Constructive cascade networks for brain-signal stress-prediction

[Name withheld]

School of Computing, Australian National University
[email withheld]

Abstract. Reliable identification of stress is a key first step in mitigating its associated negative health impacts. Rahman *et al.* recently achieved high accuracy (98.6%) predicting stress from brain signals using a single-layer neural network architecture ('Hidden30') with feature pre-selection by genetic algorithm [6]. The present work compares the prediction quality and training cost of this architecture against those of *Casper*, a constructive cascade network which is in principle capable efficient higher-order feature detection [10]. Robust performance improvements for all models are obtained by removing outliers from the original dataset and restricting attention to a single brain region (F7); however, neither Casper nor Hidden30 provide a consistent predictive advantage over simple logistic regression, which serves as a natural benchmark. Logistic regression in fact achieves perfect accuracy on the F7 dataset, while training cost is determined primarily by choice of optimiser rather than network architecture. Casper thus provides a procedural advantage over Hidden30 as it reduces initially to logistic regression, increasing the network size only as necessary. Significantly more data is required to conclusively establish these results.

1 Introduction

Chronic or traumatic stress is known to contribute both physiologically and behaviourally to negative health outcomes such as mental illness and disease [9,5]. As a first step towards effective stress-management, machine learning techniques present a promising opportunity for personalised, automated stress identification based on physiological signals such as electroencephalograms (EEGs) [3]. An EEG measures electrical activity in various regions of the brain via electrodes placed on the scalp. Beyond predicting stress, the techniques and features established in analysing such signals may prove useful in monitoring other indicators (e.g. skin temperature) and/or addressing other outcomes such as emotional state or mental illness.

Recently Rahman *et al.* [6] studied the effectiveness of several combined feature-selection and classification techniques for predicting music-induced emotional responses based on statistical features of EEG signals. In particular, genetic-algorithm feature selection followed by neural-network classification achieved 98.6% accuracy in distinguishing calm ('relaxed') and stressed ('tense') responses; it remains to determine if such prediction can be achieved more efficiently, accurately, or less intrusively. To this end, the present work performs the same classification task on several subsets of the original data, and compares the performance of three neural network models: (1) simple logistic regression; (2) the single-layer 'Hidden30' architecture from the original study; and (3) a *constructive cascade network* (CCN) known as *Casper* [10].

CCNs successively add hidden neurons to the network throughout training, encouraging each to identify the most important remaining features and allowing these features to be higher-order. In principle, CCNs can thus perform efficient intrinsic 'feature-selection' and produce smaller-sized models. The present paper therefore aims both to benchmark the results from the previous work and assess the potential benefit of Casper as a combined feature-selector and classifier for predicting stress from EEG signals.

In this work we find that additional neurons appear to decrease Casper's performance, suggesting that they do not function as effective feature-detectors. Top prediction quality is consistently achieved by logistic regression or (equivalently) Casper with no hidden neurons, particularly when trained using resilient backpropagation (Rprop). Training cost is also determined primarily by choice of optimiser, with Rprop consistently fastest. For the stress-prediction task neither Casper nor Hidden30 therefore provide an advantage over logistic regression. Moreover, the accuracy of all models is improved when training data is restricted to the single brain-region F7, with logistic regression achieving perfect accuracy. This suggests that the achievable accuracy in predicting stress from EEG data may be limited by either overfitting or the complexity of the chosen optimisation task; Casper may therefore be preferable to Hidden30 as it begins with the simplest possible network, which in this case is sufficient.

The remainder of the paper is structured as follows. Sec. 2 describes the EEG dataset and preprocessing; Sec. 3 introduces CCNs and describes the present implementation of Casper. Sec. 4 describes models 1-3 and the evaluation techniques chosen for fair, effective comparison of Casper and Hidden30. Sec. 5 discusses findings; Sec. 6 briefly summarises, and proposes directions for future work. Datasets (original/processed) and python code are included as supplementary material.

2 Dataset

In the original experiment [6], 13 male and 11 female participants each listened to 8 pieces of music across three genres, and rated their subjective response along several emotional dimensions. While listening, participants' EEG

signals were measured at 14 positions split across the pre-frontal (AF), frontal (F), temporal (T) and occipital (O) lobes of both brain hemispheres. Summary statistics were extracted from each signal after smoothing and frequency-filtering.

The available subset of this data includes 3 calm-rated and 3 stress-rated samples for each participant, and is therefore balanced both for individual participants and the entire group. 'Calm' and 'stress' ratings were (re)encoded as 0 and 1 respectively, as appropriate for binary classification with sigmoid output activation. Table 1 presents the 15 statistical features available for each EEG channel; the total number of input features across all 14 channels is therefore 210. Each feature is naturally a continuous-valued real variable, and was standardised via scaling and shifting to have mean zero and unit variance. Further description of these features and the original experiment can be found in [7].

Table 1. Available statistical features for each channel, grouped approximately by what they measure.

Linear	**Nonlinear**
Mean, sum;	Hurst exponent ('long-term memory');
Min, max, interquartile range;	Hjorth mobility ('mean frequency');
Standard deviation, variance, root-mean-square;	Approximate entropy, fuzzy entropy;
Skewness;	
Mean first difference, mean second difference;	

Clustering of the input features using principal component analysis (PCA) reveals two outlying participants P4 and P5, as shown in Fig. 1. Their distinct separation from the remaining participants contributes the largest principal component, accounting for 40% of total variance; the next-largest explains only 13% of total variance and does not visibly divide participants into distinct subgroups. The effect of removing the outliers is discussed in Sec. 5.

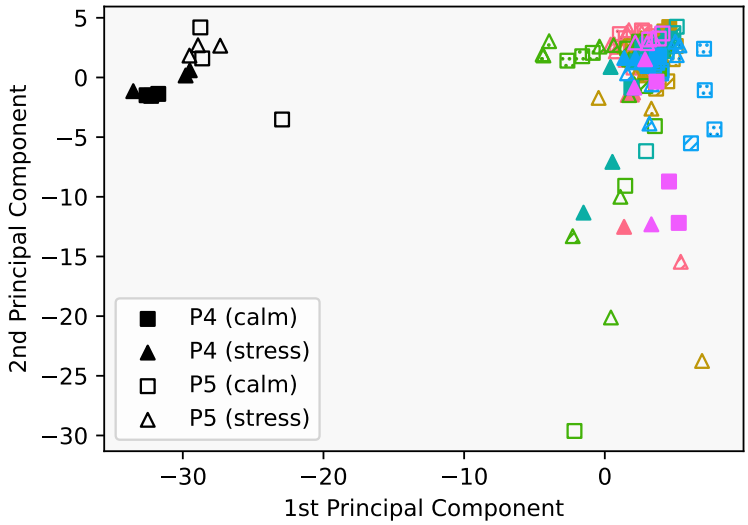

Fig. 1. First two principal components of the input feature set. Squares and triangles represent calm and stressed responses respectively, and participants are distinguished by a unique colour-fillstyle combination. Participants 4 and 5 are clear outliers.

3 Constructive cascade networks

Unlike conventional 'static' architectures, *constructive* neural networks determine their own topology by progressively adding neurons throughout training. In particular, constructive cascade networks (CCNs) begin as a linear regression architecture with inputs connected directly to the output; training then proceeds in *stages* corresponding with the addition of successive hidden neurons (see Fig. 2). The existing network is trained until performance plateaus; a new hidden neuron is then generated and added to the network, with *cascaded* fan-in from all existing input and hidden neurons and fan-out directly to the output unit(s). Adding and training hidden neurons one-by-one maximises their individual error-reduction value and avoids the so-called 'herd' phenomenon [1], while cascaded

fan-in from previous hidden units permits higher-order feature detection. In principle, CCNs therefore produce smaller networks for a given accuracy than pre-chosen static architectures.

The original *cascade correlation* CCN algorithm (Cascor) [1] trained only the output layer during each stage. New hidden neurons were instead trained independently before addition to the network, to correlate with residual output error and thus serve as effective feature-detectors; the resulting fan-in weights were then frozen. Training Cascor is thus exteremely fast, as it optimises only a single layer of weights at a time and permits caching of all neuron activations on a given dataset. However, correlation-based neuron pretraining leads to saturated weights and correspondingly sharp changes in output with respect to input [10]. Weight-freezing may also reduce the efficiency of each feature detector.

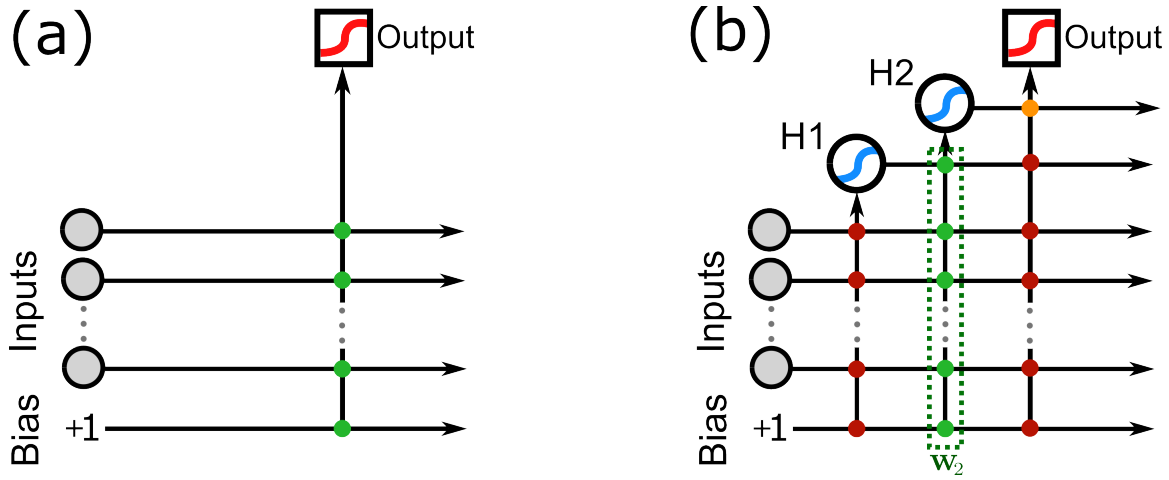

Fig. 2. The Casper constructive cascade network: (a) at initialisation, connecting inputs directly to the output via a sigmoid activation function; and (b) after the addition of two successive hidden neurons H1 and H2. Vertical lines indicate fan-in to each neuron, while horizontal lines indicate fan-out; each crossing is thus associated with a weight parameter. Fan-in weights to the most recent hidden neuron (green) are trained with learning rate $L^{(1)}$; fan-out from this neuron (orange) with rate $L^{(2)}$; and remaining weights (red) with rate $L^{(3)}$, where $L^{(1)} \gg L^{(2)} \gg L^{(3)}$.

The *Casper* algorithm [10,12] resolves these issues with Cascor by adding newly-initialised hidden neurons directly to the network and training *all* weights during each stage. Herd-avoiding, neuron-efficient feature-detection is maintained by replacing Cascor's neuron pre-training and freezing with a difference in learning rates $L^{(1)} \gg L^{(2)} \gg L^{(3)}$ for the corresponding weights, as illustrated in Fig. 2. Initial weights are trained using the $L^{(1)}$ rate. Training all weights precludes activation caching and increases the cost of optimisation.

CCNs are compatible with a wide choice of optimisers.[1] Casper was originally proposed using (progressive) *resilient backpropagation* (Rprop) [8,10]; for each parameter θ_p this algorithm maintains a separate adaptive step size δ_p, which is increased or decreased according to consistency of the corresponding loss gradient. The original Casper also implemented weight decay with simulated annealing (SARPROP [11]) as

$$\frac{\partial \ell}{\partial \theta_p} \leftarrow \frac{\partial \ell}{\partial \theta_p} - k \operatorname{sign}(\theta_p) \cdot \theta_p^2 \cdot 2^{-0.01h}$$

where h is the number of epochs since the last hidden neuron was added, and k is a decay hyperparameter. In the present case SARPROP was found to be detrimental and will not be discussed further.

The present paper uses sigmoid activations for each Casper hidden unit. To detect a performance plateau, each stage is broken into periods of $15 \cdot 1.5^n$ epochs where n is the number of hidden neurons. A new neuron is added if training loss does not decrease by 2% during this period; however, no new neurons are added once training loss falls below 10^{-7}. Values for $L^{(1,2,3)}$ and further details are given in Sec. 4.

4 Methodology

This section describes the training and architecture details of each model, as well as the evaluation procedure chosen to permit fair, holistic comparison between Casper [10] and the static neural network used by Rahman *et al.* [6].

[1] Note that each neuron receives a loss-gradient contribution from all subsequent neurons.

4.1 Models

The previous work [6] trained a feedforward neural network with a single fully-connected 30-unit hidden layer and unspecified nonlinearity; here this is deemed the 'Hidden30' architecture. To benchmark this result and assess potential performance/feature-detection benefits of Casper, the present work implemented (1) Hidden30 with ReLU nonlinearity; (2) Casper, as described in the previous section; and (3) simple logistic regression $y = \sigma(\mathbf{w} \cdot \mathbf{x})$, as a natural benchmark. Here σ is the sigmoid activation function, and biases are absorbed into the weights of each model. All models have a single output neuron with sigmoid activation and binary cross-entropy loss ℓ, as appropriate for a 0-1 classification task. The classification threshold was taken to be 0.5.

The parameters $\boldsymbol{\theta}$ of architectures 1-3 were trained with each of three optimisation methods:

(a) Full gradient descent (GD): $\boldsymbol{\theta}^{t+1} \leftarrow \boldsymbol{\theta}^t - \gamma \nabla_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}^t)$, where γ is the learning rate (lr). Batched (stochastic) gradient descent was not beneficial here.

(b) Adaptive momentum (Adam) [4] with learning rate γ and (standard) hyperparameters ($\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$). A smaller batch size (bs) was found to improve Adam's performance; note that this affects the effective learning rate.

(c) Rprop [8], with (standard) parameters $(\eta_+, \eta_-) = (1.2, 0.5)$ and $(\delta_{\max}, \delta_{\min}) = (50, 10^{-6})$.

Hypterparameters were tuned manually, based on k-fold evaluation performance (see next subsection). The optimal learning rate γ (GD/Adam) and initial step-size δ_0 (Rprop) for logistic and Hidden30 models are indicated in Tables (3, 4, 5) in Sec. 5. For Casper, the learning rates were set as $(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)}) = (0.01, 0.003, 0.0001)$ for Adam and $(\gamma^{(1)}, \gamma^{(2)}, \gamma^{(3)}) = (0.1, 0.01, 0.001)$ for GD. The initial step sizes were set as $(\delta_0^{(1)}, \delta_0^{(2)}, \delta_0^{(3)}) = (0.2, 0.005, 0.001)$ for Rprop.

For consistency, all model parameters were initialised using uniform Xavier (Glorot) initialisation [2], including fan-in weights for successive CCN neurons. The single weight connecting each new hidden neuron to the output was intialised to zero.

4.2 Evaluation

Following Ref. [6] we compare model performance based on a leave-one-subject-out cross-validation process, and measure prediction quality according to the five metrics in Table 2.

Table 2. Measures of prediction quality.

Feature	Description
Accuracy	Fraction of predicted labels which are correct.
Precision (positive predictive value)	Fraction of predicted-stressed which are true-stressed.
Sensitivity (true positive rate)	Fraction of true-stressed which are predicted-stressed.
Specificity (true negative rate)	Fraction of true-calm which are predicted-calm.
F1-score (F-score, F-measure)	Harmonic mean of precision and sensitivity.

Each model was trained for between 300 and 2000 epochs[2] and its test-participant predictions recorded periodically. Five runs were performed for each fold. Random-number generation was initialised with a distinct deterministic seed for each run and fold; for fairness, the same set of seeds was used for each model. For Casper, the initialisation weights of successive neurons were decoupled from the duration of each training stage.

Prediction quality, loss, and training cost were then accumulated over all folds and runs at each epoch. Except for accuracy, the metrics in Table 2 are not linear with respect to the confusion matrix; in this case the confusion matrix was first accumulated over all folds and runs, and each metric computed from the result. Due to the small size of the test set for each fold (1 participant, i.e. 6 samples), this produces a more faithful representation of expected model performance than taking the average of the metrics for individual folds and runs.

Sec. 5 presents the best average prediction quality of each model as well as the (first) epoch and time at which this was achieved. This was in most cases unambiguous; however, a small number of conflicts between different prediction metrics were resolved by taking the minimum of the (average) test loss. Time and epoch serve only as imperfect measured of training cost, as runtime is influenced by the machine's background processes and cost-per-epoch varies with architecture and optimiser. As proxies for parameter-count and feedforward-evaluation cost, we report also the average number of hidden neurons at this optimal epoch. This is particularly natural for comparison with Casper, whose size is not static. All training and evaluation was performed using the Pytorch framework on macOS 11.7 (64-bit) with a 1.4 GHz Dual-Core Intel Core i5 processor and 4GB 1600 MHz DDR3 RAM.

[2] depending on its convergence speed

5 Results and discussion

This section presents the top-performing hyperparameter variant of each architecture with each optimiser. Models were trained and evaluated the full set of 210 features, both with and without outlying participants (Tables 3 and 4, respectively). Models were subsequently run on a smaller dataset (Table 5) excluding outliers and consisting of only the 15 features from brain region F7, chosen as it appears frequently in the list of top features reported in the original study (Table II in [6]). All presented models achieve 100% training accuracy on all three datasets, and generally achieve average test-set accuracy above 95%. Rprop is consistently the fastest and most accurate optimiser for each model, with the exception of Hidden30 Adam in Table 4. The remainder of this section discusses: the effect of dataset choice; the relative merits of Casper and Hidden30; comparison with the original study; and remaining limitations.

Table 3. Model performance with all features (outlier participants included)

	Model training					Prediction quality (%)				
Architecture	Optimiser		Epochs	Time (s)	Size (hid.)	Acc.	Prec.	Sens.	Spec.	F1
Logistic	GD	lr=0.1	140	0.8	0	95.7	98.2	93.1	98.3	95.6
	Rprop	$\delta_0 = 0.01$	100	0.7		97.1	97.2	96.9	97.2	97.1
	Adam	bs=12; lr=0.001	90	1.5		95.7	99.1	92.2	99.2	95.5
Hidden30	GD	lr=0.1	330	1.8	30	95.4	94.6	96.4	94.4	95.5
	Rprop	$\delta_0 = 0.01$	50	0.3		95.4	95.8	95.0	95.8	95.4
	Adam	bs=12; lr=0.001	130	2.4		95.1	94.0	96.4	93.9	95.2
Casper	GD		750	0.9	2.0	94.4	99.1	89.7	99.2	94.2
	Rprop	See Sec. 4	60	0.2	0.0	97.8	96.7	98.9	96.7	97.8
	Adam		175	0.3	0.4	93.1	98.1	87.8	98.3	92.7

Table 4. Model performance with all features (outlier participants excluded)

	Model training					Prediction quality (%)				
Architecture	Optimiser		Epochs	Time (s)	Size (hid.)	Acc.	Prec.	Sens.	Spec.	F1
Logistic	GD	lr=0.1	150	0.7	0	97.9	100.0	95.8	100.0	97.9
	Rprop	$\delta_0 = 0.01$	50	0.3		97.9	100.0	95.8	100.0	97.9
	Adam	bs=12; lr=0.001	650	7.6		97.9	100.0	95.8	100.0	97.9
Hidden30	GD	lr=0.1	300	1.7	30	97.9	97.9	97.9	97.9	97.9
	Rprop	$\delta_0 = 0.01$	50	0.3		97.9	99.7	96.1	99.7	97.9
	Adam	bs=12; lr=0.001	1400	2.9		98.5	98.5	98.5	98.5	98.5
Casper	GD		500	0.6	0.0	97.6	100.0	95.2	100.0	97.5
	Rprop	See Sec. 4	70	0.2	0.0	99.5	100.0	99.1	100.0	99.5
	Adam		1000	1.8	0.95	95.8	100.0	91.5	100.0	95.6

Both simple logistic regression ('Logistic') and Casper slightly outperform Hidden30 in prediction quality on the dataset with outliers, by approximately 2% across all quality metrics. Removing outliers improves Casper and Logistic by 2-3% averaged over all metrics, with a slightly larger improvement for Hidden30 to make it comparable with the other two models. Surprisingly, the prediction quality of all models is improved by a further 1-2% by restriction of the dataset to just the brain region F7. This attests firstly to the importance of feature-selection, and in particular the importance of the frontal lobe for stress prediction as claimed in the previous study [6]. Casper does not provide a feature-selection advantage for the present task, gaining at most two hidden neurons (on average) and performing best with none; in the latter case Casper becomes equivalent to logistic regression. Casper and Logistic are comparable in prediction quality to Hidden30 on the full-featured outlier-excluded dataset, and outperform Hidden30 on the remaining two. In fact Casper and Logistic achieve perfect prediction accuracy on the F7 dataset. Together, the improved performance of simpler models and the general improvement on the F7 dataset suggest that predictive performance on this task is limited by either (1) optimisation complexity due to model size and feature count, or (2) overfitting due to the small size of the dataset. In either case, Casper provides a procedural advantage over Hidden30 in that it begins with the simplest network.

The original study [6] achieved 98.6% prediction accuracy using the Hidden30 architecture, presumably including outliers in its dataset. In the equivalent present case (see Table 3) Hidden30 achieves only 95.4%, while the top accu-

racy (97.8%; Casper Rprop) is only slightly below the original paper; note however that the original study performed optimisation with a different 150 features selected from a larger set, and used the Levenberg-Marquardt optimiser. Ref. [6] does not report precision, sensitivity, specificity or F1-score for the stress-calm (tensing-relaxing) dataset, but the values in Table 3 are comparable to those reported for other emotional measures (Table III in [6]). Subsequent datasets present an improvement over the original paper, with the top accuracies after outlier removal (99.5%; Casper Rprop) and F7-selection (100%; Casper/Logistic) exceeding the original 98.6%. Since feature-selection was also employed in the original study, its imperfect accuracy may be due to the presence of the outliers.

Table 5. Model performance on channel F7 (outlier participants excluded)

Model training						Prediction quality (%)				
Architecture	Optimiser		Epochs	Time (s)	Size (hid.)	Acc.	Prec.	Sens.	Spec.	F1
Logistic	GD	lr=0.1	500	2.5	0	99.8	100.0	99.7	100.0	99.8
	Rprop	$\delta_0 = 0.01$	**75**	**0.5**		**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
	Adam	bs=12; lr=0.001	670	9.8		99.2	98.8	99.7	98.8	99.2
Hidden30	GD	lr=0.1	1000	4.7	30	98.5	98.8	98.2	98.8	98.5
	Rprop	$\delta_0 = 0.01$	110	0.7		98.9	100.0	97.9	100.0	98.9
	Adam	bs=12; lr=0.001	500	7.6		98.6	97.9	99.4	97.9	98.6
Casper	GD	See Sec. 4	750	0.9	3.7	97.7	100.0	95.5	100.0	97.7
	Rprop		100	0.2	0.0	100.0	100.0	100.0	100.0	100.0
	Adam		300	0.3	0.0	98.3	98.5	98.2	98.5	98.3

The results presented above remain subject to several major limitations. Despite the use of multi-run k-fold cross-validation, the dataset remains small[3] and hence results subject to considerable statistical error and bias. The small dataset also means that reported training costs are significantly influenced by environmental or implementation factors rather than the computational complexity of each algorithm; reported time and epoch are thus an inaccurate indication of training cost on larger datasets. In reporting model performance we have accordingly prioritised maximum achievable prediction quality; however, for larger datasets it may prove useful to examine the prediction-quality/training-cost tradeoff more systematically, by comparing accuracy at a fixed training cost or vice versa.

There also remains significant potential for further optimisation of the presented models. A systematic hyper-parameter search for each model would certainly improve results over manual tuning. This is particularly true of Casper with its three distinct learning rates, which are infeasible to tune simultaneously by hand. Additional parameters such as weight decay and momentum could also be optimised.

6 Conclusion

This paper considered the potential benefits of Casper [10] as a constructive alternative to the static network architecture (Hidden30) previously used to predict stress based on EEG signals [6]. We find that manual outlier removal and feature selection permits simple logistic regression to achieve perfect classification accuracy. Hidden30 is in general outperformed by logistic regression, implemented either explicitly or via Casper with zero hidden neurons. Since Hidden30 is strictly more expressive, this is likely due to overfitting or excessive optimisation complexity; Casper is thus advantageous as it begins with the simplest possible network. The prediction of stress from other indicators such as electrodermal activity, blood volume pulse, skin temperature or pupil dilation [7] may therefore benefit from a Casper-like constructive architecture. A significantly larger dataset and more systematic optimisation is required to firmly establish these results.

References

1. Fahlman, S., Lebiere, C.: The cascade-correlation learning architecture. In: Touretzky, D. (ed.) Advances in Neural Information Processing Systems. vol. 2. Morgan-Kaufmann (1989)
2. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Teh, Y.W., Titterington, M. (eds.) Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research, vol. 9, pp. 249–256. PMLR, Chia Laguna Resort, Sardinia, Italy (13–15 May 2010)
3. Katmah, R., Al-Shargie, F., Tariq, U., Babiloni, F., Al-Mughairbi, F., Al-Nashash, H.: A Review on Mental Stress Assessment Methods Using EEG Signals. Sensors **21**(15) (2021). https://doi.org/10.3390/s21155043

[3] 144 examples split across 24 participants

4. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015), http://arxiv.org/abs/1412.6980
5. O'Connor, D.B., Thayer, J.F., Vedhara, K.: Stress and health: A review of psychobiological processes. Annual Review of Psychology **72**(1), 663–688 (2021). https://doi.org/10.1146/annurev-psych-062520-122331
6. Rahman, J.S., Gedeon, T., Caldwell, S., Jones, R.: Brain melody informatics: Analysing effects of music on brainwave patterns. In: 2020 International Joint Conference on Neural Networks (IJCNN). pp. 1–8 (2020). https://doi.org/10.1109/IJCNN48605.2020.9207392
7. Rahman, J.S., Gedeon, T., Caldwell, S., Jones, R., Jin, Z.: Towards effective music therapy for mental health care using machine learning tools: Human affective reasoning and music genres. Journal of Artificial Intelligence and Soft Computing Research **11**(1), 5–20 (2021). https://doi.org/doi:10.2478/jaiscr-2021-0001
8. Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: the RPROP algorithm. In: IEEE International Conference on Neural Networks. pp. 586–591 vol.1 (1993). https://doi.org/10.1109/ICNN.1993.298623
9. Schneiderman, N., Ironson, G., Siegel, S.D.: Stress and health: Psychological, behavioral, and biological determinants. Annual Review of Clinical Psychology **1**(1), 607–628 (2005). https://doi.org/10.1146/annurev.clinpsy.1.102803.144141
10. Treadgold, N.K., Gedeon, T.D.: A cascade network algorithm employing progressive rprop. In: Mira, J., Moreno-Díaz, R., Cabestany, J. (eds.) Biological and Artificial Computation: From Neuroscience to Technology. pp. 733–742. Springer Berlin Heidelberg, Berlin, Heidelberg (1997)
11. Treadgold, N., Gedeon, T.: A Simulated Annealing Enhancement to Resilient Backpropagation. In: Proc. Int. Panel Conf. Soft and Intelligent Computing. pp. 293–298 (1996)
12. Treadgold, N., Gedeon, T.: Exploring constructive cascade networks. IEEE Transactions on Neural Networks **10**(6), 1335–1350 (1999). https://doi.org/10.1109/72.809079