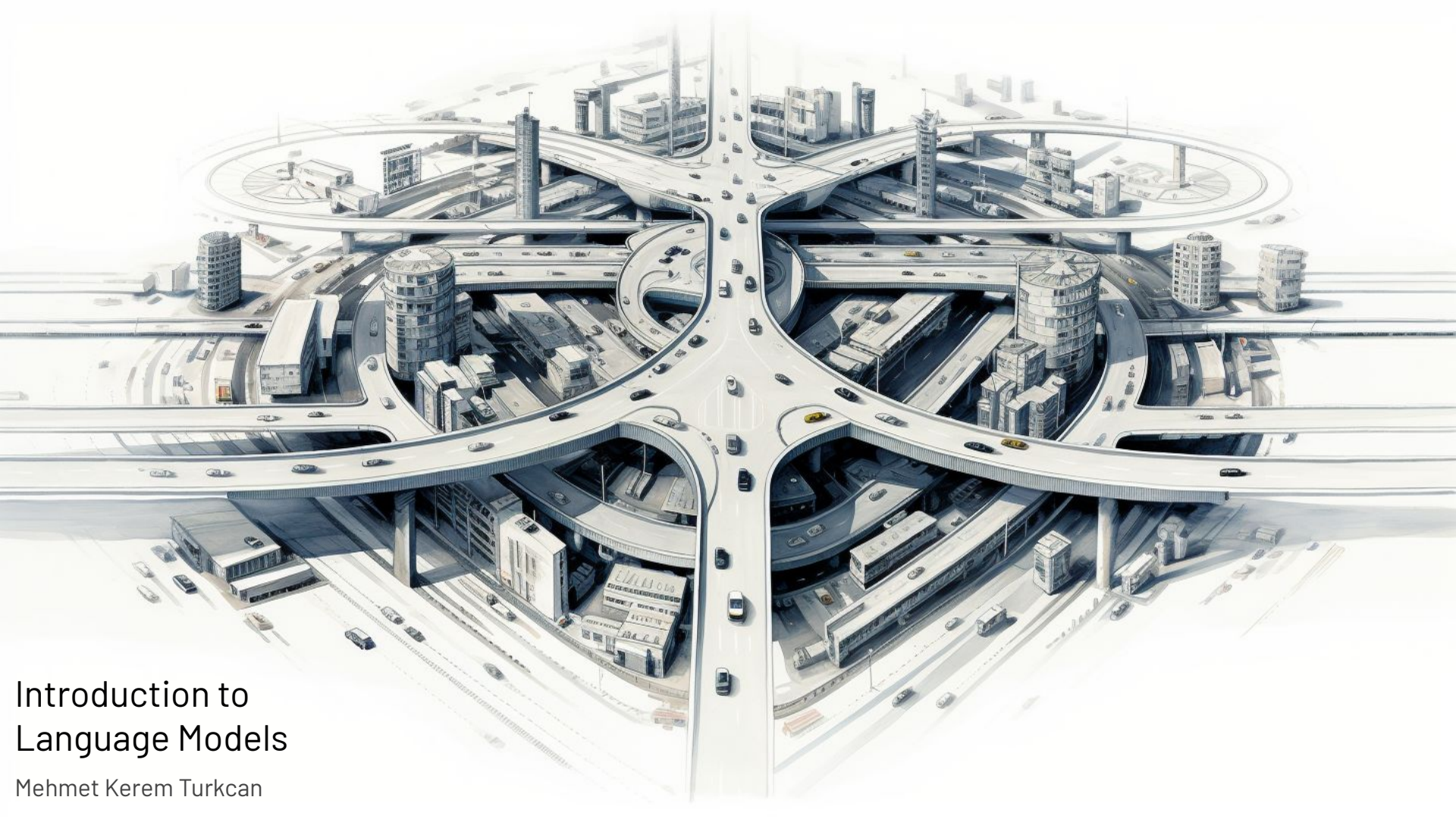
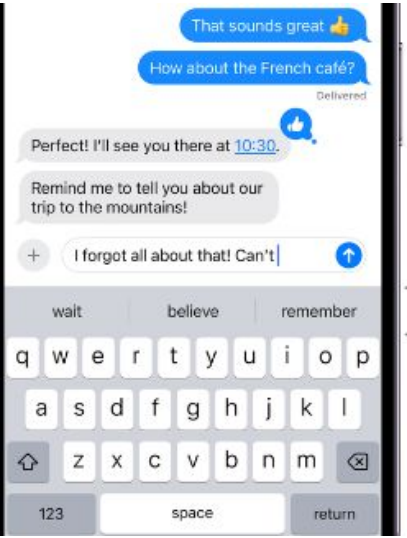


Introduction to Language Models

Mehmet Kerem Turkcan

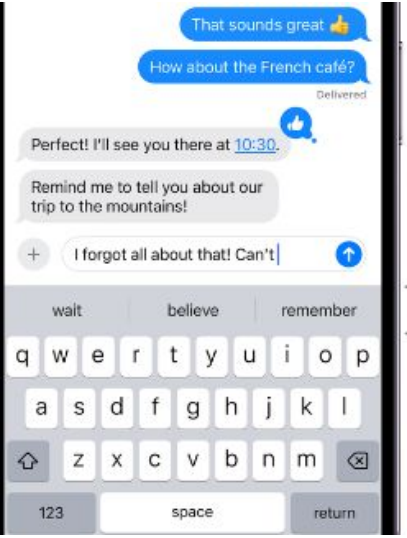
Language Models

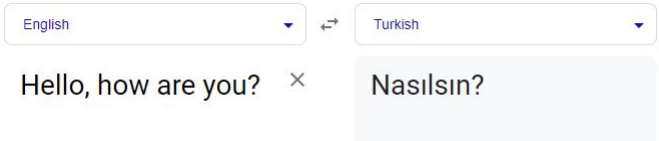
- **Fundamental Idea:**
  - Intelligent autocompletion of a sequence (for example, a text message)

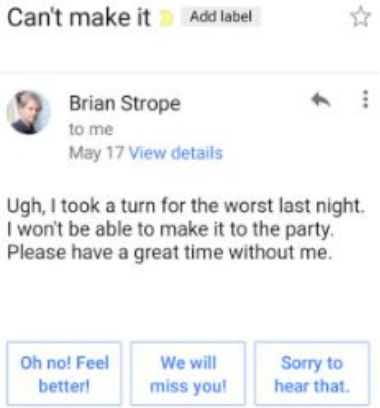


Language Models

- **Fundamental Idea:**
  - Intelligent autocompletion of a sequence (for example, a text message)
  - The “autocompletion” might have a “pause” in between, corresponding to a “change”



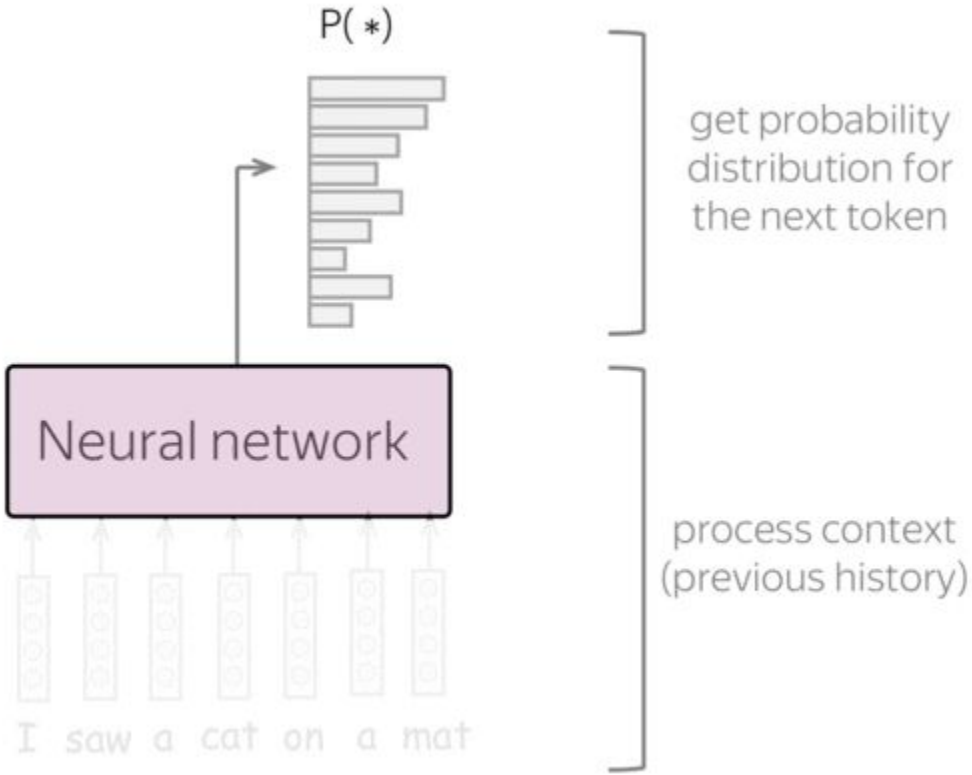




Language Models

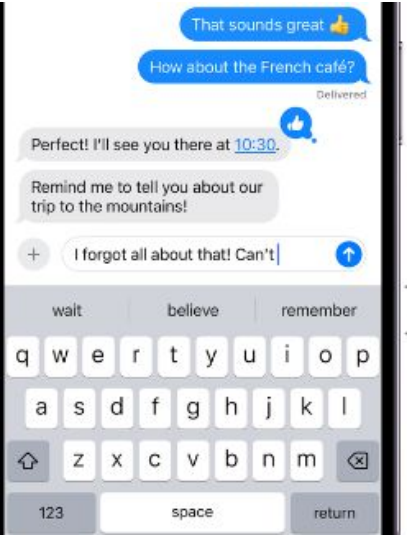
- **Fundamental Idea:**
  - Intelligent autocompletion of a sequence (for example, a text message)
- **Approach:**
  - Split a sequence into “tokens” using a “tokenizer”
    - For text, tokens could be words, single letters, syllables, or *made-up syllables that minimize the total number of “syllables” in a collection of documents*
    - For images, tokens could be “patches” of an image
    - For videos, tokens could be individual video frames
    - For audio, tokens could be 1-second sections from the audio
  - Use a neural network to answer the following question: Given a series of tokens, what is the probability for each token to be the next?

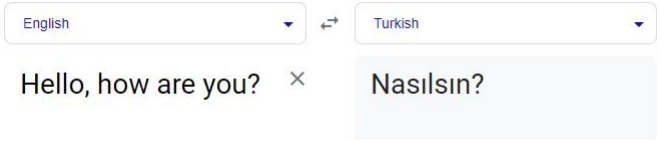
Language Models

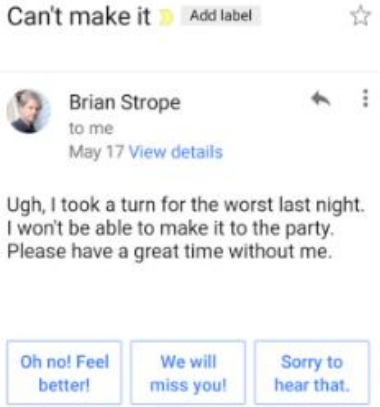


Language Models

- **Fundamental Idea:**
  - Intelligent autocompletion of a sequence (for example, a text message)
  - The “autocompletion” might have a “pause” in between, corresponding to a “change”
    - **For translation and chatting, add a special token**

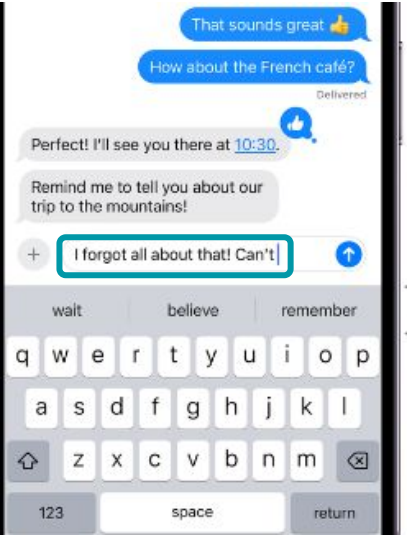


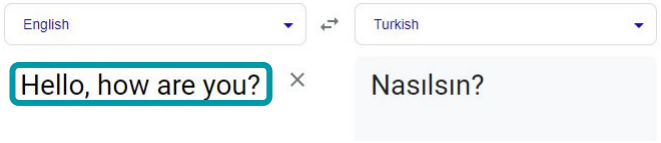


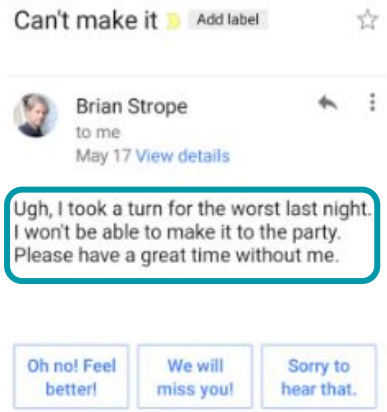


Language Models

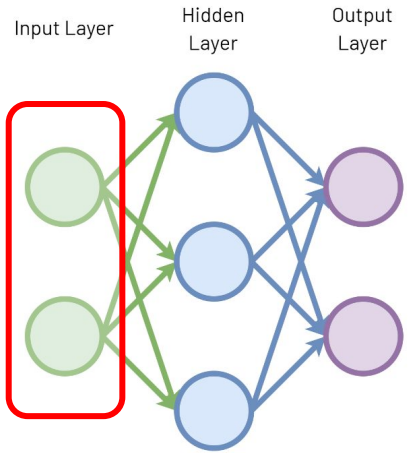
- **Fundamental Idea:**
  - Intelligent autocompletion of a sequence (for example, a text message)
  - The “autocompletion” might have a “pause” in between, corresponding to a “change”
    - **For translation and chatting, add a special token**
  - **Context:** What our model sees, which determines how far back it can observe





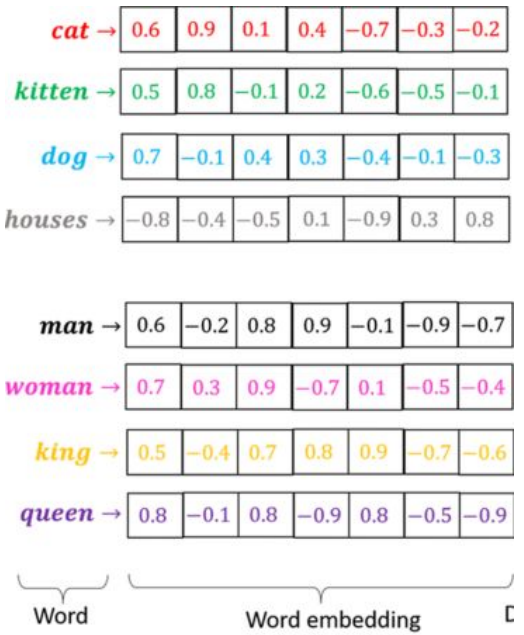


OK, but how do I calculate the inputs from tokens?

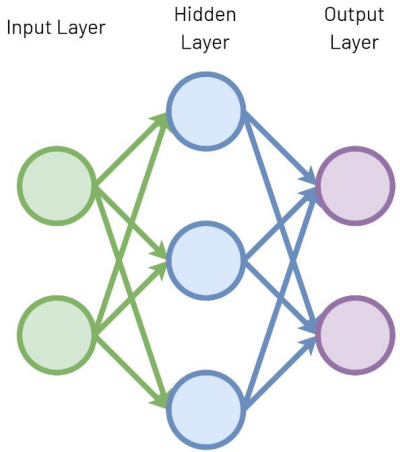


OK, but how do I calculate these from syllables?

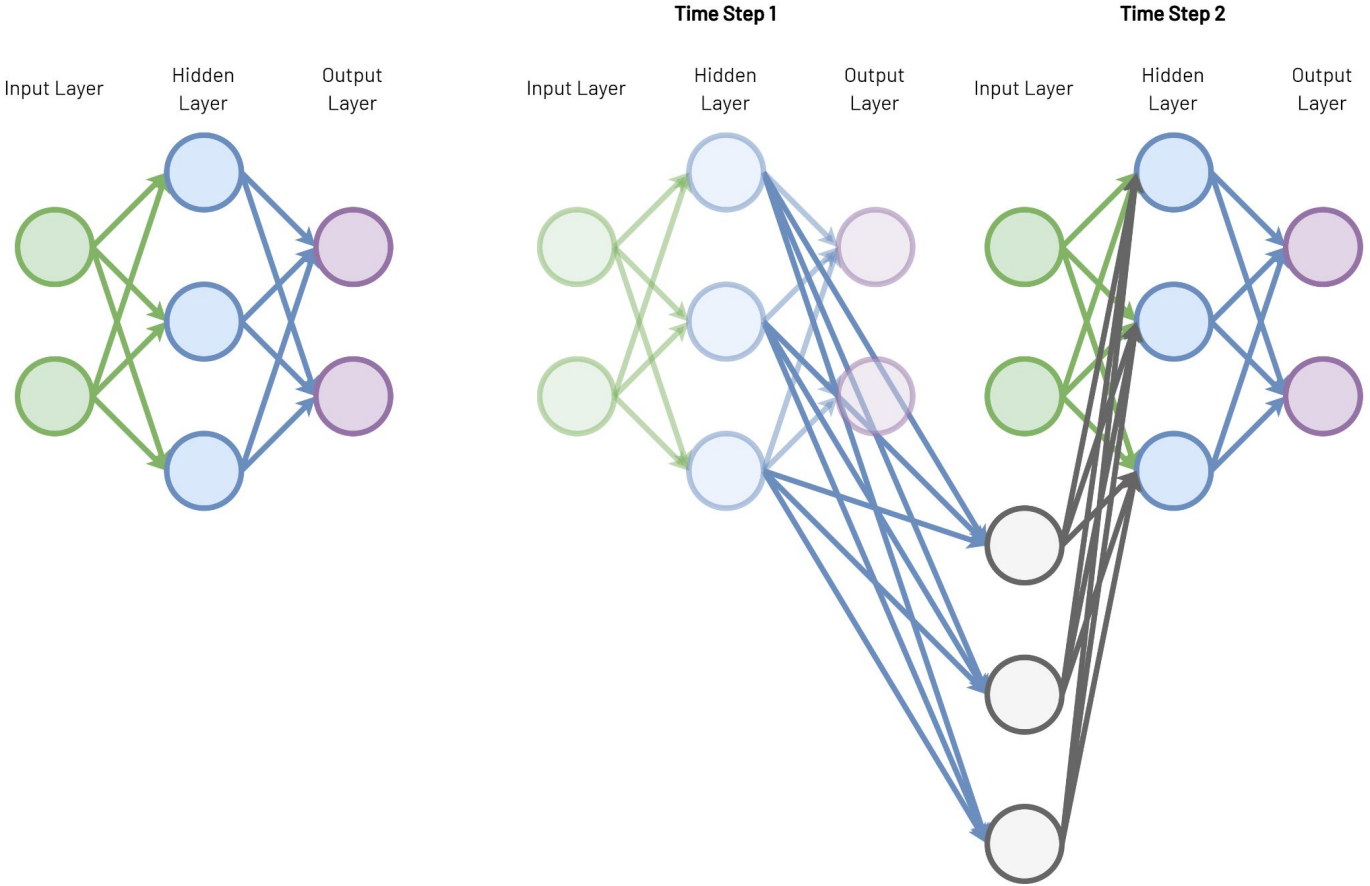
- **Fundamental Idea:**
  - We imagine tokens to correspond to some collection of numbers (a vector), corresponding to the "input layer" to the model. We update these vector as if they were another hidden layer! We call them "embeddings"
  - Think of this as a dictionary where we look up for a hidden layer for each token, instead of using an input that was given to us
    - For each vector, we can find the closest token in our dictionary to "decode" a vector into a token



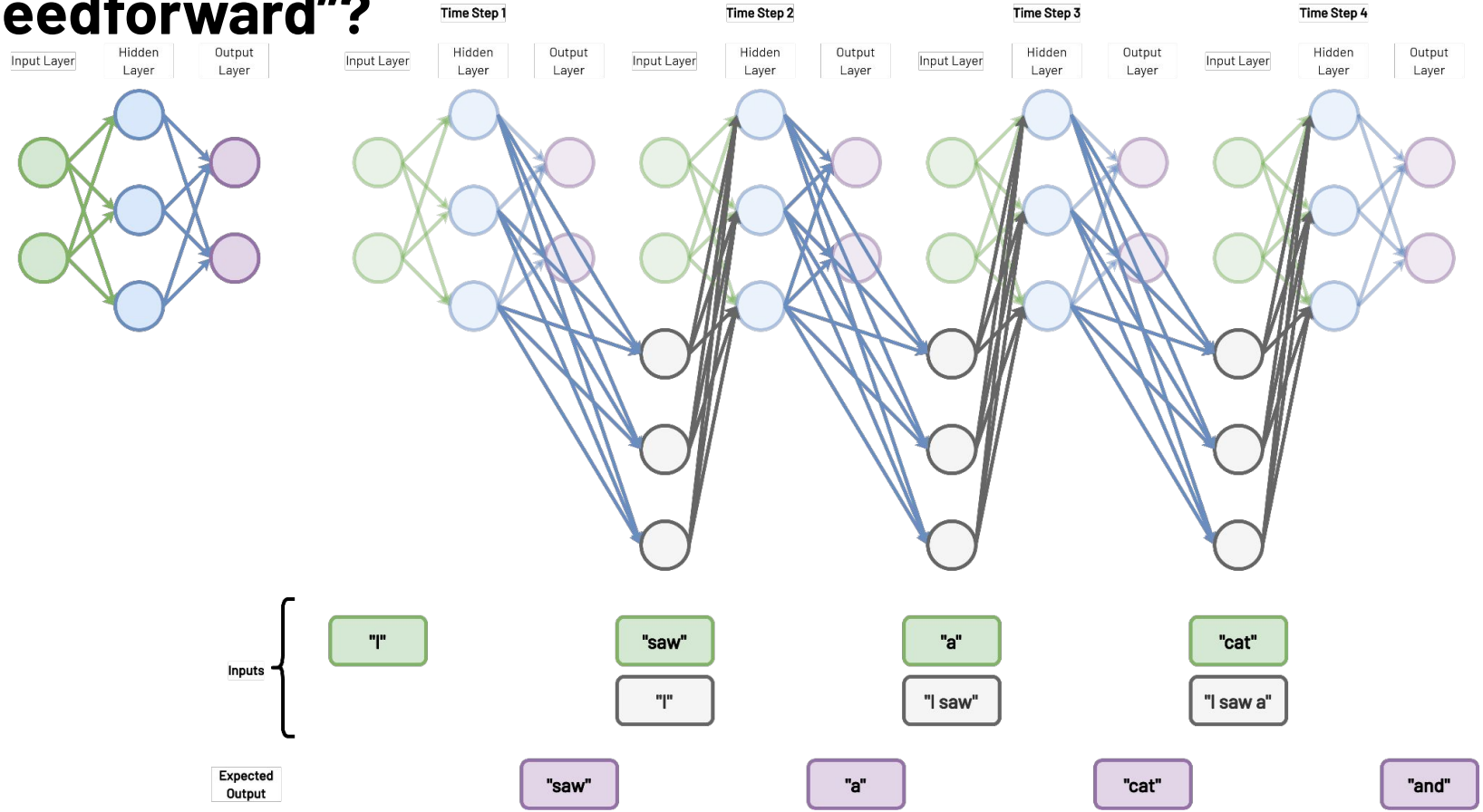
How are “Recurrent” Models Different from “Feedforward”?



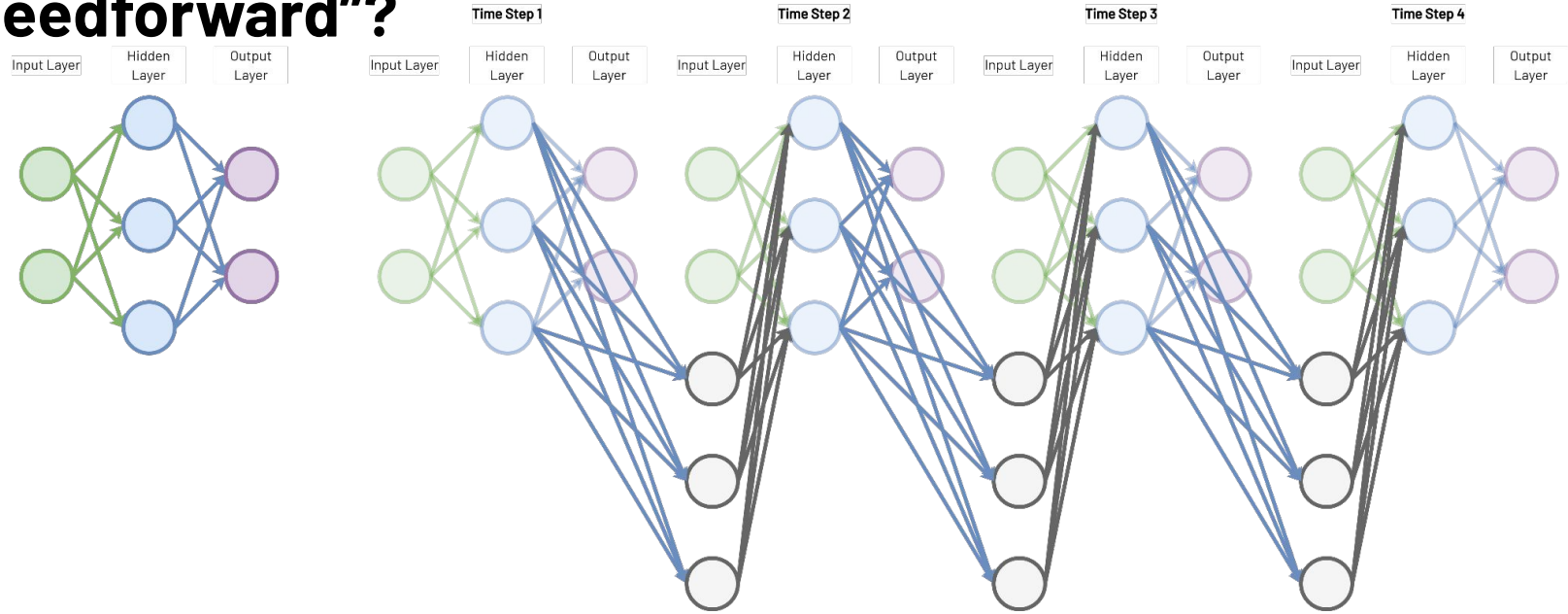
How are “Recurrent” Models Different from “Feedforward”?



How are “Recurrent” Models Different from “Feedforward”?

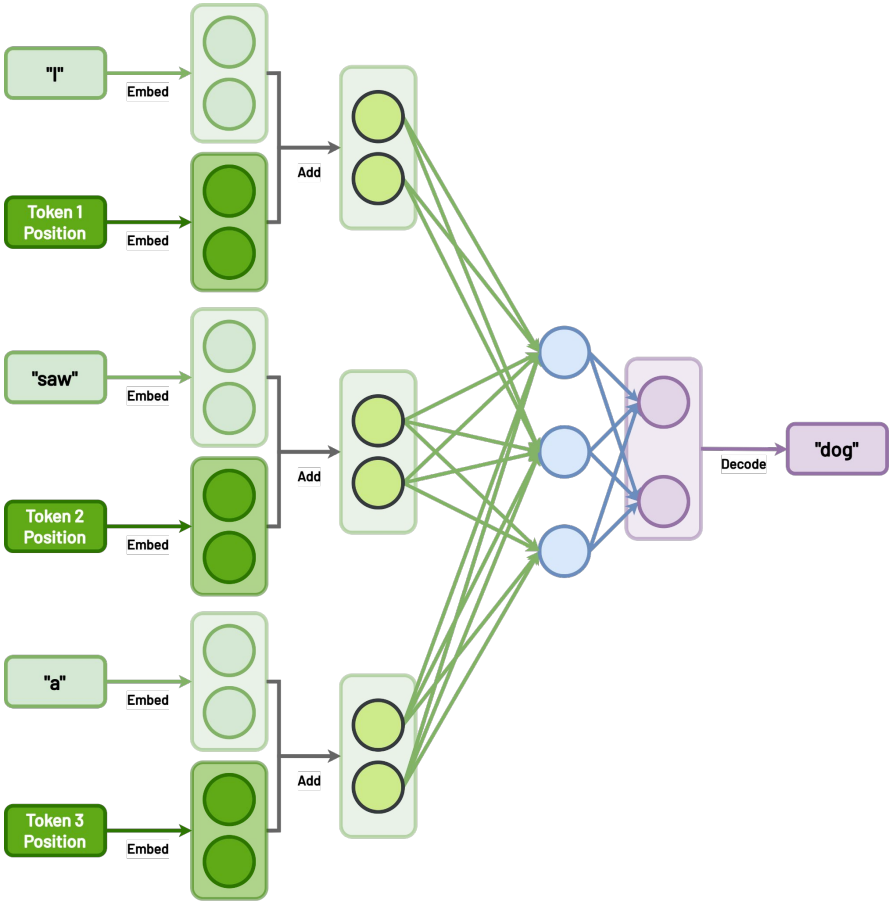


How are “Recurrent” Models Different from “Feedforward”?



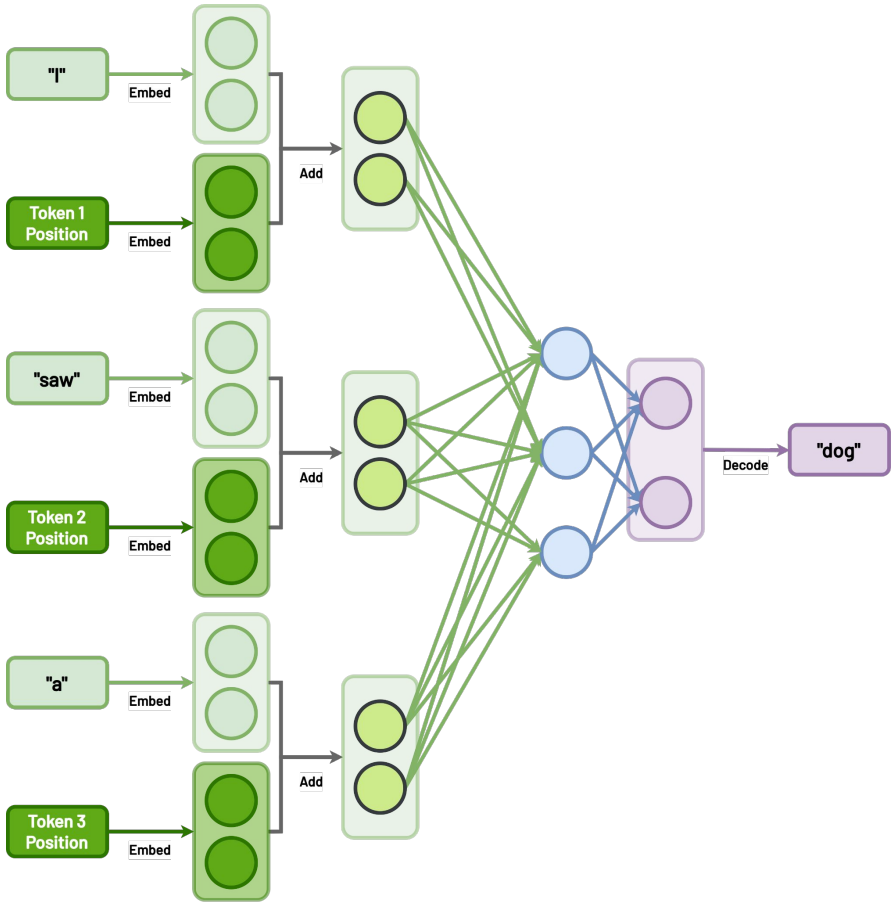
“Fake” Sequential Models

- **Fundamental Idea:**
  - We have converted tokens to vectors, so why don’t we convert time to vectors as well?
  - Then we can forget about causality!
  - Large language models follow this idea

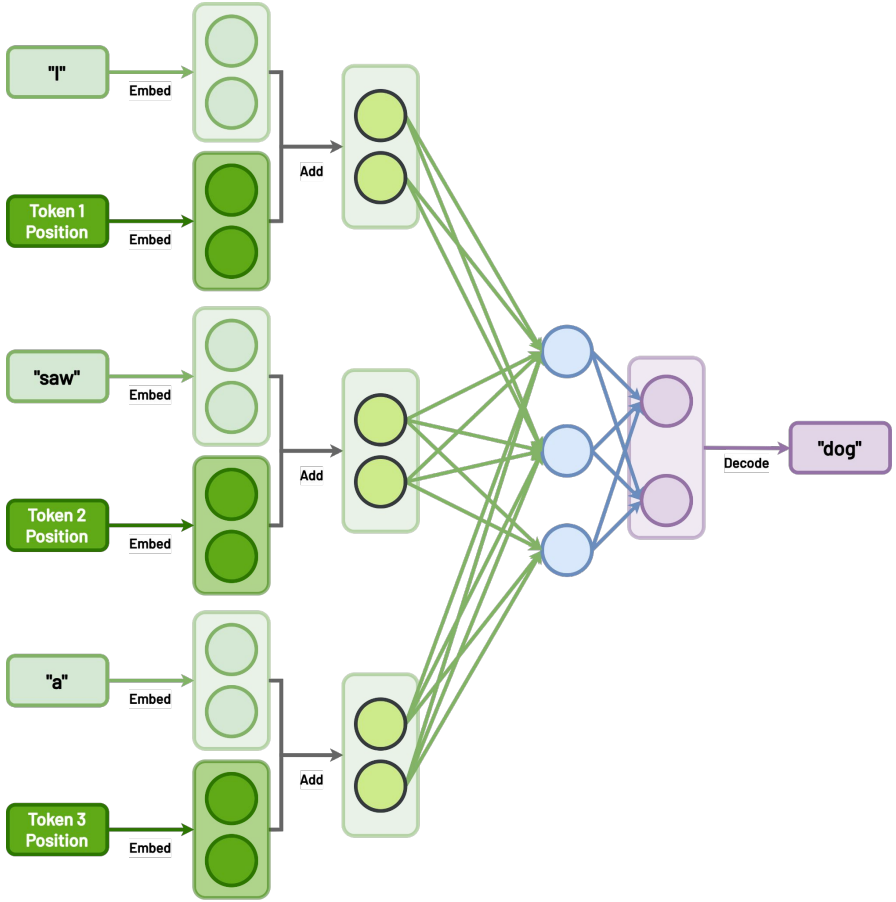


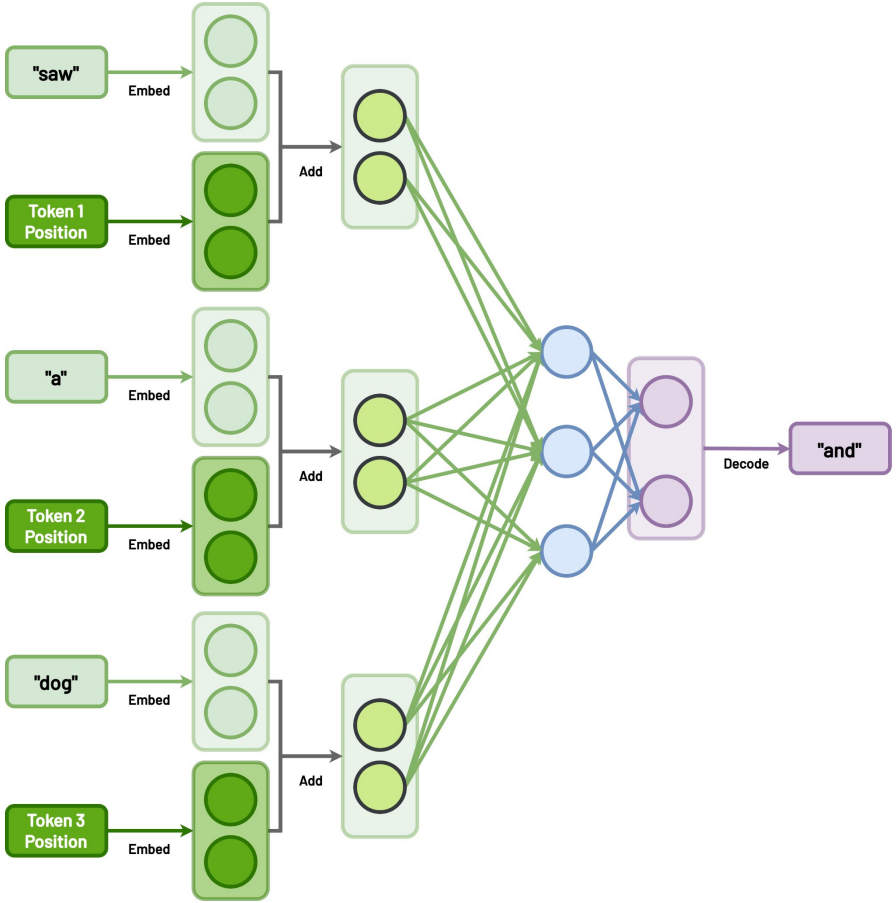
“Fake” Sequential Models don’t have to deal with this

- **Approach:**
  - Have a dictionary that maps time steps to vectors (positional embeddings)
  - Embed tokens
  - Embed “positions”
  - Add vectors of tokens with vectors of positions
  - Train a neural network to predict the next word
  - Shift input tokens by 1, add the prediction as if it was part of the start “context”



“Fake” Sequential Models don’t have to deal with this





“Language” Models

- **Important to Know**
  - Images can also be modeled this way!