Protein secondary structure prediction using multilayer fully connected neural networks

Author: María Alejandra Ulloa
Master in Molecular Techniques in Life Sciences, Stockholm University, Sweden.

Introduction

Proteins serve as the foundational functional units within biological systems, constructed from amino acid chains that intricately fold into simple or highly complex structures. Notably, proteins exhibit four distinct levels of structural organization: primary, secondary, tertiary, and quaternary structures (1,2). These structures are not easy to identify and require multiple experiments, thus extensive efforts have been made to unravel protein structure from their constituent amino acid sequences. This report delves into the realm of secondary protein structures that yield 2D conformations such as α-helices, β-sheets, and coils.

Modern approaches in secondary structure prediction harness advanced computational techniques, prominently including Artificial Neural Networks (ANN). These methods have showcased remarkable prediction accuracies of up to 84%, employing diverse model architectures such as feedforward deep networks, convolutional neural networks (CNNs), and recurrent neural networks (RNNs) (3). Notably, a recent attempt (4) demonstrated that even simple architectures, like their three-layer dense feedforward network, achieved a commendable 78% accuracy. However, a need for further enhancement persists, advocating for exploration of various hyperparameters to attain superior performance. On that note, here we introduced a fully connected neural network model (**Figure 1)** that yields a 76% overall accuracy in the prediction of secondary structures from amino acid sequences.

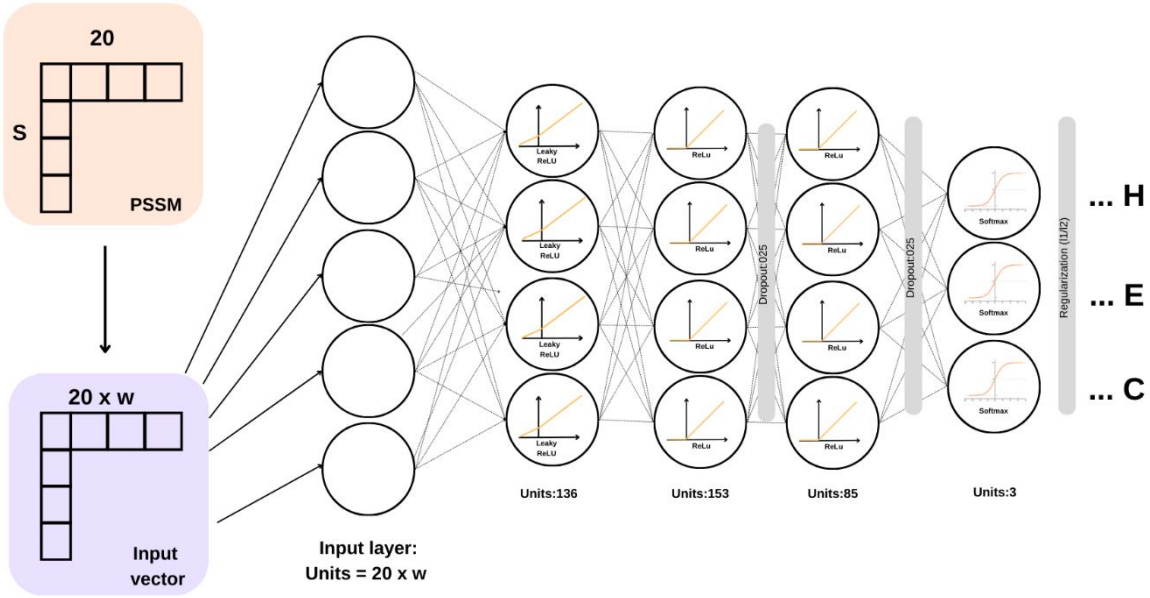


Figure 1. Multilayer feed forward neural network architecture.

Materials/methods

Datasets

Training, validation and blind-test datasets were obtained from (5) Github repository. In short, and as mentioned in her repository; the training set corresponds to proteins, originally taken from a work of (6), that had a resolution < 2.5 A, with residues in the 30-800 range, that display full domains, had DSSP information available, and full-profiles for a total of 1200 proteins with their corresponding Position Specific Scoring Matrix (PSSM) and DSSP assignment. The cross-validation set was generated by partitioning the training set in five equal folds (each one with 240 proteins). Lastly, the blind set comprised 328 sequences with their corresponding PSSM and DSSP.

Feature extraction

Extensive documentation within the literature mention the utilization of Position-Specific Scoring Matrices (PSSM) for protein structure prediction. A PSSM comprises a matrix with 20 columns representing amino acids and S rows corresponding to each residue in the sequence, where values denote the frequency of residues at each position providing evolutionary insights (7–10). To obtain positional information from the PSSM, a sliding window technique was employed to analyze the residue of interest alongside a window of neighboring residues. This entailed the creation of a new vector of dimensions 20 x W, where 'W' signifies the window size. An array containing all such vectors serves as input for the initial input layer of our neural network.

Network architecture

The model was built using Keras (12). We started with a simple network: a dense multilayer network with one input layer with an equal number of units corresponding to the width of the represented PSSM matrix. Using all the training and test data a Bayesian Optimization and Hyperband algorithms from Keras tuner (11) were used to obtain a first skeleton of an optimal architecture. Selecting for number of layers (maximum of five layers), number of units per layer (min:17, max:240), dropout layers and activation functions.

Performance optimization

Following the work of Drozdetskiy et., al (4) we focused on improving the initial network by tuning the window size for feature selection, number of epochs for training, activation functions (‘*leaky ReLU*’, *‘tanh’* and *‘ReLU’),* adding regularization techniques such as L1-L2 for weight penalization and dropout to add some randomness to the model to avoid overfitting. Batch normalization was also tested. To see all the process see: https://github.com/ma-ulloa/PSSP.

Results

Network architecture

The models chosen by Hyperband and the Bayesian algorithms were similar, but the latter was chosen for optimization **(Table 1).** Utilizing the cross-validation sets, we scrutinized different window sizes for feature extraction from the Position-Specific Scoring Matrix (PSSM). After evaluation, a window size of 17 emerged as optimal, yielding superior accuracy results across both training and validation sets **(Supplementary Figure 1a and 1b**). Furthermore, the number of epochs revealed that employing 10 epochs led to rapid overfitting, evidenced by pronounced accuracy deviations in both training and validation sets **(Supplementary Figure 1c).** Thus, 4 epochs were selected as the ideal training duration.

Table 1. Best model architecture based on Keras Tuning utils

Tuner	Bayesian		Hyperband	
Best model	activation:	ReLU	num_layers:	4
	dropout:	False	activation:	ReLU
	num_layers:	4	dropout:	False
	Input_layer:	340	Input_layer:	340
	units_0:	136	units_0:	136
	units_1:	153	units_1:	102
	units_2:	85	units_2:	204
	Output_layer:	3	Output_layer:	3
	Score: 0.7494206428527832		Score: 0.7489377856254578	

An *activityRegularization* layer incorporating L1 and L2 regularization was strategically introduced after the final layer, with experimentation revealing that a weight value of 1e-5 improved accuracy on the validation set (**Supplementary Figure 2**). While ReLU activation was employed across all layers, the implementation of *'leaky ReLU'* in the second layer notably improved the model's learning capacity and overall accuracy. Lastly, the incorporation of dropout layers within hidden layers further augmented model accuracy to a mean of 0,7592 (±0,0043) **(Figure 2).**

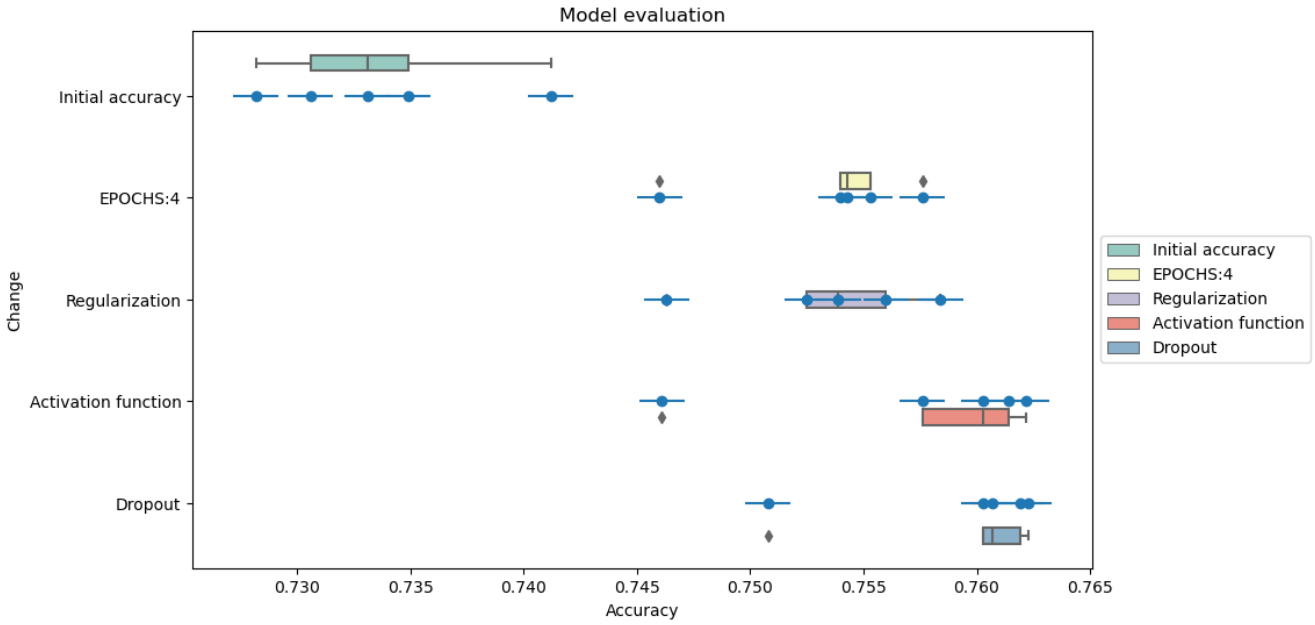


Figure 2. Model accuracy during the hyperparameter search across cross-validation sets (seen in blue) for the different changes implemented: change in the number of epochs, regularization techniques for the loss function in the last layer, addition of 'leaky ReLU' as an activation function and lastly the addition of dropout layers within hidden layers.

Network performance.

All cross-validation models were tested against a blind-set yielding an average accuracy of 76% (± 0.005) **(Supplementary table 2)**. A confusion matrix was calculated **(Figure 3 and supplementary figure 4**) from which scores were assessed for all predicted classes (H, E, C) **(Table 2)**. Helices prediction show better scores overall.

Table 2. Score metrics for predicted classes across cross-validation sets.

Class	Precision	SD	Recall	SD	F1	SD
H	0.82110142	0.0117517	0.80378451	0.01053579	0.81224197	0.00593778
E	0.72351835	0.01214003	0.60701534	0.00902925	0.66007415	0.00683902
C	0.72952938	0.00630561	0.80352486	0.008741	0.76467704	0.00257725

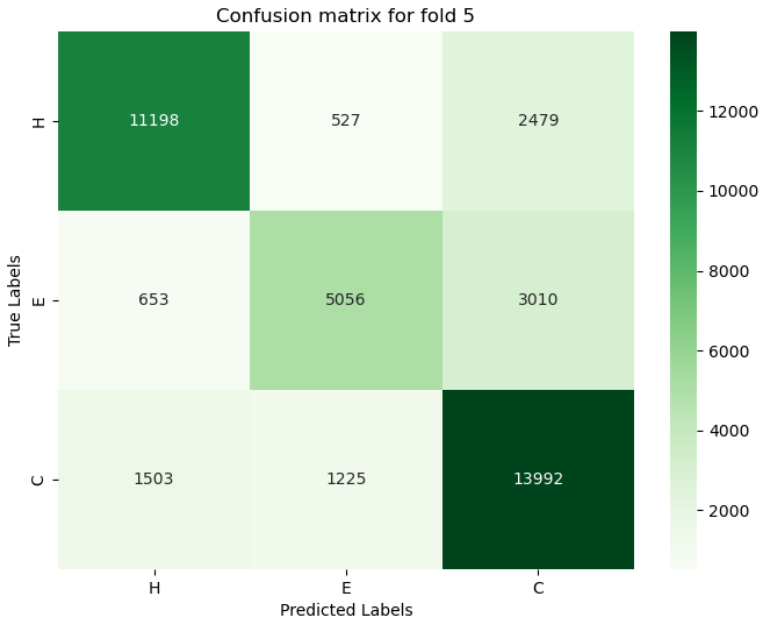


Figure 4. Confusion matrix for blind test using model of cross-validation train set number five as it is the one that shows better accuracy overall.

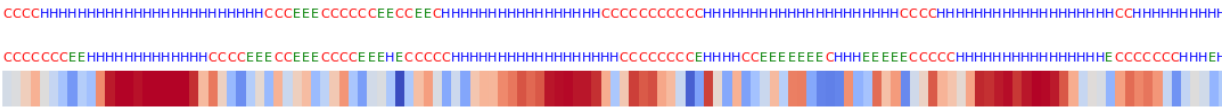


Figure 3. Expected versus predicted secondary structure for protein 4XU4_A of the blind set. Real structure (above) followed by the predicted structure and a heatmap showing the probabilities associated to every class. The sequence is not shown completely.

Conclusion/discussion

In conclusion, this project underscores the efficacy of regularization techniques, notably regularization with L1 and L2 and dropout, in mitigating overfitting during training, thereby enhancing model generalization going from an initial accuracy of 74% to 76% accuracy. The careful selection of activation functions also played a pivotal role in improving model performance. Likewise, the consistent performance observed across cross-validation sets underscores the reliability of our model. However, exploring alternative architectures and incorporating specific features to provide the network with additional structural and geometrical information may yield further improvements in accuracy. Still,

References

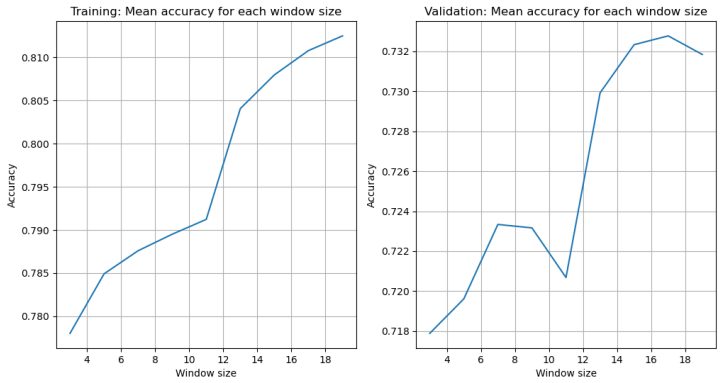
1. Jiang Q, Jin X, Lee SJ, Yao S. Protein secondary structure prediction: A survey of the state of the art. J Mol Graph Model. 2017 Sep 1;76:379–402.
2. Pakhrin SC, Shrestha B, Adhikari B, Kc DB. Deep Learning-Based Advances in Protein Structure Prediction. Int J Mol Sci. 2021 Jan;22(11):5553.
3. Wardah W, Khan MGM, Sharma A, Rashid MA. Protein secondary structure prediction using neural networks and deep learning: A review. Comput Biol Chem. 2019 Aug 1;81:1–8.
4. Dongardive J, Abraham S. Reaching optimized parameter set: protein secondary structure prediction using neural network. Neural Comput Appl. 2017 Aug 1;28(8):1947–74.
5. katarinaelez/protein-ss-pred: Protein secondary structure predictor [Internet]. [cited 2024 Mar 11]. Available from: https://github.com/katarinaelez/protein-ss-pred
6. Drozdetskiy A, Cole C, Procter J, Barton GJ. JPred4: a protein secondary structure prediction server. Nucleic Acids Res. 2015 Jul 7;43(Web Server issue):W389.
7. Heffernan R, Yang Y, Paliwal K, Zhou Y. Capturing non-local interactions by long short-term memory bidirectional recurrent neural networks for improving prediction of protein secondary structure, backbone angles, contact numbers and solvent accessibility. Bioinformatics. 2017 Sep 15;33(18):2842–9.
8. Jayasimha A, Mudambi R, Pavan P, Lokaksha BM, Bankapur S, Patil N. An effective feature extraction with deep neural network architecture for protein-secondary-structure prediction. Netw Model Anal Health Inform Bioinforma. 2021 Oct 23;10(1):58.
9. Jones DT. Protein secondary structure prediction based on position-specific scoring matrices1. J Mol Biol. 1999 Sep 17;292(2):195–202.
10. Yang B, Wu Q, Ying Z, Sui H. Predicting protein secondary structure using a mixed-modal SVM method in a compound pyramid model. Knowl-Based Syst. 2011 Mar 1;24(2):304–13.
11. O'Malley T, Bursztein E, Long J, Chollet F, Jin H, Invernizzi L, et al. Keras Tuner [Internet]. 2019. Available from: https://github.com/keras-team/keras-tuner
12. O'Malley T, Bursztein E, Long J, Chollet F, Jin H, Invernizzi L, et al. Keras Tuner [Internet]. 2019. Available from: https://github.com/keras-team/keras-tuner

Supplementary material

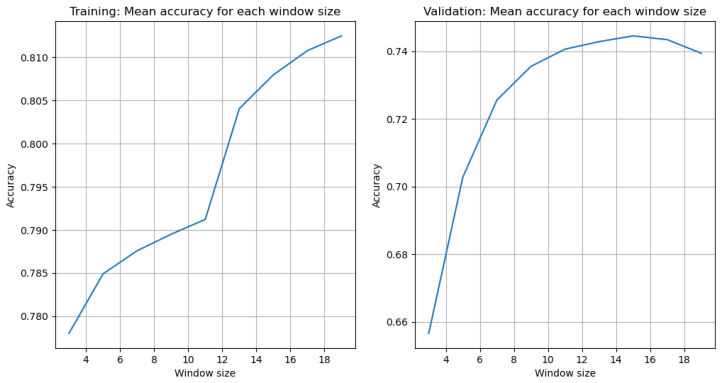
Hyperparameters tuning:

- **Window size**

(a)

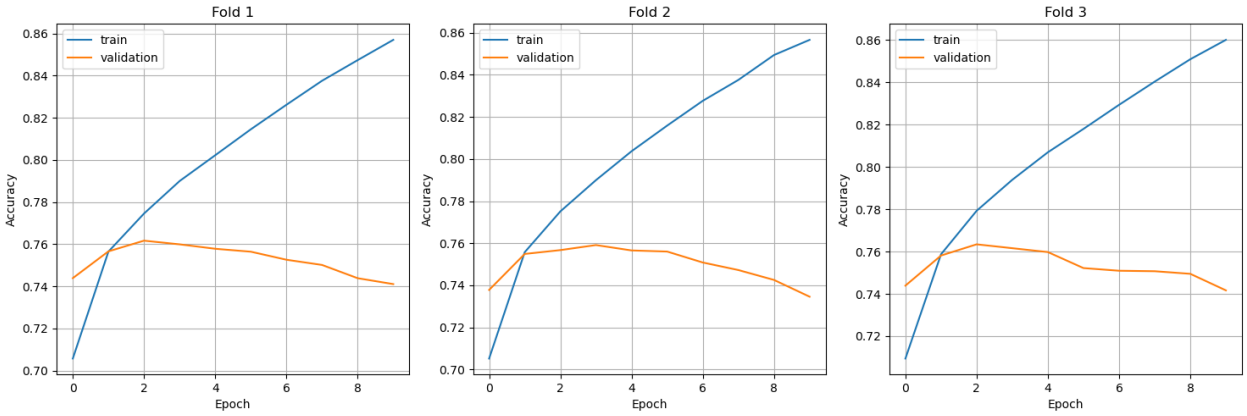


(b)



- **Number of epochs:**

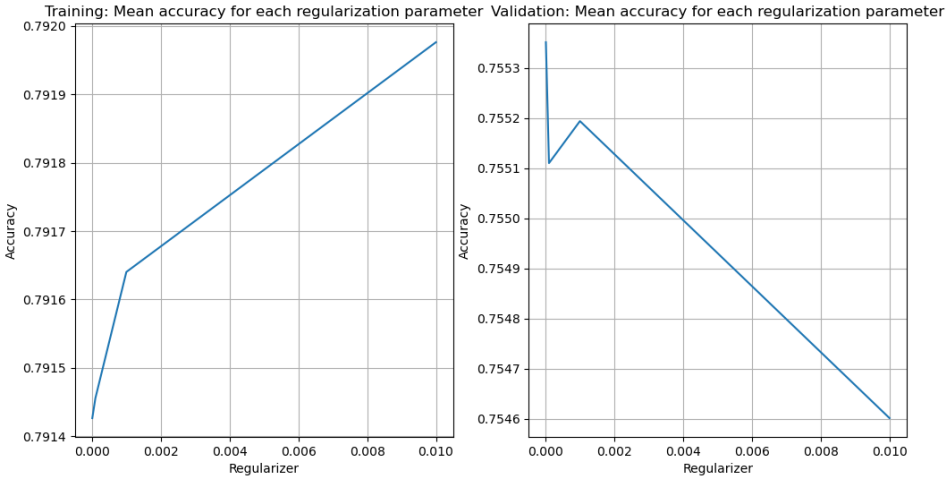
(c)



Sup 1. Window size for feature selection comparison between two model architectures selected as the best models according to the Keras hyperparameter optimization framework. (a) Model selected by the Bayesian Optimization algorithm. (b) Model selected by the Hyperband algorithm. (c) train and validation set accuracy during training across number of epochs: four epochs was selected as the appropiate number for training of the model to avoid overfitting.

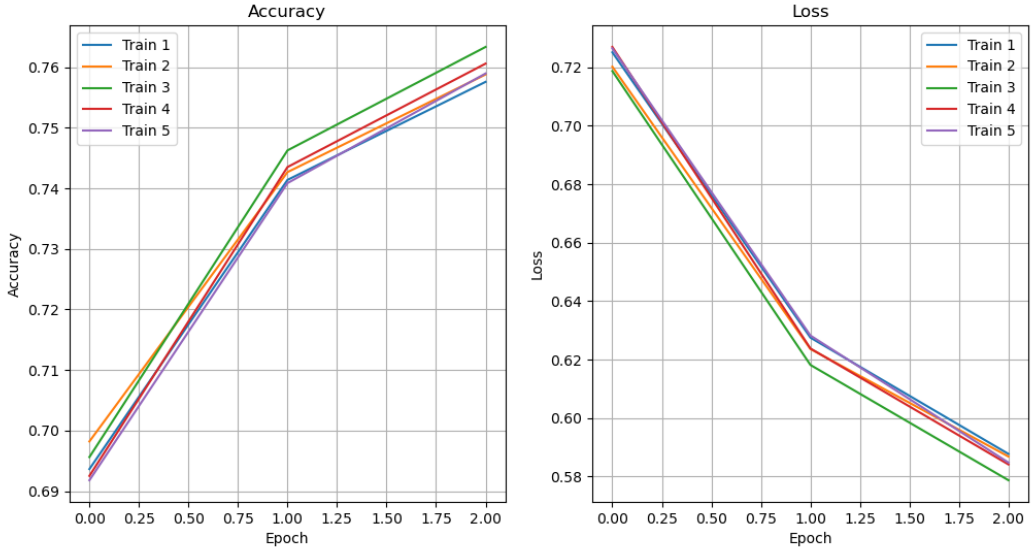
Regularization techniques:

- L1/L2

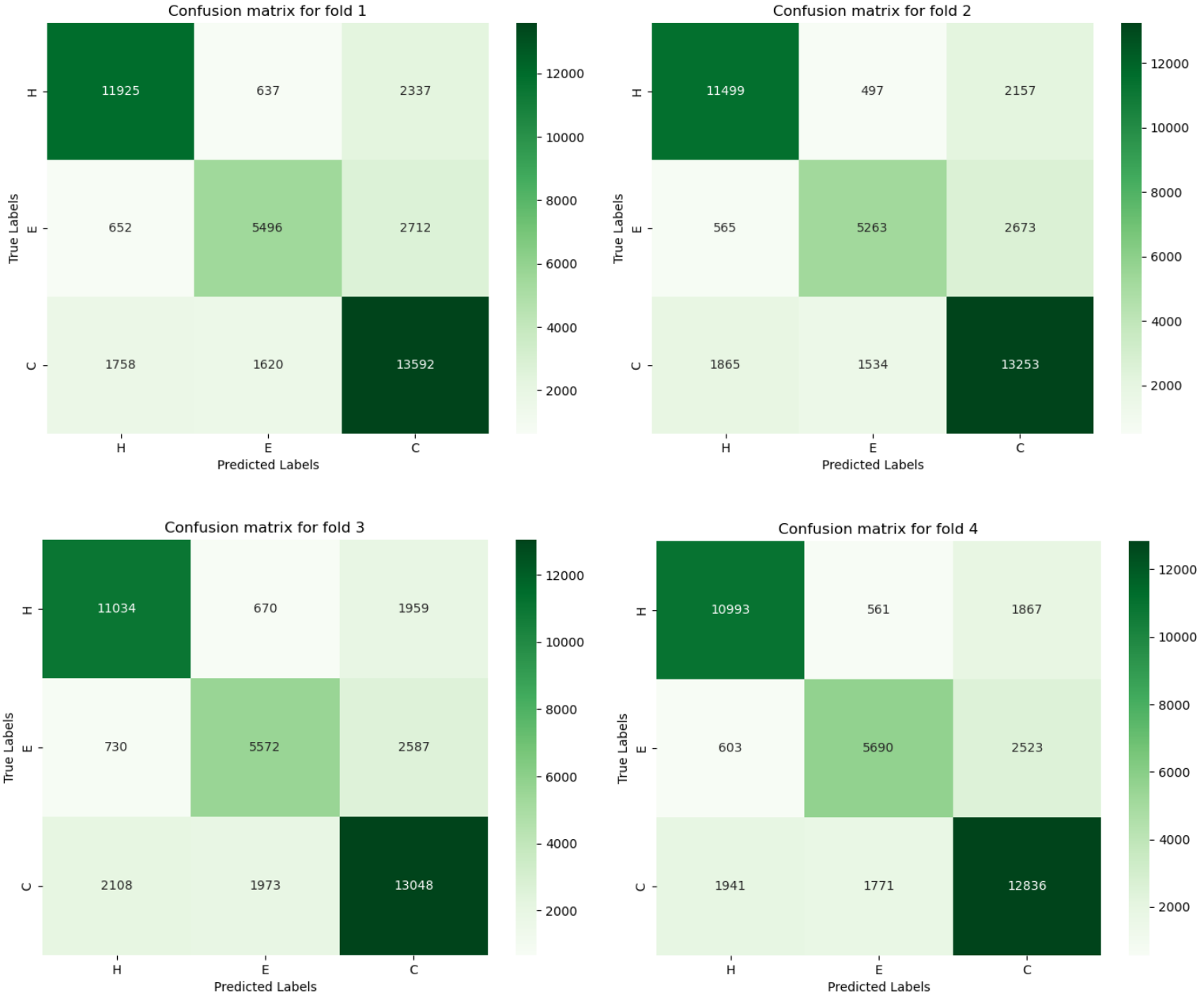


Sup 2. Test accuracy mean for the five cross-validation models evaluating four different values for the Activity regularization option [0.01, 0.001, 0.0001, 0.00001]

Accuracy and loss during model training



Sup 3. Training accuracy for cross-validation test set with final architecture.



Sup 4. Confusion matrix for cross-validation sets training and test on blind set.

Table 1. Cross-validation accuracy during training for hyperparameter tuning

CV set	Initial accuracy	EPOCHS:4	Regularization: 0.00001	Activation function	Dropout
1	0,7349	0,7540	0,7560	0,7614	0,7619
2	0,7412	0,7576	0,7539	0,7576	0,7603
3	0,7282	0,7460	0,7463	0,7461	0,7508
4	0,7331	0,7553	0,7525	0,7603	0,7607
5	0,7306	0,7543	0,7584	0,7622	0,7623

Table 2. Prediction accuracy for each cross-validation set against the blind test

Train-model	Blind-set accuracy	SD
1	0,7622	
2	0,7631	
3	0,7512	
4	0,7609	
5	0,7634	
Mean accuracy	0,7601	0,00456244