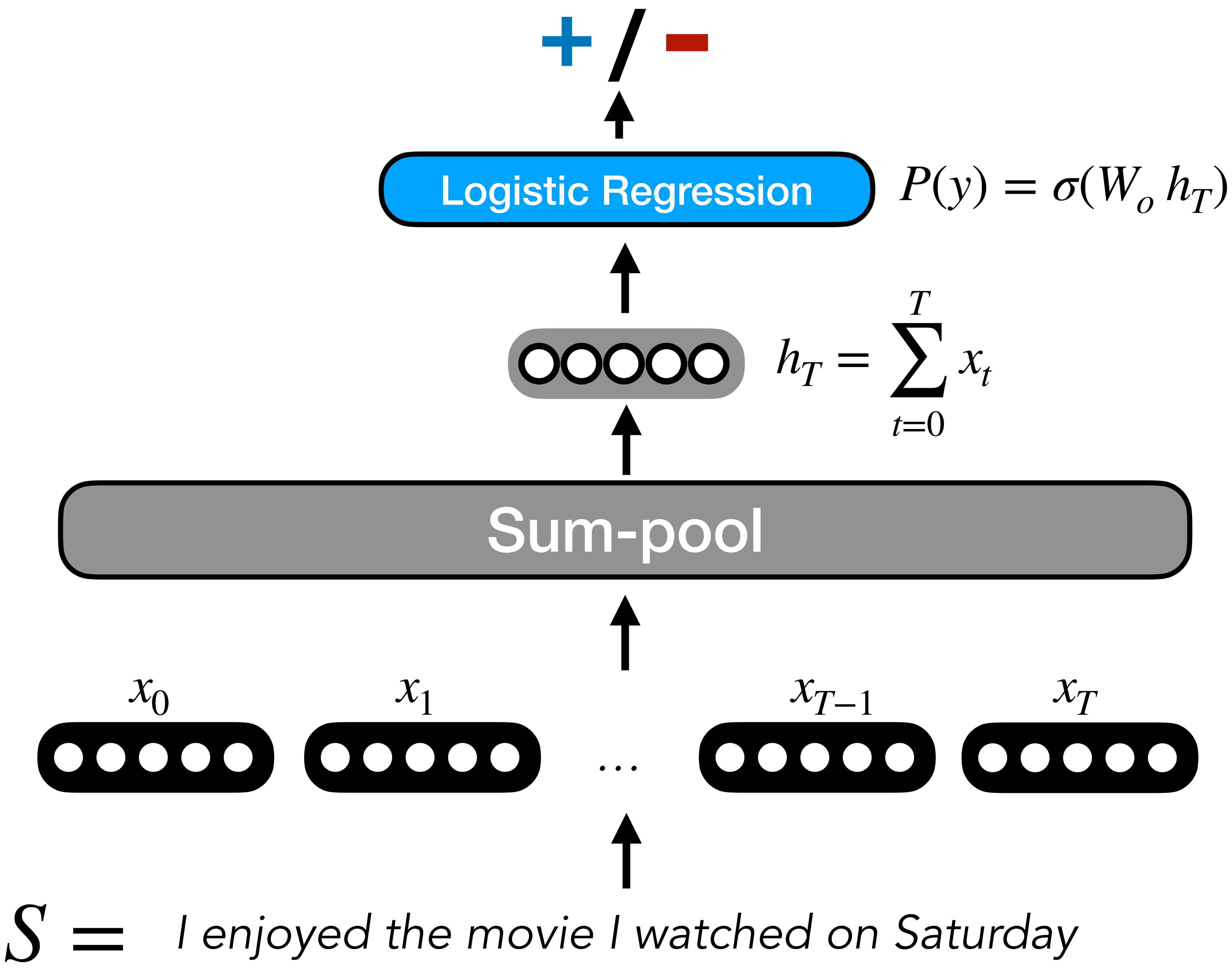


$S =$ *I enjoyed the movie I watched on Saturday*

Comprehension Questions

- What are the learnable parameters in our system?
- How many **unique** embeddings are in X for this example sentence S
- How many **unique** embeddings are in $\mathbb{E}$?

Vocabulary size V



Recap

- **Words and other tokens become vectors; no longer discrete symbols!**
- In simplest NLP System:
  - Define a vocabulary of words (or token types) *V* that our system can assign to a vector
  - Define a model that composes these vectors (or embeddings) of words into some sequence representation
  - A classifier can map this representation to a set of labels to make a prediction
  - The prediction depends on the natural language task we are trying to accomplish
  - By learning to make these predictions, we learn better embeddings for the words in the sequences

Tomorrow

What could be a better way to learn word embeddings?

Self-supervised learning of word embeddings

References

- Shen, D., Wang, G., Wang, W., Min, M., Su, Q., Zhang, Y., Li, C., Henao, R., & Carin, L. (2018). Baseline Needs More Love: On Simple Word-Embedding-Based Models and Associated Pooling Mechanisms. *Annual Meeting of the Association for Computational Linguistics.*