Transfer Learning for sEMG Hand Gestures Recognition Using Convolutional Neural Network

Ulysse Côté-Allard*, Cheikh Latyr Fall*, Alexandre Campeau-Lecours†,
Clément Gosselin†, François Laviolette‡ and Benoit Gosselin*
*Department of Computer and Electrical Engineering. Laval University, Quebec, Canada. Email: ulysse.cote-allard.1@ulaval.ca
†Department of Computer Science and Software Engineering. Laval University, Quebec, Canada
‡Department of Mechanical Engineering. Laval University, Quebec, Canada

***Abstract*—In the realm of surface electromyography (sEMG) gesture recognition, deep learning algorithms are seldom employed. This is due in part to the large quantity of data required for them to train on. Consequently, it would be prohibitively time consuming for a single user to generate a sufficient amount of data for training such algorithms.**

In this paper, two datasets of 18 and 17 able-bodied participants respectively are recorded using a low-cost, low-sampling rate (200Hz), 8-channel, consumer-grade, dry electrode sEMG device named Myo armband (Thalmic Labs). A convolutional neural network (CNN) is augmented using transfer learning techniques to leverage inter-users data from the first dataset and alleviate the data generation burden on a single individual. The results show that the proposed classifier is robust and precise enough to guide a 6DoF robotic arm (in conjunction with orientation data) with the same speed and precision as with a joystick. Furthermore, the proposed CNN achieves an average accuracy of 97.81% on 7 hand/wrist gestures on the second dataset.

I. Introduction

In recent years, robotics have been leveraged to increase the autonomy of people living with disabilities. The main objective is thus to improve the quality of life by enabling users to perform a wider range of day-to-day tasks in less time. To exploit the field of robotics to its fullest extent, artificial intelligence can be employed to make assistive robots increasingly useful and versatile. However, to be truly effective, these new assistive solutions must remain easy to integrate.

A natural and intuitive approach for creating such interfaces is to look at the muscle activity of the user. This activity can be recorded by surface electromyography (sEMG) sensors, a non-invasive technique widely adopted both in clinical settings and research. The sEMG signals are non-stationary and represents the sum of subcutaneous motor action potentials generated through muscular contraction [1]. The combination of sEMG signals and supervised learning algorithms has been used for more intuitive control of prosthesis and assistives robots [1], [2], [3]. However, collecting data from a single user to train a supervised system can be prohibitively time-consuming. In contrast, pre-training a classifier on the accumulated data of preceding subjects could potentially reduce the required amount of data for new users, whilst enhancing the system's accuracy.

This article proposes leveraging inter-user data using transfer learning on a Convolutional Neural Network (CNN) in order to introduce a more accurate classifier. This while simultaneously reducing the size of the required training dataset, typically seen with deep learning algorithms. CNNs have scarcely been used for the classification of sEMG hand gestures as they typically require a large amount of computational power. However, recently complex CNN topologies have been implemented with low energy consumption and manufacturing costs. Additionally, only the inference part must be carried out on the low power unit (e.g. embedded system) as the training can be completed in an offline setting. Thus, the use of CNNs in sEMG gesture recognition contexts is becoming possible. In fact, it has recently been suggested for classification in the time domain [4], [5] and time-frequency domain [2]. Additionally, domain adaptation techniques using CNNs have also been proposed to enhance the performance of sEMG classifiers in inter-session settings [6]. To the best of our knowledge, we propose the first deep learning framework in the context of sEMG gesture for inter-subject recognition.

This paper is organized as follows. Sec. II presents the sEMG data acquired as well as how it was acquired and processed. The details of the CNN implementation and transfer learning algorithms are presented in Sec. III and Sec. IV respectively. Sec. V-A shows that the proposed classifier is accurate and robust enough to control a robotic arm in order to perform a complex, three-dimensional task in real-time. Finally, Sec. V-B compares and analyzes the test accuracy of the system, with and without using transfer learning algorithms, to showcase the advantage of using them.

II. sEMG dataset recording and processing

In the case presented in this article, two datasets are utilized to build and evaluate the proposed CNN. The first one was recorded a year prior to the second and is referred to as the *validation dataset*. This dataset is comprised of 18 able-bodied subjects performing one round of seven gestures for 20 *s* each. This dataset was the only one employed when building the classifier's architecture and selecting hyper-parameters, as to avoid indirect overfitting at test time. The second dataset, referred to as the *test dataset*, is comprised of 17 healthy, able-bodied subjects aged between 22 and 28, performing

three rounds of seven gestures for 20 *s* each. Note that for the experiments presented in Sec. V, no distinction is made between left and right handed people as the proposed system is able to adapt automatically to those variations.

The data acquisition was approved by the Laval University Ethics committee (approbation number: 2017-026/21-02-2016) and written informed consent was obtained from all subjects.

A. Recording Hardware

Each subject's forearm electromyographic activity was recorded using the Myo Armband by Thalmic Labs[1]. The Myo is a low-cost, 8-channel, dry-electrode, low-sampling rate (200Hz), consumer-grade sEMG device. Additionally, the Myo includes a 9-axis inertial measurement unit (IMU) which will be utilized in Sec. V-A. The Myo is a non-invasive device, as the use of dry-electrodes enables users to simply slip the bracelet on without any preparation. Comparatively, gel-based electrodes require the shaving and washing of the skin to obtain optimal contact between the subject's skin and electrodes. These requirements would thus impose a significant burden on the user's daily life. However, the ease of use of the armband comes with severe limitations as dry electrodes are less accurate and robust to motion artifact, compared to gel-based ones [7]. Additionally, the low-sampling frequency (f_s) of 200 *Hz* is not sufficient to access the preferred f_s of 1 *kHz* [8]. The resulting signal-to-noise ratio recorded by the Myo is thus relatively poor compared to an optimal system. Therefore more robust machine learning algorithms are required to process it accurately.

In all cases, the armband was tightened to its maximum and slid up the user's forearm until the circumference of the armband matched that of the forearm. This was done in the same way systematically with each participant, to avoid introducing bias from the researchers. As a consequence of this, there was a great variety of armband placements (as can be seen in Fig 1) to reflect the way users without prior knowledge of optimal electrode placement might vary in their placement of the armband.

Fig. 1. Examples of the wide range of the armband placements on the subjects' forearm

B. Hand/Wrist Gestures

Fig. 2 shows the seven gestures employed in this work: open hand, closed hand, wrist extension, wrist flexion, ulnar deviation and radial deviation. The seventh gesture dubbed *neutral* refers to the natural posture of the hand of the subject when no significant muscle activity is detected.

[1]https://www.myo.com/

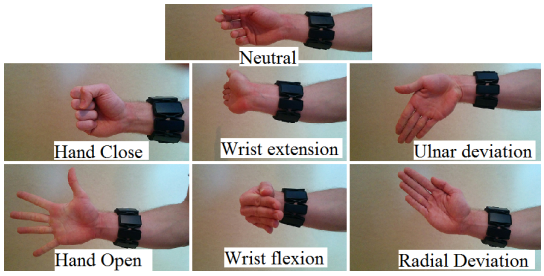

Fig. 2. The 7 hand/wrist gestures considered in this work.

C. Time-Window Length

For real-time control in a closed loop, input latency is an important factor to consider. A maximum time latency of 300 *ms* was first recommended in [9]. Even though more recent studies suggest that optimally the latency should be kept between 100-125 *ms* [10], the performance of the classifier should take priority over speed [10], [11]. As is the case in [2], due to the low frequency of the Myo the choice of 300 *ms*, was selected to achieve a reasonable number of samples between each prediction.

D. Recording labeled data

The collection of labeled data for the classifier (both for the validation and test dataset) was conducted by requiring the user to hold each gesture for 5 *s*. From there, the data is separated by applying sliding windows of 260 *ms* (to ensure sufficient time for data processing and classification) with an overlap of 235 *ms*. As suggested in [12], [13], to account for the natural variability of each gesture within the same subject (i.e. the subject will not apply the exact same strength and make the exact same movement), the process is repeated three more times. Each of these time is referred to as a *cycle*.

E. Data Processing

The use of CNNs for gesture classification instead of a more traditional approach, shifts the focus from feature engineering to feature learning. Because the feature extraction is inherently part of a CNN, the number of calculations required before feeding the input to the classifier is significantly reduced. Additionally, considering the need of the proposed algorithm to eventually run on an embedded system, considerable efforts were deployed in order to reduce the pre-processing burden as much as possible. As a result, the pre-processing was reduced into simply calculating the spectrograms of the raw sEMG data. Spectrograms are particularly well suited to capture non-stationary signals as is the case with sEMG data. For each of the eight channels of the armband, 52 samples (260 *ms*) per spectrograms are leveraged using Hann windows of 28 points per FFT and an overlap of 20. This yields a matrix of 8x4x15 (Channels x Time bins x Frequency bins). The axes are then rearranged to be 4x8x15 in order to be suitable to be fed to the CNN. Finally, to reduce baseline drift and motion artifact, the lowest frequency bin is removed. The final generated matrix is thus of shape 4x8x14

III. Classifier Overview

Deep learning algorithms, including CNNs, are prone to overfitting especially on small datasets [14], [15]. In the context of this article, acquiring a large dataset for each individual subject would be time consuming and impractical when considering real-life applications, as a user would often not endure hours of data recording for each training. To address this over-fitting issue, Batch Normalization [16], Monte Carlo Dropout [15] and early stopping are utilized and explained in greater details in the followings subsections. Considering the stochastic nature of the algorithms presented in this paper, unless stated otherwise, all experiments are reported at an average of 20 runs.

A. Batch Normalization

Batch Normalization (BN) [16] is a recent technique that proposes normalizing each batch of data through each layer during training. After the training phase, the data is fed one last time through the network to compute the data statistics in a layer wise fashion and then fixing those statistic for online classification. BN was shown to yield faster training times whilst achieving better system accuracy and regularization [16]. When removing BN, the proposed CNN failed to converge to acceptable solutions. As recommended in [16], BN was applied before the non-linearity.

B. Monte Carlo Dropout

Dropout is a regularization method aimed at reducing the co-adaptation of units in an effort to address overfitting [14]. At training time, hidden units are randomly deactivated with probability p (hyper-parameters) and reactivated at test time. Note that often in existing literature, dropout is only used after fully connected layers and not with convolutional layers [15]. On the other hand, Monte Carlo Dropout (MC Dropout) proposes to use dropout with convolutional layers and to keep dropping units at test time (sending the same example M times and predicting the resulting most probable gesture) [15]. MC Dropout was specifically developed to offer more robustness than traditional approaches to the overfitting of CNNs when trained on small datasets [15]. On the validation dataset, MC Dropout obtains on average 97.30% whereas traditional dropout obtains 97.11%.

C. CNN's architecture

Videos are a representation of how spatial information (images) change through time. This information has been successfully extracted for classification purposes using CNNs for several years now [17], [18]. One popular architecture for extracting this information is called the *slow fusion model* where temporal data starts on disconnected parallel layers and are slowly fused together throughout the network so that higher layers have access to progressively more temporal information. This model was also applied successfully to multi-view image fusion [19]. An obvious parallel between video (Time x Spatial x Spatial) and the input utilized in this work (Time x Spatial x Frequency) is that they can both represent non-stationary information. This parallel led to considering the idea of using the same type of architecture applied in video recognition for spectrogram classification. The proposed architecture, using the slow-fusion model, was by far the most successful of the ones tried on the validation set.

Considering the nature of the data, the proposed architecture does not use any pooling as is typically the case. Indeed, pooling layers produces features that are resistant to information shifts in the image [20]. This behavior would be detrimental in the present case considering that the position of the channels on the arm is highly informative of the hand/wrist gesture presently performed. Similarly, pooling layers in the dimension of the frequency component is also unused because of the relatively poor frequency resolution of the spectrograms resulting from the armband's low frequency. In the case of high-frequency sEMG apparatus, 1D pooling layers on the frequency axis might actually lead to better results and faster prediction times, but this would need further investigation.

The implementation of the proposed architecture, seen in Fig. 3, was realized using Theano [21] in conjunction with Lasagne [22] which allowed the training to run on a GPU. As usual in deep learning, the architecture was built using trial and error and inspiration from previous architectures (mainly [2], [6], [18]). The non-linear activation function is the parametric rectified linear unit (PReLU) [23] and ADAM [24] is utilized for the optimization of the CNN. The deactivation rate for MC Dropout is set at 0.65 and was found using cross-validation on the validation set. Finally, to further avoid overfitting, early stopping is utilized by saving 10% of the training data which is randomly chosen at the beginning of the training in order to periodically test the network.

Dedicated ASIC platforms can be optimized for CNN inference using batch processing [25], binary-weight CNNs [26], half precision computation [27], fixed point operation [28], network pruning [29] and memory bandwidth optimization [30]. Feasibility of a dedicated hardware material for the proposed architecture in Sec. IV-A, requires it to perform a 8x4x14 spectrogram computation every 40 ms and 265.3 MOp/s to operate the network at an equivalent throughput. High-performance digital signal processing hardware architectures including suitable discrete Fourier transform tools for spectrogram computation have already been explored [31]. Recently researchers proposed deep learning platforms able to reach peak performance of several thousands MOp/s/W (e.g. Eyeriss [32], YodaNN [33], Origami [34], Bang et al. [35]). Adopting hardware-implementation approaches, based on the state of the art techniques will lead to a power-consumption lower than 50mW for the proposed architecture, suitable for wearable applications. Note that given the use of PReLU as the non-linearity, techniques for CNNs optimization on embedded systems relying on sparse weight matrix cannot be utilized conventionally. However, by using MC Dropout at prediction time, a majority of connections are forced to be zero and hence those optimization techniques remain relevant and speed up the inference time.

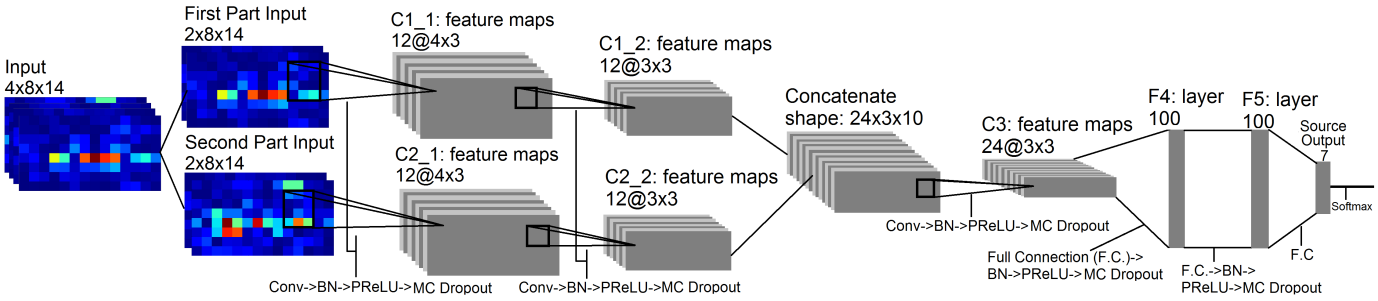

Fig. 3. The proposed CNN architecture using 39 639 parameters. To allow the slow fusion process, the input is first separated equally into two parts in respect to the time axis. The two branches are then concatenated back together before the last convolution.

IV. Transfer Learning

The underlying hypothesis of this paper is that information rich data can be extracted from sEMG signals from subjects performing a certain set of gestures. This new source of information can then be leveraged to build a more precise and reliable classifier for another subject performing the same set of gestures. The main motivation behind using transfer learning to help train the CNN is thus the idea that using this form of pre-training allows the capture of more general and robust features. The CNN is then able to leverage those features to build a more effective representation of a new subject's sEMG activity.

The method utilized for recording the data was purposely as unconstrained as possible in order to closely approximate day-to-day conditions. As such, no guarantees can be made regarding the relative orientation of the armband from one subject to the other. For this reason, the first step that allows transfer learning, is to align all the channels between each participant. The alignment for each subject was made by identifying the most active channel for each gesture on the first subject. On subsequent subjects the channels were then circularly shifted until their activation for each gesture matched the first subjects as closely as possible.

A. Progressive Neural Network

Fine-tuning is the most prevalent transfer learning technique in deep learning [36], [37], [38]. It consists of training a model on a *source domain* (often with an abundance of labeled data) and using the trained weights as a starting point when presented with a new task. However, fine-tuning can suffer from *catastrophic forgetting* [39] where relevant important features learned during pre-training might be lost on the *target domain* (i.e. new task). Moreover, by design, fine-tuning is ill-suited when significant differences exist between the source and the target domain as it can bias the network into poorly adapted features for the task at hand. Progressive Neural Networks [39] (PNN) propose to address those issues by pre-training a model on the source domain and freezing its weights. When a new task appears, a new network is created with random initialization and connected in a layer-by-layer wise fashion with the original network (See Fig 4). These models will be referred to hereafter as the *source network* and the *target network*. In this work, the new network created is identical to the source network (see Fig. 3) with the addition of merging layers which simply merge the output of each source network's layer to its corresponding new network's layer. An overview of the final proposed architecture is presented in Fig. 4. Note that during training of the source network, dropout is used, given the abundance of data. When training the target network, dropout is exchanged for MC Dropout.

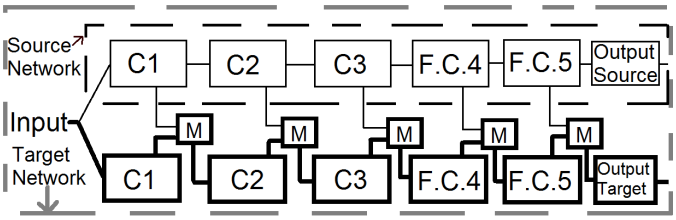

Fig. 4. The PNN architecture using 116 247 parameters. The details of the source network are shown in Fig 3. C1,2,3 and F.C.4,5 correspond to the three stages of convolutions and two stages of fully connected layers respectively. Note that for clarity's sake, the slow fusion aspect has been omitted from the representation but are present on both networks. The M boxes represent the merging of the CNNs' corresponding layers.

B. Adaptive Batch Normalization

The last form of transfer learning, named AdaBatch, utilized in this work comes from the use of BN when training both on the source and the target. [40] hypothesizes that the label-related information (i.e. difference between gestures) rests in the network model weights whereas the domain-related information is stored in their BN statistic (i.e. difference between subjects). By fixing the weights of the parameters of the network, except those related to BN, the model would be able to naturally adapt to a new user without modifying the weights of the original model. AdaBatch is a natural supplement when working with PNN considering that the source network has fixed weights when training on the target.

In this work, the model utilized a multi-stream Adabatch scheme as described in [6], but in an inter-subject fashion instead of an inter-session setting. This method learns the parameters of the BN layer of each subject individually. Additionally, these parameters are also the only ones not frozen in the source network, when training the target network. Note that a single-stream scheme (i.e. all subjects share statistics

and BN parameters are frozen on the source network) was also tried and although when training the source network on a small subset of subjects the single stream tended to yield better results, its performance rapidly declined when adding more subjects in the source dataset.

V. Experimental Results

In order to evaluate the performance of the classifier, two experiments were conducted. The first evaluates the model in real-time to ensure that the accuracy reached by the proposed CNN is sufficient for the control of an assistive robot. The second aims to give a quantitative measurement of the accuracy and robustness of the proposed system over approximately one hour. Another purpose is to showcase the benefits of using transfer learning both in terms of the amount of data required and the accuracy reached.

A. Robotic arm's guidance

To demonstrate that the proposed classifier is accurate enough to perform a complex, three-dimensional task in real-time, the participants were asked to control the robot JACO produced by Kinova. This assistive robotic arm has six degrees of freedom (6DoF) and is normally controlled using a joystick [41]. The task was built to maximise the amount of hand/wrists gesture required to complete it and consisted of grabbing a cube on a table and then dropping it at a specific location. The task was completed using both the standard joystick and the Myo described in Sec. II-A. To lessen bias as much as possible, half the subjects started using the joystick and the other half the armband.

1) Robotic arm's guidance scheme for the Myo Armband: The guidance scheme of the robotic arm utilizes both sEMG data and the IMU. Translation of the robotic arm is mapped with the IMU component. The pitch, roll and yaw of the subject's forearm, controls the Y, X and Z-axis respectively. The wrist extension, flexion, ulnar deviation and radial deviation gestures rotate the robotic hand in the same direction as the gesture performed. This behavior is achieved using the rotation system proposed in [42] which was shown to be easier to use than classic robot end-effector rotation, especially for neophytes. Finally, the closed and opened hand gesture were used to close and open the robotic hand respectively. A video accompanying this article shows the guidance system and the task performed by the participants.

2) Results of the guidance task: The average completion time for the task is 84 *s* with the Myo and 89 *s* with the joystick. As can be seen from Fig 5, both sub-tasks take roughly the same time to complete with either modality and the Wilcoxon signed-rank test find no statistically significant difference between the two (p=0.379). The proposed classification algorithm is thus accurate and robust enough to handle a complex, three-dimensional task in real-time with the same precision and speed as with the normal control method.

B. Transfer Learning algorithm accuracy

In order to assert the effects of using transfer learning, the target network is compared to a CNN using the architecture

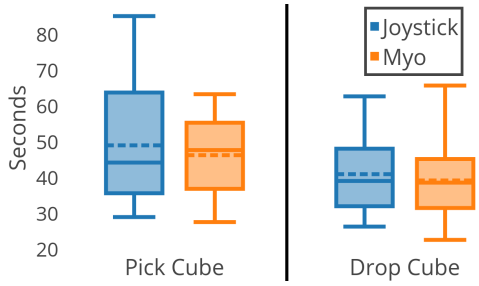

Fig. 5. Box plot of the time taken to complete the task using both modality of control (Joystick and Myo Armband).

presented in Fig. 3. For simplicity's sake, this CNN will be referred to as *Baseline-CNN*. Training of both networks was done using 1,2,3 and 4 cycles of the first round of the test dataset. The accuracy reported for both networks in Fig 6 represent the percentage of correct predictions made in the second and third round of the test dataset. As expected, training on more cycles yielded higher accuracy for both classifiers. However, even with only one cycle of training, the target network achieves an average accuracy of 93.36% and 97.81% when training on all four cycles. In all cases, the target network achieves significantly higher accuracy than the *Baseline-CNN* using the Wilcoxon signed-rank test ($p < 0.002$).

Note that using the same statistical test, no differences were found between the second and third round ($p = 0.501$) in terms of accuracy for the target network even though they were recorded up to 40 minutes apart.

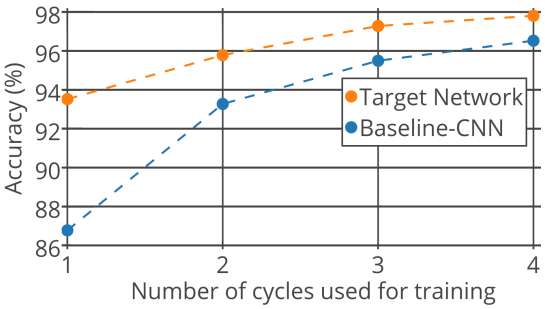

Fig. 6. The source and target network accuracy on the test set considering the amount of data used to trained them. Note that in this case, the source network is trained only on data from the test dataset. A cycle refers to 5 *s* of recorded data for each gesture as presented in Sec. II-D.

VI. Conclusion

This paper presents a novel CNN architecture, enhanced by transfer learning techniques. The proposed classifier was shown to be robust enough to control a 6DoF robotic arm with the same speed and precision as with a joystick. Additionally, the target network yielded an average accuracy of 97.81% on the test dataset when trained on 4 cycles.

Future works will focus on the long term use of the classifier using unsupervised methods to alleviate the need of periodic recalibration.

REFERENCES

[1] M. A. Oskoei and H. Hu, "Myoelectric control systems a survey," *Biomedical Signal Processing and Control*, vol. 2, no. 4, pp. 275–294, 2007.

[2] U. C. Allard, F. Nougarou, C. L. Fall, P. Giguère, C. Gosselin, F. Laviolette, and B. Gosselin, "A convolutional neural network for robotic arm guidance using semg based frequency-features," in *Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 2464–2470.

[3] D. Farina, N. Jiang, H. Rehbaum, A. Holobar, B. Graimann, H. Dietl, and O. C. Aszmann, "The extraction of neural information from the surface emg for the control of upper-limb prostheses: emerging avenues and challenges," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, no. 4, pp. 797–809, 2014.

[4] 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," *Frontiers in Neurorobotics*, vol. 10, 2016.

[5] W. Geng, Y. Du, W. Jin, W. Wei, Y. Hu, and J. Li, "Gesture recognition by instantaneous surface emg images," *Scientific Reports*, vol. 6, 2016.

[6] Y. Du, W. Jin, W. Wei, Y. Hu, and W. Geng, "Surface emg-based inter-session gesture recognition enhanced by deep domain adaptation," *Sensors*, vol. 17, no. 3, p. 458, 2017.

[7] P. Van Dijk Johannes *et al.*, "High-density surface emg: techniques and applications at a motor unit level," *Biocybernetics and Biomedical Engineering*, vol. 32, no. 3, pp. 3–27, 2012.

[8] R. Merletti and P. Di Torino, "Standards for reporting emg data," *Journal of Electomyography and Kinesiology*, vol. 9, no. 1, pp. 3–4, 1999.

[9] B. Hudgins, P. Parker, and R. N. Scott, "A new strategy for multifunction myoelectric control," *IEEE Transactions on Biomedical Engineering*, vol. 40, no. 1, pp. 82–94, 1993.

[10] T. R. Farrell and R. F. Weir, "The optimal controller delay for myoelectric prostheses," *IEEE Transactions on neural systems and rehabilitation engineering*, vol. 15, no. 1, pp. 111–118, 2007.

[11] B. Peerdeman, D. Boere, H. Witteveen, R. H. Veld, H. Hermens, S. Stramigioli *et al.*, "Myoelectric forearm prostheses: State of the art from a user-centered perspective," *Journal of Rehabilitation Research & Development*, vol. 48, no. 6, p. 719, 2011.

[12] D. Yang, W. Yang, Q. Huang, and H. Liu, "Classification of multiple finger motions during dynamic upper limb movements," *IEEE journal of biomedical and health informatics*, 2015.

[13] A. H. Al-Timemy, R. N. Khushaba, G. Bugmann, and J. Escudero, "Improving the performance against force variation of emg controlled multifunctional upper-limb prostheses for transradial amputees," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 24, no. 6, pp. 650–661, 2016.

[14] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[15] Y. Gal and Z. Ghahramani, "Bayesian convolutional neural networks with bernoulli approximate variational inference," *arXiv preprint arXiv:1506.02158*, 2015.

[16] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[17] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt, "Sequential deep learning for human action recognition," in *International Workshop on Human Behavior Understanding*. Springer, 2011, pp. 29–39.

[18] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[19] H. Guo, G. Wang, and X. Chen, "Two-stream convolutional neural network for accurate rgb-d fingertip detection using depth and edge information," in *Image Processing (ICIP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 2608–2612.

[20] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[21] R. Al-Rfou, G. Alain, A. Almahairi, C. Angermueller, D. Bahdanau *et al.*, "Theano: A python framework for fast computation of mathematical expressions," *arXiv preprint arXiv:1605.02688*, 2016.

[22] S. Dieleman, J. Schlter, C. Raffel, E. Olson, S. K. Snderby *et al.*, "Lasagne: First release." Aug. 2015. [Online]. Available: http://dx.doi.org/10.5281/zenodo.27878

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[24] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[25] C. Wang, Y. Wang, Y. Han, L. Song, Z. Quan, J. Li, and X. Li, "Cnn-based object detection solutions for embedded heterogeneous multicore socs," in *Design Automation Conference (ASP-DAC), 2017 22nd Asia and South Pacific*. IEEE, 2017, pp. 105–110.

[26] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "Xnor-net: Imagenet classification using binary convolutional neural networks," in *European Conference on Computer Vision*. Springer, 2016, pp. 525–542.

[27] A. Lokhmotov and G. Fursin, "Optimizing convolutional neural networks on embedded platforms with opencl," in *Proceedings of the 4th International Workshop on OpenCL*. ACM, 2016, p. 10.

[28] S. Anwar, K. Hwang, and W. Sung, "Fixed point optimization of deep convolutional neural networks for object recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 1131–1135.

[29] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally, "Eie: efficient inference engine on compressed deep neural network," in *Proceedings of the 43rd International Symposium on Computer Architecture*. IEEE Press, 2016, pp. 243–254.

[30] C. Li, Y. Yang, M. Feng, S. Chakradhar, and H. Zhou, "Optimizing memory efficiency for deep convolutional neural networks on gpus," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE Press, 2016, p. 54.

[31] G. Feng, L. Yao, and S. Chen, "Autonft: Architecture synthesis for hardware dft of length-of-coprime-number products," *Integration, the VLSI Journal*, 2017.

[32] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 1, pp. 127–138, 2017.

[33] R. Andri, L. Cavigelli, D. Rossi, and L. Benini, "Yodann: An architecture for ultra-low power binary-weight cnn acceleration," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2017.

[34] L. Cavigelli and L. Benini, "A 803 gop/s/w convolutional network accelerator," *IEEE Transactions on Circuits and Systems for Video Technology*, 2016.

[35] S. Bang, J. Wang, Z. Li, C. Gao, Y. Kim, Q. Dong, Y.-P. Chen, L. Fick, X. Sun, R. Dreslinski *et al.*, "14.7 a 288μw programmable deep-learning processor with 270kb on-chip weight storage using non-uniform memory hierarchy for mobile intelligence," in *Solid-State Circuits Conference (ISSCC), 2017 IEEE International*. IEEE, 2017, pp. 250–251.

[36] G. Mesnil, Y. Dauphin, X. Glorot, S. Rifai, Y. Bengio *et al.*, "Unsupervised and transfer learning challenge: a deep learning approach." *ICML Unsupervised and Transfer Learning*, vol. 27, pp. 97–110, 2012.

[37] Y. Bengio *et al.*, "Deep learning of representations for unsupervised and transfer learning." *ICML Unsupervised and Transfer Learning*, vol. 27, pp. 17–36, 2012.

[38] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in neural information processing systems*, 2014, pp. 3320–3328.

[39] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," *arXiv preprint arXiv:1606.04671*, 2016.

[40] Y. Li, N. Wang, J. Shi, J. Liu, and X. Hou, "Revisiting batch normalization for practical domain adaptation," *arXiv preprint arXiv:1603.04779*, 2016.

[41] A. Campeau-Lecours, V. Maheu, S. Lepage, S. Latour, L. Paquet, and N. Hardie, "Jaco assistive robotic device: Empowering people with disabilities through innovative algorithms." *Rehabilitation Engineering and Assistive Technology Society of North America (RESNA)*, 2016.

[42] D.-S. Vu, U. C. Allard, C. Gosselin, F. Routhier, B. Gosselin, and A. Campeau-Lecours, "Intuitive adaptive orientation control of assistive robots for people living with upper limb disabilities," ***SUBMITTED*** *International Conference on Rehabilitation Robotics*, pp. 1–6, 2017.