A review on the long short-term memory model

VAN HOUDT, Greg; Mosquera, Carlos & NAPOLES RUIZ, Gonzalo (2020) A review on the long short-term memory model. In: ARTIFICIAL INTELLIGENCE REVIEW, 53, p. 5929-5955.

DOI: 10.1007/s10462-020-09838-1
Handle: http://hdl.handle.net/1942/31139

Noname manuscript No.
(will be inserted by the editor)

A Review on the Long Short-Term Memory Model

Greg Van Houdt · Carlos Mosquera · Gonzalo Nápoles

Received: date / Accepted: date

Abstract Long Short-Term Memory (LSTM) has transformed both machine learning and neurocomputing fields. According to several online sources, this model has improved Google's speech recognition, greatly improved machine translations on Google Translate, and the answers of Amazon's Alexa. This neural system is also employed by Facebook, reaching over 4 billion LSTM-based translations per day as of 2017. Interestingly, recurrent neural networks had shown a rather discrete performance until LSTM showed up. One reason for the success of this recurrent network lies in its ability to handle the exploding / vanishing gradient problem, which stands as a difficult issue to be circumvented when training recurrent or very deep neural networks. In this paper, we present a comprehensive review that covers LSTM's formulation and training, relevant applications reported in the literature and code resources implementing this model for a toy example.

Keywords Recurrent Neural Networks · Vanishing / exploding Gradient · Long Short-Term Memory · Deep Learning

G. Van Houdt, G. Nápoles
Faculty of Business Economics,
Hasselt University, Belgium
Agoralaan gebouw D, 3590 Diepenbeek
E-mail: greg.vanhoudt@uhasselt.be

C. Mosquera
Artificial Intelligence Lab,
Vrije Universiteit Brussel, Belgium
Pleinlaan 9, 1050 Brussels

G. Nápoles
Department of Cognitive Science & Artificial Intelligence,
Tilburg University, The Netherlands
Warandelaan 2, 5037 AB Tilburg, The Netherlands
E-mail: g.r.napoles@tilburguniversity.edu

1 Introduction

Recurrent or very deep neural networks are difficult to train, as they often suffer from the exploding / vanishing gradient problem [46, 47]. To overcome this shortcoming when learning long-term dependencies, the LSTM architecture [48] was introduced. The learning ability of LSTM impacted several fields from both a practical and theoretical perspective, so that it became a state-of-the-art model. This led to the model being used by Google for its speech recognition [109], and to improve machine translations on Google Translate [88, 143]. Amazon employs the model to improve Alexa's functionalities [133], and Facebook puts it to use for over 4 billion LSTM-based translations per day as of 2017 [97].

Due to its high applicability and popularity, this neural architecture has also found its way into the world of gaming. For example, Google's Deepmind created AlphaStar [127], an artificial intelligence designed to play Starcraft II. Throughout the development of AlphaStar, it started to master the game [126], climbing up the global rankings, which was unseen before. Research in this field is of course not limited to Starcraft II, as the research interest spans the entire RTS gaming genre due to its complexity [158]. To generalise the topic of reinforcement learning in other settings, OpenAI succeeded in building a robot hand called Dactyl, which taught itself how to manipulate objects in a human-like fashion [105].

Of course, a neural architecture would not be so well adopted into practice without a strong theoretical foundation. An extensive review with respect to several LSTM variants and their performances relative to the so-called vanilla model was recently conducted by Greff et al. [42]. The vanilla LSTM is interpreted as the original LSTM block with the addition of the forget gate and peephole connections. In total, eight variants were identified for experimentation. In a nutshell, the vanilla architecture performs well on a number of tasks, and none of the eight investigated variants significantly outperforms the remaining ones. This justified most applications found in the literature to employ the vanilla LSTM.

A recent study conducted by Yu et al. [151] provides an overview of the LSTM cell, its functionalities and different architectures. A distinction is made between LSTM-dominated neural networks and integrated LSTM networks, the latter of which adds other components than LSTM to take advantage of its properties, therefore potentially hybridising neural networks. As will be illustrated in Section 4, this work complements the review study of Yu et al. [151] in terms of a broad applications overview showing where integrated networks are most useful. As each problem is largely unique, there is often a better solution than employing solely the standard LSTM model.

Therefore, in this paper, we present a comprehensive review on the LSTM model that complements the theoretical findings presented in [42, 151]. Our review study focuses on three main directions, which move from theory to practice. In the first part, we broadly describe the LSTM components, how they interact with each other and how we can estimate the learnable parameters. These considerations may become relevant for readers who want to master the model from a theoretical perspective, instead of being experienced practitioners. In the second part, we outline interesting applications that show the potential of LSTM as an undeniable state-of-the-art method within the deep learning field. Some interesting application domains include text recognition, time series forecasting, natural language processing, computer vision, and image and video captioning, among

others. In the last part, we present a code example in Tensorflow that aims to predict the next word of a sample short story.

As for the search criteria on which this literature review is based, we evaluated 409 papers containing the terms "Long short-term memory" or "LSTM" in either the title, abstract or keywords. Only non-paid journals with a recognised peer-review system were considered. Moreover, we included papers presented in mainstream conferences such as the *Conference on Neural Information Processing Systems*, the *Conference on Computer Vision and Pattern Recognition*, the *AAAI Conference on Artificial Intelligence*, the *Conference on Empirical Methods in Natural Language*, etc. Having constructed our starting library, we selected papers making a relevant contribution in terms of theory or practice in which LSTM played a key role. It does not imply however that papers not included in our review do not fulfil this criterion, but certainly we could not cover the whole literature concerning the LSTM model. For example, a rough search carried out in February 2020 by using the search criteria mentioned above reported 11,931 documents indexed by Scopus. Therefore, we have given priority to most recent contributions.

The remainder of this paper is structured as follows. In Section 2 we describe the theoretical foundations behind the LSTM model, followed by a concise description of a procedure to adjust the learnable parameters in Section 3. Section 4 zooms in on different applications of the model as found in the literature. An example implementation in Tensorflow can be found in Section 5. Finally, we summarise our conclusions in Section 6.

2 Long Short-Term Memory

The LSTM model [48] is a powerful recurrent neural system specially designed to overcome the exploding / vanishing gradient problems that typically arise when learning long-term dependencies, even when the minimal time lags are very long [49]. Overall, this can be prevented by using a *constant error carousel* (CEC), which maintains the error signal within each unit's cell. As a matter of fact, such cells are recurrent networks themselves, with an interesting architecture in the way that the CEC is extended with additional features, namely the input gate and output gate, forming the memory cell. The self-recurrent connections indicate feedback with a lag of one time step.

A vanilla LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. This forget gate was not initially a part of the LSTM network, but was proposed by Gers et al. [33] to allow the network to reset its state. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information associated with the cell. In the remainder of this section, LSTM will refer to the vanilla version as this is the most popular LSTM architecture [42]. This does not imply, however, that it is also the superior one in every situation. This will be elaborated on in Section 4.

In short, the LSTM architecture consists of a set of recurrently connected sub-networks, known as memory blocks. The idea behind the memory block is to maintain its state over time and regulate the information flow thought non-linear gating units. Fig. 1 displays the architecture of a vanilla LSTM block, which involves the gates, the input signal $x^{(t)}$, the output $y^{(t)}$, the activation functions,

and peephole connections [32]. The output of the block is recurrently connected back to the block input and all of the gates.

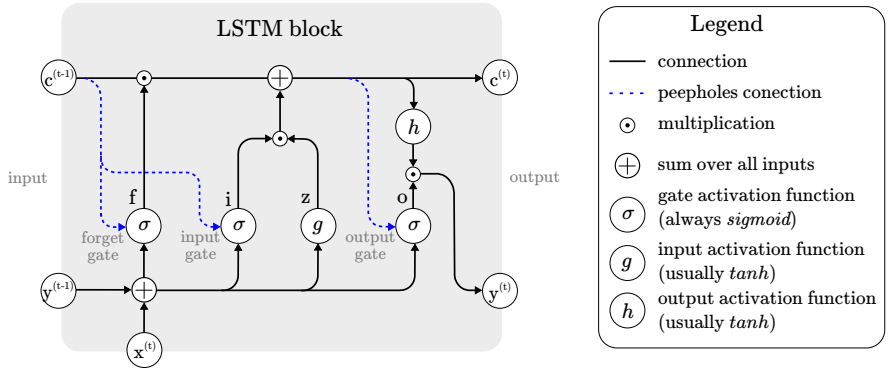

Figure 1 Architecture of a typical vanilla LSTM block.

Aiming to clarify how the LSTM model works, let us assume a network comprised of N processing blocks and M inputs. The forward pass in this recurrent neural system is described below.

Block input. This step is devoted to updating the block input component, which combines the current input $x^{(t)}$ and the output of that LSTM unit $y^{(t-1)}$ in the last iteration. This can be done as depicted below:

$$z^{(t)} = g(W_z x^{(t)} + R_z y^{(t-1)} + b_z) \tag{1}$$

where W_z and R_z are the weights associated with $x^{(t)}$ and $y^{(t-1)}$, respectively, while b_z stands for the bias weight vector.

Input gate. In this step, we update the input gate that combines the current input $x^{(t)}$, the output of that LSTM unit $y^{(t-1)}$ and the cell value $c^{(t-1)}$ in the last iteration. The following equation shows this procedure:

$$i^{(t)} = \sigma(W_i x^{(t)} + R_i y^{(t-1)} + p_i \odot c^{(t-1)} + b_i) \tag{2}$$

where $\odot$ denotes point-wise multiplication of two vectors, W_i, R_i and p_i are the weights associated with $x^{(t)}$, $y^{(t-1)}$ and $c^{(t-1)}$, respectively, while b_i represents for the bias vector associated with this component.

In the previous steps, the LSTM layer determines which information should be retained in the network's cell states $c^{(t)}$. This included the selection of the candidate values $z^{(t)}$ that could potentially be added to the cell states, and the activation values $i^{(t)}$ of the input gates.

Forget gate. In this step, the LSTM unit determines which information should be removed from its previous cell states $c^{(t-1)}$. Therefore, the activation values $f^{(t)}$ of the forget gates at time step t are calculated based on the current input $x^{(t)}$, the outputs $y^{(t-1)}$ and the state $c^{(t-1)}$ of the memory cells at the previous time step $(t-1)$, the peephole connections, and the bias terms b_f of the forget gates. This can be done as follows:

$$f^{(t)} = \sigma(W_f x^{(t)} + R_f y^{(t-1)} + p_f \odot c^{(t-1)} + b_f) \tag{3}$$

where W_f, R_f and p_f are the weights associated with $x^{(t)}$, $y^{(t-1)}$ and $c^{(t-1)}$, respectively, while b_f denotes for the bias weight vector.

Cell. This step computes the cell value, which combines the block input $z^{(t)}$, the input gate $i^{(t)}$ and the forget gate $f^{(t)}$ values, with the previous cell value. This can be done as depicted below:

$$c^{(t)} = z^{(t)} \odot i^{(t)} + c^{(t-1)} \odot f^{(t)}. \tag{4}$$

Output gate. This step calculates the output gate, which combines the current input $x^{(t)}$, the output of that LSTM unit $y^{(t-1)}$ and the cell value $c^{(t-1)}$ in the last iteration. This can be done as depicted below:

$$o^{(t)} = \sigma(W_o x^{(t)} + R_o y^{(t-1)} + p_o \odot c^{(t)} + b_o) \tag{5}$$

where W_o, R_o and p_o are the weights associated with $x^{(t)}$, $y^{(t-1)}$ and $c^{(t-1)}$, respectively, while b_o denotes for the bias weight vector.

Block output. Finally, we calculate the block output, which combines the current cell value $c^{(t)}$ with the current output gate value as follows:

$$y^{(t)} = g(c^{(t)}) \odot o^{(t)}. \tag{6}$$

In the above steps, σ, g and h denote point-wise non-linear activation functions. The logistic sigmoid $\sigma(x) = \frac{1}{1+e^{1-x}}$ is used as a gate activation function, while the hyperbolic tangent $g(x) = h(x) = \tanh(x)$ is often used as the block input and output activation function.

It seems appropriate to mention that the functionality of this architecture inspired the authors in [66] to enhance the training of very deep networks. The gating mechanism was employed in the so-called highway networks to allow for an unimpeded information flow across many layers. This could be considered as another proof-of-concept, showing that the gates work.

Even though vanilla LSTM already performs very well, several works studied the possibilities to improve performance. For example, Su and Kuo [119] developed the Extended LSTM model, further improving the accuracy of predictions in several application fields by enhancing the memory capability. This shows that theoretical improvements can still be made to an already state-of-the-art performing architecture. In the work of Bayer et al. [7], the search for improvements of the model was already ongoing. The authors looked for for an architectural alternative to LSTM to optimise sequence learning capabilities. They succeeded in evolving memory cell structures capable of learning context-sensitive formal languages through gradient descent, being in many ways comparable to LSTM performance-wise. The authors in [8] built upon recurrent networks of spiking neurons, developing Long short-term memory Spiking Neural Networks (LSNN) including adapting neurons. During tests in which the size of LSNN was comparable to that of LSTM, it was shown that the performance is very comparable to that of LSTM. This is yet another illustration of how accurate LSTM is and remains.

3 How to train your model

The LSTM model described in Section 2 uses full gradient training presented by Graves and Schmidhuber [39] to adjust the learnable parameters (weights) involved in the network. More specifically, *Backpropagation Through Time* [140] is used to compute the weights that connect the different components in the network. Therefore, during the backward pass, the cell state $c^{(t)}$ receives gradients from $y^{(t)}$ as well as the next cell state $c^{(t+1)}$. Those gradients are accumulated before being backpropagated to the current layer.

In the last iteration T, the change $\delta_y^{(T)}$ corresponds with the network error $\partial E/y^{(T)}$ such that E denotes the loss function. Otherwise, $\delta_y^{(t)}$ is the vector of delta values passed down $\Delta^{(t)}$ from the above layer including the recurrent dependencies. This can be done as follows:

$$\delta_y^{(t)} = \Delta^{(t)} + R_z^T \delta_z^{(t+1)} + R_i^T \delta_i^{(t+1)} + R_f^T \delta_f^{(t+1)} + R_o^T \delta_o^{(t+1)}. \tag{7}$$

In a second step, the change in the parameters associated with the gates and the memory cell are calculated as:

$$\delta_{o^{(t)}} = \delta_y^{(t)} \odot h(c^{(t)}) \odot \sigma'(\hat{o}^{(t)}) \tag{8}$$

$$\begin{aligned} \delta_c^{(t)} = \delta_y^{(t)} \odot o^{(t)} \odot h'(c^{(t)}) + p_o \odot \delta_o^{(t)} + \\ p_i \odot \delta_i^{(t+1)} + p_f \odot \delta_f^{(t+1)} + \delta_c^{(t+1)} \odot f^{(t+1)} \end{aligned} \tag{9}$$

$$\delta_{f^{(t)}} = \delta_c^{(t)} \odot c^{(t-1)} \odot \sigma'(\hat{f}^{(t)}) \tag{10}$$

$$\delta_{i^{(t)}} = \delta_c^{(t)} \odot z^{(t)} \odot \sigma'(\hat{i}^{(t)}) \tag{11}$$

$$\delta_{z^{(t)}} = \delta_c^{(t)} \odot i^{(t)} \odot g'(\hat{z}^{(t)}) \tag{12}$$

where $\hat{o}^{(t)}$, $\hat{i}^{(t)}$, $\hat{z}^{(t)}$ and $\hat{f}^{(t)}$ denote the *raw* values attached with the output gate, input gate, the block input and forget gate, respectively, before being transformed by the corresponding transfer function.

As pointed out in [42], the delta values for the inputs are only required if there is a layer below that needs to be trained, thus:

$$\delta_x^{(t)} = W_z^T \delta_z^{(t)} + W_i^T \delta_i^{(t)} + W_f^T \delta_f^{(t)} + W_o^T \delta_o^{(t)}. \tag{13}$$

Finally, the gradients for the weights are calculated as follows:

$$\delta_{W_*} = \sum_{t=0}^{T} \delta_*^{(t)} \otimes x^{(t)} \qquad \delta_{p_i} = \sum_{t=0}^{T-1} c^{(t)} \odot \delta_i^{(t+1)}$$

$$\delta_{R_*} = \sum_{t=0}^{T-1} \delta_*^{(t+1)} \otimes y^{(t)} \qquad \delta_{p_f} = \sum_{t=0}^{T-1} c^{(t)} \odot \delta_f^{(t+1)}$$

$$\delta_{b_*} = \sum_{t=0}^{T} \delta_*^{(t)} \qquad \delta_{p_o} = \sum_{t=0}^{T} c^{(t)} \odot \delta_o^{(t)}$$

such that $\otimes$ represents the outer product of two vectors, whereas $*$ can be any component associated with the weights: the block input $\hat{z}$, the input gate $\hat{i}$, the forget gate $\hat{f}$ or the output gate $\hat{o}$.

Alternatively, one can train the model with evolino [112, 141]. This method evolves weights to the nonlinear, hidden nodes of recurrent neural networks while computing optimal linear mappings from hidden state to output.

4 Relevant Applications

The LSTM network is applied in a wide array of problem domains, both individually and in combination with other deep learning architectures. As previously discussed, LSTM is one of the most advanced networks to process temporal sequences. For this reason, the vanilla LSTM is still one of the most popular network choices, even though it is possible to combine it with other networks to create hybrid models. LSTM is well suited to handle time series predictions, but also any other problem that requires temporal memory.

This section gives an overview of the topics the model is most suitable for and how it can be used to solve complex problems. In that regard, we will discuss the applications per problem domain.

4.1 Time Series Prediction

When it comes to temporal sequences in data, time series data immediately comes to mind. Still, this is a broad concept. In the more literal sense of time series predictions, the LSTM model has been applied to financial market predictions in, for example, Fisher and Krauss [29] and Yan and Ouyang [146]. Due to complex features such as non-linearity, non-stationarity and sequence correlation, financial data pose a huge forecasting challenge. It was shown by Fischer and Krauss, however, that the LSTM network outperforms more traditional benchmarks: the random forest, a standard deep neural network and a standard logistic regression.

Sagheer and Kotb [108] confirmed this finding that LSTM outperforms standard approaches when predicting petroleum production. In their research, they stacked multiple layers of LSTM blocks on top of one another in a hierarchical fashion. This increased the model's ability to process temporal tasks and enabled it to better capture the structure of data sequences. Attempting to forecast the oil market price, the authors in [13] came to the same conclusions. The proposed prediction model, composed of the vanilla LSTM architecture, proved to be superior. Liu [77] compared LSTM with other models, like support vector machines. This research in estimating financial stock volatility not only showed it was relatively easy to calibrate LSTM, but also that it results in accurate predictions for even large time intervals.

In [103] the authors used LSTM to model the time series observations and later on improve predictions by combining this with text data input. Using this technique, they attempted to predict taxi demand in New York, even though the proposed method is generalisable for other applications as well. This is an important benefit of the architecture, as it was empirically shown that the forecast error can be greatly reduced with this approach.

Receiving a time series as input does not necessarily mean the model will predict the next values in the series, as it can also be used to train a classifier. This is the case of the fault diagnosis in [68] where the authors used an LSTM-based framework for condition monitoring of wind turbines using only raw time series data gathered by a single or multiple sensors. They explicitly made this choice to avoid a heavy reliance on expert knowledge. LSTM was utilised to capture long-term dependencies in the data to do proper fault classification. Experiments in this study have shown that this framework is not only robust, but it also outperformed the state-of-the-art methods. Other fault classification works include batteries in electric vehicles [51], where LSTM predicts the battery voltage levels which lays the ground work for fault classification, and electro-mechanical actuators in aircraft [147] based on the recorded sensor data. The study of Saeed et al. [107] presented the necessary steps and process required to create a flexible fault diagnosis model, in which LSTM plays a pivotal role in the statistical analysis. The authors tested their model in the nuclear energy domain.

As another LSTM-based classification example, Uddin [130] demonstrated that the model can also detect which activity was performed given input data from multiple wearable healthcare sensors obtained via an edge device like a laptop. This sensor data was fed to LSTM with which twelve different human activities were modelled. Again, the proposed method was proven to be robust and achieve better results than the reported traditional models.

Given several data points produced by a number of sensors, Elsheikh et al. [23] predicted the remaining useful life (RUL) of physical systems, for example, production resources. This was done by proposing a new bidirectional LSTM (BLSTM) architecture. The term *bidirectional* in recurrent neural networks comes from the idea to provide the input sequence in both ways: forwards and backwards. From these two hidden layers, the output layer can get information about the past and the future states, respectively, simultaneously [113]. The same concept can be applied to the LSTM network, as it is a recurrent network that takes sequences as input. Graves and Schmidhuber [40] presented the BLSTM network, comparing it with the unidirectional variant in a classification context. It was already confirmed that LSTM outperforms traditional recurrent neural networks, but the findings from this research indicated that BLSTM can achieve even better results.

Since the problem at hand in [23] is only about the RUL, the authors were not interested in intermediate predictions. Therefore, the input sequence is not processed in both directions simultaneously. Rather, the forward direction is processed first, after which the LSTM final states initialise the backward processing cells. This forces the network to produce two different yet linked mappings of the data to the RUL. Since the initial wear of the physical system is usually unknown, the network is trained to anticipate requirements such as replacement of parts. This method outperforms the conventional network types according to the conducted experiments.

Li et al. [73] presented a lithium-ion battery RUL prediction based on the empirical mode decomposition algorithm in combination with a novel Elman-LSTM hybrid network. First, empirical mode decomposition extracts both high and low frequency signals from the input data. The Elman neural network was introduced to the solution as it has the capability of short-term memory [71]. Therefore, it is included to tackle the problem of capturing capacity recovery in certain cycles of the batteries. With the long-term capabilities of LSTM, the battery degradation

is regarded as the time series. With both models working together, capturing both degradation and recovery characteristics, the battery RUL is predicted. Experiments show that this novel method offers a superior performance in comparison to other state-of-the-art models.

As shown in [73], predictions can be improved by combining different model architectures. However, not only can these models work together side by side, the input sequence can also be preprocessed for LSTM by another deep learning or other architecture, also forming a hybrid model. Wu et al. [144] used ensemble empirical mode decomposition to select the proper intrinsic mode functions which then serve as input for the LSTM model to predict crude oil price movements. This proved to be a very effective methodology, even when the number of decomposition results varies.

As the reader can notice from the applications above, LSTM is an appropriate model to deal with time series data. But these are of course not all contributions which can be found in the literature. Many other applications exist for time series prediction scenarios, like emotion ratings [102], topic evolution in text streams [80], building energy usage [136], carbon emission levels [56], flight delays [87], financial trading strategy selection [110], air quality monitoring [159], wind speed forecasting [18], precipitation nowcasting [145], real-time occupancy prediction [63] and health predictive analytics [86]. The LSTM's prediction power speaks for the wide usability of the model to overcome problems in which other recurrent neural systems are likely to fail.

4.2 Natural Language Processing

LSTM is a force to be reckoned with when it comes to language learning, both context-free and context-sensitive [34, 35]. Natural language processing is the field of research that explores how computers can be used to understand and manipulate natural language text or speech to do useful things [19]. For example, dialog systems – also known as conversational agents – allow human beings to interact with a machine via speech. Speech recognition with the use of the LSTM model was first performed in [37] as it has the main benefit of dealing with long time lags. Results comparable to the hidden Markov model (HMM) [100] were obtained in this experiment. This work then initiated the further exploration of LSTM for this domain.

Dialog systems should be able to respond to *out-of-domain* speech input, instead of giving a random response. In [104] a binary classifier was built using two LSTM layers. The classifier was trained with only *in-domain* data, with the goal to later on recognise *out-of-domain* sentences. This method achieved higher prediction rates than the former state-of-the-art models.

Of course, while identifying differences between sentences is interesting from the classification perspective, understanding the meaning of the sentence can be quite a challenge for machines. Take the following sentence as an example: "*Apple will launch a trio of new iPads this spring, according to Barclays research analysts.*" Given the context provided in the sentence, algorithms should be able to understand that "*Apple*" refers to the company, not the fruit. This is the purpose of entity disambiguation. The architecture in Sun et al. [122] was comprised of two LSTM networks to learn the context of the sentences, which performed well.

However, this already strong baseline was outperformed by the memory network proposed in [123].

If the context of sentences and words is understood, then questions can be answered about them. One example of this application is visual question answering, a computer vision task where a system is provided with a text-based question about an image and the answer must be inferred [31, 58]. Section 4.4 elaborates on other computer vision applications. Alternatively, community question answering – selecting the correct answer to a question – can be performed. The authors in [139] built a hybrid attention network which considers the importance of words in the current sentence but also the mutual importance with respect to the counterpart sentence for sentence representation learning. The representations of the questions and answers are learned with an LSTM-based architecture.

Given the model's capability to remember long-term contexts, LSTM can be used to detect dialogue breakdowns. Conversational agents have to timely detect such a breakdown as it helps the agents recover from mistakes. A recent contribution from Takayama et al. [125] investigated variations when determining if a response causes breakdowns in a conversation (subjectivity), and variations in breakdown types due to designs of agents (variationality). The variationality was addressed with three models: LSTM which considers the global context, the convolutional neural network (CNN) which is sensitive to the local characteristics, and also the combination of both. The authors did investigate BLSTM as well, but chose to employ unidirectional LSTM since results were comparable.

On the other hand, Hori et al. [52] did adopt both the uni- and bidirectional networks, along with a hierarchical recurrent encoder decoder. The responses given by these three models are combined by a minimum Bayes risk based system in order to go to the generation phase of system responses in a Twitter help-desk dialog task. Note that this proposal is not for a real-time dialog system as it is impossible to feed the input in a reverse order in this case.

Building a dialog system is, as expected, also possible with BLSTM. As with text recognition, the words in a sentence that follow on each other are not only dependent on the previous one, it can also be related with the next one. The authors in [62] employed the MemN2N architecture [120] to perform automatic system responses, but added BLSTM to the beginning of their network to better reflect the temporal information. Numerical simulations showed that the performance of state-of-the-art methods for an end-to-end network is comparable to the hierarchical LSTM model. Just as in [52], this is not for a real-time dialog system.

Wang et al. [138] attempted to generate natural and informative responses for customer service oriented dialog systems. For this purpose, two frameworks were proposed: one to encode the entire dialogue history, while the other integrates external knowledge extracted from a search engine. It is in the former that the authors explored both CNN and LSTM networks. The simulation results showed that the recurrent network was more effective in improving the reply quality when compared with the CNN variant, which can only capture the problem semantics through short patterns.

Although the interaction with humans is pivotal in this kind of problems, it does not have to end with providing the human with appropriate responses or to provide the information the user is seeking. Opinions can be analysed and extracted from sentences as well, as done by Zhang et al. [154] who employed a multilayered BLSTM architecture. D'Andrea et al. [21] put the strengths of LSTM to use, in

combination with other architectures, to track opinions on vaccination. From this study, however, it seemed that LSTM was not amongst the best architectures to tackle the problem.

Likewise, text can be classified into certain categories, as done by Dabiri and Heaslip [20] with the use of CNN and LSTM, from which the contributors concluded that their approach reports superior results when compared with other algorithms. Combinatory Categorical Grammar supertagging can also be performed by means of deep learning architectures as illustrated in [57] where the authors used BLSTM and conditional random field. This hybrid architecture attained good results in a very efficient manner.

Liu et al. [78] employed both LSTM and CNN networks to build a rumor identification classifier in the social media environment. For this purpose, they performed forwarding comments analysis, they included the influence, authority and popularity of so-called influencers, and captured diffusion structures. Results show that the proposed models are quite capable of learning hidden clues and contextual information.

Of course, before any analysis can be conducted, the structure of the document must be machine-understandable, which is the domain of document representation, studied in [157]. First, an attention-based LSTM is used to generate hidden representations of word sequences. Next, a latent topic-modelling layer and tree-structured LSTM generate the semantic representations of the document. This method proved its value over the state-of-the-art.

Naturally, there are a great deal of works which contributed to the field of natural language processing using (a variant of) LSTM. Examples are Chinese word segmentation [36, 83], morphological segmentation [134], relation extraction [115], mapping natural text to knowledge base entities [61], emoji prediction [6], and translation tasks [124].

4.2.1 Sentiment Analysis

Sentiment analysis is closely related with natural language processing. Many data points can be used to detect emotions like physiological data, the environment, videos, etc. Kanjo et al. [60] put to use sensor signals coming from these multimodal data sources. More specifically, these signals originated from smartphones and wearable devices. In fact, they were the first to perform emotion recognition using physiological, environmental and location data. To analyse all the data, four models were built, all based on a CNN-LSTM architecture: the on-body data, the environmental data, the location data, and finally the fusion of all data inputs. The authors concluded that, by using this hybrid network, the accuracy level was increased by more than 20% compared to a traditional multi-layer perception model.

When it comes to video-based sentiment analysis, Li and Xu [70] introduced a new feature extraction method, hvnLBP-TOP, followed by a sequence learning module containing BLSTM. In short, they extracted from a video the different frames, in which human faces were detected to put together a human face video, a fixed-size picture sequence. From a comparison analysis with state-of-the-art models, this new method proved effective.

Identifying people's emotions can also help detect conversation anomalies. For this, Sun et al. [121] investigated the hybrid model of CNN-LSTM combined with

a Markov chain Monte Carlo method. The former is applied to identify the emotion of the conversation texts, while the latter detects the emotion transitions. A limitation of this research, however, is that the initial and stimulating emotion cannot be set at any time during the conversation without guiding it in a specific direction.

Krauss and Feuerriegel [65] proposed a method that builds upon the discourse structure of documents, namely a tensor-based, tree-structured deep neural network named Discourse-LSTM. The method is based on rhetorical structure theory which structures documents hierarchically, forming a discourse tree, which discourse-LSTM can process completely. The tensor structure reveals the salient text passages and thereby provides explanatory insights, all the while returning a superior performance.

In the work of Song et al. [116] a method of sentiment lexicon embedding in Korean was proposed. After extensive data preprocessing, attention-based LSTM was responsible for sentiment classification. The authors' approach resulted in improved accuracy of this classification. Zhou et al. [162] used an attention-based BLSTM to model bilingual texts with the goal of cross-language sentiment classification. The authors confirmed the attention mechanism proves to be very effective, especially on the word-level. A similar architecture was used in [148] with the goal of improving target-dependent sentiment classification, and achieving better or comparable results. The attention mechanism was also employed by Ma et al. [85], in their variants of LSTM, incorporating commonsense knowledge, which outperformed the competition in targeted aspect sentiment analysis.

The authors in [160] experimented with a one- and a two-dimensional CNN-LSTM architecture to identify emotions from speech data. The results show that the proposed methods achieve outstanding performance. Especially the two-dimensional variant outperformed the benchmark models: the deep belief networks and the CNN-based architectures. A similar study is performed by Huang et al. [55], with similar results: CNN-LSTM outperforms not only CNN, but also support vector machine predictions.

In speech data emotion recognition, Fayek et al. [25] conducted a review study to compare various deep learning architectures, both feed-forward and recurrent networks. In this study, CNN yielded the best accuracy. This is probably why most contributions use the hybrid network of both: to combine the strengths of both models.

4.3 Image and Video Captioning

So far, we have discussed basic time series predictions and natural language processing. However, a computer can be requested to describe what can be seen in an image or a video in a natural-like speech format. This is referred to as image and video captioning.

From the perspective of our literature study, we notice that this domain often operates with a CNN-LSTM hybrid architecture, thus following the encoder-decoder framework. For example, having a CNN feed a sequence of frames to LSTM layers to generate the word sequence, where [132] integrated linguistic knowledge from large text corpora.

The study of Chen et al. [15] built upon the state-of-the-art of computer vision to enable "robots to speak". In other words, the goal of this research was to feed the model with images of cars, as detection of cars is a hot topic in self-driving technology, which would then generate a textual description of the image in understandable human language. In this regard, a CNN was applied to extract car region proposals to embed them into fixed-size windows. The LSTM can then generate from the image input a one sentence description closely related to the input image with variable length words. Compared with four other relevant algorithms, the proposed model by Chen et al. [15] proved to be superior.

For more general image captioning, He et al. [44] attempted to exploit the structure information of a natural sentence. The CNN extracts from the image a high-level features representation. This vector described the global content of the input image. The LSTM network is then employed to generate words from the image representation in a recurrent process, guided by part of speech. A part of speech tagger is a system which automatically assigns the part of speech to words using contextual information [111]. The performance of LSTM in part of speech was studied in [98] and later confirmed in [53], thus concluding the superiority of LSTM in this domain.

To increase the fluidity and descriptive nature of the generated image captions, a deep network consisting of three steps was proposed in [64]. The first step in their proposed method is, of course, system preprocessing. For this purpose, the region proposal network is applied to generate regions of interest – the regions likely to contain objects or people. Second, to start with image description generation, the model conducts object and scene classification and attribute predictions. The third step is language "conversion". The generated attributes and class labels are converted into fully descriptive image captions. It is in these two final steps that LSTM plays a key role, as it is the language generating neural network.

Chen et al. [16] proposed a new reference-based LSTM model to caption images where the training images are considered to be the references. This way, the authors attempted to solve two problems, namely identifying the important words in a caption, and misrecognition of objects and scenes. Experiments show their approach offers superior performance.

Liu et al. [75] argued that, in order to perform proper video captioning, one should not just ignore the rich contextual information that is also available like objects, scenes, actions etc. In this work, LSTM was employed to first learn the multimodal and dynamic representation of the video. Next, the model is leveraged to generate the description words one by one.

To conclude this section, the study of Ren et al. [101] applied multimodal LSTM to perform speaker identification. This is the task of localising the face of a person corresponding to the identity of the ongoing the voice in a video. Therefore, a collective perception of both visuals and audio is required. The authors show that modelling the temporal dependency across face and voice can significantly improve the robustness. In the end, their system outperforms the state-of-the-art systems with both a lower false alarm rate and a higher recognition accuracy.

4.4 Computer Vision

LSTM can also be used for gesture and action recognition. This field includes identifying human poses and interactions. In [10] the authors investigated automatic recognition of mimicry behaviour, as mimicry has the power to influence social judgements and behaviours. Mimicry behaviour is here defined as face and head movements. Video recordings of mimicry changes are fed to the network and compared with other methods, namely cross-correlation and generalised time-warping. LSTM reported an outstanding performance due to the model's inherent ability to process spatio-temporal transformations. A significant variance in the performance was detected in these experiments, thus suggesting there was still room for improvement. Zhang et al. [153] explored the effects of attention in convolutional LSTM with regards to gesture recognition, and discovered that convolutional structures in the gates do not play the role of spatial attention. Instead, a reduction of these structures result in a better accuracy, a lower parameter size and a lower computational consumption. Therefore, they introduced a new variant of LSTM.

A similar study was conducted by Chen et al. [17]. The network architecture consists of a global LSTM network comprised of multiple blocks. The video sequence is passed to the network as a set of images, while the output consists of estimates of facial landmark coordinates of the corresponding image. As these coordinates highly correlate throughout the sequence, the output of one image is used as input for the next one. In this work, however, the global network only produces the initial coordinates. Such points are fine-tuned with two feed-forward deep neural networks, which increases the accuracy of the coordinates while maintaining the shape of the face.

Hou et al. [54] presented a facial landmark detection method for images and videos under uncontrolled conditions. This method is a unified framework which integrates, amongst others, LSTM to make full use of the spatial and temporal middle stage information to improve the accuracy. Based on experiments on publicly available datasets, the method proved to be more effective than the state-of-the-art approaches at that time.

LSTM can also recognise gestures made by hand and even track the entire human body. This is skeleton-based human activity tracking [92] where the authors used three-dimensional data sequences obtained from full-body and hand skeletons to address human activity and hand gesture recognition, respectively. To accomplish this, the hybrid model CNN-LSTM was utilised. Extensive simulations concluded that this hybrid model has a similar performance as state-of-the-art methods. Zhu et al. [163] also recognised the importance of skeleton joints as a good representation of the skeleton for describing actions. They introduced a new dropout algorithm which has proven its effectiveness in experiments. This dropout algorithm allowed the dropping of internal gates, cell and output responses for an LSTM neuron to encourage each unit to learn better parameters.

The applicability is not limited to tracking body characteristics, of course. The location of any arbitrary object in a sequence of frames can also be tracked given the initial position [14]. LSTM was employed to better keep track of the historical context while performing more reliable inferences in the current step. The authors fairly stated they did not propose a real-time algorithm in their research, as runtime speeds should be improved.

Humans also pass as objects for tracking in a sequence of frames. Pei et al. [95] proposed an LSTM-based model to predict human trajectories in crowded scenes, where one LSTM is used for each object. This model includes the assumption that humans adjust their paths to accommodate for other people's movements, as well as that of their partners. In other words, social interactions are also taken into account. However, the employed architecture failed to identify obstacles. In [1], on the other hand, the authors reported a superior performance of their human trajectory prediction study. The main difference is that Alahi et al. [1] only consider human interactions, as objects and the scene are completely left out of the equation. In this context, Zhang et al. [155] worked on joint trajectory predictions and proposed a new states refinement module for LSTM. A comparative analysis shows the effectiveness of this method.

In [137] spatio-temporal LSTM was introduced in order to predict the next images based on historical frames which did achieve state-of-the-art prediction performance. This was measured on three video prediction datasets. In the context of predicting future frames, Fan et al. [24] introduced cubic LSTM consisting of three branches, namely a spatial, a temporal and an output branch, the latter of which combines the first two branches to generate the predicted frames.

Similar to [95], the study from Li et al. [72] was performed in the context of intelligent surveillance. Firstly, a fixed-size window is slid on a static image to generate an image sequence. This sequence serves as input for a CNN which extracts a feature sequence from the image sequence. Finally, this feature sequence is passed in proper order to memorise and recognise sequential patterns. This model is designed to predict potential object locations in the scenes. In terms of accuracy, this algorithm attained the best performance on three surveillance datasets. However, the authors make note that this method posed the challenge that it was not real-time detection.

One can go further than surveillance. Zhao et al. [161] build upon the theme, the scene and the temporal structure of video footage to identify specific, potentially harmful, videos. Preventing the spreading of videos containing harmful ideas is a valuable application. As usual, LSTM is employed to process the temporal information. The theme and scene are learned from a variety of other models. The bottom line of the research is that impressive results were obtained, thus setting a benchmark architecture up.

Another application of LSTM with regards to the computer vision field is the estimation of the human pose in three dimensions when the input consists of two-dimensional images [93]. This is accomplished in three distinct steps. First of all, the two-dimensional poses are analysed during which key skeleton points are identified using CNN. Second, three-dimensional points are constructed for each key point. The initial guess is obtained by optical triangulation. From this, it was expected that convergence would be faster, given that the starting point is not random. Third, the full body pose is estimated using the LSTM network, as a series of poses is available, thus integrating the spatial and temporal information. This method performed very well when compared with state-of-the-art approaches. In Brattoli et al. [11], the human pose was analysed with CNN and LSTM to reveal distinct functional deficits and restoration of humans during recovery, of which the approach proved widely applicable.

Nguyen et al. [91] modelled the human-to-human multi-modal interactive behaviour with LSTM. This behaviour consists of speech, gaze and gestures which

are jointly modelled. To be more specific, the one-directional LSTM was used for on-line model comparison, and the bidirectional input is employed for off-line comparison. For both off-line and on-line prediction tasks, the chosen model yielded better results than the conventional benchmark methods when generating the appropriate overt actions.

For end-to-end sequence learning of actions in video, VideoLSTM was introduced by Li et al. [74]. However, their approach went the other way around: instead of adapting the data to the model with regards to input, the authors adapted the model to the data. They argued that, to reckon the spatio-temporal nature of the video using an LSTM, they should hard-wire the LSTM network with convolutions and spatio-temporal attentions, thus creating a convolutional attention LSTM architecture.

In order to tackle the problem of parallelisation on GPUs of multidimensional LSTMs, Stollenga et al. [118] proposed the PyraMiD-LSTM, a re-arrangement of the traditional cuboid order of computations in a pyramidal fashion. Experiments have shown that this model is easy to parallelise, especially for 3D data and outperformed the state-of-the-art in pixel-wise image segmentation.

Naturally, there are plenty of other contributions in computer vision [4, 27, 43, 76, 81, 84, 96, 114], with or without proposing a variation of the LSTM model to improve performance.

4.4.1 Text Recognition

Another field in which LSTM has proved to be specially proficient is text recognition, a known subtask of computer vision. For example, Naz et al. [89] investigated text recognition possibilities on cursive scripts, specifically Urdu. The challenge imposed by cursive scripts originates from the large number of character shapes, inter- and intra-word overlaps, context sensitivity and diagonality of text. In this study, sliding windows over the text lines are employed for feature extraction, of which the resulting vector is inputted into a multi-dimensional LSTM network. The final output is provided by a connectionist temporal classification layer. In other words, the used architecture consists of three stages. This technique resulted in the highest results reported for the benchmark problems. Note that, a few years earlier, Graves and Schmidhuber [41] applied a similar architecture with multi-dimensional recurrent networks [38] and connectionist temporal classification, which was at that time also a breakthrough with respect to accuracy.

A second study by Naz et al. [90] on Urdu was conducted one year later, to further improve character recognition of the cursive script. This was done with explicit feature extraction, rather than implicit, by employing CNN. The CNN extracted lower level features, then convoluted the learned kernels with text line images and finally fed the features to a multi-dimensional LSTM [38] which is used as the classifier in this system. The simulations, carried out on a public dataset, showed this architecture again outperformed the state-of-the-art models, including the three-stage model he proposed with his collaborators one year earlier.

The results in [89, 90] suggest that creating hybrid architectures provide more accurate classifications than the ones computed with the vanilla LSTM model. This would explain the wide usage of hybrid models reported in the literature. A similar framework is applied by Bhunia et al. [9] where a three-stage approach was used. First, stacked convolutional layers extract precise translation invariant

image features. These layers generate varying dimension feature vectors that are fed into an LSTM network to exploit the spatial dependencies present in text script images. Second, patch weights are obtained via an attention network followed by a softmax layer. Third, the features obtained in the second stage are integrated by employing attention based dynamic weighting.

Before going to the recognition step, Frinken et al. [30] performed several text preprocessing tasks. To summarise, after extracting the text lines, the skew angle was determined and removed by rotation. Afterwards the slant is corrected to normalise directions of long vertical strokes. After this estimation, a shear transformation is applied, to finally scale all characters both vertically and horizontally. The BLSTM neural network is then employed for keyword spotting. Keyword spotting is a detection task consisting in discovering the presence of specific spoken words in speech signals [28]. This study by Fernandez et al. used the power of BLSTM to handle information through time and showed the outperformance compared to the classic HMM in terms of accuracy. Another example BLSTM's power is provided in [2] to recognise handwritten mathematical expressions, which they see as collections of strokes, as it is the state-of-the-art model which outperformed previous models. Van Phan and Nakagawa [131] wanted a highly accurate and quick method for text/non-text classification, for which they experimented with BLSTM and achieved top-tier results, but also several future research challenges. The authors in [152] used, amongst others, the BLSTM as a text recogniser to feed it to the language modelling network. In [106] the BLSTM network was explored for text classification using both supervised and semi-supervised learning.

In [45] the authors proposed Tensorised LSTM in which the hidden states are represented by tensors and updates via a cross-layer convolution. The goal of this model is to increase capacity without adding additional parameters and with only a little longer runtime. Experiments on the MNIST dataset, in which written digits are to be recognised, showed the potential of the proposed model.

The features that CNNs can extract from text as input for an LSTM network, can also be fed in both directions, thus the CNN-BLSTM hybrid network is also an architecture to be considered. For example, in [128] a BLSTM was added to the already existing CNN model from previous research. This enabled the authors to extract semantic categories and thus populate a knowledge database with the document contents. Experimental results showed better performances than the state-of-the-art approaches.

Naturally, CNN is not the only alternative to create a hybrid model. The HMM can also be utilised. Liwicki and Bunke [79] were, as reported back in 2009, the first ones to propose the combination of such diverse recognition architectures on the decision level in the field of handwritten text line recognition. Finally, as LSTM could be combined with a connectionist temporal classification output layer, the same goes for BLSTM [150].

In contrast with time series applications, the vanilla LSTM architecture for text recognition does not always provide the most accurate results. The most convenient approaches exploit the input signal in both directions, or create hybrid deep learning architectures.

4.5 Other Application Domains

Of course, the use of the LSTM architecture is not limited to applications discussed above. A wide variety of problems are suited to the use of this model. In this section, we will illustrate how the model can be applied to this large range of problems.

For example, in [99] the maze learning performance of LSTM is compared to two other neural network architectures. A maze is defined as a network of distinctly marked rooms randomly interconnected by doors that open probabilistically. This study investigated two characteristics of the models: the retention of long-term state information and the modular use of the learned information. It appeared that the performance varied. LSTM is capable of learning the context maze tasks with non-modular training. The fact that it does have problems with modular training highlights the problem of using this model for tasks of a dynamic nature which may cause components to change, necessitating retraining.

An application in traffic analyses, namely the analysis of short-term crash risk, is proposed by Bao et al. [5]. In this work, three architectures are used: CNN to extract spatial features, LSTM for the temporal features, and finally convolutional LSTM for the spatio-temporal features, all in a stacked hierarchy. The simulations showed that the hybrid model performs better than standard machine learning approaches for capturing the spatio-temporal characteristics for the citywide short-term crash risk prediction.

Several applications are identified in the field of computer science. The authors in [26] use an LSTM as part of a Denial of Service and privacy attacks detection model, in which LSTM is focused on the identification of XSS and SQL attacks. This proposed system was not only very accurate, but also acceptably fast. Homayoun et al. [50] analysed the capability of CNN-LSTM to classify abnormalities related with ransomware activities. In [164] LSTM was applied to path planning in network traffic engineering with constrained conditions, which showed to be a superior method. Also in terms of information security did LSTM prove to be useful, as demonstrated by Kang et al. [59] in their analysis of malware detection and classification.

The network can also be found in the healthcare sector. For example, Pei et al. [94] adopted LSTM to map small bowel images to the corresponding diameters. In [69] an architecture was set up to recognise irregular entities in biomedical text. The contribution of Turan et al. [129] refers to a system that estimates the real-time pose for actively controlled endoscopic capsule robots. The authors in [3] developed an approach for automatic detection of atrial fibrillation. These applications all apply combinations of network types, where LSTM learns the temporal relations in the data. On the other hand, after several data preprocessing steps, LSTM is part of the development of an abnormal heart sound detection method in the study of Zhang et al. [156]. Perhaps more impressively, Yi et al. [149] applied the model in the fight against cancer in identifying anticancer peptides with great success. In the BLSTM context, Van Steenkiste et al. [117] contributed the value of the architecture in predicting outcomes of blood culture tests. The authors found that prediction power only decreased slightly even when predicting hours upfront.

In the field of sound recognition, Wöllmer and Schuller [142] used BLSTM in combination with bottleneck feature generation to develop a front-end allowing the production of context-sensitive probabilistic feature vectors of arbitrary size for

speech recognition. But besides speech, all kinds of sounds can be detected. In [67] several neural networks were compared with regards to detection of screams and shouts. All tested models performed virtually equally well, however, a distinction could be made when it came to speech recognition. The tests again show the temporal structure of speech, since recurrent neural networks outperformed the other networks types.

Eck and Schmidhuber [22] wondered whether LSTM would be a good candidate to learn how to compose music, since standard recurrent neural networks often lack global coherence. Their results show that LSTM can learn to play blues music, while also composing novel melodies in that style. The model also does not drift from the learned structure.

Airport runways have to be of high quality to ensure a safe landing. In this capacity, the authors in [12] developed *GrooveNet*: a classification model for identifying shallow and worn grooves in a runway consisting of two LSTM layers, two dropout layers and one fully connected layer. This methodology proved not only to be very robust, but also very accurate.

A perhaps more exotic application of LSTM concerns block-sparsity recovery [82]. Here, the authors employed the LSTM framework for better capturing the correlations and dependencies among nonzero elements of signals. This proposed method proved to be superior compared to alternative methods. Finally, Wang et al. [135] proposed a novel deep learning waveform recognition method, using two-channel CNNs combined with BLSTM. The contributors claim that this approach has a significantly better performance than the state-of-the-art.

5 Implementing LSTM in Tensorflow

As discussed above, LSTM can be utilised in a wide variety of situations. To run the model on your personal computer, some code libraries have been developed. In this section, we shall briefly describe how to build an LSTM neural network in Python, specifically using the Tensorflow framework, with code examples. Note that, for the sake of relevance, we limit ourselves to the most import code extracts.

First of all, the required code libraries must be imported.

```
import numpy as np
import tensorflow as tf
from tensorflow.contrib import rnn
import random
```

Let us suppose we want to train an LSTM to predict the next word of a sample short story. If we feed it with correct sequences of three symbols from the text as inputs and a labelled symbol, eventually the neural network will learn to predict the next symbol correctly.

The LSTM only supports numeric inputs. A way to convert symbols to numeric inputs is to assign a unique sequential number to each symbol based in order of appearance. The reverse dictionary is also generated since it will be used to decode the outputs.

```
def build_dataset(words):
    n2w = dict(enumerate(words))
```

```
    return dict(zip(n2w.values(), n2w.keys())), n2w
```

LSTM produces an output vector of probabilities of the next symbol normalised by the `softmax()` function. The index of the element with the highest probability is the predicted index of the symbol in the reverse dictionary. This model is at the core of the application, which is very simple to implement in Tensorflow.

```
def RNN(x, weights, biases):
    # reshape to [1, n_input]
    x = tf.reshape(x, [-1, n_input])
    x = tf.split(x, n_input, 1)

    # 1-layer LSTM with n_hidden units
    lstm_cell = rnn.BasicLSTMCell(n_hidden, forget_bias=1.0)

    # Get lstm cell output
    outputs, states = rnn.static_rnn(lstm_cell, x, dtype=tf.float32)

    # Linear activation, using rnn inner loop last output
    return tf.matmul(outputs[-1], weights['out']) + biases['out']
```

In the training process, at each step, three symbols are retrieved from the training data to form the input vector. These three symbols are converted to numeric values.

```
        x = [[w_dict[w]] for w in training_data[offset: offset + n_input]]
        x = np.reshape(np.array(x), [-1, n_input, 1])
```

The training label is a one-hot vector coming from the symbol after the three input symbols.

```
        y = np.zeros([n_classes], dtype=float)
        y[w_dict[training_data[offset + n_input]]] = 1.0
        y = np.reshape(y, [1, -1])
```

After reshaping to fit in the feed dictionary, the optimisation runs.

```
        # Run optimisation op (backprop)
        session.run(train_op, feed_dict={X: x, Y: y})
```

The accuracy and loss are accumulated to monitor the progress of the training.

```
        if step % display_step == 0 or step == 1:
            # Calculate batch loss and accuracy
            loss, acc = session.run([loss_op, accuracy], feed_dict={X: x, Y:
                 y})
            print("Step " + str(step) +
                  ", Minibatch Loss={:.4f}".format(loss) +
                  ", Training Accuracy= {:.3f}".format(acc))
```

The cost is a cross entropy between the label and `softmax()` prediction using a gradient descent optimiser at a learning rate of 0.001.

```
# Define loss and optimiser
```

```
loss_op =
     tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(logits=logits,
     labels=Y))
optimizer = tf.train.GradientDescentOptimizer(learning_rate=learning_rate)
train_op = optimizer.minimize(loss_op)
```

The accuracy of the LSTM can be improved by additional layers.

```
# 2-layer LSTM, each layer has n_hidden units
lstm_cell = rnn.MultiRNNCell([rnn.BasicLSTMCell(n_hidden),
    rnn.BasicLSTMCell(n_hidden)])
```

After training, we are enabled to test the network predicting the next word after input “had”, “a” and “general” words.

```
    # Next word prediction after words "had", "a" and "general"
    x = ['had', 'a', 'general']
    x = np.reshape(np.array([w_dict[w] for w in x]), [-1, n_input, 1])
    y = session.run(prediction, feed_dict={X: x})
    print(k_dict[int(tf.argmax(y, 1).eval())])
```

Also, we can test the accuracy of a batch sample.

```
# Calculate accuracy
print("Testing Accuracy:", \
    session.run(accuracy, feed_dict={X: batch_x, Y: batch_y}))
```

As the reader can notice, we only need a handful of code in the Tensorflow framework in order to predict the next word in a sequence. Alternatively, one can also opt for Keras or Pytorch to implement LSTM-based solutions.

6 Conclusions

In this paper, we have revised the recent applications on LSTM reported in the literature. Our survey has illustrated the ability of this recurrent system to handle a wide variety of problems including time series forecasting, text recognition, natural language processing, image and video captioning, sentiment analysis and computer vision. When modelling most of these problems, it was found a common practice is to hybridise CNNs with LSTM with the aim to get an optimal performance. In such hybrid models, convolution and pooling layers were used to reduce the problem dimensionality while greatly suppressing the redundancy in representations. However, as the choice of networks to integrate is so vast, Table 1 shows a summary per application domain with recommended network types. Note that recommendations are limited to either the LSTM-dominated network or (standard) integrated architectures. As discussed in Section 4, further customisation of such architectures could always be applied to improve accuracy. Keep in mind that, as described in [151], there is no variant surpassing the standard LSTM at all aspects and integrated networks could also use improvements. Second, our recommendations are based on the results reported in the literature, but it is wise to remember the heterogeneity of problems, and as such, results will vary. Therefore, it would be wise to take our recommendations with a grain of salt.

Table 1 Recommendations – LSTM-dominated or integrated networks.

Problem Domain	Recommended architecture	Elaboration
Time Series Prediction	Vanilla LSTM	– Relatively simple to configure for the problem at hand, – High accuracy rate in comparison to state-of-the-art, – Wide applicability w.r.t. predicting future values and classification.
Natural Language Processing	BLSTM, CNN-LSTM	– Language is related to both previous and the next words, so temporal information is best represented in both directions, – CNN-LSTM is most useful for text classification, – Vanilla LSTM could lack sufficient accuracy.
Sentiment Analysis	CNN-LSTM	– CNN on its own yields good results already, – The integrated network provides a significant boost in prediction accuracy.
Image & Video Captioning	CNN-LSTM	– Architecture of choice for the majority of authors covered in our review, thus implicitly showing effectiveness, – High synergy between models, as CNN executes the important feature selection process for LSTM.
Computer Vision	Integrated architectures	– LSTM-dominated model shows variance in performance, – Computer Vision tasks are of a high dimensionality and complexity, as such, integrate models to take advantages of each one's strengths, – Mainly, adding CNN for feature extraction is recommended.
Text Recognition	BLSTM, CNN-(B)LSTM	– Vanilla LSTM is often insufficient in performance, – Recognising text is dependent on both the next and previous character, so exploiting BLSTM is practical, – Hybrid architectures, like CNN-(B)LSTM, certainly have great potential but adds complexity.

Together with relevant LSTM applications, the fundamental underpinnings behind this recurrent system are detailed including its main components, the interaction with each other and a gradient-based method to compute the weight matrix. The experimental study in [42] concluded that the forget gate and the output transfer function are the most critical components of the LSTM block, whereas the learning rate is the most important hyperparameter in the backpropagation algorithm. Hence, further studying these components may lead to LSTM variants with improved prediction capabilities. Another equally relevant research line refers to less computationally demanding learning procedures to adjust the learnable parameters.

Acknowledgements We thank the reviewers for their very thoughtful and thorough reviews of our manuscript. Their input has been invaluable in increasing the quality of our paper. Also, a special thanks to prof. Jürgen Schmidhuber for taking the time to share his thoughts on the manuscript with us and making suggestions for further improvements.

References

1. Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., Savarese, S.: Social lstm: Human trajectory prediction in crowded spaces. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 961–971 (2016)
2. Álvaro, F., Sánchez, J.A., Benedí, J.M.: An integrated grammar-based approach for mathematical expression recognition. Pattern Recognition **51**, 135 – 147 (2016)
3. Andersen, R.S., Peimankar, A., Puthusserypady, S.: A deep learning approach for real-time detection of atrial fibrillation. Expert Systems with Applications **115**, 465 – 473 (2019)
4. Baddar, W.J., Ro, Y.M.: Mode variational lstm robust to unseen modes of variation: Application to facial expression recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 3215–3223 (2019)
5. Bao, J., Liu, P., Ukkusuri, S.V.: A spatiotemporal deep learning approach for citywide short-term crash risk prediction with multi-source data. Accident Analysis & Prevention **122**, 239 – 254 (2019)
6. Barbieri, F., Anke, L.E., Camacho-Collados, J., Schockaert, S., Saggion, H.: Interpretable emoji prediction via label-wise attention lstms. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 4766–4771 (2018)
7. Bayer, J., Wierstra, D., Togelius, J., Schmidhuber, J.: Evolving memory cell structures for sequence learning. In: International Conference on Artificial Neural Networks, pp. 755–764. Springer (2009)
8. Bellec, G., Salaj, D., Subramoney, A., Legenstein, R., Maass, W.: Long short-term memory and learning-to-learn in networks of spiking neurons. In: Advances in Neural Information Processing Systems, pp. 787–797 (2018)
9. Bhunia, A.K., Konwer, A., Bhunia, A.K., Bhowmick, A., Roy, P.P., Pal, U.: Script identification in natural scene image and video frames using an attention based convolutional-lstm network. Pattern Recognition **85**, 172 – 184 (2019)

10. Bilakhia, S., Petridis, S., Nijholt, A., Pantic, M.: The mahnob mimicry database: A database of naturalistic human interactions. Pattern Recognition Letters **66**, 52 – 61 (2015). Pattern Recognition in Human Computer Interaction
11. Brattoli, B., Buchler, U., Wahl, A.S., Schwab, M.E., Ommer, B.: Lstm self-supervision for detailed behavior analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6466–6475 (2017)
12. Cai, A.Z., Li, B.L., Hu, C.Y., Luo, D.W., Lin, E.C.: Automated groove identification and measurement using long short-term memory unit. Measurement **141**, 152 – 161 (2019)
13. Cen, Z., Wang, J.: Crude oil price prediction model with long short term memory deep learning based on prior knowledge data transfer. Energy **169**, 160 – 171 (2019)
14. Chen, B., Li, P., Sun, C., Wang, D., Yang, G., Lu, H.: Multi attention module for visual tracking. Pattern Recognition **87**, 80 – 93 (2019)
15. Chen, L., He, Y., Fan, L.: Let the robot tell: Describe car image with natural language via lstm. Pattern Recognition Letters **98**, 75 – 82 (2017)
16. Chen, M., Ding, G., Zhao, S., Chen, H., Liu, Q., Han, J.: Reference based lstm for image captioning. In: Thirty-First AAAI Conference on Artificial Intelligence (2017)
17. Chen, Y., Yang, J., Qian, J.: Recurrent neural network for facial landmark detection. Neurocomputing **219**, 26 – 38 (2017)
18. Chen, Y., Zhang, S., Zhang, W., Peng, J., Cai, Y.: Multifactor spatio-temporal correlation model based on a combination of convolutional neural network and long short-term memory neural network for wind speed forecasting. Energy Conversion and Management **185**, 783 – 799 (2019)
19. Chowdhury, G.G.: Natural language processing. Annual Review of Information Science and Technology **37**(1), 51–89 (2003)
20. Dabiri, S., Heaslip, K.: Developing a twitter-based traffic event detection model using deep learning architectures. Expert Systems with Applications **118**, 425 – 439 (2019)
21. D'Andrea, E., Ducange, P., Bechini, A., Renda, A., Marcelloni, F.: Monitoring the public opinion about the vaccination topic from tweets analysis. Expert Systems with Applications **116**, 209 – 226 (2019)
22. Eck, D., Schmidhuber, J.: Finding temporal structure in music: Blues improvisation with lstm recurrent networks. In: Proceedings of the 12th IEEE workshop on neural networks for signal processing, pp. 747–756. IEEE (2002)
23. Elsheikh, A., Yacout, S., Ouali, M.S.: Bidirectional handshaking lstm for remaining useful life prediction. Neurocomputing **323**, 148 – 156 (2019)
24. Fan, H., Zhu, L., Yang, Y.: Cubic lstms for video prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33 (2019)
25. Fayek, H.M., Lech, M., Cavedon, L.: Evaluating deep learning architectures for speech emotion recognition. Neural Networks **92**, 60 – 68 (2017). Advances in Cognitive Engineering Using Neural Networks
26. Feng, F., Liu, X., Yong, B., Zhou, R., Zhou, Q.: Anomaly detection in ad-hoc networks based on deep learning model: A plug and play device. Ad Hoc Networks **84**, 82 – 89 (2019)
27. Feng, Y., Ma, L., Liu, W., Luo, J.: Spatio-temporal video re-localization by warp lstm. In: Proceedings of the IEEE Conference on Computer Vision and

Pattern Recognition, pp. 1288–1297 (2019)
28. Fernández, S., Graves, A., Schmidhuber, J.: An application of recurrent neural networks to discriminative keyword spotting. In: International Conference on Artificial Neural Networks, pp. 220–229. Springer (2007)
29. Fischer, T., Krauss, C.: Deep learning with long short-term memory networks for financial market predictions. European Journal of Operational Research **270**(2), 654 – 669 (2018)
30. Frinken, V., Fischer, A., Baumgartner, M., Bunke, H.: Keyword spotting for self-training of blstm nn based handwriting recognition systems. Pattern Recognition **47**(3), 1073 – 1082 (2014). Handwriting Recognition and other PR Applications
31. Gao, H., Mao, J., Zhou, J., Huang, Z., Wang, L., Xu, W.: Are you talking to a machine? dataset and methods for multilingual image question. In: Advances in neural information processing systems, pp. 2296–2304 (2015)
32. Gers, F., Schmidhuber, J.: Recurrent nets that time and count. In: Proceedings of the International Joint Conference on Neural Networks, vol. 3, pp. 189 – 194 (2000)
33. Gers, F., Schmidhuber, J., Cummins, F.: Learning to forget: Continual prediction with lstm. Neural computation **12**, 2451–71 (2000)
34. Gers, F.A., Pérez-Ortiz, J.A., Eck, D., Schmidhuber, J.: Learning context sensitive languages with lstm trained with kalman filters. In: International Conference on Artificial Neural Networks, pp. 655–660. Springer (2002)
35. Gers, F.A., Schmidhuber, E.: Lstm recurrent networks learn simple context-free and context-sensitive languages. IEEE Transactions on Neural Networks **12**(6), 1333–1340 (2001)
36. Gong, J., Chen, X., Gui, T., Qiu, X.: Switch-lstms for multi-criteria chinese word segmentation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 6457–6464 (2019)
37. Graves, A., Eck, D., Beringer, N., Schmidhuber, J.: Biologically plausible speech recognition with lstm neural nets. In: International Workshop on Biologically Inspired Approaches to Advanced Information Technology, pp. 127–136. Springer (2004)
38. Graves, A., Fernández, S., Schmidhuber, J.: Multi-dimensional recurrent neural networks. In: International conference on artificial neural networks, pp. 549–558. Springer (2007)
39. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional lstm and other neural network architectures. Neural Networks **18**(5), 602 – 610 (2005). IJCNN 2005
40. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional lstm networks. In: Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005., vol. 4, pp. 2047–2052 vol. 4 (2005). DOI 10.1109/IJCNN.2005.1556215
41. Graves, A., Schmidhuber, J.: Offline handwriting recognition with multidimensional recurrent neural networks. In: D. Koller, D. Schuurmans, Y. Bengio, L. Bottou (eds.) Advances in Neural Information Processing Systems 21, pp. 545–552. Curran Associates, Inc. (2009)
42. Greff, K., Srivastava, R.K., Koutnik, J., Steunebrink, B.R., Schmidhuber, J.: LSTM: A Search Space Odyssey. IEEE Transactions on Neural Networks and Learning Systems **28**(10), 2222–2232 (2017)

43. Guo, D., Zhou, W., Li, H., Wang, M.: Hierarchical lstm for sign language translation. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
44. He, X., Shi, B., Bai, X., Xia, G.S., Zhang, Z., Dong, W.: Image caption generation with part of speech guidance. Pattern Recognition Letters **119**, 229 – 237 (2019). Deep Learning for Pattern Recognition
45. He, Z., Gao, S., Xiao, L., Liu, D., He, H., Barber, D.: Wider and deeper, cheaper and faster: Tensorized lstms for sequence learning. In: Advances in neural information processing systems, pp. 1–11 (2017)
46. Hochreiter, S.: Untersuchungen zu dynamischen neuronalen netzen. Diploma, Technische Universität München **91**(1) (1991)
47. Hochreiter, S., Bengio, Y., Frasconi, P., Schmidhuber, J.: Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In: S.C. Kremer, J.F. Kolen (eds.) A Field Guide to Dynamical Recurrent Neural Networks. IEEE Press (2001)
48. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**, 1735–80 (1997)
49. Hochreiter, S., Schmidhuber, J.: Lstm can solve hard long time lag problems. In: M.C. Mozer, M.I. Jordan (eds.) Advances in Neural Information Processing Systems 9, pp. 473–479. MIT Press (1997)
50. Homayoun, S., Dehghantanha, A., Ahmadzadeh, M., Hashemi, S., Khayami, R., Choo, K.K.R., Newton, D.E.: Drthis: Deep ransomware threat hunting and intelligence system at the fog layer. Future Generation Computer Systems **90**, 94 – 104 (2019)
51. Hong, J., Wang, Z., Yao, Y.: Fault prognosis of battery system based on accurate voltage abnormity prognosis using long short-term memory neural networks. Applied Energy **251**, 113381 (2019)
52. Hori, T., Wang, W., Koji, Y., Hori, C., Harsham, B., Hershey, J.R.: Adversarial training and decoding strategies for end-to-end neural conversation models. Computer Speech & Language **54**, 122 – 139 (2019)
53. Horsmann, T., Zesch, T.: Do lstms really work so well for pos tagging?– a replication study. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 727–736 (2017)
54. Hou, Q., Wang, J., Bai, R., Zhou, S., Gong, Y.: Face alignment recurrent network. Pattern Recognition **74**, 448 – 458 (2018)
55. Huang, K.Y., Wu, C.H., Su, M.H.: Attention-based convolutional neural network and long short-term memory for short-term detection of mood disorders based on elicited speech responses. Pattern Recognition **88**, 668 – 678 (2019)
56. Huang, Y., Shen, L., Liu, H.: Grey relational analysis, principal component analysis and forecasting of carbon emissions based on long short-term memory in china. Journal of Cleaner Production **209**, 415 – 423 (2019)
57. Kadari, R., Zhang, Y., Zhang, W., Liu, T.: Ccg supertagging via bidirectional lstm-crf neural architecture. Neurocomputing **283**, 31 – 37 (2018)
58. Kafle, K., Kanan, C.: Visual question answering: Datasets, algorithms, and future challenges. Computer Vision and Image Understanding **163**, 3 – 20 (2017). Language in Vision
59. Kang, J., Jang, S., Li, S., Jeong, Y.S., Sung, Y.: Long short-term memory-based malware classification method for information security. Computers & Electrical Engineering **77**, 366 – 375 (2019)

60. Kanjo, E., Younis, E.M., Ang, C.S.: Deep learning analysis of mobile physiological, environmental and location sensor data for emotion detection. Information Fusion **49**, 46 – 56 (2019)
61. Kartsaklis, D., Pilehvar, M.T., Collier, N.: Mapping text to knowledge graph entities using multi-sense lstms. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 1959–1970 (2018)
62. Kim, B., Chung, K., Lee, J., Seo, J., Koo, M.W.: A bi-lstm memory network for end-to-end goal-oriented dialog learning. Computer Speech & Language **53**, 217 – 230 (2019)
63. Kim, S., Kang, S., Ryu, K.R., Song, G.: Real-time occupancy prediction in a large exhibition hall using deep learning approach. Energy and Buildings (2019)
64. Kinghorn, P., Zhang, L., Shao, L.: A hierarchical and regional deep learning architecture for image description generation. Pattern Recognition Letters **119**, 77 – 85 (2019). Deep Learning for Pattern Recognition
65. Kraus, M., Feuerriegel, S.: Sentiment analysis based on rhetorical structure theory: Learning deep neural networks from discourse trees. Expert Systems with Applications **118**, 65 – 79 (2019)
66. Kumar Srivastava, R., Greff, K., Schmidhuber, J.: Training very deep networks. Neural Information Processing Systems (NIPS 2015 Spotlight) (2015)
67. Laffitte, P., Wang, Y., Sodoyer, D., Girin, L.: Assessing the performances of different neural network architectures for the detection of screams and shouts in public transportation. Expert Systems with Applications **117**, 29 – 41 (2019)
68. Lei, J., Liu, C., Jiang, D.: Fault diagnosis of wind turbine based on long short-term memory networks. Renewable Energy **133**, 422 – 432 (2019)
69. Li, F., Zhang, M., Tian, B., Chen, B., Fu, G., Ji, D.: Recognizing irregular entities in biomedical text via deep neural networks. Pattern Recognition Letters **105**, 105 – 113 (2018). Machine Learning and Applications in Artificial Intelligence
70. Li, H., Xu, H.: Video-based sentiment analysis with hvnlbp-top feature and bi-lstm. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 9963–9964 (2019)
71. Li, P., Li, Y., Xiong, Q., Chai, Y., Zhang, Y.: Application of a hybrid quantized elman neural network in short-term load forecasting. International Journal of Electrical Power & Energy Systems **55**, 749 – 759 (2014)
72. Li, X., Ye, M., Liu, Y., Zhang, F., Liu, D., Tang, S.: Accurate object detection using memory-based models in surveillance scenes. Pattern Recognition **67**, 73 – 84 (2017)
73. Li, X., Zhang, L., Wang, Z., Dong, P.: Remaining useful life prediction for lithium-ion batteries based on a hybrid model combining the long short-term memory and elman neural networks. Journal of Energy Storage **21**, 510 – 518 (2019)
74. Li, Z., Gavrilyuk, K., Gavves, E., Jain, M., Snoek, C.G.: Videolstm convolves, attends and flows for action recognition. Computer Vision and Image Understanding **166**, 41 – 50 (2018)
75. Liu, A.A., Xu, N., Wong, Y., Li, J., Su, Y.T., Kankanhalli, M.: Hierarchical & multimodal video captioning: Discovering and transferring multimodal knowledge for vision to language. Computer Vision and Image Understanding **163**,

113 – 125 (2017). Language in Vision
76. Liu, J., Wang, G., Hu, P., Duan, L.Y., Kot, A.C.: Global context-aware attention lstm networks for 3d action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1647–1656 (2017)
77. Liu, Y.: Novel volatility forecasting using deep learning–long short term memory recurrent neural networks. Expert Systems with Applications **132**, 99–109 (2019)
78. Liu, Y., Jin, X., Shen, H.: Towards early identification of online rumors based on long short-term memory networks. Information Processing & Management **56**(4), 1457 – 1467 (2019)
79. Liwicki, M., Bunke, H.: Combining diverse on-line and off-line systems for handwritten text line recognition. Pattern Recognition **42**(12), 3254 – 3263 (2009). New Frontiers in Handwriting Recognition
80. Lu, Z., Tan, H., Li, W.: An evolutionary context-aware sequential model for topic evolution of text stream. Information Sciences **473**, 166 – 177 (2019)
81. Luo, Y., Ren, J., Wang, Z., Sun, W., Pan, J., Liu, J., Pang, J., Lin, L.: Lstm pose machines. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5207–5215 (2018)
82. Lyu, C., Liu, Z., Yu, L.: Block-sparsity recovery via recurrent neural network. Signal Processing **154**, 129 – 135 (2019)
83. Ma, J., Ganchev, K., Weiss, D.: State-of-the-art chinese word segmentation with bi-lstms. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 4902–4908 (2018)
84. Ma, S., Sigal, L., Sclaroff, S.: Learning activity progression in lstms for activity detection and early detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1942–1950 (2016)
85. Ma, Y., Peng, H., Cambria, E.: Targeted aspect-based sentiment analysis via embedding commonsense knowledge into an attentive lstm. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
86. Manashty, A., Light, J.: Life model: A novel representation of life-long temporal sequences in health predictive analytics. Future Generation Computer Systems **92**, 141 – 156 (2019)
87. McCarthy, N., Karzand, M., Lecue, F.: Amsterdam to dublin eventually delayed? lstm and transfer learning for predicting delays of low cost airlines. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 9541–9546 (2019)
88. Metz, C.: An infusion of ai makes google translate more powerful than ever (2016). URL https://www.wired.com/2016/09/google-claims-ai-breakthrough-machine-translation/
89. Naz, S., Umar, A.I., Ahmad, R., Ahmed, S.B., Shirazi, S.H., Siddiqi, I., Razzak, M.I.: Offline cursive urdu-nastaliq script recognition using multidimensional recurrent neural networks. Neurocomputing **177**, 228 – 241 (2016)
90. Naz, S., Umar, A.I., Ahmad, R., Siddiqi, I., Ahmed, S.B., Razzak, M.I., Shafait, F.: Urdu nastaliq recognition using convolutionalrecursive deep learning. Neurocomputing **243**, 80 – 87 (2017)
91. Nguyen, D.C., Bailly, G., Elisei, F.: Learning off-line vs. on-line models of interactive multimodal behaviors with recurrent neural networks. Pattern Recognition Letters **100**, 29 – 36 (2017)

92. Núñez, J.C., Cabido, R., Pantrigo, J.J., Montemayor, A.S., Vélez, J.F.: Convolutional neural networks and long short-term memory for skeleton-based human activity and hand gesture recognition. Pattern Recognition **76**, 80 – 94 (2018)
93. Núñez, J.C., Cabido, R., Vélez, J.F., Montemayor, A.S., Pantrigo, J.J.: Multiview 3d human pose estimation using improved least-squares and lstm networks. Neurocomputing **323**, 335 – 343 (2019)
94. Pei, M., Wu, X., Guo, Y., Fujita, H.: Small bowel motility assessment based on fully convolutional networks and long short-term memory. Knowledge-Based Systems **121**, 163 – 172 (2017)
95. Pei, Z., Qi, X., Zhang, Y., Ma, M., Yang, Y.H.: Human trajectory prediction in crowded scene using social-affinity long short-term memory. Pattern Recognition **93**, 273 – 282 (2019)
96. Perrett, T., Damen, D.: Ddlstm: Dual-domain lstm for cross-dataset action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7852–7861 (2019)
97. Pino, J.M., Sidorov, A., Ayan, N.F.: Transitioning entirely to neural machine translation (2017). URL https://engineering.fb.com/ml-applications/transitioning-entirely-to-neural-machine-translation/
98. Plank, B., Søgaard, A., Goldberg, Y.: Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 412–418 (2016)
99. Portegys, T.E.: A maze learning comparison of elman, long short-term memory, and mona neural networks. Neural Networks **23**(2), 306 – 313 (2010)
100. Rabiner, L.R.: An introduction to hidden markov models. IEEE ASSP Magazine **3**(1), 4–16 (1986)
101. Ren, J., Hu, Y., Tai, Y.W., Wang, C., Xu, L., Sun, W., Yan, Q.: Look, listen and learna multimodal lstm for speaker identification. In: Thirtieth AAAI Conference on Artificial Intelligence (2016)
102. Ringeval, F., Eyben, F., Kroupi, E., Yuce, A., Thiran, J.P., Ebrahimi, T., Lalanne, D., Schuller, B.: Prediction of asynchronous dimensional emotion ratings from audiovisual and physiological data. Pattern Recognition Letters **66**, 22 – 30 (2015). Pattern Recognition in Human Computer Interaction
103. Rodrigues, F., Markou, I., Pereira, F.C.: Combining time-series and textual data for taxi demand prediction in event areas: A deep learning approach. Information Fusion **49**, 120 – 129 (2019)
104. Ryu, S., Kim, S., Choi, J., Yu, H., Lee, G.G.: Neural sentence embedding using only in-domain sentences for out-of-domain sentence detection in dialog systems. Pattern Recognition Letters **88**, 26 – 32 (2017)
105. Sabina Aouf, R.: Openai creates dactyl robot hand with "unprecedented" dexterity (2019). URL https://www.dezeen.com/2018/08/07/openai-musk-dactyl-robot-hand-unprecedented-dexterity-technology/
106. Sachan, D.S., Zaheer, M., Salakhutdinov, R.: Revisiting lstm networks for semi-supervised text classification via mixed objective function. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 6940–6948 (2019)
107. Saeed, H.A., jun Peng, M., Wang, H., wen Zhang, B.: Novel fault diagnosis scheme utilizing deep learning networks. Progress in Nuclear Energy **118**,

103066 (2020)
108. Sagheer, A., Kotb, M.: Time series forecasting of petroleum production using deep lstm recurrent networks. Neurocomputing **323**, 203 – 213 (2019)
109. Sak, H., Senior, A., Rao, K., Beaufays, F., Schalkwyk, J.: Google voice search: faster and more accurate (2015). URL https://ai.googleblog.com/2015/09/google-voice-search-faster-and-more.html
110. Sang, C., Pierro, M.D.: Improving trading technical analysis with tensorflow long short-term memory (lstm) neural network. The Journal of Finance and Data Science **5**(1), 1 – 11 (2019)
111. Schmid, H.: Part-of-speech tagging with neural networks. In: Proceedings of the 15th conference on Computational linguistics-Volume 1, pp. 172–176. Association for Computational Linguistics (1994)
112. Schmidhuber, J., Wierstra, D., Gagliolo, M., Gomez, F.: Training recurrent networks by evolino. Neural computation **19**(3), 757–779 (2007)
113. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing **45**(11), 2673–2681 (1997). DOI 10.1109/78.650093
114. Si, C., Chen, W., Wang, W., Wang, L., Tan, T.: An attention enhanced graph convolutional lstm network for skeleton-based action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1227–1236 (2019)
115. Song, L., Zhang, Y., Wang, Z., Gildea, D.: N-ary relation extraction using graph-state lstm. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 2226–2235 (2018)
116. Song, M., Park, H., shik Shin, K.: Attention-based long short-term memory network using sentiment lexicon embedding for aspect-level sentiment analysis in korean. Information Processing & Management **56**(3), 637 – 653 (2019)
117. Steenkiste, T.V., Ruyssinck, J., Baets, L.D., Decruyenaere, J., Turck, F.D., Ongenae, F., Dhaene, T.: Accurate prediction of blood culture outcome in the intensive care unit using long short-term memory neural networks. Artificial Intelligence in Medicine **97**, 38 – 43 (2019)
118. Stollenga, M.F., Byeon, W., Liwicki, M., Schmidhuber, J.: Parallel multi-dimensional lstm, with application to fast biomedical volumetric image segmentation. In: Advances in neural information processing systems, pp. 2998–3006 (2015)
119. Su, Y., Kuo, C.C.J.: On extended long short-term memory and dependent bi-directional recurrent neural network. Neurocomputing **356**, 151 – 161 (2019)
120. Sukhbaatar, S., Weston, J., Fergus, R., et al.: End-to-end memory networks. In: Advances in neural information processing systems, pp. 2440–2448 (2015)
121. Sun, X., Zhang, C., Li, L.: Dynamic emotion modelling and anomaly detection in conversation based on emotional transition tensor. Information Fusion **46**, 11 – 22 (2019)
122. Sun, Y., Ji, Z., Lin, L., Tang, D., Wang, X.: Entity disambiguation with decomposable neural networks. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **7**(5), e1215 (2017)
123. Sun, Y., Ji, Z., Lin, L., Wang, X., Tang, D.: Entity disambiguation with memory network. Neurocomputing **275**, 2367 – 2373 (2018)

124. Sutskever, I., Vinyals, O., Le, Q.: Sequence to sequence learning with neural networks. In: Advances in neural information processing systems (2014)
125. Takayama, J., Nomoto, E., Arase, Y.: Dialogue breakdown detection robust to variations in annotators and dialogue systems. Computer Speech & Language **54**, 31 – 43 (2019)
126. The AlphaStar Team: Alphastar: Grandmaster level in starcraft ii using multi-agent reinforcement learning (2019). URL https://deepmind.com/blog/article/AlphaStar-Grandmaster-level-in-StarCraft-II-using-multi-agent-reinforcement-learning
127. The AlphaStar Team: Alphastar: Mastering the real-time strategy game starcraft ii (2019). URL https://deepmind.com/blog/article/alphastar-mastering-real-time-strategy-game-starcraft-ii
128. Toledo, J.I., Carbonell, M., Fornés, A., Lladós, J.: Information extraction from historical handwritten document images with a context-aware neural model. Pattern Recognition **86**, 27 – 36 (2019)
129. Turan, M., Almalioglu, Y., Araujo, H., Konukoglu, E., Sitti, M.: Deep endovo: A recurrent convolutional neural network (rcnn) based visual odometry approach for endoscopic capsule robots. Neurocomputing **275**, 1861 – 1870 (2018)
130. Uddin, M.Z.: A wearable sensor-based activity prediction system to facilitate edge computing in smart healthcare system. Journal of Parallel and Distributed Computing **123**, 46 – 53 (2019)
131. Van Phan, T., Nakagawa, M.: Combination of global and local contexts for text/non-text classification in heterogeneous online handwritten documents. Pattern Recognition **51**, 112–124 (2016)
132. Venugopalan, S., Hendricks, L.A., Mooney, R., Saenko, K.: Improving lstm-based video description with linguistic knowledge mined from text. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 1961–1966 (2016)
133. Vogels, W.: Bringing the magic of amazon ai and alexa to apps on aws (2016). URL https://www.allthingsdistributed.com/2016/11/amazon-ai-and-alexa-for-all-aws-apps.html
134. Wang, L., Cao, Z., Xia, Y., De Melo, G.: Morphological segmentation with window lstm neural networks. In: Thirtieth AAAI Conference on Artificial Intelligence (2016)
135. Wang, Q., Du, P., Yang, J., Wang, G., Lei, J., Hou, C.: Transferred deep learning based waveform recognition for cognitive passive radar. Signal Processing **155**, 259 – 267 (2019)
136. Wang, W., Hong, T., Xu, X., Chen, J., Liu, Z., Xu, N.: Forecasting district-scale energy dynamics through integrating building network and long short-term memory learning algorithm. Applied Energy **248**, 217 – 230 (2019)
137. Wang, Y., Long, M., Wang, J., Gao, Z., Philip, S.Y.: Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms. In: Advances in Neural Information Processing Systems, pp. 879–888 (2017)
138. Wang, Z., Wang, Z., Long, Y., Wang, J., Xu, Z., Wang, B.: Enhancing generative conversational service agents with dialog history and external knowledge. Computer Speech & Language **54**, 71 – 85 (2019)
139. Wen, J., Tu, H., Cheng, X., Xie, R., Yin, W.: Joint modeling of users, questions and answers for answer selection in cqa. Expert Systems with Applic-

ations **118**, 563 – 572 (2019)
140. Werbos, P.J., et al.: Backpropagation through time: what it does and how to do it. Proceedings of the IEEE **78**(10), 1550–1560 (1990)
141. Wierstra, D., Gomez, F.J., Schmidhuber, J.: Modeling systems with internal state using evolino. In: Proceedings of the 7th annual conference on Genetic and evolutionary computation, pp. 1795–1802. ACM (2005)
142. Wöllmer, M., Schuller, B.: Probabilistic speech feature extraction with context-sensitive bottleneck neural networks. Neurocomputing **132**, 113 – 120 (2014). Innovations in Nature Inspired Optimization and Learning Methods Machines learning for Non-Linear Processing
143. Wu, Y., Schuster, M., Chen, Z., Le, Q.V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al.: Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv preprint arXiv:1609.08144 (2016)
144. Wu, Y.X., Wu, Q.B., Zhu, J.Q.: Improved eemd-based crude oil price forecasting using lstm networks. Physica A: Statistical Mechanics and its Applications **516**, 114 – 124 (2019)
145. Xingjian, S., Chen, Z., Wang, H., Yeung, D.Y., Wong, W.K., Woo, W.c.: Convolutional lstm network: A machine learning approach for precipitation nowcasting. In: Advances in neural information processing systems, pp. 802–810 (2015)
146. Yan, H., Ouyang, H.: Financial time series prediction based on deep learning. Wireless Personal Communications **102**, 1–18 (2017)
147. Yang, J., Guo, Y., Zhao, W.: Long short-term memory neural network based fault detection and isolation for electro-mechanical actuators. Neurocomputing (2019)
148. Yang, M., Tu, W., Wang, J., Xu, F., Chen, X.: Attention based lstm for target dependent sentiment classification. In: Thirty-First AAAI Conference on Artificial Intelligence (2017)
149. Yi, H.C., You, Z.H., Zhou, X., Cheng, L., Li, X., Jiang, T.H., Chen, Z.H.: Acp-dl: A deep learning long short-term memory model to predict anticancer peptides using high-efficiency feature representation. Molecular Therapy - Nucleic Acids **17**, 1 – 9 (2019)
150. Yousfi, S., Berrani, S.A., Garcia, C.: Contribution of recurrent connectionist language models in improving lstm-based arabic text recognition in videos. Pattern Recognition **64**, 245 – 254 (2017)
151. Yu, Y., Si, X., Hu, C., Zhang, J.: A review of recurrent neural networks: Lstm cells and network architectures. Neural computation **31**(7), 1235–1270 (2019)
152. Zamora-Martínez, F., Frinken, V., España-Boquera, S., Castro-Bleda, M., Fischer, A., Bunke, H.: Neural network language models for off-line handwriting recognition. Pattern Recognition **47**(4), 1642 – 1652 (2014)
153. Zhang, L., Zhu, G., Mei, L., Shen, P., Shah, S.A.A., Bennamoun, M.: Attention in convolutional lstm for gesture recognition. In: Advances in Neural Information Processing Systems, pp. 1953–1962 (2018)
154. Zhang, M., Wang, Q., Fu, G.: End-to-end neural opinion extraction with a transition-based model. Information Systems **80**, 56 – 63 (2019)
155. Zhang, P., Ouyang, W., Zhang, P., Xue, J., Zheng, N.: Sr-lstm: State refinement for lstm towards pedestrian trajectory prediction. In: Proceedings

of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 12085–12094 (2019)
156. Zhang, W., Han, J., Deng, S.: Abnormal heart sound detection using temporal quasi-periodic features and long short-term memory without segmentation. Biomedical Signal Processing and Control **53**, 101560 (2019)
157. Zhang, W., Li, Y., Wang, S.: Learning document representation via topic-enhanced lstm model. Knowledge-Based Systems **174**, 194 – 204 (2019)
158. Zhang, Z., Li, H., Zhang, L., Zheng, T., Zhang, T., Hao, X., Chen, X., Chen, M., Xiao, F., Zhou, W.: Hierarchical reinforcement learning for multi-agent moba game. arXiv preprint arXiv:1901.08004 (2019)
159. Zhao, J., Deng, F., Cai, Y., Chen, J.: Long short-term memory - fully connected (lstm-fc) neural network for pm2.5 concentration prediction. Chemosphere **220**, 486 – 492 (2019)
160. Zhao, J., Mao, X., Chen, L.: Speech emotion recognition using deep 1d & 2d cnn lstm networks. Biomedical Signal Processing and Control **47**, 312 – 323 (2019)
161. Zhao, Z., Song, Y., Su, F.: Specific video identification via joint learning of latent semantic concept, scene and temporal structure. Neurocomputing **208**, 378 – 386 (2016). SI: BridgingSemantic
162. Zhou, X., Wan, X., Xiao, J.: Attention-based lstm network for cross-lingual sentiment classification. In: Proceedings of the 2016 conference on empirical methods in natural language processing, pp. 247–256 (2016)
163. Zhu, W., Lan, C., Xing, J., Zeng, W., Li, Y., Shen, L., Xie, X.: Co-occurrence feature learning for skeleton based action recognition using regularized deep lstm networks. In: Thirtieth AAAI Conference on Artificial Intelligence (2016)
164. Zuo, Y., Wu, Y., Min, G., Cui, L.: Learning-based network path planning for traffic engineering. Future Generation Computer Systems **92**, 59 – 67 (2019)