2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)
Bari, Italy. October 6-9, 2019

Parkinson's disease EMG signal prediction using Neural Networks

Rafael Anicet Zanini[1], Esther Luna Colombini[1] and Maria Claudia Ferrari de Castro[2]

Abstract— **This paper proposes a comparison between different neural network models, using multilayer perceptron (MLPs) and recurrent neural network (RNN) models, for predicting Parkinson's disease electromyography (EMG) signals, to anticipate resulting resting tremor patterns. The experimental results indicate that the proposed models can adapt to different frequencies and amplitudes of tremor, and provide reasonable predictions for both EMG envelopes and EMG raw signals. Therefore, one could use these models as input for a control strategy for functional electrical stimulation (FES) devices used on tremor suppression, by dynamically predicting and improving FES control parameters based on tremor forecast.**

I. INTRODUCTION

Parkinsons Disease (PD) is a neurodegenerative disease that affects approximately 2% of world's population over 60 years old (7-10 million people), being the second most common age-related neurodegenerative disorder after Alzheimer's disease [1]. It is characterized by the degeneration of dopamine production neurons found at the deepest part of the encephalum called substantia nigra, with causes still unknown [2].

The most characteristic symptoms are resting and action tremor, bradykinesia (slowness of movement), postural instability (unstable and prone to falls) and rigidity (stiffness), which may also cause non-motor symptoms such as difficulty to sleep, depression, and fatigue [3].

In this context, assisting technology that can diminish all kinds of tremor impact and burden for patients has a crucial role, allowing patients to live without all impairments imposed by PD. Furthermore, minimizing the effects of tremor on patients for multiple tasks is a huge benefit that one can address with different techniques, such as functional electrical stimulation (FES), which usually rely on analyzing movement and tremor signals based on electromyography (EMG) [4].

Functional electrical stimulation (FES) has been proposed by many authors [5] as a potential method for reducing and controlling the pathological tremor. However, the feasibility and accuracy of FES depend heavily on the estimation of amplitude and frequency of the involuntary tremor signals, and the side-effect of FES on patient's voluntary movements. In most cases, these measurements are based on surface electromyography (sEMG) for measuring muscle activation, and accelerometers for measuring the resulting tremor on the limbs.

[1]Rafael Anicet Zanini and Esther Luna Colombini are with Laboratory of Robotics and Cognitive Science (LaRoCS), UNICAMP, Campinas, Brazil rzanini@gmail.com, esther@ic.unicamp.br

[2]Maria Claudia Ferrari de Castro with the Department of Computer Sciente, Fundao Educacional Inaciana Padre Sabia de Medeiros (FEI), Sao Bernardo do Campo, Brazil mclaudia@fei.edu.br

To effectively use EMG as input, one has to use different filtering and pre-processing techniques to be able to correctly identify movement and tremor patterns main characteristics, such as base frequency, amplitude, mean value, which are commonly used as parameters for FES control [6] [7] [8] [9].

Recent work shows that traditional FES control strategies typically include: open loop, feed-forward, and closed loop [10]. There are different strategies for achieving a closed loop, like Proportional Integral Derivative Control (PID), Fuzzy Logic Control (FLC) and Model-based Control. However, Souza et al. [10] suggest that these traditional controllers performance is still limited since they cannot optimally address some critical challenges on PD tremor suppression, such as:

- Tremor pattern, amplitude, and frequency varies from patient to patient;
- Tremor magnitude and frequencies vary during the day according to levels of levodopa and other medications consumed by the patient, and
- Tremor and muscle-stimulus vary according to movement patterns.

Recent work [5] has demonstrated the feasibility of neural oscillator based control, using a mixed feed-forward and feedback control approach that could provide a faster response when compared to traditional feedback control, with promising results.

However, control variables such as the frequency gains, feedback gain, stimulus width, and amplitude and other parameters are all fixed to experimental values. Hence, any disturbances could make these parameters work outside the optimized area, deteriorating the performance of the tremor suppression. Also, adapting the control parameters from one patient to another requires a new configuration set-up and fine-tuning for individual characteristics.

One of the main reasons for this is because most of FES controllers only use past data to adjust control parameters. So the generated stimulus is always related to the last tremors, most times sub-optimal to the real-time tremor signal. Therefore, being able to predict tremor signals based on specific patient tremor pattern effectively can provide great value for improving FES control, and make it automatically adaptable for different patients and different tremor conditions.

Recent strategies for simulating or classifying EMG signals include Neural Networks (NN) that can learn EMG patterns and simulate a similar signal. According to [11], it is possible to combine RNNs based on Long short-term memory (LSTM) and multilayer perceptron (MLP) networks

978-1-7281-4569-3/19/$31.00 ©2019 IEEE

to recognize patterns and classify different movements efficiently. The LSTM network captures temporal dependencies of the sEMG signals, while the MLP focuses on the static and amplitude characteristics.

According to [12], it is possible to use other types of RNN such as gated recurring units (GRUs) to create a model that can predict 1 point in the future, which can then be used to generate a closed-loop model that mimics EMG patterns based on a sliding window that adds the predicted point to the input data set. Although very successful, the work does not provide a study for PD.

In this work, we propose to evaluate and compare different neural network architectures, using MLPs, LSTM, and combined models to predict PD's sEMG signals. Our primary target is to be able to successfully predict a full window of the tremor pattern (approximately 0.2s) so that we can use it later for FES control optimization. With such a prediction model, we can also create a Parkison EMG simulator, which can assist further research and studies on Parkinson's disease.

We obtained the used private data set from surface electromyography (sEMG) and accelerometer data acquired from real PD's patients on previous work [13]. There are 18 different record sets available, each one with more than 116,000 points per data set, acquired from multiple sessions from 5 patients, 1 patient diagnosed with Essential tremor (ET) and the remaining 4 diagnosed with PD according to UK Parkinsons Disease Society Brain Bank Clinical Diagnostic Criteria [14]. All acquisition procedures were previously submitted and approved by the Plataforma Brasil ethical committee, and the patient selection was performed by neurologists from the Federal University of Sao Paulo (UNIFESP).

II. METHODS

Multi-layer perceptron (MLP) is a computational model that processes information through a series of interconnected computational nodes. A limitation of the MLP architecture is that it assumes that all inputs and outputs are independent of each other, which for predicting time-series is not desirable, as each point depends directly from the previous points in time. For an MLP to model a time series (such as a sensor signal), it is necessary to include some temporal information in the input data. Recurrent neural networks (RNNs) are neural networks specifically designed to tackle this problem, making use of a recurrent connection in every unit, allowing the neurons' connections to capture temporal relationships.

Like any Recurrent Neural Network (RNN) neurons, long short-term memory (LSTM) neurons keep a context of memory within their pipeline [15] to allow for tackling sequential and temporal problems without the issue of the vanishing gradient affecting their performance. They have additional states that control which memories must be copied or deleted from the neuron, combining the transformed input with the predecessor (x_{t-1}) and successor (x_{t+1}). Multi-layer LSTM networks have been successfully used and tested for Sequence to Sequence problems [16], such as translation and sequence prediction [17].

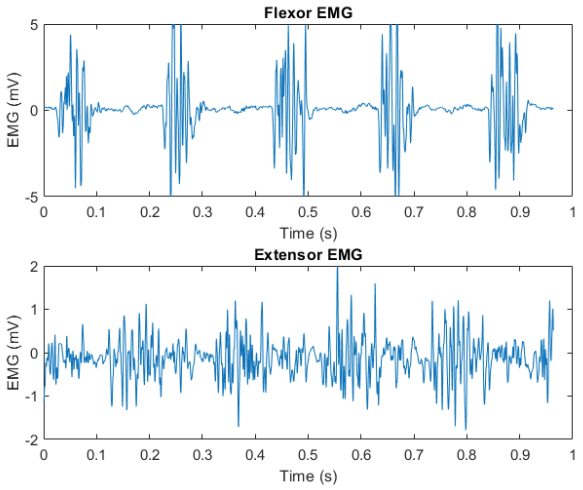

Fig. 1. Flexor (a) and Extensor (b) EMG signals, acquired with surface EMG sensors attached to patient's forearm.

Another promising and new technique called Residual Networks (ResNet)[18] has also been proposed to solve the vanishing gradient problem. Researchers have noticed that reintroducing the original inputs together with layer intermediate outputs could effectively reduce the vanishing problem, making deep networks easier to optimize and with higher accuracy. This can be applied with any network topology but is especially useful for deep MLP networks.

In this work, we present the proposed LSTM architectures for EMG signal prediction. We approached and evaluated different combinations, focusing on the following approaches:

- MLP networks
- LSTM networks
- Combined LSTM and MLP networks
- MLP Autoencoders
- LSTM Autoencoders

A. Data acquisition and pre-processing

The data collection and pre-processing methods follow a similar approach as proposed by [13], where EMG signals from wrist extensor and flexor muscles were acquired with a 2kHz DELSYS Trigno EMG system. We then applied a low pass filter with a cut-off frequency of 500Hz for higher frequencies noise attenuation. A notch filter was applied to attenuate 60Hz component from the power line, and analog EMG was converted into digital format through a 12-bit A/D converter.

The collected data was provided into 60-second window experiments (approximately 120,000 data points per experiment) in multiple sessions for different patients. Figure 1 shows the raw input data acquired for each analysis. The flexor and extensor EMG signals were acquired and analyzed during experiments, but for comparison reasons, the results only show 1 feature at a time for each trained model.

For this work, we used two different representations of the EMG signals to evaluate the proposed methods.

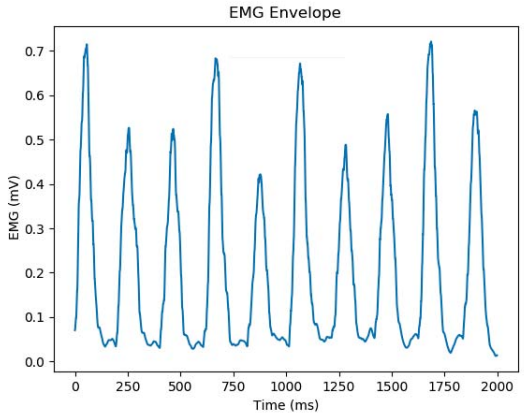

Fig. 2. Flexor EMG data after a rectification and smooth transformation, generating the EMG envelopes according to tremor pattern.

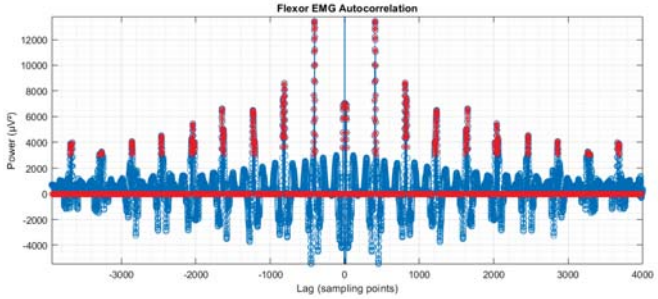

Fig. 3. Flexor EMG signal autocorrelation: as we can see, at every 400 points there is a peak on the correlation. The far we move in time, the lower the correlation.

The first one is based on the EMG envelopes, which we obtained from the rectified EMG signal, which is then smoothed with a 100-point moving average filter (equivalent to applying an FIR filter with impulse response consisting of 100 equal value samples). All signals were then normalized to have a common scale between different experiments, without distorting differences in the ranges of values for different patients and experiments. Figure 2 shows how the original EMG signal is rectified and then smoothed and normalized to obtain the EMG envelopes. Predicting this signal allows FES control devices to anticipate the behavior of the tremor, providing rich information about the base tremor frequencies and amplitudes.

The second one consists of a smoothed version of the raw EMG signal (10-point moving average filter), which provides much more information about the electrical stimulation on muscles. All signals were re-scaled between -1.0 and 1.0 to have a common scale between different experiments and samples. The higher the variation of the signal, the higher is the difficulty to create a model that can successfully predict it. However, being able to recognize and generate such behavior successfully is also desired, to develop an EMG tremor simulator to train FES control devices.

B. Time-series prediction

Predicting Parkinson's Disease EMG signals based on patient's readings as input is a typical time series sequence prediction problem. Since PD's tremor typical frequency varies between 4-6Hz, each tremor stimulus happens on a 0.2-second window. Considering a sampling rate of 2kHz, we need to cover at least 400 points in the future to be able to predict when is the next tremor stimulus happening. Also, the further we move towards the past EMG signals, the lower is the correlation to the future tremor pattern. By analyzing the autocorrelation of EMG tremor signals (Figure 3), it is possible to notice that tremor patterns repeat on the expected frequency range, reaching peaks every 400 points (equivalent to 200ms in time). The far we move in time, the lower is the correlation. So for this study, we fixed the input window and predicted window sizes, by predicting the next 400 points based on the the last 4,000 points.

In this paper, training was performed on 80% of the data set, grouped on windows of 4,000 points and then randomly sampled. Different window sizes were also evaluated, showing that the best configuration for a 2kHz acquisition rate is a window with 4,000 points, representing a period of 2s. It is also possible to use smaller periods of input time, however, the resulting prediction error increases considerably. The test was performed in the 20% remaining data, while validation during training was performed over 10% of the training data set.

The batch size determines how many samples each training batch will use. After evaluating multiple values, the training batch was fixed on 400 window samples.

Multiple topologies were tested, with a different number of LSTM layers (1 to 8), dense output layers (1 to 6) and a number of LSTM neurons (20 to 200). Different activation functions were used according to the input data set (ReLu, tanh, sigmoid).

For the EMG envelopes, the activation function preferred was ReLU. For the raw EMG signal, the activation function used was tanh, as it works best with negative and positive non-linear data. The selected loss function was the mean squared error (MSE) and ADAM [19] the selected optimizer.

C. Proposed architectures

1) Multilayer perceptron (MLP): The first architecture used was a simple multilayer perceptron network (MLP), combining variable hidden units to generate the predicted output. The input and output signals had only one feature (either the Flexor or Extensor EMG signal), predicting 400 timesteps in the future based on 4,000 steps from the past. Figure 4(a) shows the representation of such architecture.

2) Multilayer LSTM: The second tested architecture was a multilayer LSTM network, combining variable hidden and recurrent units to a single dense layer for the predicted output. The input and output signals had also only one feature, predicting 400 steps in the future based on 4,000 previous steps. Figure 4(b) shows the representation of such architecture.

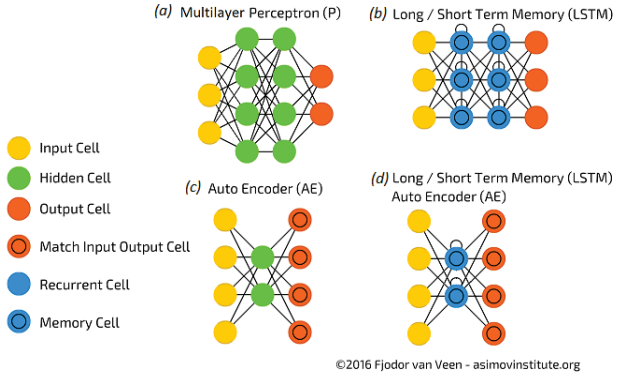

Fig. 4. Proposed architectures: (a) Multilayer Perceptron (MLP) (b) LSTM (c) MLP Autoencoder (d) LSTM Autoencoder. Adapted from [20]

3) MLP Autoencoder: We often use autoencoders for reducing the dimensionality of input signals, trying to identify a more compact set of features that adequately describes the signal and its temporal relations. With this approach, two models have been trained: the encoder, which reduces the dimension of the input vector to a more compact representation, and the decoder, that expands the signal back to its original representation. Once we train the model, the encoder can be used as input for other neural networks (NN) models, providing a more compact representation without so much loss of information.

Two different encoders were trained in this paper: one for the representation of the EMG envelopes and another for learning the compact representation of a raw EMG signal. Both signals were trained using an MLP network with 3 hidden layers ([4,000 2,000 800]). Figure 4(c) shows the typical architecture for this scenario. After training, we used the encoder as input for training decoders that can predict the next 400 time-steps based on 800 encoded time-steps from an original 4,000 time-steps input signal.

The combination of input/encoding/output can vary, depending on the level of abstraction and data compression we want to achieve. For the EMG envelope, it was possible to compress the signal to only 10% of the original data set (4,000 $\rightarrow$ 400), without losing much information. For raw EMG data, we tested different variations of the compression to achieve a good decoded signal. However, due to the complexity of the signal, we could only compress it to 20% of the original data set (4,000 $\rightarrow$ 800). For comparison purposes, we kept the same 20% compression rate for both signals (raw EMG and EMG envelope). Figure 5 shows how raw EMG data can be compacted and later used for signal prediction.

4) LSTM Autoencoder: One approach to sequence to sequence (seq2seq) prediction problems that have proven very effective is called the Encoder-Decoder LSTM [21]. Similar to the MLP autoencoder strategy, this architecture consists of two models: one for reading the input sequence and encoding it into a fixed-length vector, and a second for decoding the fixed-length vector and outputting the predicted

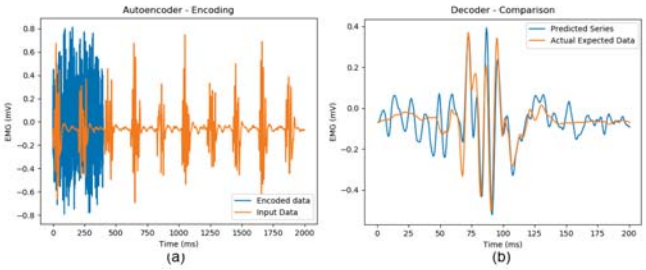

Fig. 5. MLP autoencoder: (a) shows how input data is compacted into a lower dimensional space (b) shows how decoder can use the incoding to predict future steps

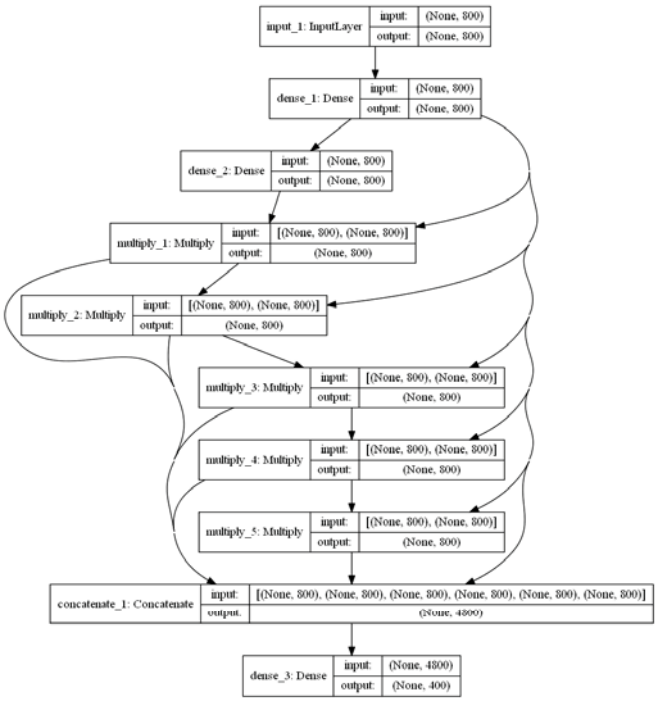

Fig. 6. Custom MLP decoder: re-injecting the first MLP output to other layers through linear combinations

sequence. The Encoder-Decoder LSTM was developed for natural language processing problems where it demonstrated state-of-the-art performance [16].

In this work, the LSTM Autoencoder architecture was used to train an encoder capable of reducing the dimensionality of the input EMG and accelerometer signals, and then a mixed LSTM and MLP decoder was used to predict the future EMG time-steps. Figure 4(d) shows the representation of such architecture for both signals, EMG envelopes, and EMG raw signals.

III. EXPERIMENTS AND RESULTS

This section describes the results for the EMG signal prediction based on the proposed architectures. We tested the proposed architectures for all the patients in the dataset, following the 2 proposed pre-processing techniques: the first one based on EMG envelopes, and the second one based on

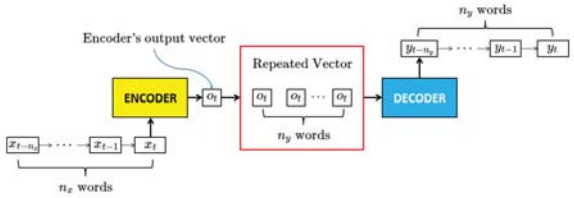

Fig. 7. The LSTM autoencoder architecture. The output of the encoding LSTM layer is a vector with the same number of dimensions from the number of cells on the LSTM layer. If we want to reshape that to a new sequence with size "ny", we need to repeat such output "ny" times, and making the decoding LSTM layer to output the input sequence. [22]

EMG raw signals. We then compare the results to evaluate the performance of each approach for each given input.

A. Implementation and Metric evaluation

All networks were designed and implemented using Keras framework on top of Tensorflow backend. MLP networks were implemented using a combination of dense layers with a different number of hidden units. Different activation functions were also evaluated, and the best results were achieved with rectified linear units (ReLU) and hyperbolic tangent (tanh).

For the LSTM networks, the proposed architectures used only LSTM layers with a dense regression layer in the end for generating the individual time steps outputs. For the LSTM autoencoder architecture, it is necessary to repeat the output weight vector from the encoding LSTM to the decoding LSTM layer n times, where n is the number of timesteps that we want to predict since LSTMs always expect a sequence of parameters as input. As the output of a LSTM is a vector with another representation of the sequence, we have to manually create the sequence again, so that the output of the decoder LSTM matches the number of inputs from the encoder. This architecture can be better visualized in Figure 7.

As the units of the EMG are in mV, it is useful to have an error metric that is also in the same unit as the expected output, so we can better evaluate the variance of the predicted signal. Both Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) fit these criteria, however, RMSE is more commonly used for biological signals evaluation, since the errors are squared before they are averaged, the RMSE gives a relatively high weight to outliers and significant bigger errors.

We evaluated all the models using RMSE. However, after comparing predicted signals and their RMSE values, we could see that for some cases the metric was not providing a good representation on the quality of the predicted signal (see Figure 9 and Table II). So after evaluating different loss functions (MSE, MAE, MAPE, MPE, Log CosH), which also did not improve the distinguishing between bad and good predictions, we designed a new "EMG Error" function γ, that penalizes large differences between the predicted signal p_i and actual signal a_i with a multiplication factor α, whenever the difference module is higher than a certain threshold ε. The result is also summed with the Mean Squared Error (MSE) calculated for the whole sample, so we are able to keep optimizing the loss value for the mean squared error and incorporating the penalty factor for diverging predictions. The custom designed loss function is defined based on the following equation:

$$\gamma = \frac{1}{n}\Sigma_{i=1}^{n}(p_i - a_i)^2 + \frac{1}{n}\Sigma_{i=1}^{j}|p_j - a_j| * \alpha, |p_j - a_j| \geq \varepsilon$$

As a result, we have a predicted signal that is much closer to the actual signal, specially on the large amplitudes for the high frequency component, as we can see on Figure 9 (items 7 and 8).

B. Comparison of proposed architectures

1) Prediction of EMG Envelopes: Table I shows the comparison of the proposed architectures, size of the model, training and test performance over epochs for all trained models. To evaluate the generalization of the model, we have included the test of the model on a second experiment session for the same patient (Test Loss 2) and a different patient (Test Loss 3).

MLP models were trained over 20 epochs, while LSTM models were trained over 5 epochs since the model loss did not improve significantly after that.

For a relatively simple pattern like the EMG envelope, most models can predict with a good level of confidence (RMSE $<$ 0.1mV). We can see that the best results are achieved with the decoder models after applying the MLP encoding (Table I items 8a and 9a). We can also see the impact of including regularization methods like dropout, making predicted signals more smooth and better adapted to the actual signal. We can also notice a better generalization result while testing for other data sets (Test Loss 2 and Test Loss 3).

Variations on the activation function can cause a change in the quality of the output signal, especially without regularization (Figure 8 items (b) compared against (a)). The output signal from the MLP network generated with ReLU is much more unstable than the one produced with the tanh activation function. The comparison also shows that using the Autoencoder approach for reducing the dimensionality of the signal before applying the decoding layers can provide a much better fit of the predicted curves.

2) EMG Raw Signal: Table II shows the comparison of the trained models for EMG raw signal. Models with the custom loss function "EMG Error" are marked by (**).

Considering the higher complexity of the EMG raw signal, with varying high frequencies combined with different amplitudes, the proposed MLP and LSTM models did not perform well. We can see that the best performance prediction model was the one based on an MLP decoder with 3 layers and a linear combination of layers (Architecture presented in Figure 6).

However, when comparing the Test Loss results, MLP models seem to perform well, which can lead us to the conclusion that even though the RMSE results are low, this

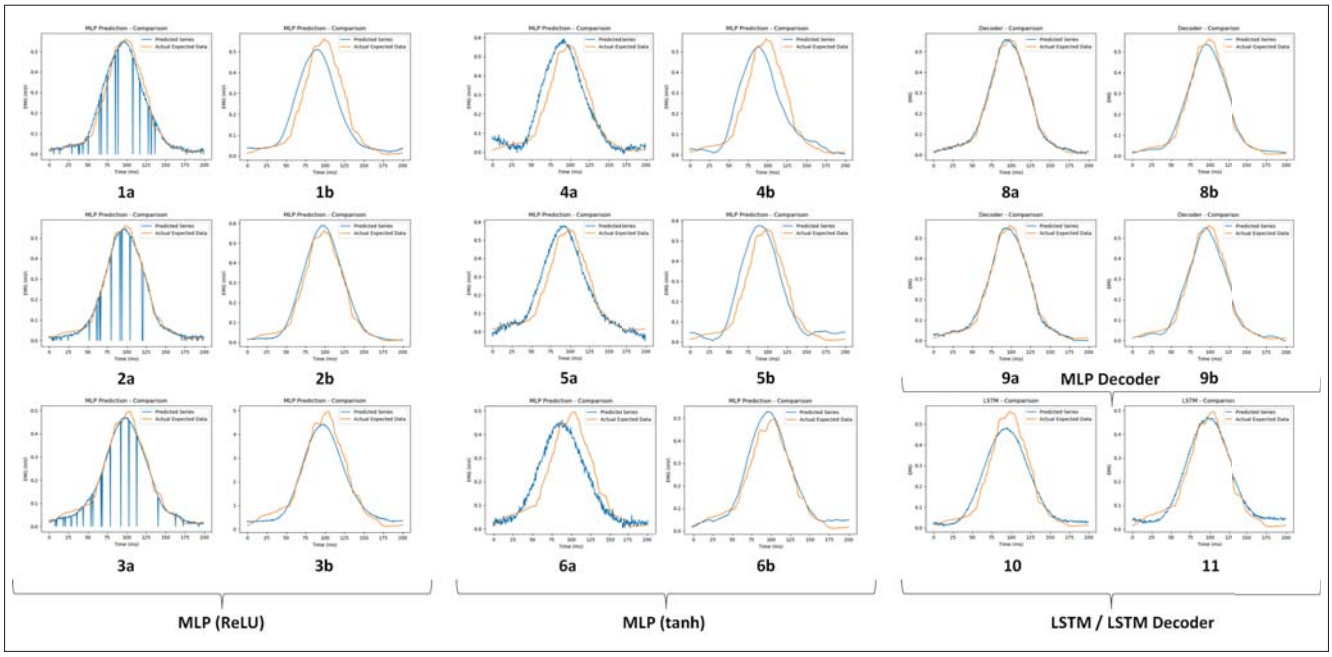

Fig. 8. EMG Envelope prediction for all proposed architectures. 1-3 are predictions based on MLP with ReLu activation. 4-6 are predictions based on MLP with tanh activation. 8-9 are predictions with MLP decoders and 10-11 are predictions using LSTM. (a) are predictions without and (b) predictions with 20% dropout between layers for regularization.

TABLE I
EMG ENVELOPE PREDICTION COMPARISON

Architecture (Layers) - Activation Function	# hidden units	Size	Regularization	Train Loss	Test Loss	Test Loss 2	Test Loss 3
1a. MLP - ReLU	[4000,1000,400]	80MB	None	0.06	0.06	0.09	0.20
1b. MLP - ReLU	[4000,1000,400]	80MB	Dropout(0.2)	0.05	0.04	0.07	0.18
2a. MLP - ReLU	[4000,2000,1000,800,400]	106MB	None	0.07	0.07	0.10	0.21
2b. MLP- ReLU	**[4000,2000,1000,800,400]**	**106MB**	**Dropout(0.2)**	**0.03**	**0.02**	**0.07**	**0.19**
3a. MLP- ReLU	[4000,2000,2000,800,800,400,400]	120MB	None	0.06	0.06	0.09	0.21
3b. MLP- ReLU	[4000,2000,2000,800,800,400,400]	120MB	Dropout(0.2)	0.04	0.03	0.07	0.21
4a. MLP - tanh	[4000,1000,400]	80MB	None	0.05	0.04	0.07	0.20
4b. MLP- tanh	[4000,1000,400]	80MB	Dropout(0.2)	0.05	0.04	0.07	0.19
5a. MLP- tanh	[4000,2000,1000,800,400]	106MB	None	0.04	0.04	0.07	0.20
5b. MLP- tanh	[4000,2000,1000,800,400]	106MB	Dropout(0.2)	0.05	0.04	0.07	0.19
6a. MLP- tanh	[4000,2000,2000,800,800,400,400]	120MB	None	0.04	0.04	0.08	0.23
6b. MLP- tanh	[4000,2000,2000,800,800,400,400]	120MB	Dropout(0.2)	0.05	0.05	0.07	0.22
7. MLP Autoencoder (*) - ReLU	Encoder:[4000,2000,800] Decoder:[800,2000,4000]	Enc.: 100MB Dec.: 40MB	Dropout(0.2)	0.04	0.03	0.04	0.15
8a. MLP Decoder - ReLU	**[800,800,400]**	**7MB**	**None**	**0.01**	**0.01**	**0.07**	**0.23**
8b. MLP Decoder - ReLU	[800,800,400]	7MB	Dropout(0.2)	0.02	0.02	0.07	0.21
9a. MLP Decoder Multiply - ReLU	**[800,800,400]***	**7MB**	**None**	**0.01**	**0.01**	**0.08**	**0.22**
9b. MLP Decoder Multiply - ReLU	[800,800,400]*	7MB	Dropout(0.2)	0.02	0.01	0.07	0.20
10. LSTM - tanh	LSTM[20]+Dense[400]	0.144MB	Dropout(0.2)	0.08	0.08	0.07	0.23
11. LSTM Autoencoder - tanh	Encoder: LSTM(20) Decoder: LSTM(20) + Dense(400)	5.6MB	Dropout(0.2)	0.55	0.57	0.68	0.78

does not indicate a good prediction of the signal, since we are considering the mean values of all points.

These results lead us to the definition of a new loss function, that can better capture the mix of the high and low frequency of the EMG signal, resulting on a better metric to evaluate signal prediction quality.

IV. DISCUSSION

From a general perspective, both models, based on MLP and LSTM performed well for the EMG envelope. For the raw EMG, only the MLP decoder approach was able to perform well with the encoded input from a previously trained MLP autoencoder. The pure MLP and the LSTM approaches did not provide satisfactory results for the given length of prediction time-steps in the future (400 points).

Autoencoder models also performed better in general, for both MLP and LSTM architectures. By providing a way to reduce the dimensionality of the input data set, they were able to reduce the complexity of forecast task, resulting in

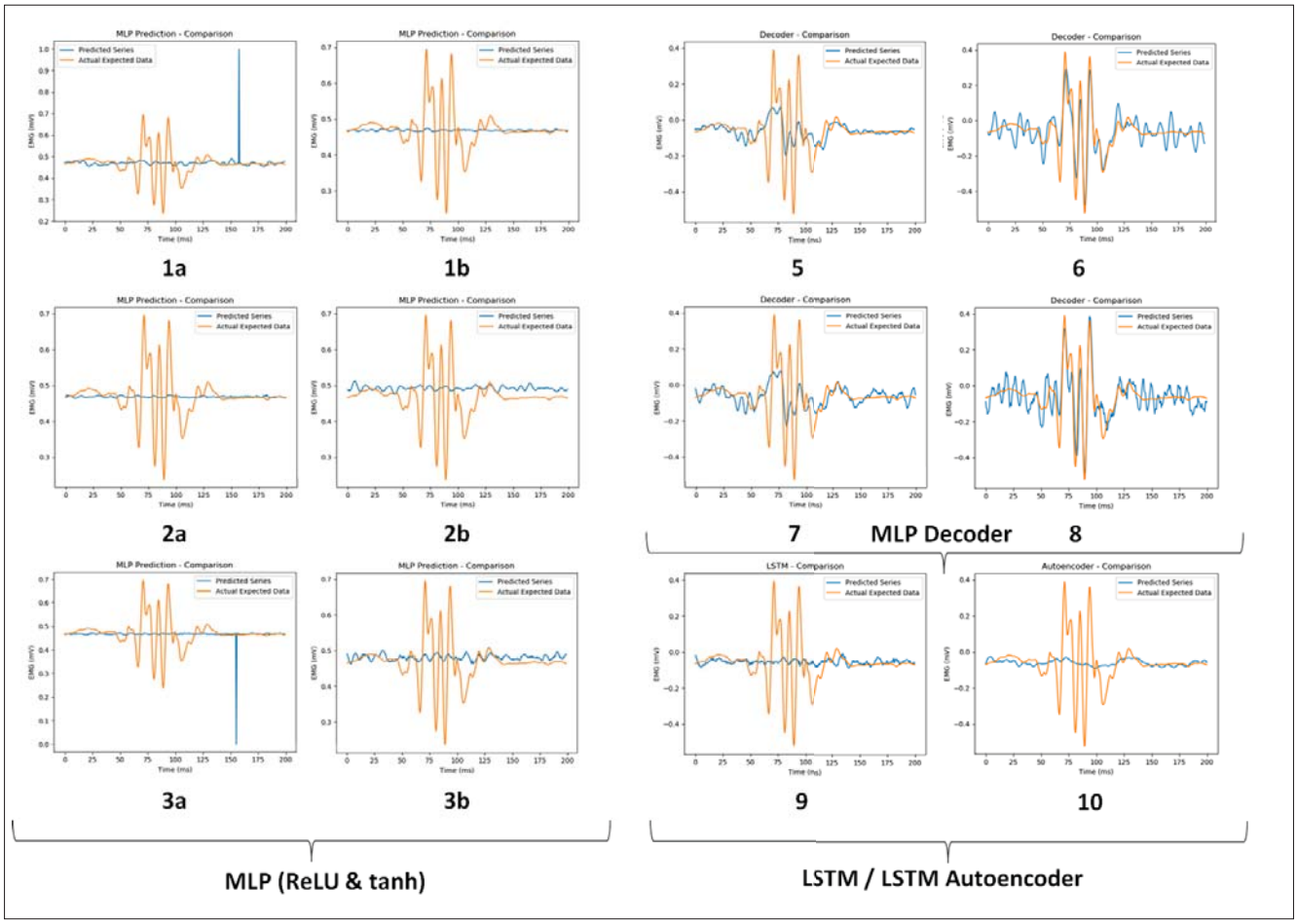

Fig. 9. Raw EMG signal prediction for all proposed architectures. 1-3 are predictions based on MLP with ReLu activation. 5-6 are predictions with MLP decoders. 7-8 are predictions with MLP decoders trained with custom loss function "EMG Error" and 9-10 are predictions using LSTM and LSTM autoencoder.

TABLE II

EMG RAW SIGNAL PREDICTION COMPARISON

Architecture (Layers) - Activation Function	# hidden units	Size	Regularization	Train Loss	Test Loss	Test Loss 2	Test Loss 3
1a. MLP (3) - ReLU	[4000,1000,400]	80MB	Dropout(0.2)	0.06	0.06	0.07	0.08
1b. MLP (3) - tanh	[4000,1000,400]	80MB	Dropout(0.2)	0.07	0.07	0.08	0.08
2a. MLP (5) - ReLU	[4000,2000,1000,800,400]	106MB	Dropout(0.2)	0.07	0.06	0.07	0.08
2b. MLP (5) - tanh	[4000,2000,1000,800,400]	106MB	Dropout(0.2)	0.06	0.07	0.07	0.07
3a. MLP (7) - ReLU	[4000,2000,2000,800,800,400,400]	120MB	Dropout(0.2)	0.07	0.07	0.07	0.08
3b. MLP (7) - tanh	[4000,2000,2000,800,800,400,400]	120MB	Dropout(0.2)	0.07	0.07	0.07	0.07
4. MLP Autoencoder (*) - tanh	Enc.:[4000,2000,800] Dec:[800,2000,4000]	Encoder:100MB Decoder: 7MB	Dropout(0.2)	0.06	0.07	0.16	0.18
5. MLP Decoder - tanh	[800,800,400]	6MB	Dropout(0.2)	0.12	0.13	0.15	0.16
6. MLP Decoder Multiply - tanh	[800,800,400]	12MB	None	0.05	0.17	0.23	0.23
7. MLP Decoder (3) - tanh (**)	[800,800,400]	6MB	None	2.13	0.13	0.15	0.17
8. MLP Decoder Multiply (3) - tanh (**)	**[800,800,400]**	**12MB**	**None**	**1.27**	**0.16**	**0.22**	**0.22**
9. LSTM (2) - tanh	LSTM(100)+Dense(400)	1MB	Dropout(0.2)	0.07	0.05	0.07	0.06
10. LSTM Autoencoder (3) - tanh	Enc: LSTM[100] Dec: LSTM[100] + Dense[400]	0.68MB	Dropout(0.2)	0.08	0.14	-	-

better approximations of the future signal. Tables I and II provide a comparison among tested models, evaluating the number of neurons, time and size of the proposed trained model, for both EMG Envelope and EMG raw signals.

The increase in the number of MLP layers and neurons shows a slight improvement in test error values and how fast the network converges. However, the higher the number of layers and neurons, the bigger the size of the output network,

therefore more memory and time needed to train the model. The increase in the number of dense layers and number of neurons at the end of the decoder models did not improve the remaining loss value for predicted signals on both MLP and LSTM networks.

By testing the trained models on EMG data from one patient to another (Test Loss 3), we were able to see that the prediction results were not as good as expected, which can indicate that the proposed models are not generalized enough for direct application on multiple patients. Trained models with the combination of data sets proved to be more generalized. However, the prediction error was too big. However, by using learning transfer techniques, it should be possible to reduce the time for re-training the models among different patients drastically.

Recent work has been done using Neural Networks for movement classification based on EMG data. Most commonly used architectures use hybrid approaches, combining CNNs with LSTM and MLPs.

This work distinguishes from recent work as it is the first one proposing to use NN not only to classify but to predict EMG signals, focusing on Parkinson's disease tremor patterns. He et al. [11] have shown that a hybrid LSTM + MLP approach performs well for pattern recognition when compared to pure MLP, CNN or traditional classification approached like SVM or KNN.

Laptev et. al. [23] shows that the LSTM Autoencoder approach can have distinguished better results than other types of neural networks, especially for anomaly detection. In [24], the proposed Autoencoder LSTM network was used in conjunction with a probabilistic formulation for uncertainty estimation, which provided better results, with almost 96% of coverage over a 95% confidence interval.

Future work could incorporate convolution layers (Conv1D) before the MLP or before the LSTM networks, to capture new spatial patterns from the time series data, such as those included in [25]. We can also propose the usage of these prediction models as inputs for other types of NN that can optimize the control parameters of FES devices.

V. CONCLUSIONS

This paper contributes with three main findings to its field. First, we have shown that the usage of both MLP and LSTM based networks can successfully predict EMG tremor behavior, not only predicting EMG envelopes but also the EMG raw signals. Such achievement can support the development of new ways to control and optimize FES devices for reducing resting tremor on PD patients. Second, by utilizing the autoencoder approach, we were able to successfully train encoding models that are capable of compacting EMG information into a lower dimensional space, which can still be used for pattern recognition and signal prediction. Finally, we compared the different MLP and LSTM topologies, evaluating the influence of hyperparameters on the models, and how the function loss can also affect the quality of the prediction, proposing a new loss metric to evaluate and train EMG prediction models.

REFERENCES

[1] W. H. Organization., "Neurological disorders: public health challenges," 2006.

[2] T. Fritsch, K. A. Smyth, M. S. Wallendal, T. Hyde, G. Leo, and D. S. Geldmacher, "Parkinson disease: Research update and clinical management," *Southern Medical Journal*, vol. 105, pp. 650–656, 2012.

[3] D. B. from Thomson reuters, "Disease briefing: Parkinsons disease - an abbreviated entry," 2014.

[4] P. H. Peckham and J. S. Knutson, "Functional electrical stimulation for neuromuscular applications," *Annual Review of Biomedical Engineering*, vol. 7, no. 1, pp. 327–360, 2005. PMID: 16004574.

[5] D. Zhang, P. Poignet, F. Widjaja, and W. T. Ang, "Neural oscillator based control for pathological tremor suppression via functional electrical stimulation," *Control Engineering Practice*, vol. 19, no. 1, pp. 74 – 88, 2011.

[6] H. A. G. Rojas and E. W. Jensen, "Methods for a movement and vibration analyzer," 2009. US Patent 12/442,784.

[7] K. H. Rosenbluth, S. L. Delp, J. Paderi, V. Rajasekhar, and T. Altman, "Devices and methods for controlling tremor," 2016. US Patent 2017/0014625 A1.

[8] E. Siores and L. Swallow, "Detection and suppression of muscle tremors," 2011. UK Patent GB 2444393.

[9] A. de Oliveira Andrade, A. R. de Pdua Machado, J. A. F. B. Jnior, G. L. Cavalheiro, E. A. de Oliveira, and A. P. S. S. Paixo, "Luva instrumentada e rtese ativa para a quantificao e atenuao do tremor humano," 2016. BR 102014023282-6 A2.

[10] A. C. Souza, F. M. Ramos, M. Dorado, L. Fonseca, and A. Bo, "A comparative study on control strategies for fes cycling using a detailed musculoskeletal model," *IFAC-PapersOnLine*, vol. 49-32, p. 204209, 2016.

[11] Y. He, O. Fukuda, N. Bu, H. Okumura, and N. Yamaguchi, "Surface emg pattern recognition using long short-term memory combined with multilayer perceptron," *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 5636–5639, 2018.

[12] D. Belo, J. Rodrigues, J. Vaz, P. Pezarat-Correia, and H. Gamboa, "Biosignals learning and synthesis using deep neural networks," *BioMedical Engineering Online*, vol. 16, 9 2017.

[13] W. C. Pinheiro, B. E. Bittencourt, L. B. Luiz, L. A. Marcello, V. F. Antonio, P. H. A. de Lira, R. G. Stolf, and M. C. F. Castro, "Parkinsons disease tremor suppression," *Proceedings of the 10th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2017)*, pp. 149–155, 2017.

[14] A. Hughes, S. Daniel, L. Kilford, and A. Lees, "Accuracy of clinical diagnosis of idiopathic parkinson's disease: A clinico-pathological study of 100 cases," *J Neurol Neurosurg Psychiatry*, vol. 56, pp. 938–939, 03 1992.

[15] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, pp. 1735–1780, 1997.

[16] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *NIPS*, 2014.

[17] K. Cho, B. van Merrienboer, aglar Gülehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation," in *EMNLP 2014*, 2014.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[19] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations*, 12 2014.

[20] F. van Veen, "Neural network zoo," 2017.

[21] J. Brownlee, *Long Short-Term Memory Networks With Python - Develop Sequence Prediction Models With Deep Learning*. Jason Brownlee, v1.5 ed., 2018.

[22] T. Tran, "Sequence to sequence," 2016.

[23] N. Laptev, J. Yosinski, L. E. Li, and S. Smyl, "Time-series extreme event forecasting with neural networks at uber," *ICML 2017 Time Series Workshop, Sydney, Australia*, 2017.

[24] L. Zhu and N. Laptev, "Deep and confident prediction for time series at uber," *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 103–110, 2017.

[25] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4580–4584, April 2015.