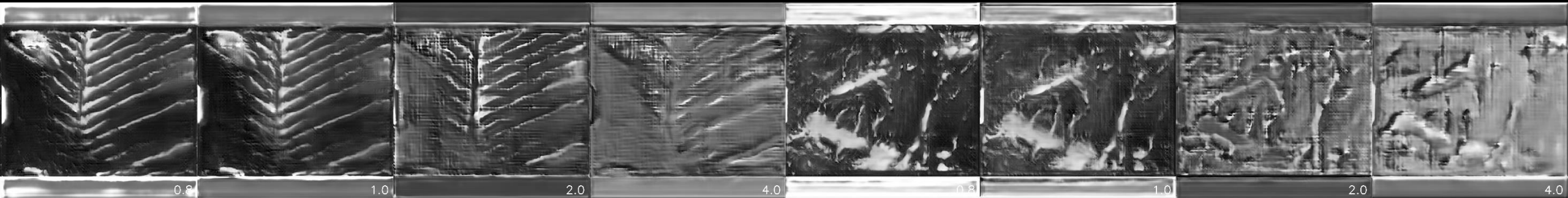
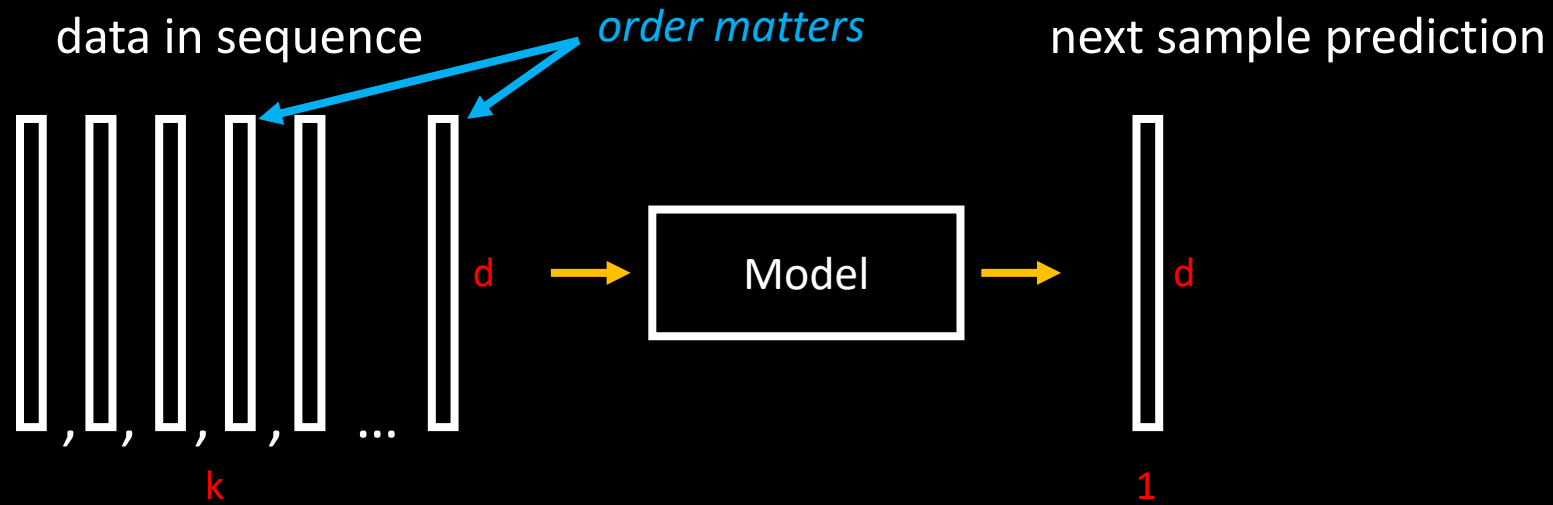


Exploring Machine Intelligence

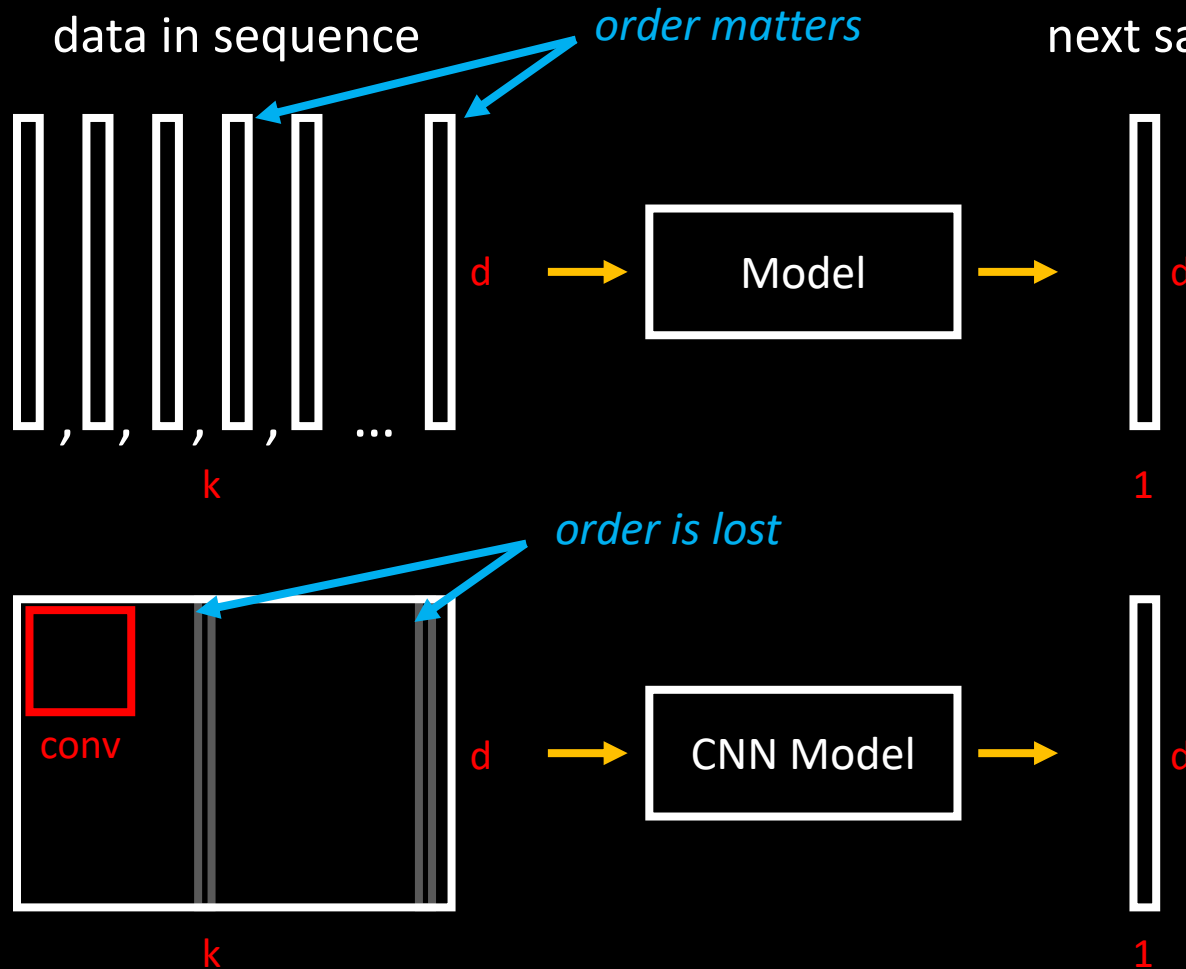
Week 7, Sequential Modelling



Motivation for today



Motivation for today



We could use any model we talked about (*CNN*, *fully connected NN*) to map this data – but there is a **type of models which is designed with sequential data in mind** and maps this sequentiality better.

Today

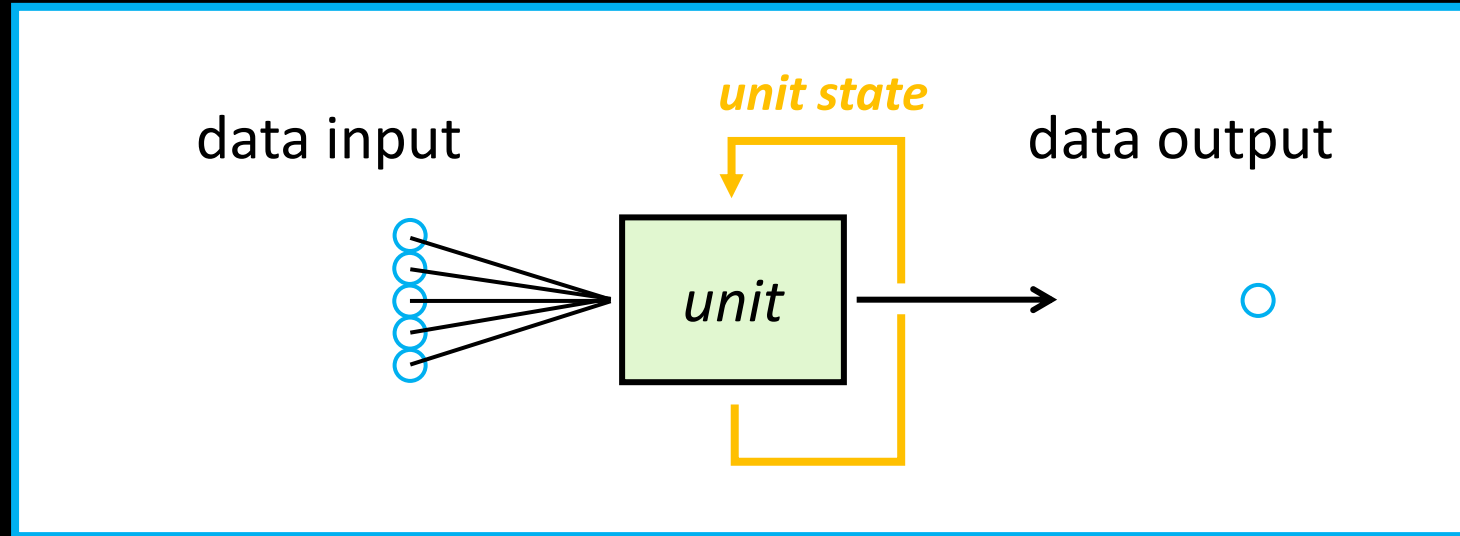
Sequential Modelling

- Creating networks from **units with memory**
- Plugging in **textual and musical data**

Practical session:

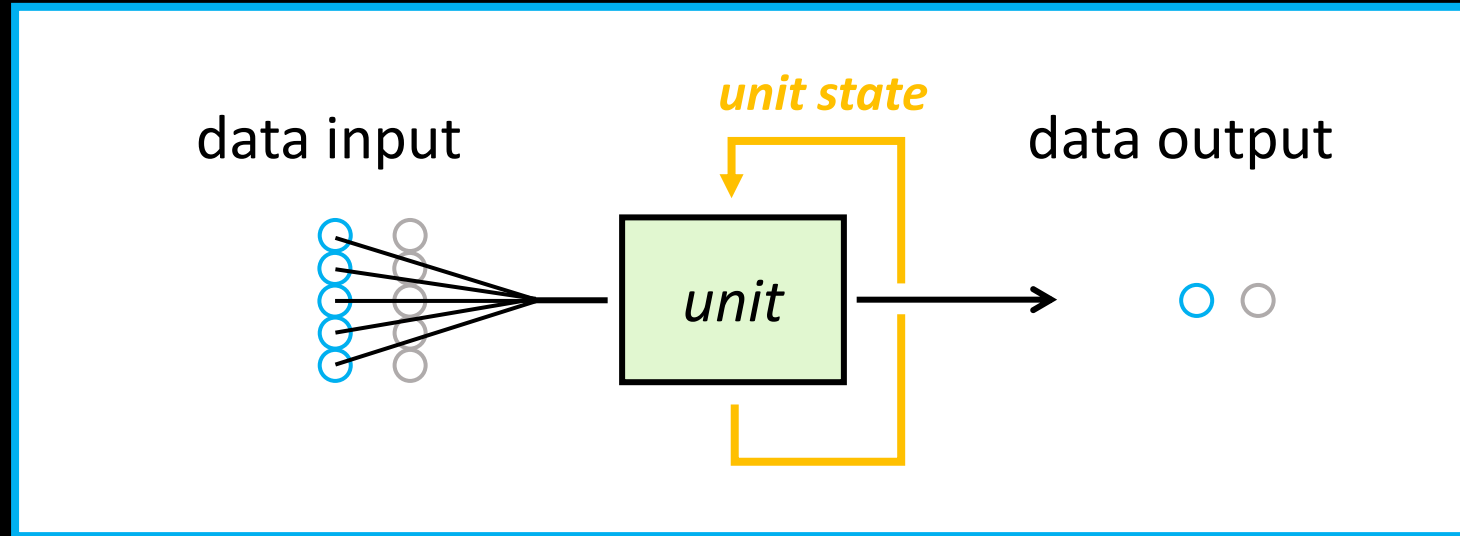
- **Q&A session** for the Final Exam

Unit with a memory



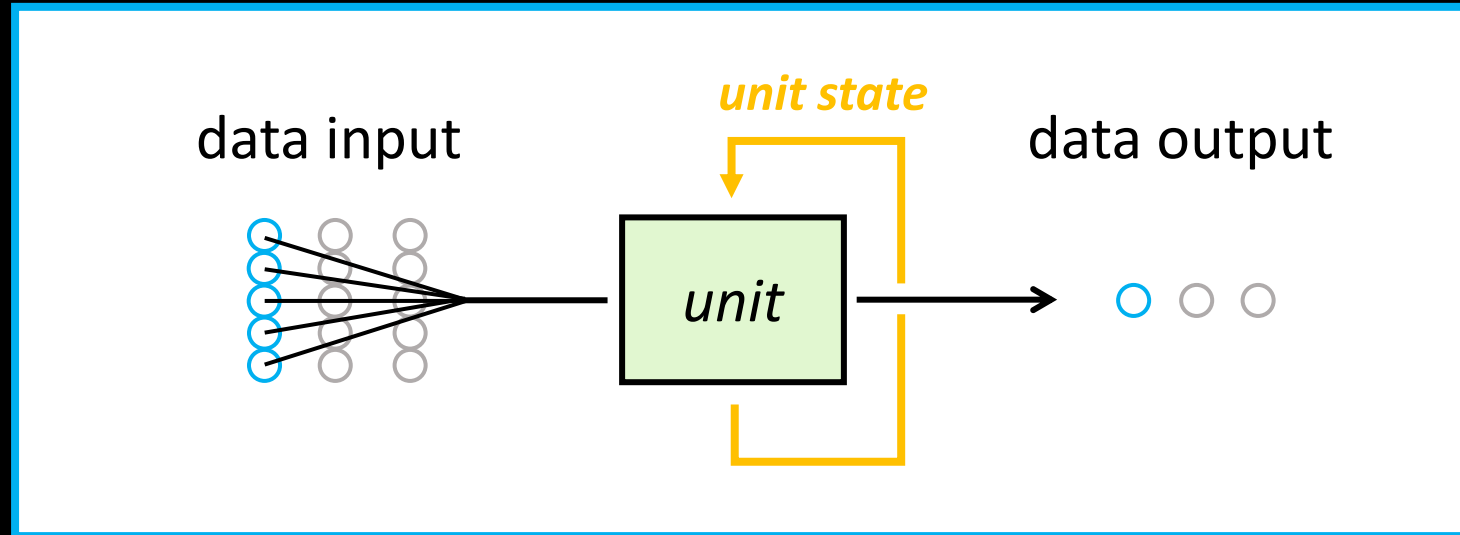
- Not only we **produce some output from the input**, we also **alter the unit state** (**memorize** some data from the next input in sequence)
- How both of this happens is something we learn from the structure of data

Unit with a memory



- Not only we **produce some output from the input**, we also **alter the unit state** (**memorize** some data from the next input in sequence)
- How both of this happens is something we learn from the structure of data

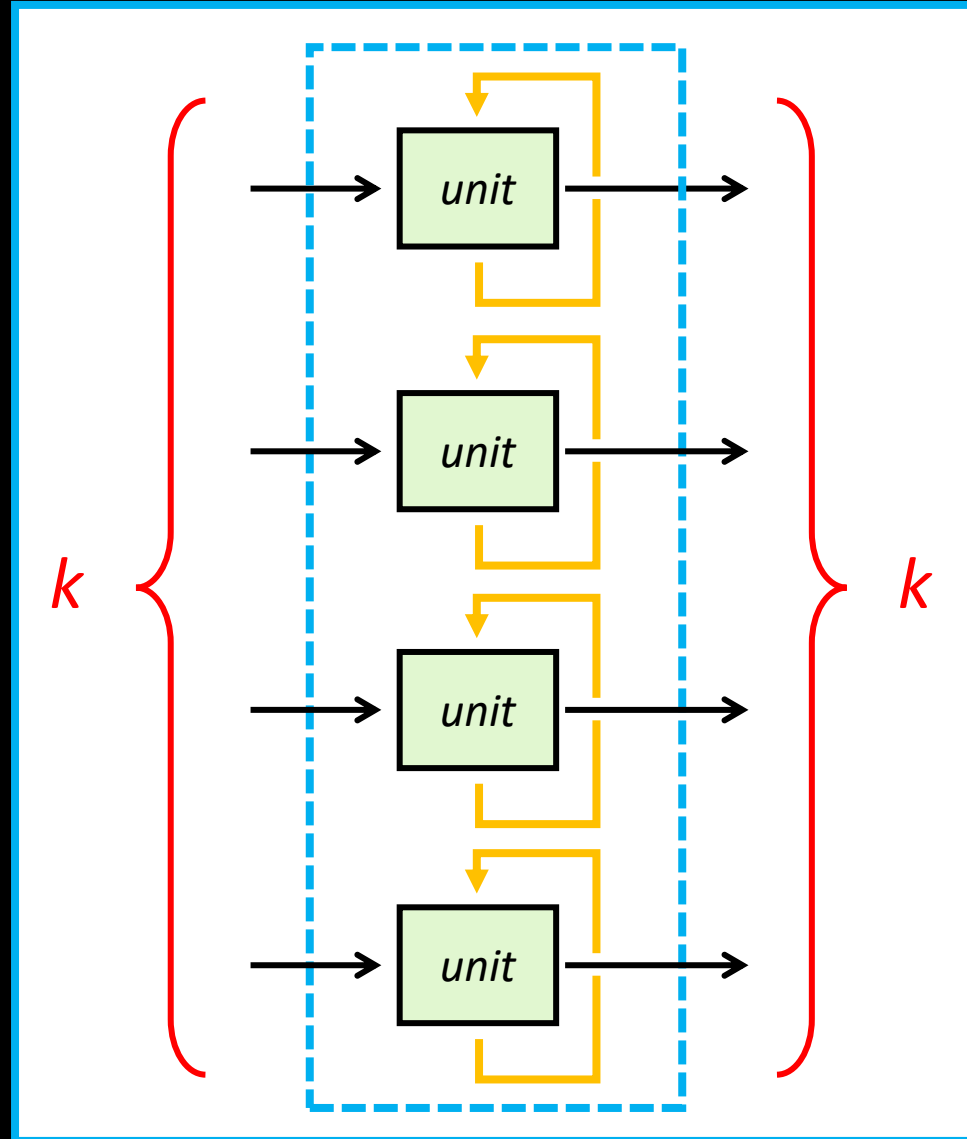
Unit with a memory



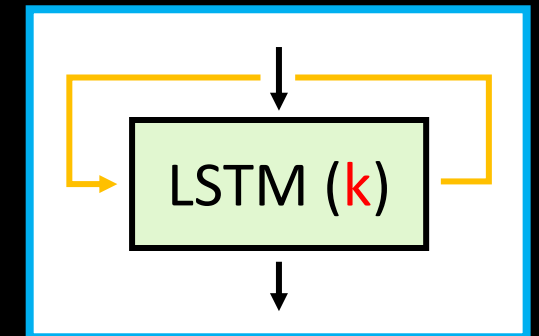
- Not only we **produce some output from the input**, we also **alter the unit state** (**memorize** some data from the next input in sequence)
- How both of this happens is something we learn from the structure of data

Recurrent layer

- These **recurrent units** with memory can be put into **layers**

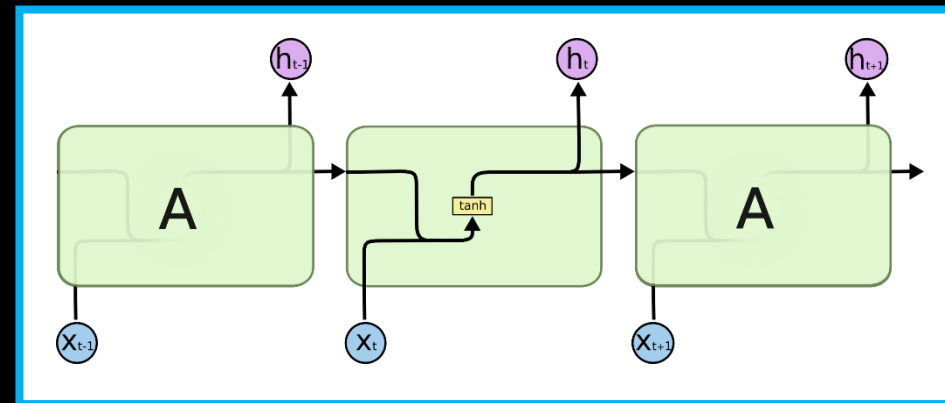


- Single **LSTM layer** with k units is usually simplified as:



Unit types: RNNs and LSTMs

- **Recurrent Neural Networks (RNN):**
 - Simpler unit design

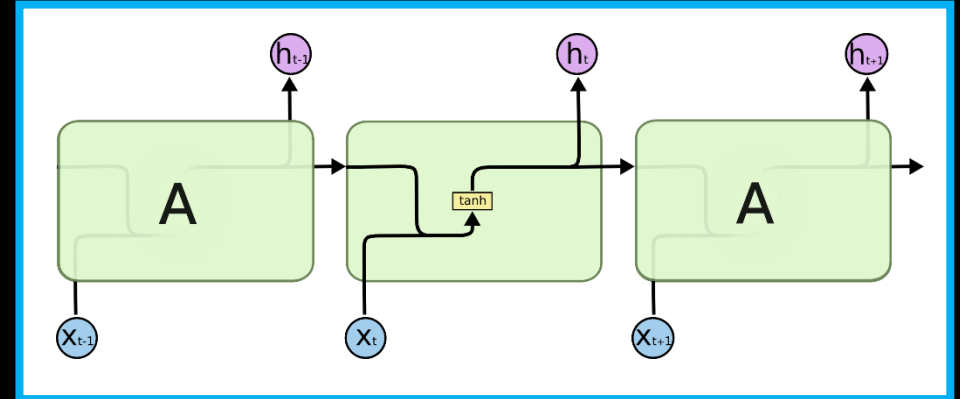


“I grew up in *France*... I speak fluent *French*.”

Unit types: RNNs and LSTMs

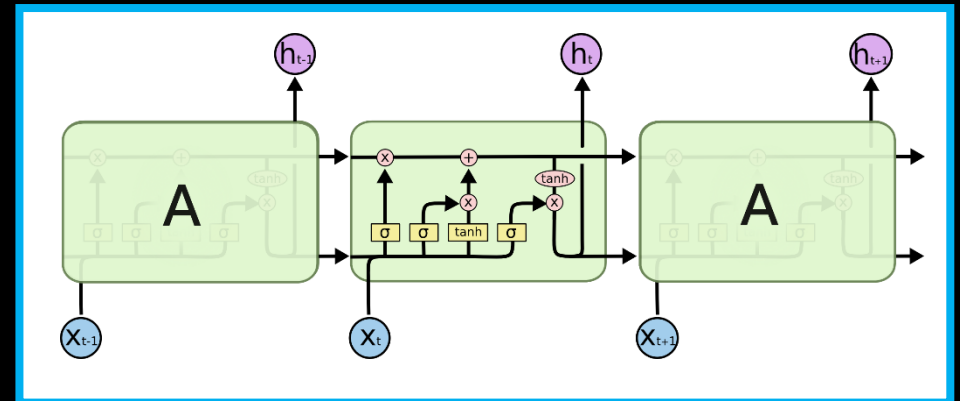
- **Recurrent Neural Networks (RNN):**

- Simpler unit design



- **Long-Short Term Memory (LSTM):**

- More complex unit design, made to **remember longer dependencies** inside the data



“I grew up in *France*... I speak fluent *French*.”

^ forget and input gates controlled by learned parameters (hence needs more of them)

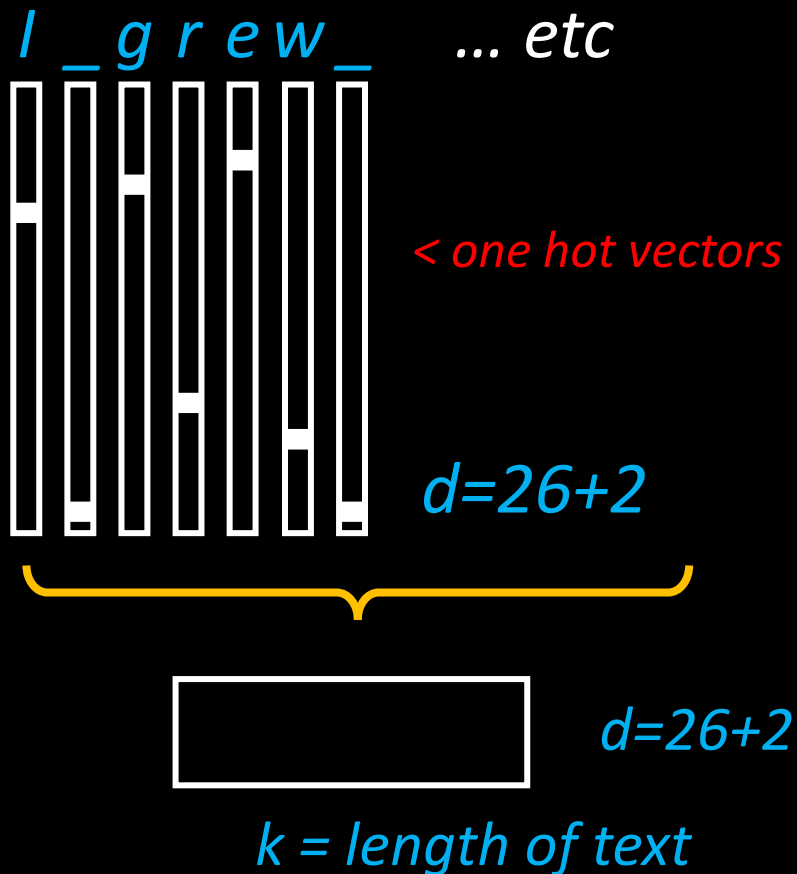
Plugging in text data

“I grew up in France... I speak fluent French.” *< original data*

- **Vectorize text using a dictionary**

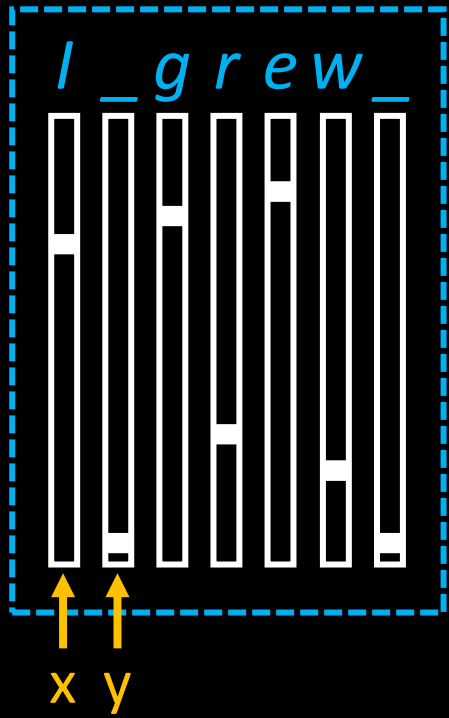
Plugging in text data

"I grew up in France... I speak fluent French." < original data



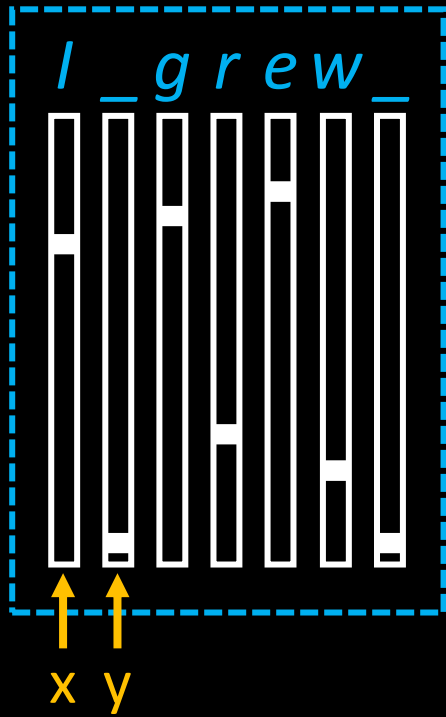
- **Vectorize text using a dictionary**
- Each letter is **encoded as a one-hot vector** using alphabet as dictionary (or full words using a word dictionary)
- We can **later decode the predictions** looking into that dictionary

dataset

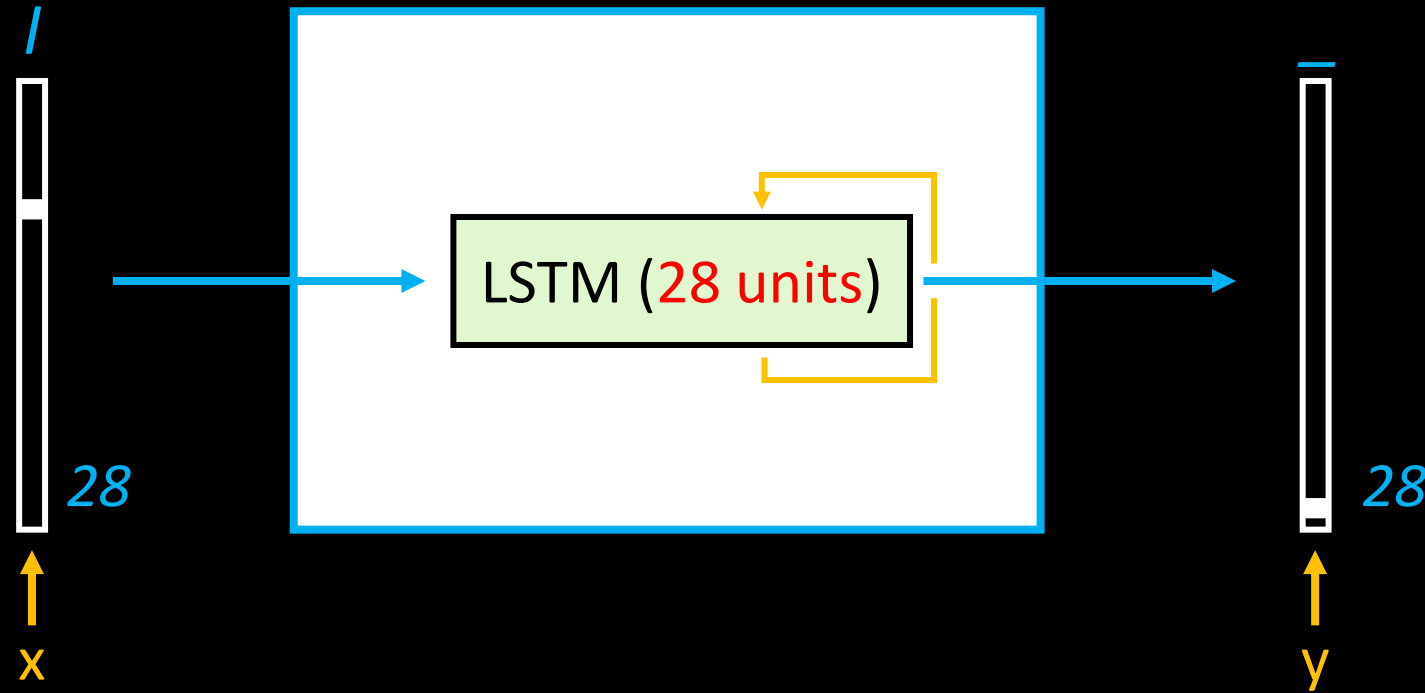


Simple model

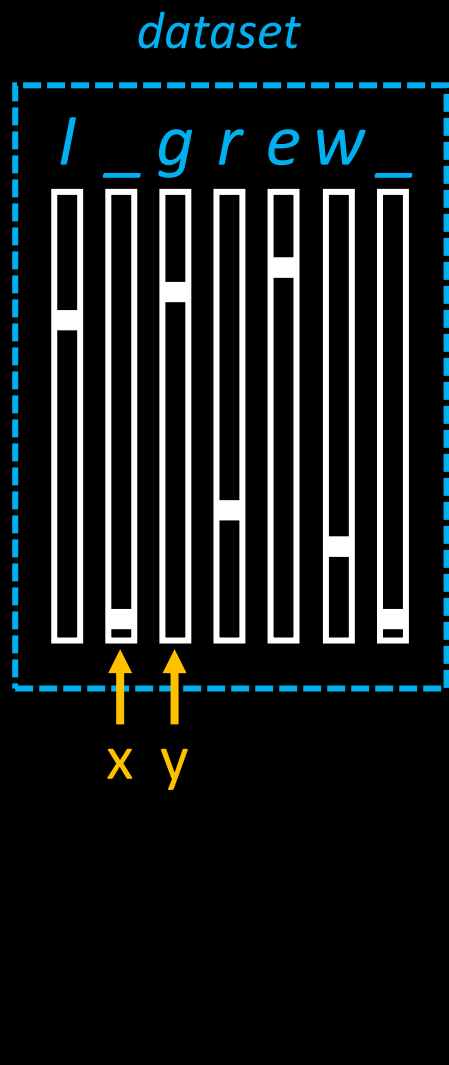
dataset



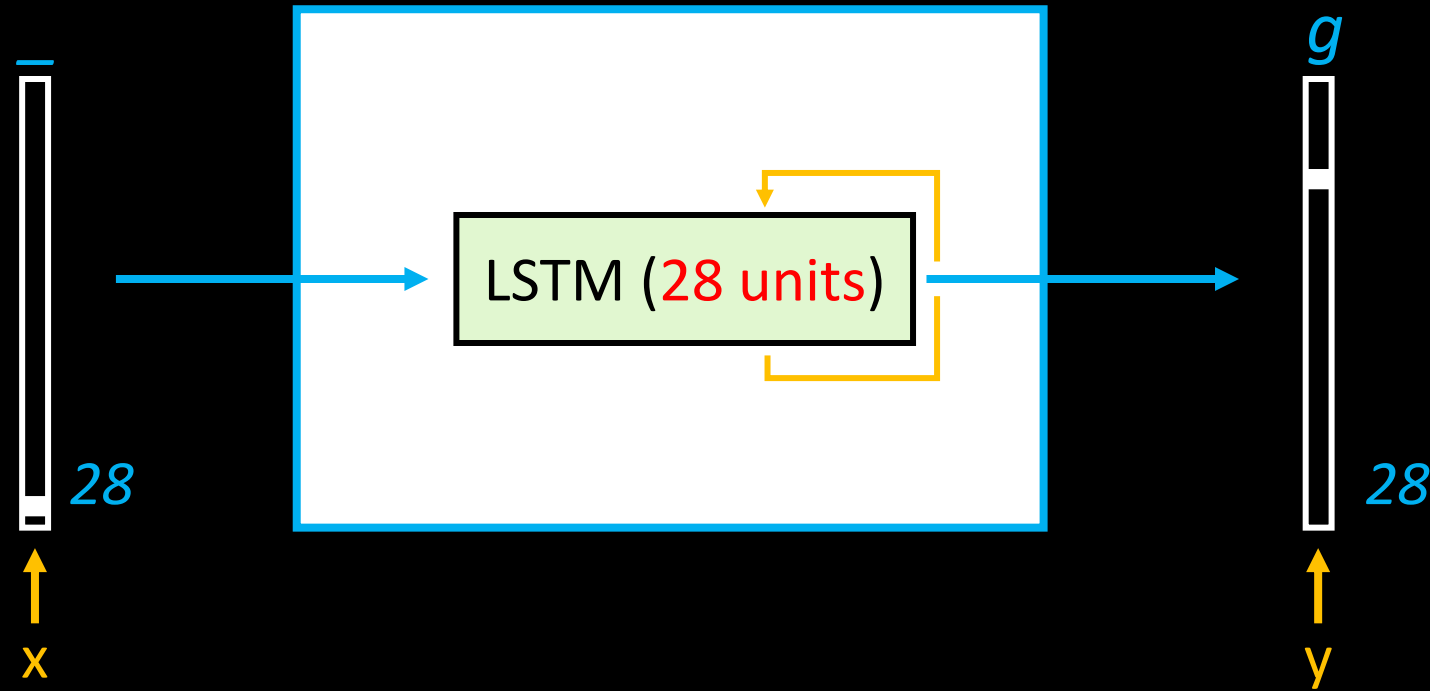
Simple model



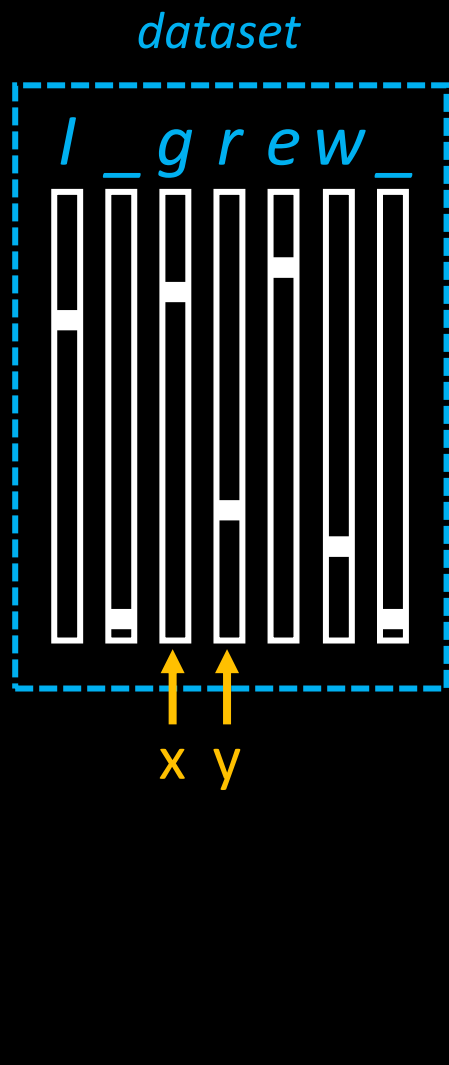
- We are **training the model** to **learn the transformation $x \rightarrow y$** while it also **learns what** is useful **to save** inside the 28 units (what preceded before this x).



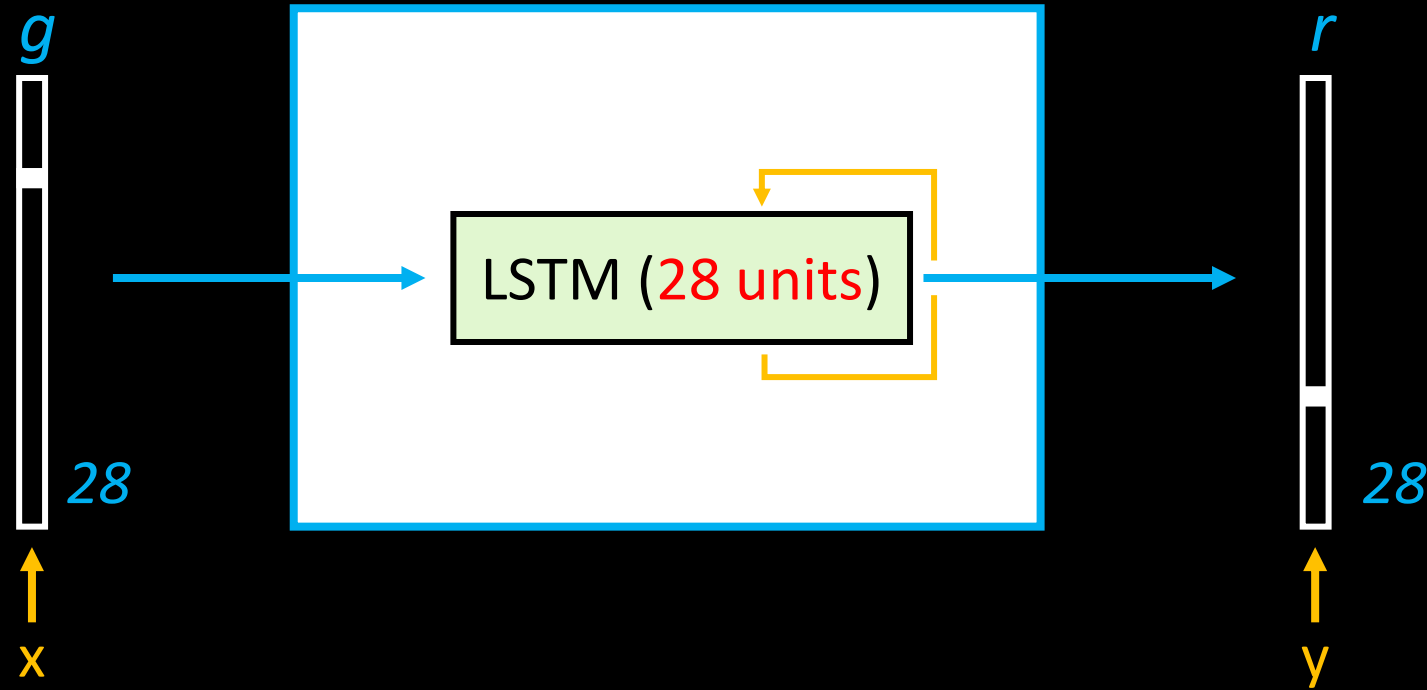
Simple model



- We are **training the model** to **learn the transformation $x \rightarrow y$** while it also **learns what** is useful **to save** inside the 28 units (what preceded before this x).

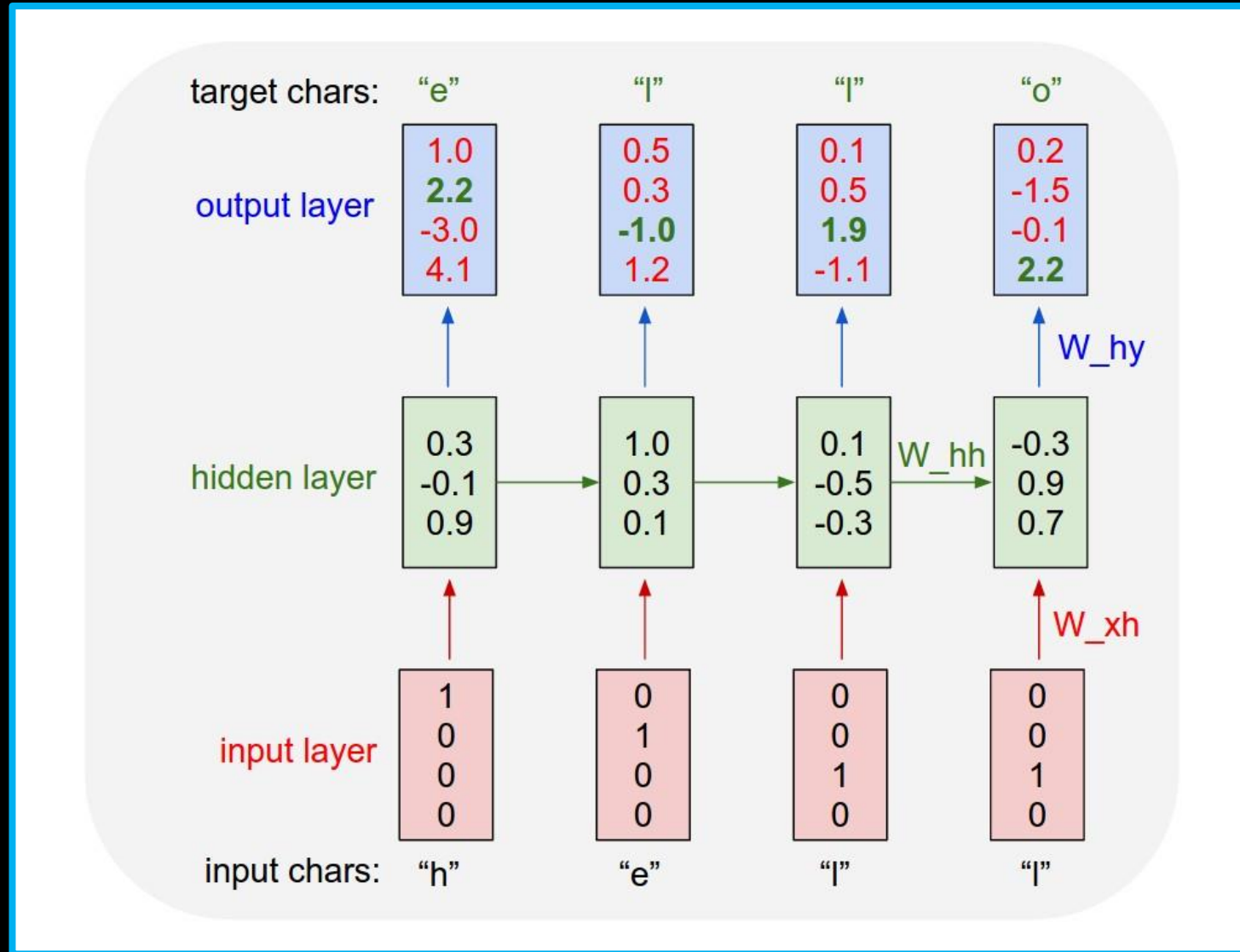


Simple model

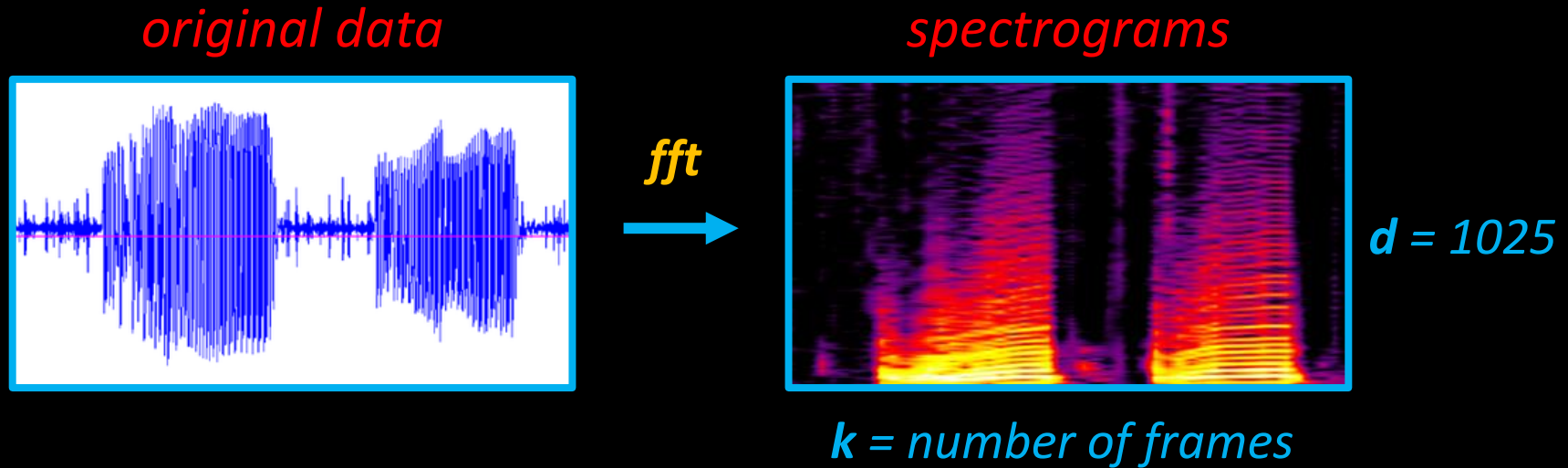


- We are **training the model** to **learn the transformation $x \rightarrow y$** while it also **learns what** is useful **to save** inside the 28 units (what preceded before this x).

Text data

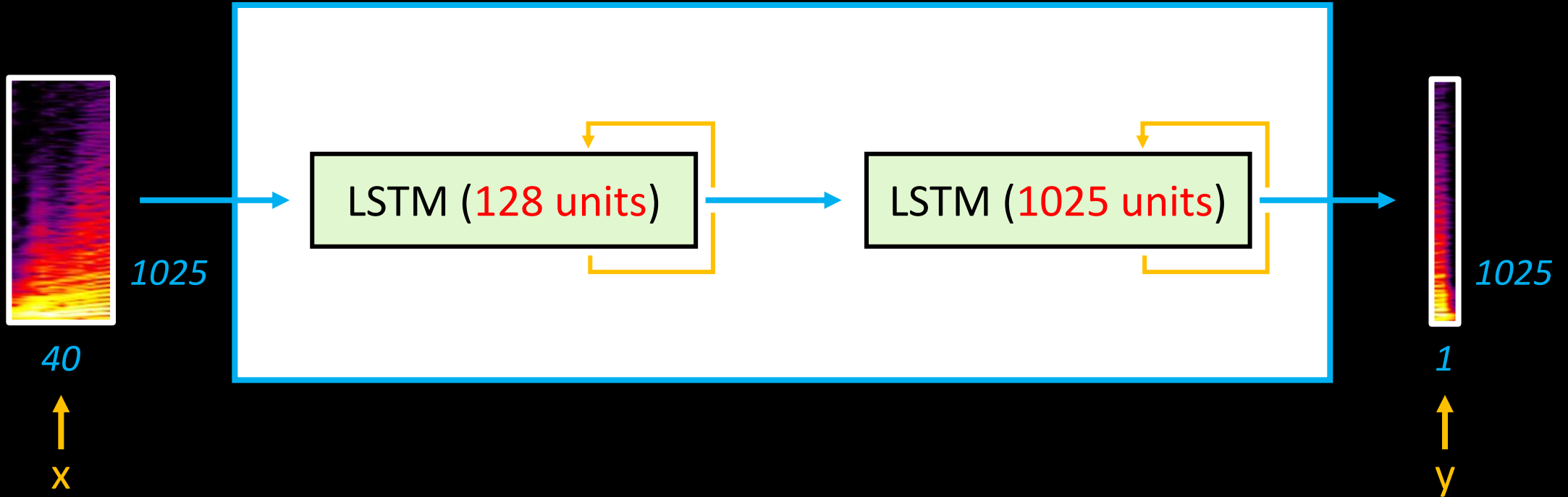


Plugging in music data



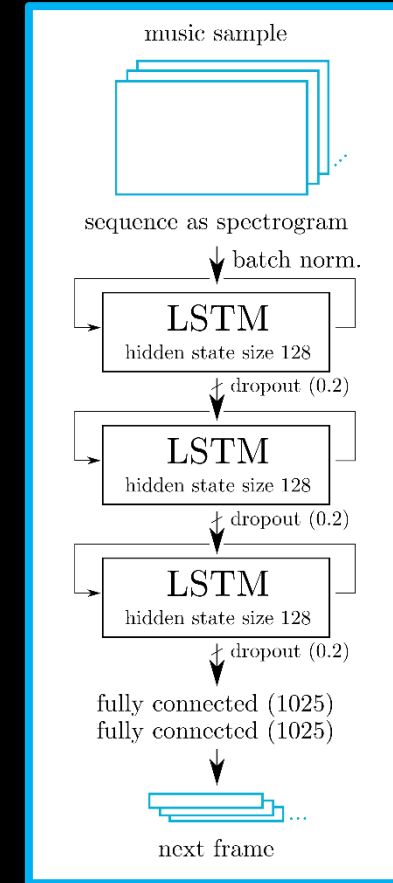
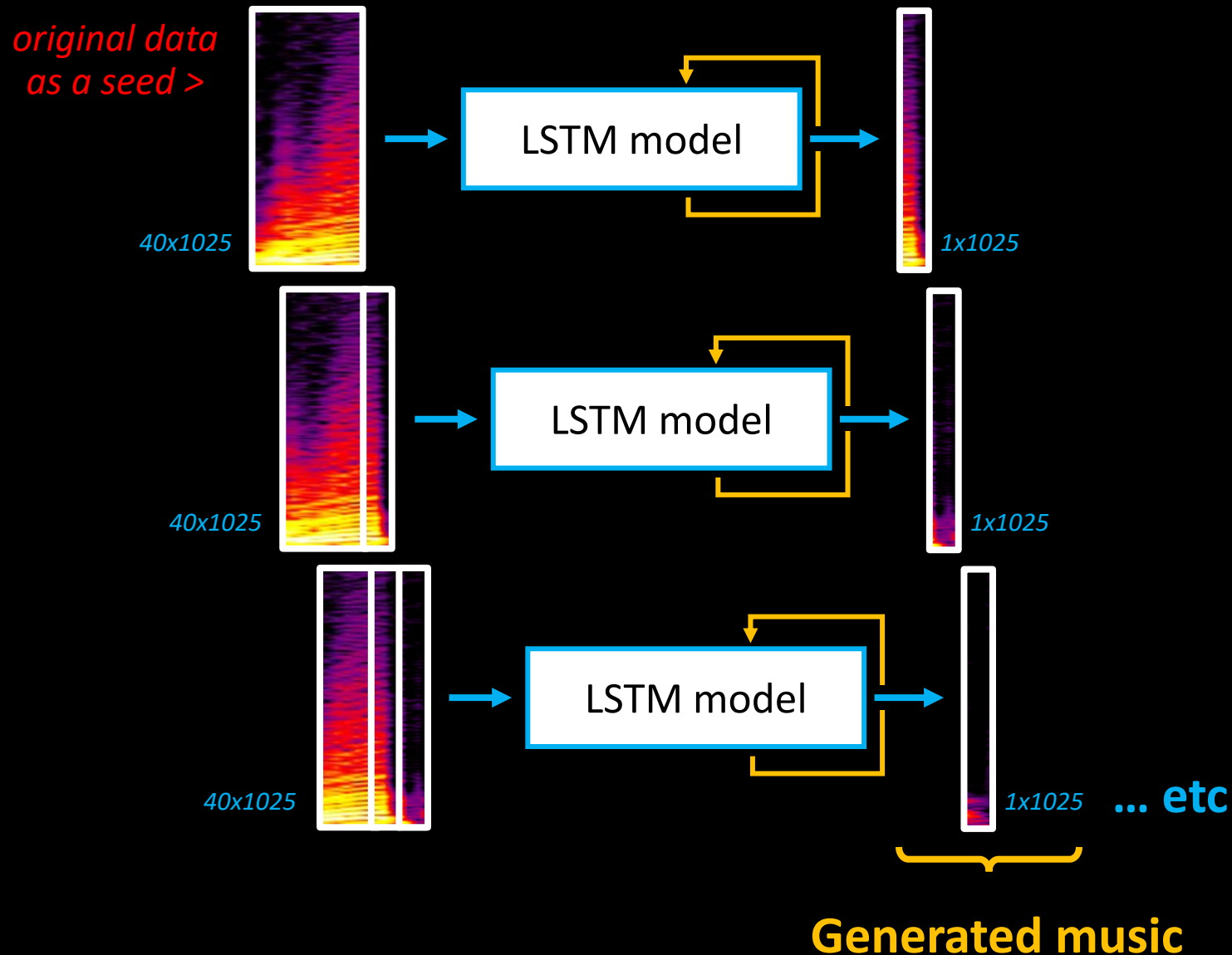
- **Encode music using the Fourier Transform** (*fft*) to get **spectrogram** (which can be considered as image representation)
- We can **later decode the predictions** using the *inverse fft*

Larger model



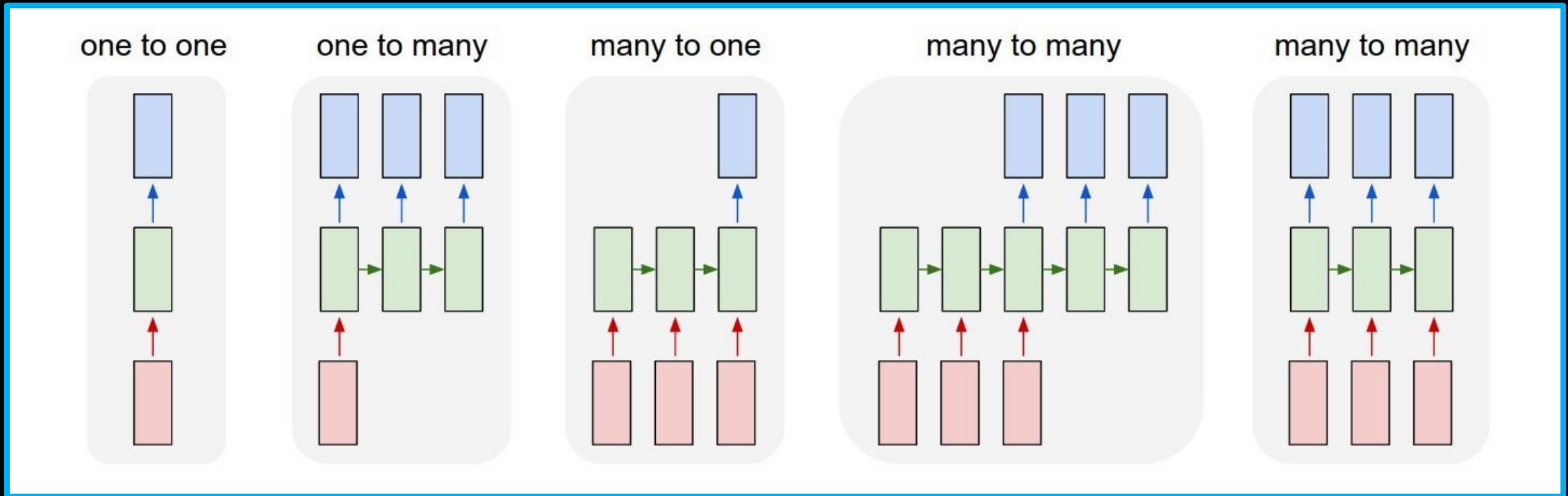
- We are **training the model** on a longer sequence of data (*“many to one”*), the model still **learns the transformation** of **x -> y**.

Using a trained LSTM model:



- Check the **generated music sample**: [ml-jazz-meanderings-ml-generated-sounds-1](#)

Types of models and data schemes



- The movement to the right on this illustration means **moving one timestep further** (next sample in the sequence)

Practicum live session

- This week's live session will be organized as a **Q&A session** about all the material that we went through in the class
- **Assessment requirements** details – [on our moodle page](#)

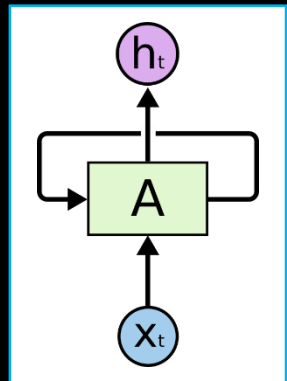
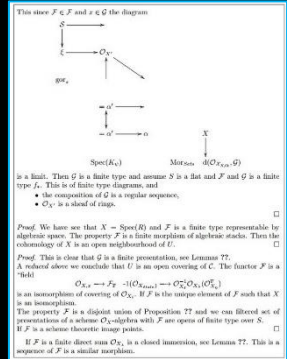
Links and additional readings:

- **Bonus readings:**

- **Andrej Karpathy blog** “The Unreasonable Effectiveness of Recurrent Neural Networks” – [rnn-effectiveness](#)
 - Citation: *There’s something magical about Recurrent Neural Networks (RNNs). ... This post is about sharing some of that magic with you. We’ll train RNNs to generate text character by character and ponder the question “how is that even possible?”*
- **Blog** “Understanding LSTM Networks” - [Understanding-LSTMs](#)

- **Code samples:**

- **WordRNN:** Working repository for training and using a multilayer RNN and LSTM models – [word-rnn-tensorflow](#)
- **Text analysis:** Gensim library [examples](#)



The end