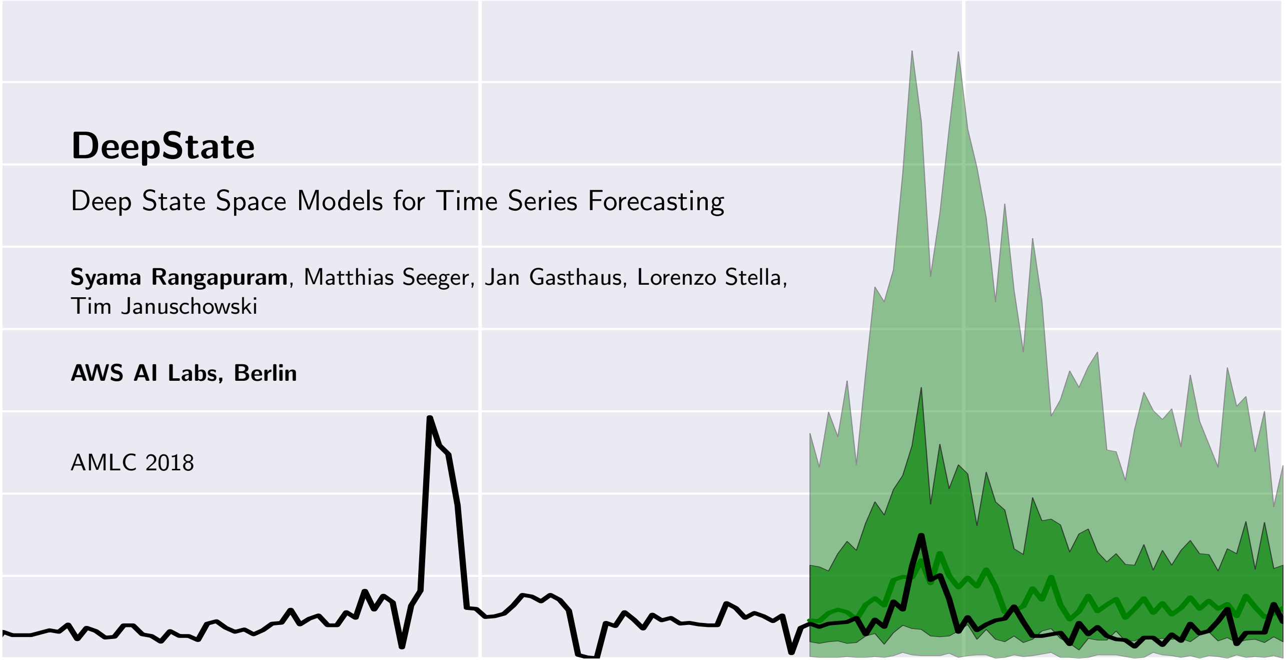
DeepState

Deep State Space Models for Time Series Forecasting

Syama Rangapuram, Matthias Seeger, Jan Gasthaus, Lorenzo Stella, Tim Januschowski

AWS AI Labs, Berlin

AMLC 2018



Forecasting Team


Lorenzo Stella


Matthias Seeger


Jan Gasthaus


Bernie Wang


David Salinas


Tim Januschowski

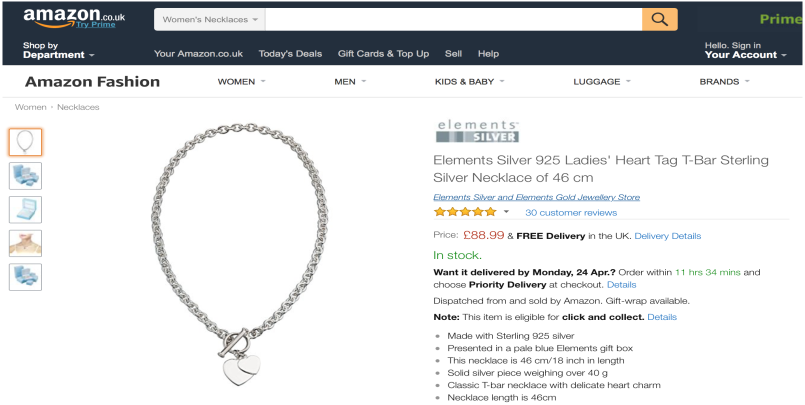

Syama Rangapuram

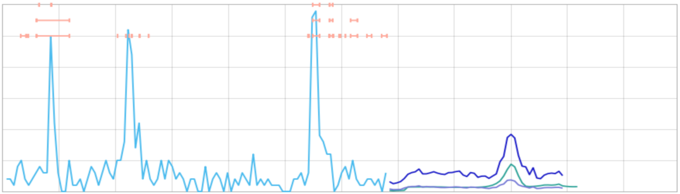

Valentin Flunkert


Alexander Alexandrov


Michael Bohlke-Schneider

Why Forecasting





Forecasting in a Nutshell

- Predict the future behaviour of a time series given its past

$$z_1, z_2, \dots, z_T \Longrightarrow P(z_{T+1}, z_{T+2}, \dots z_{T+\tau})$$

Forecasting in a Nutshell

- Predict the future behaviour of a time series given its past

$$z_1, z_2, \ldots, z_T \Longrightarrow P(z_{T+1}, z_{T+2}, \ldots z_{T+\tau})$$

- If I knew the future ...

$$\text{best action} = \operatorname*{argmin}_a \text{cost}(a, z_{T+1}, z_{T+2}, \ldots z_{T+\tau})$$

Forecasting in a Nutshell

- Predict the future behaviour of a time series given its past

$$z_1, z_2, \ldots, z_T \Longrightarrow P(z_{T+1}, z_{T+2}, \ldots z_{T+\tau})$$

- If I knew the future ...

$$\text{best action} = \underset{a}{\operatorname{argmin}} \operatorname{cost}(a, z_{T+1}, z_{T+2}, \ldots z_{T+\tau})$$

- If I have a calibrated forecast $P(z_{T+1}, z_{T+2}, \ldots z_{T+\tau})$ ...

$$\text{best action} = \underset{a}{\operatorname{argmin}} \mathrm{E}_{\mathrm{P}}[\operatorname{cost}(a, z_{T+1}, z_{T+2}, \ldots z_{T+\tau})]$$

Forecasting in a Nutshell

- Predict the future behaviour of a time series given its past

$$z_1, z_2, \ldots, z_T \Longrightarrow P(z_{T+1}, z_{T+2}, \ldots z_{T+\tau})$$

- If I knew the future ...

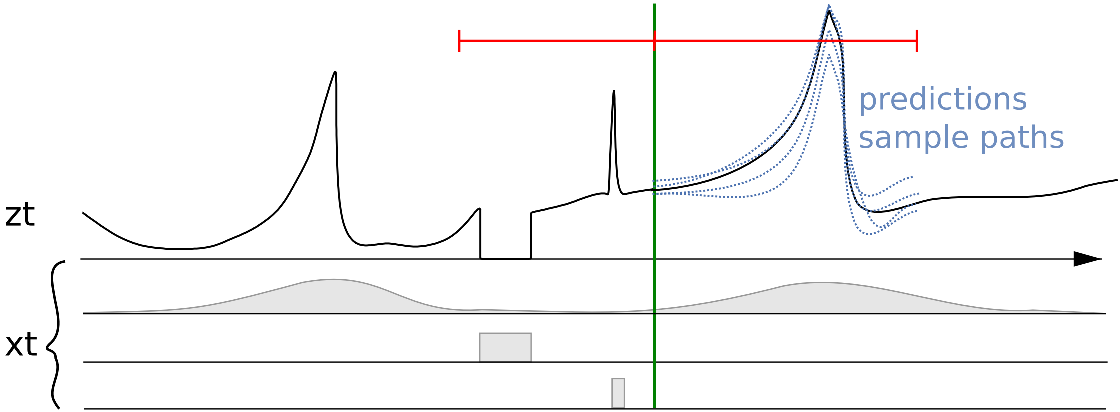
$$\text{best action} = \underset{a}{\operatorname{argmin}} \operatorname{cost}(a, z_{T+1}, z_{T+2}, \ldots z_{T+\tau})$$

- If I have a calibrated forecast $P(z_{T+1}, z_{T+2}, \ldots z_{T+\tau})$ ...

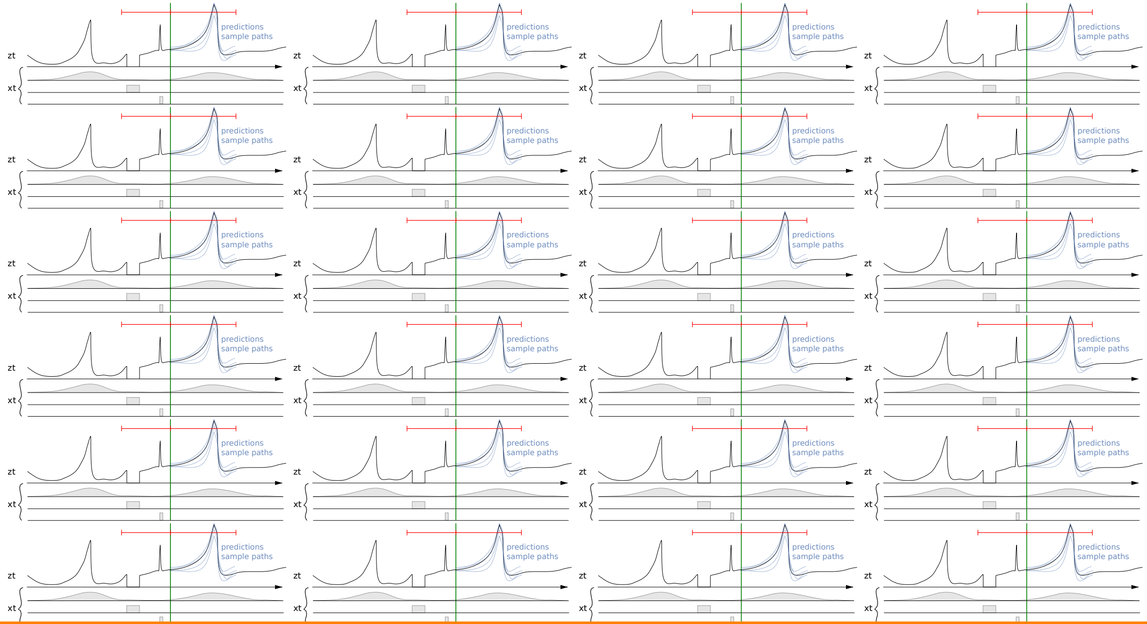
$$\text{best action} = \underset{a}{\operatorname{argmin}} \mathrm{E}_{\mathrm{P}}[\operatorname{cost}(a, z_{T+1}, z_{T+2}, \ldots z_{T+\tau})]$$

- Forecasts can be leveraged for optimising business problems in retail, AWS, logistics, professional services, etc.

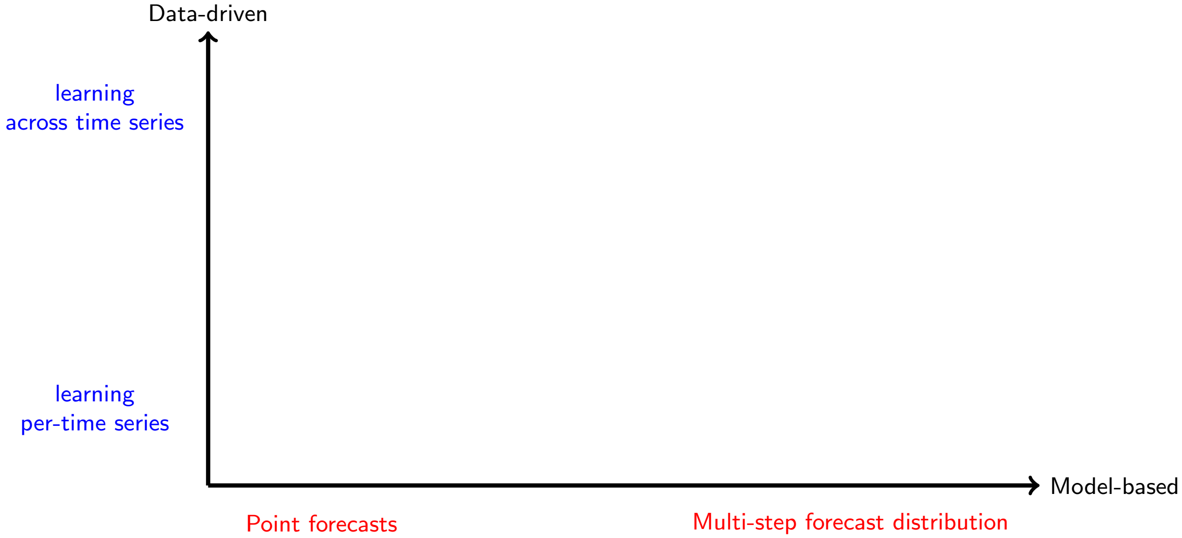
General Setup



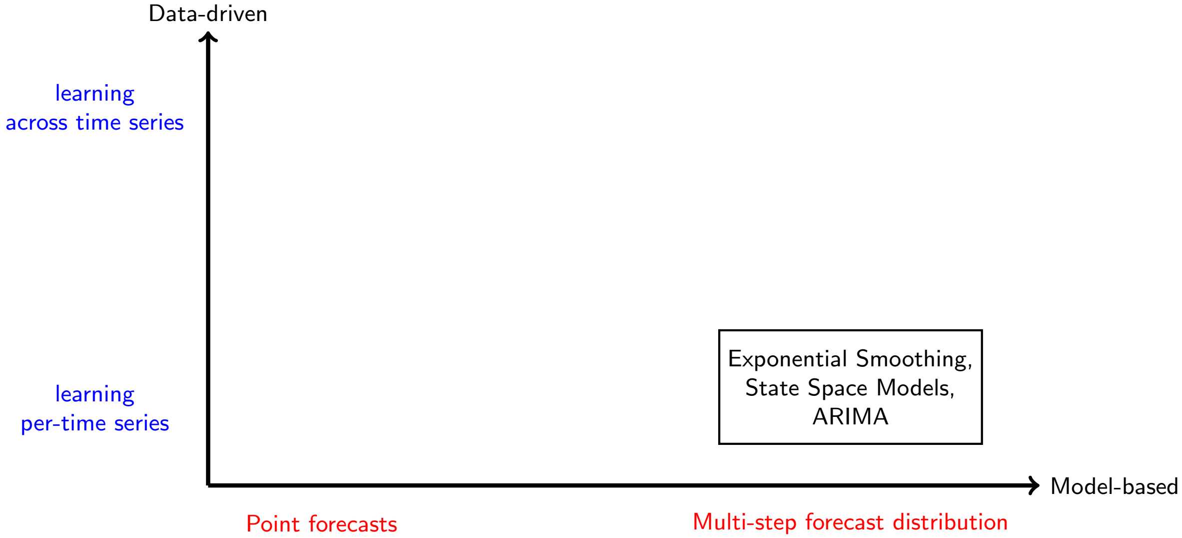
General Setup



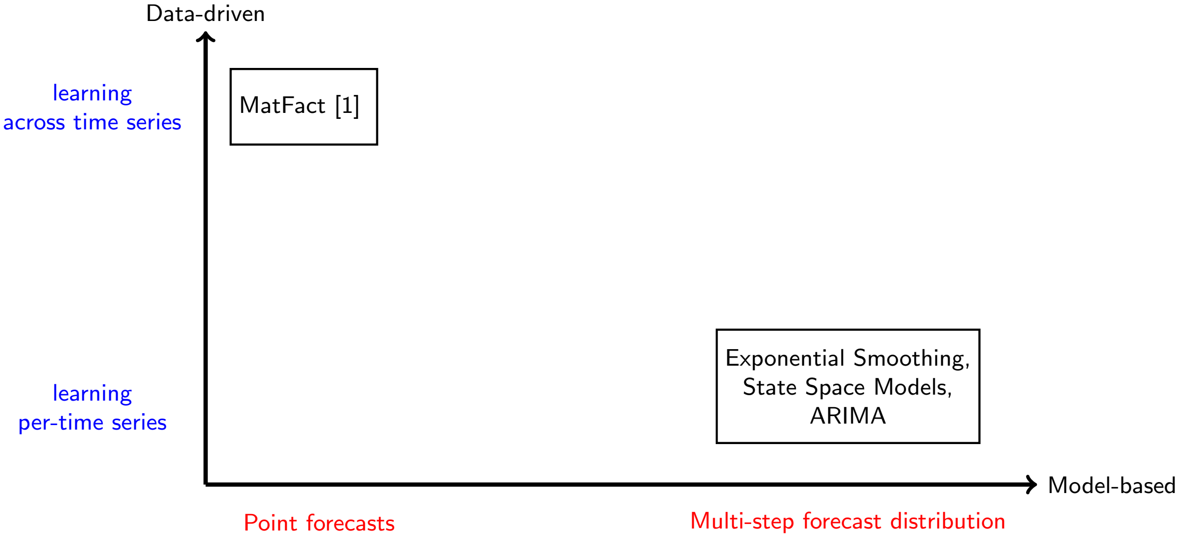
Forecasting Literature



Forecasting Literature

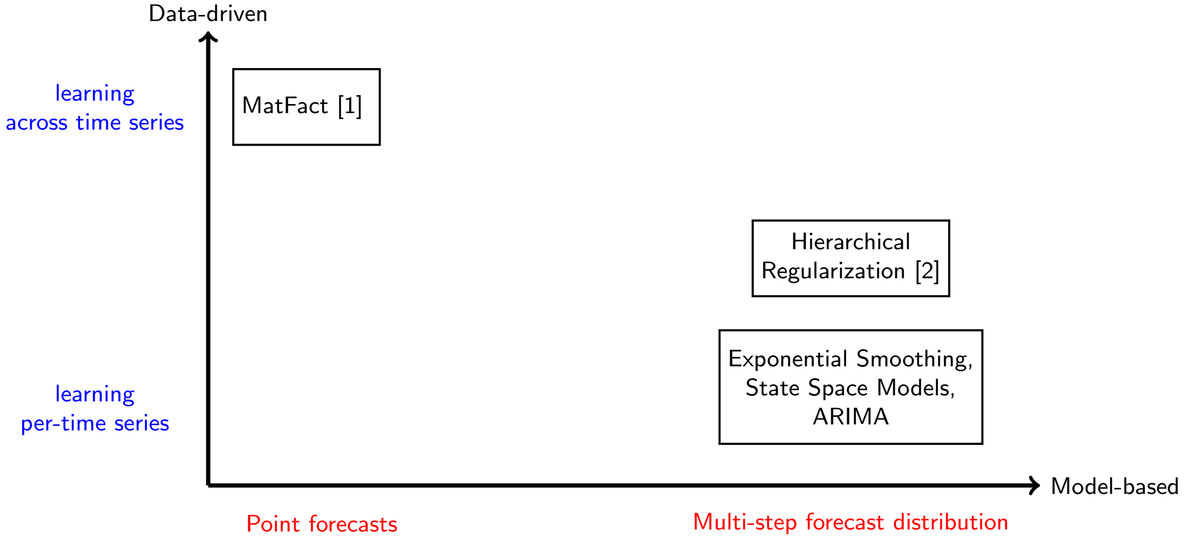


Forecasting Literature



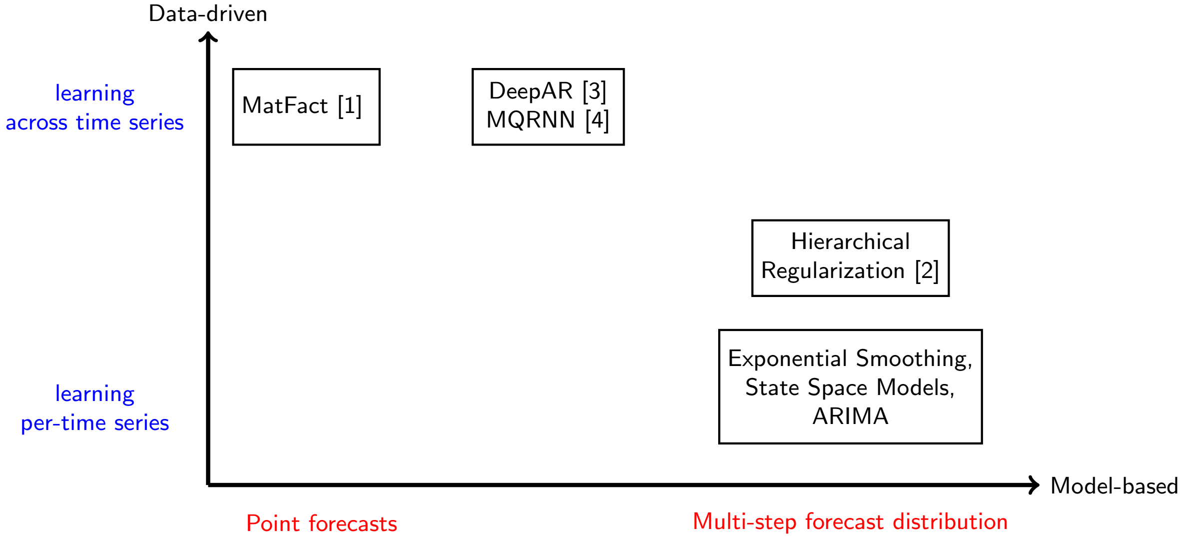
[1] Yu, Rao, Dhillon. Temporal Regularized Matrix Factorization for High-dimensional Time Series Prediction. NIPS 2016

Forecasting Literature



[2] N. Chapados. Effective bayesian modeling of groups of related count time series. ICML 2014

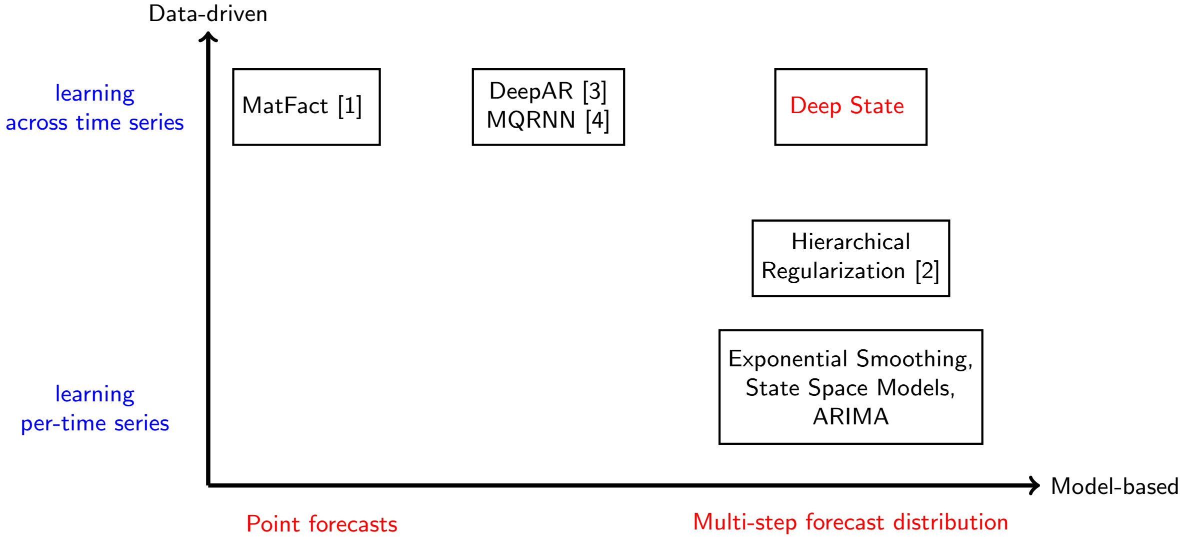
Forecasting Literature



[3] Flunkert, Salinas, Gasthaus. Probabilistic Forecasting with Autoregressive Recurrent Networks. arXiv 2017

[4] Wen, Torkkola, Narayanaswamy. A Multi-Horizon Quantile Recurrent Forecaster. arXiv 2017

Forecasting Literature



State Space Model

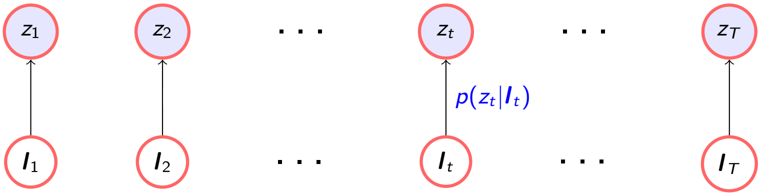
- Generative model for observations: $(z_1, \ldots, z_T)$

State Space Model

- Generative model for observations: $(z_1, \ldots, z_T)$

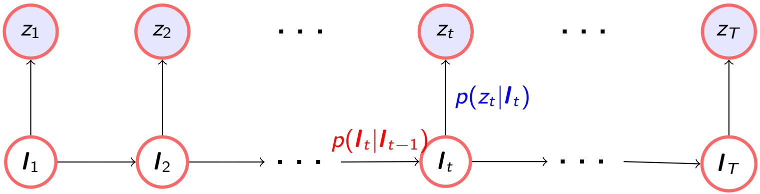
State Space Model

- Generative model for observations: $(z_1, \ldots, z_T)$



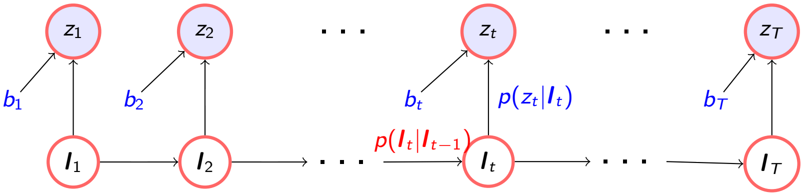
State Space Model

- Generative model for observations: $(z_1, \ldots, z_T)$



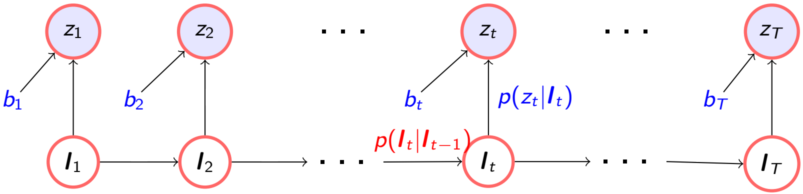
State Space Model

- Generative model for observations: $(z_1, \ldots, z_T)$



State Space Model

- Generative model for observations: $(z_1, \ldots, z_T)$



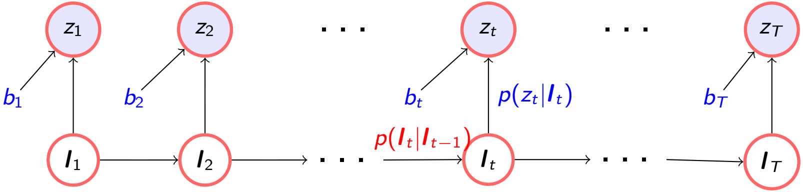
- Linear Gaussian Model

$$\boldsymbol{l}_t = \boldsymbol{l}_{t-1} + g_t \varepsilon_t, \qquad \varepsilon_t \sim N(0,1) \qquad \text{(state transistion)}$$

$$z_t = b_t + \boldsymbol{a}_t{}^T \boldsymbol{l}_{t-1} + \sigma_t \epsilon_t, \qquad \epsilon_t \sim N(0,1) \qquad \text{(measurements)}$$

State Space Model

- Generative model for observations: $(z_1, \ldots, z_T)$



- Linear Gaussian Model

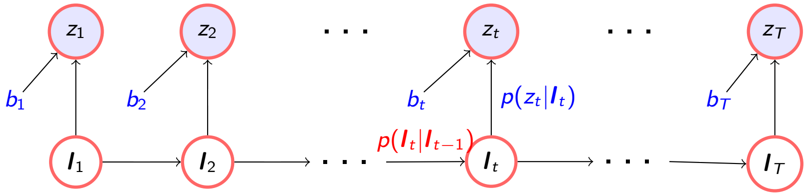
$$\boldsymbol{l}_t = \boldsymbol{l}_{t-1} + g_t \varepsilon_t, \qquad \varepsilon_t \sim N(0,1) \qquad \text{(state transistion)}$$

$$z_t = b_t + \boldsymbol{a}_t{}^T \boldsymbol{l}_{t-1} + \sigma_t \epsilon_t, \qquad \epsilon_t \sim N(0,1) \qquad \text{(measurements)}$$

- Free parameters: $\Theta = \{b_t, \boldsymbol{a}_t, \boldsymbol{g}_t, \sigma_t | \ \forall t\} \cup \{\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0\}$

State Space Model

- Generative model for observations: $(z_1, \dots, z_T)$



- Linear Gaussian Model

$$\boldsymbol{l}_t = \boldsymbol{l}_{t-1} + g_t \varepsilon_t, \qquad \varepsilon_t \sim N(0,1) \qquad \text{(state transistion)}$$

$$z_t = b_t + \boldsymbol{a}_t{}^T \boldsymbol{l}_{t-1} + \sigma_t \epsilon_t, \qquad \epsilon_t \sim N(0,1) \qquad \text{(measurements)}$$

- Free parameters: $\Theta = \{b_t, \boldsymbol{a}_t, \boldsymbol{g}_t, \sigma_t | \, \forall t\} \cup \{\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0\}$
- Θ learned using maximum likelihood principle

$$\Theta^* = \underset{\Theta}{\operatorname{argmax}}\, p_{\mathrm{SS}}(z_1, \dots, z_T | \Theta)$$

Deep State Space Model in a Nutshell

Similar features $\rightarrow$ similar state space models (modulo scale)

$$\Theta_t = \Psi(\mathbf{x}_{1:t}, \Phi), \quad \forall t$$

Deep State Space Model in a Nutshell

Similar features $\rightarrow$ similar state space models (modulo scale)

$$\Theta_t = \Psi(\mathbf{x}_{1:t}, \Phi), \quad \forall t$$

Generative Model:

Deep State Space Model in a Nutshell

Similar features $\rightarrow$ similar state space models (modulo scale)

$$\Theta_t = \Psi(\mathbf{x}_{1:t}, \Phi), \quad \forall t$$

Generative Model:

Features: $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$

Deep State Space Model in a Nutshell

Similar features $\rightarrow$ similar state space models (modulo scale)

$$\Theta_t = \Psi(\mathbf{x}_{1:t}, \Phi), \quad \forall t$$

Generative Model:

LSTM

$$\Theta_t = \theta(\mathbf{h}_t)$$
$$\mathbf{h}_t = \psi(\mathbf{h}_{t-1}, \mathbf{x}_t, \Phi)$$

$$\uparrow$$

Features: $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$

Deep State Space Model in a Nutshell

Similar features $\rightarrow$ similar state space models (modulo scale)

$$\Theta_t = \Psi(\mathbf{x}_{1:t}, \Phi), \quad \forall t$$

Generative Model:

SSM

$$\mathbf{z} \sim p_{\mathrm{SS}}(z_{1:T} | \{\Theta_t\})$$

$\Theta_1, \Theta_2, \ldots, \Theta_T$

LSTM

$$\Theta_t = \theta(\mathbf{h}_t)$$

$$\mathbf{h}_t = \psi(\mathbf{h}_{t-1}, \mathbf{x}_t, \Phi)$$

Features: $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$

Deep State Space Model in a Nutshell

Similar features $\rightarrow$ similar state space models (modulo scale)

$$\Theta_t = \Psi(\mathbf{x}_{1:t}, \Phi), \quad \forall t$$

Generative Model:

$z_1, \ldots, z_T$

$\uparrow$

SSM

$$\mathbf{z} \sim p_{\mathrm{SS}}(z_{1:T} | \{\Theta_t\})$$

$\uparrow$ $\Theta_1, \Theta_2, \ldots, \Theta_T$

LSTM

$$\Theta_t = \theta(\mathbf{h}_t)$$
$$\mathbf{h}_t = \psi(\mathbf{h}_{t-1}, \mathbf{x}_t, \Phi)$$

$\uparrow$

Features: $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$

DeepState - Training

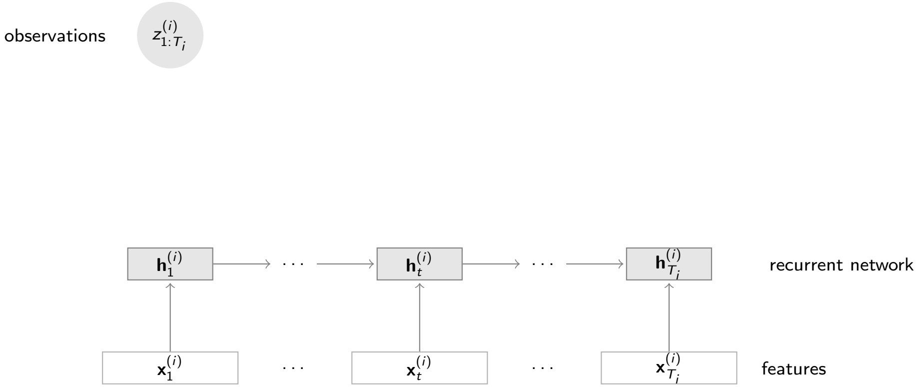
- **Loss:** negative log-likelihood of the data under our model given features

observations $z^{(i)}_{1:T_i}$

$\mathbf{x}^{(i)}_1$ $\cdots$ $\mathbf{x}^{(i)}_t$ $\cdots$ $\mathbf{x}^{(i)}_{T_i}$ features

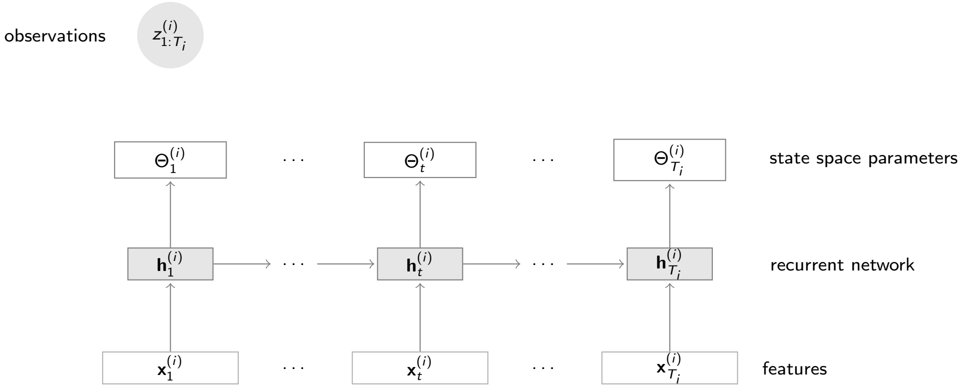
DeepState - Training

- Loss: negative log-likelihood of the data under our model given features



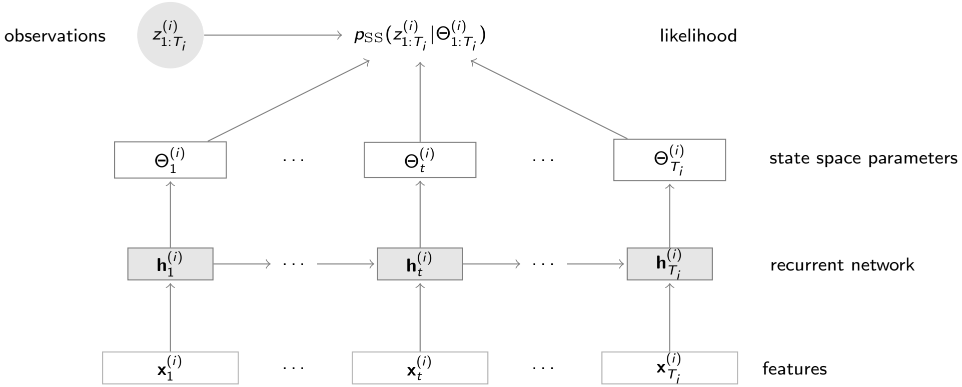
DeepState - Training

- **Loss:** negative log-likelihood of the data under our model given features



DeepState - Training

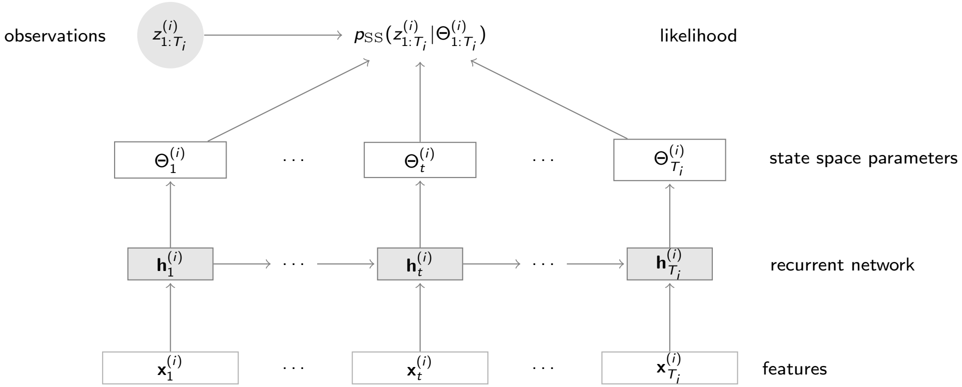
- Loss: negative log-likelihood of the data under our model given features



- Kalman filtering in the case of linear Gaussian model

DeepState - Training

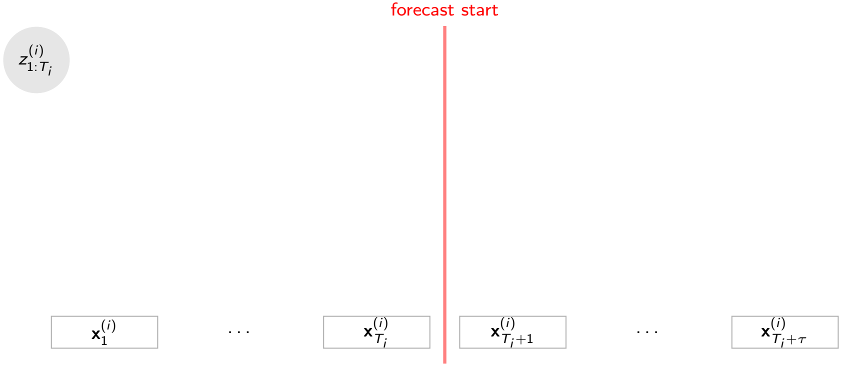
- Loss: negative log-likelihood of the data under our model given features



- Kalman filtering in the case of linear Gaussian model
- Robust to outliers and can handle missing data (z_{t-1} is not fed back as feature for time step t)

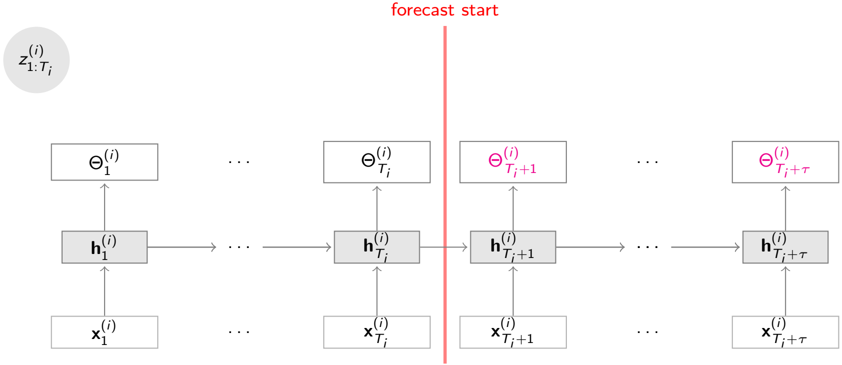
DeepState - Prediction

Test time series (possibly new/unseen) with features in both training and prediction ranges



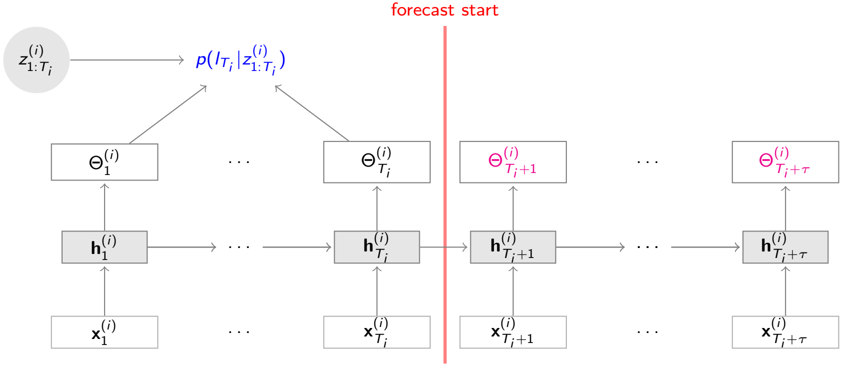
DeepState - Prediction

Test time series (possibly new/unseen) with features in both training and prediction ranges



DeepState - Prediction

Test time series (possibly new/unseen) with features in both training and prediction ranges

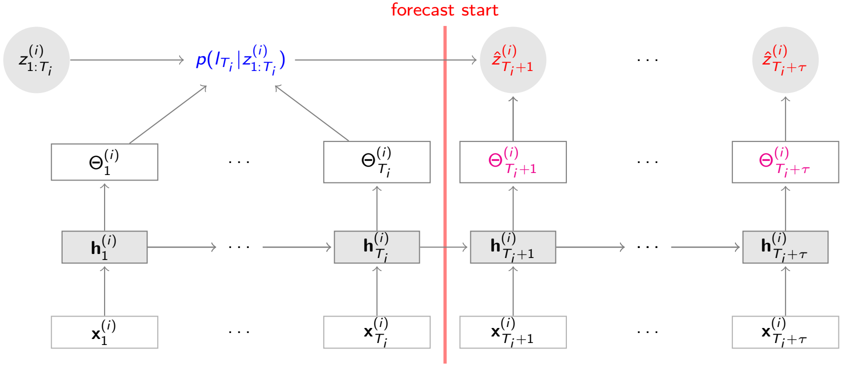


- Given the posterior of the final state and state space parameters, one can obtain the forecast distribution

$$P(z_{T_i+1}, z_{T_i+2}, \ldots, z_{T_i+\tau} | z_1, z_2, \ldots, z_{T_i})$$

DeepState - Prediction

Test time series (possibly new/unseen) with features in both training and prediction ranges



- Given the posterior of the final state and state space parameters, one can obtain the forecast distribution

$$P(z_{T_i+1}, z_{T_i+2}, \ldots, z_{T_i+\tau} | z_1, z_2, \ldots, z_{T_i})$$

- Analytical form or samples (sampling here is much faster than in DeepAR)

Data Efficiency:

- **Public datasets:**
  - `traffic`: 963 hourly time series measuring occupancy rates of car lanes
  - `electricity`: 370 hourly time series measuring electricity consumption of different consumers
- **Forecast horizon:** 7 days
- **Metrics:** P50QL, P90QL: normalized quantile regression loss

	2-weeks		3-weeks		Full-data	
	P50QL	P90QL	P50QL	P90QL	P50QL	P90QL
`DeepAR`	0.177	0.153	0.126	0.096	0.132	0.104
`DeepState`	**0.126**	**0.069**	**0.125**	**0.068**	**0.124**	**0.07**

Results on `traffic` dataset

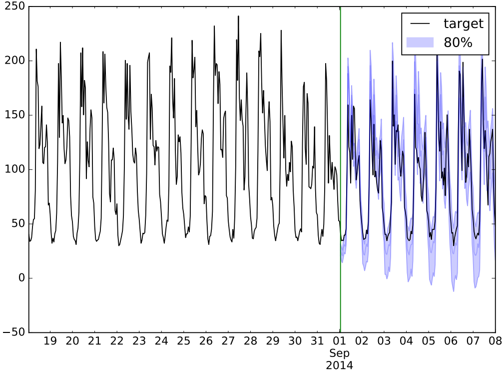
Data Efficiency:

- **Public datasets:**
  - `traffic`: 963 hourly time series measuring occupancy rates of car lanes
  - `electricity`: 370 hourly time series measuring electricity consumption of different consumers
- **Forecast horizon:** 7 days
- **Metrics:** P50QL, P90QL: normalized quantile regression loss

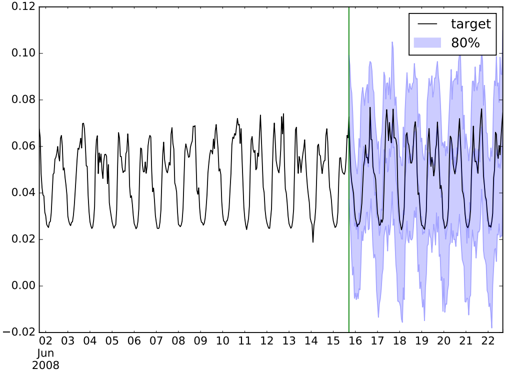
	2-weeks		3-weeks		Full-data	
	P50QL	P90QL	P50QL	P90QL	P50QL	P90QL
`DeepAR`	0.153	0.147	0.147	0.132	**0.09**	0.051
`DeepState`	**0.091**	**0.049**	**0.095**	**0.051**	0.093	**0.049**

Results on `electricity` dataset

Example Predictions



electricity



traffic

More quantitative experiments

- **Public datasets:**
  - `traffic`: 963 hourly time series measuring occupancy rates of car lanes
  - `electricity`: 370 hourly time series measuring electricity consumption of different consumers
- **Forecast horizon:** 24 hours
- **Metrics:** P50QL/P90QL (Matrix factorization method only produces point forecasts)

	`electricity`	`traffic`
`MatFact`	0.16/-	0.20/-
`DeepAR`	0.079/**0.0415**	**0.128**/0.0961
`DeepState`	**0.076**/0.0418	0.14/**0.0957**

Average P50QL/P90QL for rolling day prediction for seven days

Conclusion

DeepState

- New approach by marrying state space models with deep recurrent neural networks
- Explicitly incorporates structural assumptions and hence data-efficient
- More robust to outliers and can handle missing data
- Learns a joint global model and can forecast time series with little history (cold start problem)

Further directions

- Extensions to other likelihoods