Machine Learning for Healthcare: Project 1

Mert Ertugrul, Johan Lokna, Nora Schneider

I. Introduction

An electrocardiogram (ECG) is a non-invasive method to record the electrical activity of the heart, based on which heart diseases can be detected. Different waveforms, morphologies and noise in the signal make their classification more difficult. Therefore, there is a need for accurate and low-cost diagnostic methods [6, 17]. In this report, we provide and evaluate different deep learning based solutions for classifying ECG timeseries.

For our experiments we use two thoroughly studied ECG datasets: the MIT-BIH Arrythimia Database [3, 10] and the PTB Diagnostic ECG Database [1, 3]. Both were preprocessed in a similar manner. The main characteristic of the given data is the sequential structure. Further, both datasets have high class imbalance. The MIT-BIH data has normal and different arrhythmia cases resulting in a total of 5 classes, whereas the PTBDB data only differentiates between normal and abnormal heatbeats.

The models can be grouped into four tasks and are discussed in section II. First, Vanilla CNN and Vanilla RNN models are tested. Second, ResNet, as an improvement on Vanilla CNN, Bidirectional LSTM and ConvLSTM as improvements on Vanilla RNN are implemented and evaluated. We further create a representation learning approach using U-Net. As a third task, we implement two ensemble approaches: Ensembles based on previous models and boosting of simple CNNs. Finally, transfer learning is applied on the best performing CNN and RNN variations.

The results of our models' performances are summarized in Table I. For reasons of clarity, additional model results are reported in Section III-A.

A. Methodology

For all tasks, whenever computational resources and time permit, hyperparameters deemed influential for the models are tuned via grid search with 5-fold cross validation and accuracy used as the decision metric. Further we implemented various generalization methods in our models, such as early stopping, learning rate reduction or dropout. For detailed information we refer to the provided repository.

II. Models

A. Task 1: Vanilla Models

Vanilla CNN: We test three different Vanilla CNN architectures. The final Vanilla CNN for the MIT-BIH data consists of alternating convolutional and max-pool layers. Contrary, a VGG-net architecture similar to [17] is used as a final Vanilla CNN model on the PTBDB data. The Vanilla CNN perform slightly worse than the baseline model on the MIT-BIH data, but it outperforms the baseline on the PTBDB data (see Ia and Ib). Therefore, the vanilla CNN captures the sequential structure of the data well and the extracted deep features are meaningful for the classification task.

Vanilla RNN: For the vanilla RNN, a RNN cell with a variable hidden state size is combined with a fully connected layer. Accuracy on MIT-BIH data is better than on PTBDB, potentially due to the size of the dataset. However, F1-score shows the opposite trend reflecting the class imbalance issues that may be arising. The Vanilla RNNs perform worse than the given baseline throughout all performance metrics (see Ia and Ib). It is well known, that Vanilla RNNs are prone to vanishing gradients due to the same hidden state parameters being used over backpropogation through time, and the hidden state memory remains short term without a gate mechanism to control gradient flow through it. Therefore, we also tested a single LSTM cell. Yet, no parameter combination converges. Changing model capacity through different hidden state sizes, stacking LSTM cells, different numbers of fully connected layers or addressing class imbalance through a weighted loss approach do not change this outcome. Thus, this model is excluded from reported results.

B. Task 2: Additional Models

ResNet: As a CNN structure becomes deeper its performance usually increases. However, problems of vanishing gradients and saturating accuracy arise. As a solution, ResNet implements residual connections between consecutive convolutional layers [4, 6]. Motivated by this, we evaluate the applicability of ResNet to our problem. While observing an improved performance compared to the Vanilla CNN on the PTBDB data (Ib), the performance slightly decreased on the MITBIH data

(Ia). This suggests, that the VanillaCNN on the MITBIH data is not effected by vanishing gradient or saturating accuracy. Contrary, these problems might have lead to a weaker performance of the Vanilla CNN on the PTBDB data.

Bidirectional LSTM: Bidirectional LSTM models are able to leverage information that is present before and after the currently processed section of the input sequence as it processes the sequence in both forward and reverse order. Consequently, they are known to improve performance in related tasks such as timeseries forecasting [15]. This model's performance indicate whether a non-causal approach to reading the sequence can improve predictive capability or whether the reverse order of the sequence can reveal useful information. For outputs of the forward and reverse runs, the default approach of concatenating them is used. Test performance improved compared to the Vanilla RNN and the fact that this model can converge despite the standard LSTM not converging is unusual. Yet, it does not outperform the baseline.

ConvLSTM: This model combines convolutional layers, an LSTM cell with variable hidden state size and fully connected layers. We theorize that the convolutional layers extract features in a parameter efficient way and the LSTM cell can interpret these features in a sequential manner [13]. Among all RNN based models, the ConvLSTM model is the best performer with respect to every evaluation metric for both datasets. Compared to both the Vanilla RNN and the variants of LSTM, the model reaches validation accuracies of 0.8 and 0.9 at earlier epochs, with a final test set accuracy of 0.97. Moreover, throughout the grid search, parameter combinations that enable convergence are more broad compared to Vanilla RNN, indicating better model stability. Overall, we infer that the feature extraction capability of the convolutional layers is responsible for faster convergence and higher performance. Despite outperforming other RNN based models, this architecture cannot outperform our purely CNN based models (e.g. baseline).

U-Net: Training classifiers on unsupervised representations instead of raw input signals can indeed provide significant performance gains [9], which has also been shown for heartbeat classification [7, 11]. We therefore implement an encoder based on U-Net [12], whose structure resembles that of an auto-encoder [5] as can be seen in Figure 1. The U-Net-encoder is trained in a self-supervised manner on partially masked heartbeat signals; an example is given in Figure 2. This allows us to train the encoder on all heartbeat signals from both datasets. We use a weighted mean-square error reconstruction loss, where the reconstruction error on the unmasked section is penalized by a factor $\alpha \in [0, 1]$. α controls the trade-off between the standard mean square reconstruction loss and the forecasting loss [9]. We find that a too high α causes the network to collapse into simply predicting the mean signal, whereas a too low α results in an uninformative embedding. α is tuned based on extrinsic evaluation of down-stream models' performance.

After the training of the U-Net, we extract the latent representation of the heartbeat signals from its middle layer, as indicated by the orange arrow in Figure 1. We then train dense neural network classifiers based on the latent representation. On the PTBDB dataset, the final classifier's performance is slightly above 0.98 with respect to all metrics, slightly worse than the other CNN-based methods. However, on the MIT-BIH dataset the model is able to outperform every other model with an F1-score of 0.9504 and an accuracy of 0.9918. By projecting the learnt embedding using t-SNE [16], we observe a significant better concentration of the minority classes to specific regions in the latent space for the MIT-BIH compared to the PTBDB dataset as seen in Figure 3 and 4. We suspect this effect to be caused by the minority classes in the MIT-BIH possessing more characteristic heartbeat signals. This gives a heuristic explanation for the improved performance over the MIT-BIH dataset; the concentration of the minority classes to specific regions in the latent space seems to allow the classifier to more efficiently learn the minority classes.

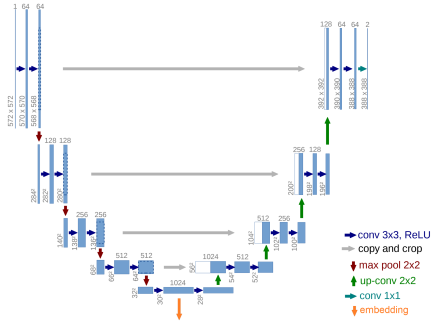

Fig. 1: U-Net architecture. Orange arrow indicates where the embedding is extracted

C. Task 3: Ensemble Models

Stacking Ensembles: Stacking is a widely used technique for improving the performance of a single classifier by combining different learning algorithms [14]. This inspired us to implement an ensemble stacking the models from the previous tasks. We selected the three

best performing methods based on their hold-out cross-validation performance during training. This achieves better performance than selecting all models, because of the performance gap. Further we test different methods for combining the pre-trained classifiers; averaging their predicted probabilities, using a majority vote and training a logistic regression model on top of their predicted probabilities. Averaging resulted in a slightly higher performance compared to the other methods, although they do not differ a lot. In Table Ia and Ib the results of the averaging ensemble are reported. On the MIT-BIH dataset all stacking approaches are worse than the best performing model (UNET), which we suspect is caused by the performance gap. In contrast, we see that the stacked ensemble perform exceptionally well with respect to every metric on the PTB dataset; the only model performing better is the boosted CNN-based ensemble. Consistent with previous results, it seems that the stacking ensemble is able to leverage the predictive power of several, individually well-performing models simultaneously to increase the overall performance.

CNN-Based Boosting: Boosting is another widely used ensemble method which combines multiple weak learners trained on different subsections of the data into a single classifier. We construct an ensemble using the Adaboost algorithm [2]. However, instead of using a forest of decision trees as is most commonly done, we use CNNs as our base learners. In order to train our model within reasonable time, each base learner is restricted to a few number of convolutional and dense layers. On the PTBDB dataset this method attains top performance with respect to every metric. Moreover, we observe that boosting significantly increase the performance of each base learner. The MIT-BIH dataset is more challenging for this technique. Firstly, we observe that due to the high class-imbalance, the base learners struggle with performing well on all classes. Secondly, the size of the dataset forces us to significantly restrict the size of our base learners due to computational limitations. We find that down-sampling the majority class and up-sampling the minority classes remedies both these problems to some degree. However, the final model still performs significantly worse than all other CNN-based models on the MIT-BIH dataset. It might therefore seem that boosting with CNNs as base learners is particularly well suited for smaller, somewhat balanced datasets.

D. Task 4: Transfer Learning

Transfer learning is a well-known method to transfer knowledge between two different but related problems. It is often used to overcome the deficit of training data in one problem and increase performance [8]. Therefore, we train our models on the larger MIT-BIH data and then retrain selected layers on PTBDB data.

Transfer learning on ResNets can show a performance increase on ECG classification tasks [8]. We evaluate three different approaches for our model: First, only the fully connected layers are retrained. Second, the entire model is retrained. In the third approach, we start by retraining the fully connected layers and upon convergence the entire model is fine-tuned. The first two approaches perform significantly worse than the ResNet trained on PTBDB data only. The third model performs similar to the original ResNet (see Ib). We suspect, that there is no additional knowledge transferred from the MIT-BIH training. However, retraining the model requires less computational power.

For transfer learning with RNN based models, we choose two best performing architectures, which are Bidirectional LSTM and ConvRNN. For both architectures, we evaluate the same first two approaches as for the ResNet models. Completely retraining the models with pretrained weights gives much better performance compared to only training the fully connected layers for both architectures. The resulting models for both architectures also outperform their non-transfer learning counterparts in all evaluation metrics. This outcome is almost orthogonal to that of ResNet transfer learning. A plausible reason is that there is more room for improvement for RNN models compared to ResNet models. It is also worth noting that RNN based transfer learning models converge faster than their original counterparts. In table Ib only the best performing transfer learning models are provided.

E. Discussion

We tested and evaluated different state of the art deep learning methods for classifying ECG timeseries. It is important to mention, that the baseline is already high performing with accuracies of 0.9852 and 0.9893, respectively. It is therefore challenging to achieve a significant improvement. RNN based models, including Vanilla RNN and bidirectional LSTM, perform worse than the baseline for both dataset. We conclude that purely RNN-based methods are less suited than CNNs for the given problem and we identify capacity limits and non-parallelizability as plausible causes. However, CNN based methods show great potential and are able to handle both class imbalance and the sequential nature of the data. On MIT-BIH data, U-Net performs best and achieves a significant increase of F1-score and also improves other evaluated metrics. On PTBDB data, a boosting ensemble of simple CNNs achieves best performance. By using stacking ensemble methods we also ob-

serve an increase in performance on the PTBDB dataset. Furthermore, transfer learning significantly improves the original counterparts for RNN based models. In contrast, no improvement can be reached by transfer learning on the CNN based models. Overall, we have empirically demonstrated that deep learning based methods are able to attain very high performance on both the PTDB and MIT-BIH datasets.

Model	F1-Score	Accuracy
Baseline	0.9175	0.9852
Vanilla CNN	0.9072	0.9841
Vanilla RNN	0.8582	0.9728
ResNet	0.9089	0.9842
Bidirectional LSTM	0.8917	0.9782
ConvLSTM	0.9154	0.9842
U-Net	**0.9504**	**0.9918**
Stacking	0.9300	0.9893
Boosting	0.7460	0.9333

(a) Model Performances on MIT-BIH Dataset

Model	F1-Score	Accuracy	AUROC	AURC
Baseline	0.9926	0.9893	0.9873	0.9911
Vanilla CNN	0.9943	0.9918	0.9886	0.9917
Vanilla RNN	0.9515	0.9309	0.9244	0.9495
ResNet	0.9960	0.9942	0.9925	0.9947
Bidirectional LSTM	0.9663	0.9509	0.9306	0.9511
ConvLSTM	0.9827	0.9749	0.9647	0.9747
U-Net	0.9871	0.9814	0.9818	0.9881
Stacking	0.9945	0.9959	0.9937	0.9953
Boosting	0.9957	**0.9966**	**0.9950**	**0.9963**
TL: ResNet	**0.9962**	0.9945	0.9924	0.9945
TL: ConvRNN	0.9928	0.9896	0.9879	0.9916
TL: Bidirectional LSTM	0.9762	0.9656	0.9564	0.9695

(b) Model Performances on PTB Database

TABLE I: Model Performances; the largest value for each column is highlighted.

REFERENCES

[1] R Bousseljot, D Kreiseler, and A Schnabel. Nutzung der ekg-signaldatenbank cardiodat der ptb über das internet. 1995.

[2] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[3] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. Ch. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley. PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation*, 101(23):e215–e220, 2000 (June 13). Circulation Electronic Pages: http://circ.ahajournals.org/content/101/23/e215.full PMID:1085218; doi: 10.1161/01.CIR.101.23.e215.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[5] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[6] Enbiao Jing, Haiyang Zhang, ZhiGang Li, Yazhi Liu, Zhanlin Ji, and Ivan Ganchev. Ecg heartbeat classification based on an improved resnet-18 model. *Computational and Mathematical Methods in Medicine*, 2021, 2021.

[7] Mohammad Kachuee, Shayan Fazeli, and Majid Sarrafzadeh. ECG heartbeat classification: A deep transferable representation. volume abs/1805.00794, 2018.

[8] Mohammad Kachuee, Shayan Fazeli, and Majid Sarrafzadeh. ECG heartbeat classification: A deep transferable representation. In *2018 IEEE International Conference on Healthcare Informatics (ICHI)*. IEEE, jun 2018.

[9] Xinrui Lyu, Matthias Hüser, Stephanie L. Hyland, George Zerveas, and Gunnar Rätsch. Improving clinical predictions through unsupervised time series representation learning. *ArXiv*, abs/1812.00490, 2018.

[10] George B Moody and Roger G Mark. The impact of the mit-bih arrhythmia database. *IEEE Engineering in Medicine and Biology Magazine*, 20(3):45–50, 2001.

[11] M.M. Al Rahhal, Yakoub Bazi, Haikel AlHichri, Naif Alajlan, Farid Melgani, and R.R. Yager. Deep learning approach for active classification of electrocardiogram signals. *Information Sciences*, 345:340–354, 2016.

[12] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[13] Senior Sainath, Vinyals, Haiyang, and Sak. Convolutional, long short-term memory, fully connected deep neural networks. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, 2015.

[14] M. Paz Sesmero, Agapito I. Ledezma, and Araceli

Sanchis. Generating ensembles of heterogeneous classifiers using stacked generalization. *WIREs Data Mining and Knowledge Discovery*, 5(1):21–34, 2015.

[15] Tavakoli Siami-Namini and Namin. The performance of lstm and bilstm in forecasting time series. *IEEE International Conference on Big Data*, 2019, 2019.

[16] Laurens van der Maaten and Geoffrey Hinton. Viualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 11 2008.

[17] Dengqing Zhang, Yuxuan Chen, Yunyi Chen, Shengyi Ye, Wenyu Cai, and Ming Chen. An ecg heartbeat classification method based on deep convolutional neural network. *Journal of Healthcare Engineering*, 2021, 2021.

III. Appendix

A. Further Results

In our report we mention further models, that we evaluated, but did not achieve best performances. They are summarized in Table II.

Model	F1-Score	Accuracy
Baseline	0.9175	0.9852
Vanilla LSTM	0.1863	0.8282
Stacking All: Averaging all models	0.9197	0.9871
Stacking All: Majority Vote	0.9139	0.9859
Stacking All: Logistic Regression	0.9869	0.9333
Stacking Top 3: Averaging all models	0.9300	0.9893
Stacking Top 3: Majority Vote	0.9315	0.9889
Stacking Top 3: Logistic Regression	0.9193	0.9865

(a) Additional Model Performances on MIT-BIH Dataset

Model	F1-Score	Accuracy	AUROC	AURC
Baseline	0.9926	0.9893	0.9873	0.9911
Vanilla LSTM	0.8386	0.7220	0.5	0.7220
Stacking All: Averaging	0.9969	0.9955	0.9923	0.9941
Stacking All: Majority Vote	0.9964	0.9948	0.9934	0.9953
Stacking All: Logistic Regression	0.9974	0.9962	0.9943	0.9958
Stacking Top 3: Averaging	0.9971	0.9956	0.9937	0.9953
Stacking Top 3: Majority Vote	0.9964	0.9948	0.9926	0.9946
Stacking Top 3: Logistic Regression	0.9971	0.9958	0.9937	0.9953

(b) Additional Model Performances on PTB Database

TABLE II: Model Performances

B. U-Net: Additional Plots

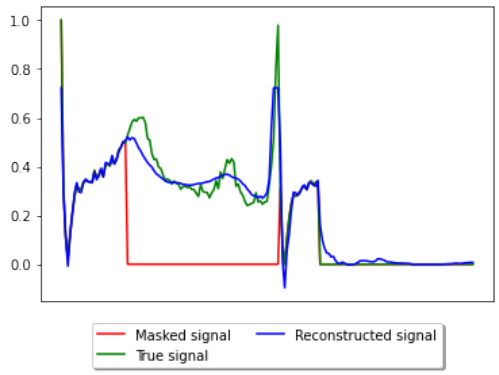

Fig. 2: A quite typical partially masked sequence. Red line is the input signal to the U-Net, the blue line is the reconstructed signal and the green line is the ground truth.

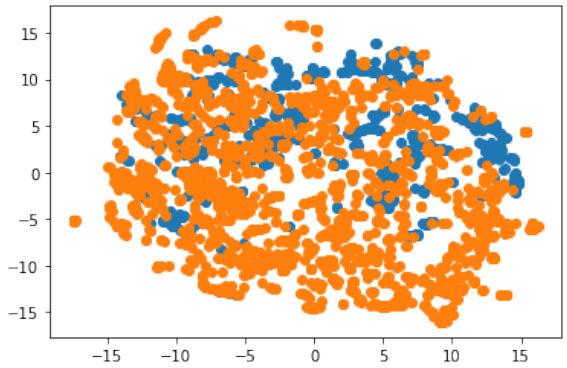

Fig. 3: t-SNE of embedding on PTBDB dataset

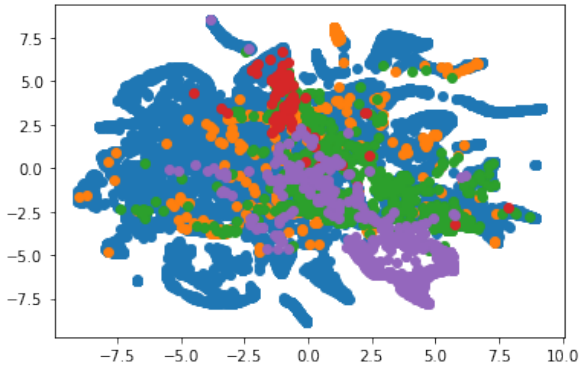

Fig. 4: t-SNE of embedding on MIT-BIH dataset