Course Project: Exploring Neural Network Architectures for EEG Data Classification

Mohammad Akbarnezhad
UCLA
miles85@ucla.edu

Abstract

This study investigates the efficacy of various neural network architectures in classifying electroencephalography (EEG) data in order to predict the task that a subject has intended or performed. EEG data, comprising 2115 examples from 22 electrodes over 1000 time bins, presents unique challenges in capturing both spatial and temporal dimensions. I explored and compared three classes of architectures: Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM), and a hybrid of CNN and LSTM. Initial experiments focused on generalization capabilities across different subjects while a model trained based on a single subject. Results indicated the necessity of training on a broad subject pool for effective generalization. Further analysis revealed the importance of selecting appropriate temporal cutoffs to improve model performance. I discovered that CNN-based architectures, specifically CNN1 and a hybrid model, outperformed others, achieving test accuracies above 70%. The study also examined various configurations of layer sequencing and hyperparameters. While LSTMs alone were less effective, likely due to the spatial-temporal complexity of EEG data, hybrid models integrating CNNs showed potential, although they did not consistently outperform standalone CNN models. These findings underscore the complexity of EEG data classification and the potential of tailored neural network architectures in improving accuracy and generalization.

1. Introduction

1.1. Architectural Exploration

The exploration began with a preliminary study of multiple architectures, convolutional layers differing in filter size and the number of convolution layers. Architectures with smaller kernel sizes compensated with additional layers, while those with larger filter sizes had fewer layers. Residual networks was also considerd, which incorporate shortcut connections to bypass layers, addressing the vanishing gradient problem [1, 3] in deep networks. This allows for the construction of deeper networks, potentially improving performance on complex tasks. Despite the theoretical advantages, I was unable to conduct adequate hyperparameter tuning in this case, yet the results from linear data flow still outperformed others. A similar issue arose when attempting to construct a network akin to the inception concept in GoogleNet [4], which allows the network to select the most appropriate filter size. Consequently, I decided against using this approach as the initial investigation did not yield promising results, potentially due to issues with hyperparameter tuning.

1.2. Convolutional Neural Networks (CNN)

Two CNN architectures were investigated: CNN1 and CNN2 (Figures 1 and 3 in the appendix, respectively). CNN1 convolves the temporal dimension first, followed by spatial convolution, and concludes with pooling [2]. In contrast, CNN2 treats the spatial dimension as depth, akin to channels in image data. Despite CNN1's logical approach to handling spatial information, exploring CNN2's performance was deemed valuable.

1.3. Recurrent Neural Networks (LSTM)

Given the temporal nature of EEG data, an LSTM architecture using Long Short-Term Memory (LSTM) layers was also tested (Figure 2 in the appendix). However, the combination of spatial data characteristics and long time steps in the EEG data presented challenges, making LSTMs less favorable in this context.

1.4. Hybrid CNN-LSTM

Finally, hybrid architectures combining CNNs and LSTMs were considered similar to CNN1 and CNN2. These hybrids integrate an LSTM layer before the final dense layer in both the CNN1 and CNN2 architectures. While the design convolves both the spatial and temporal dimensions, it ensures that the preceding CNN layers retain enough temporal information for effective LSTM processing.

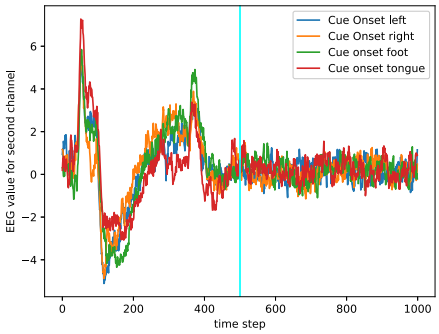

Figure 1. Value of EEG (time-series) averaged over all examples (e.g. for first channel)

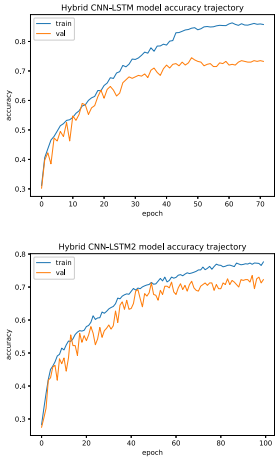

Figure 2. learning curve for hybrid-1 and hybrid-2

2. Results

2.1. Single Subject Generalization

Initially, the generalization capability of models trained on individual subjects was assessed. Using the CNN1 architecture, models were individually trained on subjects 0 and 5, with adjustments made until each achieved over 70% validation accuracy. However, as Table 1 in the appendix illustrates, neither model demonstrated high performance when applied to test data from all subjects. This indicates that relying on models trained on a broader subject pool is essential for better generalization.

2.2. Temporal Dimension Analysis

An analysis of the temporal dimension across all four classes, averaged over all examples, revealed a lack of meaningful data towards the end of the recordings (Figure 1). This suggests that the recording duration exceeded the actual duration of the tasks. To address this, three different cutoff times – 500, 600, and 700 time bins – were evaluated to eliminate less informative data segments. The 500-time-bin cutoff demonstrated the best performance, improving accuracy on test data across all considered architectures.

3. Discussion

3.1. Architecture Performance

Among the various architectures tested, CNN1 and CNN2 achieved test accuracies above 70%. In order to find optimal hyperparameters the process involved manual experimentation with multiple variables, including filter size, number of filters, type of padding, nonlinearity function (elu, relu), pooling type (max or average) and size, dropout ratios (too many choices and combination for different layers manually), optimizer (Adam, RMSProp), learning rate (too many choices were tried), and learning rate scheduler. I could not use grid search for all these combinations that I wanted to examine as it would involve extensive computation. Although initially I did not use dropout for the convolution layers I ended up using them as later I noticed without dropout for convolution layers the model was overfitting (gap between training and validation performance), while excessive dropout ratios were avoided due to their negative impact on model learning as was obvious from stall in model learning the training dataset. An early stopping strategy was employed to prevent overfitting, ceasing training when validation accuracy lagged behind training accuracy. An example of learning curve for hybrid-1 and hybrid-2 is shown in figure 2 in which the training of hybrid-1 has stopped because of early stopping schema. However, hybrid-2 has stopped at epoch 100 not due to early stopping but just because of the max number of epochs. As shown, the validation score is steadily increasing together with the training score and there is not sign of overfitting. The graph tells us there is still capacity in model to learn more and improve the performance by training for more epochs.

3.2. Layer Ordering and Architecture Variations

Experimentation with the sequence of convolution, activation, pooling, batch normalization, and dropout layers led to different optimal configurations for CNN1 and CNN2. In contrary to my understanding to use batchnorm layer before activation layer as we did in homework, I ended up using two different cases. For CNN1, the sequence of convolution, batch normalization, activation, average pooling, and dropout yielded the best results. This order particularly enhanced the network's ability to normalize features prior to activation, improving the stability and performance of the learning process. In contrast, CNN2 performed best with a sequence of convolution, activation, max pooling, batch normalization, and dropout.

3.3. LSTM Limitations and Hybrid Models

The LSTM architecture did not perform as well as CNN, likely due to the dataset's spatial components and the long duration of time series data, which poses a challenge for LSTMs. To mitigate these issues, a convolution layer was introduced at the start of the network to handle spatial information and compress temporal dimensions, making it more manageable for the LSTM. The results improved in comparison to pure LSTM. Interestingly, while it was hypothesized that hybrid models would outperform pure CNN models, this was not consistently the case. For example, CNN1 outperformed hybrid-CNN-LSTM1, though this could be attributed to potentially insufficient hyperparameter tuning.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part IV 14*, pages 630–645. Springer, 2016. 1

[2] Vernon J Lawhern, Amelia J Solon, Nicholas R Waytowich, Stephen M Gordon, Chou P Hung, and Brent J Lance. Eegnet: a compact convolutional neural network for eeg-based brain–computer interfaces. *Journal of neural engineering*, 15(5):056013, 2018. 1

[3] Robin Tibor Schirrmeister, Jost Tobias Springenberg, Lukas Dominique Josef Fiederer, Martin Glasstetter, Katharina Eggensperger, Michael Tangermann, Frank Hutter, Wolfram Burgard, and Tonio Ball. Deep learning with convolutional neural networks for eeg decoding and visualization. *Human brain mapping*, 38(11):5391–5420, 2017. 1

[4] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 1

Table 1: Performance of model trained for a single subject.

Person Id	Training (augmented)/Validation size	Accuracy on test for all persons
0	748/50	44.0%
5	744/50	37.2%

Table 2: Summary of the performance of considered architectures.

Model	Validation Accuracy	Test Accuracy	Early stopping at epoch
CNN-1	76.7 %	74.9 %	Force stopped (100)
CNN-2	71.5 %	70.8 %	Force stopped (100)
LSTM	40 %	39 %	55
Hybrid CNN-LSTM-1	73.3 %	69.6 %	70
Hybrid CNN-LSTM-2	73.2 %	68.8 %	Force stopped (100)

Architectures:

Hybrid CNN- LSTM -1 is same as CNN-1 for which Dense(40) and LSTM(10) are also added before last dense (4, softmax) layer. Among these two, only **CNN1** is shown in Figure 1.

Hybrid CNN-LSTM-2 is same as CNN-2 for which Dense(40) and LSTM(10) are also added before last dense (4, softmax) layer. Among these two, only **Hybrid CNN-LSTM-2** is shown in Figure 2.

Difference between CNN1 and CNN2:

In CNN1, initially temporal dimension is convolved followed by convolution in spatial dimension (channels) followed by average pool in temporal dimension. Finally temporal dimension convolved and pooled. However, in CNN2, the spatial dimension is just convolved in the beginning through the depth of the filters.

Model	CNN-1	CNN-2	LSTM	Hybrid -1	Hybrid -2
activation	elu	elu	elu	elu	elu
Pool	Average	Max	-	Average	Max
Dropout ratio	0.4	0.4	0.4	0.6	0.6
optimizer	Adam	Adam	Adam	Adam	Adam
learning rate	1e-3	5e-4	1e-3	1e-3	5e-4
learning rate (patience, factor)	5, 0.6	5, 0.4	10, 0.3	5, 0.6	10, 0.4

Data augmentation techniques such as max pooling, averaging with noise, and subsampling with noise are applied to the training data.

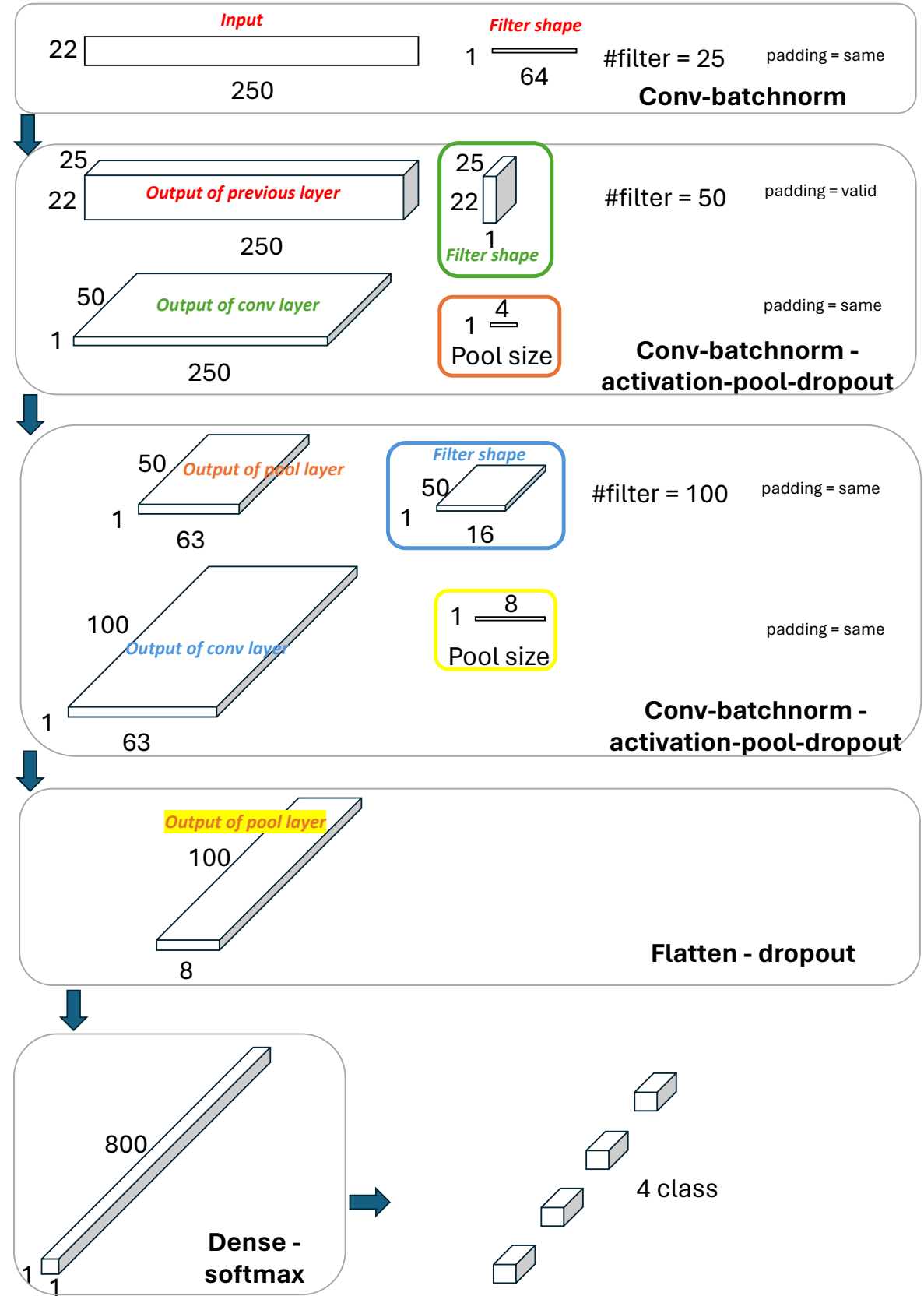

Figure 1: Architecture of CNN-1

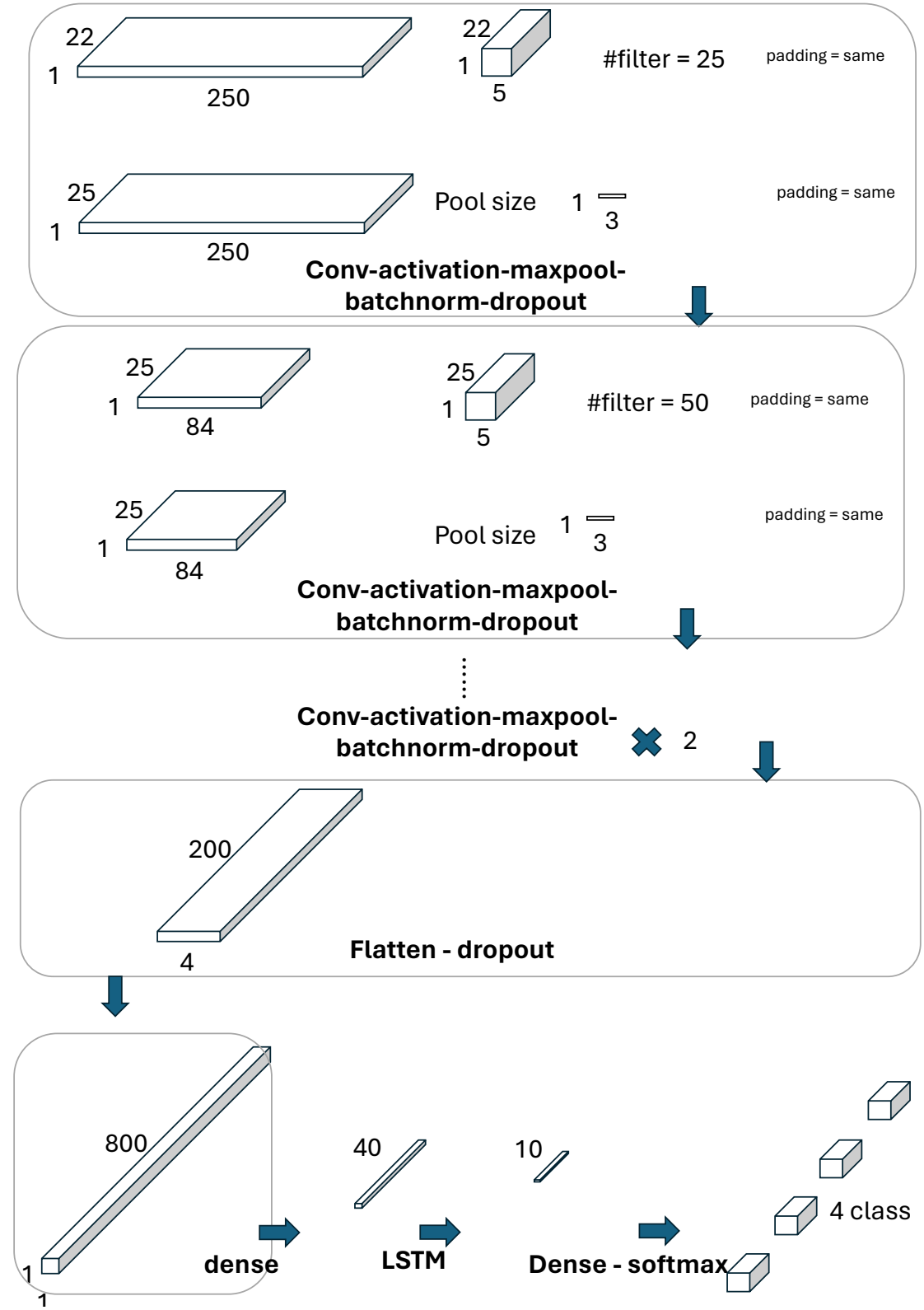

Figure 2: Architecture of Hybrid-CNN-LSTM-2

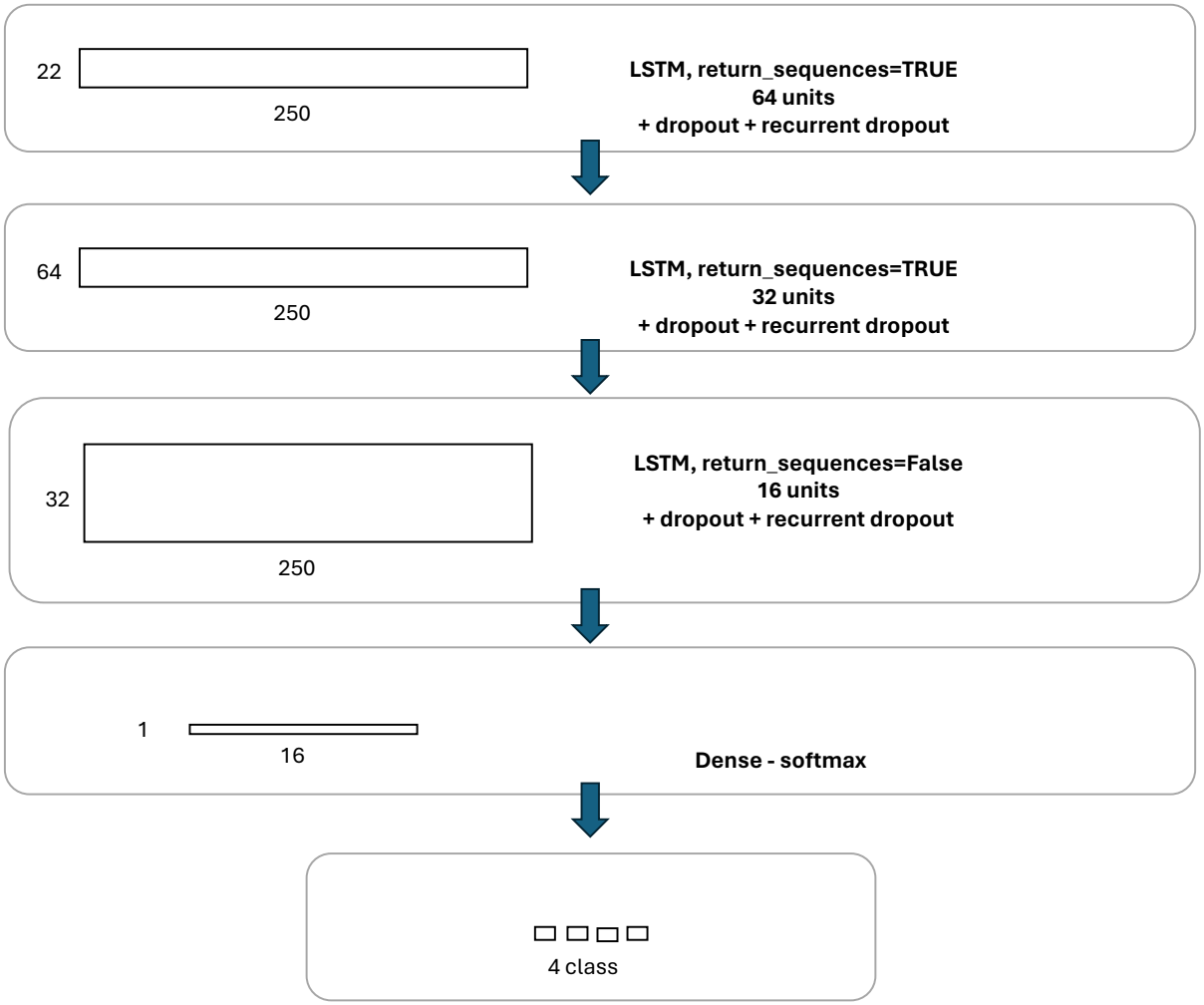

Figure 3: Architecture of LSTM only