Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/MBE

MBE, 17(5): 5432–5448.
DOI: 10.3934/mbe.2020293
Received: 14 May 2020
Accepted: 20 July 2020
Published: 12 August 2020

Research article

Multi-Stroke handwriting character recognition based on sEMG using convolutional-recurrent neural networks

Jose Guadalupe Beltran-Hernandez, Jose Ruiz-Pinales*, Pedro Lopez-Rodriguez, Jose Luis Lopez-Ramirez and Juan Gabriel Avina-Cervantes

Digital Signal Processing and Telematics Groups, Engineering Division of the Campus Irapuato-Salamanca (DICIS), Universidad de Guanajuato, Palo Blanco, Salamanca, Guanajuato, Mexico

* **Correspondence:** Email: pinales@ugto.mx; Tel: +524646479940.

Abstract: Despite the increasing use of technology, handwriting has remained to date as an efficient means of communication. Certainly, handwriting is a critical motor skill for childrens cognitive development and academic success. This article presents a new methodology based on electromyographic signals to recognize multi-user free-style multi-stroke handwriting characters. The approach proposes using powerful Deep Learning (DL) architectures for feature extraction and sequence recognition, such as convolutional and recurrent neural networks. This framework was thoroughly evaluated, obtaining an accuracy of 94.85%. The development of handwriting devices can be potentially applied in the creation of artificial intelligence applications to enhance communication and assist people with disabilities.

Keywords: surface EMG; long short-term memory; gated recurrent unit; convolutional neural networks

1. Introduction

Handwriting is a complex and essential communication skill for humans that is often disregarded. Nonetheless, handwriting (both manuscript and cursive) is a critical motor skill for children's cognitive development and academic success [1]. Considering as an example the case of a musical composition, experts in this area show a preference for handwriting editions over the use of specialized, user-friendly interfaces. The latter due to the lack of confidence in using a computer for artistic applications. Despite this fact, the automatic detection of hand-written musical notation in digital media is highly recommended and profitable in such an application, similar to characters recognition in text analysis. Calvo-Zaragoza and Oncina proposed a music notation recognition method based on an electronic pen

and a finite-state machine to detect user strokes competitively [2].

In hand gesture analysis, Mendes et al. studied the interaction human and robot based on gesture recognition by developing an Electromyography sensor to capture and discriminate valid gestures using a Deep Learning technique [3].

Analogously, Słapekand and Paskiel proposed a method based on Bèzier curves to extract predefined hand gestures without using a marker at the start or end in the sequence of points [4]. Such type of methods tries to detect the handwriting style to develop improved methodologies in smart human-computer interfaces and handwriting recognition.

On the other hand, the development of brain-computer interfaces offers an alternative for clinical and assisted applications involving motor biosignals. For instance, the communication abilities of people with or without disabilities can be enhanced through devices interfaced with Electroencephalography (EEG) and/or Electromyography (EMG) signals.

There are two kinds of EMG [5]: Surface EMG and intramuscular EMG. Intramuscular EMG is a technique that is performed by inserting electrodes into a skeletal muscle. In contrast, surface EMG is performed by placing electrodes on the skin surface above a muscle. Compared to intramuscular EMG, surface EMG has the advantage of being a non invasive technique which is clinically superior in the diagnosis, rehabilitation, and testing of new therapies linked to motor activity. Surface EMG (sEMG) signals form an intricate interference pattern of bioelectrical signals generated mainly from the dynamical activity of skeletal muscles. As a consequence, sEMG signals from individual muscles cannot be reliably distinguished.

Additionally, sEMG has found several new applications supported by Machine Learning and Deep Learning methods as movement analysis, muscular fatigue, sports therapies, wearable devices, and particularly in handwriting recognition [6]. sEMG data acquisition and the corresponding waveforms are affected by many factors such as the muscles' physiological properties, added noise in the instrumentation used for sensing, as well as the chosen location of the electrodes related to the active muscle units. Hence, conventional features extraction methods involving sEMG generally require robust pre-processing steps and filtering. Moreover, the sEMG signal complexity can also be explained by the fact that there are various ways in which a motor task can be performed (motor redundancy) [7, 8]. Simao et al. made a recent review of the sEMG-based systems using pattern recognition techniques and describing sensitive processes as signal acquisition and filtering. Their study focused on new human-machine interaction modalities [9].

Recently, advances in biomedicine have made possible the invention of myoelectric prostheses capable of performing real arm or leg movements [10]. Neural prostheses strive to restore the mobility of limbs, and the ability to communicate of subjects with disabilities, through the interconnection of brain potentials, actuators, or artificial devices in functional electrical stimulation. A significant innovation was the implementation of a device to predict a premature birth several weeks in advance. In this context, bioelectric interfaces allow extracting handwriting patterns for diagnostic purposes in patients with Parkinson's disease, and more recently, dysgraphia disability. Therefore, sEMG signals generated during the handwriting can be used to characterize neurological disorders [11]. In handwriting, the sEMG signals are generated and captured from the hand and forearm muscles during the handwriting activity, to be decoded by the specific purpose recognition systems [12]. In early studies, Linderman et al. developed a reliable system using sEMG signals to recognize handwriting characters to diagnose or measure disease progression in medical treatments [13].

In general, handwriting recognition from sEMG signals is addressed using the following approaches: Template Matching [13], Dynamic Time Warping (DTW) [12, 14], Radial Basis Functions (RBF) Neural Networks [11]. It is noteworthy that sEMG signals convey a great amount of information about limb movements. Following this reasoning, several applications have been proposed in the literature for the recognition, modeling, and reconstruction of handwriting movements. Additionally, the reconstruction of handwriting from sEMG signals has been addressed using the Wiener filters [13], Recursive Least Squares [15], RBF neural networks [16], and Kalman filters [17]. Therefore, these advances are not only limited to medical purposes, but also toys and video games can be controlled through the musculatures bioelectric activity. In a related work, Wei et al. devised a gesture recognition method by using multi-view features for both unimodal and multimodal sEMG-based gestures driven by a Deep convolutional neural network (DCNN) model [18].

Dash et al. [19] designed an approach called AirScript, for visualizing and recognizing characters written in the air. In this case, IMU (Inertial Measurement Unit) data from the Myo armband were applied to obtain a 2D coordinates sequence to visualize a handwritten digit as an image. For handwritten digit recognition, a convolutional neural network (CNN), and two GRU (Gated Recurrent Unit) neural networks were analyzed. The images were used as inputs to the CNN, the generated sequence of 2D coordinates was utilized as input to the first GRU neural network, and the raw IMU signal as input to the second GRU neural network. Subsequently, the outputs of those three neural networks were fused to yield the class of the recognized digit. The system reached an accuracy of 91.7 and 96.7% for independent and dependent tasks, respectively. In the method implemented by Roy et al. [20], single stroke digits were written in the air using a marker of uniform color. A video camera recorded the handwriting trajectory. After segmenting each video frame, the marker tip position was used to approximate the handwriting trajectory and to obtain an image of the handwritten digit. Next, a CNN was used for recognizing the image of each handwritten digit reporting an accuracy of 97.7, 95.4 and 93.7% for the English, Bengali, and Devanagari digits, respectively.

In some studies, the recognition of handwriting characters was limited to use small databases, imposing several restrictions on the placement of the electrodes, and in the number and complexity of handwriting strokes. Most handwriting recognition methods using sEMG signals were mainly based on pre-processing and feature extraction [13], while some others use the DTW method after pre-processing [14].

The approach discussed in this paper addreses handwriting character recognition problem from a more general point of view. To cope with the variability of the sEMG signals, the use of CNNs is suggested for the signal processing and feature extraction [21, 22]. Hence, a Recurrent Neural Network (RNN) based on the Long Short-Term Memory (LSTM) and GRU networks is used to recognize sEMG signals, considering that LSTMs have also been previously applied successfully to gesture recognition [23]. Implemented CNN architectures are based on several studies demonstrating the efficacy and robustness of CNNs for extracting discriminant features directly from raw data. CNNs use a multi-level representation to learn and decode original data from specialized layers applied to an optimal recognition task [24].

2. Materials and methods

2.1. Convolutional neural networks

A convolutional neural network (CNN), is a well-known Deep Learning architecture originally inspired by findings in visual perception. In 1959, Hubel & Wiesel discovered the existence of simple cells in the visual cortex of cats [25]. Simple cells are neurons that respond to bars or edges directed to a specific orientation. Hubel & Wiesel also recommended a hierarchical model to explain the behavior of simple cells. According to Marr [26], the receptive fields of simple cells are either bar o edge-shaped, and they compute a linear convolution with a bar or edge-shaped mask. Inspired on these findings, Fukushima suggested the Neocognitron in 1980 [27], which perhaps constitutes one of the first CNN architectures. The Neocognitron introduced two basic layer types: Convolutional layer and subsampling layer. Weng et al. proposed the Cresceptron architecture, which introduced max-pooling [28] and a variant called blurring [29].

In 1990, LeCun et al. created the LeNet architecture to classify handwritten digits [30]. The LeNet architecture and variants were the first to be trained with the backpropagation algorithm [31]. This architecture obtained effective representations directly from the image, which made it possible to recognize visual patterns from raw pixels with little-to-none pre-processing. At the same time, Zhang et al. used a Shift-Invariant Artificial Neural Network (SIANN) to recognize characters in an image [32].

In the beginning, despite the success of the LeNet-5 for digit recognition, CNNs were not successful for more complex tasks. The main reasons included the low computing power available, the small number of training samples, and diverse problems that arose especially for Deep CNNs (DCNN) [33]. After several efforts to surmount the difficulties for training DCNNs and thanks to the availability of new powerful computing resources, Krizhevsky et al. presented the AlexNet architecture, which won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC2012) [34]. The overall AlexNet architecture [35] is basically the same as the LeNet-5 architecture but considerably deeper, with stacked convolutions and with more filters per layer. Among the innovations incorporated in AlexNet, were the use of the Rectified Linear Unit (ReLU) activation functions [36], max-pooling layers, dropout [37], data augmentation [38], and ad-hoc initialization algorithms, Jiuxiang Gu et al. detail the recent improvements of CNNs related to the aforementioned innovations [33].

Several years later, He et al. proposed the ResNet architecture, which won the ILSVRC 2015 [39]. It was several times deeper than AlexNet and incorporated innovations such as residual blocks, and batch-normalization layers [40].

2.2. Recurrent neural networks

Recurrent neural networks are a class of neural networks originally conceived as en effort to incorporate time in neural networks. In 1986 Jordan et al. [41] proposed a neural network with dynamic memory which incorporated recurrent connections from the output layer to the hidden layer. Several years later, Elman et al. [42] proposed another recurrent network based on the Jordan network. This model incorporated recurrent connections directly from the output of the hidden layer allowing the network to maintain a sort of state or memory in the hidden layer. This allowed the network to perform difficult tasks such as temporal sequence prediction, sequence-to-sequence mapping, etc.

Recurrent neural networks are trained by using Backpropagation-Through-Time (BPTT). First, the network is unfolded in time. As a result, a very deep shared weights feedforward neural network is obtained. Then training is performed by using the back-propagation algorithm. At that time, recurrent neural networks had a limited success because they presented learning problems when dealing with very long sequences [43, 44]. This was called the vanishing gradient problem because the backpropagated gradients used to update the weights vanished rapidly to zero. As a result, the network was not able to learn patterns contained at the beginning of the sequence. This means that the network was not able to learn long-range dependencies.

2.2.1. Long short-term memory

In 1997 Hochreiter & Schmidhuber [45] proposed Long Short-Term Memory (LSTM) networks as solution to the vanishing gradient problem. The basic structure of an LSTM network is composed of a chain of units that can decide when to remember, forget, and acquire new information. Consequently, an LSTM unit is composed of one memory cell and three multiplicative gates: Forget, input, and output gates. Figure 1 illustrates the basic structure of an LSTM unit, including in essence sigmoid and tanh units, adders, and multipliers.

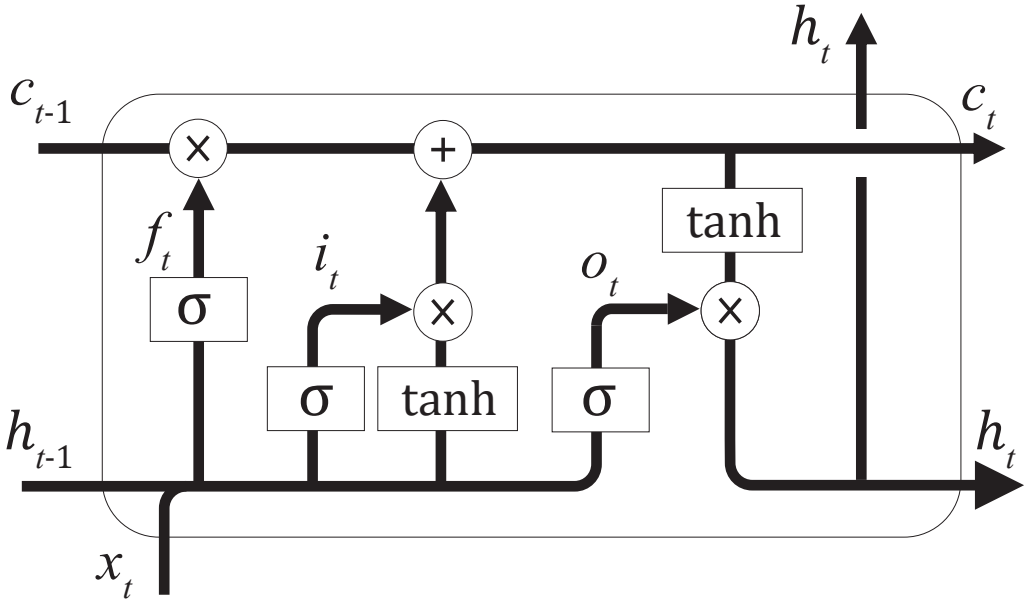


Figure 1. Structure of an LSTM unit.

Mathematically, the equations governing the operation of an LSTM unit are described next.
The forget gate output is given by Eq (2.1):

$$f_t = \sigma(W_{xf}\, x_t + W_{hf}\, h_{t-1} + b_f), \tag{2.1}$$

where $\sigma(x) = \frac{1}{1+e^{-x}} \in [0, 1]$ is the sigmoid activation function, x_t is the current input, h_t is the current output of the LSTM unit, W and b are weight and bias matrices, respectively.

The output of the input gate i_t is given by Eq (2.2):

$$i_t = \sigma(W_{xi}\, x_t + W_{hi}\, h_{t-1} + b_i), \tag{2.2}$$

where h_{t-1} is the previous output, W and b are respectively, the weight matrix and bias parameters that must be learned. Besides, the current output of the cell state c_t, is obtained by Eq (2.3):

$$c_t = f_t\, c_{t-1} + i_t\, \tanh(W_{xc}\, x_t + W_{hc}\, h_{t-1} + b_c), \tag{2.3}$$

where $tanh(x)$ is the hyperbolic tangent activation function. Finally, o_t is the output of the output gate defined as Eq (2.4):

$$o_t = \sigma(W_{xo}\, x_t + W_{ho}\, h_{t-1} + b_o). \tag{2.4}$$

LSTM networks have found countless applications since their invention, which involve speech recognition, text analysis, language processing, machine translation, handwriting generation, among others.

2.2.2. Gated recurrent unit

Gated Recurrent Unit (GRU) networks were introduced by Cho [46] as an alternative to LSTM networks with fewer parameters. A GRU unit utilizes only two gates to decide when to acquire new information and its hidden state is also its output. Basically, a GRU unit is composed of one memory cell and two multiplicative gates: Update, and output gates. Figure 2 illustrates the structure of a fundamental GRU unit.

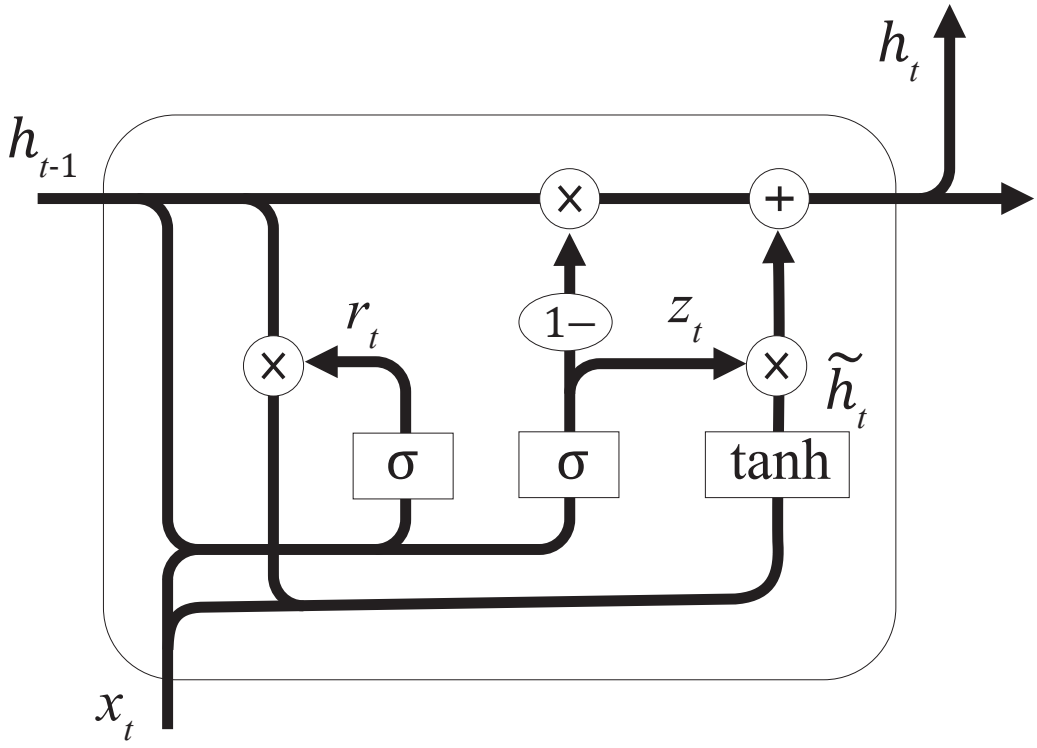


Figure 2. Structure of a fundamental GRU unit.

The computations performed by a GRU unit are described next. The reset gate's output r_t, which decides the amount of information to forget, is defined as Eq (2.5)

$$r_t = \sigma(W_{xr}\, x_t + W_{hr}\, h_{t-1} + b_r), \tag{2.5}$$

where σ is a sigmoid activation function, x_t is the current input, h_{t-1} is the hidden state of the previous time step, W and b are the weight matrix and bias parameters, respectively. The output of the update gate for time step t, z_t, is described by Eq (2.6)

$$z_t = \sigma(W_{xz}\, x_t + W_{hz}\, h_{t-1} + b_i). \tag{2.6}$$

Finally, the current hidden state (or output) h_t of the GRU unit is formally given by Eq (2.7)

$$h_t = (1 - z_t)\, h_{t-1} + z_t\, \tilde{h}_t, \tag{2.7}$$

where $\tilde{h}_t$ is simply defined as,

$$\tilde{h}_t = \tanh(W_{xh}\, x_t + r_t\, W_{hh}\, h_{t-1} + b_h), \tag{2.8}$$

which uses the reset gate output to feedback past relevant information.

2.3. Proposed approach

Most previous approaches to handwriting character recognition from sEMG signals were based on pre-processing and feature extraction followed by a linear classifier such as LDA [13]. Whereas some other approaches were based on pre-processing followed by the DTW method. Nevertheless, all these approaches are highly restrictive, in the sense that they require the electrodes to be carefully placed on the muscles to facilitate classification. Additionally, all those approaches were only designed for single-stroke characters.

In this paper, a more general and reliable approach is proposed. This method disposes of highlighted features as multi-user, free-style, multi-stroke handwriting character recognition. Each test-subject wears a Myo armband device on the forearm, and no restriction is made regarding a predefined position of the electrodes. Figure 3 shows the global components of the database construction of the characters recognition approach.

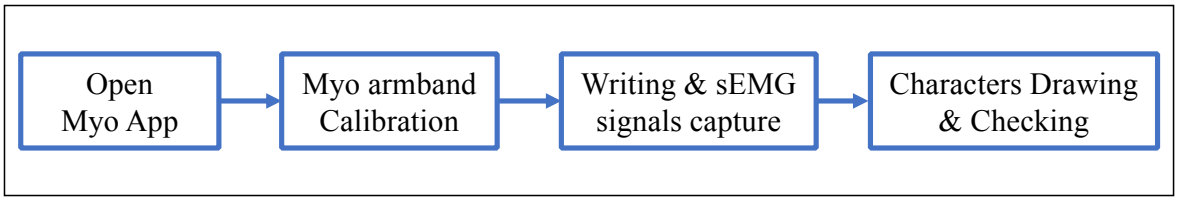


Figure 3. General framework for the database construction used in the recognition system.

The main contributions are focused on the database creation and the development of a discriminant and versatile characters recognition system. The four blocks are dedicated to the character's acquisition system and especially to the construction of the database using software programmed to control the

captured patterns (see details in Section 3). To cope with the variability of sEMG signals, powerful techniques for feature extraction, and sequence processing, such as CNNs and RNNs (LSTM, GRU) networks, were respectively used. From a multi-channel sEMG signal approach, four neural network architectures were designed and implemented, which were named DCNN-LSTM, DCNN-2LSTM, DCNN-GRU, and DCNN-2GRU architectures.

Figure 4 shows the internal layers of the proposed CNN-LSTM architecture.

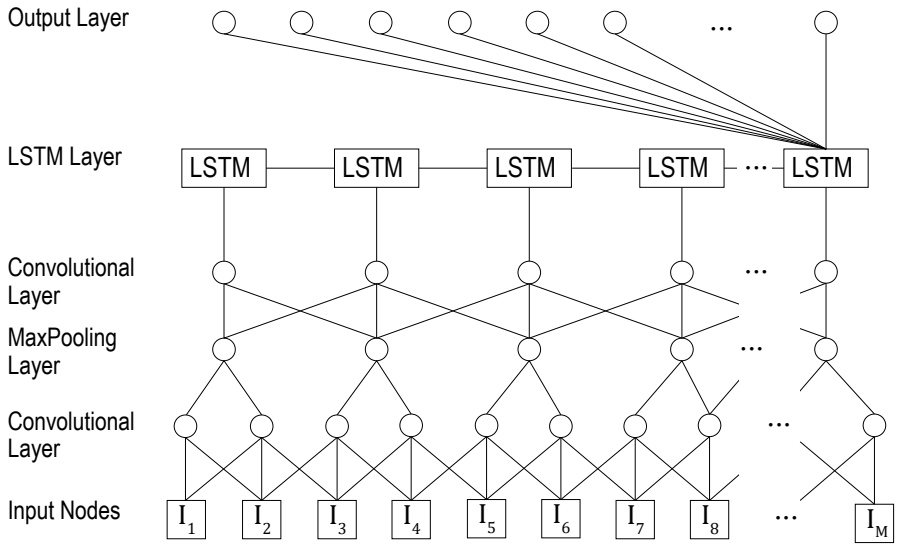


Figure 4. CNN-LSTM architecture.

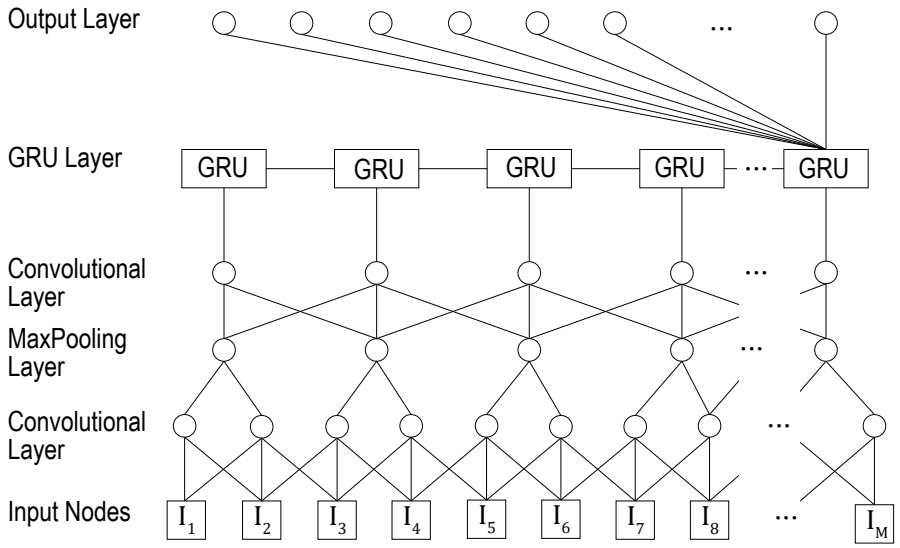


Figure 5. CNN-GRU architecture.

Such a DCNN-LSTM architecture comprises two convolutional layers, one pooling layer, one LSTM layer, and one fully connected layer. Each convolutional layer has a ReLU activation function

and 32 filters of size 3×1. A max-pooling layer follows the first convolutional layer. Thereunto, a dropout layer is added to avoid over-training and improve generalization in each convolutional layer. Besides, the LSTM layer is composed of 180 units. Conversely, the fully connected layer has 36 outputs and a SoftMax activation function.

The neural architectures were implemented in Python using Keras with Tensorflow backend with: The loss function based on categorical-cross-entropy, the optimizer known as Nadam, the fundamental metrics defined as loss and accuracy.

Apart from the above, the internal layers of the proposed CNN-GRU architecture are shown in Figure 5.

It is worth mentioning that the DCNN-GRU architecture is similar to the DCNN-LSTM architecture. The main difference relies on using a GRU layer instead of an LSTM layer. Whereas, the DCNN-2LSTM and DCNN-2GRU architectures use two stacked LSTM and GRU layers, respectively.

Figure 6 presents the sequence of processes involved in the identification of the character class.

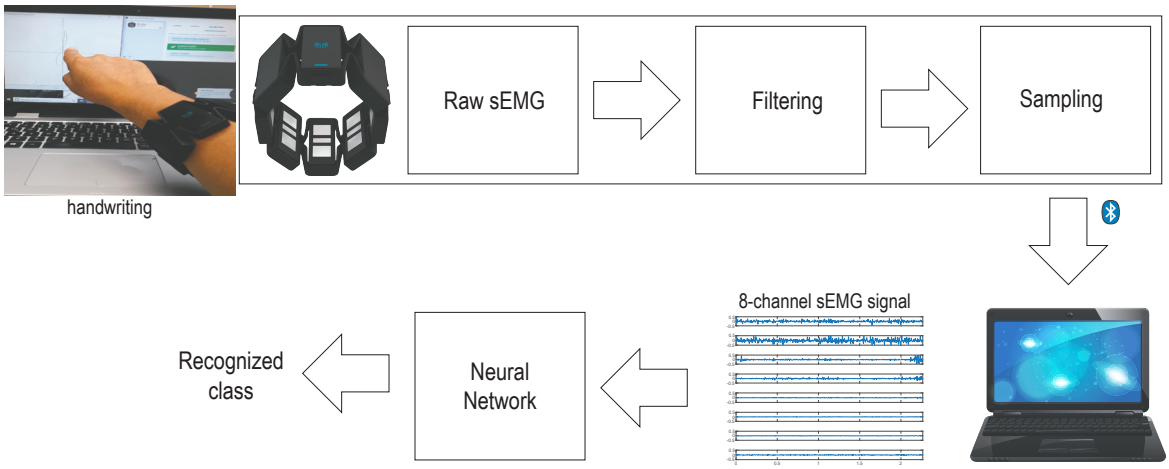


Figure 6. Overview of the recognition system based on convolutional and recurrent neural networks.

The most important steps are armband calibration, handwriting task, sEMG capture (Myo armband: Raw sEMG signals generation, filtering and sampling), magnitude and sequence length normalization to finally send the 8-channel sEMG data to the neural architecture for recognition. In concrete terms, the input to the Neural Network is an array of size $(900, 8)$ with eight-channel sEMG signals, one pattern obtained from the armband at each capture, the number of outputs are 36. Each output represents an estimation of a class probability.

3. Data collection

The raw sEMG data used in the numerical tests were collected from a Myo™ armband. The Myo™ armband is a commercial electronic device developed by Thalmic Labs®. Such a system is a low-cost, non-invasive, user-friendly wireless eight-channel device with dry electrodes working at a sampling frequency of 200 Hz.

Figure 7. A subject wears a Myo armband on the forearm.

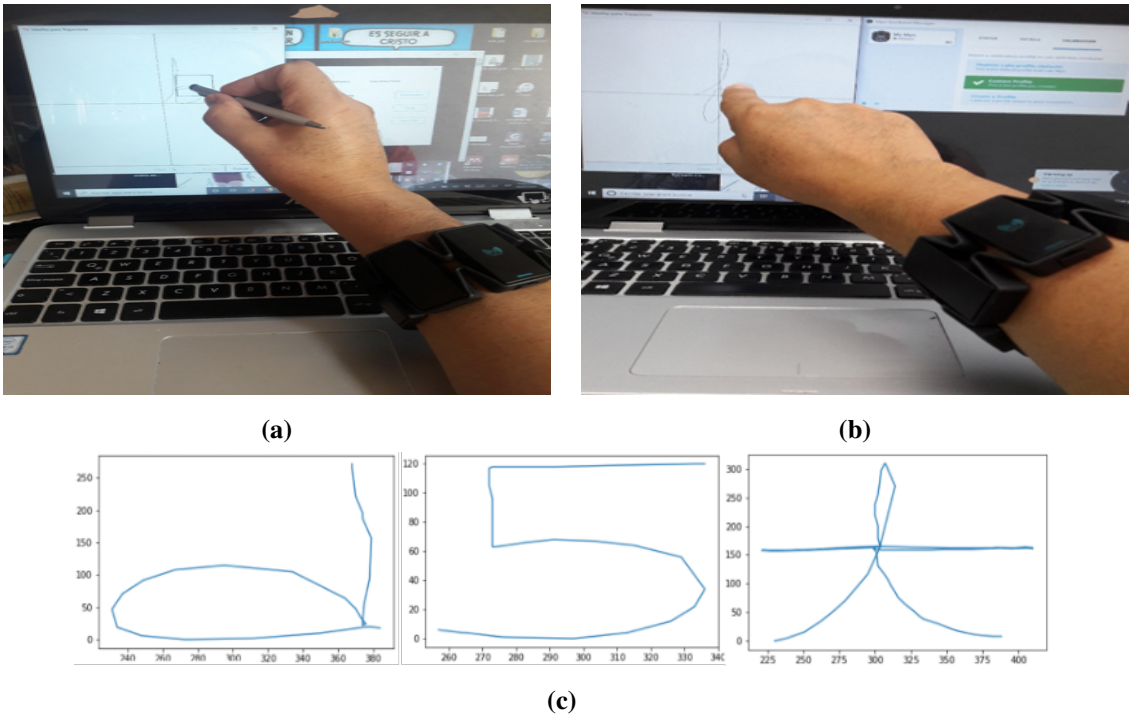


Figure 8. Example of a capture task. (a) A subject writing a letter by using a stylus pen. (b) A subject writing a letter by using the index finger. (c) Some examples of handwritten characters.

A capture application was developed in Python, under Windows 10. Synthetically, the capture procedure operates as follows, first, the application to capture sEMG signals is started from a command prompt. The application displays a GUI with an option of terminating the application at any time. Next, a subject wears a Myo armband on the left or right forearm (see Figure 7). Posteriorly, the armband is turned on and connected to the PC via a Bluetooth dongle that is plugged into a USB port. Once the device has been correctly detected, a personal calibration profile must be created and selected by using the Myo armband Manager software. Then, the user types a character (letter or digit) into a text field in the GUI. If the typed character (or class) is wrongly chosen, the software still provides the possibility

to cancel the capture. After that, the user clicks on Start button to begin the capture process. At this time the subject starts writing with the index finger (or a stylus pen) on the touchscreen. The capture process finishes when the subject clicks on Stop button. Now, the subject must click on Save button to write the captured data to a CSV file. The captured data comprises eight signals corresponding to the eight sensors of the Myo armband. Such characters are written in a freehand style with the index finger or a stylus pen. Additionally, the (x,y) coordinates of the handwriting movements are visualized on the screen. Thereunto, remember that before each handwriting task, the Myo armband is calibrated so that the device always started with the same personal configuration for different users.

Three healthy subjects seated in a comfortable position participated in the creation of the database.

Each task consisted of drawing with the index finger (or stylus pen), 26 lowercase characters ('a'-'z'), and 10 decimal digits ('0'-'9'). The subjects were free to use their particular handwriting style with one or more strokes.

Because a total of 3 subjects participated in creating the database, the number of classes of characters is 36 (26 letters and 10 digits), and each subject wrote each character class one thousand times, the total number of patterns is 108,000 (i.e., $3 \times 36 \times 1000$). Ergo, the database contains 108,000 patterns of multi-channel sEMG signals divided into 36 classes and 3,000 patterns per class. Additionally, the database also provides the trajectory of each handwritten character along with the stroke number. This can be useful for visualization or research purposes (i.e., handwriting trajectory reconstruction). Figure 9 shows the representative (average) pattern over all classes. The amplitudes are small in the right side because there are fewer patterns with lengths above 2.5 seconds.

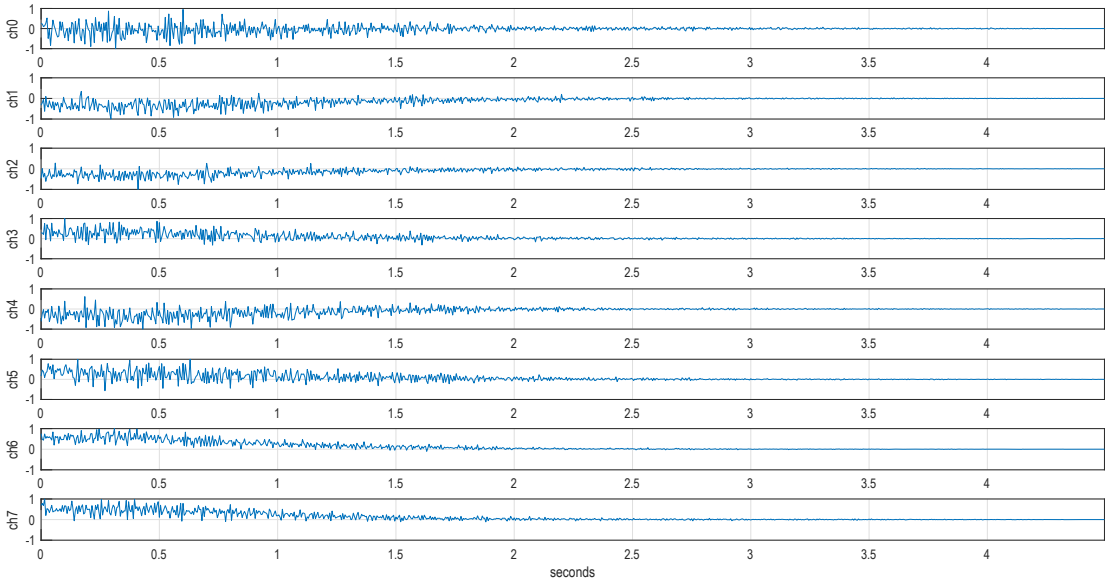


Figure 9. The representative (average) pattern of the database over all classes.

4. Numerical results

The database was split into training and testing sets to assess the performance of the proposed architectures. The testing set contains 10% of the samples, whereas the rest of the samples formed the

training set. Thus, 97,200 samples were used for training and 10,800 samples were used for testing.

All architectures were trained and tested on a workstation with an Intel® Core™ i7 processor, 16 GB RAM, an Nvidia® GeForce RTX™ 2080 GPU, and Windows 10 using Keras with a Tensorflow™ backend. For all cases, the Nesterov-accelerated ADAptive Moment (NADAM) algorithm [47] was used for training using the cross-entropy loss function and a batch size of 1024.

Firstly, each architecture was trained with similar tuning parameters for 8,000 epochs using a cyclical learning rate schedule [48]. Thus, the minimum and maximum learning rates were set to 0.000001 and 0.001, respectively. Furthermore, the step size was defined to five times the number of iterations in each epoch. Secondly, each training essay was repeated several times with different initial conditions to help avoid local minima. Moreover, the dropout rate was varied from 0.8 to 0.6 to help reduce the effects of overfitting. Likewise, the number of units in the LSTM and GRU layers varied from 80 to 200.

Table 1 shows the results obtained for the four architectures using only samples for one subject.

Table 1. Results obtained for each architecture.

Architecture	Accuracy (%)
CNN-LSTM	96.37
CNN-2LSTM	96.48
CNN-GRU	97.03
CNN-2GRU	96.29

As can be seen, the best accuracy (97.03%) is achieved by the CNN-GRU architecture, whereas the CNN-2GRU architecture achieves the lowest accuracy (96.29%).

Table 2 shows the results for the CNN-GRU architecture for each and all subjects.

Table 2. Results obtained for the CNN-GRU architecture.

Subjects	Accuracy (%)
Subject1	97.04
Subject2	95.05
Subject3	94.89
All	94.85

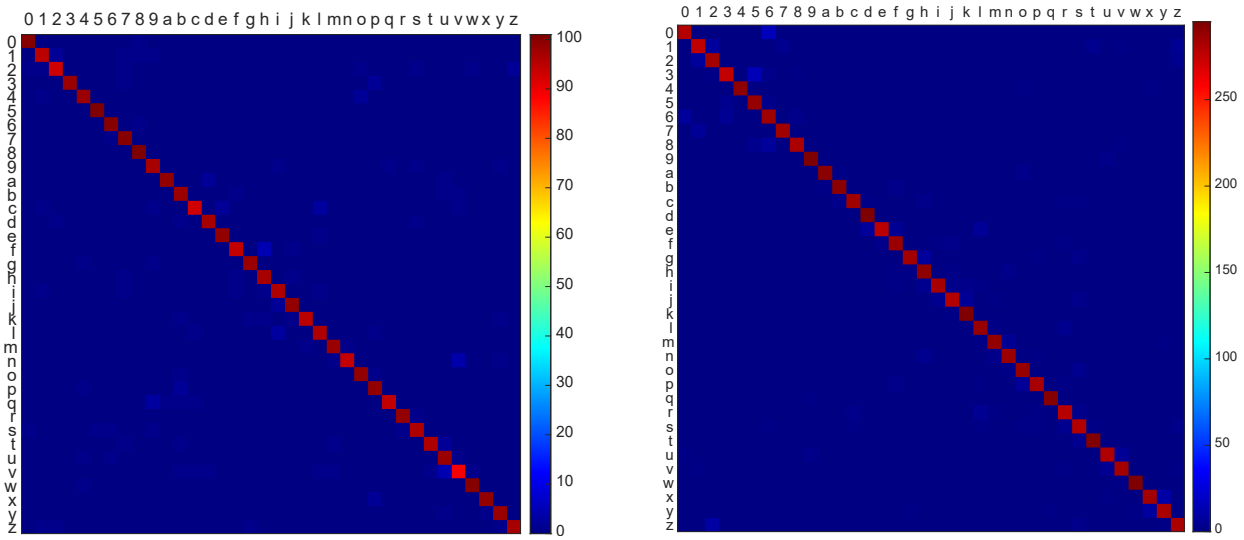
Note that the highest accuracy was obtained for subject1 and the lowest for subject3. The accuracy of all subjects was 94.85%, which is slower than the accuracy of a single subject. This effect is due to the higher variability of the sEMG signals of all subjects. Table 3 shows a quantitative comparison with representative state of the art studies.

Table 3. Comparison of the proposed method with state-of-the-art approaches.

Method	Samples	Subjects	Classes	Accuracy (%)
Template Matching [13]	350	6	10	97
DTW [12]	520	3	26	84.29
DTW [14]	780	4	26	78.24 to 92.42
Proposed	108k	3	30	94.85

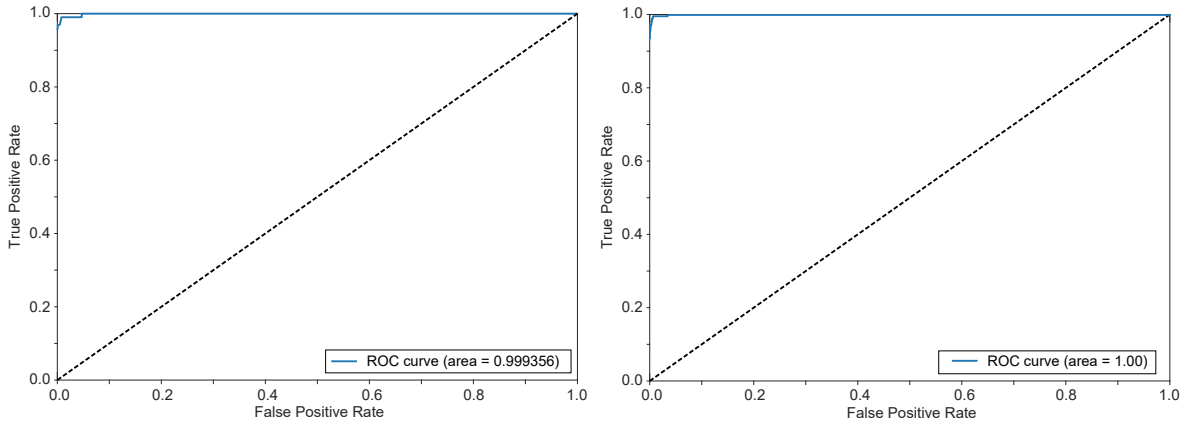
From the numerical results, it can be observed and concluded that the proposed approach is highly competitive, and in general, compares well with the already-existing methods in the state of the art. Figure 10 shows the confusion matrix obtained for the CNN-GRU architecture.

It is noteworthy that all the classes are recognized with high accuracy for samples from all test subjects. Figure 11 shows the ROC curve obtained for class a by using the CNN-GRU architecture.



(a) Subject 1. **(b)** All subjects.

Figure 10. Confusion matrix obtained for the CNN-GRU architecture.



(a) Subject 1. **(b)** All subjects.

Figure 11. ROC curve for class a and CNN-GRU architecture.

As can be seen, the ROC area is very close to one, which indicates an excellent separation capability between classes. A similar result was obtained for the other 35 classes.

5. Discussions

The numerical results showed that the four architectures proposed presented comparable accuracies. However, the fastest architecture (CNN-GRU) also presented the highest average accuracy. It is worth mentioning that these results yielded a high accuracy, although no pre-processing was performed. For these reasons, the CNN-GRU architecture was chosen for the task of multi-user, free-style, handwriting character recognition. Notwithstanding, there are always options for amending the actual systems. For instance, an opportunity of improvement is represented by the possibility of testing our approach using a more extensive multi-user database of sEMG signals.

6. Conclusions

A new methodology for recognizing multi-stroke handwriting characters from sEMG signals using an Myo^{TM} armband device has been presented. The proposed approach consists of using robust Deep Learning architectures for feature extraction, and sequence recognition algorithms such as CNNs and RNNs. One advantage of the discussed approach is that characters can be written either by using a pen or a finger. Another advantage is that characters can also be written in the air.

A CNN-GRU architecture achieved average accuracy of 94.85% for the task of multi-user handwriting character recognition. In comparison with previous works, the results obtained in this work are promising even though sEMG signals presented complex variations on their shapes due to the differences in the placement of the armband device. Therefore, this study is an essential contribution to the area of brain-computer interfaces. Brain-computer interfaces offer an alternative to restore or enhance the communications skills of people with and without disabilities.

Acknowledgements

This project was fully supported by the Electronics Department of the Universidad de Guanajuato under the Program POA 2020. Especially, authors would like to give special thanks to Perla Villegas-Torres for her support in the edition of this manuscript. Authors are also grateful to the Mexican Council of Science and Technology (CONACyT), for granting the scholarships (CVU/SGN: 495556/455203 and 495754/284009) to fund their doctoral studies.

Conflict of interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

1. J. E. Maldarelli, B. A. Kahrs, S. C. Hunt, J. J. Lockman, Development of early handwriting: Visual-motor control during letter copying, *Dev. Psychol.*, **51** (2015), 879–888.
2. J. Calvo-Zaragoza, J. Oncina, Recognition of pen-based music notation with finite-state machines, *Expert Syst. Appl.*, **72** (2017), 395–406.

3. N. Mendes, M. Sim£o, P. Neto, *Segmentation of electromyography signals for pattern recognition*, IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society, 2019, 732–737.
4. M. Słapek, S. Paszkiel, Detection of gestures without begin and end markers by fitting into Bèzier curves with least squares method, *Pattern Recognit. Lett.*, **100** (2017), 83–88.
5. K. A. Lamkin-Kennard, M. B. Popovic, Sensors: Natural and Synthetic Sensors, *Biomechatronics*, Elsevier, 2019, 81–107.
6. J. Wu, X. Li, W. Liu, Z. Jane Wang, sEMG Signal Processing Methods: A Review, *J. Phys.*, **1237** (2019), 032008.
7. E. Guigon, P. Baraduc, M. Desmurget, Computational Motor Control: Redundancy and Invariance, *J. Neurophysiol.*, **97** (2007), 331–347.
8. C. J. De Luca, Physiology and Mathematics of Myoelectric Signals, *IEEE Trans. Biomed. Eng.*, **BME-26** (1979), 313–325.
9. M. Sim£o, N. Mendes, O. Gibaru, P. Neto, A Review on Electromyography Decoding and Pattern Recognition for Human-Machine Interaction, *IEEE Access*, **7** (2019), 39564–39582.
10. Y. Gloumakov, J. Bimbo, A. M. Dollar, *Trajectory Control For a Myoelectric Prosthetic Wrist*, Myoelectric Controls Symposium, 2020.
11. A. Lansari, F. Bouslama, M. Khasawneh, A. Al-Rawi, *A novel electromyography (EMG) based classification approach for Arabic handwriting*, Proceedings of the International Joint Conference on Neural Networks, 2003.
12. G. Huang, D. Zhang, X. Zheng, X. Zhu, *An EMG-based handwriting recognition through dynamic time warping*, in 2010 Annual International Conference of the IEEE Engineering in Medicine and Biology, IEEE, 2010, 4902–4905.
13. M. Linderman, M. A. Lebedev, J. S. Erlichman, Recognition of Handwriting from Electromyography, *PLoS ONE*, **4** (2009), e6791.
14. C. Li, Z. Ma, L. Yao, D. Zhang, *Improvements on EMG-based handwriting recognition with DTW algorithm*, in 2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), IEEE, 2013.
15. I. Chihi, A. Afef, B. Mohamed, Analysis of Handwriting Velocity to Identify Handwriting Process from Electromyographic Signals, *Am. J. Appl. Sci.*, **9** (2012), 1742–1756.
16. M. A. Slim, A. Abdelkrim, M. Benrejeb, *An efficient handwriting velocity modelling for electromyographic signals reconstruction using Radial Basis Function neural networks*, 2015 7th International Conference on Modelling, Identification and Control (ICMIC), IEEE, 2015, 1–6.
17. E. Okorokova, M. Lebedev, M. Linderman, A. Ossadtchi, A dynamical model improves reconstruction of handwriting from multichannel electromyographic recordings, *Front. Neurosci.*, **9** (2015), 1–15.
18. W. Wei, Q. Dai, Y. Wong, Y. Hu, M. Kankanhalli, W. Geng, Surface-Electromyography-Based Gesture Recognition by Multi-View Deep Learning, *IEEE Trans. Biomed. Eng.*, **66** (2019), 2964–2973.

19. A. Dash, A. Sahu, R. Shringi, J. Gamboa, M. Z. Afzal, M. I. Malik, et al., *AirScript-Creating Documents in Air*, 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), IEEE, 2017.
20. P. Roy, S. Ghosh, U. Pal, *A CNN Based Framework for Unistroke Numeral Recognition in Air-Writing*, 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR), IEEE, 2018, 404–409.
21. N. Rusk, Deep learning, *Nat. Methods*, **13** (2016), 35.
22. H. Chen, Y. Zhang, G. Li, Y. Fang, H. Liu, Surface electromyography feature extraction via convolutional neural network, *Int. J. Mach. Learn. Cybern.*, **11** (2020), 185–196.
23. M. Simão, P. Neto, O. Gibaru, EMG-based online classification of gestures with recurrent neural networks, *Pattern Recognit. Lett.*, **128** (2019), 45–51.
24. Y. Lecun, Y. Bengio, G. Hinton, Deep learning, *Nature*, **521** (2015), 436–444.
25. D. H. Hubel, T. N. Wiesel, Receptive fields and functional architecture of monkey striate cortex, *J. Physiol.*, **195** (1968), 215–243.
26. D. Marr, *Analyzing natural images: A computational theory of texture vision.*, Cold Spring Harbor symposia on quantitative biology, Cold Spring Harbor Laboratory Press, 1976, 647–662.
27. K. Fukushima, S. Miyake, *Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position*, Competition and Cooperation in Neural Nets, Springer, Berlin, Heidelberg, 1980, 267–285.
28. C. Lee, P. W. Gallagher, Z. Tu, Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree, Artificial intelligence and statistics, 2016, 464-472.
29. J. Weng, N. Ahuja, T. S. Huang, *Cresceptron: A self-organizing neural network which grows adaptively*, International Joint Conference on Neural Networks (IJCNN), 1992, 576–581.
30. Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, et al., Backpropagation Applied to Handwritten Zip Code Recognition, *Neural Comput.*, **1** (1989), 541–551.
31. Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, *Proc. IEEE*, **86** (1998), 2278–2324.
32. W. Zhang, K. Itoh, J. Tanida, Y. Ichioka, Parallel distributed processing model with local space-invariant interconnections and its optical architecture, *Appl. Optics*, **29** (1990), 4790–4797.
33. J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, et al., Recent advances in convolutional neural networks, *Pattern Recognit.*, **77** (2018), 354 – 377.
34. A. Krizhevsky, I. Sutskever, G. E. Hinton, *Imagenet classification with deep convolutional neural networks*, Advances in Neural Information Processing Systems 25, 2012.
35. O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, et al., ImageNet Large Scale Visual Recognition Challenge, *Int. J. Comput. Vis.*, **115** (2015), 211–252.
36. V. Nair, G. E. Hinton, *Rectified linear units improve restricted boltzmann machines*, in Proceedings of the 27th International Conference on International Conference on Machine Learning, ICML10, Omnipress, Madison, WI, USA, 2010, 807814.

37. 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.*, **15** (2014), 1929–1958.
38. J. Wang, L. Perez, The effectiveness of data augmentation in image classification using deep learning, *Convolutional Neural Networks Vis. Recognit.*, **11** (2017).
39. K. He, X. Zhang, S. Ren, J. Sun, *Deep Residual Learning for Image Recognition*, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2016, 770–778.
40. S. Ioffe, C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, Proceedings of the 32nd International Conference on Machine Learning, ICML 2015.
41. M. I. Jordan, *Attractor dynamics and parallelism in a connectionist sequential machine*, Artificial neural networks: Concept learning. 1990, 112-127.
42. J. L. Elman, Finding Structure in Time, *Cognit. Sci.*, **14** (1990), 179–211.
43. S. Hochreiter, Untersuchungen zu dynamischen neuronalen Netzen, Master's thesis, Institut für Informatik, Technische Universität, Munchen, 1991.
44. Y. Bengio, P. Simard, P. Frasconi, Learning Long-Term Dependencies with Gradient Descent is Difficult, *IEEE Trans. Neural Networks*, **5** (1994), 157–166.
45. S. Hochreiter, J. Schmidhuber, Long Short-Term Memory, *Neural Comput.*, **9** (1997), 1735–1780.
46. K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, et al., Learning phrase representations using RNN encoder-decoder for statistical machine translation, *arXiv preprint arXiv:1406.1078*, 1724–1734.
47. T. Dozat, *Incorporating nesterov momentum into adam*, in International Conference on Learning Representations, ICLR, 2016.
48. L. N. Smith, *Cyclical Learning Rates for Training Neural Networks*, 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), IEEE, 2017, 464–472.

Supplementary Material

The dataset is available in the public repository Discover Mendeley Data. Available from: https://data.mendeley.com/datasets/ms3sbpbrgp/1.