Analyzing the Mental Fatigue Findings via Physiological Signals

1st Ömer Faruk IŞIK
Dept. of Computer Engineering
Istanbul Kultur University
1700003708@stu.iku.edu.tr

2nd Yasemin NUKAN
Dept. of Computer Engineering
Istanbul Kultur University
1600001635@stu.iku.edu.tr

3rd Alper Yusuf DURSUN
Dept. of Computer Engineering
Istanbul Kultur University
1600001752@stu.iku.edu.tr

***Abstract*—Mental fatigue is one of the primary causes of diminished cognitive capacity, situational awareness, and decision-making ability and is one of the leading causes of accidents in daily life. Physiological and, in most cases, temporary mental or cognitive fatigue is perceived as insignificant but fatal in work areas such as aviation, automobiles, and shipyards, especially in vehicle use may lead to severe accidents. Due to this reason, it is crucial to analyze the mental capacity of working individuals in an age where today's technology increasingly requires cognitive awareness. The most accurate way to gauge mental weariness is via physiological sensors. However, it is a sophisticated task because mental fatigue manifests itself in different ways in different people. As a case study, this article presents the usage of various deep-structured learning algorithms and physiological sensors to identify mental fatigue at work. The results substantiate that deep learning algorithms like CNN and LSTM may reach high classification accuracy levels for mental fatigue because they can extract difficult-to-read and unmeaningful raw data. The findings express that rather than traditional neural networks(NNs), more complex NNs and further deep-structured learning algorithms containing LSTM or CNN algorithms are more precise with mental fatigue classification.**

***Index Terms*—Electroencephalograph (EEG), Signal processing, Deep learning, Mental fatigue**

I. Introduction

Mental fatigue often manifested as cognitive impairment, is one of the main grounds of accidents occurring in everyday life. However, it is known that mental fatigue associated with the performance of cognitive tasks is not caused by perturbation of neural mechanisms.

Physiological and, in most cases, temporary mental or cognitive fatigue is perceived as insignificant but fatal in work areas such as aviation, automobiles, and shipyards, especially in vehicle driving. In addition, it may lead to serious accidents. For this reason alone, it is essential to analyze the mental capacity of working individuals in an age where today's technology increasingly requires cognitive awareness. The most crucial factor is to create a realistic experimental environment, which allows the research that is or will be conducted to contribute in this direction. A key factor for this is the ability to analyze real-time the data used to detect cognitive fatigue from physiological signals.

The main goal of this initiative is to identify through physiological signals whether factors associated with mental fatigue cause decreased physical performance. Another research topic is to assess the effects of physical task duration and intensity to provide an overview of the potential factors underlying this effect.

The Fatigue Assessment Scale that we will use in our project will be used to determine whether the subjects are fatigued or not [1]. People will measure whether they are fatigued with this method. This article will explain the deep learning methods on the subject's physiological signals such as an electroencephalogram, accelerometer, blood volume pulse, electrodermal activity, and temperature.

We may encounter some problems when physiological sensors detect mental fatigue. The data obtained from these sensors are often complex, and they do not agree with each other because they measure based on frequency [2]. Also, we cannot say there is a precise fatigue scale as these data may differ from person to person. Some deep learning methods can provide high accuracy and efficiency and reduce the requirement for more data.

In this research, we search for the best deep learning method that fits the mental fatigue data. The rest of the research is as follows: Section II describes the data collected from the subjects and how this data will be processed. Then, in section III, The results are determined and explained. Finally, IV evaluates the results and advises on future work.

II. Methodology

To assess the effectiveness of several deep-learning techniques for classifying mental weariness, the experiment was conducted while the subjects were at work. This section describes our experimental work, the sensors that we used, and an overview of the Data set.

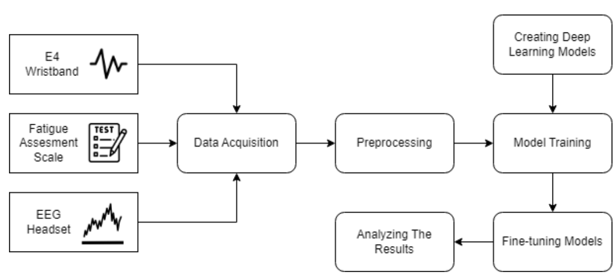

Fig. 1. Main steps of the entire process

A. Experimental Work

This experiment was performed with two wearable devices provided by Istanbul Kultur University. These devices were used to monitor and collect raw physiological signal data from 30 different subjects. Subjects are working in 3 different fields of proficiency, and they have a similar number of working hours, which is around 8-9 hours. A survey was filled out by each subject. This survey includes information on each subject, such as their gender, age, and working hours. Wearable devices were used to gather data from subjects for 30 minutes, both before the start of their shift and after their shift. Both devices collected data simultaneously since the software applications are timed to start collecting data at the same time.

TABLE I
FATIGUE ASSESSMENT SCORE INTERPRETATION

Total Score	Levels of Fatigue
Less than 22	Normal
Between 22 and 34	Mild to moderate
35 or more	Severe

At the end of each 30 minutes, FAS was used to determine if the subject was fatigued or not [1]. A description of the fatigue experienced is represented in Table I.

B. Wearable Sensor Description

One of the devices we use in our project is the Empatica E4. All subjects wore wristband on their left hand. The Empatica E4 wristband contains four sensors. Electrode for Electrodermal activity (EDA), a 3-axis accelerometer (ACC), a temperature (TEMP), and a photoplethysmography for blood volume pulse (BVP) are specified in Table II [3].

TABLE II
DATA SAMPLING OF EMPATICA E4

Data	Sampling Frequency
ACC	32Hz
BVP	64Hz
EDA	4Hz
TEMP	4Hz

The other device is Neurosky Mindwave Mobile 2. Electroencephalography EEG is a powerful device for recording the brain's electrical activity. Electroencephalography is non-invasive because the brain's electrical activity is recorded from the scalp surface after being picked up by electrodes. Patients are hardly at any risk from it, normal adults and children. It can be used repeatedly without limits. The brainwaves pattern captured by EEG has been categorized into four groups, as shown in Table III [4]:

We had to find an app to download the device's data because the kit that measures brain signals does not record the data. So we used the application called NeuroExperimenter, where we can save the EEG data in CSV files [5].

TABLE III
EEG BRAINWAVE SIGNALS

Signals	Frequency
Alpha	8-13 Hz
Beta	Above 13 Hz
Theta	4-8 Hz
Delta	0.5-4 Hz

C. Data Acquisition and Preprocessing

As described in section II-B, an Empatica E4 wristband and MindWave Mobile headset were used to collect data from subjects. Both devices have several sensors, and these sensors monitor and collect different kinds of signals. The wristband collects raw data and exports them into different CSV files for each sensor because every sensor has a different sampling rate. Because these sensors have different sampling rates, every sensor output has a distinct amount of data at the end of the collection of the data.

Some data preprocessing operations were necessary to have a meaningful data set to train different kinds of deep structured learning algorithms and models. As the first part of the preprocessing, raw data files were analyzed, and the sampling rates were examined.

Every physiological signal sensor has a different sampling rate (Hz), so each sensor represents data in a different time interval. A Python script has been created to express all the sensor data appropriately to fix this problem and have a meaningful data set.

The first step of the script was to cut out the first and last 5 minutes of the raw data since there might have been some connection issues or noise in the data. Starting from the 300th second to the last 5 minutes, the data have been cropped out once for every 15 seconds. This procedure has been done for every raw sensor data for each subject and saved into new data frames. Finally, these new data frames were exported to CSV files, titled with their sensor names, respectively.

After the first script, meaningful data for every sensor was generated, but the data was still not a whole; instead, they were separated by each subject number.

A secondary script was created in Python to combine every subject's CSV files categorized by sensor type. For example, the entire BVP data for each subject has been gathered into a single CSV file. The same procedure has been done for each sensor type, such as EDA, ACC, and EEG signals generated by headset SDK.

After combining every single sensor data in CSV files, the data at hand were analyzed to gather the entire data in a single CSV file so that different kinds of deep learning algorithms could be trained with a single data frame. After the analysis, it was discovered that even though all the sensors operated for the same amount of time (30 minutes each), they generated different amounts of data. To overcome this problem and have a data set that represents the entirety of the data appropriately, all the data except the BVP data has been oversampled by their sampling rate (Hz) with the help of the oversampling libraries in Python so that the final data frame would be a meaningful and appropriate data set.

Another crucial step of data preprocessing is standardization. Standardization is a widely used scaling technique that makes data scale-free by transforming the statistical distribution of the data into the following form:

$$z = \frac{x - \mu}{\sigma}$$

Fig. 2. Mathematical Representation of Standardization

- mean - 0 (zero)
- standard deviation - 1

This globally scales the entire dataset with zero mean and unit variance.

With the help of StandardScaler function from sklearn preprocessing library, this process can easily be done.

D. Deep Learning Models

Deep learning (also called deep structured learning) is a branch of the larger ML technique tree derived from ANNs through representation learning. Deep learning models represent a new learning paradigm in various fields, such as machine learning and artificial intelligence (AI). On the downside, the mathematical and computational methodologies that form the basis of deep learning models can be complicated and confusing, especially for scientific researchers working with big data. To solve this problem, this section provides detailed information about various deep learning models.

1) Deep Neural Network (DNN) : In the simplest case, a NN that contains some complexity, most of the time containing at least two layers, can be called as DNN, also known as the deep net. Deep nets deal with data in complex ways by deploying complicated mathematical modeling.

First of all, machine learning techniques got developed so that deep learning techniques would evolve by them. Machine learning (ML) is a structure to automate statistical models with the help of different algorithms. An ML model is a specific model that predicts some data variable. Then, those predictions are made, and a result is generated as accuracy.

The training part of generating deep learning algorithms and models initiated the development phase of ANNs. The hidden layer is used by ANNs as a location to reserve and assess the importance of some of the inputs to the output. The hidden layer also establishes correlations between the value of groupings of inputs and holds information about the significance of each input.

The fact that this works well for model correction means that every element in one of the hidden layers brings up a relationship and evaluates the importance of the entry data in creating the output data. So who says we can't just heap

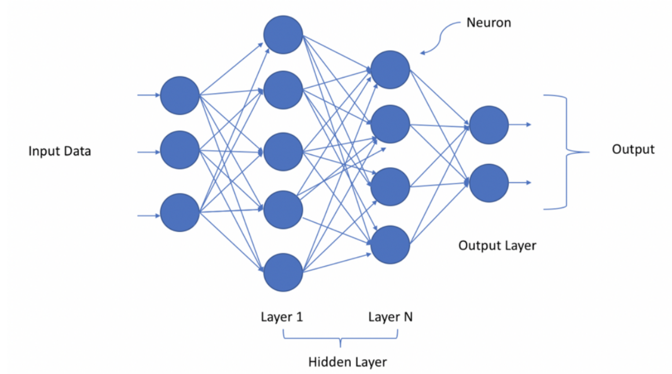

Fig. 3. Deep Neural Network Architecture

these and make them more useful? The deep web, therefore, contains several hidden layers. The word "deep" actually refers to multiple model layers.

2) Recurrent Neural Network (RNN) : An RNN is an NN that includes loops that allow data to be cycled within the network. In other words, recurrent neural networks use assumptions from past experiences to signal upcoming events. Repetitive models are more capable of sequencing vectors and opening APIs to perform more complex tasks [6].

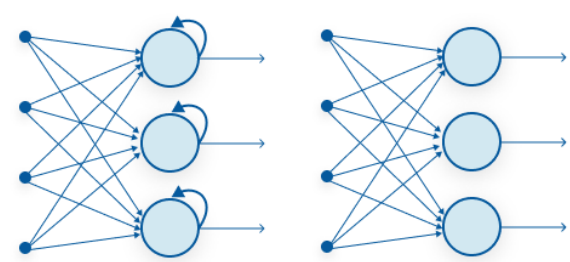

Fig. 4. Difference Between RNN and a Feed-Forward Neural Network

One way to conceptualize an RNN is as a series of interconnected networks. They often have a chain-like architecture, making them suitable for tasks such as language translation, and speech recognition. RNNs can be designed to process sequences of vectors at their inputs, outputs, or both [6].

3) Long Short-Term Memory (LSTM) : LSTM is an enhanced version of RNNs that can learn long-term dependencies. It was first presented by Hochreiter Schmidhuber (Hochreiter & Schmidhuber, 1997 [7]), and has been improved and generalized by many people in later research.

A series of repeatedly connected subnets referred to as memory blocks make up an LSTM algorithm, and also known as an upgraded version of RNNs [8]. The LSTM's design objective is to address this intrinsic problem of vanishing gradients. These blocks are recognizable versions of the memory chip on a computer (Hochreiter & Schmidhuber, 1997 [7]).

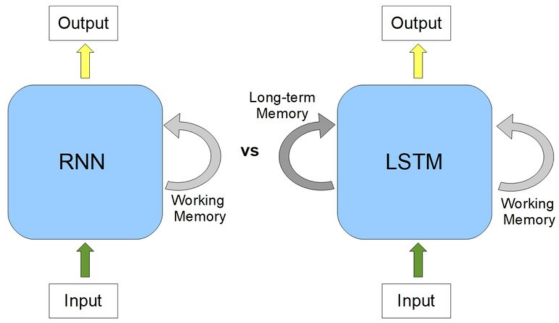

Fig. 5. Difference Between RNN and LSTM

4) Convolutional Neural Network (CNN) : Traditional ANNs and CNNs share some important similarities. They consist of neurons that can refine themselves through experience [9].

From the raw input data to the last class score output, The entirety of the network represents a single perceptual score function (weight). The last layer contains the classes and all the associated loss functions. The usual handy parts mainly generated to help ANNs also valid for CNNs.

Just one notable distinction between this model technique and conventional ANNs can be expressed as that CNNs are mainly employed in the field of classifying pictures or images. By doing so, it will be easier to incorporate image-specific elements into the network's design, which will improve the network's suitability for tasks that are mainly focused on pictures and lower the number of parameters needed to set up a model.

As mentioned earlier, CNNs are primarily focused on the input consisting of images. But this does not conclude that it can only be used on image classification. The emphasis is on configuring the architecture to best match the needs for managing particular sorts of data. One of the primary distinctions is that a CNN's layer of neurons is made up of neurons arranged in three dimensions [9].

III. Result and Discussion

After dealing with raw data and deciding on which deep learning models to use, three different deep learning models have been created. After some brief research, we have decided that a vanilla DNN, a 1-D CNN, and a hybrid LSTM model would be the best options for training our data set.

The next step was to analyze and determine which feature combinations would be more helpful in classifying mental fatigue. According to that work, three combinations were designed so that different deep learning algorithms would give us a proper outcome about the different physiological signals' effects on mental fatigue and cognition.

The first and probably the most important combination was to evaluate EEG signals separately from other signals such as BVP, EDA, etc. One of the earlier studies on the effects of mental weariness on brain efficiency shows that these states have impeccable distinctions between rested and fatigued states and can be useful for anticipating and detecting accidents in a variety of domains. [10].

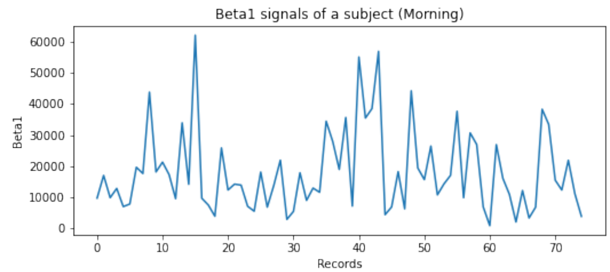

Fig. 6. A Sample of Beta Signal of A Subject in Rested State

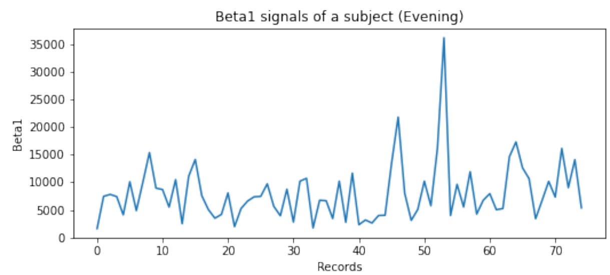

Fig. 7. A Sample of Beta Signal Of a Subject in Fatigued State

Our second feature combination was to train physiological signal data such as , EDA, and more. This combination mainly focuses on physical signals that affect fatigue, such as stress. As depicted in figure 9, high BVP is an outcome of mental fatigue. And this mental fatigue can also cause critical accidents in different fields of proficiency [11]. Being stressed out is a part of our daily life, and may get even higher after a long and exhausting day at work. We created a secondary combination to train our models to analyze this, specifically focusing on more physical effects.

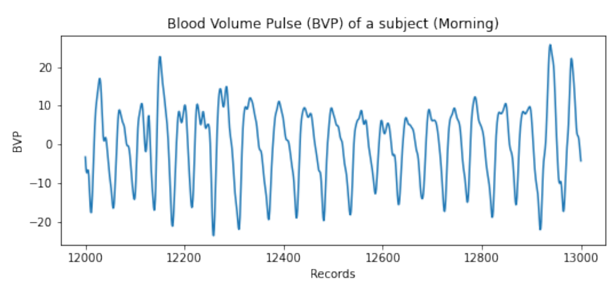

Fig. 8. Blood Volume Pulse in Rested State

We created three deep-learning model architectures, a vanilla DNN, a 1-D CNN, and an LSTM in Python to analyze and train our data. The first step was to convert our categorical data "status", which represents the fatigue state of subjects, into dummy variables, also known as indicator variables.

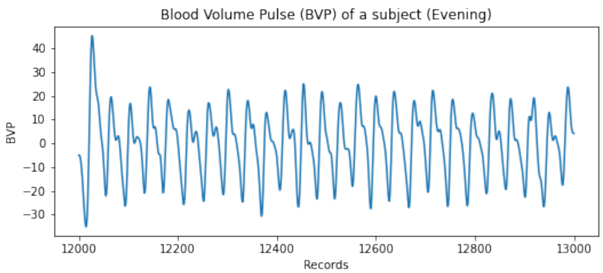

Fig. 9. Blood Volume Pulse in Fatigued State

After this process, data was split into testing and training sets for further classification. The size ratio for the test said is determined as either 15% or 20% (based on which model it may vary).

Until this step, the majority of the operations that we have done are similar for all of our deep-learning models. However, every different model type requires different steps, such as defining the input layer, and the usage of different activation methods. Some models also require additional layers such as Dropout to help us prevent overfitting.

To begin with DNN, after the standard processes, an input layer with an input shape (for EEG models it's 9) was first created. The second layer is a dense layer. A dense layer is strongly linked to its previous layer and serves to change the dimension of the output by performing matrix-vector multiplication. And most of the time, it's also used as an output layer with desired output shape. As the third layer, a Batch Normalization layer was used.

The next step was to add a dropout layer. A Dropout layer implements dropout to the input. In the dropout stage, input units are randomized from 0 to 0 for every step. This will be a little bit helpful in preventing overfitting [12]. Inputs that are not 0 have been increased by 1/(1 - Rate), so the addition of all inputs will still be intact. As the layer of output, another dense layer is used. The unit numbers of dense layers are either 64 or 128 depending on the position of it.

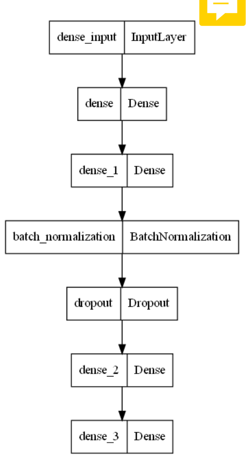

Fig. 10. DNN Model Architecture

At this point, the vanilla DNN model was almost ready to be trained with our EEG data. The last step was to determine the best epoch size and batch size.

The batch number is a parameter that we can imagine as a cycle that repeats more than one time and generates predictions. From this error, we use an update algorithm to advance the model.

A hyperparameter that controls how frequently the learning algorithm iterates over the training data set is the number of epochs [13].

There is not a pre-determined excellent value for batch and epoch size. Different values have to be tested out to figure out a fit value for the data at hand. For our work, the best value we could come up with was 32 for batch size, and 110 for epoch size. After this, we compiled our model.

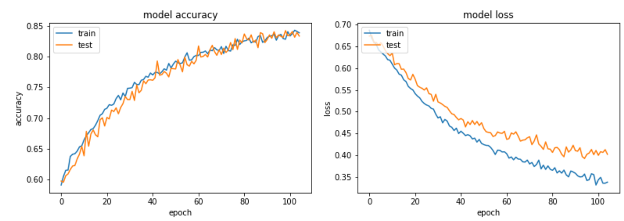

Fig. 11. DNN model for EEG Data

As a result, our model achieved 84% accuracy on the training set. On the other hand, test set accuracy was 80%. The training and test set's loss values were around 0.33 and 0.40, respectively. So this sort of expresses that the test train split percentage was not enough for the test set, but at the same time, increasing the size of the test set would result in overfitting.

Three fundamental layers are commonly present in our second model, CNN: convolutional, pooling and finally a fully connected layer.

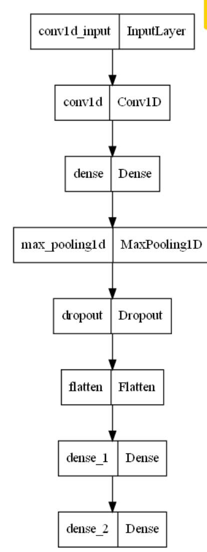

Fig. 12. CNN Model Architecture

A pooling layer modifies output at a particular location by reproducing outline statistics for nearby outputs. This reduces the representation's spatial size and the required computation and weights. Pooling operations are processed separately for each slice of the representation.

After all the similar steps in all kinds of deep learning models, instead of a 2-D or 3-D layer like image classification models require, A 1-D convolutional layer was set as the input layer because we have numerical data. Our parameters for this layer were filter size determined at 32 and kernel size 2. Filter size is an integer that denotes the output space's dimensionality, such as the quantity of output filters used in the convolution. Conv1D window is represented by the integer or variety of single integers known as the kernel size [9].

The second step was to generate a dense layer with size 64, followed by a 1-D pooling layer. Since CNNs are mostly used in image classification, our model needed extra layers like dropout and flattening layers. Before our output layer, another dense layer was generated. At our output layer, an activation function called sigmoid was used to have better accuracy and loss values at the end.

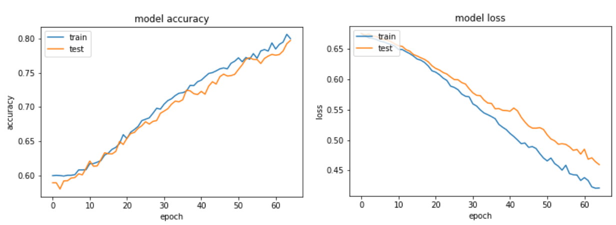

Fig. 13. CNN model for EEG Data

As a result, our CNN model achieved 80% accuracy on the training set. However, test set accuracy was 78%. The training and test set's loss values were around 0.42 and 0.45, respectively. The performance is closer to our vanilla DNN model, but DNN performed a little bit better in classifying our data.

The first step of creating the model was to initiate with our input layer according to the size of our EEG data. The second step is to create a dense layer with size 64 with the activation function called ReLU.

ReLU is a componential linear function that transforms an input strictly to the upcoming output, with the condition of input being positive. Models that use it are easy to train and often perform better, making it the most common activation function used on many types of neural networks [14].

The next layer includes a bidirectional LSTM layer. This layer brings up extra interactions with the input, during training it might enhance gradient flow over lengthy periods. This layer was used two times to have better results at the end. A dropout layer was added between two LSTM layers, and also after the second LSTM layer. As the output layer, a dense layer was created with the sigmoid function. Also, the most appropriate batch size was 128, and the best value for the epoch was 25.

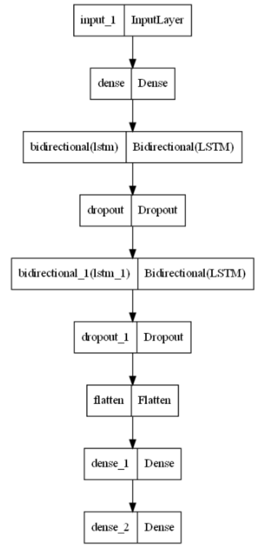

Fig. 14. LSTM Model Architecture

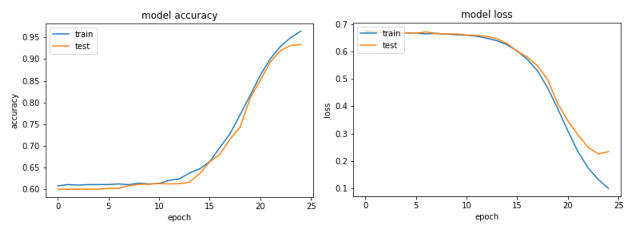

Fig. 15. LSTM model for EEG Data

As a result, our model achieved 96% training accuracy and testing accuracy was 93%. The training and test set's loss values were around 0.10 and 0.23, respectively. So this sort of expresses that with our data, the best choice was a bidirectional hybrid LSTM model. This is expectable since our data is numerical, the LSTM architecture has a loop of subnet blocks working on the input data continuously until the circle size defined by the user is ended. So in a way, it's easier for LSTM to learn numerical data better.

Next step was to create deep-learning models for Empatica data, such as temperature, BVP, and EDA. We used the same approach to our models as we did with EEG data. The same 3 model architectures were used. A vanilla DNN, a 1-D CNN and a hybrid LSTM model was used with the same parameters. Some slight differences have been applied to the parameters like batch size and epoch number. Since we had a bigger data set for our Empatica E4-only data analysis, we did not need so many layers and a bigger train test split ratio.

As seen in figure 16, our DNN model achieved 89% training accuracy and test accuracy was 87%. Training loss is 0.24 and validation loss is 0.27. Until now, we can say that this model has worked out well.

For our CNN model that would train with Empatica data, the same process was followed as we did with EEG data. Since our data is bigger now, we dropped the secondary max pooling

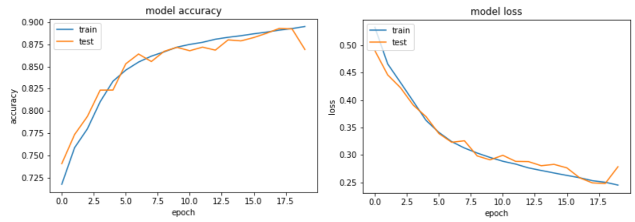

Fig. 16. DNN model for Empatica Data

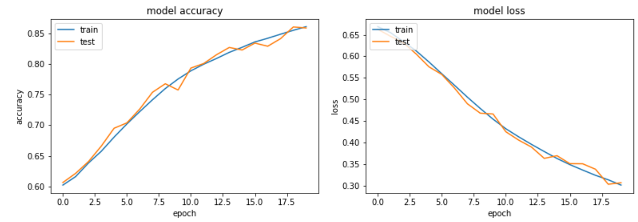

Fig. 18. CNN model for EEG & Empatica Data

and dropout layers to have a more accurate model. Sigmoid was selected as the activation parameter. And obviously, the input shape was changed. The batch size was increased to 128, and the epoch number was decreased to 20. And for the last part of our modifications, the filter size in the 1-D input layer has been decreased to 16.

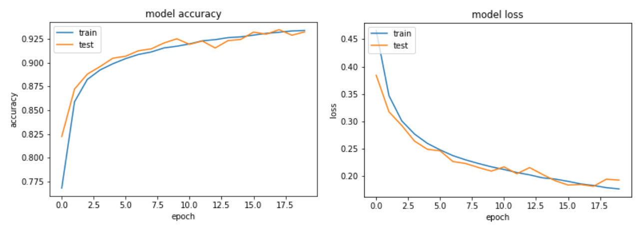

Fig. 17. CNN model for Empatica Data

As depicted in figure 17, our CNN model achieved 93% training accuracy and test accuracy was 92%. The training and test set loss values were calculated as 17% and 19%, respectively. Compared, it performed better than our vanilla DNN model in classifying fatigue.

When it comes to testing out the LSTM model, we found out that LSTM was not useful for our Empatica data. We have applied several improvements to the model to prevent and overcome overfitting. But we analyzed that the model kept overfitting at the early epochs, which was unavoidable. Test accuracy was mostly higher than training accuracy, and it was the same deal for loss value. So in the end we concluded that LSTM was not useful for the Empatica data. This was probably caused by a lack of features in our Empatica data.

The third combination was to try out a data set that contained both EEG and Empatica data, but there were some problems. None of the models we tried produced good results because both feature sides (EEG and Empatica) are irrelevant to each other.

Our DNN model achieved 84% training accuracy and 85% test accuracy. The training and test set loss values were calculated at 33% and 32%, respectively. We performed various optimizations to the model, but the results were always as we described. Test results were always better than training results. The CNN model was a little bit better compared to DNN; training accuracy was 86%, and test accuracy was 85%. Loss values were calculated as 30% and 31%, respectively.

The results were way worse compared to the other two models when it came to LSTM. Whatever the size of the test set or whatever optimizations were done, the results were always better on the test set. This was probably caused by a data set that contained irrelevant features. The numbers were not realistic, so they are not demonstrated in this work.

TABLE IV
RESULTS OF VARIOUS DEEP LEARNING MODELS

	Input	Train	Test
DNN	EEG	0.84 ± 0.33	0.80 ± 0.40
	EMP	0.89 ± 0.24	0.87 ± 0.27
	EEG+EMP	0.84 ± 0.33	0.85 ± 0.32
CNN	EEG	0.80 ± 0.42	0.78 ± 0.45
	EMP	0.93 ± 0.17	0.91 ± 0.19
	EEG+EMP	0.86 ± 0.30	0.85 ± 0.31
LSTM	EEG	0.96 ± 0.10	0.93 ± 0.23
	EMP	0.88 ± 0.20	0.92 ± 0.19
	EEG+EMP	-	-

IV. CONCLUSION

This research determined working people's mental fatigue using three basic deep learning methods. We observed that we need to make some improvements so that the devices from which we receive the data match each other in Hz. Using all of the data from the Empatica device turned out badly, so we focused only on the EDA, TEMP, and BVP data. We also removed some features from the EEG data that did not affect learning, so our results were better.

First, LSTM yielded the best classification accuracy on EEG data. DNN performed better than CNN. We realized that CNN is not a good algorithm for EEG data. Second, CNN yielded the best training accuracy and loss in Empatica data. Also, test accuracy and loss are good as well. DNN yielded much better accuracy compared to LSTM. We observed that LSTM provides high test accuracy over training accuracy on a low-specification dataset, so it is unsuitable for low-specification datasets.

When both EEG and Empatica data were combined, the results were not satisfying. This was most probably caused by the irrelevance of features on both sides. We found out that

LSTM did not perform properly on the combined data. CNN was a good choice, but it also had problems. The reasonable choice was to train EEG and Empatica data separately, not as a whole.

In future works, improvements in the performance of classification algorithms can be made by optimizing the hyperparameters in more complicated ways. Another upgrade that can be done is creating hybrid model architectures such as CNN-RF or other combinations of different deep learning algorithms. Subjects can be classified by their gender or age in future work so that the classification of mental fatigue based on a person's information can be done in a more accurate way.

References

[1] de Vries, J., Michielsen, H., Van Heck, G.L. and Drent, M. (2004), Measuring fatigue in sarcoidosis: The Fatigue Assessment Scale (FAS). British Journal of Health Psychology, 9: 279-291. https://doi.org/10.1348/1359107041557048

[2] M. Langkvist, L. Karlsson, and A. Loutfi, “A review of unsupervised ¨ feature learning and deep learning for time-series modeling,” Pattern Recognition Letters, vol. 42, pp. 11–24, 2014

[3] Empatica. E4 wristband User's manual. Retrieved 12.27.20222 from https://empatica.app.box.com/v/E4-User-Manual

[4] M. Teplan, ”Fundamentals of EEG measurement,” Measurement science review, vol. 2, pp. 1-11, 2002.

[5] NeuroExperimenter Users Guide. Retrieved 27.12.2022 from https://drive.google.com/file/d/1QliMc7bU7AlpkNKdrRPvdcR9PwWwV7jD

[6] Recurrent Neural Networks: Design and Applications (International Series on Computational Intelligence) 1st Edition, by Larry Medsker (Editor), Lakhmi C. Jain (Editor)

[7] Hochreiter S, Schmidhuber J. 1997. Long short-term memory. Neural Comput. 9(8):1735–178

[8] Dyer C, Ballesteros M, Ling W, Matthews A, Smith NA. 2015. Transition-based dependency parsing with stack long short-term memory. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics; p. 334–343

[9] Nash, R. (2015). An Introduction to Convolutional Neural Networks. arXiv. https://doi.org/10.48550/arXiv.1511.08458

[10] Li, G., Huang, S., Xu, W. et al. The impact of mental fatigue on brain activity: a comparative study both in resting state and task state using EEG. BMC Neurosci 21, 20 (2020). https://doi.org/10.1186/s12868-020-00569-1

[11] Posada-Quintero HF, Bolkhovsky JB. Machine Learning models for the Identification of Cognitive Tasks using Autonomic Reactions from Heart Rate Variability and Electrodermal Activity. Behavioral Sciences. 2019; 9(4):45. https://doi.org/10.3390/bs9040045

[12] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), 1929-1958.

[13] Brownlee, J. (2018). What is the Difference Between a Batch and an Epoch in a Neural Network. Machine Learning Mastery, 20.

[14] J. Si, S. L. Harris and E. Yfantis, ”A Dynamic ReLU on Neural Network,” 2018 IEEE 13th Dallas Circuits and Systems Conference (DCAS), 2018, pp. 1-6, doi: 10.1109/DCAS.2018.8620116.