CSCI-5952 E01: BIG DATA SCIENCE: TERM PROJECT PROGRESS REPORT(DRAFT)

Rumana Sultana
College of Engineering, Design & Computing
The University of Colorado Denver
Colorado, USA.
rumana.sultana@ucdenver.edu

Project Title— **Automatic Sleep Stage Classification Using Deep Transfer Learning**

.

Role— Project planning, project design, data preparation/pre-processing, project development and project implementation

I. PROBLEM STATEMENT AND BACKGROUND

One invisible enemy, lot of deaths and a complete trapped world. This situation forces us for alternative solution with minimized human attachment. So, self-driven or data-driven technology has significant importance for current pandemic world to confirm the human safety. My project proposal includes a unique idea to initiate an automatic sleep stage classification to improve the human lifestyle. Depression is a long-term disorder of human life. It is increasing now a days due to the lockdown of COVID-19 outbreak. Hundreds of thousands of people are suffering from sleep disorder caused by depression and other underlying health condition that consequence to heart attack and even death. Therefore, sleep stage analysis has a significant importance in clinical science to diagnose the symptoms and diseases. However, still it is a hard task as the labeling of sleep stages from polysomnography (PSD) is done manually by clinicians applying a developed guideline. Unfortunately, it is a highly time-consuming process with the involvement of human resources and cost. Moreover, accurate performance is possible from the analysis of a long period data of a subject. Home based sleep monitoring with edge devices come in attraction to overcome the limitations and difficulties. Current world has innovated a portable home or mobile devices for Electroencephalography (EEG) and Electrocardiogram (EKG) for monitoring sleep diagnosis and heart rhythms. However, the accuracy and reliability are not satisfactory that people still dependable on hospital diagnosis. Machine learning and deep learning model like CNN, DNN, RNN LSTM, etc. has been widely applied for sleep study [1-5]. But machine and deep leaning model need high performance computing resources with big dataset. To ensure these resources to a home or mobile device are challenging. That is why an edge device is depended to a cloud server for decision making data. This process needed continuous internet service that causes latency and sometimes failure of service due to unavailability of connection. The survey paper [6-7] describe about edge computing (DL on edge) and model compression and acceleration for DNN that motivates me to develop a data-efficient model (less data used) for using in an edge device with a performance accuracy 84%-90%

II. THE DATA SOURCE(S) USED

A. *Sleep-EDF Database*

The phusionet database Sleep-EDF [11] which is used by the DeepSleepNet [1] that includes 197 whole-night Polysomnographic sleep recordings, including EEG, EOG, Chin EMG, and event markers of around 20 subjects. Some records also have respiration and body temperature. The corresponding hypnograms were manually scored by well-trained technicians. I have already downloaded it from [12]. In my project, single-channel EEGs (F4-EOG(Left), Fpz-Cz and Pz-Oz) is using from this public sleep data set. Some characteristics of the data set is given below:

- **Downloaded Link:** https://www.physionet.org/static/published- projects/sleep-edfx/sleep-edf-database- expanded-1.0.0.zip.
- **Data and Annotation Files:**
  -PSG.edf files- whole night recordings of EEG, EOG, submental chin EMG and even marker.
  -SC*PSG.edf- contain oro-nasal respiration and rectal body temperature.
  *Hypnogram.edf- annotation of sleep pattern correspond to PSGs consist of sleep stages W, R,1,2,3,4, M and? (not scored)
  -Unrecorded signals removed from ST*PSG.edf
  -Stages: W=wakefulness, R- rapid eye movement, N1-light sleep, N2-deeper sleep, N3&4-deep sleep.
- **Sleep Cassette Study and Data:**
  -153 SC*files of healthy Caucasians aged 25-101, with no medication.
  -**Variables**: Files are named in the form of SC4ssNE0-PSG.edf. Where, ss=subject number, N=night.
  -**Missing Values**: The first night of subjects 36 and 52, and the second night of subject 13, were lost due to failing cassette or laserdisk.
  -EEG and EOG signals were sampled at 100Hz.
  -EMG signal was high-pass filtered, rectified and low-pass filtered and resulted EMG, expressed in uV ms (root-mean-square) was sampled at 1 HZ.
  -Oro-nasal airflow, body temperature and event marker sampled at 1 HZ.

- **Sleep Telemetry Study and Data:**
  -44 ST*files of temazepam effects on sleep in 22 Caucasians males and females, difficulty with falling asleep.
  -Variables: Files are named in the form of ST7ssNJ0-PSG.edf, where ss=subject number, N=night
  -EOG, EMG, and EEG signals sampled at 100Hz.

III. Methodology of My Data Science Solution

The main goal of this project is to design a data-efficient model with less data used. Most of the small dataset dependent DL sleep staging model's performance hampered by the data variability or mismatch and data inefficiency. Moreover, almost all edge device or AI sensor have limited resources with storage, internet connectivity and proper GPU support to deploy large DL model on it. A cloud or data server dependency makes the end node latent and less effective. This solution is expected to reduce the data dependency and increase the data efficiency as well as touched the highest possible accuracy on the AI device with limited resources. AI device can improve their intelligence in instant decision-making support. The ideas used on [5] and [13] helped me to define a tentative design includes 1) to train a model based on [1] "DeepSleepNet: a model for automatic sleep stage scoring based on raw Single-Channel EEG" approaches that used two CNNs with different filter sizes and bidirectional-LSTMs techniques. This model will be deployed into a cloud server (source domain) 2) then, the pre-trained model from step 1 will be finetuned into a small sub-model (target domain) by using deep transfer learning techniques to enable transferring knowledge from the big dataset (on cloud) to the small cohort that will deploy into an edge device for automatic sleep staging. 3) Unknown user-level data at edge device will be passed through the re-trained (sub-DSN) model first for a prediction with a threshold confidence level. 4) If score below the threshold, then data will be sent to the source model on cloud.

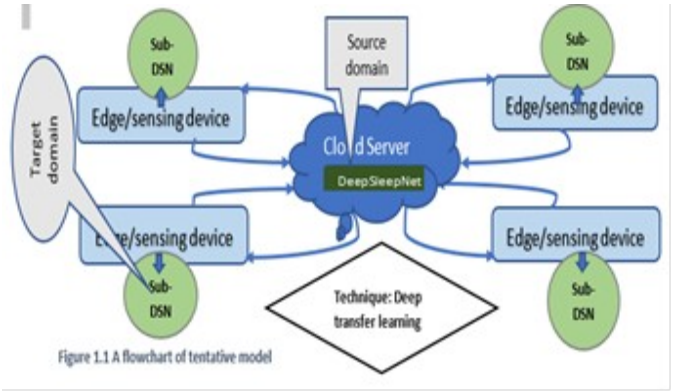


Figure 1: Flow chart

A. Source Model

In paper [5], they used two source model trained based on MASS database. They used the pre-trained model SeqSleepNet+ and DeepSleepNet+. In my project, I am using the DeepSleepNet described in paper [1] as base model. The network receives raw signal as input.This model is trained based on two CNN with 4 convolutional layers. Each convolutional layer is linked with batch normalization and rectified linear units (RELU) activation. A biRNN depend on the LSTM cell and designed with two bi-directional LSTM layers. Design of DeepSleepNet is shown in Figure 2. I am trying to increase the use of convolutional layers in this network to increase the performance.

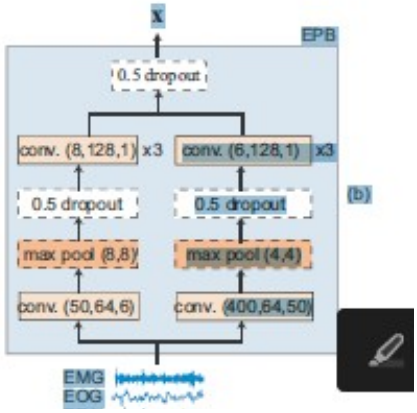


Figure 2: DeepSleepNet [5]

B. Target Model

Then SleepEDF dataset is used as target domain to finetune the source model in target domain.

C. Transfer Learning

Normally, Ds={Xs, Ys} denote the source domain with Xs is the feature space and Ys is the label space. And Ts is the task in the source domain with conditional probability P(ys|xs), where xs belongd to Xs and ys belongs to Ys. Similarly, Dt ={Xt,Yt} denote the target domain with Xt is the feature space and Yt is the label space. And Tt is the task in the target domain with conditional probability P(yt|xt), where xt belong to Xt and yt belong to Yt. The objective of transfer learning is to improve the P(yt|xt) with the information achieved from Ds and Ts. In my project same theory applied, with a source domain trained based on DeepsleepNet and then finetuned the learned knowledge to the target domain. WLOG, the pretraining process minimize the loss Ls over source domain data with the model parameter, $\boldsymbol{\theta}$. The pretrained model is used the starting point in the target domain. In transfer learning, a subset of pretrained model's parameter finetuned with the target domain data whereas the rest remains unchanged.

IV. Implementation and Results

The main method I am using is deep transfer learning techniques on edge computing. Because Deep learning is extremely expensive due to its' highly configured computing requirement and time consumption characteristics. Limited

resources devices cannot ensure these requirements. But transfer learning is a process of deep leaning/machine learning through which knowledge can be transferred form one big model to a small model. This approach has an outstanding performance in the circumstance of lack of data, speed, and system's limitation. These difficulties reduce the scope of edge computing. So, the method is the best match to solve the problem. My ultimate effort will be 84%-90% accuracy achievement performed by both source model and target model shown in Figure 1.1 in the training and testing outcome analysis based on these two databases. I have started to train my based model with the tutorial described in [14]. They have developed the model based on the paper [15]. Currently, I used a Butterworth filter instead of their filtering process to process the raw data and my accuracy increased almost 10% than them. I used 30 HZ cutoff frequency to eliminate high frequency noise as my filter parameter.

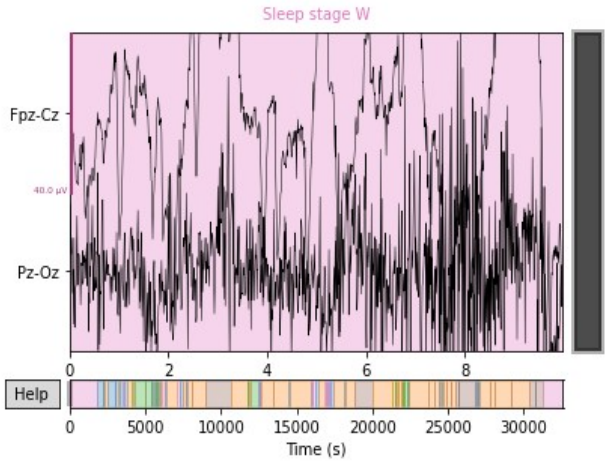


Figure 3: Plot of raw data

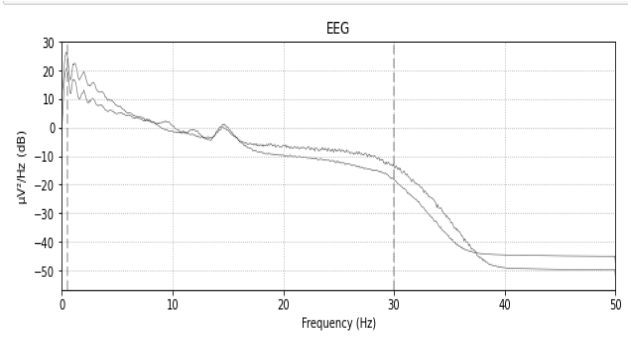


Figure 4: plot of filtered data with Butterworth filter.

I have trained the model with the preprocess data that is shown in Figure 5 and 6. The primary training examples are: **Training set=18845, Validation set=4094 and test set=9850.**

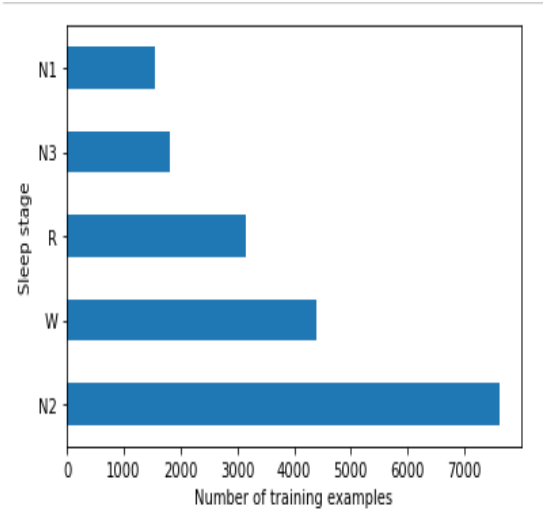


Figure 5: Number of training examples

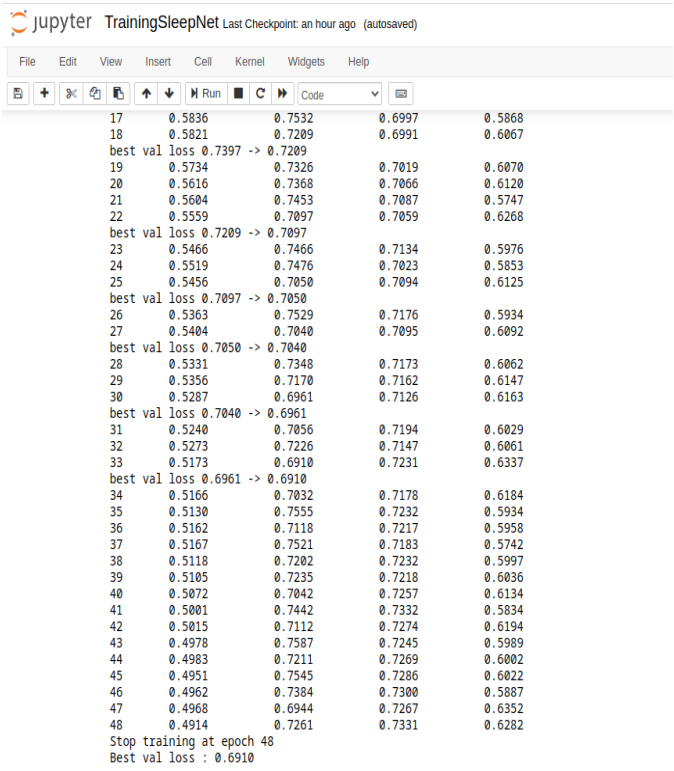
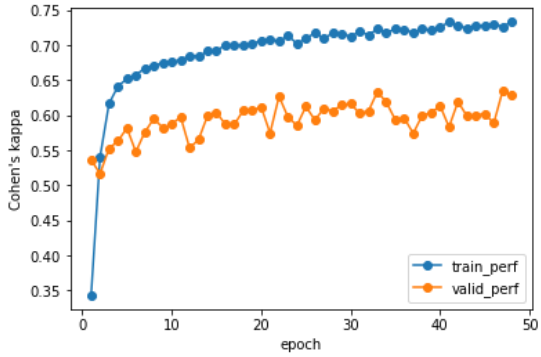


Figure 6: Primary Training.

The preliminary training shows a test balance accuracy of 0.75 and test Cohen's kappa of 0.714. The accuracy is not as my expectation. So, I am trying to modify the base model to increase performance. Preliminary result is shown in Figure 7 and 8.



F

Figure 7: Visualization of loss

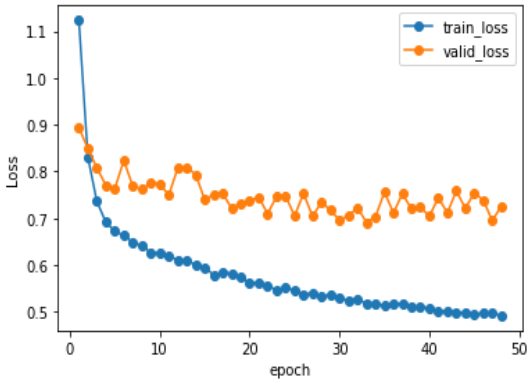


Figure 8: Visualization of performance.

Transfer learning model is developed based on the base model performance. The preparation of data has been done based on the method on paper [5] that is shown in Figure 9 and 10. Unfortunately, no result could be achieved to show due to killing most of the time on developing the base model.

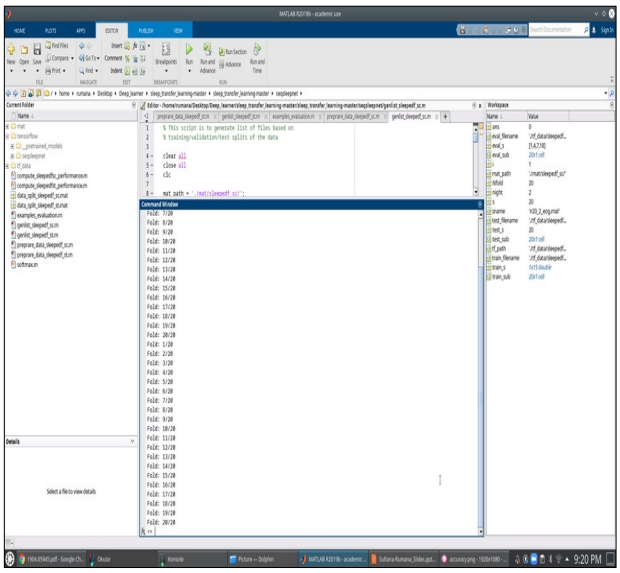


Figure 9: MATLAB Data preprocessing (SC)

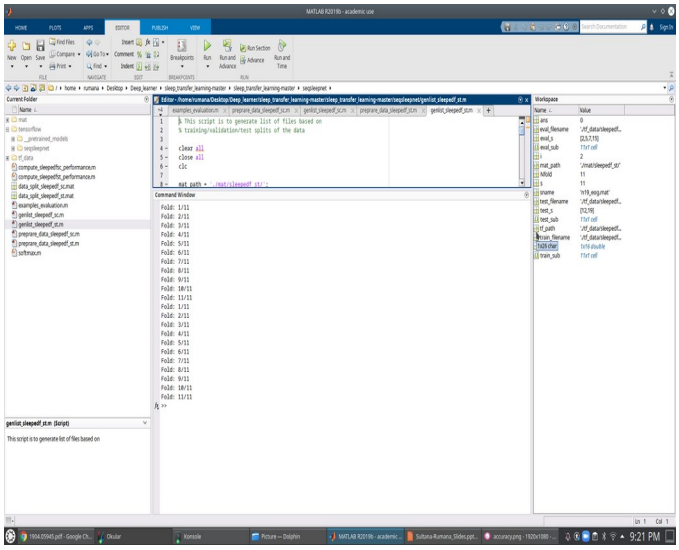


Figure 10: MATLAB data preprocessing (ST)

As my primary model accuracy is not as per my expectation, I have trained DeepSleepNet but I couldn't fixed some problems that could be happened due to data preprocessing gap and the weights becomes zero in fold0. That's why, I couldn't use this model as a base model shown in Figure 11.

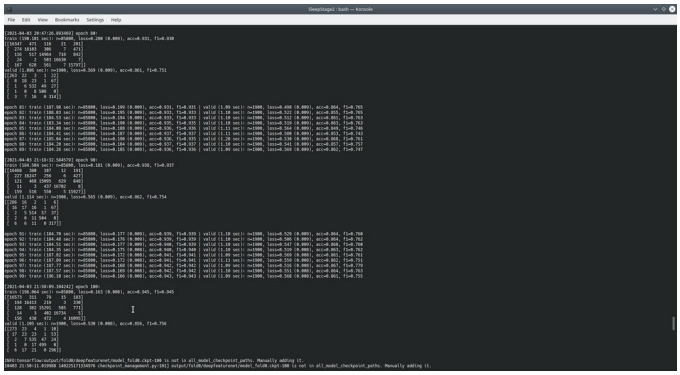


Figure 11: DeepSleepNet training session.

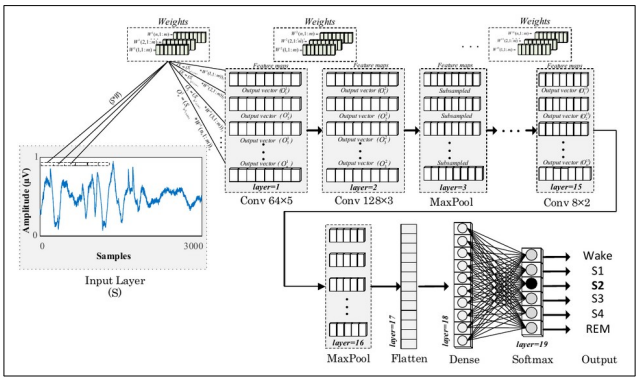


Figure 12: 1D CNN model for sleep staging[16]

I have developed a new model with the pre-processed database based on paper [16]. In this paper, they used the model shown in Figure 12 and Figure 13. I have got the values

almost near to them. They claimed that they got about 88-89% accuracy for training, validation and testing for sleep-edfx dataset with this model. The model developed by me is shown in Figure 14.

Num.	Layer Name	No. of Filter × Kernel Size	Region/Unit Size	Layer Parameters	No. of Trainable Parameters	Output Size
1	1D Conv	64 × 5	-	ReLU, Stride = 3	384	64 × 999
2	1D Conv	128 × 5	-	ReLU, Stride = 1	24,704	128 × 997
3	MaxPool	-	2	Stride = 2	0	128 × 498
4	Dropout	-	-	Rate = 0.2	0	128 × 498
5	1D Conv	128 × 13	-	ReLU, Stride = 1	213,120	128 × 486
6	1D Conv	256 × 7	-	ReLU, Stride = 1	229,632	256 × 480
7	MaxPool	-	2	Stride = 2	0	256 × 240
8	1D Conv	256 × 7	-	ReLU, Stride = 1	262,272	128 × 233
9	1D Conv	64 × 4	-	ReLU, Stride = 1	32,832	64 × 230
10	MaxPool	-	2	Stride = 2	0	64 × 115
11	1D Conv	32 × 3	-	ReLU, Stride = 1	6176	32 × 113
12	1D Conv	64 × 6	-	ReLU, Stride = 1	12,352	64 × 108
13	MaxPool	-	2	Stride = 2	0	64 × 54
14	1D Conv	8 × 5	-	ReLU, Stride = 1	2568	8 × 50
15	1D Conv	8 × 2	-	ReLU, Stride = 1	136	8 × 49
16	MaxPool	-	2	Stride = 2	0	8 × 24
17	Flatten	-	-	-	0	1 × 192
18	Dense	-	64	ReLU, Drop = 0.2	12,352	1 × 64
19	Dense	-	nb_class	Softmax	195	1 × nb_class

Figure 13: Details of used layers and parameters[16]

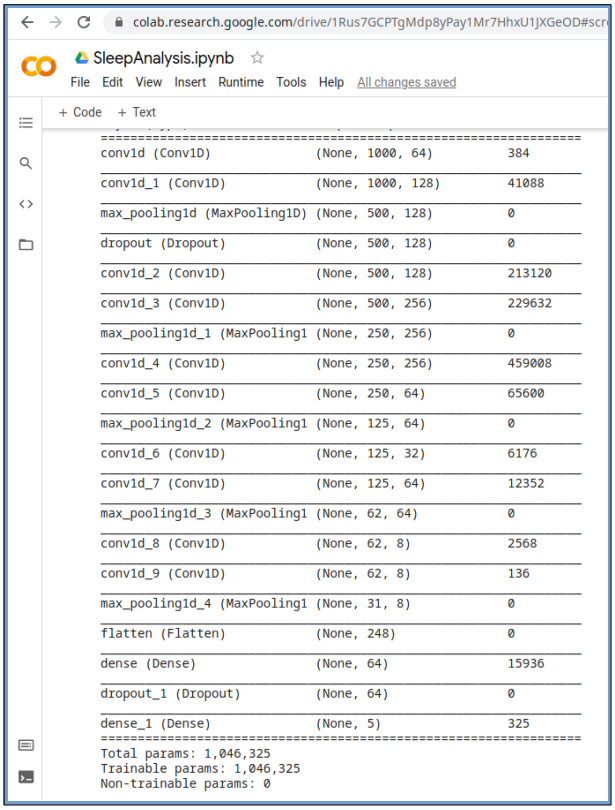


Figure 14: Model developed

First, I tried to fit the model using the primary model I developed with PyTorch data set, but Keras model didn't approved me to fit model with PyTorch data set. Then I researched some on it and come to a conclusion maybe Keras model is not supportive with PyTorch data frame data set. Then I tried to develop a PyTorch model with the same parameters. Again, I was unsuccessful and the reason could be my primary model was trained with 2 dimensional convolution neural network whereas the new model is using 1 dimensional convolution neural network and the data set has prepared with multiple channels. But I need to per-process data set with single channel for fitting into the model. Then I have prepared my data set again suitable for the new model based on [17]. The new dataset has same sleep label with the frequency of labels shown in Figure 15.

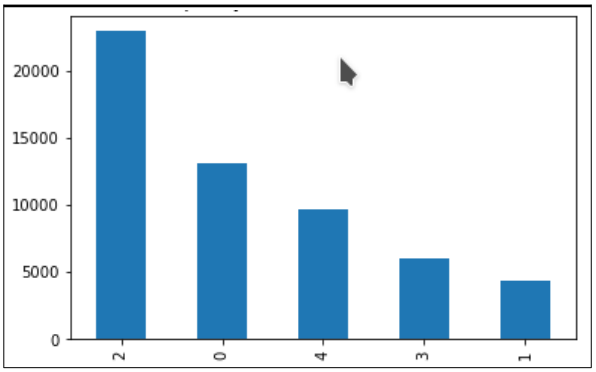


Figure 15: Frequency of the labels in data set

I have trained this model both with unfiltered data set as said in the paper[16] but without normalization and it gave me a overall accuracy and macro F1-score of 85%-78 whereas the DeepSleepNet has overall accuracy and macro F1-score for Sleep-EDF data set is 82.0%-76.9. Training is done for 30 epochs with 45351 training sleep data set of shape (3000,1). The training loss and validation loss are shown in figure 16 and the graph is shown in Figure 17. The accuracy and validation accuracy with epochs is shown in Figure 18. The output from training is shown in Figure 19. The model performance graph for 5 classes is shown in the Figure 20.

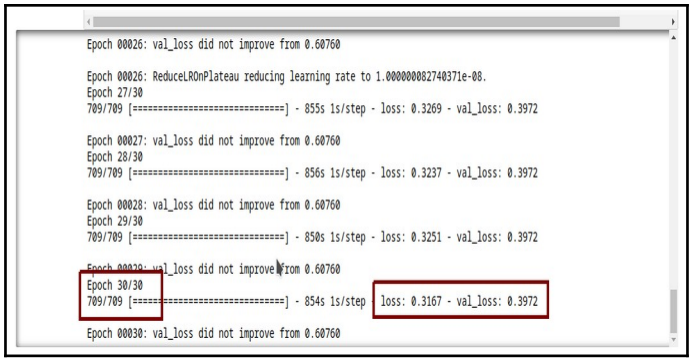


Figure 16: training with unfiltered data set

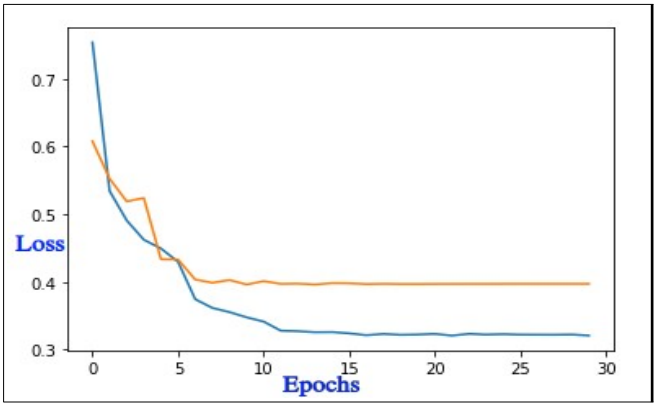


Figure 17: History of loss and validation loss with epochs

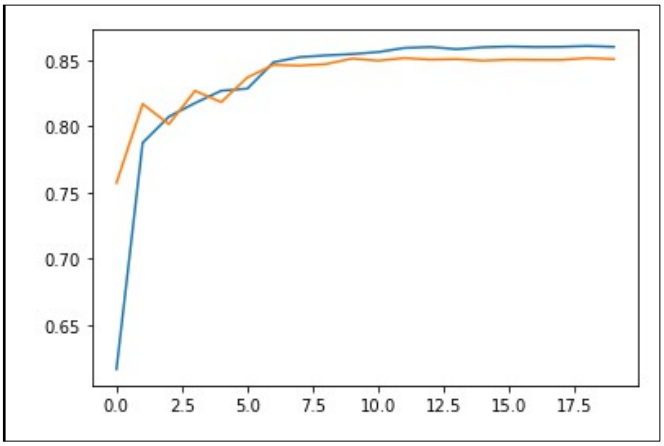


Figure 18: History of accuracy and validation accuracy

F1 score: 0.7799128954339096

	precision	recall	f1-score	support
0	0.93	0.93	0.93	1231
1	0.52	0.35	0.42	435
2	0.87	0.90	0.89	2341
3	0.88	0.85	0.87	613
4	0.77	0.82	0.79	979
accuracy			0.85	5599
macro avg	0.79	0.77	0.78	5599
weighted avg	0.84	0.85	0.84	5599

Figure 19: Model output for unfiltered data.

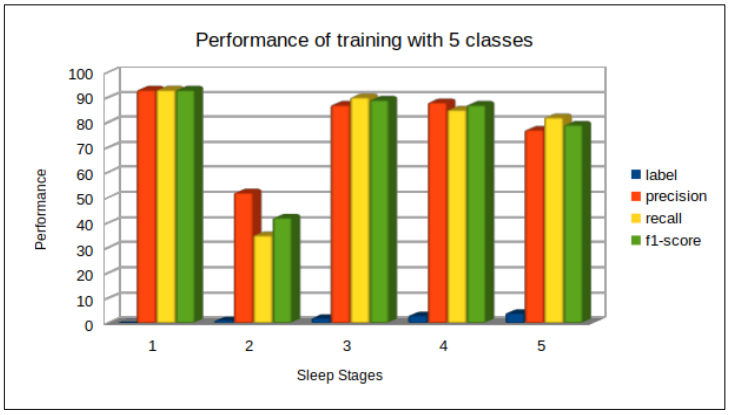


Figure 20: Performance of training

Then I trained the same model with applying Butterworth filter on data set and without normalization. This time the result increase a little that is near about 87%. Unfortunately, I have replaced the training session when modifying and couldn't finish a whole training session after that due to technical problem. I have completed 20 epochs only which give me same result as above. But, before I had completed 30 epochs and got 87.99 % accuracy and I hope if I can do 100 epochs the result will increase as the base paper did 100 epochs with same model.

The new model gave me higher performance than DeepSleepNet. Then I tried use this model as base model to fine tune the target model. Currently, I couldn't this part of my project. I hope after fine tuning this model with transfer learning will give me more better result than the base paper.

Evaluation and Comparison with base model:

In base model [5] they used SeqSleepNet+ and DeepSleepNet+ for source model. In both cases the performance is lesser than the new model I have developed based on [16]. I hope after fine tuning the new model to target model will increase the performance factors obtained by fine tuning by base paper [5] shown in Figure 21. I expect the new trained model can increase the performance factors about 3% more than the base paper [5].

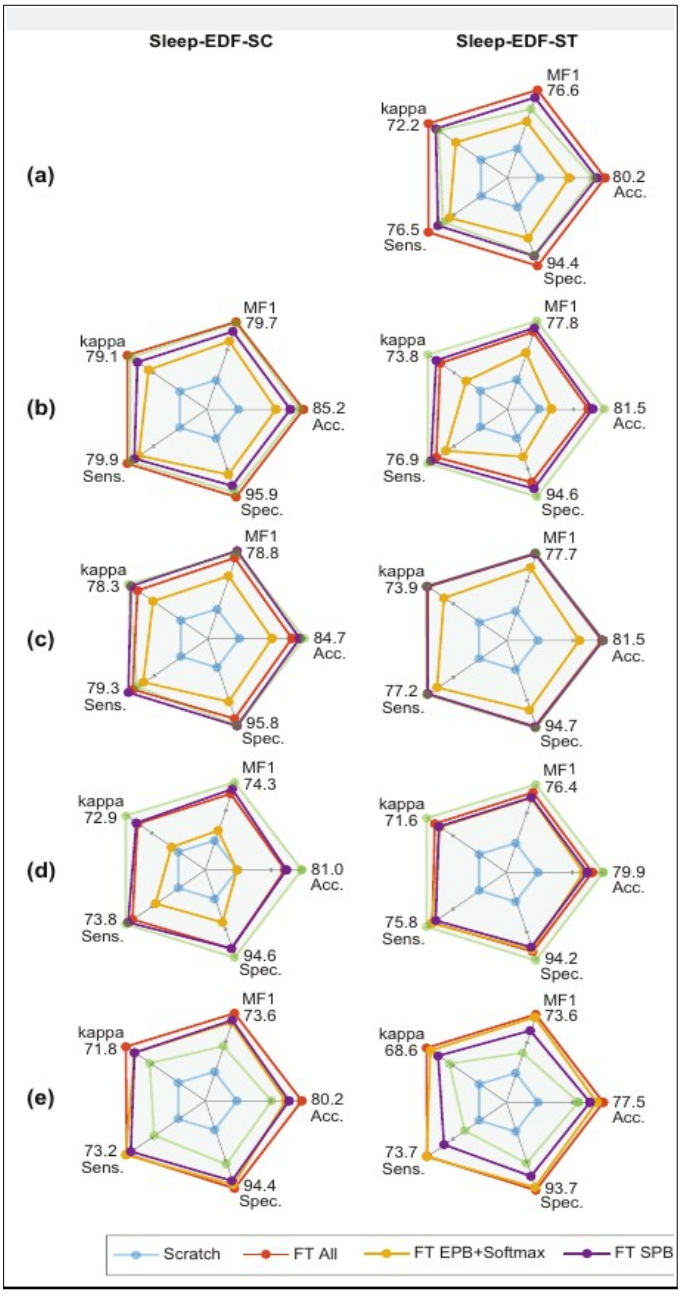


Figure 21: Performance factors obtained by fine tuning in base model[5]

V. Description of Tools is Using

The tools I am using through my project are MATLAB (for data preparation), Python3, TensorFlow 2 for network training and evaluation, NumPy, SciPy, sklearn, h5py, etc. The tools will be employed in case of necessity. The reason of choosing the tools is:

- **MATLAB** has user-friendly EEGLAB toolbox to process the EEG data with applying necessary high pass and low pass filter.
- **Python3** and **TensorFlow 2** is very enriched with machine and deep learning libraries.
- **NumPy** makes easy complex machine and deep learning numerical operations with large dataset.
- **SciPy** contains different modules for optimization, linear algebra, integration and statistics that is very helpful for data analysis.
- **Scikit-learn**, a machine learning library for Python has various algorithms like support vector machine, random forests, and k-neighbors, and it supports Python numerical and scientific libraries like NumPy and SciPy.
- **H5py** is helpful for viewing datasets of different formats in a tabular way or as an image.
- **PyTorch** is open-source machine learning library. I am using this tool currently to develop a base model using tutorial [13]. I will try to increase the performance of this model. If I don't achieve accuracy level, then I will try another way.
- **Keras** is a open source software library in python that is used for deep learning model development. It is easy to DL build model and evaluate it with necessary tools.

  I tried to use PyTorch first for pre-processing data set and training model. But, as Keras is easy to understand and develop a model, I have used Keras instead of PyTorch. In my cases, the Keras model worked better than the primary model using PyTorch data processing and build model techniques. However, it help me how to start and how to find solution as per my necessity. I am a new developer for MNE sleep data set, TensorFlow and Keras and PyTorch. That's why Currently, I have used the most easier library that can help me to understand the data set and how to process a big data set using available tools.

VI. Lesson Learned

I have gained many knowledge by doing this project. First of all I come to familiar with Polysomnogram (PSG) data set of signals and how to process it to use in a deep learning model. I have learned how to use the available tools to build DL model. Before I mainly worked on Matlab platform for deep learning. Though I had used TensorFlow before for another Fall course but this project give me a clear idea about PyTorch, TensorFlow, Keras and other available data processing and deep learning tools. I have learned many important big data DL techniques along with meta learning from my course content. I learnt how meta learning used LSTM method to train a model. And also other some different methods under meta learning topics help me to get idea and make my concept clear. And finally, I come to know how we can use a big data model to a small target model with less memory and data availability domain by applying transfer learning. The project is very interesting. I will continue the project to get a final outcome.

VII. Team Contributions

As I am doing the project alone. 100% of the contribution on this project done by me.

VIII. Future Work and Challenges

First of all my priority future work is to finetune the trained model to target domain. Next, I have a plan to use a pretrained model like Alexnet or Resnet with spectrogram to check if it works better than the current trained network. I have already started to read a meta-learning sleep analysis model. I will work to develop it and modify it in future for my own interest.

IX. Conclusion

Sound sleeps ensure a healthy life. Automatic sleep analysis is high-preference task for current world. Whereas automatic sleep scoring or home-based monitoring is very difficult and less reliable. The approach described above will be able to address the difficulty of sleep research very well and helpful to identify proper solution.

References

[1] Supratak, A., Dong, H., Wu, C., & Guo, Y. (2017). DeepSleepNet: A model for automatic sleep stage scoring based on raw single-channel EEG. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 25(11), 1998-2008. [Google Scholar]

[2] Nguyen, A., Raghebi, Z., Banaei-Kashani, F., Halbower, A. C., & Vu, T. (2016, October). LIBS: a low-cost in-ear bioelectrical sensing solution for healthcare applications. In Proceedings of the Eighth Wireless of the Students, by the Students, and for the Students Workshop (pp. 33-35). [Google Scholar]

[3] Munk, A. M., Olesen, K. V., Gangstad, S. W., & Hansen, L. K. (2018, April). Semi-supervised sleep-stage scoring based on single channel EEG. In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 2551-2555). IEEE. [Google Scholar]

[4] Banville, H., Chehab, O., Hyvarinen, A., Engemann, D., & Gramfort, A. (2020). Uncovering the structure of clinical EEG signals with self-supervised learning. Journal of Neural Engineering. [Google Scholar]

[5] Phan, H., Chén, O. Y., Koch, P., Lu, Z., McLoughlin, I., Mertins, A., & De Vos, M. (2020). Towards more accurate automatic sleep staging via deep transfer learning. IEEE Transactions on Biomedical Engineering. [Google Scholar]

[6] Wang, X., Han, Y., Leung, V. C., Niyato, D., Yan, X., & Chen, X. (2020). Convergence of edge computing and deep learning: A

comprehensive survey. IEEE Communications Surveys & Tutorials, 22(2), 869-904. [Google Scholar]

[7] Cheng, Y., Wang, D., Zhou, P., & Zhang, T. (2017). A survey of model compression and acceleration for deep neural networks. arXiv preprint arXiv:1710.09282. [Google Scholar]

[8] O'reilly, C., Gosselin, N., Carrier, J., & Nielsen, T. (2014). Montreal Archive of Sleep Studies: an open‐access resource for instrument benchmarking and exploratory research. Journal of sleep research, 23(6), 628-635. [Google Scholar]

[9] Iber, C. (2007). The AASM manual for the scoring of sleep and associated events: Rules. Terminology and Technical Specification. [Google Scholar]

[10] J. A. Hobson, "A manual of standardized terminology, techniques and scoring system for sleep stages of human subjects," Electroencephalography and Clinical Neurophysiology, vol. 26, no. 6, pp. 644, 1969. [Google Scholar]

[11] Kemp, B., Zwinderman, A. H., Tuk, B., Kamphuisen, H. A., & Oberye, J. J. (2000). Analysis of a sleep-dependent neuronal feedback loop: the slow-wave microcontinuity of the EEG. IEEE Transactions on Biomedical Engineering, 47(9), 1185-1194. [Google Scholar]

[12] The sleep-EDF database download link, https://www.physionet.org/static/published-projects/sleep-edfx/sleep-edf-database-expanded-1.0.0.zip.Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301,1982].

[13] Li, E., Zeng, L., Zhou, Z., & Chen, X. (2019). Edge AI: On-demand accelerating deep neural network inference via edge computing. IEEE Transactions on Wireless Communications, 19(1), 447-457. [Google Scholar]

[14] https://colab.research.google.com/github/hubertjb/dl-eeg-tutorial/blob/main/sleep_staging_physionet.ipynb#0.-Setting-up-the-environment.

[15] Chambon, S., Galtier, M. N., Arnal, P. J., Wainrib, G., & Gramfort, A. (2018). A deep learning architecture for temporal sleep stage classification using multivariate and multimodal time series. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 26(4), 758-769

[16] Yildirim, O., Baloglu, U. B., & Acharya, U. R. (2019). A deep learning model for automated sleep stages classification using PSG signals. *International journal of environmental research and public health*, *16*(4), 599.

[17] https://github.com/swayanshu/Sleep-Stage-Classification