Emotion (Valence and Arousal) Prediction from DEAP Dataset

Mr. Jirasak Buranathawornsom
Mr. Arnajak Tungchoksongchai
Mr. Sunny Kumar Tuladhar
Mr. Amanda Raj Shrestha

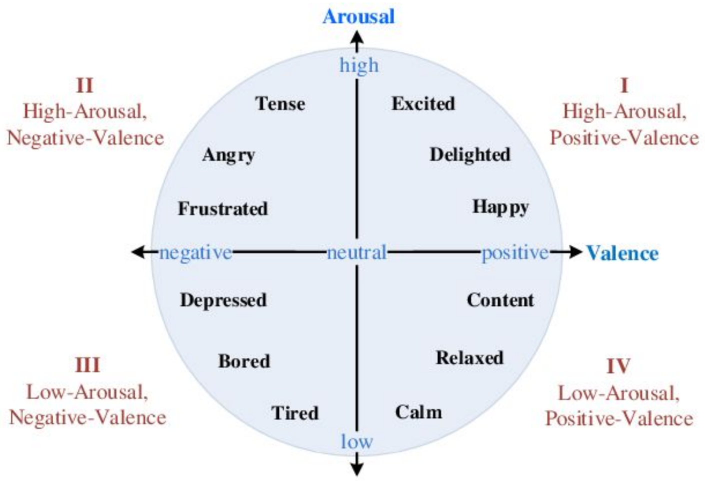
Objective

Try to predict the Emotion from DEAP dataset using various features and machine learning methods.

About the DEAP Dataset

A database for Emotion Analysis using Physiological Signals.

Main Goal of the dataset to create an adaptive music video recommendation system.
Uses **Valence-Arousal** Scale by **Russell** for emotion categorisation



UI filled by participants

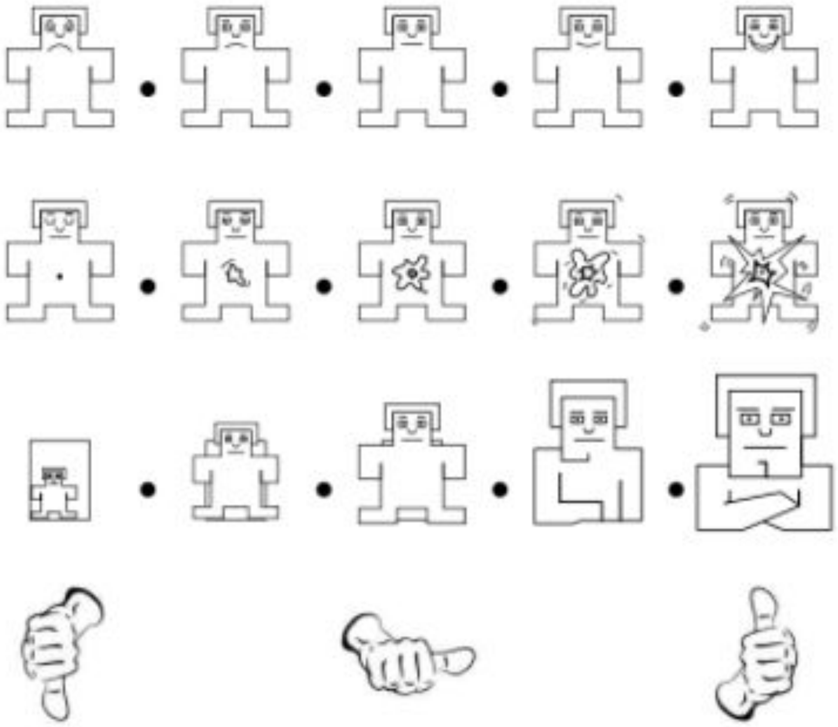
Major predictors of emotion

VALENCE

AROUSAL

DOMINANCE

LIKING

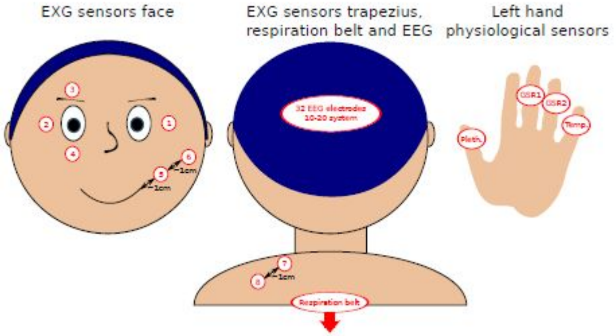


About the dataset

32 participants. 32 EEG electrodes (10-20 system) sampled originally at 5**12 Hz** and **8 peripheral physiological signals** recorded.

40 videos(stimuli) of **1 minute** each shown to each participant with a break after 20 mins.

Videos were music videos selected on basis of **LALV/ LAHV /HALV /HAHV**



Placement of peripheral physiological sensors.
For Electrodes were used to record EOG and 4 for EMG
(zygomaticus major and trapezius muscles). In addition,
GSR, blood volume pressure (BVP), temperature and
respiration were measured.

About the dataset (Preprocesssed)

Array name	Array shape	Array contents
data	40 x 40 x 8064	video/trial x channel x data
labels	40 x 4	video/trial x label (valence, arousal, dominance, liking)

1. The data was downsampled to 128Hz from 512 tto 256 and then to 128.
2. EOG artefacts were removed. ***electrooculogram*** *(EOG) represents the eyeblinking signals*
3. A bandpass frequency filter from 4.0-45.0Hz was applied.
4. The data was averaged to the common reference.
5. The EEG channels were reordered so that they all follow the Geneva order as above.
6. The data was segmented into 60 second trials and a 3 second pre-trial baseline removed. 63 x 128 = 8064
7. The trials were reordered from presentation order to video (Experiment_id) order.

Sunny Kumar Tuladhar

Experiments

1. Power Spectral Density (PSD) + Machine Learning(SVM, NVB, KNN, XGBoost)
2. Power Spectral Density (PSD) + ANN
3. Short Term Fourier Transform + 3dCNN

PSD using Welch's Periodogram

Power Spectral Density was extracted from the signal with a window of **4 seconds.** Our signal is sampled at **128 Hz** and has a **band pass** at **4.0-45.0Hz .**

The original array is 32 subjects x 40 clips x 40 channels x 8064 channels.

Converted to first to 32sub x 40 clips x 32 *eegchannels* x 7680 signals

(first 3 seconds are baseline so 3 x 128 were removed)

The brainwave frequencies

Theta brainwaves (4-7 Hz) represent a day dreamy, spacey state of mind mental inefficiency. At very slow levels, theta brain wave activity is a very relaxed state, representing the twilight zone between waking and sleep.

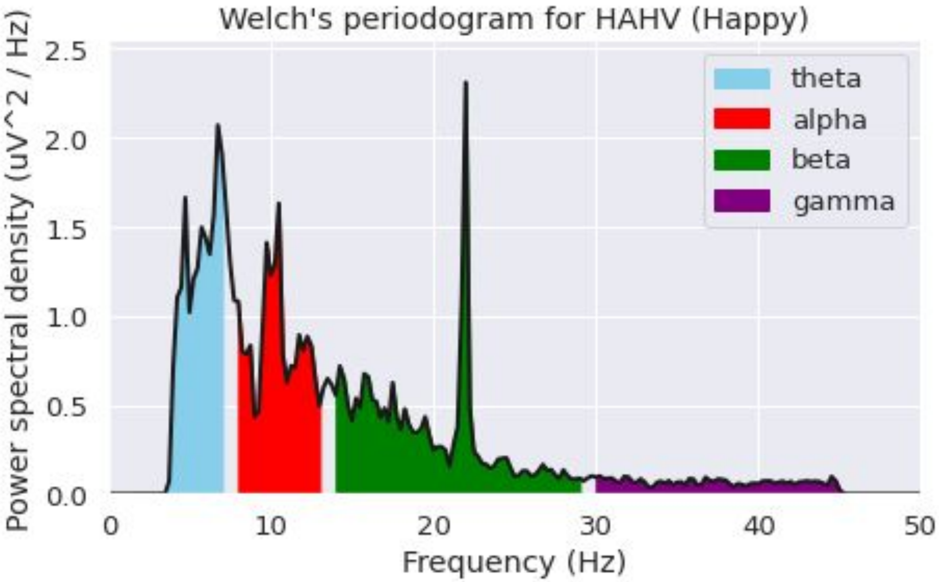
Alpha brainwaves (8-12 Hz.) are slower and larger. Associated with a state of relaxation and represent the brain shifting into an idling gear, waiting to respond when needed. If we close our eyes and begin picturing something peaceful, there is an increase in alpha brainwave

Beta brainwaves (13 – 38 Hz) are small, faster brainwaves associated with a state of mental, intellectual activity and outwardly focused concentration. This is basically state of alertness.

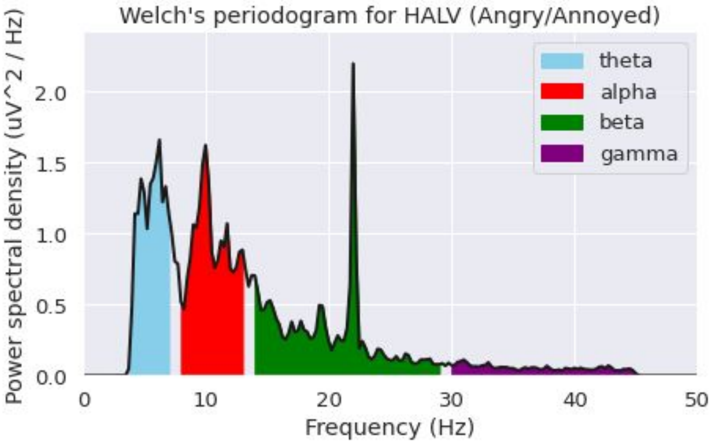
Gamma brainwaves (39 – 42 Hz) are the fastest and most subtle brain waves. Gamma rhythms modulate perception and consciousness.

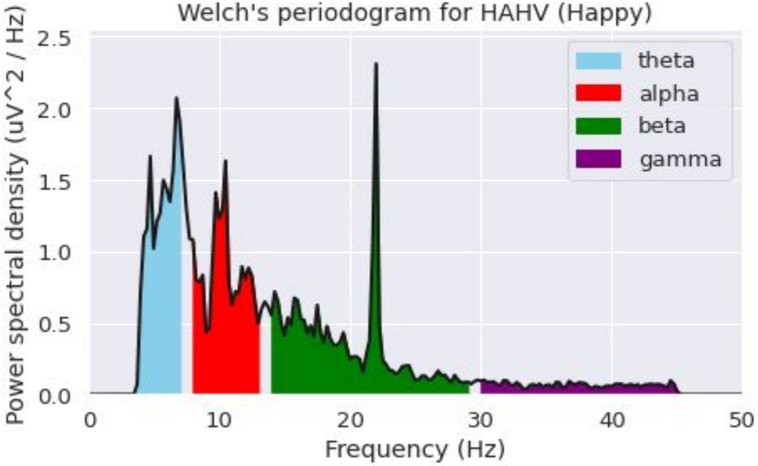
Power Spectral Density

Each channel was converted to the band power of each of these frequencies. A window of 4 seconds was chosen for the PSD extraction.

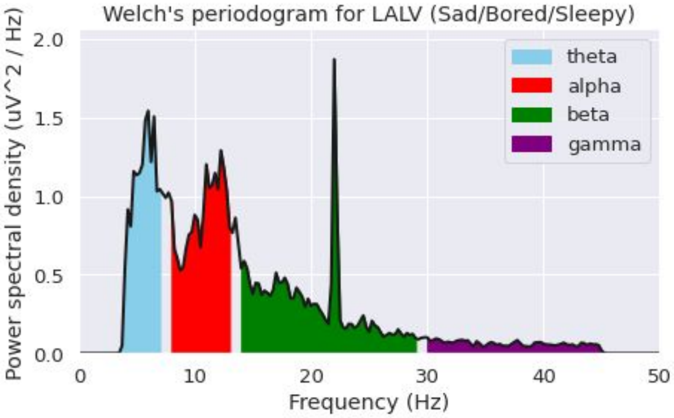


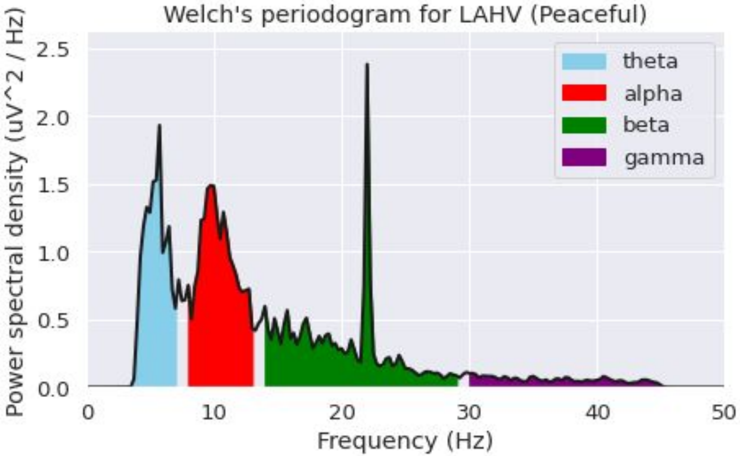
Welch’s periodogram for the 1st channel of the first person’s 1st video.





Periodogram for the 1st subject in the 1st channel of four clips that caused four emotions





PSD + Machine Learning

Data was split into 25 Train subjects and 7 test subjects.

Then 4 freq power bands were extracted from the signal.

First **each isolated power band** from the **32 channels** was taken as the feature separately and fed into ML algorithms(SVM, NVB, KNN, XGBoost)

Experiment/ Accuracy	PSD + NVB			
	Theta	Alpha	Beta	Gamma
Valence	0.38	0.37	0.53	0.6
Arousal	0.53	0.54	0.54	0.54

Experiment/ Accuracy	PSD + SVC (rbf)			
	Theta	Alpha	Beta	Gamma
Valence	0.62	0.62	0.62	0.62
Arousal	0.54	0.54	0.54	0.54

Experiment/ Accuracy	PSD + KNN			
	Theta	Alpha	Beta	Gamma
Valence	0.57	0.55	0.48	0.46
Arousal	0.54	0.56	0.57	0.58

Experiment/ Accuracy	PSD + XgboostClassifier			
	Theta	Alpha	Beta	Gamma
Valence	0.47	0.51	0.43	0.49
Arousal	0.53	0.48	0.48	0.53

PSD all bands + ML

Later all **4 bands** of each of the **32 channel (32 x 4 = 128 features)** was fed into the ML algorithms.

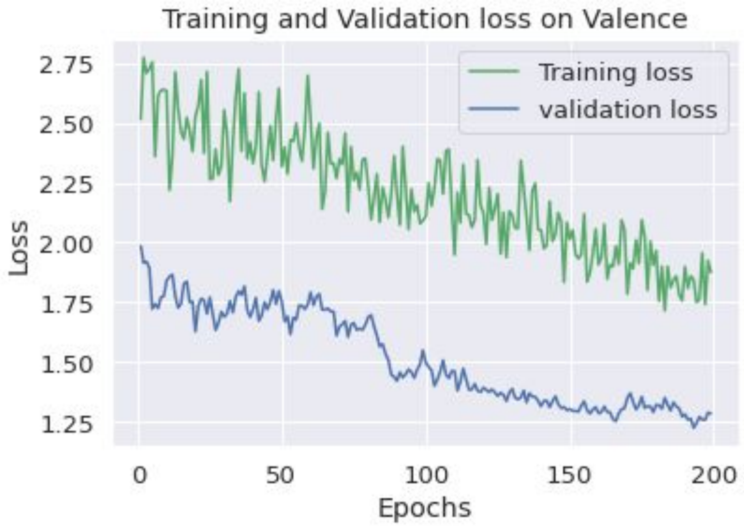
Experiment/ Accuracy	PSD (All Bands) + ML			
	NVB	SVC-rbf	KNN	XGBoost
Valence	0.53	0.62	0.55 (n = 3)	0.6
Arousal	0.54	0.54	0.57 (n = 18)	0.55

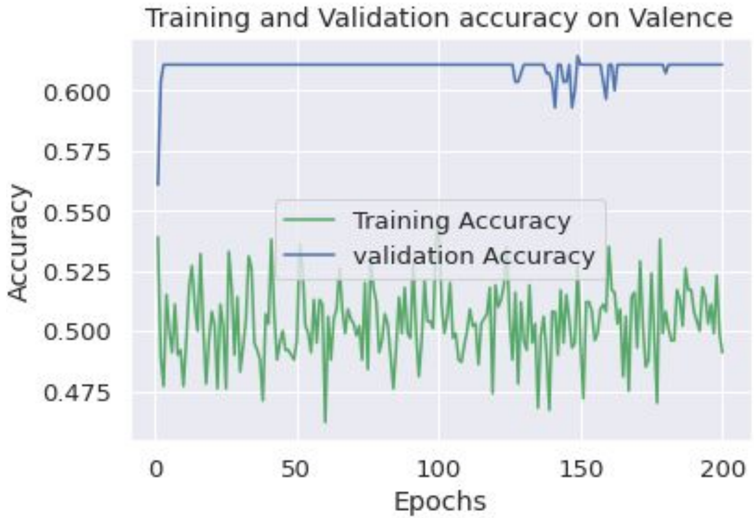
PSD + ANN

Next the PSD with 128 features was run through this Neural Network

```
model = Sequential()
model.add(Dense(256,kernel_initializer='he_uniform',activation = 'elu'))
model.add(BatchNormalization())
model.add(Dropout(.8))
model.add(Dense(64,kernel_initializer='he_uniform',activation = 'elu'))
model.add(BatchNormalization())
model.add(Dropout(.8))
model.add(Dense(16,kernel_initializer='he_uniform',activation = 'elu'))
model.add(Dropout(.8))
model.add(Dense(2)) #data of 2 types
model.add(Activation('softmax'))
model.compile(loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              optimizer=tf.keras.optimizers.Adam(learning_rate=0.00001),metrics=['accuracy'])
```

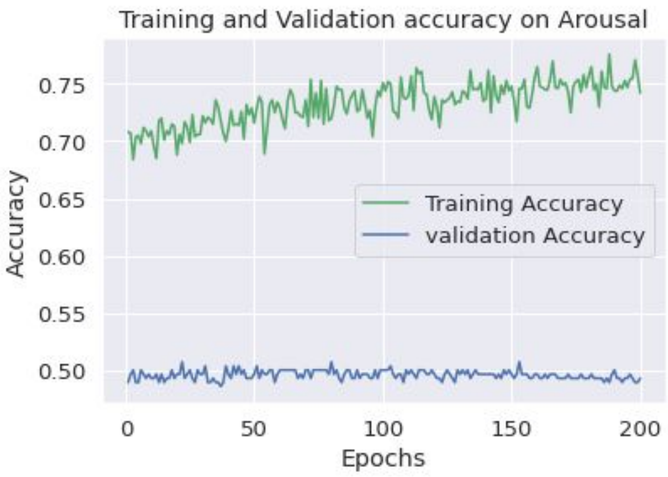
PSD + ANN on Valence

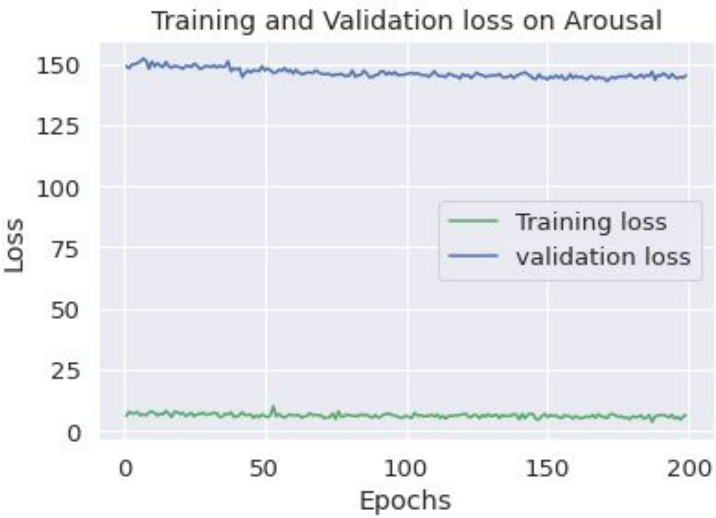




Final validation accuracy on Valence was 0.61 after 200 epochs

PSD + ANN on Arousal





Final validation accuracy on Arousal was 0.49 after 500 epochs

PSD Conclusion

From the above results we have see that PSD of band features combined with Machine Learning does not perform well in predicting Valence and Arousal with **single** or **all** of the frequency bands.

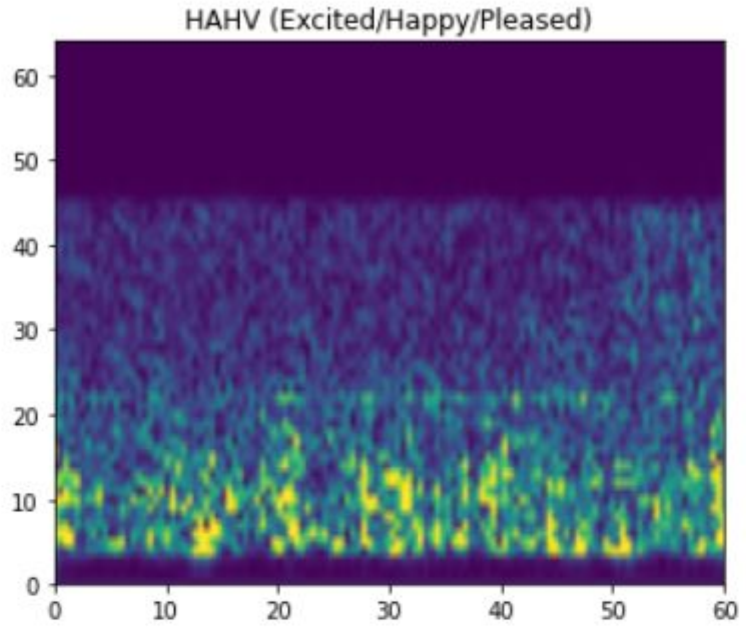
KNN seems to be performing the best among them with 0.56 and 0.63 accuracy.

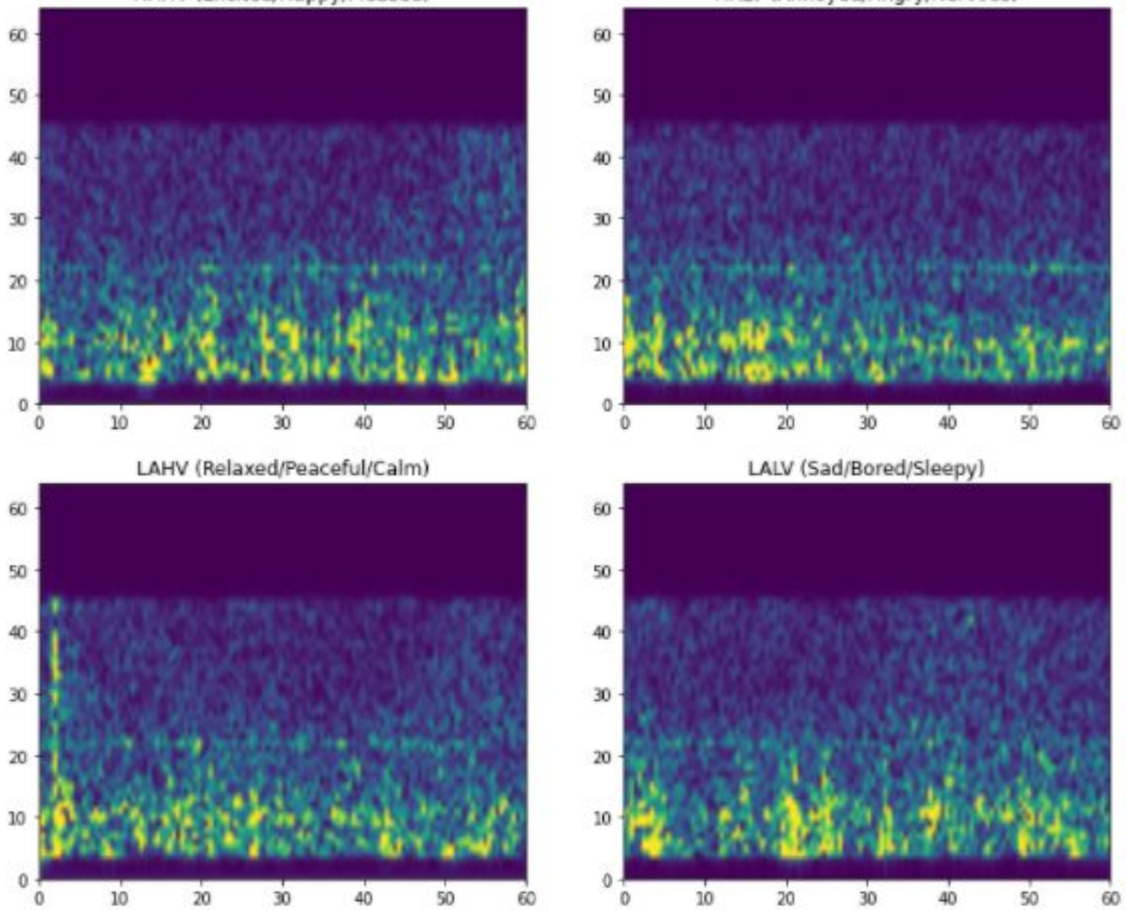
It could be due to lack of temporal data in the extracted features and that emotion is temporal. Possibility.

So next we try STFT with 3dCNN

Short Term Fourier Transform

Next STFT was performed on all 32 EEG channels with a window of 4 secs, 512 samples as recommended by this.

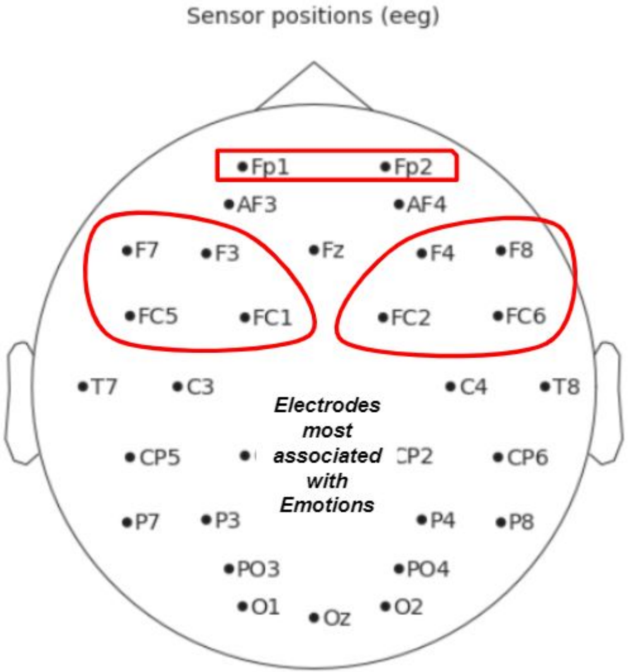




Spectrogram visualisation of 4 emotions of same subjects when watching 4 different clips of channel 5.

10 electrodes for Emotion

Next 10 electrodes were selected based on this [paper] which were most associated with emotion

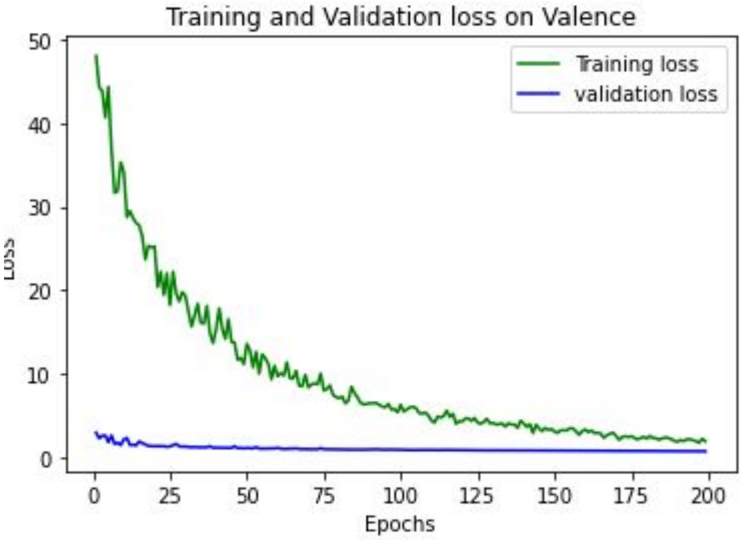


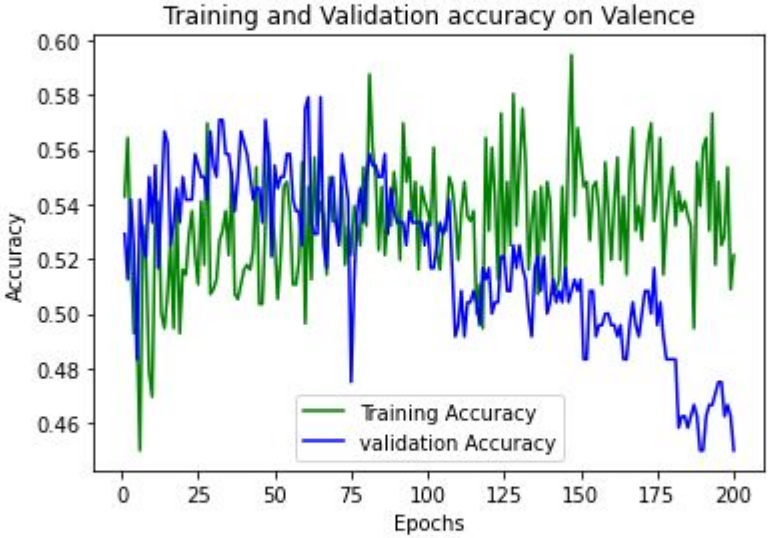
STFT + 3dCNN

```
model = Sequential()
model.add(Conv3D(32, (3,3,3), input_shape=(10, 65, 120,1),)) #input_shape matches our input i
model.add(Conv3D(32, (3,3,3), padding = 'same'))
model.add(Dropout(.4))
model.add(MaxPooling3D(pool_size=(2,2,2)))
model.add(Activation('relu'))
model.add(Dropout(.4))

model.add(Conv3D(64, (3,3,3),padding = 'same'))
model.add(Conv3D(64, (3,3,3)))
model.add(Activation('relu'))
model.add(Dropout(.4))
model.add(MaxPooling3D(pool_size=(2,2,2)))

model.add(Flatten())
model.add(Dense(128))
model.add(Dropout(.7))
model.add(Dense(64))
model.add(Dropout(.7))

model.add(Dense(2)) #data of 2 types
model.add(Activation('softmax'))
model.compile(loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              optimizer=tf.keras.optimizers.Adam(learning_rate=0.00001),metrics=['accuracy'])
```

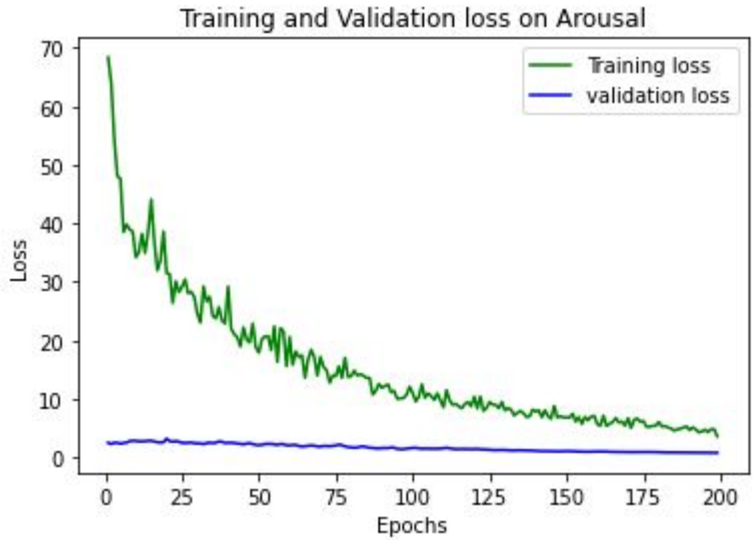
Next we feed the spectrogram image 65 x 120 into a 3DCNN. Our input shape is 32chan x 65 x 120 x 1.

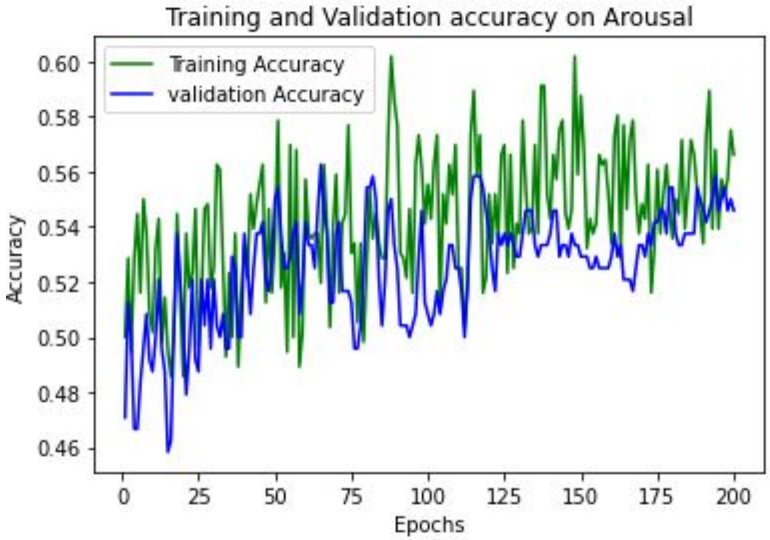
STFT + 3dCNN (Valence)





STFT + 3dCNN (Arousal)





STFT+3dCNN Conclusion

It seems that even with STFT and temporal data the model is not performing well. More research, other methods of feature extraction and different models could be tried to get better results

Also combining the Valence and Arousal into one emotion label with HAHV, HALV, LAHV and LALV as the 4 classes in the label could also be tried out.

References

Emotion recognition from multichannel EEG signals using K-nearest neighbor classification

DEAP: A Database for Emotion Analysis using Physiological Signals

deap-eeg-classification

Amanda Shrestha

- (PDF) A Comparative Study of Subject-Dependent and Subject-Independent Strategies for EEG-Based Emotion Recognition using LSTM Network (researchgate.net)
- Title : A Comparative Study of Subject-Dependent and Subject-Independent Strategies for EEG-Based Emotion Recognition using LSTM Network

Followed Paper

Subject dependent analysis for valence and arousal using different models (KNN, SVM, LSTM)

Papers Proposed LSTM model and accuracies

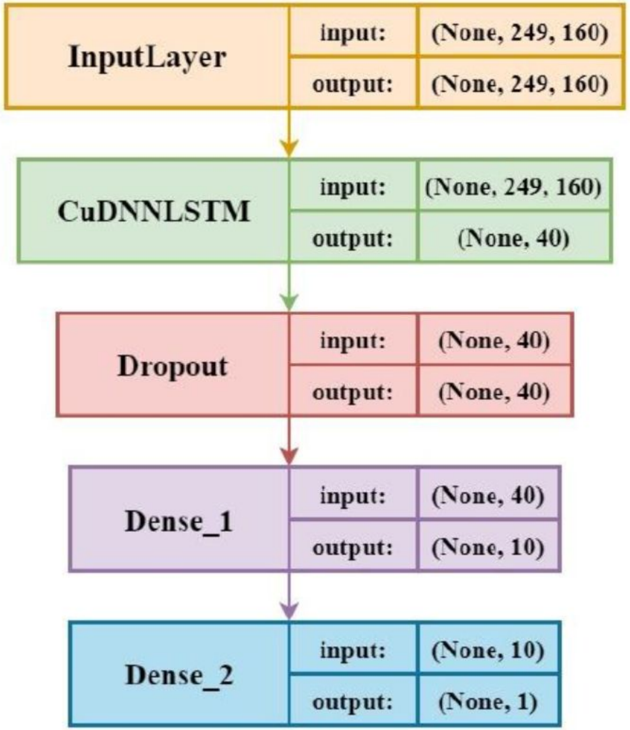


Table 2. Testing accuracies for Subject-Dependent and Subject-Independent models

Models	Subject-Dependent		Subject-Independent	
	Valence	Arousal	Valence	Arousal
KNN	86.03	79.64	70.86	68.36
SVM	76.56	72.66	**72.19**	**71.25**
Decision Tree	71.10	67.97	58.13	55.63
Random Forest	81.25	81.19	61.95	61.25
LSTM	**94.69**	**93.13**	70.31	69.53

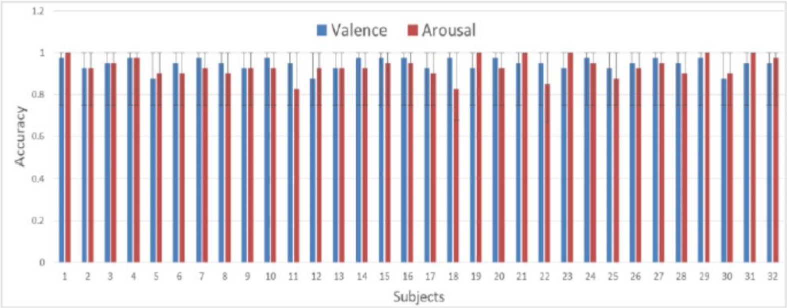
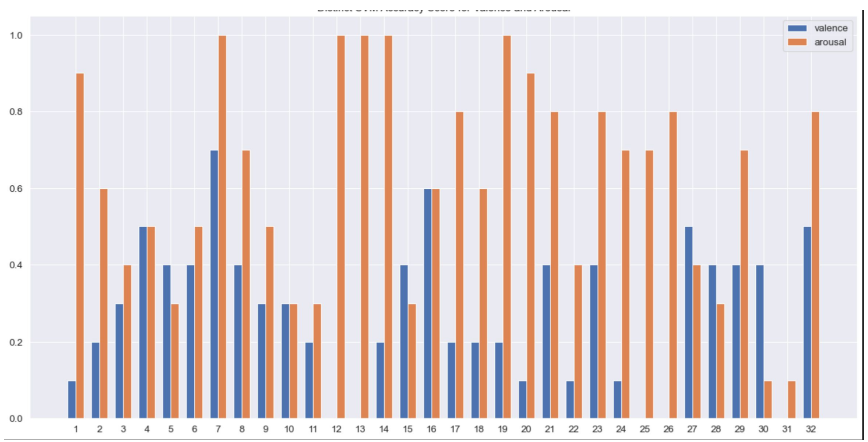


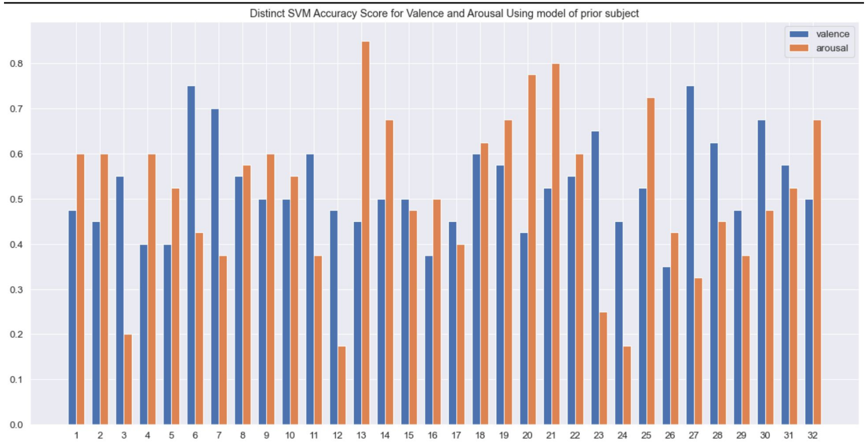
Figure 4. Testing accuracies for 32 subjects using LSTM model for subject-dependent model

Accuracies I got from SVM



Train and test split from each subject

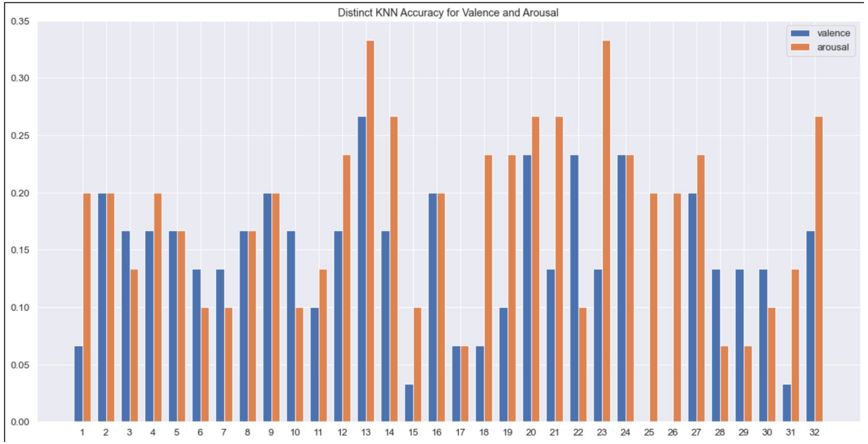
```
Mean Valence Accuracy on SVM: 28.0%
Mean Arousal Accuracy on SVM: 62.0%
```



Train from different subject and test from current subject

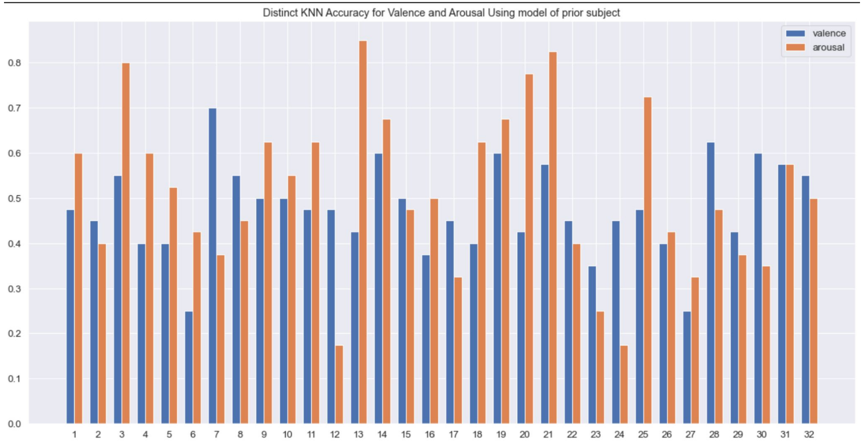
```
Mean Valence Accuracy on SVM: 53.0%
Mean Arousal Accuracy on SVM: 51.0%
```

Accuracies I got from KNN



Train and test split from each subject

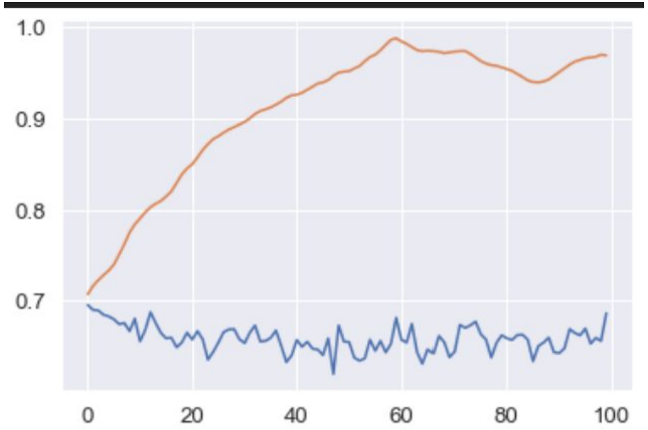
```
Mean Valence Accuracy on KNN: 14.0%
Mean Arousal Accuracy on KNN: 18.0%
```



Train from different subject and test from current subject

```
Mean Valence Accuracy on KNN: 48.0%
Mean Arousal Accuracy on KNN: 51.0%
```

Proposed Model Train and Validation Loss



Overfitting

```
Paper Proposed model accuracy with seperate subjects for train and test:  55.0
```

Arnajak Tungchoksongchai

Over all experiment

- Cross subject with 2 labels (high and low)
  - RNN
  - CNN
- Individual subject with 10 labels (0 - 9)
  - RNN

<u>Note</u>   all this experiment use 16 channel with are important to emotion classification

DEAP DATASET

Modality	Arousal ACC	Arousal F1	Valence ACC	Valence F1
EEG	0.620	0.583**	0.576	0.563**
Peripheral	0.570	0.533*	0.627	0.608**
MCA	0.651	0.618**	0.618	0.605**
Random	0.500	0.483	0.500	0.494
Majority class	0.644	0.389	0.586	0.368
Class ratio	0.562	0.500	0.525	0.500

DEAP experiment use SVM model and got acc around 60%

https://www.eecs.qmul.ac.uk/mmv/datasets/deap/doc/tac_special_issue_2011.pdf

Table 2
Methods of the Emotional Feature Extraction and Classification Using EEG Signals.

Reference	Classification Methods	Feature Extraction	Accuracy (%)
G. Zhao et al. [13]	SVM	STFT	81.08 Extraversion; 86.11 Agreeableness; 80.56 Conscientiousness; 83.78 Openness
J. Liu et al. [28]			86.43 Amusement VS. Joy VS. Tenderness; 65.09 Anger VS. Disgust VS. Fear VS. Sadness; 92.26 Non-Neutrality VS. Neutrality; 86.63 Positive VS. Negative
R. Nawaz et al. [45]			77.62 Valence; 78.96 Arousal; 77.60 Dominance
Z. Li et al. [56]		MLDW-PSO	89.50
Zhang et al. [66]		EMD, SE	94.98
Q. Gao et al. [67]		STFT, PS, WT, WEE	85.67 Neutral; 87.11 Happiness; 89.17 Sadness
T. B. Alakus et al. [47]	SVM (RBF)	DWT, Statistical	75.00 HAPV; 96.00 HANV; 71.00 LAPV; 79.00 LANV
H. Zamanian et al. [48]		EMD, Gabor Wavelet	93.86
N. Zhuang et al. [44]		EMD	69.10 Valence; 71.99 Arousal
H. Chao et al. [68]	SVM (Linear)	MIC	70.21 Valence; 71.85 Arousal
S. K. Khare et al. [55]	MC-LSSVM (RBF)	Adaptive TQWT	95.70
T. Chen et al. [46]	LIBSVM (Gaussian)	EMD	82.63 Valence; 74.88 Arousal
Z.Yin et al. [69]	LSSVM	PSD, Statistical	70.00 Valence; 67.00 Arousal
S. Taran et al. [70]	MC-LSSVM	EMD, VMD	92.79 Happiness; 87.62 Fear; 88.98 Sadness; 93.13 Relax
T. Song et al. [23]	DGCNN	DE	90.40
T. Zhang et al. [54]	STRNN	DE	89.50
S. Liu et al. [39]	DECNN	DDE	97.56
H. Cui et al. [71]	RACNN	Temporal, Regional, Asymmetric	96.65 Valence; 97.11 Arousal
A. Rahman et al. [72]	ANN	PCA, t-statistics	86.57
Alnafjan et al. [73]	NeuCube-Based SNN	–	84.62
S. K. Khare et al. [74]	Configurable CNN	SPWVD	96.71 Fear; 86.08 Happy; 93.83 Relax; 95.45 Sadness
Z. Liu et al. [30]	KNN	EMD	86.46 Valence; 84.90 Arousal
L. Piho et al. [38]		WT, SF, PSE, HOS, HOC	86.00 Valence; 87.20 Arousal
M. A. Ashar et al. [75]		DFC, DE	92.30
A. E. Putra et al. [76]	KNN (k = 21)	WD	57.50 Valence; 64.00 Arousal
D. Nath et al. [77]	LSTM	PSD	94.69 Valence; 93.13 Arousal (Subject-Dependent)
Y. Q. Yin et al. [78]		DE	90.45 Valence; 90.60 Arousal (Subject-Dependent) 84.81 Valence; 85.27 Arousal (Subject-Independent)
J. Yang et al. [79]	BiLSTM	DE	84.21
A. Garg et al. [80]	Merged LSTM	WT, Statistical	84.89 Valence; 83.85 Arousal; 84.37 Dominance
W. L. Zheng et al. [12]	DBN	DE	86.08
Y. Huang et al. [81]		W-WPCC	74.84 Anger; 71.91 Boredom; 66.32 Fear; 63.38 Joy; 69.98 Neutral; 72.68 Sadness
L. Zheng et al. [52]	GELM	STFT, PSD, DE, DASM, RASM	91.07
W. Zheng [53]	GSCCA	DE	82.45
E. S. Pane et al. [49]	RF	PSD, Statistical, DWT	75.60
H. Dabas et al. [50]	NB	PSD	78.06

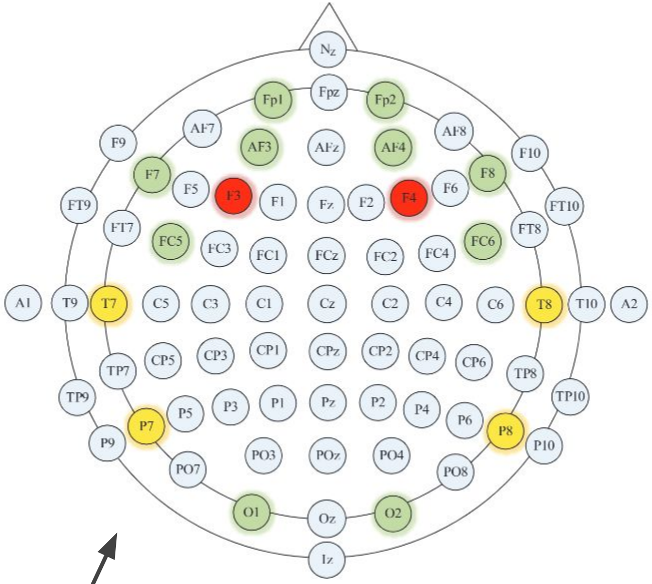


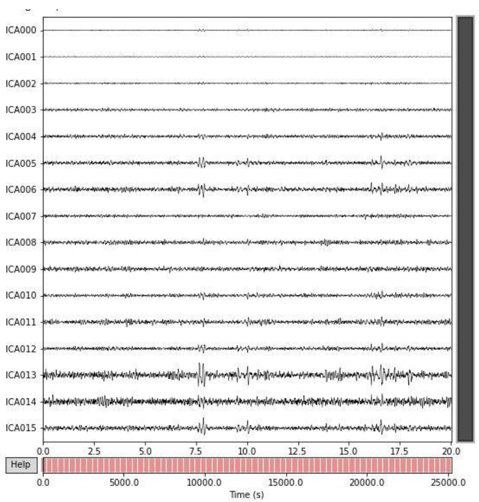
Fig. 5. 10–10 standard electrodes lead system.

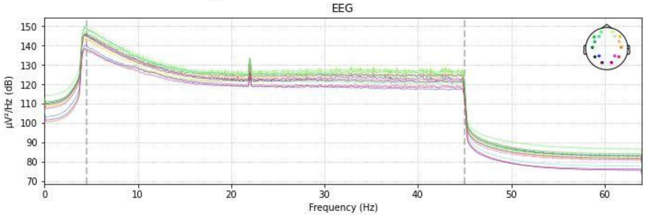
All experiment after this use only 16 important channel for emotion (when I try with 32 channel and this 16 channel result is similar)

It can be seen that the classification effect of the traditional machine learning
algorithms is generally not as good as the deep learning algorithms.

https://reader.elsevier.com/reader/sd/pii/S2667241321000033?token=BEC809AAF6D8967DF1F871898C4DB7728542CA96180E20154099C0C2C1413BCB193C45F6F8CA9F87BCAF3E9E29264FFE&originRegion=eu-west-1&originCreation=20211106183238

Recurrent Neural Networks





LSTM = 50.86%

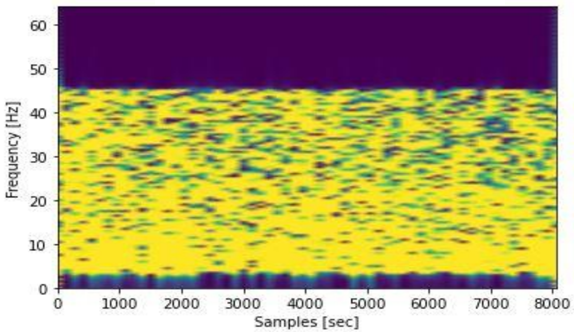
PSD + CONV1D + LSTM = 47.08%

CONV1D + LSTM + Attention = 54.03%

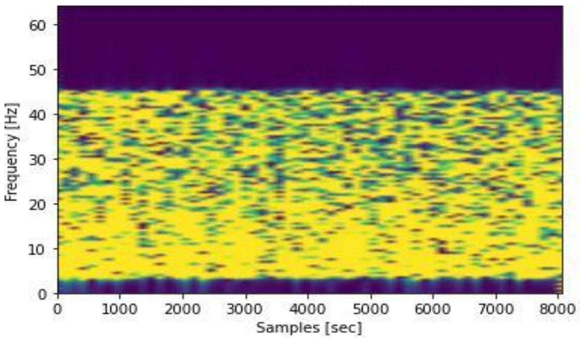
ICA + CONV1D + LSTM + Attention = 57.08%

Convolutional neural network

High valence



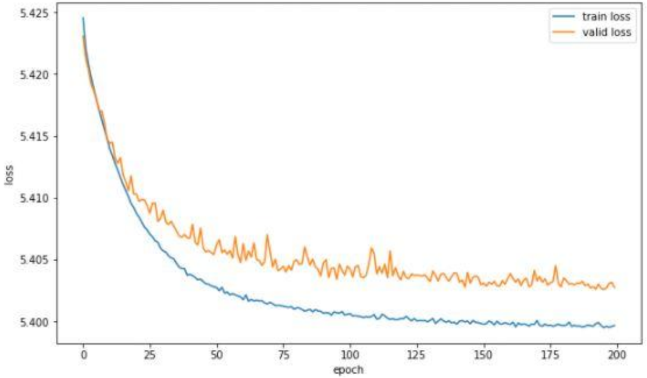
Low valence

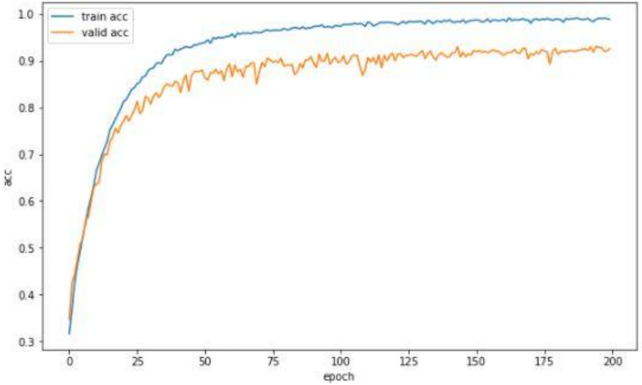


STFT + CONV2D = 54.78%

STFT + CONV2D + L2 = 60.94%

Individual subject





As the experiment before I separate label to 2 classes (high and low) so in individual subject make a label to 10 class(0-9)

CONV1D + LSTM + Attention = **92.57%**

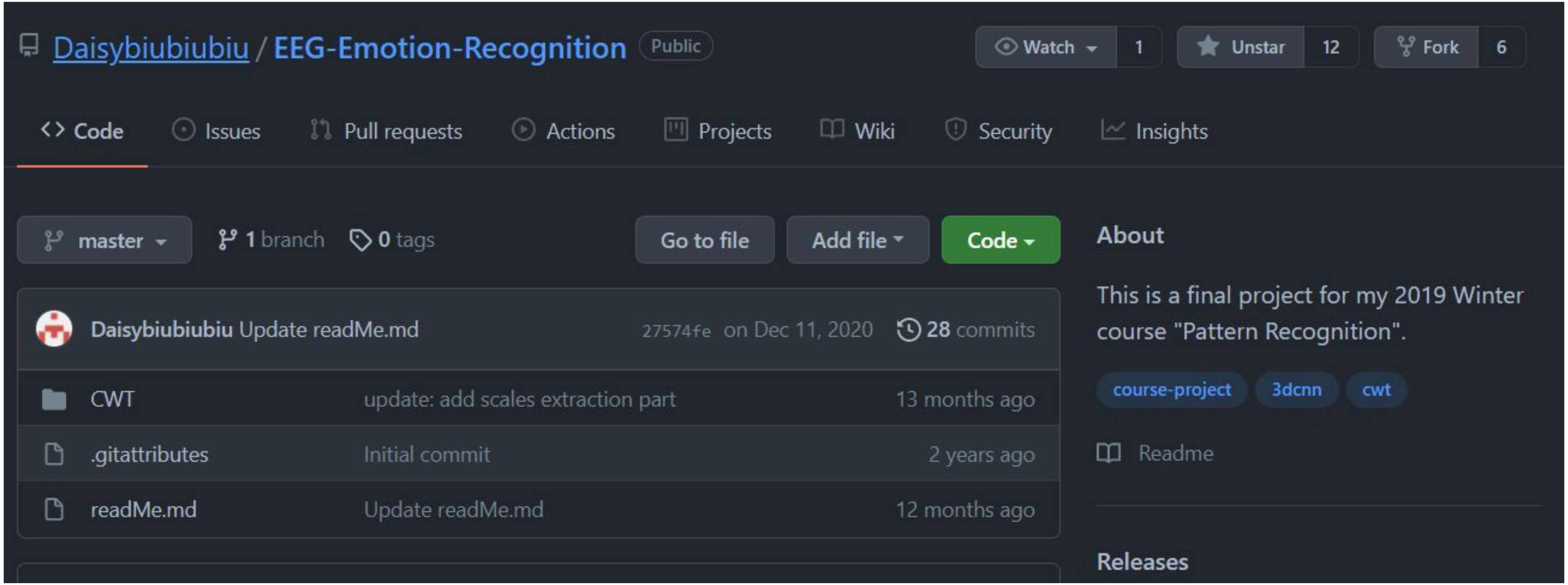
Conclusion :
Because participants performed a self-assessment of thiers levels of arousal and valence. In this way, there will be a lot of variance and bias when combining multiple people. This is why it get high accuracy in individual subject but not good enough in cross-subject.

Jirasak Buranathawornsom

Original Data

- 32 Participants
- 40 Clips
- 32 Channels
- 8064 datapoint

32*40, 32, 8064 -> 1280, 32, 8064



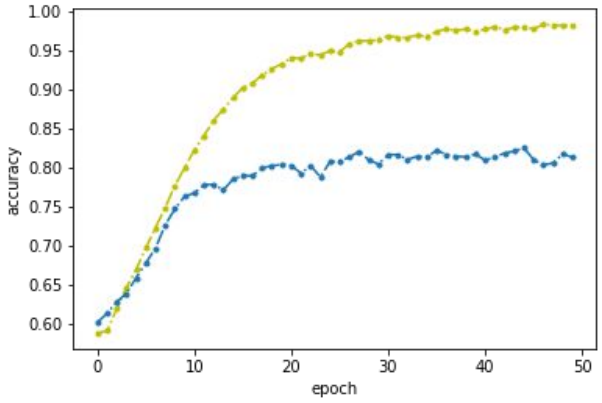
Daisybiubiubiu / EEG-Emotion-Recognition
Public
Watch 1
Unstar 12
Fork 6
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
master
1 branch
0 tags
Go to file
Add file
Code
Daisybiubiubiu Update readMe.md
27574fe on Dec 11, 2020
28 commits
CWT
update: add scales extraction part
13 months ago
.gitattributes
Initial commit
2 years ago
readMe.md
Update readMe.md
12 months ago
About
This is a final project for my 2019 Winter course "Pattern Recognition".
course-project
3dcnn
cwt
Readme
Releases

Preprocessing

1. Remove 3s baseline
    a. (1280, 32, **8064**) -> (1280, 32, **7680**)
2. Transform data in each channel using continuous wavelet transform, scale ranging from 1-64 Hz (64, 7680)
    a. (1280, **32**, 7680) -> (1280, 32, **64,** 7680)
3. Select only 32 scales from (8-40 Hz)
    a. (1280, 32, **64,** 7680) -> (1280, 32, **32,** 7680)
4. Sample only 60 datapoint out of 7680
    a. (1280, 32, 32**, 7680**) -> (1280, 32, 32**, 60**)

Train

1. Segment from 60 frame every 3 frame to get 20 more frame per original sample
    a. (**1280**, 32, 32**, 60**) -> (**1280 * 20 = 25600**, 32, 32**, 3**)
2. Randomly split the data into train and test
3. Feed the data with shape (32, 32**,** 3) into 3D CNN network



Result from their github

Emotion Recognition from Multi-Channel EEG through Parallel Convolutional Recurrent Neural Network

Emotion Recognition from Multi-Channel EEG through Parallel Convolutional Recurrent Neural Network

Yilong Yang
Software School
Xiamen University
Xiamen, China
yilongyang@stu.xmu.edu.cn

Qingfeng Wu*
Software School
Xiamen University
Xiamen, China
qfwu@xmu.edu.cn

Ming Qiu
Software School
Xiamen University
Xiamen, China
mingqiu@xmu.edu.cn

Yingdong Wang
Software School
Xiamen University
Xiamen, China
yingdongwang@stu.xmu.edu.cn

Xiaowei Chen
Software School
Xiamen University
Xiamen, China
wdenxw@stu.xmu.edu.cn

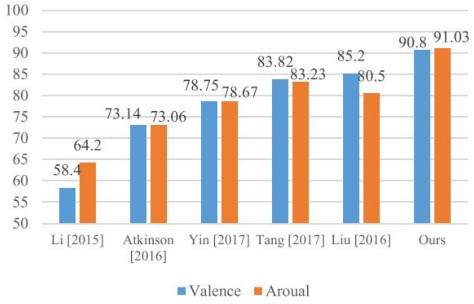
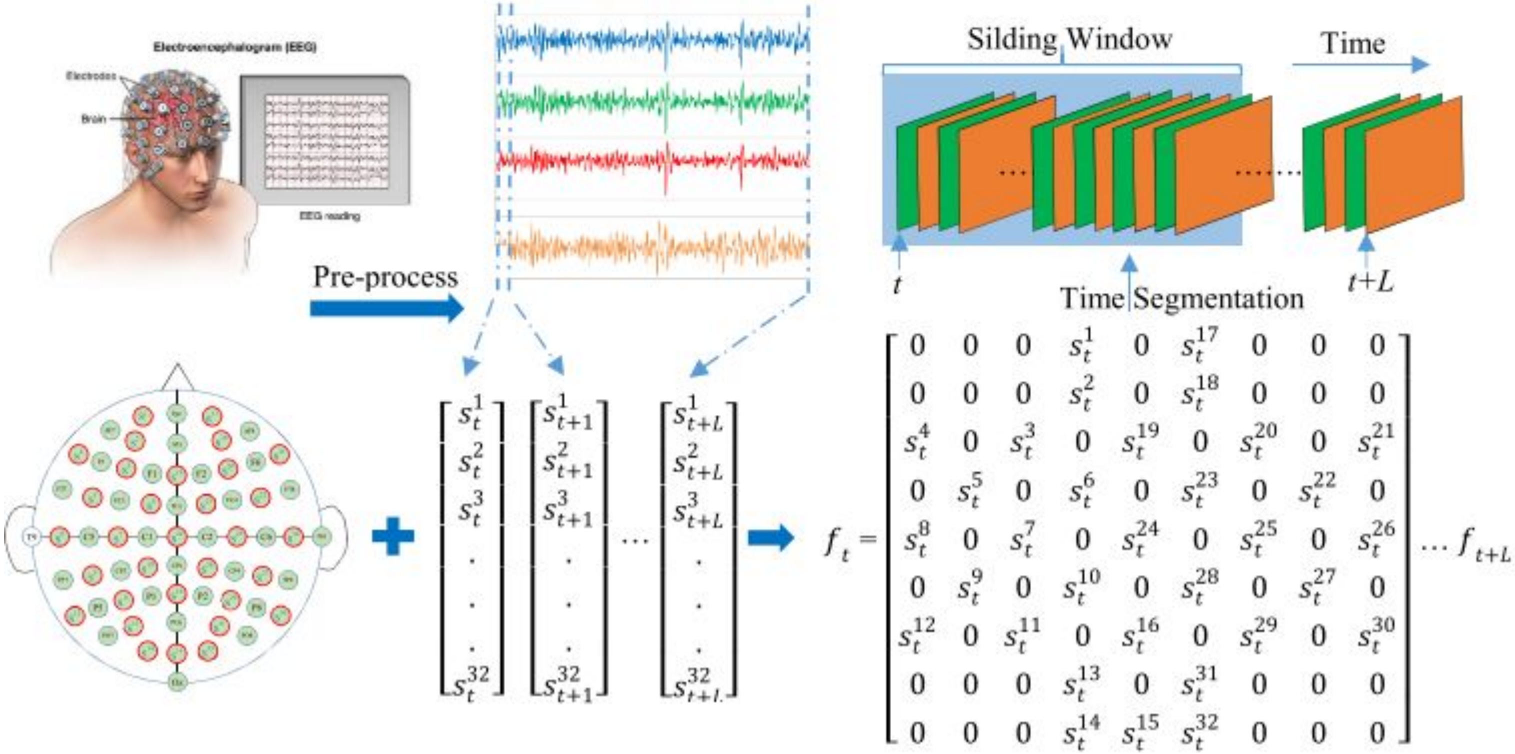


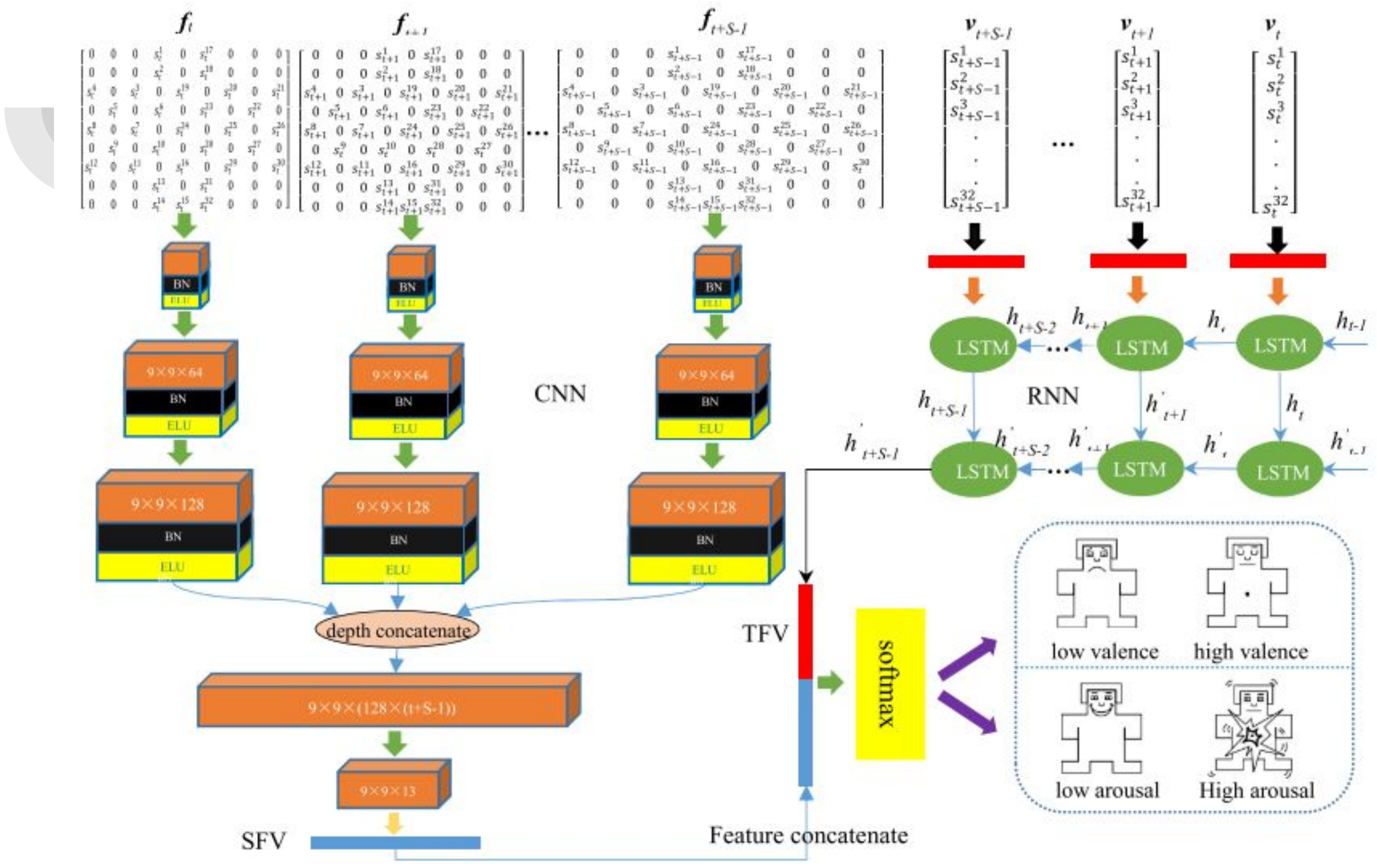
Fig. 4. Performance comparison between relevant approaches.

Preprocessing

1. Remove 3s baseline and normalize the data with baseline
    a. 1280, 32, **8064** -> 1280, 32, **7680**
2. Standardize data across 32 channels
3. Create CNN and RNN dataset
    a. CNN dataset
        i. Converting 1D EEG signals into 2D EEG frames
            1. 1280, **32**, 7680 -> 1280, **(9, 9)**, 7680
        ii. Segment the data with time window of 1s (128 points) so we get 7680/128=60 more samples per original sample
            1. **1280**, (9, 9), **7680** -> **(1280 * 60=76800)**, (9, 9), **128**
    b. RNN dataset
        i. Segment the data with time window of 1s (128 points)
            1. **1280**, 32, **7680** -> **(1280 * 60=76800)**, 32, **128**
4. Save CNN and RNN data (using numpy memmap)



Electroencephalogram (EEG)
Electrodes
Brain
EEG reading
Pre-process
Silding Window
Time
t
t+L
Time Segmentation



CNN
RNN
BN
ELU
9×9×64
9×9×128
depth concatenate
9×9×(128×(t+S-1))
9×9×13
SFV
TFV
softmax
Feature concatenate
LSTM
low valence
high valence
low arousal
High arousal

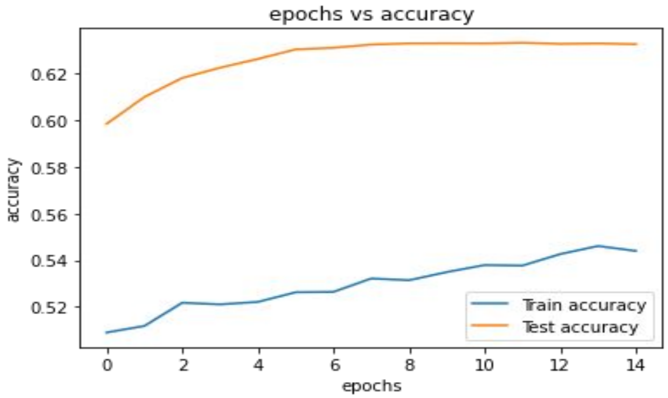
```python
class RNNCNN(nn.Module):
    def __init__(self, ):
        super().__init__()
        self.cnn_part = nn.Sequential(
            nn.Conv2d(in_channels=1, out_channels=32, kernel_size=(4,4), padding='same'),
            nn.BatchNorm2d(32),
            nn.ELU(),
            nn.Conv2d(in_channels=32, out_channels=64, kernel_size=(4,4), padding='same'),
            nn.BatchNorm2d(64),
            nn.ELU(),
            nn.Conv2d(in_channels=64, out_channels=128, kernel_size=(4,4), padding='same'),
            nn.BatchNorm2d(128),
            nn.ELU(),
        )
        self.one_one_conv = nn.Conv2d(in_channels=16384, out_channels=13, kernel_size=(1,1))

        self.lstm_part = nn.LSTM(input_size=128, hidden_size=32, num_layers=2, dropout=0.6, batch_first=True)

        self.final_part = nn.Sequential(nn.Linear(1085, 512),
                                        nn.Dropout(0.6),
                                        nn.Linear(512, 256),
                                        nn.Dropout(0.6),
                                        nn.Linear(256, 128),
                                        nn.Dropout(0.6),
                                        nn.Linear(128, 2)
                                        )
    def forward(self, cnn_data, rnn_data):

        # CNN
        cnn_data = cnn_data.view(-1, 1, 9, 9, 128)
        to_be_concatenated = []
        for channel in range(cnn_data.shape[-1]):
            result = self.cnn_part(cnn_data[...,channel])
            to_be_concatenated.append(result)

        result_concatenated = torch.concat(to_be_concatenated, axis=1)
        result_concatenated_one_one_conv = self.one_one_conv(result_concatenated)
        spatial_feature_vector = result_concatenated_one_one_conv.view(-1, 9*9*13)

        # RNN
        out, (h_n,c_n) = self.lstm_part(rnn_data)
        temporal_feature_vector = h_n[-1]

        # Combine CNN and RNN
        spatial_temporal_concat = torch.concat([spatial_feature_vector, temporal_feature_vector],1)

        output = self.final_part(spatial_temporal_concat)
        return output
```

Experiment

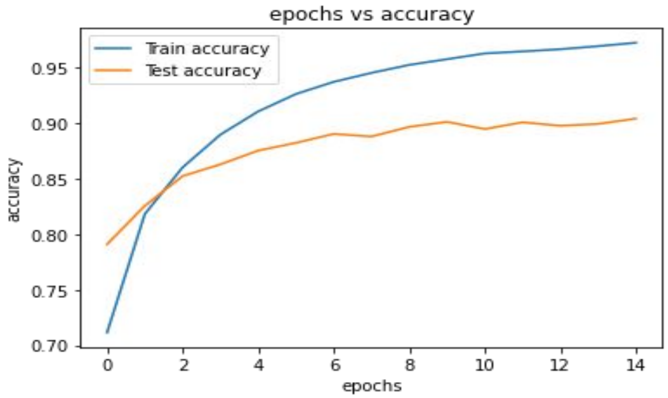
- Train on all participants
  - Case 1: From 32 participants,
    - First 25 participants: Train
    - Last 7 participants: Test
  - Case 2: Random Split with test_size = 0.2
- Train on one participants (1st participant)
  - Case 1: From 40 clips,
    - First 32 clips: Train
    - Last 8 clips: Test
  - Case 2: Random Split with test_size = 0.2

Train on all participants

- Case 1: From 32 participants,
  - First 25 participants: Train
  - Last 7 participants: Test
- Case 2: Random Split with test_size = 0.2



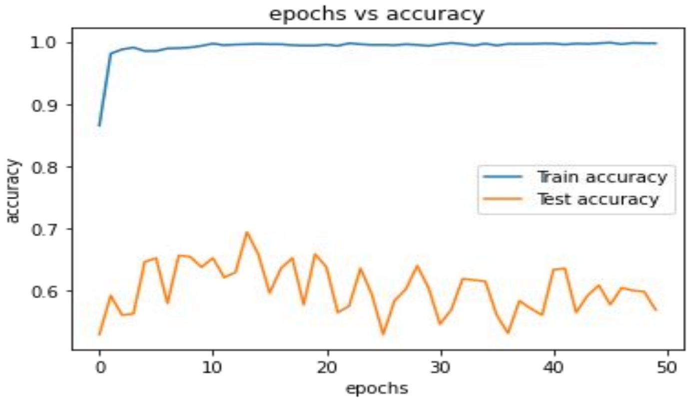
Case 1



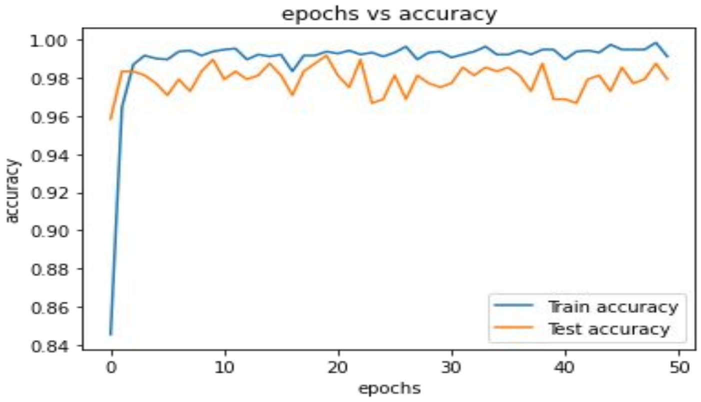
Case 2

Train on one participants (1st participant)

- Case 1: From 40 clips,
  - First 32 clips: Train
  - Last 8 clips: Test
- Case 2: Random Split with test_size = 0.2



Case 1



Case 2

Conclusion

- DEAP dataset is about Emotion and classification of emotions was harder than expected.
- Including temporal data could increase accuracy
- Better techniques ,feature extractions could be applied for better results.
- The data split could be more standardised
- While subject dependent could get high accuracy, subject independent could not achieve this in this dataset using our methods.

Things we have learned

- Learned how to handle signal data
- Learned about different signal processing techniques through different libraries
- Learned that even deep learning cannot learn everything without proper understanding of the domain.
- Got general idea of the BCI domain
- Learned to perform research and think scientifically.

THANK YOU