RNN With Stacked Architecture for sEMG based Sequence-to-Sequence Hand Gesture Recognition

Philipp Koch*, Mark Dreier*, Marco Maass*, Huy Phan†, and Alfred Mertins*
*Institute for Signal Processing, University of Lübeck, Germany
†School of Electronic Engineering and Computer Science, Queen Mary University of London, United Kingdom
{koch, maass, mertins}@isip.uni-luebeck.de,
mark.dreier@student.uni-luebeck.de, h.phan@qmul.ac.uk

***Abstract*—Having proven their suitability for real world applications, surface electromyography signals are the means of choice for hand gesture recognition especially in medical applications like upper limb prosthesis. So far, mostly hand-crafted features combined with a standard classifier or neural networks are adopted for signal analysis. However, the performance of the standard approaches is insufficient and the networks are inappropriate for embedded applications due to their sheer size. To address these problems, a small recurrent neural network is proposed to fully utilize the sequential nature of the biosignals. Our network architecture features a special recurrent neural network cell for feature learning and extraction instead of convolutional layers and another type of cell for further processing. To evaluate the suitability of this inhomogenously stacked recurrent neural network, experiments on three different databases were conducted. The results reveal that this small network significantly outperforms state-of-the-art systems and sets new records. In addition, we demonstrate that it is possible to achieve relatively equal performance across all subjects.**

***Index Terms*—hand movement classification, surface electromyography, recurrent neural network, hand prosthesis**

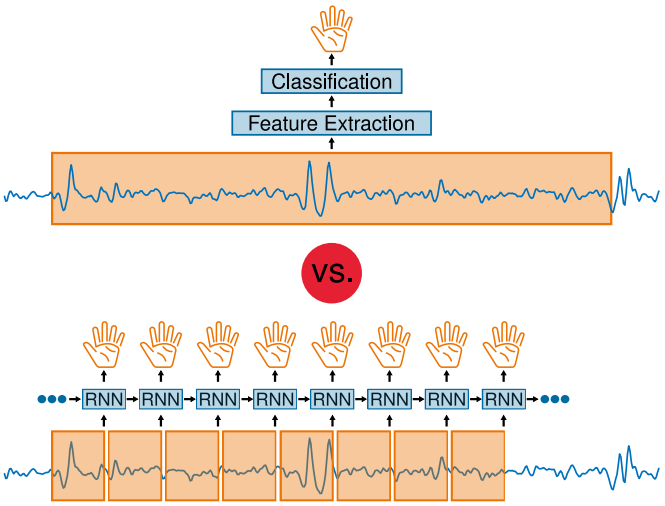

Fig. 1. Illustration of the differences between the standard classification approach (above) and the classification with an inhomogeneously stacked RNN (below).

I. Introduction

As interactions between humans and machines should become more intuitive and simpler, the interest in human-machine-interfaces increases. Arguably, a fairly convenient interaction for humans is through hand gestures [1], [2]. Especially the possible applications of hand gesture recognition systems that are not based on stationary external sensors like a camera system, but rely on wearable and mobile sensors, are versatile. By using biosignals acquired by wearable and relatively inexpensive sensors for decoding hand movements, hand gesture recognition systems become interesting for medical applications such as the control of hand prosthesis [3] or exoskeletons [4], [5].

Usually mobile systems, such as hand prosthesis control systems, use surface electromyography (sEMG) signals. The common hand gesture recognition systems are based on hand-crafted features and a conventional classifier like a random forest and support vector machine (SVM) [6]–[8]. Here feature extraction and classification are relatively computationally inexpensive. However, these approaches have some drawbacks such as the introduction of significant delays due to the required window length (at least 100 ms) and a limited classification accuracy.

With the emergence of deep neural networks, the recent research has focused more on such machine learning techniques. As convolutional neural networks (CNNs) are designed to be deployed on raw data and learning a feature extractor within the classifier, they are used to address the problem caused by the hand-crafted features, however, with mixed results [9], [10]. With CNNs it is possible to learn an individual feature extractor for each user. Furthermore, the networks can handle very short windows, but admittedly, for the final hand gesture, a voting across multiple consecutive windows is performed to achieve a better accuracy. The sheer size of the networks, making them nearly impossible to deploy on mobile systems, is worth noticing. Besides, a delay is introduced due to the voting, and arguably, a voting is not necessarily a feasible method as the temporal dependencies are ignored.

Lately, recurrent neural networks (RNNs) have been used to address the sequential nature of the sEMG signals [11], [12]. With these, many different hand movements can be distinguished with satisfying accuracy. Also the deployed networks are fairly small and the RNN concept is designed for the classification of a sequence of data, making it ideal for the

usage in a hand gesture recognition system. As standard RNNs are not particularly designed for processing raw data, the proposed networks rely on hand-crafted features, resulting in the need for longer windows (100 ms). However, a combination of different RNN cells has successfully been adopted to decode hand gestures from raw data of magnetometer, gyroscope, and accelerometer in combination and individually [13]–[15] (see Fig. 1 for an illustration of a standard and an RNN based approach). The reported results show the possibility of using small windows while achieving sufficient accuracies. In contrast to sEMG signals, these types of signals have not been proven to be applicable in real-world applications yet. So far they have only been used in a few research projects and could only shine under these experimental conditions. Note that in all publications, the signals of all three modalities were upsampled by a high factor.

Since sEMG signals are the means of choice especially for practical applications and hand-crafted features appear to be a weak point, an RNN for processing raw sEMG signals is developed in contrast to previous works. The proposed architecture is small and consists of two different types of RNN cells, one designed for feature extraction and utilizing spatial dependencies within the data and a standard one for processing the extracted features. Unlike in most of the previous publications, the training of the RNN is performed in a sequence-to-sequence manner as it is assumed that this helps with regularization and correctly classifying beginnings of sequences. The windows presented to the network are just 5 ms long, allowing for preventing the introduction of unnecessary delays that significantly impact usability of a hand gesture recognition system. The experiments conducted on three different databases clearly reveal the suitability of the proposed RNN since state-of-the-art results are obtained for both able-bodied and amputated subjects.

II. Mathematical Formulation of the Classification Problem

Before presenting the proposed approach, a mathematical formulation of the actual task is introduced. Let $\boldsymbol{X_t}$ denote the representation of raw data belonging to a window of time t and $\mathbb{L}$ be the set containing all possible labels, the classification problem can be described as

$$\mathcal{C} : (\boldsymbol{X}_{t-i\cdot\eta})_{i=0}^{N-1} \longmapsto y_{t+\tau}, \quad t \geq i \cdot \eta \tag{1}$$

with $y_{t+\tau} \in \mathbb{L}$ being the corresponding label, N the number of processed sequence elements and η the hop size. This formulation allows us to incorporate sequential information into the classification processes. In general, for hand gesture recognition a stream of data is recorded resulting in constantly growing sequences. To address this, we reformulate the classification problem. Up to time T the mapping

$$\mathcal{C}_{\text{seq}} : (\boldsymbol{X}_{t-i\cdot\eta})_{i=0}^{N-1} \longmapsto (y_{t+\tau-i\cdot\eta})_{i=0}^{N-1}, \quad t \geq i \cdot \eta \tag{2}$$

with $y_{t+\tau-i\cdot\eta} \in \mathbb{L}$ for $i \in \{0, 1, \ldots, N-1\}$ is performed. In this work, we set the hop $\eta \geq 0$ to the window size to create a sequence of consecutive, but not overlapping windows. In order to avoid delays, the hand gesture should be assigned to the end of the window. Therefore, the offset $\tau \geq 0$ is chosen to be half of the window length. Note that in contrast to other approaches, this formulation does not cause a delay of half of the window size as the label is assigned to the end of the window instead of classifying the gesture belonging to the window center.

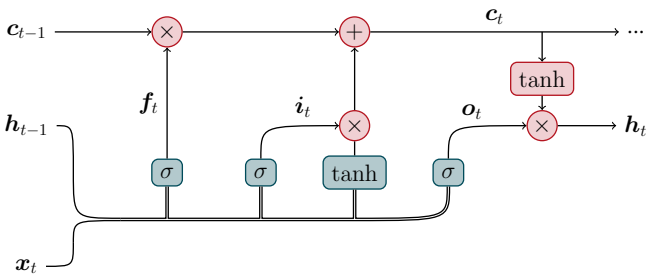

Fig. 2. Illustration of the concept behind an LSTM cell.

III. Inhomogeneously Stacked RNN Architecture

RNNs are networks particularly developed for processing sequences. These networks are capable of producing an output for each element of a sequence before processing the consecutive sequence element. Furthermore, the principles of RNNs enable processing an element with respect to its predecessors in the sequence. Consequently, RNNs used in a sequence-to-sequence manner are suitable to perform the sequential mapping stated in (2).

As already implied, a key aspect of this work is to exploit temporal and spatial dependencies within the sEMG data. Therefore, by design, the network should be capable of inherently utilizing sensor positions as well as the waveform of the acquired signals. Consequently, in this work, we use a slightly modified convolutional LSTM (ConvLSTM) cell [16] as an upstream feature extractor directly processing the sequence of raw data followed by a standard long short-term memory (LSTM) cell [17].

A. LSTM Cell

The LSTM cell is probably one of the most popular RNN cells having shown its versatility on many different tasks. An illustration of its basic principle is depicted in Fig. 2. The recursive nature of this cell is quite obvious. The cell state $\boldsymbol{c}_{t-1}$ is carefully updated by so-called gates which manipulate the state based on the current input $\boldsymbol{x}_t$ and the previous cell output $\boldsymbol{h}_{t-1}$. The final output is computed by multiplying the output gate $\boldsymbol{o}_t$ with the updated state $\boldsymbol{c}_t$. Mathematically the LSTM cell can be described through its nonlinear hidden layer function $\mathcal{H}$

$$(\boldsymbol{h}_t, \boldsymbol{c}_t) = \mathcal{H}(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}, \boldsymbol{c}_{t-1}) \tag{3}$$

where $t \in \mathbb{N}$ denotes the current timestep. With $\odot$ representing the Hadamard product, the update of the cell state is given by

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tanh\left(\boldsymbol{W}_{xc}\boldsymbol{x}_t + \boldsymbol{W}_{hc}\boldsymbol{h}_{t-1} + \boldsymbol{b}_c\right) \tag{4}$$

with the gates being

$$\boldsymbol{i}_t = \sigma\left(\boldsymbol{W}_{xi}\boldsymbol{x}_t + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{b}_i\right) \tag{5}$$

and

$$\boldsymbol{f}_t = \sigma\left(\boldsymbol{W}_{xf}\boldsymbol{x}_t + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{b}_f\right). \tag{6}$$

As mentioned before, the final output depends on the output gate

$$\boldsymbol{o}_t = \sigma\left(\boldsymbol{W}_{xo}\boldsymbol{x}_t + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{b}_o\right) \tag{7}$$

and is defined as

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh\left(\boldsymbol{c}_t\right). \tag{8}$$

Note that all of the previous matrices $\boldsymbol{W}_{w\in\mathbb{W}}$ with $\mathbb{W} = \{xc, xi, xf, xo, hc, hi, hf, ho\}$ represent trainable matrices and the vectors $\boldsymbol{b}_{b\in\mathbb{B}}$ with $\mathbb{B} = \{c, i, f, o\}$ the trainable biases.

B. Modified ConvLSTM Cell

In contrast to the LSTM cell which processes only one-dimensional input data, the ConvLSTM cell works with multi-dimensional tensors. For instance, this allows us to encode the relative position of sensors within the data and thus present it to the network. To fully explore the additional information, the principle of convolutional layers, which are the core of CNNs, is adopted. This means that all $\boldsymbol{W}_{w\in\mathbb{W}}$ matrices are replaced by one or multiple trainable convolutional kernels which are then convolved with the current input $\boldsymbol{X}_t$ or the previous output $\boldsymbol{H}_{t-1}$. Consistently, all bias vectors of the LSTM cell become tensors in a ConvLSTM cell.

Convolution operations usually lead to smaller outputs compared to the inputs. In a ConvLSTM cell this shrinkage is problematic as it causes dimension mismatches. Usually, the problem is addressed by zero-padding the input before performing the convolution. As in our case the input tensors are quite small, the padding can have a major impact. Knowing that usually the electrodes are roughly arranged as circles or grids placed around the forearm, we perform a cyclic padding for dimensions representing electrode positions. Because the processed windows are quite short, symmetric or constant repetition padding is used for the time dimension in order to prevent potentially rapid changes that would be introduced by zero padding. For an input with a single sensor dimension and a 3×3 convolutional kernel, the padded input is given by

$$\left[\begin{array}{c|cccccc|c}
x_{1,12} & x_{1,1} & x_{1,2} & x_{1,3} & \cdots & x_{1,12} & x_{1,1} \\ \hline
x_{1,12} & x_{1,1} & x_{1,2} & x_{1,3} & \cdots & x_{1,12} & x_{1,1} \\
x_{2,12} & x_{2,1} & x_{2,2} & x_{2,3} & \cdots & x_{2,12} & x_{2,1} \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
x_{10,12} & x_{10,1} & x_{10,2} & x_{10,3} & \cdots & x_{10,12} & x_{10,12} \\ \hline
x_{10,12} & x_{10,1} & x_{10,2} & x_{10,3} & \cdots & x_{10,12} & x_{10,12}
\end{array}\right].$$

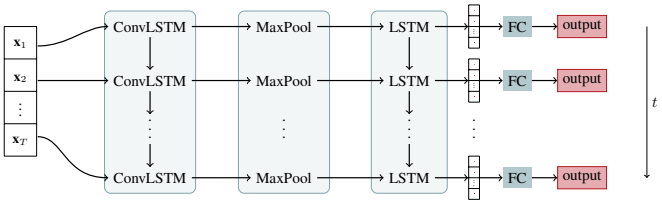

Fig. 3. Illustration of the proposed network architecture enrolled over time.

C. Architecture

Before explaining the actual network architecture, it is worth elaborating on the network's input in order to emphasize the versatility of the approach. As mentioned implicitly, the network can handle tensor inputs. Consequently, it is possible to represent a variety of different sensor placements, even a combination of implanted and surface sensors. Note that if the sensor positions just roughly correspond to the grid positions, it generally does not affect the network's performance. Even if some sensors are placed completely independently from the regular placement scheme, and probably be placed quite far away from the rest, the network can adapt to that if they are assumed to be part of the grid, as the results will show.

The proposed network uses two kinds of RNN cells and is illustrated in Fig. 3. As one would expect, firstly a ConvLSTM cell is deployed to process the raw sEMG signals. By using multiple (in our case 24) convolution kernels the ConvLSTM cell is capable of performing complex analysis. As with a growing number of kernels the size of the state and consequently the output increases. In order to reduce the dimensionality, the ConvLSTM cell is followed by a max-pooling layer that shrinks the cell's output to half of its original size. Also, it adds an additional nonlinearity to the network. Next, the resulting tensor is vectorized and presented to a standard LSTM cell. For this, we chose a cell with state size of 256. The final output of the network is then produced by a fully-connected layer with a softmax activation function, as we solve a classification problem.

IV. Training and Validation

The network is trained and tested in sequence-to-sequence fashion. For training we used sequences of fixed length T. This can help to accelerate training and makes data augmentation, in our case oversampling, easy. By oversampling we mean that from a training signal, multiple highly overlapping sequences are extracted. It is a well-known strategy to increase the amount of training data.

The network's optimization is based on the cross-entropy loss calculated for all time steps. In the following, the ground-truth label of time step t is represented as one-hot encoded column vector $\boldsymbol{y}_t$ and the network's prediction of it is given by the vector $\hat{\boldsymbol{y}}_t$. With matrix $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_T)$ representing the ground truth vectors of a sequence and $\hat{\boldsymbol{Y}} = (\hat{\boldsymbol{y}}_1, \hat{\boldsymbol{y}}_2, \ldots, \hat{\boldsymbol{y}}_T)$

TABLE I
OVERVIEW OVER THE DIFFERENT DATASETS.

	DB II	DB III	DB VII
Subjects (able-bodied / amputated)	40 / 0	0 / 11	20 / 2
Repetitions	6	6	6
Number of hand gestures	50	50	40

being the corresponding predictions of a network, the loss is calculated by

$$E\left(\boldsymbol{Y}, \hat{\boldsymbol{Y}}\right) = \sum_{t=1}^{T} -\boldsymbol{y}_t^{\top} \log\left(\hat{\boldsymbol{y}}_t\right). \tag{9}$$

The optimizer used to minimize this loss function is the Adam optimizer [18]. To regularize the network during training, we applied dropout [19] with the dropout probability set to $50\,\%$. By randomly switching neurons off through setting all weights to zero, overfitting can be prevented. In order to further improve and accelerate the optimization process, mini batches with a size of 200 were used.

In testing, the entire sequences were classified even though they differed in length. So, unlike in training, the test sequences were not split into smaller sequences of same length. This is done, because in an actual application, e.g. in a hand prosthesis, a stream of data has to be constantly analysed to predict the current movement. This is arguably a fair setting for testing the network as it is close to the tasks in actual applications. For the evaluation of the network's performance, the network predictions of every window of a sequence were taken into account and compared with the corresponding ground truth.

V. EXPERIMENTS

A. Databases

In order to evaluate the capabilities of the network, we conducted experiments on publicly available databases allowing for comparisons with state-of-the-art approaches. We used the databases DB II and DB III [6] as well as DB VII [22]. In all three databases the signals were acquired with sEMG electrodes. In DB II and DB III, the data were recorded by a Delsys® Trigno™Wireless system while in DB VII Delsys® Trigno™IM Wireless sensors were used. Most of the sensors were placed in a ring around the subjects' forearm, except for a few sensors that were positioned on certain muscles located in the upper and lower arm. Generally, the subjects were equipped with 12 electrodes except for some amputees for which the number of electrodes had to be reduced as their amputation prevented the correct placement of all electrodes. After equipping the subjects with the electrodes, all subjects were asked to perform a variety of different hand movements while the data were recorded. All movements were repeated several times and separated by a resting period. An overview of the databases is given in Table I.

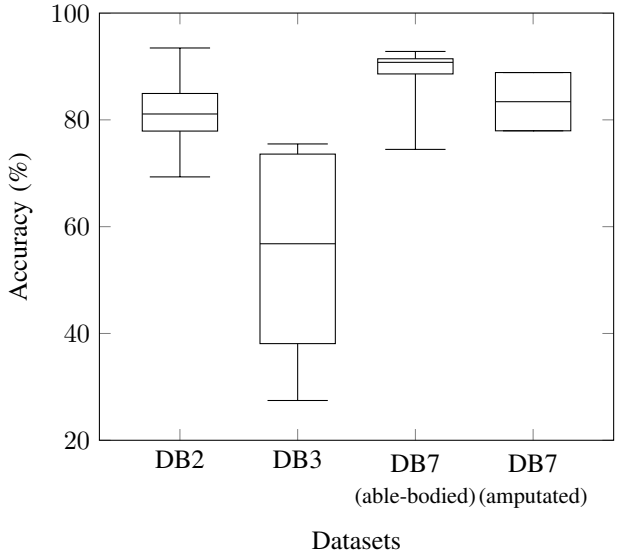

Fig. 4. Boxplots of the obtained results.

B. Data Handling

The data of each database were split into two disjunct sets for training and testing according to [6], [22]. By following the suggested data splitting, the obtained results can be compared to those of previous works. We chose a window size of $5\,\mathrm{ms}$, as this has turned out experimentally to be a suitable choice for such RNN approaches. In training the sequence length was set to $1\,\mathrm{s}$. Before applying the network, the data were normalized to achieve zero mean and unit standard deviation. This normalization was performed for each sensor individually and the necessary statistical parameters were calculated using the training data only.

Even though the sensors might not all be placed circularly around the forearm, the data were organized as if they were arranged according to the circular scheme. This means we created an input matrix for the network where the rows represent the sensors in correct order as much as possible and the columns belong to the samples of the window.

C. Results

For DB II and DB III, the reported results are the averages calculated across all subjects of a database while for DB VII the medians are reported. As DB VII contains able-bodied subjects and amputees, the results are reported for both groups separately.

The obtained results are shown in Table II. The comparison reveals that the proposed network outperforms the state-of-the-art approaches for all three databases. Regarding DB II, the performance of our network is only nearly matched by a CNN-RNN system [21] even though this system is a large network containing several convolutional and locally-connected layers as well as RNN cells making it inapplicable for embedded applications. The proposed system significantly outperforms the approach proposed in [12] that uses $100\,\mathrm{ms}$ long windows, hand-crafted features, and a single RNN cell. On DB VII we

TABLE II
COMPARISON OF THE OBTAINED ACCURACIES WITH THOSE OF STATE-OF-THE-ART SYSTEMS. A $\sim$ DENOTES THAT THE EXACT ACCURACY IS UNKNOWN.

System	window length	DB II	DB III	DB VII (able-bodied)	DB VII (amputated)
Proposed RNN	5 ms	82.81 %	60.58 %	92.30 %	85.41 %
Self-Recalibration system based on CNN [20]	200 ms	78.71 %	$\sim$ 40 %	–	–
CNN-RNN [21]	200 ms	82.20 %	–	–	–
Feature plus RNN [12]	100 ms	78.0 %	55.3 %	–	–
Features plus Random Forest [6]	200 ms	75.27 %	–	–	–
Features plus SVM [6]	200 ms	–	46.27 %	–	–
Features plus LDA [22]	256 ms	–	–	$\sim$ 60 %	$\sim$ 42 %

achieve outstanding performance even when comparing to results obtained by a system relying on a standard classification pipeline using the data of multiple modalities (accelerometer, gyroscope, magnetometer, and sEMG electrodes) which achieved an accuracy of $81.7\,\%$ for able-bodied persons and an accuracy of $77.7\,\%$ for amputees [22]. These results emphasise the suitability of a dedicated RNN cell for feature extraction.

Fig. 4 shows boxplots of the obtained results for different databases. As one can see, for DB II and DB VII, the variation between subjects is relatively small, whereas it is significantly larger for DB III. A major reason for the variation in performance on DB III may be the fact that there are large anatomical differences between the amputated subjects. A comparison with the results reported in [6], [22] reveals that even for DB III the performance of our network varies relatively little from subject to subject compared to other methods in the literature. These results suggest that it is beneficial for the robustness and reliability of a gesture recognition system to train the combination of feature extraction and classifier end-to-end for each subject individually in a data-driven fashion.

VI. CONCLUSION

Within the scope of this work we showed that it is possible to reliably classify hand movements with small RNNs. The proposed inhomogenously stacked sequence-to-sequence RNN outperforms state-of-the-art systems and even proves to work stable for all subjects. Note that the network works especially well for amputated subjects making it interesting for the application in upper limb prosthesis. As the network is designed for processing sequences and returns an output for a given window before processing the next one, it is especially suitable for real time applications. Taking also into account that the RNN requires $5\,\mathrm{ms}$ short windows and therefore helps to avoid delays, the proposed system is a promising approach for further research and possible applications.

REFERENCES

[1] J. Cheng, X. Chen, Z. Lu, K. Wang, and M. Shen, "Key-press gestures recognition and interaction based on sEMG signals," in *Proc. Int. Conf. Multimodal Interact. and Mach. Learn. Multimodal Interact.*, 2010.

[2] F. Muri, C. Carbajal, A. M. Echenique, H. Fernández, and N. M. López, "Virtual reality upper limb model controlled by EMG signals," *J. Phys. Conf. Ser.*, vol. 477, 2013.

[3] C. Cipriani, F. Zaccone, S. Micera, and M. C. Carrozza, "On the shared control of an EMG-controlled prosthetic hand: Analysis of user-prosthesis interaction," *IEEE Trans. Robot.*, vol. 24, no. 1, pp. 170–184, 2008.

[4] J. Rosen, M. Brand, M. B. Fuchs, and M. Arcan, "A myosignal-based powered exoskeleton system," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 31, no. 3, pp. 210–222, 2001.

[5] K. Kiguchi and Y. Hayashi, "An EMG-based control for an upper-limb power-assist exoskeleton robot," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 4, pp. 1064–1071, 2012.

[6] M. Atzori, A. Gijsberts, C. Castellini, B. Caputo, A.-G. Mittaz Hager, S. Elsig, G. Giatsidis, F. Bassetto, and H. Müller, "Electromyography data for non-invasive naturally-controlled robotic hand prostheses," *Sci. Data*, vol. 1, no. 140053, 2014.

[7] X. Zhang, X. Chen, Y. Li, V. Lantz, K. Wang, and J. Yang, "A framework for hand gesture recognition based on accelerometer and EMG sensors," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 41, no. 6, pp. 1064–1076, 2011.

[8] K. Englehart and B. Hudgins, "A robust, real-time control scheme for multifunction myoelectric control," *IEEE Trans. Biomed. Eng.*, vol. 50, no. 7, pp. 848–854, 2003.

[9] M. Atzori, M. Cognolato, and H. Müller, "Deep learning with convolutional neural networks applied to electromyography data: A resource for the classification of movements for prosthetic hands," *Front. Neurorobot.*, vol. 10, no. 9, 2016.

[10] W. Geng, Y. Du, W. Jin, W. Wei, Y. Hu, and J. Li, "Gesture recognition by instantaneous surface EMG images," *Sci. Rep.*, vol. 6, no. 36571, 2016.

[11] P. Koch, H. Phan, M. Maass, F. Katzberg, and A. Mertins, "Recurrent neural network based early prediction of future hand movements," in *Proc. IEEE Eng. Med. Biol. Soc.*, 2018.

[12] P. Koch, H. Phan, M. Maass, F. Katzberg, R. Mazur, and A. Mertins, "Recurrent neural networks with weighting loss for early prediction of hand movements," in *Proc. Eur. Signal Process. Conf.*, 2018.

[13] P. Koch, M. Dreier, M. Maass, M. Böhme, H. Phan, and A. Mertins, "A recurrent neural network for hand gesture recognition based on accelerometer data," in *Proc. IEEE Eng. Med. Biol. Soc.*, 2019.

[14] P. Koch, M. Dreier, M. Böhme, M. Maass, H. Phan, and A. Mertins, "Inhomogeneously stacked RNN for recognizing hand gestures from magnetometer data," in *Proc. Eur. Signal Process. Conf.*, 2019.

[15] P. Koch, N. Brügge, H. Phan, M. Maass, and A. Mertins, "Forked recurrent neural network for hand gesture classification using inertial measurement data," in *Proc. IEEE Int. Conf. Acoust. Speech Signal*, 2019.

[16] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proc. 28th Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 802–810.

[17] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[18] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2014.

[19] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[20] X. Zhai, B. Jelfs, R. H. M. Chan, and C. Tin, "Self-recalibrating surface EMG pattern recognition for neuroprosthesis control based on convolutional neural network," *Front. Neurosci.*, vol. 11, no. 379, 2017.

[21] Y. Hu, Y. Wong, W. Wei, Y. Du, M. Kankanhalli, and W. Geng, "A novel attention-based hybrid CNN-RNN architecture for sEMG-based gesture recognition," *PLOS ONE*, vol. 13, pp. 1–18, 2018.

[22] A. Krasoulis, I. Kyranou, M. S. Erden, K. Nazarpour, and S. Vijayakumar, "Improved prosthetic hand control with concurrent use of myoelectric and inertial measurements," *J. Neuroeng. Rehabil.*, vol. 14, no. 71, 2017.