A1. Background on the team

Competition Name: ICR - Identifying Age-Related Conditions

Team Name: room722

Private Leaderboard Score: 0.30626

Private Leaderboard Place: 1st

Name: Roman Ottenbakher

Location: Russia

Email: [r\_o\_m\_a\_h\_a@mail.ru](mailto:r_o_m_a_h_a@mail.ru)

A2. Background on me

I am a chemist working in science, and I am PhD in physical chemistry. To be honest, I like data science so much (and more than my current job), that I want become a data scientist in the future. I had some experience in data science competitions, previously participated in competitions at Kaggle and other platforms. I entered this competition to try my best and win it. Overall, I spend 2 months solving this comp.

A3. Summary

The model I used is DNN based on Gated Residual Network (GRN) and Variable Selection Network (VSN). They were described as a part of the bigger model for time series prediction in paper “Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting” [1]. GRN was tuned using novel “smish” activation function [2] instead of vanilla “relu” and provided the flexibility to the model to apply non-linear processing only where needed. VSN allowed the model to softly remove any unnecessary noisy inputs which could negatively impact performance. Together, VSN and GRN helped improving the learning capacity of the deep neural network model.

A4. Features Selection / Engineering

The model used all 56 anonymous features. No external data were used.

The binary [‘EJ’] feature was transformed to 0/1. The missing values were filled with median values of the train dataset if there were no missing values in the train, and with feature’s min minus max values otherwise.

No standardization or normalization was employed (according to my experiments, it worsened the performance).

A5. Training Method(s)

The model was trained using standard methods from tensorflow/keras library: the gradient descent with Adam optimizer, decreasing learning rate and early stopping after 25 epochs without improvements. The dropout values for 3 VSN layers were 0.75, 0.5 and 0.25, respectively. Such big values were necessary for the model not to overfit, but also led to very high variance in the validation score. The training was performed using 10-fold multi-label cross-validation. The first label was the target, and the second label was “the hardness to predict”. The latter was derived from the baseline model, which was actually the same model but with only 2 VSN layers. The hardness was postulated as 1 if (y\_true=1 and y\_pred<0.2) or (y\_true=0 and y\_pred>0.8), 0 otherwise. The training was repeated for 10 times for each fold due to high variance in scores. In the end, I manually picked two models with the best validation score for each fold and concatenated them in equal weights (arithmetic mean).

A6. Interesting findings

There were not many. The most fascinating finding for me personally was that high variance in the validation scores allowed to obtain models with better than average score and high generalization capability even on as small dataset as we had.

A7. Model Execution Time

It took nearly 10 hours to train 100 single models (10 repeats of 10-fold cross-validation). And the inference time is nearly 15 seconds (major part of which was tensorflow building the model). To speed up inference time to 1-2 seconds, one can save built tensorflow model on disk and then read it and load model weights into it, which is much faster.

A8. References

[1] https://arxiv.org/abs/1912.09363

[2] https://www.mdpi.com/2079-9292/11/4/540