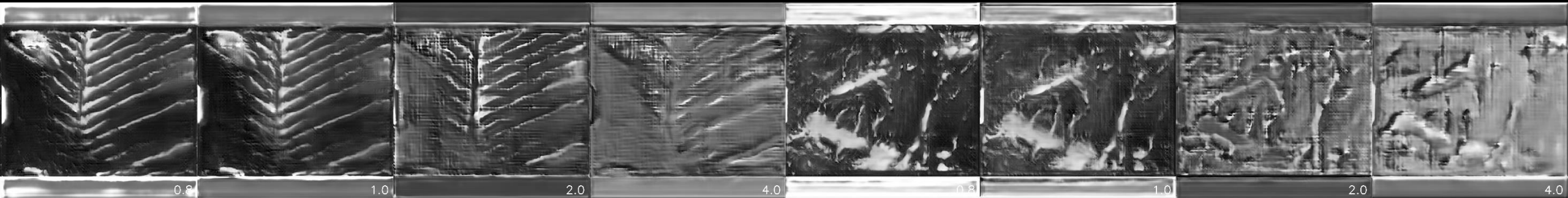


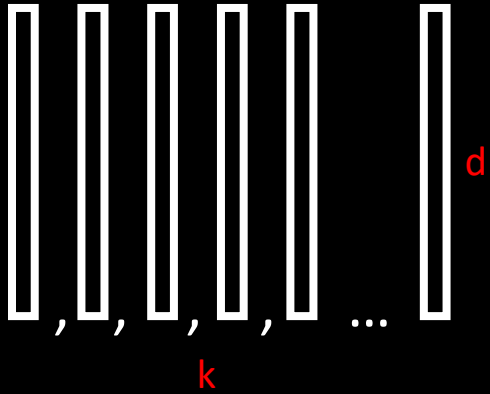
Exploring Machine Intelligence

Week 7, Sequential Modelling

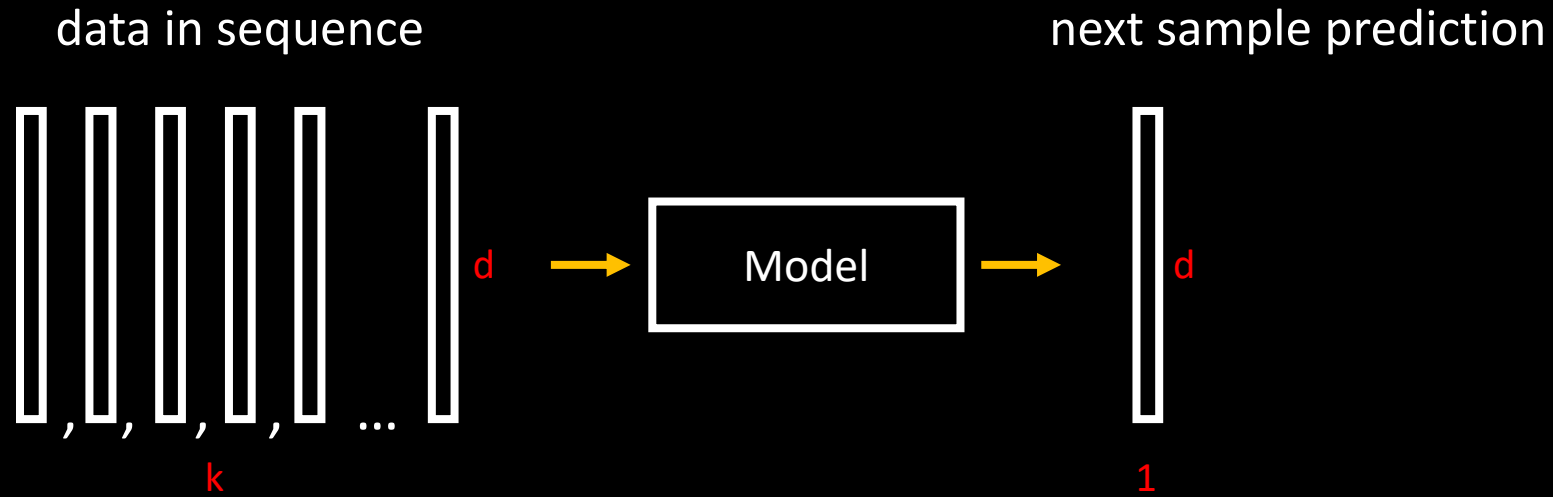


Motivation for today

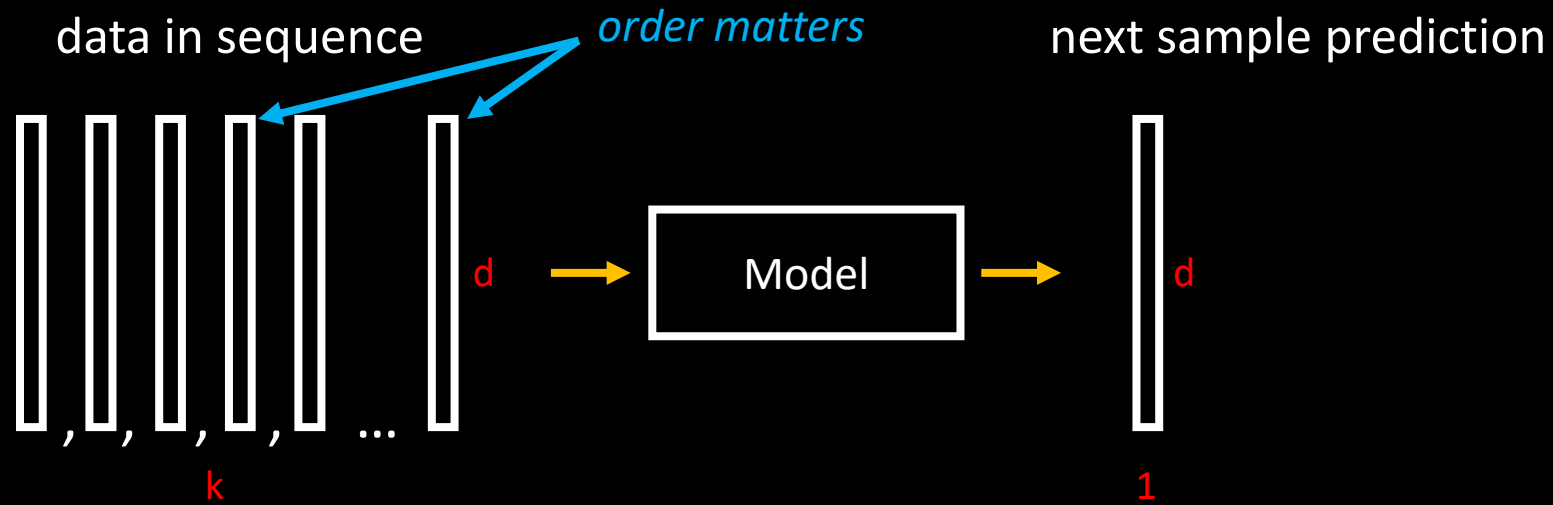
data in sequence



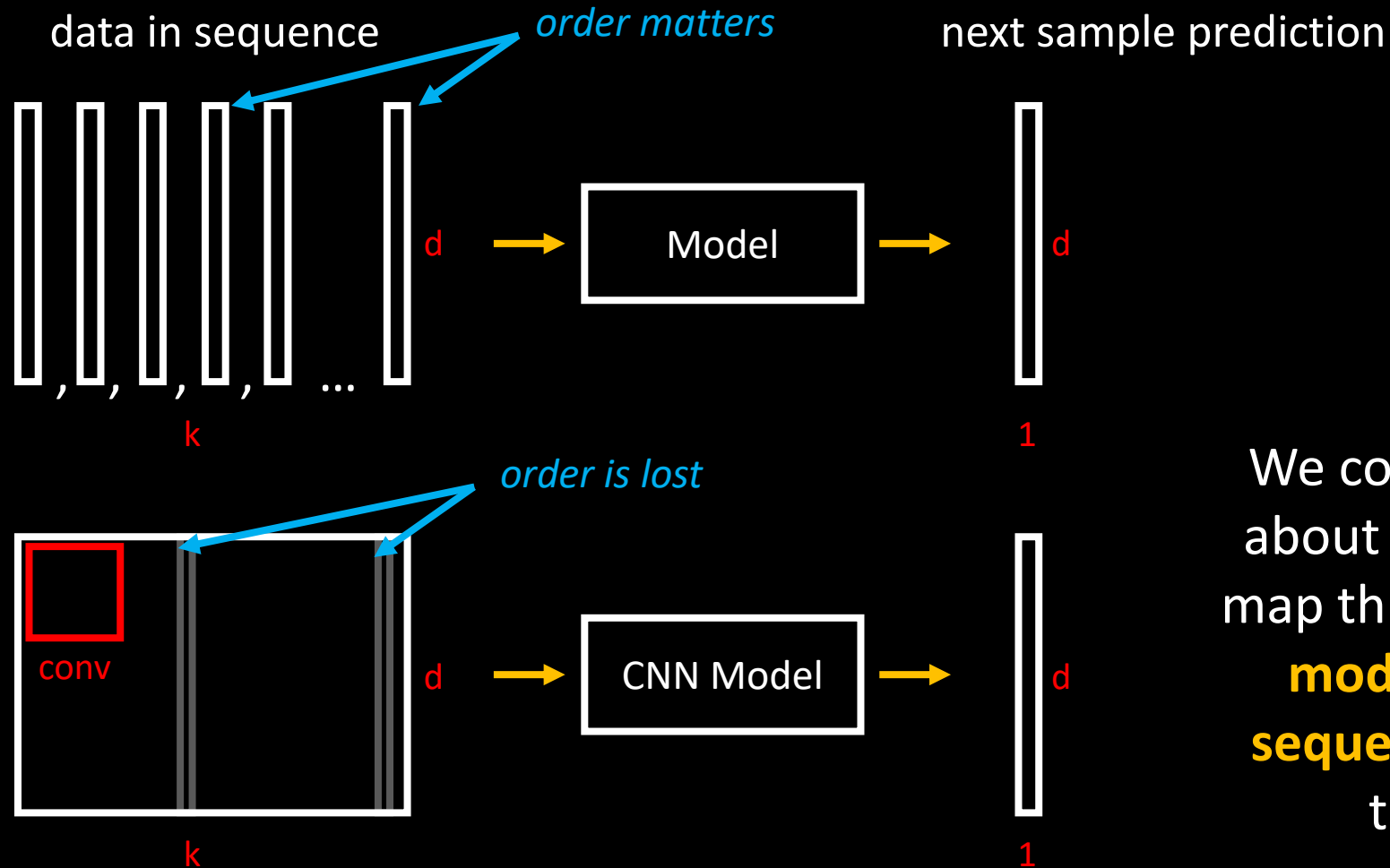
Motivation for today



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.

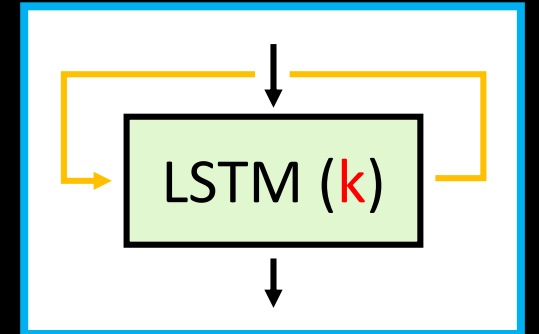
Today

Sequential Modelling

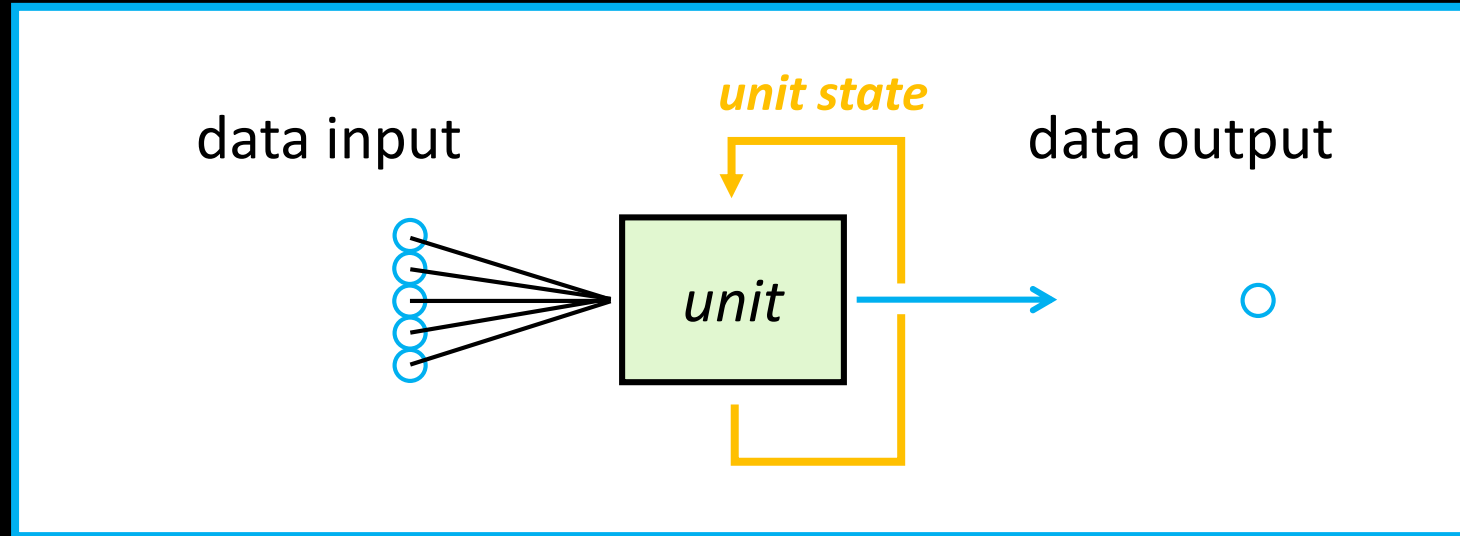
- Creating networks from **units with memory**
- Plugging in **textual and musical data**

Practical session:

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

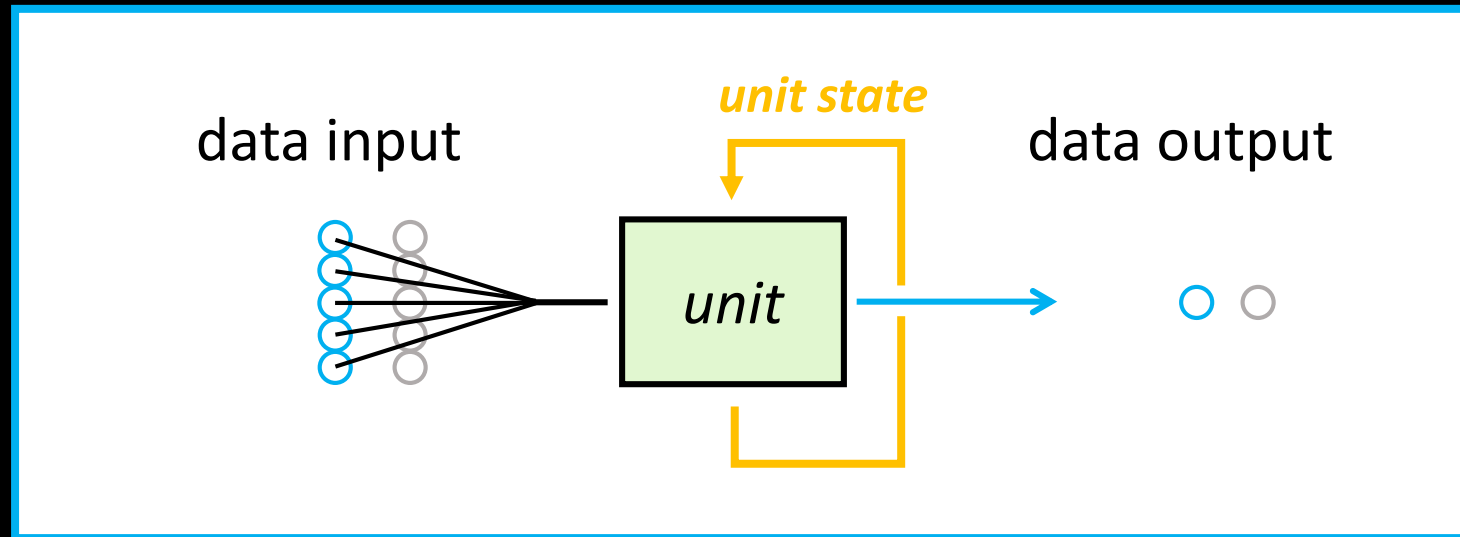


Unit with a memory



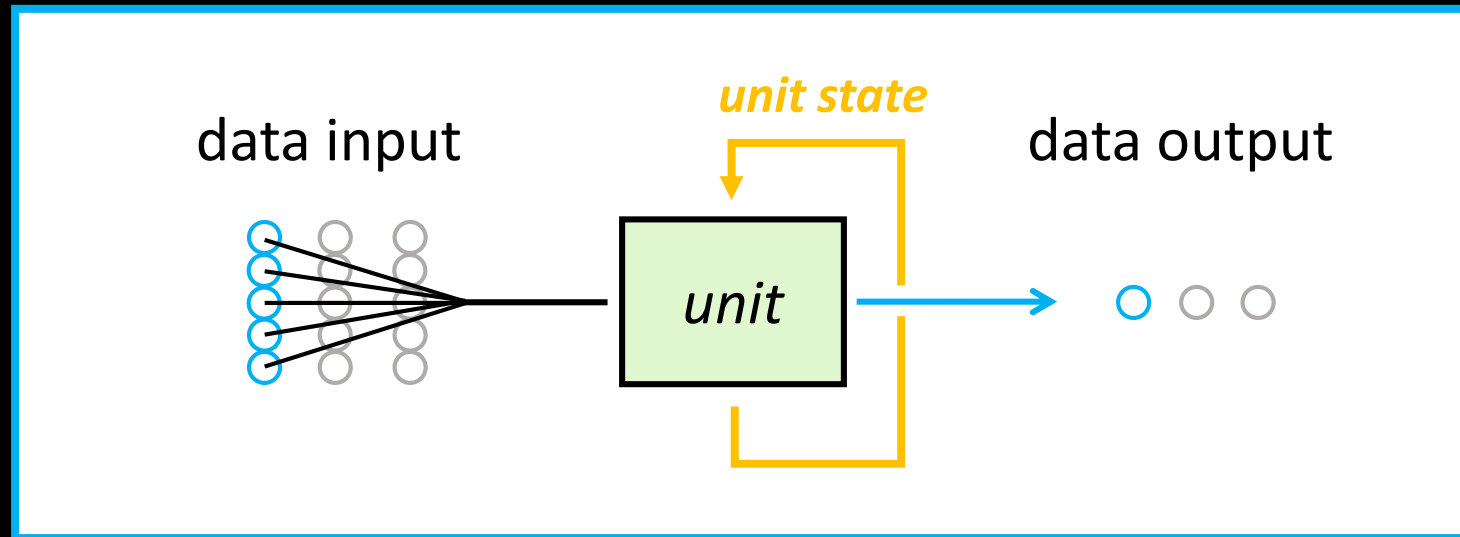
- Not only we **produce some output from the input**, we also **alter the unit state** (**memorize** some data from the next input in sequence)
- How both of this happens is something we **learn from the structure of data**

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**

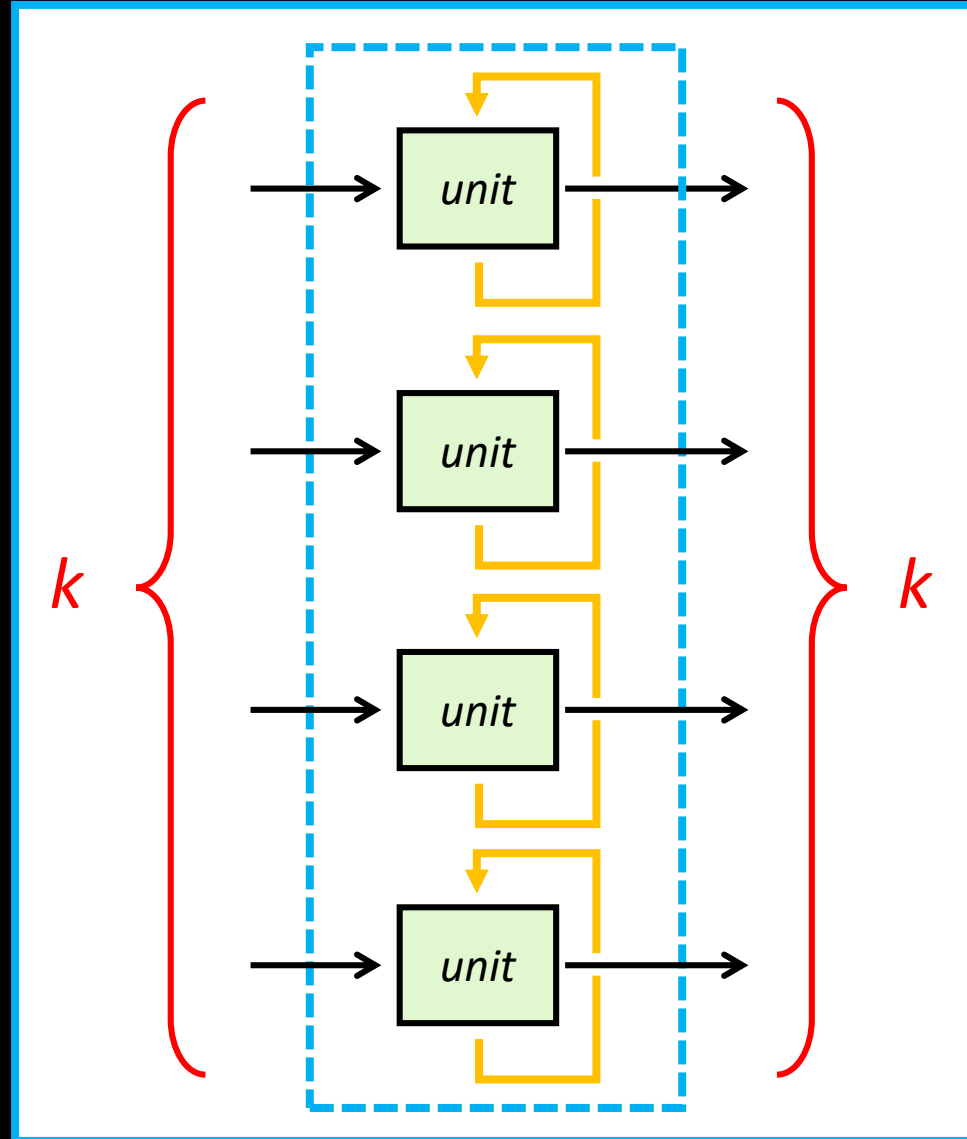
Unit with a memory



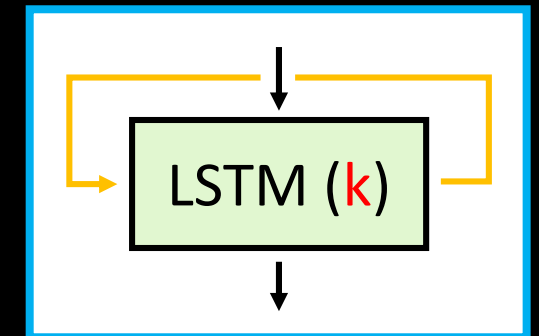
- Not only we **produce some output from the input**, we also **alter the unit state** (**memorize** some data from the next input in sequence)
- How both of this happens is something we **learn from the structure of data**

Recurrent layer

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

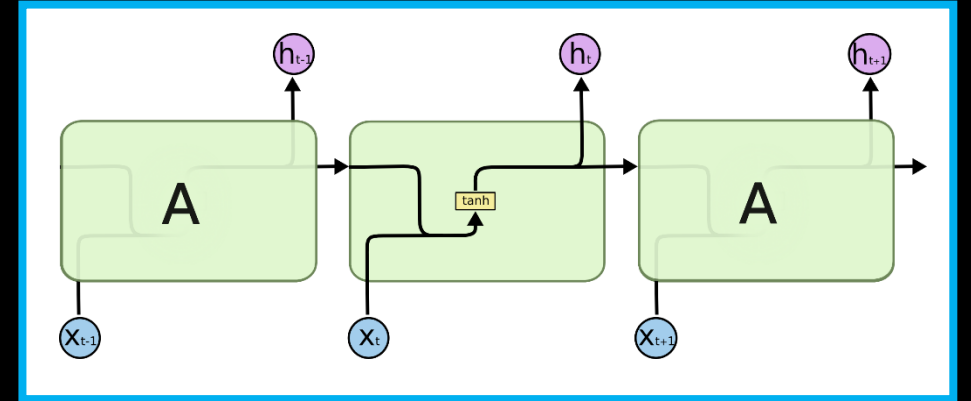


- Single **LSTM layer** with k units is usually simplified as:



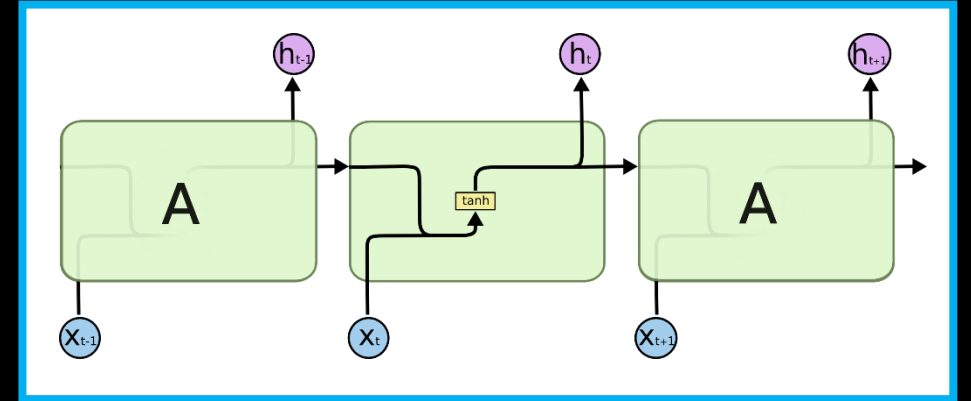
Unit types: RNNs and LSTMs

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



Unit types: RNNs and LSTMs

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

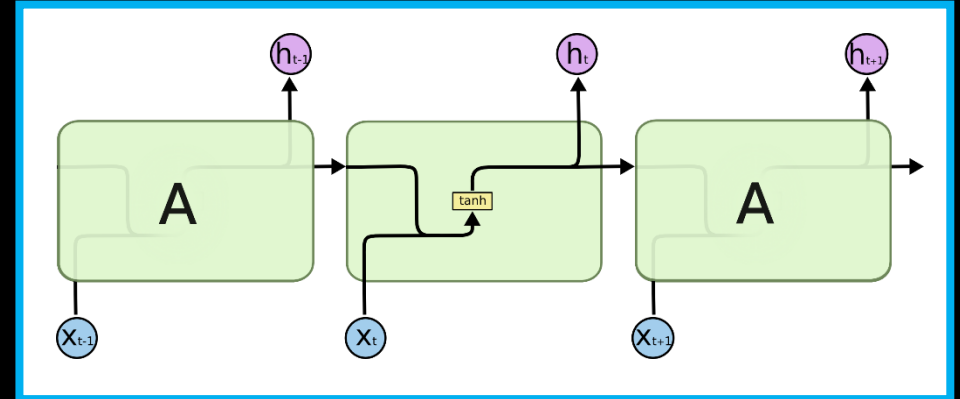


“I grew up in *France*... I speak fluent *French*.”

Unit types: RNNs and LSTMs

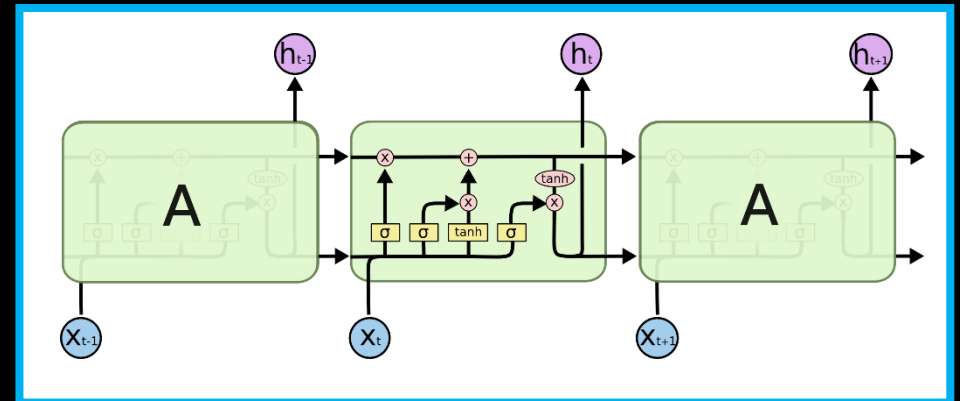
- **Recurrent Neural Networks (RNN):**

- 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)

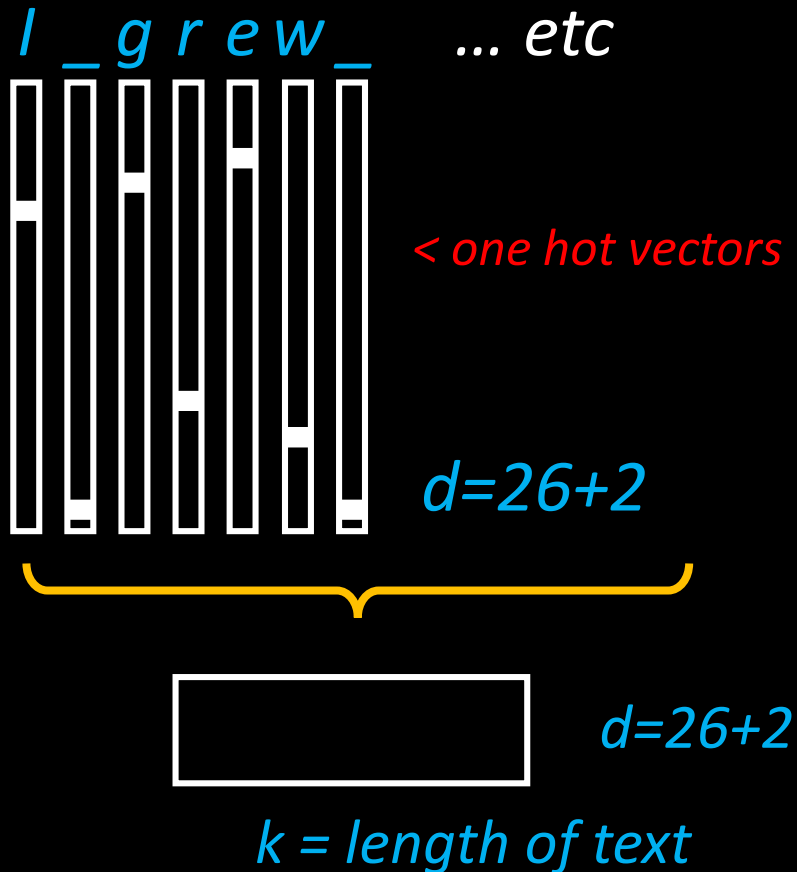
Plugging in text data

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

- **Vectorize text using a dictionary**

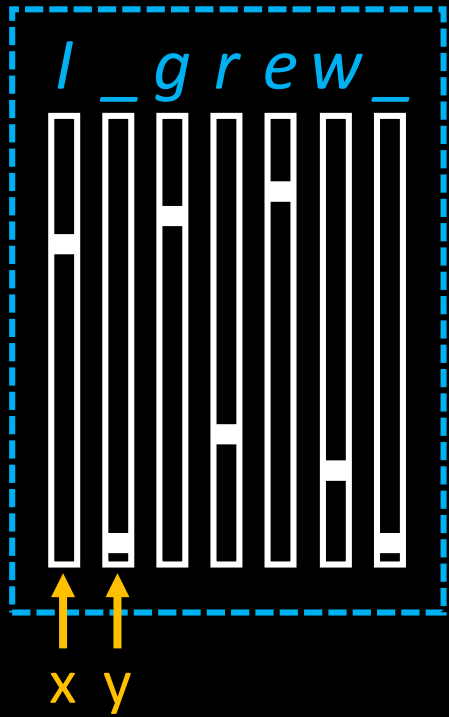
Plugging in text data

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



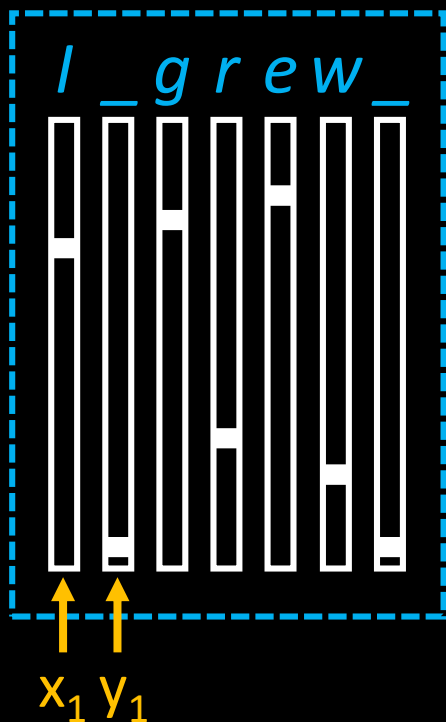
- **Vectorize text using a dictionary**
- Each letter is **encoded as a one-hot vector** using alphabet as dictionary (or full words using a word dictionary)
- We can **later decode the predictions** looking into that dictionary

dataset

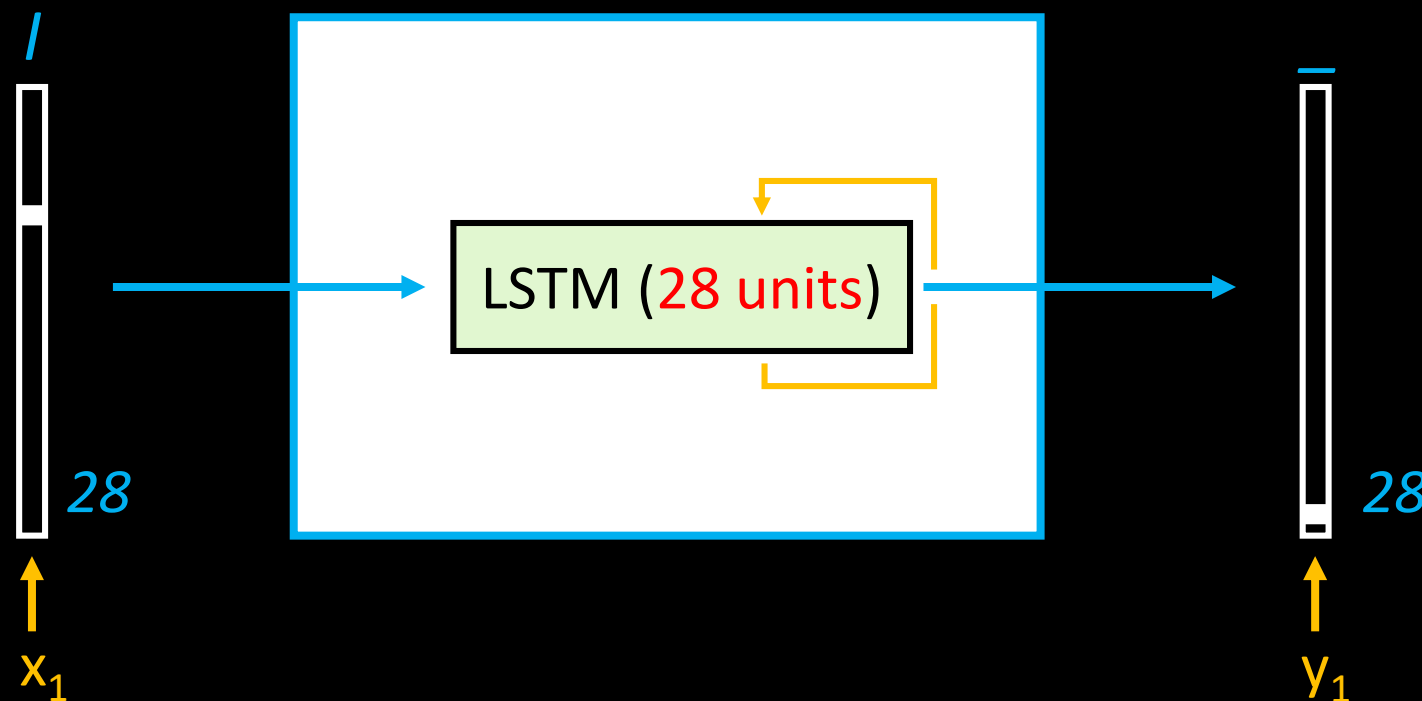


Simple model

dataset

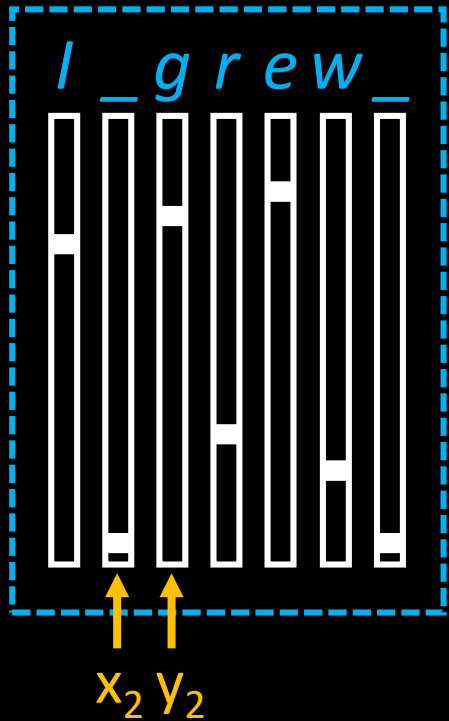


Simple model

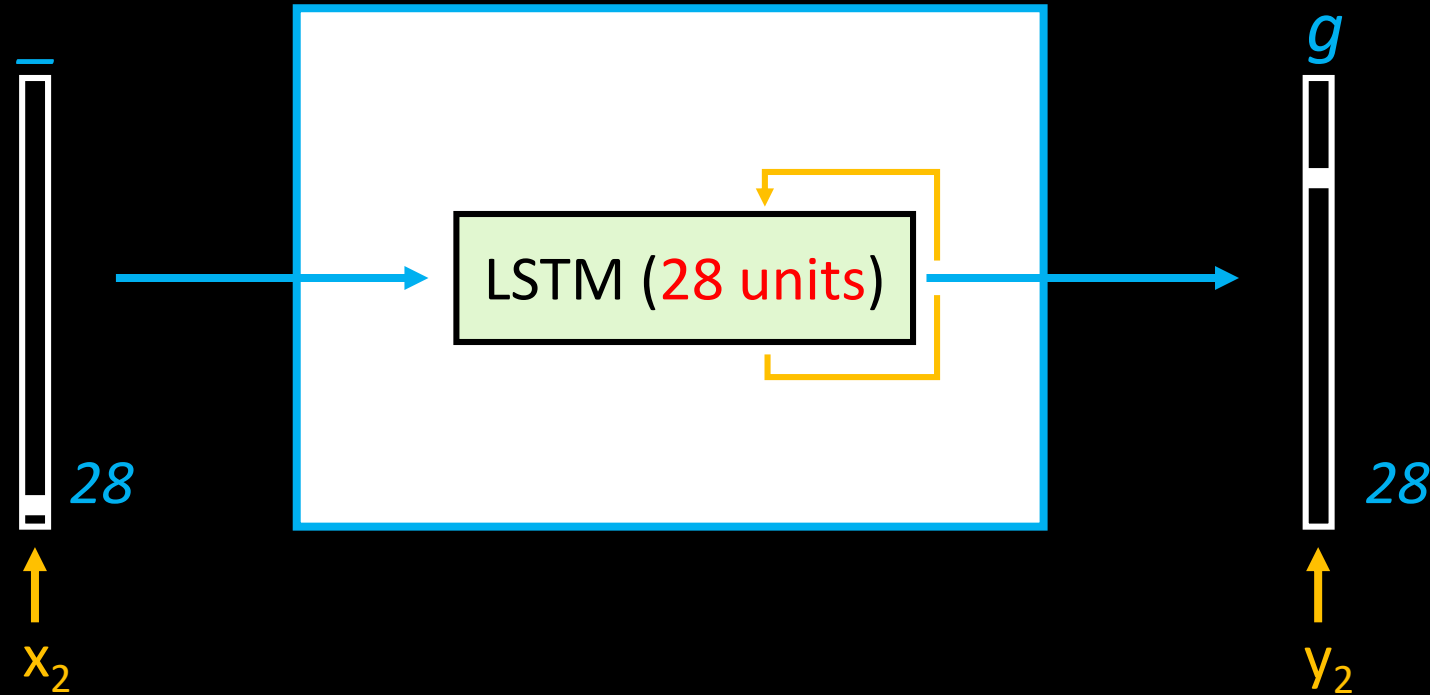


- We are **training the model** to **learn the transformation $x \rightarrow y$** while it also **learns what** is useful **to save** inside the 28 units (what preceded before this x).

dataset

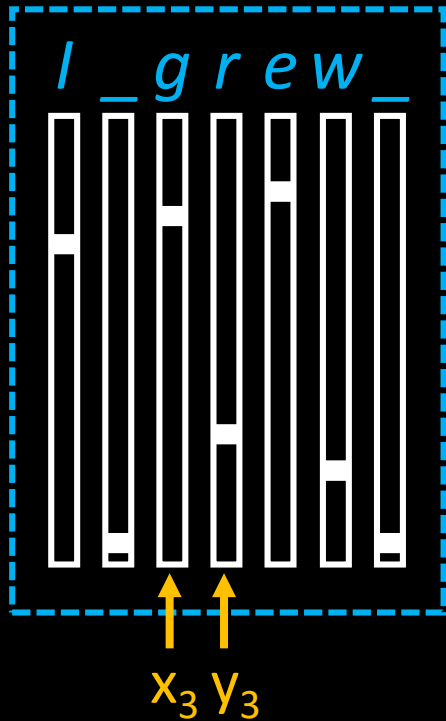


Simple model

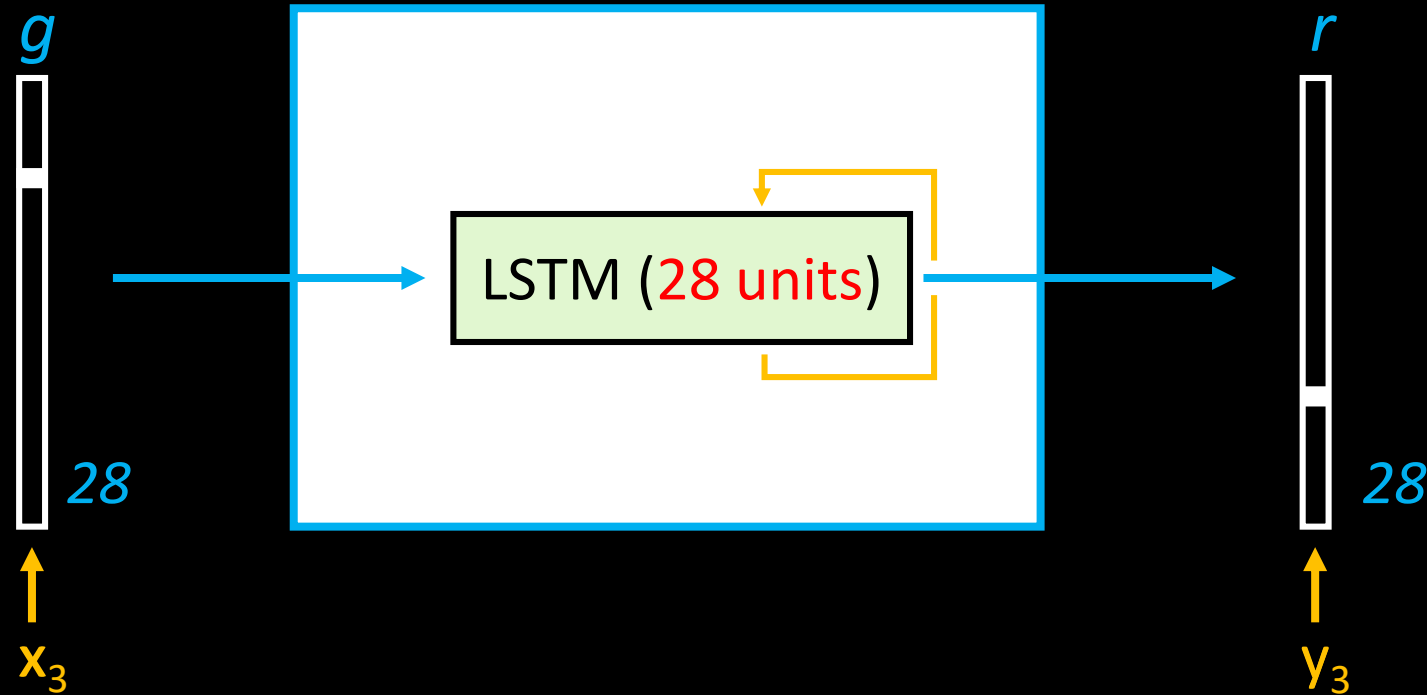


- We are **training the model** to **learn the transformation $x \rightarrow y$** while it also **learns what** is useful **to save** inside the 28 units (what preceded before this x).

dataset



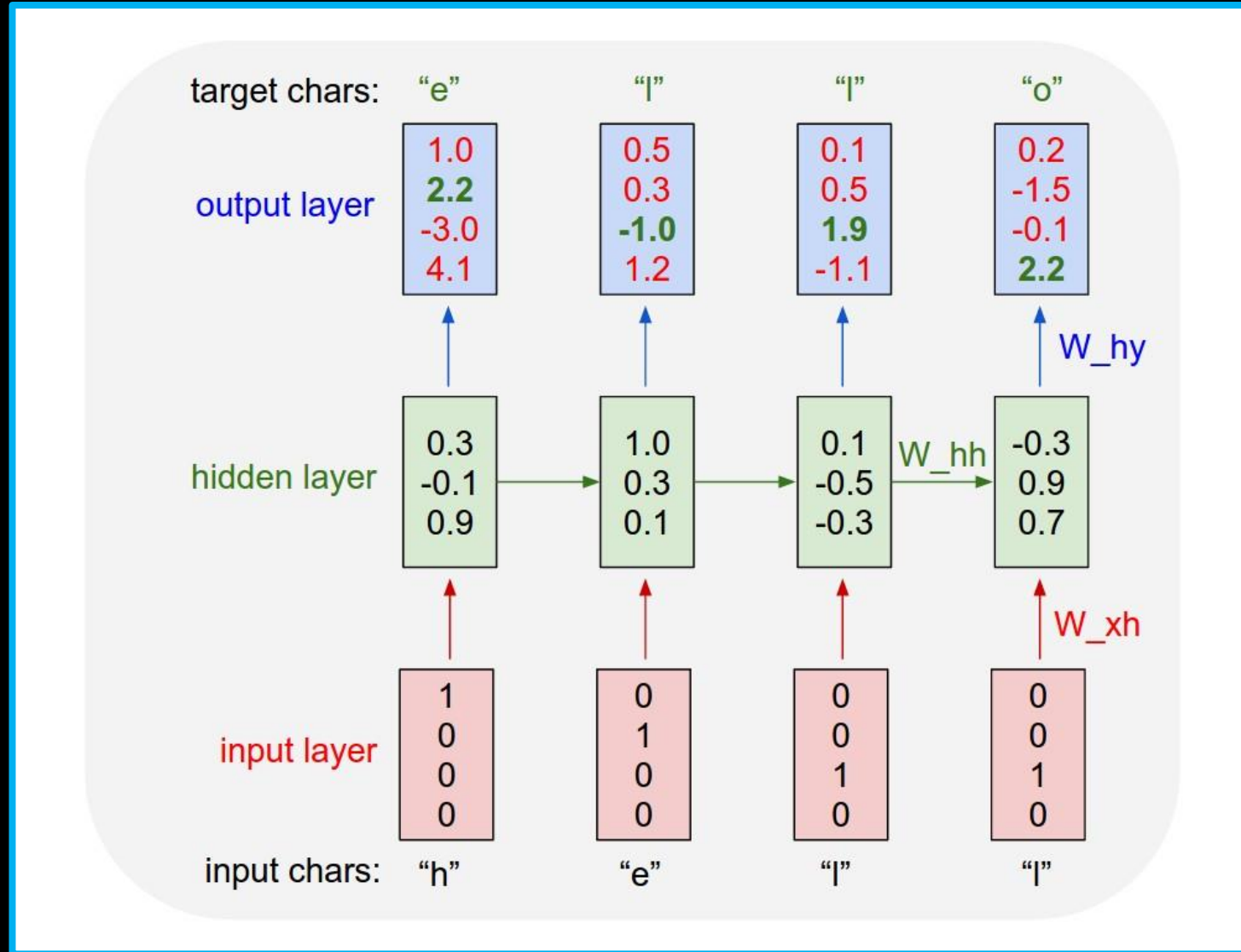
Simple model



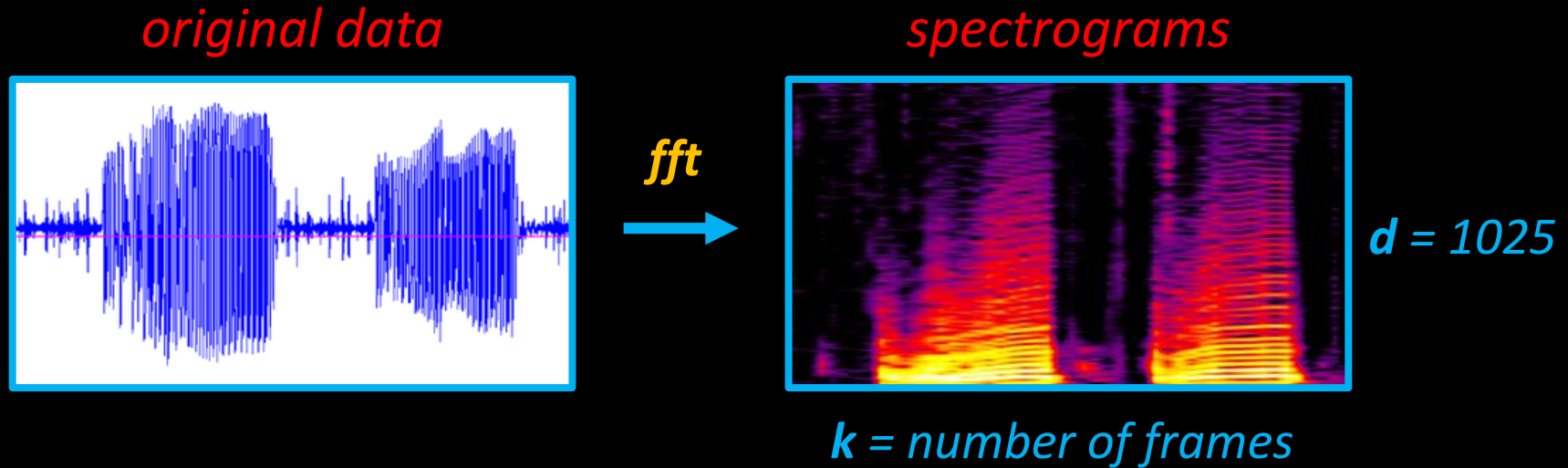
- We are **training the model** to **learn the transformation $x \rightarrow y$** while it also **learns what** is useful **to save** inside the 28 units (what preceded before this x).

Text data

- Example:

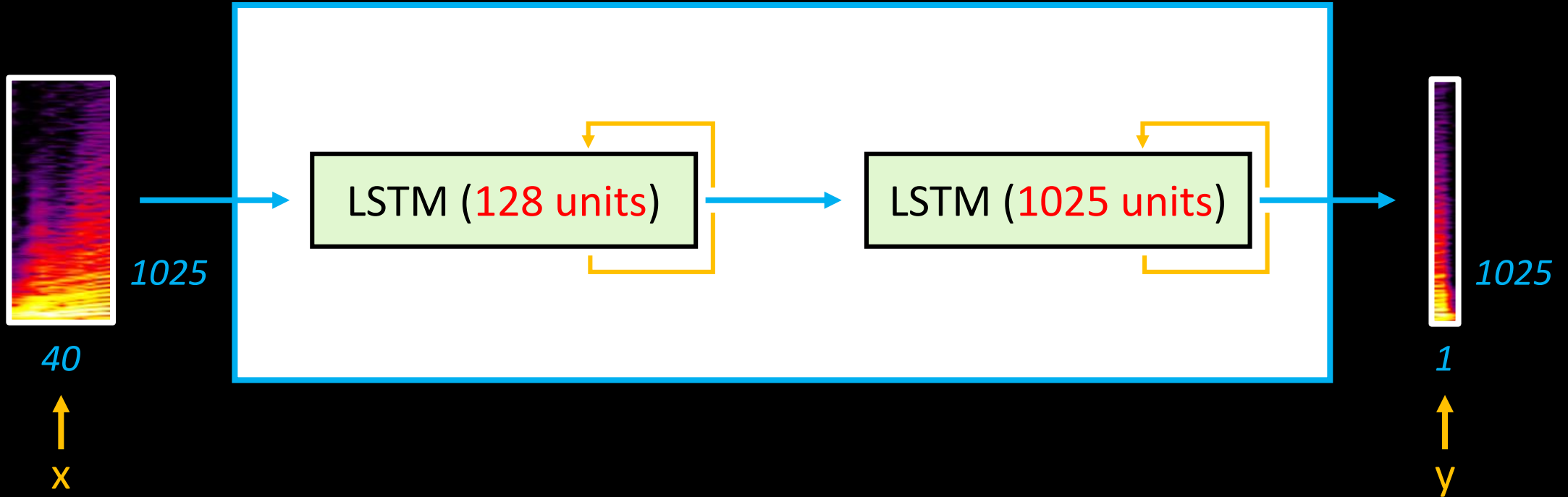


Plugging in music data



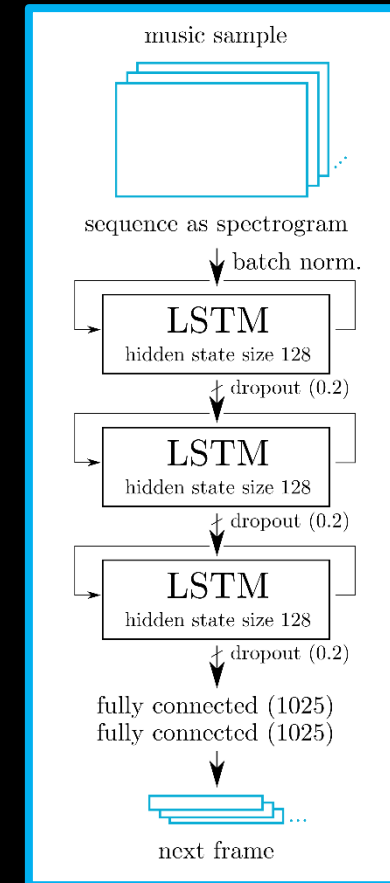
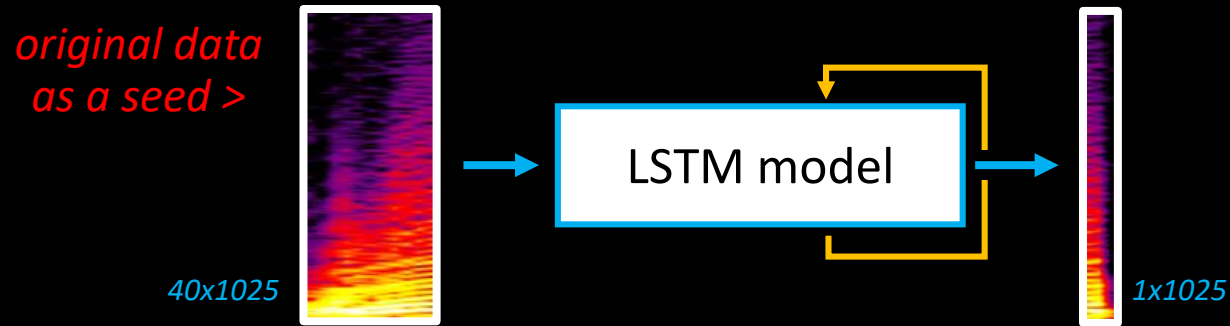
- **Encode music using the Fourier Transform** (*fft*) to get **spectrogram** (which can be considered as image representation)
- We can **later decode the predictions** using the *inverse fft*

Larger model

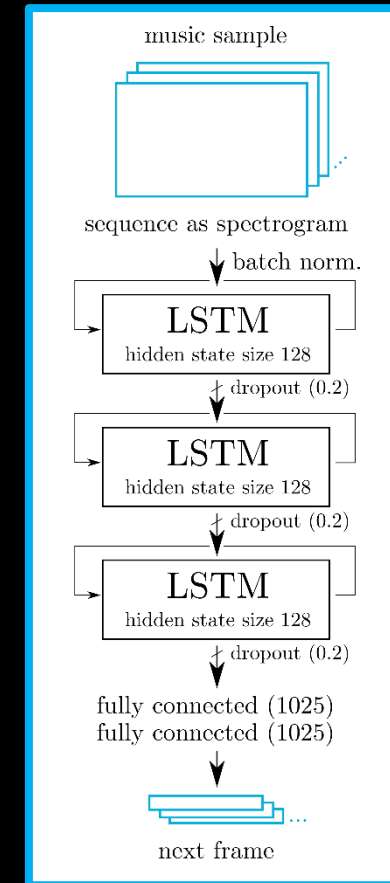
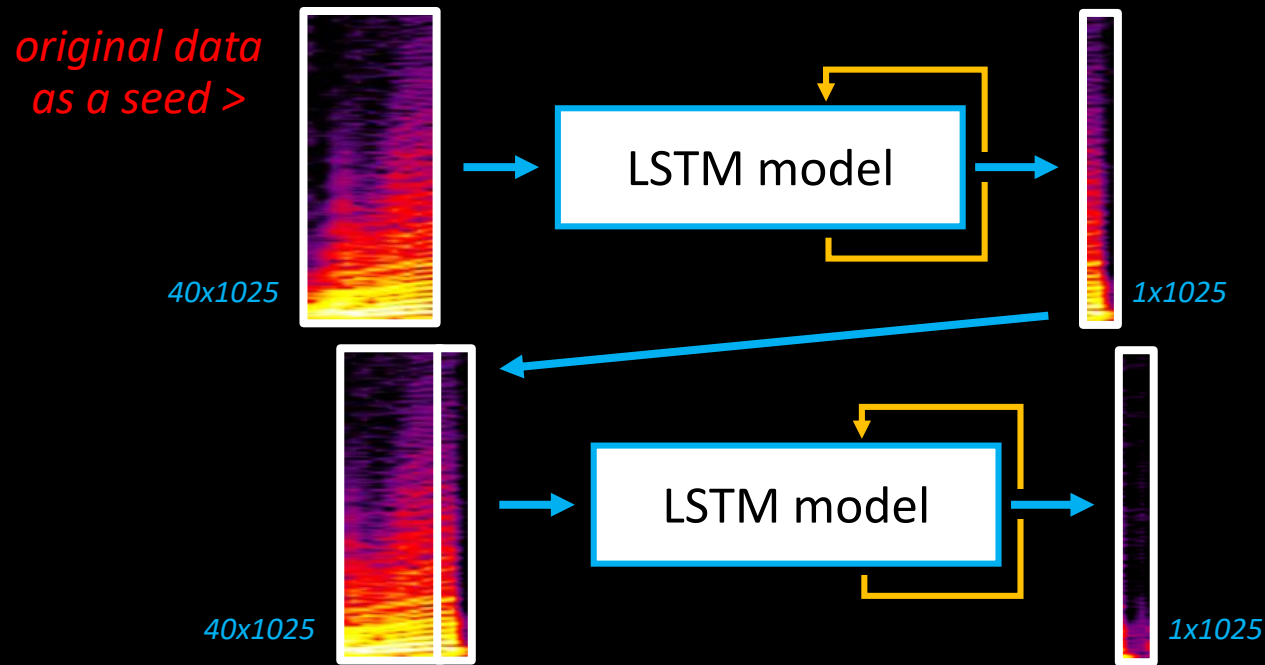


- We are **training the model** on a longer sequence of data (*"many to one"*), the model still **learns the transformation** of **x -> y**.

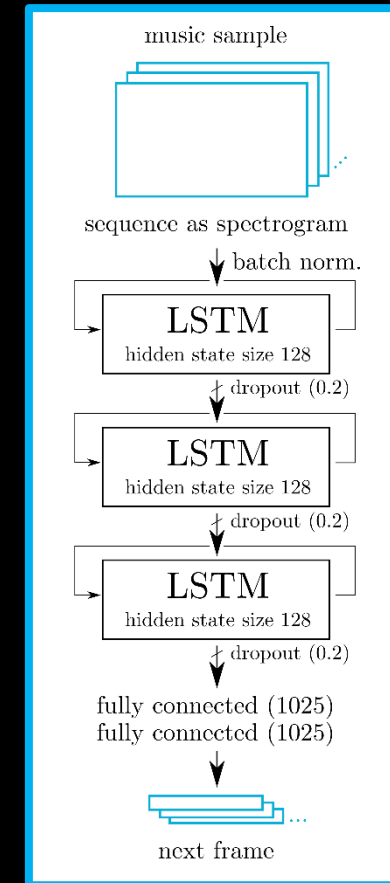
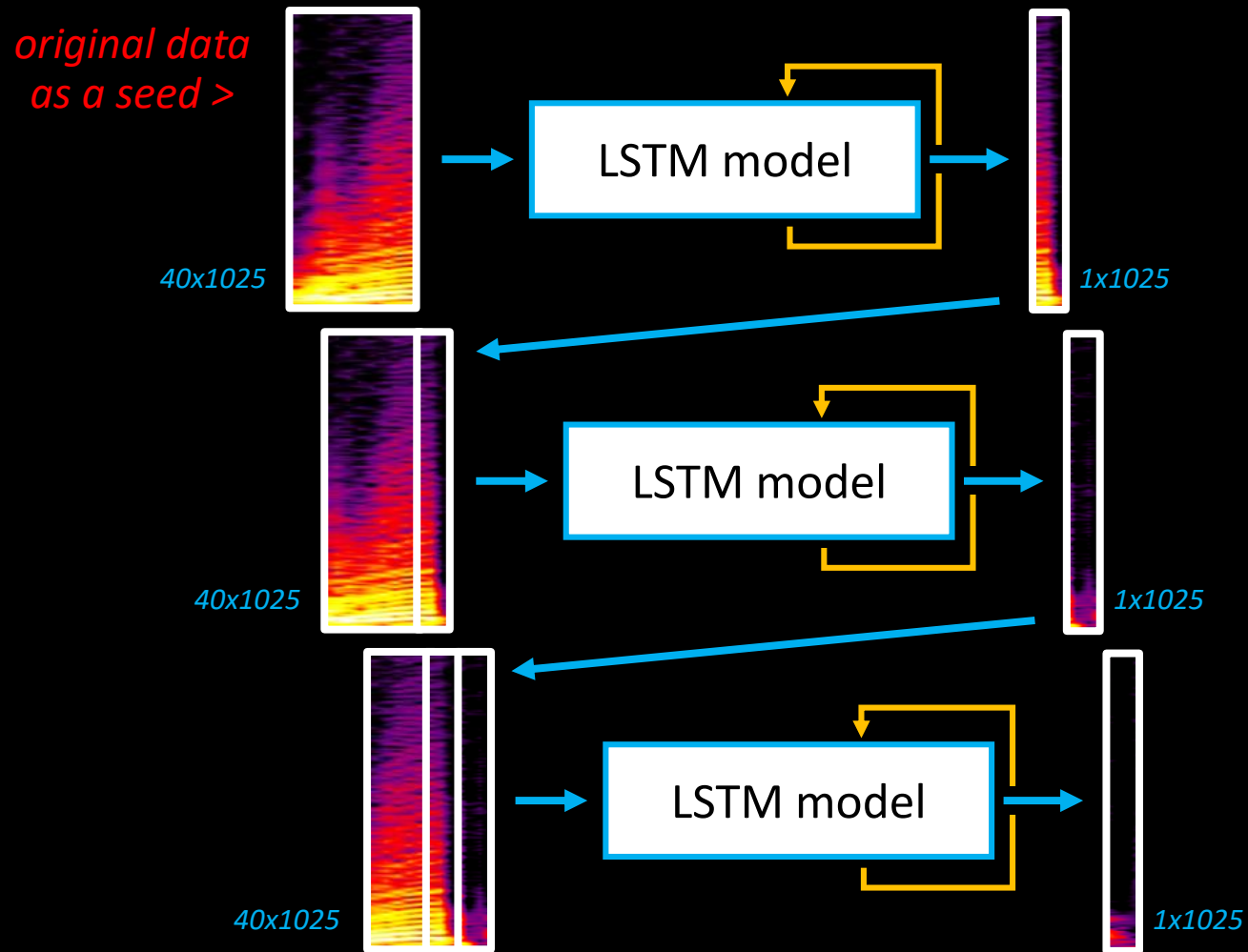
Using a trained LSTM model:



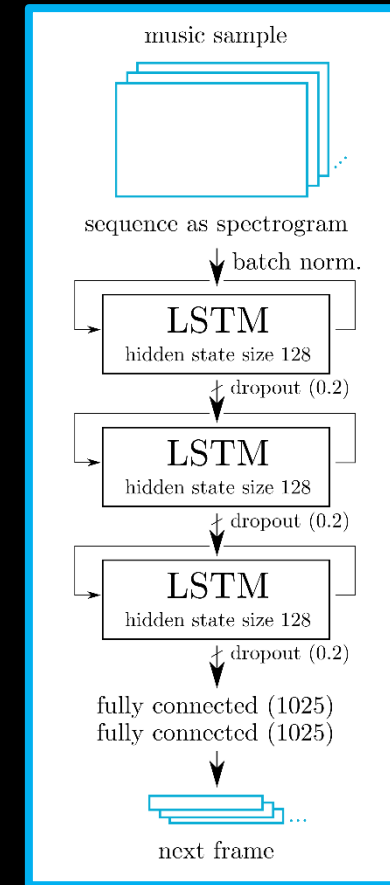
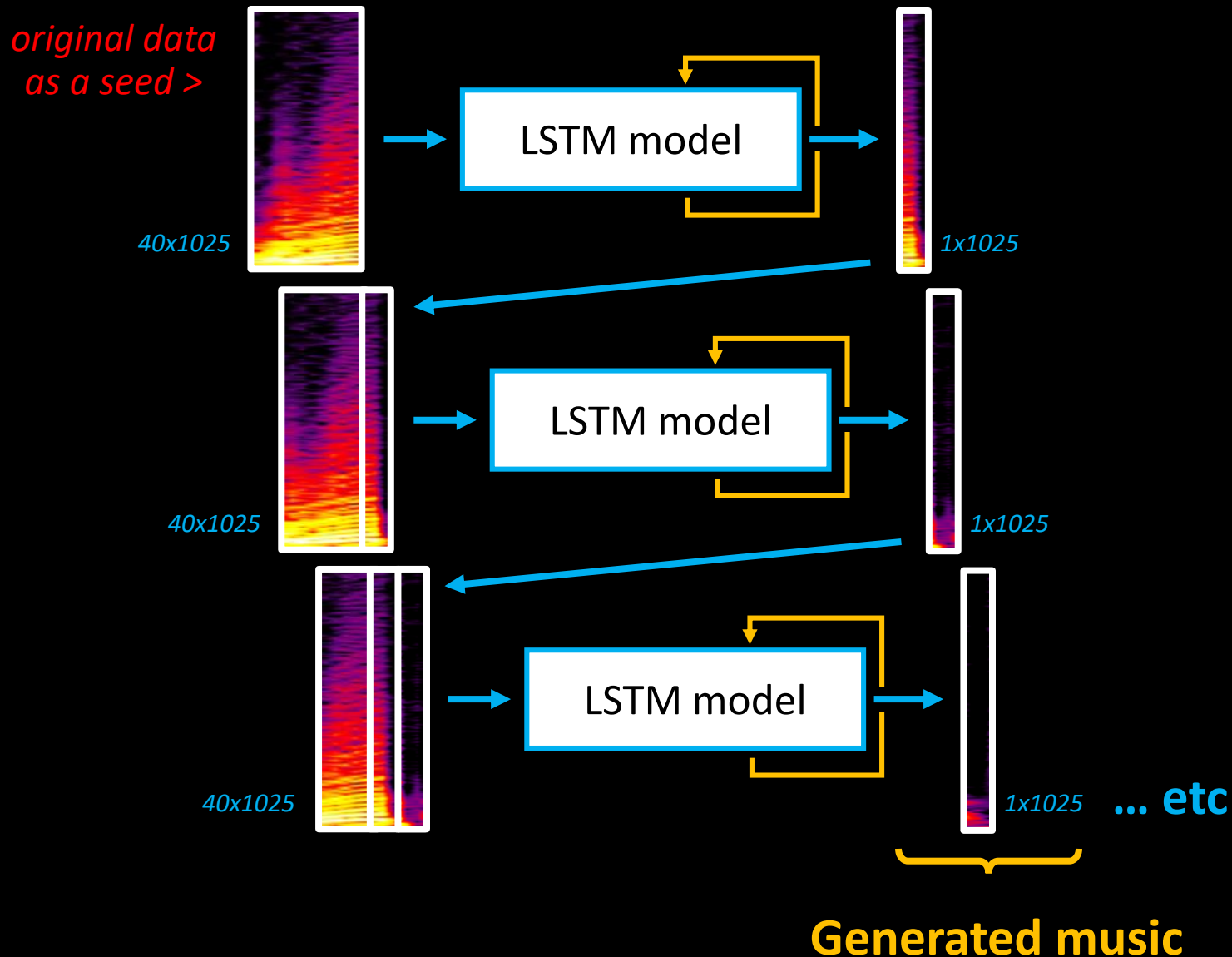
Using a trained LSTM model:



Using a trained LSTM model:

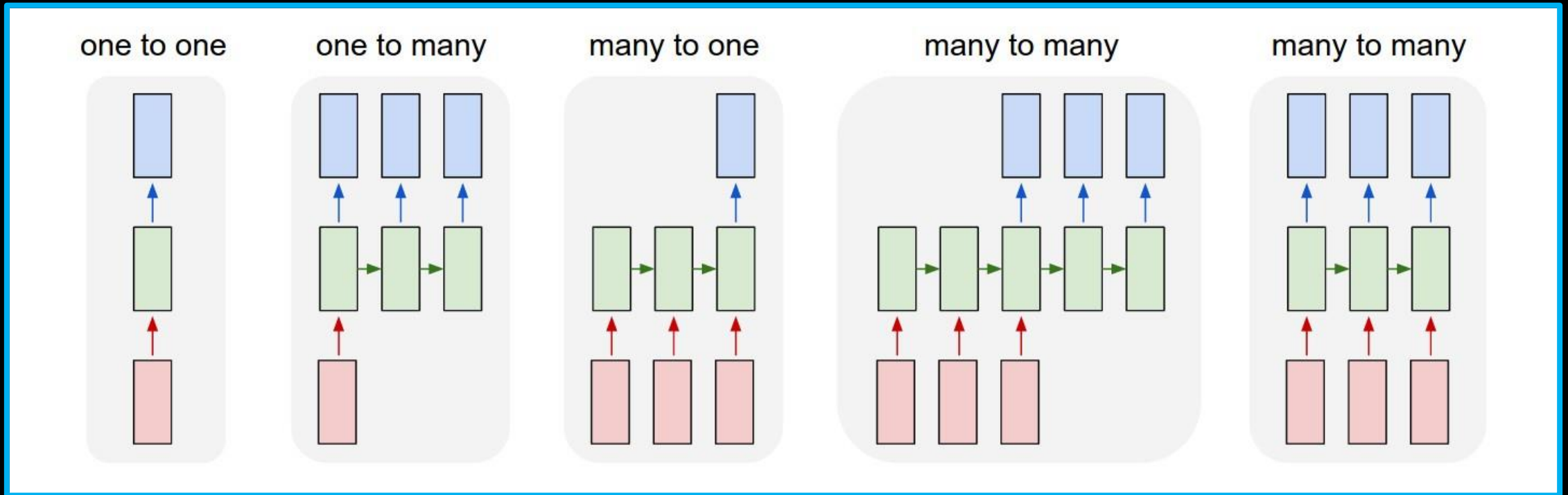


Using a trained LSTM model:



- 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)

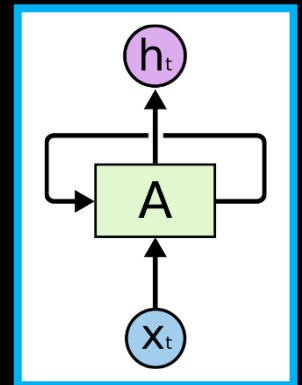
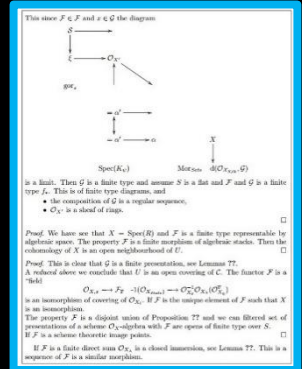
Links and additional readings:

- **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](#)

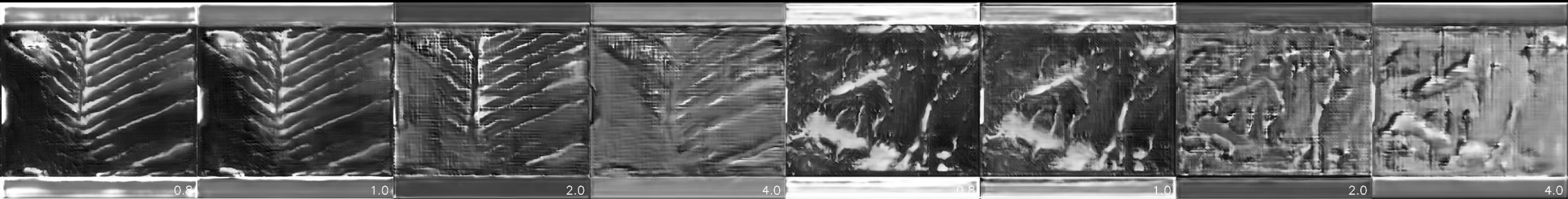


Practicum live session

- 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](#)

Exploring Machine Intelligence

Week 7, Sequential Modelling



Final Exam Q&A

Weeks overview

- **I.** Intro & Motivation
- **II.** Fully Connected Neural Networks
- **III.** Convolutional Neural Networks
- **IV.** Interactive ML
- **V.** AutoEncoders and GANs
- **VI.** Domain2domain (pix2pix), Style Transfer, Deep Dream
- **VII.** Sequential models

Element 1: Online testing: Moodle quizzes with practical and theoretical questions. (50%) **[Finished!]**

Element 2: Final examination: Online quiz with 20 multiple choice questions and 3 open ended questions (expect short essay answers). (50%) **[opens at 12.06.]**

- Following slides include some **example topics and questions** that may appear in the exam (*PS: these are on purpose hard; in the exam they will be easier*).

I. Intro & Motivation

- **Linear regression**

II. Fully Connected Neural Networks

- **Neuron parameters**
- **Fully Connected Neural Network**
 - Connectivity of a neuron
- **Error between prediction and label**

III. Convolutional Neural Networks

- **Model overfitting**
- **Convolution**
 - As a filter
 - Parameters
- **What allows layers of CNN to specialize?**
- **Describe how we can think about an existing models as having the feature extractor and classifier section?**

V. AutoEncoders and GANs

- **Describe AutoEncoder as an identity operation learning**
- **Describe GAN as a two-player game**
 - What is the purpose of Generator?
 - What is the purpose of Discriminator?
- **Differences between default versions of AE and GAN**
 - Encoding-ability
 - Loss function
- **Progressive Growing GAN details**

VI. pix2pix, style transfer, deep dream

- **Describe pix2pix as a translation**
- **Describe style transfer as a content and style separation**
 - How do we edit the input image?
- **Describe the deep dream technique**
 - What are we trying to maximize?

VII. Sequential models

- **Describe what do we want from sequential models?**
- **Describe text vectorization (encoding)**
- **Describe music vectorization (encoding)**

The end