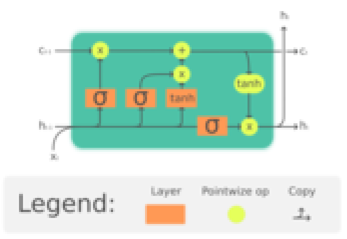
WIKIPEDIA

Long short-term memory

Long short-term memory (**LSTM**) units are units of a recurrent neural network (RNN). An RNN composed of LSTM units is often called an *LSTM network*. A common LSTM unit is composed of a **cell**, an **input gate**, an **output gate** and a **forget gate**. The cell remembers values over arbitrary time intervals and the three *gates* regulate the flow of information into and out of the cell.



The Long Short-Term Memory (LSTM) cell can process data sequentially and keep its hidden state through time.

LSTM networks are well-suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. LSTMs were developed to deal with the exploding and vanishing gradient problems that can be encountered when training traditional RNNs. Relative insensitivity to gap length is an advantage of LSTM over RNNs, hidden Markov models and other sequence learning methods in numerous applications.

Contents

History

LSTM was proposed in 1997 by Sepp Hochreiter and Jürgen Schmidhuber[1] and improved in 2000 by Felix Gers' team.[2]

Among other successes, LSTM achieved record results in natural language text compression,[3] unsegmented connected handwriting recognition[4] and won the ICDAR handwriting competition (2009). LSTM networks were a major component of a network that achieved a record 17.7% phoneme error rate on the classic TIMIT natural speech dataset (2013).[5]

As of 2016, major technology companies including Google, Apple, and Microsoft were using LSTM as fundamental components in new products.[6] For example, Google used LSTM for speech recognition on the smartphone,[7][8] for the smart assistant Allo[9] and for Google Translate.[10][11] Apple uses LSTM for the "Quicktype" function on the iPhone[12][13] and for Siri.[14] Amazon uses LSTM for Amazon Alexa.[15]

In 2017 Microsoft reported reaching 95.1% recognition accuracy on the Switchboard corpus, incorporating a vocabulary of 165,000 words. The approach used "dialog session-based long-short-term memory".[16]

Architectures

There are several architectures of LSTM units. A common architecture is composed of a memory *cell*, an *input gate*, an *output gate* and a *forget gate*.

An LSTM *cell* takes an input and stores it for some period of time. This is equivalent to applying the identity function ($f(x) = x$) to the input. Because the derivative of the identity function is constant, when an LSTM network is trained with backpropagation through time, the gradient does not vanish.

The activation function of the LSTM *gates* is often the logistic function. Intuitively, the *input gate* controls the extent to which a new value flows into the cell, the *forget gate* controls the extent to which a value remains in the cell and the *output gate* controls the extent to which the value in the cell is used to compute the output activation of the LSTM unit.

There are connections into and out of the LSTM *gates*, a few of which are recurrent. The weights of these connections, which need to be learned during training, determine how the gates operate.

Variants

In the equations below, the lowercase variables represent vectors. Matrices W_q and U_q contain, respectively, the weights of the input and recurrent connections, where q can either be the input gate i, output gate o, the forget gate f or the memory cell c, depending on the activation being calculated.

LSTM with a forget gate

The compact forms of the equations for the forward pass of an LSTM unit with a forget gate are:[1][2]

$$\begin{aligned}
f_t &= \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \\
i_t &= \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \\
o_t &= \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \\
c_t &= f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \\
h_t &= o_t \circ \sigma_h(c_t)
\end{aligned}$$

where the initial values are $c_0 = 0$ and $h_0 = 0$ and the operator $\circ$ denotes the Hadamard product (element-wise product). The subscript t indexes the time step.

Variables

- $x_t \in \mathbb{R}^d$: input vector to the LSTM unit
- $f_t \in \mathbb{R}^h$: forget gate's activation vector
- $i_t \in \mathbb{R}^h$: input gate's activation vector
- $o_t \in \mathbb{R}^h$: output gate's activation vector
- $h_t \in \mathbb{R}^h$: output vector of the LSTM unit
- $c_t \in \mathbb{R}^h$: cell state vector

- $W \in \mathbb{R}^{h \times d}$, $U \in \mathbb{R}^{h \times h}$ and $b \in \mathbb{R}^{h}$: weight matrices and bias vector parameters which need to be learned during training

where the superscripts d and h refer to the number of input features and number of hidden units, respectively.

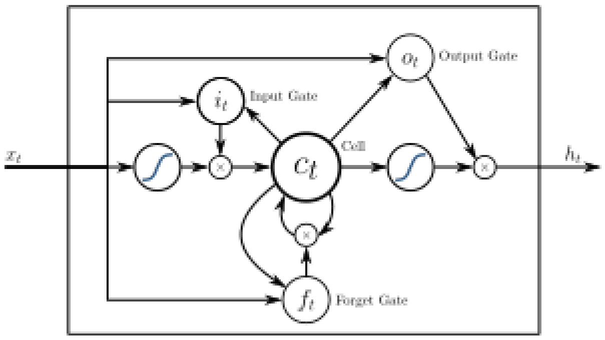
Activation functions

- σ_g: sigmoid function.
- σ_c: hyperbolic tangent function.
- σ_h: hyperbolic tangent function or, as the peephole LSTM paper suggests, $\sigma_h(x) = x$.[17][18]

Peephole LSTM

The figure on the right is a graphical representation of an LSTM unit with peephole connections (i.e. a peephole LSTM).[17][18] Peephole connections allow the gates to access the constant error carousel (CEC), whose activation is the cell state.[20] h_{t-1} is not used, c_{t-1} is used instead in most places.

$$\begin{aligned}
f_t &= \sigma_g(W_f x_t + U_f c_{t-1} + b_f) \\
i_t &= \sigma_g(W_i x_t + U_i c_{t-1} + b_i) \\
o_t &= \sigma_g(W_o x_t + U_o c_{t-1} + b_o) \\
c_t &= f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c c_{t-1} + b_c) \\
h_t &= o_t \circ \sigma_h(c_t)
\end{aligned}$$



A peephole LSTM unit with input (i.e. i), output (i.e. o), and forget (i.e. f) gates. Each of these gates can be thought as a "standard" neuron in a feed-forward (or multi-layer) neural network: that is, they compute an activation (using an activation function) of a weighted sum. i_t, o_t and f_t represent the activations of respectively the input, output and forget gates, at time step t. The 3 exit arrows from the memory cell c to the 3 gates i, o and f represent the *peephole* connections. These peephole connections actually denote the contributions of the activation of the memory cell c at time step $t-1$, i.e. the contribution of c_{t-1} (and not c_t, as the picture may suggest). In other words, the gates i, o and f calculate their activations at time step t (i.e., respectively, i_t, o_t and f_t) also considering the activation of the memory cell c at time step $t-1$, i.e. c_{t-1}. The single left-to-right arrow exiting the memory cell is *not* a peephole connection and denotes c_t. The little circles containing a $\times$ symbol represent an element-wise multiplication between its inputs. The big circles containing an *S*-like curve represent the application of a differentiable function (like the sigmoid function) to a weighted sum. There are many other kinds of LSTMs as well.[19]

Peephole convolutional LSTM

Peephole convolutional LSTM.[21] The $*$ denotes the convolution operator.

$$\begin{aligned}
f_t &= \sigma_g(W_f * x_t + U_f * h_{t-1} + V_f \circ c_{t-1} + b_f) \\
i_t &= \sigma_g(W_i * x_t + U_i * h_{t-1} + V_i \circ c_{t-1} + b_i) \\
o_t &= \sigma_g(W_o * x_t + U_o * h_{t-1} + V_o \circ c_{t-1} + b_o) \\
c_t &= f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c * x_t + U_c * h_{t-1} + b_c) \\
h_t &= o_t \circ \sigma_h(c_t)
\end{aligned}$$

Training

To minimize LSTM's total error on a set of training sequences, iterative gradient descent such as backpropagation through time can be used to change each weight in proportion to the derivative of the error with respect to it. A problem with using gradient descent for standard RNNs is that error gradients vanish exponentially quickly with the size of the time lag between important events. This is due to $\lim_{n\to\infty} W^n = 0$ if the spectral radius of W is smaller than 1.[22][23] With LSTM units, however, when error values are back-propagated from the output, the error remains in the unit's memory. This "error carousel" continuously feeds error back to each of the gates until they learn to cut off the value. Thus, regular backpropagation is effective at training an LSTM unit to remember values for long durations.

LSTM can also be trained by a combination of artificial evolution for weights to the hidden units, and pseudo-inverse or support vector machines for weights to the output units.[24] In reinforcement learning applications LSTM can be trained by policy gradient methods, evolution strategies or genetic algorithms.

CTC score function

Many applications use stacks of LSTM RNNs[25] and train them by connectionist temporal classification (CTC)[26] to find an RNN weight matrix that maximizes the probability of the label sequences in a training set, given the corresponding input sequences. CTC achieves both alignment and recognition.

Applications

Applications of LSTM include:

- Robot control[27]
- Time series prediction[28]
- Speech recognition[29][30][31]
- Rhythm learning[18]
- Music composition[32]
- Grammar learning[33][17][34]
- Handwriting recognition[35][36]
- Human action recognition[37]
- Sign Language Translation[38]
- Protein Homology Detection[39]
- Predicting subcellular localization of proteins[40]
- Time series anomaly detection[41]
- Several prediction tasks in the area of business process management[42]
- Prediction in medical care pathways[43]
- Semantic parsing[44]
- Object Co-segmentation[45][46]

LSTM has Turing completeness in the sense that given enough network units it can compute any result that a conventional computer can compute, provided it has the proper weight matrix, which may be viewed as its program.

See also

- Differentiable neural computer
- Gated recurrent unit
- Long-term potentiation
- Prefrontal cortex basal ganglia working memory
- Recurrent neural network
- Time series

External links

- Recurrent Neural Networks (http://www.idsia.ch/~juergen/rnn.html) with over 30 LSTM papers by Jürgen Schmidhuber's group at IDSIA
- Gers PhD thesis (http://www.felixgers.de/papers/phd.pdf) on LSTM networks.
- Fraud detection paper (https://web.archive.org/web/20120522234026/http://etd.uwc.ac.za/usrfiles/modules/etd/docs/etd_init_3937_1174040706.pdf) (original (http://etd.uwc.ac.za/bitstream/handle/11394/249/Abidogun_MSC_2005.pdf)) with two chapters devoted to explaining recurrent neural networks, especially LSTM.
- Paper (http://www.cs.umd.edu/~dmonner/papers/nn2012.pdf) on a high-performing extension of LSTM that has been simplified to a single node type and can train arbitrary architectures.
- Tutorial: How to implement LSTM in Python with Theano (http://christianherta.de/lehre/dataScience/machineLearning/neuralNetworks/LSTM.html)
- Gers, Felix A., Nicol N. Schraudolph, and Jürgen Schmidhuber. "Learning precise timing with LSTM recurrent networks." Journal of machine learning research 3, no. Aug (2002): 115-143. (http://www.jmlr.org/papers/volume3/gers02a/gers02a.pdf)[47]
- Tutorial: How to use LSTMs with TensorFlow in Python on cellphone sensor data (https://github.com/guillaume-chevalier/LSTM-Human-Activity-Recognition)

References

1. Sepp Hochreiter; Jürgen Schmidhuber (1997). "Long short-term memory" (https://www.researchgate.net/publication/13853244_Long_Short-term_Memory). *Neural Computation*. **9** (8): 1735–1780. doi:10.1162/neco.1997.9.8.1735 (https://doi.org/10.1162%2Fneco.1997.9.8.1735). PMID 9377276 (https://www.ncbi.nlm.nih.gov/pubmed/9377276).
2. Felix A. Gers; Jürgen Schmidhuber; Fred Cummins (2000). "Learning to Forget: Continual Prediction with LSTM" (http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.55.5709). *Neural Computation*. **12** (10): 2451–2471. doi:10.1162/089976600300015015 (https://doi.org/10.1162%2F089976600300015015).
3. "The Large Text Compression Benchmark" (http://www.mattmahoney.net/dc/text.html#1218). Retrieved 2017-01-13.
4. Graves, A.; Liwicki, M.; Fernández, S.; Bertolami, R.; Bunke, H.; Schmidhuber, J. (May 2009). "A Novel Connectionist System for Unconstrained Handwriting Recognition" (http://ieeexplore.ieee.org/document/4531750/). *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **31** (5): 855–868. doi:10.1109/tpami.2008.137 (https://doi.org/10.1109%2Ftpami.2008.137). ISSN 0162-8828 (https://www.worldcat.org/issn/0162-8828).
5. Graves, Alex; Mohamed, Abdel-rahman; Hinton, Geoffrey (2013-03-22). "Speech Recognition with Deep Recurrent Neural Networks". arXiv:1303.5778 (https://arxiv.org/abs/1303.5778) [cs.NE (https://arxiv.org/archive/cs.NE)].
6. "With QuickType, Apple wants to do more than guess your next text. It wants to give you an AI" (https://www.wired.com/2016/06/apple-bringing-ai-revolution-iphone/). *WIRED*. Retrieved 2016-06-16.
7. Beaufays, Françoise (August 11, 2015). "The neural networks behind Google Voice transcription" (http://googleresearch.blogspot.co.at/2015/08/the-neural-networks-behind-google-voice.html). *Research Blog*. Retrieved 2017-06-27.

8. Sak, Haşim; Senior, Andrew; Rao, Kanishka; Beaufays, Françoise; Schalkwyk, Johan (September 24, 2015). "Google voice search: faster and more accurate" (http://googleresearch.blogspot.co.uk/2015/09/google-voice-search-faster-and-more.html). *Research Blog*. Retrieved 2017-06-27.
9. Khaitan, Pranav (May 18, 2016). "Chat Smarter with Allo" (http://googleresearch.blogspot.co.at/2016/05/chat-smarter-with-allo.html). *Research Blog*. Retrieved 2017-06-27.
10. Wu, Yonghui; Schuster, Mike; Chen, Zhifeng; Le, Quoc V.; Norouzi, Mohammad; Macherey, Wolfgang; Krikun, Maxim; Cao, Yuan; Gao, Qin (2016-09-26). "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation". arXiv:1609.08144 (https://arxiv.org/abs/1609.08144) [cs.CL (https://arxiv.org/archive/cs.CL)].
11. Metz, Cade (September 27, 2016). "An Infusion of AI Makes Google Translate More Powerful Than Ever | WIRED" (https://www.wired.com/2016/09/google-claims-ai-breakthrough-machine-translation/). *www.wired.com*. Retrieved 2017-06-27.
12. Efrati, Amir (June 13, 2016). "Apple's Machines Can Learn Too" (https://www.theinformation.com/apples-machines-can-learn-too). *The Information*. Retrieved 2017-06-27.
13. Ranger, Steve (June 14, 2016). "iPhone, AI and big data: Here's how Apple plans to protect your privacy | ZDNet" (http://www.zdnet.com/article/ai-big-data-and-the-iphone-heres-how-apple-plans-to-protect-your-privacy). *ZDNet*. Retrieved 2017-06-27.
14. Smith, Chris (2016-06-13). "iOS 10: Siri now works in third-party apps, comes with extra AI features" (http://bgr.com/2016/06/13/ios-10-siri-third-party-apps/). *BGR*. Retrieved 2017-06-27.
15. Vogels, Werner (30 November 2016). "Bringing the Magic of Amazon AI and Alexa to Apps on AWS. - All Things Distributed" (http://www.allthingsdistributed.com/2016/11/amazon-ai-and-alexa-for-all-aws-apps.html). *www.allthingsdistributed.com*. Retrieved 2017-06-27.
16. Haridy, Rich (August 21, 2017). "Microsoft's speech recognition system is now as good as a human" (http://newatlas.com/microsoft-speech-recognition-equals-humans/50999). *newatlas.com*. Retrieved 2017-08-27.
17. Gers, F. A.; Schmidhuber, J. (2001). "LSTM Recurrent Networks Learn Simple Context Free and Context Sensitive Languages" (ftp://ftp.idsia.ch/pub/juergen/L-IEEE.pdf) (PDF). *IEEE Transactions on Neural Networks*. **12** (6): 1333–1340. doi:10.1109/72.963769 (https://doi.org/10.1109%2F72.963769).
18. Gers, F.; Schraudolph, N.; Schmidhuber, J. (2002). "Learning precise timing with LSTM recurrent networks" (http://www.jmlr.org/papers/volume3/gers02a/gers02a.pdf) (PDF). *Journal of Machine Learning Research*. **3**: 115–143.
19. Klaus Greff; Rupesh Kumar Srivastava; Jan Koutník; Bas R. Steunebrink; Jürgen Schmidhuber (2015). "LSTM: A Search Space Odyssey". *IEEE Transactions on Neural Networks and Learning Systems*. **28** (10): 2222. arXiv:1503.04069 (https://arxiv.org/abs/1503.04069). doi:10.1109/TNNLS.2016.2582924 (https://doi.org/10.1109%2FTNNLS.2016.2582924).
20. Gers, F. A.; Schmidhuber, E. (November 2001). "LSTM recurrent networks learn simple context-free and context-sensitive languages" (ftp://ftp.idsia.ch/pub/juergen/L-IEEE.pdf) (PDF). *IEEE Transactions on Neural Networks*. **12** (6): 1333–1340. doi:10.1109/72.963769 (https://doi.org/10.1109%2F72.963769). ISSN 1045-9227 (https://www.worldcat.org/issn/1045-9227).
21. Xingjian Shi; Zhourong Chen; Hao Wang; Dit-Yan Yeung; Wai-kin Wong; Wang-chun Woo (2015). "Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting". *Proceedings of the 28th International Conference on Neural Information Processing Systems*: 802–810. arXiv:1506.04214 (https://arxiv.org/abs/1506.04214). Bibcode:2015arXiv150604214S (http://adsabs.harvard.edu/abs/2015arXiv150604214S).
22. S. Hochreiter. Untersuchungen zu dynamischen neuronalen Netzen. Diploma thesis, Institut f. Informatik, Technische Univ. Munich, 1991.
23. Hochreiter, S.; Bengio, Y.; Frasconi, P.; Schmidhuber, J. (2001). "Gradient Flow in Recurrent Nets: the Difficulty of Learning Long-Term Dependencies (PDF Download Available)" (https://www.researchgate.net/publication/2839938_Gradient_Flow_in_Recurrent_Nets_the_Difficulty_of_Learning_Long-Term_Dependencies). In Kremer and, S. C.; Kolen, J. F. *A Field Guide to Dynamical Recurrent Neural Networks*. *ResearchGate*. IEEE Press. Retrieved 2017-06-27.
24. Schmidhuber, J.; Wierstra, D.; Gagliolo, M.; Gomez, F. (2007). "Training Recurrent Networks by Evolino". *Neural Computation*. **19** (3): 757–779. doi:10.1162/neco.2007.19.3.757 (https://doi.org/10.1162%2Fneco.2007.19.3.757).

25. Fernández, Santiago; Graves, Alex; Schmidhuber, Jürgen (2007). "Sequence labelling in structured domains with hierarchical recurrent neural networks". *Proc. 20th Int. Joint Conf. on Artificial In℡ligence, Ijcai 2007*: 774–779. CiteSeerX 10.1.1.79.1887 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.79.1887).
26. Graves, Alex; Fernández, Santiago; Gomez, Faustino (2006). "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks". *In Proceedings of the International Conference on Machine Learning, ICML 2006*: 369–376. CiteSeerX 10.1.1.75.6306 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.75.6306).
27. Mayer, H.; Gomez, F.; Wierstra, D.; Nagy, I.; Knoll, A.; Schmidhuber, J. (October 2006). "A System for Robotic Heart Surgery that Learns to Tie Knots Using Recurrent Neural Networks" (http://ieeexplore.ieee.org/document/4059310/). *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*: 543–548. doi:10.1109/IROS.2006.282190 (https://doi.org/10.1109%2FIROS.2006.282190). ISBN 1-4244-0258-1.
28. Wierstra, Daan; Schmidhuber, J.; Gomez, F. J. (2005). "Evolino: Hybrid Neuroevolution/Optimal Linear Search for Sequence Learning" (https://www.academia.edu/5830256/Evolino_Hybrid_Neuroevolution_Optimal_Linear_Search_for_Sequence_Learning). *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI), Edinburgh*: 853–858.
29. Graves, A.; Schmidhuber, J. (2005). "Framewise phoneme classification with bidirectional LSTM and other neural network architectures". *Neural Networks*. **18** (5–6): 602–610. doi:10.1016/j.neunet.2005.06.042 (https://doi.org/10.1016%2Fj.neunet.2005.06.042).
30. Fernández, Santiago; Graves, Alex; Schmidhuber, Jürgen (2007). "An Application of Recurrent Neural Networks to Discriminative Keyword Spotting" (http://dl.acm.org/citation.cfm?id=1778066.1778092). *Proceedings of the 17th International Conference on Artificial Neural Networks*. ICANN'07. Berlin, Heidelberg: Springer-Verlag: 220–229. ISBN 3540746935.
31. Graves, Alex; Mohamed, Abdel-rahman; Hinton, Geoffrey (2013). "Speech Recognition with Deep Recurrent Neural Networks". *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*: 6645–6649.
32. Eck, Douglas; Schmidhuber, Jürgen (2002-08-28). "Learning the Long-Term Structure of the Blues" (https://link.springer.com/chapter/10.1007/3-540-46084-5_47). *Artificial Neural Networks — ICANN 2002*. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg. **2415**: 284–289. doi:10.1007/3-540-46084-5_47 (https://doi.org/10.1007%2F3-540-46084-5_47). ISBN 3540460845.
33. Schmidhuber, J.; Gers, F.; Eck, D.; Schmidhuber, J.; Gers, F. (2002). "Learning nonregular languages: A comparison of simple recurrent networks and LSTM". *Neural Computation*. **14** (9): 2039–2041. doi:10.1162/089976602320263980 (https://doi.org/10.1162%2F089976602320263980).
34. Perez-Ortiz, J. A.; Gers, F. A.; Eck, D.; Schmidhuber, J. (2003). "Kalman filters improve LSTM network performance in problems unsolvable by traditional recurrent nets". *Neural Networks*. **16** (2): 241–250. doi:10.1016/s0893-6080(02)00219-8 (https://doi.org/10.1016%2Fs0893-6080%2802%2900219-8).
35. A. Graves, J. Schmidhuber. Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks. Advances in Neural Information Processing Systems 22, NIPS'22, pp 545–552, Vancouver, MIT Press, 2009.
36. Graves, Alex; Fernández, Santiago; Liwicki, Marcus; Bunke, Horst; Schmidhuber, Jürgen (2007). "Unconstrained Online Handwriting Recognition with Recurrent Neural Networks" (http://dl.acm.org/citation.cfm?id=2981562.2981635). *Proceedings of the 20th International Conference on Neural Information Processing Systems*. NIPS'07. USA: Curran Associates Inc.: 577–584. ISBN 9781605603520.
37. M. Baccouche, F. Mamalet, C Wolf, C. Garcia, A. Baskurt. Sequential Deep Learning for Human Action Recognition. 2nd International Workshop on Human Behavior Understanding (HBU), A.A. Salah, B. Lepri ed. Amsterdam, Netherlands. pp. 29–39. Lecture Notes in Computer Science 7065. Springer. 2011
38. Huang, Jie; Zhou, Wengang; Zhang, Qilin; Li, Houqiang; Li, Weiping (2018-01-30). "Video-based Sign Language Recognition without Temporal Segmentation" (https://arxiv.org/pdf/1801.10111.pdf) (PDF).
39. Hochreiter, S.; Heusel, M.; Obermayer, K. (2007). "Fast model-based protein homology detection without alignment". *Bioinformatics*. **23** (14): 1728–1736. doi:10.1093/bioinformatics/btm247 (https://doi.org/10.1093%2Fbioinformatics%2Fbtm247). PMID 17488755 (https://www.ncbi.nlm.nih.gov/pubmed/17488755).

40. Thireou, T.; Reczko, M. (2007). "Bidirectional Long Short-Term Memory Networks for predicting the subcellular localization of eukaryotic proteins". *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*. **4** (3): 441–446. doi:10.1109/tcbb.2007.1015 (https://doi.org/10.1109%2Ftcbb.2007.1015).
41. Malhotra, Pankaj; Vig, Lovekesh; Shroff, Gautam; Agarwal, Puneet (April 2015). "Long Short Term Memory Networks for Anomaly Detection in Time Series" (https://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2015-56.pdf) (PDF). *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning — ESANN 2015*.
42. Tax, N.; Verenich, I.; La Rosa, M.; Dumas, M. (2017). "Predictive Business Process Monitoring with LSTM neural networks" (https://link.springer.com/chapter/10.1007/978-3-319-59536-8_30). *Proceedings of the International Conference on Advanced Information Systems Engineering (CAiSE)*: 477–492. arXiv:1612.02130 (https://arxiv.org/abs/1612.02130). doi:10.1007/978-3-319-59536-8_30 (https://doi.org/10.1007%2F978-3-319-59536-8_30).
43. Choi, E.; Bahadori, M.T.; Schuetz, E.; Stewart, W.; Sun, J. (2016). "Doctor AI: Predicting Clinical Events via Recurrent Neural Networks" (http://proceedings.mlr.press/v56/Choi16.html). *Proceedings of the 1st Machine Learning for Healthcare Conference*: 301–318.
44. Jia, Robin; Liang, Percy (2016-06-11). "Data Recombination for Neural Semantic Parsing". *arXiv:1606.03622 [cs]*.
45. Wang, Le; Duan, Xuhuan; Zhang, Qilin; Niu, Zhenxing; Hua, Gang; Zheng, Nanning (2018-05-22). "Segment-Tube: Spatio-Temporal Action Localization in Untrimmed Videos with Per-Frame Segmentation" (https://qilin-zhang.github.io/_pages/pdfs/Segment-Tube_Spatio-Temporal_Action_Localization_in_Untrimmed_Videos_with_Per-Frame_Segmentation.pdf) (PDF). *Sensors*. MDPI AG. **18** (5): 1657. doi:10.3390/s18051657 (https://doi.org/10.3390%2Fs18051657). ISSN 1424-8220 (https://www.worldcat.org/issn/1424-8220).
46. Duan, Xuhuan; Wang, Le; Zhai, Changbo; Zheng, Nanning; Zhang, Qilin; Niu, Zhenxing; Hua, Gang (2018). *Joint Spatio-Temporal Action Localization in Untrimmed Videos with Per-Frame Segmentation*. 25th IEEE International Conference on Image Processing (ICIP). doi:10.1109/icip.2018.8451692 (https://doi.org/10.1109%2Ficip.2018.8451692). ISBN 978-1-4799-7061-2.
47. [Gers, Felix A., Nicol N. Schraudolph, and Jürgen Schmidhuber. "Learning precise timing with LSTM recurrent networks." Journal of machine learning research 3, no. Aug (2002): 115-143.]

Retrieved from "https://en.wikipedia.org/w/index.php?title=Long_short-term_memory&oldid=869039504"