Air Quality Forecasting Using Deep Learning Techniques

Alican MERTAN
Istanbul Technical University
mertana@itu.edu.tr

Alper Unal
Istanbul Technical University
aunal@itu.edu.tr

Overview

Input

Previous PM_{10} measurements

→

Neural Network Model

→

Output

Future PM_{10} concentration

Overview

Dataset

Approach

Experiments

Results

Conclusion

Dataset

Hourly PM_{10} measurements of Adana

Dataset

Hourly PM_{10} measurements of Adana

Dataset

Hourly PM_{10} measurements of Adana from 2009 to 2018

- Training: 2009-2016
- Validation: 2017
- Testing: 2018

Dataset

Hourly PM_{10} measurements of Adana from 2009 to 2018

- Training: 2009-2016
- Validation: 2017
- Testing: 2018

Features

- PM_{10} concentration
- Date
- Hour

Dataset

Hourly PM_{10} measurements of Adana
from 2009 to 2018

- Training: 2009-2016
- Validation: 2017
- Testing: 2018

Features

- PM_{10} concentration (float)
- Date (categorical)
- Hour (categorical)

Dataset

Hourly PM_{10} measurements of Adana
from 2009 to 2018

- Training: 2009-2016
- Validation: 2017
- Testing: 2018

Features

- PM_{10} concentration (float)
- Date (categorical)
- Hour (categorical)

one-hot encoding ex: February [0 1 0 0 0 0 0 0 0 0 0 0]
float ex: February 0.09

Dataset

Hourly PM_{10} measurements of Adana
from 2009 to 2018

- Training: 2009-2016
- Validation: 2017
- Testing: 2018

Features

- PM_{10} concentration (float)
- Date (categorical)
- Hour (categorical)

one-hot encoding ex: February [0 1 0 0 0 0 0 0 0 0 0 0]
float ex: February 0.09

Problems

- Missing data (due to failure at the station)

Dataset

Hourly PM_{10} measurements of Adana
from 2009 to 2018

Training: 2009-2016

Validation: 2017

Testing: 2018

Features

PM_{10} concentration (float)

Date (categorical)

Hour (categorical)

one-hot encoding ex: February [0 1 0 0 0 0 0 0 0 0 0 0]

float ex: February 0.09

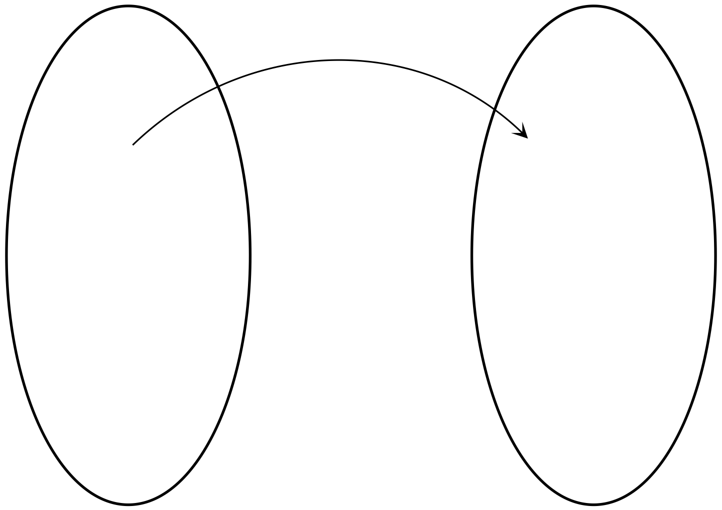
Problems

Missing data (due to failure at the station)

Solution: linear interpolation (only for training data and up to two consecutive hours)

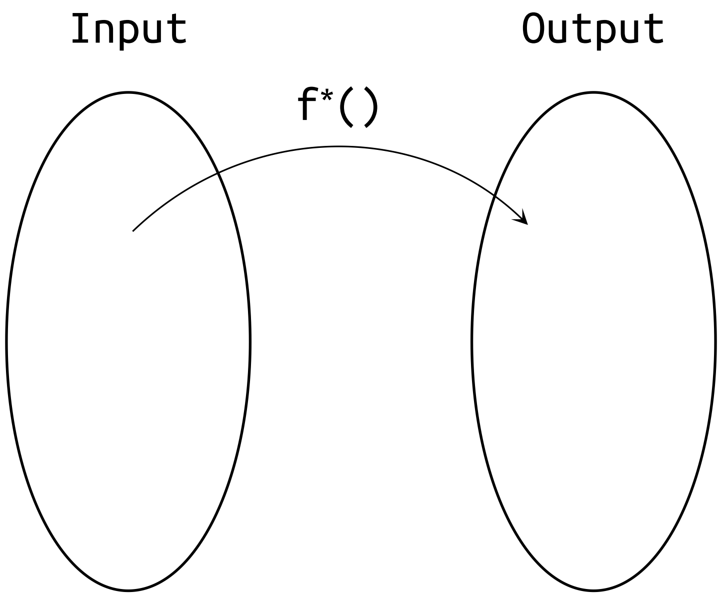
Basics

Input Output



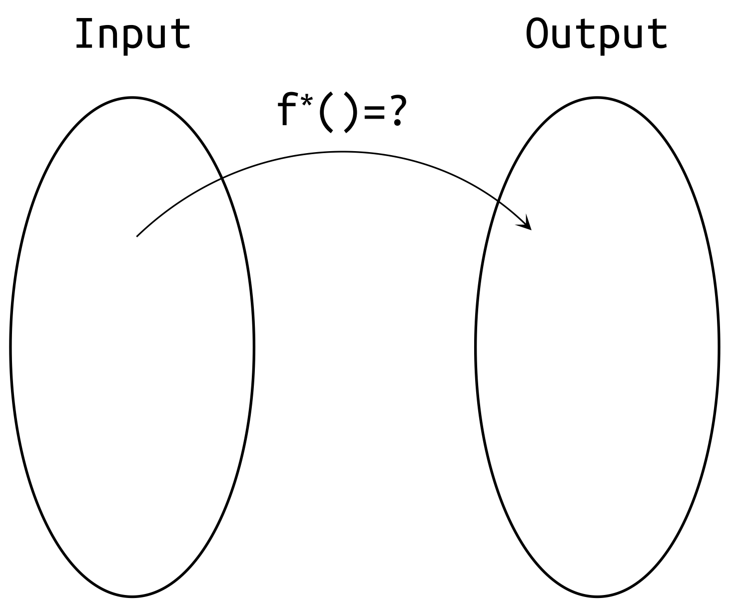
Approach

Basics



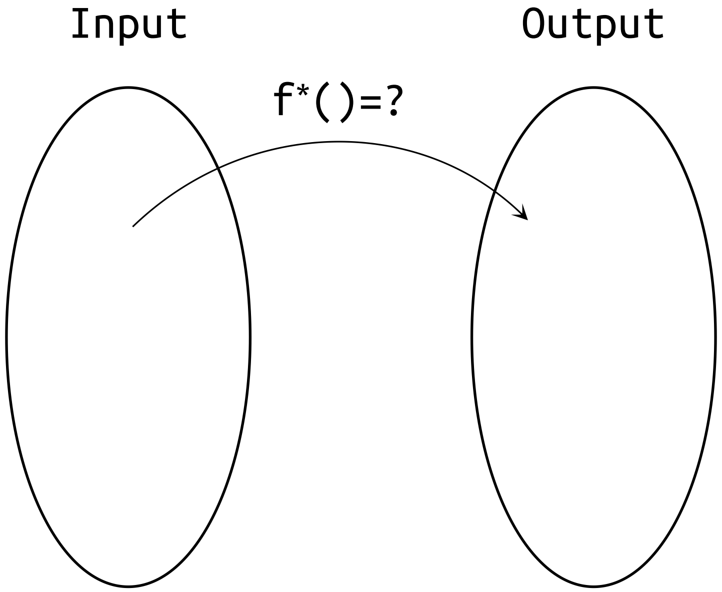
Approach

Basics



Approach

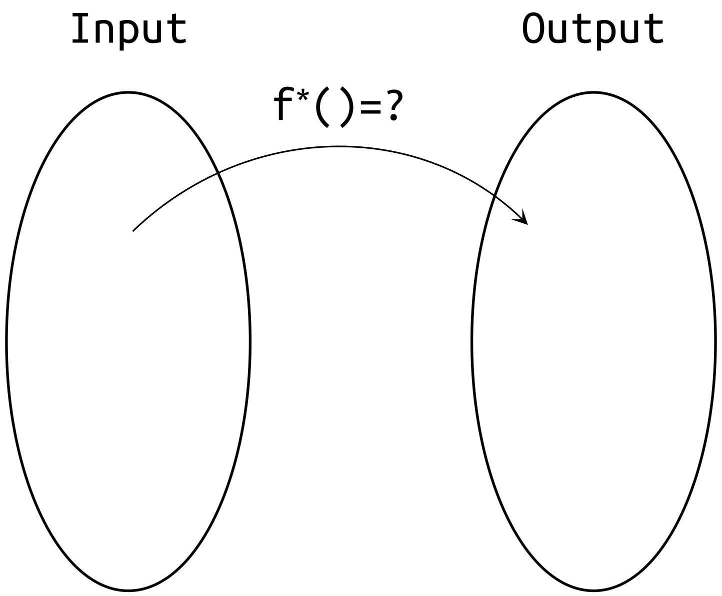
Basics



Approximate f*() with f(;w), and learn w from data

Approach

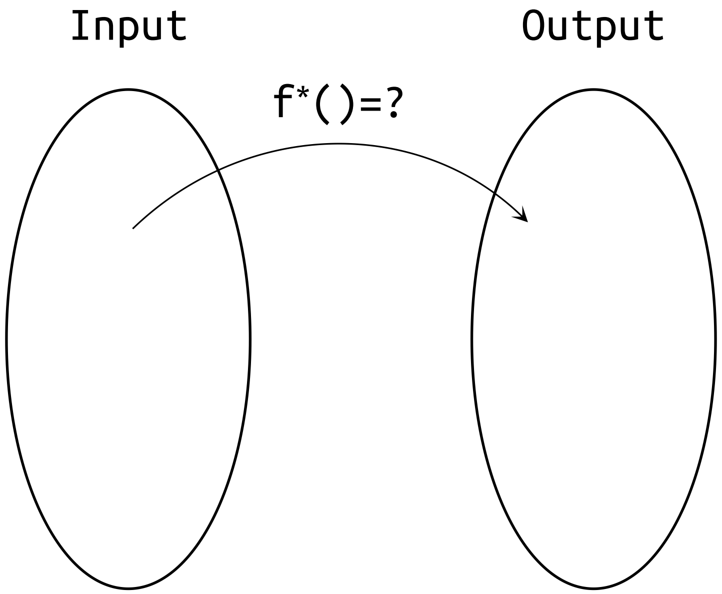
Basics



Approximate $f^*()$ with $f(;w)$, and learn w from data

$$w = \underset{w}{\operatorname{argmin}} \sum \left(f^*(input) - f(input;w) \right)^2$$

Basics



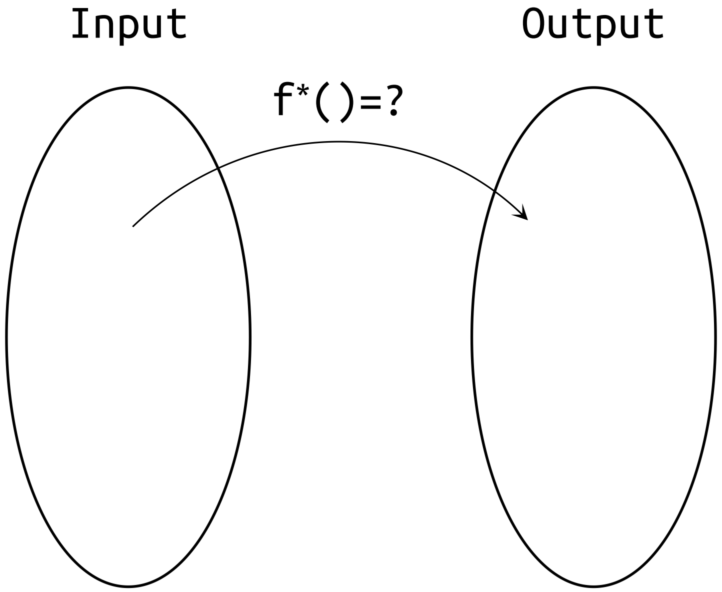
Approximate $f^*()$ with $f(;w)$, and learn w from data

$$w = \underset{w}{\operatorname{argmin}} \sum \left(f^*(input) - f(input;w) \right)^2$$

$f()$ is modeled with a neural network

Approach

Basics



Approximate $f^*()$ with $f(;w)$, and learn w from data

$$w = \underset{w}{\operatorname{argmin}} \sum \left(f^*(input) - f(input;w) \right)^2$$

f() is modeled with a neural network
Important to choose right architecture

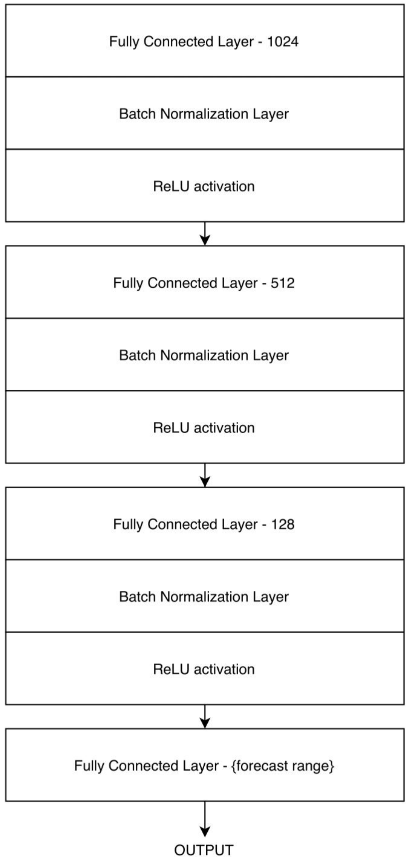
Approach

Experimented architectures

Fully connected   Convolutional   RNN   LSTM   seq2seq

Experimented architectures

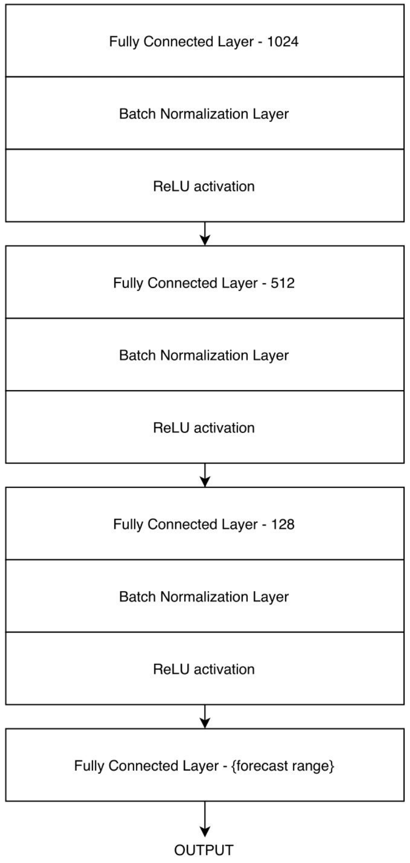
Fully connected Convolutional RNN LSTM seq2seq

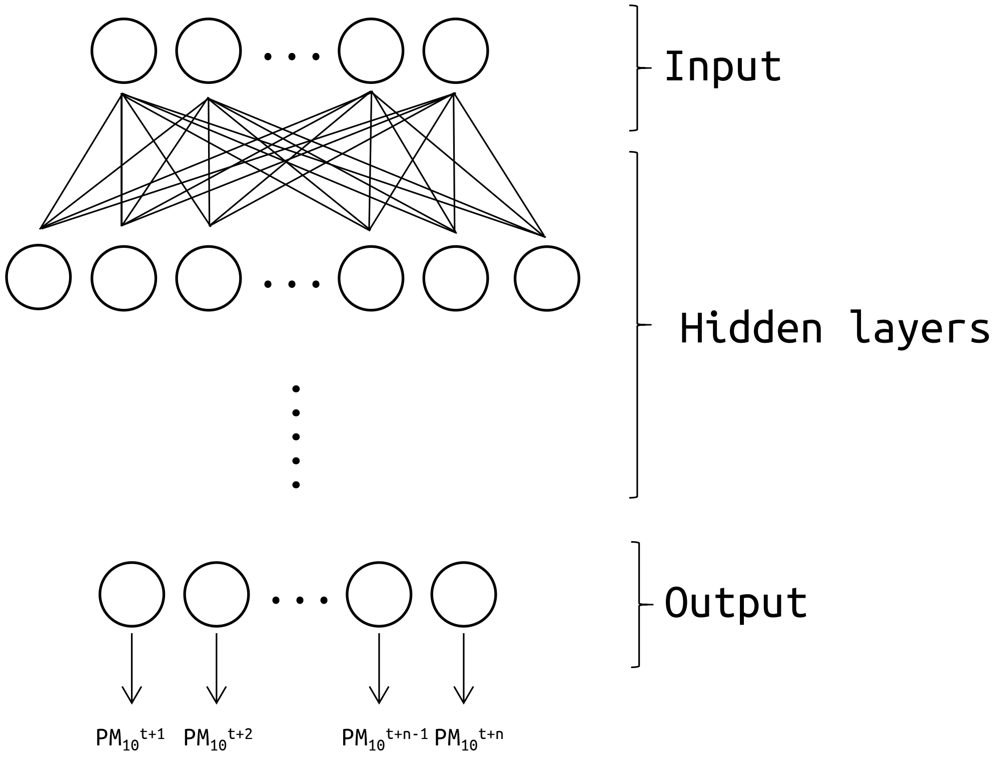


Approach

Experimented architectures

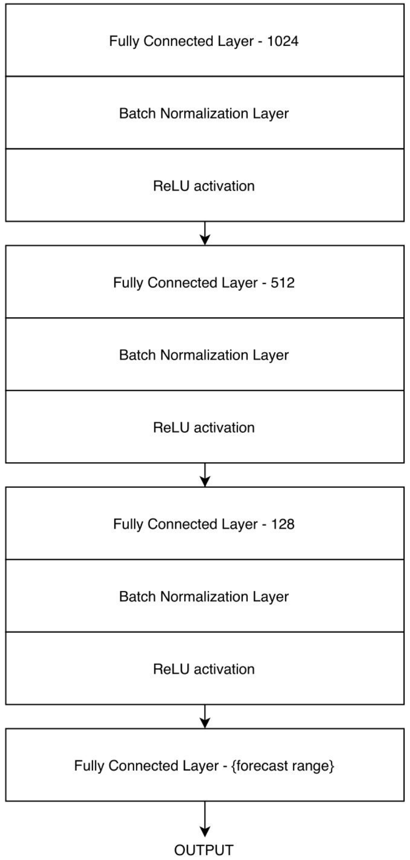
Fully connected Convolutional RNN LSTM seq2seq

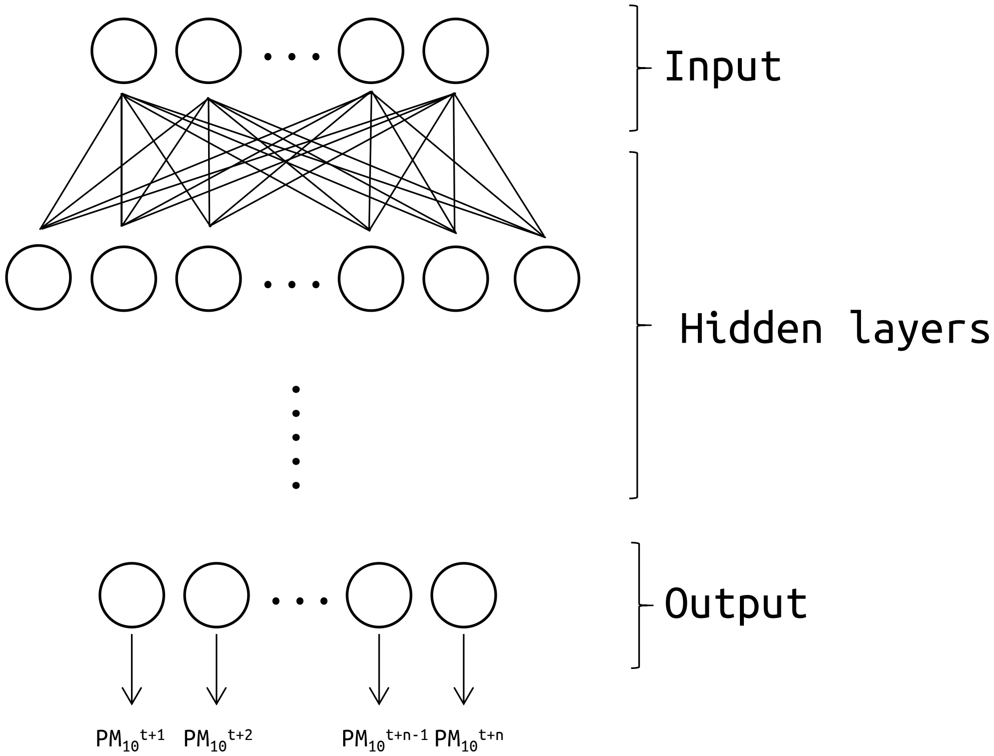




Experimented architectures

Fully connected Convolutional RNN LSTM seq2seq



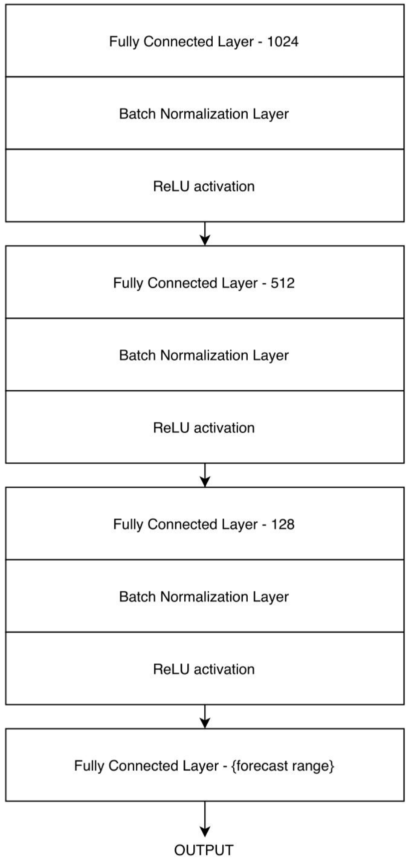


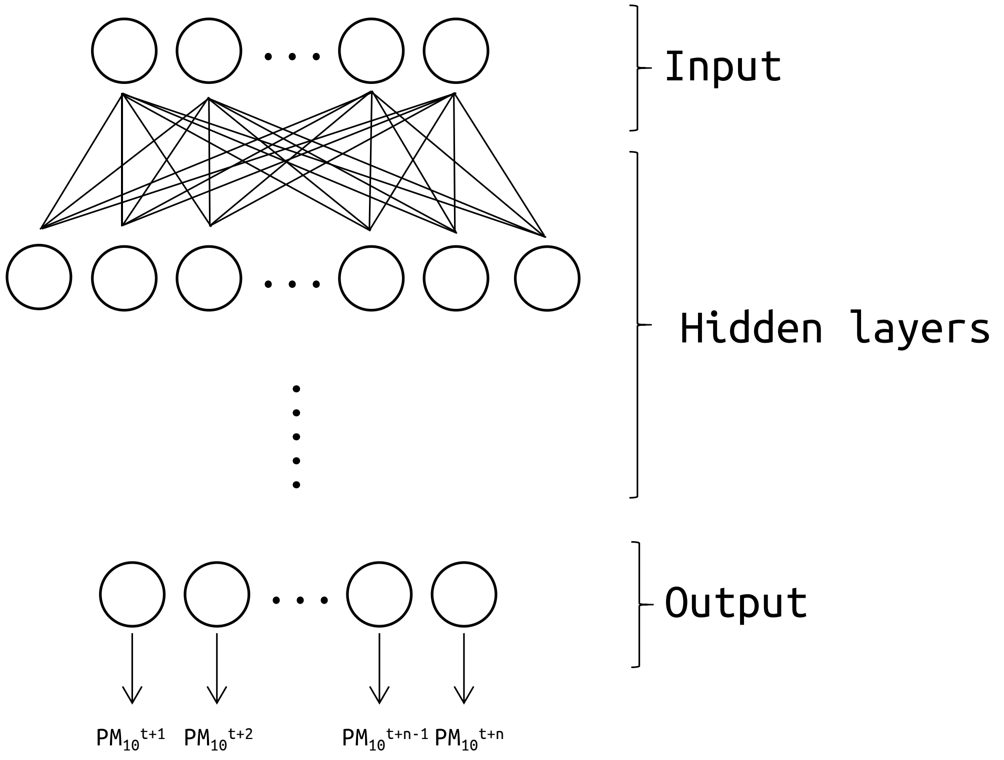
Most basic model

Approach

Experimented architectures

Fully connected | Convolutional | RNN | LSTM | seq2seq



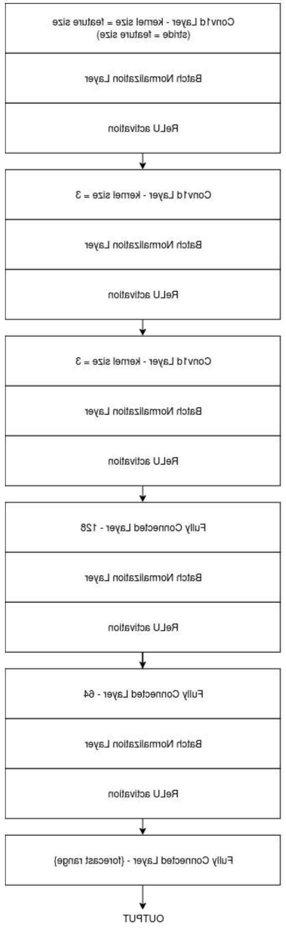


Most basic model

Heuristically chosen

Experimented architectures

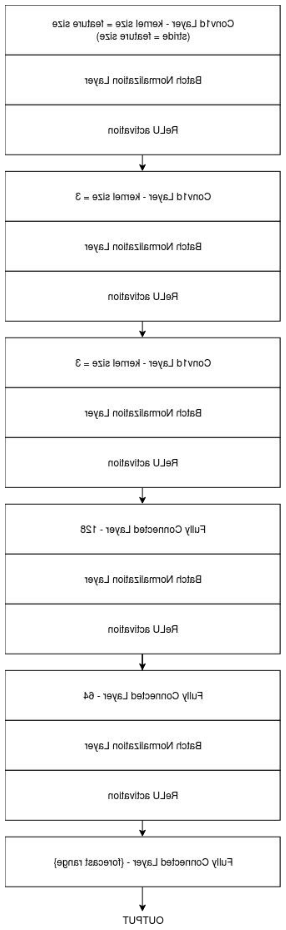
Fully connected | **Convolutional** | RNN | LSTM | seq2seq

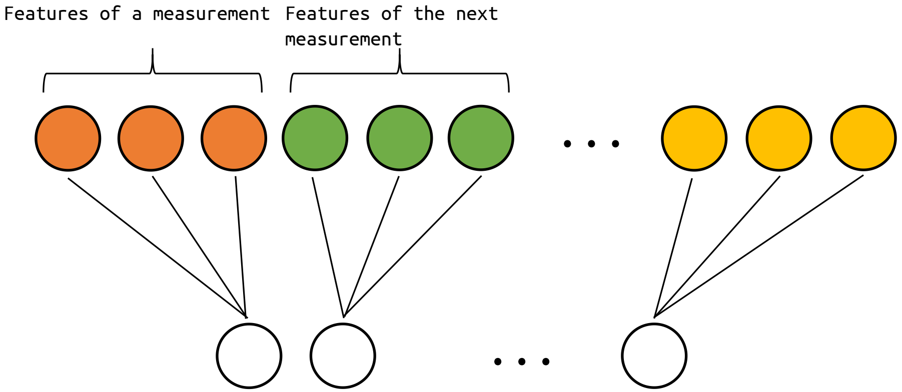


Approach

Experimented architectures

Fully connected | **Convolutional** | RNN | LSTM | seq2seq



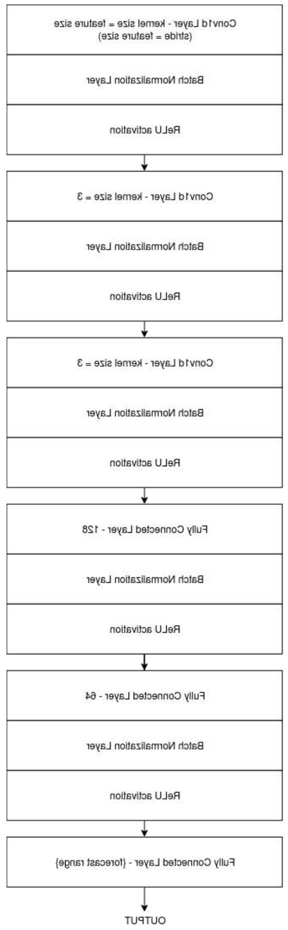


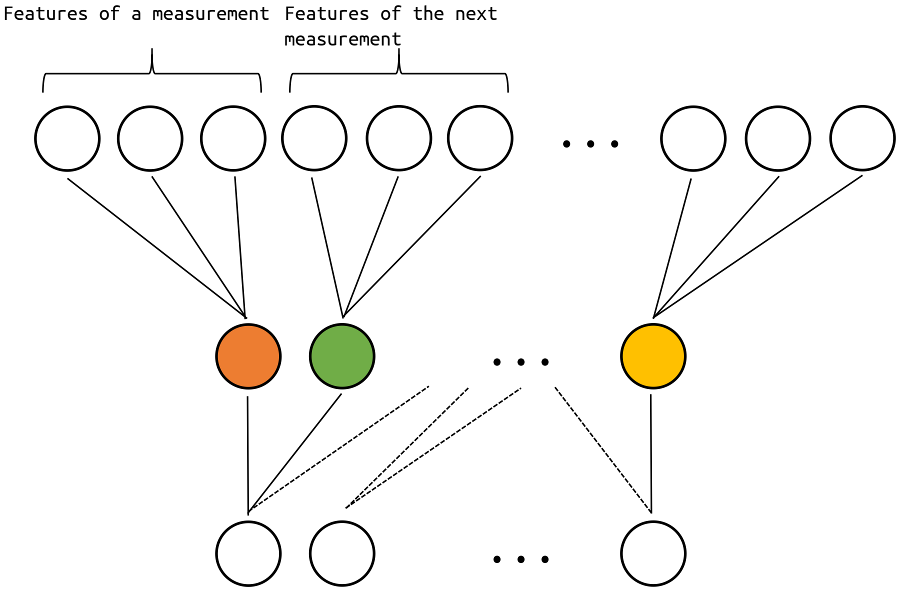
First layer summarizes each measurement

Approach

Experimented architectures

Fully connected · **Convolutional** · RNN · LSTM · seq2seq





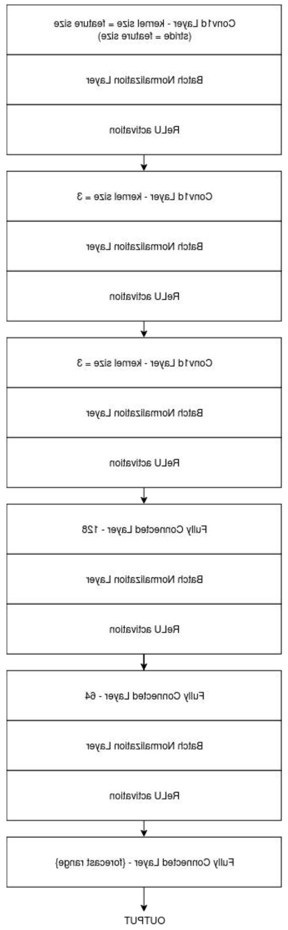
First layer summarizes each measurement

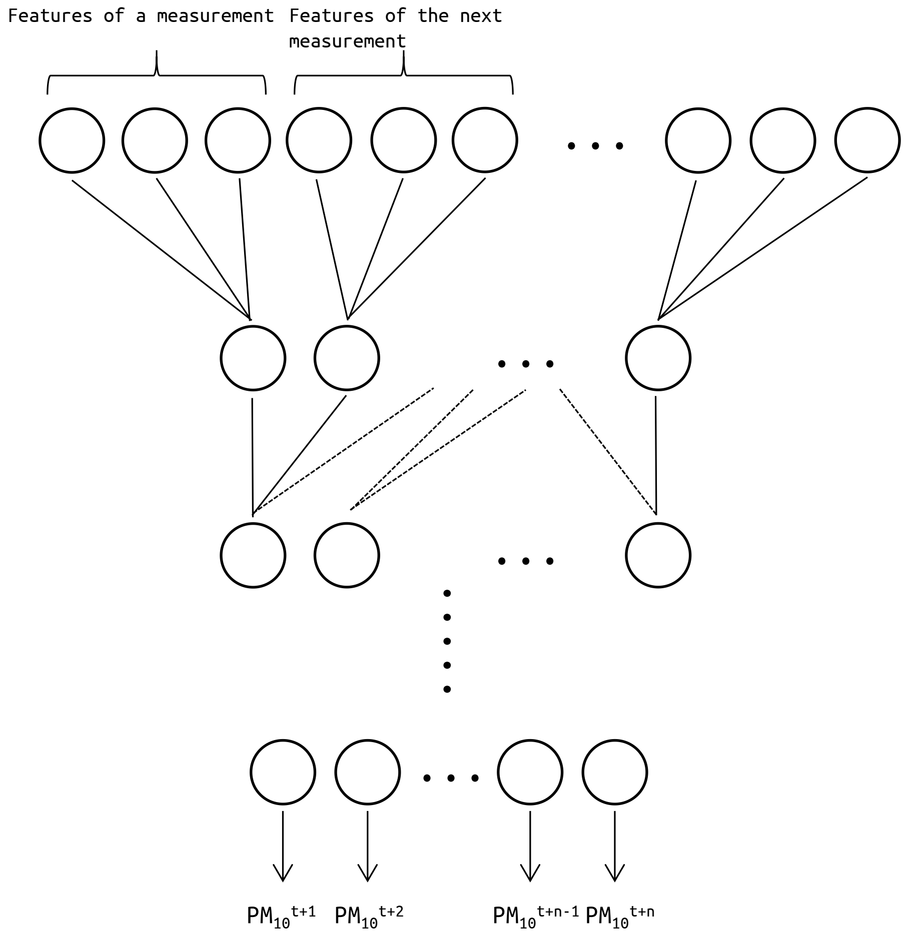
Next layers fuse information of temporally neighboring measurements

Approach

Experimented architectures

Fully connected | **Convolutional** | RNN | LSTM | seq2seq





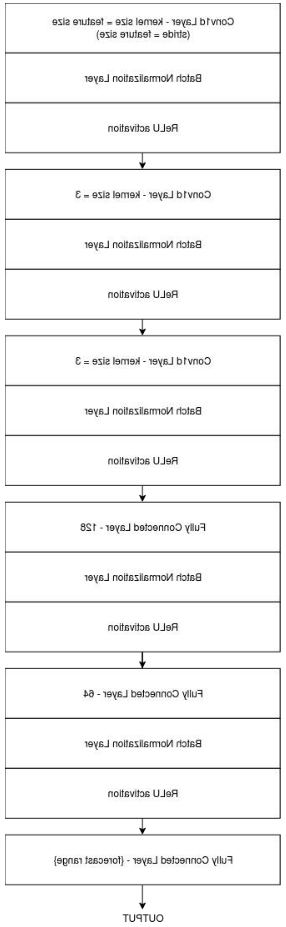
First layer summarizes each measurement

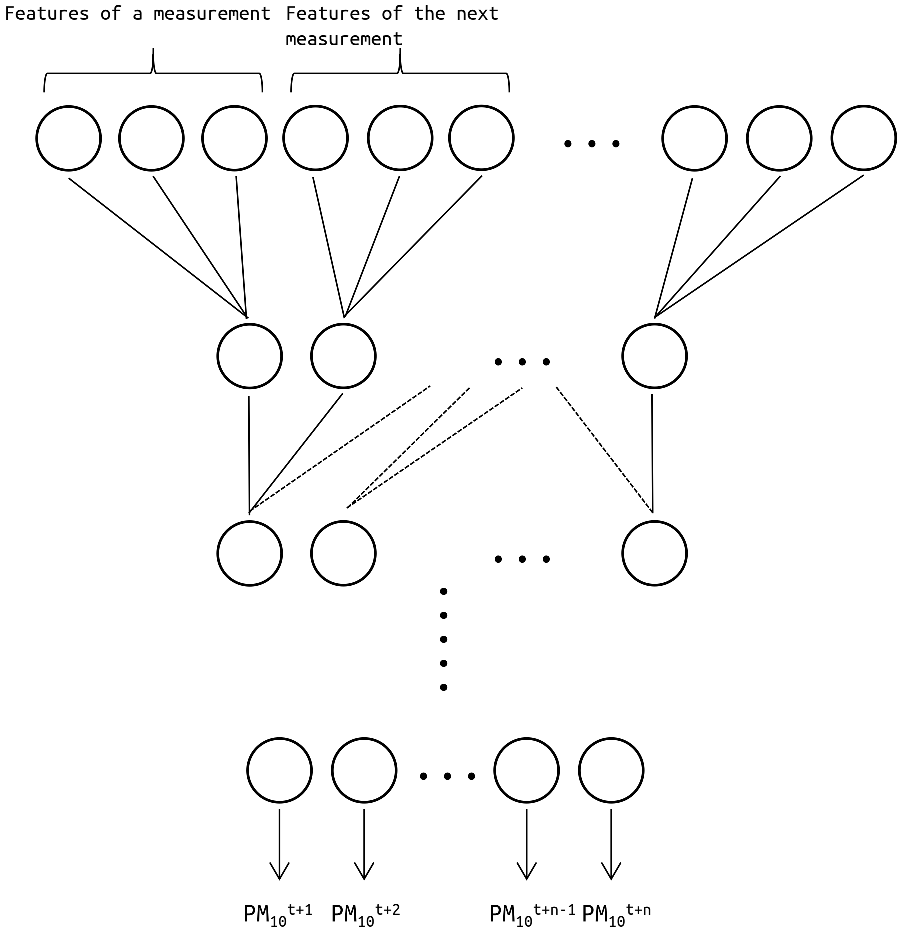
Next layers fuse information of temporally neighboring measurements

Approach

Experimented architectures

Fully connected   **Convolutional**   RNN   LSTM   seq2seq





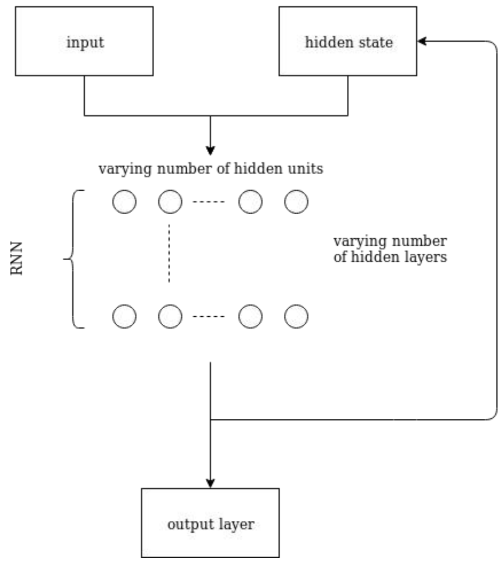
First layer summarizes each measurement

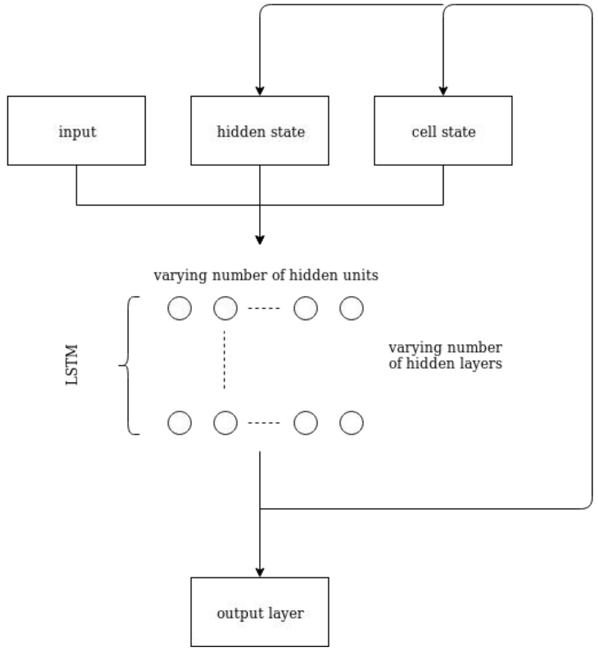
Next layers fuse information of temporally neighboring measurements

Heuristically chosen

Experimented architectures

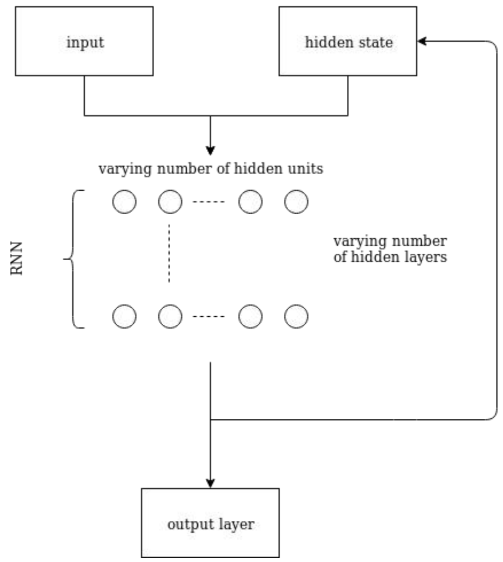
Fully connected | Convolutional | RNN | LSTM | seq2seq

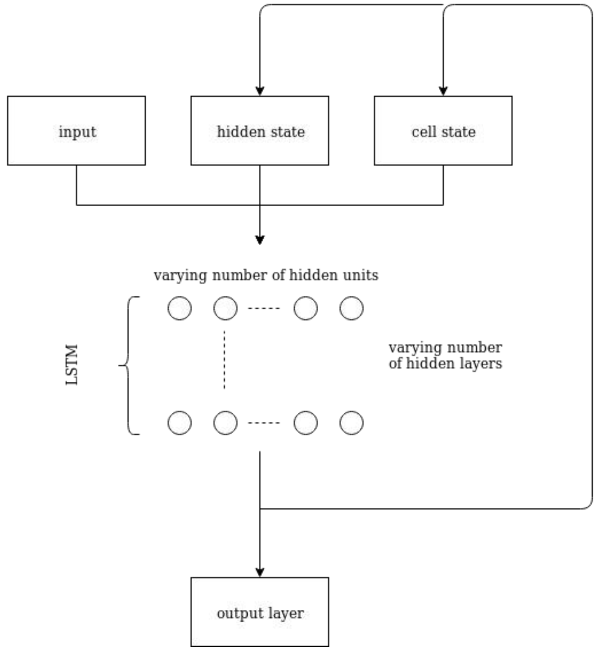




Experimented architectures

Fully connected Convolutional RNN LSTM seq2seq

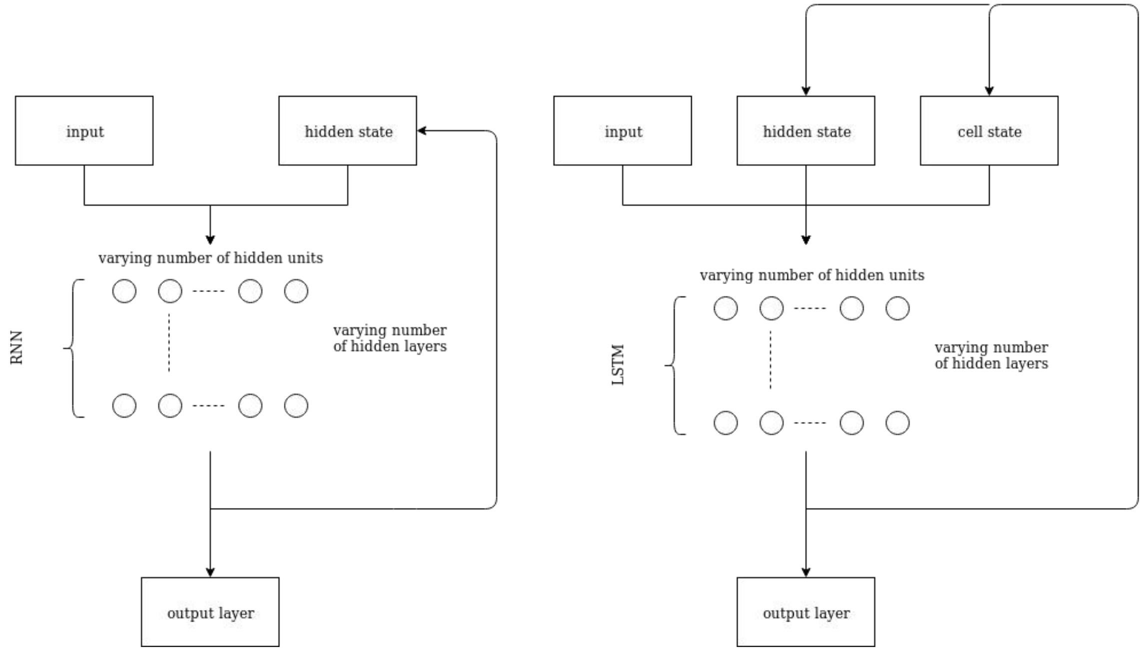




Specifically designed for sequential data

Experimented architectures

Fully connected | Convolutional | RNN | LSTM | seq2seq



Specifically designed for sequential data

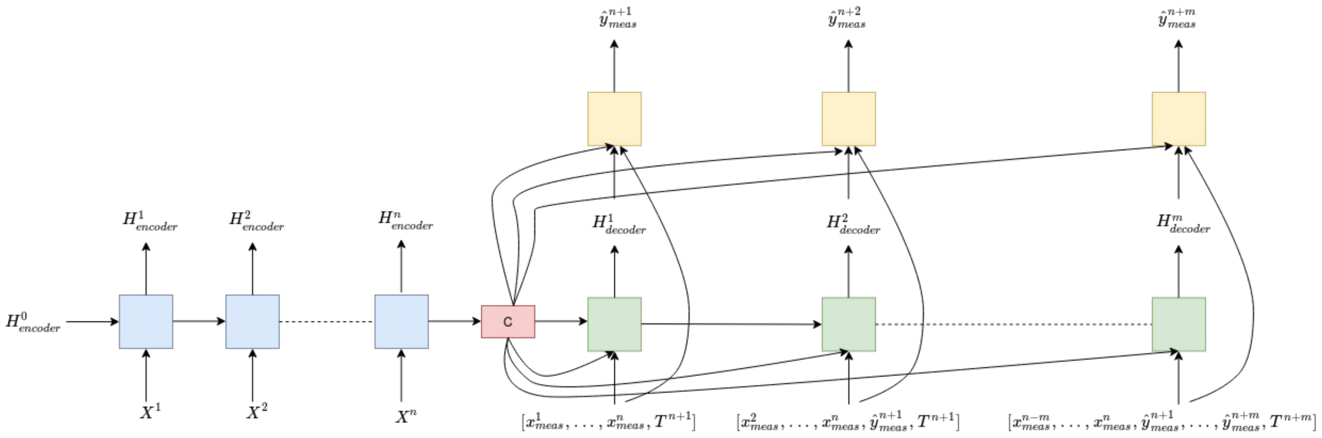
Experimented architectural choices

- Number of hidden layers
- Number of hidden units in each layer

Experimented architectures

Fully connected Convolutional RNN LSTM **seq2seq**

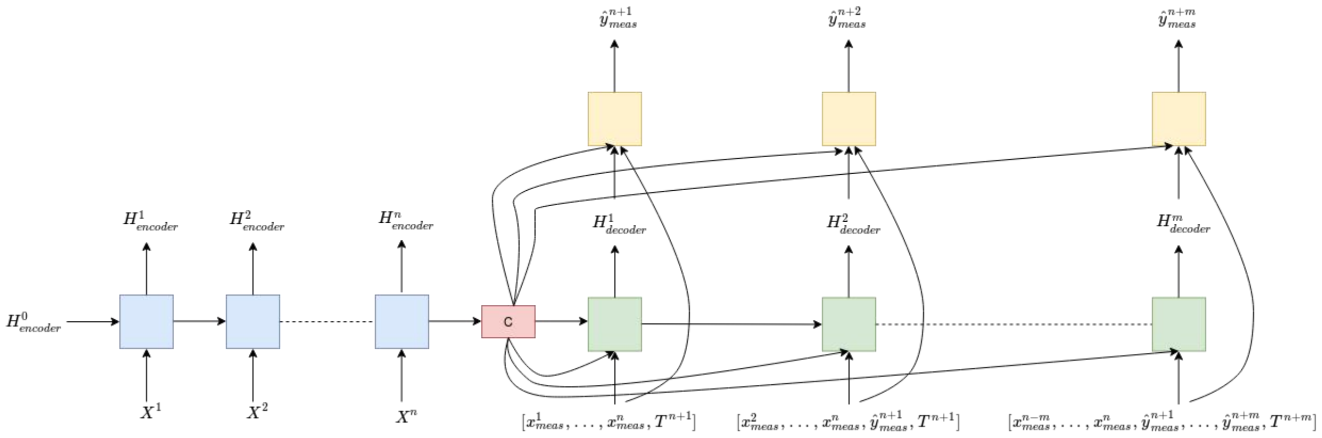
Adapted from Marino et al.[1]



[1] D. L. Marino, K. Amarasinghe and M. Manic, "Building energy load forecasting using Deep Neural Networks," IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, Florence, 2016, pp. 7046-7051, doi: 10.1109/IECON.2016.7793413.

Experimented architectures

Fully connected Convolutional RNN LSTM **seq2seq**



Adapted from Marino et al.[1]

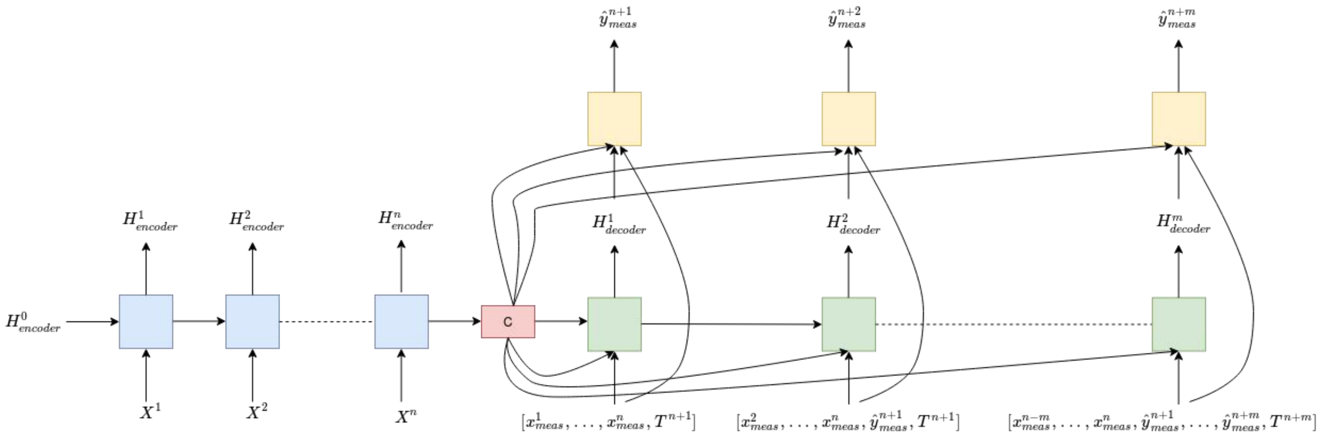
Most complex model

[1] D. L. Marino, K. Amarasinghe and M. Manic, "Building energy load forecasting using Deep Neural Networks," IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, Florence, 2016, pp. 7046-7051, doi: 10.1109/IECON.2016.7793413.

Approach

Experimented architectures

Fully connected | Convolutional | RNN | LSTM | seq2seq



Adapted from Marino et al.[1]

Most complex model

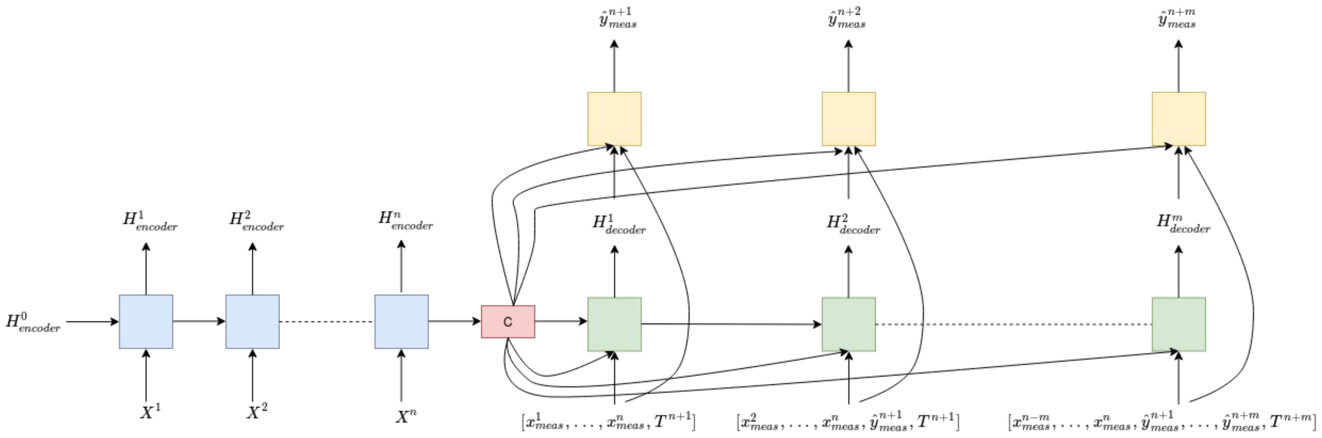
Consists of GRU units

[1] D. L. Marino, K. Amarasinghe and M. Manic, "Building energy load forecasting using Deep Neural Networks," IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, Florence, 2016, pp. 7046-7051, doi: 10.1109/IECON.2016.7793413.

Approach

Experimented architectures

Fully connected | Convolutional | RNN | LSTM | **seq2seq**



Adapted from Marino et al.[1]

Most complex model

Consists of GRU units

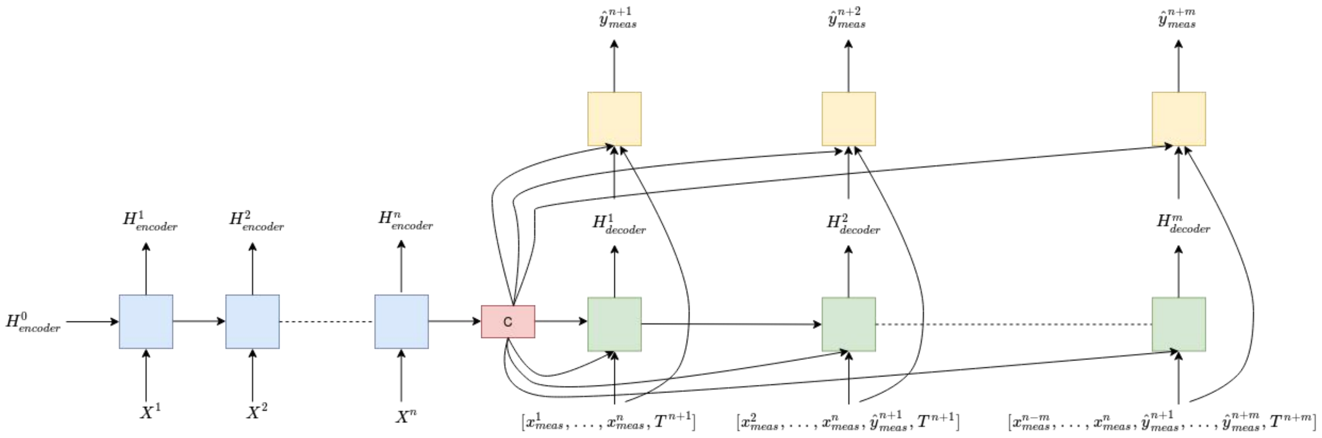
Encoder summarizes contextual information
Decoder outputs PM_{10} concentrations

[1] D. L. Marino, K. Amarasinghe and M. Manic, "Building energy load forecasting using Deep Neural Networks," IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, Florence, 2016, pp. 7046-7051, doi: 10.1109/IECON.2016.7793413.

Approach

Experimented architectures

Fully connected Convolutional RNN LSTM **seq2seq**



Adapted from Marino et al.[1]

Most complex model

Consists of GRU units

Encoder summarizes contextual information
Decoder outputs PM_{10} concentrations

Experimented architectural choices
- Number of hidden layers
- Number of hidden units in each layer

[1] D. L. Marino, K. Amarasinghe and M. Manic, "Building energy load forecasting using Deep Neural Networks," IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, Florence, 2016, pp. 7046-7051, doi: 10.1109/IECON.2016.7793413.

Experiments

Feature types

- One-hot
- Float

Experiments

- Feature types
  - One-hot
  - Float
- Architectures
  - Fully connected
  - Convolutional
  - RNN
  - LSTM
  - seq2seq

Experiments

- Feature types
  - One-hot
  - Float
- Architectures
  - Fully connected
  - Convolutional
  - RNN
  - LSTM
  - seq2seq
- Architectural hyperparameters
  - Number of hidden layers
  - Number of units in hidden layers

- Feature types
  - One-hot
  - Float
- Architectures
  - Fully connected
  - Convolutional
  - RNN
  - LSTM
  - seq2seq
- Architectural hyperparameters
  - Number of hidden layers
  - Number of units in hidden layers
- Forecast range
  - 1 (next hour)
  - 24 (next day)

Experiments

- Feature types
  - One-hot
  - Float
- Architectures
  - Fully connected
  - Convolutional
  - RNN
  - LSTM
  - seq2seq
- Architectural hyperparameters
  - Number of hidden layers
  - Number of units in hidden layers
- Forecast range
  - 1 (next hour)
  - 24 (next day)
- Look back
  - 1 week (168 measurement)

Experiments

Feature types
- One-hot
- Float

Architectures
- Fully connected
- Convolutional
- RNN
- LSTM
- seq2seq

Architectural hyperparameters
- Number of hidden layers
- Number of units in hidden layers

Forecast range
- 1 (next hour)
- 24 (next day)

Look back
- 1 week (168 measurement)

Performance Measure

$$\text{relL1} = \frac{|PM_{10}{}^{groundtruth} - PM_{10}{}^{prediction}|}{PM_{10}{}^{groundtruth}}$$

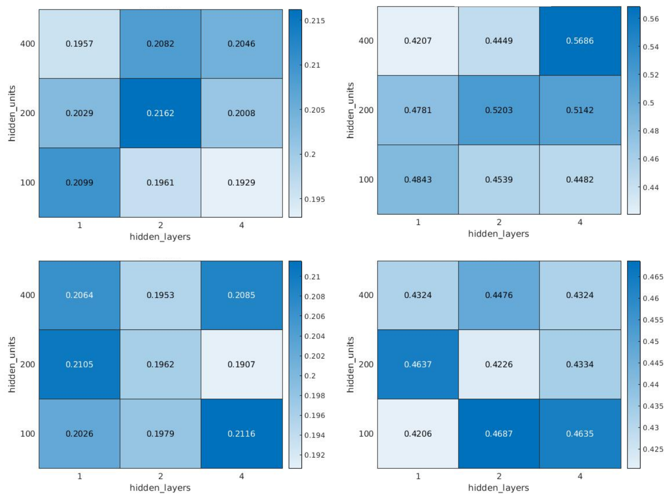
Results

RNN

forecast range 1 | forecast range 24

One-hot

Float

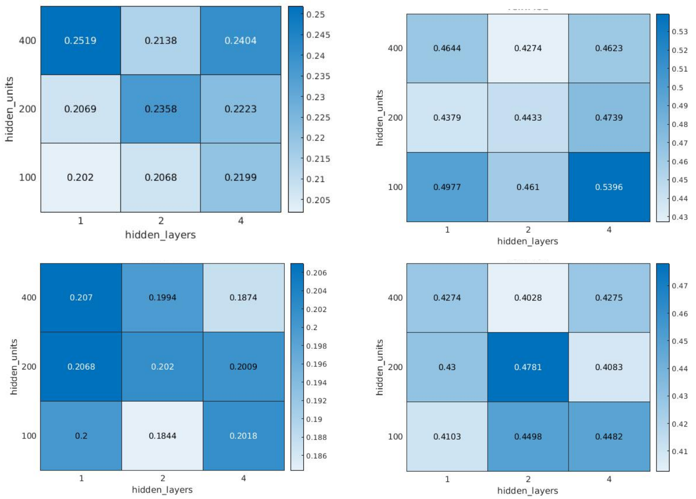


LSTM

forecast range 1 | forecast range 24

One-hot

Float

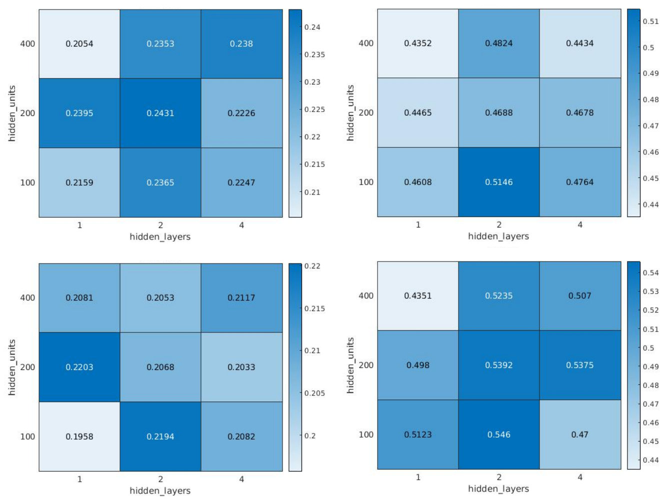


seq2seq

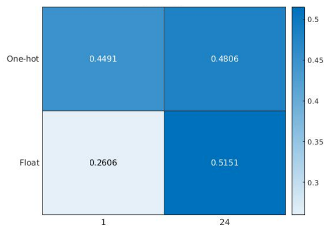
forecast range 1 | forecast range 24

One-hot

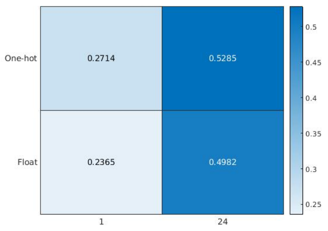
Float



Fcnet



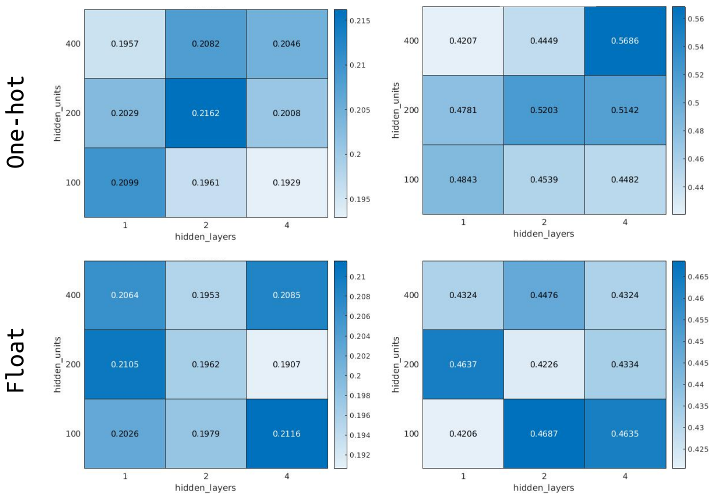
Convnet



Results

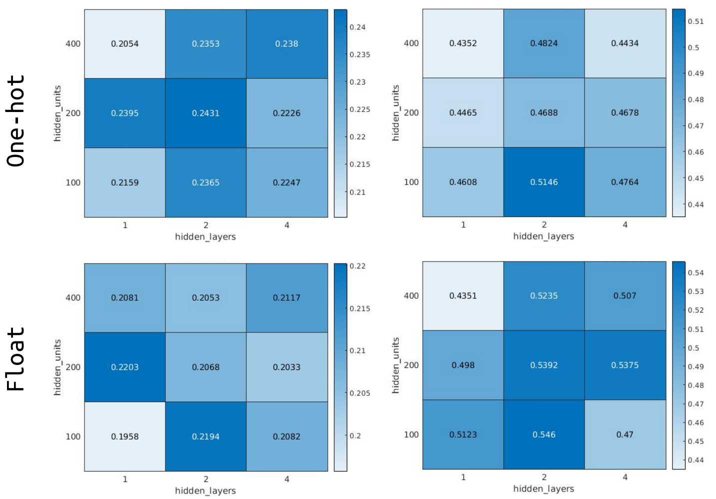
RNN

forecast range 1 forecast range 24



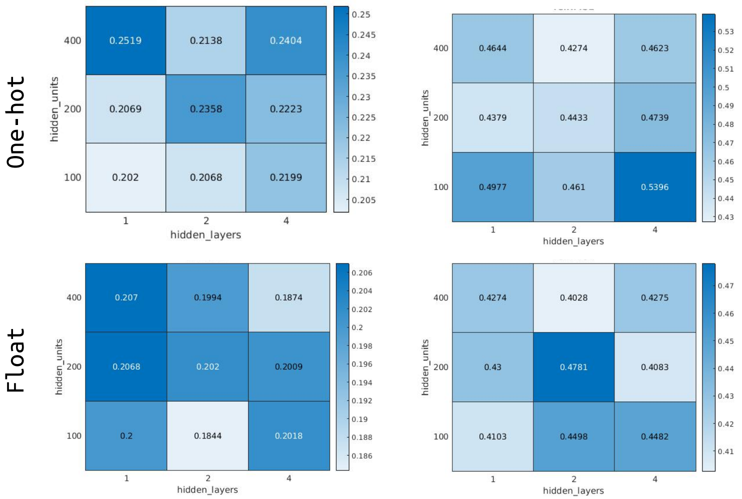
seq2seq

forecast range 1 forecast range 24

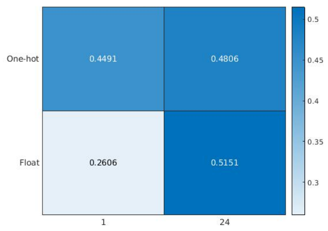


LSTM

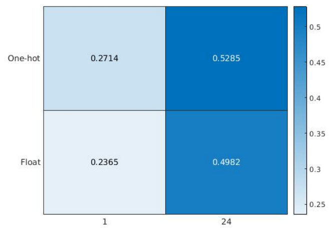
forecast range 1 forecast range 24



Fcnet



Convnet



Takeaways

Float representation is better

No general trend in model size

Results

Architecture	Forecast range 1	Architecture	Forecast range 24
Baseline	0.2032	Baseline	0.6021

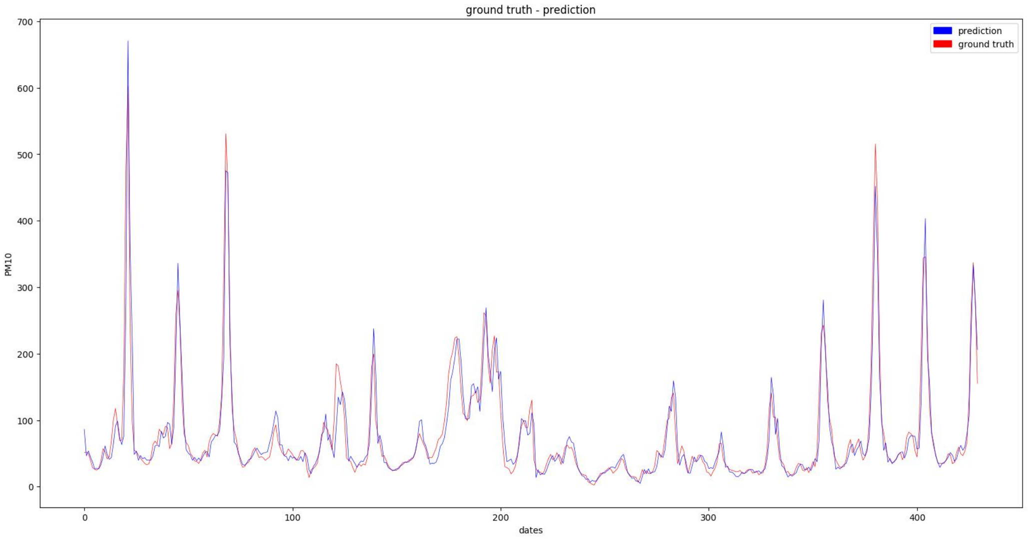
Results

Architecture	Forecast range 1	Architecture	Forecast range 24
Baseline	0.2032	Baseline	0.6021
Fcnet_float	0.2606	Fcnet_onehot	0.4806
Convnet_float	0.2365	Convnet_float	0.4982
RNN_float_4_100	<u>0.1929</u>	RNN_float_1_100	<u>0.4206</u>
LSTM_float_2_100	**0.1844**	LSTM_float_2_400	**0.4028**
seq2seq_float_1_100	0.1958	seq2seq_float_1_400	0.4351

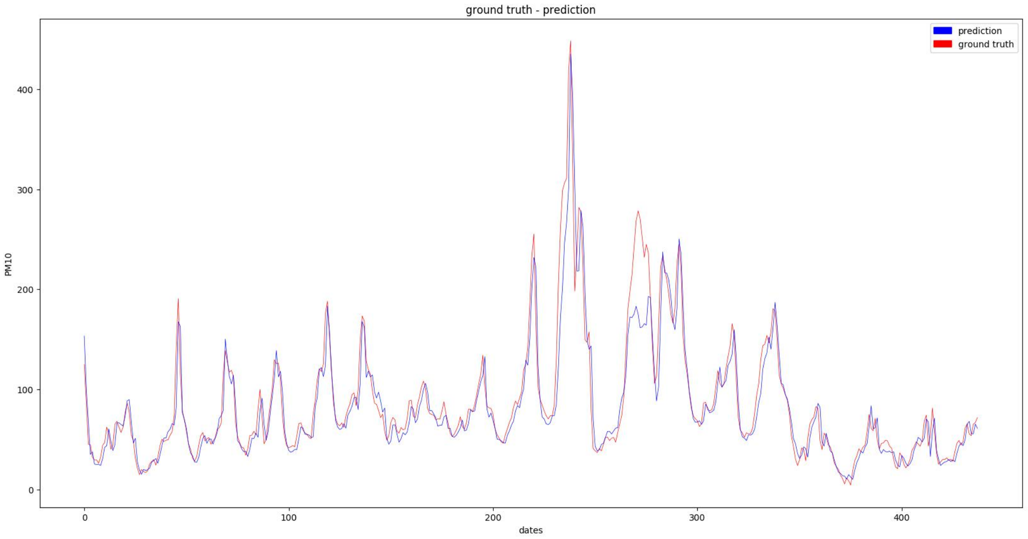
Results

Example predictions from test set

LSTM_float_2_100 forecast range 1



Jan 2018

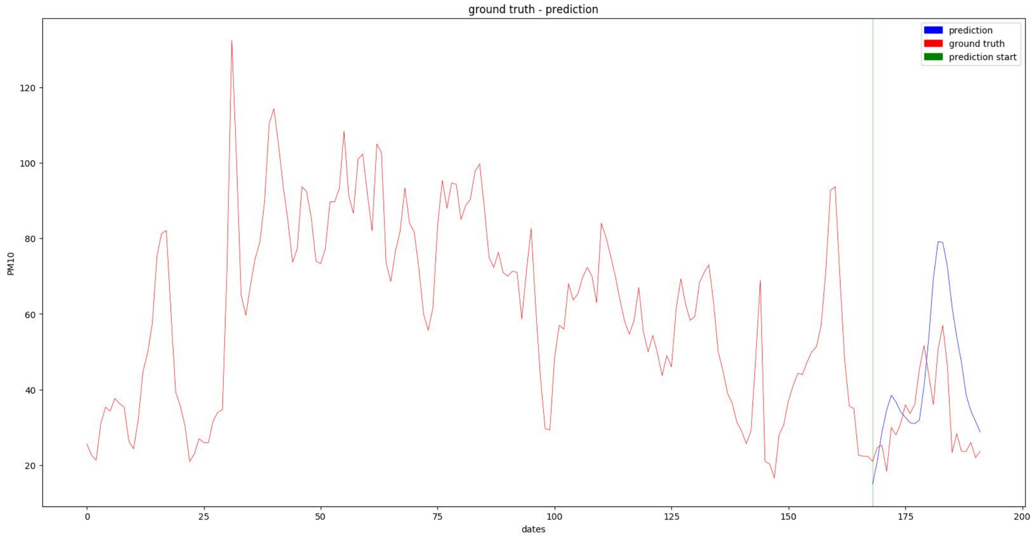


March 2018

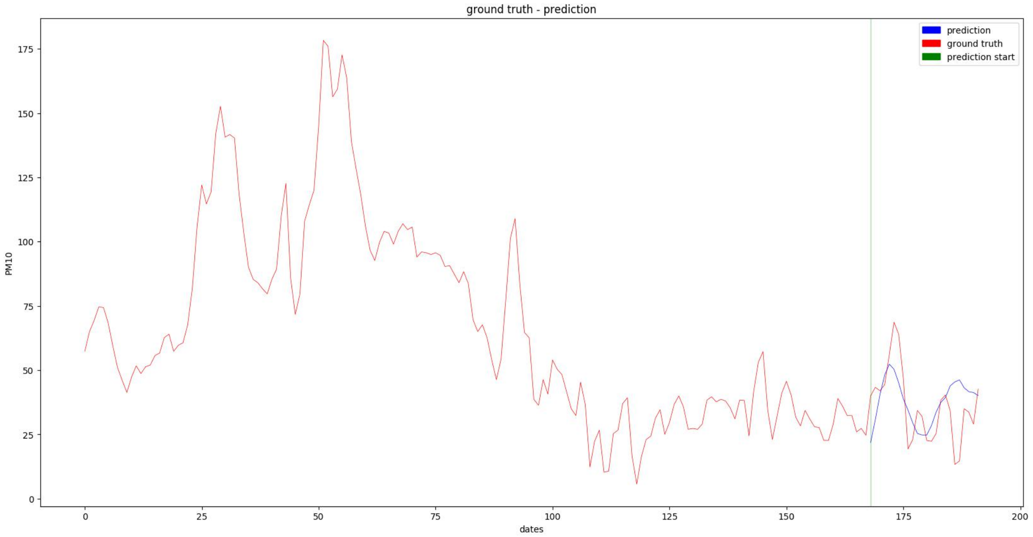
Results

Example predictions from test set

LSTM_float_2_400 forecast range 24



input: 02.05.2018-09.05.2018 output: 10.05.2018



input: 10.05.2018-17.05.2018 output: 18.05.2018

Conclusion

Experimented with different

- representations
- architectures
- hyperparameters
- forecast ranges

Conclusion

Experimented with different
- representations
- architectures
- hyperparameters
- forecast ranges

RNN and its variants outperform simpler models

Conclusion

Experimented with different

- representations
- architectures
- hyperparameters
- forecast ranges

RNN and its variants outperform simpler models

Performance drops as the forecast range increases

Additional information can be utilized

THANK YOU FOR LISTENING!