A comparative evaluation of time-delay, deep learning and echo state neural networks when used as simulated transhumeral prosthesis controllers

Charles R. Day
School of Computing and Mathematics
Keele University
Keele, United Kingdom
c.r.day@keele.ac.uk

Edward K. Chadwick
School of Engineering
University of Aberdeen
Aberdeen, Scotland
edward.chadwick@abdn.ac.uk

Dimitra Blana
School of Medical Sciences & Nutrition
University of Aberdeen
Aberdeen, Scotland
dimitra.blana@abdn.ac.uk

***Abstract* — Transhumeral amputation has a considerable detrimental effect on the amputee's quality of life and independence. Previous work has already established the potential for exploiting proximal humerus myoelectric and kinematic signals for the effective control of a myoelectric prosthesis. That previous work used a Time-Delay Neural Network (TDNN) to perform the mapping of six electromyographic (EMG) and six kinematic proximal humerus signals to predict elbow flexion/extension. Since that earlier work alternative deep learning and recurrent neural network architectures, well-suited to the processing of high-dimensional time-series data, have come to the fore. The work reported here is a comparative evaluation using the metric of RMS error for the predicted elbow flexion/extension angles output by TDNN, Long Short Term Memory (LSTM) and Echo State Network (ESN) architectures. For the most effective comparison we successfully reproduced TDNN results that were comparable to previous work (with average RMSE of 11.8 degrees on unseen test data). Then using the same training and testing datasets, and networks of broadly similar complexity, we evaluated the effectiveness of the LSTM and ESN approaches. The LSTMs trained here delivered an average RMSE of 10.4 degrees on unseen test data. The ESNs delivered an average RMSE of 16.3 degrees on unseen test data. The current work was not intended to find the best possible LSTM or ESN solution for this problem. Instead the intention was to see if any particular aspects of the network architectures worked better with the particular challenges of transhumeral biomedical engineering data of this sort.**

Keywords — Echo State Networks, Time-Delay Neural Networks, Long Short-Term Memory, transhumeral prosthesis control, time-series processing.

I. INTRODUCTION

Each year in the UK alone thousands of major limb amputations are carried out [1]. More than half of these amputees are under 55 years old and trauma is the major reason for upper limb amputation. Therefore, loss of the upper limb significantly affects an individual's ability to work, independence and overall quality of life. Amputees who choose to fit a prosthetic limb onto the remaining arm often have the option of an *active prosthesis*: one that has the potential to restore some upper limb function. Active prostheses can be body powered, using straps and cables to transfer energy from controlled upper-body movements, or *myoelectric*, which are electrically powered and use elements of the amputee's residual neuromuscular system for its control.

Myoelectric prostheses are controlled by *electromyographic signals* (EMG) generated by contractions of the amputee's residual muscles. There is an on-going need to increase the functionality and ease of control of myoelectric prostheses in order to facilitate their wider adoption. To this end biomedical engineers have sought to develop sophisticated control algorithms and signal processing techniques to allow more EMG signals to be used. Pulliam et al. (2011) for have shown that EMG from seven proximal residual muscles could be used to predict elbow flexion and forearm pronation/supination [2]. However, Fougner et al. (2011) have shown that by incorporating kinematic signals, derived from accelerometers sensing residual body motions, along with EMG prosthesis control could be further improved [3].

This sort of multi-sensor data fusion, combining EMG and inertial measurement unit (IMU) kinematic sensor data, is an area that is well suited to the application of artificial neural networks [4]. Additional promise derives from the possibility of using these sensor data as they evolve through time to deliver improved prosthesis control. The ability of one time-series processing neural network architecture, Time-Delayed Neural Network (TDNN) was the subject of work by Blana et al. (2015) [5]. In that study it was shown that TDNN processing of combined EMG and kinematic data could deliver effective predictions for elbow and forearm movements. Since those encouraging results were obtained additional highly effective time-series processing neural network architectures have come to the fore e.g. Long Short-Term Memory (LSTM) networks are leading proponents from the domain of deep-learning and Echo-State Networks (ESNs) from reservoir computing. The work reported here is an initial comparative investigation into the effectiveness of all three network architectures (TDNN, LSTM and ESN) when applied to this important and challenging real-world biomedical engineering task.

The approach taken here is not to try to obtain the very best performing networks of each type for the prosthesis controller task – that might result in networks with radically different characteristics. Instead we have attempted to create and

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

evaluate networks with broadly similar characteristics to see how well they each cope with the challenges posed by this particular task.

The remainder of the paper is organised as follows. Section II briefly introduces neural network architectures. Section III explains the biomedical engineering data acquisition techniques used and how the training and testing data for the networks were prepared. Section IV explains how the networks were trained and tested. Section V presents the results and Section VI discusses the results obtained so far and Section VII concludes the paper and outlines directions for future work.

II. Time-Series Networks

Time Delay Neural Networks (TDNNs), see Figure 1, are an established time-series processing neural network architecture, first introduced by Waibel et al. in 1989 [6]. TDNNs deal with the temporal extent of key features from the input signal using tapped-delay lines to provide a kind of short-term memory that feeds into an otherwise conventional feed-forward network architecture.

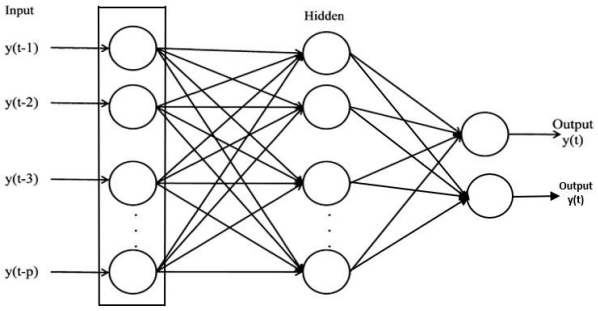

Figure 1: A schematic overview of a time-delay neural network.

The TDNN networks used here were trained and tested using the MATLAB Deep Learning toolbox [7].

Long Short-Term Memory (LSTM) networks were introduced by Hochreiter & Schmidhuber [8, 9] in order to more effectively deal time-series data where important components of key input signal characteristics had differing temporal extents. The fundamental building block of LSTM networks is the LSTM-cell (Figure 2).

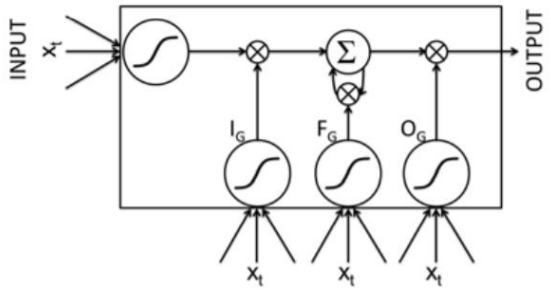

Figure 2: Overview of an LSTM-cell – the fundamental building block of LSTM neural networks.

Some of the most important features of an LSTM-cell are three tuneable parameters known as the *input gate* (I_G), *forget gate* (F_G) and the *output gate* (O_G). These gates control the extent to which: the current features from the input signal; the persistence of the current cell's state; and the significance of the current cell's state, respectively, will have on the output activation of the cell. The action of these gates are one of the key aspects of LSTM networks comprised of several cells that allows then to simultaneously respond to both short-term and long-term features of time-varying signals. The LSTM networks used here were trained and tested using the MATLAB Deep Learning toolbox [7].

Echo State Networks (ESNs) are a type of recurrent artificial neural network from the field of reservoir computing [10]. What sets ESNs apart from many other neural networks is the existence of a sparsely interconnected reservoir, since the recurrent connections within this reservoir allow time-series data to resonate, effectively giving the network a short term memory. A typical ESN topology can be seen in Figure 3.

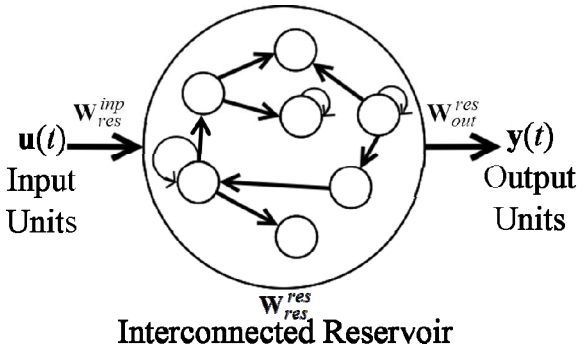

Figure 3: Schematic overview of a typical Echo-state neural network.

The input units are fully connected with the reservoir, whereas the reservoir units themselves are only sparsely interconnected, and all of these weighted connections are randomly generated at the initialisation of the reservoir and kept constant throughout. Only the weights on the connections between each reservoir unit and each output unit are trained, usually by a simple regression technique such as ridge regression [11–13]. The ESNs used here were trained and tested using the reservoir computing toolbox for MATLAB [14].

The ability of ESNs to exhibit a short term memory means that they have often been used to process time-series data in a number of fields, with recent examples including industry [15], medicine [16] and structural health monitoring [17].

In the ESN architecture used here, at any time t, the output of the vector of ESN reservoir neurons, $\mathbf{x}$, was given by (1).

$$\mathbf{x}(t) = f((1-\delta)\mathbf{x}(t-1) + \delta(\mathbf{W}_{res}^{inp}\mathbf{u}(t-1) + \mathbf{W}_{res}^{res}\mathbf{x}(t-1))) \quad (1)$$

Here, $\mathbf{W}_{res}^{inp}$ is the input to reservoir weight matrix, f is the tanh activation function that was used (other commonly used activation functions include the sigmoid function), $t-1$ is the

previous time step, $\mathbf{W}_{res}^{res}$ is the reservoir weight matrix, $\mathbf{u}(t)$ is the vector of the input data at time t and δ is the leak rate.

One advantage of ESNs is that there are several tuneable parameters, allowing networks to be configured for particular tasks. For example, although $\mathbf{W}_{res}^{inp}$ is randomly generated, it can be uniformly scaled according to the task, with a greater input scaling increasing non-linearity and increasing the relative effect of the input at time t compared to past inputs [19]. In this case, the scaled input weights were calculated by (2),

$$\mathbf{W}_{res}^{inp} = \rho \times \mathbf{W}_{res}^{inp} \quad (2)$$

where $\mathbf{W}'^{inp}_{res}$ represents the input to reservoir weight matrix prior to scaling and ρ is the input scaling.

Similarly, the randomly generated internal reservoir weights can be scaled in order to adjust the length of the ESN's short term memory. This can be done by multiplying the initial reservoir weight matrix by the spectral radius scaling factor and dividing by the maximum eigenvalue of this initial matrix [10], as shown in (3).

$$\mathbf{W}_{res}^{res} = \alpha \times \mathbf{W}'^{res}_{res} / |\lambda_{\max}| \quad (3)$$

Here, α is the spectral radius scaling factor and λ_{max} is the maximum eigenvalue of $\mathbf{W}'^{res}_{res}$, which represents the initial reservoir weights. The spectral radius is usually limited to a maximum value of 1 in order to ensure the echo state property for the network, and it can be seen in (3) that as the spectral radius decreases, so too will the resonance of past inputs in the reservoir, since the values of the reservoir weights will decrease.

Other parameters that can be adjusted include the reservoir size (i.e. the number of neurons in the interconnected reservoir) and the connectivity factor of the reservoir, which is the proportion of connections within the reservoir to reservoir units. The final tuneable parameter, the leak rate, acts as a time constant, speeding up or slowing down the reservoir dynamics. The effect of the leak rate on a reservoir unit can be seen in (1).

Each of the above feed-forward, deep-learning and reservoir-computing network architectures have proven track records in terms of their effectiveness in handling time-series data that have particular characteristics; this may make them particularly suitable for modelling the dynamics of limb motion. In addition, each of the network architectures have particular characteristics that will make direct neural network comparisons difficult. Despite these difficulties, in this study we seek to compare their effectiveness with the particular biomedical engineering data that will arise from this important developing application domain.

III. Data Acquisition

In Blana et al. [5] TDNNs were first evaluated to assess their effectiveness as transhumeral prosthesis controllers. In that work several able bodied participants were asked to repeat several controlled reaching and grabbing exercises with gesture targets rendered using a virtual reality headset. Before performing these reaching and grabbing exercises the transhumeral skin surface of each the participants had a number of EMG and IMU sensors attached. The EMG and IMU sensors provided a total of twelve input signal channels (i.e. six EMG signals and six IMU kinematic signals) that needed to be mapped into the correct elbow flexing and wrist pronation/supination angles. The ground truth values for these target angles were contemporaneously derived from suitable IMU sensors situated distally on the participants' humerus and forearms. The study was approved by the Keele University Ethics Panel and all participants gave informed consent. An overview of the data recording set-up is shown in Figure 4.

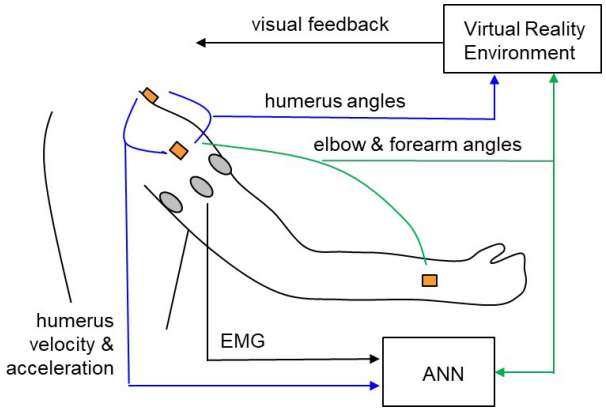

Figure 4: Overview of the data acquisition set-up. Gray ovals indicate EMG sensors; orange boxes indicate IMU sensors for kinematic data. IMU sensor data provide information that allows the ground truth elbow-flexing and wrist-rotation angles to be determined.

Examples of the kind of time-series data that were acquired in this way can be seen in Figure 5. Example input data streams are shown the top two panels depicting six channels of normalised EMG data and six channels of normalised IMU data. The bottom panel shows the target data stream i.e. elbow flexing/extending angle.

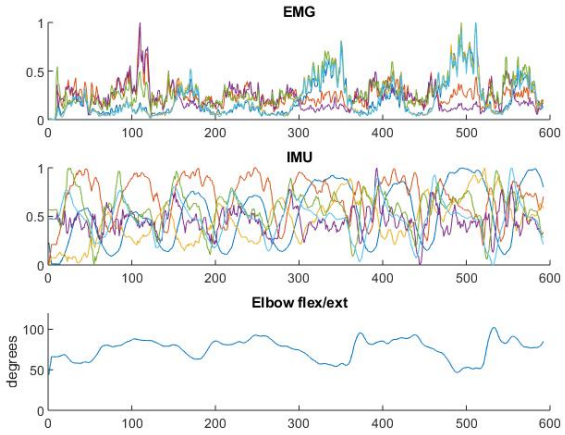

Figure 5: a typical sequence of normalised EMG and IMU input data and the corresponding target elbow flex/ext angle.

This data acquisition set-up was used with a total of twelve able-bodied human participants. Prompted by the virtual reality headset, each participant was shown a range of reaching and grabbing target objects that they were asked to capture several times. Delivering between 10 to 15 recorded time-series for each able-bodied participant. Each recorded time-series was typically around 30 seconds duration, sampled every 50ms (20Hz).

IV. Network Training and Testing

Blana et al. [5] experimented by training several TDNNs to act as transhumeral prosthesis controllers. Their TDNNs accepted the twelve dimensional EMG and IMU data, described above, as input and provided estimates of the desired elbow or wrist angles as the outputs. Separate elbow angle and wrist angle prediction TDNN networks were trained for each participant. That is, their TDNNs were participant and task (i.e. elbow or wrist angle prediction) specific. Their best performing TDNNs were determined using the RMS error of the network outputs and used a single layer of just 6 hidden nodes and three input time delays. Table I provides a summary of the best performing TDNNs reported by Blana et al.

TABLE I. The RMS prediction error, in degrees, for the ten participant specific TDNNs trained and evaluated by Blana et al. (2015). The mean performance for the TDNNs is also given (standard deviation in brackets).

Participant	Elbow flex RMSE (°)	
	Training data	Test Data
S1	2.8	15.8
S2	2.9	8.4
S3	3.4	11.5
S4	2.4	17.6
S5	2.6	11.4
S6	3.0	20.8
S7	3.3	17.1
S8	2.0	6.1
S9	2.1	9.0
S10	2.5	19.4
Mean (std)	2.7 (0.5)	13.7 (5.1)

The main objective of the work reported here is to compare the performance of the three network architectures to assess their effectiveness in this biomedical engineering domain.

One approach for this type of machine learning comparison would be to rigorously configure networks with equivalent numbers of free-parameters. For neural networks this would largely be determined by the total number of weights and biases in each of the networks. However, this is not the approach that has been taken here.

As described above in Section II each of the network architectures under consideration are radically different. For example, TDNNs use fully connected hidden layers and several delay lines, LSTMs require individual cells with some significant internal structure and ESNs usually operate with large, but only sparsely interconnected, reservoirs. So, instead of rigidly constraining networks to have the same numbers of free parameters, our approach was to consider each network architecture's principal operational features and, based on those network configuration options, compare how well they were able to complete the prosthesis control task.

An important aspect of the comparison carried out in this work is that it was possible to control for the preparation and treatment of the training and testing datasets that would be used by each of the network architectures. A total of 109 recordings of time-series data were available from Blana et al's data acquisition process. In each time-series the 12 input signals and the ground-truth elbow flexion angles were present and for each time-series used to train/test the networks the inputs were all normalised over the range [0, 1]. Each of these participant-specific time series were then divided into sub-sequences, in the proportions [0.7, 0.15, 0.15], to be used as either: network training data; training-validation data; or the testing data sets, respectively.

Blana et al. [5] have already determined that TDNNs with 12 input nodes, 6 hidden nodes and a single output node worked best with this type of prosthesis related data. Our first step was to see if the TDNN results reported in that work were reproducible using the particular treatment of the available data described above.

In keeping with the motivation outlined in Section I, above, a series of LSTM networks with broadly similar overall characteristics i.e. 12 input units, 6 LSTM-cells and a single output unit were created. One of the important training options for deep-learning networks such as LSTMs is to train using penultimate layers of *drop-out* nodes to prevent premature overfitting [18]. An initial grid-search to assess the effectiveness of using drop-out indicated that there was no significant benefit to be gained here. So for all of the the LSTM networks that we report on below drop-out was not used, and just the training-validation datasets were used to prevent overfitting.

An equivalent set of ESN networks were also configured to have 12 input units and a single output unit, but here the intervening reservoir was comprised of 60 reservoir units. This reservoir size was chosen to reflect the fact that ESN reservoirs (the seat of the ESN's short-term memory time-series processing power) are, necessarily, only sparsely interconnected. For our ESNs only 10% of the possible interconnections were realised. An initial grid-search was again performed to estimate some useful settings for several of the other ESN specific training parameters. With the small subset of data used for the grid-search an ESN spectral radius of 0.1 and reservoir-unit leak-rate of 0.5 were selected for use with all of the ESNs.

For each of the time-series recordings that were available from all of the participants ten separate neural networks of each architectural type were trained and tested, using the appropriate

training-validation dataset to avoid overfitting. In this way the best performing participant-specific network could be found for each of the architectural approaches. In addition, it was also possible to determine the mean participant-specific network performance for each of the network architectures: averaging network performance over all of the time-series recorded by each participant. (N.B. in most realistic transhumeral prosthesis usage scenarios it is likely that a patient-specific controller would be desirable. In the case of amputees this might best be developed using bespoke patient-specific data captured from a remaining upper-limb.)

V. Results

Tables II, III and IV, below, summarise the performance of each of the network types evaluated in this paper. Table II shows the results of attempting to reproduce Blana et al's previous results using TDNNs. Table IIII shows equivalent results for the LSTMs and Table IV gives the results for the ESNs. Each of these tables shows for each of the participants: the best RMSE obtained from any network using its training data; as well as the average RMSE obtained from all of that participant's networks when using its training data. The final two columns of each of these tables show the best and average RMSEs obtained – but this time when using each network's testing data.

TABLE II. The TDNN RMS prediction error, in degrees, for the ten participants in this study. The mean performance is also given (standard deviations in brackets).

Participant	Elbow flex RMSE (º) (training data)		Elbow flex RMSE (º) (testing data)	
	Best	Mean (std)	Best	Mean (std)
S1	1.9	8.4 (4.9)	4.2	9.9 (4.0)
S2	2.8	9.1 (4.0)	1.0	11.3 (4.5)
S3	1.9	11.6 (6.7)	3.6	14.6 (7.5)
S4	2.1	11.4 (8.0)	2.3	12.9 (8.4)
S5	2.5	10.6 (4.6)	2.0	11.2 (6.0)
S6	0.9	11.9 (9.7)	4.1	13.0 (7.5)
S7	5.1	13.1 (7.5)	6.1	14.7 (7.7)
S8	2.4	8.3 (5.2)	2.8	9.4 (5.3)
S9	2.1	8.6 (4.5)	3.0	10.8 (5.5)
S10	2.1	8.7 (4.2)	3.5	10.4 (5.1)
Mean (std)	2.3 (1.1)	10.2 (1.8)	3.3 (1.4)	11.8 (1.9)

Comparing the findings shown in Table II with those in Table I shows that some aspects of the previous TDNN results were reproducible here. Broadly similar network performance was obtained when comparing the best networks for each participant when evaluated using their training data. Inspecting the results when the TDNNs were evaluated using their testing data though shows that generally superior performance was obtained from the networks trained for the current study and that their performance was less variable.

The results presented in Tables III and IV below, allow similar comparisons to be made for the LSTM and ESN approaches. Generally, the best LSTM networks were less able to fit their training data than any of the best TDNNs or the best ESNs. (Though interestingly all of the network types seem to have struggled with training data recorded from participant S7.) The standard deviations for this set of comparators also suggested similar levels of variability for the most pre-eminent networks when fitting their training data.

TABLE III. The LSTM RMS prediction error, in degrees, for the ten participants in this study. The mean performance is also given (standard deviations in brackets).

Participant	Elbow flex RMSE (º) (training data)		Elbow flex RMSE (º) (testing data)	
	Best	Mean (std)	Best	Mean (std)
S1	3.0	5.8 (1.3)	3.5	8.7 (3.3)
S2	4.1	7.0 (1.4)	3.0	11.5 (4.0)
S3	3.9	8.1 (1.8)	4.2	14.0 (6.1)
S4	4.3	7.6 (1.7)	7.1	10.8 (4.5)
S5	3.3	7.2 (1.8)	4.4	10.4 (5.4)
S6	4.3	7.5 (1.5)	2.7	9.8 (3.5)
S7	5.7	8.4 (1.3)	5.2	12.6 (4.3)
S8	3.3	6.1 (1.7)	2.7	7.9 (3.4)
S9	3.6	6.2 (1.5)	4.3	9.0 (3.4)
S10	4.2	6.3 (1.4)	7.2	9.3 (2.4)
Mean (std)	3.9 (0.8)	7.0 (0.9)	4.4 (1.6)	10.4 (1.9)

TABLE IV. The ESN RMS prediction error, in degrees, for the ten participants in this study. The mean performance is also given (standard deviations in brackets).

Participant	Elbow flex RMSE (º) (training data)		Elbow flex RMSE (º) (testing data)	
	Best	Mean (std)	Best	Mean (std)
S1	2.2	3.9 (0.9)	4.1	10.6 (6.0)
S2	2.8	4.1 (0.7)	3.7	17.1 (13.9)
S3	0.6	4.2 (1.2)	4.8	17.5 (12.3)
S4	2.5	3.9 (0.9)	2.6	14.8 (8.8)
S5	1.0	3.9 (1.2)	6.9	15.7 (9.3)
S6	0.9	4.1 (1.2)	1.8	25.7 (35.2)
S7	4.2	5.3 (1.5)	9.2	23.3 (22.1)
S8	1.7	3.3 (0.9)	3.4	13.7 (9.4)
S9	2.8	3.6 (0.7)	4.0	11.7 (7.7)
S10	2.9	3.7 (0.7)	4.8	12.9 (9.1)
Mean (std)	2.6 (1.1)	4.0 (0.5)	4.5 (2.1)	16.3 (4.9)

The evaluation is significantly different when considering the performance of each architecture with their testing datasets. The mean performance of the TDNNs in the current study is once again slightly superior to the test performance reported by Blana et al. and the standard deviations suggest their test data

performance is also less variable. On average the best performing networks with previously unseen test data were the LSTM networks. On average the worst performing networks with previously unseen test data were the ESNs.

VI. Discussion

The comparatively poor average ESN performance with unseen test data is suggestive of potential overfitting to the training data. However, a more detailed inspection of the ESN outputs when processing many of their test datasets, see Figure 6, suggests that, in some cases, another ESN reservoir phenomenon might be partially influencing their degraded performance.

When some trained ESNs need to perform a mapping that involves a marked transition from either a non-linear to a less variable output characteristic, or vice versa, their reservoir dynamics can sometimes give rise to highly eccentric output activations that can persist for some time after the transition [19, 20]. The precise input data and trained reservoir characteristics that give rise to this behaviour are not well understood. ESNs seem perfectly capable of processing non-linear data or data that transitions relatively slowly. The issue appears to arise when there is a marked transition. It may be that the degraded ESN performance with some of the test data evaluated here is an example of this kind of behaviour. Figure 6 shows how the output of one of the trained ESNs upon encountering a need to transition its outputs from relatively variable to relatively constant characteristic instead begins to behave erratically.

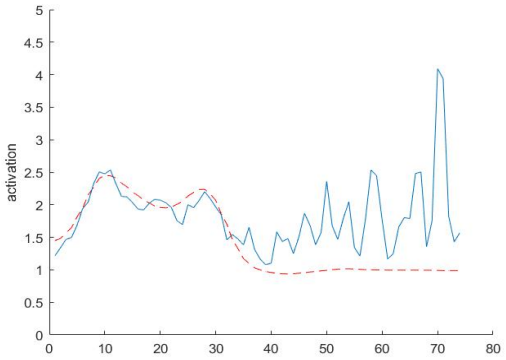

Figure 6: ESN target activation (dashed red) and output unit activation (solid blue) when processing previously unseen test data for participant S6.

LSTM networks by virtue of the LSTM-cell's built-in adaptive internal gate-structures (allowing different parts of LSTM networks to become attuned to either slowly or more rapidly varying data characteristics) may already be pre-disposed to cope well with these sort of phenomena.

ESN networks have no such reservoir-adaptation capability: the ESN reservoir's connectivity is fixed at network initialisation and if such characteristic data transitions are not a significant part of the ESNs training data this might explain why the trained ESNs could episodically deliver poorer test data performance than expected.

VII. Conclusion and Future Work

Currently the LSTM networks seem to hold the greatest promise for the development of neural controllers for upper limb prostheses of the sort modelled here.

This conclusion is based on the overall performance rates of the LSTM networks when processing their previously unseen test datasets. This is the scenario that best models the potential future operating environment for any neural controller for such prostheses.

None of the results reported above have involved any filtering or smoothing of the network outputs (in case they might obscure potentially interesting network properties in this comparative evaluation). However, such filtering and smoothing of the outputs is likely to be highly beneficial, to emulate some of the inertial properties of real limbs, if the alternative biomedical engineering objective of developing the most effective prosthesis controller were brought to the fore.

More fundamental neural network work, attempting to address the issues summarised in Figure 6 and Section V above, will investigate the effectiveness of a hybrid ESN architecture, *Reservoir with Random Static Projections* (R^2SP), [19, 20] with this domain data.

The revised R^2SP ESN model was first proposed by Butcher at al. to show that by supplementing the standard ESN reservoir architecture with two additional, strategically placed, feed-forward layers, the augmented architecture did help to overcome some of the non-linear/linear data transition challenges. The modestly sized ESN reservoirs and the time series data used for the current study may well make a more detailed investigation using R^2SP ESN variants insightful. Both for further neural network research as well as for biomedical engineers seeking to develop ever more effective prosthesis controllers.

Acknowledgment

The authors are grateful to ten anonymous, able-bodied, human participants who participated in the recording of all of the datasets used to train and test the above neural networks.

References

[1] NASDAB, 2005. National Amputee Statistical Database (NASDAB). The Amputee Statistical Database for the United Kingdom 2004/05.

[2] Pulliam, C. L., Lambrecht, J. M., Kirsch, R. F., 2011. Electromyogram-based neural network control of transhumeral prostheses. The Journal of Rehabilitation Research and Development 48 (6), 739.

[3] Fougner, A., Scheme, E., Chan, A. D. C., Englehart, K., Stavdahl, O., Dec. 2011. Resolving the limb position e ect in myoelectric pattern recognition. IEEE transactions on neural systems and rehabilitation engineering : a

publication of the IEEE Engineering in Medicine and Biology Society 19 (6), 644–51.

[4] Smith, D.L., & Singh, S. (2006). Approaches to Multisensor Data Fusion in Target Tracking: A Survey. IEEE Transactions on Knowledge and Data Engineering, 18, 1696-1710.

[5] D. Blana, T. Kyriacou, J.M. Lambrecht, E.K. Chadwick, Feasibility of using combined EMG and kinematic signals for prosthesis control: a simulation study using a virtual reality environment, Journal of Electromyography and Kinesiology (2015), doi: http://dx.doi.org/10.1016/j.jelekin.2015.06.010

[6] Waibel, A.H., Hanazawa, T., Hinton, G.E., Shikano, K., & Lang, K.J. (1989). Phoneme recognition using time-delay neural networks. IEEE Trans. Acoustics, Speech, and Signal Processing, 37, 328-339.

[7] MATLAB and Deep Learning Toolbox Release 2018b, The MathWorks, Inc., Natick, Massachusetts, United States.

[8] Hochreiter, S., Schmidhuber, J. (1997). "Long short-term memory". Neural Computation. 9 (8): 1735–1780. doi:10.1162/neco.1997.9.8.1735.

[9] Hochreiter, S., Schmidhuber, J. (1996). LSTM can solve hard long time lag problems. Advances in Neural Information Processing Systems. 473-479.

[10] H. Jaeger, "The 'echo state' approach to analysing and training recurrent neural networks," Fraunhofer Institute for Autonomous Intelligent Systems, 2001.

[11] X. Dutoit, B. Schrauwen, J. V. Campenhout, D. Stroobandt, H. V. Brussel, and M. Nuttin, "Pruning and regularization in reservoir computing," *Neurocomputing*, vol. 72, pp. 1534–1546, 2009.

[12] A. E. Hoerl and R. W. Kennard, "Ridge Regression: Biased Estimation for Nonorthogonal Problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[13] D. C. Montgomery, E. A. Peck, and C. G. Vining, *Introduction to Linear Regression Analysis*. Wiley, 1982.

[14] D. Verstraeten, B. Schrauwen, M. D'Haene, and D. Stroobandt, "An experimental unification of reservoir computing methods," *Neural Networks*, vol. 20, no. 3, pp. 391–403, 2007.

[15] J. Park, B. Lee, S. Kang, P. Y. Kim, and H. J. Kim, "Online Learning Control of Hydraulic Excavators Based on Echo-State Networks," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 1, pp. 249–259, Jan. 2017.

[16] Z. H. Khan, N. Hussain, and M. I. Tiwana, "Classification of EEG signals for wrist and grip movements using echo state network," *Biomedical Research*, vol. 28, pp. 1095–1102, 2017.

[17] E. A. Antonelo, E. Camponogara, and B. Foss, "Echo State Networks for data-driven downhole pressure estimation in gas-lift oil wells," *Neural Networks*, vol. 85, pp. 106–117, 2017.

[18] Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting." Journal of Machine Learning Research. Vol. 15, pp. 1929-1958, 2014.

[19] Butcher JB, Verstraeten D, Schrauwen B, Day CR, Haycock, P.W.. 2010. Extending reservoir computing with random static projections: a hybrid between extreme learning and RC. 18th European Symposium on Artificial Neural Networks (ESANN 2010) (pp. 303-308). Evere, Belgium: D-Side.

[20] Butcher JB, Verstraeten D, Schrauwen B, Day CR, Haycock PW. 2013. Reservoir Computing and extreme learning machines for non-linear time-series data analysis. Neural Networks, vol. 38, 76-89.