POLYPUBLIE

Improved Alzheimer disease diagnosis with a machine learning approach and neuroimaging: Case study development. Supplément

Lilia Lazli

Document en libre accès dans PolyPublie
Open Access document in PolyPublie

Date	2025
Type	Article de revue/Article
Version	Matériel supplémentaire / Supplementary material
URL de PolyPublie PolyPublie URL	https://publications.polymtl.ca/64723/
Évaluation par les pairs Peer reviewed	Révisé par les pairs / Refereed
Conditions d'utilisation Terms of use	Creative Commons Attribution 4.0 International (CC BY)
Référence Citation	Lazli, L. (2025). Improved Alzheimer disease diagnosis with a machine learning approach and neuroimaging: Case study development. JMIRx Med, 6, e60866 (13 pages). https://doi.org/10.2196/60866

Publication officielle
Official publication

Titre de la revue Journal title	JMIRx Med (vol. 6)
Maison d'édition Publisher	JMIR Publications Inc.
URL officielle Official URL	https://doi.org/10.2196/60866
Mention légale Legal notice	This is an open-access article distributed under the terms of the Creative Commons Attribution License (https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work, first published in JMIRx Med, is properly cited. The complete bibliographic information, a link to the original publication on https://med.jmirx.org/, as well as this copyright and license information must be included.

Ce fichier a été téléchargé à partir de PolyPublie, le dépôt institutionnel de Polytechnique Montréal
This file has been downloaded from PolyPublie, the institutional repository of Polytechnique Montréal
https://publications.polymtl.ca

Multimedia Appendix 3: FFNN

The FFNN computation process with mathematical foundations is explained as follows:

Activation function

We used the hyperbolic tangent sigmoid, which is a logistic function and ranges from 0 to 1.

$$y_{v_i} = 2/\left(1 + exp(-2 * v_i)\right) - 1 \tag{3.1}$$

where y_i is the output of the ith node (neuron) and v_i is the weighted sum of the input connections.

Training

Learning occurs by changing connection weights after each piece of data is processed, based on the amount of error in the output compared to the expected result.

We can represent the degree of error in an output node j in the n th data point (training example) by :

$$e_j(n) = d_j(n) - y_j(n) \tag{3.2}$$

where $d_j(n)$ is the desired target value for nth data point at node j, and $y_j(n)$ is the value produced at node j when the n th data point is given as an input.

The node weights can then be adjusted based on corrections that minimize the error in the entire output for the n th data point, given by:

$$\varepsilon(n) = \frac{1}{2} \sum_{output\ node\ j} e_j^2(n) \tag{3.3}$$

To change the hidden layer weights, the output layer weights change according to the derivative of the activation function (backpropagation of the activation function).