Indian Institute of Technology Palakkad

IIT PALAKKAD

Summer Internship 2021

Report On

EMG based Classification of Basic Hand Movements using Neural Networks

Submitted by

Medapati Krupa Ananda Reddy

(Indian Institute of Engineering Science and Technology, Shibpur)

Supervisor:

Dr. Mahesh R Panicker

Electrical Engineering

September 18, 2021

Abstract

Electromyography (EMG) Signals, the biosignals collected from the skeletal muscles are widely being employed in medical applications for controlling assistive devices like Prosthetic/Orthotic devices and also in other applications of Human-Computer Interactions, Virtual Reality, etc. This project report explains the Classification of basic hand movements with the Single Channel Surface EMG Signals collected from a muscle on the right-hand forearm. Many research works were done to classify the EMG Signals in many different methods. The different methods include extracting features from the EMG Signal and feeding them into various Machine Learning algorithms like Support Vector Machines (SVM), etc, and also using various Deep Learning (a powerful subset of Machine Learning) Algorithms like Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN), Autoencoders, etc. In this project, a 1D CNN model has been employed to classify 6 different classes of Single-channel EMG Signals. Although the main research problem of the Internship project is to be able to deploy the Neural Network Model onto an Embedded System, i.e., to use the Neural Network as an Embedded Machine Learning (also called Tiny ML) application, due to the complexity in understanding the Embedded System's computing hardware in the context of Tiny ML and also due to some constraints with the EMG Sensor Electrodes, it could not be achieved. However, the Neural Network was trained to some extent in the context of Tiny ML. Many constraints were followed in making and training the Neural Network in the context of Tiny ML, and finally, the Neural Network Model has achieved nearly 67% accuracy.

Contents

1 Introduction

1.1 Electromyography (EMG) Signal

A biosignal is defined as any signal continually measured and monitored from a biological being. The biosignals, depending on the location of the body that produces and on the observed record, need different techniques for recording.

Electromyography (EMG) is a technique for evaluating and recording the electrical activity produced by skeletal muscles. The distribution of electricity is spread into the muscle when an action potential is at a muscle fiber. The signal is generated by the superposition of evoked action potentials of all active motor units in a muscle. EMG is performed using an instrument called an electromyograph to produce a record called an electromyogram.

So, an EMG Signal is a bioelectrical signal that measures the activity produced by skeletal muscles during their contraction representing neuromuscular activities[2].

1.2 Recording EMG Signal

There are two methods of recording the EMG Signal: Intramuscular EMG and Surface EMG.

(1) Intramuscular EMG assesses muscle function by recording muscle activity by inserting needle electrodes through the skin into the muscle tissue. Generally, these electrodes are divided into two categories:

(a) Monopolar Needle Electrode is a simplest needle electrode which is a fine wire inserted into a muscle with a surface electrode as a reference; or two fine wires inserted into muscle referenced to each other. Made of stainless steel, the monopolar needle electrode has a very finely sharpened point and is covered with Teflon or other insulating material over its entire length, except for a 0.5 mm exposure at the tip.

(a) Monopolar Needle Electrode

(b) Concentric Needle Electrode

Figure 1: Intramuscular EMG Electrodes

(b) Concentric Needle Electrode is a slightly more complex needle electrode in design. These needles have a fine wire, embedded in a layer of insulation that fills the barrel of a hypodermic needle, that has an exposed shaft called as Cannula, and the shaft serves as the reference electrode. The exposed tip of the fine wire serves as the active electrode.

(2) Surface EMG assesses muscle function by recording muscle activity from the surface above the muscle on the skin. These electrodes are divided into two categories:

(a) Passive Surface Electrodes are connected to an external amplifier circuit with the help of cables for proper signal acquisition. They consist of a metal disc, usually silver or chlorargyrite (Ag or AgCl), an adhesive disk, contain insulation everywhere except the point of contact with the skin and detect the average muscle activity on the surface. They can be disposable or reusable.

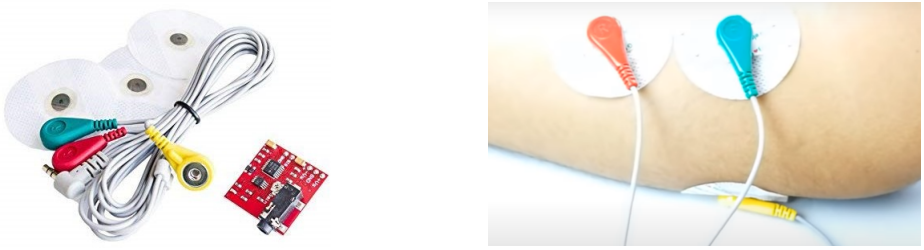


Figure 2: Passive Surface EMG Electrodes

(b) Active Surface Electrodes have attachment containing a pre-amplification for surface electrodes and are commonly referred as dry electrodes because no preparation of the skin surface is required in the area to be measured.

Figure 3: Two Channel Active Surface EMG Electrodes

Surface EMG has some limitations. Since these electrodes are applied on the skin, they are generally used for superficial muscles only. Crosstalk from other muscles is a major problem. Their position must be kept stable with the skin; otherwise, the signal is distorted[3].

1.3 Applications of EMG

For different Muscle Movements, we can analyse those respective EMG Signals and use them in many areas including medical applications like diagnosis of neuromuscular diseases, controlling assistive devices (i) for prosthesis when some portion of the limb of a person is missing, (ii) for orthesis to correct, rehabilitate or support parts of a person's body, (iii) for exoskeleton which works in parallel with the body to assist the user in their movements[4]. Other areas include Human Computer Interfaces (HCI) which helps to control and monitor the machines in Industries and Virtual Reality games, etc[5].

1.4 Classification of EMG Signals

So, Since EMG can be used in these many applications, research on 'Gesture Recognition' from those EMG Signals can be done in order to create a system that can identify specific human gestures and use them to convey information or for device control.

However, in order to identify the human gestures, Pattern Recognition of the EMG Signals is the most difficult part. This is because of large variations in EMG signals having different signatures depending on age, muscles activity, motor unit paths, skin-fat layer, and gesture style. Compared to other biosignals, EMG signal contains complicated types of noise that are caused by inherent equipment and environment noise, electromagnetic radiation, motion artifacts (alterations not related to EMG, caused due to the human shaking), and the interaction of different tissues[5].

Addressing all these issues, to classify the gestures, many research works have been done using many various techniques and obtained an acceptable classification performance. The various techniques include (i)Extracting the various features from the EMG Signal that uses parameters in time domain, frequency domain and time-frequency domain and then, classifying those extracted features using various Machine Learning Classification Algorithms. (ii)Using Deep Learning Classification Algorithms in which features are automatically extracted and are classified.

In this project, EMG Signals collected from right hand forearm are used to classify 6 basic hand movements. Since EMG Signals are Time Series Data, 1 Dimensional Convolutional Neural Networks (CNNs) are implemented to classify the EMG Signals.

The rest of the project report discusses the related existing work, Network Architecture, Dataset used, evaluation methods and the results.

2 Related Work

As mentioned before, existing research work classifies EMG Signals using many various techniques. In any technique, the general process is the Feature Extraction and Classification. The various techniques are:

Using Traditional ML Algorithms: Various Features are extracted from the EMG Signals in time domain, frequency domain, time-frequency domain. In the time-domain, there are common features such as Integrated Electromyogram (IEMG), zero-crossing, Willison amplitude, variance, skewness and kurtosis. The common frequency-domain features are the coefficients of short-time Fourier transform (STFT), the coefficients of wavelet transform (WT), and the coefficients of auto-regression (AR) of the spectrum of EMG signals.

All these extracted features are fed into an ML classifier such as Support Vector Machines (SVM), Linear Discriminant Analysis (LDA), Fuzzy Classifiers, etc.

Using Deep Learning Algorithms: Deep Learning is a powerful subfield of Machine Learning. Here, the signals are directly fed into a Deep Learning Neural Network such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Autoencoders, Deep Belief Networks (DBNs), etc. And the process of Automatic Feature Extraction and Classification is carried in the model.

Many research works suggest that "if the data used is large", Neural Networks outperforms the traditional Machine Learning algorithms. One of Deep Learning's main advantages over other Machine Learning algorithms is its capacity to execute feature engineering on its own. A deep learning algorithm will scan the data to search for features that correlate and combine them to enable faster learning without being explicitly told to do so[6].

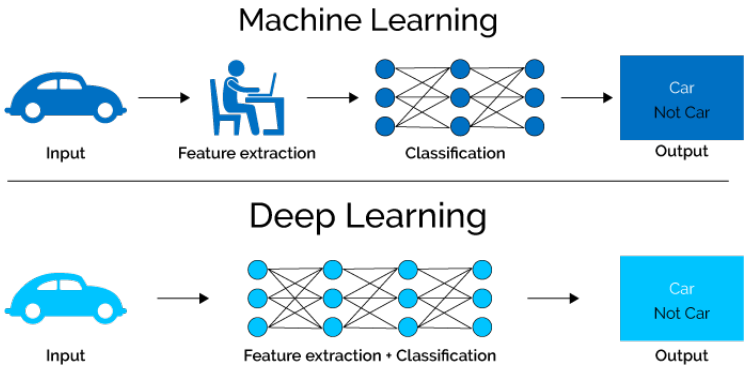


Figure 4: Showing the difference b/w ML and DL for a basic image recognition[7]

EMG Signals are time series data. Generally Long Short-Term Memory (LSTMs) networks, a type of RNNs are used for Time Series Data in Supervised Learning based training i.e., when 'labelled data' is provided for training an ML model. But it is observed that '1D CNNs' achieve better accuracy results than LSTMs and also take less time for training the model as compared to LSTMs which take a lot of time.

3 Approach

Since EMG Signal is a Time Series data, it is a 1 dimensional data. For the given no.of samples in a given time, the respective EMG value for each sample will be recorded. So, for classification of these signals, in this project, 1 Dimensional Convolutional Neural Networks (CNNs) are implemented.

Generally, CNNs are being extensively used in Image Classification problems to classify a 2 dimensional grayscale or RGB image. But CNNs can also be used for 1 Dimensional data.

The CNNs are feed forward neural networks basically consisting of an (i)Input layer, (ii)Hidden layers, (iii)Output layer. The hidden layers in CNNs mainly consist of 3 types of layers. They are:

1. Convolutional layer
2. Pooling layer
3. Fully-connected (FC) layer

The Convolutional layer can be followed by additional convolutional layers or pooling layers. The fully connected layer is the last hidden layer.

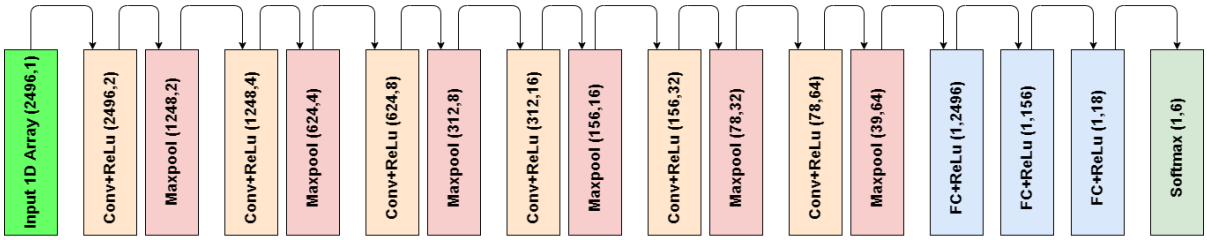


Figure 5: 1D CNN Architecture

Fig.5 shows the 1D CNN Architecture used in the project in which each main layer is discussed below and full architecture in Sec.4.3.

1. Convolutional layer: This layer is the core building block of the CNNs where the major computation occurs. The main purpose of the Convolutional layer is to detect different patterns or features from the input data. The input data is a 1 dimensional 'tensor' of shape 2496 x 1 in which 2496 is the no.of samples of each EMG Signal and 1 is the no.of channels.

The Convolutional layer contains a feature detector also known as 'kernel' or 'filter' which slides over the input data to detect the feature. This process is known as 'Convolution'. The kernel, a 1D array of weights which represents a part of the input 1D array is applied to the input 1D array and a dot product is calculated between the input values and the kernel. This dot product is then fed into an output array. Afterwards, the filter shifts by a stride, repeating the process until the kernel has swept across the entire input 1D array.

The final output from the series of dot products from the input and the filter is known as a 'feature map' or 'activation map'.[8]

Here, the weights in the Kernel remain fixed as it moves across the array. Some parameters, like the weight values, adjust during training through the process of back-propagation and gradient descent. However, there are 3 hyperparameters which affect the volume size of the output that need to be set before the training of the neural network begins. These include:

No. of filters: Applying multiple kernels on a single array will result in multiple channel of outputs or the feature maps. This process of leveraging different kernels allows the network to find different patterns in the signal thereby extracting more features.

Stride: It is the distance that the kernel moves over the input array. As the filter shifts by a stride, the process of convolution gets repeated until the kernel has swept across the entire input 1D array. So, it can be understood that a larger stride yields a smaller output feature map.

Padding: It is usually used when the filters do not fit the input array. This makes all elements that fall outside of the input array to zero, producing a larger or equally sized output. There are three types of padding:

(i). Valid padding: This is also known as no padding. In this case, the last convolution is dropped if dimensions do not align.

(ii). Same padding: This padding ensures that the output layer has the same size as the input layer.

(iii). Full padding: This type of padding increases the size of the output by adding zeros to the border of the input.

After each convolution operation, a CNN applies an activation function called 'Rectified Linear Unit' **(ReLU)** to the feature map. ReLu is a nonlinear activation function which goes through all outputs of the Convolutional layer, and wherever a negative number occurs, it swaps out a 0. The advantage of ReLu is to allow networks to learn faster and perform better.

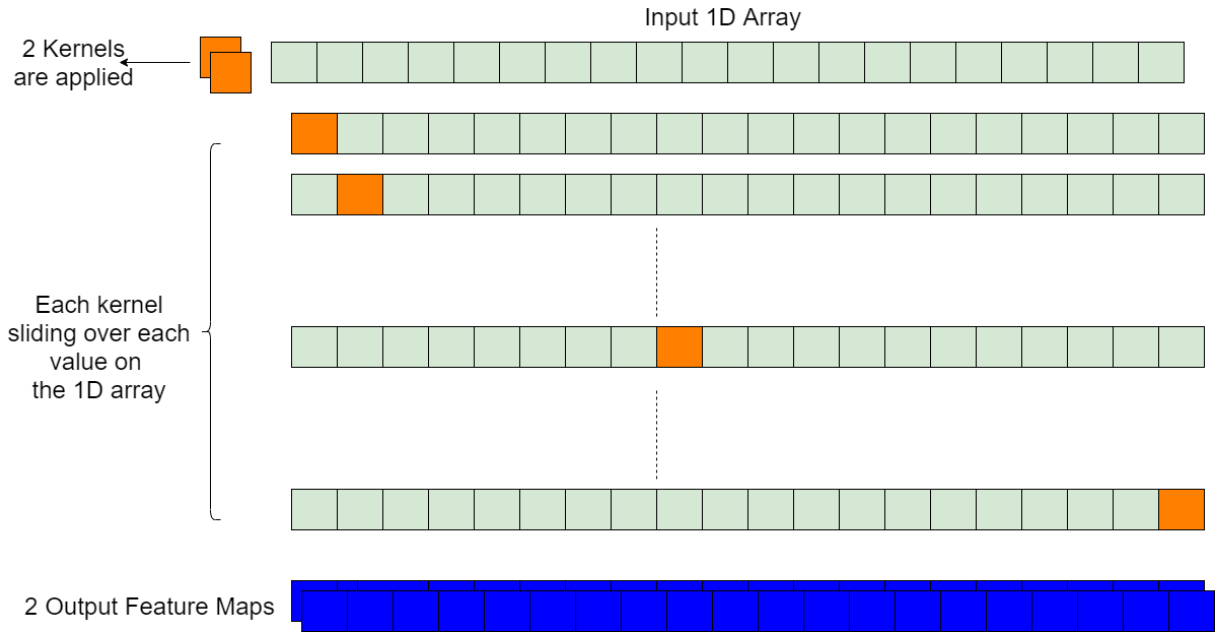


Figure 6: Explanation of Convolutional layer

2. Pooling layer: This layer down-samples the convolved output from the ReLu function, conducting 'Dimensionality Reduction'. In a sense, it turns the low level data into higher level information thereby reducing the no.of parameters of the input. Hence reducing the size of the model. This layer is also similar to Convolutional layer where the kernel is sweeped over the entire input 1D array but the difference is that this kernel does not have any weights. Instead, the kernel applies an aggregation function to the input values across the kernel. There are two types of Pooling layers:

(a). Max Pooling: As the kernel moves across the input array, it selects the the maximum value to send to the output array.

(b). Average Pooling: Here, the kernel selects and sends the average value to the output array.

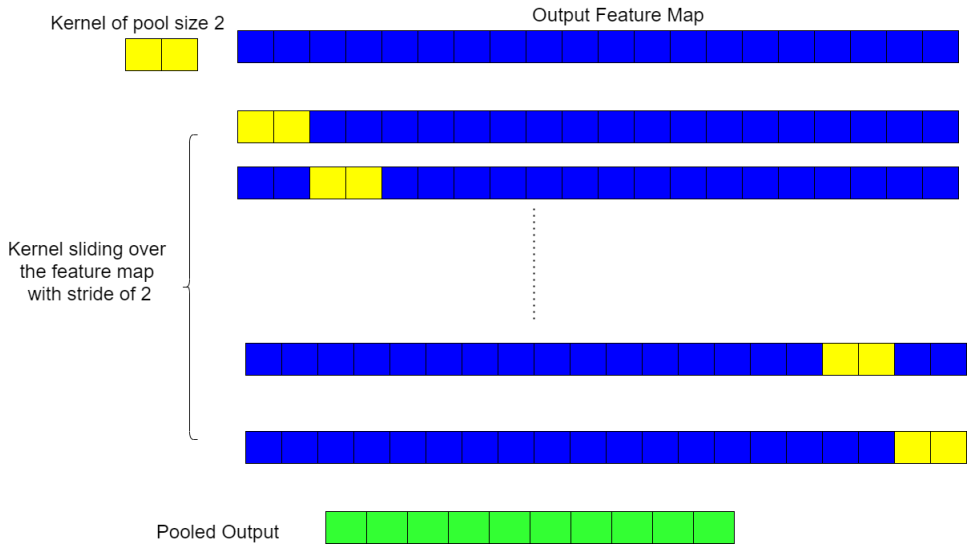


Figure 7: Explanation of Pooling layer

3. Fully-connected (FC) layer: This layer combines all the previous multiple channel outputs of 1D arrays into a single 1D vector. It is Fully-Connected layer because each node from the previous layer is connected to all the nodes of this layer.

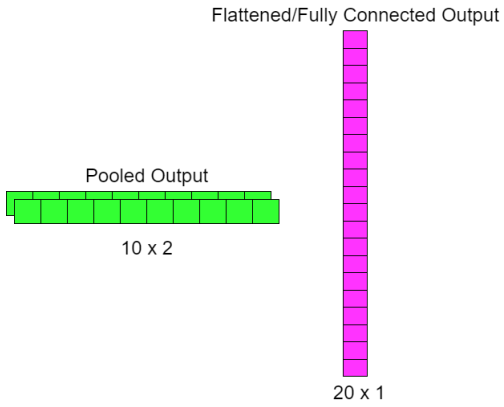


Figure 8: Explanation of Fully Connected layer

This layer performs the task of classification based on the features extracted through the previous layers and their different filters. FC layers also use ReLu and additionally leverage a **softmax** activation function to classify inputs according to its each class appropriately, producing a probability distribution from 0 to 1 for each class.

Dropout layer: These layers are used after the FC layers in order to avoid 'Overfitting', a problem causing the network to be more biased only to the training data. Dropout layers remove some random neurons from the layers without affecting the network's inference accuracy. Dropout rate represents the percentage of the neurons which are randomly removed.

4 Experiments

This section contains the following.

4.1 Data

The Data used in the project is from sEMG for Basic Hand movements Data Set[10] available at UC Irvine Machine Learning Repository[11]. This dataset includes 2 databases of Surface EMG Signals in which Database 2 is used for this project. This database consists of mat files of 2 Channels of sEMG signals collected from two muscles on right hand forearm 'Flexor Capri Ulnaris' and 'Extensor Capri Radialis' shown in Fig.9 from a single healthy subject (male, 22-year-old). In this project, the signals collected from the muscle 'Flexor Carpi Ulnaris' are used in training the network.

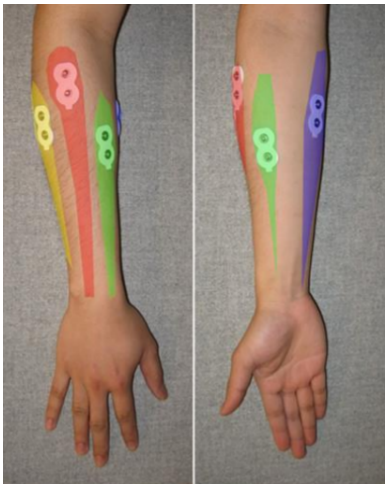


Figure 9: Flexor carpi ulnaris (red) and Extensor carpi radialis (blue) muscles[10]

The subject was asked to perform following 6 movements as shown in Fig.10 repeatedly which can be considered as daily hand grasps:

(a). Spherical: for holding spherical tools

(b). Tip: for holding small tools

(c). Palmar: for grasping with palm facing the object

(d). Lateral: for holding thin, flat objects

(e). Cylindrical: for holding cylindrical tools

(f). Hook: for supporting a heavy load

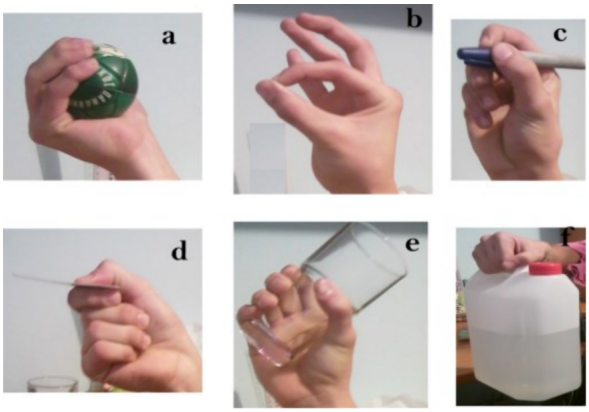


Figure 10: 6 Hand Grasps[12]

The subject performed the six grasps for 100 times each for 3 consecutive days. So, there are $6 * 100 * 3 = 1800$ signals.

Instrumentation: The data were collected using Active Surface EMG Electrodes at a sampling rate of 500 Hz, using as a programming kernel the National Instruments (NI) Labview for 5 sec. So, there will be $500Hz * 5sec = 2500$ samples. As written in the dataset information, the signals were band-pass filtered using a Butterworth Band Pass filter with low and high cutoff at 15Hz and 500Hz respectively and a notch filter at 50Hz to eliminate line interference artifacts.
The hardware that was used was an NI analog/digital conversion card NI USB- 009, mounted on a PC. The signal was taken from two Differential EMG Sensors and the signals were transmitted to a 2-channel EMG system by Delsys Bagnoliâ Handheld EMG Systems.

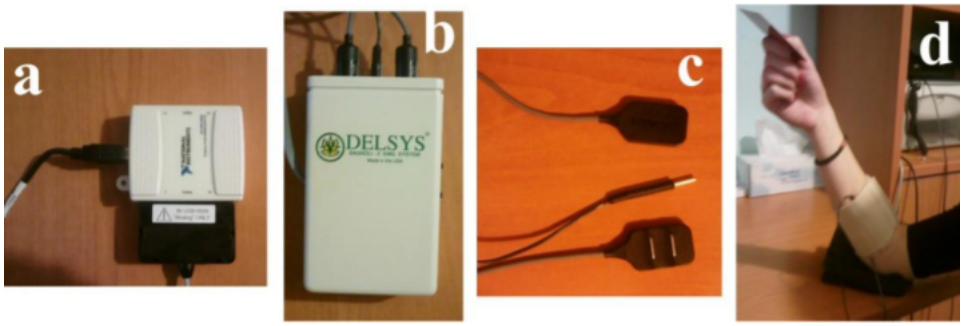


Figure 11: a) National Instruments analog/digital conversion card, b) 2-channel EMG system, c) 2 Differential and 1 reference EMG Sensor and d) the setup applied on a subject[12].

In order to remove the data when the muscle is inactive, within a sliding window of 40 msecs the average Integrated EMG value was calculated and a threshold was set. When the value of a sample of the signal exceeds this threshold, the signal needed is obtained.

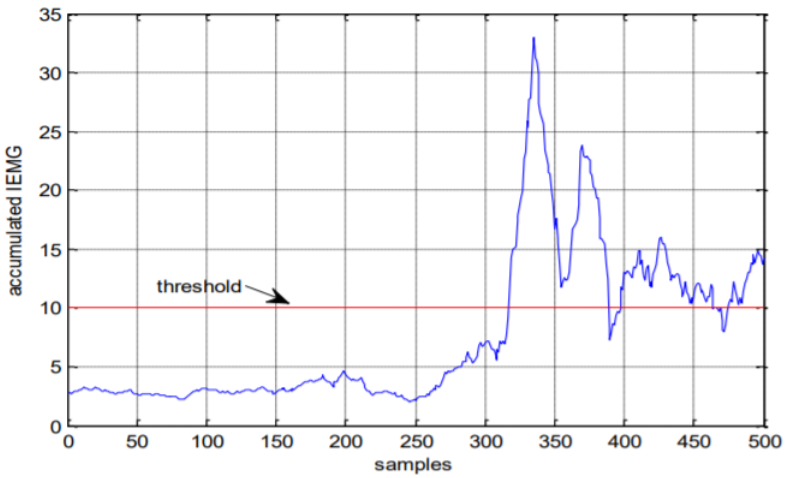


Figure 12: Illustration of evolution of the accumulated IEMG values moving from a resting phase to full muscle contraction, along with the predefined threshold; based on the threshold, the muscle is contracted at the 320th sample.[12]

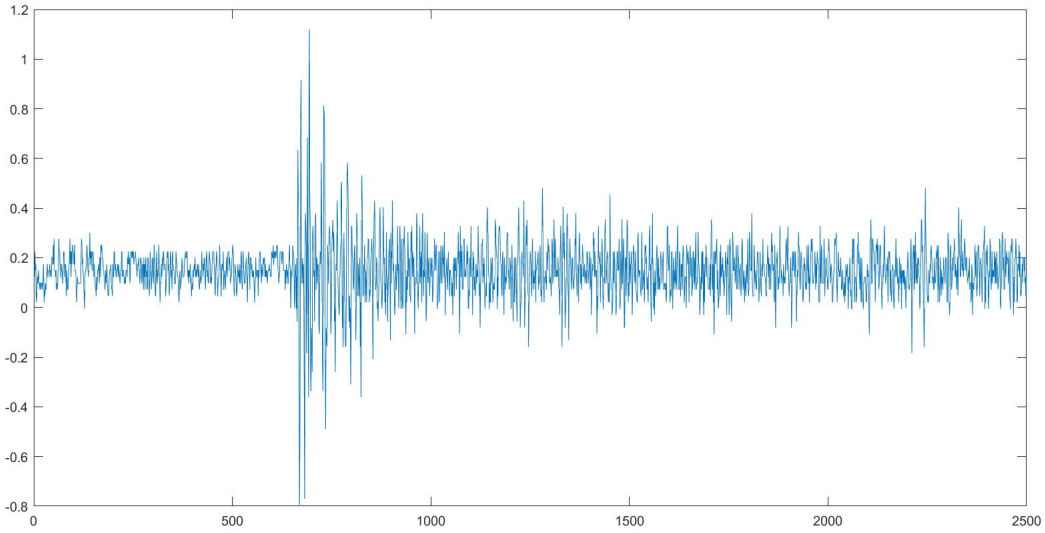


Figure 13: sEMG Signal of a hand grasp obtained after a series of preprocessing

Splitting of the data: So, the Dataset consists of 1800 preprocessed Single channel sEMG signals representing 6 classes of hand grasps. This data is to be used for training the Neural Network. However, all the data is not used for training the network. Some data is left behind. This is because, if all the data is used only for training the network, the network may perform with 100% accuracy for the classification of the data used in training. But if the network is tested with some other data not used in training, the network may not perform well. This problem is called 'Overfitting' i.e, the network only performs predominantly well on the training data. This is really a great problem because, in real time application, the network definitely has to perform on the data which it has not seen.

So, in order to avoid Overfitting, some data is left behind from training. That data can be used as Validation set and Test set. Validation set can be used to measure the accuracy while training and Test set can be used to measure the accuracy after the training. In this way, network can be trained accurately.

Generally, 60-70% of the data is used only for training and the remaining 30-40% is used for Validation and testing. But in this project, since EMG Signals have more variations in terms of age, skin-fat layer, muscle activity, etc., more data is used for testing the data. 40% of the data is used for testing, 45% of the data is used fot training, 15% of the data for Validation. Out of the 1800 signals, 720 signals are used for testing, 810 signals are used for training, 270 signals are used for validation.

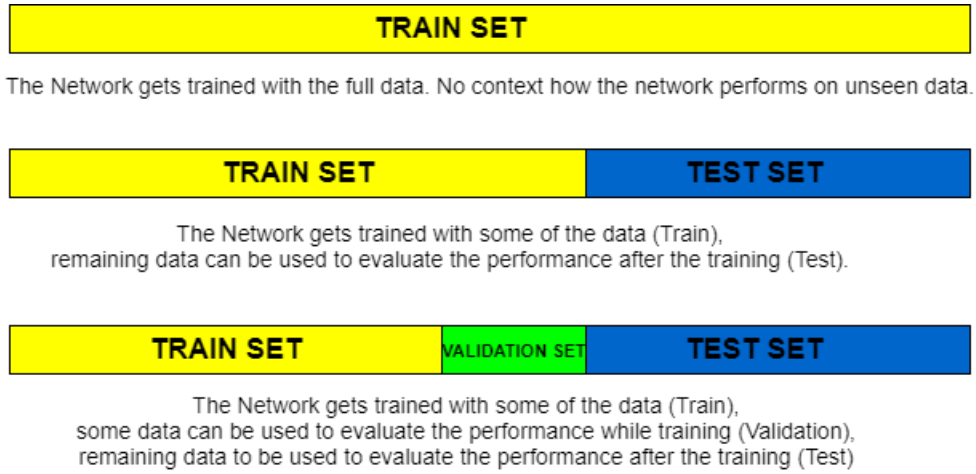


Figure 14: Train, Validation and Test sets

4.2 Evaluation method

Various metrics are necessary to be employed to evaluate a trained neural network on the test set. Metrics can be easily understood from the 'Confusion Matrix'. A confusion matrix is a tabular way of visualizing the performance of your prediction model. Each entry in a confusion matrix denotes the number of predictions made by the model where it classified the classes correctly or incorrectly[13].

A confusion matrix can be basically understood using a binary classification problem which has only two classes to classify, preferably a positive and a negative class.

True Positive (TP): It refers to the number of predictions where the classifier correctly predicts the positive class as positive.

True Negative (TN): It refers to the number of predictions where the classifier correctly predicts the negative class as negative.

False Positive (FP): It refers to the number of predictions where the classifier incorrectly predicts the negative class as positive.

False Negative (FN): It refers to the number of predictions where the classifier incorrectly predicts the positive class as negative.

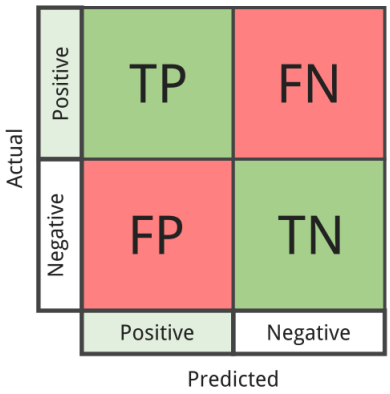


Figure 15: Confusion Matrix for Binary Classification

Based on this confusion matrix, the following metrics are explained and these are the metrics employed in the project.

Accuracy: This is the overall accuracy of the model, meaning the fraction of the total samples that were correctly classified by the classifier. Accuracy is calculated by the formula: $\frac{(TP+TN)}{(TP+TN+FP+FN)}$.

Precision: It tells what fraction of predictions as a positive class were actually positive. To calculate precision, use the following formula: $\frac{TP}{(TP+FP)}$.

Recall: It tells what fraction of all positive samples were correctly predicted as positive by the classifier. To calculate Recall, use the following formula: $\frac{TP}{(TP+FN)}$.

F1-score: It combines precision and recall into a single measure. Mathematically it's the harmonic mean of precision and recall. It can be calculated as follows: $2 \times \frac{Precision \times Recall}{Precision+Recall} = \frac{2TP}{2TP+FP+FN}$

So, as understandable from the above formulae, since achieving 100% accuracy is not possible, higher values of any metric evaluates the model to be more efficient. The same procedure can also be used for Multiclass Classification problems where each class has its own positives and negatives.

4.3 Experimental details

In this project, as shown in Fig.5 and also in Fig.16, EMG Signals representing 6 basic hand movements are fed into the neural network. In the initial layers, chain of Convolution, ReLu, Max Pool operations gets repeated. 6 convolutional + ReLu + Max-pool layers are followed by 3 Fully-Connected + ReLu layers which is in turn followed by softmax in the last layer.

The conv + ReLu + Maxpool layers are used for Feature Extraction and FC + ReLu + Softmax layers are used for Classification.

Noting the effect of each single Conv, ReLu and Max-Pool operations passing through the input 1D array, it can be observed that the size of the individual 1D array gets halfed, and the depth of the no.of outputs or the feature maps gets doubled.

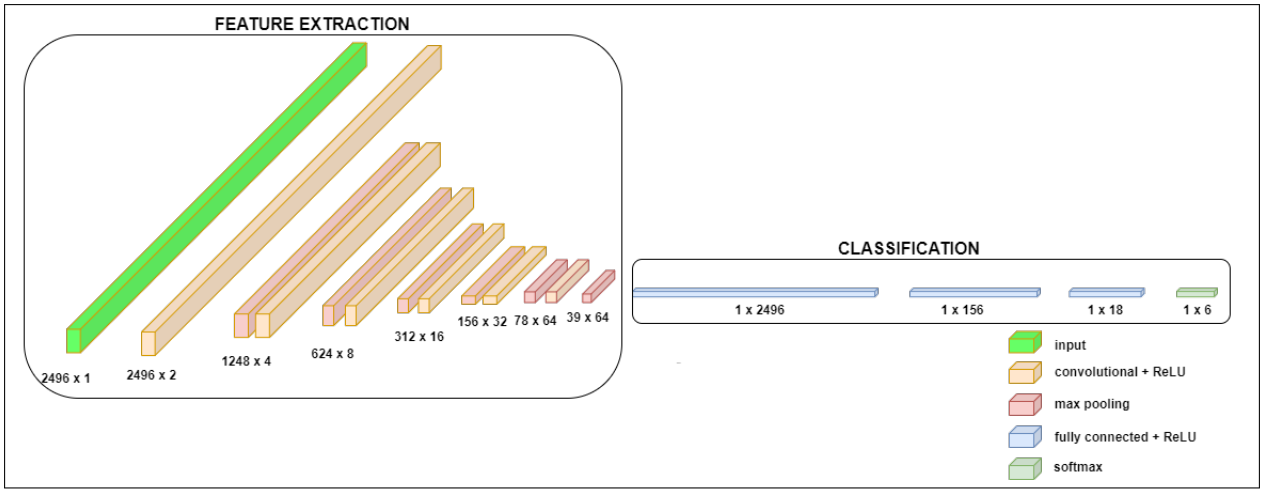


Figure 16: 1D CNN - Feature Extraction and Classification

For example, the output of the first convolutional layer is an array of size 2496 and depth 2, and the output of the second layer is an array of size 1248 and depth 4. It continues till the array of size 39 and depth 64. This is very important, because using multiple layers, CNN will be able to break down the complex patterns into a series of simpler patterns.

So, the network architecture is designed such that if more layers are used, more complex patterns are broke down into simpler patterns. This is the reason even though the actual no.of samples used for each EMG Signal in the dataset is 2500, in order to use more layers, last four samples are deleted and 2496 samples are used.

To achieve this, throughout all the convolutional layers, kernel of size 1, stride 1, Same padding are used. For the Conv1, 2 no.of such kernels are used and they go on doubling in the subsequent Conv layers in order to break into more simpler patterns.

And throughout all the maxpool layers, pool size of 2, strides 2 and Same padding are used. Pool size of 2 makes the array size to half after each convolutional layer.

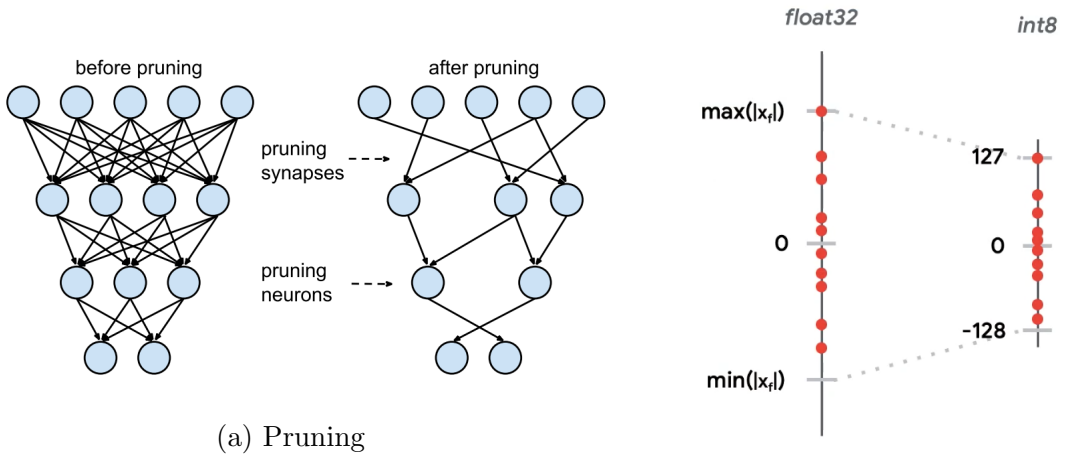
Now, the output array (39,64) gets flattened and becomes a 1D vector of length 39 x 64 = 2496. Again it is fed into an FC+ReLu layer of length 156, and again 18 and finally, Softmax is used in the last layer to classify 6 classes of hand movements.

Hyperparameters: This whole 1D CNN is trained using the following hyperparameters: Constant Learning Rate: 0.00001. Number of Epochs: 2500. Batch Size in Training and Inference: 100. This low learning rate of 0.00001 is used because when trained even with a rate slight higher than this, there is a potential rise in the training accuracy resulting in Overfitting.

In the context of Tiny ML: The main intention of the neural network is to deploy it onto an Embedded System, i.e, to use it as an Embedded ML (or Tiny ML) model. This model with these many layers has a large size which cannot be fit into an Embedded

System, so the size of the model has to be decreased. So, techniques like 'Pruning' and 'Quantization' helps to achieve the task.

Pruning: It is one of the methods for inference to efficiently produce models smaller in size, more memory-efficient, more power-efficient and faster at inference with minimal loss in accuracy.[14]



(a) Pruning

(b) Quantization

Figure 17

As seen in the fig.17a, Pruning removes some connections and neurons which have very less contribution to the network which results in a smaller and faster network. However, there may be small reduction in the accuracy of the model. Depending on the application, pruning can be done in many ways. In this project pruning is done for the whole model to reduce the size by nearly 3 times.

Quantization: It is an optimization that works by reducing the precision of the numbers used to represent a model's parameters, which by default are 32-bit floating point numbers. This results in a smaller model size, better portability and faster computation.

Reducing the precision means that the values expressed in Float 32-bits which takes 4 bytes are compressed down into an int 8 which takes only 1 byte. So, there is 4x reduction in the size as shown in the Fig.17b.

There are basically two ways of using Quantization: 1.Post Training Quantization (PTQ), 2.Quantization Aware Training (QAT). As the names suggest, PTQ is used after training the model and QAT is used while training the model. In this project, PTQ is employed to reduce the size of the model. This technique is used when converting the trained model into a model which is used for inference on small systems like Mobile phones, Embedded Systems, etc.

So, while using Pruning while training the model and PTQ after training the model, the model needed for inference has approximately 10x reduction in the model size from 1.3 MB to 134 KB.

4.4 Results

After training the model for 2500 epochs, the network has obtained an accuracy of 61% on the training set and an accuracy of 58% on the validation set. For the Test set, the network has performed with an accuracy of 68%.

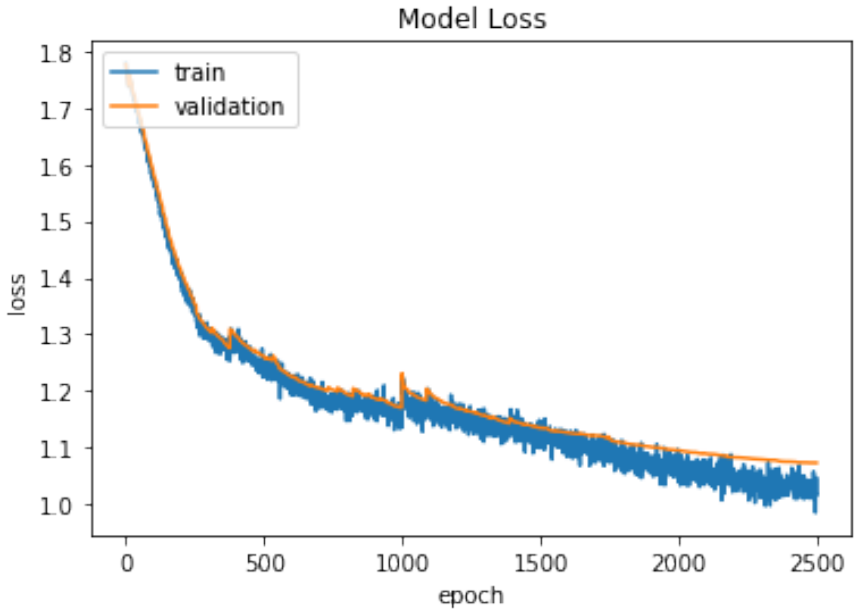


Figure 18: Train and Validation losses plotted against the epochs

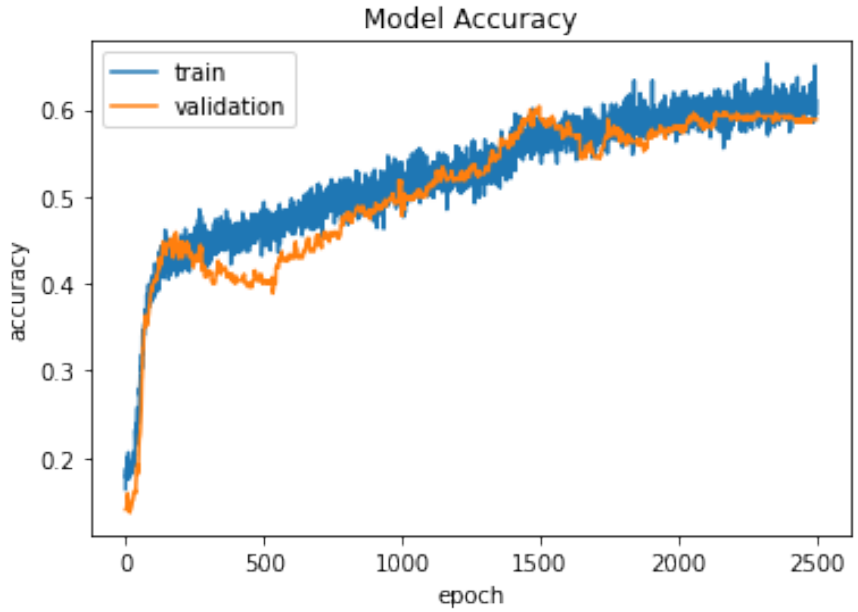


Figure 19: Train and Validation accuracies plotted against the epochs

The Confusion Matrix on the test set is shown in Table.1 and the metric evaluation for each class is shown in Table.2.

Table 1: **Confusion Matrix**

		Predicted Class					
		Cylindrical	***Hook***	***Lateral***	***Palmer***	***Spherical***	***Tip***
True Class	***Cylindrical***	**51**	12	13	13	16	8
	Hook	15	**106**	1	6	0	8
	Lateral	18	4	**52**	21	5	14
	Palmer	12	3	10	**56**	2	25
	Spherical	5	0	1	0	**131**	2
	Tip	3	2	3	10	0	**92**

Table 2: **Classification Report**

	Precision	**Recall**	**F1-score**
Cylindrical	0.49	0.45	0.47
Hook	0.83	0.78	0.81
Lateral	0.65	0.46	0.54
Palmer	0.53	0.52	0.52
Spherical	0.85	0.94	0.89
Tip	0.62	0.84	0.71

So, comparing the accuracies of Train, Validation, Test sets, it is clear that the network has not gone into overfitting. Infact, the network performs well for test set than the train set. This is due to the consideration of taking large amount of data for testing. It is observed that the accuracy of the network on the test set without Pruning the network is varying between 70-75% whereas when Pruning is introduced in the network, the accuracy drops to 60-70%. This is expected, because since Pruning compresses the model, there will be a drop in accuracy.

5 Analysis

From the Classification Report, it can be analysed that, for the sEMG signals collected from the muscle 'Flexor Carpi Ulnaris', the Neural Network performs very well in classifying the hand grasps holding Spherical tools and supporting heavy loads (hook). And also, performs fairly well for holding small tools (tip). But there is large drop in the performance of the network for holding thin/flat objects (lateral), for grasping with palm facing the object (palmer). And the least performance for holding Cylindrical tools.

The poor performance of the network on the last 3 classes discussed above shows that the muscle 'Flexor Carpi Ulnaris' has low contribution to these 3 hand grasps. So, collecting signals using more channels from various muscles may help in classifying these signals more accurately.

And also, due to the limitation of the Surface EMG electrodes, as discussed in Sec.1.2, crosstalk from other muscles on the superficial muscles causes problem in understanding the patterns of the signals.

Moreover, Generally, Neural Networks perform exceptionally well when they are fed with very large amount of data. But the dataset being used has only 1800 signals. So, using more number of EMG Signals may give good overall accuracy. Although there are some large EMG Signal datasets, they are multiple channel based datasets. Single channel EMG datasets are rarely found. However, collecting large amount of data of sEMG signals is also not easy as anyother application since it requires to go through many complex stages.

6 Conclusion

Apart from using CNNs for Image classification problems, CNNs can also be used for Time Series Classification problems. Especially, since EMG Signals have really complex patterns, leveraging the powerful CNNs on these biosignals can identify the patterns efficiently. In order to build an efficient CNN architecture, it does need a good understanding of EMG, its measurements and preprocessing. The no.of channels being used to collect the EMG Signals, the type of EMG electrodes being employed are very important for an efficient architecture. In order to use the architecture for a Tiny ML application, the architecture needs to be built understanding the requirements and constraints of the Embedded System.

In this project, the EMG Signals collected from the Active Surface EMG electrodes are classified with an acceptable accuracy using the 1D CNNs and the network is evaluated with respect to each class. However, the accuracy of the model can be improved by improving the network architecture, using large data, using multiple channel signals.

The Model can also be potentially used as a Tiny ML application, but it requires a considerable understanding of the Embedded System's computing hardware, its preprocessing and the postprocessing stages.

References

[1] Wikipedia Contributors. Biosignal. `https://en.wikipedia.org/wiki/Biosignal`.

[2] Wikipedia Contributors. Electromyography. `https://en.wikipedia.org/wiki/Electromyography`.

[3] Muhammad Zahak Jamal. Signal acquisition using surface emg and circuit design considerations for robotic prosthesis. *Computational Intelligence in Electromyography Analysis-A Perspective on Current Applications and Future Challenges*, 18:427–448, 2012.

[4] Alex Hill. Exoskeleton, orthosis, prosthetic... so what's the difference? *LinkedIn*, 2016.

[5] Md Rezwanul Ahsan, Muhammad Ibn Ibrahimy, and Othman O Khalifa. Electromygraphy (emg) signal based hand gesture recognition using artificial neural network (ann). In *2011 4th international conference on mechatronics (ICOM)*, pages 1–6. IEEE, 2011.

[6] Valeryia Shchutskaya. Deep learning: Strengths and challenges. *InData Labs*, 2018.

[7] Sambit Mahapatra. Why deep learning over traditional machine learning? *Towards Data Science*, 2018.

[8] IBM Cloud Education. Convolutional neural networks. *IBM Cloud Learn Hub*, 2020.

[9] Jason Brownlee. 1d convolutional neural network models for human activity recognition. *Machine Learning Mastery*, 2018.

[10] Christos Sapsanis. Recognition of basic hand movements using electromyography. Master's thesis, University of Patras, 2013.

[11] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[12] Christos Sapsanis, George Georgoulas, and Anthony Tzes. Emg based classification of basic hand movements based on time-frequency features. In *21st Mediterranean conference on control and automation*, pages 716–722. IEEE, 2013.

[13] Joydwip Mohajon. Confusion matrix for your multi-class machine learning model. *Towards Data Science*, 2020.

[14] Ranjeet Singh. Pruning deep neural networks. *Towards Data Science*, 2019.