An Analysis of Muscle Gesture Classification with Neural Networks

Submitted by:
SHIVRAM RAMKUMAR

Research Question

This research project seeks to identify the impact of the various network layers and parameters on the learning curve of a neural network that attempts to classify muscle gestures. As artificial intelligence and machine learning drive most modernization trends, a greater understanding of the fundamental algorithms will serve to reach objectives more efficiently. Regarding the algorithms, many users employ high-level software that simplifies the process and consequently does not require the fullest understanding of the equations. However, a better knowledge of those principles can lead to more efficient generalization models with a better ability to explain how it works. This begs the question: how do the parameters involved optimize and impact machine learning models? The research approaches the concept using a project-driven framework, specifically fine-tuning the algorithm segments for real-time wrist gesture classifiers using electrical data streaming.

Introduction

Today, the world's most influential organizations are exploring machine learning to the fullest extent possible, yet much progress remains to be made on the detailed implementation of the AI models. The algorithms consist of pure mathematics; using massive sets of data, these mathematical expressions execute different functions to make a predictive model. Out of these, the simplest iteration would be 2-dimensional linear scatterplots. Given a series of known data, a general prediction model can be created such that the inputs predict an output very close to the actual value, as tested during model validation. These have progressed to logistic regressions (using logistic curves) for binary classification, support vector machines (a line that divides two groups in scatterplots – predicts based on side of the line), and gradient descent (multivariate expressions to modify parameters based on a loss function, or how far away an output is from the expected). Regardless of their basic structure, they suffice for many of the necessary data programs in generating accurate predictions.

However, the level of complexity can be increased. The expressions above can be manually programmed to generate an output, but when the data ascend to higher dimensions (more inputs for output) or have more complicated structures (like a circle cluster on a scatterplot as opposed to a line), the simple models do not always suffice. Even then, pure-math solutions reflect the structure adequately. Eventually, the data either becomes too expansive or too convoluted, at which point neural networks were developed. Their designs are modeled after a human brain, with their different layers and inputs called "neurons," but they are so far only specific task oriented. Researchers quickly abandoned the biological roots with decades of manipulation and experimentation and have instead adopted a variety of algorithms that focused on different learning methods and ways of accommodating error (backpropagation – change parameters to reflect accumulated error). As processing power has increased, simpler programming languages have been created to achieve more powerful tasks such as creating different types of neural networks.

Currently, public interest in AI and machine learning has spiked especially since the introduction of self-driving cars, facial recognition, and search engines. The abundance of public resources has provided stable foundations to delve into this field easily. With many of the classification and prediction problems that exist today, machine learning (although its impact should not be exaggerated) has much potential in these areas.

One specific area is teleoperated robotics. Many tasks in the workforce and research environments are unpredictable and consequently require human control. Some worksites are hazardous to human health or survival, for example, outer space, deep ocean, nuclear or biologically/chemically toxic environments, mines, construction sites, or police operations According to Sheridan (1989), teleoperation can be used as a solution, and is the extension of a person's sensing and manipulation ability (p. 487). Telerobotics is a form of teleoperation where a human operator supervises a telerobot as it executes the task set by the human. Another use for telerobotics is in handicapped aids - in such a case, the haptic devices are required to be non-static for mobility and inconspicuous to be more appealing. More uses of non-static haptics would be in firefighting and military operations, where avoiding loss of life and maintaining security are paramount. To aid in these tasks, however, operator stress and task errors can be notably reduced through a haptic device that allows the operator to perceive forces and obstacles (Diolaiti, 2002, p. 67).

With the growing shortage of doctors in the world, such teleoperated techniques will become increasingly useful for remote operations and training simulations. In these cases, the ability to pass down and spread knowledge is crucial. Using an armband, for example, adopting machine learning algorithms to classify (based on muscle inputs) the different actions in real-time can greatly assist in understanding the surgical process, and possibly even replicated with machinery and their precision equipment. It may even help in VR simulations, where training newer generations can occur using haptic feedback to pass existing knowledge more efficiently. And in creating the neural networks for classification, a better understanding of the parameters, layers, and impacts may serve to bolster a variety of industries, medical among them.

Research Methodology

The armband in question for this research is the Myo Armband, an electromyography band worn on the forearm that streams relative muscle data at 200Hz. Using the documentation and API for the device it was possible to obtain samples of the EMG readings for later analysis. The numerical data was collected as a function of time using the following guidelines:

1. Five hand/wrist positions were set (up, down, left, right, and at rest)
2. Each of those positions were held for a minute, with data recording begun
3. Specific muscles targeted
4. Two hundred arrays of eight EMG values per second

There are then several components for identifying any impacts. Using the results, different applications of statistics were used to determine any features that could be exploited in creating a classification algorithm. For example, quantitative 2D scatterplots for each gesture (the plots representing EMG values) illuminate any trends by the signal strength alone and not time. Deviations and variances were calculated to find more holistic trends.

Following the data science and statistics is the programmatic side. Using Keras, a TensorFlow API created by Google in the Python programming language, different neural network models were created, and their learning curve analyzed. This includes the time taken to train the model to a certain percentage of validation accuracy (validation to make sure the model is not memorizing data and predictions). The shape of the curve was closely examined to determine a stable learning curve (at which point the data was not learning too quickly and overshooting nor taking too long to reach high accuracy). Different neural network layers were introduced in different models; an automatic callback-based learning rate was used; different prediction windows demonstrated the impact of lengthening or decreasing the number of signals per classification.

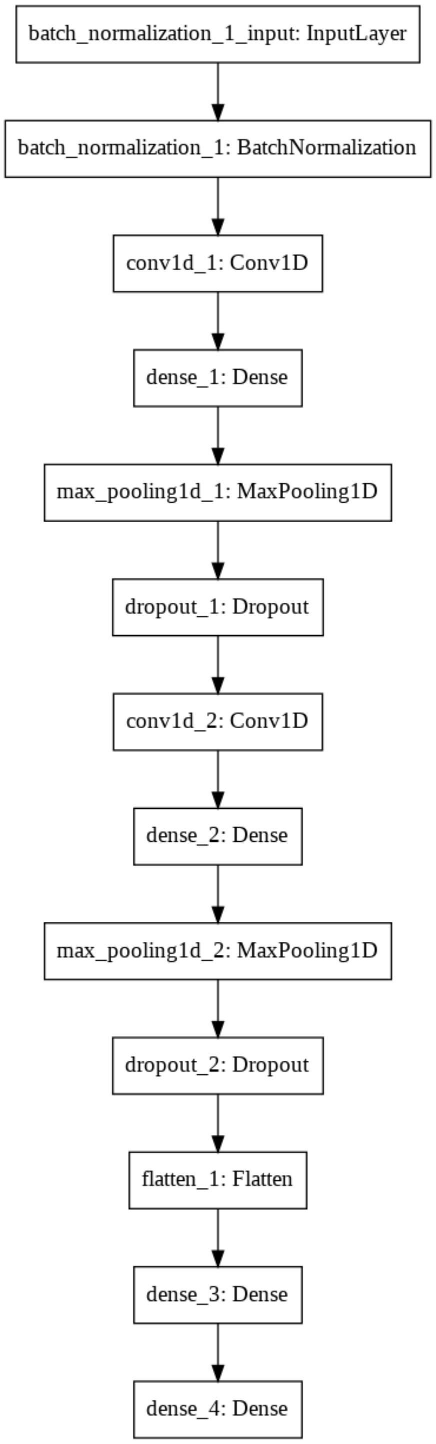


Figure 1

Analysis and Discussion

The goal of this study was to optimize the parameter-tuning process in creating neural networks and to compare the model metrics to find the impact of each feature. This propels a fundamental and intuitive understanding of how network layers and higher-dimensional mathematics work. With such research, teaching neural networks can be streamlined to better indicate how each network layer affects the model architecture and the model itself**.** The analysis was completed via statistics and visualization algorithms, as well as using Python's data libraries and the Keras API to find the training and validation accuracy, loss, and training time.

After the five positions were recorded separately and joined (in the order of down, left, rest, right, up), the data was graphed as a line chart for holistic visualization.

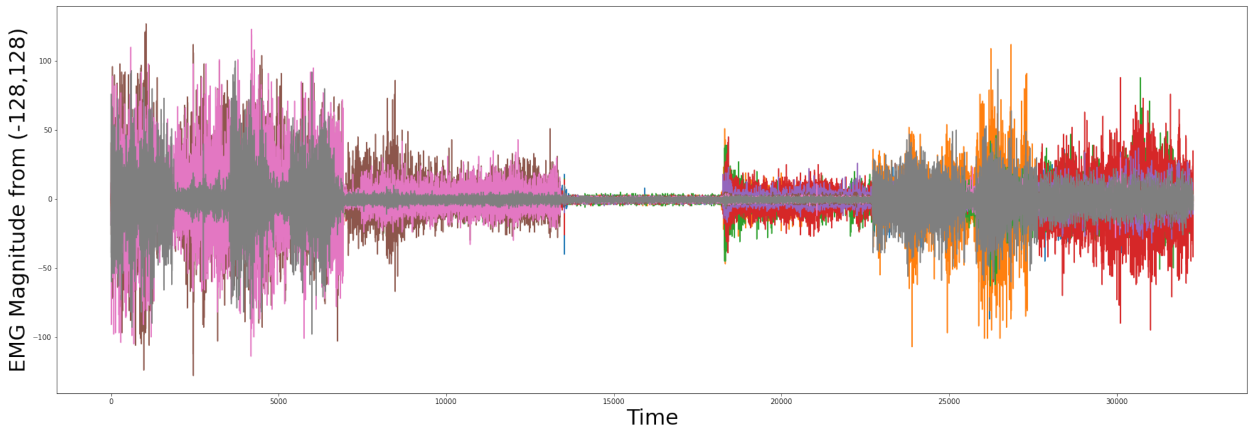


Figure 2

A significant limitation of the Myo armband was that it returned EMG values not in terms of electrical readings but as unitless magnitudes from (-128, 128). In terms of repeatability, this limits the scope of the Armband for widespread use, since the training algorithm will have to rely on complete retraining; because the units are not absolute, biological standards are not applicable and the existing variances in forearm physiology due to different ratios of muscle, fat, nerves, skin, and chemical compounds. Regardless, the graph illustrates several visually obvious trends:

1. The third position, rest, is visually and numerically deviant from all other categories in that the muscular activity remains low for all eight EMG streams. The rest position is therefore easier to classify assuming proper relaxation is achieved.
2. The down and up positions have the greatest magnitude of readings for the most streams, whereas the left and right streams are nearly half in greatest magnitudes. This relates to the biology and physical structure of wrist movement and related muscle activation.
3. The down and left positions, at least for this user, resemble greater activation in the same muscles (i.e. similar pink, gray, and red streams visually identifiable). Distinguishing between these positions thus relies heavily on the magnitude of the reading.
4. Even for consistent positions, the EMG streams fluctuate and vary greatly as a function of time, and in the case of the up position, even different muscles can activate at inconsistent magnitudes for the same position. This was considered when developing a classification algorithm using Keras and TensorFlow.

After verifying these trends and creating a Keras classification algorithm using the layer architecture defined in *Figure 1*, the Myo Armband API was used to develop a Python program that performed instantaneous, real-time prediction using the armband data stream and the exported model. The classification itself had a prediction accuracy of ~80%, but the model training revealed a peculiarity:

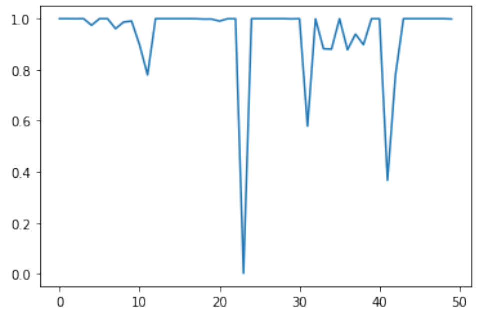


Figure 3: Validation accuracy as a function of training iterations

When validating on a random portion of the training data defined in *Figure 2* to verify accuracy, the model starts with nearly perfect accuracy and fluctuates wildly (note the massive accuracy drop around 23 iterations). Ideally, the learning curve should resemble a gradual exponential with from beginning at zero (because the model has not learned yet) to a plateau at the end, indicative of the decreasing training loss and parametric optimization.

The likely sources of error were narrowed down to overfitting (where the model trains too much on the noise and random fluctuations in the data instead of the features), data memorization, programmatic bugs, or data preprocessing errors. In debugging, the first step is generally data visualization to determine the quality of the data, and whether it contains too much noise to develop a functioning algorithm. Some of these visualization tools are machine learning algorithms in themselves; for instance, Principal Component Analyses (PCA) and t-Distributed Stochastic Neighbor Embedding (t-SNE) are two plots that can make it easy to identify any data errors by creating a scatterplot of high dimensional data into two dimensions.

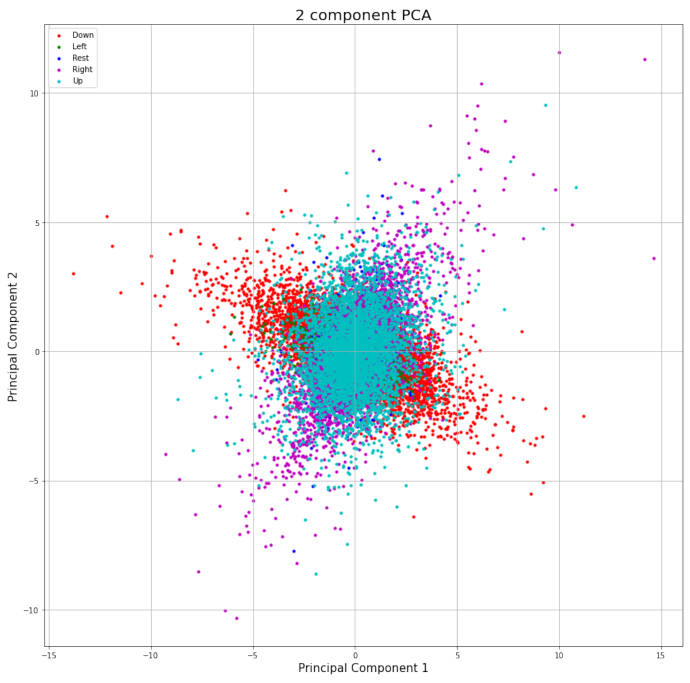


Figure 4: Principal Component Analysis

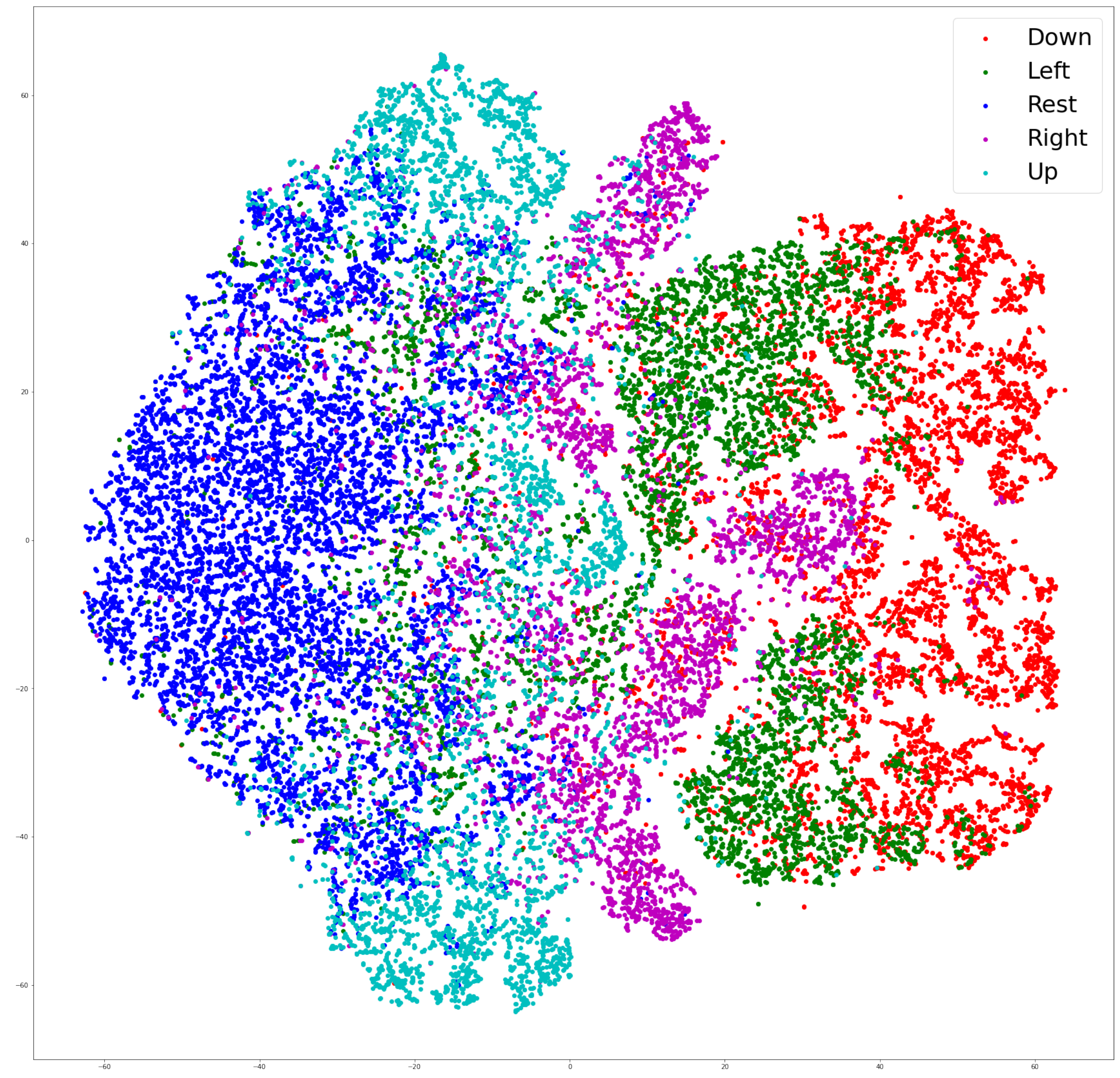
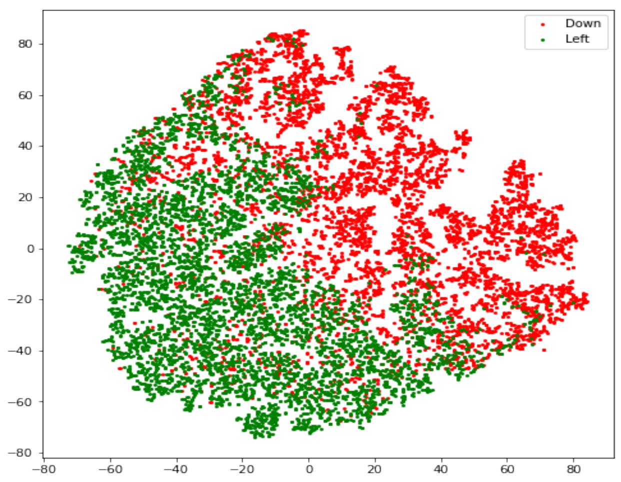
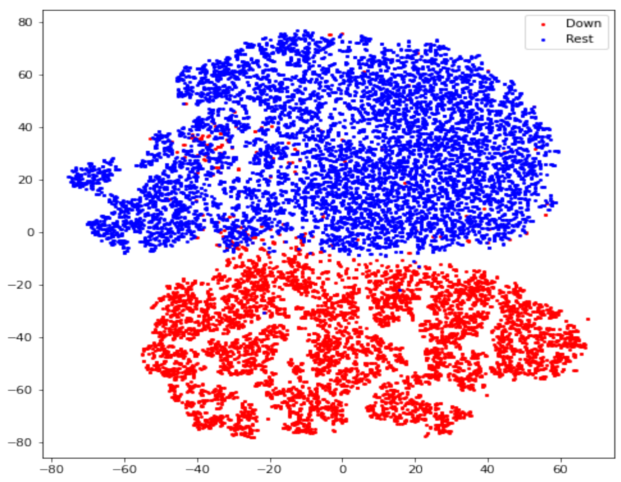
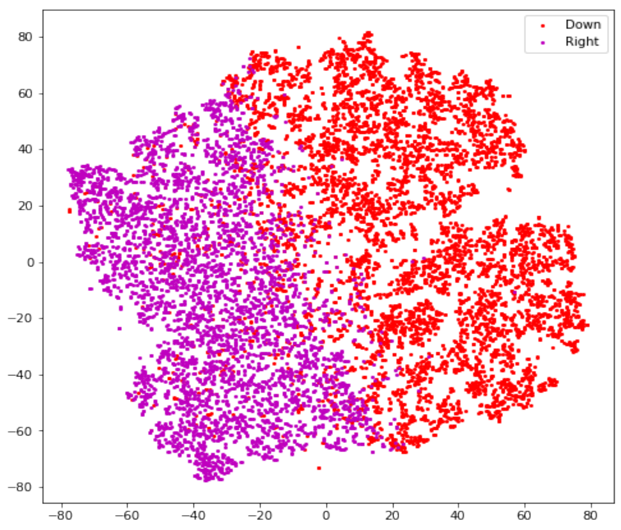


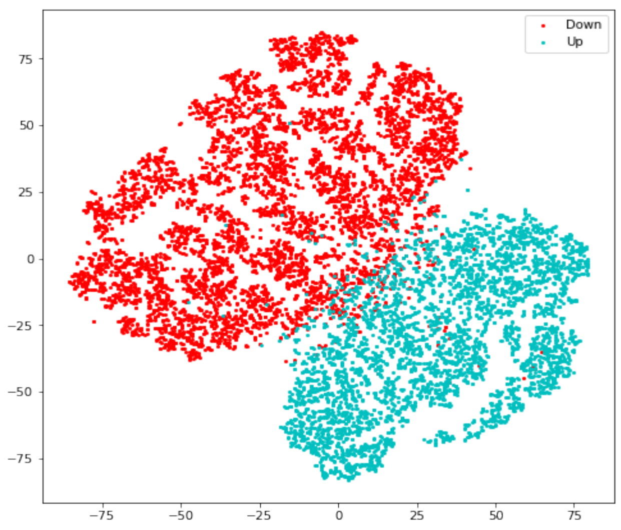
Figure 5: 5-class t-SNE

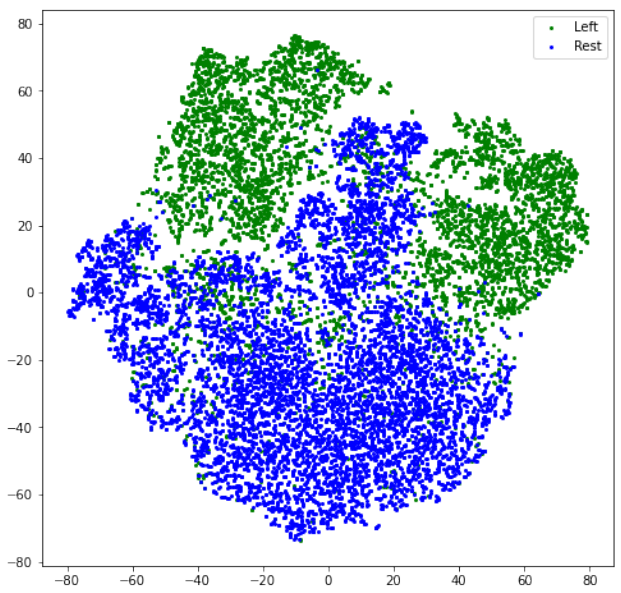
Although the PCA shows complete overlap, the t-SNE plot indicates a substantial divergence between the positions. However, there does exist some similar results between the down/left and the up/right positions; consequently, the data overlap is significantly responsible for the unusual training. The success of the algorithm indicates that it functions as developed, and the tendency of the algorithm to misclassify certain positions such as up and right can be attributed to the overlap. Even further, two-class t-SNEs were completed for the individual sets.

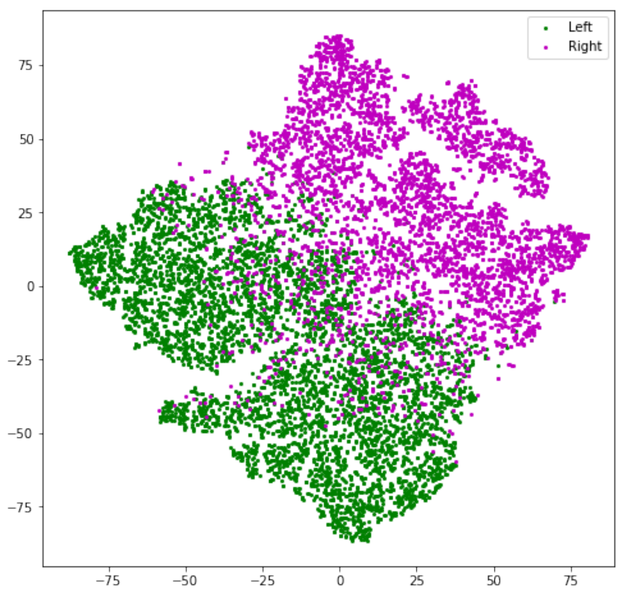

Down
Left

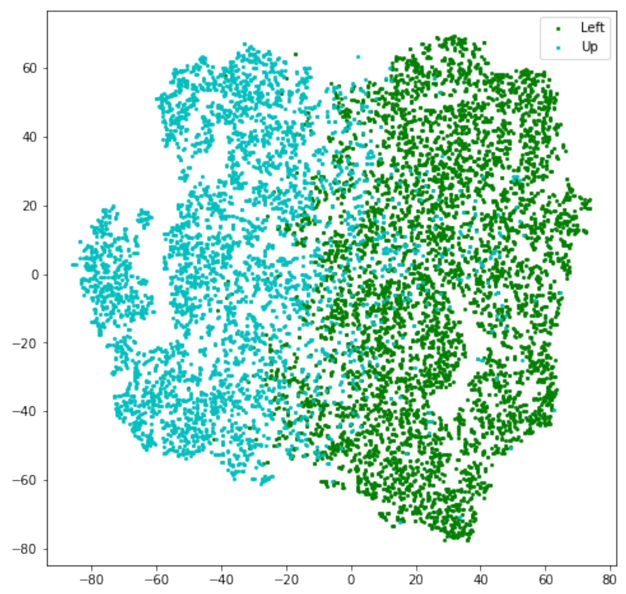

Down
Rest

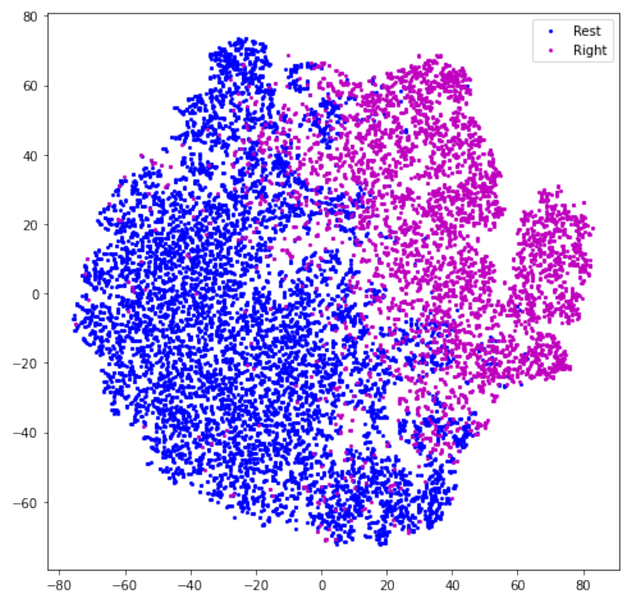

Down
Right

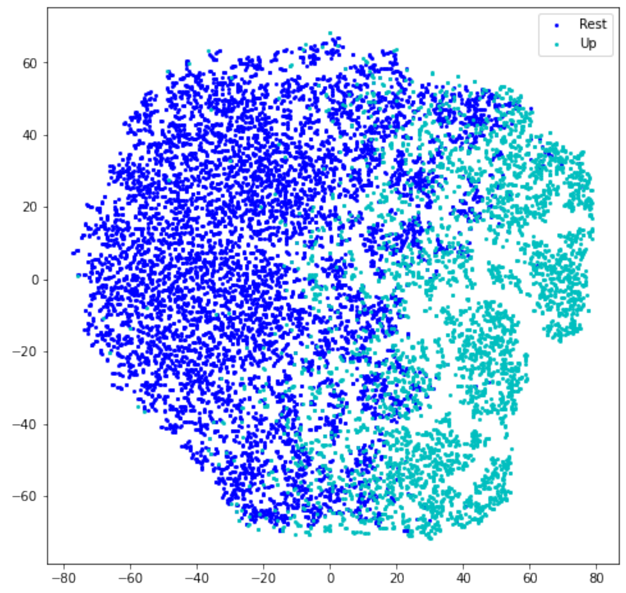

Down
Up

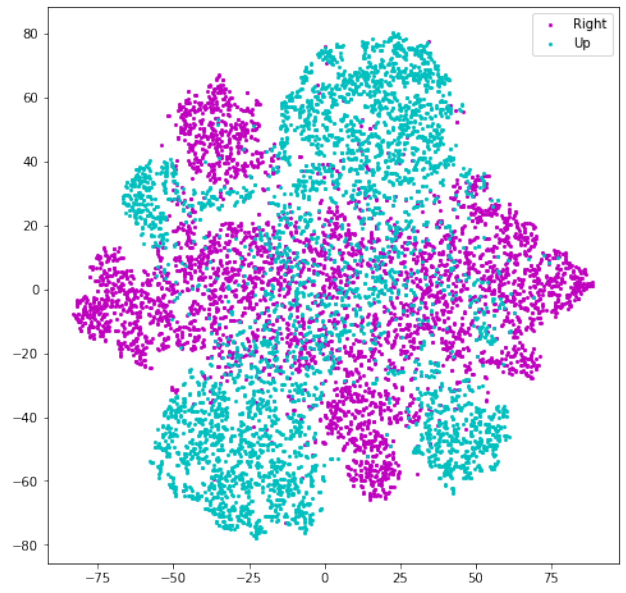

Left
Rest


Left
Right









Of all the plots, the down/left and especially up/right similarity resemble datasets that were easily confused by the algorithm and accounted for its inaccuracy. The rest set is remarkably separate and remains distinctly separate from other classes; the up/down differences also display an easily classifiable difference.

These results indicate that the first step in optimizing algorithm performance is choosing accurate training data. More of it does not necessarily lead to better results – a smaller, less noisy set will perform better in training for holistic generalization so that the model properly learns the data features. An alternate approach would be to address the flawed data in the training stage. Various noise reduction algorithms can efficiently be applied to the EMG streams to reduce the local fluctuations and decrease the amount of overfitting. Another option involves randomly dropping some of the trained network parameters so that the network avoids heavily relying on any one feature of a dataset (dropout layers – minimize overfitting). Further research should identify the impact of tuning hyperparameters such as learning rate and batch size, and retraining the algorithm using smaller networks and miniscule data sizes should be tested to resolve whether additional bugs persist beyond these adjustments and impact the algorithm itself.

Conclusion

The purpose of this research was to investigate the training process of machine learning model development in the context of telerobotics and muscle gesture classification for human-computer interfaces. From the exploration, it was determined that the most important step was resorting to carefully collected and discrete sets of data for optimal model performance. To verify adequate deviation between sets, higher-dimensional plotting algorithms such as t-SNE plots should be used to test overlap and collect new data if required. Also crucial is the use of normalization techniques and data smoothing to reduce noise and overfitting. From there, real-time classification can benefit from applications of noise suppression to augment the accuracy. More work is necessary to determine precisely the impact of parameter tuning and model architecture improvements. Logical extensions of this experiment would involve in-depth study of the learning curves, model layers, hyperparameters, algorithmic training / execution speed, and human interface feasibility, assuming proper data.

Bibliography

Diolaiti, N., & Melchiorri, C. (2002). Teleoperation of a mobile robot through haptic feedback. *IEEE International Workshop on Haptic, Audio and Visual Environments and Their Applications*, pp. 67-72. https://doi.org/10.1109/

Dominjon, L., Richir, S., Lecuyer, A., & Burkhardt, J. M. (2006). Haptic Hybrid Rotations: Overcoming Hardware Angular Limitations of Force-Feedback Devices. *IEEE Virtual Reality Conference (VR 2006)*, pp. 167-174. https://doi.org/10.1109/VR.2006.68

Juliani, A., Berges, V.-P., Vckay, E., Gao, Y., Henry, H., Mattar, M., & Lange, D. (2018, September 7). Unity: A General Platform for Intelligent Agents. Retrieved January 15, 2020, from arxiv.org website: https://arxiv.org/abs/1809.02627

Godard, C., Aodha, O. M., & Brostow, G. J. (2017). Unsupervised Monocular Depth Estimation with Left-Right Consistency. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6602-6611. https://doi.org/10.1109/CVPR.2017.699

Sheridan, T. B. (1989). Telerobotics. *Automatica*, *25*(4), 487-507. https://doi.org/10.1016/0005-1098(89)90093-9

Sledd, A., & O'Malley, M. K. (2006). Performance Enhancement of a Haptic Arm Exoskeleton. *2006 14th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pp. 375-381. https://doi.org/10.1109/HAPTIC.2006.1627127