|  |  |  |
| --- | --- | --- |
| **Spring 2024** | **Report #4– 04/28/2024** | **Dingyi Nie** |

**——————————————————————————————————————**

**I. Task Achieved Last Week**

**——————————————————————————————————————**

* Met with Yixing and Sanket, learned about DFT/RFT, LNT, and Sanket’s approach of handling earthquake sensor data
* Experimented with different SSL configs and achieved a performance gain by manually searching the best hyperparameter set
* Introduced LNT into pipeline and experimented with it

**——————————————————————————————————————**

**II. Feedback and Interaction**

**——————————————————————————————————————**

* **Prof. Kuo’s Feedback**
  + Last week’s:
    - Padding and alignment: make sure the end is aligned
    - Decision based on windowed intervals (window len = avg seq len)
    - In certain layer, combine all channels together
  + This week’s
    - Placeholder

**——————————————————————————————————————**

**III. Report**

**——————————————————————————————————————**

As suggested by Prof. Kuo last week, this week I first played with the hyperparameters in the SSL module. Recall that our initial feature shape is (BATCH\_SIZE, C=1, 336, 7), and the 4-layer SSL module was defined as follows (call it config.1):

1. Kernel size = (4,1), stride = (4,1): this reduces the last 3 dimensions into (4, 84, 7)
2. Kernel size = (4,1), stride = (4,1): this reduces the last 3 dimensions into (16, 21, 7)
3. Kernel size = (4,2), stride = (4,2): this reduces the last 3 dimensions into (128, 5, 3)
4. Kernel size = (4,2), stride = (4,2): this reduces the last 3 dimensions into (1024, 1, 1)

There was a question raised by Prof. Kuo that we are progressively merging consecutive sensors, rather than fusing all of the channels together. This may introduce some unnecessary assumptions. So, I adjusted the parameters so that the combination of all sensor channels happens in the same layer as follows (config.2):

1. Kernel size = (4,**7**), stride = (4,1): this reduces the last 3 dimensions into (28, 84, 1)
2. Kernel size = (4,1), stride = (4,1): this reduces the last 3 dimensions into (112, 21, 1)
3. Kernel size = (4,**1**), stride = (4,**1**): this reduces the last 3 dimensions into (448, 5, 1)
4. Kernel size = (4,**1**), stride = (4,**1**): this reduces the last 3 dimensions into (1792, 1, 1)

This adjustment will result in a larger reduced feature space of 1792-dim. This adjustment alone will boost the performance to an **85.3%** AUROC score, compared to last week’s best 84.1%.

Since we were using LR for downstream task head (classifier) initially, I thought maybe XGBoost can bring another performance boost, so I tried it, and it turns out our pipeline reaches an **87.2% score** already.

Next, since we discussed about the slicing-combining paradigm to deal with varying input sequence length, I thought about improving the way I used for aligning input episodes. Previously, I simply padded all 7-channel episodes into the same length (max\_seq\_len = 336) with 0’s in the end. Prof. Kuo suggested we make sure the episodes are aligned by the end (current) time, rather than start time, because we are making predictions for all of them at the time being, rather than some fixed time that’s either present or in the future, i.e. hour 336. So, I tried adjusting the padding to **the beginning** of the sequence. However, this adjustment caused a performance drop to only ~60% AUROC score.

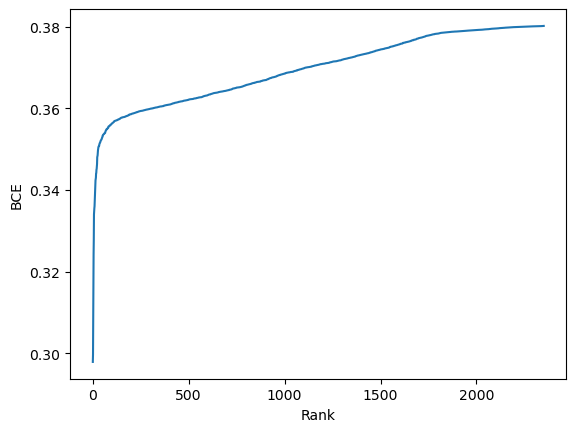
After carefully examining possible reasons, I think I found a key point – when we set the kernel length same as the strid in the Saab transform, **it will drop some values at the end of the sequence if the unified sequence length is not divisible by kernel length**. When we pad in the beginning instead of the end, this may result in meaningful values being dropped. Hence, I redesigned the SSL module again with config.3:

1. Kernel size = (3,7), stride = (3,1): this reduces the last 3 dimensions into (21, 112, 1)
2. Kernel size = (4,1), stride = (4,1): this reduces the last 3 dimensions into (84, 28, 1)
3. Kernel size = (4,1), stride = (4,1): this reduces the last 3 dimensions into (336, 7, 1)
4. Kernel size = (7,1), stride = (7,1): this reduces the last 3 dimensions into (2352, 1, 1)

This setting makes sure each layer is an exact division.

With the new setting and a beginning padding, our model’s evaluation score comes to **88.6%.**

Finally, I tried DFT for feature selection on the 2352-dim feature. The BCE-Rank curve is shown here:



I have tried a couple of settings of n\_selected, but the performance only decreases. This means the 2352-dim feature space we obtain from SSL is already pretty meaningful and compact, without too much redundancy or noise that could affect downstream classifier’s performance negatively.

Here list all the results by far (rows in grey are baseline competitors, reported in [Raindrop](https://arxiv.org/pdf/2110.05357.pdf) (ICLR22), plus a newest research [ViTST](https://proceedings.neurips.cc/paper_files/paper/2023/file/9a17c1eb808cf012065e9db47b7ca80d-Paper-Conference.pdf) (NeurIPS23) on the same topic):

|  |  |
| --- | --- |
| Method | AUROC |
| Lin. imp. + SSL (config.1) + LR | 81.9% |
| XGB imp. (strat.1) + SSL (config.1) + LR | 84.1% |
| XGB imp. (strat.2) + SSL (config.1) + LR | 84.0% |
| XGB imp. (strat.1) + SSL (config.2) + LR | 85.3% |
| XGB imp. (strat.1) + SSL (config.2) + XGB Classifier | 87.2% |
| XGB imp. (strat.1) + SSL (config.3) + XGB Classifier | **88.6%** |
| Transformer | 83.2% |
| Transformer-mean | 84.1% |
| GRU-D | 83.9% |
| SeFT | 78.7% |
| mTAND | 80.4% |
| Raindrop | 87.0% |
| ViTST | **89.2%** |

**——————————————————————————————————————**

**IV. Next Steps**

——————————————————————————————————————

* Finish designing and coding for ensemble learning
* Experiment with slicing methods for enabling **early sepsis prediction** rather than classification
* Experiment with possible ways of utilizing demographic info and clinical text data (maybe word bags?)

**——————————————————————————————————————**

**V. Milestone**

——————————————————————————————————————

* A new performance gain in P19 classification task; now best AUROC score = 88.6%
* Codes are available here: <https://github.com/d9sus4/GL-TS>