

Recurrent Neural Networks

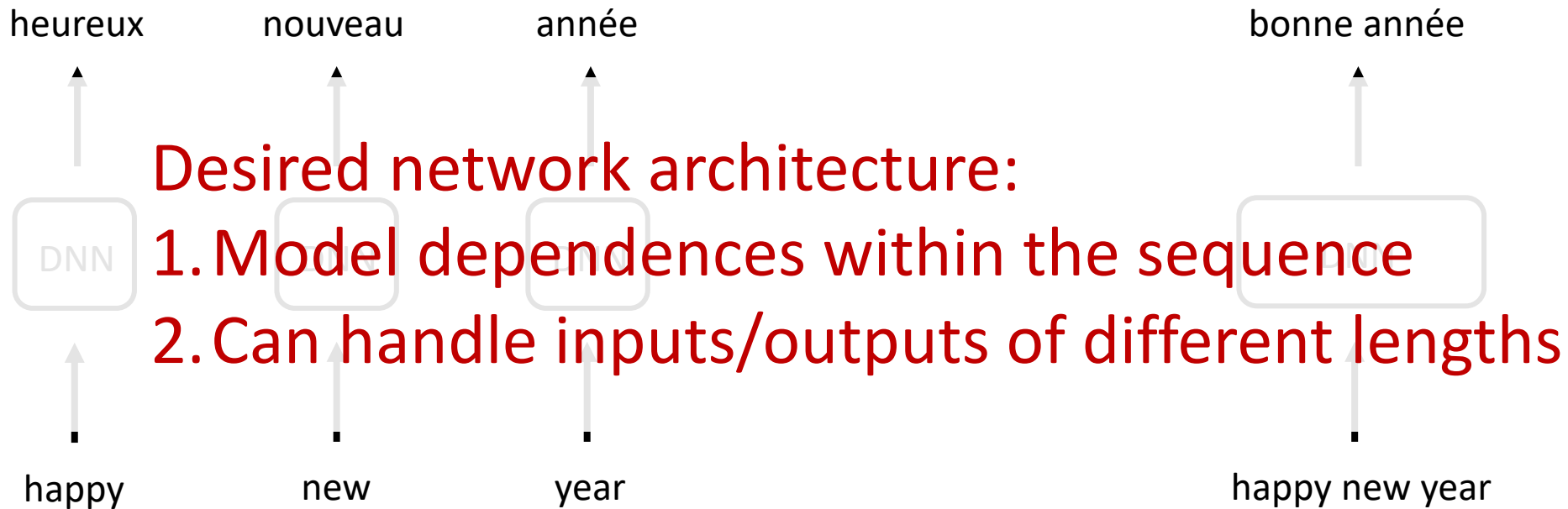
Applications

Deep Learning Course 2021

Yingzhen Li (yingzhen.li@imperial.ac.uk)

Why Recurrent Neural Networks

Machine translation as a motivating example:



Desired network architecture:

1. Model dependencies within the sequence
2. Can handle inputs/outputs of different lengths

ignoring dependencies



cannot handle inputs of varied lengths



Sequence-to-Sequence Model

Machine translation (e.g. EN to FR):

- Data sequence: $x_{1:T} = (x_1, \dots, x_T)$
 - x_t : the t^{th} word in the English sentence
- Label sequence: $y_{1:L} = (y_1, \dots, y_L)$
 - y_l : the l^{th} word in the French sentence

- Goal: learn the **conditional distribution**

$$p_{\theta}(y_{1:L}|x_{1:T})$$

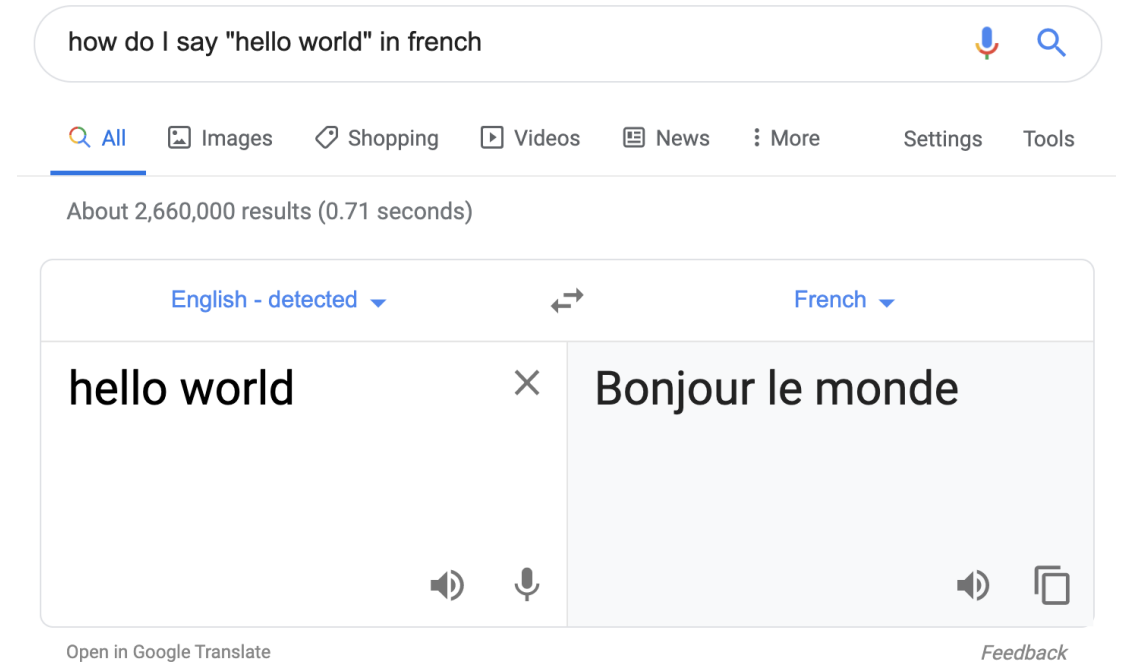
for x, y sequences of **any length**

- Idea: define an **auto-regressive model**

$$p_{\theta}(y_{1:L}|x_{1:T}) = \prod_{l=1}^L p_{\theta}(y_l|y_{<l}, v), v = \text{enc}(x_{1:T})$$

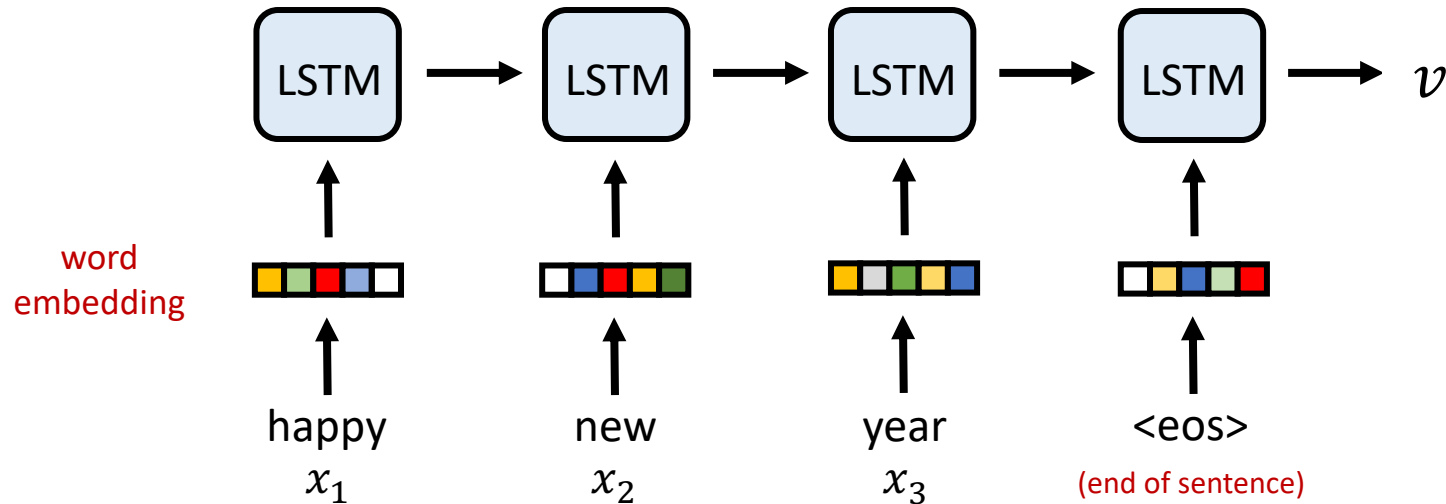
Sequence decoder

Sequence encoder



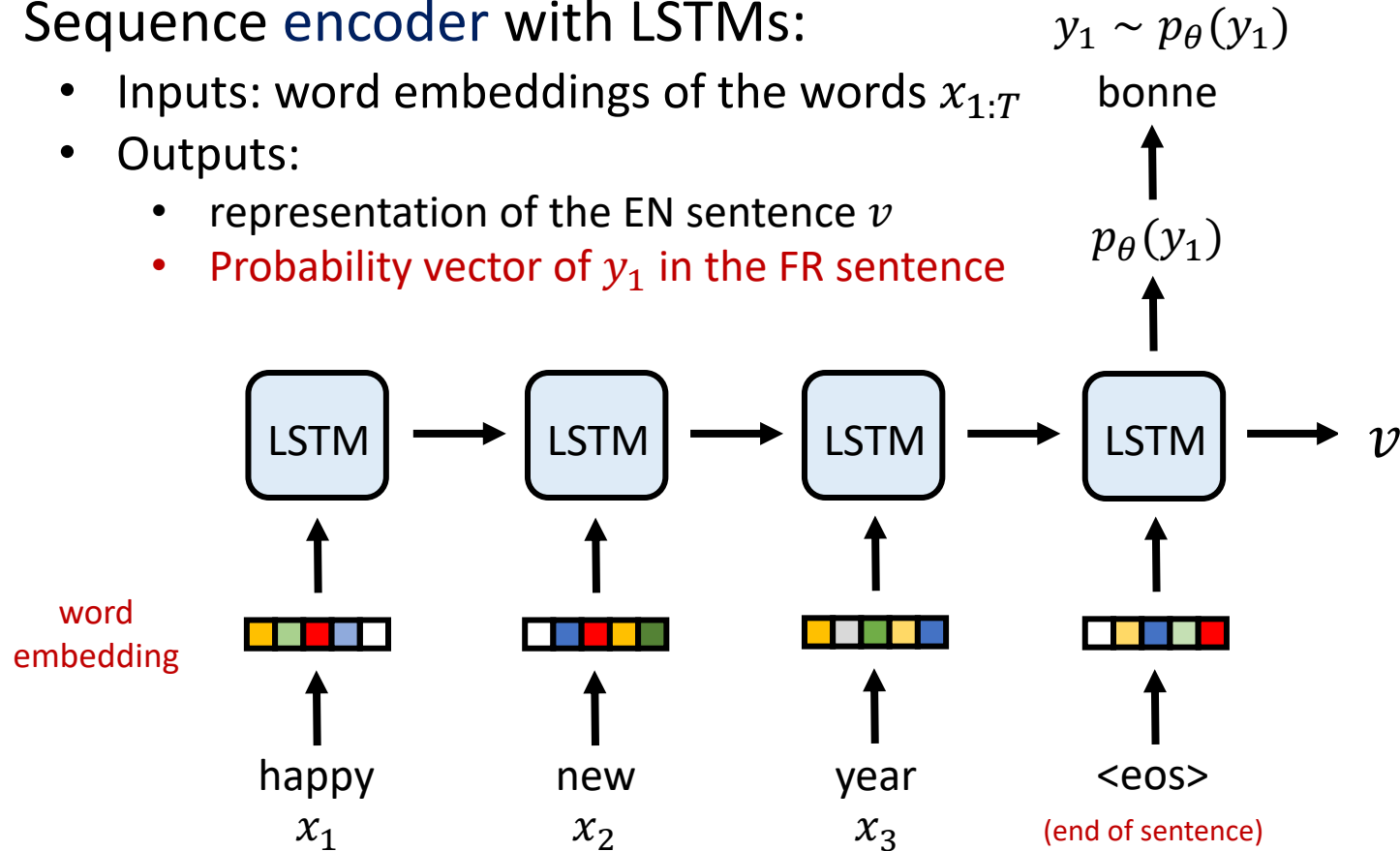
Sequence-to-Sequence Model

- Sequence **encoder** with LSTMs:
 - Inputs: word embeddings of the words $x_{1:T}$
 - Outputs:
 - representation of the EN sentence v



Sequence-to-Sequence Model

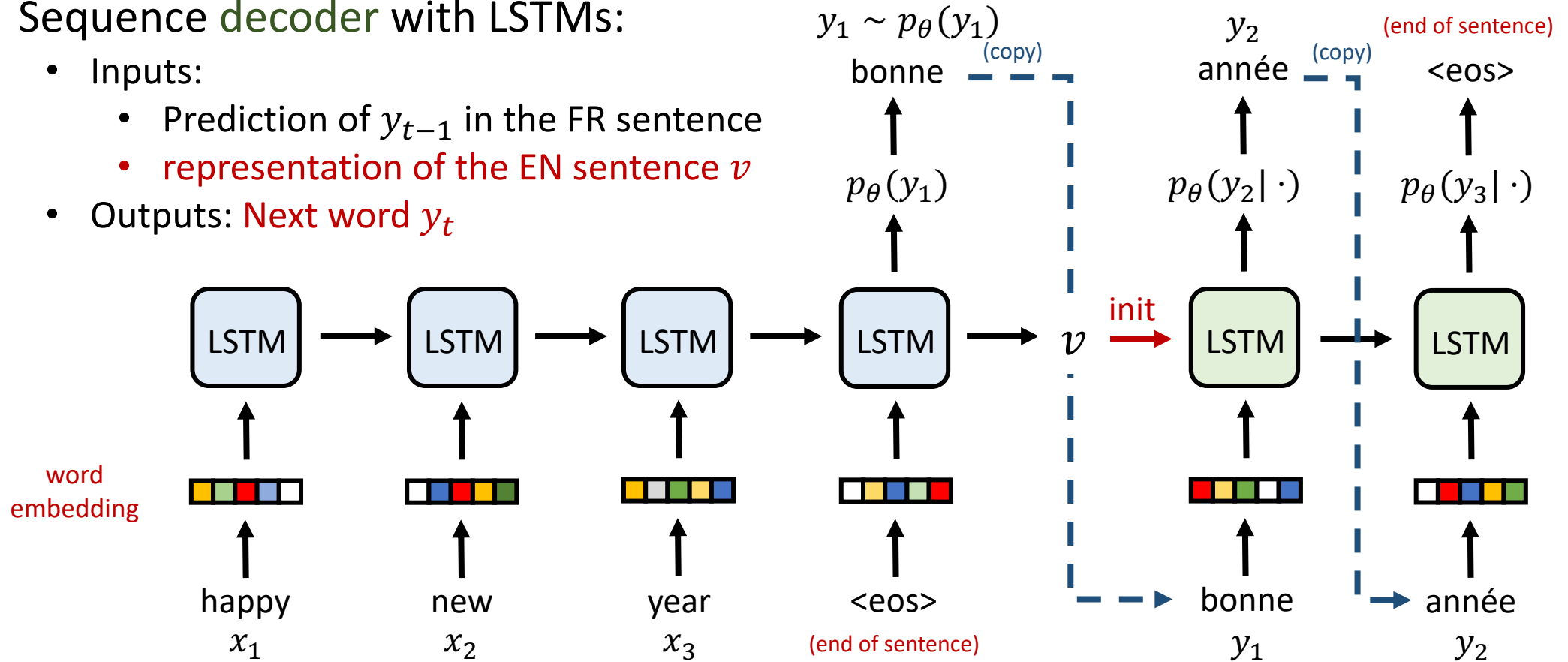
- Sequence **encoder** with LSTMs:
 - Inputs: word embeddings of the words $x_{1:T}$
 - Outputs:
 - representation of the EN sentence v
 - **Probability vector of y_1 in the FR sentence**



Sequence-to-Sequence Model

- Sequence decoder with LSTMs:

- Inputs:
 - Prediction of y_{t-1} in the FR sentence
 - representation of the EN sentence v
- Outputs: Next word y_t



Sequence-to-Sequence Model

- Sequence decoder inputs during training/test:

Training:

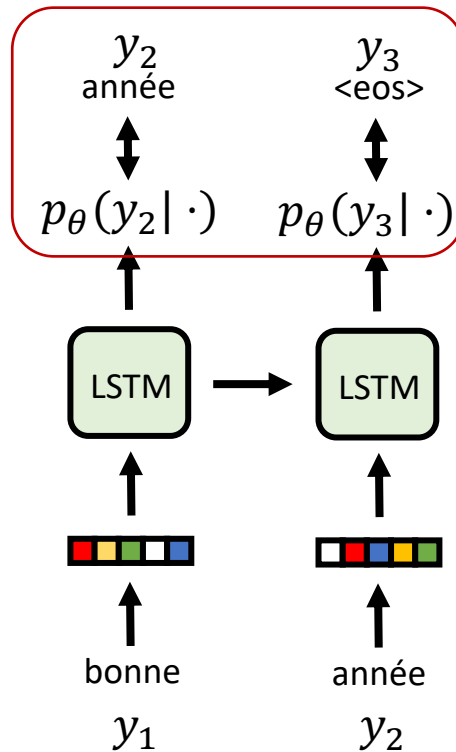
Data: $(x_{1:T}, y_{1:L})$

Inputs: the y_l word in the provided output supervision sequence

MLE training require computing

$p_\theta(y_{1:L}|x_{1:T})$ using data

Apply loss $L(p_\theta, y_{1:L})$



Test:

Data: $x_{1:T}$

Inputs: the y_l word predicted from the last decoding step

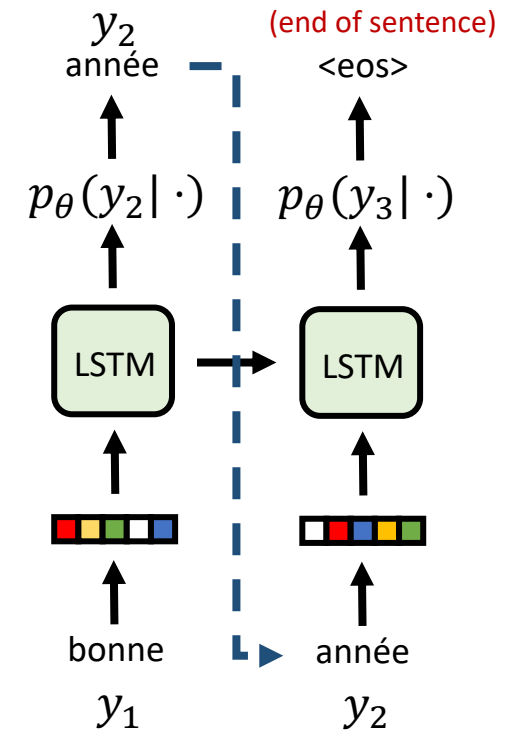
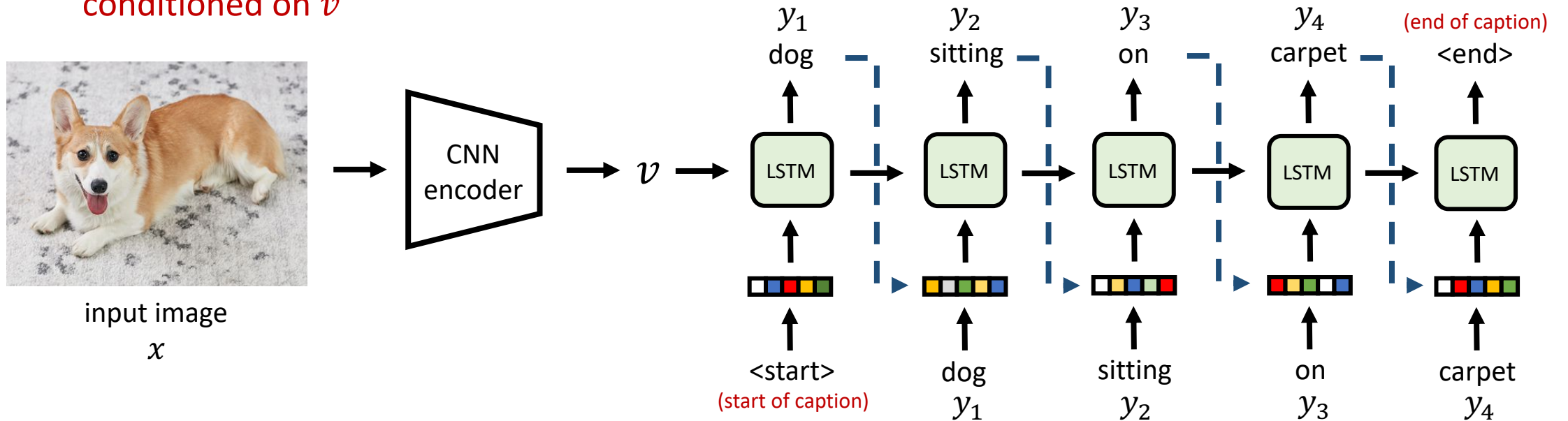


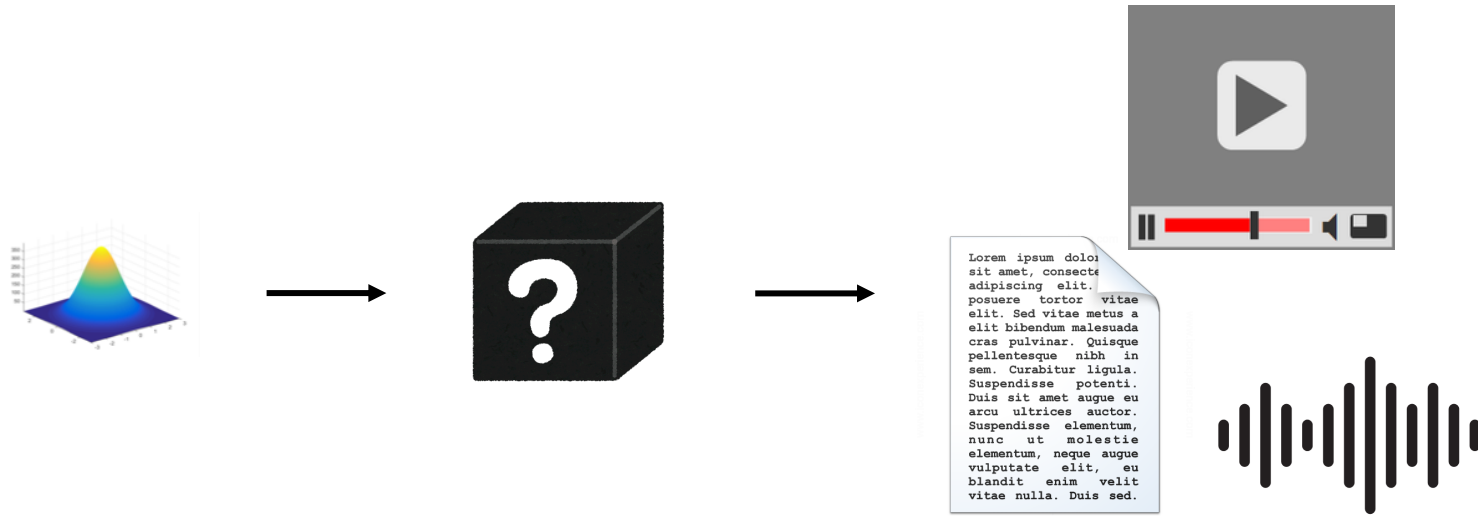
Image Captioning

- Image captioning with CNN encoder + LSTM decoder:
 - CNN encoder extract representation v of image x
 - LSTM decoder generate caption conditioned on v



Sequence generation models

- Latent variable model for sequence generation:



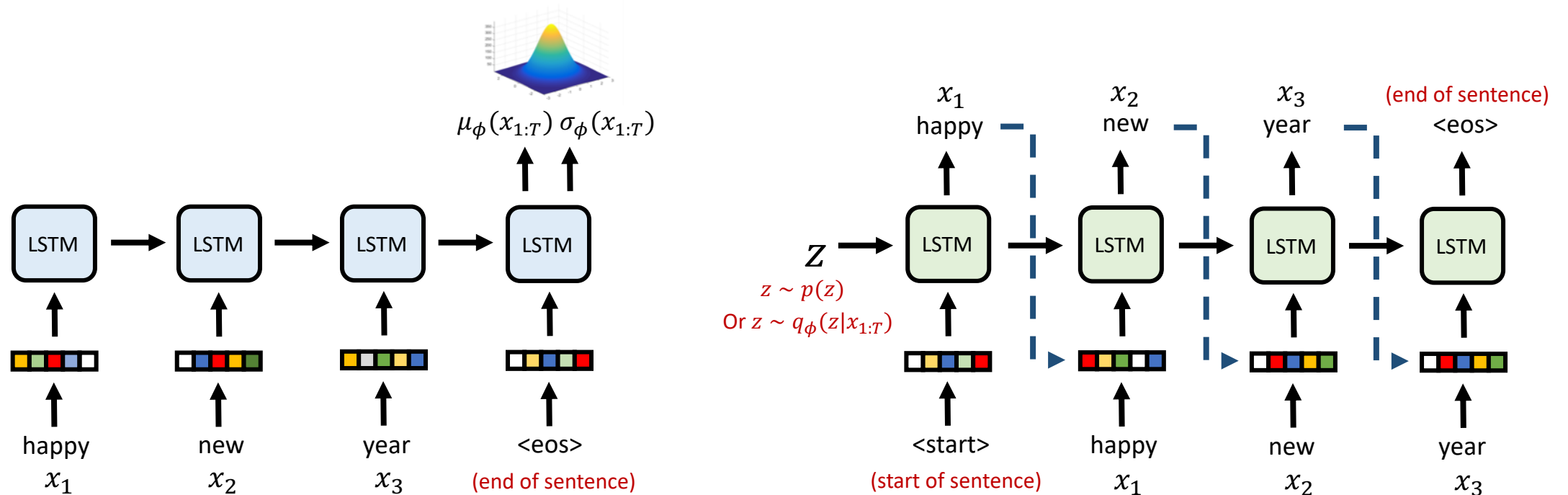
$$z \sim p(z), \quad x_{1:T} \sim p_{\theta}(x_{1:T}|z)$$
$$p_{\theta}(x_{1:T}) = \int p_{\theta}(x_{1:T}|z)p(z)dz$$

Sequence generation models

- Sequence VAE for language modelling:

Encoder: $q_\phi(z|x_{1:T}) = N(z; \mu_\phi(x_{1:T}), \text{diag}(\sigma_\phi^2(x_{1:T})))$

Generator: $p_\theta(x_{1:T}|z) = \prod_{t=1}^T p_\theta(x_t|x_{<t}, z)$

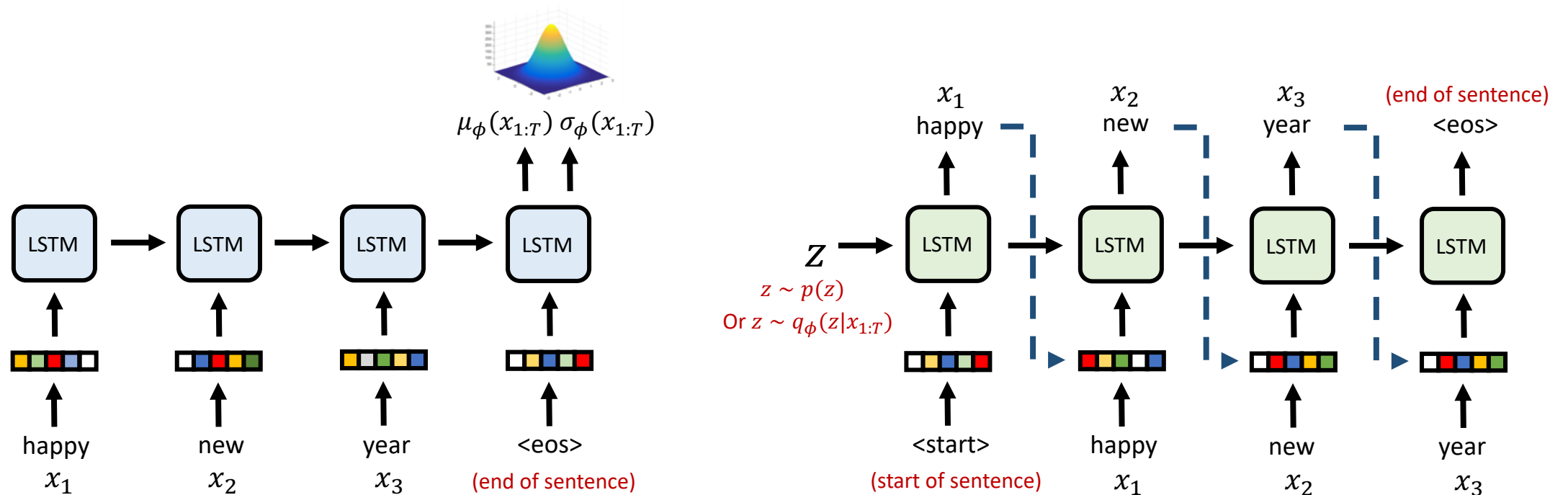


Sequence generation models

- Sequence VAE for language modelling:

$$= \sum_{t=1}^T \log p_{\theta}(x_t | x_{<t}, z)$$

$$L(\theta, \phi) = E_{p_{data}(x_{1:T})} [E_{q_{\phi}(z|x_{1:T})} [\log p_{\theta}(x_{1:T}|z)] - \beta KL[q_{\phi}(z|x_{1:T}) || p(z)]]$$



Sequence generation models

- Combining state-space models and RNNs:

State-space models:

- Stochastic dynamic model for the latent state:

$$p_{\theta}(z_{1:T}) = p_{\theta}(z_1) \prod_{t=2}^T p_{\theta}(z_t|z_{t-1})$$

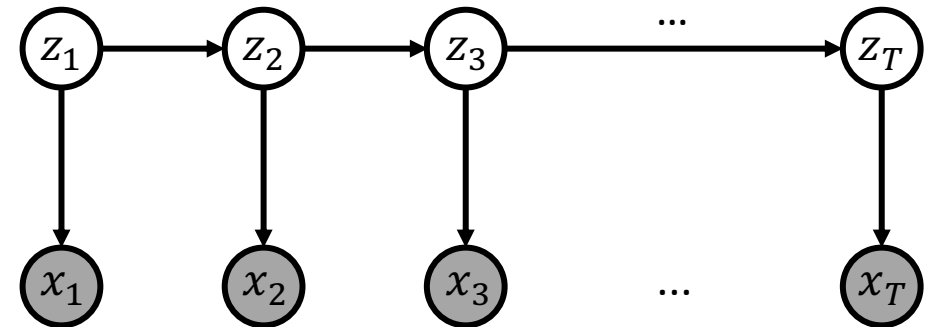
- Emission model:

$$p_{\theta}(x_t|z_t)$$

- Joint distribution:

$$p_{\theta}(x_{1:T}, z_{1:T}) = \prod_{t=1}^T p_{\theta}(z_t|z_{t-1})p_{\theta}(x_t|z_t)$$

(with the convention that $p_{\theta}(z_1|z_0) := p_{\theta}(z_1)$)



Sequence generation models

- Combining state-space models with RNNs:

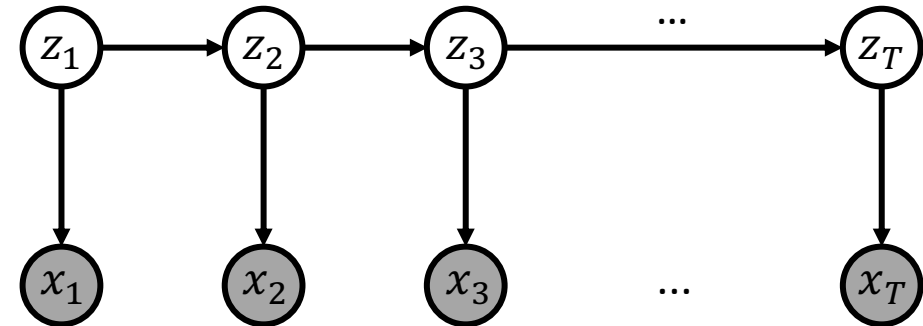
Example: Hidden Markov Model (HMM)

- Stochastic **linear** dynamic model for the latent state:

$$z_t = Az_{t-1} + B\epsilon_t, \epsilon_t \sim N(0, I)$$

- Linear Gaussian** emission model:

$$x_t = Cz_t + D\psi_t, \psi_t \sim N(0, I)$$



Sequence generation models

- Combining state-space models with RNNs:

State-space models + non-linear dynamics:

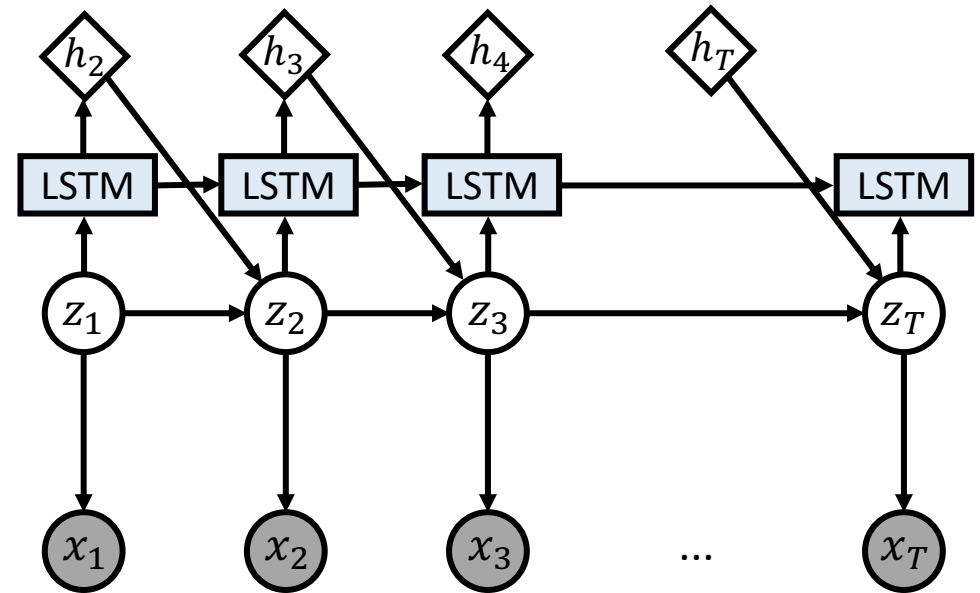
- Stochastic dynamic model **parameterized by RNNs**:

$$z_t = \mu_{\theta}^z(t) + \sigma_{\theta}^z(t)\epsilon_t, \epsilon_t \sim N(0, I),$$

$$\mu_{\theta}^z(t), \sigma_{\theta}^z(t) = NN_{\theta}(h_t^d), [h_t^d, c_t^d] = LSTM_{\theta}(z_{t-1}, h_{t-1}^d, c_{t-1}^d)$$

- Non-linear** emission model:

$$x_t = \mu_{\theta}^x(z_t) + \sigma_{\theta}^x(z_t)\psi_t, \psi_t \sim N(0, I)$$



Sequence generation models

- Combining state-space models with RNNs:

State-space models + non-linear dynamics:

- Stochastic dynamic model **parameterized by RNNs**:

$$p_{\theta}(z_{1:T}) = \prod_{t=1}^T p_{\theta}(z_t | z_{<t})$$

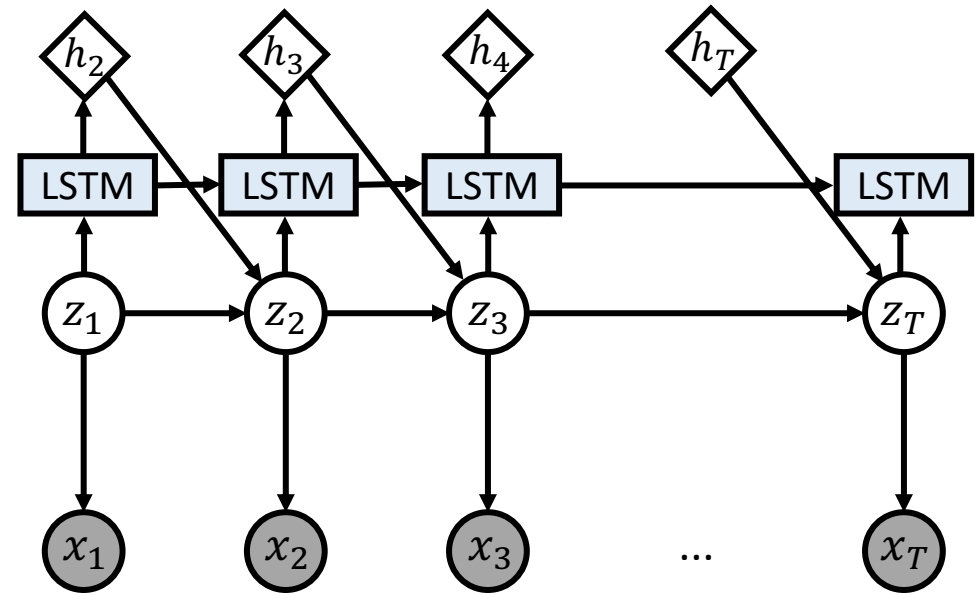
$p_{\theta}(z_t | z_{<t}) \neq p_{\theta}(z_t | z_{t-1})$ (LSTM makes all historical states relevant)

- Non-linear** emission model:

$$p_{\theta}(y_t | z_t): x_t = \mu_{\theta}^x(z_t) + \sigma_{\theta}^x(z_t)\psi_t, \psi_t \sim N(0, I)$$

- Joint distribution:

$$p_{\theta}(x_{1:T}, z_{1:T}) = \prod_{t=1}^T p_{\theta}(z_t | z_{<t}) p_{\theta}(x_t | z_t)$$



Sequence generation models

- Combining state-space models with RNNs:

Training:

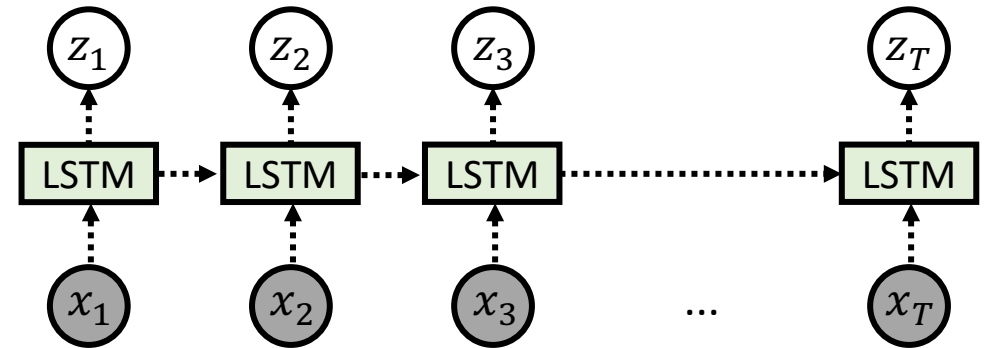
$$E_{p_{data}(x_{1:T})}[\log p_{\theta}(x_{1:T})] \geq E_{p_{data}(x_{1:T})}[E_{q_{\phi}(z_{1:T}|x_{1:T})}[\log p_{\theta}(x_{1:T}|z_{1:T})] - KL[q_{\phi}(z_{1:T}|x_{1:T})||p_{\theta}(z_{1:T})]]$$

Prior parameters to be learned!

- Generative model: $p_{\theta}(x_{1:T}, z_{1:T}) = \prod_{t=1}^T p_{\theta}(z_t|z_{<t})p_{\theta}(x_t|z_t)$
- Designing an LSTM-based encoder:

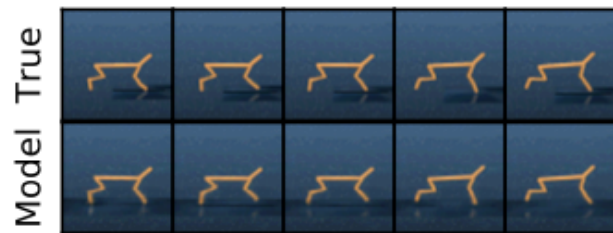
$$q_{\phi}(z_{1:T}|x_{1:T}) = \prod_{t=1}^T q_{\phi}(z_t|x_{\leq t})$$

$$q_{\phi}(z_t|x_{\leq t}) = N(z_t; \mu_{\phi}^z(h_t^e), \text{diag}(\sigma_{\phi}^z(h_t^e)^2))$$
$$[h_t^e, c_t^e] = LSTM_{\phi}(x_t, h_{t-1}^e, c_{t-1}^e)$$

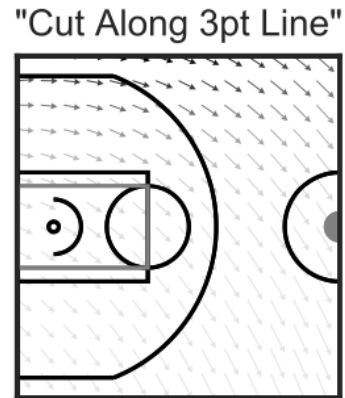


Sequence generation models

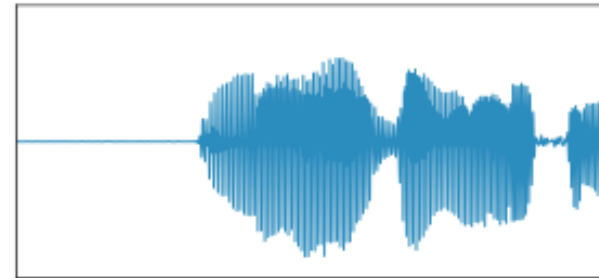
Neural state-space models have been applied to:



Model-based RL



Basketball player trajectory analysis



Speech synthesis

Fraccaro et al. Sequential Neural Models with Stochastic Layers. NeuIPS 2016
Linderman et al. Recurrent Switching Linear Dynamical Systems. AISTATS 2017
Hafner et al. Learning Latent Dynamics for Planning from Pixels. ICML 2019