Statistical Comparison of Valence-Arousal Classifiers from EEG on DEAP and MANHOB Datasets

Riccardo Graziosi
E-mail: riccardo.graziosi@studenti.unimi.it
Natural Interaction and Affective Computing, A.Y. 2020-2021
University of Milan, Milan, Italy

Abstract—As devices for recording electroencephalography (EEG) signals become cheaper, there is growing interest for applications that use EEG data to predict human affective state. However, research papers in this field often suffer from poor reproducibility [1] and the reported results are quite fragile, lacking statistical significance and often based upon testing on a single dataset.

As a consequence, the aim of this paper is twofold: first, an attempt is made to reproduce an academic research on the topic and, second, the obtained models are tested through statistical experiments in order to compare different models and datasets.

For the first goal, the methodology for valence-arousal classification from EEG proposed in [2] is followed closely in order to try to reproduce the results reported therein. To reach the second goal, the models are then compared against each other using McNemar's and 5x2cv tests, and also on two different datasets, DEAP [3] and MAHNOB [4], with the aim to understand whether a model can perform similarly or not on two different but related datasets.

Unfortunately, we were unable to reproduce the results of [2]. After having established the unfeasibility of reproducing [2], slightly different choices in model architecture and training have been made in order to reach satisfactory performances.

Of the two models taken into consideration, a Deep Neural Network (DNN) and a Convolutional Neural Network (CNN), the first was able to reach the maximum accuracy on a particular training set, but the CNN proved to outperform the DNN on average. Using the same models, it was also found that higher accuracy is achieved on DEAP than on MAHNOB, but only to a small extent, showing that these models are robust enough to perform nearly equivalently well on both datasets.

1 Introduction

For quite some time, emotion recognition has been mostly based on video or audio recordings, because of the cheap cost of sensors like cameras and microphones. However, as technology progresses, it has been possible to build relatively low cost sensors to capture physiological signals as well, so in the affective computing community the interest in using this kind of data has sensibly grown lately. Electroencephalography (EEG) signals make no exception.

Parallel to this, there has been a huge increase in deep learning techniques usage, so it comes to no surprise that a lot of recent academic research focuses on training deep neural networks to recognize emotions from EEG. Moreover, since EEG data is known to be a complex signal to understand, the ability of deep neural networks to automatically learn features sounds promising.

Recent researches in this field confirm this hypothesis, with results showing deep neural models outperforming traditional techniques. However, a lot of these researches have proven themselves difficult or even impossible to reproduce, and rely on a single dataset for testing their models. Some studies, like [1], report alarming data about this problem: on average, researches on deep learning for EEG do not make publicly available the used datasets (50% of the times) or the code of the models (90%), and the difficulty of reproducibility is usually hard to impossible (90%).

The first goal of the present study is to repro-

duce the steps and obtain predictors with similar performances as the ones reported in [2]. In that research, two neural network models are trained, namely a simple Deep Neural Network (DNN) and a Convolutional Neural Network (CNN), to classify emotions from EEG data. The dataset used is DEAP [3], which is a famous benchmark database for affective computing applications. The study focuses on predicting emotional state based on the two continuous dimensions of valence and arousal, as proposed by Russell. In particular, the focus is on binary and 3-class classification of valence and arousal, whereas in the present study only binary classification is taken into consideration.

Even though all the steps described in [2] were followed closely, our model accuracy was nowhere near their reported one, leading to the conclusion that some data preprocessing steps have been omitted from their paper.

Another objective of the present research is to statistically compare different models (specifically, DNN and CNN) to understand if a significant difference between the two exists or not. Moreover, the models have been tested on two EEG datasets annotated with valence-arousal labels, namely DEAP and MAHNOB, in order to find out whether the same architecture can work well in both domains.

Results show that models trained and evaluated on DEAP tend to perform a bit better than those trained and evaluated on MAHNOB, even though this may be because of the difference in size between the two datasets. In general, it has been found that both models are able to perform with similar performance on DEAP and MAHNOB as well.

DNN and CNN models have also been compared against each other, on both datasets, using McNemar's test and 5x2cv paired t test. These tests have been chosen because of their low Type I error and decent statistical power, as pointed out in [5], and because they are the de facto standard nowadays. While McNemar's test was unable to spot any significant difference between the models, the 5x2cv test, being a bit more powerful, has been able to show that the CNN model is better than the DNN one in a statistically significant way.

The report is structured as follows. Section 2 summarizes the preprocessing steps, methodology and results of [2], and also contains a discussion about how much of the paper we were able to reproduce. Section 3 describes the datasets and the preprocessing steps applied to each one, whereas Section 4 details the employed neural architectures and the relative hyperpameters and training procedures. Then, Section 5 contains a summary of results and statistical tests for the comparison between models and datasets. Lastly, Section 6 elaborates on the results of the research with respect to the proposed goals. The report is completed by a small Appendix that can be used as a reference to easily navigate the source code of models and experiments provided.

2 Related Work

The paper that inspired this reasearch is [2], published in 2017 by Tripathi et al. The authors employed simple neural network models to predict valence and arousal from EEG data. The problem of predicting valence-arousal is defined as a classification problem, specifically they tested for binary and 3-class classification. For binary classification, valence-arousal values under 5 are considered low activation, whereas values over 5 are considered high activation.

The data used comes from the preprocessed version of the DEAP dataset [6]. Then, to allow for training with reasonable computational resources, they processed the dataset in order to reduce the dimensionality of the EEG data, by dividing each EEG trial into batches and summarizing each batch using statistical values like mean, standard deviation, minimum, maximum, etc.

The two employed models are basic deep neural networks. The first is a simple 4-layer neural network, constitued of fully connected layers, the other is a convolutional neural network with 2 convolutional layers, a max pooling layer and 2 fully connected layers.

The paper then reports results obtained through 32-fold cross validation by DNN and CNN using different hyperpameters configurations. The DNN model is able to reach 75.8% and 73.1% accuracies on valence and arousal respectively, whereas the CNN model reaches an impressive 81.4% and 73.4%.

2.1 Reproducibility of Related Work

The current section discusses what we have been able to reproduce from [2]. Since reproduced results were not so satisfactory, some changes have been made to the preprocessing procedures and to model architectures and hyperpameters. For this reason,

this section has been placed before Sections 3 and 4, which describe the final dataset preprocessing steps and models used for the present research.

Even though code and data of that research were not publicly available, reproducing the same preprocessing steps and models have not been a challenge, as the data processing was straightforward and the neural models pretty basic. However, the trained predictors were unable to learn from EEG data: models were either underfitting or overfitting, but no general patterns were discovered by them.

The problem was mitigated by standardizing the dataset, as explained in Section 3, with models being able to learn some patterns from the data. The standardization step was not explicitly cited in [2], but it is such a common step that it could be claimed it is an implicit one to perform.

However, even after standardization, the accuracy of the resulting models was nowhere near the one stated in [2], reaching a maximum of 80% for certain training/test splits, but averaging at about 60% accuracy, as opposed to the average 75% accuracy of the reproduced research. The results reported in Section 5.1, even though they are based on slightly different hyperpameter choices and model architectures, are nearly identical to the ones obtained by reproducing the paper exactly, so they can be used as a metric to compare the expected accuracy against the real obtained one.

Since the training procedure and model architectures are no different from standard neural networks used in other fields, the problem probably relies in the data. Different types of standardization have been tried (per channel, per trial, per participant, global; after or before dimensionality reduction) but no gains in accuracy were obtained. Therefore, it is likely that authors of [2] used a custom preprocessed version of DEAP, even though they never explicitly mentions that other preprocessing procedures were performed.

After being unable to reproduce the results of [2], some choices were made that diverge from that research: for example, only the 32 EEG channels were used instead of the full 40 channels which also contain other physiological signals, in order to have the exact same set of features in both DEAP and MAHNOB. Model architectures and hyperpameters have also been slightly modified. A detailed description of both datasets and models can be found in Sections 3 and 4.

3 Datasets

DEAP and MAHNOB datasets have been chosen for this research because they both contain EEG data and valence-arousal annotations. Valence and arousal annotations are based on Russels' scale, which is widely used in affective computing. Using Russels' valence-arousal scale, each emotional state is a point on a 2D plane, with valence and arousal being the horizontal and vertical axes respectively (see Figure 1). So, a combination of valence and arousal results in a specific emotion. In particular, valence can range between unpleasant and pleasant, and arousal can range between inactive or active.

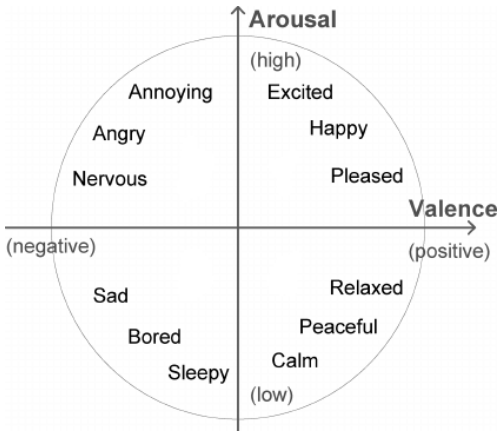

Fig. 1: Russels' valence-arousal circumplex. Emotion positions are approximate. Image taken from [7].

3.1 DEAP

DEAP [3] is a dataset for emotion analysis released in 2014. It is one of the biggest publicly available datasets in the affective computing field and also contains a wide variety of different physiological and video signals.

The DEAP datasets consists of two parts:

1) A database of 120 one-minute music videos, each one rated by 14-16 volunteers based on valence, arousal and dominance.
2) A subset of 40 of the above music videos, each one with the respective EEG and physiological signals for each of the 32 participants. As in the first part, every video was rated on valence, arousal and dominance dimensions.

For the sake of this report, only the second part of the DEAP dataset is used, which contains EEG signals.

EEG signals have been collected using the Biosemi ActiveTwo device, which records 32 EEG channels with configurable sampling rate. DEAP was collected at 512Hz, but the creators of the dataset also provide a preprocessed version of EEG signals, downsampled to 128Hz and with frequency filters and other useful preprocessing steps already applied.

In particular, for each of the 32 participant, the following preprocessed information is present:

- data: a 40 x 40 x 8064 array that contains 8064 recordings for each of the 40 channels and for each of the 40 music videos. There are 8064 recordings per channel per video because the trials are 63 seconds long (3 seconds pre-trial baseline + 60 seconds trial) and the sampling rate is 128Hz (63 x 128 = 8064).
- labels: a 40 x 4 array that contains an annotation of valence, arousal, dominance and linking for each of the 40 music videos.

This preprocessed information has been, in turn, processed again as explained in Section 3.3.

3.2 MAHNOB

MAHNOB [4] is a dataset for affect recognition released in 2012. It is a multimodal dataset that provides audio, video and physiological signals, as well as eye gaze data. All the data is synchronized and annotated with respect to valence and arousal affective dimensions.

Four different types of experiments have been conducted:

1) In the first type of experiments, a video was shown to participants who had to annotate their level of valence and arousal in response to the video stimulus.
2) In the other three types of experiments, a tag was placed at the bottom of the screen: the tag could or could not be related to the movie shown. In this case, participants were asked to rate how much the tag was pertinent to the video.

For the sake of this report, only data from the first type of experiments is used.

EEG signals have been recorded using the same device used to collect the DEAP dataset, the Biosemi ActiveTwo. For this reason, EEG signals have 32 channels as well but MAHNOB was collected at 256Hz instead of 512Hz. As opposed to DEAP, MAHNOB does not provide a preprocessed version of the dataset, but rather the raw collected files, which are of .bdf format for EEG signals. In order to work with this data it is necessary to perform more preprocessing steps than with DEAP, which are described in Section 3.3.

3.3 Dataset Preprocessing

Data from both DEAP and MAHNOB has been preprocessed. The following two subsections explain the preprocessing steps applied on both datasets in detail.

3.3.1 DEAP Preprocessing

Data dimensionality has been reduced. The 40 channels have been trimmed down to 32, keeping only the EEG signals, and the 8064 readings per channel have been reduced to 99 values.

In order to perform this latter processing, which was taken by [2], the 8064 recordings have been divided in 10 batches of approximately 807 readings each. Then, for each batch the following statistical values have been extracted: mean, median, maximum, minimum, standard deviation, variance, range, skewness and kurtosis, yielding 9 values per batch (90 values for 10 batches). Then the same values are calculated for the entire 8064 reading, yielding 9 additional values for a total of 99 values.

These summary values are then standardized on an example basis, yielding a mean of 0 and a standard deviation of 1, using the following formula:

$$x_{i,j} = \frac{x_{i,j} - mean(X)}{std(X)}, \tag{1}$$

where X is the entire 32x99 example, and $x_{i,j}$ is the value of the j-th reading of the i-th channel.

3.3.2 MAHNOB Preprocessing

This dataset provides raw EEG data in .bdf format, which has been collected with the Biosemi ActiveTwo device. Since this data is not preprocessed, some additional work had to be done. In order to work with raw EEG signals, the MNE Python library [8], which is specifically addressed to process and visualize human neurophysiological data, has been used.

The same preprocessing steps applied on the official preprocessed version of the DEAP dataset, as reported in [6], has been applied. In particular, EEG signals have been referenced to the channel "Cz", which is a common channel for referencing, even suggested in Biosemi FAQ [9]. A 4-45Hz bandpass frequency filter was applied, but actually yielded poorer results, so it was removed. Moreover, since MAHNOB does not provide a fixed number of recordings per session and also contains recordings for 30 seconds before and after the experiments, the needed recordings have been extracted from the middle of the trial.

Then, the same preprocessing steps used on DEAP (explained in Section 3.3.1) have been applied, with just a minor adjustment: 16128 (8064 x 2) readings are taken into consideration instead of 8064 and preprocessing batches size is also doubled, because the MAHNOB dataset provides raw data collected at 256Hz, as opposed to DEAP which provides a downsampled 128Hz version of the data. In this way the temporal window covered by a batch is the same for the two datasets.

3.3.3 Preprocessed Datasets Summary

After the preprocessing steps explained in the previous sections, both datasets contains data with the same shape, as described in Table 1.

Dataset	Number of examples	Data shape	Label shape
DEAP [3]	1280	32 x 99	2
MAHNOB [4]	546	32 x 99	2

TABLE 1: Dataset size and data shape after preprocessing steps. Data contains 32 channels with 99 recordings each, whereas labels contain 2 values (valence and arousal).

The scripts to perform these processing steps are available in the repository of the project under the names *prepare_deap.py* and *prepare_mahnob.py* respectively.

Both datasets have been divided into train and test sets, with split ratios (1180, 100) and (460, 86) for DEAP and MAHNOB respectively. Unfortunately, the original MAHNOB dataset contains 1183 sessions, but only 546 out of these are annoted with valence and arousal, yielding a dataset which is quite small for the present use case.

4 MODELS

Two different neural network architectures have been adopted for this research: a deep neural network (DNN) with fully connected layers and a convolutional neural network (CNN), which have been taken from [2], with just some minor modifications. Both models have been developed using Python and PyTorch [10], and the source code can be found in $scripts/nn/models.py$.

The following subsections explain each of these models and the training techniques in detail.

4.1 Deep Neural Network (DNN)

The DNN model is a deep neural network with 3 hidden layers. An approximate graphical scheme of the architecture is depicted in Figure 2, whereas the exact details of each layer are described in Table 2.

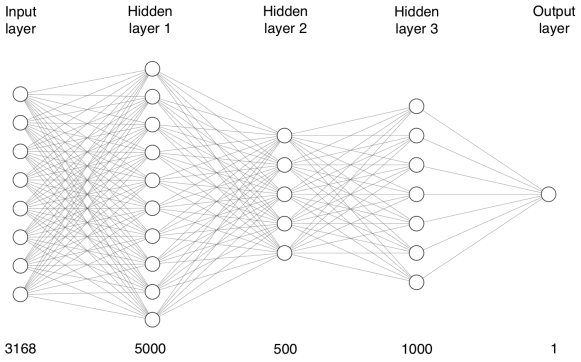

Fig. 2: DNN architecture. The number of depicted neurons is just for representation purposes, the real number of neurons is reported under each layer.

ReLU activation functions were used after each dense layer (except for the last) to introduce nonlinearity to the model, whereas a sigmoid is applied after the last layer to squeeze the output to the interval [0, 1]. Since in this paper valence/arousal classification is treated as a binary classification problem (low or high), a single output neuron with value in [0, 1] represents the network inferred probability that the input signal refers to a high valence/arousal emotional state.

Dropout technique is used heavily because of the small amount of data available for training, in order to avoid overfitting.

All weights of the network are initialized with the Xavier's normal method [11], whereas all biases are initialized with value 0.

Layer type	Layer params	Output shape
Flatten	-	(3168)
Dropout	p = 0.25	(3168)
Dense	in = 3168, out = 5000	(5000)
Dropout	p = 0.50	(5000)
Dense	in = 5000, out = 500	(500)
Dropout	p = 0.50	(500)
Dense	in = 500, out = 1000	(1000)
Dropout	p = 0.50	(1000)
Dense	in = 1000, out = 1	(1)
Number of parameters = 18'847'501		

TABLE 2: Deep Neural Network (DNN) architecture.

Hyperparameters, optimizer and loss function used for training are reported in Table 3. These slightly differ between the two datasets.

	DEAP	MAHNOB
Batch size	310	70
Epochs	250	250
Loss function	BCE	BCE
Optimizer	RMSProp	RMSProp
Learning rate	0.0001	0.0001
Momentum	0	0

TABLE 3: Hyperparameters, loss function, optimizer for the training procedure of the DNN. BCE = Binary Cross Entropy; RMSProp = Root Mean Squared Propagation.

4.2 Convolutional Neural Network (CNN)

The CNN model makes use of convolutional layers and treats data as a two-dimensional input of shape 32 x 99. Figure 3 depicts the architecture and Table 4 describes each layer in detail.

In short, the model consists of two convolutional layers followed by a max pooling layer, and, at last, two fully connected layers. Convolutional layers

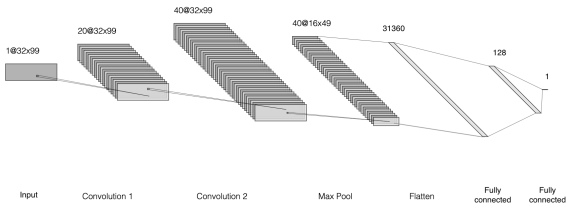

Fig. 3: CNN architecture.

Layer type	Layer params	Out shape
Dropout	p = 0.25	(32, 99)
Convolutional	kernel size = 3x3 num. of kernels = 20 padding = 1 x 1	(20, 32, 99)
Dropout	p = 0.15	(20,32,99)
Convolutional	kernel size = 3x3 num. of kernels = 40 padding = 1 x 1	(40, 32, 99)
Dropout	p = 0.15	(40, 32, 99)
MaxPool	kernel size = 2x2	(40, 16, 49)
Flatten	-	(31360)
Dropout	p = 0.50	(31360)
Dense	in = 31360, out = 128	(128)
Dropout	p = 0.25	(128)
Dense	in = 128, out = 1	(1)
Number of parameters = 4'021'777		

TABLE 4: Convolutional Neural Network (CNN) architecture.

treat the input as a 2D image, applying 3x3 filters through the convolution operation. This type of layers is predominantly used in tasks involving images. Max pooling layers are used to reduce the spatial dimension of the data, sliding a 2x2 window across the image which gets reduced to a single value: the value of the neuron with highest activation. Max pooling reduces the spatial dimensions of the image, hence reducing the number of parameters needed in the final fully connected layers and helping the network to avoid overfit.

As with the DNN model, CNN weights are initialized with Xavier's normal technique and biases are set to 0.

Hyperparameters, optimizer and loss function used

for training are reported in Table 5. These slightly differ between the two datasets.

	DEAP	**MAHNOB**
Batch size	50	70
Epochs	250	250
Loss function	BCE	BCE
Optimizer	SGD	SGD
Learning rate	0.001	0.0001
Momentum	0.9	0.9

TABLE 5: Hyperparameters, loss function, optimizer for the training procedure of the CNN. BCE = Binary Cross Entropy; SGD = Stochastic Gradient Descent.

5 Analysis of Results

This section is divided into multiple subsections.

Section 5.1 focuses on the comparison of the obtained results with the expected ones from the reproduced research [2] and on the difference between model performances on DEAP and MAHNOB.

On the other hand, Section 5.2, describes the statistical tests performed in order to compare the DNN and CNN model against each other, with the goal of discovering whether there exists a significant difference between the two models or not.

Lastly, Section 5.3 is devoted to a small summary of the performances of models for arousal classification.

5.1 Analysis of Results Between Datasets

The first way in which models have been evaluated is the simplest one. As already mentioned in Section 3.3, each dataset have been split in two subsets: training and test parts. For this experiment, models have been trained on the training part of the datasets and tested on the test set of the respective dataset.

The results for binary valence classification can be found in Table 6. These particular results refer to the best model obtained during the training process.

From these results it seems that, in general, models perform better on DEAP than on MAHNOB. A factor that certainly contribute to this is that MAHNOB dataset has less than half the number of examples of DEAP, which makes the model harder to train and more prone to overfitting. Looking at Table 6, the DNN model seems to outperform the CNN model on both datasets, but especially on DEAP; anyway, this informal observation has been challenged in Section 5.2, which statistically compares the two models.

Model	**Dataset**	**Accuracy**	**Conf. int.**
DNN	DEAP	71.0%	62.1% - 79.9%
	MAHNOB	65.2%	56.3% - 76.3%
CNN	DEAP	66.0%	56.7% - 75.3%
	MAHNOB	65.1%	55.0% - 75.2%

TABLE 6: Results of DNN and CNN models for valence classification on DEAP and MAHNOB datasets. Confidence interval refers to a significance level of 95% and have been computed by approximating the binomial distribution of test set evaluation to a Gaussian distribution. The script *confidence-intervals.py* contains the code used for the computation.

Models have also been evaluated using K-fold cross validation. For this technique, the dataset is partitioned in K folds of same size (if possible), then, in turn, each fold is used as a test set, whereas the rest of the dataset is used as the training set. As a result, K models gets trained and their accuracies evaluated, so the final reported accuracy for K-fold cross validation is the mean of these accuracies.

The results for 32-fold cross validation on DEAP and 6-fold cross validation on MAHNOB are reported in Table 7.

Model	**Dataset**	**Accuracy**
DNN	DEAP	58.7%
	MAHNOB	55.1%
CNN	DEAP	59.2%
	MAHNOB	56.4 %

TABLE 7: Results of K-fold cross validation for DNN and CNN on DEAP and MAHNOB. DEAP runs used 32 folds, whereas MAHNOB runs used 6 folds. The script to reproduce this experiment can be found under the name *kfold_cross_validation.py*.

The accuracies found using K-fold cross validation are far inferior than those found using a fixed training/testing split. As a consequence, it can be

said that models suffer from high variance error, i.e. its performance is highly correlated to the specific train and test sets fed to them. It is likely that the train/test split operated on the datasets for results of Table 6 were "lucky" splits, yielding high accuracy by chance.

The high-variance conjecture is also confirmed by the specific fold accuracies obtained during K-fold cross validation. For example, in the K-fold run for the DNN model on DEAP, fold accuracies ranged from 43% to 78%, showing how different dataset splits can radically change accuracy results. The same behaviour has been observed with MAHNOB as well, though to a less extreme degree.

K-fold results on DEAP can be compared to the ones reported in [2], since that research also used 32-fold cross validation as an evalutation technique. They obtained 75% and 81% accuracy with DNN and CNN respectively, against our 58% and 59%. The gap in accuracy is huge, and, even though the dataset and models used in the present research diverged from the ones of [2], quite similar results as the ones of Table 7, as explained in Section 2.1, have been obtained using the same exact data preprocessing steps and model architectures of [2].

K-fold results also confirm the previous results in that both models perform better on DEAP rather than MAHNOB. Another interesting observation is that the CNN model slightly outperformed the DNN model on both datasets, whereas, when evaluating on a single train/test split, the DNN model was able to reach higher maximum accuracy.

5.2 Statistical Tests for Comparing Models

For simplicity, all statistical tests have been performed on models for valence prediction. However, based on the results of Section 5.3, we believe the outcome of statistical tests would be similar for arousal models.

5.2.1 McNemar's Test

McNemar's test has been adopted in order to test whether there is a statistically significant difference between the performance of DNN and CNN models. To carry out this test, the predictors whose results are reported in Table 6, i.e. the predictors trained on the default train/test splits of DEAP and MAHNOB, have been used. The script to reproduce the McNemar's tests presented in this section is *mcnemar-test.py*.

McNemar's test works as follows [5]: the predictors to compare, in this case f_{DNN} and f_{CNN}, are evaluated against the test set, and in the meanwhile the following contingency table is constructed:

n_{00}	n_{01}
n_{10}	n_{11}

where n_{00} is the number of samples in the test set misclassified by both predictors, n_{01} the number of samples misclassified by f_{DNN} but not by f_{CNN}, n_{10} the number of samples misclassified by f_{CNN} but not by f_{DNN} and n_{11} the number of samples correctly classified by both predictors. As a result, $n_{00}+n_{01}+n_{10}+n_{11}$ equals the number of examples in the test set.

The null hypothesis of McNemar's test is that both predictors have the same error rate, i.e. $n_{01} = n_{10}$. The test compares the expected counts of n_{01} and n_{10} to the real obtained counts, using a goodness-of-fit Chi-Square test.

In practice, the following McNemar's test statistic is greater than $X^2_{1,0.95} = 3.841$ with probability less than 5%:

$$\frac{(|n_{01} - n_{10}| - 1)^2}{n_{01} + n_{10}} \tag{2}$$

so, in that case, the null hypothesis can be confidently rejected, i. e. the two predictors have significantly different performances on the chosen training and test sets.

The contigency table obtained with DNN and CNN models trained on DEAP is the following:

15	14
19	52

and the resulting statistic is 0.487, which is not enough to confidently reject the null hypothesis. So, even though the DNN and CNN predictors have different performances as reported in Section 5, the McNemar's test suggests we should accept the null hypothesis, i.e. the two predictors have no significantly different performances.

For predictors on MAHNOB, the following contingency table has been obtained:

14	15
16	41

Even without making any computations, it can be seen that n_{01} and n_{10} are nearly the same, so, in

this case as well, it can be said that, following McNemar's test, the two predictors have basically the same performance.

5.2.2 5x2cv Paired t Test

While McNemar's test is about the comparison of two predictors (where a predictor is considered the result of running a learning algorithm, i.e. the resulting model), the 5x2cv test compares two learning algorithms. So, in order to carry out this test, there is no need to use the pretrained models presented in Section 5, as it was the case with McNemar's test.

The 5x2cv paired *t* test is a statistical test based on 5 repetitions of 2-fold cross validation, which aims to discover whether there is a significant performance difference between two learning algorithms [5]. The test exhibit low Type I error, even though not as low as McNemar's test. On the other hand, the power of the 5x2cv test is higher than McNemar's, i.e. the test is better at detecting a difference when that difference really exists.

One big disadvantage of 5x2cv test is that it is quite computationally expensive, ten times more than McNemar's test. Dietterich, in [5], suggests to use 5x2cv over McNemar's when it is computionally feasible to do so, and fortunately that is the case for the data and models of this research.

The test works as follows. 5 iterations of 2-fold cross validation are performed. In each iteration, data is partitioned in two sets, S_1 and S_2, then the two learning algorithms A and B are both trained first on S_1 and tested on S_2, then vice versa. As a result, four error estimates are obtained: $p_A^{(1)}$, $p_B^{(1)}$, $p_A^{(2)}$ and $p_B^{(2)}$. For each fold, estimated differences can be computed as follows: $p^{(1)} = p_A^{(1)} - p_B^{(1)}$ and $p^2 = p_A^{(2)} - p_B^{(2)}$. Then, estimated variance is: $s^2 = (p^{(1)} - \overline{p})^2 + (p^{(2)} - \overline{p})^2$, where $\overline{p} = \frac{(p^{(1)}+p^{(2)})}{2}$. Since this computations are repeated for each iteration, we get s_i^2 for $i = 1, ..., 5$. Then, the test statistic can be computed as follows:

$$\tilde{t} = \frac{p_1^{(1)}}{\sqrt{\frac{1}{5}\sum_{i=1}^{5} s_i^2}} \tag{3}$$

Under the null hypothesis, $\tilde{t}$ follows a *t* distribution with 5 degrees of freedom. So, by setting alpha at 0.05, the null hypothesis can be rejected if $\tilde{t} > 2.571$ or $\tilde{t} < -2.571$.

The 5x2cv test has been used to compare DNN and CNN models on DEAP and MAHNOB. The same architectures and hyperpameters reported in Sections 4.1 and 4.2 have been used for these tests, except for the number of epochs which have been reduced to 150 in order to meet hardware limitations. The script to reproduce these results can be found under the name *5x2cv-test.py*.

On DEAP, the resulting statistic $\tilde{t}$ was -2.502, which is very close to -2.571, the threshold value for rejecting the null hypothesis with 95% confidence. For a slightly higher alpha value, for example 0.06, it is possible to reject the null hypothesis, meaning that there is probably a statistically significant difference between the two compared learning algorithms.

Surprisingly, while in the results of Table 6 of Section 5 the DNN network was able to reach higher accuracy than the CNN, in this case the average accuracy of the two models was 54.3% and 57.2% for DNN and CNN respectively, so the CNN model outperformed DNN. Note that accuracies for 2-fold cross validation are worse than the one reported in Section 5 because the training set is much smaller in this case, which likely leads to overfitting.

On the other hand, on MAHNOB, the $\tilde{t}$ statistic computed by the test was 0.306, showing that the two models perform similarly on this dataset.

5.3 Results for Arousal Classification

The present research mainly focused its attention of valence classification, but some experiments were also performed on arousal classification. Specifically, K-fold cross validation has been carried out for it as well, yielding the results of Table 8.

Model	Dataset	Accuracy
DNN	DEAP	56.9%
	MAHNOB	54.2%
CNN	DEAP	59.1%
	MAHNOB	54.4 %

TABLE 8: Results of K-fold cross validation for DNN and CNN on DEAP and MAHNOB for arousal binary classification. DEAP runs used 32 folds, whereas MAHNOB runs used 6 folds.

These results are consistent with the ones from valence classification, highlighting that the CNN model seems to outperform the DNN model a little. They are also in agreement with the results reported

in [2], since they also show a slight deterioration of accuracy with respect to valence classification.

6 Conclusions

In this work we first attempted to reproduce the results of another paper, [2], but we were unable to do so, with models reaching far inferior accuracies that the ones reported in the reproduced paper.

However, it was found that both tested models are able to perform similarly on DEAP and MAHNOB, which means they have proven to be pretty robust classifiers of valence-arousal from EEG that could probably be employed with small to no changes for other EEG-based datasets. Since these results are obtained with basic and generic neural network models like the ones described in this study, it is reasonable to think that more ad-hoc and complex neural architectures may perform even better on emotion classification from EEG.

Moreover, the CNN architecture turned out to be significantly better, from a stastical standpoint, than the DNN model, at least on DEAP. This result is important because future research may benefit more from experimenting with different CNN architectures than with DNN architectures.

7 Appendix: Source Code Structure

- *dataset*: scripts that perform all the preprocessing steps detailed in Section 3.
  - *prepare_deap.py*: prepares the DEAP dataset.
  - *prepare_mahnob.py*: prepares the MAHNOB dataset.
  - *reduce_dim.py*: utility function that performs the dimensionality reduction on input EEG data. Used by both *prepare_deap.py* and *prepare_mahnob.py* scripts.
- *nn*: scripts and configurations for datasets, models, training procedures.
  - *datasets.py*: PyTorch's class to read examples from DEAP and MAHNOB.
  - *models.py*: PyTorch's models for DNN and CNN architectures, detailed in Section 4.
  - *train_utils.py*: contains the code for the training procedure.
  - *train.py*: launches the training procedure.
  - *utils.py*: contains utility functions used across other scripts.
  - *configs/*: contains YAML configuration files for models, hyperpameters, training procedures
- *statistical_analysis*: scripts to perform the statistical tests used in the paper.
  - *5x2cv_test.py*: performs the 5x2 cross validation paired t test (Section 5.2.2).
  - *confidence_intervals.py*: computes the confidence intervals for the results of Section 5.
  - *kfold_cross_validation.py*: performs k-fold cross validation. Used for carrying out the 5x2cv test but can also be used on its own.
  - *mcnemar_test.py*: performs McNemar's test (Section 5.2.1) on pretrained DNN and CNN models.
- *pretrained_models*: contains the 4 pretrained models (DNN on DEAP, CNN on DEAP, DNN on MAHNOB, CNN on MAHNOB) whose results are in the first part of Section 5.

References

[1] Y. Roy, H. Banville, I. Albuquerque, A. Gramfort, T. H. Falk, and J. Faubert, "Deep learning-based electroencephalography analysis: a systematic review," *Journal of neural engineering*, vol. 16, no. 5, p. 051001, 2019.

[2] S. Tripathi, S. Acharya, R. D. Sharma, S. Mittal, and S. Bhattacharya, "Using deep and convolutional neural networks for accurate emotion classification on deap dataset." *Twenty-ninth IAAI conference*, 2017.

[3] S. Koelstra, C. Muhl, M. Soleymani, J.-S. Lee, A. Yazdani, T. Ebrahimi, T. Pun, A. Nijholt, and I. Patras, "Deap: A database for emotion analysis; using physiological signals," *IEEE transactions on affective computing*, vol. 3, no. 1, pp. 18–31, 2011.

[4] M. Soleymani, J. Lichtenauer, T. Pun, and M. Pantic, "A multimodal database for affect recognition and implicit tagging," *IEEE transactions on affective computing*, vol. 3, no. 1, pp. 42–55, 2011.

[5] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural computation*, vol. 10, no. 7, pp. 1895–1923, 1998.

[6] "Deap dataset description." [Online]. Available: http://www.eecs.qmul.ac.uk/mmv/datasets/deap/readme.html

[7] y.-h. Yang and H. Chen, "Machine recognition of music emotion: A review," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 3, 05 2012.

[8] "Mne python library." [Online]. Available: https://mne.tools/stable/index.html

[9] "Biosemi, frequently asked questions." [Online]. Available: https://www.biosemi.com/faq/cms&drl.htm

[10] "Pytorch official website." [Online]. Available: https://pytorch.org/

[11] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.