Karlsruhe Institute of Technology

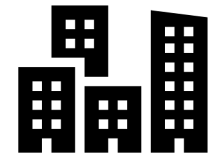
Deep Learning for Energy Time Series

Kai Schmieder

ENERGY INFORMATICS SEMINAR
INSTITUTE FOR AUTOMATION AND APPLIED INFORMATICS (IAI)

Motivation

- Buildings are major energy consumer worldwide (Pérez-Lombard et al. 2008)



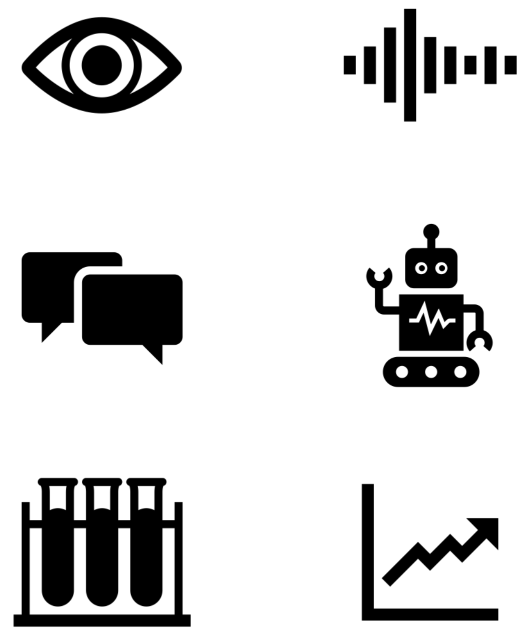
- Minimize the energy wastage and making power generation and distribution more efficient by intelligent control decisions (Marino et al. 2016, p. 7046)

- Load forecasting crucial for mitigating uncertainties of the future

Motivation

- Deep Learning proved to be useful in manifold fields
  - Computer vision
  - Speech and audio processing
  - Natural language processing
  - Robotics
  - Bioinformatics and chemistry
  - Finance
  - …

(Goodfellow et al. 2016, p.8f)



Agenda

- Foundation
- Long Short-Term Memory for Energy Time Series
- Evaluation
- Discussion
- Conclusion

FOUNDATION

Energy Time Series

- Aggregate level and building level forecasting categorized by
  - Short-term → one hour to one week
  - Medium-term → one week to one year
  - Long-term → ranges longer than one year

  (Mocanu et al. 2016, p. 91)

- Two main approaches to forecast Energy Time Series
  - Statistical and Machine Learning based models
  - Physical Principles based models

  (Mocanu et al. 2016, p. 91; Marino et al. 2016, p. 7046)

- → here: Short-term & Statistical and Machine Learning based models

Deep Learning

- Central goal of Machine Learning
  - Learn useful representations of input data
  - That get us closer to expected output

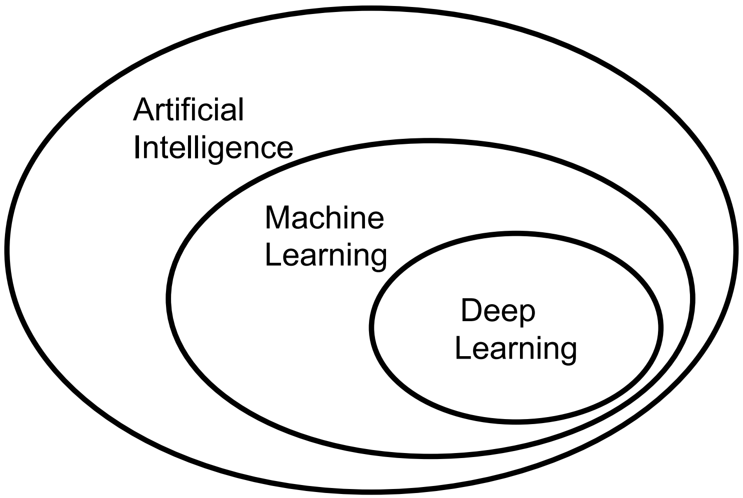


Figure 1: Own representation based on Chollet, 2018, p.4

- Deep Learning
  - Specific subfield of Machine Learning
  - Idea of successive layers of representations
  - Depth: how many layers contribute to a model of the data
  - Layered representations (almost always) learned via neural networks

(Chollet 2018, p. 6ff)

Overview of relevant Deep Architectures

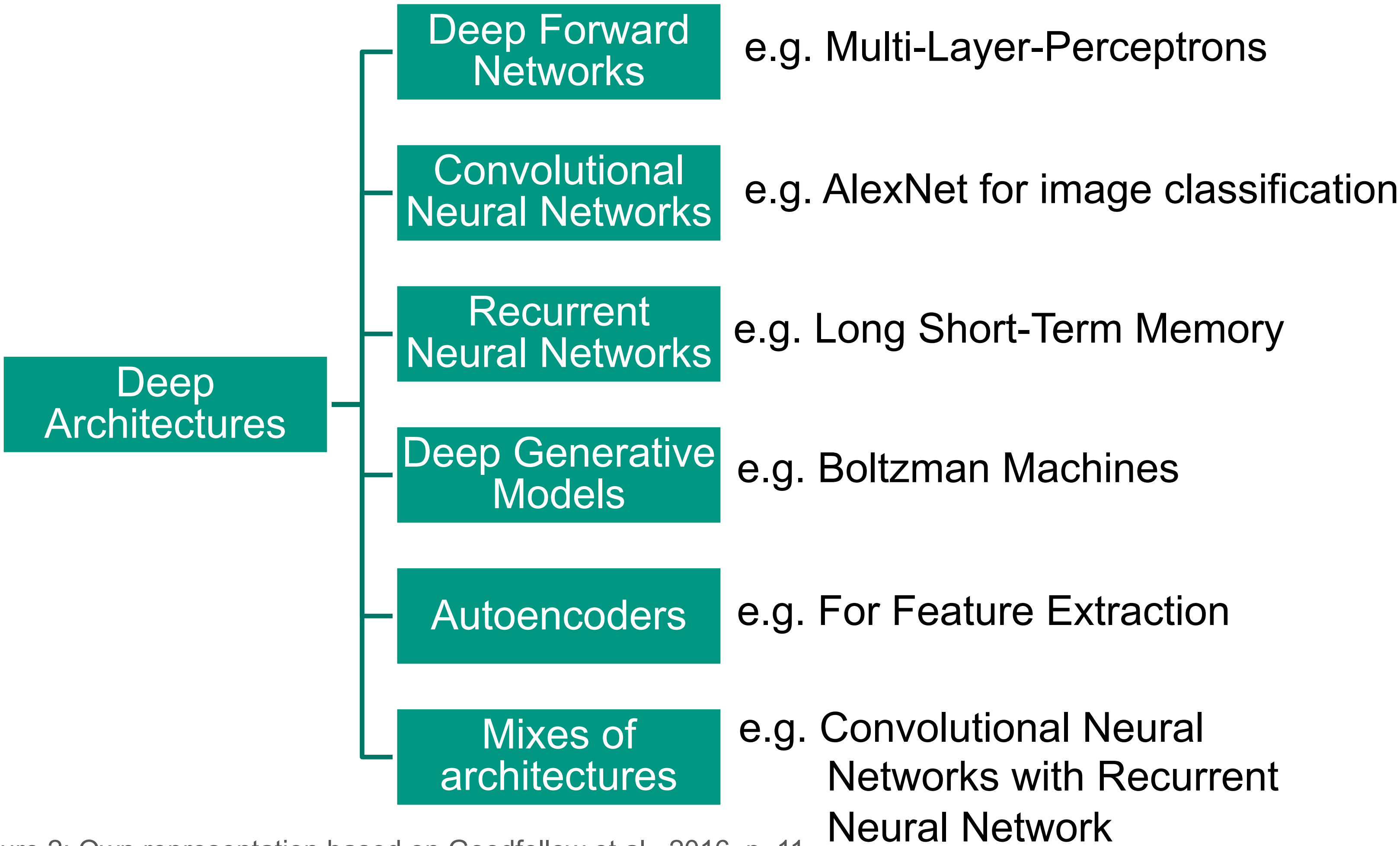


Figure 2: Own representation based on Goodfellow et al., 2016, p. 11

Foundation » Long Short-Term Memory for Energy Time Series » Evaluation » Discussion » Conclusion

Recurrent Neural Networks

- Family of neural networks for processing sequential data
- Basic principle
  - Each member of the output is a function of the previous member of the output
  - Unrolled Recursive Neural Network

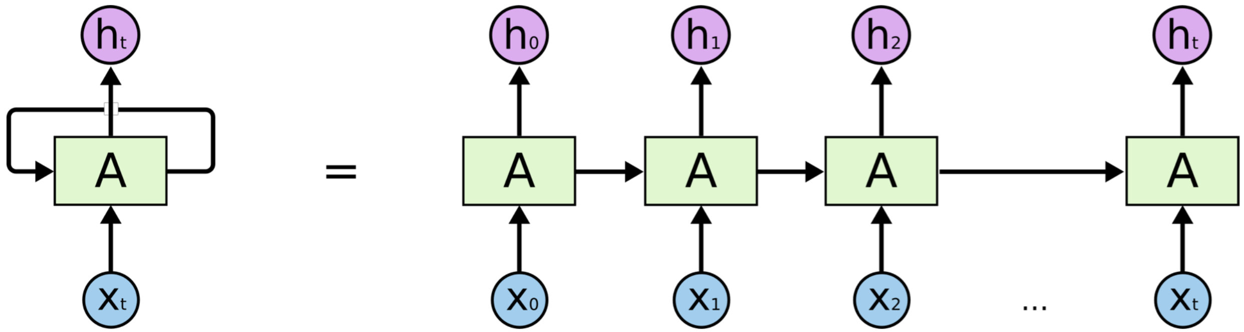


Figure 3: Olah, 2015, http://colah.github.io/posts/2015-08-Understanding-LSTMs/img/RNN-unrolled.png, Retrieved on 16.12.18

(Goodfellow et al. 2016, p. 363f)

- Vanishing gradient problem: challenge to learn long-term dependencies (Hochreiter 1991; Bengio et al. 1993, 1994)

LONG SHORT-TERM MEMORY FOR ENERGY TIME SERIES

Short abstract of Marino et al. (2016)

- Application of two architectural variations of Long Short-Term Memory
  - Stacked Long Short-Term Memory
  - Sequence-to-Sequence Long Short-Term Memory

  (Marino et al. 2016)

- Experiments on benchmark dataset of electricity consumption for a single residential customer (“Individual household electric power consumption”)

  (Dheeru and Taniskidou 2017)

Building Energy Load Forecasting using Deep Neural Networks

Daniel L. Marino, Kasun Amarasinghe, Milos Manic
Department of Computer Science
Virginia Commonwealth University
Richmond, Virginia
marinodl@vcu.edu, amarasinghek@vcu.edu, misko@ieee.org

***Abstract*—Ensuring sustainability demands more efficient energy management with minimized energy wastage. Therefore, the power grid of the future should provide an unprecedented level of flexibility in energy management. To that end, intelligent decision making requires accurate predictions of future energy demand/load, both at aggregate and individual site level. Thus, energy load forecasting have received increased attention in the recent past. However, it has proven to be a difficult problem. This paper presents a novel energy load forecasting methodology based on Deep Neural Networks, specifically, Long Short Term Memory (LSTM) algorithms. The presented work investigates two LSTM based architectures: 1) standard LSTM and 2) LSTM-based Sequence to Sequence (S2S) architecture. Both methods were implemented on a benchmark data set of electricity consumption data from one residential customer. Both architectures were trained and tested on one hour and one-minute time-step resolution datasets. Experimental results showed that the standard LSTM failed at one-minute resolution data while performing well in one-hour resolution data. It was shown that S2S architecture performed well on both datasets. Further, it was shown that the presented methods produced comparable results with the other deep learning methods for energy forecasting in literature.**

Keywords*—Deep Learning; Deep Neural Networks; Long-Short-Term memory; LSTM; Energy; Building Energy; Energy Load forecasting

I. Introduction

Buildings are identified as a major energy consumer worldwide, accounting for 20%-40% of the total energy production [1]-[3]. In addition to being a major energy consumer, buildings are shown to account for a significant portion of energy wastage as well [4]. As energy wastage poses a threat to sustainability, making buildings energy efficient is extremely crucial. Therefore, in making building energy consumption more efficient, it is necessary to have accurate predictions of its future energy consumption.

At the grid level, to minimizing the energy wastage and making the power generation and distribution more efficient, the future of the power grid is moving to a new paradigm of smart grids [5], [6]. Smart grids are promising, unprecedented flexibility in energy generation and distribution [7]. In order to provide that flexibility, the power grid has to be able to dynamically adapt to the changes in demand and efficiently distribute the generated energy from the various sources such as renewables [8]. Therefore, intelligent control decisions should be made continuously at aggregate level as well as modular level in the grid. In achieving that goal and ensuring the reliability of the grid, the ability of forecasting the future demands is important. [6], [9].

Further, demand or load forecasting is crucial for mitigating uncertainties of the future [6]. In that, individual building level demand forecasting is crucial as well as forecasting aggregate loads. In terms of demand response, building level forecasting helps carry out demand response locally since the smart grids incorporate distributed energy generation [6]. The advent of smart meters have made the acquisition of energy consumption data at building and individual site level feasible. Thus data driven and statistical forecasting models are made possible [7].

Aggregate level and building level load forecasting can be viewed in three different categories: 1) Short-term 2) Medium-term and 3) Long-term [6]. It has been determined that the load forecasting is a hard problem and in that, individual building level load forecasting is even harder than aggregate load forecasting [6], [10]. Thus, it has received increased attention from researchers. In literature, two main methods can be found for performing energy load forecasting: 1) Physics principles based models and 2) Statistical and machine learning based models. Focus of the presented work is on the second category of statistical load forecasting. In [7], the authors used Artificial Neural Network (ANN) ensembles to perform the building level load forecasting. ANNs have been explored in detail for the purpose of all three categories of load forecasting [9], [11]-[13]. In [14], the authors use a support vector machines based regression model coupled with empirical mode decomposition to for long-term load forecasting. In [15], electricity demand is forecast using a kernel based multi-task learning methodologies. In [10], authors model individual household electricity loads using sparse coding to perform medium term load forecasting. In the interest of brevity, not all methods in literature are introduced in the paper. For surveys of different techniques used for load forecasting, readers are referred to [16], [17] and [8]. Despite the extensive research carried out in the area, individual site level load forecasting remains to be a difficult problem.

Therefore, the work presented in this paper investigates a deep learning based methodology for performing individual building level load forecasting. Deep learning allows models composed of multiple layers to learn representations in data. The use of multiple layers allow the learning process to be carried out with multiple layers of abstraction. A comprehensive overview and a review of deep learning methodologies can be

978-1-5090-3474-1/16/$31.00 ©2016 IEEE 7046

Figure 4: Marino et al., 2016, p. 7046

Overview of presented Long Short-Term Memory architectures

Stacked Long Short-Term Memory

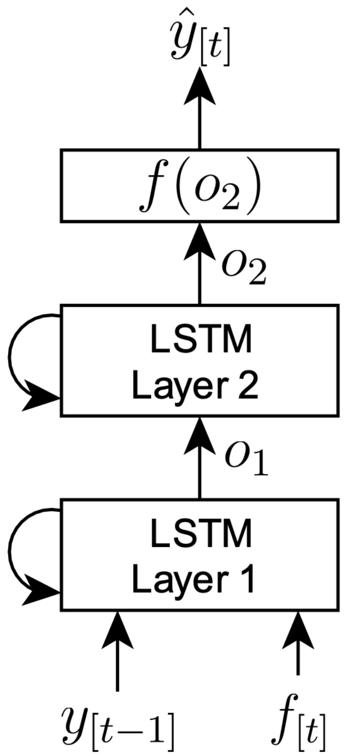


Figure 5:
Own representation based on Marino et al., 2016, p. 7047

Sequence-to-Sequence Long Short-Term Memory

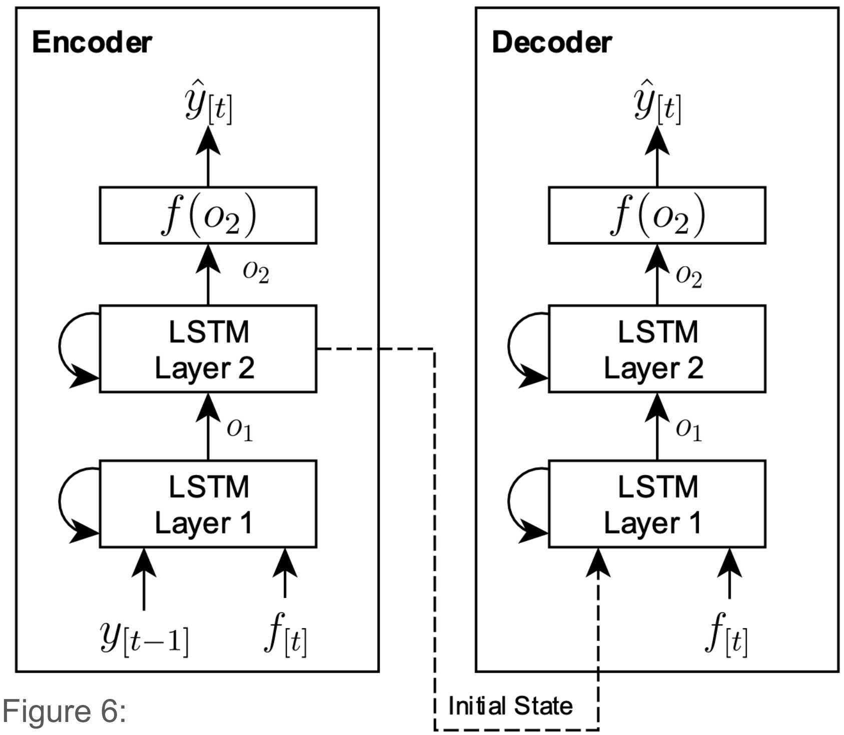


Figure 6:
Own representation

Long Short-Term Memory

Karlsruhe Institute of Technology

- Type of Recurrent Neural Network introduced by Sepp Hochreiter and Jürgen Schmidhuber (1997)
  - Great success in various applications (Goodfellow et al. 2016, p. 363f)
  - Solves vanishing gradient problem

- Key concepts
  - Cell State
  - Gates
    - Input
    - Forget
    - Output

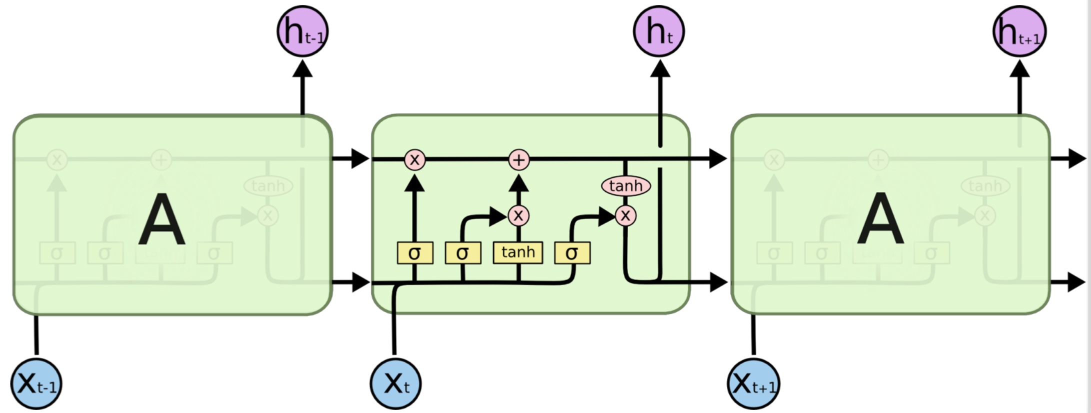
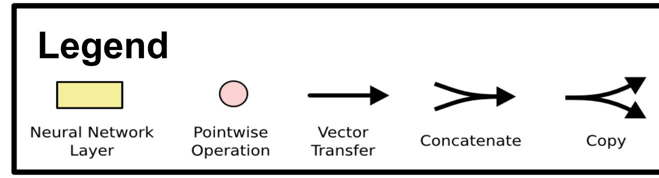


Figure 7: Olah, 2015, http://colah.github.io/posts/2015-08-Understanding-LSTMs/img/LSTM3-chain.png, Retrieved on 03.12.2018

Long Short-Term Memory Cells



0 Cell State

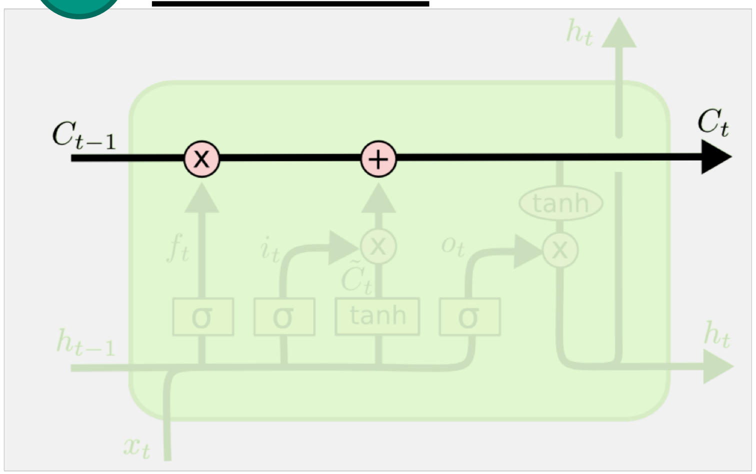
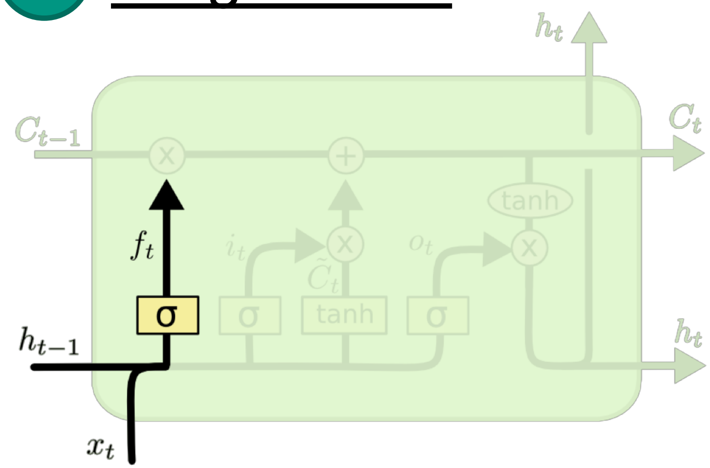


Figure 8: Olah, 2015, http://colah.github.io/posts/2015-08-Understanding-LSTMs/img/LSTM3-C-line.png, Retrieved on 03.12.2018

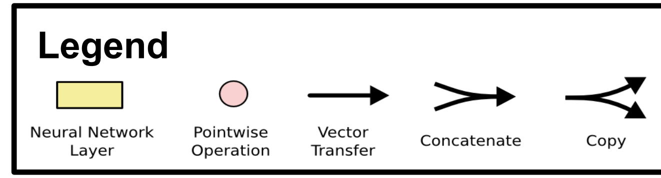
1 Forget Gate



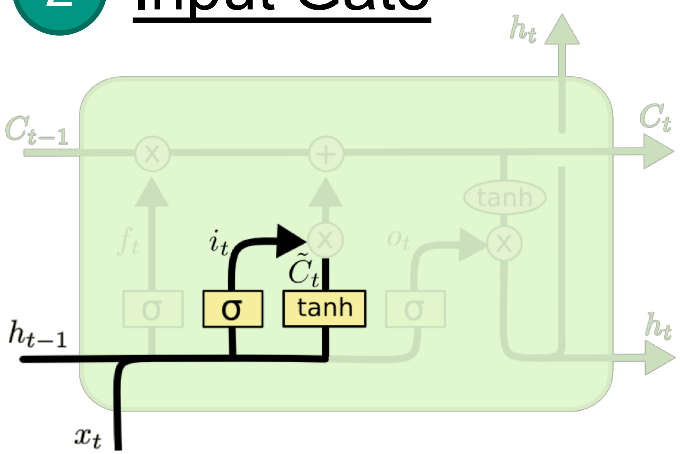
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figure 9: Adapted from Olah, 2015, http://colah.github.io/posts/2015-08-Understanding-LSTMs/img/LSTM3-focus-f.png, Retrieved on 03.12.2018

Long Short-Term Memory Cells



2 Input Gate

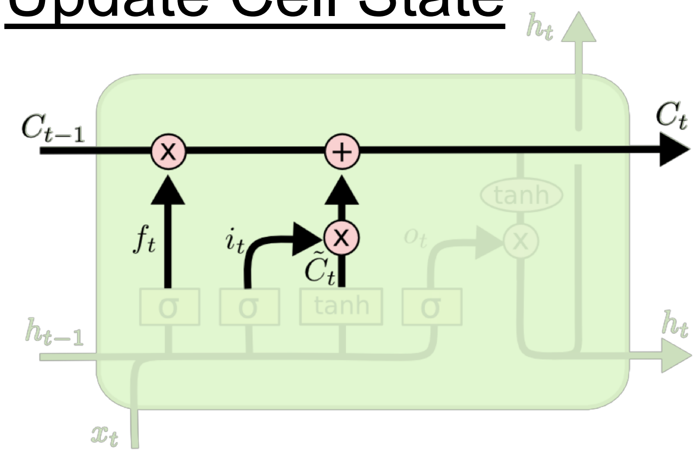


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

Figure 10: Adapted from Olah, 2015, http://colah.github.io/posts/2015-08-Understanding-LSTMs/img/LSTM3-focus-i.png, Retrieved on 03.12.2018

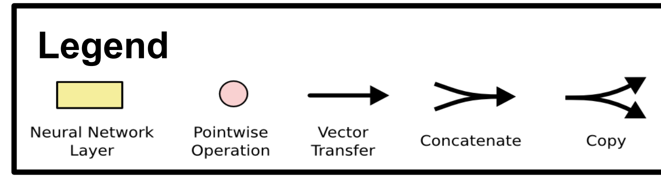
3 Update Cell State



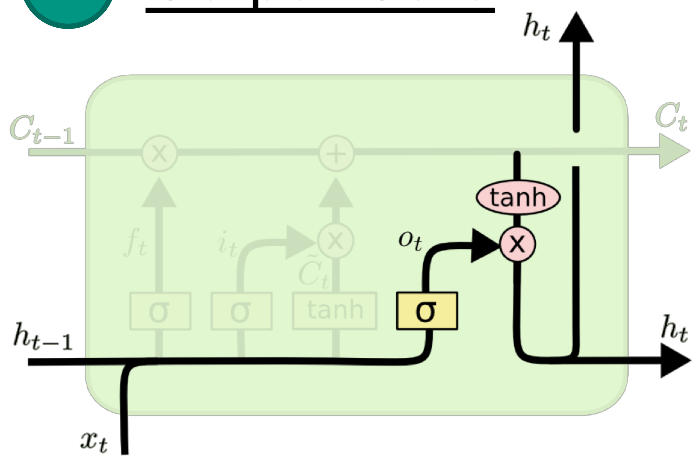
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figure 11: Adapted from Olah, 2015, http://colah.github.io/posts/2015-08-Understanding-LSTMs/img/LSTM3-focus-C.png, Retrieved on 03.12.2018

Long Short-Term Memory Cells



4 Output Gate



$$o_t = \sigma\left(W_o\ [h_{t-1}, x_t]\ +\ b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

Figure 12: Adapted from Olah, 2015, http://colah.github.io/posts/2015-08-Understanding-LSTMs/img/LSTM3-focus-o.png, Retrieved on 03.12.2018

Stacked Long Short-Term Memory architecture

- Stacked architecture
  - Stack multiple Long Short-Term Memory Cells into a multi-layer architecture
  - Dense layer

- Input vector: $i_{[t]} = [\ y_{[t-1]}\ \underbrace{day_{[t]}\ \ day\text{-}of\text{-}week_{[t]}\ \ hour_{[t]}}_{f_{[t]}}\ ]$
  - Use also more than one time step as “context”

- Output vector: $\hat{y}_{[t]}$

(Marino et al. 2016)

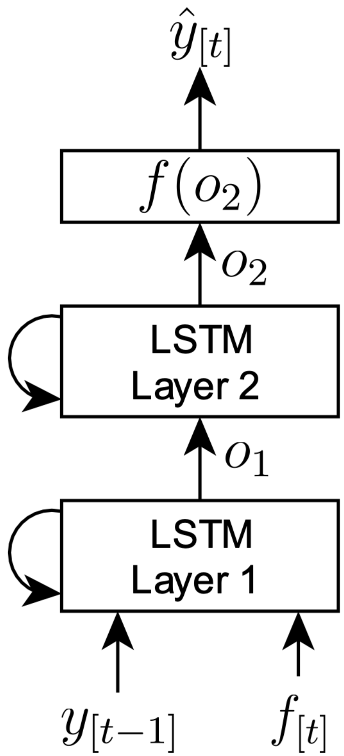


Figure 5: Own representation based on Marino et al., 2016, p. 7047

Overview of presented Long Short-Term Memory architectures

Slide from before

Stacked Long Short-Term Memory

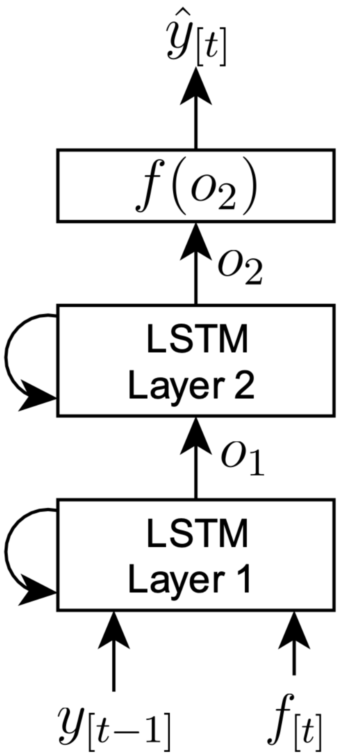


Figure 5: Own representation based on Marino et al., 2016, p. 7047

Sequence-to-Sequence Long Short-Term Memory

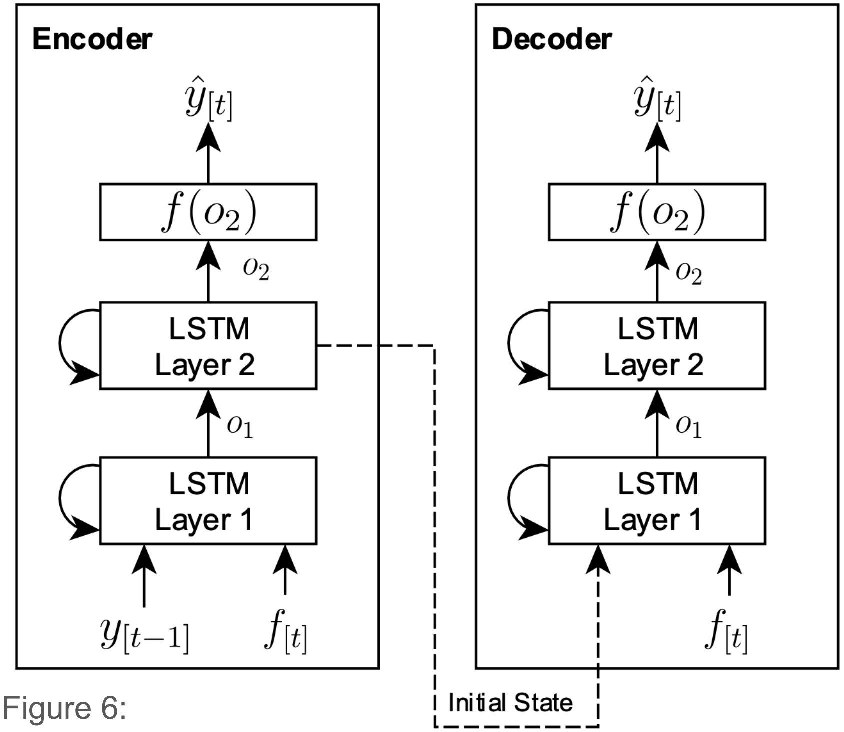


Figure 6: Own representation

Foundation › **Long Short-Term Memory for Energy Time Series** › Evaluation › Discussion › Conclusion

The idea of Sequence-to-Sequence Learning

- Introduced by Sutskever et al. (2014) to map sequences of different lengths

- Comprised of two sub-models
  - Encoder
    - Convert input sequences of variable length and
    - Encode them in a fixed length vector, which is then used as input state for the decoder
  - Decoder
    - Generates an output sequence of fixed length

  (Marino et al. 2016, p. 7048)

- Applications in speech recognition, machine translation and question answering (Goodfellow et al. 2016, p. 385)

Sequence-to-Sequence
Long Short-Term Memory architecture

- Encoder
  - Input vector:
    $i_{[t]} = [\ y_{[t-1]}\ \underbrace{day_{[t]}\ \ day\text{-}of\text{-}week_{[t]}\ \ hour_{[t]}}_{f_{[t]}}\ ]$
  - "Output": Cell State
- Decoder
  - Input vector: $f_{[t]}$
  - Output vector: $\hat{y}_{[t]}$

(Marino et al. 2016)

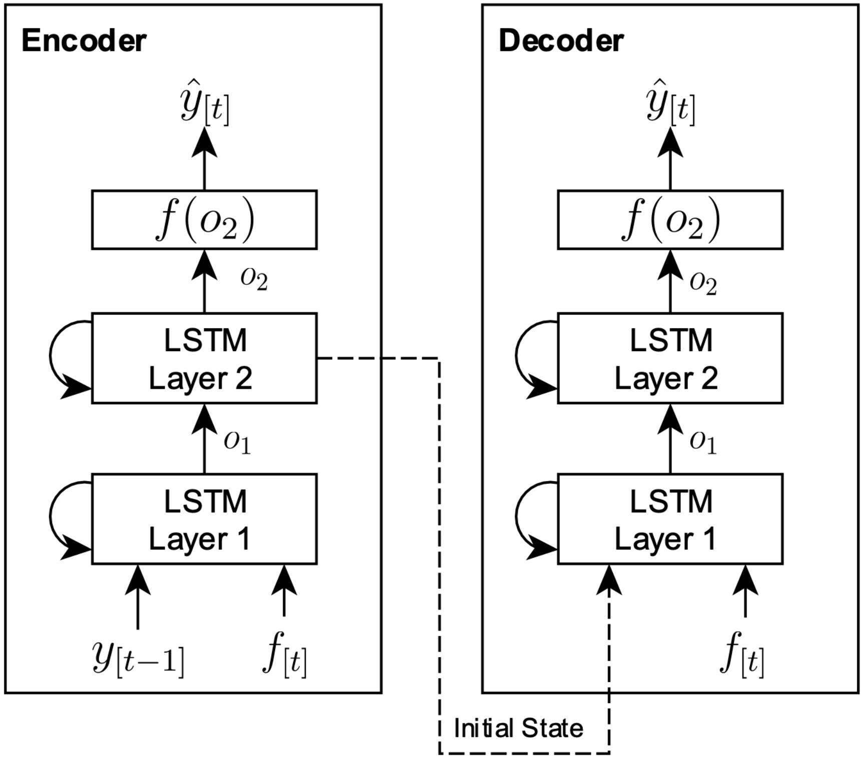


Figure 6: Own representation

EVALUATION

Experiments in Marino et al. (2016)

Data

- "Individual household electric power consumption" (Dheeru and Taniskidou 2017)
- Aggregation Level
  - 1 minute (original)
  - 1 hour
- Training set: 3 years; Test set: 1 year

Model

- Stacked LSTM
- Sequence-to-Sequence LSTM

Horizon

- One Step
- 60 Steps

Evaluation and results

- Stacked LSTM performs well with inputs further in the past for hourly data
- Sequence-to-Sequence LSTM performs well in both datasets
- Comparable results to Mocanu et al. (2016)

Implementation and hardware resources

- Jupyter Notebooks at
  https://github.com/nicoleludwig/EnergyInformatics

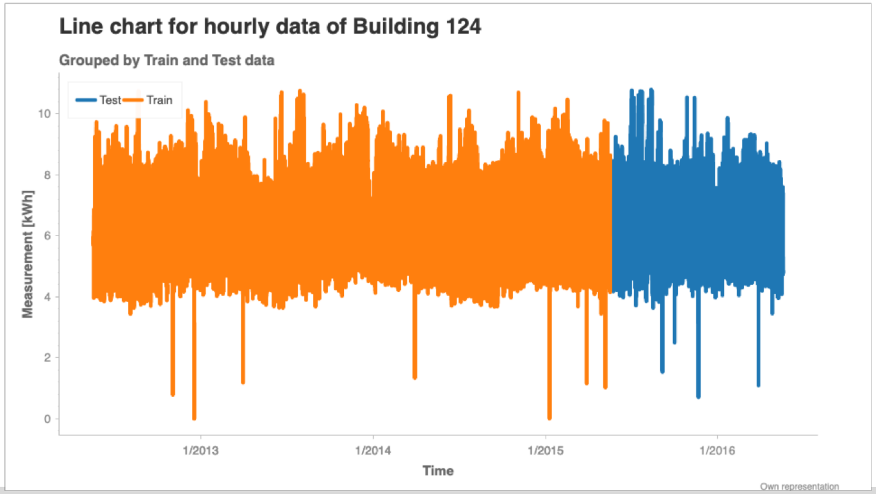
- Implementation with Python
  - Pandas
  - NumPy
  - scikit-learn
  - Keras with TensorFlow as backend
  - Chartify

- Hardware resources for training
  - GPU: NVIDIA Tesla V100*

* Made possible by Andreas Bartschat (IAI), thanks again! ☺

Data for evaluation

- Energy consumption data of KIT Campus Nord
  - Building 124
  - 15 minutes resolution
  - Training set of three years
    (Mid May 2012 until Mid May 2015)
  - Test set of one year
    (Mid May 2015 until Mid May 2016)



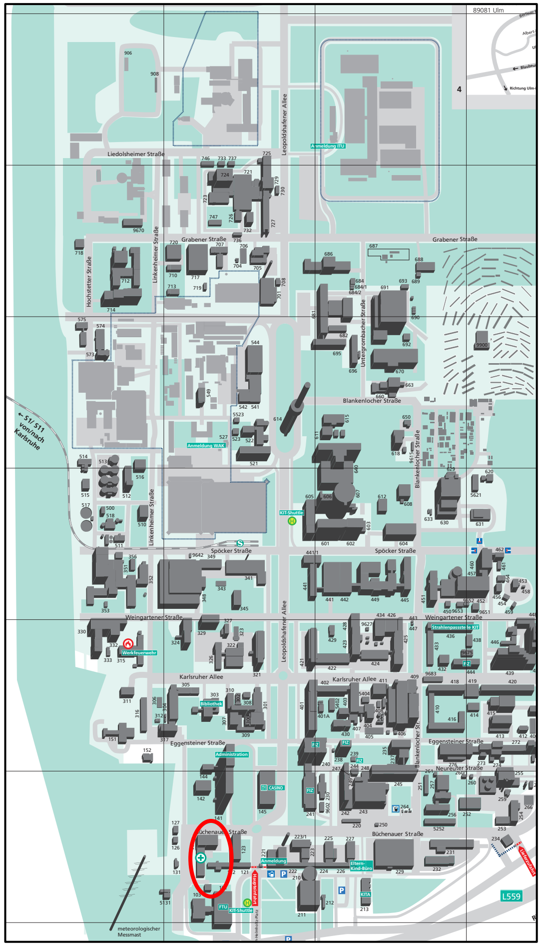


Figure 13: https://www.kit.edu/downloads/Campus-Nord.pdf, Retrieved on 05.01.2019

Foundation › Long Short-Term Memory for Energy Time Series › **Evaluation** › Discussion › Conclusion

Additional time series approaches

- Stationarity
  - "A stationary time series is one whose properties do not depend on the time at which the series is observed." (Hyndman and Athanasopoulos 2018)
  - Implementation with differencing to remove effects
- Normalization (Brownlee 2018, p. 249, Chollet 2018, p. 210f)
  - Min-Max-Scaling
    - Transform features by scaling each feature to a given range
    - Implementation with range [0, 1]
  - Standardization
    - Standardize features by removing the mean and scaling to unit variance (z-transformation)
    - Implementation with mean = 0 and standard deviation = 1

Overview of evaluation scenario characteristics

Characteristic	Category		
Level of Aggregation	15 minutes	1 hour	
Stationarity	Stationary	Non-Stationary	
Normalization	None	Min-Max-Scaling	Standardization
Models	Stacked Long Short-Term Memory	Sequence-to-Sequence Long Short-Term Memory	
Horizon	One step	One Day	One Week

→ 72 scenarios per building

Evaluation metric

Karlsruhe Institute of Technology

- Root mean squared error (RMSE)

$$\text{RMSE} = \sqrt{\frac{1}{T \ * n_v} \sum_{t=1}^{T} \sum_{i=1}^{n_v} (v_{i,t} \ - \ \hat{v}_{i,t})^2}$$

n_v → number of steps in horizon
T → total number of steps predicted into the future
v_i → real values for time-step t
$\hat{v}_i$ → value predicted by the model for time-step t

(Mocanu et al. 2016, p. 95)

Baseline models

Legend: (15 minutes / 1 hour)

- Naïve Forecast
  - Value of the last week on the same day of the week at the same time
  - Prediction for timestep t: (t - 672 / t - 168)

- Ordinary Regression
  - Regression of the respective load time series with the following regressors
    - Load before one day (t - 96 / t - 24)
    - Load before two days (t - 192 / t - 48)
    - Load before one week (t - 672 / t - 168)
    - Dummy variables for weekend, month, hour and minute (only for 15 minutes)

Results for Building 124

Level of Aggregation	Stationary	Normalization	RMSE per time horizon: Stacked-LSTM	S2S-LSTM	Naïve Forecast	Ordinary Regression
15 Minutes	No	None	(0,015; 0,853; 0,893)	(1,474; 1,496; 1,491)	0,909	(0,397; 0,789; 0,825)
		Min-Max	(0,01; 0,792; 0,865)	(1,482; 1,473; 1,491)		
		Standard	(0,011; 0,816; 0,868)	(1,475; 1,489; 1,494)		
	Yes	None	(0,007; 1,193; 1,726)	(0,427; 1,755; 2,022)		
		Min-Max	(0,016; 1,039; 3,196)	(0,427; 1,761; 2,093)		
		Standard	(0,008; 1,177; 1,693)	(0,427; 1,755; 2,023)		
1 Hour	No	None	(0,021; 0,742; 0,871)	(1,453; 1,449; 1,452)	0,829	(0,311; 0,724; 0,76)
		Min-Max	(0,026; 0,712; 0,81)	(1,433; 1,447; 1,453)		
		Standard	(0,03; 0,73; 0,864)	(1,443; 1,456; 1,452)		
	Yes	None	(0,008; 0,839; 1,291)	(0,558; 1,693; 1,964)		
		Min-Max	(0,049; 0,885; 1,899)	(0,558; 1,695; 2,015)		
		Standard	(0,009; 0,856; 1,208)	(0,558; 1,7; 1,964)		

Legend
Root Mean Squared Error (One Step; One Day; One Week) | LSTM – Long Short-Term Memory | S2S – Sequence-to-Sequence

Remarks on results

- Technical and practical insights
  - Handling time shift and some missing days in 2012
  - Data pre-processing pipeline
    - Traditional time series analysis: 2D [samples, features]
    - Deep Learning: 3D tensor [samples, time-steps, features]
  - Distinction between one-step and multi-step forecasts
  - Parameters

- Time for training per Long Short-Term Memory model approx. 6,5 hours for all scenarios

DISCUSSION

Critical reflection on Marino et al. (2016)

Missing parameter configurations

- Training batch and epochs
- Norm clipping
- Dropout
- Activation function at Dense layer
- …

Comparable results with Mocanu et al. (2016)

- In Mocanu et al. (2016) seven experiments with different resolutions and time horizons on the same dataset (Dheeru and Taniskidou 2017)
- Implementation and evaluation of Conditional Restricted Boltzmann Machine and Factored Conditional Restricted Boltzmann Machine
- Comparable result in one of seven scenarios → Results of other scenarios?

Related work

Paper Reference / Deep Architecture	Deep Forward Networks	Con-volutional Neural Networks	Recurrent Neural Networks	Deep Generative Networks	Auto-encoders
Amarasinghe et al. 2017		CNN			
Gensler et al. 2016			LSTM*		AE*
He 2017		CNN*	LSTM*		
Heghedus et al. 2018			GRU		
Jarábek et al. 2017			S2S-LSTM		
Li et al. 2017	ELM*				SAE*
Marino et al. 2016			Stacked- & S2S-LSTM		
Mocanu et al. 2016			RNN	CRBM & FCRBM	
Ryu et al. 2016				RBM	
Voß et al. 2018		WaveNet			

Legend: * - Combined Approach | CNN – Convolutional Neural Networks | (F)(C)RBM – (Factored) (Conditional) Restricted Boltzmann Machine | ELM – Extreme Learning Machine | GRU – Gated Recurrent Unit | LSTM – Long Short-Term Memory | RNN – Recurrent Neural Network | S2S – Sequence-to-Sequence | (S)AE – (Stacked) Autoencoder

CONCLUSION

Summary

- Achievements
  - Long Short-Term Memory as one Deep Learning method for Energy Time Series Forecasting
  - Overview of current Deep Architectures for Energy Time Series
  - Implementation and evaluation of
    - Stacked Long Short-Term Memory and
    - Sequence-to-Sequence Long Short-Term Memory

    based on Marino et. al. (2016)

- Summarised results
  - Stacked Long Short-Term Memory outperformed baseline models for hourly data in one-step and one-day predictions
  - Good performance of Sequence-to-Sequence Long Short-Term Memory could not be confirmed (without hyperparameter optimization)

Outlook

Deep Learning

- Hyperparameter optimization
  - Grid Search
  - Random Search
  - Bayesian Optimization, etc.
- Variations of Long Short-Term Memory

Time Series

- Verify results with more buildings
- “Real” univariate Time Series Forecast
- Multivariate Time Series Forecast

Questions

Karlsruhe Institute of Technology

?

Bibliography (1/5)

- Amarasinghe, K., Marino, D. L., & Manic, M. (2017). Deep neural networks for energy load forecasting. In *26th {IEEE} International Symposium on Industrial Electronics, {ISIE} 2017, Edinburgh, United Kingdom, June 19-21, 2017* (pp. 1483–1488). IEEE.
- Bengio, Y., Frasconi, P., & Simard, P. (1993). The problem of learning long-term dependencies in recurrent networks. In *Neural Networks, 1993., IEEE International Conference on* (pp. 1183–1188).
- Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, *5*(2), 157–166.
- Brownlee, J. (2018). Deep Learning for Time Series Forecasting. Machine Learning Mastery. Retrieved from https://machinelearningmastery.com/deep-learning-for-time-series-forecasting/
- Li, C., Ding, Z., Zhao, D., Yi, J., & Zhang, G. (2017). Building energy consumption prediction: An extreme deep learning approach. *Energies*, *10*(10), 1–20.

Bibliography (2/5)

- Chollet, F. 2018. Deep Learning with Python. Manning. Shelter Island.
- Dheeru, D., & Karra Taniskidou, E. (2017). “Individual household electric power consumption”. UCI Machine Learning Repository. Retrieved from http://archive.ics.uci.edu/ml
- Goodfellow, I., Bengio, Y. & Courville, A. (2016). Deep Learning. Cambridge, MA, USA: MIT Press.
- He, W. (2017). Load Forecasting via Deep Neural Networks. In V. Ahuja, Y. Shi, D. Khazanchi, N. Abidi, Y. Tian, D. Berg, & J. M. Tien (Eds.), *Proceedings of the 5th International Conference on Information Technology and Quantitative Management, {ITQM} 2017, December 8-10, 2017, New Delhi, India* (Vol. 122, pp. 308–314). Elsevier.
- Heghedus, C., Chakravorty, A., & Rong, C. (2018). Energy Load Forecasting Using Deep Learning. In *2018 IEEE International Conference on Energy Internet (ICEI)* (pp. 146–151).

Bibliography (3/5)

- Hochreiter, S. (1991). Untersuchungen zu dynamischen neuronalen Netzen. *Diploma, Technische Universität München*, *91*(1).
- Hochreiter, S. & Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 9(8), 1735–1780.
- Hyndman, R. J. & Athanasopoulos, G. Forecasting: Principles and Practice. https://otexts.org/fpp2/stationarity.html#fn14, Retrieved on 17.12.18
- Jarábek, T., Laurinec, P., & Lucká, M. (2017). Energy load forecast using S2S deep neural networks with k-Shape clustering. In *2017 IEEE 14th International Scientific Conference on Informatics* (pp. 140–145).
- Li, C., Ding, Z., Zhao, D., Yi, J., & Zhang, G. (2017). Building energy consumption prediction: An extreme deep learning approach. *Energies*, *10*(10), 1–20.

Bibliography (4/5)

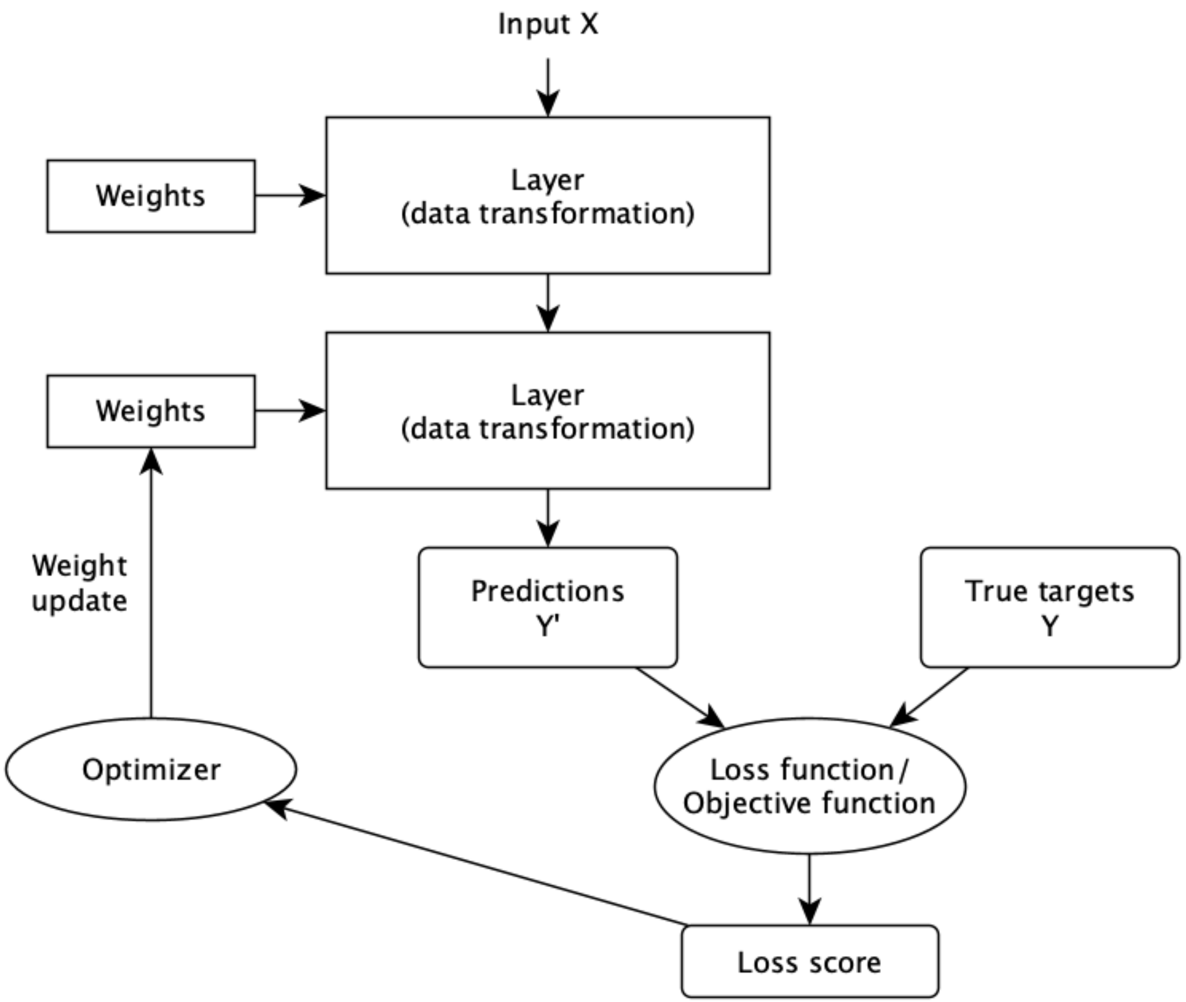
- Marino, D. L., Amarasinghe, K., & Manic, M. (2016). Building energy load forecasting using Deep Neural Networks. IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, 7046–7051.
- Mocanu, E., Nguyen, P. H., Gibescu, M., & Kling, W. L. (2016). Deep learning for estimating building energy consumption. Sustainable Energy, Grids and Networks, 6, 91–99.
- Pérez-Lombard, L., Ortiz, J., & Pout, C. (2008). A review on buildings energy consumption information. *Energy and Buildings*, *40*(3), 394–398.
- Ryu, S., Noh, J., & Kim, H. (2016). Deep neural network based demand side short term load forecasting. *2016 IEEE International Conference on Smart Grid Communications, SmartGridComm 2016*, 308–313.
- Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. Advances in Neural Information Processing Systems (NIPS), 3104–3112.

Bibliography (5/5)

- Voß, M., Bender-Saebelkampf, C., & Albayrak, S. (2018). Residential Short-Term Load Forecasting Using Convolutional Neural Networks. In *IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*.

APPENDIX AND BACKUP

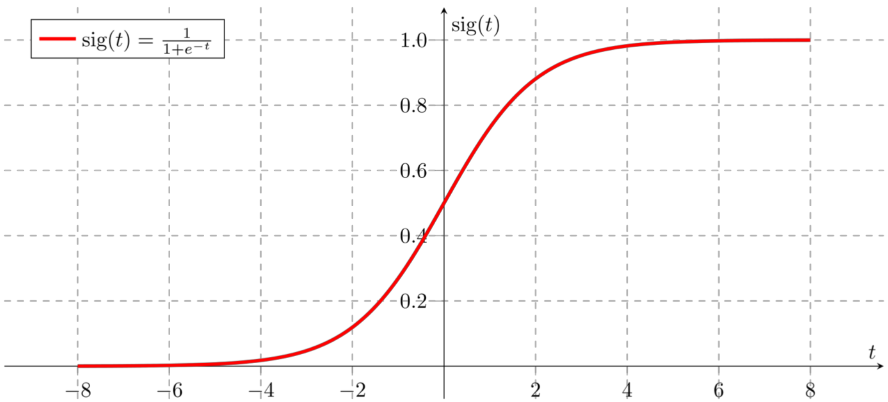
How does deep learning basically work?



Source: Own representation based on Chollet (2018, p.11)

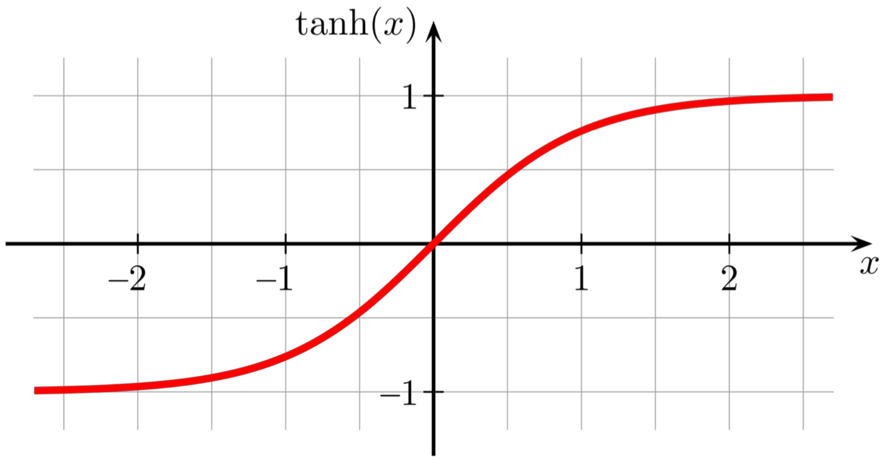
Activation functions

- Sigmoid (σ)
  - $\sigma(t) = \frac{1}{1+e^{-t}}$
  - Value range [0; 1]



Source: https://upload.wikimedia.org/wikipedia/commons/thumb/5/53/Sigmoid-function-2.svg/1920px-Sigmoid-function-2.svg.png (Retrieved on 06.01.2019)

- **Tangens hyperbolicus (tanh)**
  - $\tanh(t) = \frac{e^{t} - e^{-t}}{e^{t} + e^{-t}}$
  - Value range [-1; 1]



Source: https://upload.wikimedia.org/wikipedia/commons/thumb/8/87/Hyperbolic_Tangent.svg/1920px-Hyperbolic_Tangent.svg.png (Retrieved on 06.01.2019)

Details: Stacked Long Short-Term Memory

- Objective function $L = \sum_{t=1}^{M} (y_{[t]} - \hat{y}_{[t]})^2$
- Unrolling implemented with M=50
- Optimizer Adam
- Training with Backpropagation Through Time
- Optimization with Norm clipping

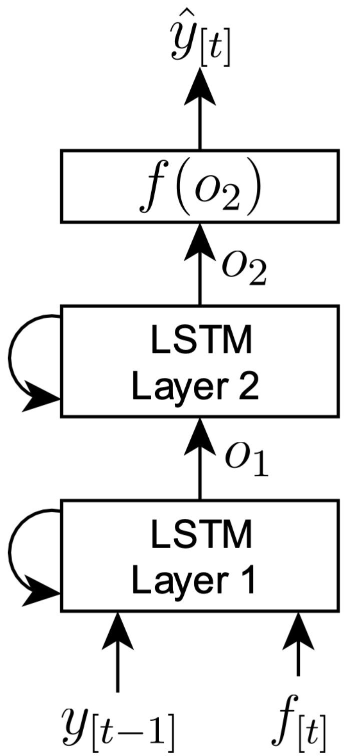


Figure 5:
Own representation based on Marino et al., 2016, p. 7047

Details: S2S Long Short-Term Memory

- Similar setup as Stacked Long Short-Term Memory

- Encoder objective function

$$L_E = \sum_{t=1}^{M} (y_{[t]} - \hat{y}_{[t]})^2$$

- Decoder objective function

$$L_D = \sum_{t=M+1}^{T} (y_{[t]} - \hat{y}_{[t]})^2$$

- Optimization with Norm clipping
- Regularization with Dropout

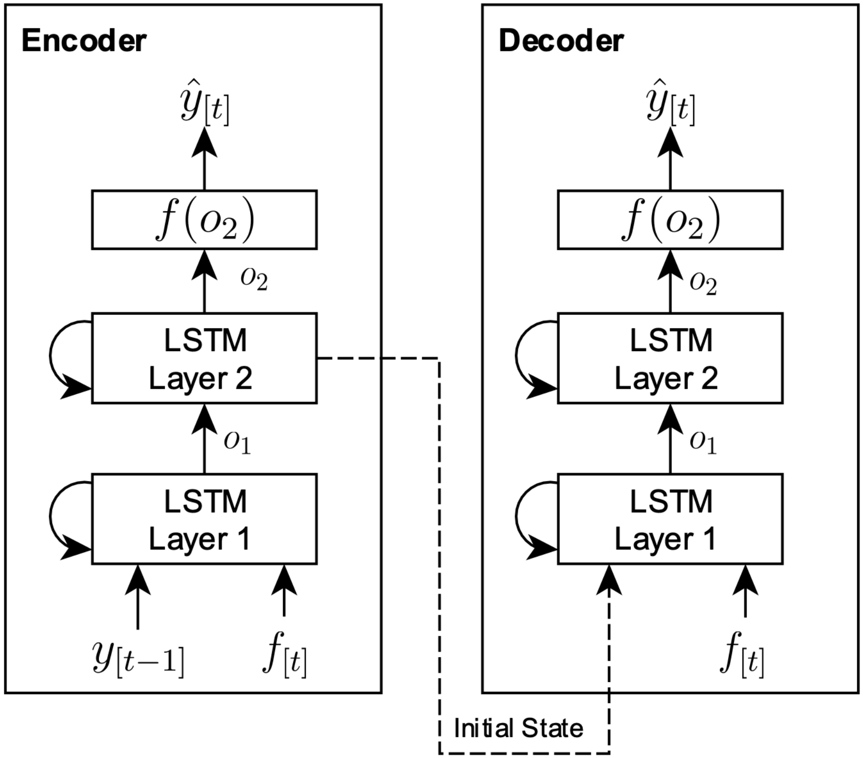


Figure 6:
Own representation

Backpropagation Through Time for S2S LSTM