A Bio-inspired model for audio processing

Anonymous Author1
Anonymous Department
Anonymous Institution
Anonymous City, Country
anonymous1@email.com

Anonymous Author2
Anonymous Department
Anonymous Institution
Anonymous City, Country
anonymous2@email.com

***Abstract*—Homeostatic Activity Dependant Structural Plasticity (HADSP) is a recently introduced technique to generate network using structural plasticity. The algorithm use only homeostatic plasticity but let emerge principles of Hebbian learning. A previous study suggested that HADSP was able to generate networks that effectively leverage the inter-relationships of the network's inputs but the idea was tested only on simple benchmarks.**

This paper investigates how HADSP performs on specific tasks in speech recognition. In these tasks, a single-variable time series is turned into a multi-variable time series using audio processing. The primary focus of our approach was not necessarily to develop an optimal algorithm, but to explore and elucidate a set of techniques and concepts with the potential to enhance existing algorithms and stimulate conversations on speech dataset processing.

Our principal results are that using spectral representation of the audio signal greatly improves the performance of speech recognition for the tested dataset as well as enhancing the capabilities of HADSP in generating better ESN connectivity.

I. Introduction

Structural plasticity, the ability of neurons to create new connections, is a fundamental mechanism in the formation and maturation of biological neural circuits [1] [2] [3] but it is essentially absent in standard artificial neural networks. In the context of echo state networks (ESNs), the rate-based version of reservoir computing, connections between neurons are generated randomly, an approach that has been described as "the antithesis of the 'optimal'" approach [4].

In a previous study [5], it was demonstrated that the Homeostatic activity-dependent structural plasticity (HADSP) algorithm was capable of generating reservoir architectures that are adapted to a particular Mackey-Glass benchmark [6] and showed better performance on this benchmark. Those improvements were not due to hyperparameter tuning and showed that HADSP can improve performance on specific types of data by exploiting the correlation in the band-filtered version of the dataset and using it to improve performance. In this paper we aim to first generalize this technique and extend the capabilities of the HADSP algorithm to a benchmark speech dataset.

This project has received funding from a research and innovation program under a grant agreement No XXXXXX.

To do so we use a type of spectral density-based time series synthesis to generate a multivariate time series from a univariate one. This process mimics the way the human ear encodes sound waves and variation of it have already been used widely in the Deep learning field [7].

In what follows, we first present the necessary background for echo state networks and describe the rate-based HADSP as well as its main properties (section II). In section III we also introduce the main elements of our experiment and the techniques used to generate the multivariate dataset from the univariate speech dataset. In section IV we test the performance of the different parts of our framework on the Free Spoken Digits Datasets [8] and discuss the results.

II. Background

A. Echo state networks

Testing reservoir computing capabilities on a given architecture can be interesting because it may give insight into the capacity of a given architecture to support certain types of computations, particularly those involving temporal dynamics or complex patterns. This can be especially valuable when trying to understand how an architecture might perform with different types of recurrent tasks, how effectively it can handle temporal data, and how it might be optimized for different use cases such as time series prediction, chaotic systems modeling or in our case speech recognition.

An illustration of the Echo state networks framework [4] [9] [10] is given in Fig. 1. It normally consists of a a randomly initiated recurrent neural network of n neurons (the *reservoir*), and a trained *readout layer*

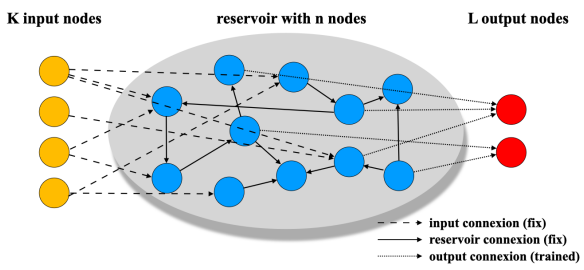

Fig. 1. Schematic architecture of reservoir computing

that creates the outputs as a linear combination of the reservoir neuron states.

The evolution of the vector states $\mathbf{x}$ of the n neurons is determined by the interactions between the connection matrix $\mathbf{W}$, the input matrix $\mathbf{W_{in}}$, the input $u[t]$, the bias vector $\mathbf{b}$ and the activation function σ, combined in the following equation :

$$\mathbf{x}[t+1] = \sigma(\mathbf{W} \times \mathbf{x}[t] + \mathbf{W_{in}} \times u[t]) + \mathbf{b} \quad (1)$$

We take σ as the hyperbolic tangent function. The purpose of the bias term $\mathbf{b}$ is to enrich the dynamics of the network, it is however kept low enough to not be preponderant. $\mathbf{b}$ is a $1 \times n$ vector and dimension of the matrices are $n \times n$ for $\mathbf{W}$ and $1 \times n$ for $\mathbf{W_{in}}$. In ESNs, these weights are typically randomly initialised from a chosen distribution. However, as we explain below, HADSP uses a different weight initialisation scheme where these weights evolve during operation according to a self-organised local rule.

Once $\mathbf{W}$, $\mathbf{W_{in}}$ and $\mathbf{b}$ are fixed, the desired output is obtained through the following equation :

$$y[t+1] = \mathbf{W_{out}} \times \mathbf{x}[t+1] + \mathbf{b_{out}} \quad (2)$$

where $\mathbf{W_{out}}$, $\mathbf{b_{out}}$ are learned using ridge regression such that the generated train output, $y[t]$, optimally approximates a desired target output, $y^{target}[t]$ [4].

B. The HADSP algorithm

1) Growing rule: HADSP uses a biologically inspired *growing rule* to determine the evolution of the network connectivity. The rule is derived from biology/computational neuroscience context, where the calcium concentration in the soma of neurons is thought to be proportional to the firing rate and to potentially impact the growth of dendrites and axons [2] [3]. In this model, neurons have multiple synapses of fixed weight. Similarly, in our *rate based* algorithm, we consider that neurons can have zero, one or multiple fixed weight connections between them. For k connections from node i to node j, the value $w_{ij} = k \times 0.05$. Self-connections are not allowed in our model, so $w_{jj} = 0$.

We limit the maximum number of synapses between two neuron γ, meaning that we always have $k \leq \gamma$ and that each connection $w_{ij} \leq k \times \gamma$.

The value of weight increment influences the regime's dynamics. This parameter should be kept low enough to allow the HADSP to recombine the inputs without creating an unstable regime. In the context of our experiments, a weight increment value of 0.05 was found to be optimal.

Every ΔT steps, a growth indicator, $\mathbf{\Delta z}$, is calculated for each neuron according to the following formula :

$$\mathbf{\Delta z} = -(\langle \mathbf{x} \rangle - \rho) \div \beta \quad (3)$$

where $\langle \mathbf{x} \rangle$ is the average of the state vector $\mathbf{x}$ over the increment period. $\mathbf{\Delta z}$ depends on two hyperparameters : the growth parameter β and the target rate ρ.

Based on the resulting value of Δz_j of the j-th neuron, one input connection is added if $\Delta z_j < -1$ or pruned if $\Delta z_j > +1$.

Thus, when a neuron's activity falls below the -1 threshold, a new connection is established to increase its activity. The creation of new connections is restricted to neurons that have been identified as requiring additional connections. Among all those possible partners, one is selected randomly to create a new connection. Similarly, when a neuron's activity exceeds the +1 threshold, an existing connection is pruned in order to decrease its activity. Note that the pruning of connections is performed independently of the state of the neuron's partners, regardless of whether they also need to decrease their activity or not. In other words, it does not take into account the relation between the activities of the neurons pairs, which differentiates it from Hebbian mechanisms.

2) On the need of excitatory neurons: In the brain, neurons are broadly classified into two types: excitatory and inhibitory. Excitatory neurons are those whose increased activity contributes to the increased firing rate of the post-synaptic neurons, thereby promoting signal transmission. Conversely, inhibitory neurons decrease the firing rate of post-synaptic neurons, reducing the signal transmission.

The algorithm is not effective in the presence of mixed inhibitory and excitatory neurons. Indeed, since it aims to regulate the firing rate of neurons through the creation and pruning of connections, it requires a clear understanding of the effect of input on neuron activity: in order to increase the firing rate of a neuron, it is necessary to determine whether the input will have a positive or negative impact on the rate.

The easiest option to ensure this is to make sure each neuron has a strictly positive or negative output to properly guide the process. Consequently, the HADSP algorithm uses only excitatory neurons, this choice limits the computational properties of the ESN, in order to maintain the desired level of interpretability and functionality in the algorithm.

3) Hebbian like plasticity: The HADSP algorithm, although being an homeostatic algorithm, implements a form of Hebbian plasticity. It was shown in previous work [5] that neurons under the rule of HADSP create connections based on their previous correlated overstimulation. This outcome shows that our algorithm, although purely homeostatic, displays a form of Hebbian learning. This Hebbian mechanism can improve network performance: when presented with inputs that are similar, the network can combine those inputs to create new and more useful dynamics.

4) Interpretation of the rule and convergence of the algorithm: For each neuron i, the associated growth

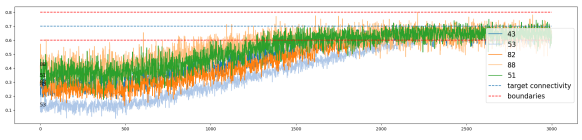

Fig. 2. Evolution of the activity upon the effects of the activity-dependant homeostatic plasticity for 5 randomly selected neurons. The blue dotted line represents the target rate ρ. The activation value of the neurons is constrained between the red lines of value $\rho + \beta$ and $\rho - \beta$

indicator Δz_i is linked to its current firing rate x_i.

$$\Delta z_i = \frac{1}{\beta}(\rho - x_i) \tag{4}$$

In order to reach equilibrium, it is necessary for the growth indicator to satisfy the condition $\Delta z_i \in]-1, 1[$, which translates mathematically to the following constraint for equilibrium.

$$x_i \in]\rho - \beta, \rho + \beta[\tag{5}$$

Therefore, the equilibrium condition of the algorithm can be directly related to its hyperparameters β and ρ. This is illustrated in Fig. 2 where the HADSP influence causes the activity of neurons to fall within the specific range that we mentioned.

In some instances, the equilibrium of activity cannot be reached due to the narrow range of the equilibrium area, causing the network to continuously oscillate between over and under the threshold of a suitable activity range through quick pruning and adding of connections, leading to an unstable regime.

Generally, if the algorithm fails to converge, the network undergoes constant reorganization, leading to a changing dynamic over time. As a result, for a given input time series, the network will not form a stable representation of the inputs, making it impossible to train using the Ridge regression method.

In addition to the target rate, ρ, and the growth parameter, β, hyperparameters include the maximum number of partners γ, the number of step ΔT at which the . γ does not limit the number of connections but rather their distribution. For the rest of the study we set $\gamma = 12$.

III. Experimental setup

Our approach was not to create a high-performance algorithm. Rather, we sought to investigate how this techniques and concepts could enhance existing algorithms and invigorate conversations on speech dataset processing. Our principal contribution lies in the systematic evaluation of transforming audio signals into multivariate time series for their application in Echo State Networks (ESN). Concurrently, we critically assess the performance and adaptability of the Homeostatic Activity-Dependent Structural Plasticity (HADSP) algorithm when applied to speech datasets. These investigations strive to uncover novel insights into the applicability of these techniques in the field of speech processing and artificial intelligence research.

A. Dataset used

In this paper, we use the Free Spoken Digit Dataset (FSDD) [8]. It is a simple audio/speech dataset consisting of recordings of spoken digits in wav files at 8kHz. The recordings are trimmed so that they have near-minimal silence at the beginnings and ends. It contains data from 6 speakers and 3,000 recordings. The task is to classify a sequence into 10 possible digits, hence the output space is 10-dimensional. The label is the index of the output neuron with maximum activation.

The dataset was divided into two subsets of equal size (n=1500 each), namely a training set and a test set. Care was taken to stratify the split on the basis of speakers, resulting in each subset containing exclusive instances of three distinct speakers. This stratified approach ensures that the model's performance evaluation is unbiased, as no speaker appears in both subsets.

B. Pretraining dataset

In previous work it was shown that HADSP perform best when confronted to multivariate equivalent of time series that was extracted by applying several band pass filter on the original time series. Our contribution here is the automation of this process that was previously done extracting the values of the band pass filter by looking at the spectrum of the signal.

In situations where the dataset is primarily univariate, a transformation into a multivariate structure can be achieved via spectral density analysis of the training dataset. The frequencies corresponding to power peaks are identified and employed as the basis for creating bandpass filters. Each filter is centered around a specific frequency with the low and high cutoff frequencies derived as half the lower and upper peak frequencies respectively. The application of these $n_{component}$ bandpass filters to the dataset generates an $n_{component}$-dimension time signal.

1) Spectral Density Analysis: Our approach starts with the estimation of the power spectral density (PSD) of the time series data (specifically the Free Spoken Digits Dataset in this context) using Welch's method via the `signal.welch` function from the scipy library. Subsequent identification of the PSD peaks is accomplished with `signal.find_peaks`, thereby facilitating the extraction of the associated frequencies and power values. The calculation of the power threshold, established as 1e-5 times the maximum power, enables the filtering of frequencies associated with powers exceeding this threshold. This action effectively eliminates weak PSD peaks potentially attributable to noise.

2) Application of Butterworth Filter: Following the identification of power peaks, we proceed with the creation of bandpass filters, each centered around a specific peak frequency. These filters are then applied to the original dataset, yielding an $n_{component}$ dimension time signal, or a multivariate time series, comprising the filtered time series corresponding to each frequency.

3) Scaling: Finally, data scaling is conducted using sklearn's MinMaxScaler to normalize or rescale the filtered time series, bounding the range within -1 to 1. This measure ensures data range uniformity, thereby enabling the HADSP model to better discern dependencies within the datasets.

C. Important characteristics of the reservoir

The bias vector $\mathbf{b}$ is sampled randomly from a normal distribution of mean 0.1 and standard deviation 0.1 [4]. Key characteristics to the reservoir behavior include the spectral radius of $\mathbf{W}$ and the scaling of $\mathbf{W_{in}}$.

The *spectral radius* is the largest eigenvalue λ_i of the weight matrix :

$$sr(\mathbf{W}) = \max_{i=1}^{n} |\lambda_i| \tag{6}$$

It influences the rate at which information decays or amplifies in the reservoir over time. Proper selection of the spectral radius ensures that the reservoir is capable of producing good output predictions. In practice, $sr(\mathbf{W})$ is set to maximize performance on a task, usually to a value around 1.

Input scaling of the weight matrix $\mathbf{W_{in}}$ is important in echo state networks because it affects the dynamic range of inputs that are fed into the network. A properly scaled input ensures that the inputs are within a range that allows the network to capture the relevant information from the input signal. If the input is not scaled properly, then the network may become saturated, leading to distorted and inaccurate output.

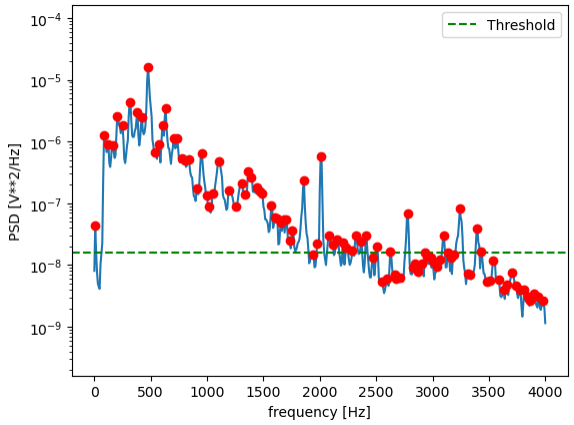

Fig. 3. Typical power spectral density using Welch's method with peak identified as red dots and threshold used to selected the frequencies

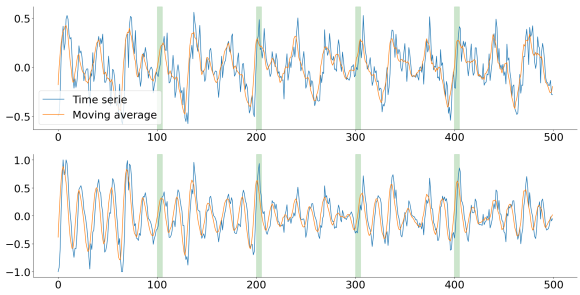

Fig. 4. Typical time series and moving average obtained by applying a band filter on the inputs, the green stripes are to represent the size of ΔT on which HADSP is calculated

In our context, input scaling is twice as important because if the magnitude of the inputs is too high, the network activation will already be too high and our algorithm will not be able to combine them to reach the desired target activation. To define the scaling of the inputs, we try to keep the perceived input at a maximum amplitude, that is 3 times lower than the target rate of our HADSP algorithm.

D. Supervised training of W_{out} and inference for classification task

The subsequent procedure in our methodology focuses on the computation of the output weight matrix, denoted as W_{out}, for classification tasks.

During this phase we expose the network to an input sequence and allow it to evolve autonomously. Once each sequence reaches its termination point, we gather the resulting state of the reservoir. To ensure non-interference between the network's responses to various sequences, we reset the network state following the conclusion of each sequence. This sequence-to-state process is repeated for all given sequences, resulting in a $1500 \times n$ matrix that embodies the final network states post-sequence execution.

This resulting matrix operates as the output vector which forms the basis of the subsequent linear regression stage. The linear regression maps the states to 10 output neurons, each associated with a distinct class. The class corresponding to the neuron with the maximum activation is the predicted output of the network for the input sequence.

We employed a ridge regression algorithm, with the regularization parameter to $r = 10^{-7}$, a value that has been typically associated with successful outcomes in the literature. [4]

The inference phase follows a parallel approach, wherein each sequence is permitted to execute until completion. The model's prediction corresponds to the output layer neuron exhibiting the maximum activation.

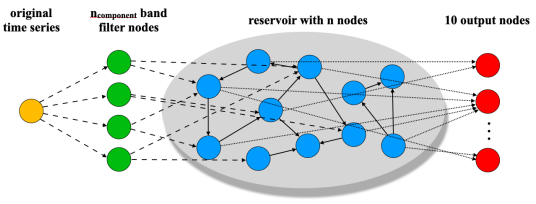

Fig. 5. Setup for our experiments for the classification architecture with the band filtered inputs

IV. Results and Discussion

In the execution of this study, we carried out the implementation of the Homeostatic Activity-Dependent Structural Plasticity (HADSP) algorithm. For the training and inference of the reservoir, the ReservoirPy library was employed [11].

A. Evaluation for different spectral radius of the framework

To condition the network, a pre-training set was utilized, which consisted 20 concatenated instances derived from the training dataset.For this experiment and for the purpose of accurately capturing the correlation within the time series, we selected a small increment of value $\Delta T = 5$ for our rule. Consequently the network was feed the pretraining set and every 5 steps HADSP was used to initiate, add or prune connections. Our choice of hyperparameters was dictated by the need for stability and convergence; $\rho = 0.8$, $\beta = 0.2$, $\gamma = 12$ and input scaling was set to 0.3 were chosen to assure that the algorithm would reliably converge towards a stable representation of the input. This implies that the connection formation and pruning process would eventually stabilize, leading to a healthy network dynamic.

We trained the ESN and calculated the W_{out} matrix in a supervised manner as explained in section III-D.

We compare this first network with matrix $\mathbf{W_{HADSP}}$ against a additional networks $\mathbf{W_{random}}$ of equal size, $n = 294$, with same connectivity but whose with weights sampled from a uniform distribution between 0 and 1. We scale the $\mathbf{W_{random}}$ and the $\mathbf{W_{HADSP}}$ matrices so that their spectral radius are equal to 0.9, 1, 1.1 and 1.2. For the two networks the $\mathbf{W_{in}}$ input matrix is taken from a normalize distribution of mean 1 and standard deviation 0.5 with connectivity of 0.5. For the two networks the $\mathbf{W_{in}}$ input matrix is taken from a normalize distribution of mean 1 and standard deviation 0.5.

The results in Table I show that for the selected growth and target rate parameters the HADSP algorithm demonstrated superior performance to the other randomly generated network on multivariate datasets. However those results where not always obtain for all combination of growth and target rate, and it remains unclear how to tweaks those hyperparameters so that it always succeed and why we obtain such a drastic change between the two different values.

B. Influence of the multivariate conversion

We also compare those multivariate network to their univariate counterpart using this time again two reservoirs: One of connectivity matrix $\mathbf{W_{HADSP}}$ generated with HADSP by the same methods as the previous one with $\rho = 0.8$, $\beta = 0.1$, $\gamma = 12$, and the increment of the algorithm set to 5 steps and another one $\mathbf{W_{random}}$ of equal size, connectivity and spectral radius but randomly instantiated. Once again, when the plastic unsupervised learning process has been completed, we let the network self-evolve with each input sequence. After each sequence completes, we collect the final reservoir state and reset the network to avoid interference. Repeating this for all sequences generates a $1500 \times n$ matrix representing the final network states post each sequence. This matrix serves for the ridge regression to calculate the output weight matrix W_{out}.

Upon evaluation summarized in II, we observe that the conversion of the dataset to its multivariate form significantly enhances the performance metrics, while maintaining identical reservoir dimensions. This suggests that multivariate representation provides a more efficient and robust platform for the dataset under analysis. There could be several reasons for this.

- *Capturing Interactions:* Multivariate datasets can capture interactions between different variables that might be missed in univariate data. For example, if two features have a combined effect on the target variable, a model trained on the multivariate data can potentially learn this relationship.

TABLE I
POURCENTAGE OF CORRECTLY CLASSIFIED INSTANCE, FOR FREE SPOKEN DIGITS DATASET DATASET USING MULTIVARIATE DATASET FOR A NETWORK GENERATED RANDOMLY AND ANOTHER THROUGH HADSP AND FOR DIFFERENT SPECTRAL RADIUS. THE PERFORMANCE WAS AVERAGE OVER 3 TRIAL

Model	SR	HADSP + multi	random + multi
FSDD	0.9	17.267%	17.267%
	1	17.533%	16.867%
	1.1	17.333%	16.867%
	1.2	17.333%	17.400%

TABLE II
POURCENTAGE OF CORRECTLY CLASSIFIED INSTANCE, FOR FREE SPOKEN DIGITS DATASET DATASET USING UNIVARIATE TIME SERIES FOR A NETWORK GENERATED RANDOMLY AND ANOTHER THROUGH HADSP AND FOR DIFFERENT SPECTRAL RADIUS. THE PERFORMANCE WAS AVERAGE OVER 3 TRIAL

Model	τ	HADSP + uni	random + uni
FSDD	0.9	9.267%	9.467%
	0.99	9.267	9.400
	1.1	9.067	9.467
	1.2	9.667	9.467

TABLE III
HYPERPARAMETER SEARCH VALUES

Hyperparameter	min value	max value	step size
spectral_radius	0.8	1.2	0.05
target_rate	0.6	1.0	0.1
growth_parameter	0.1	0.3	0.1
input_scaling	0.1	0.3	0.05

TABLE IV
COMPARISON OF HADSP AND RANDOM NETWORK PERFORMANCES

Metric	HADSP	Random
Accuracy	18.9%	18.4%

- *Higher Dimensionality:* The higher dimensionality of a multivariate dataset can provide more information for the model to learn from. In reservoir computing, this can lead to a higher dimensional reservoir state space, which can capture more complex dynamics in the input data.
- *Noise Reduction:* Upon our multivariate extraction we selected the peak frequencies given a certain threshold. If the dataset contains noise, this transformation might help in reducing the impact of the noise on the model's performance.

C. Best performing network

In the pursuit of an optimized performance of our algorithm, we utilized Optuna, a versatile hyperparameter optimization framework. The chosen parameters under examination included the following hyperparameters: spectral radius, target rate, growth parameter, and input scaling. These parameters were subjected to a hyperparameter search, exploring their respective ranges as specified in III. Similarly, the hyperparameter search was conducted for the randomly instantiated reservoir with the spectral radius and the input scaling.

The best results, summarize in table IV show that HADSP generated network performs slightly better than it's randomly instantiated counterpart.

While the results offer an initial insight into how well the algorithm works, they might not entirely capture all the abilities of the ESN. The small size of the reservoir used in our experiment and our decision to leave out complex, multi-layered reservoir computing structures could downplay how effective and advantageous it might be to use a larger, multi-layered ESN setup.

V. CONCLUSION

In this study, we introduced a novel bio-inspired framework designed specifically for speech datasets. This framework utilizes a transformation of audio signals into multivariate time series using a spectral density-based time series synthesis and the HADSP algorithm, allowing the model to take advantage of the correlations among the various time series.

Our empirical findings highlight that when two networks are engaged with either the Free Spoken Digits Dataset time series or an equivalent representation involving multiple band-filtered Free Spoken Digits Dataset inputs, the latter consistently exhibits superior performance. Our second finding is that under certain conditions HADSP efficaciously exploits the interrelationships of network inputs to manifest richer dynamics but that finding the right set of hyperparameters to do so is not straighforward.

In the pursuit of future research, we aim to delve deeper into the conditions that optimize the performance of the algorithm and seek to elucidate the underlying mechanisms driving these enhancements, particularly focusing on the synergistic interplay between Hebbian learning and homeostatic mechanisms. We remain steadfast in our ambition to broaden the scope of datasets the HADSP algorithm can be applied to and conduct a meticulous analysis of the resultant networks, to further understand the strengths and limitations of the algorithm. Ultimately, our goal is to establish a comprehensive understanding of the intertwining relationship between Hebbian learning and homeostasis and explore how this intricate balance can be harnessed within artificial contexts.

REFERENCES

[1] Markus Butz, Florentin Wörgötter, and Arjen van Ooyen. Activity-dependent structural plasticity. *Brain Research Reviews*, 60(2):287–305, may 2009.

[2] Markus Butz and Arjen van Ooyen. A simple rule for dendritic spine and axonal bouton formation can account for cortical reorganization after focal retinal lesions. *PLoS Computational Biology*, 9(10):e1003259, oct 2013.

[3] Júlia V. Gallinaro and Stefan Rotter. Associative properties of structural plasticity based on firing rate homeostasis in recurrent neuronal networks. *Scientific Reports*, 8(1), feb 2018.

[4] Mantas Lukoševičius. A practical guide to applying echo state networks. In *Lecture Notes in Computer Science*, pages 659–686. Springer Berlin Heidelberg, 2012.

[5] Tanguy Cazalets and Joni Dambre. An homeostatic activity-dependent structural plasticity algorithm for richer input combination, 2023.

[6] Michael C. Mackey and Leon Glass. Oscillation and chaos in physiological control systems. *Science*, 197(4300):287–289, jul 1977.

[7] Maren Buermann and Tom A J P Van Meer. Speech recognition using very deep neural networks: Spectrograms vs cochleagrams, 2020.

[8] Zohar Jackson, César Souza, Jason Flaks, Yuxin Pan, Hereman Nicolas, and Adhish Thite. Jakobovski/free-spoken-digit-dataset: v1.0.8, 2018.

[9] Herbert Jaeger and Harald Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, 304(5667):78–80, apr 2004.

[10] Mantas Lukoševičius and Herbert Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149, aug 2009.

[11] Nathan Trouvain, Luca Pedrelli, Thanh Trung Dinh, and Xavier Hinaut. ReservoirPy: An efficient and user-friendly library to design echo state networks. In *Artificial Neural Networks and Machine Learning – ICANN 2020*, pages 494–505. Springer International Publishing, 2020.