Time Series Modeling using Neural Networks

Dušan Fedorčák

11/2021 – RBI

Machine Learning College

Background

- Ph.D. in computer science at VŠB-TU Ostrava
  - Neural networks & unsupervised self-organization
- Experienced in simulations
  - flood prediction system for MSK
  - traffic monitoring & prediction systems
- Experienced in computer graphics & scientific visualization
  - GIS related real-time 3D visualizations
- 5+ years in applied ML and artificial intelligence
  - Lead researcher in GoodAI – general artificial intelligence
  - CTO in Neuron Soundware – sound processing via Deep Learning
  - Lead ML in Merlon Intelligence Inc. – natural language processing

Machine Learning College

Content

DAY 1

- **Classical time series analysis**
  - *Decomposition of time series*
  - *ARIMA models family*
  - *State space models generalization*
- **Theoretical window**
  - *Neural Networks & Recurrent NNs*
  - *Time series specifics*
- **Practical examples**
  - *Simple regression – toy example*
  - *Rainfall-runoff simulation – regression*

– lunch break –

- **Practical examples**
  - *Trampoline jumping – classification*
  - *Local Weather Forecast – regression*

DAY 2

- **Product Design & ML**
  - *Integration of ML models into products*
  - *Tips & tricks for debuging NNs*
- **Practical Examples** (in random order)
  - *Exoplanets Hunting*
  - *Mobile Motion Sensing*

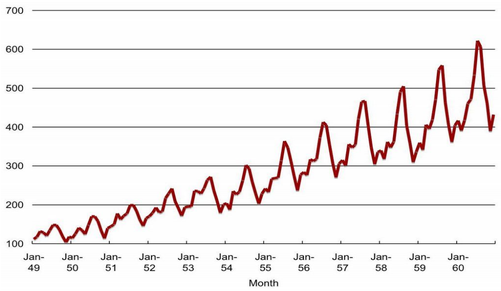
– lunch break –

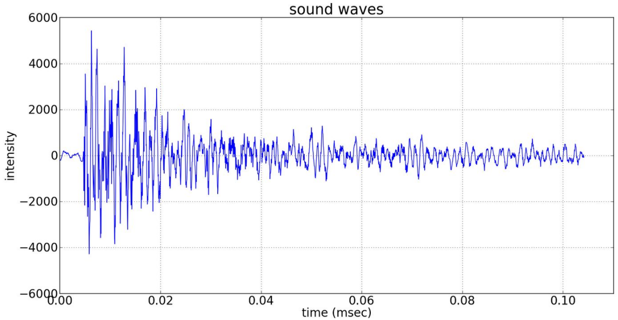
  - *Manufacturing Process Modeling*
  - *Financial distress prediction*

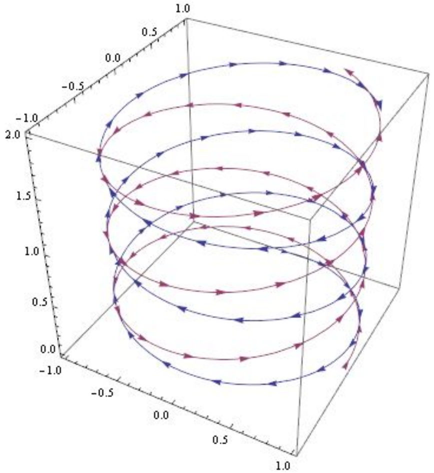
- *Google Drive Folder with data*
- *GitHub repository with example sources*

Machine Learning College

Time series – example data

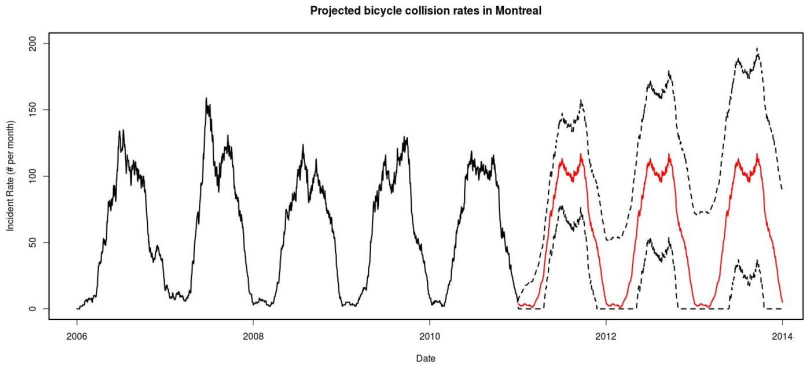


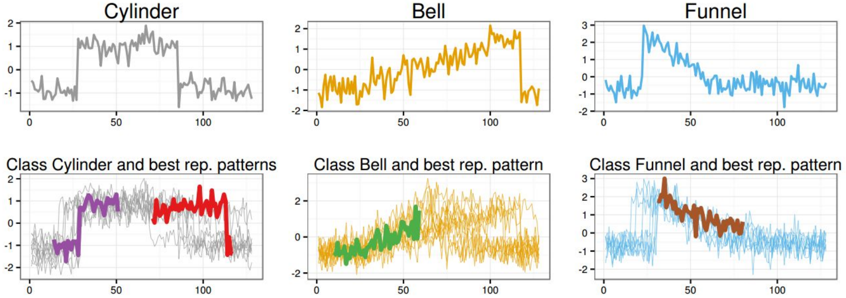


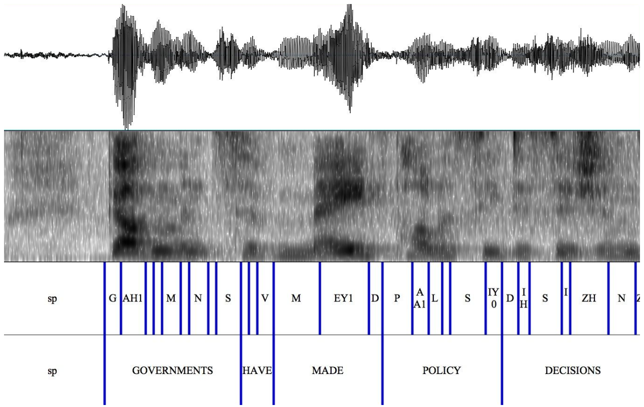


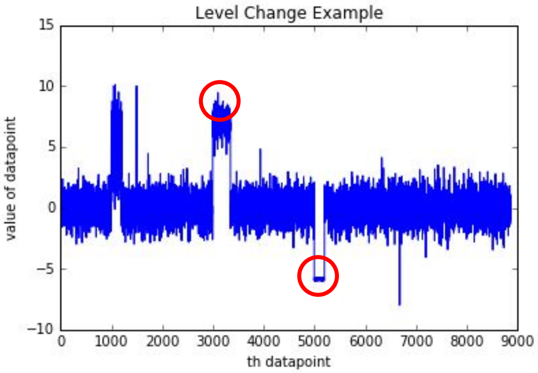
Machine Learning College

Time series – example tasks





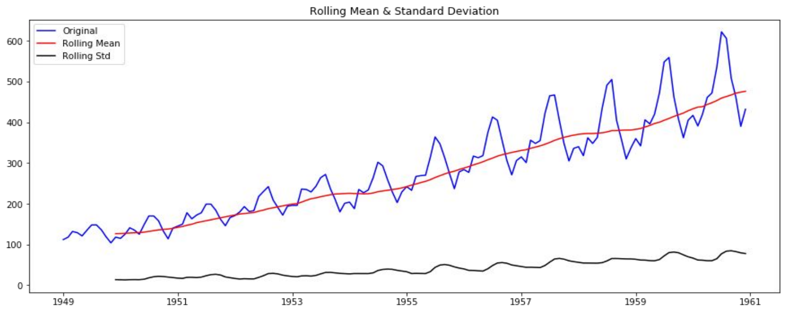


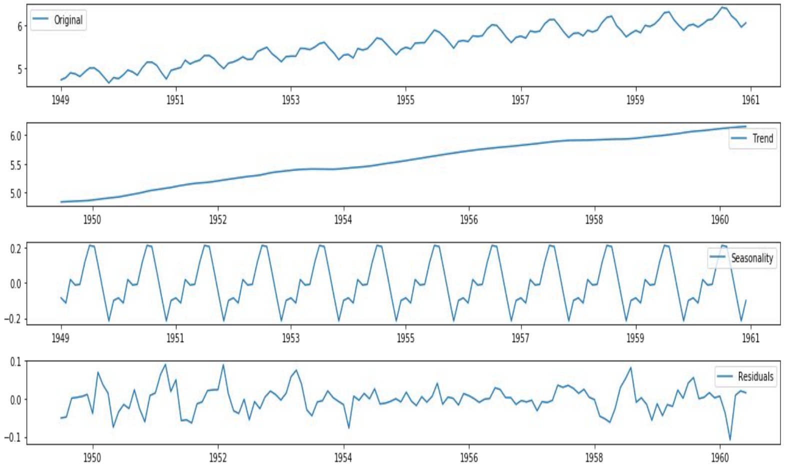


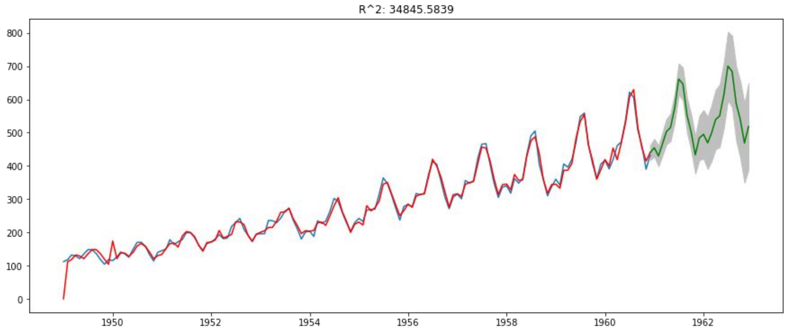
Machine Learning College

Time Series – classical analysis & modeling

- Time Series Decomposition
  - Inflation, trend, seasonality, differencing
- ARIMA models
  - http://people.duke.edu/~rnau/411home.htm







Machine Learning College

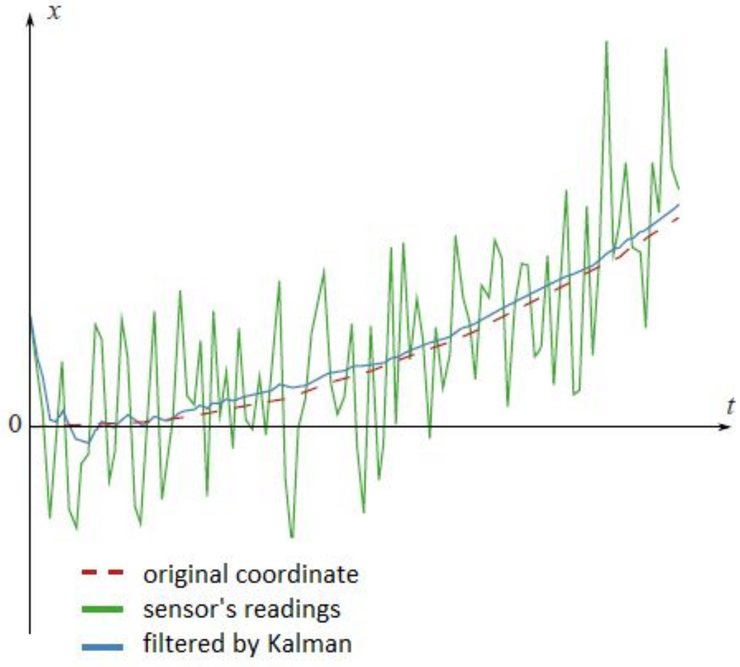
State Space Models

- State Space Models
  - A dynamic system that **evolves over time**
  - Knowing **the current state of the model is enough** to predict the future
  - The true state of the system might **not** be **directly observable**
- Model Description
  - State
    - $\boldsymbol{x}_t \sim N(\boldsymbol{x}_t, \boldsymbol{P}_t)$
  - State Equation
    - $\boldsymbol{x}_t = \boldsymbol{F}\boldsymbol{x}_{t-1} + N(0, \boldsymbol{Q})$ – *sometimes without noise*
  - Observation Equation
    - $\boldsymbol{y}_t = \boldsymbol{H}\boldsymbol{x}_t + N(0, \boldsymbol{R})$

Machine Learning College

Kalman Filter

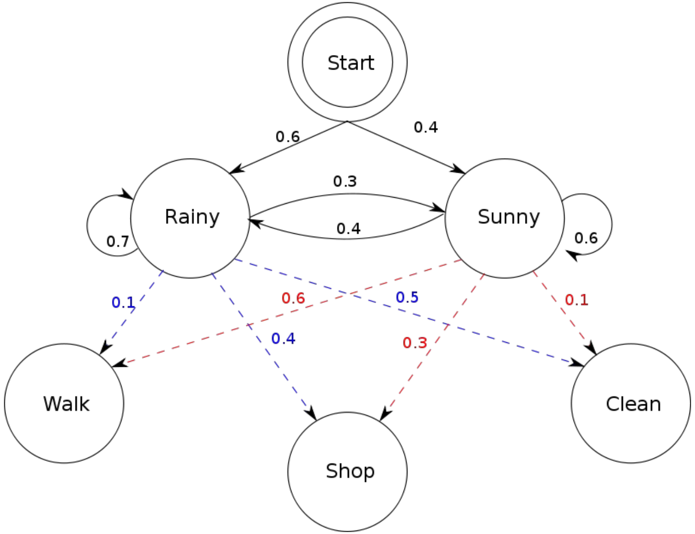
- Evolve state
  - $x'_t = Fx_{t-1}$
  - $P'_t = F\,P_{t-1}F^T + Q$
- Integrate observation
  - $x_t = x'_t + K_t(y_t - Hx'_t)$
  - $P_t = (I - K_tH)\,P'_t$
- **Kalman Gain**
  - $K_t = P'_t\,H^T(HP'_tH^T + R)^{-1}$
- ARIMA and Kalman Filter
  - ARIMA can be viewed as a state space model
  - ARIMA can be fitted with MLE via Kalman Filter
  - https://bookdown.org/rdpeng/timeseriesbook/maximum-likelihood-with-the-kalman-filter.html
  - https://towardsdatascience.com/the-kalman-filter-and-maximum-likelihood-9861666f6742



Machine Learning College

Hidden Markov Model

- Model Description
  - HMM (**λ**) can be viewed as a **state space model**
  - Finite set of hidden states
    - $\mathbf{Q} = \{q_1, q_2, \dots, q_n\}$, $\boldsymbol{\pi} = \{\pi_1, \pi_2, \dots, \pi_n\}$ – init
    - ***n*** – number of states (hyperparameter)
  - Set of observations
    - $\mathbf{O}_i = (o^1, o^2, o^3, \dots, o^T)$
  - Transition probability matrix & emissions
    - $\mathbf{A} = (a_{00}, \dots, a_{nn})$, $\mathbf{B} = q_i \rightarrow o$



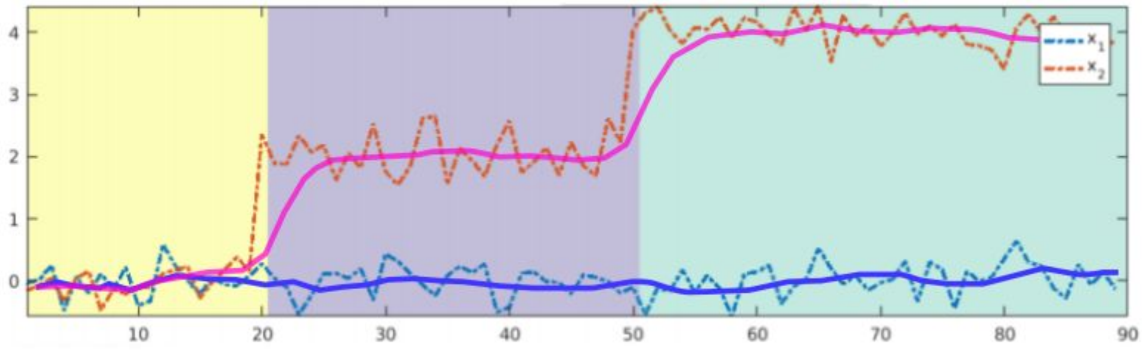
- Model Capabilities
  - $P(\mathbf{O}|\boldsymbol{\lambda})$ – Give prob. of **O** being produced by **λ** – *forward-backward alg.*
  - $P(\mathbf{q}_1, \dots, \mathbf{q}_t|\mathbf{O}, \boldsymbol{\lambda})$ – Give most likely sequence of states for given **O** – *Viterbi alg.*
  - **O ⇒ λ** – Model must be trainable with **O** – *Baum-Welch alg.*

Machine Learning College

Kalman Filter vs. Hidden Markov Model

- **Kalman Filter**
  - Continuous state
  - Generic state & observation equation
  - Linear dynamic system
  - *Fusion of sensor readings and controls*
  - *ARMA models implementation*

- **Hidden Markov Model**
  - Discrete set of states
  - N-states hyperparameter
  - Emission & Transition tables
  - *Speech recognition*
  - *Time series segmentation*



http://lasa.epfl.ch/teaching/lectures/ML_MSc_Advanced/Slides/Lec_XI_HMM_extensions.pdf

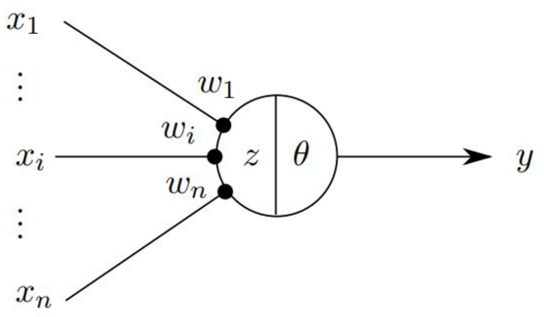
Machine Learning College

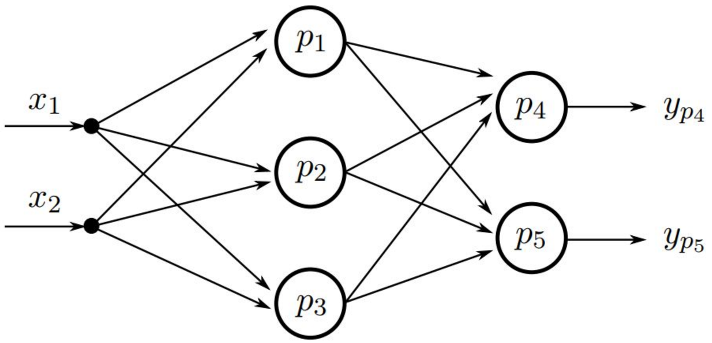
Time Series – goal in classical terminology

- **Forecasting**
  - Given the past and the present observation, what will the future look like?
- **Time scale analysis**
  - Given the observations, what time scales dominate when observing temporal variation in the data
- **Filtering**
  - Given the past and the present observation, how should I update my estimate of the true state of nature?
- **Smoothing**
  - Given a complete dataset, what can I infer about the true state of nature in the past?
- **Regression**
  - Given a time series of two phenomena, what is the association between them?

https://bookdown.org/rdpeng/timeseriesbook/

Neural networks

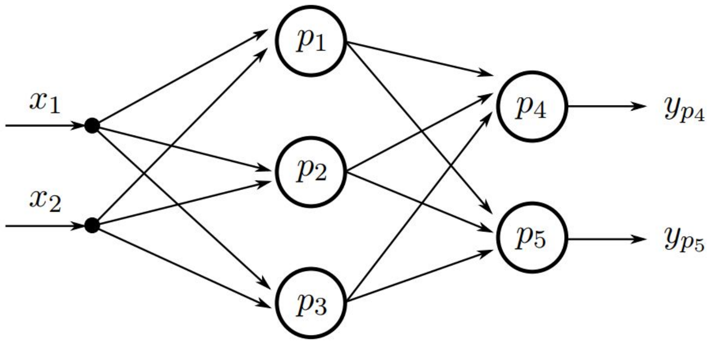


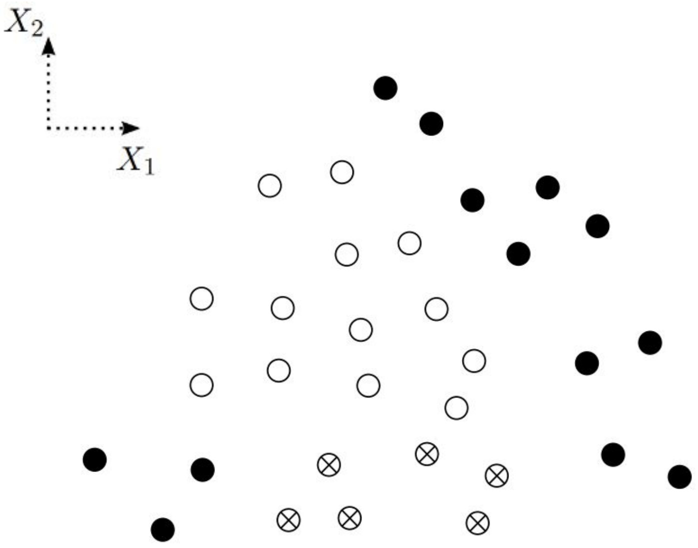


- Artificial Neural Cells
  - Linear combination of inputs
  - Non-linear activation function
- Connected Neurons
  - Directed graph
  - Layered structure
    - Dense connections
    - Convolutions & pooling
    - Recurrency, signal gates
    - Masking & attention heads
- Universal function approximator
  - Trainable with data
  - Backpropagation
  - Deep vs. shallow architecture

Machine Learning College

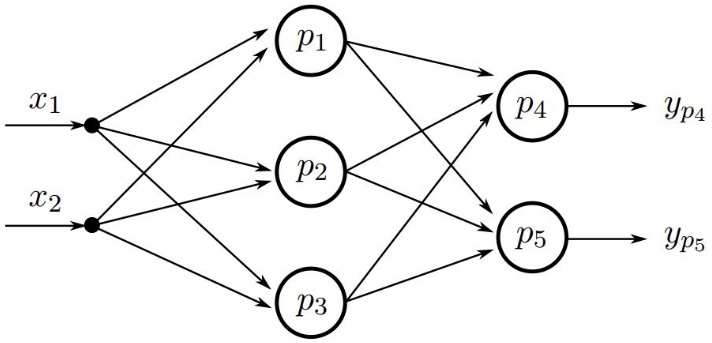
Neural networks – Input space separation

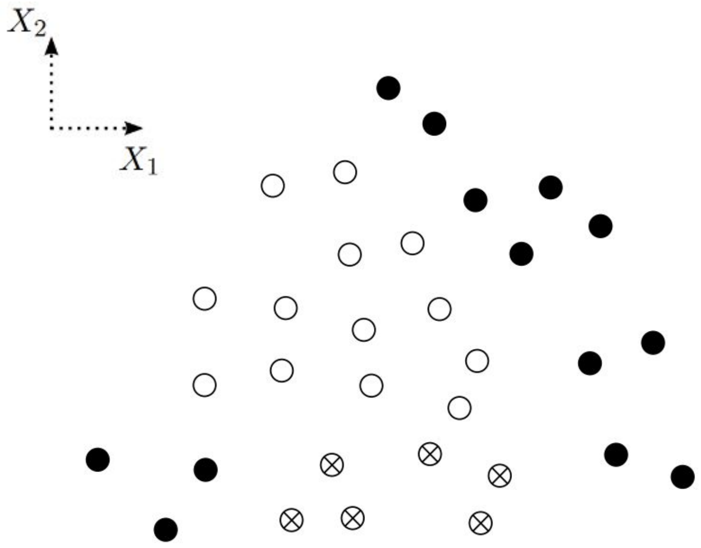




Machine Learning College

Neural networks – Input space separation

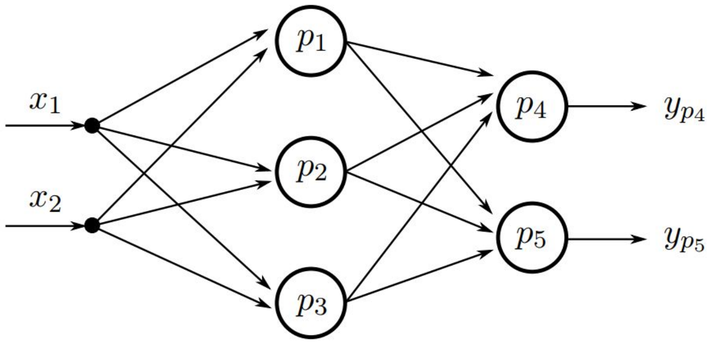


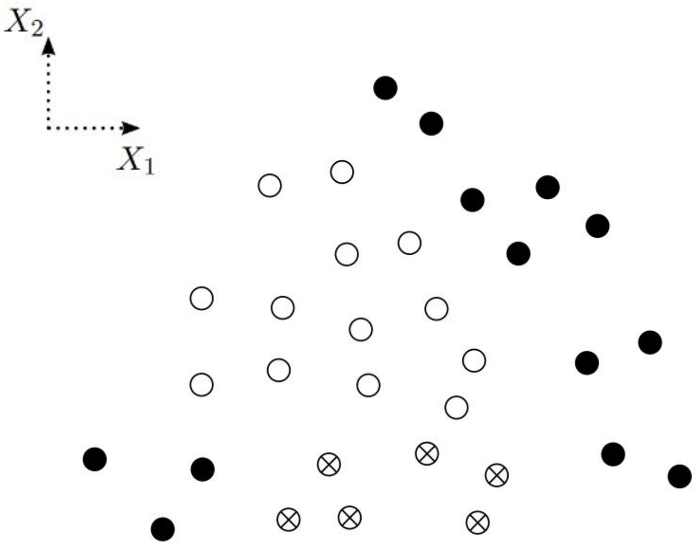


$$y = s(\Sigma\, w_i x_i - \theta)$$

Machine Learning College

Neural networks – Input space separation

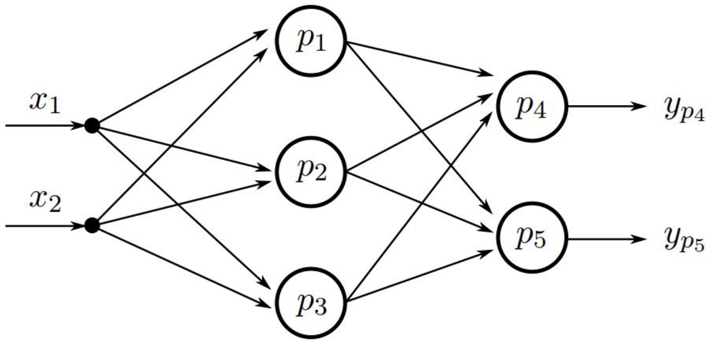


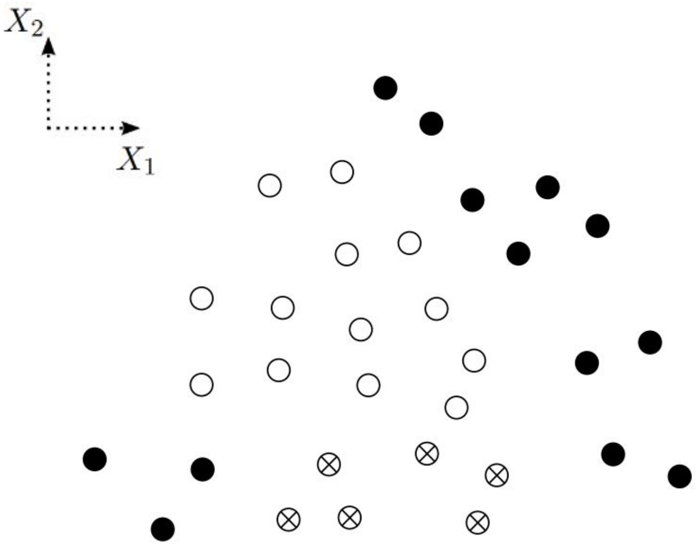


$$y = s(\Sigma w_i x_i - \theta) = s(w_1 x_1 + w_2 x_2 - \theta)$$

Machine Learning College

Neural networks – Input space separation



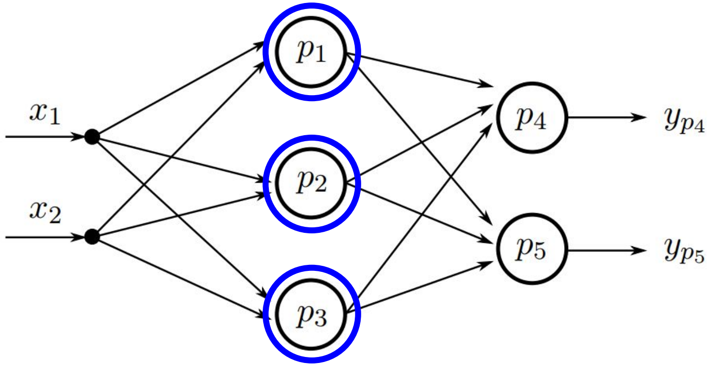


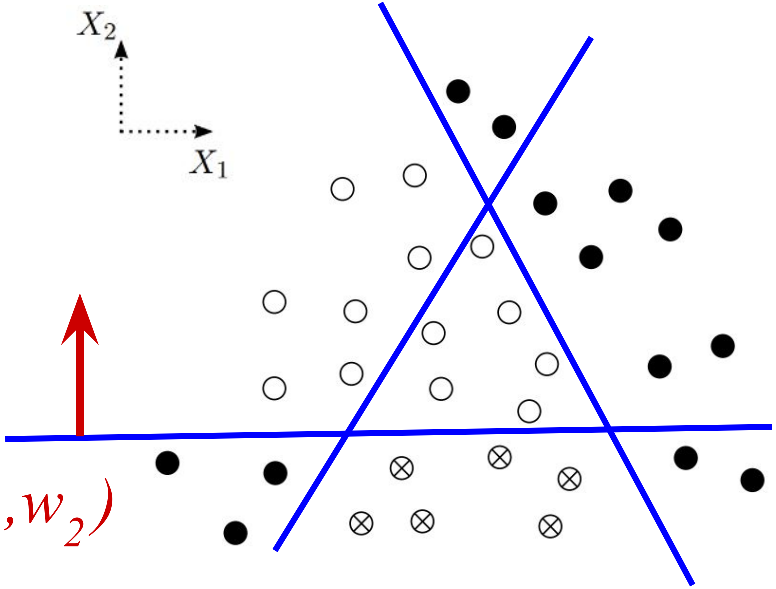
$$ax + by + c = 0$$

$$y = s(\Sigma w_i x_i - \theta) = s(w_1 x_1 + w_2 x_2 - \theta)$$

Machine Learning College

Neural networks – Input space separation





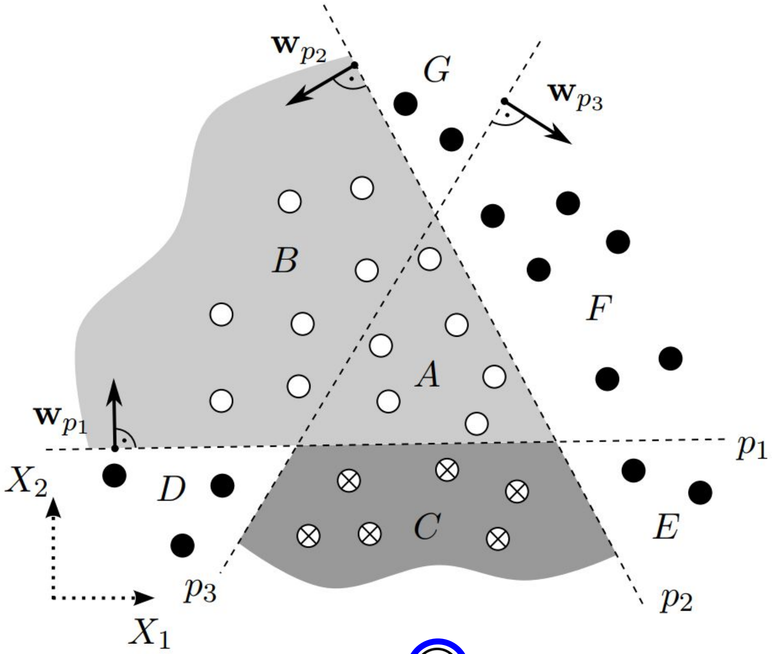
$$(a,b) = (w_1, w_2)$$

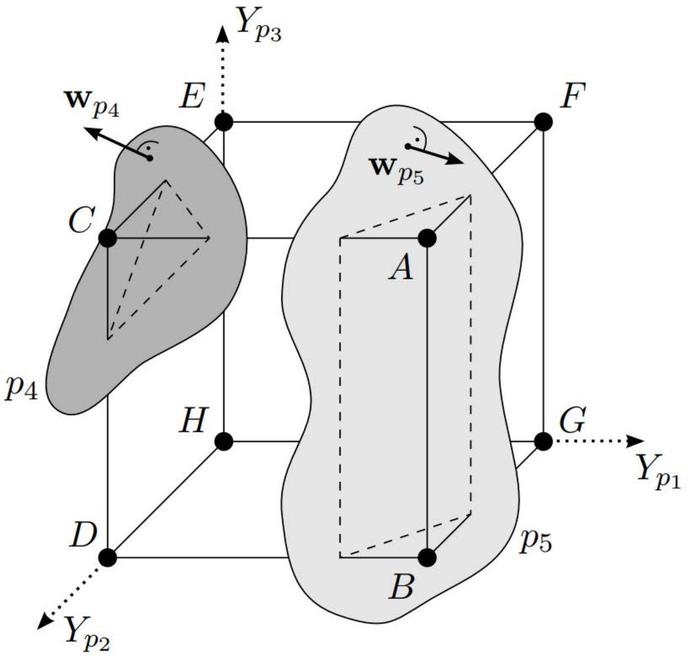
$$ax + by + c = 0$$

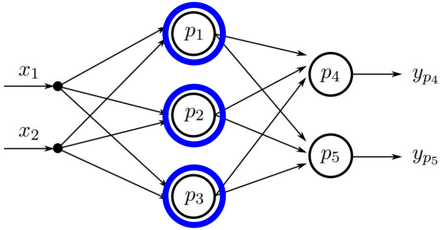
$$y = s(\Sigma\, w_i x_i - \theta) = s(w_1 x_1 + w_2 x_2 - \theta)$$

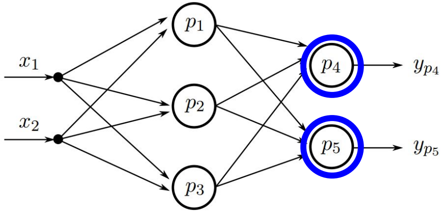
Machine Learning College

Neural networks – Input space separation

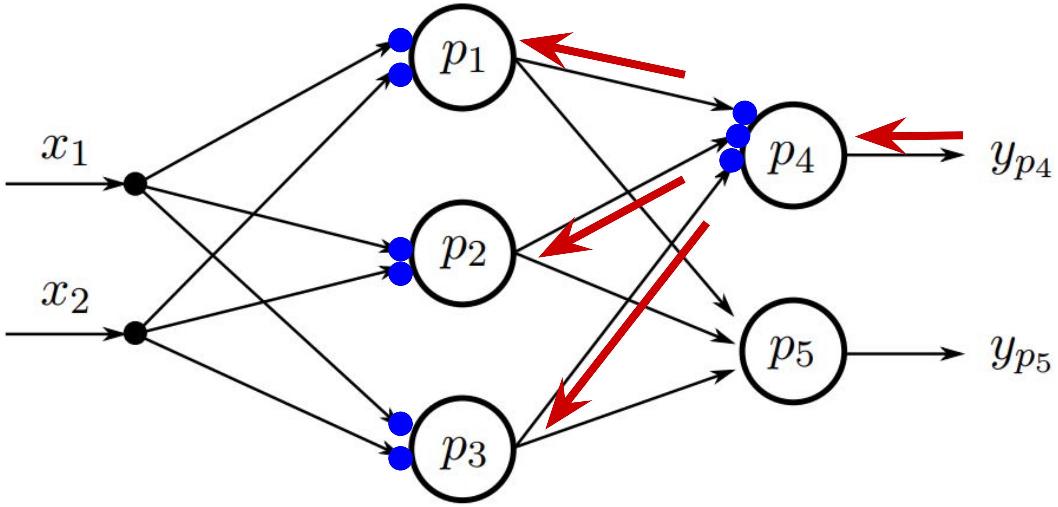






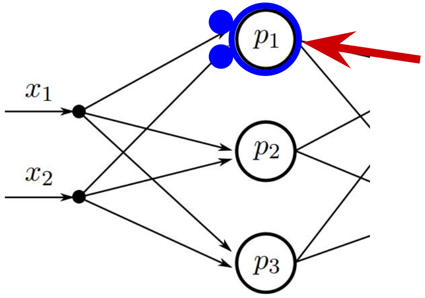


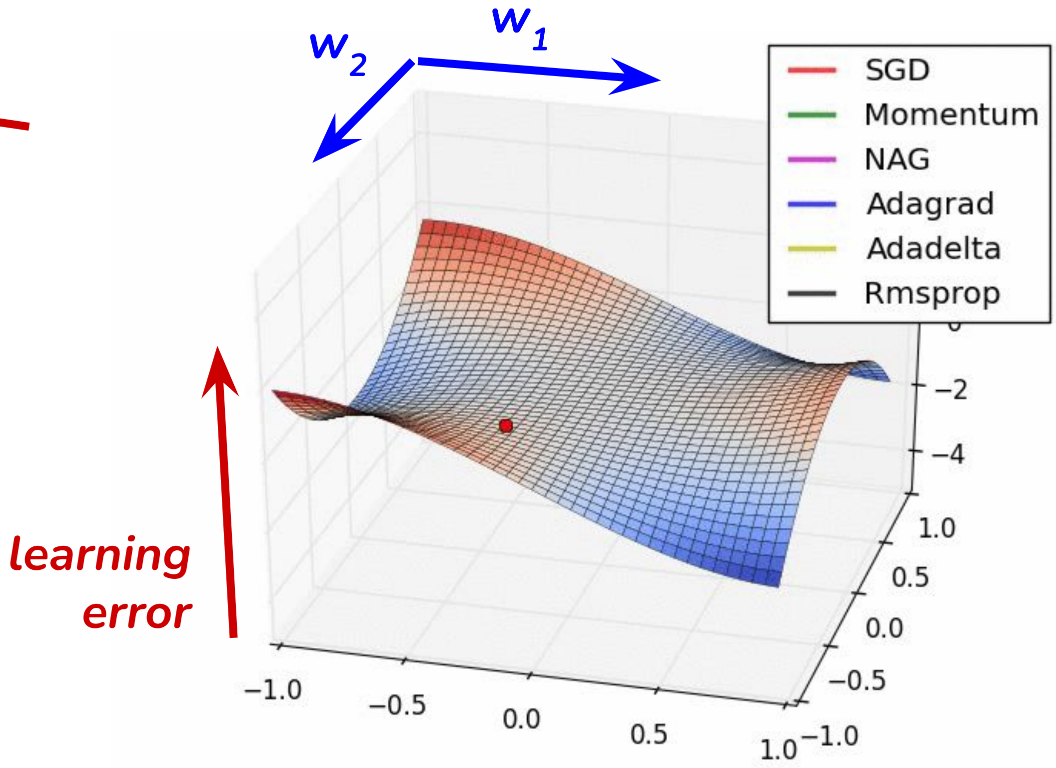
Neural networks – Backpropagation



Machine Learning College

Neural networks - Backpropagation

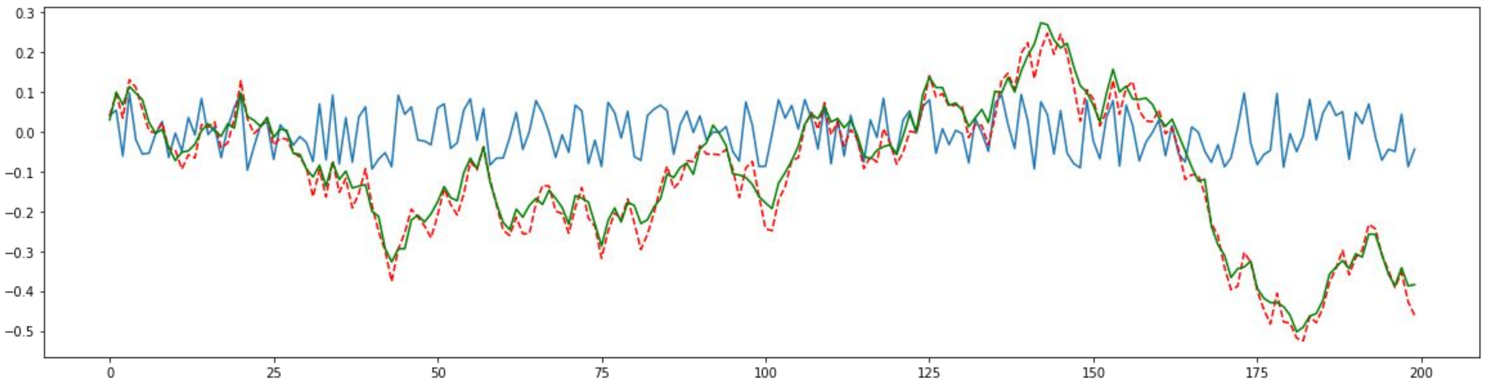




http://www.denizyuret.com/2015/03/alec-radfords-animations-for.html

Machine Learning College

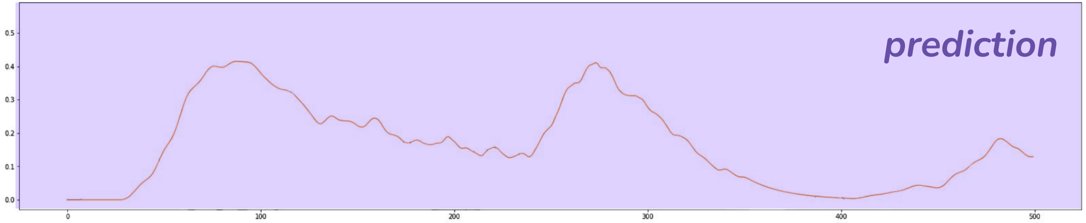
Time Series with Neural Networks

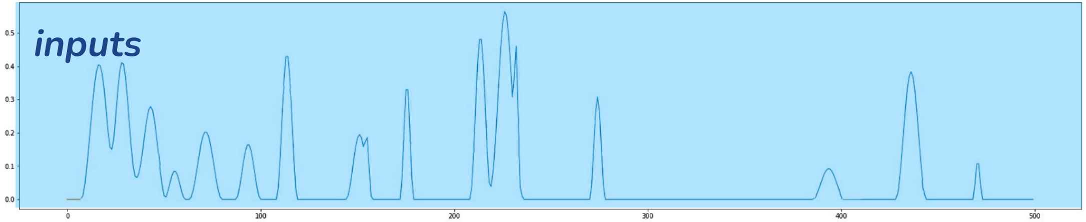


- Neural Networks
  - How to **express time domain**
  - How to **prepare training data**
  - How to **design the model**
  - How to **train & test the model**

Machine Learning College

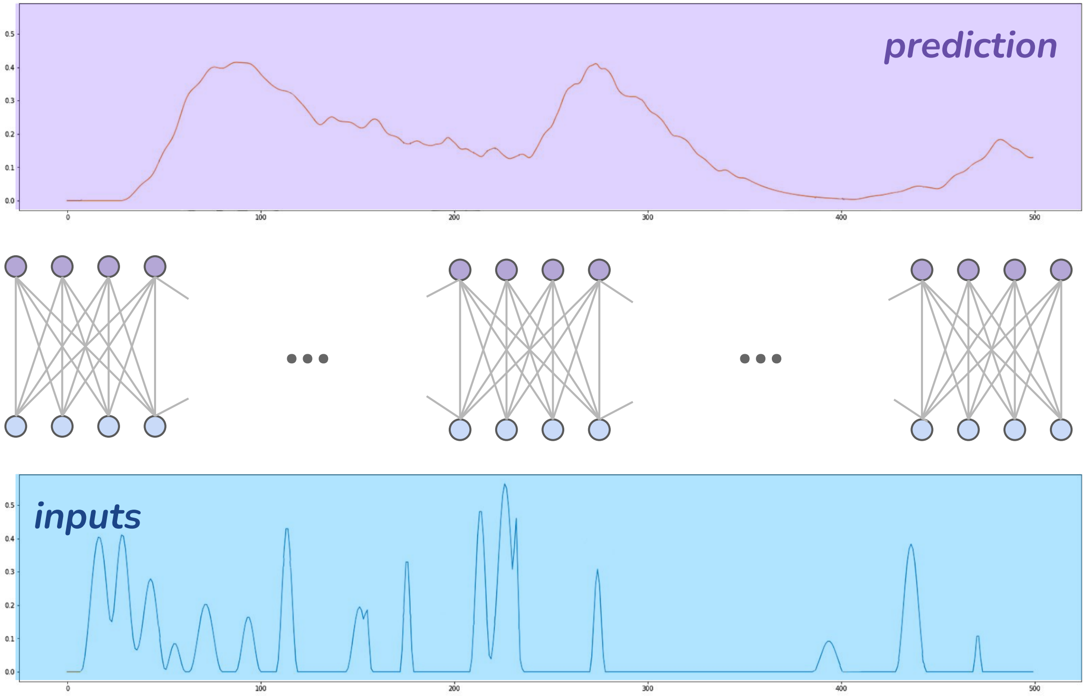
How neural network fits?





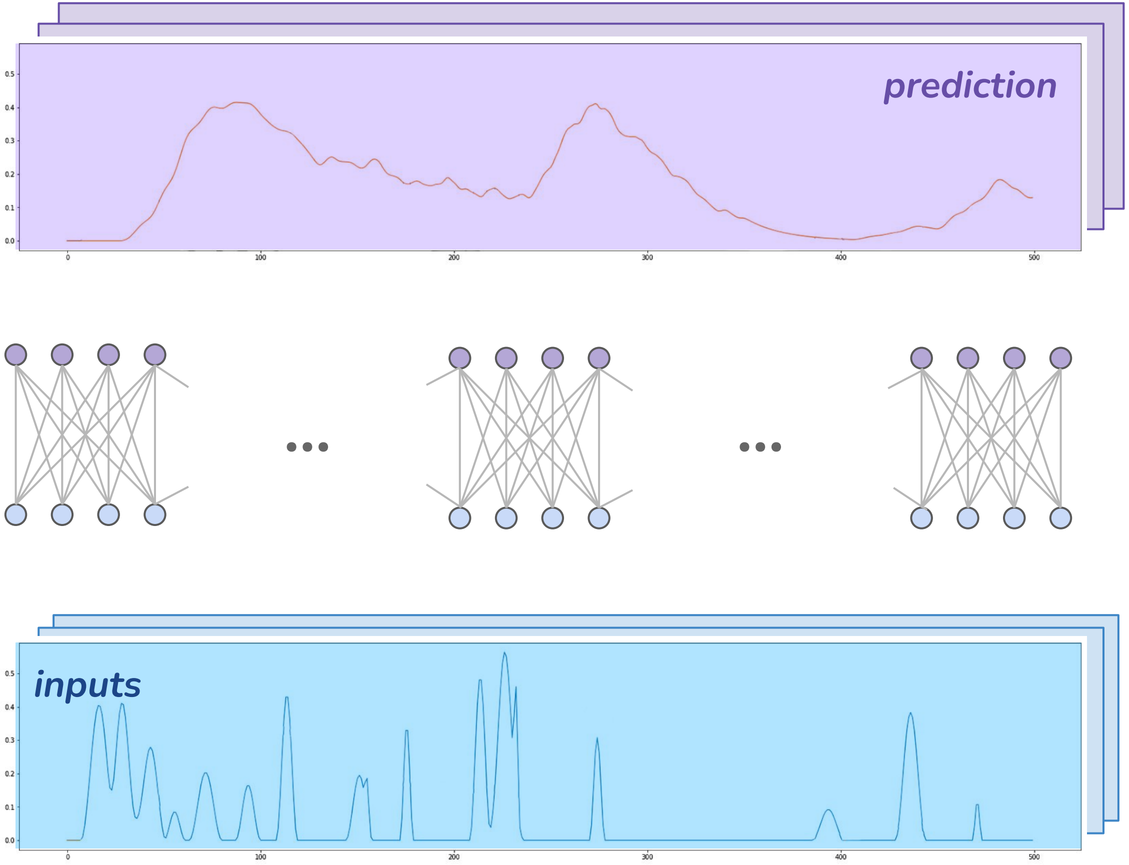
Machine Learning College

Simple feed-forward network



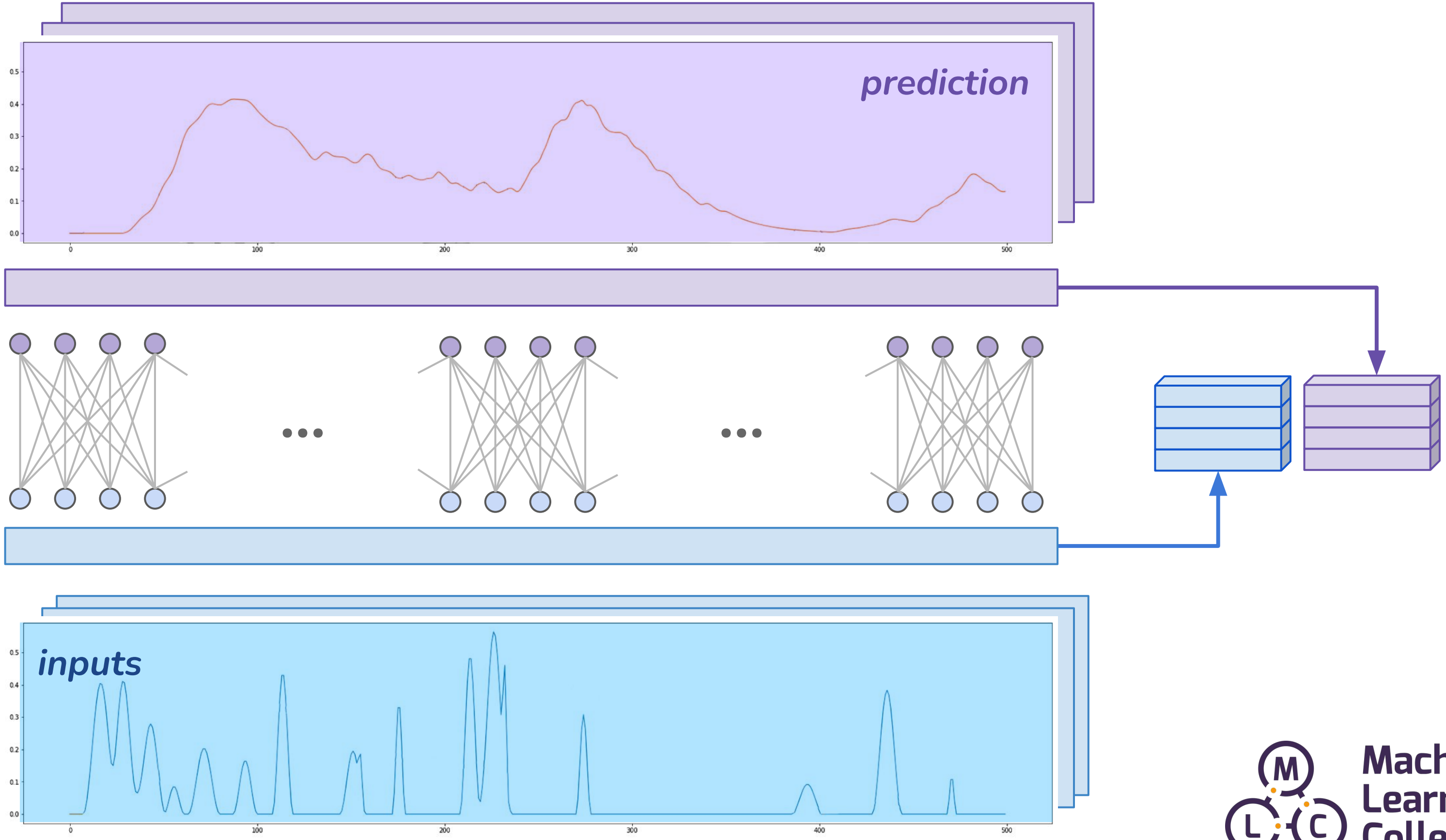
Machine Learning College

Simple feed-forward network



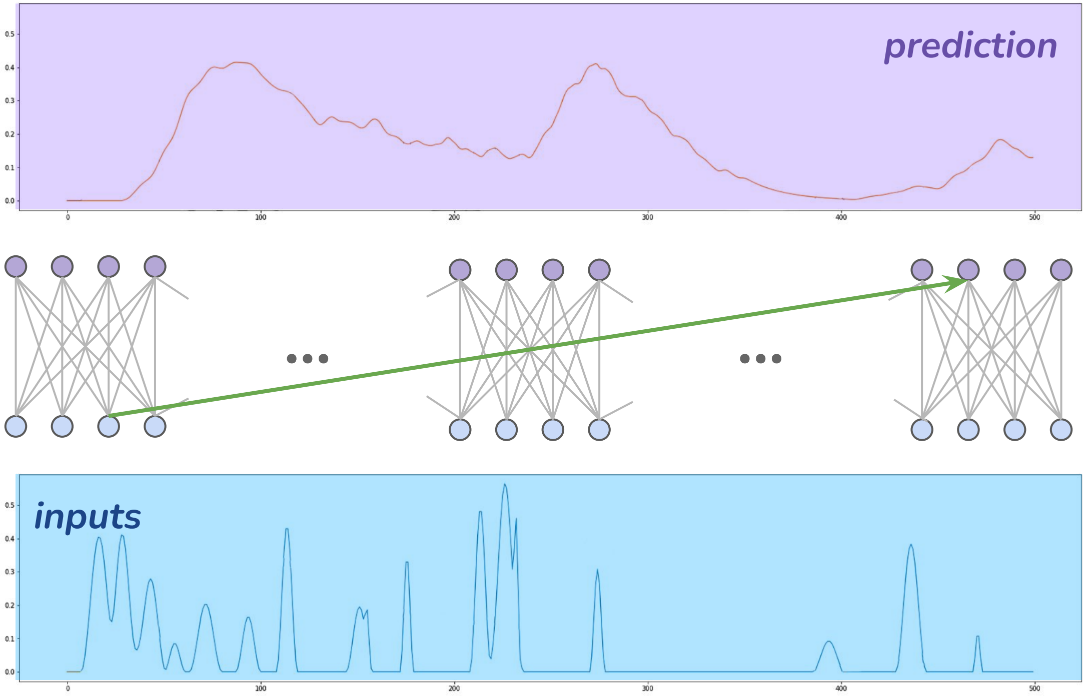
Machine Learning College

Simple feed-forward network



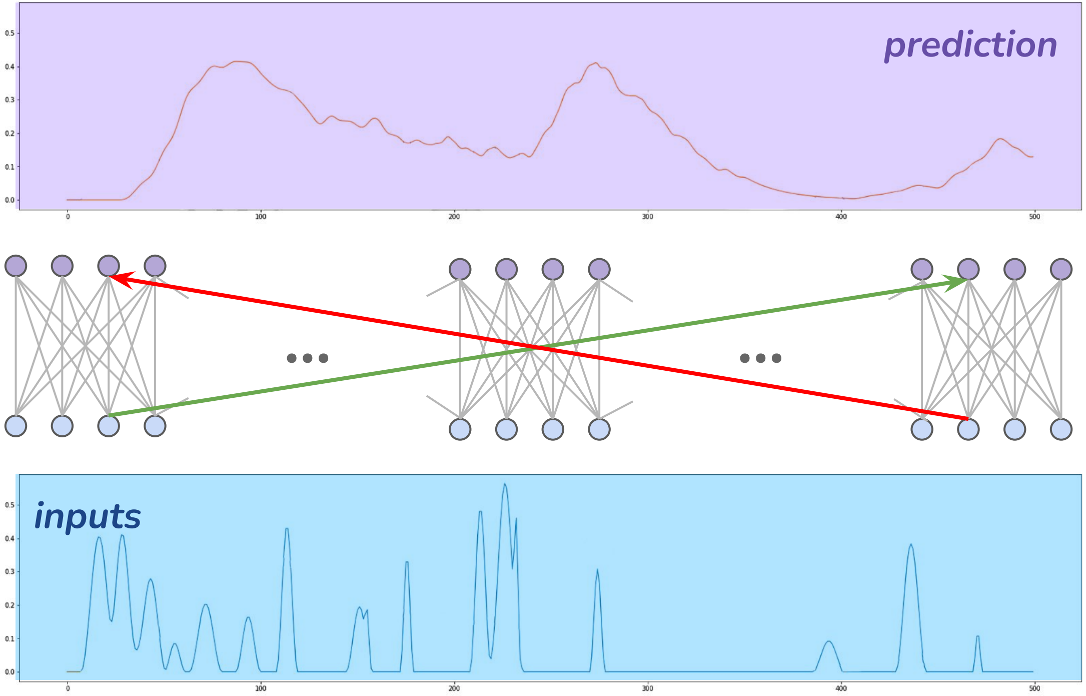
Machine Learning College

Simple feed-forward network



Machine Learning College

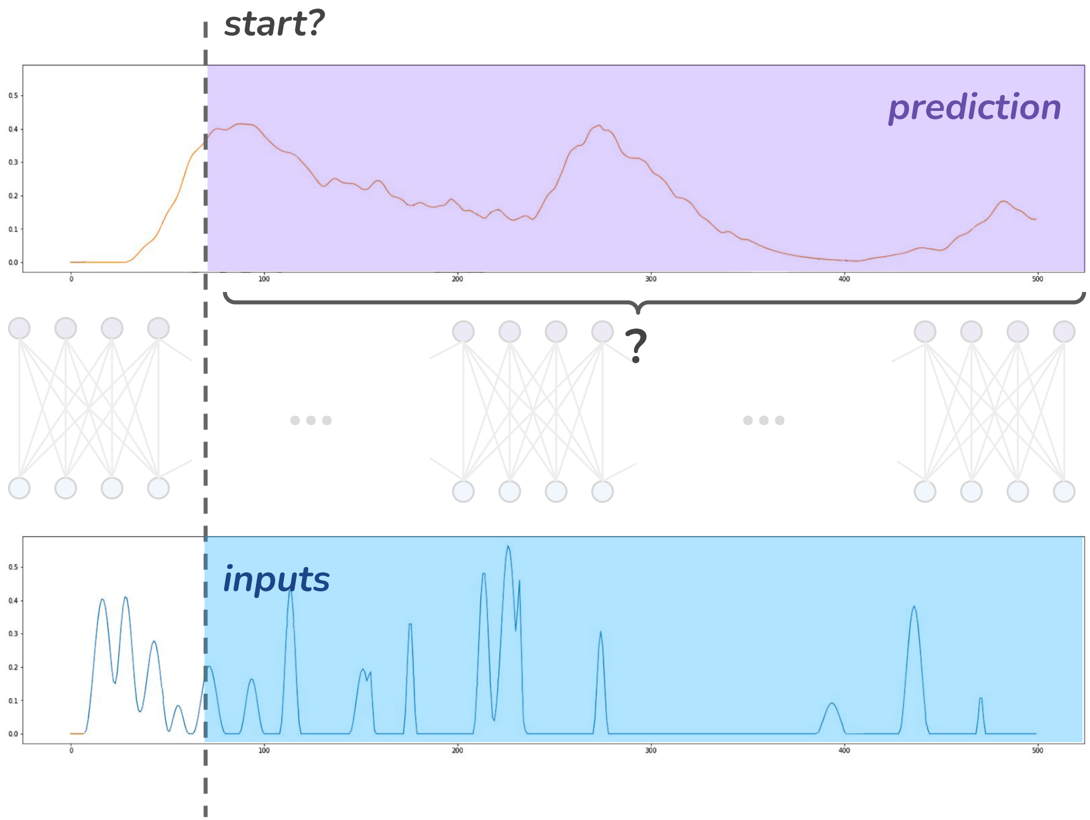
Simple feed-forward network



predicting from future ?

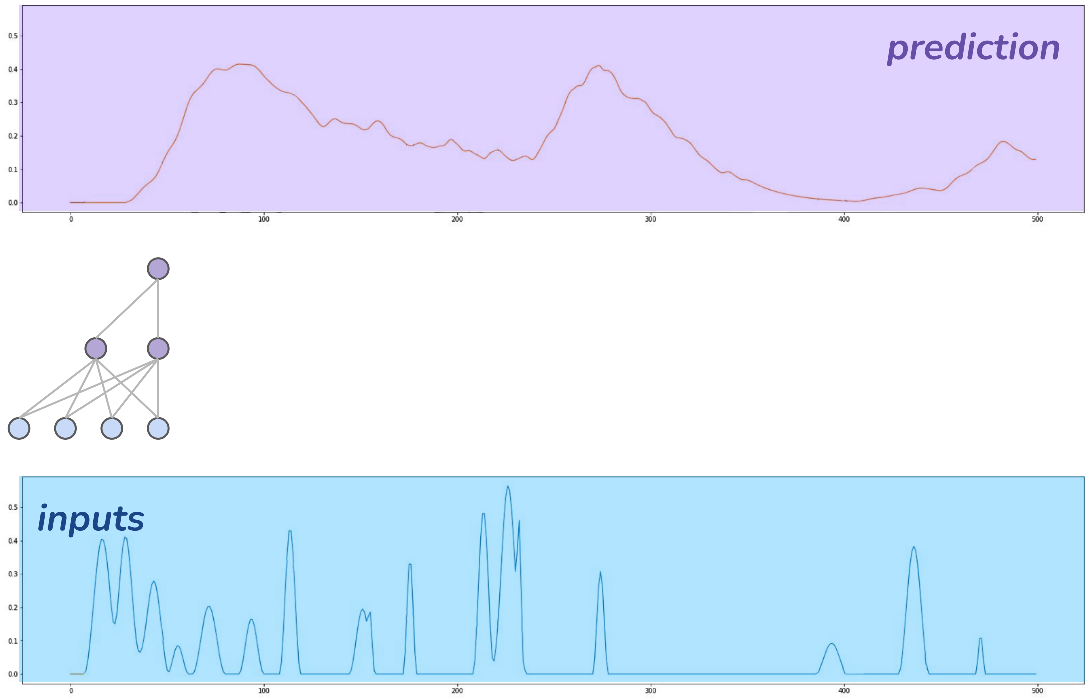
Machine Learning College

Simple feed-forward network



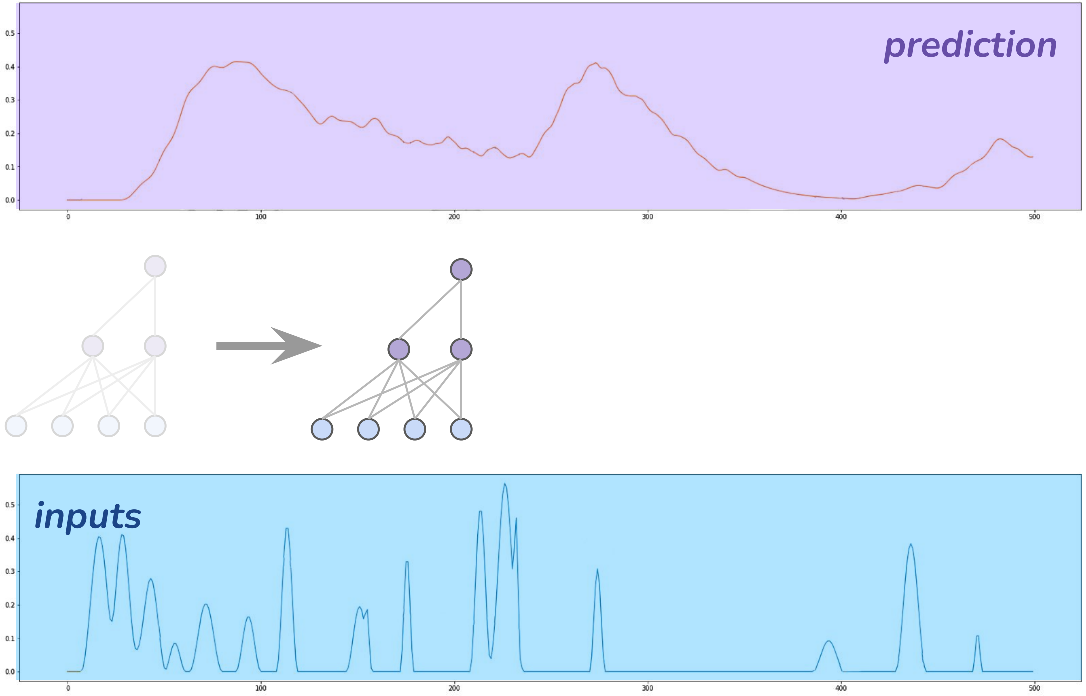
Machine Learning College

Sliding feed-forward network



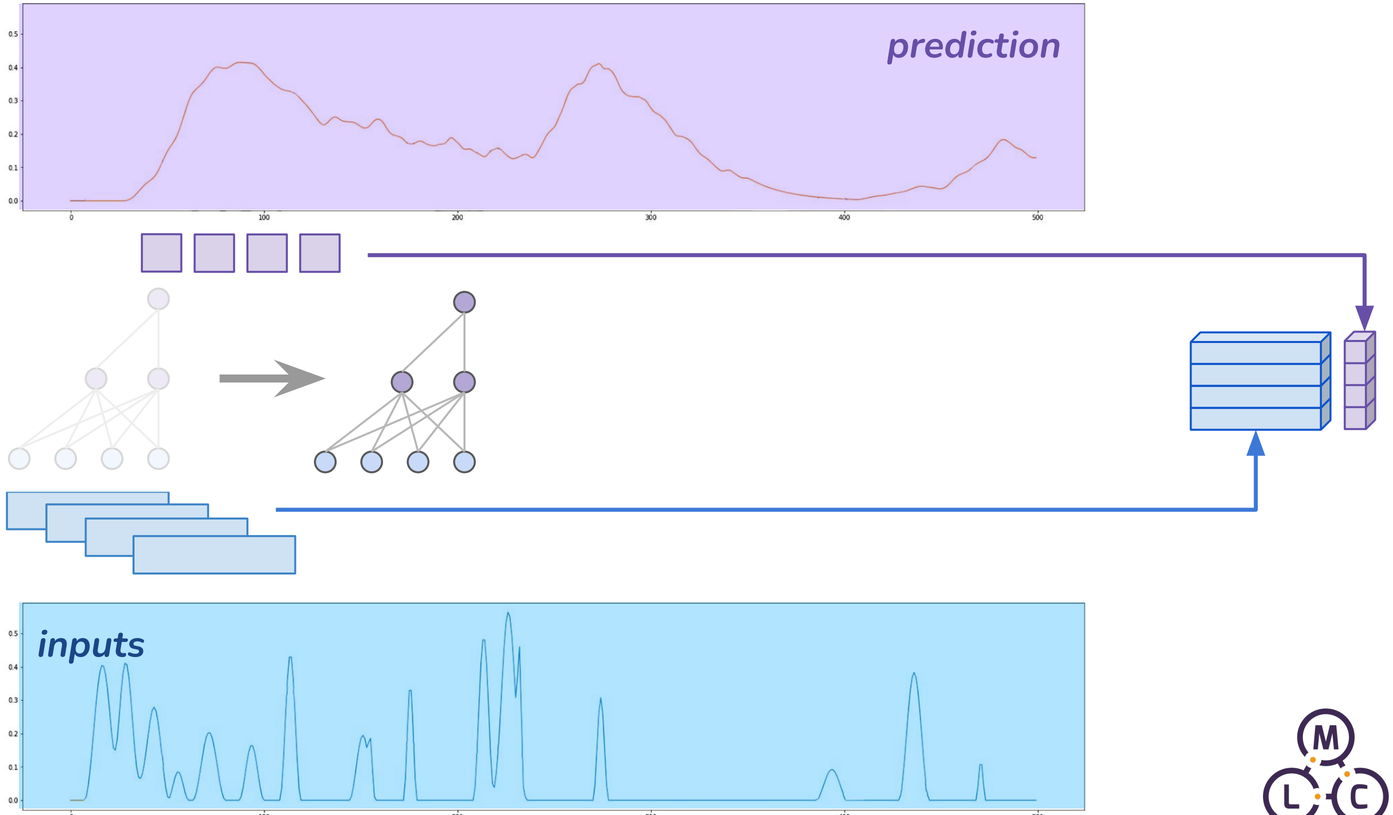
Machine Learning College

Sliding feed-forward network



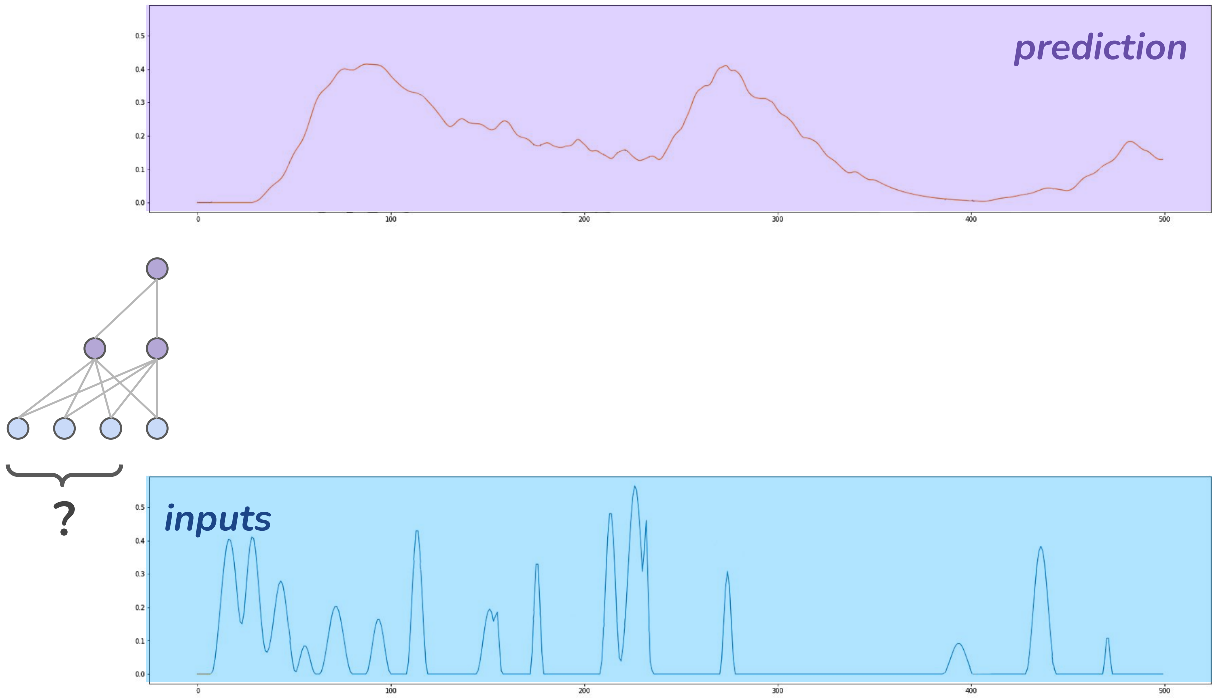
Machine Learning College

Sliding feed-forward network



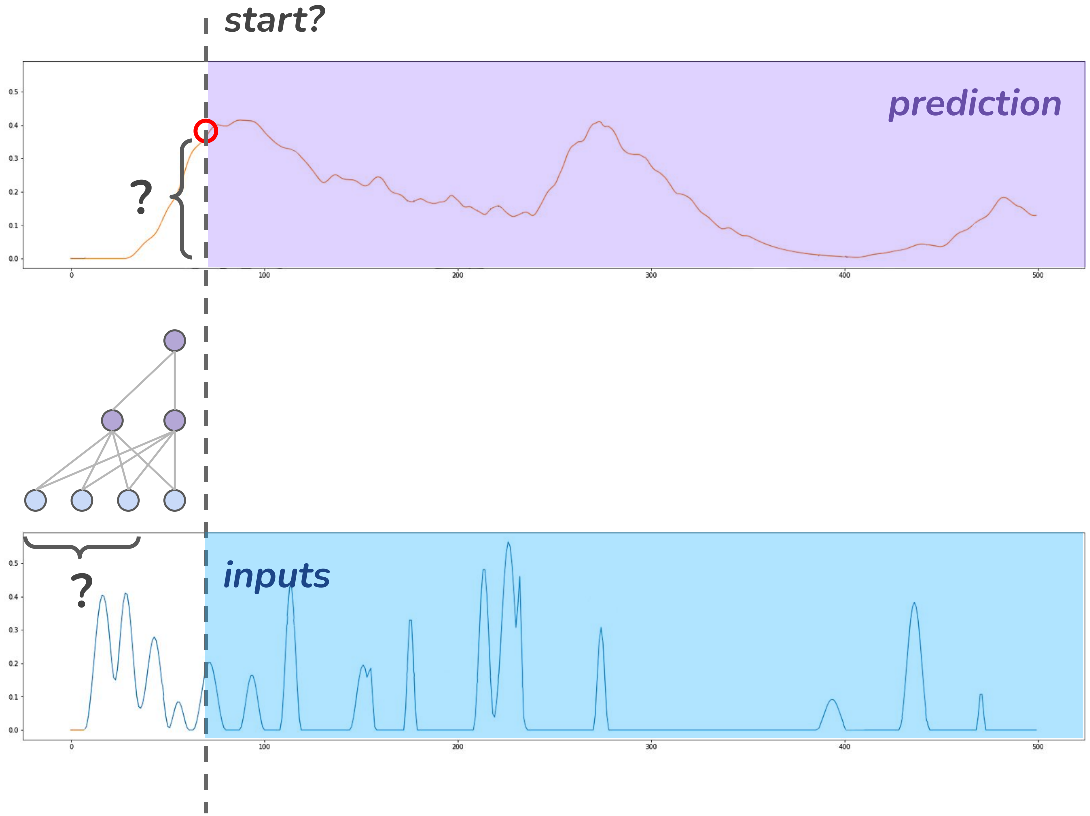
Machine Learning College

Sliding feed-forward network



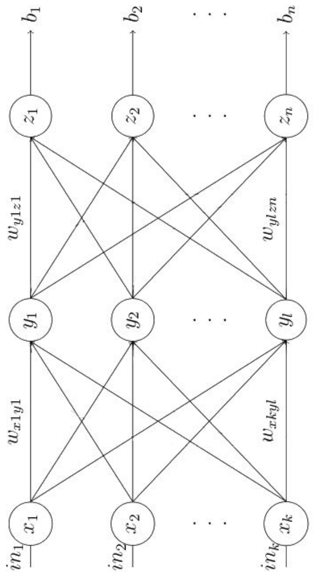
Sliding feed-forward network

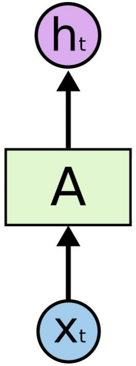
Sliding feed-forward network

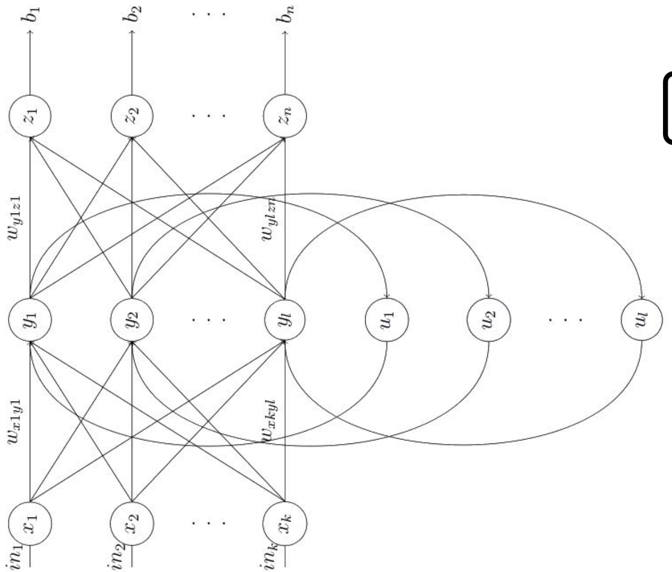


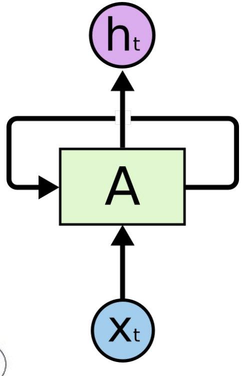
Machine Learning College

Recurrent Neural Networks





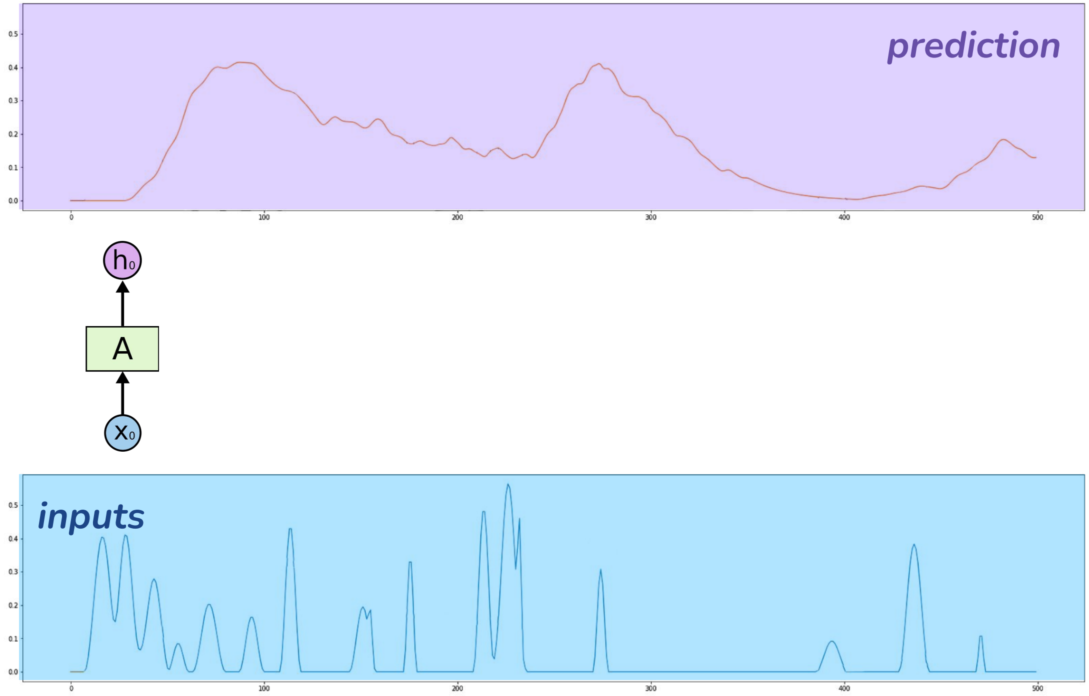




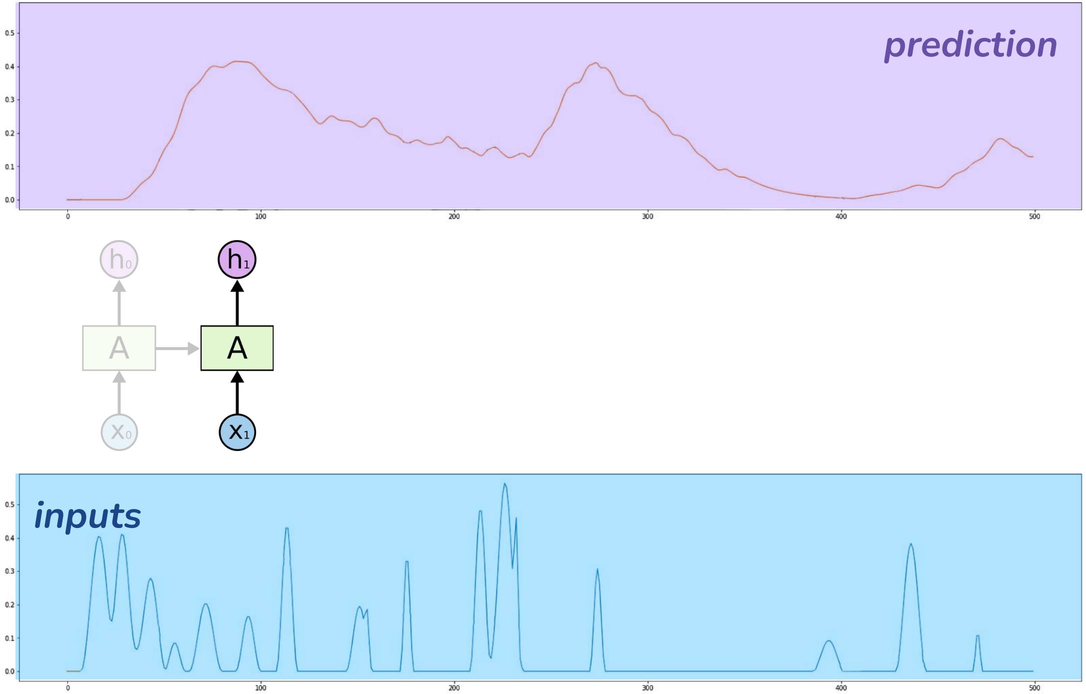
*http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Machine Learning College

RNN for time series prediction

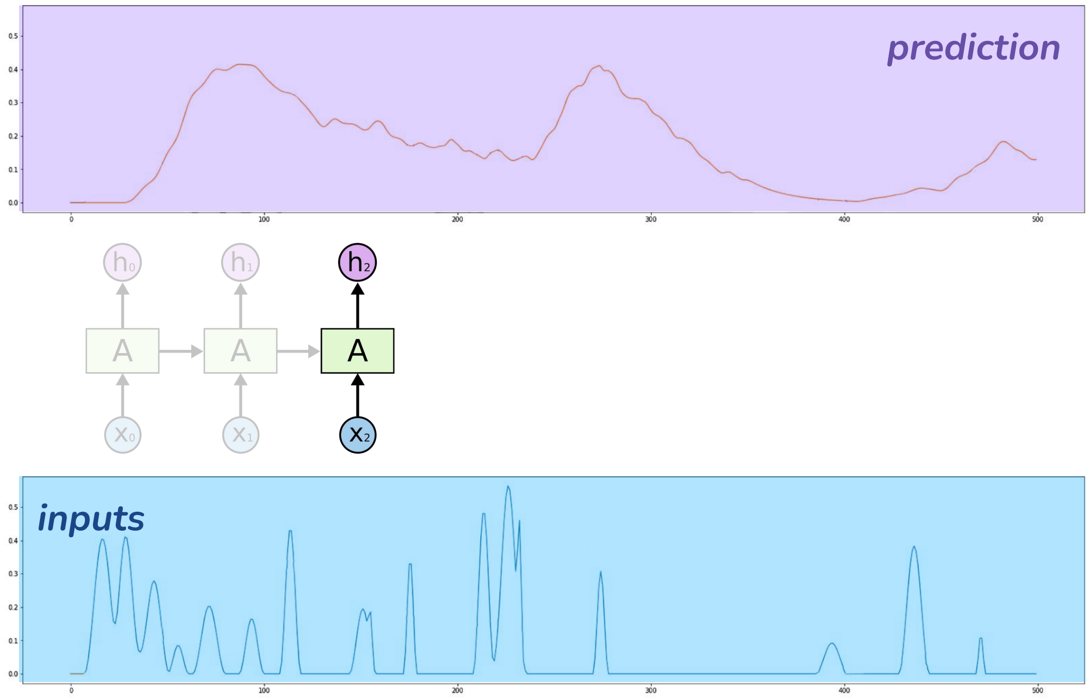


RNN for time series prediction



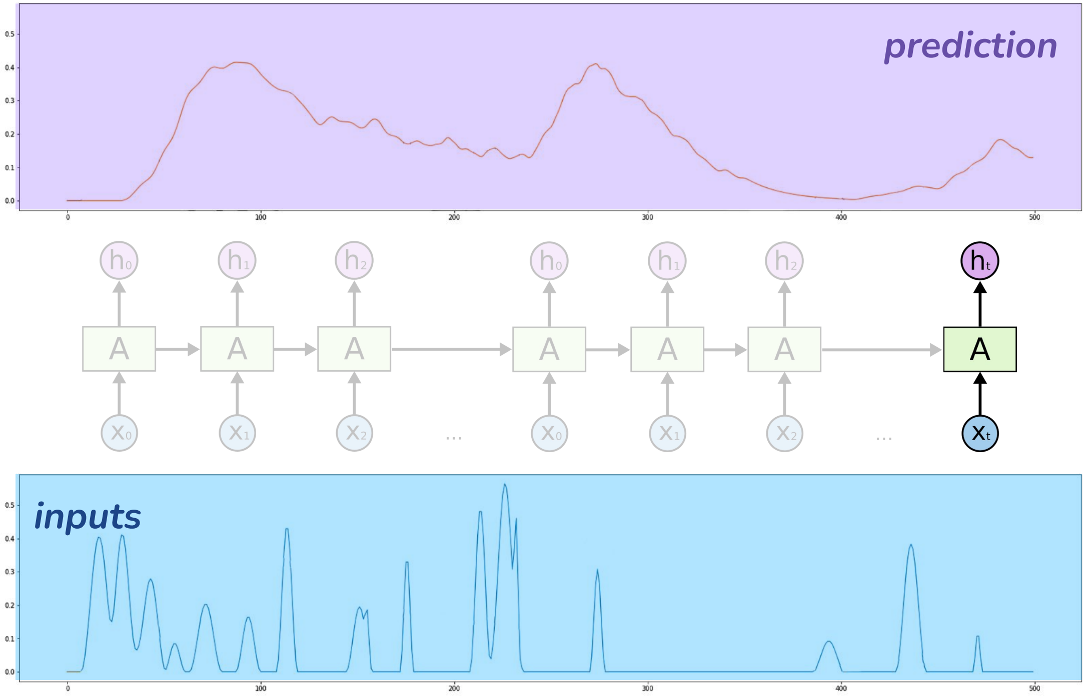
Machine Learning College

RNN for time series prediction

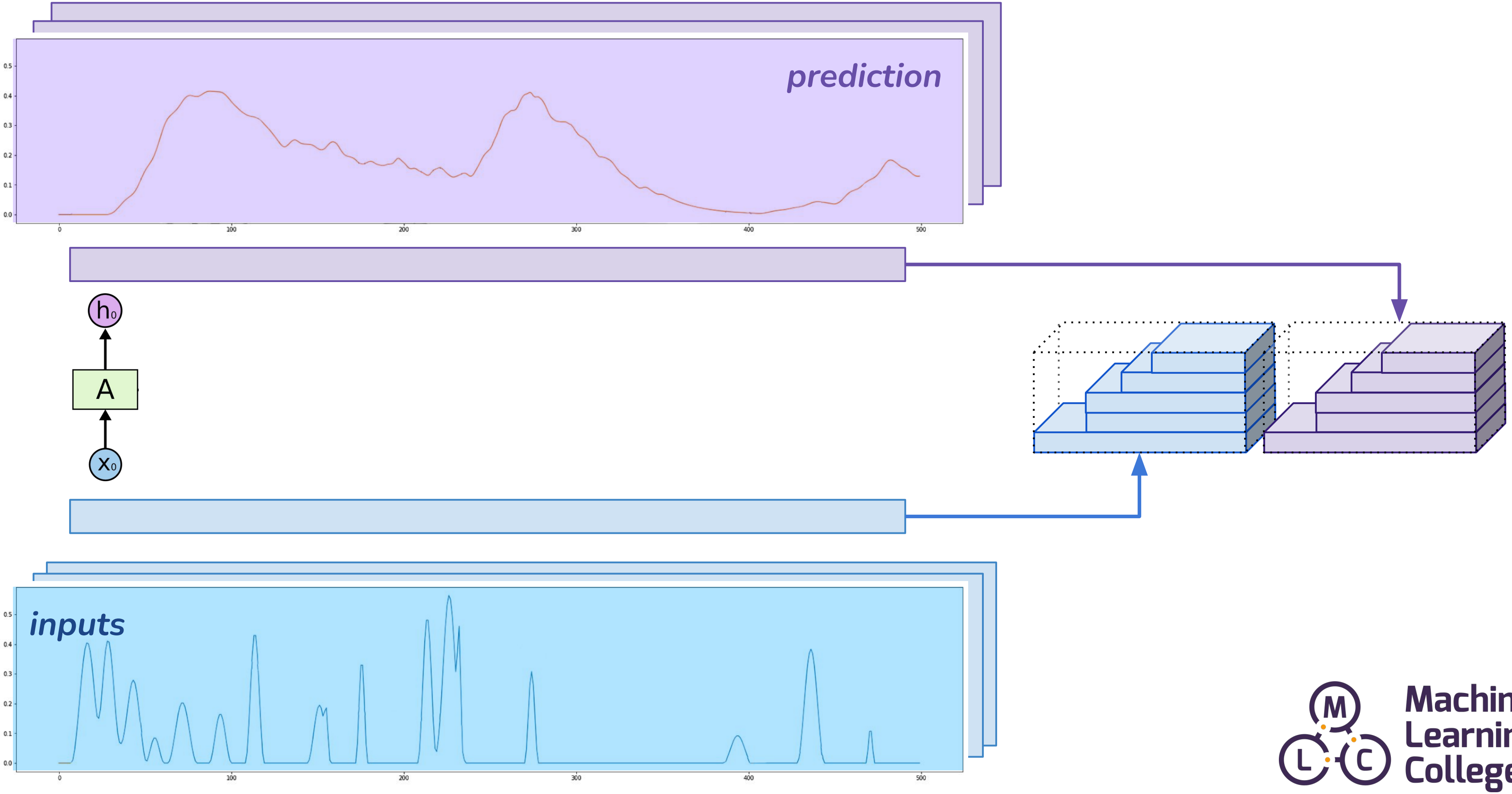


Machine Learning College

RNN for time series prediction

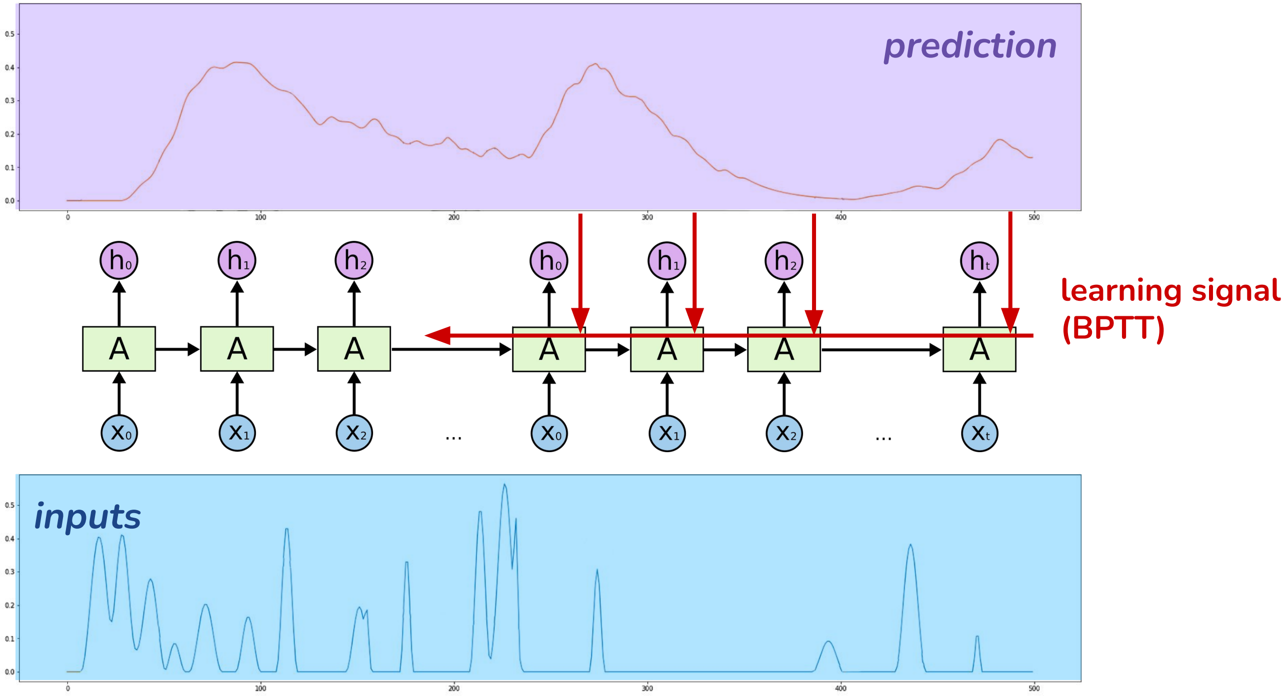


RNN for time series prediction



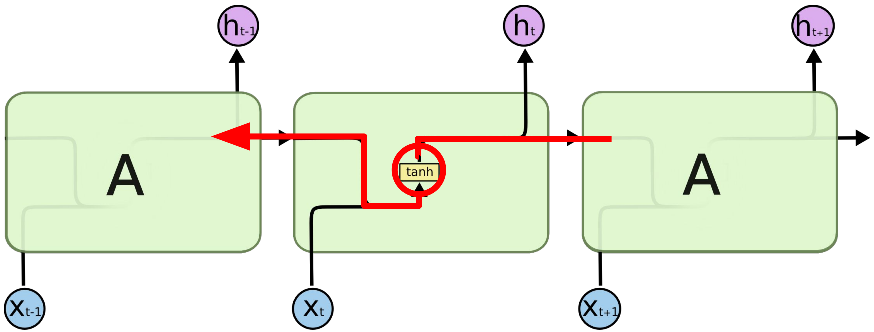
Machine Learning College

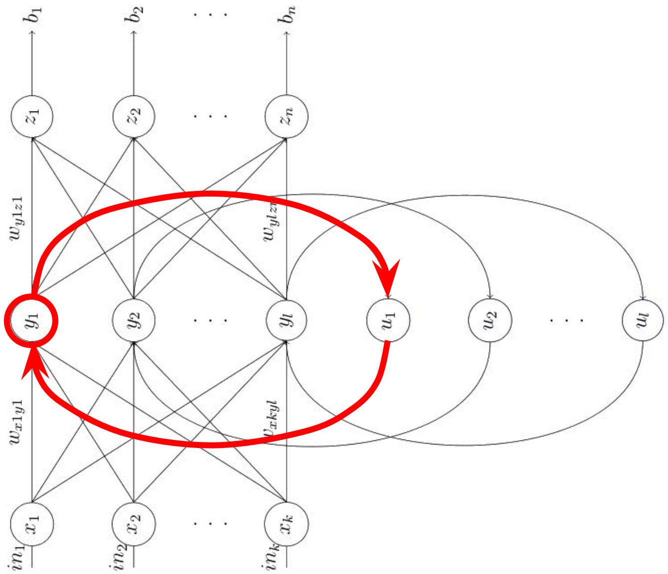
RNN for time series prediction



Machine Learning College

RNN – Vanishing gradients

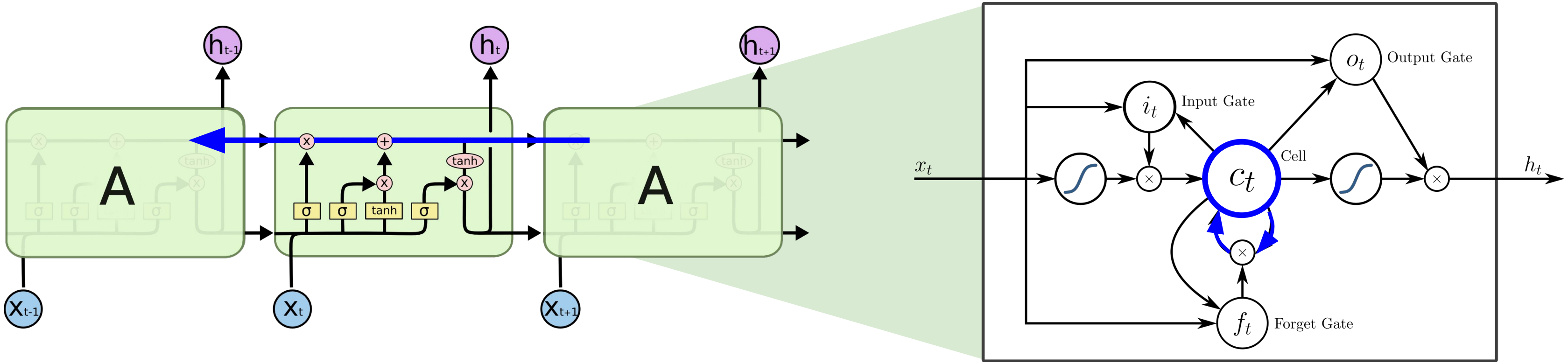




*http://colah.github.io/posts/2015-08-Understanding-LSTMs/

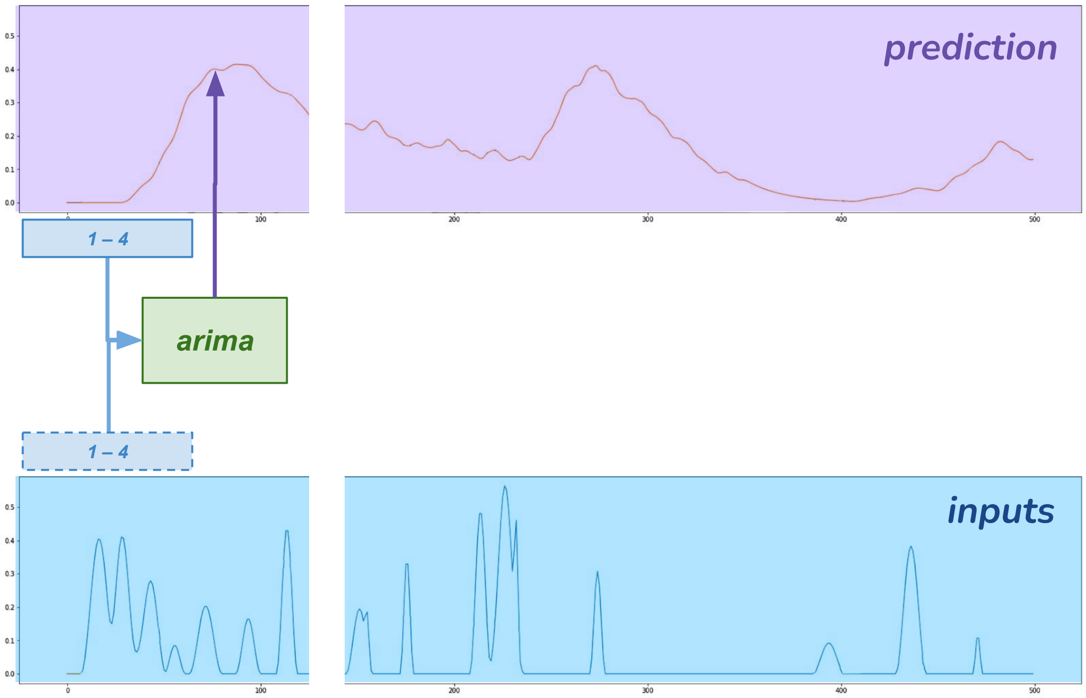
Machine Learning College

Long short-term memory – LSTM



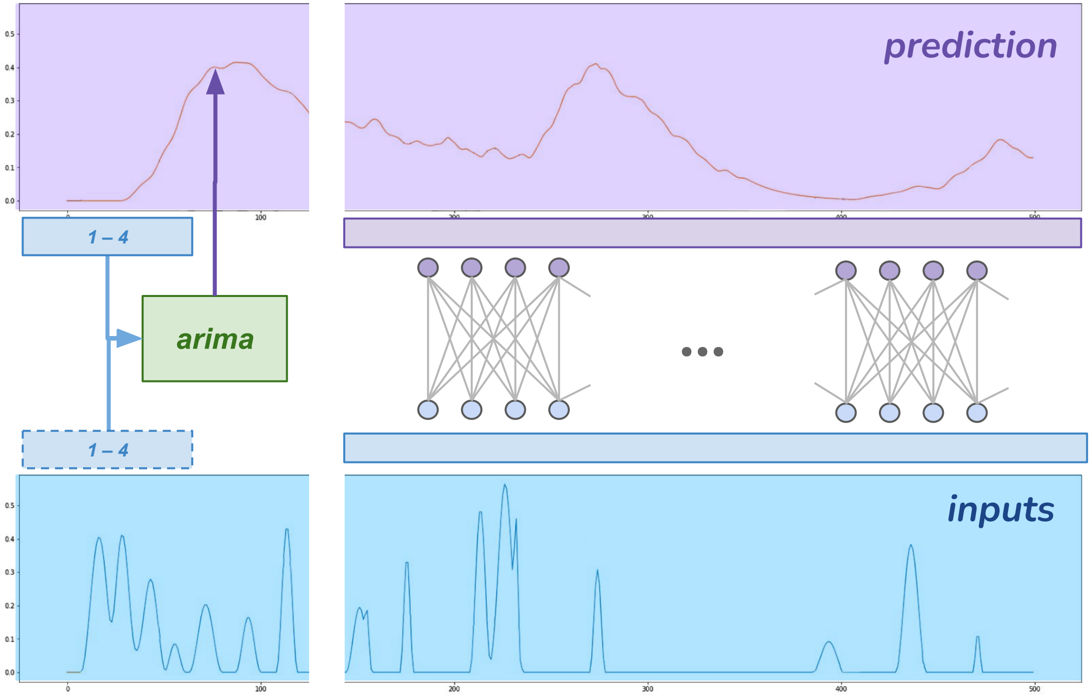
*http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Classical model vs. neural network



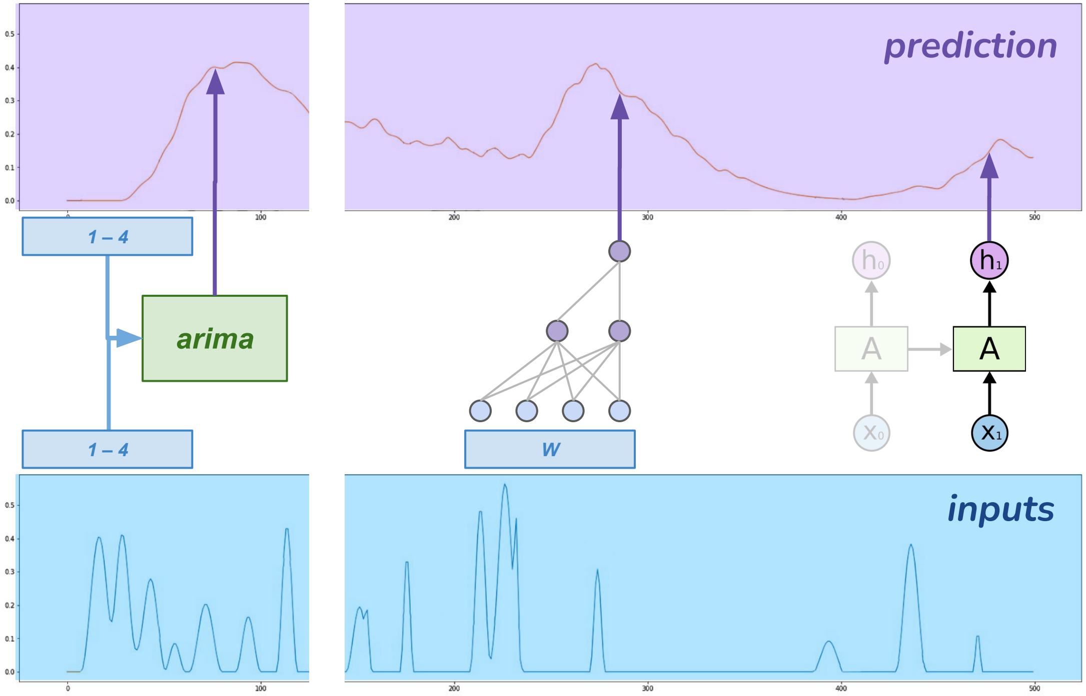
Machine Learning College

Classical model vs. neutal network



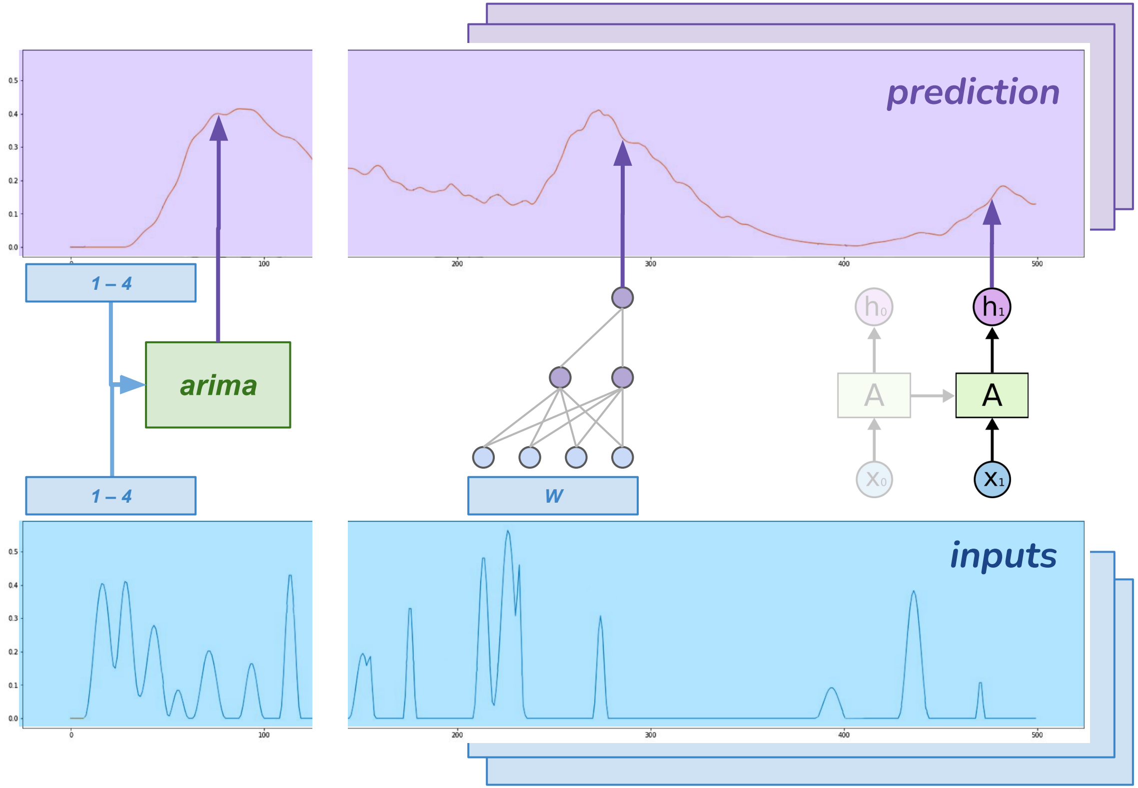
Machine Learning College

Classical model vs. neutal network



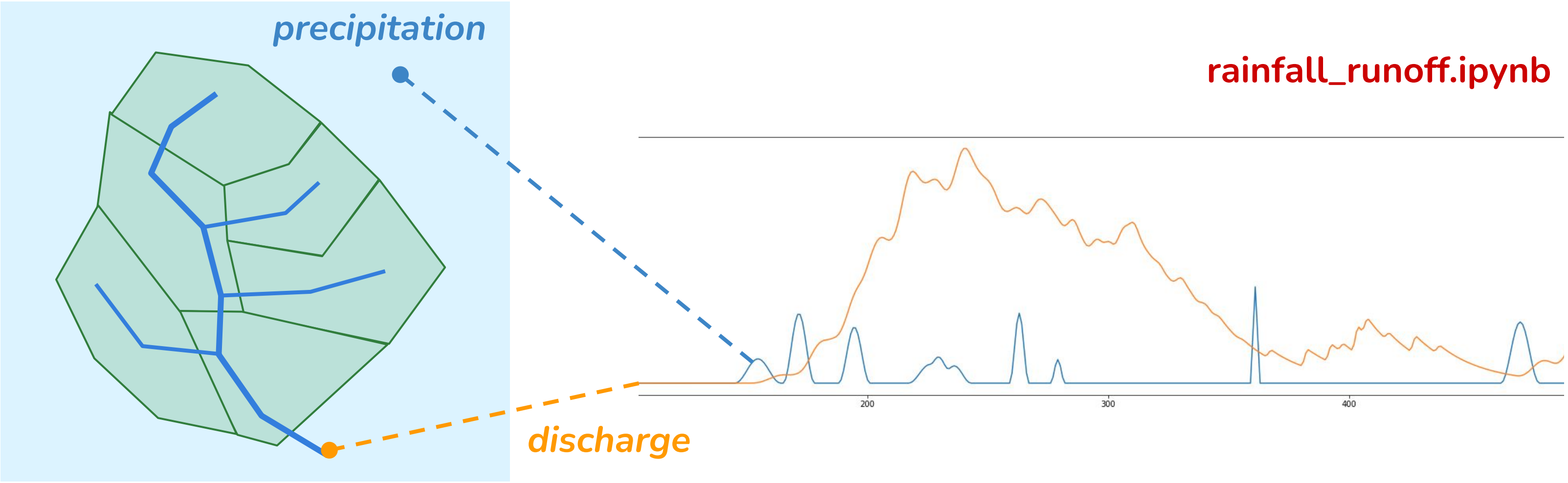
Machine Learning College

Classical model vs. neutal network



Machine Learning College

Rainfall-runoff example

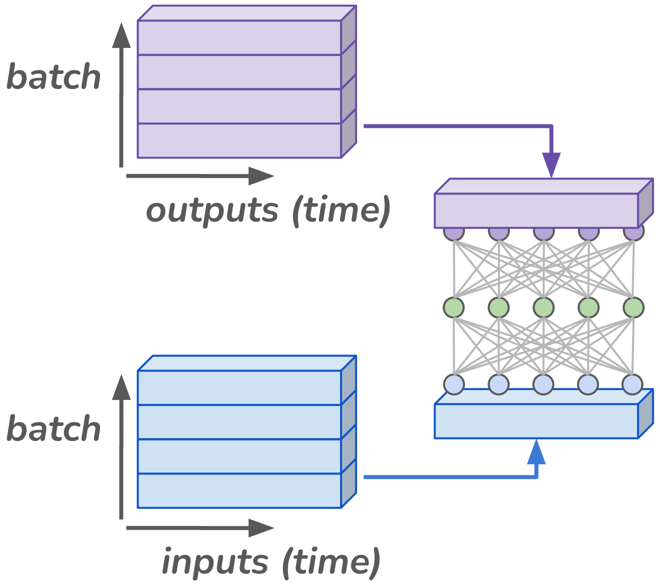


- Test out neural networks on simple **generated rainfall-runoff dataset**
  - Simulated **long-time dependencies** in data
  - Test various neural network architectures
    - Flat feed-forward network
    - LSTM

Machine Learning College

Tensors & dimentions – univariate regression

Flat NN = 2D training data

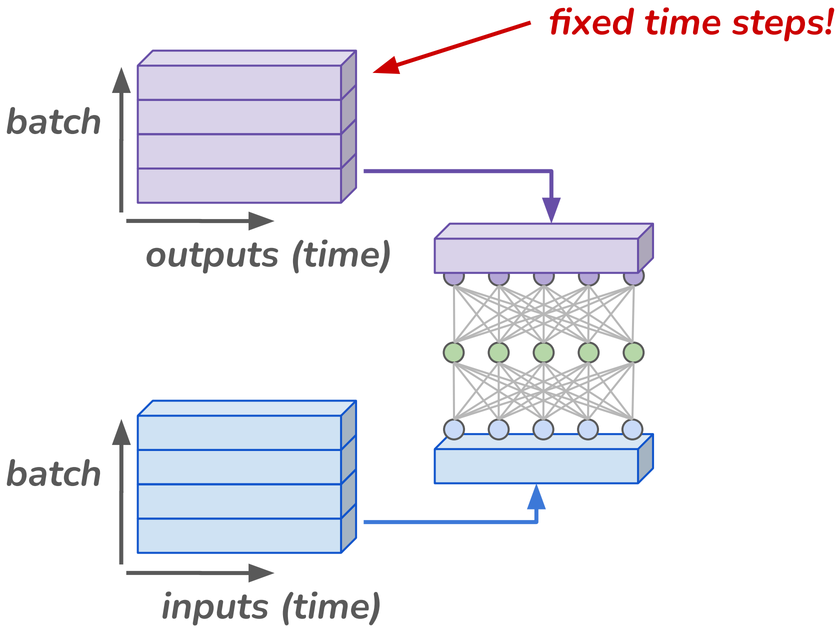


$(B, T) \longrightarrow (B, T)$

Machine Learning College

Tensors & dimentions – univariate regression

Flat NN = 2D training data

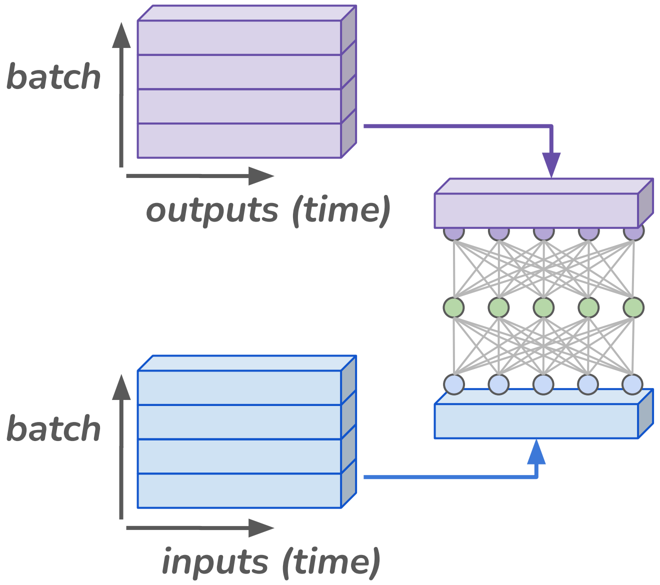


(B, T)⟶(B, T)

Machine Learning College

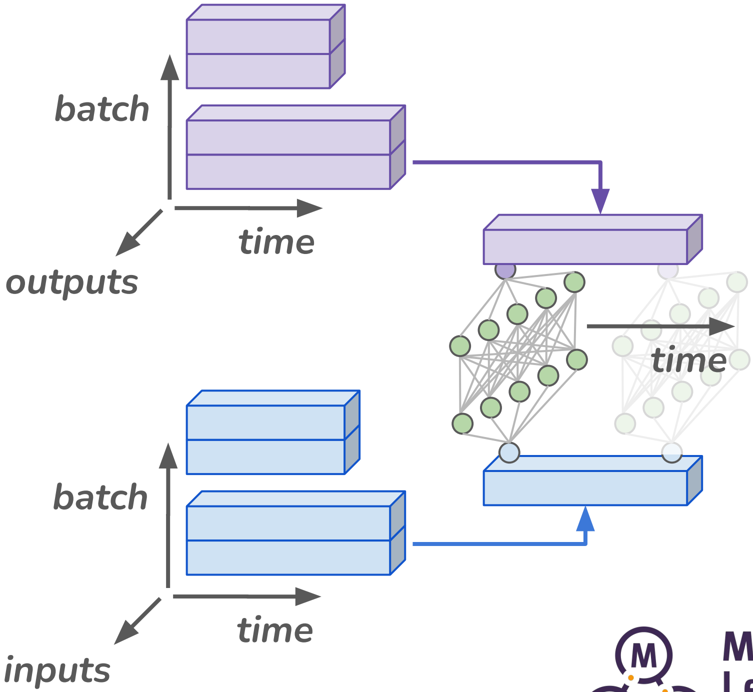
Tensors & dimentions – univariate regression

Flat NN = 2D training data



(B, T)⟶(B, T)

Recurrent NN = 3D training data

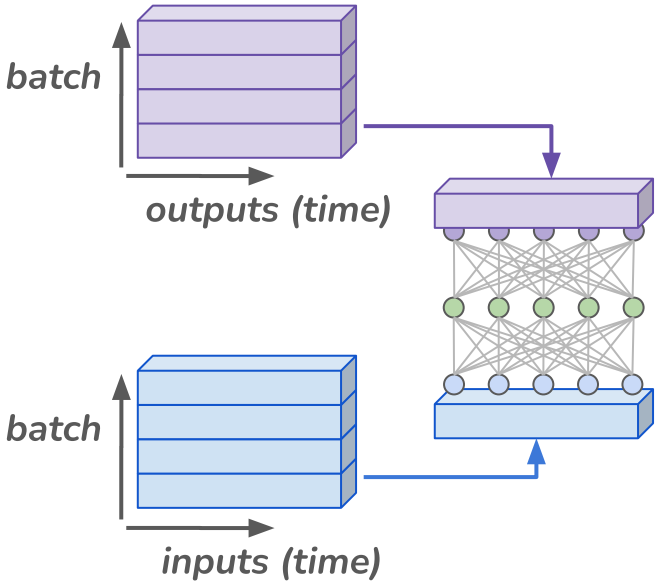


(B, T, 1)⟶(B, T, 1)

Machine Learning College

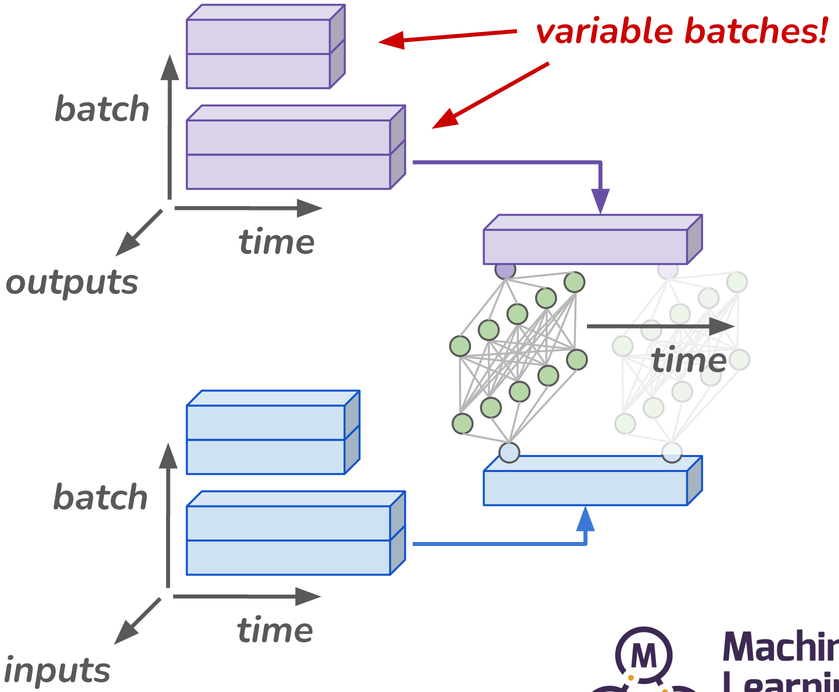
Tensors & dimentions – univariate regression

Flat NN = 2D training data



(B, T)⟶(B, T)

Recurrent NN = 3D training data

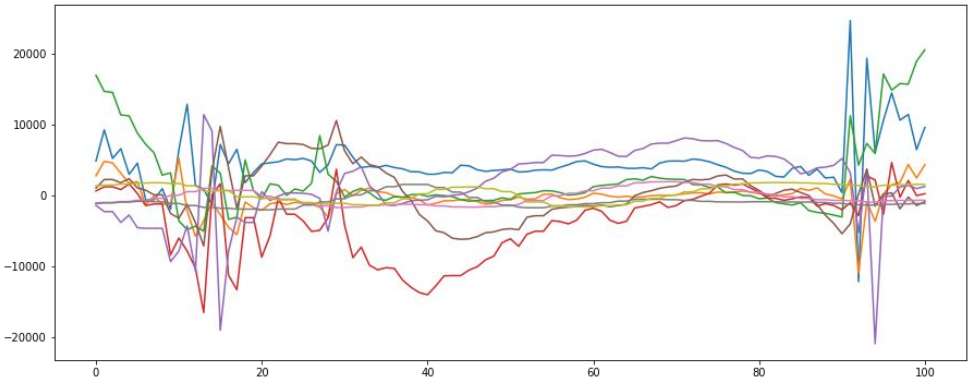


(B, T, 1)⟶(B, T, 1)

Machine Learning College

Trampoline jumping example

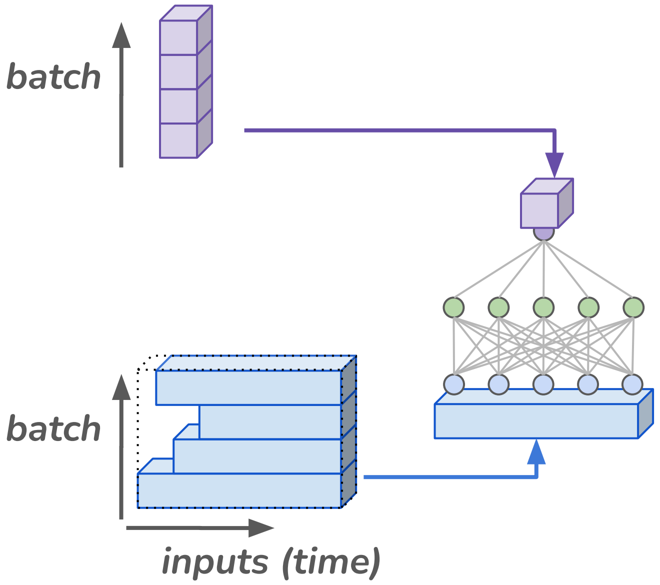
- Data preparation
  - Dataset normalization
  - Sequence padding
- Binary classification task
  - Target values & dimensions
  - Loss functions
- Training & evaluation
  - Inference visualization
  - Evaluation metrics



Machine Learning College

Tensors & dimentions – binary classification

Flat NN = 2D training data

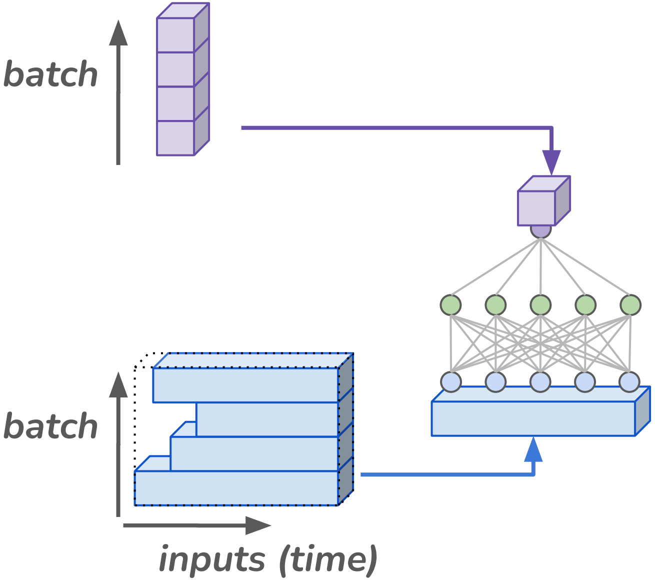


(B, T)⟶(B, 1)

Machine Learning College

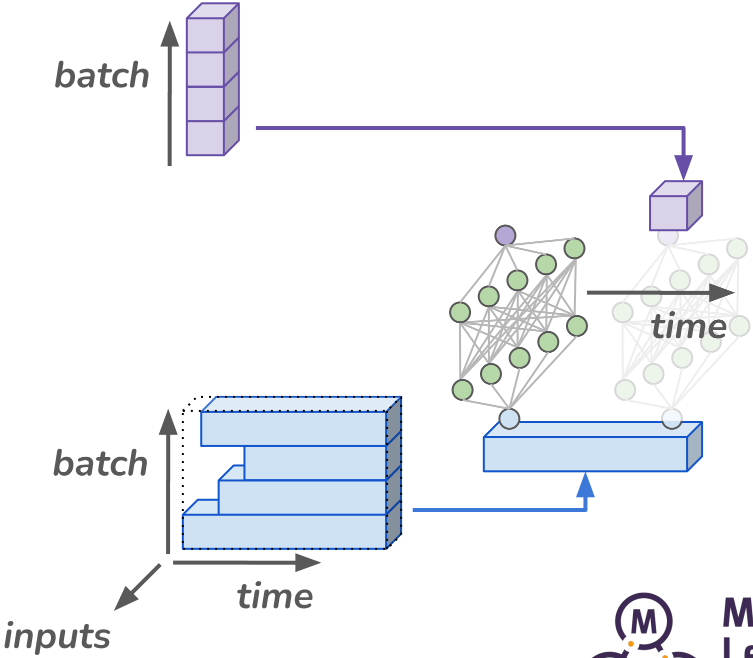
Tensors & dimentions – binary classification

Flat NN = 2D training data



$(B, T) \longrightarrow (B, 1)$

Recurrent NN = 3D training data

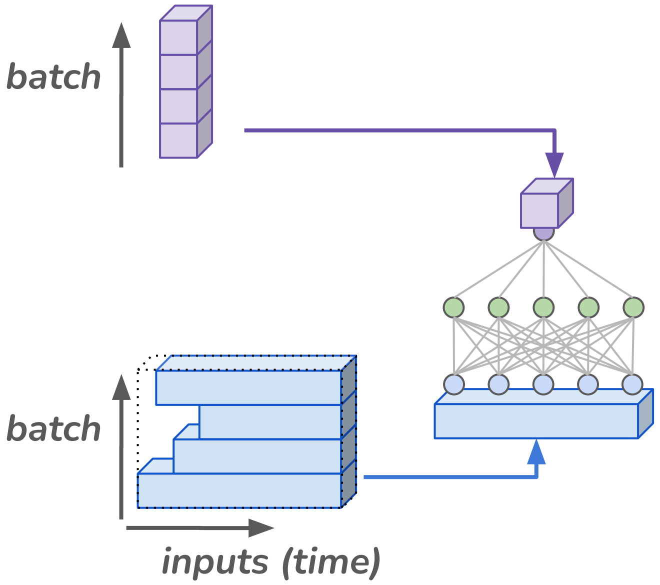


$(B, T, 1) \longrightarrow (B, 1)$

Machine Learning College

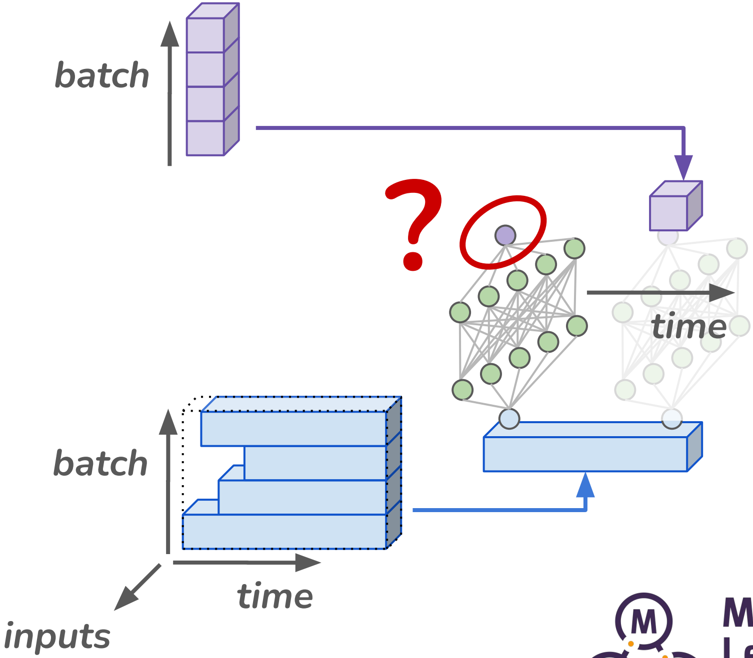
Tensors & dimentions – binary classification

Flat NN = 2D training data



(B, T)⟶(B, 1)

Recurrent NN = 3D training data

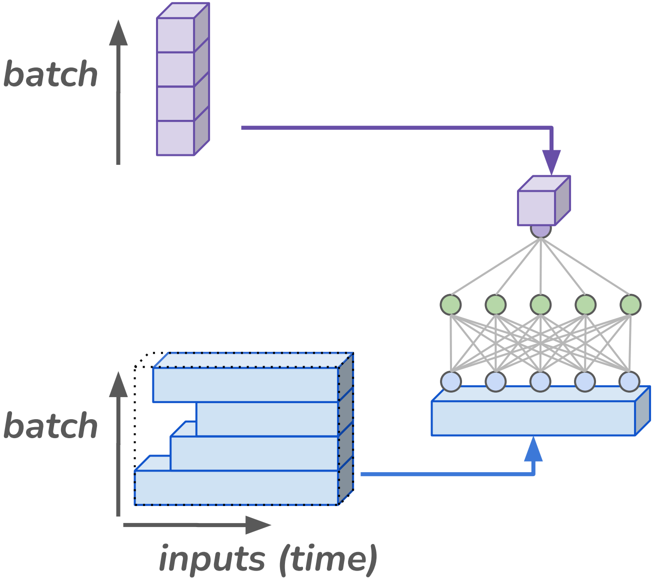


(B, T, 1)⟶(B, 1)

Machine Learning College

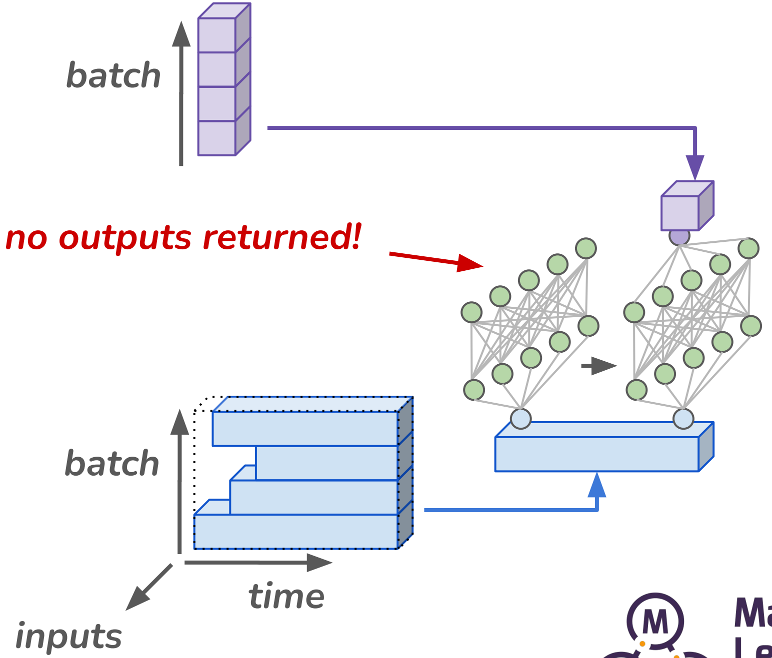
Tensors & dimentions – binary classification

Flat NN = 2D training data



(B, T)⟶(B, 1)

Recurrent NN = 3D training data



(B, T, 1)⟶(B, 1)

Machine Learning College

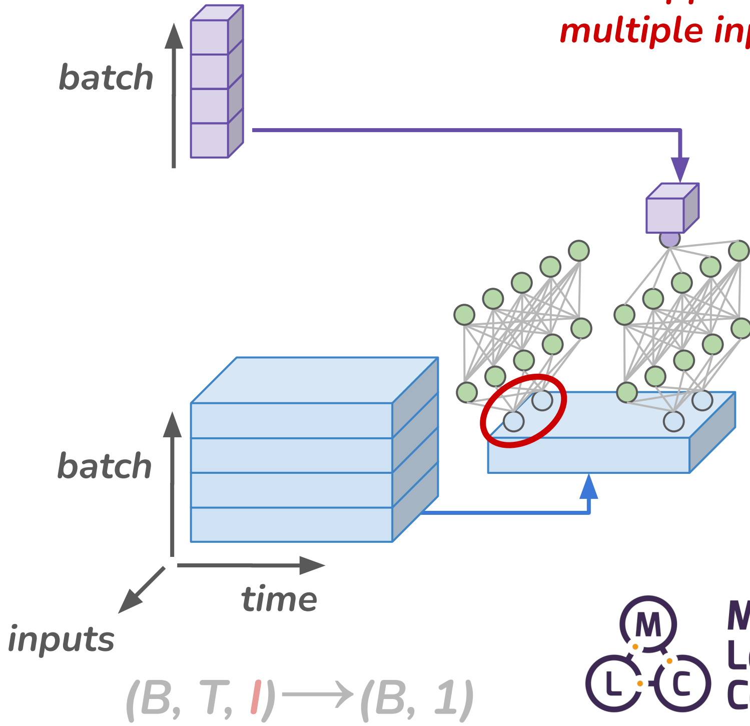
Tensors & dimentions – multivariate b. classification

Flat NN = 2D training data

?

Recurrent NN = 3D training data

natural support for multiple inputs

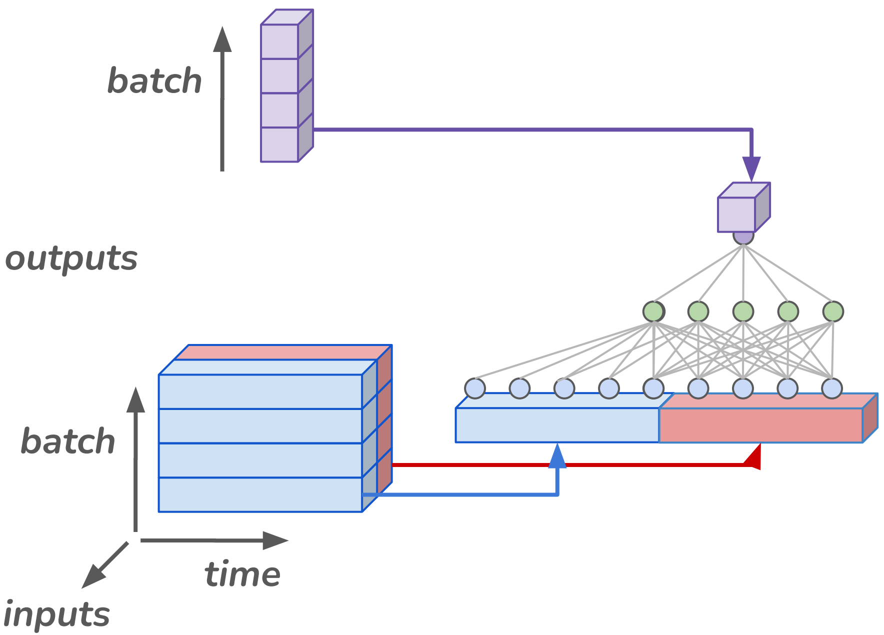


Machine Learning College

Tensors & dimentions – multivariate b. classification

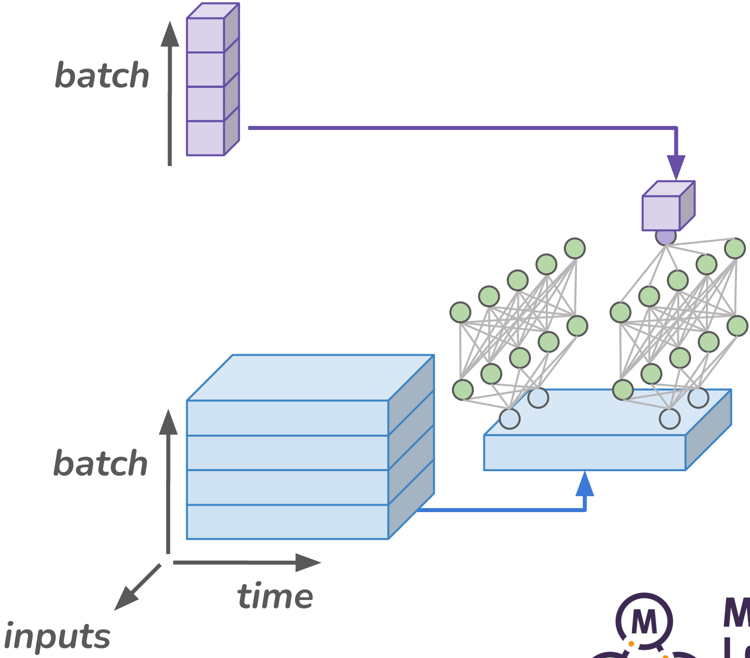
Flat NN = 3D⟶2D training data

flattening / interleaving



(B, T×I)⟶(B, T×O)

Recurrent NN = 3D training data

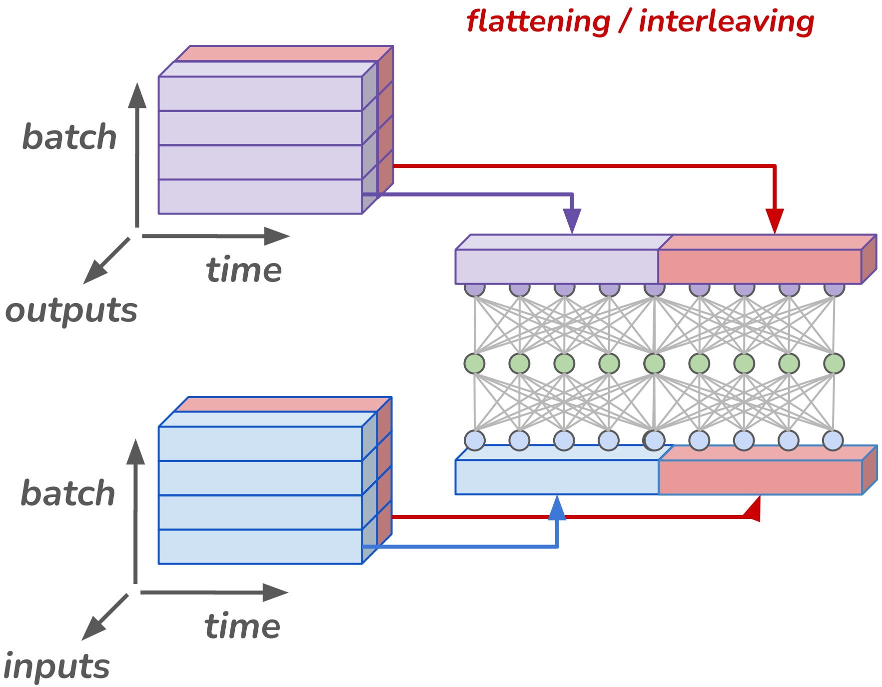


(B, T, I)⟶(B, 1)

Machine Learning College

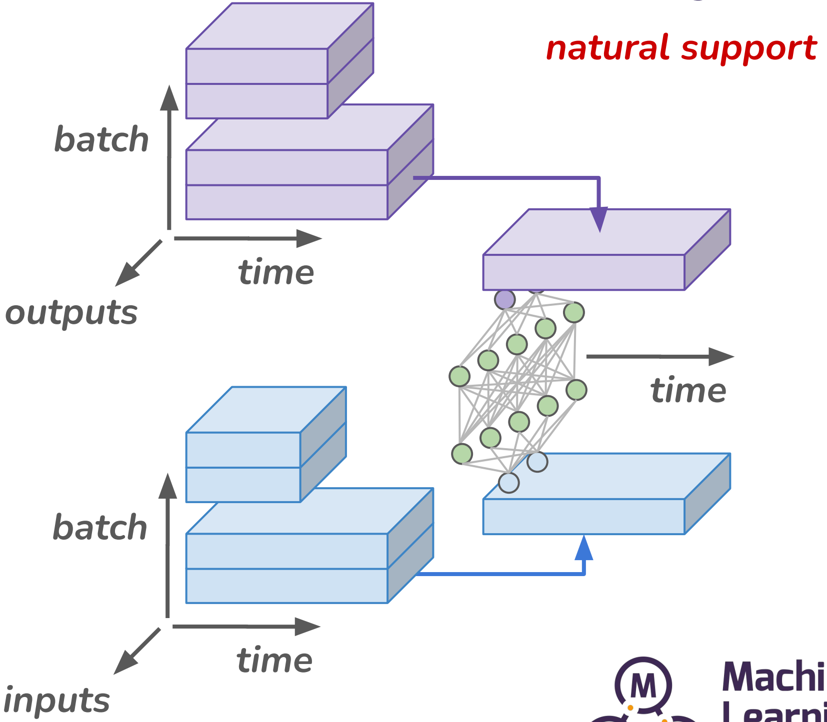
Tensors & dimentions – multivariate regression

Flat NN = 3D⟶2D training data



(B, T×I)⟶(B, T×O)

Recurrent NN = 3D training data



(B, T, I)⟶(B, T, O)

Machine Learning College

Time Series Modeling using Neural Networks – DAY 2

Dušan Fedorčák

11/2021 – RBI

Machine Learning College

Content

DAY 1

- **Classical time series analysis**
  - *Decomposition of time series*
  - *ARIMA models family*
  - *State space models generalization*
- **Theoretical window**
  - *Neural Networks & Recurrent NNs*
  - *Time series specifics*
- **Practical examples**
  - *Simple regression – toy example*
  - *Rainfall-runoff simulation – regression*

– lunch break –

- **Practical examples**
  - *Trampoline jumping – classification*
  - *Local Weather Forecast – regression*

DAY 2

- **Product Design & ML**
  - *Integration of ML models into products*
- **Practical Examples** (in random order)
  - *Exoplanets Hunting*
  - *Mobile Motion Sensing*

– lunch break –

- **Tips & tricks for debuging NNs**
- **Practical Examples**
  - *Manufacturing Process Modeling*
  - *Financial distress prediction*

Machine Learning College

ML & Product Design – problems & decisions

- ML should solve **problems**
  - AI is cool ⇒ let's use it in the product! ✖
  - We need to solve this problem ⇒ can we apply ML? ✔
  - *Q: What other means of solving the problem are available?*

ML & Product Design – problems & decisions

- ML should solve **problems**
  - AI is cool ⇒ let's use it in the product! ✖
  - We need to solve this problem ⇒ can we apply ML? ✔
  - *Q: What other means of solving the problem are available?*

- Problems vary in **difficulty**
  - Easily solvable by human ⇒ ML helps to scale up & automate
    - spam filter, face recognition, driving a car
  - Not easily solvable by human ⇒ ML can bring some solution
    - weather forecast, stock market prediction
  - *Q: What are we optimizing for? (costs, risk reduction, better service, …)*

Machine Learning College

ML & Product Design – problems & decisions

- ML should solve **problems**
  - AI is cool ⇒ let's use it in the product! ✖
  - We need to solve this problem ⇒ can we apply ML? ✔
  - *Q: What other means of solving the problem are available?*
- Problems vary in **difficulty**
  - Easily solvable by human ⇒ ML helps to scale up & automate
    - spam filter, face recognition, driving a car
  - Not easily solvable by human ⇒ ML can bring some solution
    - weather forecast, stock market prediction
  - *Q: What are we optimizing for? (costs, risk reduction, better service, …)*
- Problems boils down to **decisions**
  - ML can **assist** with decisions
  - ML can **automate** decisions
  - *Q: Could assistance model work for us or full automation is needed?*

Machine Learning College

ML & Product Design – automation

- ML solutions are **imperfect**
    - Expectation control & automation bias ⇒ trust issues
    - Scaling up implerfect models ⇒ quality issues
    - Right evaluation metrics – model evaluation vs. UX evaluation
    - *Q: Do all involved parties understand the problem & solution?*
    - *Q: Is there an evaluation metric everybody understands and agrees with?*

ML & Product Design – automation

- ML solutions are **imperfect**
  - Expectation control & automation bias ⇒ trust issues
  - Scaling up implerfect models ⇒ quality issues
  - Right evaluation metrics – model evaluation vs. UX evaluation
  - *Q: Do all involved parties understand the problem & solution?*
  - *Q: Is there an evaluation metric everybody understands and agrees with?*

- **Black-box** models are tricky
  - Black-box model + automation ⇒ trust issues
  - Expensive configurability & finetuning
  - *Q: Can we build an understandable stress-test evaluation dataset?*
  - *Q: What tools or probes are avaiable for analysis of our learned black-box model?*

Machine Learning College

ML & Product Design – automation

- ML solutions are **imperfect**
  - Expectation control & automation bias ⇒ trust issues
  - Scaling up implerfect models ⇒ quality issues
  - Right evaluation metrics – model evaluation vs. UX evaluation
  - *Q: Do all involved parties understand the problem & solution?*
  - *Q: Is there an evaluation metric everybody understands and agrees with?*

- **Black-box** models are tricky
  - Black-box model + automation ⇒ trust issues
  - Expensive configurability & finetuning
  - *Q: Can we build an understandable stress-test evaluation dataset?*
  - *Q: What tools or probes are avaiable for analysis of our learned black-box model?*

- Assistance ⇔ Automation – there is a **spectrum**
  1. No automation
  2. Scored set of possible decisions
  3. Narrowed set of decision to approve
  4. Veto before automatic execution
  5. Full automation
  - *Q: What is the lowest level of automation that brings value*

Machine Learning College

ML & Product Design – data analysis

- ML solutions **depends on data**
  - More complex models ⇒ more data required
  - Constant battle agaings overfitting
  - Distributions shift over time
  - *Q: What data is available and will be available in the future*

Machine Learning College

ML & Product Design – data analysis

- ML solutions **depends on data**
  - More complex models ⇒ more data required
  - Constant battle agaings overfitting
  - Distributions shift over time
  - *Q: What data is available and will be available in the future*
- **Base rates** in data influences products
  - Base rates can have unintuitive effects on the product
  - Sampling reality often produces imbalanced data
  - *Q: What would be the performace of near-perfect model given the base rates*

Machine Learning College

ML & Product Design – data analysis

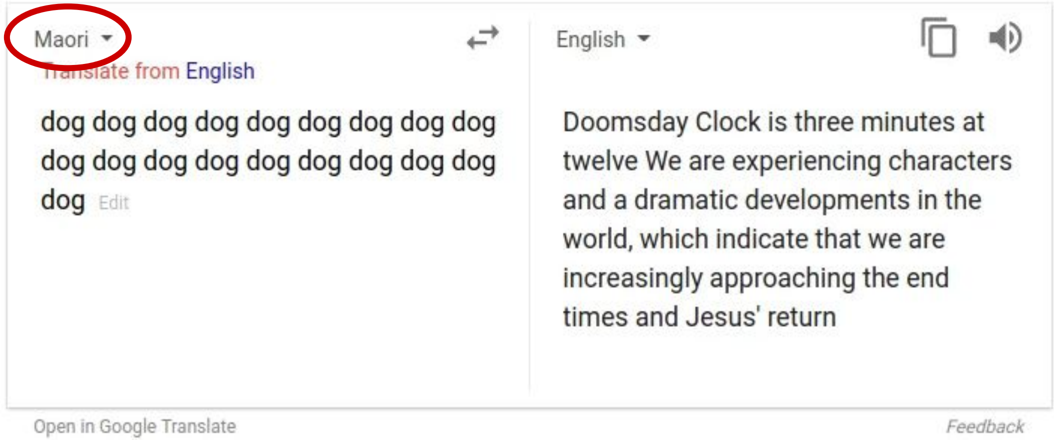
- ML solutions **depends on data**
  - More complex models ⇒ more data required
  - Constant battle agaings overfitting
  - Distributions shift over time
  - *Q: What data is available and will be available in the future*
- **Base rates** in data influences products
  - Base rates can have unintuitive effects on the product
  - Sampling reality often produces imbalanced data
  - *Q: What would be the performace of near-perfect model given the base rates*
- All datasets are **biased**
  - Inconsistency between data sampling and model goals
  - Biased evaluation sets
  - *Q: Does our historical data reliably capture the goal of the model*

Machine Learning College

Dataset issues – insufficient number of samples

Machine Learning College

Dataset issues – insufficient number of samples



https://www.skynettoday.com/briefs/google-nmt-prophecies

Machine Learning College

Dataset issues – biased training set

TECHNOLOGY NEWS OCTOBER 10, 2018 / 5:43 AM / UPDATED 2 YEARS AGO

Insight - Amazon scraps secret AI recruiting tool that showed bias against women

By Jeffrey Dastin

8 MIN READ

SAN FRANCISCO (Reuters) - Amazon.com Inc's AMZN.O machine-learning specialists uncovered a big problem: their new recruiting engine did not like women.

Machine Learning College

https://www.reuters.com/article/us-amazon-com-jobs-automation-insight-idUSKCN1MK08G

Dataset issues – biased training set

Machine Learning College

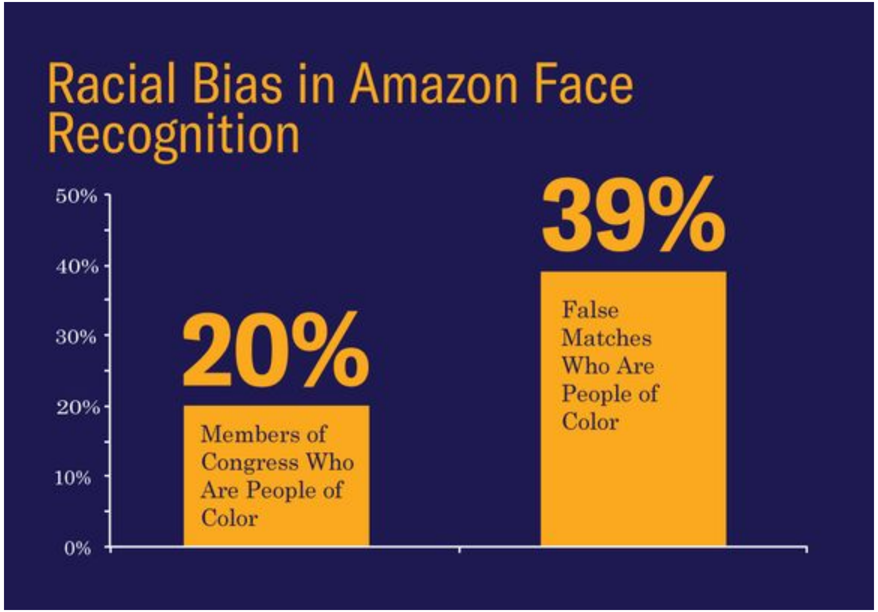
https://www.aclu.org/blog/privacy-technology/surveillance-technologies/amazons-face-recognition-falsely-matched-28

Dataset issues – biased training set

https://www.aclu.org/blog/privacy-technology/surveillance-technologies/amazons-face-recognition-falsely-matched-28

Machine Learning College

Dataset issues – biased training set



Machine Learning College

https://www.aclu.org/blog/privacy-technology/surveillance-technologies/amazons-face-recognition-falsely-matched-28

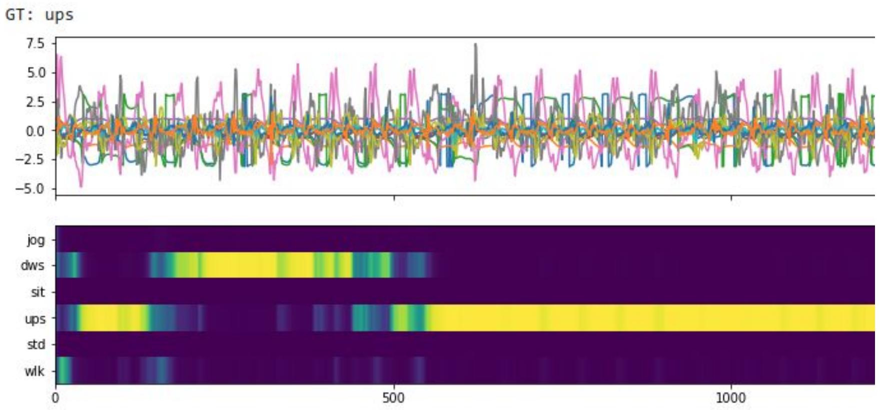
Human perception – overfitted to faces

https://en.wikipedia.org/wiki/Pareidolia

Machine Learning College

Motion sensing example

- Data preparation
  - Dataset normalization
  - Slicing long sequences
- Categorical classification task
  - Predict activity type
  - Use correct activation & loss function
- Training & evaluation
  - Try different architectures
  - Evaluate resuslt with standard metrics
- Secondary task
  - Subject identification



Machine Learning College

ML Tips & Tricks

http://karpathy.github.io/2019/04/25/recipe/

ML Tips & Tricks

- **Known your data**
    - Visualize everything you can
    - Try to find patterns ⇒ *become the model yourself*
    - Look for noisy labels / missing data
    - Make sure your preprocessing is correct (especially vectorized code)

http://karpathy.github.io/2019/04/25/recipe/

ML Tips & Tricks

- **Known your data**
  - Visualize everything you can
  - Try to find patterns ⇒ *become the model yourself*
  - Look for noisy labels / missing data
  - Make sure your preprocessing is correct (especially vectorized code)
- **Simple models first**
  - Build training & evaluation loop
  - Choose simple architectures first ⇒ *less room for errors*
  - Build baseline models for comparison ⇒ *even simple heuristics are useful*

http://karpathy.github.io/2019/04/25/recipe/

ML Tips & Tricks

- **Known your data**
  - Visualize everything you can
  - Try to find patterns ⇒ *become the model yourself*
  - Look for noisy labels / missing data
  - Make sure your preprocessing is correct (especially vectorized code)
- **Simple models first**
  - Build training & evaluation loop
  - Choose simple architectures first ⇒ *less room for errors*
  - Build baseline models for comparison ⇒ *even simple heuristics are useful*
- **Train iteratively**
  - Train without inputs ⇒ *yields another baseline model*
  - Overfit one batch ⇒ *something is off if you can't get zero loss*
  - Overfit the training set as far as you can

http://karpathy.github.io/2019/04/25/recipe/

Machine Learning College

ML Tips & Tricks

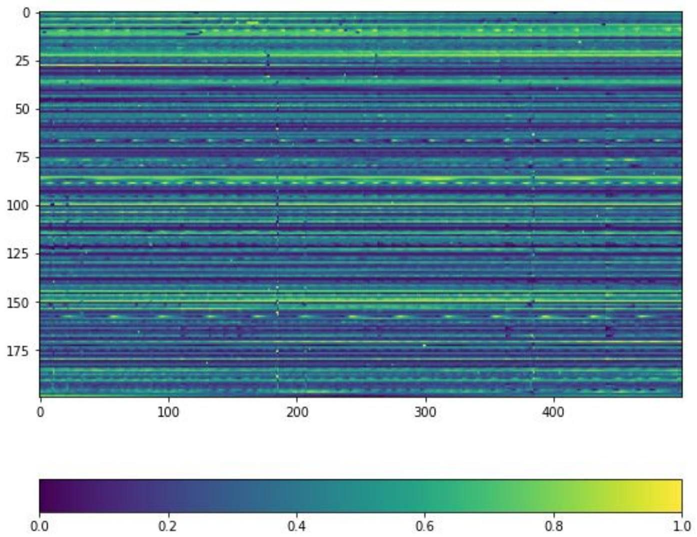
- **Known your data**
  - Visualize everything you can
  - Try to find patterns ⇒ *become the model yourself*
  - Look for noisy labels / missing data
  - Make sure your preprocessing is correct (especially vectorized code)
- **Simple models first**
  - Build training & evaluation loop
  - Choose simple architectures first ⇒ *less room for errors*
  - Build baseline models for comparison ⇒ *even simple heuristics are useful*
- **Train iteratively**
  - Train without inputs ⇒ *yields another baseline model*
  - Overfit one batch ⇒ *something is off if you can't get zero loss*
  - Overfit the training set as far as you can
- **Regularize**
  - Early stopping ⇒ *best evaluation loss*
  - Make the model smaller ⇒ *less space for overfitting*
  - Get more training data ⇒ *more labels, data augmentation, pre-training*

http://karpathy.github.io/2019/04/25/recipe/

Machine Learning College

Exoplanets hunting example

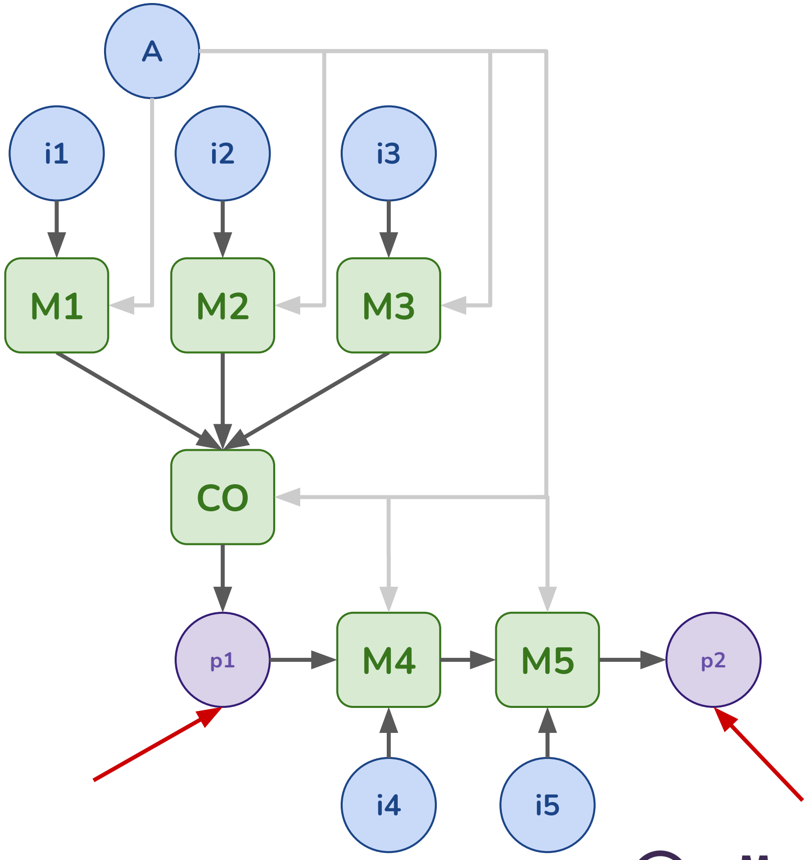
- Data preparation
  - Dataset normalization
  - Highly **imbalanced dataset**
- Binary classification/detection task
  - Detect starts with planets
- Model architecture
  - Dense, LSTM, Bidir. LSTM, CNN
- Training & evaluation
  - Use right evaluation metrics for imbalanced datasets



Machine Learning College

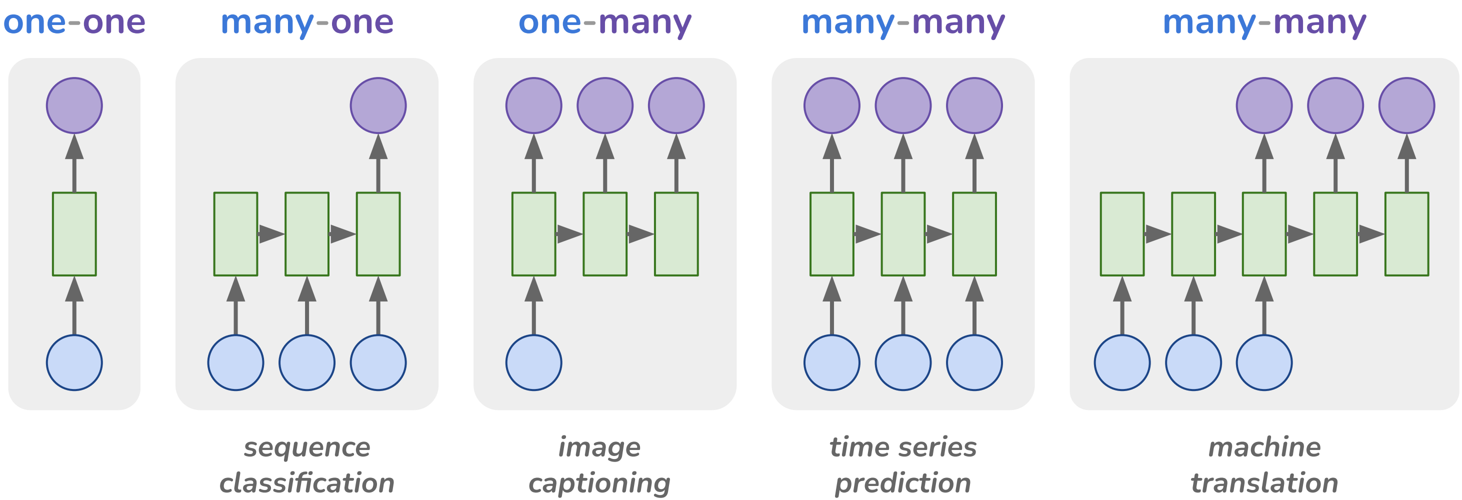
Factory process example

- Data preparation
  - Dealing with **missing values**
  - Dataset normalization
  - Slicing long sequences
- Regression task
  - Predict target variables in future
- Model architecture
  - **Model architecture mimics the process**
- Training & evaluation
  - Masking out missing labels with **custom loss function**



Machine Learning College

RNN and sequence data

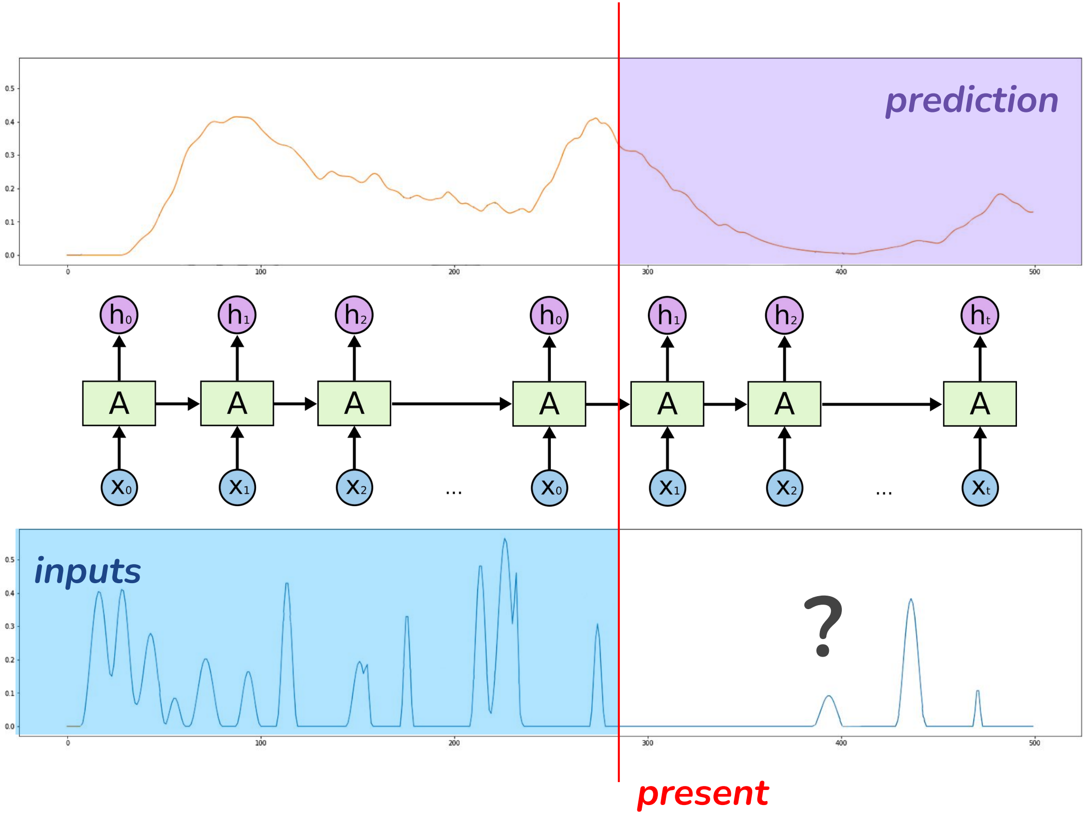


Machine Learning College

*http://karpathy.github.io/2015/05/21/rnn-effectiveness/

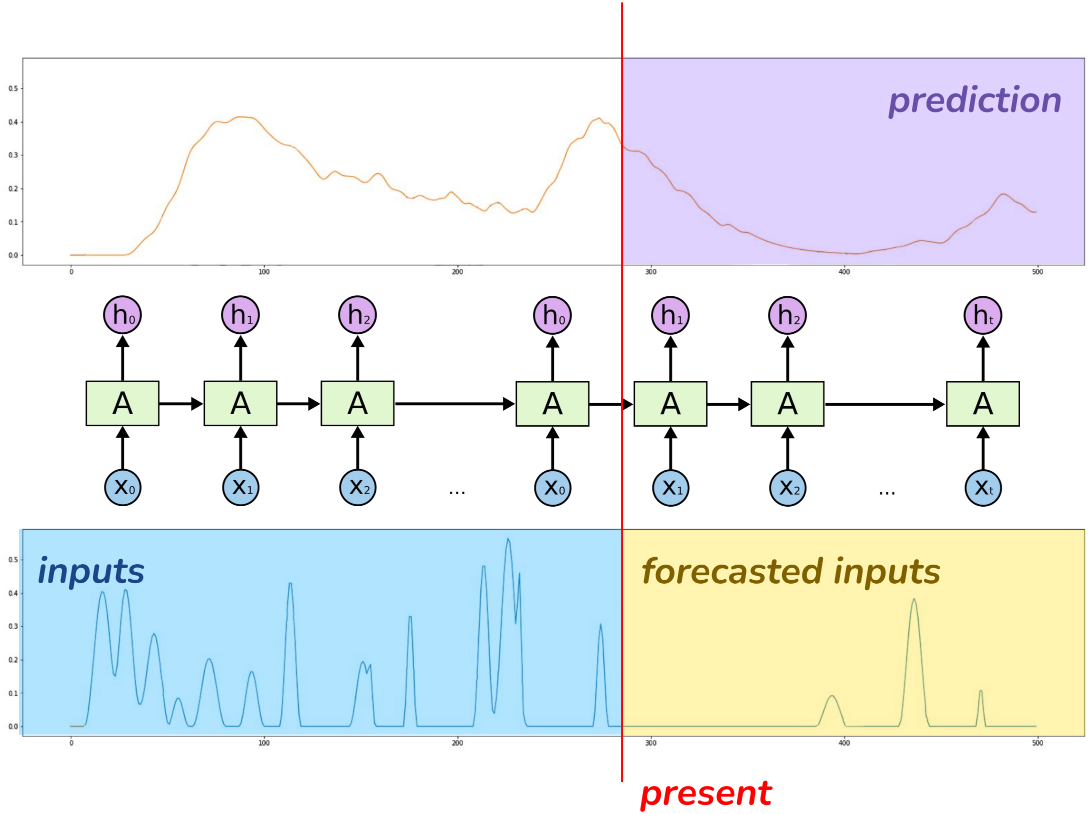
RNN for time series prediction

Forecasting from input variables

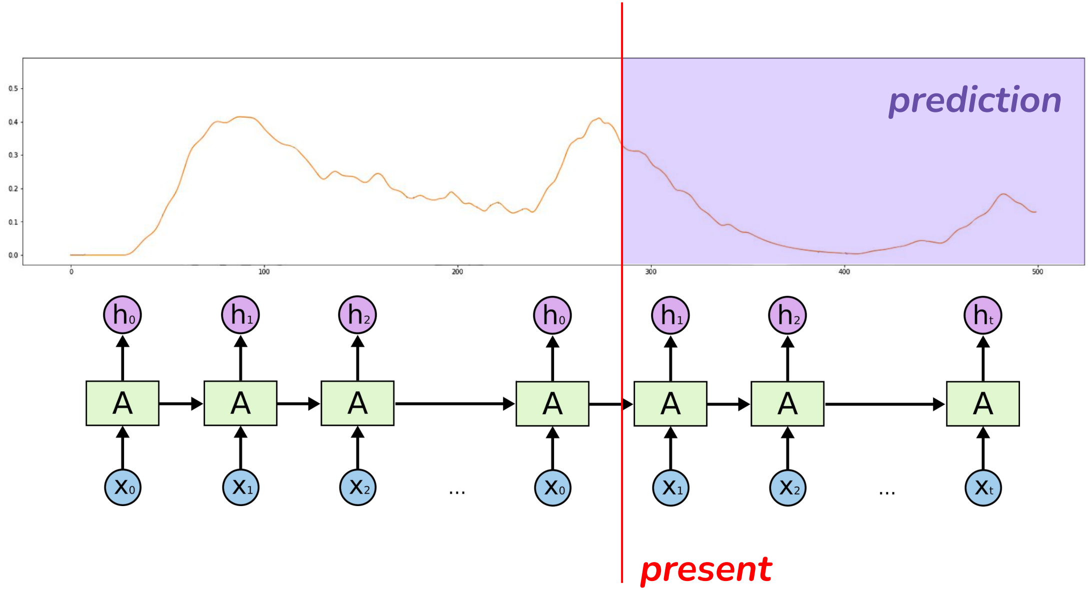


Machine Learning College

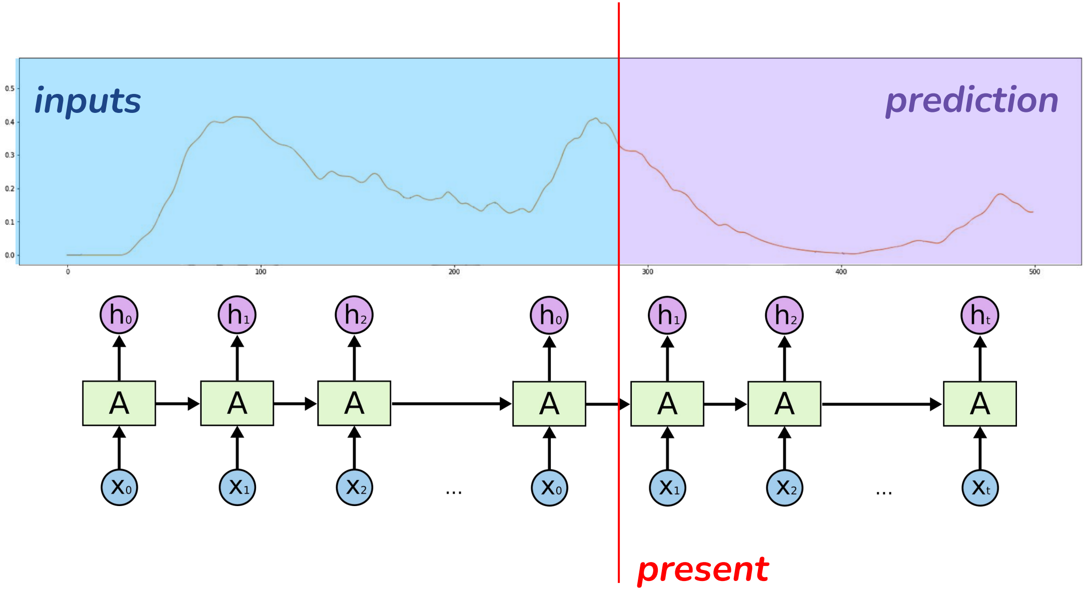
Forecasting from input variables



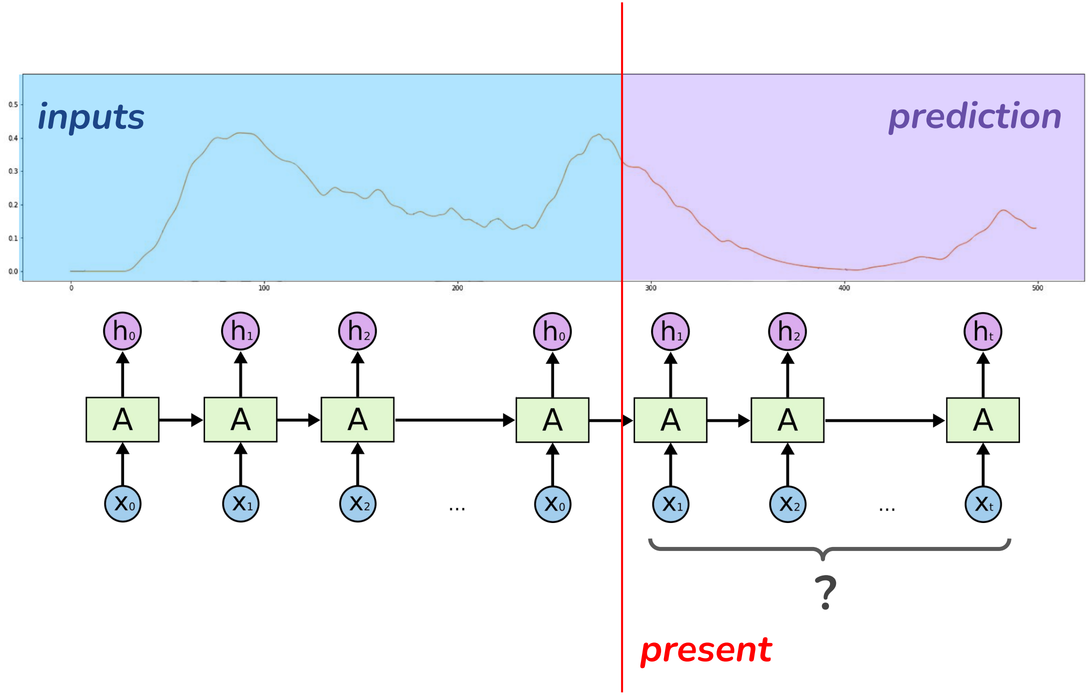
Forecasting from historical values



Forecasting from historical values

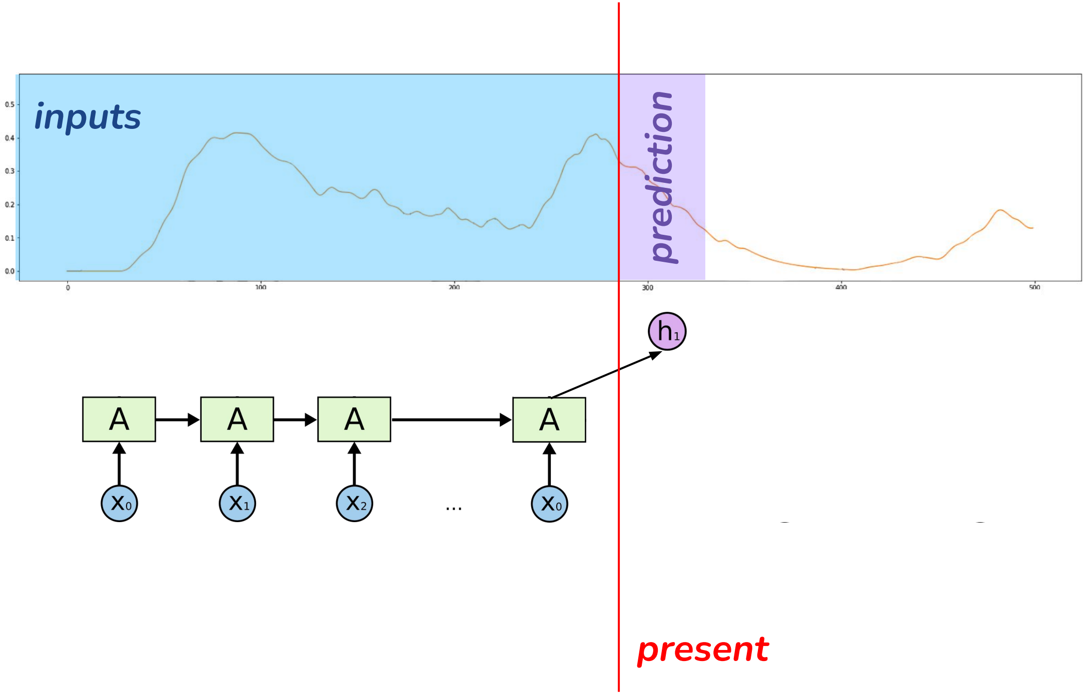


Forecasting from historical values



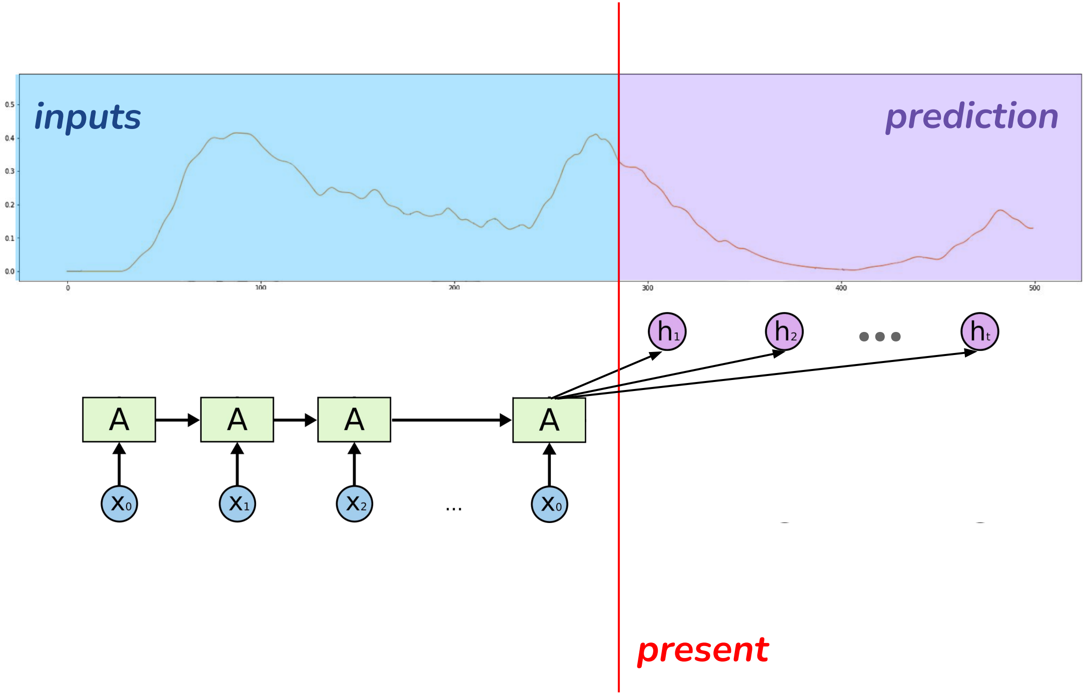
Machine Learning College

Forecasting – one step ahead

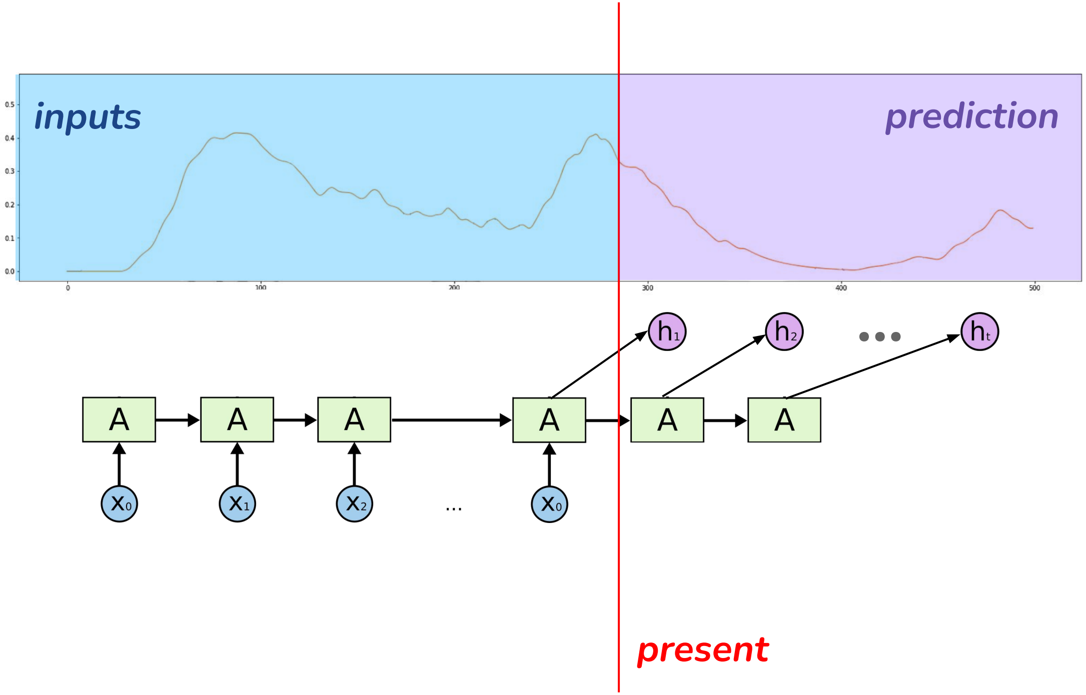


Machine Learning College

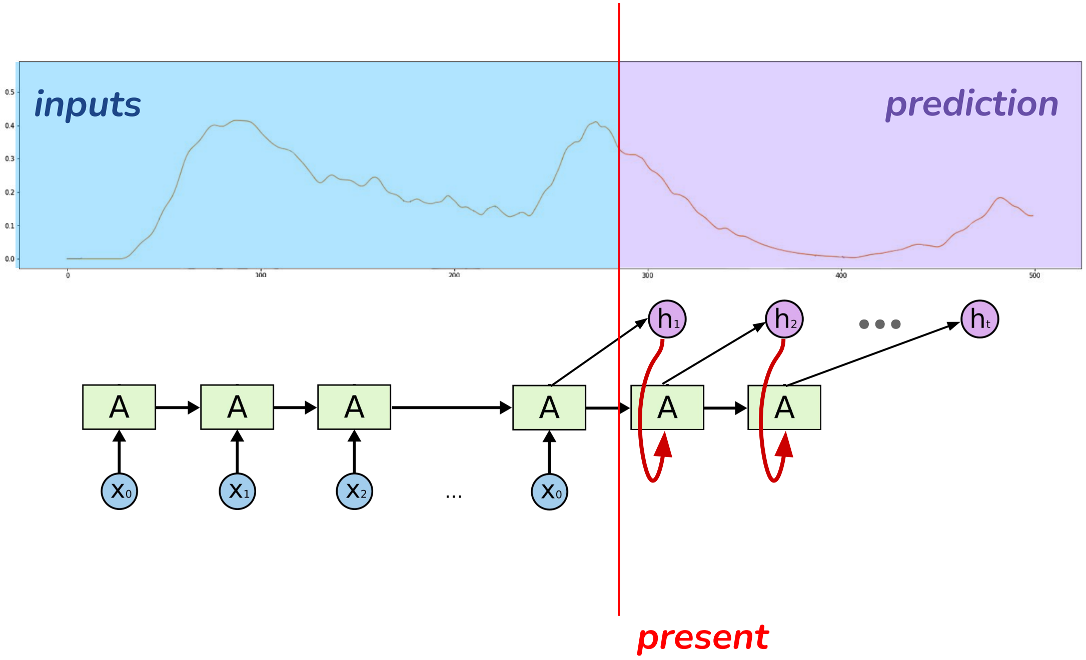
Forecasting – flat multi-step prediction



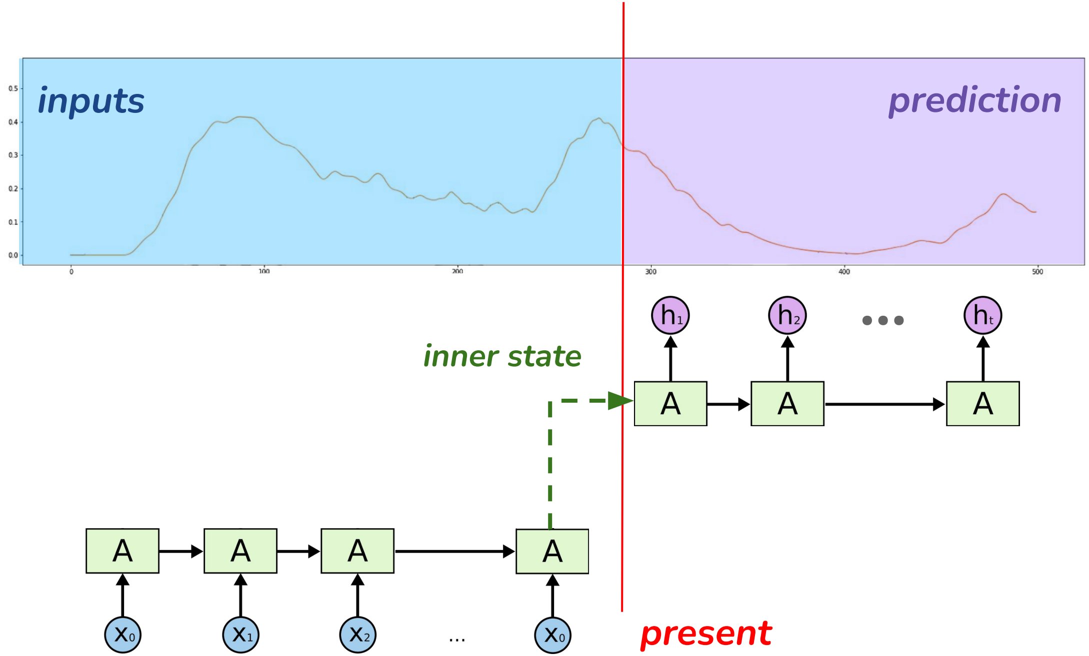
Forecasting – developed multi-step predition



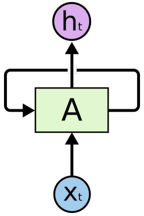
Forecasting – developed multi-step predition

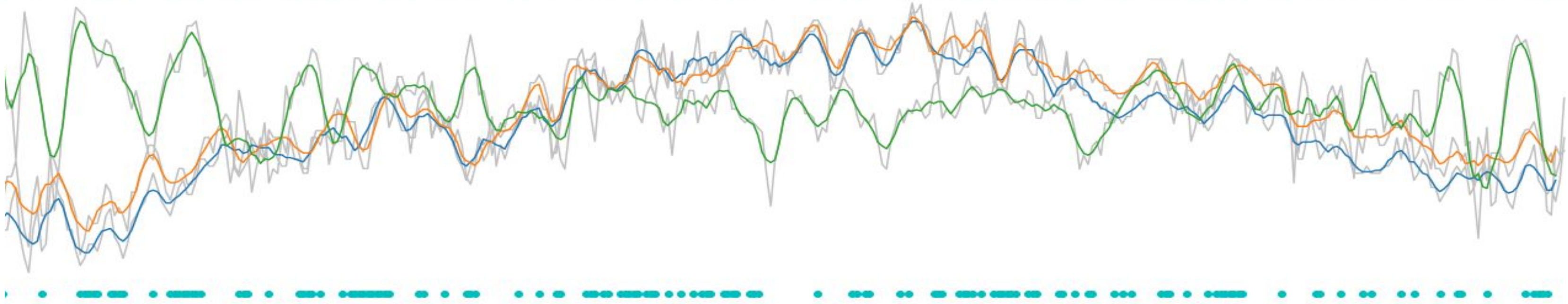


Forecasting – encoder & decoder

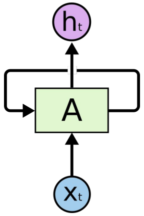


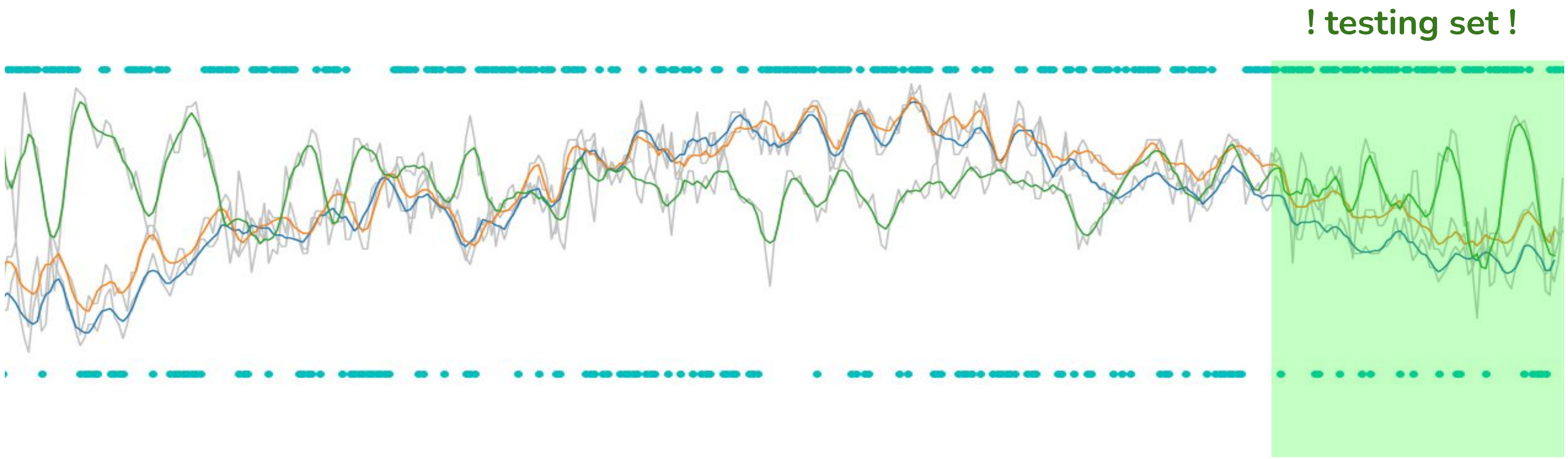
Training set construction



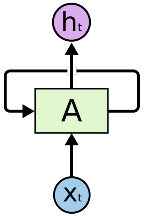


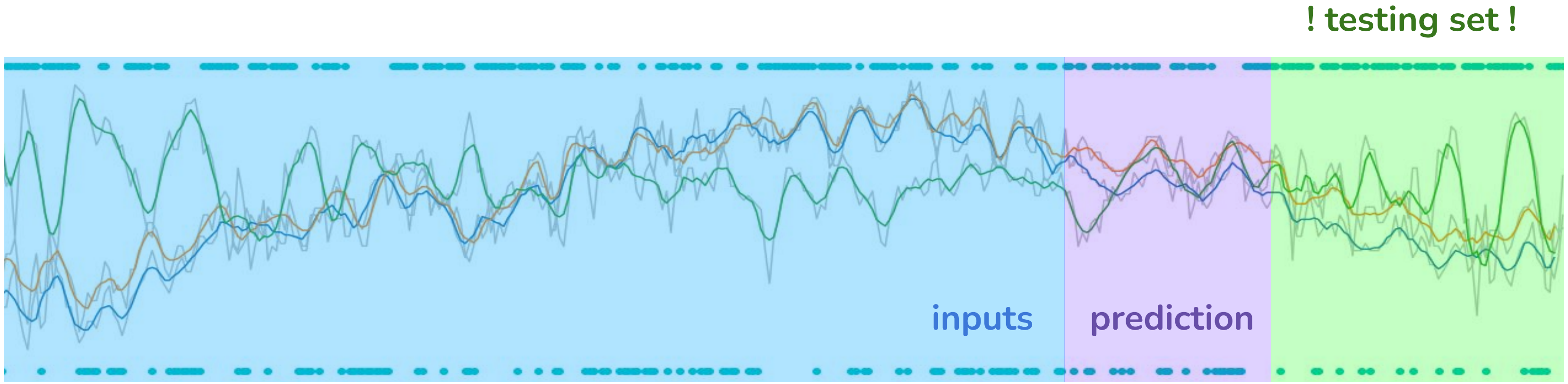
Training set construction



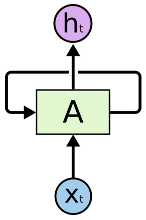


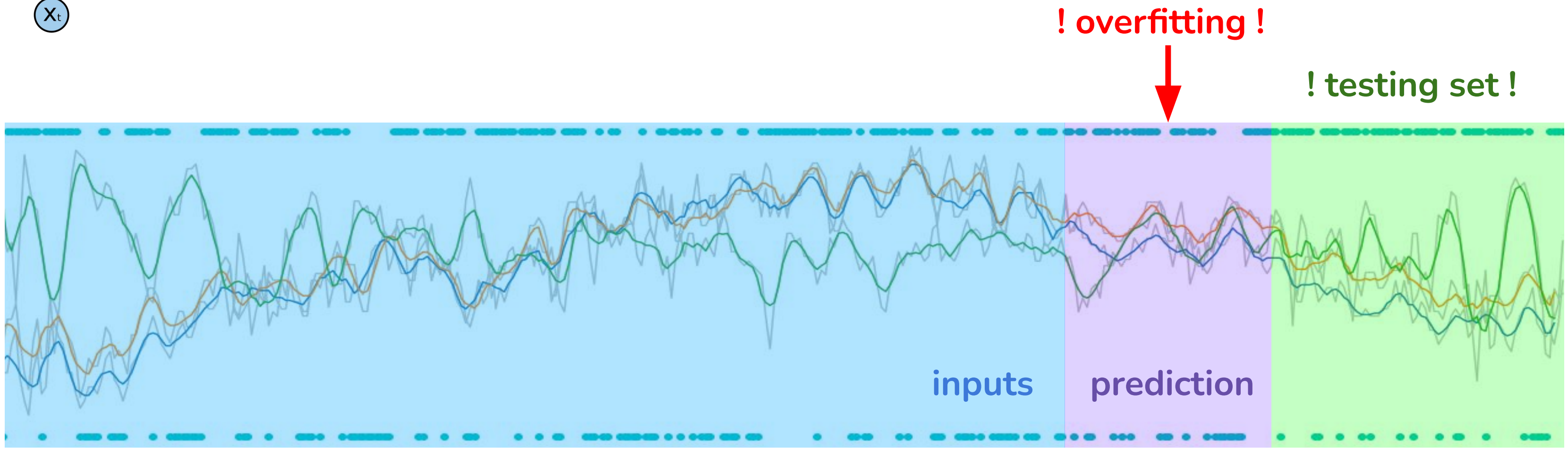
Training set construction



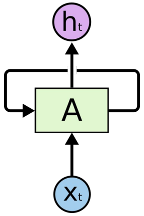


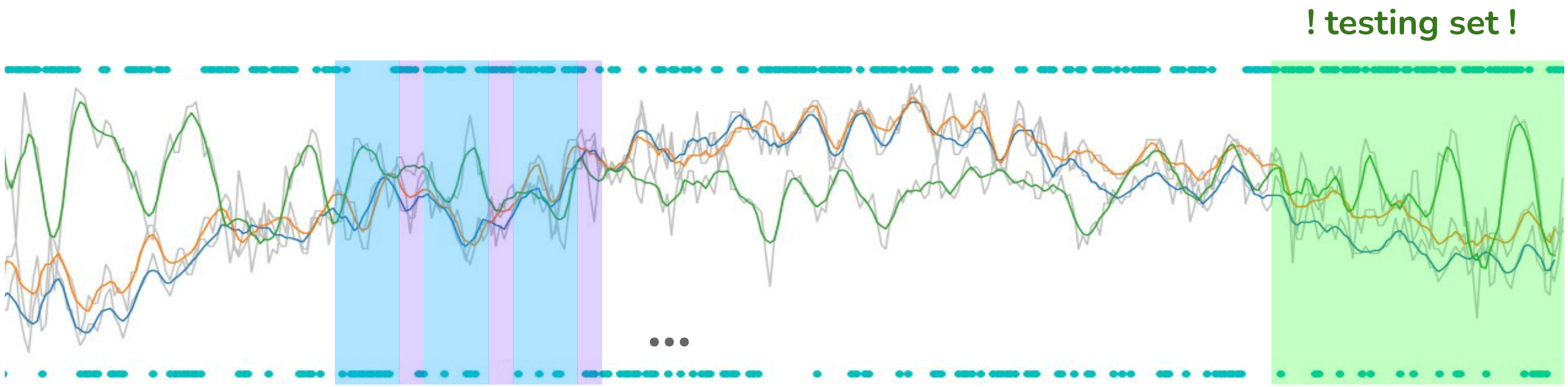
Training set construction



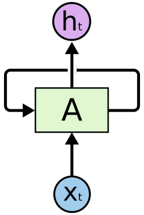


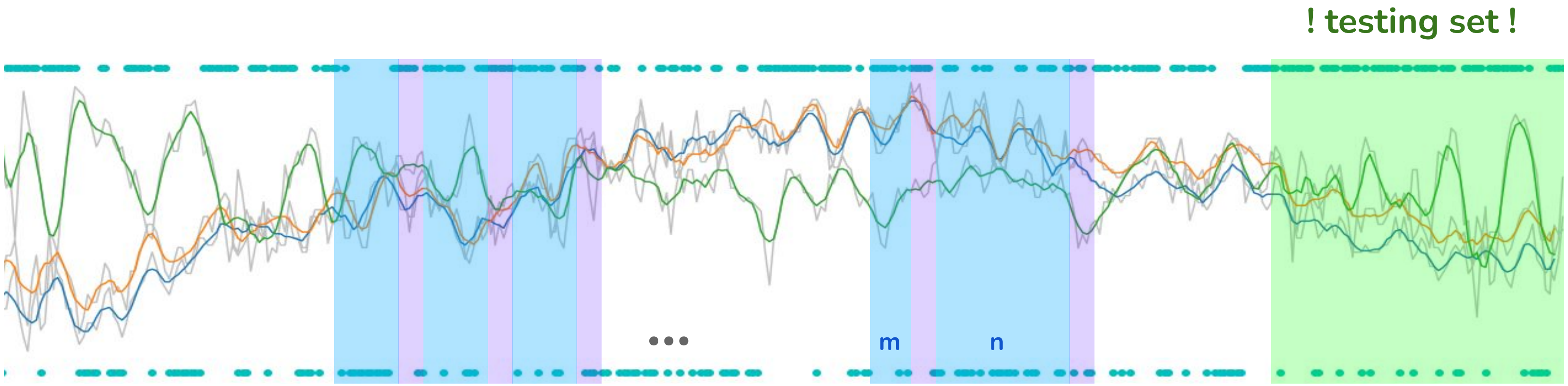
Training set construction



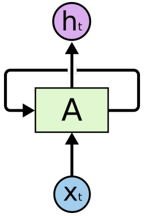


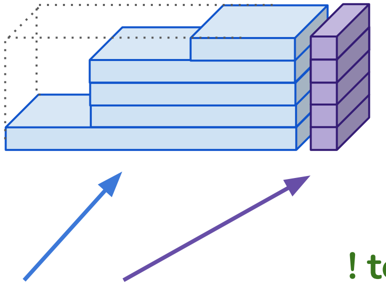
Training set construction

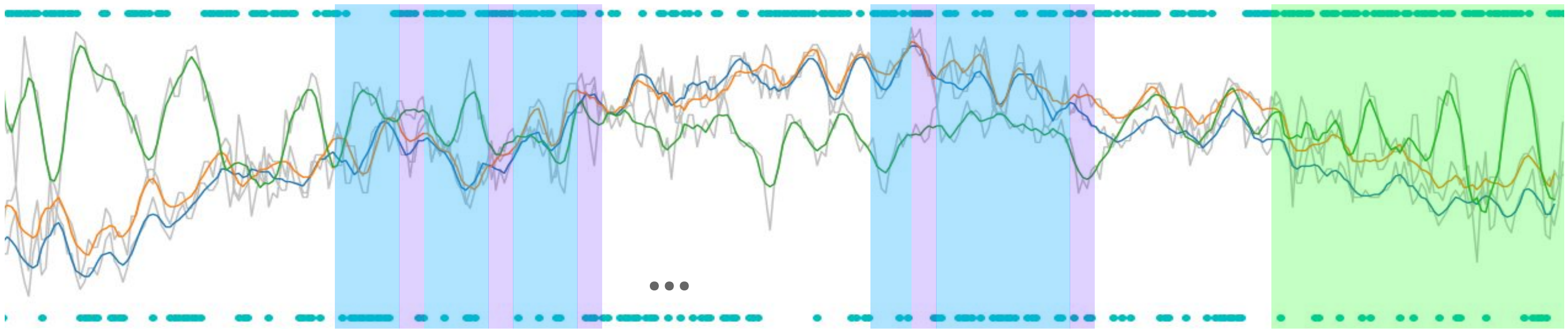




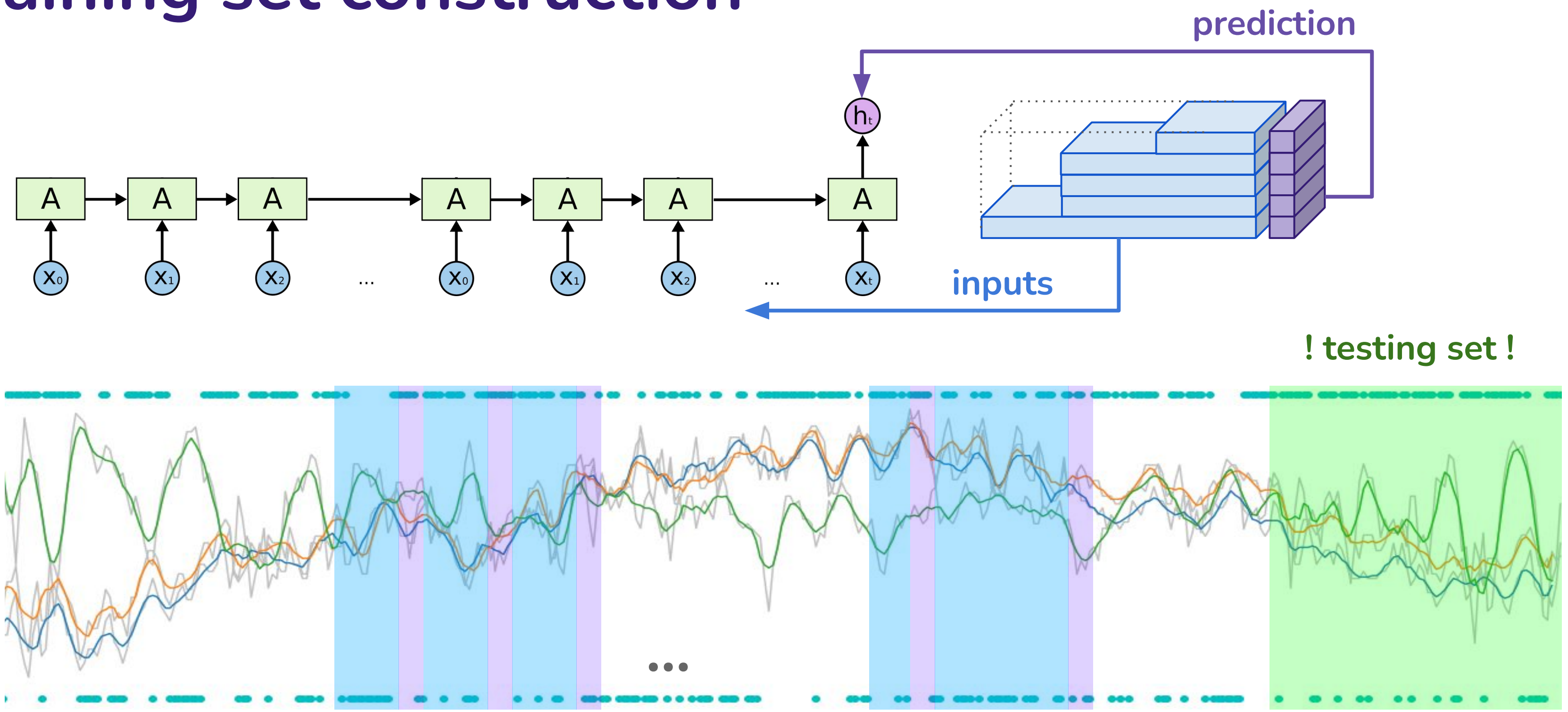
Training set construction



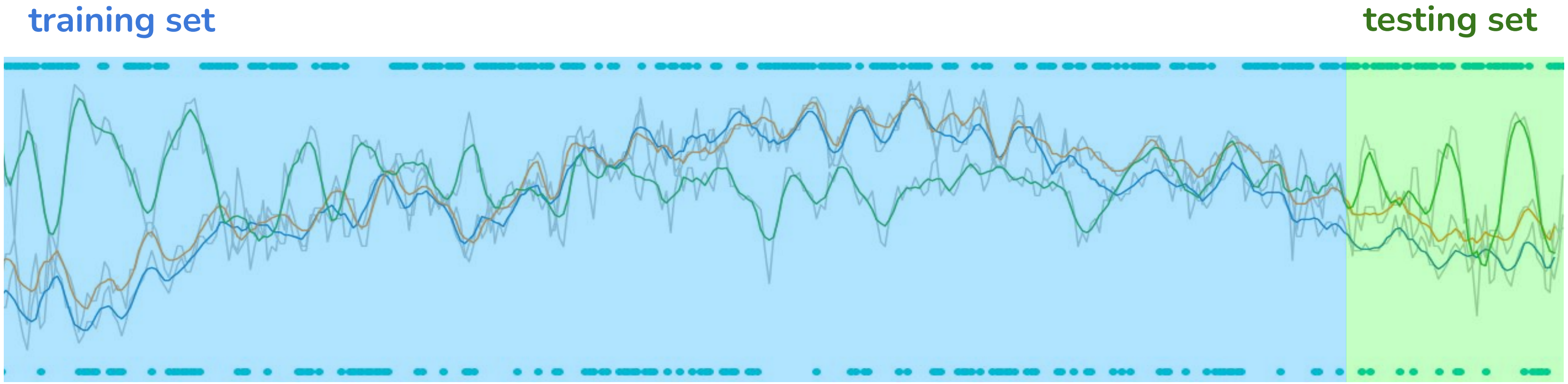




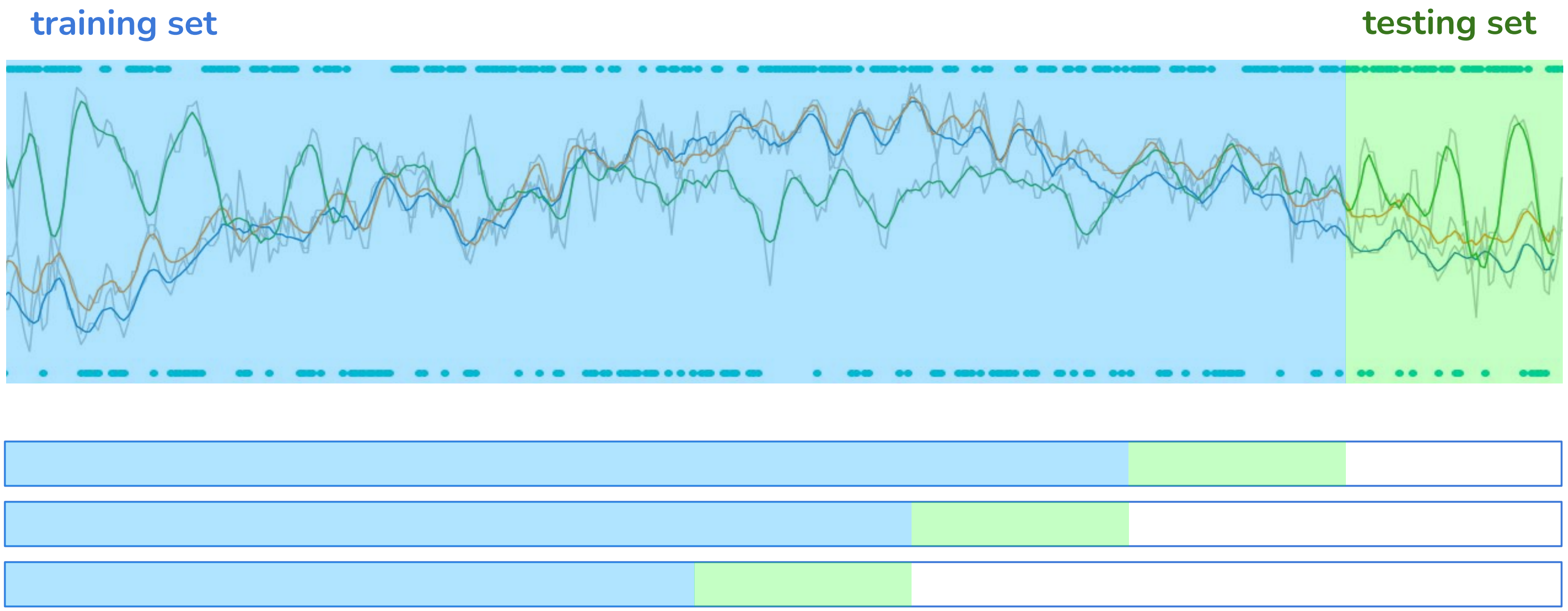
Training set construction



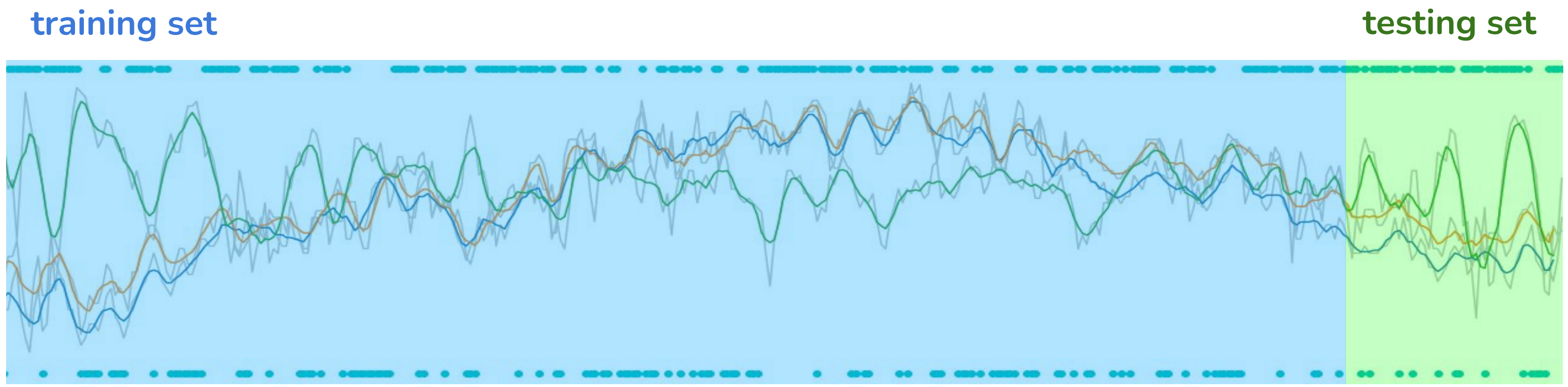
Training set construction

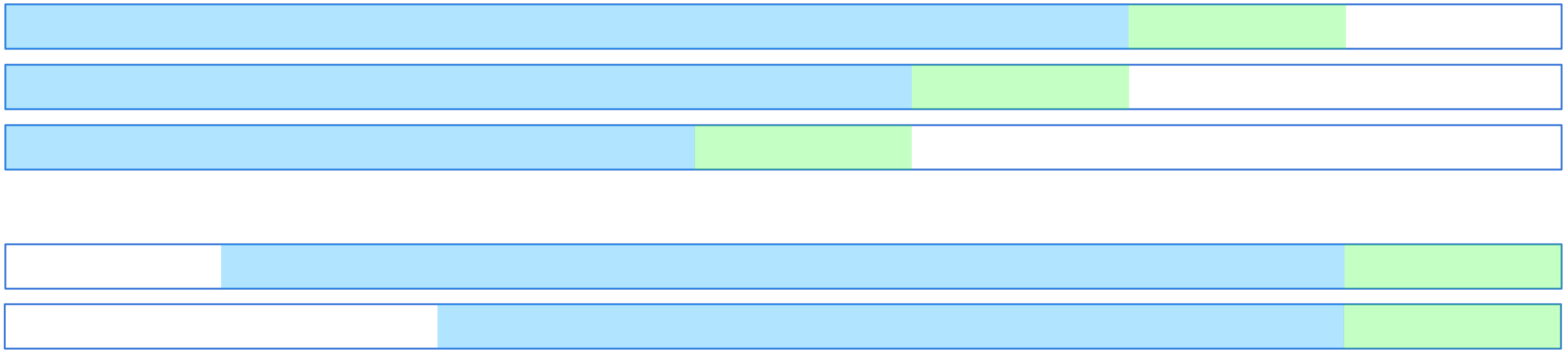


Training set construction



Training set construction

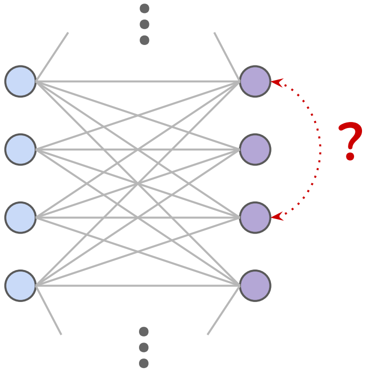




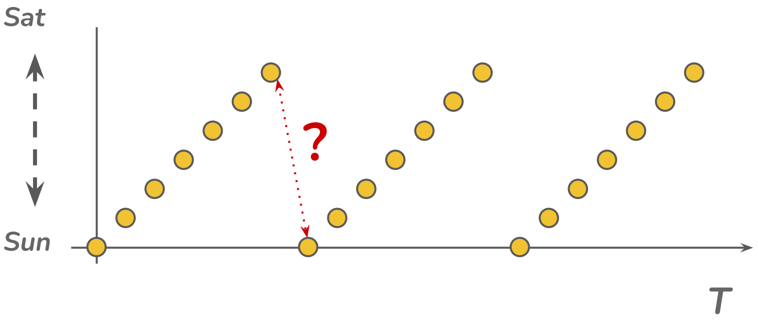
Feature Encoding – Seasonal dummy variables

Sun	0	0	0	0
Mon	**1**	0	0	0
Tue	0	**1**	0	0
Wed	0	0	**1**	0
Thu	0	0	0	**1**
Fri	0	0	0	0
Sat	0	0	0	0

T →

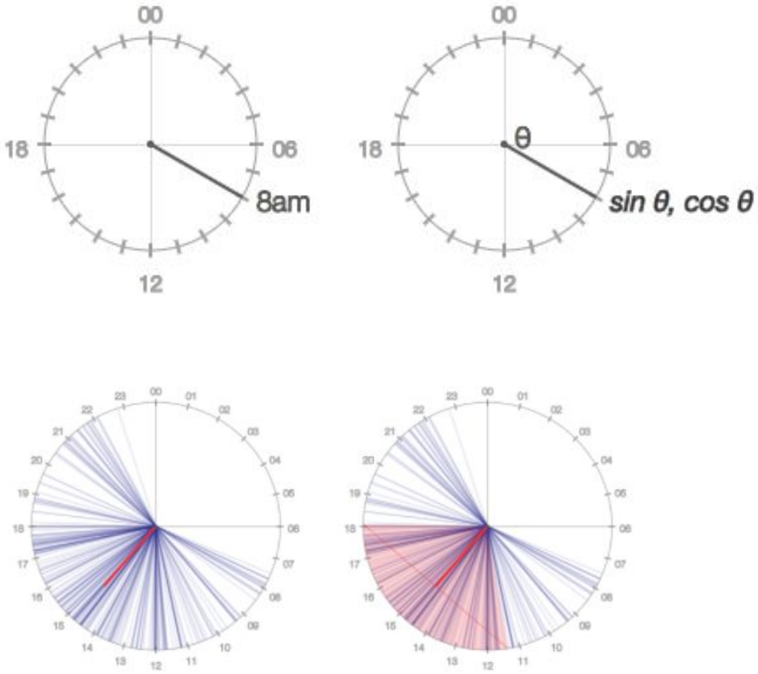


- *(hour of day, day of week, ...)*
- *Numerical variables*
- *One-hot encoding*

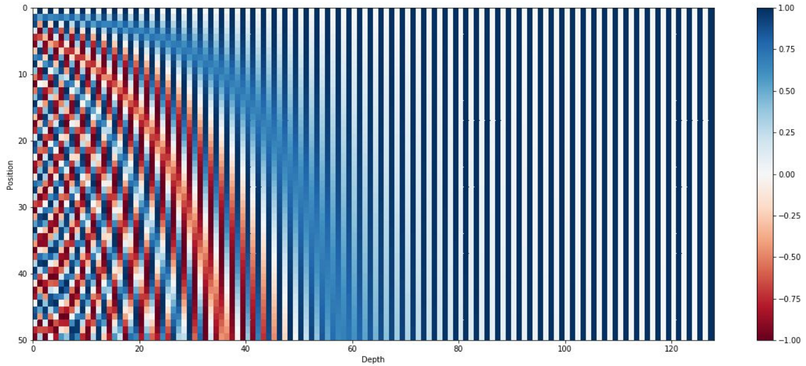


https://medium.com/life-at-hopper/ai-in-travel-part-2-representing-cyclic-and-geographic-features-4ada33dd0b22
https://kazemnejad.com/blog/transformer_architecture_positional_encoding/

Feature Encoding – Seasonal dummy variables



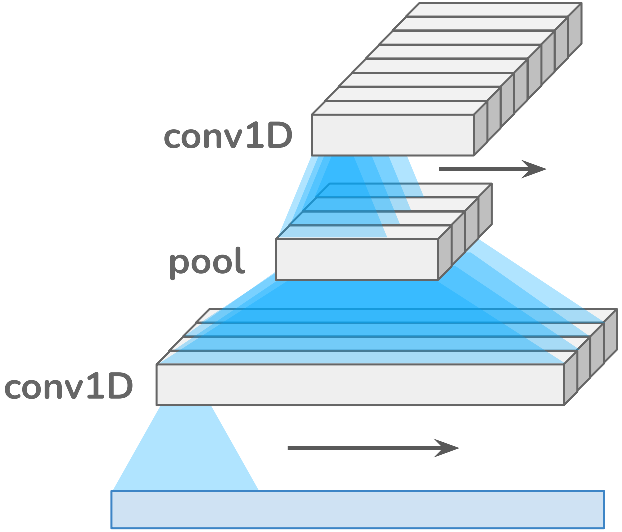
- *Circular encoding*
- *Positional embedding (transformers)*

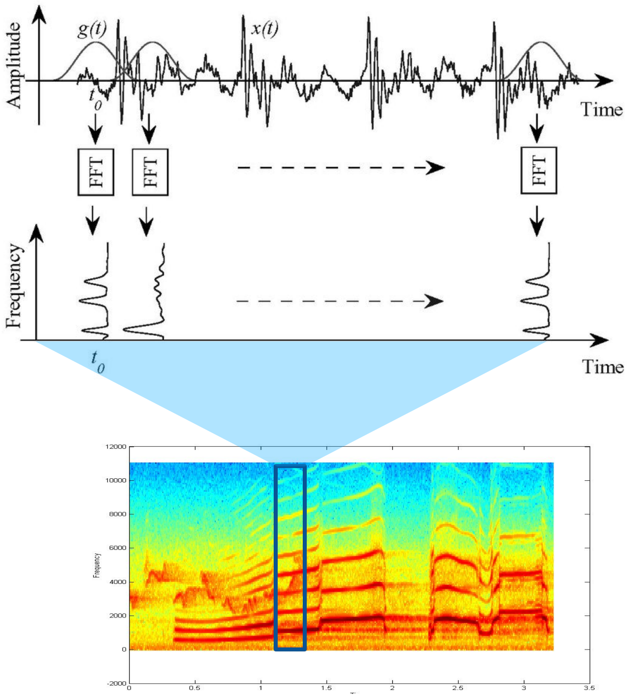


https://medium.com/life-at-hopper/ai-in-travel-part-2-representing-cyclic-and-geographic-features-4ada33dd0b22
https://kazemnejad.com/blog/transformer_architecture_positional_encoding/

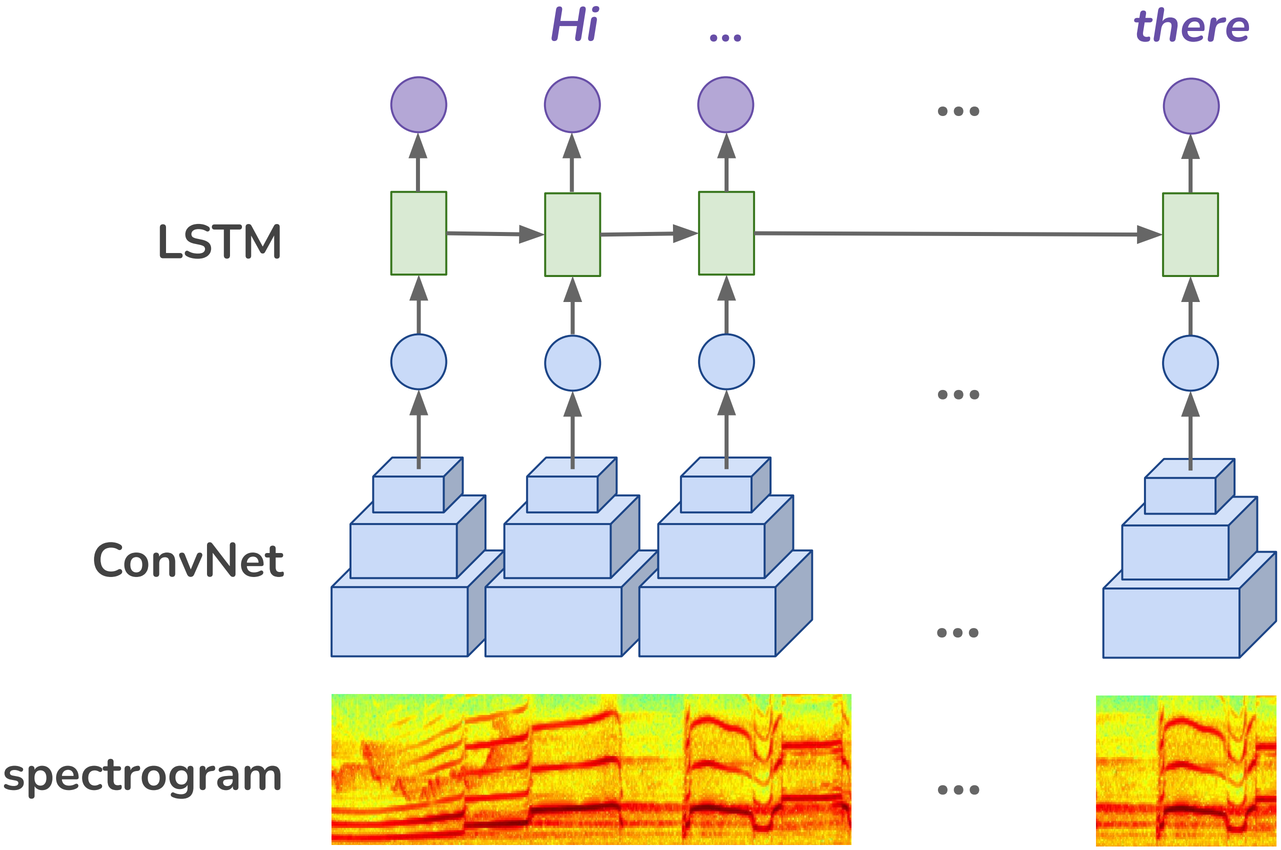
Feature Encoding II

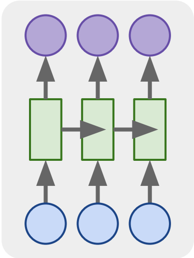
- Exchange for extra dimension
  - 1D Convolution & pooling
  - Short-Time Fourier Transform





Advanced architectures





ML Tips & Tricks

- Known your data
  - Visualize everything you can
  - Try to find patterns ⇒ *become the model yourself*
  - Look for noisy labels / missing data
  - Make sure your preprocessing is correct (especially vectorized code)
- Start with simple models
  - Build training & evaluation loop
  - Choose simple architectures first ⇒ *less room for errors*
  - Build baseline models for comparison ⇒ *even simple heuristics are useful*
- Train iteratively
  - Train without inputs ⇒ *yields another baseline model*
  - Overfit one batch ⇒ *something is off if you can't get zero loss*
  - Overfit the training set as far as you can
- Regularize
  - Early stopping ⇒ *best evaluation loss*
  - Make the model smaller ⇒ *less space for overfitting*
  - Get more training data ⇒ *more labels, data augmentation, pre-training*

http://karpathy.github.io/2019/04/25/recipe/

Additional learning materials

- Classical time series
  - Time series course – *a book-like explanation of basic principles of time-series and classical analysis*
  - Statistical forecasting – *detailed notes on classical time series analysis and ARIMA models*

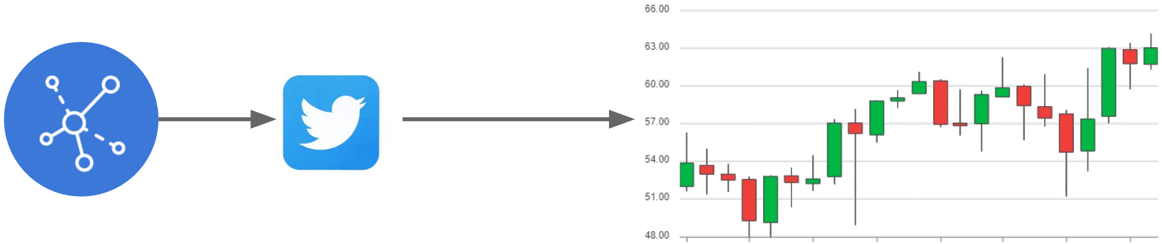
- Keras
  - Guides - *code examples for most of the basics in Keras*
  - Examples - *huge selection of code examples from different areas (time series, vision, ...)*
  - Blog – *good selection of advanced application of Keras on practical problems*

- Interesting blogs
  - Adam Geitgey – *Machine learning is fun – great selection of simple examples from various areas*
  - Christopher Olah – *very well-described principles of neural networsk (with a lot of visual insights)*
  - Andrej Karpathy – *some very interesting insights (including the debug recipes for NNs)*
  - Distill – *Chris Olah and Shan Carter collaboration – open problems in deep learning & advanced topics*

- Tech companies blogs
  - DeepMind (Google) – *top research in artificial intelligence – usually acompanied with science papers*
  - OpenAI – *started as non-commercial research group / answer to Deepmind*
  - Facebook – *many interesting projects sometimes with free-to-use pre-learned models*
  - Amazon – *many interesting machine learning articles sometimes with detailed papers*

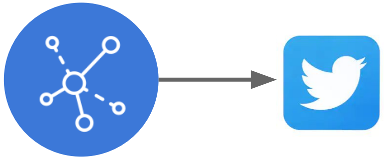
Machine Learning College

Time series prediction from textual data

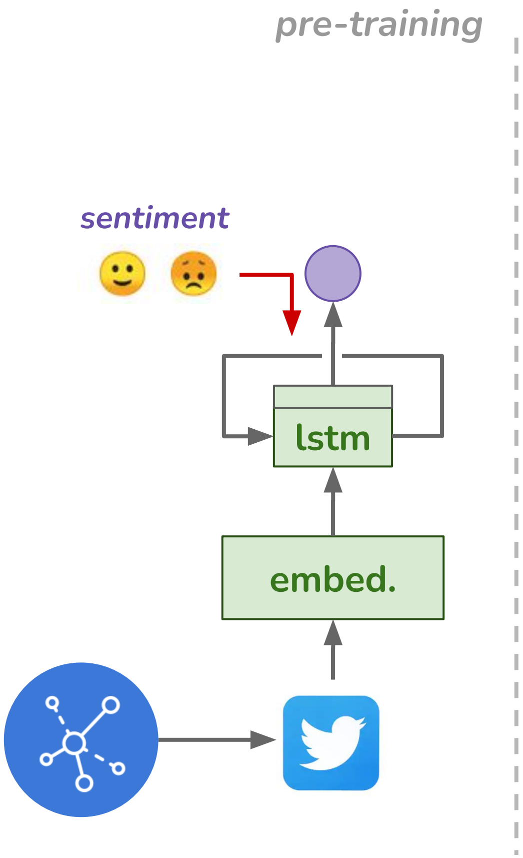


Machine
Learning
College

Time series prediction from textual data

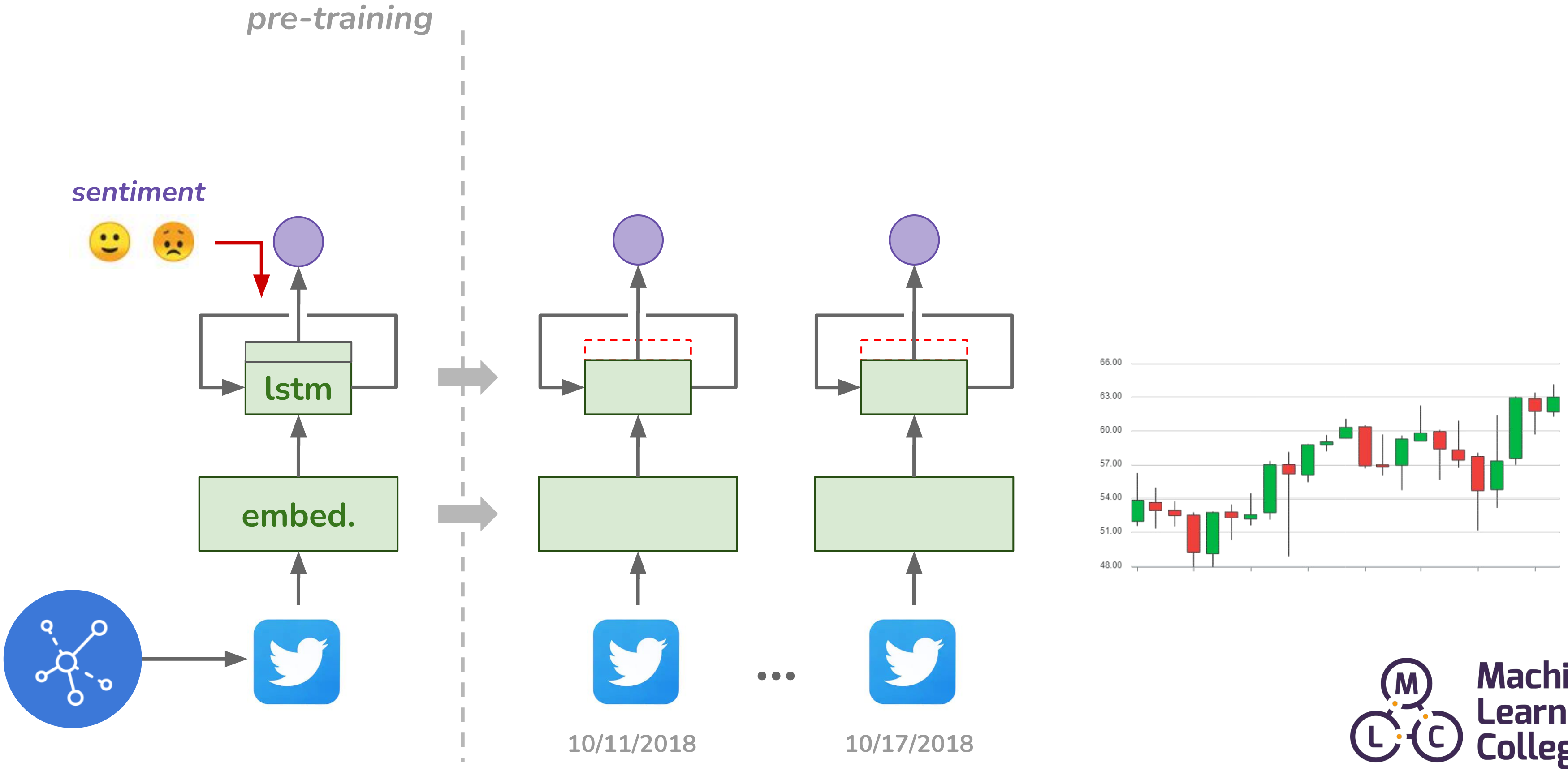


Pre-training with additional data



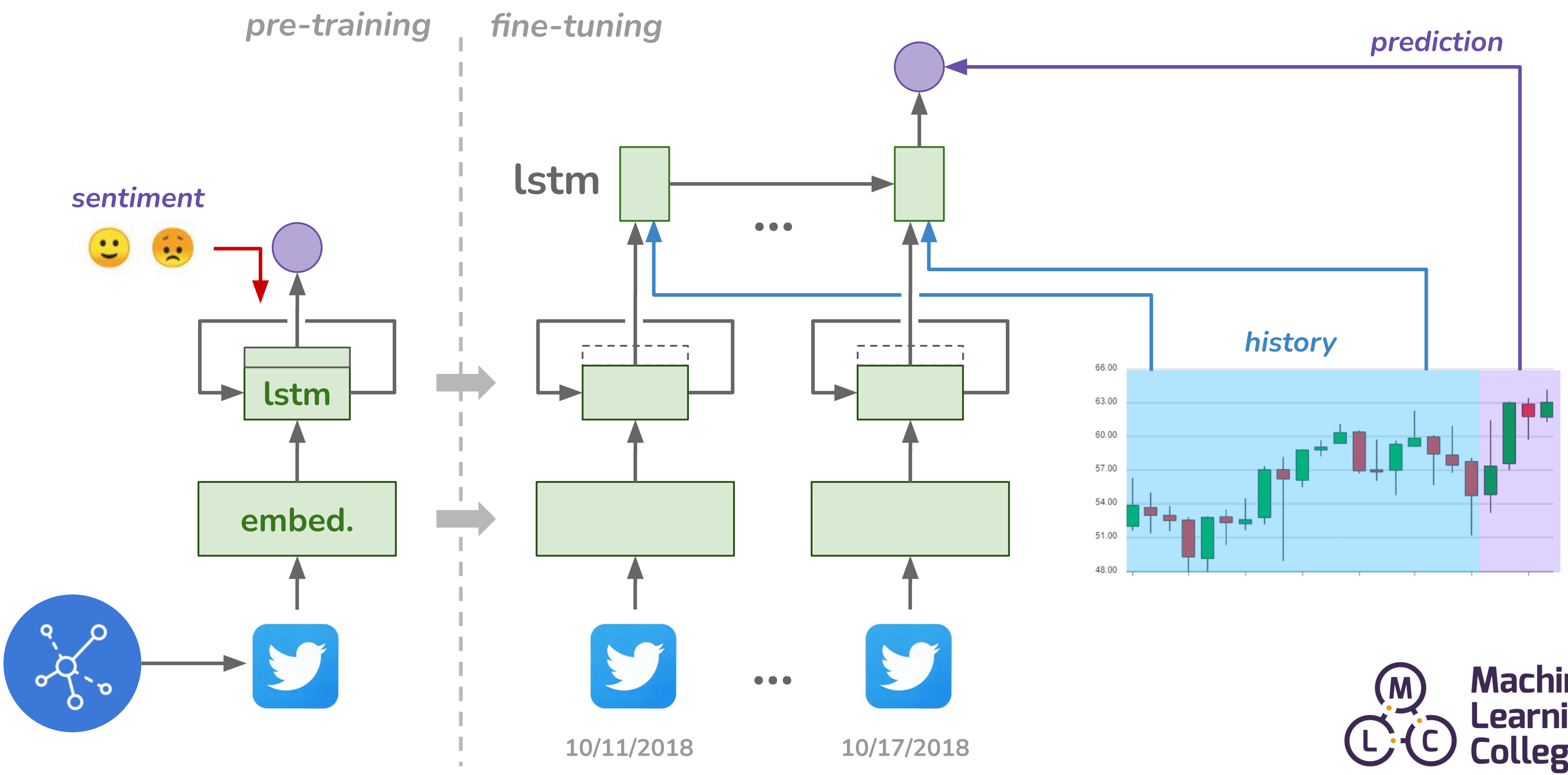
Machine Learning College

Transfering model & exposing feature layer



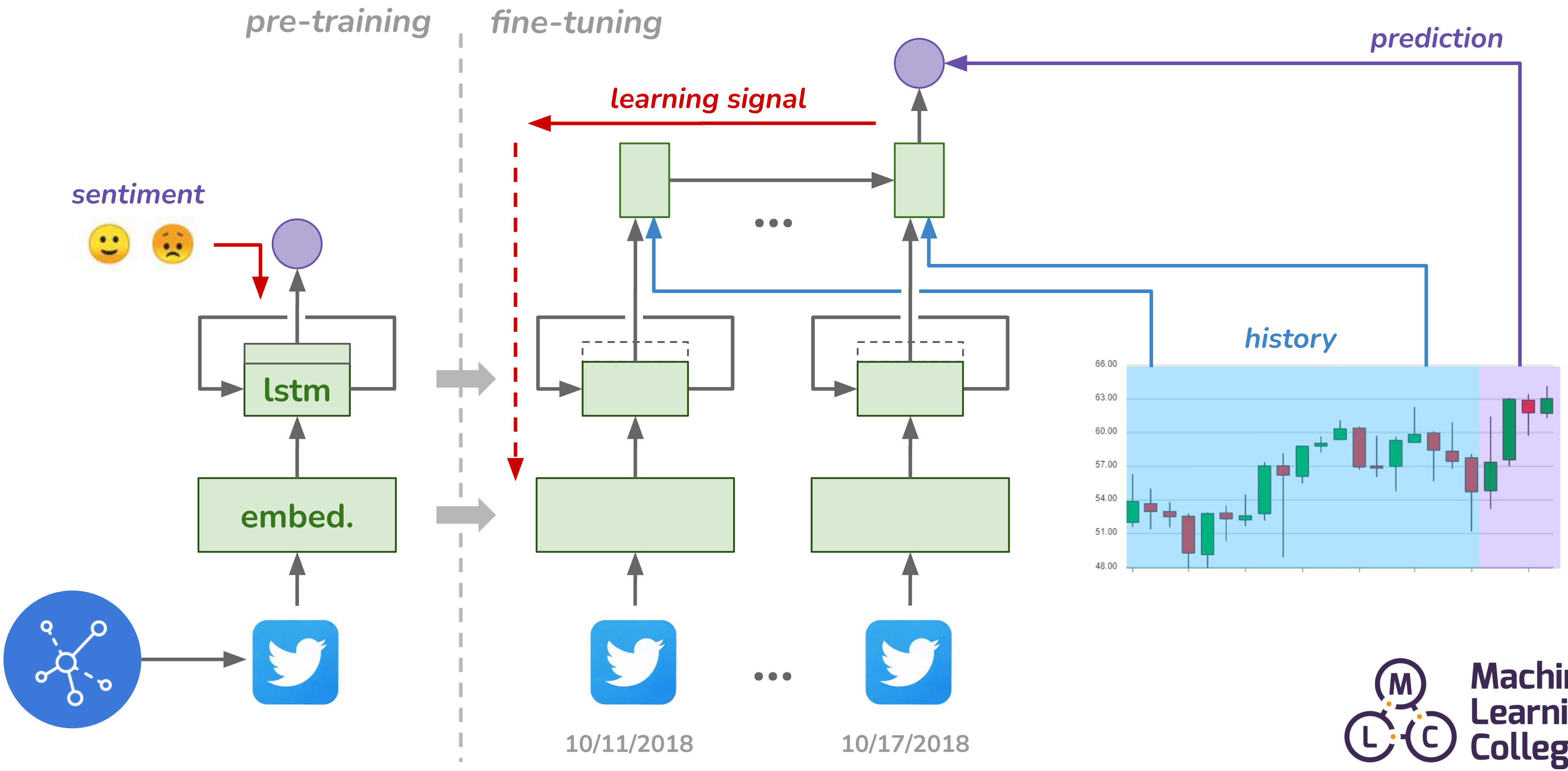
Machine Learning College

Fine-tuning with time series target data



Machine Learning College

Fine-tuning with time series target data



Machine Learning College