UHASSELT

KNOWLEDGE IN ACTION

Faculteit Bedrijfseconomische Wetenschappen

master in de toegepaste economische wetenschappen: handelsingenieur in de beleidsinformatica

Masterthesis

Deep Learning: the power behind the Long Short-term Memory model

Greg Van Houdt

Scriptie ingediend tot het behalen van de graad van master in de toegepaste economische wetenschappen: handelsingenieur in de beleidsinformatica

PROMOTOR :

dr. Gonzalo NAPOLES RUIZ

2018
2019

UHASSELT

KNOWLEDGE IN ACTION

www.uhasselt.be
Universiteit Hasselt
Campus Hasselt:
Martelarenlaan 42 | 3500 Hasselt
Campus Diepenbeek:
Agoralaan Gebouw D | 3590 Diepenbeek

UHASSELT

KNOWLEDGE IN ACTION

Faculteit Bedrijfseconomische Wetenschappen

master in de toegepaste economische wetenschappen: handelsingenieur in de beleidsinformatica

Masterthesis

Deep Learning: the power behind the Long Short-term Memory model

Greg Van Houdt

Scriptie ingediend tot het behalen van de graad van master in de toegepaste economische wetenschappen: handelsingenieur in de beleidsinformatica

PROMOTOR :

dr. Gonzalo NAPOLES RUIZ

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 the webpage of one of the LSTM's fathers — Prof. Jürgen Schmidhuber — this model improved speech recognition on over 2 billion Android phones, greatly improved machine translation through Google Translate, and the answers of Amazon's Alexa. This neural system is also employed by Facebook, by 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 on 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 comprehensible review that covers LSTM's formulation and training, relevant applications reported in the literature and code resources implementing this neural system for a toy example.

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

1 Introduction

Recurrent or very deep neural networks are really difficult to train as they often suffer from the exploding & vanishing gradient problem [25]. To overcome this shortcoming when learning long-term dependencies, the Long Short-Term Memory architecture [26], or LSTM for short, was introduced. The learning

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

ability of LSTM impacted several fields from both a practical and theoretical perspective, so that it became a state-of-the-art model.

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. [23]. 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 improves performance. This justified most applications found in the literature to employ the vanilla LSTM.

In this paper, we bring a comprehensive review on the LSTM model that complements the theoretical findings presented in [23]. Our review study focuses on three main directions, which move from theory to practice. In the first part, we widely 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, image and video captioning, among others. In the last part, we present a code example in Keras that aims at predicting the next word from sample short story.

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 Keras can be found in Secion 5. Finally, we summarise our conclusions in Section 6.

2 Long Short-Term Memory

The LSTM model [26] is a powerful recurrent neural system specially designed to overcome the exploding and vanishing gradient problems that typically arise when learning long-term dependencies, even when the minimal time lags are very long [27]. Overall, this can be prevented by using a *constant error carousel* 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 that will be unveiled in this section.

An 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 proposed by Gers et al. [19] 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 attached with the cell.

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. Figure 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 [18]. The output of the block is recurrently connected back to the block input and all of the gates.

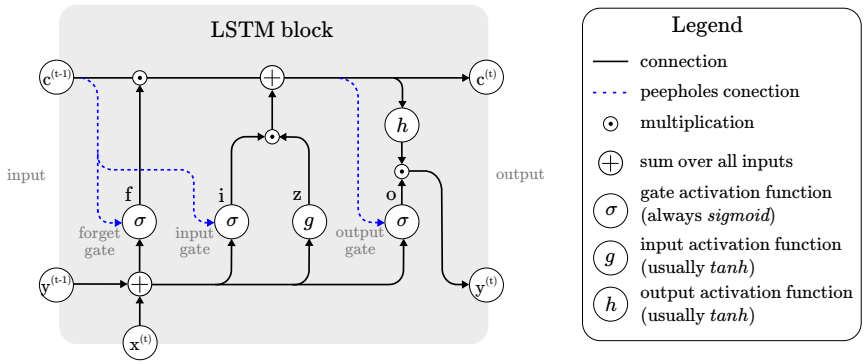

Fig. 1 Architecture of a typical vanilla LSTM block.

Aiming at clarifying how the LSTM model works, let us assume a network comprised of N processing blocks and M the number of inputs. The forward pass in this recurrent neural system can be fully described by six well-defined steps, which are summarized below.

Step 1. 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:

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

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.

Step 2. 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:

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

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.

Step 3. In this step, the LSTM layer 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.

Step 4. 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:

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

Step 5. 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:

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

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.

Step 6. Finally, we calculate the block output, which combines the current cell value $c^{(t)}$ with current output gate value as follows:

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

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 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 of [37] 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.

3 How to train your model

The LSTM model described in Section 2 uses full gradient training presented by Graves and Schmidhuber [20] to adjust the learnable parameters involved in the network. More specifically, *Backpropagation Through Time* 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:

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

$$\delta_c^{(t)} = \delta_y^{(t)} \odot o^{(t)} \odot h'(c^{(t)}) + p_o \odot \delta_{\hat{o}}^{(t)} + p_i \odot \delta_{\hat{i}}^{(t+1)} \tag{9}$$

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

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

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

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.

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

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

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

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

where $\otimes$ 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 $\hat{o}$.

4 Relevant Applications

The LSTM network is used 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 ideal to handle time series predictions, but also any other problem that requires temporal memory.

This section will give an overview in what topics the model is most suitable and how to use it to solve complex problems. In that regard, we will discuss the applications per problem domain.

4.1 Time Series Prediction

When speaking of 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. Fisher and Krauss [16] were amongst the first to attempt this, along with Yan and Ouyang [76]. Due to complex features such as non-linearity, non-stationarity and sequence correlation, financial data poses 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 net and a standard logistic regression.

Sagheer and Kotb [59] 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 increases the model's ability to process temporal tasks and enable it to better capture the structure of data sequences.

In [57], 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 generalizable 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 [39] 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 utilized to capture long-term dependencies in the data to do proper fault classification. Experiments in this study show that this framework is not only robust, but it also outperformed the state-of-the-art methods.

As another LSTM-based classification example, Uddin [70] 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 your laptop. This sensor data was fed to LSTM with which twelve different human activities were modeled. 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. [13] 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 [61]. The same concept can be applied to the LSTM network, as it is a recurrent network that takes sequences as input. Graves and Schmidhuber [21] 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 indicate that BLSTM can achieve even better results.

Since the problem at hand in [13] 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 initialize the backward processing cells. This forces the network to get 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.

Lei et al. [39], as discussed above, made the conscious choice of using raw input data, but this is not at all required. The input sequence can be preprocessed by another deep learning or other architecture, forming a hybrid model. Wu et al. [75] 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 not all. Other applications exist for time series prediction scenarios, like emotion ratings [56], topic evolution in text streams [45] and health predictive analytics [47]. The LSTM's prediction power speaks for the wide usability of LSTM to overcome problems in which other recurrent neural systems are likely to fail.

4.2 Text Recognition

Another field in which LSTM has proved to be specially proficient is text recognition. For example, Naz et al. [48] 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 (CTC) layer. In other words, the used architecture consists of three stages. This technique resulted in the highest results reported for the benchmark problems. However, a few years earlier, Graves and Schmidhuber [22] applied a similar architecture with multi-dimensional recurrent networks and CTC, which was at that time also a breakthrough with respect to accuracy.

A second study by Naz et al. [49] 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 a convolutional neural network (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 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 [48,49] 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. [4] 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. This research is the first to propose attention mechanisms in script identification, which provided state-of-the-art accuracy rates.

Before going to the recognition step, Frinken et al. [17] performed several text preprocessing tasks. To summarize: after extracting the text lines, the skew angle was determined and removed by rotation. Afterward the slant is corrected to normalize 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. [1] utilized BLSTM to recognize 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 [54] 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 of [78] used, amongst others, the BLSTM as a text recognizer to feed it to the language modeling network.

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. This was done in [68]: 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 hidden Markov model can also be utilized. Liwicki and Bunke [44] 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 CTC output layer, the same goes for BLSTM [77].

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.

4.3 Natural Language Processing

Building upon text recognition is natural language processing, the field of research that explores how computers can be used to understand and manipulate natural language text or speech to do useful things [9].

For example, dialog systems — also known as conversational agents — allow human beings to interact with a machine via speech. The system should be able to respond to *out-of-domain* speech input, instead of giving a random response. In [58] a binary classifier was built using two LSTM layers. The classifier was trained with only *in-domain* data, with the goal to later on recognize *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 sentences, 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 [65] 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 [66].

If the context of sentences and words is understood, then questions can be answered about them. One example of this application is visual question an-

swering, a computer vision task where a system is provided with a text-based question about an image and the answer must be inferred [32]. Section 4.5 elaborates on other computer vision applications. Alternatively, community question answering, selecting the correct answer to a question, can be performed. The authors of [73] 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. [67] investigated variations in determining a response that breakdowns a conversation (subjectivity), and variation in breakdown types due to designs of agents (variationality). The variationality was addressed with three models: LSTM which considers the global context, 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. [29] 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.

Building a dialog system is, as expected, also possible with BLSTM. Again, 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 of [34] employed the MemN2N architecture [63] 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.

Wang et al. [72] 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 with 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 analyzed and extracted from sentences as well, as done by Zhang et al. [79] who employed multilayered BLSTM architecture. D'Andrea et al. [11] 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 [10] with the use of CNN and LSTM, from which the contributors concluded that their approach reported superior results when compared with other algorithms. Combinatory Categorical Grammar supertagging can also be performed by means of deep learning architectures as illustrated in [31] where the authors used BLSTM and conditional random field. This hybrid architecture attained good results in a very efficient manner.

4.4 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. [33] put to use sensor signals coming from these multi-modal 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.

Identifying people's emotions can also help detect conversation anomalies. For this, Sun et al. [64] 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 at a specific direction.

Krauss and Feuerriegel [36] proposed a method that builds upon the discourse structure of documents, namely 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.

The authors in [80] 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.

In speech data emotion recognition, Fayek et al. [14] 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.

4.5 Computer Vision

LSTM can also be used for gesture and action recognition. This field includes identifying human poses and interactions. In [5] the authors investigated automatic recognition of mimicry behavior, as mimicry has the power to influence social judgements and behaviors. Mimicry behavior 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 generalized 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.

A similar study was conducted by Chen et al. [8]. The network architecture consists in 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. [30] 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 [51] 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 utilized. Extensive simulations concluded that this hybrid model performs comparably when contrasted with state-of-the-art methods.

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 [6]. 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 run-time speeds should be improved.

Similarly, the study from Li et al. [41] 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 memorize and recognize 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.

Yet one can go further than surveillance. Zhao et al. [81] 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 [52]. 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.

Nguyen et al. [50] modeled the human-to-human multi-modal interactive behavior with LSTM. This behavior consists of speech, gaze and gestures which are jointly modeled. 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. [42]. 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.

4.6 Image and Video Captioning

So far, we have discussed a number of application domains. However, we can go further than tracking objects when it is about computer vision: a computer

can be requested to describe what can be seen in an image or a video. 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. The study of Chen et al. [7] 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 one sentence description closely relating to the input image with variable length words. Compared with four other relevant algorithms, the proposed model by Chen et al. [7] proved to be superior.

For more general image captioning, He et al. [24] 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 [60].

To increase the fluidity and descriptive nature of the generated image captions, a deep network was proposed by [35], consisting of three steps. The first step in their proposed method is, of course, system pre-processing. 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.

Liu et al. [43] 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.

4.7 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. For example, in [55] 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 having 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. [3]. In this work, three architectures are used: CNN to extract spatial features, LSTM for the temporal features, and finally convolutional LSTM for the spatiotemporal features, all in a stacked hierarchy. The simulations show that the hybrid model performs better than standard machine learning approaches for capturing the spatiotemporal characteristics for the citywide short-term crash risk prediction.

Several applications are identified in the field of computer science. The authors in [15] use a LSTM as a 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. [28] analysed the capability of CNN-LSTM to classify abnormalities related with ransomware activities. In [82] LSTM was applied to path planning in network traffic engineering with constrained conditions, which showed to be a superior method.

The network can also be found in the healthcare sector. For example, Pei et al. [53] adopted LSTM to map small bowel images to the corresponding diameters. In [40] an architecture was set up to recognize irregular entities in biomedical text. The contribution of Turan et al. [69] refers to a system that estimates the real-time pose for actively controlled endoscopic capsule robots. The authors in [2] 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.

In the field of sound recognition, Wöllmer and Schuller [74] 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 [38] 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 [12] 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.

A perhaps more exotic application of LSTM concerns block-sparsity recovery [46]. Here, the authors employed the LSTM framework for better capturing the correlations and dependencies among nonzero elements of signals. This pro-

posed method proved to be superior compared to alternative methods. Finally, Wang et al. [71] proposed a novel deep learning waveform recognition method, using two-channel CNNs combined with BLSTM. The contributors that this approach has a significantly better performance than the state-of-the-art.

5 Implementing LSTM in Keras

As discussed above, LSTM can be utilized 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 a generic LSTM classifier in Python, specifically using Keras.

First of all, in order to construct the LSTM network, we need to import a couple of modules from Keras. This can be done as follows:

Code Block 1 Importing libraries

```
from keras.models import Sequential
from keras.layers import Dense, LSTM, Dropout
```

With regards to the network construction itself, we need to instantiate Sequential, the model class to which we will add LSTM, Dropout [62] and Dense layers. The Dropout serves as countermeasure to over-fitting the data by dropping random nodes at each training stage. For instance, in case of a classification problem, this can be done as depicted below:

Code Block 2 Building the LSTM network

```
model = Sequential()

model.add(LSTM(unit=50, return_sequences=True,
    input_shape=(n_features, 1)))
model.add(Dropout(0.2))

model.add(LSTM(unit=50))
model.add(Dropout(0.2))

model.add(Dense(2, activation='softmax'))
model.compile(loss='categorical_crossentropy',
    optimizer='adam', metrics=['accuracy'])
```

To expand the model with a new layer, the add method is used. Inside the add method, we passed our LSTM layer. The first parameter to the LSTM layer refers to the number of neurons. The second parameter, return_sequences, determines whether to pass to the next layer the last output in the output sequence or the full sequences. The first entry to the input_shape parameter is the number of time steps while the last is the number of indicators. After that, to compile the model we use the method model.compile() with the adam optimizer and the loss function categorical_crossentropy.

Next, we use the backpropagation algorithm to fit the model by using 100 epochs with a batch size of 32. The model expects our data to be in a specific format, usually a three-dimensional array with training samples, timestamps, and features at each step. In the code below, X and Y represent the features and the target class, respectively, as we are illustrating a classification problem. The parameter train_size determines the proportion of the dataset to be used as the training set.

Code Block 3 Training the model

```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, train_size=0.8)

model.fit(X_train, Y_train, epochs=100, batch_size=32)
```

Finally, to exploit our network, we can use the method model.predict(X_test), or we can do as follows to evaluate the model:

Code Block 4 Measuring score and accuracy on test set

```
score, acc = model.evaluate(X_test, Y_test, batch_size=32)
print("Logloss score: %.2f" % (score))
print("Test accuracy: %.2f" % (acc))
```

As the reader can notice, building the model only takes a few steps in Keras, namely (1) building, (2) training and (3) evaluating the model. Alternatively, Python users can also opt for the Tensorflow framework.

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 modeling most of these problems, it was found a common practice is to hybridize 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.

Together with relevant LSTM applications, we have detailed the fundamental underpinnings behind this recurrent system including its main components, the interaction with each other and a gradient-based method to compute the weight matrix. The experimental study in [23] 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.

References

1. Álvaro, F., Sánchez, J.A., Benedí, J.M.: An integrated grammar-based approach for mathematical expression recognition. Pattern Recognition **51**, 135 – 147 (2016). DOI https://doi.org/10.1016/j.patcog.2015.09.013. URL http://www.sciencedirect.com/science/article/pii/S0031320315003441
2. 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). DOI https://doi.org/10.1016/j.eswa.2018.08.011. URL http://www.sciencedirect.com/science/article/pii/S0957417418305190
3. 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). DOI https://doi.org/10.1016/j.aap.2018.10.015. URL http://www.sciencedirect.com/science/article/pii/S0001457518303877
4. 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). DOI https://doi.org/10.1016/j.patcog.2018.07.034. URL http://www.sciencedirect.com/science/article/pii/S0031320318302590
5. 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). DOI https://doi.org/10.1016/j.patrec.2015.03.005. URL http://www.sciencedirect.com/science/article/pii/S0167865515000768. Pattern Recognition in Human Computer Interaction
6. Chen, B., Li, P., Sun, C., Wang, D., Yang, G., Lu, H.: Multi attention module for visual tracking. Pattern Recognition **87**, 80 – 93 (2019). DOI https://doi.org/10.1016/j.patcog.2018.10.005. URL http://www.sciencedirect.com/science/article/pii/S0031320318303509
7. 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). DOI https://doi.org/10.1016/j.patrec.2017.09.007. URL http://www.sciencedirect.com/science/article/pii/S016786551730315X
8. Chen, Y., Yang, J., Qian, J.: Recurrent neural network for facial landmark detection. Neurocomputing **219**, 26 – 38 (2017). DOI https://doi.org/10.1016/j.neucom.2016.09.015. URL http://www.sciencedirect.com/science/article/pii/S0925231216310165
9. Chowdhury, G.G.: Natural language processing. Annual Review of Information Science and Technology **37**(1), 51–89 (2003). DOI 10.1002/aris.1440370103. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/aris.1440370103
10. Dabiri, S., Heaslip, K.: Developing a twitter-based traffic event detection model using deep learning architectures. Expert Systems with Applications **118**, 425 – 439 (2019). DOI https://doi.org/10.1016/j.eswa.2018.10.017. URL http://www.sciencedirect.com/science/article/pii/S0957417418306651
11. 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). DOI https://doi.org/10.1016/j.eswa.2018.09.009. URL http://www.sciencedirect.com/science/article/pii/S0957417418305803
12. 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)
13. Elsheikh, A., Yacout, S., Ouali, M.S.: Bidirectional handshaking lstm for remaining useful life prediction. Neurocomputing **323**, 148 – 156 (2019). DOI https://doi.org/10.1016/j.neucom.2018.09.076. URL http://www.sciencedirect.com/science/article/pii/S0925231218311573

14. Fayek, H.M., Lech, M., Cavedon, L.: Evaluating deep learning architectures for speech emotion recognition. Neural Networks **92**, 60 – 68 (2017). DOI https://doi.org/10.1016/j.neunet.2017.02.013. URL http://www.sciencedirect.com/science/article/pii/S089360801730059X. Advances in Cognitive Engineering Using Neural Networks
15. 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). DOI https://doi.org/10.1016/j.adhoc.2018.09.014. URL http://www.sciencedirect.com/science/article/pii/S1570870518306887
16. 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). DOI https://doi.org/10.1016/j.ejor.2017.11.054. URL http://www.sciencedirect.com/science/article/pii/S0377221717310652
17. 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). DOI https://doi.org/10.1016/j.patcog.2013.06.030. URL http://www.sciencedirect.com/science/article/pii/S0031320313002823. Handwriting Recognition and other PR Applications
18. 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). DOI 10.1109/IJCNN.2000.861302
19. Gers, F., Schmidhuber, J., Cummins, F.: Learning to forget: Continual prediction with lstm. Neural computation **12**, 2451–71 (2000). DOI 10.1162/089976600300015015
20. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional lstm and other neural network architectures. Neural Networks **18**(5), 602 – 610 (2005). DOI https://doi.org/10.1016/j.neunet.2005.06.042. URL http://www.sciencedirect.com/science/article/pii/S0893608005001206. IJCNN 2005
21. 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
22. 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)
23. 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). DOI 10.1109/TNNLS.2016.2582924
24. 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). DOI https://doi.org/10.1016/j.patrec.2017.10.018. URL http://www.sciencedirect.com/science/article/pii/S0167865517303811. Deep Learning for Pattern Recognition
25. Hochreiter, S., Bengio, Y., Frasconi, P., Schmidhuber, J., Kremer, S.C., Kolen, J.F.: Gradient Flow in Recurrent Nets: The Difficulty of Learning Long-Term Dependencies. In: A Field Guide to Dynamical Recurrent Networks. Wiley-IEEE Press (2001). DOI 10.1109/9780470544037.ch14. URL http://ieeexplore.ieee.org/search/srchabstract.jsp?arnumber=5264952
26. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**, 1735–80 (1997). DOI 10.1162/neco.1997.9.8.1735
27. Hochreiter, S., Schmidhuber, J.: Lstm can solve hard long time lag problems. In: Advances in Neural Information Processing Systems 9, pp. 473–479. MIT Press (1997)
28. 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). DOI https://doi.org/10.1016/j.future.2018.07.045. URL http://www.sciencedirect.com/science/article/pii/S0167739X17328467
29. 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). DOI https://doi.org/10.1016/j.csl.2018.08.006. URL http://www.sciencedirect.com/science/article/pii/S0885230818300949

30. Hou, Q., Wang, J., Bai, R., Zhou, S., Gong, Y.: Face alignment recurrent network. Pattern Recognition **74**, 448 – 458 (2018). DOI https://doi.org/10.1016/j.patcog.2017.09.028. URL http://www.sciencedirect.com/science/article/pii/S0031320317303813
31. Kadari, R., Zhang, Y., Zhang, W., Liu, T.: Ccg supertagging via bidirectional lstm-crf neural architecture. Neurocomputing **283**, 31 – 37 (2018). DOI https://doi.org/10.1016/j.neucom.2017.12.050. URL http://www.sciencedirect.com/science/article/pii/S0925231217319124
32. Kafle, K., Kanan, C.: Visual question answering: Datasets, algorithms, and future challenges. Computer Vision and Image Understanding **163**, 3 – 20 (2017). DOI https://doi.org/10.1016/j.cviu.2017.06.005. URL http://www.sciencedirect.com/science/article/pii/S1077314217301170. Language in Vision
33. 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). DOI https://doi.org/10.1016/j.inffus.2018.09.001. URL http://www.sciencedirect.com/science/article/pii/S1566253518300460
34. 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). DOI https://doi.org/10.1016/j.csl.2018.06.005. URL http://www.sciencedirect.com/science/article/pii/S0885230818300913
35. Kinghorn, P., Zhang, L., Shao, L.: A hierarchical and regional deep learning architecture for image description generation. Pattern Recognition Letters **119**, 77 – 85 (2019). DOI https://doi.org/10.1016/j.patrec.2017.09.013. URL http://www.sciencedirect.com/science/article/pii/S0167865517303240. Deep Learning for Pattern Recognition
36. 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). DOI https://doi.org/10.1016/j.eswa.2018.10.002. URL http://www.sciencedirect.com/science/article/pii/S0957417418306432
37. Kumar Srivastava, R., Greff, K., Schmidhuber, J.: Training very deep networks. Neural Information Processing Systems (NIPS 2015 Spotlight) (2015)
38. 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). DOI https://doi.org/10.1016/j.eswa.2018.08.052. URL http://www.sciencedirect.com/science/article/pii/S0957417418305657
39. Lei, J., Liu, C., Jiang, D.: Fault diagnosis of wind turbine based on long short-term memory networks. Renewable Energy **133**, 422 – 432 (2019). DOI https://doi.org/10.1016/j.renene.2018.10.031. URL http://www.sciencedirect.com/science/article/pii/S0960148118312151
40. 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). DOI https://doi.org/10.1016/j.patrec.2017.06.009. URL http://www.sciencedirect.com/science/article/pii/S0167865517302155. Machine Learning and Applications in Artificial Intelligence
41. 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). DOI https://doi.org/10.1016/j.patcog.2017.01.030. URL http://www.sciencedirect.com/science/article/pii/S0031320317300389
42. 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). DOI https://doi.org/10.1016/j.cviu.2017.10.011. URL http://www.sciencedirect.com/science/article/pii/S1077314217301741
43. Liu, A.A., Xu, N., Wong, Y., Li, J., Su, Y.T., Kankanhalli, M.: Hierarchical & multi-modal video captioning: Discovering and transferring multimodal knowledge for vision to language. Computer Vision and Image Understanding **163**, 113 – 125 (2017). DOI https://doi.org/10.1016/j.cviu.2017.04.013. URL http://www.sciencedirect.com/science/article/pii/S1077314217300735. Language in Vision
44. Liwicki, M., Bunke, H.: Combining diverse on-line and off-line systems for handwritten text line recognition. Pattern Recognition **42**(12), 3254 – 3263 (2009). DOI https://

doi.org/10.1016/j.patcog.2008.10.030. URL http://www.sciencedirect.com/science/article/pii/S0031320308004652. New Frontiers in Handwriting Recognition
45. Lu, Z., Tan, H., Li, W.: An evolutionary context-aware sequential model for topic evolution of text stream. Information Sciences **473**, 166 – 177 (2019). DOI https://doi.org/10.1016/j.ins.2018.09.027. URL http://www.sciencedirect.com/science/article/pii/S0020025518307266
46. Lyu, C., Liu, Z., Yu, L.: Block-sparsity recovery via recurrent neural network. Signal Processing **154**, 129 – 135 (2019). DOI https://doi.org/10.1016/j.sigpro.2018.08.014. URL http://www.sciencedirect.com/science/article/pii/S0165168418302810
47. 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). DOI https://doi.org/10.1016/j.future.2018.09.033. URL http://www.sciencedirect.com/science/article/pii/S0167739X17326523
48. 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). DOI https://doi.org/10.1016/j.neucom.2015.11.030. URL http://www.sciencedirect.com/science/article/pii/S092523121501749X
49. 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). DOI https://doi.org/10.1016/j.neucom.2017.02.081. URL http://www.sciencedirect.com/science/article/pii/S0925231217304654
50. 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). DOI https://doi.org/10.1016/j.patrec.2017.09.033. URL http://www.sciencedirect.com/science/article/pii/S0167865517303525
51. 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). DOI https://doi.org/10.1016/j.patcog.2017.10.033. URL http://www.sciencedirect.com/science/article/pii/S0031320317304405
52. 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). DOI https://doi.org/10.1016/j.neucom.2018.10.009. URL http://www.sciencedirect.com/science/article/pii/S0925231218311858
53. 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). DOI https://doi.org/10.1016/j.knosys.2017.01.023. URL http://www.sciencedirect.com/science/article/pii/S0950705117300369
54. Phan, T.V., Nakagawa, M.: Combination of global and local contexts for text/non-text classification in heterogeneous online handwritten documents. Pattern Recognition **51**, 112 – 124 (2016). DOI https://doi.org/10.1016/j.patcog.2015.07.012. URL http://www.sciencedirect.com/science/article/pii/S0031320315002721
55. Portegys, T.E.: A maze learning comparison of elman, long short-term memory, and mona neural networks. Neural Networks **23**(2), 306 – 313 (2010). DOI https://doi.org/10.1016/j.neunet.2009.11.002. URL http://www.sciencedirect.com/science/article/pii/S0893608009002871
56. 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). DOI https://doi.org/10.1016/j.patrec.2014.11.007. URL http://www.sciencedirect.com/science/article/pii/S0167865514003572. Pattern Recognition in Human Computer Interaction
57. 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). DOI https://doi.org/10.1016/j.inffus.2018.07.007. URL http://www.sciencedirect.com/science/article/pii/S1566253517308175
58. 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). DOI https://doi.org/10.1016/j.patrec.2017.01.008. URL http://www.sciencedirect.com/science/article/pii/S0167865517300089
59. Sagheer, A., Kotb, M.: Time series forecasting of petroleum production using deep lstm recurrent networks. Neurocomputing **323**, 203 – 213 (2019). DOI https://doi.org/10.1016/j.neucom.2018.09.082. URL http://www.sciencedirect.com/science/article/pii/S0925231218311639
60. 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)
61. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing **45**(11), 2673–2681 (1997). DOI 10.1109/78.650093
62. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research **15**, 1929–1958 (2014)
63. Sukhbaatar, S., Weston, J., Fergus, R., et al.: End-to-end memory networks. In: Advances in neural information processing systems, pp. 2440–2448 (2015)
64. 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). DOI https://doi.org/10.1016/j.inffus.2018.04.001. URL http://www.sciencedirect.com/science/article/pii/S156625351730667X
65. 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)
66. Sun, Y., Ji, Z., Lin, L., Wang, X., Tang, D.: Entity disambiguation with memory network. Neurocomputing **275**, 2367 – 2373 (2018). DOI https://doi.org/10.1016/j.neucom.2017.11.013. URL http://www.sciencedirect.com/science/article/pii/S0925231217317277
67. Takayama, J., Nomoto, E., Arase, Y.: Dialogue breakdown detection robust to variations in annotators and dialogue systems. Computer Speech & Language **54**, 31 – 43 (2019). DOI https://doi.org/10.1016/j.csl.2018.08.007. URL http://www.sciencedirect.com/science/article/pii/S0885230818300901
68. 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). DOI https://doi.org/10.1016/j.patcog.2018.08.020. URL http://www.sciencedirect.com/science/article/pii/S0031320318303145
69. 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). DOI https://doi.org/10.1016/j.neucom.2017.10.014. URL http://www.sciencedirect.com/science/article/pii/S092523121731665X
70. 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). DOI https://doi.org/10.1016/j.jpdc.2018.08.010. URL http://www.sciencedirect.com/science/article/pii/S0743731518306270
71. 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). DOI https://doi.org/10.1016/j.sigpro.2018.09.038. URL http://www.sciencedirect.com/science/article/pii/S0165168418303256
72. 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). DOI https://doi.org/10.1016/j.csl.2018.09.003. URL http://www.sciencedirect.com/science/article/pii/S0885230818300834
73. 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 Applications **118**, 563 – 572 (2019). DOI https://doi.org/10.1016/j.eswa.2018.10.038. URL http://www.sciencedirect.com/science/article/pii/S0957417418306961
74. Wöllmer, M., Schuller, B.: Probabilistic speech feature extraction with context-sensitive bottleneck neural networks. Neurocomputing **132**, 113 – 120 (2014). DOI https://doi.org/10.1016/j.neucom.2012.06.064. URL http://www.sciencedirect.com/

science/article/pii/S0925231213008813. Innovations in Nature Inspired Optimization and Learning Methods Machines learning for Non-Linear Processing

75. 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). DOI https://doi.org/10.1016/j.physa.2018.09.120. URL http://www.sciencedirect.com/science/article/pii/S0378437118312536
76. Yan, H., Ouyang, H.: Financial time series prediction based on deep learning. Wireless Personal Communications **102**, 1–18 (2017). DOI 10.1007/s11277-017-5086-2
77. 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). DOI https://doi.org/10.1016/j.patcog.2016.11.011. URL http://www.sciencedirect.com/science/article/pii/S0031320316303697
78. 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). DOI https://doi.org/10.1016/j.patcog.2013.10.020. URL http://www.sciencedirect.com/science/article/pii/S0031320313004494
79. Zhang, M., Wang, Q., Fu, G.: End-to-end neural opinion extraction with a transition-based model. Information Systems **80**, 56 – 63 (2019). DOI https://doi.org/10.1016/j.is.2018.09.006. URL http://www.sciencedirect.com/science/article/pii/S0306437918301182
80. 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). DOI https://doi.org/10.1016/j.bspc.2018.08.035. URL http://www.sciencedirect.com/science/article/pii/S1746809418302337
81. 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). DOI https://doi.org/10.1016/j.neucom.2016.06.002. URL http://www.sciencedirect.com/science/article/pii/S0925231216305276. SI: BridgingSemantic
82. Zuo, Y., Wu, Y., Min, G., Cui, L.: Learning-based network path planning for traffic engineering. Future Generation Computer Systems **92**, 59 – 67 (2019). DOI https://doi.org/10.1016/j.future.2018.09.043. URL http://www.sciencedirect.com/science/article/pii/S0167739X18313244