Using Transfer Learning from a Self-Supervised model for ECG Classification

Xander Cartwright - 6532786, Rina Fumoto - 6498806, Liam McIntosh - 6471883

University of Surrey – COM3025

Introduction

In this investigation we will be classifying data from a collection of heartbeat signals from the ECG Heartbeat Categorization Dataset [1]. The ECG Heartbeat Categorization Dataset is a collection of two famous datasets, the MIT-BIH Arrhythmia Database [2] and the PTB Diagnostic ECG Database [3]. The aim of this project is to train a self-supervised model on the classification of the MIT-BIH data, and then use transfer learning to classify the PTB data into normal or abnormal classes. The datasets we are using contain data from Electrocardiography (ECG) tests. ECG signals are electrical signals detected on the skin as the heart contracts.

The MIT-BIH Arrhythmia Database consists of arrhythmia data collected at Boston's Beth Israel Hospital Arrhythmia Laboratory and MIT between 1975 and 1979. The database contains 48 half-hour excerpts of two-channel ambulatory ECG readings, 24 randomly selected and 25 selected ECG readings from a set of 4000 24-hour ambulatory ECG. The 25 were selected to give rarer, but clinically significant, arrhythmias better representation, as they would not be well-represented in the small 24 randomly selected samples. This dataset consists of 109,446 labelled samples, each cropped and padded, and has five labels: 0 - Normal beat (N), 1 - Supraventricular ectopic beat (S), 2 - Ventricular ectopic beat (V), 3 - Fusion beat (F), 4 - Unknown beat (Q) [4].

The PTB Diagnostic ECG Database is a collection of 549 records from 290 subjects taken at Physikalisch-Technische Bundesanstalt (PTB) in Germany. From the 290 subjects: 148 were diagnosed with myocardial infarction (MI), 52 were healthy and the remaining 90 had been diagnosed with seven different diseases. The ECG Heartbeat Categorization Dataset only uses the healthy and MI subject's data, meaning there are only two categories for this dataset: normal (healthy) and abnormal (MI).

Literature Review

There are mainly four steps for classifying arrhythmia using ECG: ECG signal pre-processing, heartbeat segmentation, feature extraction and classification [6]. The first two steps have already been done for the datasets we are using. Instead of doing any specific feature extraction we decided to experiment with using the entire signal so that the model could be used more generally. For the classification step we decided to research and experiment with different methods, we will now review these.

The first method is the Multi-Layer Perceptron (MLP), which is considered to be the most basic implementation of the artificial neural network. Constructed as a series of fully connected layers with non-linear activations and intermixed with dropout layers for data regularization. [8] The MLP has been demonstrated to be a simple but effective approach for arrhythmia classification [9].

The Convolutional Neural Network (CNN) is a highly popular model often employed in tasks such as Object Classification. However, studies have pointed towards an effective use of CNNs in the areas of arrhythmia and MI classification [1][7]. The CNN itself is a type of deep neural network architecture that makes use of a series of convolution, pooling, and activation layers. The purpose of the convolutional layer is to extract a feature from an area within some target data. The pooling layer on the other hand is designed to perform sub-sampling where the dimensions of the input data is reduced by a given factor. By stacking convolutional, activation function and pooling layers, a classical CNN architecture can be created.

Recurrent Neural Networks (RNNs) differ from their more typical feed-forward counterparts in that RNNs contain connections that form cycles. This property allows these networks to form internal memory and retain contained information over extended periods of time. With this RNNs can be used to model temporal behaviour making them effective when used against sequences of data across a temporal dimension [10]. However, they can suffer from a vanishing gradient. A more sophisticated version that addresses this is the Long Short-Term Memory architecture [11]. Another notable approach involves the model ensemble. Which takes multiple different models and combines the output of each in order to form an overall output. This method has been used to enhance the accuracy of classification tasks. This investigation will also consider the applications of transfer learning.

This is the process by which a given subset (or complete set) of weights are taken from a pre-trained model and used to form the base model for a similar task. Using transfer learning has often been demonstrated to result in far more effective models than ones that were initialized with completely random weights. In some cases, the weights will be set across the whole model, or a section of a model. These weights can also be locked during training in order to create a fixed feature extractor. When considering the problem of arrhythmia and MI classification, there is evidence to suggest that transfer learning, whereby a pretrained model for classifying arrhythmia is applied to MI classification, can yield improved accuracy [1].

Finally, we will be utilizing the method of representation learning. This is the process of using a self-supervised model against a dataset that has been labelled automatically. The process by which automatic labelling has been achieved in the past is via the applications of transformations. By applying transformations such as: rotations, stretches, translations and others. The subsequently generated data can then be labelled automatically by its transformation. By training a model on this transformed dataset, high level features can be learnt resulting in notable improvements to accuracy [5]. In addition to this, a multi-task version of representation learning can be used which divides the self-supervised task into several binary sub-problems which can then be optimized in parallel with each other. Multi-task learning has shown to be successful for ECG classification [5], as the different transformation classification tasks can help to learn the different aspects of ECG scans.

Problem Analysis

We performed some initial experimentation using 3 different classifiers: the MLP, CNN and LSTM. These models were trained on the raw datasets in a supervised manner. This was done in order to gain an understanding of the general effectiveness of these models against both datasets. Both datasets were split up with a train to test ratio of 9:1. Upon preliminary experimentation, it was noted that these methods performed slightly better on the PTB Diagnostic ECG Database then on the MIT-BIH Arrhythmia Database. There is however an exception for the LSTM which is shown in Table 1, the LSTM performed very well against the MIT dataset with a test accuracy of approximately 87.25%, but only achieving 83.85% for PTB. For PTB, the MLP also achieved an accuracy of 91.68%, whilst the CNN managed to achieve the best result with 97.46% accuracy.

Model Type	Test Accuracy (MIT)	Test Accuracy (PTB)
MLP	90.38%	91.68%
CNN	92.28%	97.46%
LSTM	87.25%	83.85%

Table 1: Results from preliminary experiments on the MIT and PTB datasets

However, to try and improve these results without altering the structure of the models, we would have to either acquire more data or perform transfer learning. In this investigation we chose the latter option. Specifically, we intend to train a model using the MIT dataset and then transfer the weights to a secondary model to classify the PTB & MIT-BIH datasets. However, this may not be enough to extract meaningful and truly generalized features that could be transferred to the new models. So, we decided to use a representational learning approach, so that we can improve the classifiers' ability to learn high-level features. Whereby a self-supervised model would be used to train on an artificially modified version of the MIT-BIH dataset. These weights would then be transferred to the secondary model to be used to enhance the model's effectiveness against both datasets. Whilst some models perform better than others, some may extract different kinds of features from the data. To address this, we decided to combine the resulting final classifiers into an ensemble to further enhance their collective performance.

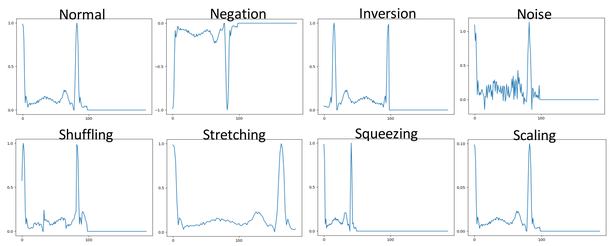


Figure 1: Examples of each Transformation

Implementation

As an implementation, we begin by addressing the representational model. The first part of this is to construct an artificial self-labelling dataset based on the MIT-BIH Arrhythmia dataset. We applied 7 different transformations to each instance within the preexisting dataset (with the existing labels removed). These transformations are then applied after the padding has been removed, and the padding is then reapplied after the transformation. These transformations will be listed as follows. Negation, this maps out each signal by flipping it about the x axis e.g., y' = -f(x). Inversion, this reverses the direction of the signal in the x-axis. Noise, this transformations adds a certain amount of Gaussian noise to the signal. Shuffling, this splits the signal into n different segments and randomly shuffles them to form a mixed-up signal. Stretching, this stretches the signal by a factor of 2 in the x-axis. Squeezing, this stretches the signal by a factor of ½ in the x-axis. Scaling, this reduces the amplitude of the signal globally by a factor of 10. By applying these transformations, 8 different versions of the dataset will be created. And each of these are automatically labelled from 0-7. Examples of the transformations can be seen in Figure 1. The created dataset was split into a train to test ratio of 4:1.

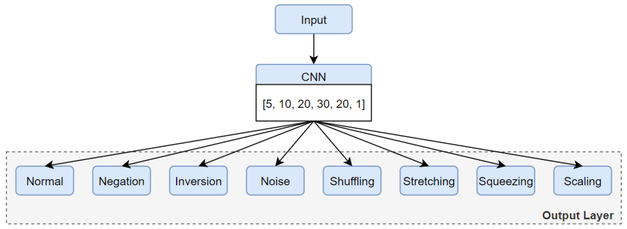


Figure 2: Architecture of the self-supervised CNN

With the representational dataset created, we then build the self-supervised model architecture for each model type, an example of this with the CNN can be seen in Figure 2. This architecture is repeated for each of the individual model types as seen Table 1. The individual models are built as follows.

The MLP consists of 7 fully connected layers. A dropout layer is included in the network to introduce regularization The network primarily uses the ReLU activation function with a Sigmoid activation for the output layer. The CNN is constructed via pairs of one-dimensional convolution and pooling pairs with a Leaky ReLU activation for the intermediate detector layer [12]. These pairs are repeated 3 times. The type of pooling used is max pooling. Upon completion of these layers, the output is flattened and is fed through 3 fully connected layers. These layers also utilize the Leaky ReLU activation function, except for the output layer which uses the Sigmoid activation. Finally, the LSTM is configured according to the Keras implementation of the LSTM. The LSTM itself takes in 64 units as a parameter and the output from this is fed through 2 fully connected layers. The hidden layer uses a ReLU activation function, whilst the final layer uses the sigmoid activation function.

A multi-task framework is also added to the representational models. In this design, there are 8 tasks. Each task is used to represent a binary classification problem for its respective transformation. Meaning that each task will attempt to classify if the incoming signal is or is not the respective transformation (including one for no transformation). This is represented in the network by 8 different fully connected layers, which all receive the output from the previous. These models are then trained on the representational dataset for 5 epochs. The model weights are then transferred to supervised architecture variants (see figure 3) and are trained on the PTB dataset for 20 epochs. Each model is trained with Binary Cross-Entropy Loss for PTB and Categorial Cross-Entropy for MIT-BIH. All of them are trained using Adam, and a mini-batch size of 64.

Finally, to improve the accuracies of the individual models, an ensemble is constructed out of the 3 trained models. This ensemble uses an averaging operation to fuse the outputs from each individual classifier to form an overall output. This final model is again trained for 20 epochs. This overall forms 3 different layers of models requiring 3 different weight transferences. This full process is depicted in Figure 3.

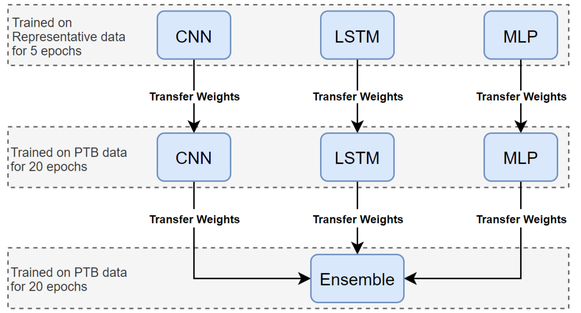


Figure 3: Depiction of all the weight transferences and training regimes needed to create the final model.

Evaluation

Upon experimentation with the additional transferred representation learning, we are met with positive results. For MIT, shown in Table 2, the CNN and the LSTM perform very well. With the LSTM making the largest gain in terms of accuracy with a percentage accuracy gain of 4.91%. Whilst this is small as an absolute value, it should be noted that all these models are operating at high degrees of accuracy, meaning that small gains can be quite significant. However, in the case of the MLP, the pre-training has had a detrimental effect causing to lose an accuracy of 3.2% when compared to its performance in Table 1. This implies that the MLP may not have responded well to the representational learning, causing it diverge away from learning meaningful features. The final ensemble scores the highest accuracy of 93.01%, verifying the small, but effective use of combining classifiers.

For PTB, both the CNN and the LSTM make positive gains, with the CNN reaching an accuracy of 98.01%. The MLP also fails to improve, but not by the same margin as for MIT. The final ensemble for PTB scores an impressive highest test accuracy of 98.49%.

Model Type	Test Accuracy (MIT)	Test Accuracy (PTB)
MLP with Rep	87.18%	89.14%
CNN with Rep	92.69%	98.01%
LSTM with Rep	92.16%	97.04%
Ensemble with Rep	93.01%	98.49%

Table 2: Results from preliminary experiments and the Transferred ensemble for the PTB & MIT-BIH datasets.

Conclusion

In summary, this investigation has highlighted the potential effective use of both representational learning and transfer learning to build superior models for ECG arrhythmia classification. Whilst this does not work in all cases, it does point towards an overall positive trend of improvement. In future, this investigation could be expanded to include a larger amount of both transformations for representational learning, and individual classifiers to push the limits of this concept's ability to improve classification performance. Therefore, this investigation can be considered a success.

Key References

[1] Kachuee, M., Fazeli, S., and Sarrafzadeh, M. (2018). ECG Heartbeat Classification: A Deep Transferable Representation. 2018 IEEE International Conference on Healthcare Informatics (ICHI).

[2] Moody, G. and Mark R. (2001). The impact of the MIT-BIH Arrhythmia Database. IEEE Eng in Med and Biol 20(3):45-50 (PMID: 11446209)

[3] Bousseljot, R., Kreiseler, D. and Schnabel, A. (1995). Nutzung der EKG-Signaldatenbank CARDIODAT der PTB über das Internet. Biomedizinische Technik, Band 40, Ergänzungsband 1 S 317

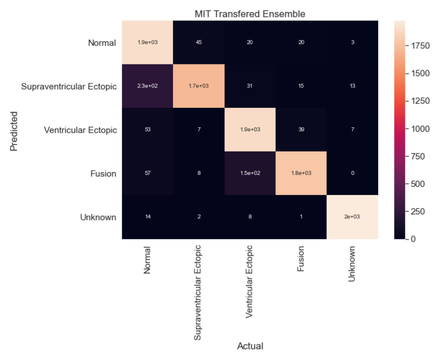
[5] Sarkar, P. and Etemad, A., (2021). Self-supervised ECG Representation Learning for Emotion Recognition. IEEE Transactions on Affective Computing, pp.1-1.

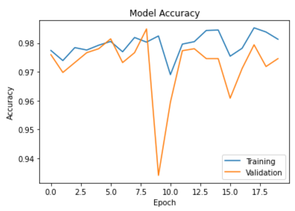
References

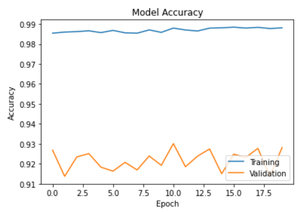
[1] Kachuee, M., Fazeli, S., and Sarrafzadeh, M. (2018). ECG Heartbeat Classification: A Deep Transferable Representation. IEEE International Conference on Healthcare Informatics (ICHI).
[2] Moody, G. and Mark R. (2001). The impact of the MIT-BIH Arrhythmia Database. IEEE Eng in Med and Biol 20(3):45-50 (PMID: 11446209).
[3] Bousseljot, R., Kreiseler, D. and Schnabel, A. (1995). Nutzung der EKG-Signaldatenbank CARDIODAT der PTB über das Internet. Biomedizinische Technik, Band 40, Ergänzungsband 1 S 317.
[4] De Chazal, P., O'Dwyer, M. and Reilly, R.B. (2004). Automatic classification of heartbeats using ECG morphology and heartbeat interval features. IEEE transactions on biomedical engineering, 51(7), pp.1196-1206.
[5] Sarkar, P. and Etemad, A., (2021). Self-supervised ECG Representation Learning for Emotion Recognition. IEEE Transactions on Affective Computing, pp.1-1.
[6] Luz, E.J.D.S., Schwartz, W.R., Cámara-Chávez, G. and Menotti, D. (2016). ECG-based heartbeat classification for arrhythmia detection: A survey. Computer methods and programs in biomedicine, 127, pp.144-164.
[7] Strodthoff, N., and Strodthoff, C. (2019). Detecting and interpreting myocardial infarction using fully convolutional neural networks. Physiological Measurement, 40(1), 015001.
[8] Rizal, A., and Hadiyoso, S. (2015). ECG Signal Classification Using Hjorth Descriptor. International Conference on Automation, Cognitive Science, Optics, Micro Electro-Mechanical System, and Information Technology (ICACOMIT), 2015, pp. 87-90, doi: 10.1109/ICACOMIT.2015.7440181.
[9] Xiang, C., Ding, S.Q., and Lee, T.H. (2005). Geometrical interpretation and architecture selection of MLP. IEEE Transactions on Neural Networks, vol. 16, no. 1, pp. 84-96, Jan. 2005, doi: 10.1109/TNN.2004.836197.
[10] Yildirim, O., Baloglu, U.B., Tan, R.-S., Ciaccio, E.J. and Acharya U.R. (2019). A new approach for arrhythmia classification using deep coded features and LSTM networks. Computer Methods and Programs in Biomedicine, 176 , pp. 121-133.
[11] Singh, S., Pandey, S.K., Pawar, U. and Janghel, R.R. (2018). Classification of ECG Arrhythmia using Recurrent Neural Networks. Procedia Computer Science, 132 , pp. 1290-1297.
[12] Acharya, U.R., Oh, S.L., Hagiwara, Y., Tan, J.H., Adam, M., Gertych, A. and San Tan, R., 2017. A deep convolutional neural network model to classify heartbeats. Computers in biology and medicine, 89, pp.389-396.

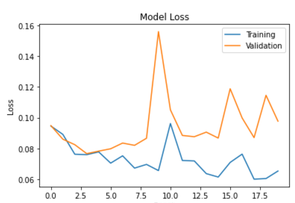
Appendix A – Raw Results

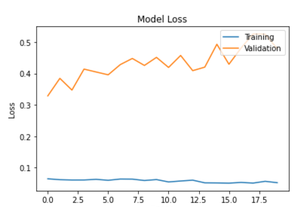
Training results of the Ensemble model with transferred representation learning





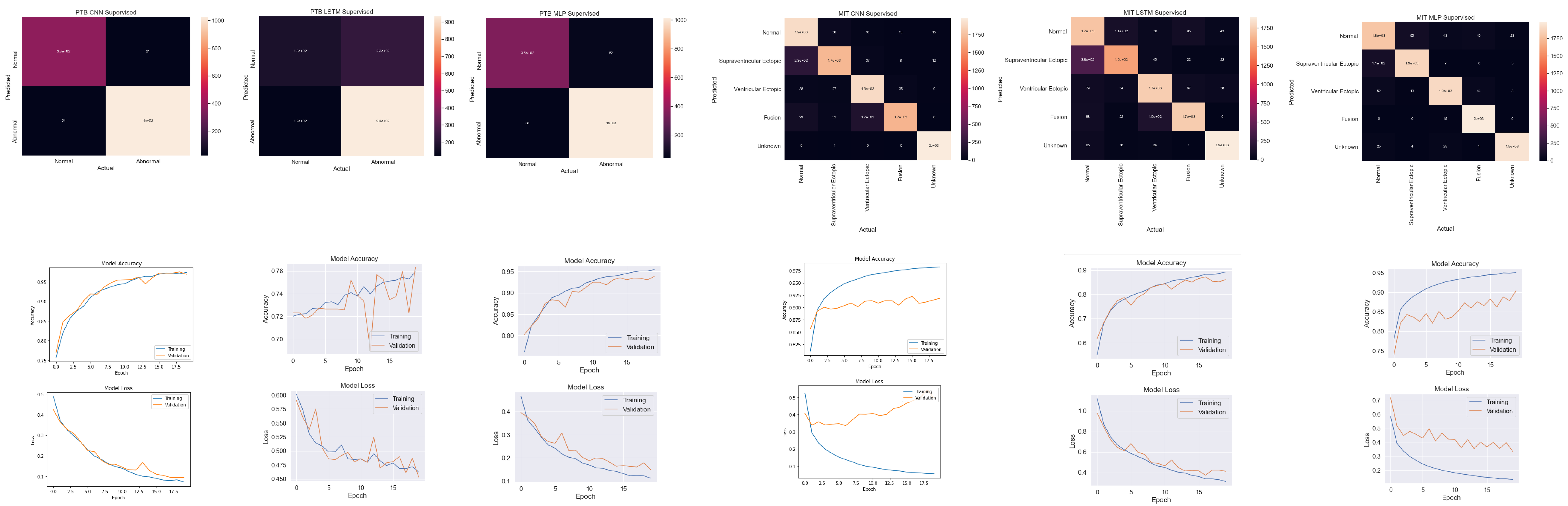






Training results of CNN, LSTM and MLP models with transferred representation learning

Initial Experiments of CNN, LSTM and MLP models



Appendix B – Screenshots

Screenshots

Supervised Only Experiments

MIT-BIH Dataset

-- MLP

```
Epoch 18/20
1563/1563 [==============================] - 3s 2ms/step - loss: 0.1369 - accuracy: 0.9497 - val_loss: 0.3551 - val_accuracy:
0.8880
Epoch 19/20
1563/1563 [==============================] - 2s 1ms/step - loss: 0.1396 - accuracy: 0.9481 - val_loss: 0.3967 - val_accuracy:
0.8784
Epoch 20/20
1563/1563 [==============================] - 2s 1ms/step - loss: 0.1317 - accuracy: 0.9520 - val_loss: 0.3356 - val_accuracy:
0.9038
```

- CNN

```
Epoch 15/20
1563/1563 [==============================] - 10s 7ms/step - loss: 0.0692 - accuracy: 0.9771 - val_loss: 0.4436 - val_accuracy:
0.9171
Epoch 16/20
1563/1563 [==============================] - 10s 7ms/step - loss: 0.0647 - accuracy: 0.9788 - val_loss: 0.4661 - val_accuracy:
0.9228
Epoch 17/20
1563/1563 [==============================] - 11s 7ms/step - loss: 0.0600 - accuracy: 0.9808 - val_loss: 0.4815 - val_accuracy:
0.9085
```

- LSTM

```
Epoch 16/20
1563/1563 [==============================] - 120s 77ms/step - loss: 0.3633 - accuracy: 0.8742 - val_loss: 0.4148 - val_accurac
y: 0.8654
Epoch 17/20
1563/1563 [==============================] - 120s 77ms/step - loss: 0.3417 - accuracy: 0.8817 - val_loss: 0.3727 - val_accurac
y: 0.8725
Epoch 18/20
1563/1563 [==============================] - 126s 81ms/step - loss: 0.3455 - accuracy: 0.8806 - val_loss: 0.4215 - val_accurac
y: 0.8547
```

PTB Dataset

- MLP

```
Epoch 18/20
205/205 [==============================] - 11s 51ms/step - loss: 0.2148 - accuracy: 0.9140 - val_loss: 0.2293 - val_accuracy:
0.9162
Epoch 19/20
205/205 [==============================] - 10s 51ms/step - loss: 0.2342 - accuracy: 0.9101 - val_loss: 0.2163 - val_accuracy:
0.9168
Epoch 20/20
205/205 [==============================] - 10s 51ms/step - loss: 0.2202 - accuracy: 0.9136 - val_loss: 0.2507 - val_accuracy:
0.8907
```

- CNN

```
Epoch 18/20
205/205 [==============================] - 1s 7ms/step - loss: 0.0589 - accuracy: 0.9792 - val_loss: 0.0942 - val_accuracy: 0.9
698
Epoch 19/20
205/205 [==============================] - 1s 7ms/step - loss: 0.0715 - accuracy: 0.9736 - val_loss: 0.0804 - val_accuracy: 0.9
746
Epoch 20/20
205/205 [==============================] - 1s 7ms/step - loss: 0.0542 - accuracy: 0.9810 - val_loss: 0.0851 - val_accuracy: 0.9
739
```

- LSTM

```
Epoch 18/20
205/205 [==============================] - 26s 129ms/step - loss: 0.4368 - accuracy: 0.7890 - val_loss: 0.4812 - val_accuracy:
0.7546
Epoch 19/20
205/205 [==============================] - 27s 129ms/step - loss: 0.4404 - accuracy: 0.7894 - val_loss: 0.4922 - val_accuracy:
0.7540
Epoch 20/20
205/205 [==============================] - 29s 140ms/step - loss: 0.3863 - accuracy: 0.8220 - val_loss: 0.3682 - val_accuracy:
0.8385
```

Added Representational Learning Experiments

MIT-BIH Dataset – Representational Models

- MLP

```
Epoch 5/20
1563/1563 [==============================] - 78s 50ms/step - loss: 0.2475 - accuracy: 0.9128 - val_loss: 0.4240 - val_accuracy:
0.8714
Epoch 6/20
1563/1563 [==============================] - 78s 50ms/step - loss: 0.2232 - accuracy: 0.9217 - val_loss: 0.4539 - val_accuracy:
0.8719
Epoch 7/20
1563/1563 [==============================] - 83s 53ms/step - loss: 0.2112 - accuracy: 0.9269 - val_loss: 0.4769 - val_accuracy:
0.8706
```

- CNN

```
Epoch 8/20
1563/1563 [==============================] - 11s 7ms/step - loss: 0.0740 - accuracy: 0.9766 - val_loss: 0.3582 - val_accurac
y: 0.9189
Epoch 9/20
1563/1563 [==============================] - 12s 8ms/step - loss: 0.0726 - accuracy: 0.9760 - val_loss: 0.3485 - val_accurac
y: 0.9260
Epoch 10/20
1563/1563 [==============================] - 13s 8ms/step - loss: 0.0658 - accuracy: 0.9788 - val_loss: 0.3815 - val_accurac
y: 0.9183
```

- LSTM

```
Epoch 16/20
1563/1563 [==============================] - 107s 68ms/step - loss: 0.1401 - accuracy: 0.9520 - val_loss: 0.3124 - val_accurac
y: 0.9179
Epoch 17/20
1563/1563 [==============================] - 107s 68ms/step - loss: 0.1300 - accuracy: 0.9557 - val_loss: 0.3052 - val_accurac
y: 0.9210
Epoch 18/20
1563/1563 [==============================] - 107s 69ms/step - loss: 0.1330 - accuracy: 0.9536 - val_loss: 0.3611 - val_accurac
y: 0.9009
```

PTB Dataset – Representational Models

- MLP

```
Epoch 18/20
205/205 [==============================] - 17s 84ms/step - loss: 0.2571 - accuracy: 0.8960 - val_loss: 0.2585 - val_accuracy:
0.8893
Epoch 19/20
205/205 [==============================] - 17s 82ms/step - loss: 0.2514 - accuracy: 0.8928 - val_loss: 0.2637 - val_accuracy:
0.8866
Epoch 20/20
205/205 [==============================] - 16s 79ms/step - loss: 0.2390 - accuracy: 0.8994 - val_loss: 0.2633 - val_accuracy:
0.8914
```

- CNN

```
Epoch 16/20
205/205 [==============================] - 1s 7ms/step - loss: 0.0565 - accuracy: 0.9791 - val_loss: 0.0673 - val_accuracy: 0.9
787
Epoch 17/20
205/205 [==============================] - 1s 7ms/step - loss: 0.0514 - accuracy: 0.9834 - val_loss: 0.0614 - val_accuracy: 0.9
801
Epoch 18/20
205/205 [==============================] - 1s 7ms/step - loss: 0.0499 - accuracy: 0.9808 - val_loss: 0.0745 - val_accuracy: 0.9
753
```

- LSTM

```
Epoch 18/20
205/205 [==============================] - 14s 70ms/step - loss: 0.0896 - accuracy: 0.9684 - val_loss: 0.0881 - val_accuracy:
0.9704
Epoch 19/20
205/205 [==============================] - 14s 70ms/step - loss: 0.0950 - accuracy: 0.9696 - val_loss: 0.1784 - val_accuracy:
0.9320
Epoch 20/20
205/205 [==============================] - 15s 71ms/step - loss: 0.1050 - accuracy: 0.9645 - val_loss: 0.0841 - val_accuracy:
0.9677
```

Final Ensembles

MIT-BIH Dataset – Final Ensemble

```
-
Epoch 10/20
1563/1563 [==============================] - 172s 110ms/step - loss: 0.0594 - accuracy: 0.9865 - val_loss: 0.4513 - val_accurac
y: 0.9192
Epoch 11/20
1563/1563 [==============================] - 174s 111ms/step - loss: 0.0518 - accuracy: 0.9887 - val_loss: 0.4193 - val_accurac
y: 0.9301
Epoch 12/20
1563/1563 [==============================] - 188s 120ms/step - loss: 0.0563 - accuracy: 0.9880 - val_loss: 0.4574 - val_accurac
y: 0.9185
```

-

PTB Dataset – Final Ensemble

```
-
Epoch 8/20
205/205 [==============================] - 21s 105ms/step - loss: 0.0701 - accuracy: 0.9805 - val_loss: 0.0820 - val_accurac
y: 0.9766
Epoch 9/20
205/205 [==============================] - 22s 109ms/step - loss: 0.0701 - accuracy: 0.9816 - val_loss: 0.0866 - val_accurac
y: 0.9840
Epoch 10/20
205/205 [==============================] - 24s 117ms/step - loss: 0.0630 - accuracy: 0.9847 - val_loss: 0.1560 - val_accurac
y: 0.9340
```