2023 IEEE Symposium Series on Computational Intelligence (SSCI)
Mexico City, Mexico. December 5-8, 2023

Enhancing Gesture Recognition for Musical Conducting: A Study on Diverse Data Classification and Stacked Neural Network Architectures

Gideon Woo, Faith Tan, and Herbert H. Tsang
Applied Research Lab, Trinity Western University, Langley, British Columbia, Canada

Abstract— **This study addresses the limitations of many gesture recognition algorithms, which predominantly employ machine learning-based approaches tailored to specific types of gestures, leaving niche gestures such as musical conducting gestures largely unexplored. To advance the research in musical conducting gesture recognition, we focus on two key aspects: (1) broadening the dataset to encompass various conducting speeds and investigating its impact on performance, and (2) introducing a stacked neural network architecture to explore performance improvements beyond conventional node increase. The study demonstrates that incorporating diverse data significantly enhances performance and that stacking neural network layers yields notable performance gains.**

I. Introduction

In the context of musical conducting, conducting gesture are the specific movements and gestures made by a conductor to convey essential information to the musicians during a performance. These gestures serve as a non-verbal communication method, guiding the ensemble on aspects of tempo, dynamics, phrasing, articulation, entrances, and other musical elements.

Using advanced technologies, researchers began to explore complex and nuanced gestures using machine learning methods. Recently, our research has addressed timing and articulation aspects of the gesture with some successes [1] [2] [3]. In this paper, we will explore the use of stacked neural network architecture for this special type of gesture recognition.

II. Literature Review

In general, there are two approaches in gesture recognition using machine learning: vision-based and sensor-based.

A. Machine Learning for Gesture Recognition

1) Vision-Based: Munasinghe et al. had used a feed-forward neural network (NN) and implemented a real-time, computer-vision-based gesture recognition workload [4]. This NN consisted of three hidden layers, with 100 nodes per layer. The maximum accuracy he achieved with this method, even with "better" lighting, was 85% [4].

Cristina Mata tested two methods for hand pose estimation using image recognition [5]. She implemented Conditional Random Field (CRF) for fine parts segmentation in two different ways, one using a convolutional neural network (CNN), and the second using a recurrent neural network (RNN). When Mata implemented the CNN, efficiency was an evident problem, but in conjunction with the RNN, the efficiency improved. However, implementing the CNN with the RNN created an issue with memory [5].

A comparison between four different machine learning algorithms for hand gesture recognition was performed by Triguerios, et al. [6] (k-Nearest Neighbour (kNN), Naive Bayes (NB), Support Vector Machines (SVM), and Artificial Neural Networks (ANNs)). Data was collected as low-resolution depth images and pre-processed by feature extraction and hand segmentation. Of the four algorithms, the ANN returned the highest accuracy at 96.99%, though it was noted that it took the longest time to train. Following the ANN was the k-NN method, with an accuracy of 95.45% [6].

Kim, et al. developed a new hand gesture recognition method using a restricted column energy (RCE) neural network with dynamic time warping (DTW) [7]. DTW is a popular algorithm used in gesture recognition, but has drawbacks that mainly include time especially with large data sets. RCE neural networks use Euclidean distance in distance measurements; to solve this issue, the researchers have used DTW distance in place of it. Using an IMU sensor, they got 3-axis gyroscope data and 3-axis accelerometer data, which was preprocessed by Arduino to standardize the length of the data. This combined method of DTW + RCENN returned a maximum accuracy of 98.6%, which is at minimum 4% higher than either of those methods on their own [7].

2) Sensor-based: Chistyakov and Chepin went with a different approach using a Microsoft Kinect to collect gesture data in three dimensions [8]. All of this data was then converted into a bit matrix through the mapping of the hand trajectories and subsequently fed into a convolutional neural network model that consisted of two convolution layers with a kernel size of 5 x 5. This system was designed with continuous motions in mind where there are no clear starting or end points to the gesture. Although accuracy numbers were not published, this is still a solid framework to refer to when conducting any type of research with gesture recognition.

Sign language recognition represents one of the more intricate hand gesture applications that has garnered significant attention. Mekala et al. conducted comprehensive research in this domain, developing a system capable of discerning the signed letters [9]. Their approach involved image processing techniques with substantial computational load to isolate the hand from the background. Subsequently, this processed information was fed into a convolutional neural network comprising three layers.

Remarkably, the algorithm achieved a remarkable 100% accuracy in classifying all the letters of the alphabet. However, when introducing noise into the images, the overall accuracy diminished. This exposed the system's vulnerability to noise, with an average noise immunity of 48% [9].

B. Neural Networks

Once the data are collected, we will employ a classification framework to analyse our data. In our previous studies, we have found Recurrent Neural Network (RNN), Long-Short Term Memory (LSTM), and Gated Recurrent Unit (GRU) to be most accurate [1] [2]. At that time, we only tested using data that were collected with one speed, 80 BPM.

1) Recurrent Neural Network (RNN): The recurrent neural network (RNN) represents an enhanced version of the feed-forward neural network. This improvement is achieved by enabling each node to access data from prior time steps, thereby incorporating contextual information and leading to improved overall performance. RNNs demonstrate heightened proficiency in recognizing patterns and features within time series data, such as conducting gestures. Although not the highest-performing algorithm, RNNs are included as a baseline for comparison against more advanced neural networks in our study.

2) Long-Short Term Memory (LSTM): The long-short term memory (LSTM) neural network represents an advancement over the recurrent neural network (RNN) model. Each LSTM node possesses a memory component. The memory capability of LSTM nodes allows them to retain crucial information, impacting their future outputs. This enhancement in comparison to a conventional RNN mirrors how past information influences decision-making in the present, proving particularly valuable in longer sequences where RNNs often encounter challenges due to the vanishing gradient problem.

The project leverages NVIDIA's CUDA Deep Neural Network (cuDNN) optimized implementation of LSTM within the Keras library, significantly reducing training time [10].

3) Gated Recurrent Unit (GRU): The gated recurrent unit (GRU) represents another advancement in recurrent neural networks. Similar to LSTM, GRU allows each node to access memory, but with a streamlined approach using only two gates: update and reset. The update gate governs the utilization of incoming data, selectively storing useful and relevant information in memory while discarding the rest. On the other hand, the reset gate is responsible for maintaining existing memory by clearing away less relevant portions.

Due to the reduced number of gates compared to LSTM, GRU is expected to offer faster training performance. Additionally, an optimized version of GRU, tailored for efficiency, is available in the Keras library, which was employed in this project.

C. Stacking Neural Networks

In the image recognition world with their use of convolutional neural networks (CNN), getting extra performance out of training the neural network can be difficult. As discussed in Stanford's convolutional neural network class, the overall structure of a CNN is three-dimensional [11]. Typical variables to increase include the kernel size and filter size of a specific neural network layer which increases the complexity in hopes of the algorithm learning more. With the 3-D structure of the convolutional neural network, any increase of complexity will also require a quadratic increase of computational power to compensate. This can quickly get out of hand as there is only so much the algorithm can learn, which means that increasing complexity is subject to the principle of diminishing returns. Another option would be to stack multiple simple convolutional neural network layers on top of one another to improve performance.

Simonyan and Zisserman suggested an alternative approach based on a stack of convolutional neural networks [12]. The images were passed through a stack of convolutional neural network layers with 3 x 3 filter size, which is the smallest size to capture information around a given pixel. They were able to show that adding more "depth" to the neural network structure was more effective than fewer layers with more complex parameters. They demonstrated by one model with half the number of layers with a filter size of 5 x 5 being outperformed by their 3 x 3 filter size model [12].

Another excellent stacking technique example comes from Adavanne et al. where it was used for bird audio detection [13]. Their neural network structure used 16 2-D convolutional layers, each with a 3 x 3 filter size. This data was then fed into a GRU layer before the final output. Results were respectable, the algorithm identify the presence of a bird call/tweet within an audio clip with a validation accuracy of 88.1% and a training accuracy of 95.1%.

A good indication that the same technique of deepening the neural network would prove useful outside of just image recognition with convolutional neural networks comes from Chen et al. [14]. They combined the use of one-dimensional convolutional neural networks with Surface Electromyography signals (the electrical signals that the brain sends to muscles to control them) from sensors on the wrist to test gesture recognition. They were able to show how stacking multiple layers of simple CNNs produced the best results even when compared to a model combining LSTM and convolutional layers [14]. This is a great sign that the same principle of simplifying could apply to stacking multiple layers of other neural networks like LSTM and GRU to get increased performance on gesture recognition workloads.

The objectives of this paper are as follows:

- To evaluate the performance differences caused by implementing a stacked neural network structure on the musical conducting gesture workload.
- To look at the effects from expanding the data set to include conducting gestures of various speeds.

III. Methods

A. Data Collection

We are using an mobile phone to capture our data. This phone gyroscope and accelerometer. An app was built that could be easily shared between users for easy data collection.

This enabled easier scaling for data collection due to circumstances that require everything be done remotely (e.g., COVID-19 pandemic). A metronome was used to help keep the gesture speed consistent with the android phone in the right hand recording the sensor data using the app that was developed.

Previously, we have collected beating patterns at 80 beats per minute (BPM) [1] [2]. Then we collected new data by varying the BPM. In this study, we have collected data at 60, 70, 80, 90, and 100 bpm. Also, these data contain different beat patterns (e.g., two, three, and four beats) and articulations (e.g., legato, normal, and staccato).

B. Neural Network Implementations

We employ a neural network that can be trained on a single graphics processing unit (GPU) and requires less than four gigabytes of video RAM. This ensures that the experiments can be realistically conducted on typical consumer hardware. All of the neural networks had 4,096 internal nodes and were given 500 epochs to give achieve the best results from a 60, 20, 20 split of all the data: 60% for training, 20% for testing, and 20% for validation. The models that gave the best validation accuracy were kept and evaluated for our final results. There are two other implementations that were tested in the experiments to see if there were any improvements to be had without increasing the number of internal nodes drastically.

1) Bi-directional Implementation: This implementation requires the recurrent neural network layer be duplicated where one is being fed data starting from the past and into the future while the other is fed data starting from the future and into past. The output of the node and its corresponding duplicate are then combined.

2) Stacking Recurrent Layers: This is a commonly employed technique in the field of image recognition. For convolutional neural networks (CNNs), utilizing multiple simpler layers with lower filter sizes (analogous to internal nodes in our context) yields superior accuracy while being computationally efficient and requiring less training time compared to using a single complex layer.

The advantage of layer stacking lies in promoting deeper learning, as information learned from one layer is passed down to subsequent layers, enabling the extraction of more detailed and abstract features. The same principle holds potential for time-series data and other neural network structures, as observed in this project. To investigate this, the neural network implementations include a stacked counterpart with four recurrent neural network (RNN) layers instead of one.

The internal nodes of the first two layers were set at 1,024 respectively, the third layer having 512, and the fourth layer having 256 internal nodes resulting in a total of 2,816 internal nodes while the single-layered implementations have the full 4,096 internal nodes. Figure 1 shows the visualization of this stacked structure.

IV. Results and Discussion

In order to test the stacked structure's improvement over a single layer implementation of neural networks, a direct comparison is done by training two sets of neural networks: one with the stacked implementation and one without.

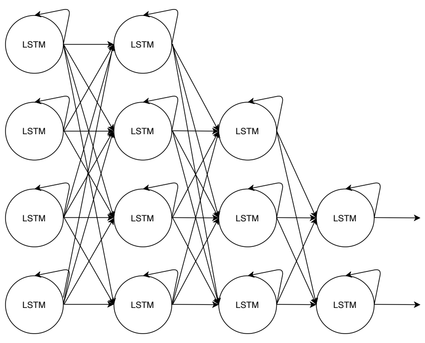

Fig. 1. Visualization of the stacked LSTM layers

To test the influence of more diverse gesture data at different speeds, every algorithm is trained on 4 different sets of data (60 and 80 BPM; 80 and 100 BPM; 60, 80 and 100 BPM; and all the data available). All of the models trained, without the inclusion of 70 and 90 BPM gesture data, reveal how an algorithm might respond to the inclusion of just slower data, just faster data, and both slower and faster data. The models produced using all the data available (including 70 and 90 BPM) is used to evaluate overall algorithm performance when given even more data to work with. Table I shows all the different experiments that were run as part of this study.

TABLE I
THIS TABLE SUMMARIZED THE DATA SETS AND THEIR CORRESPONDING CONDUCTING SPEED USED IN THIS STUDY.

Experiments	Beats Per Minute				
	60	70	80	90	100
Stacked vs. Single Layer			X		
Slower Data	X		X		
Faster Data			X		X
Slower and Faster Data	X		X		X
All Data	X	X	X	X	X

A. Stacked Layer vs. Single Layer

In our testing, adding stacked layers only resulted in a marginal performance increase for models trained with 80 BPM gesture data when classifying beat patterns, as depicted in Table II. A standard RNN, on the other hand, not only failed to show improvement but also exhibited a performance decline with the stacked structure. Considering the overall subpar performance of both RNN implementations in beat pattern recognition, it's possible that recurrent neural networks are not well-suited for this specific classification task. LSTM did show a slight increase in accuracy, while GRU maintained its performance but demonstrated improved standard deviation, indicating greater consistency across multiple runs. One potential reason for this modest improvement in performance could be attributed to the strong performance of the single-layer implementation. When occasional subpar input data is taken into account, there is limited scope for substantial enhancements with the addition of stacked layers.

TABLE II

RESULTS FOR BEAT PATTERN AND ARTICULATION FOR 80 BPM, WHERE S.D. = STANDARD DEVIATION

Algorithms	Single Layer Accuracy %	Stacked Layers Accuracy %	Single Layer Cohen's Kappa	Stacked Layers Cohen's Kappa	Single Layer S.D.	Stacked Layers S.D.
Beat Pattern (Two, three and four beats)						
RNN	45.90	44.14	0.188	0.162	0.0010485	0.0008627
LSTM	99.80	99.84	0.997	0.998	0.0000073	0.0000103
GRU	99.80	99.80	0.997	0.997	0.0000137	0.0000073
Articulation (e.g., legato, normal, and staccato)						
RNN	62.82	71.84	0.442	0.578	0.0022447	0.0064769
LSTM	98.82	99.29	0.982	0.989	0.0000308	0.0000178
GRU	99.21	99.25	0.988	0.989	0.0000338	0.0000198

On the articulation recognition side, the stacked implementations all gave improved performance compared to the single-layered ones. The improvement for RNN is the most obvious with a 9.02% increase in overall accuracy. Improvement was small in the case of GRU while LSTM showed a larger increase in performance when moving to a stacked neural network architecture. The same reduction in the standard deviation as seen in the beat pattern recognition is also present here and is a good sign that the stacked structure improves consistency of performance.

The overall positive effects of the stacked neural network architecture shown in this part of the study meant that it made sense to continue with the stacked structure going forward. The poor performance exhibited by the basic RNN in this portion of the study suggested that it is not a good fit for musical conducting gesture classification with data collected from a smartphone. This is why it was removed from further experimentation with the expanded data set.

B. Expanded Data Set

Working with an expanded data set, one intuitively would assume that the overall accuracy of the algorithms would decrease since there is more variety in the data for the algorithms to learn from causing more confusion. However, Table III, Table IV, Table V, and Table VI show that it was not necessarily the case. In the following sections, we will examine the results in more detail.

1) Performance with Slower Data: Table III displays the outcomes of neural networks trained on data at 60 and 80 BPM. The performance differences, when compared to the exclusively 80 BPM results, exhibit a varied pattern.

The overall accuracy for beat pattern recognition showed a slight decline, which was anticipated due to the increased data variety and corresponding complexity for the algorithms to analyze. Notably, the implementation of bi-directional neural networks demonstrated a significant improvement, surpassing the results achieved solely with 80 BPM data.

The articulation recognition performance demonstrates improvement for both LSTM and GRU models, indicating that the inclusion of slower data contributed to better generalization in distinguishing between different articulations in the gesture. However, it is noteworthy that the bi-directional implementations had no notable impact on LSTM, while GRU exhibited a slight improvement in performance.

2) Performance with Faster Data: Table IV shows the results from the neural networks being trained on 80 and 100 BPM data tell a different story when compared to the models trained on 60 and 80 BPM.

Using the 80 BPM results as a baseline, beat pattern recognition barely fell in the case of GRU while LSTM performed better. These results suggest that the faster data was easier to handle showing better results compared to the slower data models. This could be due to the window size. Since all training was done with the same window size, the slower 60 BPM data meant that there is less of the gesture in each window, which could contribute to the lower performance with slower data. The higher speed 100 BPM data meant that more of the gesture is within the same window size, which could explain the better results. The difference between bi-directional and uni-directional implementations was non-existent with LSTM and only slightly better with GRU. This can also be the result of the easier workload due to the window size limitation.

BiGRU here did outperform GRU by a sizable amount, but BiLSTM actually performed worse compared to LSTM.

Articulation recognition performance exhibited a slight improvement compared to the original 80 BPM models. Surprisingly, the slower data models outperformed the faster data models. This observation may be attributed to the nature of the gesture classification in this study. In musical conducting, the distinctions between articulations become more discernible at lower speeds, allowing more time for each beat to occur. Conversely, at faster speeds, the differences between articulations diminish, as there is less time available for each articulation to be expressed during each beat.

Notably, BiGRU outperformed GRU significantly, while BiLSTM performed worse compared to LSTM.

3) Performance with Faster and Slower Data : Table V shows the results where each dataset contain all speeds. In beat pattern recognition, both GRU and LSTM models yielded overall inferior results compared to the original 80 BPM models, as anticipated. However, BiGRU showed a remarkable improvement, achieving an impressive overall accuracy of 99.92%. BiLSTM also provided a performance boost compared to LSTM, though not as substantial as the improvement seen with BiGRU over GRU. This is the only scenario where a GRU algorithm outperformed LSTM and BiLSTM.

In the context of articulation recognition, all models con-

TABLE III

RESULTS OF EXAMINING THE PERFORMANCE WITH SLOWER DATA (60 AND 80 BPM)

	Beat Pattern			Articulation		
Algorithm	Overall Accuracy	Cohen's Kappa	Standard Deviation	Overall Accuracy	Cohen's Kappa	Standard Deviation
BiGRU	99.84%	0.9976	0.0000044	99.37%	0.9905	0.0000114
BiLSTM	99.89%	0.9984	0.0000044	99.55%	0.9933	0.0000042
GRU	99.66%	0.9949	0.0000070	99.34%	0.9901	0.0000170
LSTM	99.79%	0.9969	0.0000039	99.55%	0.9933	0.0000070

TABLE IV

RESULTS OF EXAMINING THE PERFORMANCE WITH SLOWER DATA (80 AND 100 BPM)

	Beat Pattern			Articulation		
Algorithm	Overall Accuracy	Cohen's Kappa	Standard Deviation	Overall Accuracy	Cohen's Kappa	Standard Deviation
BiGRU	99.78%	0.9968	0.0000012	99.36%	0.9904	0.0000173
BiLSTM	99.89%	0.9984	0.0000047	99.12%	0.9868	0.0000354
GRU	99.76%	0.9964	0.0000036	99.28%	0.9892	0.0000100
LSTM	99.89%	0.9984	0.0000032	99.47%	0.9920	0.0000240

TABLE V

RESULTS OF EXAMINING THE PERFORMANCE WITH A MIXED OF FASTER AND SLOWER DATA (60, 80, AND 100 BPM)

	Beat Pattern			Articulation		
Algorithm	Overall Accuracy	Cohen's Kappa	Standard Deviation	Overall Accuracy	Cohen's Kappa	Standard Deviation
BiGRU	99.92%	0.9988	0.0000009	99.46%	0.9919	0.0000089
BiLSTM	99.82%	0.9973	0.0000035	99.58%	0.9937	0.0000068
GRU	99.76%	0.9964	0.0000030	99.35%	0.9904	0.0000127
LSTM	99.78%	0.9967	0.0000051	99.42%	0.9913	0.0000124

TABLE VI

OVERALL BEAT RECOGNITION PERFORMANCE RESULTS (60 - 100 BPM).

	Beat Pattern			Articulation		
Algorithm	Overall Accuracy	Cohen's Kappa	Standard Deviation	Overall Accuracy	Cohen's Kappa	Standard Deviation
BiGRU	99.91%	0.9986	0.0000018	99.59%	0.9938	0.0000044
BiLSTM	99.93%	0.9990	0.0000008	99.62%	0.9942	0.0000096
GRU	99.83%	0.9974	0.0000025	99.60%	0.9940	0.0000045
LSTM	99.83%	0.9974	0.0000025	99.64%	0.9946	0.0000025

tinued to surpass the 80 BPM models. When compared to models trained solely on slower or faster data, the performance was slightly better for most cases, except for LSTM. This indicates that with both slower and faster data, the algorithms demonstrated better ability to identify significant commonalities between the speed-variable gestures, leading to improved generalization, albeit moderately for this data combination. Notably, both bi-directional models outperformed their uni-directional counterparts in this context.

4) Performance with All Data Available: The most compelling results arise from the models generated using the entire available dataset (Table VI). Contrary to the initial hypothesis, introducing greater data variety does not lead to poorer algorithm performance.

Overall, beat recognition performance is commendable, with LSTM and GRU models surpassing their 80 BPM counterparts. Notably, the BiLSTM model outshines all others, achieving an unprecedented 99.93% overall accuracy. Additionally, the GRU model demonstrates superior performance compared to all other GRU models trained in this study.

Articulation recognition performance is consistently strong. All models trained on this dataset outperformed models trained with any combination of the 60, 80, and 100 BPM data.

Notably, GRU and LSTM surpassed the original 80 BPM models in beat recognition, and all algorithms excelled in articulation recognition. This suggests that the classification algorithms have reached a tipping point where increasing data variety improves performance rather than diminishing it.

One plausible explanation for these results lies in the neural networks' ability to extract common features. The inclusion of 70 and 90 BPM data expanded the dataset, thereby increasing the differences between gestures. Consequently, there are fewer shared features among the various gestures, making it easier for algorithms to identify distinctive patterns and characteristics for beat patterns and articulation. This enhanced model generalization leads to improved algorithm performance.

V. Conclusion

The art of musical conducting involves intricate gestures that possess the ability to convey a wealth of information when executed with skill. Each of these gestures serves the purpose of communicating crucial aspects such as the time signature, articulation, and tempo of the musical piece. Extracting this information presents a daunting challenge for computational intelligence methods to grapple with.

In this research, we sought to move the computational intelligence paradigm forward by improving gesture recognition specific for musical conducting gestures. Drawing influence from convolutional neural networks used in image recognition, a stacked neural network architecture was implemented in hopes of improving performance without increasing the number of internal nodes. One additional facet of the musical conducting gesture, speed, was also added as a variable to see how effective the neural network algorithms can be in a previously untested way. By looking at the results of these experiments, several important conclusions can be made.

Firstly, the basic recurrent neural network (RNN) proves unsuitable for classifying musical conducting gestures. It falls short in comparison to LSTM and GRU, particularly when implemented in a stacked manner. As computing hardware advances, the advantage of faster training times with simple RNNs becomes less relevant, especially as more complex gestures emerge.

Secondly, stacked neural network structures do indeed offer a performance benefit when compared to their conventional single layer counterparts when it comes to classifying the musical conducting gesture. This can be seen by an improvement in overall accuracy in most situations and a reduction of standard deviation pointing to increased performance consistency. Other than the improvement of LSTM classifying articulation, the improvement from single layer to stacked layers was slight. This can be attributed to the fact that the single layer implementations performed very well already achieving overall accuracy of no lower than 98.82% in any workload. An increase in data variety might have caused a wider gap in performance between the single layer implementation and the stacked architecture, but that was not tested in this study and is a further step that can be taken in future research.

Thirdly, experiments with an expanded dataset reveal that algorithm performance deteriorates when exposed to one or two additional varying-speed datasets for beat pattern classification in conducting gestures. Beyond that, introducing more than two varying-speed datasets pushes the algorithms to perform better compared to training on just one speed.

Fourthly, articulation recognition does not suffer any drop in performance when different speeds of the same gesture are introduced. This shows that the differences and features distinguishing one articulation from another is easier for neural networks to pick out consistently compared to beat pattern.

As gesture recognition evolves, techniques must adapt to handle increasingly complex gestures. This study demonstrated the efficacy of a layered approach with neural networks, stacking multiple simple layers. These insights can enhance future machine learning models. Additionally, the experiments highlighted the impact of varied gesture speeds on algorithm performance, offering valuable knowledge for future research in time-series data classification and dataset creation strategies.

References

[1] J. van Heek, G. Woo, J. Park, and H. H. Tsang, "Analysis and selection of classifiers for gesture characteristics recognition based on MGRS framework," in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2019, pp. 2972–2980.

[2] J. van Heek, G. Woo, J. Park, and H. H. Tsang, "A comparison of computational intelligence techniques for real-time discrete multivariate time series classification of conducting gestures," in *Computer Vision Systems*, D. Tzovaras, D. Giakoumis, M. Vincze, and A. Argyros, Eds. Cham: Springer International Publishing, 2019, pp. 573–585.

[3] F. Tan, G. Woo, and H. H. Tsang, "CGLER: Laban effort framework analysis with conducting gestures using neural networks," in *IEEE Symposium Series on Computational Intelligence (SSCI)*, 2020, pp. 1452–1459.

[4] M. I. N. P. Munasinghe, "Dynamic hand gesture recognition using computer vision and neural networks," in *2018 3rd International Conference for Convergence in Technology (I2CT)*, 2018, pp. 1–5.

[5] C. Mata, "Two approaches to robust hand pose estimation: Generative modeling and semantic relations," Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, 2019.

[6] P. Trigueiros, F. Ribeiro, and L. P. Reis, "A comparison of machine learning algorithms applied to hand gesture recognition," in *7th Iberian Conference on Information Systems and Technologies (CISTI 2012)*, 2012, pp. 1–6.

[7] M. Kim, J. Cho, S. Lee, and Y. Jung, "IMU sensor-based hand gesture recognition for human-machine interfaces," *Sensors*, vol. 19, no. 18, p. 3827, Sep 2019. [Online]. Available: http://dx.doi.org/10.3390/s19183827

[8] I. S. Chistyakov and E. V. Chepin, "Gesture recognition system based on convolutional neural networks," *IOP Conference Series: Materials Science and Engineering*, vol. 498, p. 012023, apr 2019. [Online]. Available: https://doi.org/10.1088%2F1757-899x%2F498%2F1%2F012023

[9] P. Mekala, Y. Gao, J. Fan, and A. Davari, "Real-time sign language recognition based on neural network architecture," in *2011 IEEE 43rd Southeastern Symposium on System Theory*, 2011, pp. 195–199.

[10] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[11] "Convolutional neural networks for visual recognition," [Online; accessed July 20, 2019]. [Online]. Available: http://cs231n.github.io/convolutional-networks

[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

[13] S. Adavanne, K. Drossos, E. Çakir, and T. Virtanen, "Stacked convolutional and recurrent neural networks for bird audio detection," in *2017 25th European Signal Processing Conference (EUSIPCO)*, 2017, pp. 1729–1733.

[14] L. Chen, J. Fu, Y. Wu, H. Li, and B. Zheng, "Hand gesture recognition using compact cnn via surface electromyography signals," *Sensors*, vol. 20, no. 3, 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/3/672