TARGET IDENTIFICATION WITH DEEP LEARNING FOR SSVEP-BASED BRAIN-COMPUTER INTERFACE SPELLERS

by
OSMAN BERKE GÜNEY

Submitted to the Graduate School of Engineering and Natural Sciences
in partial fulfilment of
the requirements for the degree of Master of Science

Sabancı University
July 2022

TARGET IDENTIFICATION WITH DEEP LEARNING FOR SSVEP-BASED BRAIN-COMPUTER INTERFACE SPELLERS

Approved by:

. . . . . .

. . . . . .

. . . . . .

Date of Approval: 25 July 2022

ABSTRACT

TARGET IDENTIFICATION WITH DEEP LEARNING FOR SSVEP-BASED BRAIN-COMPUTER INTERFACE SPELLERS

OSMAN BERKE GÜNEY

ELECTRONICS ENGINEERING M.S. THESIS, JULY 2022

Thesis Supervisor: Assist. Prof. Dr. Hüseyin Özkan

Keywords: Deep learning, Brain-computer interface, BCI, Steady state visually evoked potentials, SSVEP, Speller, Ensemble, Transfer learning, Unsupervised adaptation

Target identification in brain-computer interface (BCI) spellers refers to the electroencephalogram (EEG) classification for predicting the target character that the user intends to spell. When the visual stimulus of each character is tagged with a distinct frequency, the EEG records steady-state visually evoked potentials (SSVEP) whose spectrum is dominated by the harmonics of that tagging frequency. In this setting, we address the target identification and propose three novel methods, which are a deep neural network (DNN) architecture, an ensemble method, and an unsupervised adaptation of the proposed network. Our DNN is trained exclusively for the information transfer rate (ITR) maximization, in two-stages with user-specific labeled data. The first stage obtains a global model and the second fine-tunes it to the user. The SSVEP signals are processed through convolutions across the sub-bands of harmonics, channels, time, and classified at the fully connected layer. We achieve 265.23 bits/min and 196.59 bits/min ITRs on the publicly available and widely used benchmark and BETA datasets, respectively. These ITRs are the highest ever reported performance results in the literature to date.

Our ensemble method prioritizes the user comfort over the ITR by not requiring any additional data from new users and classifies the instances based on a weighted linear combination of our DNNs trained with previously collected data from several individuals (training subjects). The proposed ensemble method attains 155.51

bits/min ITR on the benchmark dataset and 114.36 bits/min ITR on BETA. In order to achieve both the ITR maximization and user comfort goals, our third unsupervised method pre-trains the DNN with data of the training subjects and then adapts to the new user while utilizing her/his unlabeled data. The adaptation is accomplished by minimizing a new loss function that we also propose, which consists of self-adaptation and local-regularity terms. We observe ITRs of 201.15 bits/min and 145.02 bits/min on the benchmark and BETA datasets, respectively. These results show that both our ensemble and unsupervised methods outperform the state-of-the-art alternative user-comfort-oriented techniques, such as ttf-CCA and OACCA.

ÖZET

GUDHP TABANLI BEYİN-BİLGİSAYAR ARAYÜZÜ HECELETİCİLERİ İÇİN DERİN ÖĞRENME İLE HEDEF TESPİTİ

OSMAN BERKE GÜNEY

ELEKTRONİK MÜHENDİSLİĞİ YÜKSEK LİSANS TEZİ, TEMMUZ 2022

Tez Danışmanı: Dr. Öğr. Üyesi Hüseyin Özkan

Anahtar Kelimeler: Derin öğrenme, Beyin-bilgisayar arayüzü, BBA, Görsel uyarılmış durağan hal potansiyelleri, GUDHP, Heceletici, Topluluk öğrenmesi, Aktarım öğrenmesi, Gözetimsiz uyarlama

Beyin-bilgisayar arayüzü (BBA) heceleticilerinde hedef tanıma, kullanıcının hecelemek istediği hedef karakteri tahmin etmek için elektroensefalogram (EEG) sinyallerinin sınıflandırılmasına karşılık gelmektedir. Her karakterin görsel uyarıcısı farklı bir frekansla etiketlendiğinde, EEG, spektrumu bu etiketleme frekansının harmoniklerinin baskın olduğu görsel uyarılmış durağan hal potansiyelleri (GUDHP) sinyallerini ölçer. Biz bahsedilen bu durumdaki hedef tanımlama problemini ele almaktayız ve bu problem için bir derin sinir ağı (DSA) mimarisi, bir topluluk modeli ve önerilen ağın gözetimsiz bir uyarlaması olan toplam üç adet yenilikçi yöntem önermekteyiz. Kullanıcıya özel veriler kullanılarak iki aşamada eğitilen DSA mimarimiz bilgi transfer hızını (BTH) enbüyütme amacı ile geliştirilmiştir. İlk aşamada global bir model elde edilir ve ikinci aşamada elde edilen modelin kullanıcıya uyarlanması yapılır. GUDHP sinyalleri harmoniklerin alt bantları, kanallar, zaman boyunca konvolüsyonlar yoluyla işlenir ve tam bağlantılı katmanda sınıflandırılır. Bu yöntem ile beraber, herkese açık ve yaygın olarak kullanılan benchmark ve BETA veri setlerinde sırasıyla 265.23 bit/dk ve 196.59 bit/dk BTH değerleri elde etmekteyiz. Bu BTH değerleri literatürde bugüne kadar raporlanan en yüksek BTH değerleridir.

Topluluk yöntemimiz, yeni kullanıcılardan herhangi bir ek veri gerektirmeyerek kullanıcı rahatlığına BTH'ye göre öncelik verir ve verileri, birkaç kişiden önceden toplanmış veriler (katılımcı verileri) kullanılarak eğitilmiş DSA mimarilerimizin

ağırlıklı doğrusal kombinasyonu ile sınıflandırır. Geliştirdiğimiz topluluk modeli, benchmark veri setinde 155.51 bit/dk BTH'ye ve BETA veri setinde 114.36 bit/dk BTH'ye ulaşır. Üçüncü denetimsiz yöntemimiz, hem ITR maksimizasyonu hem de kullanıcı konforu hedeflerine ulaşmak için katılımcı verileriyle DSA mimarisinin ön eğitimini yapar, sonrasında ön eğitimi yapılmış modeli yeni kullanıcının etiketlenmemiş verilerini kullanarak yeni kullanıcıya uyarlar. Üçüncü denetimsiz yöntemimiz BTH enbüyütme ve kullanıcı konforu hedeflerine ulaşmak amacıyla, katılımcı verileri ile önceden eğitilmiş DSA mimarimizi kullanıcının etiketsiz verilerini kullanarak kullanıcıya uyarlar. Uyarlama kendi kendine adaptasyon ve yerel-düzenlilik terimlerinden oluşan yeni geliştirdiğimiz kayıp fonksiyonunu minimize ederek gerçekleştirilir. Bu yöntemimiz ile benchmark ve BETA veri setlerinde sırasıyla 201.15 bit/dk ve 145.02 bit/dk BTH değerlerini gözlemliyoruz. Bu sonuçlar, hem topluluk hem de gözetimsiz yöntemlerimizin, ttf-CCA ve OACCA gibi güncel en iyi alternatif kullanıcı konforu odaklı tekniklerden daha iyi performansa ulaştıklarını göstermektedir.

ACKNOWLEDGEMENTS

First of all, I would like to thank my thesis advisor Dr. Huseyin Ozkan for his support in almost every phase of my academic life. If he did not guide and help me, I could not manage to complete this thesis with this quality.

I would also like to thank my thesis jury members, Dr. Ozgur Ercetin and Dr. Erdem Akagunduz, for their valuable time and participation in my thesis evaluation.

Additionally, I would like to thank Muhtasham, Sami, Kerem, and all my friends, whose names I cannot list here, for their help, friendship, and briefly for everything. Also, I would like to thank Emirhan, Can, Yigit, Deniz, Pelinsu, and Giray for their collaborations. I sincerely appreciate their contributions to this thesis and joint publications.

Also, I would like to thank my father Huseyin, and my mom Nil, for their continuous support.

Lastly, but most importantly, all the praise to God for giving me strength in everything.

This thesis study was supported by The Scientific and Technological Research Council of Turkey (TUBITAK) under Contracts 118E268 and 121E452.

to Eymen

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

Brain signals can be translated to external commands by means of brain-computer interfaces (BCI) [1], which have received increasing attention in recent years [2] and led to substantial progress in many applications such as gaming [3], stroke rehabilitation [4], and cursor control [5]. Another prominent application is the BCI speller [6] that assists patients with severe motor disabilities (e.g. amyotrophic lateral sclerosis), so that they can communicate by spelling via solely brain signals without muscular activity. BCI speller research is recently more focused on the use of steady-state visually evoked potentials (SSVEP) in EEG (electroencephalogram) signals [7, 8] as the SSVEP is of relatively higher signal-to-noise ratio (SNR) than, for instance, the event related potentials (ERP) in P300 spellers [9]. Consequently, BCI SSVEP spellers achieve higher information transfer rate (ITR) with ease of system configuration [10, 11].

The steady-state brain response to a visual stimulus flickering at a certain frequency induces the SSVEP signal that is characteristically dominated in its spectrum by the harmonics of the applied input flickering frequency. This enables the use of SSVEP in BCI speller designs [12]. In the experimental paradigm of BCI SSVEP spellers, a matrix of certain alphanumeric characters, each of which flickers at a unique frequency, is presented on the computer screen (Fig. 1.1), and the subject attends to the character that she/he intends to spell. The goal is to predict (i.e. identify) the intended (i.e. targeted) character based on the received SSVEP signal while managing the trade-off between the prediction accuracy and the signal duration such that the maximum ITR is achieved. Since the frequency spectrum is typically exploited up to almost 100 Hz with the largest harmonic, high temporal precision and at least 200 Hz sampling rate are necessary. Hence, EEG is a popular and appropriate choice for its high speed acquisition with a non-invasive and low cost implementation [13].

We address the target identification in BCI SSVEP speller systems as a multi-class classification problem, and propose novel algorithms for each of the following three scenarios. Firstly, we propose a deep neural network (DNN) architecture to the

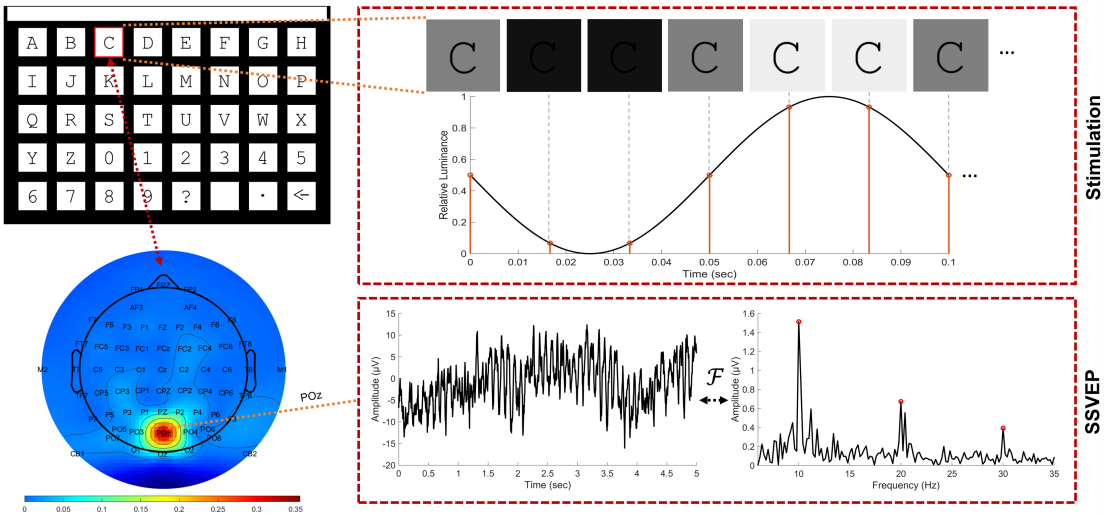


Figure 1.1 A typical system setup of a BCI SSVEP speller is illustrated. A matrix of thumbnail images of certain alphanumeric characters with IDs $j \in \mathcal{M} = \{1, 2, \cdots, M\}$, e.g., $M = 40$, is visually presented to the user on the screen. Each character is contrast-modulated in time by a sinusoid of the assigned unique frequency F_j, e.g., $F_j \in \{8, 8.2, \cdots, 15.8\}$, thereby generating a flickering effect during the T, e.g., $T = 1$, seconds of visual presentation. For example, the character "C" flickers at 10 Hz as illustrated above. If the user wishes to spell a character $y \in \mathcal{M}$ and attends to the corresponding thumbnail, then the steady state brain response (when sensed with EEG from particularly the occipital region, cf. the topographic map representing the user) manifests the multi-channel SSVEP signal $\mathbf{x}$ that is dominated in its spectrum by the harmonics $\{kF_y\}$ of the input frequency F_y, as also illustrated above in the case of $y \equiv$"C". The goal is the target identification for spelling that is to predict the target character y as $\hat{y}$ based on the received multi-channel SSVEP signal $\mathbf{x}$ with C channels, e.g., $C = 9$ or $C = 64$.

goal of ITR maximization. The proposed DNN processes SSVEP signals in time domain as an end-to-end system from the EEG to the prediction of the targeted character, and we train this DNN with two-stages. In the first stage, the DNN is globally trained using all the available SSVEP data, then in the second stage, the globally trained DNN is fine-tuned individually to each person who wants to use the system separately, using the labeled data of the corresponding person. Our proposed DNN architecture significantly outperforms the state-of-the-art as well as the most recently proposed techniques in the literature. We achieve impressive ITRs with only 0.4 seconds of stimulation, 265.23 bits/min and 196.59 bits/min, on the two publicly available large scale benchmark [14] and BETA [15] datasets that consist of the EEG data of 105 subjects with 40 target characters. To our best knowledge, these are the highest ever ITRs reported on these datasets.

Although our proposed DNN is the best performing method, its training requires labeled data from each individual who wants to use the system, performance is not the only criterion that needs to be considered. There are also other factors,

such as users' comfort, that one should take into account for convenient daily-life use. Because of the labeled data requirement, to be able to use our proposed DNN for each person who wants to use the system as the first time, additional EEG experiments must be conducted for data collection, which is typically extensive and burdensome. To remove the hassle of long and tiring data collection periods, we have also proposed an ensemble-based classification method. Our ensemble-based method uses transfer and ensemble learning approaches such that models that have been previously trained on different subjects[1] provide the ensemble and are transferred to the new user. The target characters are then decoded based on a weighted combination of the predictions of k many selected models from the ensemble. As the model, we have used our proposed DNN architecture, since it significantly outperforms the state-of-the-art methods with the highest ITR, when the user-specific training is used. We tested the performance of our ensemble-based method on same datasets (i.e., the benchmark [14] and the BETA [15] datasets). Our ensemble-based method achieves *155.51 bits/min* on the benchmark dataset, and *114.36 bits/min* on the BETA dataset. To the best of our knowledge, among the methods that do not require the user specific data from each new user, and do not utilize any sort of user-specific adaptation, these ITR values are the highest ever reported performance results on these datasets.

Even though our ensemble-based method is the best performing method within its category (i.e., among the methods that do not utilize any user-specific adaptation), its performance is far behind the performance of our proposed DNN trained using user specific labeled data. This fact actually shows that the user-specific adaptation is in fact a must. However, as discussed, the users' comforts must also be considered. To be able to satisfy both requirements (i.e., the performance and the users' comfort), we have proposed another method that firstly transfers a global DNN model that is trained using the data of all available subjects to the new user. Then, our method adapts the global DNN model to the user in an unsupervised fashion by utilizing the user's collected unlabeled data, which are accumulated as the user uses the system. For the DNN model, again we have used our proposed DNN architecture. In the adaptation phase, our proposed novel custom loss function that consists of the self-adaptation and local-regularity terms is minimized. The performance evaluation of this method, is conducted on the same datasets. Our adaptation method achieves *201.15 bits/min* on the benchmark dataset, and *145.02 bits/min* on the BETA dataset. The results show that to the best our knowledge, our adaptation method similar with our other methods is the best performing method within its

[1] In this thesis, we use the words "user" and "subject" interchangeably but with a slight difference in the meaning. "User" typically refers to the end-user of a BCI system whereas "subject" refers to the participants of an EEG experiment conducted for data collection and development purposes.

category.

All of our proposed methods can be straightforwardly extended (as it is not specific to spellers) to general BCI systems for the broader purpose of translating brain signals to external commands. Therefore, we believe that our techniques will produce a great impact and immensely valuable use in a plethora of real-life applications of SSVEP-based BCIs such as rehabilitation, control, and gaming.

The rest of the thesis is organized as follows. In Chapter 2, related methods in the literature that were developed for the identification problem in SSVEP-based BCIs are discussed. In Chapter 3, we provide problem statement. In Chapter 4, a description of both the benchmark and BETA datasets is presented. The details of our proposed DNN architecture, and its experimental results on the used datasets are in Chapter 5. Chapter 6 presents our ensemble based method and its experimental results. In Chapter 7, we give the explanations and details of our loss function proposed for the unsupervised adaptation as well as its performance evaluation results on the datasets. We conclude the thesis in Chapter 8 with a final discussion.

2. RELATED WORK

In this chapter, we discuss the prominent target identification methods developed for the SSVEP-based BCI speller systems. These methods can be grouped into four different categories based on utilized training strategies and data availabilities. These categories are: **1)** both transfer and training free algorithms, **2)** algorithms with supervised training, **3)** training free and transferred algorithms, and **4)** algorithms with unsupervised training. The expression "transferred" indicates that there exist some data collected previously from some subjects with the EEG experiments, and the models trained using these data are transferred to a new user. The expression "trained" indicates that the algorithm is adapted for the new user, and the expression "supervised" indicates that this training/adaptation requires label information. In the following, the methods in each category will be explained under separate subsections.

2.1 Both Transfer and Training Free Algorithms

The methods from the "both transfer and training free algorithms" category rely on mathematical formulations/models of the SSVEP signals, hence they do not need any data. Because these methods are free of any data requirement, they are practical and they can be used directly in new applications of the SSVEP based BCIs, where the previously collected data either is limited or not available or collecting the labeled data from the user is not feasible. However, their performances (i.e., ITR and target identification accuracy) are generally much worse than the algorithms from the other categories.

One of the conventional target character identification methods from this category is based on the power spectrum density analysis (PSDA) of the received SSVEP signal [16], in which the SNRs of the components of the stimulus frequencies are calculated

and then the frequency of the highest SNR is selected as the final prediction. The minimum energy combination (MEC) method [17] linearly combines the SSVEP signals from multiple EEG channels to enhance the identification performance by minimizing the energy of the undesired SSVEP component. Another method is the canonical correlation analysis (CCA) (we call this method as "Standard-CCA" throughout this thesis) in [18], which measures the maximal correlation between the SSVEP signal (of the optimal channel combination yielding that maximum) and the reference of a flickering frequency of interest (of the optimal harmonics combination yielding that maximum). Then, the frequency of the largest maximal correlation is selected as the final prediction. Standard-CCA generally demonstrates better ITR performance than PSDA and MEC methods [18, 19].

As an improved extension of Standard-CCA, a method called filter bank canonical correlation analysis (FBCCA), which is the best performing method in this category, was proposed in [20]. In FBCCA, the Standard-CCA algorithm is run in parallel on the multiple SSVEP signals obtained by applying a filter-bank (multiple band-pass filters) first and the results are combined afterwards. The reason of FBCCA performance improvement over Standard-CCA is that the filter-bank approach evaluates the contribution (to the identification) of each harmonic degree separately by using various sub-bands in the spectrum. This is supported in [21] by that, as the degree increases, the harmonic magnitude drops but the SNR does not necessarily decrease since the noise reduces faster. We refer to the seventh figure of [21] for a demonstration, where it is shown that the harmonics up to 50 Hz maintains a relatively high SNR. We also observe this -by inspection- in our own signal analysis (cf. the spectrum example provided in Fig. 1.1 in the case of the harmonics of 10 Hz up to the 3rd degree). The filter-bank approach has become a standard procedure thereafter and many researchers have followed by utilizing it to increase the performance [7, 8, 11, 22].

2.2 Algorithms with Supervised Training

EEG signals are well-known to exhibit data statistics that can drastically change from one person to another [23, 24], because of this fact, incorporating with individual labeled data provides a significant ITR performance improvement [25]. The methods in this category requiring the labeled data from each user, show much better accuracy and ITR results than the methods in other categories. Therefore,

many extensions of the Standard-CCA that incorporate with the individual data, such as L1-MCCA [26], ITCCA [27], MwayCCA [28], MsetCCA [29], PCCA [30], CACC [31], as well as a combination of Standard-CCA and the ITCCA method yielding Extended-CCA [25, 32] are proposed to improve the performance of the Standard-CCA. Among those methods, Extended-CCA and its improved version m-Extended-CCA are reported to outperform the others [11, 33].

The correlated component analysis (CORRCA) [8] maximizes the correlation between the multi-channel template signals (which are calculated by averaging the SSVEP signals across multiple trials in the training set for each frequency) and the multi-channel test signal, and then the frequency of the highest correlation yields the final prediction[1] [8]. The maximization in CORRCA [8] is with respect to a single projection across channels, whereas the maximization in Standard-CCA [18] is with respect to two projections one of which is across channels and the other is across harmonics in the references. As for the several extensions of CORRCA, the filter bank approach is used in FBCORRCA [8], the information from other correlation coefficients is exploited via carefully fusing them with exponentially decaying weights in HFCORRCA [34], and spatial filters of all stimulus frequencies are utilized in TSCORRCA [8] yielding the best performing extension.

A method called task related component analysis (TRCA) is used for BCI SSVEP spellers in [7]. The formulation models the SSVEP signal as a task-related information signal that is linearly contaminated with noise. It is shown in [7] that TRCA, when used for suppressing the noise in SSVEP by maximizing the inter-trial covariance, delivers higher ITR performance than the Extended-CCA method. TRCA can be enhanced by the filter bank approach along with spatial filters yielding the Ensemble-TRCA (eTRCA) technique [7]. A multi stimulus scheme (ms-eTRCA) is further incorporated in [22] which is an advancement over the methods Extended-CCA and eTRCA.

There exist a few deep learning studies that are related to SSVEP signal classification and BCI spellers [35–41]. These studies aim to improve the current state with the joint learning of temporal and spatial EEG features via deep neural networks. The joint feature learning not only generates high level representations through cascaded layers but also helps to alleviate the need for a separate preprocessing step. In addition, DNNs allow the inference of nonlinear interactions between such features and the stimulus decoding, which is typically not explored in the conventional techniques.

[1] Since each character corresponds by design to a unique frequency, we use the phrases "target character" or "frequency" and "identification" or "prediction" or "classification" interchangeably depending on the context.

A convolutional neural network (CNN) is designed in [35] to suppress the non-task-related signals in an ambulatory context of SSVEP signal classification. Their network of three layers of multiple feature maps processes the data in the frequency domain and yields a better identification performance than the CCA based methods. The CNN of [36] is composed of temporal and spatial processing layers that are followed by pooling and fully connected layers (FC). It performs favorably compared to the baselines of linear discriminant analysis and random forest in the case of hand movement classification with low frequency EEG (non-SSVEP). A recurrent neural network and a CNN are compared in [37] against various traditional approaches such as k-nearest neighbor classification, adaboost, decision trees and SVM (together with feature selection), where the CNN (a single convolutional layer, pooling and FC) has been concluded to outperform. The networks of [37] learns higher level representations starting from power spectral density based EEG features. In contrast, the proposed CNN (a single convolutional layer followed by pooling, batch normalization and FC) in [38] is an end-to-end system (input is raw signal) without preprocessing, and shown to perform better than the approach of [37] for particularly the dry EEG. The idea of fine tuning the pre-trained model with transfer learning for subject specific adaptation has been observed in [40] to largely improve the identification performance. One further conclusion in [40] is that their CNN outperforms the conventional approaches such as CCA, FBCCA and TRCA. Another DNN (Conv-CA) is designed in [41] for the speller application and reported to deliver a better target identification performance than the method eTRCA [7].

2.3 Training Free and Transferred Algorithms

The algorithms in this category transfer the models or the template signals obtained based on the available/existing subjects' data to the new user. Because these algorithms do not require any user data, they are free of the hassle of long and tiring data collection periods. Hence, the methods in this category are much more practical than the methods in "Algorithms with Supervised Training", and since they are data-driven approaches, their performances are superior than the performances of algorithms from "Both Transfer and Training Free Algorithms".

One of the earliest proposed methods from this category is tt-CCA [42], in which the target character is predicted based on the correlation coefficients between the received test signal and the template signals that are formed in the train-

ing/construction phase of the method. For each target character, all of the training subjects' data (having the same label with the corresponding target character) are averaged to form a template signal. Then, CCA is applied between this template signal and the synthetic reference signal (formed with the frequency and its harmonics of the target) so that the channel combination giving the maximum correlation is selected as a channel. After completing this for all target characters, the signals and channel combinations are transferred to the new user. At the stage of predicting the intended character of the new user instance, tt-CCA calculates and combines three correlation coefficients for each target character, and the one having the maximum combined correlation coefficient is selected as the prediction. The first correlation coefficient is the maximum CCA coefficient between the new instance and the synthetic template signal. The channel combination giving this maximum CCA coefficient combines channels of the new instance and the template signal, and then the correlation coefficient between them is calculated as the second. For the third coefficient, the same procedure is applied, but with the transferred channel combination.

Combined-tCCA [43] follows a similar approach with the tt-CCA method, except that it uses different correlation coefficients that are defined in [25] for the supervised setting. Rather than employing user-specific coefficients, those of [25] are modified in [43] by only using transferred signals and channel combinations.

In the ttf-CCA [44] method, channel combinations are learned for each subject separately using individual templates. After reducing all such subject-specific channel combinations to a certain number of common combinations, for each target character, correlation coefficients between the new instance and the corresponding template signal (constructed with the same way in the tt-CCA method) are calculated. The maximum of these coefficients as well as the one between the new insance and the related synthetic template signal reveals the final prediction.

2.4 Algorithms with Unsupervised Training

The user-independent methods (i.e., methods from "Both Transfer and Training Free Algorithms" and "Training Free and Transferred Algorithms" sections) are free of preparation and so appealing in the sense of the user comfort, but they underperform on the ITR side as the EEG statistics vary inter subjects largely [23, 24]. This

shows that user-specific adaptation is in fact a must. However, the user comfort is also another important point that needs to be considered. The methods, such as online tt-CCA (ott-CCA) [42], adaptive combined-tCCA (adaptive-C3A), and online adaptive CCA (OACCA), in this group aim to satisfy both the user comfort and the detection performance with using or transferring a user-independent method and adapting it in an unsupervised fashion.

The ott-CCA method [42] is the online unsupervised extension of the tt-CCA method, where the template signals used in the tt-CCA method are updated using the unlabeled new user data. It has been observed that ott-CCA increases the performance of the tt-CCA method when the signal length is more than 1 second [42]. A very similar approach is followed by the adaptive-C3A method, which is the online extension of combined-tCCA [43]. Adaptive-C3A updates the used template signals in combined-tCCA in an unsupervised manner. It is reported that the adaptive-C3A method improves the performance of the combined-tCCA method [43].

One recent method called online adaptive CCA (OACCA) [45] follows a bit different approach than the ott-CCA and adaptive-C3A methods. The OACCA method does not adapt the transfer-based method instead it adapts the CCA method to the new user by introducing the new user-specific channel and harmonic combinations. The new user-specific channel and harmonic combinations are found by the methods introduced in [22] and [46] using the calculated combinations for the previous trials.

3. PROBLEM DESCRIPTION

During a trial in a BCI SSVEP speller session (illustrated in Figure 1.1), the subject is visually presented a matrix of M alphanumeric characters each of which flickers at unique frequency $F_j : j \in \mathcal{M} = \{1, 2, \cdots, M\}$ (in Hz), e.g., $F_j \in \{8, 8.2, \cdots, 15.8\}$ with $M = 40$. Then, she/he is asked to concentrate on the target character with the identification number $y \in \mathcal{M}$ that is to be spelled. The brain response, as a result of the stimulation by the intended target character y flickering at the frequency F_y, is measured with EEG as the multi-channel SSVEP signal $\mathbf{x} \in \mathbb{R}^{C \times N_t}$. Here, C is the number of channels and $N_t = T \times F_s$ is the number of samples in each channel (with T and F_s being the signal or stimulation duration in seconds and the sampling frequency in Hz, respectively). The measured SSVEP signal $\mathbf{x}$ mostly comprises of the frequency components $A_F \cos(2\pi F t + \phi_f)$ (where $t = n/F_s$ due to sampling) at the harmonics $F = kF_y$ (integer k) of the stimulation frequency F_y. The entire spectrum (up to typically 100 Hz as far as the information content, which is corrupted by the noise and interfered with other ongoing processes in the brain, is concerned) is spanned, but the components of the harmonics are larger, i.e., $A_{kF_y} >> A_F > 0$ for $F \neq kF_y$ [11, 12] (cf. also the spectrum example in Fig. 1.1). Then the target identification problem in this setting can be perhaps solved by the detection of the peaks across harmonics up to a certain degree in the Fourier spectrum of the SSVEP signal. Namely, one can decide for the character whose harmonics are most covered by the spectrum. However, the harmonics are generally not observable in the spectrum as orthogonal components since the signal duration T yields only a low frequency resolution $\delta\hat{w} = \frac{2\pi}{TF_s}$ rad in normalized radian frequency and $\delta F = \frac{\delta\hat{w}}{2\pi} F_s = \frac{1}{T}$ Hz in cyclic frequency (where T is short). Therefore, the information in the harmonics of $A_{k\delta F} \cos(2\pi k\delta F t + \phi_{k\delta F})$ (where $t = n/F_s$ due to sampling) do leak onto the entire Fourier spectrum of the SSVEP signal due to the correlation between the harmonics and the spectrum components. If one still insists on using Fourier spectrum based decoding, then at least 5 seconds of stimulation duration (i.e., $T = 5$) is required in the case of a total of 40 characters that flicker at unique frequencies with 0.2 Hz increments. However, please note that a BCI SSVEP speller system is typically designed for enabling a severely motor disabled individual

to communicate flawlessly at a fast rate which requires a high speed accurate speller. Therefore, the main design goal is to maximize the ITR [47] that is a function of the target identification accuracy and the stimulation duration. If the prediction is perfectly accurate, then the trial-by-trial spelling of a length-l word requires $T \times l$ seconds which is equivalent to the ITR $\log_2 M \frac{60}{T}$ bits/min, i.e.,

$$\begin{aligned} \mathrm{ITR}(P,T) &= (\log_2 M + P\log_2 P + (1-P)\log_2\left[\frac{1-P}{M-1}\right])\frac{60}{T} \\ &= (\log_2 M)\frac{60}{T} \text{ (when } P=1) \text{ .} \end{aligned} \tag{3.1}$$

Note that the prediction accuracy $0 \leq P \leq 1$ is almost never perfect; nevertheless, if the identification method is optimal (with the minimum possible error rate, i.e., $1-P$), then the P can be improved only by requesting a longer stimulation via increasing T (resulting in a larger amount of data). However, in this case of lengthening the stimulation duration, the trials of the spelling slow down and consequently the ITR does not necessarily improve. For example, the long stimulation $T = \infty$, results in the perfect accuracy $P = 1$ that is, though, 0 ITR. Hence, when the identification method is optimal, it is not possible to expedite the spelling while also improving the P since the two are incompatible. This requires to manage a trade-off between P and T for the ITR maximization. On the other hand, when the target identification is itself not optimal, improving the P is possible without increasing the T up to the point where the trade-off starts dominating. The ITR maximization for a fixed T is equivalent to accuracy maximization, our strategy is to minimize the $1-P$ for each T, and observe the pair (P^*, T^*) that yields the maximum ITR. In addition to ITR maximization, the user comfort also needs to be considered for the convenient daily use.

In this respect, we formulate the character identification as a multi-class classification problem based on the available data. There are three different problem setups based on the available data types, which are "Supervised Training Setup", "Training Free and Transferred Setup", and "Unsupervised Training Setup". In each problem setting, different classification goals are prioritized because of their nature, as explained in the following subsections.

3.1 Supervised Training Setup

In the supervised training setup, there are D_u many available labeled data of the user for training a user-specific model: $\{(\mathbf{x}_i, y_i)\}_{i=1}^{D_u}$. And there is a set of labeled data from a set of subjects $\{\{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^{D_s}\}_{s=1}^{N_{tr}}$, where N_{tr} is the total number training subjects, and D_s is the total number of the EEG instances from the s'th subject. These available subjects' data are used to ease and improve the training of the user-specific model. The models trained in this problem setup generally show very promising ITR results at the expense of a tiring data collection period as detailed in Chapter 2. We propose a DNN architecture (Chapter 5) that is trained and tested in this setting.

3.2 Training Free and Transferred Setup

In this problem setup, we do not have any available data from the user, but we have the data of some other existing subjects: $\{\{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^{D_s}\}_{s=1}^{N_{tr}}$. The models are generated using the subjects' data and then directly transferred to the new user. As there is not any data of the user, the methods trained in this setting are much more convenient for the user, but they underperform on the ITR side. We propose an ensemble based method that is trained and tested in this setting.

3.3 Unsupervised Training Setup

In this setting, there is not any available labeled data of the user, but we have some unlabeled data of the user, which is accumulated as the new user uses the system: $\{(\mathbf{x}_i)\}_{i=1}^{D_u}$. These unlabeled data are used to adapt a model that is generated using the available data of the subjects: $\{\{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^{D_s}\}_{s=1}^{N_{tr}}$. In this problem setup, both the user comfort and ITR performance are tried to be satisfied. Since there is not any labeled data, the long and tiring EEG data collection experiments are not conducted; and the ITR performance is considered with adapting the methods using the accumulated unlabeled data of the user. We introduce a new loss function that allows adaptation in this unsupervised setting.

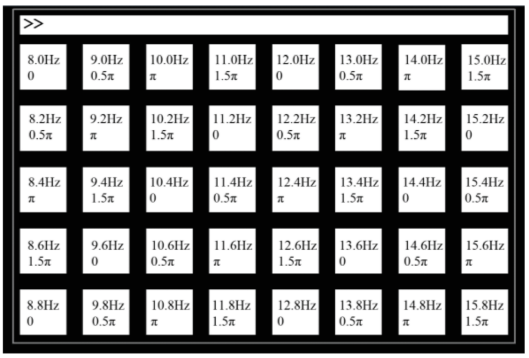
4. DATASETS

To test the performance of all the our proposed methods, we have used the publicly available and widely used two large scale datasets, which are the benchmark [14] and the BETA [15] datasets. The details of the each dataset are presented in the following.

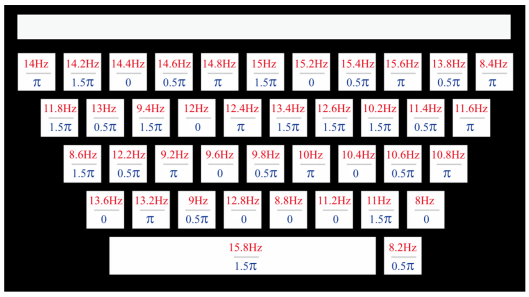
4.1 The Benchmark Dataset

The benchmark dataset has been recorded in BCI SSVEP speller experiments with 35 healthy subjects. Each experiment consists of 6 blocks, i.e., sessions. The reason of collecting the data with several blocks is to provide a break time to the subjects. During a block, the subject is shown on the screen[1] (Figure 4.1a) a matrix (5×8) of 40 target characters flickering at various frequencies (in the range $8 - 15.8$ Hz with 0.2 Hz increments) with at least 0.5π phase difference between adjacent frequencies. The EEG data is recorded through 64 channels. Each block includes 40 random-order trials (one trial per each target character). Each trial starts with a visual cue that is displayed for 0.5 seconds on the screen to guide subject's gaze to the desired target, and then conducts the stimulation for 5 seconds that is followed by an offset of 0.5 seconds. The EEG is downsampled to 250 Hz. Average visual latency of the subjects is approximately estimated as 140 ms in this dataset. We refer to [14] for further details.

[1] Figure 4.1a is taken from [14].



(a) The character matrix layout for the stimulus presentation in the experiments of the benchmark dataset is shown.



(b) The character matrix layout for the stimulus presentation in the experiments of the BETA dataset is shown.

4.2 The BETA Dataset

The BETA dataset and the benchmark dataset are similar, but also have certain important differences. We note the differences in the following (the remaining attributes are the same). This BETA dataset has been recorded with 70 healthy subjects. Each experiment consists of 4 blocks. The flickering target characters are shown in the form of a keyboard[2] (Figure 4.1b). The experiments are conducted outside of the laboratory environment, resulting in a lower SNR compared to the benchmark dataset. Hence, the target identification is more challenging in this case. The stimulation lasts 2 seconds for the first 15 subjects and 3 seconds for the remaining subjects. Average visual latency of the subjects is approximately estimated as 130 ms in this dataset. We refer to [15] for further details.

[2]Figure 4.1b is taken from [15].

5. PROPOSED SOLUTION IN THE SUPERVISED TRAINING SETTING: A DNN ARCHITECTURE

In this chapter, we present our proposed DNN architecture, which processes SSVEP signals in time domain as an end-to-end system from the EEG to the prediction of the target character. The proposed DNN strongly outperforms (with uniformly the highest ITR results for all signal lengths) the state-of-the-art as well as recently proposed deep learning techniques. It consists of 5 layers, each layer has a specific purpose. In the following, we detail the layers of the proposed DNN and discuss its performance evaluation results on the benchmark [14] and the BETA datasets. The text in this chapter and Fig. 1.1 and 5.1 are mainly from our publication about our DNN architecture [48].

5.1 The Proposed DNN Architecture

Our DNN architecture operates as an end-to-end system which receives the multi-channel SSVEP signal $\mathbf{x}$ and processes it in a feed-forward manner to the final prediction $\hat{y}$. The proposed DNN (Figure 5.1) consists of 4 convolutional layers and 1 fully connected layer. Hence, we have the processing $\mathbf{x} \rightarrow$ preprocessing: $[\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(r)}, \ldots, \mathbf{x}^{(N_s)}] \rightarrow \underline{\mathbf{x}} \rightarrow \mathbf{z}_1 \rightarrow \mathbf{z}_2 \rightarrow \mathbf{z}_3 \rightarrow \mathbf{s} \rightarrow \hat{y} = \arg\max \mathbf{s}_j$, where the preprocessing is for generating the sub-bands of harmonics $[\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N_s)}]$ that are combined in the first layer to produce the $\underline{\mathbf{x}}$ which is processed in the second layer for spatial filtering to produce the $\mathbf{z}_1$. Here, N_s is the number of sub-bands and r is the corresponding index. Downsampling follows in the third layer, yielding $\mathbf{z}_2$, then features are extracted in the fourth layer as $\mathbf{z}_3$ passing to the classification in the fully connected layer to produce the prediction $\hat{y} = \arg\max \mathbf{s}_j$ ($\mathbf{s} \in [0,1]^{M \times 1}$ is the softmax output).

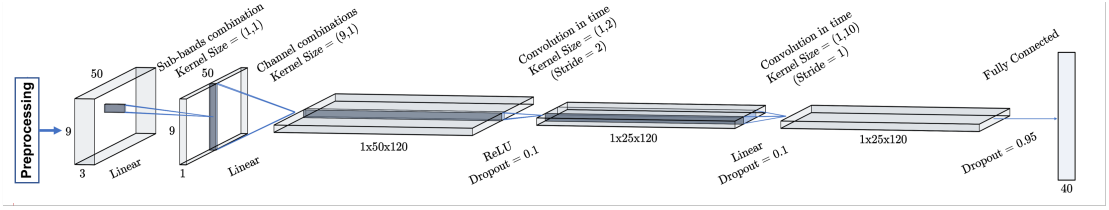


Figure 5.1 We propose a DNN architecture (with 4 convolutional and 1 fully connected layers) for the target identification problem in SSVEP-based BCIs. The proposed DNN strongly outperforms the state-of-the-art and recently proposed techniques uniformly across all signal durations $T \in \{0.2, 0.3, \cdots, 1.0\}$, but in particular delivers impressive information transfer rate (ITR) results that are 265.23 bits/min and 196.59 bits/min in as short as $T = 0.4$ seconds of stimulation with $C = 64$ channels on the two publicly available benchmark and BETA datasets. *To the best of our knowledge, our ITRs are the highest ever reported performance results on these datasets.*

Remark: We point out that since there is only one nonlinear activation (ReLu) in the proposed DNN, it can be reduced to a single hidden layer network with the ReLU activation at the hidden layer. However, that would only lead to a non-intuitive complicated network. In our DNN, the information flow is natural through an intuitive and conceptually simple design. Therefore, we present the DNN as the composition of 5 functionalities, i.e., layers.

In the following, we describe our proposed DNN. For each layer, we first motivate its use and then provide its definition. Next, the training scheme and the further details are explained.

5.1.1 First layer for harmonics (sub-bands) combination

Under the stimulation by the target character flickering at the frequency F_y, the contributions of the harmonics to the generation of the SSVEP signal might vary from one harmonic to another. For example, a lower order harmonic generally has a larger magnitude compared to a higher degree one [11]. Nevertheless, since the higher order harmonics tend to be less (for example, compared to the alpha band around 10 Hz) interfered with other ongoing brain activities, they tend to manifest perhaps surprisingly a relatively high SNR [21] (as we also observe by inspection in the spectrum example of Fig. 1.1 in the case of the stimulation frequency 10 Hz up to the 3rd degree). We also refer to the study [21] for a general SNR investigation of SSVEP harmonics. However, it is not straightforward to assess which harmonic is more informative in the SSVEP classification, and hence how to normalize in

the spectrum across the harmonics could be a fairly difficult task. This issue is handled in the literature by processing several sub-bands of the SSVEP spectrum separately, but then the results are fused in a rule based manner or are fused based on a fairly restrictive model, e.g., [11, 20]. Therefore, how to choose the weight of a certain harmonic is not sufficiently addressed in the literature due to their manual handling.

In our DNN design, we opt to stay agnostic about this normalization of harmonics and instead let the network decide about the normalization weights by training in a data driven manner. For this purpose, we band-pass filter (denoted by $\mathcal{G}_r$, with MATLAB filtfilt function) the SSVEP signal $\mathbf{x} \in \mathbb{R}^{C \times N}$ in each channel (multiple times $1 \leq r \leq N_s$), where the lower cut-off is $r \times \min\{F_j\} - \epsilon$ Hz (e.g., ~ 8 Hz for $r = 1$ in both of the datasets [14, 15]) and the upper cut-off is $6 \times \max\{f_j\} + \epsilon$ Hz (e.g., ~ 90 Hz in both of the datasets [14,15]) with ϵ being a small margin. The filter is designed as (using MATLAB designfilt function) zero-phase Chebyshev-Type 1 with filter order 2 and 1 dB pass band ripple. Hence, each filter $\mathcal{G}_r$ excludes the harmonics of the degree that is less than r while including the rest up to the 6'th degree (the maximum degree is set to 6 since beyond 100 Hz in the EEG is typically noise in BCIs). This yields the filtered output $\mathbf{x}^{(r)} \in \mathbb{R}^{C \times N}$ that includes a specific sub-band of harmonics.

The first layer of our DNN (with the weights $\mathbf{w}_s \in \mathbb{R}^{N_s \times 1}$) linearly combine these sub-bands for a normalization across the harmonics as $\underline{\mathbf{x}} = \sum_{s=1}^{N_s} \mathbf{w}_s^{(s)} \mathbf{x}^{(s)}$, where the input to the layer is $[\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(r)}, \ldots, \mathbf{x}^{(N_s)}] \in \mathbb{R}^{C \times N_t \times N_s}$ (i.e. a volume of $9 \times 50 \times 3$ in the case of $C = 9$, $N_t = 50 = T \times F_s$ with $T = 0.2$ seconds, $F_s = 250$ Hz, and $N_s = 3$) whereas the output is $\underline{\mathbf{x}} \in \mathbb{R}^{C \times N_t}$ (i.e., a plane of $9 \times 50 \times 1$ when $C = 9$ and $N_t = 50$). Hence, (if desired) our DNN has the capability to amplify the higher order harmonics by choosing the corresponding weights relatively high.

5.1.2 Second layer for channel combination

The SSVEP signal is a multi-channel signal. The channels, on the one hand, bear valuable distinct information from the brain regions that they sense from but, on the other, produce signals that are also largely correlated. A combination across channels shall be considered to extract and accumulate the whole information while discarding the redundancy or non-informative variations. Also, multiple combinations are probably needed since extracting the information living in one subspace (of the complete channel space) can suppress the one living in another subspace.

The required combinations might be even more than the number of channels as those informative subspaces are not necessarily orthogonal, requiring in return a nonlinear processing of combinations. The existing CCA analyses in the literature (such as [8, 11]) allows a separate channel combination for each and every single test instance. Here, we criticise this since it not only A) creates an unnecessarily large degree of freedom and in return a large detrimental effect due to the induced strong proneness to overfitting, but also B) risks suppressing, in each case of the test instances, valuable information that can be extracted by other but not utilized combinations. To alleviate the issue B, those techniques incorporate multiple combinations -for each and every testing again- by fusing the correlation coefficients of the CCA analysis, but this further worsens the issue A. Even then, the number of combinations is limited by the number of channels due to linearity, and the coefficient fusing is typically rule-based without a data-driven learning or is based on a simple fitting to a rather restrictive model. Further, a regularization step is generally not incorporated though it is certainly needed.

In the following, we explain from another perspective to motivate our approach. Since the neural circuitry and nonlinear processes that are involved in brain to generate the SSVEP responses vary from lower degree harmonics (as well as intermodulations) to upper degree ones [49], we certainly expect that different brain regions are more responsive to different harmonic degrees which necessitates the use of multiple channel combinations. Moreover, a different combination might be more appropriate to emphasize a certain stimulation frequency and its harmonics while suppressing the others which further necessitates multiple combinations for each classes.

Unlike the state-of-the-art methods, in our DNN design, we use the same set of multiple channel combinations that is common for all of the instances. This set in our study includes as many combinations as the number N_s of sub-bands for each stimulation frequency F_j, yielding in total, for instance, $N_{ch} = 120 = N_s \times M$ combinations when $N_s = 3$ and the number of characters is $M = 40$. We emphasize that if the number N_{ch} of channel combinations is more than the number C of channels, then one needs nonlinear processing (to avoid degeneracy and) to make use of the combinations effectively. Overall, this setting keeps the parameter complexity at a manageable level and mitigate the overfitting when compared to using a separate combination for each and every single test instance as in the existing techniques of literature. At the same time, our setting is also sufficiently powerful since we can use combinations as many as needed. To this end, the second layer (parameterized over the weights $\mathbf{w}_c \in \mathbb{R}^{C \times N_{ch}}$) of our DNN combines the channels by receiving the input plane $\underline{\mathbf{x}} \in \mathbb{R}^{C \times N_t}$ and returning the plane $\mathbf{z}_1 \in \mathbb{R}^{N_t \times N_{ch}}$, i.e., $\mathbf{z}_1 = \underline{\mathbf{x}}'\mathbf{w}_c$, where

$\underline{\mathbf{x}}'$ is the transpose of $\underline{\mathbf{x}}$. In order to achieve nonlinearity, we also apply the ReLU (rectified linear unit) activation but postpone it until the end of the third layer.

5.1.3 Third layer for filtering in time, downsampling and nonlinearity

This layer has two functions. First, it applies a filter of size 2 in time (with also a full third dimension along the depth) with stride 2, thereby halving the dimension (downsampling by 2) and reducing the parameter complexity in the network. This operation can be considered to represent the anti-aliasing filtering that is commonly used with downsampling. The filtering in this layer additionally serves for roughly adjusting the spectral bandwidth for each information flow over the channel combinations in the network. Hence, multiple such filters (as many as the number of channel combinations, i.e. $N_{ch} = 120$) are used. Second function of this layer is applying the nonlinearity. Note that when we have $N_{ch} > C$, the input plane of this third layer $\mathbf{z}_1 \in \mathbb{R}^{N_t \times N_{ch}}$ is rank-deficient with a rank at most C even if $\underline{\mathbf{x}} \in \mathbb{R}^{C \times N_t}$ (producing $\mathbf{z}_1 = \underline{\mathbf{x}}'\mathbf{w}_c$) is full rank. This defeats the purpose of producing multiple channels combinations in the previous layer. Hence, to tackle the rank-deficiency and enable the effective use of the channel combinations, the nonlinear ReLU activation is applied after downsampling to produce the output plane $\mathbf{z}_2 \in \mathbb{R}^{(N_t/2) \times N_{ch}}$.

5.1.4 Convolutional fourth and fully connected fifth layers

The fourth convolutional layer filters the input $\mathbf{z}_2$ with multiple finite impulse response filters (FIRs, each being of length 10 with also a full third dimension along the depth) to produce the features in $\mathbf{z}_3$ that is finally classified by the following fully connected (FC) layer. Hence, the very first input $\mathbf{x}$ is predicted as $\hat{y} = \arg\max \mathbf{s}_j$ ($\mathbf{s} \in [0,1]^{M \times 1}$ is the softmax output of the FC layer). The FIR filters in the fourth layer are expected to achieve frequency responses that are tuned to the spectral patterns of each stimulation class (1 FIR for each sub-band per each $M = 40$ classes, yielding in total 120 filters when we have $N_s = 3$ sub-bands) for extracting powerful features. Hence, in these two layers, all of the FIR filters as well as all of the FC weights are optimized.

5.1.5 Two-staged training and further details

The proposed DNN is initialized by sampling the network weights from the Gaussian distribution with 0 mean and 0.01 standard deviation, except that all of the weights in the first layer are initialized with 1's. The exception of the first layer is due to an intuitive choice for assigning equal weights to the sub-bands initially without affecting the order of magnitudes of the input filtered signals. We train the network in each iteration based on the training batch data $\{(\mathbf{x}_i, y_i)\}_{i=1}^{D_b}$, where D_b is the number of trials in the batch, by minimizing the categorical cross entropy loss

$$\frac{1}{D_b}\sum_{i=1}^{D_b} -\log(\mathbf{s}_{i,y_i}) + \lambda|\mathbf{w}|^2 \tag{5.1}$$

via the Adam optimizer [50] with the learning rate $\nu = 0.0001$ (without decaying), where λ is the constant of the L2 regularization which we set as $\lambda = 0.001$, $s_i \in [0,1]^{M\times 1}$ is the softmax output for the instance $\mathbf{x}_i$, $\mathbf{s}_{i,y_i}$ is the y_i'th entry of $\mathbf{s}_i$ and the final prediction is $\hat{y}_i = \arg\max \mathbf{s}_{i,j}$. Here, $\mathbf{w}$ represents all the DNN weights. Dropout layers are incorporated between the second and third, third and fourth, and fourth and fifth layers with dropout probabilities 0.1, 0.1, and 0.95, respectively.

We also point out that the total number of trainable parameters in our proposed DNN can be found by $N_s + CN_{ch} + 2N_{ch}^2 + 10N_{ch}^2 + \frac{N}{2}N_{ch}M$ (each term is for a layer including the output layer), which yields $413,883$ parameters in the plausible setting of $N_s = 3, C = 9, N_{ch} = 120, T = 0.4$ sec, $f_s = 250$ Hz with $M = 40$. Since there are at most 8400 training instances in the considered datasets, which is low compared to the parameter complexity, we opt for a relative strong regularization by using a large dropout probability (0.95) in the last layer.

We train the network in two stages. The first stage takes a global perspective by training with all of the data (in the training set) whereas the second stage re-initializes the network with the global model and fine-tunes it to each subject separately by training with only the corresponding subject data (of the training set). Hence, each subject has her/his own model. Most of the existing studies do either develop only a local model (e.g., [7, 8]) or only a global model (e.g., [35]), which indicates that our introduced two-stage training is also a novel contribution to BCI SSVEP spellers. We observe that this idea of transfer learning with two-staged learning, since it takes into account the inter-subject statistical variations, provides significant ITR improvements. In the following section of the performance evaluations, we study with two datasets independently. Namely, the global model of the first stage training is obtained for each dataset separately rather than training

a single global model based on the union of the two datasets.

5.2 Performance Evaluations

We test our DNN on publicly available two datasets, which are the benchmark [14] and the BETA [15] datasets. The state-of-the-art techniques have been previously tested on these datasets; and in our evaluations, we compare against specifically those that have been reported to perform well. In particular, we compare against 7 methods: Conv-CA, ms-eTRCA, eTRCA, TSCORRCA, m-Extended-CCA, Extended-CCA and CORRCA. In our comparisons, we follow the same test procedures for all these methods.

As explained in Chapter 4, the BCI SSVEP speller experiment consists of several blocks, so that the subject can have a break between two blocks. For example, there are 6 and 4 blocks in the benchmark and BETA datasets, respectively. In our performance evaluations, we conduct the comparisons (following the same procedure in the literature) in a leave-one-block-out fashion. We train on 5 (or 3) blocks and test on the remaining one and repeat this process 6 (or 4) times to test on each block in the benchmark (or the BETA) dataset. For each signal duration T in the range $T \in \{0.2, 0.3, \cdots, 1.0\}$, we report the mean classification accuracy and ITR along with the standard errors. We take into account a 0.5 seconds gaze shift time while computing the ITR. We test with the pre-determined set of 9 channels (Pz, PO3, PO5, PO4, PO6, POz, O1, Oz, and O2) again for fair comparisons since these channels have been used in the compared methods, but we also test with all of the available 64 channels to fully demonstrate the efficacy of our DNN. In fact, we observe improvements with 64 channels over the pre-determined set. Confusion matrices are also presented for further insights into our classification results. Additionally, we analyze the effect of the number of sub-bands and channels on the identification performance. We also report the topographic channel distributions to demonstrate the weight of each channel's contribution to the our DNN performance.

Since the available data shrinks in the second stage of our DNN training, to achieve a better regularization, the probabilities of the first two dropout layers are increased to 0.6 for the benchmark dataset [14] and to 0.7 for the BETA dataset [15]. A larger dropout probability is used for the BETA dataset as it is smaller in size (per subject) and more noisy (Chapter 4). The number of epochs (without early stopping) are

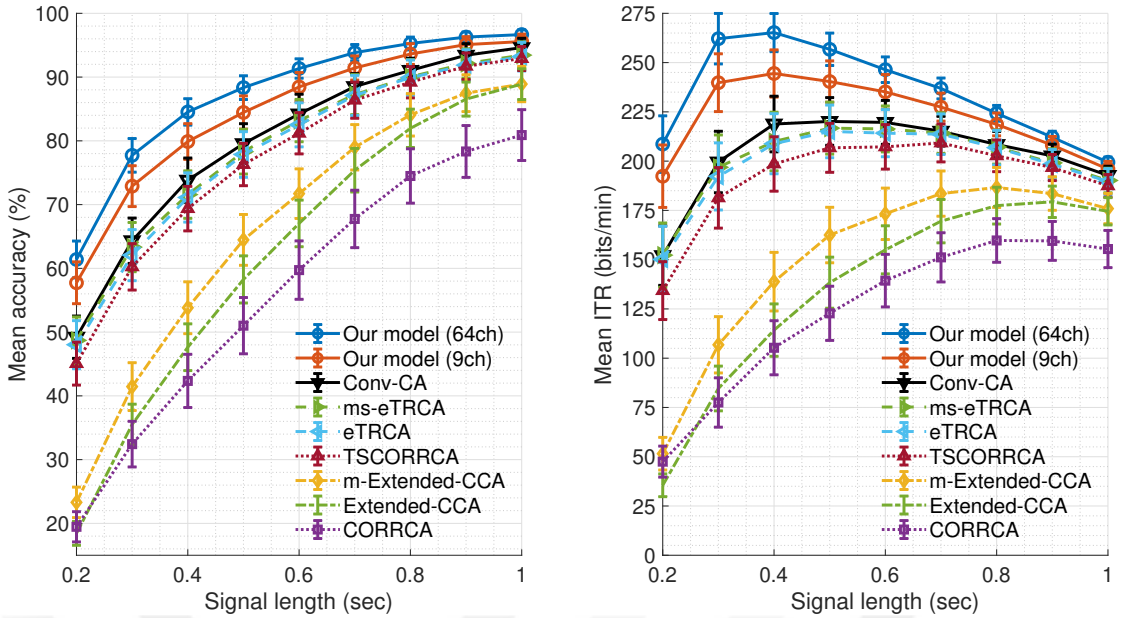


Figure 5.2 The mean classification accuracy on the left and the mean information transfer rate (ITR) on the right are presented across all 35 subjects in the benchmark dataset, together with the standard errors indicated by the bars.

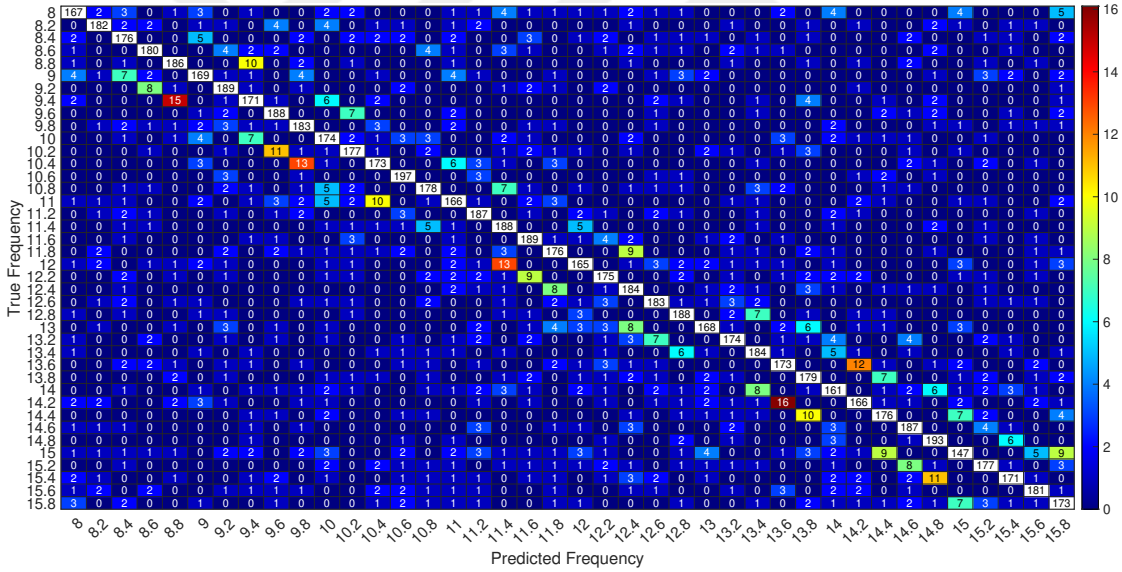


Figure 5.3 The confusion matrix of the proposed DNN with 64 channels on the benchmark dataset at 0.4 seconds of stimulation.

1000 and 800 in the first stage for the benchmark dataset and the BETA dataset, respectively, where the batch size is 100 for the both. In the second stage, the number of epochs (without early stopping) are the same and 1000 for both of the datasets and the batch sizes are 200 and 120 for the benchmark dataset and the BETA dataset, respectively. All the other settings of the proposed DNN are exactly the same between the two stages and also between the two datasets.

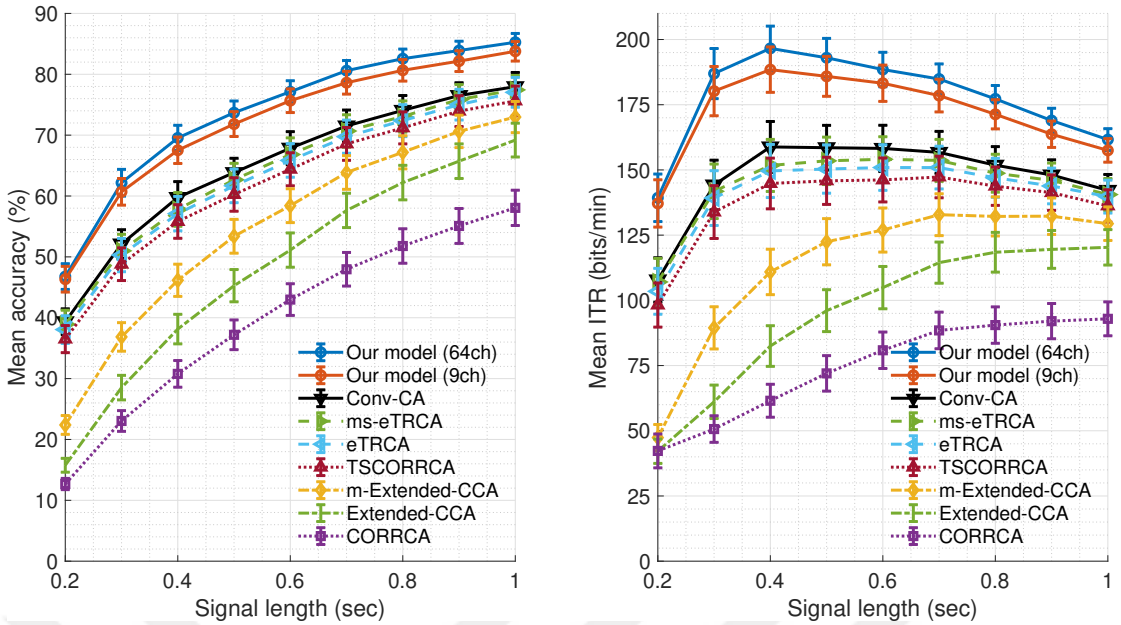


Figure 5.4 The mean classification accuracy on the left and the mean information transfer rate (ITR) on the right are presented across all 70 subjects in the BETA dataset, together with the standard errors indicated by the bars.

5.2.1 Results

The proposed DNN is observed to achieve 265.23 bits/min ($\sim 84\%$ accuracy with 64 channels) and 244.45 bits/min ($\sim 80\%$ accuracy with 9 channels) maximum ITRs (cf. Figure 5.2, and the corresponding confusion matrix in Figure 5.3 with 64 channels and 0.4 seconds of stimulation) on the benchmark dataset, and 196.59 ($\sim 70\%$ accuracy with 64 channels) bits/min and 188.45 ($\sim 68\%$ accuracy with 9 channels) bits/min on the BETA dataset (cf. Figure 5.4, and the corresponding confusion matrix in Figure 5.5 with 64 channels and 0.4 seconds of stimulation). These results are achieved in only $T = 0.4$ seconds of stimulation with using $N_s = 3$ sub-bands. *The fact that we observe such impressive results with the same exact setting in both of these independent datasets is particularly important and thereby providing further reassurance about the robustness of our presented results.* In fact, across all stimulation durations, the proposed DNN strongly outperforms all the other techniques in terms of both the accuracy and ITR, in both datasets.

In the rest of our performance evaluations, we fix the stimulation duration to $T = 0.4$ seconds as it yields the maximum ITR in both of the datasets. Also, we continue with reporting only the accuracy since it is a direct performance measure with a more intuitive interpretation and the ITR is an invertible function of the accuracy.

As reported in Table 5.1, for both of the datasets, using $N_s = 3$ sub-bands with

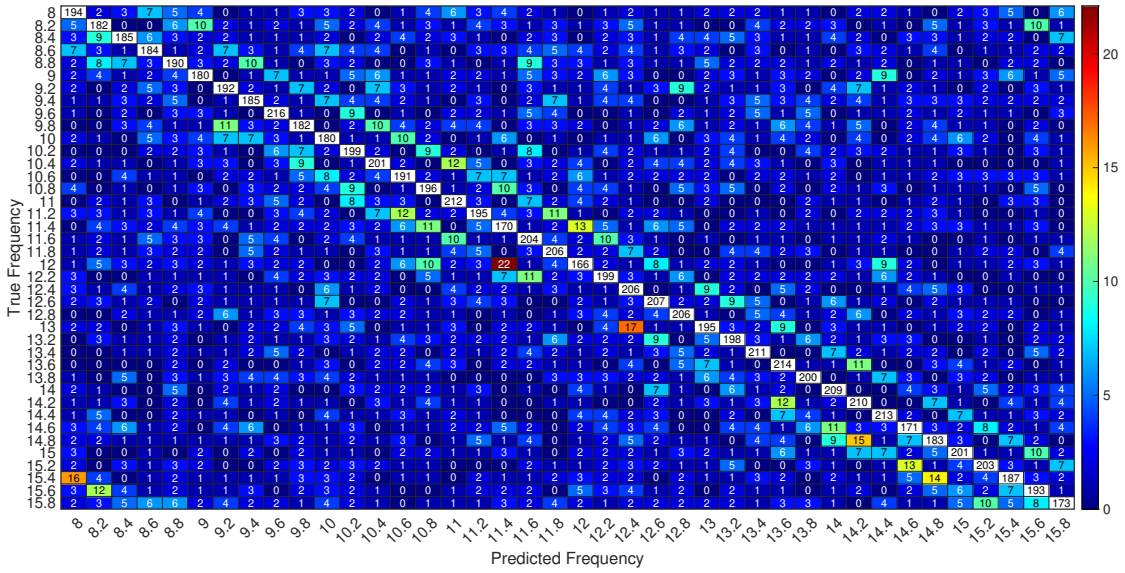


Figure 5.5 The confusion matrix of the proposed DNN with 64 channels on the BETA dataset at 0.4 seconds of stimulation.

$C = 9$ channels improves the accuracy by about $2-2.5\%$ compared to using $N_s = 1$ sub-band only. Whereas using 2 more sub-bands with $N_s = 5$ neither improves the accuracy nor does it degrade. This indicates that the first three harmonics should be taken into account differently by an appropriate combination as we do in the first layer of the proposed DNN, and harmonics beyond the third degree can be grouped and processed together. Therefore, we continue with using $N_s = 3$ sub-bands in the following.

Table 5.2 reports the accuracy with 3 (O1, Oz, O2), 6 (O1, Oz, O2, POz, PO3, PO4), 9 (Pz, PO3, PO5, PO4, PO6, POz, O1, Oz, O2) channels as typically used in the literature, and 32 channels (all channels from occipital, parietal, central-parietal regions and C3, C1, Cz, C2, C4, FCz) as well as all 64 channels. Both of using all 64 channels or 32 channels improve the accuracy but that also reduce the practicality from the user's point of view. Second, our DNN also works fairly well with 3 and 6 channels, which indicates that our algorithm can also be successfully used in the more practical systems where only few channels can be used. Overall, using 9 channels is a reasonable choice in this trade-off between the accuracy and practicality. Hence, we offer the proposed DNN with the settings of $T = 0.4$ seconds of stimulation, $N_s = 3$ sub-bands and $C = 9$ channels.

Table 5.3 presents the mean classification accuracy (along with the standard error) achieved by the proposed DNN for stages of our training, in the setting of $T = 0.4$ second of stimulation, $N_s = 3$ sub-bands and $C = 9$ channels. We observe that using only the first stage (our global model) or using only the second stage (directly training our individual models) perform comparably on the BETA dataset but the

Table 5.1 The mean classification accuracy (%), with the standard error, of our DNN is reported versus varying number of sub-bands with 9 channels and 0.4 seconds of stimulation.

	Benchmark	BETA
1 sub-band	77.89 ± 2.89	64.98 ± 2.25
2 sub-bands	78.80 ± 2.92	66.84 ± 2.17
3 sub-bands	79.89 ± 2.81	67.52 ± 2.17
4 sub-bands	79.86 ± 2.89	67.54 ± 2.12
5 sub-bands	79.96 ± 2.82	67.66 ± 2.17

Table 5.2 The mean classification accuracy (%), with the standard error, of our DNN is reported versus varying number of channels with 3 sub-bands and 0.4 seconds of stimulation.

	Benchmark	BETA
3 channels	51.04 ± 4.09	42.73 ± 2.60
6 channels	74.55 ± 3.12	58.86 ± 2.49
9 channels	79.89 ± 2.81	67.52 ± 2.17
32 channels	82.70 ± 2.63	70.21 ± 2.05
64 channels	84.54 ± 2.08	69.54 ± 2.07

latter delivers a better performance on the benchmark dataset. On the other hand, using both stages sequentially as proposed by first obtaining a global model and then fine tuning it for each individual model outperforms using only the first stage or using only the second stage on both datasets by 26% on average. This demonstrates the efficacy of our training approach. The same table also records the required running-times per epoch in each case, when run on an NVIDIA GPU (Tesla V100 Volta with memory 32GB). We observe that the second stage training takes negligible time, compared to the first stage taking ~ 30 minutes in the case of the benchmark dataset with 1000 epochs (it takes longer in the case of the BETA dataset). Note that this processing unit is specialized for deep learning algorithms, hence the running times for the first stage would scale up to several hours (completing all 1000 epochs for a single stimulation duration) on a standard computer of daily use. As for the test time (with $T = 0.4$ seconds of stimulation and $C = 9$ channels), the classification of a single instance takes about 0.008 seconds with our DNN whereas, Conv-CA and m-Extended-CCA require about 0.026 and 0.019 seconds, respectively, on the same machine.

5.2.2 Statistical Significance Analyses

Table 5.3 The mean classification accuracy (%) of our DNN is reported along with the standard error for the stages of our training. Running-times per epoch are also provided below each line.

	Benchmark	BETA
Only first stage	46.93 ± 3.30 (~ 1.80 sec.)	38.44 ± 2.12 (~ 2.25 sec.)
Only second stage	56.39 ± 4.08 (~ 0.055 sec.)	34.45 ± 2.43 (~ 0.05 sec.)
Two-stage	79.89 ± 2.81 (~ 1.855 sec.)	67.52 ± 2.17 (~ 2.30 sec.)

This part presents our statistical significance test results. Although we achieve a better performance with 64 channels, the 9 channels version of our technique is considered in the following for fairness, since all of the other compared methods use 9 channels. Also, we present ANOVA results where the effect of number of channels and sub-bands are investigated.

For a specific stimulation duration T, we conduct 7 paired t-tests each one of which analyzes the performance difference, observed in Fig. 5.2 and Fig. 5.4, between our proposed DNN (9 channels) and one of the 7 compared algorithms. These tests are repeated for each $T \in \{0.2, 0.4, 0.6, 0.8, 1\}$, and the unadjusted p-values are reported. We call an observed difference "statistically significant" (*) if the p-value is less than $\frac{0.05}{7}$ (applying "single" Bonferroni correction by 1/7 since we have 7 comparisons for each T) and "statistically highly significant" (**) if the p-value is less than $\frac{0.05}{7\times5}$ (applying "double" Bonferroni correction by 1/35 since we have 35 comparisons in total across all methods and T choices).

In the case of the benchmark dataset: In terms of the accuracy (Fig. 5.2), the least significant difference between our DNN (9 channels) and the compared methods is observed with (1) Conv-CA ($^{**}p = 3.40 \times 10^{-10}$) for $T = 0.2$, (2) eTRCA ($^{**}p = 1.04 \times 10^{-6}$) for $T = 0.4$, (3) Conv-CA ($^{**}p = 5.96 \times 10^{-4}$) for $T = 0.6$, (4) eTRCA ($p = 0.76 \times 10^{-2}$) for $T = 0.8$, and (5) Conv-CA ($p = 6.81 \times 10^{-2}$) for $T = 1$. Here, for $T = 0.8$, the difference with eTRCA is not significant; but it is significant (*) with Conv-CA and ms-eTRCA, and highly significant (**) with all the others. For $T = 1$, the difference with Conv-CA, ms-eTRCA and eTRCA are not significant; but it is significant (*) with TSCORRCA, and highly significant (**) with all the others. In terms of ITR (Fig. 5.2), the least significant difference between our DNN (9 channels) and the compared methods is observed with (1) Conv-CA ($^{**}p = 7.38 \times 10^{-10}$) for $T = 0.2$, (2) eTRCA ($^{**}p = 3.60 \times 10^{-7}$) for $T = 0.4$, (3) Conv-CA ($^{**}p = 2.55 \times 10^{-4}$) for $T = 0.6$, (4) eTRCA ($^{*}p = 0.27 \times 10^{-2}$) for $T = 0.8$, and (5) eTRCA ($p = 4.94 \times 10^{-2}$) for $T = 1$. Here, for $T = 0.8$, the difference is significant (*) with ms-eTRCA and eTRCA, and highly significant (**) with all the others. For

$T = 1$, the difference with Conv-CA, ms-eTRCA and eTRCA are not significant; but it is significant (*) with TSCORRCA, and highly significant (**) with all the others.

In the case of the BETA dataset: In terms of the accuracy (Fig. 5.4), the least significant difference between our DNN (9 channels) and the compared methods is observed with (1) Conv-CA ($^{**}p = 5.13 \times 10^{-13}$) for $T = 0.2$, (2) Conv-CA ($^{**}p = 3.58 \times 10^{-13}$) for $T = 0.4$, (3) Conv-CA ($^{**}p = 3.49 \times 10^{-10}$) for $T = 0.6$, (4) Conv-CA ($^{**}p = 3.80 \times 10^{-9}$) for $T = 0.8$, and (5) Conv-CA ($^{**}p = 3.64 \times 10^{-7}$) for $T = 1$. In terms of ITR (Fig. 5.4), the least significant difference between our DNN (9 channels) and the compared methods is observed with (1) Conv-CA ($^{**}p = 3.88 \times 10^{-12}$) for $T = 0.2$, (2) Conv-CA ($^{**}p = 1.50 \times 10^{-13}$) for $T = 0.4$, (3) Conv-CA ($^{**}p = 1.75 \times 10^{-10}$) for $T = 0.6$, (4) Conv-CA ($^{**}p = 3.84 \times 10^{-10}$) for $T = 0.8$, and (5) Conv-CA ($^{**}p = 1.32 \times 10^{-7}$) for $T = 1$. Thus, the difference (in terms of both accuracy and ITR) is always highly significant (**) with all the other compared methods and for all T's.

On the other hand, a one-way repeated measures ANOVA reveals a main effect of the number of sub-bands on the accuracy (Benchmark: $F(4,136) = 30.271$, $p < 5.18 \times 10^{-18}$; BETA: $F(4,276) = 39.793$, $p < 2.64 \times 10^{-26}$) with our proposed DNN in Table 5.1, where a paired t-test indicates a highly significant difference between using 1 sub-band and 3 sub-bands (Benchmark: $p = 3.05 \times 10^{-9}$; BETA: $p = 6.55 \times 10^{-13}$). Similarly, the number of channels (Table 5.2) has a main effect on the accuracy (Benchmark: $F(4,136) = 86.007$, $p < 2.82 \times 10^{-36}$; BETA: $F(4,276) = 162.24$, $p < 3.23 \times 10^{-71}$), where a paired t-test indicates a highly significant difference between using 9 channels and 64 channels (Benchmark: $p = 5.29 \times 10^{-4}$; BETA: $p = 3.89 \times 10^{-4}$). The training strategy (Table 5.3) also has a main effect (Benchmark: $F(2,68) = 87.466$, $p < 1.58 \times 10^{-19}$; BETA: $F(2,138) = 218.08$, $p < 1.901 \times 10^{-43}$), where a paired t-test indicates a highly significant difference between employing only the second stage and two-stage (Benchmark: $p = 1.13 \times 10^{-13}$; BETA: $p = 2.93 \times 10^{-37}$).

5.2.3 Error Patterns

Considering the inter-class confusions presented in Figure 5.3 and Figure 5.5, we observe two prominent error patterns. The first pattern is that there exists a pronounced rate of error diagonally along the two lines $F_{\text{true}} = F_{\text{predict}} \pm 0.6$, but this pattern completely disappears at even the adjacent closest neighbors of $F_{\text{true}} = F_{\text{predict}} \pm 0.2$ or $F_{\text{true}} = F_{\text{predict}} \pm 0.4$. This is surprising as one would expect

to confuse the target character with the character of the closest frequency. The reason we discover for this finding is the following. Firstly, we define the mean absolute distance $M_D(i,j)$ between two contrast modulating sinusoids with frequencies F_i, F_j from the set $\{8, 8.2, \cdots, 15.8\}$ and phases θ_i, θ_j from the set $\{0, 0.5\pi, \pi, 1.5\pi\}$ as the following.

$$M_D(i,j) = \frac{1}{T \times R} \sum_{k=0}^{T \times R - 1} |s(F_i, \theta_i, k) - s(F_j, \theta_j, k)|, \tag{5.2}$$

where $R = 60$ Hz is the refresh rate of the monitor, k is the discrete time index, T is the stimulation duration, and the contrast modulating sinusoid is defined as $s(F, \theta, k) = \frac{1}{2}(1 + \sin(2\pi F k/R + \theta))$ (see the dataset descriptions in [14] or [15]). The distance between the samples, which multiply the luminance of the character thumbnails as shown in Figure 1.1, from two contrast modulating sinusoids with the frequencies F_1 and F_2 is the smallest when the frequencies are chosen as $F_2 = F_1 \pm 0.6$ whereas it is the largest when chosen as $F_2 = F_1 \pm 0.2$ or $F_1 \pm 0.4$. This is demonstrated in the matrix of distances in Figure 5.6, and please see the strong correlation between the pattern in the Figure 5.6 and the pattern in the confusion matrices of Figure 5.3 and Figure 5.5. Consequently, during stimulation, the luminance variations falling onto the retina and the corresponding early projections to the visual cortex are maximally similar when the frequencies are F_1 and $F_1 \pm 0.6$ and maximally dissimilar when the frequencies are F_1 and $F_1 \pm 0.2$ or $F_1 \pm 0.4$. This similarity appears to reduce the discrimination power, hence negatively affects the performance.

This first pattern is perhaps best understood with the vertical intermodulations (IM) resulting from the layout used in the character matrix, which emerges in our study as the second error pattern that is uniformly observed in the confusion matrix (Figure 5.3) of the benchmark dataset. The source of confusion by this second error pattern is the well-known IM phenomenon (cf. [51] and the references therein) which generates the IM components in the SSVEP spectrum at the integer multiples of the spatially nearby flickering frequencies that the subject is exposed to. If an IM is generated that is close to the frequency of a non-target character and also close to the one of the target character itself, then there seems to happen an error due to confusion. Since this is consistently and strongly observed in the confusion matrix (Figure 5.3) of the benchmark dataset, we present as an important finding of our study that the vertical IM, together with the first pattern of sinusoidal distances, has an important effect on how the errors happen in an emphasizing-the-first-pattern (if the IM exists) or suppressing-the-first-pattern (if the IM does not exist) manner. Namely, the relatively large rate of confusion in the first pattern between the

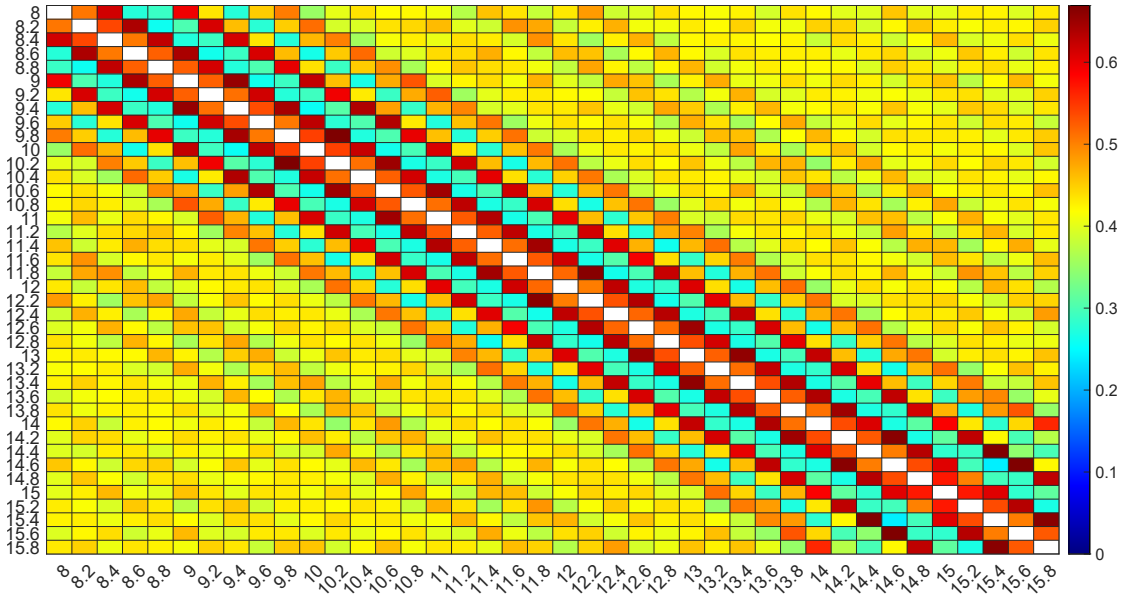


Figure 5.6 The matrix M_D of the mean absolute distances for any pair of contrast modulating sinusoids with frequencies $\{8, 8.2, \cdots, 15.8\}$ that are used for frequency tagging in the BCI SSVEP spellers of both the benchmark and BETA datasets. The distance is the smallest when two frequencies are 0.6 or 0.8 Hz apart, and the largest when $0.2, 0.4$ or 1 Hz apart.

frequencies F_1 (predicted frequency) and F_2 (true frequency) is persistent, A) only when the target character (F_2 for $F_2 = F_1 + 0.6$) is on the first, second or the third rows of the character matrix (Figure 4.1a) and also B) only when the target character (F_2 for $F_2 = F_1 - 0.6$) is on the third, fourth or fifth rows. We attribute this to the interference by the fifth degree vertical IM generated in both the cases A and B: In the case A, the confusing predicted frequency F_1 can be obtained as the 5th IM $F_1 = 3 \times F_2 - F_2^{l1} - F_2^{l2}$, and in the case B, F_1 can be obtained as the 5'th IM $F_1 = 3 \times F_2 - F_2^{u1} - F_2^{u2}$. Here, l_k or u_k are the k^{th} lower/upper adjacent frequency in the character matrix for $k = 1$ or $k = 2$. For example, the true (target) frequency $F_2 = 14.2$ Hz (character "O" on the second row in Fig. 1.1 and Fig. 4.1a) has the two lower neighbors $F_2^{l1} = 14.4$ Hz and $F_2^{l2} = 14.6$ Hz, generating the 5'th degree IM of case A as $3 \times F_2 - F_2^{l1} - F_2^{l2} = 3 \times 14.2 - 14.4 - 14.6 = 13.6$ Hz. Since this IM component appears in the received EEG signal and when this existence is strong enough, it is predicted as the true frequency, i.e., $F_1 = 13.6$ Hz (character "3"), which actually corresponds to the most frequent error (16 times of confusion 14.2 Hz vs 13.6 Hz) in Fig. 5.3. Such errors can be visually traced by noting the colored patterns below (case A) and above (case B) the diagonal of the confusion matrix in Fig. 5.3. Note that we do not observe a detrimental IM when the target character is on the first or second rows in the case A and if it is on the fourth and fifth rows in the case B, namely, if it does not have two vertical adjacent neighbors. Also, a detrimental IM is not observed horizontally or also not observed at lower

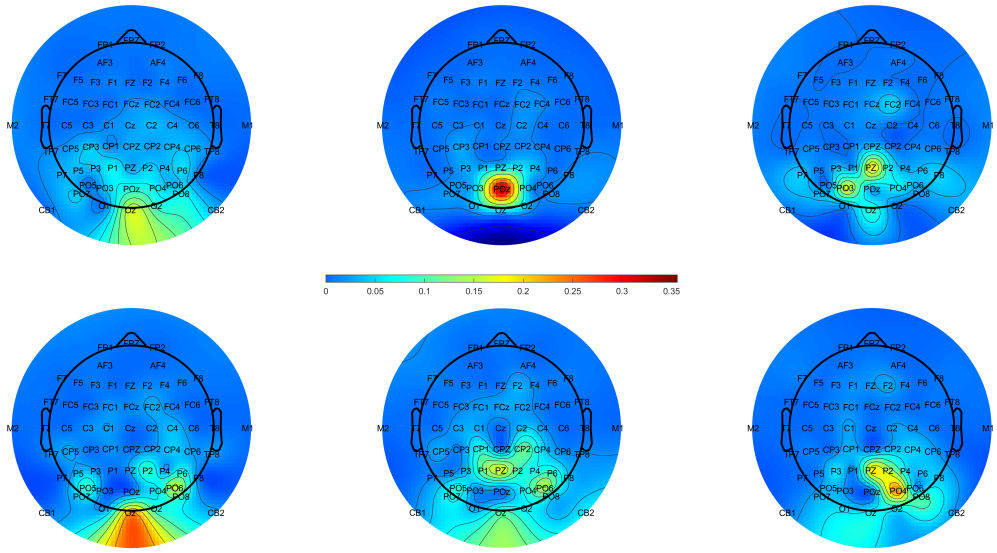


Figure 5.7 Upper row (and the lower row) presents the topographic map of the 3 channel combinations, in no particular order, learned by the proposed DNN in the case of the benchmark (and the BETA) dataset.

degrees (lower than 5) because of not that the horizontal and lower degree IMs are not generated (they are certainly generated) but that they are not any close to the frequency of the target character. Hence, an error does not happen in that specific way. We emphasize that the intermodulation effect is specific to the character matrix used in the stimulation. For this reason, we do not observe the same exact second pattern in the keyboard presentation of the BETA dataset (cf. Figure 5.5) as the characters are shuffled on the matrix. *These two error patterns can provide key insights to the matrix and stimulation design in future studies.*

5.2.4 Topographic Maps

Lastly, we study the importance of the electrodes (i.e. channels), in terms of their contribution to the target identification accuracy, by analyzing the channel combinations learned by the proposed DNN. For this purpose, we concentrate on only 3 combinations (since analyzing all 120 combinations given by the network would not be practical) such that for instance a large weight (in absolute value) in the combinations for a channel indicates a high importance. In the current setting, our network learns 120 channel combinations in its second layer (cf. Figure 1.1) for the best achievable accuracy but does not rank them with respect to their importance to the accuracy. Hence, currently, it is not possible (without a post-component

analysis of the 120 combinations which is not in the scope of the presented work) to immediately choose the most important 3 combinations out of those 120 ones. Nevertheless, in this part only, in order to have the network determine the 3 combinations, we set the size of the second layer output as $1 \times 50 \times 3$ instead of the original size, and then train the network based on the entire set of available data when fed with all available 64 channels. Based on this approach, Figure 5.7 presents the topographic maps[1] of the 3 channel combinations (without ranking them) that are learned by the proposed DNN in the both datasets. In Figure 5.7, each channel has a color closer to red (blue) if it has proportionally higher (lower) weight in the absolute value. We first observe that the channels that are recommended in the study [53] are also mostly covered by our network, which is an independent verification of a previous result. Therefore, this indicates that our network does not or limitedly overfit. Additionally, the important channels are more concentrated in the case of the benchmark dataset, whereas they are spread more in the case of the BETA dataset. We attribute this to the low SNR of the BETA dataset. Therefore, unlike [53], when the SNR is low, we tend to recommend more channels from the parietal region (such as P1, P2). Whereas we strongly recommend the channels Oz and Pz as they are shared by the both datasets. Also, our topographic maps are complementary to each other indicating the necessity of combining the channels, and that is to be nonlinearly because a totally linear approach could exploit combinations as many as the number of channels.

[1]The maps are generated by EEGLAB [52].

6. PROPOSED SOLUTION IN THE TRAINING FREE AND TRANSFERRED SETTING: ENSEMBLE-DNN

In this chapter we present our ensemble based method that does not require any labeled or unlabeled data from the new user, so that the new user immediately can start to use the system. Our ensemble based method trains an ensemble of DNNs with previously existing data from different training subjects. Then, this ensemble is transferred to the new user, and thereafter the weighted combination of the constituent DNNs are combined for making predictions. A certain similarity metric (as explained in the following section) from the training subjects to the new (test) user is also used before the prediction for determining the most representative DNNs. We have used our proposed DNN architecture, since it significantly outperforms the state-of-the-art methods with the highest ITR, when the user-specific training is used. In the following, we provide the details of this ensemble based method, and its performance evaluations' results on the benchmark [14] and the BETA [15] datasets.

6.1 Ensemble-DNN

To construct the ensemble, we have used a training strategy that is similar to the training of our DNN in one aspect as the both utilize a two-stage approach, but completely different in another aspect. The fundamental difference is that here there is no user-specific training. Namely, we use two-stage training to construct an ensemble of classifiers based on all the available data from all subjects except one subject that is reserved for testing. Since the data of the reserved subject does not join the training phase, the proposed ensemble method is free of user-specific training. In the DNN architecture, we use our suggested setting, i.e., we have used three sub-bands and nine channels (Pz, PO3, PO5, PO4, PO6, POz, O1, Oz, O2),

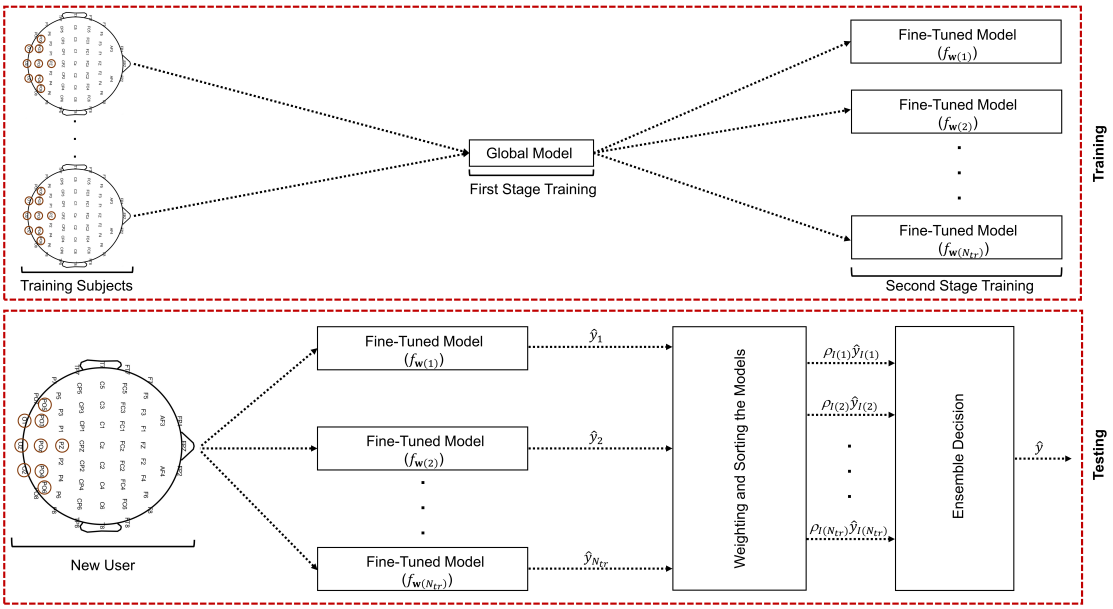


Figure 6.1 The proposed ensemble based method is illustrated. As a model, our DNN structure is adapted. In the training phase, previously collected data of N_{tr} different subjects are used to train a global model first. Then, this global model is fine-tuned to each subject which results in N_{tr} different fine-tuned models ($f_{\mathbf{w}}(n)$). In the testing phase, most representative k fine-tuned models are chosen based on the similarity between the new (test) user data and the subject data used to train $f_{\mathbf{w}}(n)$. The predictions of these k models are combined in weighted manner via giving proportional weights to the models based on their similarity, to make the final prediction $\hat{y}$. Please refer to the text for the details of our method. In our experiments, we have observed that our method is significantly superior over the state-of-the-art alternatives in terms of both the target identification accuracy and information transfer rate.

whose physical locations on the skull also are labeled in Fig. 6.1. The details of the training/construction of models ensemble are as follows.

In the first training stage, the data of a specific subject is excluded to be used in testing and the model is trained with all the available data of the rest of the subjects using the same values of the DNN's parameters in Chapter 5. Specifically, the first layer weights of the DNN architecture are initialized with unity, and initialization of other layers' weights are via sampling from Gaussian distribution with 0 mean and standard deviation 0.01. To prevent the network from overfitting, several dropouts layers are used, which ignores some of the neurons during the training. In these dropouts layers, 0.1, 0.1, and 0.95 dropout probabilities are applied between second and third, third and fourth, and fourth and fifth layers, respectively. The network is trained based on the training batch data to minimize the categorical cross entropy

loss that is defined as

$$\frac{1}{D_b}\sum_{i=1}^{D_b} -\log(\mathbf{s}_{i,y_i}) + \lambda|\mathbf{w}|^2,$$

where $\lambda = 0.001$ is the weight of the L2 regularization, D_b is the number of trials in the batch, $\mathbf{s}_{i,y_i}$ is the y_i'th index of the softmax output for the instance $\mathbf{x}_i$, y_i is the true label, and $\mathbf{w}$ is the weights of all the layers in the DNN. In the second stage, the global model learned in the first stage is fine-tuned to each subject except the excluded one in the first stage. The network is re-initialized to be trained by only the corresponding subject data, with the weights learned in the first training stage. Hence, we obtain 34 (69) fine-tuned models that will be used to classify data of the test subject in the case of the benchmark dataset [14] (BETA dataset [15]).

The details of the selection of the subject-specific models, whose decisions are used to determine final prediction, are explained next. As discussed, we determine these subject-specific models based on the similarities between the training subjects' data and the user's data. The idea behind this selection process is that the fine-tuned networks of the training subjects, whose statistics are more similar to the statistics of the new user, should provide more reliable predictions because the statistical variation is minimized among similar subjects. In the calculation of the similarity, a training subject's instances labeled with the final target character prediction of the corresponding subject's fine-tuned network are used rather than all of his or her data; because using all the data would be computationally costly. Also, we assume that the subject's instances having the same label as the model's character prediction is the most similar instances among all the subject's data, so it is enough to use only those instances to measure the similarity.

We let $f_{\mathbf{w}(n)}$ be the model with the parameters $\mathbf{w}(n)$ that is fine-tuned to the n^{th} training subject, and $\hat{y}_n$ be the final target character prediction of that subject specific model to the new user preprocessed instance $\mathbf{x}$[1], i.e., $\hat{y}_n = f_{\mathbf{w}(n)}(\mathbf{x})$. Here, $\mathbf{x} = [\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N_s)}] \in \mathbb{R}^{C \times N_t \times N_s}$ is the multi-dimensional brain response, measured by EEG using C channels/electrodes with N_t sample points (i.e., $N_t = T \times F_s$ is the number of time samples with F_s sampling frequency in Hz and T signal length in seconds), and preprocessed with N_s many band-pass filters. We also denote the n^{th} training subject's individual template for the i^{th} target character by $\overline{\mathbf{x}}_n^i$, which is basically mean of the instances having the i^{th} label. To measure the similarity (ρ_n) between the user and the n^{th} training subject, a correlation-based metric is used to measure the similarity as most techniques in the literature (cf. Chapter 2) are

[1]For the ease of exposition, in this chapter we denote the preprocessed instance by $\mathbf{x}$.

Algorithm 1 Target identification of our method for a preprocessed test instance $\mathbf{x}$

1: **for** $n = 1$ *to* N_{tr} **do**
2: $\hat{y}_n = f_{\mathbf{w}(n)}(\mathbf{x})$: prediction of the n^{th} model.
3: Determine the corresponding training template data $\overline{\mathbf{x}}_n^{\hat{y}_n}$.
4: $\underline{\mathbf{x}}_n = \sum_{s=1}^{N_s} \mathbf{w}_s^{(s)}(n)\mathbf{x}^{(s)}$: output of the sub-bands combination layer of the n^{th} model to the x.
5: $\underline{\overline{\mathbf{x}}}_n^{\hat{y}_n} = \sum_{s=1}^{N_s} \mathbf{w}_s^{(s)}(n)\overline{\mathbf{x}}_n^{\hat{y}_n(s)}$: output of the sub-bands combination layer of the n^{th} model to the $\overline{\mathbf{x}}_n^{\hat{y}_n}$.
6: Calculate the similarity ρ_n using equation (6.1) and (6.2).
7: Sort ρ_n in descending order to get the indices I (the most similar subject index is $I(1)$, the second most similar subject index is $I(2)$ etc.).
8: Find the final prediction $\hat{y}$ using equation (6.3).

correlation-based, showing correlation is an appropriate metric for measuring the similarity.

The utilized correlation metric is the summation of two correlation coefficients. The first one ($\boldsymbol{\rho}_{n,1}$) is the correlation between the n^{th} subject's template $\overline{\mathbf{x}}_n^{\hat{y}_n}$ and the user instance $\mathbf{x}$. The second one ($\boldsymbol{\rho}_{n,2}$) is the correlation between the instance $\mathbf{x}$ and the artificial reference signal $\mathbf{Y}^{\hat{y}_n}$ that is formed for the $\hat{y}_n^{th}$ target character. The artificial signal $\mathbf{Y} \in \mathbb{R}^{2N_h \times N_t}$, where N_h is the number of harmonics (we use 5 harmonics, i.e., $N_h = 5$), is constituted for each target character. And, each artificial reference signal is composition of the sinusoidal signals with the corresponding target character tagging frequency and its harmonics.

The first correlation coefficient $\boldsymbol{\rho}_{n,1}$ measures the direct similarity between the n^{th} subject and the new user. In the second correlation $\boldsymbol{\rho}_{n,2}$, the correctness of the n^{th} subject network's prediction is aimed to be measured. We have intuitively assumed that if the subject's network classify correctly the new instance, the second correlation between the corresponding subject and the new user will be high.

Furthermore, all the variables ($\mathbf{x}$, $\overline{\mathbf{x}}$, $\mathbf{Y}$) are multi-dimensional, so we can not calculate the correlation coefficients between these variables directly. And, spatial similarity should also be measured between the subjects and the user, as the spatial characteristics can be significantly different between the individuals. We calculate the correlation coefficients using a channel combination $\mathbf{w}_c^{(*)} \in \mathbb{R}^{C \times 1}$ selected from the channel combination layers of the subjects' networks, which are expected to describe spatial characteristics of the subjects. Using similar notation with Chapter 5, the channel combination weights from the second layer of the n^{th} subjects's network are denoted by $\mathbf{w}_c(n)$, i.e., $\mathbf{w}_c(n) = \{\mathbf{w}_c^{(1)}(n), \mathbf{w}_c^{(2)}(n), ..., \mathbf{w}_c^{(N_{ch})}(n)\} \in \mathbb{R}^{C \times N_{ch}}$, where N_{ch} is the number of channel combinations that equals to 120 in our case

(Fig. 5.1). Please also note that the channel combination layer comes after the sub-bands combination layer in the DNN (Chapter 5, Fig. 5.1), so the learned weights in the channel combination layers are for the inputs, whose sub-bands are combined by the sub-bands combination weights that are denoted by $\mathbf{w}_s(n)$ (i.e. $\mathbf{w}_s(n) = \{\mathbf{w}_s^{(1)}(n), \mathbf{w}_s^{(2)}(n), ..., \mathbf{w}_s^{(N_s)}(n)\} \in \mathbb{R}^{N_s \times 1}$). Therefore, to use the learned channel combinations effectively and to get benefit from preprocessing step also in the calculation of the similarity, we use the user instance $\mathbf{x}$, and the n^{th} subject's template $\overline{\mathbf{x}}_n$ by combining their sub-bands with $\mathbf{w}_s(n)$ in the calculation of the similarity ρ_n. After combining the sub-bands of the input $\mathbf{x}$ with $\mathbf{w}_s(n)$, the resulted variable is denoted by $\underline{\mathbf{x}}_n$, i.e. $\underline{\mathbf{x}}_n = \sum_{s=1}^{N_s} \mathbf{w}_s^{(s)}(n)\mathbf{x}^{(s)}$. Similarly, we denote the resulted variable after the sub-bands combination of the n^{th} subject's template by $\underline{\overline{\mathbf{x}}}_n^i$.

Also, a harmonic combination $\mathbf{w}_Y \in \mathbb{R}^{2N_h \times 1}$ is needed in the second correlation coefficient $\boldsymbol{\rho}_{n,2}$, to combine the harmonics in $\mathbf{Y}$. This harmonic combination $\mathbf{w}_Y$ is found by CCA for the given channel combination $\mathbf{w}_c^{(*)}$. With these parameters, we define a correlation vector $\boldsymbol{\rho}_n(\mathbf{w}_c^{(*)})$, which is:

$$\boldsymbol{\rho}_n(\mathbf{w}_c^{(*)}) = \begin{bmatrix} \boldsymbol{\rho}_{n,1}(\mathbf{w}_c^{(*)}) \\ \boldsymbol{\rho}_{n,2}(\mathbf{w}_c^{(*)}) \end{bmatrix} = \begin{bmatrix} \rho((\mathbf{w}_c^{(*)})^T \underline{\mathbf{x}}_n, (\mathbf{w}_c^{(*)})^T \underline{\overline{\mathbf{x}}}_n^{\hat{y}_n}) \\ \rho((\mathbf{w}_c^{(*)})^T \underline{\mathbf{x}}_n, (\mathbf{w}_Y)^T Y^{\hat{y}_n}) \end{bmatrix}.$$

The channel combination $\mathbf{w}_c^{(*)}$ is selected from $\mathbf{w}_c(n)$ to maximize the summation of the correlation coefficients term, as follows:

$$\mathbf{w}_c^{(*)} = \underset{\mathbf{w}_c^{(i)} \in \mathbf{w}_c(n)}{\arg\max} \sum_{k=1}^{2} (\boldsymbol{\rho}_{n,k}(\mathbf{w}_c^{(i)}))^2. \tag{6.1}$$

Then, with the found $\mathbf{w}_c^{(*)}$, the similarity ρ_n between the n^{th} training subject and the user is the summation of the correlation terms:

$$\rho_n = \sum_{k=1}^{2} (\boldsymbol{\rho}_{n,k}(\mathbf{w}_c^{(*)}))^2. \tag{6.2}$$

After obtaining ρ_n's for each training subject and sorting in the descending order (I keeps the order and $I(1)$ gives the index of the most correlative/similar subject),

the final prediction is made by k most similar subjects:

$$\hat{y} = \underset{i \in \{1,\ldots,M\}}{\arg\max} \sum_{j=1}^{k} \rho_{I(j)} \mathbb{1}(\hat{y}_{I(j)} = i), \tag{6.3}$$

where $\mathbb{1}(.)$ is the indicator function which equals to 1 if the condition is satisfied, otherwise 0. The parameter k is a hyperparameter, which can be chosen empirically or by cross-validation using training subjects' data. However, since the new user's statistics are different from training subjects' statistics, using cross-validation might not be reliable. Choosing by hand empirically is also not feasible as the best k could vary for each new user. Overfitting could also effect detrimentally. Therefore, we opted to follow a more sound strategy. After calculating the predictions for all $k \in \{1, 2, \cdots, N_{tr}\}$ using (3), the value with the largest prediction confidence is chosen. Here the confidence is measured with the weight difference between the most weighted character and the second most weighted character. We call this proposed ensemble selection strategy as "Dynamic Selection" in the rest of the chapter. All the steps to classify the preprocessed test instance $\mathbf{x}$ are presented in Algorithm 1.

6.2 Performance Evaluations

To test the performance of our method, we have used leave-one-subject-out procedure. Namely, for each dataset, the data of one subject is excluded, and the data of the remaining subjects are used to train the DNNs. Then, the proposed method performance is evaluated on the data of the excluded subject and this procedure is repeated for each available subject in the dataset (35 and 70 times for the benchmark and BETA datasets, respectively). We compare our method against the tt-CCA, combined-tCCA, ttf-CCA, and FBCCA methods and report the mean target identification (i.e., multiclass classification) accuracy and ITR along with the corresponding standard errors (across leave-one-out folds). Same leave-one-subject-out procedure is used while evaluating performances of tt-CCA, combined-tCCA and ttf-CCA methods. Additionally, we compare our dynamic ensemble selection procedure with the weighted combination as well as the majority voting procedures to demonstrate effectiveness of the proposed dynamic selection procedure.

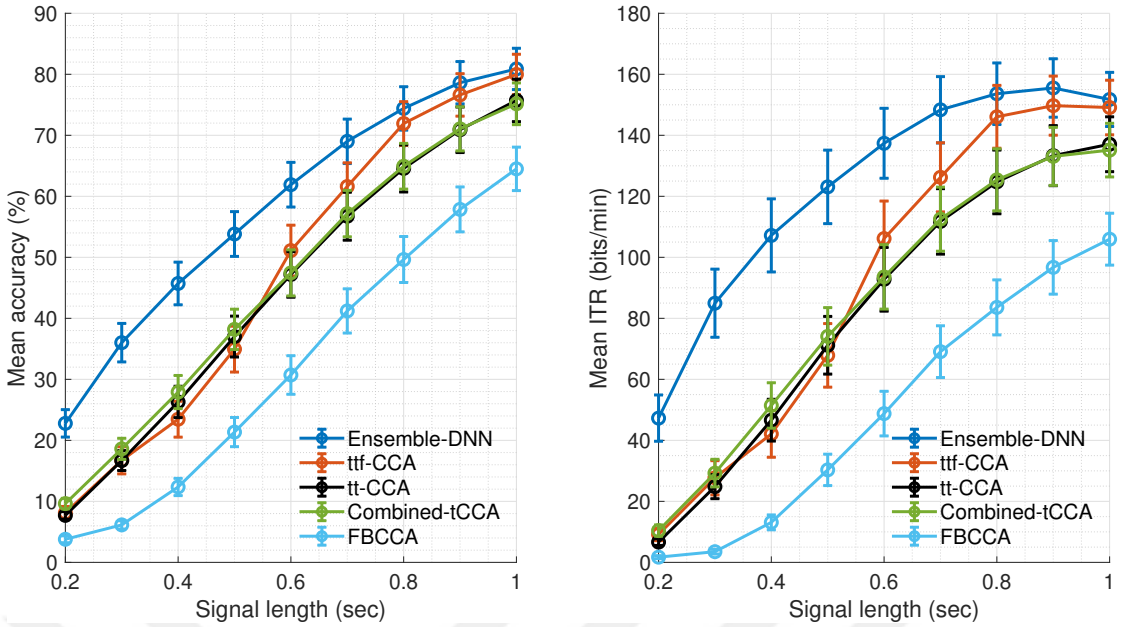


Figure 6.2 The mean classification accuracy on the left and the mean information transfer rate (ITR) on the right are presented across all 35 subjects in the benchmark dataset, together with the standard errors indicated by the bars.

6.2.1 Results and Discussion

In the benchmark dataset (Fig. 6.2), our proposed method achieves the maximum ITR 155.51 bits/min (78.61% target identification accuracy at 0.9 seconds of signal length). We have also presented the performance findings of the ttf-CCA, tt-CCA, Combined-tCCA and FBCCA methods in the same plots. Our results indicate that the proposed ensemble method significantly outperforms all the other compared methods within the range of stimulation (i.e., signal length) of $[0.2, 1]$ seconds. This range is the most used range in the literature, probably because the prolonged exposure to flickering stimulus in the SSVEP experiments quickly becomes too tiring for the subjects [54]. Hence, the shortest stimulation duration is certainly preferable where one also needs to achieve a sufficiently high ITR. In this sense, our proposed method is significantly superior. In the BETA dataset (Fig. 6.3), our proposed method achieves the maximum ITR 114.36 bits/min (64.34% accuracy at 0.8 seconds). Similar to the benchmark dataset, we also demonstrate that our method significantly outperforms the other techniques in the BETA dataset as well. To the best of our knowledge, among the methods that do not require the user specific data from each new user, and do not utilize any sort of user-specific adaptation, these ITR values are the highest ever reported performance results on these datasets.

Fig. 6.4 shows the mean target identification accuracy comparisons among k models used in the weighted combination and majority voting procedures, as well as

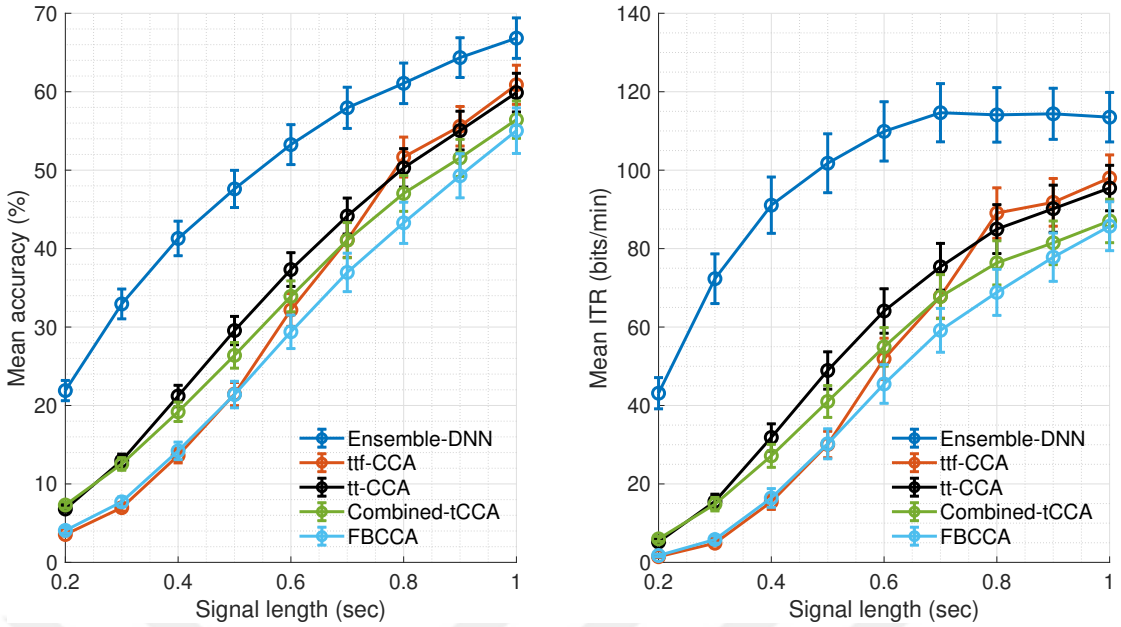


Figure 6.3 The mean classification accuracy on the left and the mean information transfer rate (ITR) on the right are presented across all 70 subjects in the BETA dataset, together with the standard errors indicated by the bars.

shows the our dynamic ensemble selection strategy's mean classification accuracy. The results reveal that our proposed dynamic ensemble selection strategy performs better than the weighted combination and the majority voting procedures' best performances among various k values. Also, in the same figure, better performance of the weighted combination procedure than the majority voting procedure shows the effectiveness of the weighting the classifiers based on their similarities.

It is also possible to make predictions with the global model (the first stage training model without the second stage), but as illustrated in Table 6.1, the accuracy and ITR results are observed to be lower than the proposed ensemble method. Comparisons are given for only 0.8, 0.9 and 1 seconds as we generally observe the highest ITRs at these signal lengths. Our proposed ensemble approach greatly enhances the global model in terms of the ITR and accuracy. The poor performance of the global model can be perhaps explained best by the significant statistical difference between most of the training subjects and the test subject. Fig. 6.5 demonstrates this fact by showing a relationship of the global model mean accuracy with the mean and standard deviation of that of the selected subject-specific models, whose selection is determined by our dynamic selection algorithm. There are total of 35 (70) dots, each of which represents a subject, on the left (right) in Fig. 6.5 for the benchmark (BETA) dataset. Our ensemble method is expected to consistently select more models for the test subjects whose statistics are very similar to the statistics of many training subjects. Global model performances on these test subjects are

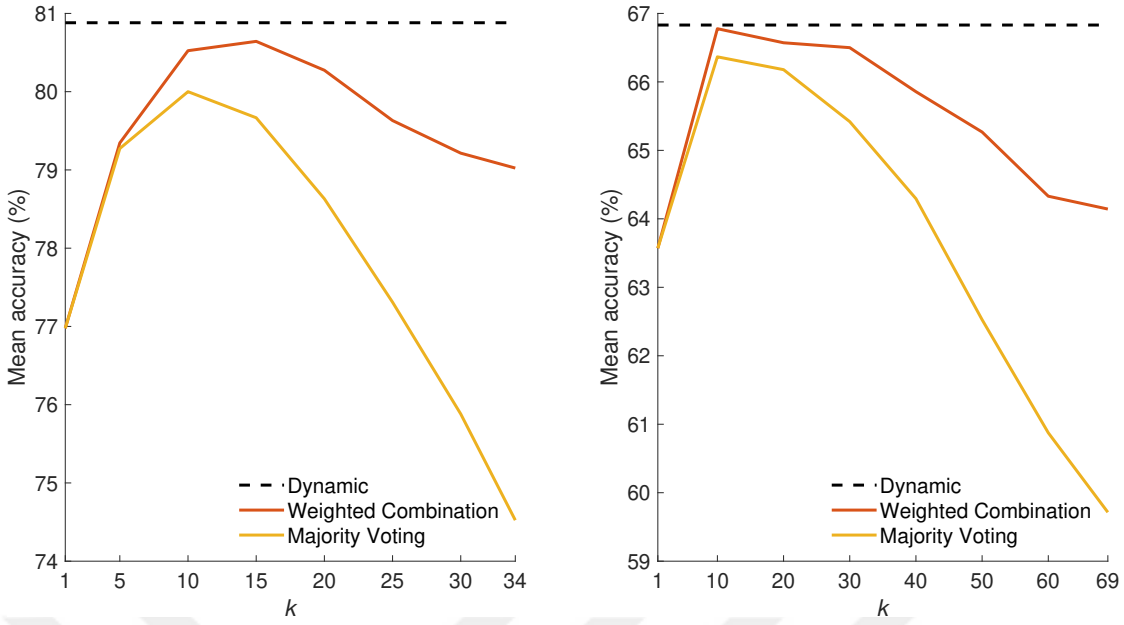


Figure 6.4 The mean classification accuracy of our dynamical ensemble selection procedure, as well as the weighted combination and the majority voting procedures for varying k values, are presented on the benchmark dataset (on the left) and the BETA dataset (on the right) at 1 seconds of signal duration.

Table 6.1 The mean target identification accuracy (on the first rows) and the ITR comparisons (on the second rows) between the proposed ensemble method and the global model (of our first stage training without fine-tuning) are presented in both the benchmark and BETA datasets.

	Benchmark		BETA	
	Ensemble	Global	Ensemble	Global
	80.88	70.61	66.83	56.58
1 sec	151.75	123.31	113.50	88.12
	78.61	68.58	64.34	54.56
0.9 sec	155.51	126.58	114.36	89.22
	74.40	64.42	61.07	51.94
0.8 sec	153.62	123.54	114.09	89.21

also expected to be high, as the average statistical similarity is higher. This is the reason for the observed relation/trend in Fig. 6.5, where it is demonstrated in the cases of that our dynamic selection methodology uses a consistently higher number of subject-specific models (i.e., if the standard deviation of k is low and the mean of k is high), the global model accuracy is observed to be also high. And, since in most of the test subjects, the mean of k is low and the standard deviation of k is high, the global model performance on average is not as good as our ensemble method.

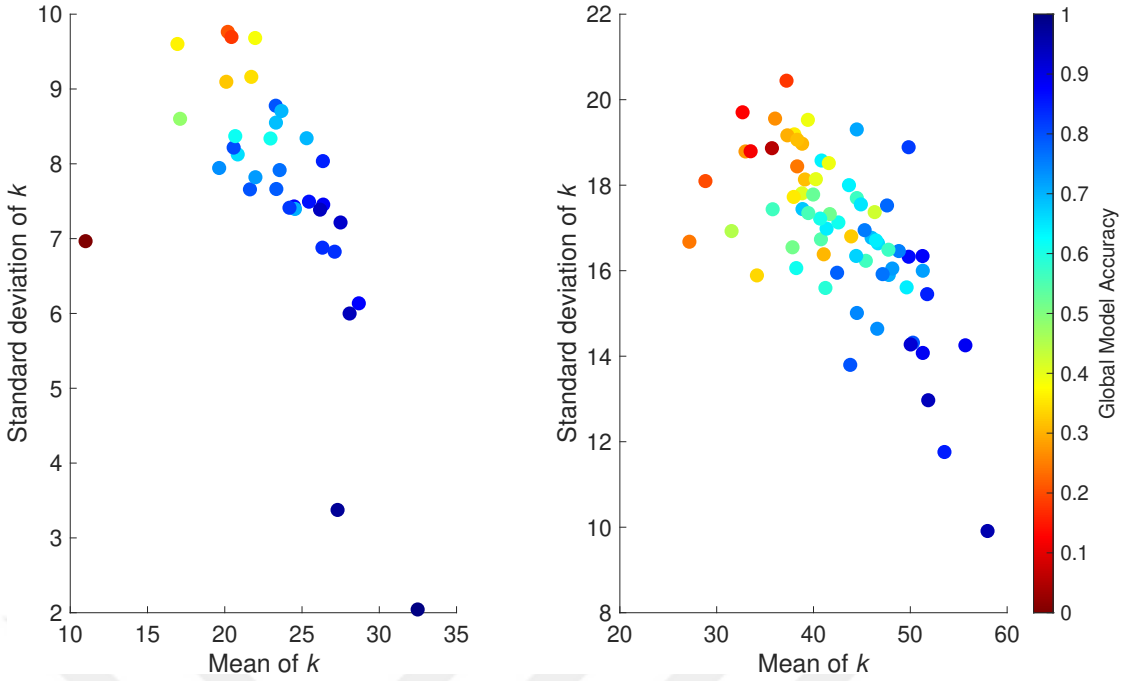


Figure 6.5 While each dot represents one test subject, the x-axis shows the average number of classifiers that our ensemble model selection method uses to classify the test subjects' data, and the y-axis shows the standard deviation of the number of used models across the instances of test subjects. The color map shows the accuracy rate of the global model on test subjects. The left figure is for the benchmark dataset, and the right figure is for the BETA dataset.

6.2.2 Statistical Significance Analyses

In this part, we present our statistical significance test results. In a similar way explained in Section 5.2.2, for each $T \in \{0.2, 0.4, 0.6, 0.8, 1\}$, we conduct 4 paired t-tests, pairing our proposed "Ensemble-DNN" method with the compared methods in Fig. 6.2 and Fig. 6.3. We report unadjusted p-values, and the observed difference is called as "statistically significant" (*) if the p-value is less than $\frac{0.05}{4}$ and "statistically highly significant" (**) if the p-value is less than $\frac{0.05}{4\times5}$. For the "statistically significant" case, we apply "single" Bonferroni correction by dividing 0.05 by 1/4, since for each T there are 4 comparisons. And for the "statistically highly significant" case, we apply "double" Bonferroni correction by 1/20, since across all methods and T choices there are 20 comparisons.

In the case of the benchmark dataset: In terms of the accuracy (Fig. 6.2) the least significant difference between our "Ensemble-DNN" method and the compared methods is observed with (1) ttf-CCA ($^{**}p = 9.40 \times 10^{-9}$) for $T = 0.2$, (2) ttf-CCA ($^{**}p = 7.37 \times 10^{-9}$) for $T = 0.4$, (3) ttf-CCA ($^{**}p = 5.82 \times 10^{-6}$) for $T = 0.6$, (4) ttf-CCA ($p = 0.27 \times 10^{-1}$) for $T = 0.8$, (5) ttf-CCA ($p = 0.48$) for $T = 1.0$. For $T = 0.8$,

the difference with ttf-CCA is not significant; but it is highly significant (**) with all the others. For $T = 1.0$, the difference with ttf-CCA is not significant; but it is significant (*) with tt-CCA and highly significant (**) with all the others.

In terms of ITR (Fig. 6.2), the least significant difference between our "Ensemble-DNN" method and the compared methods is observed with (1) ttf-CCA ($^{**}p = 1.91 \times 10^{-6}$) for $T = 0.2$, (2) ttf-CCA ($^{**}p = 1.38 \times 10^{-7}$) for $T = 0.4$, (3) ttf-CCA ($^{**}p = 5.72 \times 10^{-6}$) for $T = 0.6$, (4) ttf-CCA ($p = 0.18 \times 10^{-1}$) for $T = 0.8$, and (5) ttf-CCA ($p = 0.43$) for $T = 1$. For $T = 0.8$, the difference with ttf-CCA is not significant; but it is highly significant (**) with all the others. For $T = 1.0$, the difference with ttf-CCA is not significant; but it is significant (*) with tt-CCA and highly significant (**) with all the others.

In the case of the BETA dataset: In terms of the accuracy (Fig. 6.3) the least significant difference between our "Ensemble-DNN" method and the compared methods is observed with (1) FBCCA ($^{**}p = 6.89 \times 10^{-21}$) for $T = 0.2$, (2) FBCCA ($^{**}p = 3.21 \times 10^{-22}$) for $T = 0.4$, (3) FBCCA ($^{**}p = 4.96 \times 10^{-17}$) for $T = 0.6$, (4) FBCCA ($^{**}p = 2.03 \times 10^{-11}$) for $T = 0.8$, (5) tt-CCA ($^{**}p = 1.43 \times 10^{-5}$) for $T = 1.0$.

In terms of ITR (Fig. 6.3), the least significant difference between our "Ensemble-DNN" method and the compared methods is observed with (1) Combined-tCCA ($^{**}p = 6.55 \times 10^{-16}$) for $T = 0.2$, (2) Combined-tCCA ($^{**}p = 1.05 \times 10^{-18}$) for $T = 0.4$, (3) tt-CCA ($^{**}p = 1.21 \times 10^{-16}$) for $T = 0.6$, (4) ttf-CCA ($^{**}p = 2.21 \times 10^{-11}$) for $T = 0.8$, and (5) tt-CCA ($^{**}p = 2.48 \times 10^{-6}$) for $T = 1$. In both accuracy and ITR, the difference with all the other compared methods for all T, is always "statistically highly significant".

7. PROPOSED SOLUTION IN THE UNSUPERVISED TRAINING SETTING: A NEW LOSS FUNCTION

This chapter describes our adaptation based method that has been proposed to satisfy both user comfort and high performance. Our adaptation based method adapts the global DNN, defined in the previous chapter (Chapter 6). Because the global model is trained with the previously collected data of the subjects and is transferred to the new user, it does not require labeled data from each new user. This model is then adapted to the new user by utilizing the accumulated unlabeled data. The adaptation is achieved by minimizing the proposed custom loss function, which is detailed in the following. Also, we provide experimental results of the proposed method on the benchmark [14], and the BETA [15] datasets.

7.1 Proposed Loss Function

In this section, we explain our proposed novel loss function for unsupervised adaptation of the global DNN $f_{\mathbf{w}}$ (using similar notation with the previous chapter, we denote the global model as $f_{\mathbf{w}}$, where $\mathbf{w}$ keeps the network parameters) to the new user in the SSVEP based BCI systems. The model is adapted to the new user by minimizing the proposed loss using collected unlabeled data of the new user, in an expectation-maximization (EM) framework fashion. The proposed loss mainly consists of two terms, which are self-adaptation loss (sl) and local-regularity loss (ll). In the following subsections, we explain and give functionalities of the each term separately.

7.1.1 Self-Adaptation Loss (sl)

In the presence of the true labels (y_i) of the new user's data, the model $f_{\mathbf{w}}$ can be adapted to the new user by minimizing standard cross entropy loss, as explained in Chapter 5:

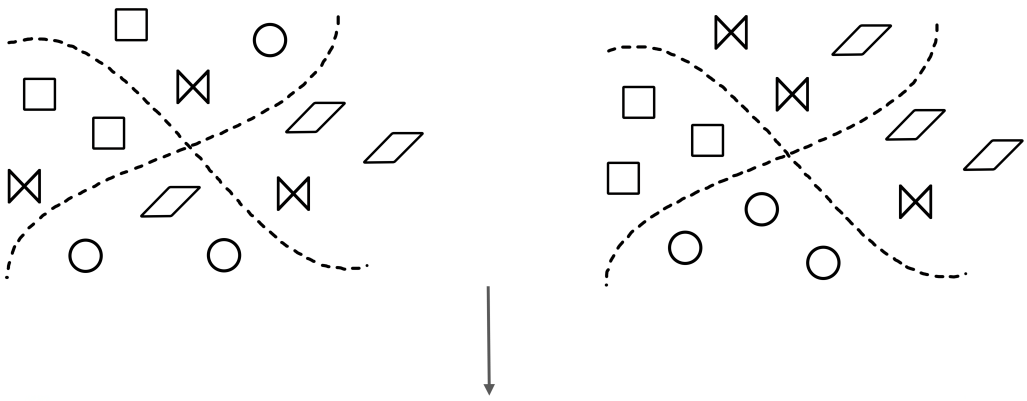
$$-\frac{1}{D_u}\sum_{i=1}^{D_u}\log(\mathbf{s}_{i,y_i}), \tag{7.1}$$

where D_u is the total number of the new user's data (Chapter 3). However, in the unsupervised setting we do not have the true labels, so instead of the true labels, predictions ($\hat{y}_i$) of the model could be used. Then, we can adapt the model $f_{\mathbf{w}}$ to the new user in an unsupervised fashion by minimizing the $-\sum_{j=1}^{D_u}\log(\mathbf{s}_{i,\hat{y}_i})/D_u$. But here, the parameters $\mathbf{w}$ of the model is updated at each iteration t, so the responses of the network are changed over iterations. Let $\mathbf{w}^t$ to be the updated parameters at the iteration t, and let the $\mathbf{s}_i^t$ and y_i^t to be the network soft response and the network prediction respectively, for the input $\mathbf{x}_i$ at the iteration t using the updated network parameters $\mathbf{w}^t$, i.e, $\mathbf{s}_i^t = f_{\mathbf{w}^t}(\mathbf{x}_i)$ and $\hat{y}_i^t = \arg\max_j \mathbf{s}_{i,j}^t$. Therefore, in an EM fashion at each iteration t, the predictions of the model at previous iteration ($\hat{y}_i^{t-1}$) could be used as pseudo-labels, with this idea the self-adaptation loss (sl) becomes:

$$\mathcal{L}_{sl} = -\frac{1}{D_u}\sum_{i=1}^{D_u}\log(\mathbf{s}_{i,\hat{y}_i^{t-1}}^t) \tag{7.2}$$

The effectiveness of minimizing such loss functions using the pseudo-labels have been shown in many unsupervised [55] and semi-supervised [56–58] settings. In effect, minimizing this loss function favors the network to have decisions boundaries between classes that are well separated by minimizing the class overlaps [56]. As discussed in [56], the minimization of this loss is equivalent to "entropy regularization" framework, which aims to get benefit from the unlabeled data in the maximum a posterior scheme, proposed in [57].

Also, the effectiveness of the minimization of sl term (Equation 7.2) in the neural network models with dropout layers [59] (please note that our DNN model also utilizes the dropout layers) may be understood in an another way, as explained in the following. The dropout layer was proposed in [59] to reduce the overfitting. When the dropout layer is used in any model, during the training phase, the model $f_{\mathbf{w}}$ gives the output only using its subpart that means during the training, instead of the full model, only the subpart of the model produces the outputs. However the model $f_{\mathbf{w}}$ gives the response using the whole full model during the test phase, and



$$\mathcal{L}_{total} = \lambda \underbrace{[-\frac{1}{D_u}\sum_{i=1}^{D_u} \log(\mathbf{s}^t_{i,\hat{y}_i^{t-1}})]}_{\mathcal{L}_{sl}} + (1-\lambda) \underbrace{[-\frac{1}{D_u}\sum_{i=1}^{D_u}\frac{1}{k_i}\sum_{i=1}^{k_i} \log(\mathbf{s}^t_{i,\hat{y}^{t-1}_{I_i(j)}})]}_{\mathcal{L}_{ll}}$$

Figure 7.1 A global model's representative decision boundaries is shown on the left. Different shapes are for different classes. After adapting the global model by minimizing our proposed loss $\mathcal{L}_{total}$ that is shown on the bottom, a representative new decision boundaries on the right is presented. Our proposed loss consists of two terms: self-adaptation loss ($\mathcal{L}_{sl}$) and local-regularity loss ($\mathcal{L}_{ll}$). The adapted global model by minimizing our proposed loss achieves *201.15 bits/min* and *145.02 bits/min* ITR results, on the benchmark and BETA datasets, respectively. These results show that our proposed method provides significant ITR performance improvements over the state-of-the-art alternative methods.

the responses produced using the whole model are better than (i.e., has less error) the responses produced using the subpart of the model [59] (cf. Figure 11 in [59]) in general. And, while generating pseudo labels, the whole network is used; but during the minimization of the loss term the subparts of the model are used. By this strategy, because of the mentioned quality difference between the predictions of the model inferred during the training and testing, the model performance is generally increased.

Although minimizing self-adaptation loss is effective, it does not take into account the structure of the data, which could improve the performance; so combining self-adaptation loss with the loss term that exploits the data structure is intuitive. In the following, we explain the proposed local-regularity loss that exploits data structure in detail.

7.1.2 Local-Regularity Loss (ll)

It is intuitive to assume that very correlative instances should have the same label (Fig. 7.1). Actually, similar ideas have been used and its effectiveness has been shown in many different areas, such as metric learning [60, 61], and kernel learning [62]. With this idea, to force the model $f_{\mathbf{w}}$ to output the same predicted target to the correlative/similar instances, we propose the local-regularity loss (ll), which is:

$$\mathcal{L}_{ll} = -\frac{1}{D_u}\sum_{i=1}^{D_u}\frac{1}{k}\sum_{i=1}^{k}\log(\mathbf{s}_{i,\hat{y}_{I_i(j)}}),$$

where I_i is the set of indexes that are sorted in descending order based on the correlation coefficient values, i.e., the index of the most correlative instance to the instance i is $I_i(1)$. The minimization of this loss enforces $f_{\mathbf{w}}$ to give the same predictions to the closest instances. The similarity between the instances is determined via the correlation-coefficient, because it is known that the correlation-coefficient is a suitable metric to measure similarities for the SSVEP signals, as most of the methods explained in Chapter 2 rely on the correlation-coefficient.

In this loss term, actually it is assumed that for each instance, there are k many closest instances among D_u many total instances, and the instance should share the same label with them. However, practically it could not be the case. For example, the new user could intend to spell character 'A' more than character 'B'. Therefore, the number of closest instances should be different for each instance. With this idea, the ll term becomes:

$$\mathcal{L}_{ll} = -\frac{1}{D_u}\sum_{i=1}^{D_u}\frac{1}{k_i}\sum_{i=1}^{k_i}\log(\mathbf{s}_{i,\hat{y}_{I_i(j)}}),$$

where k_i is the number of closest instances to the instance i. Here, the difficulty is to determine k_i properly. However, it is natural to think that the correlation coefficient values between the instance i, and the real neighbours of the instance i (i.e., instances that should share the same label with the instance i) are similar and high compared to the correlation coefficient values between the i^{th} instance, and the other instances. Therefore, there is a significant drop between the highest k_i'th correlation coefficient value and the highest k_i+1'th correlation coefficient value[1]:

$$\frac{\rho(\mathbf{x}_i, \mathbf{x}_{I_i(k_i)}) - \rho(\mathbf{x}_i, \mathbf{x}_{I_i(k_i+1)})}{|\rho(\mathbf{x}_i, \mathbf{x}_{I_i(k_i)})|} \geq \delta, \tag{7.3}$$

where δ is a predetermined threshold. When the drop is higher than the δ, it is

[1] As explained in the previous Chapter, we cannot calculate the correlation coefficient between two instances directly by $\rho(\mathbf{x}_i, \mathbf{x}_j)$, since the instances $\mathbf{x}$ are multidimensional. We calculate this correlation by combining the channels of the instances with a channel combination. However, for ease of explanation, throughout this section, the correlation coefficient calculation is given as $\rho(\mathbf{x}_i, \mathbf{x}_j)$. The details of selecting the channel combination and other implementation details are explained in the "Implementation Details" section (Section 7.3).

considered significant, and we take the instances as the actual neighbours of the instance i, until this significant drop occurs.

Also, as mentioned in the sl term, the network is updated iteration by iteration, so its responses may change over iterations. With taking into account the iteration, the ll term becomes:

$$\mathcal{L}_{ll} = -\frac{1}{D_u}\sum_{i=1}^{D_u}\frac{1}{k_i}\sum_{i=1}^{k_i}\log(\mathbf{s}^t_{i,\hat{y}^{t-1}_{I_i(j)}}).$$

Both self-adaptation and local-regularity losses have unique functionalities, and it is not easy to determine which one to use for the new user as a prior. Therefore, we use both terms together, giving each of them a unique weight. In the following subsection, this proposed total less term is explained.

7.1.3 Total Loss

As a total loss term, we use the combination of the self-adaptation and local-regularity losses, weighting them as follows:

$$\mathcal{L}_{total} = \lambda\mathcal{L}_{sl} + (1-\lambda)\mathcal{L}_{ll},$$

where $\lambda \in [0,1]$ term controls the importance of the loss terms. As a prior, it is almost impossible to determine the best λ value for each new user, so we choose the λ value empirically, as detailed in the following. Firstly, we adapt the $f_{\mathbf{w}}$ to the new user by minimizing $\mathcal{L}_{total}$ using different values of the λ. The resulted updated network parameters for the specific λ value, are shown as $\mathbf{w}(\lambda)$. Similarly, the model predictions are shown as $\hat{y}_i(\lambda)$. Then, to decide which updated network parameters $\mathbf{w}(\lambda)$ are to be used, we check how well each updated parameter clusters the new user's unlabeled data. Since we could not have the true labels in the unsupervised setting, we cannot choose which updated network parameters to use directly by checking the accuracy performance of the updated parameters, so, we choose the parameters by checking the clustering performances of the updated network parameters.

To measure the clustering performance, "silhoutte" clustering metric [63] is used. In this metric, for each instance $\mathbf{x}_i$ there is a silhouette score $m_i(\lambda)$[2] that measures how well i^{th} instance is clustered based on a difference between the instance tightness to

[2]For the $m_i(\lambda)$ similar notation with $\hat{y}_i(\lambda)$ is used.

its own cluster and the instance separation to the other clusters, which is:

$$
\begin{aligned}
a_i(\lambda) &= \frac{1}{q_{\hat{y}_i}(\lambda)-1} \sum_{\substack{j=1 \\ j\neq i}}^{D_u} d(\mathbf{x}_j,\mathbf{x}_i)\mathbb{1}(\hat{y}_j(\lambda)=\hat{y}_i(\lambda)) \\
b_i(\lambda) &= \min_{l\in\{1,\cdots,M\}\setminus\{\hat{y}_i(\lambda)\}} \frac{1}{q_l(\lambda)} \sum_{\substack{j=1 \\ j\neq i}}^{D_u} d(\mathbf{x}_j,\mathbf{x}_i)\mathbb{1}(\hat{y}_j(\lambda)=l) \\
m_i(\lambda) &= \frac{b_i(\lambda)-a_i(\lambda)}{\max(a_i(\lambda),b_i(\lambda))},
\end{aligned}
\tag{7.4}
$$

where $\mathbb{1}(.)$ is the indicator function that equals to 1 if the condition is satisfied, otherwise 0; $q_l(\lambda)$ is the total number of the instances labeled as l by the DNN with the parameters $\mathbf{w}(\lambda)$ (i.e. $q_l(\lambda)=\sum_{j=1}^{D_u}\mathbb{1}(\hat{y}_j(\lambda)=l)$); $d(\mathbf{x}_j,\mathbf{x}_i)$ is the distance metric, for this we use $d(\mathbf{x}_j,\mathbf{x}_i)=1-\rho(\mathbf{x}_j,\mathbf{x}_i)$, and it equals to cosine distance if $\mathbf{x}_i$'s are the zero mean signals; $a_i(\lambda)$ is the average distance of the i^{th} instance to the other instances that share same predicted labels ($a_i(\lambda)$ equals to 1, if $q_{\hat{y}_i}(\lambda)=1$); $b_i(\lambda)$ is the minimum average distance of the i^{th} instance to the instances that are in other cluster, minimization is done over the other clusters. Then, the overall silhouette clustering score for the predictions of the model with the parameters $\mathbf{w}(\lambda)$ is the average of the silhouette scores of all the instances (i.e. $m(\lambda)=\sum_{i=1}^{D_u} m_i(\lambda)/D_u$). The network parameters $\mathbf{w}(\lambda)$ with the associated highest silhouette clustering score $m(\lambda)$ is selected to be used.

Moreover, prediction confidence has been shown to be useful in the unsupervised learning literature [64], [65]. We also take the confidence into account because each instance is clustered with a different degree of confidence; while some of them are clustered well, the others do not. With this idea, we only consider the instances clustered well. For this, we use the pseudo-labels of the instances with positive silhouette score m_i^{λ}, since if the instance has a negative silhouette score m_i^{λ}, there is a cluster, which is on average closer to this instance than the cluster that this instance belongs to. Because of this fact, the instances having negative silhouette scores are more likely to be misclassified, and they are more likely to be classified differently as the DNN's weights are updated, so to avoid updating the DNN using the pseudo-labels that are very likely to be wrong, we update the DNN using only the pseudo-labels of the instances having positive silhouette score. Along with the confidence detail, for all the implementation details, we kindly refer readers to the "Implementation Details" section.

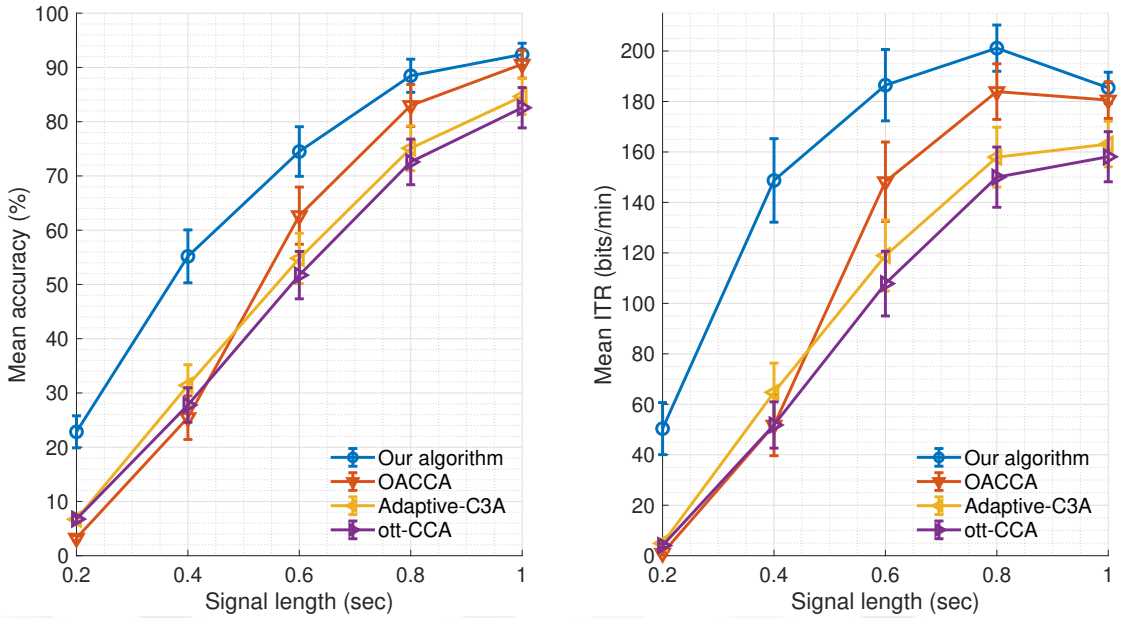


Figure 7.2 The mean classification accuracy on the left and the mean information transfer rate (ITR) on the right are presented across all 35 subjects in the benchmark dataset, together with the standard errors indicated by the bars.

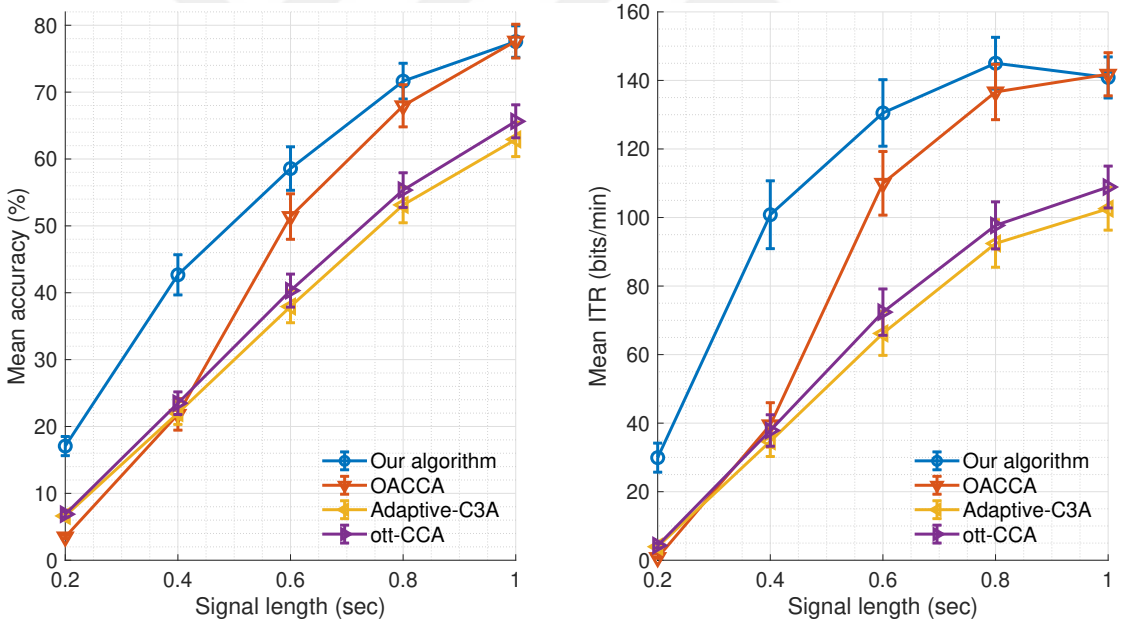


Figure 7.3 The mean classification accuracy on the left and the mean information transfer rate (ITR) on the right are presented across all 70 subjects in the BETA dataset, together with the standard errors indicated by the bars.

7.2 Performance Evaluations

We compare our method with three state-of-the-art alternatives: OACCA, adaptive-C3A, and ott-CCA methods on the benchmark [14] and the BETA [15] datasets.

The performance evaluation is conducted in a leave-one-subject-out fashion. More specifically, in each dataset separately, one subject is reserved and considered as the new user, the remaining subjects' data are used to generate the subject independent model, and this model is adapted to the reserved subject using all the data of the reserved subject in an unsupervised manner to test our model. The adaptation is continued until the loss is converged (please check the "Implementation Details" Section (Section 7.3) for further information). For the other methods, the same procedure is applied, but the adaptation is continued until all predictions of the methods stay the same in two consecutive iterations for a fair comparison (since in the respective studies of the compared methods, there is no notion of loss or convergence). Meaning, the compared methods are implemented and the same initial input is given to them to receive their first predictions in the first iteration. These predictions of the methods are kept as pseudo-labels and the updated labels are given to those method in the next iteration. This process is continued until the pseudo-labels are no longer updated in the new iteration. Therefore, every method compared in this paper have a unique number of iterations as it has been decided according to their state of convergences. And, the methods' performances on the reserved subject are evaluated on all the data of the reserved subject as done in our method. This process is repeated for each subject and the mean classification accuracy and ITR (in the calculation of the ITR, a 0.5 seconds gaze shift time that exists in both datasets is taken into account) with the standard errors for the signal duration T in the range $T \in \{0.2, 0.4, \ldots, 1.0\}$. It can also be noted that the predetermined 9 channels (Pz, PO3, PO5, PO4, PO6, POz, O1, Oz, and O2) have been used from the occipital and parietal regions in our method, and all the compared methods.

7.2.1 Results and Discussion

In this subsection, while comparing the methods, we only report the ITR results, as it is the primary performance metric and a function of the accuracy. However, we only report the accuracy results for the other experimental results since it is more intuitive and easier to understand.

The results show that our method significantly outperforms other methods in every signal length tested, especially in lower signal lengths (i.e., 0.2, 0.4, 0.6 seconds). For instance, in 0.4 seconds, we achieve 148.72 bits/min and 100.81 bits/min ITR for benchmark and BETA datasets, respectively, whereas the closest performance

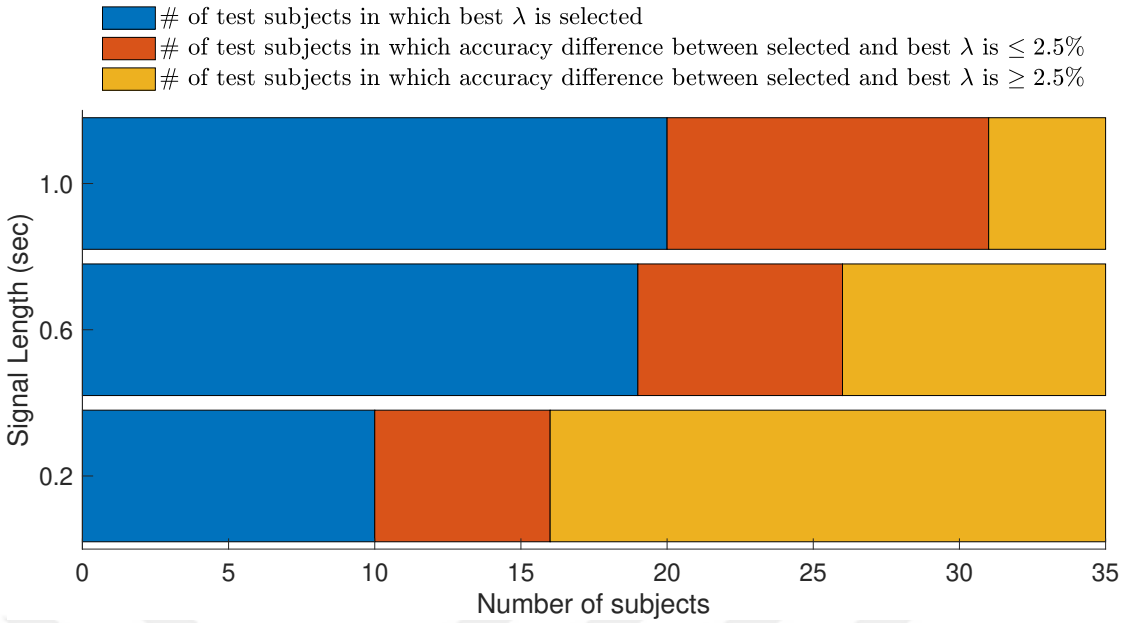


Figure 7.4 The mean classification accuracy on the left and the mean information transfer rate (ITR) on the right are presented across all 70 subjects in the BETA dataset, together with the standard errors indicated by the bars.

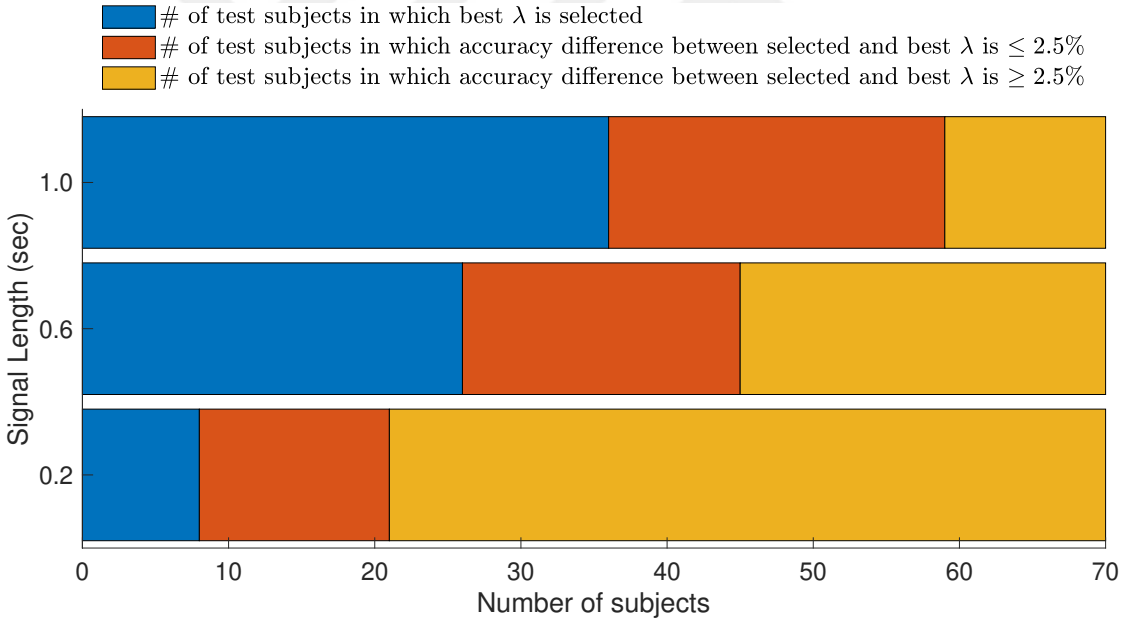


Figure 7.5 The mean classification accuracy on the left and the mean information transfer rate (ITR) on the right are presented across all 70 subjects in the BETA dataset, together with the standard errors indicated by the bars.

among compared methods is 64.72 bits/min (Adaptive-C3A) for benchmark dataset and 39.50 bits/min (OACCA) for BETA dataset. Thus, these results suggest that we investigated a significant improvement in the literature for short signal lengths in an unsupervised setting.

In addition, the highest ITR among all the methods and signal lengths is in 0.8

seconds, obtained with our method for both benchmark and BETA datasets. The proposed method achieves 201.15 bits/min ITR in benchmark dataset and 145.02 bits/min ITR in BETA dataset. The closest performance to ours, considering all signal lengths, is obtained by the OACCA method, which achieves 183.92 bits/min in 0.8 seconds for benchmark and 141.80 bits/min in 1 second for BETA dataset. This shows that, for BETA dataset, the peak of ITR among compared methods is reported in 1 seconds, whereas our method outperforms this ITR in 0.8 seconds.

The proposed algorithm outperforms other methods in every signal length tested with respect to ITR as it can be seen in Fig. 7.2 and Fig. 7.3.

The performance of our method with respect to the proposed λ selection has been tested, and the results are shown in Fig. 7.4 and Fig. 7.5, for the benchmark and the BETA datasets, respectively. We adapt the global DNN with six different λ values (0,0.2,0.4, ..., 1), for every individual test subject. The 'best λ' is the λ value giving the highest performance of the model after the training. We have chosen subjects' λ values according to the overall silhouette score as explained in Section 7.1.3. There are three bins with the number of subjects that falls in the described performance interval with respect to their λ values. The subjects fall in the first bin if the best λ is selected. The subjects fall in the second bin when their best λ's accuracy performance is more 2.5% than the selected λ's accuracy performance. Lastly, subjects fall in the third bin if their selected λ gives less than 2.5% accuracy compared to the best λ value possible. The reason for using 2.5% accuracy in the comparison is that it is the trivial accuracy, i.e., $(1/40 = 2.5\%)$.

In the benchmark dataset, for 10, 19 and 20 test subjects, the best λ is selected for 0.2, 0.6 and 1 seconds respectively. In the BETA dataset, for 8, 26 and 36 subjects, the best λ is selected for 0.2, 0.6 and 1 seconds respectively. Also, the λ values selected that cause more than 2.5% performance loss is the majority (49 for BETA and 16 for Benchmark) among subjects in 0.2 seconds for both datasets. It can be observed that as the signal length increases, the number of subjects that have the best λ selection increases. The reason is that since the SNR value increase for longer signals, the used silhouette clustering metric is more reliable. Therefore, the decided λ value for each subject is more promising for higher signal lengths as expected. This is also explaining our λ selection strategy performance difference on the benchmark and the BETA datasets, as the collected signals in the BETA dataset are more noisier.

7.2.2 Statistical Significance Analyses

Our statistical significance test results are presented in this section. We follow same procedure explained in previous two Chapters' "Statistical Significance Analyses" sections. Namely, for each $T \in \{0.2, 0.4, 0.6, 0.8, 1\}$, we conduct 3 paired t-tests, pairing our proposed adaptation algorithm with the compared methods in Fig. 7.2 and Fig. 7.3. Unadjusted p-values are reported, and we call the observed difference as "statistically significant" (*) if the p-value is less than $\frac{0.05}{3}$ and "statistically highly significant" (**) if the p-value is less than $\frac{0.05}{3\times5}$. For the "statistically significant" case, "single" Bonferroni correction is applied by dividing 0.05 by 1/3, since for each T there are 3 comparisons. And for the "statistically highly significant" case, we apply "double" Bonferroni correction by 1/15, since across all methods and T choices there are 15 comparisons.

In the case of the benchmark dataset: In terms of the accuracy (Fig. 7.2) the least significant difference between our algorithm and the compared methods is observed with (1) ott-CCA method (**$p = 3.49 \times 10^{-7}$) for $T = 0.2$, (2) OACCA method (**$p = 2.26 \times 10^{-7}$) for $T = 0.4$, (3) OACCA method (*$p = 0.94 \times 10^{-2}$) for $T = 0.6$, (4) OACCA method (*$p = 0.01$) for $T = 0.8$, (5) OACCA method ($p = 0.07$) for $T = 1.0$. For $T = 1.0$, the difference with OACCA is not significant; but highly significant (**) with all the others.

In terms of ITR (Fig. 7.2), the least significant difference between our algorithm and the compared methods is observed with (1) ott-CCA (**$p = 3.40 \times 10^{-5}$) for $T = 0.2$, (2) OACCA (**$p = 6.06 \times 10^{-8}$) for $T = 0.4$, (3) OACCA (**$p = 0.32 \times 10^{-2}$) for $T = 0.6$, (4) OACCA (*$p = 0.44 \times 10^{-2}$) for $T = 0.8$, and (5) OACCA ($p = 0.09$) for $T = 1$. For $T = 1.0$, the difference with OACCA is not significant; but highly significant (**) with all the others.

In the case of the BETA dataset: In terms of the accuracy (Fig. 7.3) the least significant difference between our algorithm and the compared methods is observed with (1) ott-CCA (**$p = 2.77 \times 10^{-12}$) for $T = 0.2$, (2) OACCA (**$p = 1.32 \times 10^{-11}$) for $T = 0.4$, (3) OACCA (*$p = 0.51 \times 10^{-2}$) for $T = 0.6$, (4) OACCA ($p = 0.02$) for $T = 0.8$, (5) OACCA ($p = 0.96$) for $T = 1.0$. For $T = 0.8$ and $T = 1.0$, the difference with OACCA is not significant; but highly significant (**) with all the others.

In terms of ITR (Fig. 7.3), the least significant difference between our algorithm and the compared methods is observed with (1) Adaptive-C3A (**$p = 3.60 \times 10^{-9}$) for $T = 0.2$, (2) OACCA (**$p = 5.38 \times 10^{-10}$) for $T = 0.4$, (3) OACCA (**$p = 0.33 \times 10^{-2}$) for $T = 0.6$, (4) OACCA ($p = 0.02$) for $T = 0.8$, and (5) OACCA ($p = 0.76$)

for $T = 1$. For $T = 0.8$ and $T = 1.0$, the difference with OACCA is not significant; but highly significant (**) with all the others.

7.3 Implementation Details

In this section, we explain the details of the the neighbour selection, confidence of the instances as well as the convergence check of the loss function.

7.3.1 Neighbour Selection & Convergence Check

As described in the first footnote of this Chapter, the instances $\mathbf{x}_i$'s are multidimensional so that the correlation coefficient between two instances can not be calculated directly by $\rho(\mathbf{x}_i, \mathbf{x}_j)$. We calculate this correlation coefficient by combining the channels of the instances with the channel combination weight $\mathbf{w}_c^{(*)} \in \mathbb{R}^{C \times 1}$ as follows:

$$\rho((\mathbf{w}_c^{(*)})'\mathbf{x}_i, (\mathbf{w}_c^{(*)})'\mathbf{x}_j), \tag{7.5}$$

where $(\mathbf{w}_c^{(*)})'$ is the transpose of $\mathbf{w}_c^{(*)}$. Hence, all the equations containing the correlation coefficient term are dependent of $\mathbf{w}_c^{(*)}$, including the terms at equation (7.4), in which the distance term ($d(\mathbf{x}_j, \mathbf{x}_i)$) rely on $\mathbf{w}_c^{(*)}$. Consequently, the silhouette score of the instance $m_i(\lambda)$ and the overall silhouette score $m(\lambda)$ are also dependent of $\mathbf{w}_c^{(*)}$, we denote this dependency by $m_i(\lambda, \mathbf{w}_c^{(*)})$, $m(\lambda, \mathbf{w}_c^{(*)})$, respectively. The channel combination $\mathbf{w}_c^{(*)}$ is selected to maximize the overall silhouette score for the given predictions; since, it is intuitive to consider that a channel combination, which maximizes the metric performance (i.e., silhouette score), also describes the user's spatial characteristic well. However, the direct maximization of this score is intractable. Therefore, by following a different strategy, we select the channel combination $\mathbf{w}_c^{(*)}$ that maximizes the overall silhouette score $m(\lambda, \mathbf{w}_c^{(*)})$ among the channel combination weights from the second layer (i.e., channel combination layer) of the DNN, which are expected to characterize optimum channel combinations' characteristic of the user, as follows:

$$\mathbf{w}_c^{(*)} = \underset{\mathbf{w}_c^{(i)} \in \mathbf{w}_c(\lambda)}{\arg\max} \frac{1}{N} \sum_{i=1}^{N} m_i(\lambda, \mathbf{w}_c^{(i)}). \tag{7.6}$$

As detailed in the Chapter 5 the channel combination layer comes after the sub-bands combination layer, so, actually the weights in the channel combination layer learned/tuned for the inputs (of this layer), whose sub-bands of harmonics (generated in the preprocessing step) are combined. Therefore, to effectively use the selected channel combination weights and to take advantage of the preprocessing step in the neighbours and silhouette score $\mathbf{w}_c^{(*)}$ calculation as well, we here also follow similar approach explained in the Chapter 6 as detailed in the next. We first preprocess the input $\mathbf{x}$, then we combine resulted sub-bands of harmonics with the weights from the sub-bands combination layers, and produce $\underline{\mathbf{x}}$ that is used in the calculation of the silhouette score as well as in the determination of the instances' neighbours.

Also, note that the weights are updated over iterations, so the channel and sub-bands combination weights are updated, which affects the neighbours of the instances calculation. Because of that the neighbours of the instances must be recalculated at each iteration. Therefore, at each iteration t, firstly the channel combination is selected to maximize the silhouette score $m(\lambda, \mathbf{w}_c^{(*)})$ for the given predictions. Then, with the selected channel combination the neighbours of the instances are determined at the iteration t, which is denoted by I_i^t.

Since at each iteration, the silhouette score is calculated, it could be used for the convergence check as well. With this idea, we stop the adapting the DNN, as the overall silhouette score is converged. The overall silhouette score, at iteration t is denoted by $m^t(\lambda, \mathbf{w}_c^{(*)})$. Also note that there is no guarantee that the overall silhouette score will get better at each iteration. Therefore, we compare the current overall silhouette score $m^t(\lambda, \mathbf{w}_c^{(*)})$ with the silhouette score of the previous iteration $m^{t-1}(\lambda, \mathbf{w}_c^{(*)})$, and if the current score is lower, we return the updated weights $\mathbf{w}^t$ to its previous state $\mathbf{w}^{t-1}$, then update it again. Because of the randomness coming from the dropout layers, the weights most probably converge to a different state at the new try, and if the network is not converged to a better state at three consecutive tries, we terminate the adaptation with assuming that the network has already converged.

7.3.2 Confidence of the Instances

In the loss term, we use the labels of the instances having the positive silhouette score as explained in the main text. However, with this strategy, there is a probability that either the self-adaptation term or the local-regularity term becomes void for an instance $\mathbf{x}_i$ at any iteration t. If the instance itself has the negative silhouette score (i.e., $m_i^t(\lambda, \mathbf{w}_c^{(*)}) < 0$), the self-adaptation term becomes void; and if all the determined instance's neighbours have the negative silhouette score (i.e., $m_j^t(\lambda, \mathbf{w}_c^{(*)}) < 0$, $\forall j \in I_i^t$), the local-regularity term becomes void. Therefore, to unable the instances contributing equally to the loss term, if one of the term becomes void for the instance x_i we update the λ term for that instance, as follows:

$$\mathcal{L}_{total} = \frac{-1}{N}\sum_{i=1}^{N}\lambda_i^t \log(\mathbf{s}^t_{i,\hat{y}_i^{t-1}}) + \frac{-1}{N}\sum_{i=1}^{N}[\frac{1-\lambda_i^t}{k_i}\sum_{i=1}^{k_i}\log(\mathbf{s}^t_{i,\hat{y}^{t-1}_{I_i(j)}})], \tag{7.7}$$

where λ_i^t is the instance specific λ term at the iteration t and it equals to the global λ, if the both terms are effective; and it equals to 0 or 1, if one of the term is void:

$$\lambda_i^t = \begin{cases} 1 & \text{if } m_j^t(\lambda, \mathbf{w}_c^{(*)}) < 0,\ \forall j \in I_i^t \\ 0 & \text{if } m_i^t(\lambda, \mathbf{w}_c^{(*)}) < 0 \\ \lambda & \text{otherwise} \end{cases}. \tag{7.8}$$

It is also possible to have that the both terms are void for some instances, in those cases we do not use these instances in the loss term.

7.3.3 Initial Predictions

We have stated that the adaptation is started by the predictions of the transferred global model. However, the global model initial performance could not be satisfactory for some users, especially when the training subjects' statistics are much different than the new user's data statistic as explained in the previous Chapter (cf. Fig 6.5). In those cases, the completely training free algorithms, such as FBCCA, could perform better than the global model. Because of this fact, we start the

adaptation either using the transferred global model's predictions or the FBCCA method's predictions. We choose which one to use by measuring their silhouette scores, and we start the adaptation by selecting the one having higher silhouette score.

7.3.4 Other Information

As the global models, we have used the global models trained in the previous Chapter. Hence, in here also, we have used our suggested setting in the DNN architecture: 3 sub-bands and 9 channels (Pz, PO3, PO5, PO4, PO6, POz, O1, Oz, O2). We minimize our proposed loss function via gradient descent with the 0.0001 learning rate. We set the threshold δ, which is used in the determination of the instances' neighbours, as $\delta = 0.05$.

8. CONCLUSION

In this thesis, we study the target identification of BCI SSVEP spellers, which is a multi-class classification problem with 40 classes where the goal is to classify the SSVEP signal received through EEG with minimum possible signal duration so that ITR is maximized. To this end, we proposed a novel DNN architecture that consists of 4 convolutional (sub-band and channel combinations as well as downsampling and filtering in time) and 1 fully connected layers. The proposed DNN strongly outperforms the state-of-the-art as well as the most recently proposed techniques in the literature on two publicly available large scale benchmark and BETA datasets. We achieve ITRs with only 0.4 seconds of stimulation: 265.23 bits/min on the benchmark and 196.59 bits/min on the BETA dataset. To our best knowledge, these are the highest (and significantly larger than the nearest competitor) performance results ever reported on these datasets.

In addition to ITR maximization, there are other classification goals, such as user comfort, which are needed to be considered. Even though our proposed DNN achieves the state-of-the-art ITR performance results, the two-stage training of our DNN requires labeled data from each new user. To collect these labeled data, the long and tiring EEG experiments are conducted that are usually uncomfortable and burdensome to the user. With prioritizing the user comfort, we also proposed an ensemble based classification method using our proposed DNN architecture as a base classifier. Our proposed ensemble based method trains multiple subject specific DNNs using the pre-existing data of various subjects and combines these DNNs to predict the target character during a spelling session. Thus, our method does not require any additional training data from a new user who wants to use the system without much hassle. Our ensemble based method achieves 155.51 bits/min and 114.36 bits/min maximum ITR results, on the benchmark dataset and the BETA dataset, respectively. These results demonstrated that the proposed method achieves significant improvements in the accuracy and ITR performances compared to other alternative techniques.

Although our ensemble based method's target identification performance is much

better than the alternative methods, the performance difference between our proposed DNN trained with the two-stage training strategy that uses user specific labeled data and our ensemble based method reveals the need and importance of user-specific adaptation. To satisfy both target identification and user-comfort goals, we have also proposed another method that adapts the DNN model to the user by utilizing unlabeled data of the user. Our unsupervised adaptation based method firstly train the global DNN model using previously collected subjects data, and then transfer that DNN to the user. Transferred DNN is adapted to the user in an unsupervised fashion by minimizing our proposed custom loss function utilizing the accumulated unlabeled data of the user. This loss function consists of two terms (self-adaptation loss and local-regularity loss), each of which has unique functionalities. With this approach, we achieve 201.15 bits/min ITR on benchmark dataset and 145.02 bits/min ITR on BETA dataset. Thus, our unsupervised adaptation based method also provides significant ITR performance improvements compared to the alternative techniques.

BIBLIOGRAPHY

[1] E. Yin *et al.*, "A dynamically optimized ssvep brain–computer interface (bci) speller," *IEEE Transactions on Biomedical Engineering*, vol. 62, no. 6, pp. 1447–1456, 2015.

[2] S. Gao *et al.*, "Visual and auditory brain–computer interfaces," *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 5, pp. 1436–1447, 2014.

[3] A. Kreilinger *et al.*, "Single versus multiple events error potential detection in a bci-controlled car game with continuous and discrete feedback," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 3, pp. 519–529, 2016.

[4] A. J. Westerveld, , *et al.*, "A damper driven robotic end-point manipulator for functional rehabilitation exercises after stroke," *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 10, pp. 2646–2654, 2014.

[5] H. Nezamfar, S. S. Mohseni Salehi, M. Moghadamfalahi, and D. Erdogmus, "Flashtype$^{\text{tm}}$: A context-aware c-vep-based bci typing interface using eeg signals," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 5, pp. 932–941, 2016.

[6] H. Cecotti and A. Graser, "Convolutional neural networks for p300 detection with application to brain-computer interfaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 433–445, 2011.

[7] M. Nakanishi, Y. Wang, X. Chen, Y. Wang, X. Gao, and T. Jung, "Enhancing detection of ssveps for a high-speed brain speller using task-related component analysis," *IEEE Transactions on Biomedical Engineering*, vol. 65, no. 1, pp. 104–112, 2018.

[8] Y. Zhang, E. Yin, F. Li, Y. Zhang, T. Tanaka, Q. Zhao, Y. Cui, P. Xu, D. Yao, and D. Guo, "Two-stage frequency recognition method based on correlated component analysis for ssvep-based bci," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 26, no. 7, pp. 1314–1323, 2018.

[9] A. M. Norcia, L. G. Appelbaum, J. M. Ales, B. R. Cottereau, and B. Rossion, "The steady-state visual evoked potential in vision research: A review," *Journal of Vision*, vol. 15, pp. 4–4, 05 2015.

[10] M. Bittencourt-Villalpando and N. M. Maurits, "Stimuli and feature extraction algorithms for brain-computer interfaces: A systematic comparison," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 26, no. 9, pp. 1669–1679, 2018.

[11] X. Chen, Y. Wang, M. Nakanishi, X. Gao, T.-P. Jung, and S. Gao, "High-speed spelling with a noninvasive brain-computer interface," *Proceedings of the National Academy of Sciences*, vol. 112, no. 44, pp. E6058–E6067, 2015.

[12] A. Rezeika, M. Benda, P. Stawicki, F. Gembler, A. Saboor, and I. Volosyak, "Brain–computer interface spellers: A review," *Brain Sciences*, vol. 8, no. 4, 2018.

[13] L. R. Hochberg and J. P. Donoghue, "Sensors for brain-computer interfaces," *IEEE Engineering in Medicine and Biology Magazine*, vol. 25, no. 5, pp. 32–38, 2006.

[14] Y. Wang, X. Chen, X. Gao, and S. Gao, "A benchmark dataset for ssvep-based brain–computer interfaces," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 10, pp. 1746–1752, 2017.

[15] B. Liu, X. Huang, Y. Wang, X. Chen, and X. Gao, "Beta: A large benchmark database toward ssvep-bci application," *Frontiers in Neuroscience*, vol. 14, 2020.

[16] W. Yijun, W. Ruiping, G. Xiaorong, and G. Shangkai, "Brain-computer interface based on the high-frequency steady-state visual evoked potential," *International Conference on Neural Interface and Control*, pp. 37–39, 2005.

[17] O. Friman, I. Volosyak, and A. Graser, "Multiple channel detection of steady-state visual evoked potentials for brain-computer interfaces," *IEEE Transactions on Biomedical Engineering*, vol. 54, no. 4, pp. 742–750, 2007.

[18] Z. Lin, C. Zhang, W. Wu, and X. Gao, "Frequency recognition based on canonical correlation analysis for ssvep-based bcis," *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 12, pp. 2610–2614, 2006.

[19] W. Nan, C. M. Wong, B. Wang, F. Wan, P. U. Mak, P. I. Mak, and M. I. Vai, "A comparison of minimum energy combination and canonical correlation

analysis for ssvep detection," *International IEEE/EMBS Conference on Neural Engineering*, pp. 469–472, 2011.

[20] X. Chen, Y. Wang, S. Gao, T.-P. Jung, and X. Gao, "Filter bank canonical correlation analysis for implementing a high-speed ssvep-based brain–computer interface," *Journal of Neural Engineering*, vol. 12, no. 4, p. 046008, 2015.

[21] F.-C. Lin, J. K. Zao, K.-C. Tu, Y. Wang, Y.-P. Huang, C.-W. Chuang, H.-Y. Kuo, Y.-Y. Chien, C.-C. Chou, and T.-P. Jung, "Snr analysis of high-frequency steady-state visual evoked potentials from the foveal and extrafoveal regions of human retina," *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 1810–1814, 2012.

[22] C. M. Wong, F. Wan, B. Wang, Z. Wang, W. Nan, K. F. Lao, P. U. Mak, M. I. Vai, and A. Rosa, "Learning across multi-stimulus enhances target recognition methods in SSVEP-based BCIs," *Journal of Neural Engineering*, vol. 17, no. 1, p. 016026, 2020.

[23] R. Zerafa, T. Camilleri, O. Falzon, and K. P. Camilleri, "To train or not to train? a survey on training of feature extraction methods for SSVEP-based BCIs," *Journal of Neural Engineering*, vol. 15, no. 5, p. 051001, 2018.

[24] M. Krauledat, M. Tangermann, B. Blankertz, and K.-R. Müller, "Towards zero training for brain-computer interfacing," *PLOS ONE*, vol. 3, no. 8, pp. 1–12, 2008.

[25] Y. Wang, M. Nakanishi, Y.-T. Wang, and T.-P. Jung, "Enhancing detection of steady-state visual evoked potentials using individual training data," *Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 3037–3040, 2014.

[26] Y. Zhang, G. Zhou, J. Jin, M. Wang, X. Wang, and A. Cichocki, "L1-regularized multiway canonical correlation analysis for ssvep-based bci," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 21, no. 6, pp. 887–896, 2013.

[27] G. Bin, X. Gao, Y. Wang, Y. Li, B. Hong, and S. Gao, "A high-speed BCI based on code modulation VEP," *Journal of Neural Engineering*, vol. 8, no. 2, p. 025015, 2011.

[28] Y. Zhang, G. Zhou, Q. Zhao, A. Onishi, J. Jin, X. Wang, and A. Cichocki, "Multiway canonical correlation analysis for frequency components recognition

in ssvep-based bcis," *International Conference on Neural information processing*, pp. 287–295, 2011.

[29] Y. Zhang, G. Zhou, J. Jin, X. Wang, and A. Cichocki, "Frequency recognition in ssvep-based bci using multiset canonical correlation analysis," *International journal of neural systems*, vol. 24, no. 04, p. 1450013, 2014.

[30] J. Pan, X. Gao, F. Duan, Z. Yan, and S. Gao, "Enhancing the classification accuracy of steady-state visual evoked potential-based brain-computer interfaces using phase constrained canonical correlation analysis," *Journal of Neural Engineering*, vol. 8, no. 3, p. 036027, 2011.

[31] P. Poryzala and A. Materka, "Cluster analysis of cca coefficients for robust detection of the asynchronous ssveps in brain–computer interfaces," *Biomedical Signal Processing and Control*, vol. 10, pp. 201 – 208, 2014.

[32] M. Nakanishi, Y. Wang, Y.-T. Wang, Y. Mitsukura, and T.-P. Jung, "A high-speed brain speller using steady-state visual evoked potentials," *International Journal of Neural Systems*, vol. 24, no. 06, p. 1450019, 2014.

[33] M. Nakanishi, Y. Wang, Y.-T. Wang, and T.-P. Jung, "A comparison study of canonical correlation analysis based methods for detecting steady-state visual evoked potentials," *PLOS ONE*, vol. 10, no. 10, pp. 1–18, 2015.

[34] Y. Zhang, E. Yin, F. Li, Y. Zhang, D. Guo, D. Yao, and P. Xu, "Hierarchical feature fusion framework for frequency recognition in ssvep-based bcis," *Neural Networks*, vol. 119, pp. 1 – 9, 2019.

[35] N.-S. Kwak, K.-R. Müller, and S.-W. Lee, "A convolutional neural network for steady state visual evoked potential classification under ambulatory environment," *PLOS ONE*, vol. 12, no. 2, pp. 1–20, 2017.

[36] G. Bressan, G. Cisotto, G. R. Müller-Putz, and S. C. Wriessnegger, "Deep learning-based classification of fine hand movements from low frequency eeg," *Future Internet*, vol. 13, no. 5, p. 103, 2021.

[37] J. Thomas, T. Maszczyk, N. Sinha, T. Kluge, and J. Dauwels, "Deep learning-based classification for brain-computer interfaces," *IEEE International Conference on Systems, Man and Cybernetics*, pp. 234–239, 2017.

[38] N. K. Nik Aznan, S. Bonner, J. Connolly, N. Al Moubayed, and T. Breckon, "On the classification of ssvep-based dry-eeg signals via convolutional neural networks," *IEEE International Conference on Systems, Man and Cybernetics*, pp. 3726–3731, 2018.

[39] N. Waytowich, V. J. Lawhern, J. O. Garcia, J. Cummings, J. Faller, P. Sajda, and J. M. Vettel, "Compact convolutional neural networks for classification of asynchronous steady-state visual evoked potentials," *Journal of Neural Engineering*, vol. 15, no. 6, p. 066031, 2018.

[40] A. Ravi, N. H. Beni, J. Manuel, and N. Jiang, "Comparing user-dependent and user-independent training of CNN for SSVEP BCI," *Journal of Neural Engineering*, vol. 17, no. 2, p. 026028, 2020.

[41] Y. Li, J. Xiang, and T. Kesavadas, "Convolutional correlation analysis for enhancing the performance of ssvep-based brain-computer interface," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 28, no. 12, pp. 2681–2690, 2020.

[42] P. Yuan, X. Chen, Y. Wang, X. Gao, and S. Gao, "Enhancing performances of ssvep-based brain–computer interfaces via exploiting inter-subject information," *Journal of Neural Engineering*, vol. 12, no. 4, 2015.

[43] N. R. Waytowich, J. Faller, J. O. Garcia, J. M. Vettel, and P. Sajda, "Unsupervised adaptive transfer learning for steady-state visual evoked potential brain-computer interfaces," *IEEE International Conference on Systems, Man, and Cybernetics*, pp. 004135–004140, 2016.

[44] K. F. Lao, C. M. Wong, Z. Wang, and F. Wan, "Learning prototype spatial filters for subject-independent ssvep-based brain-computer interface," *IEEE International Conference on Systems, Man, and Cybernetics*, pp. 485–490, 2018.

[45] C. M. Wong, Z. Wang, M. Nakanishi, B. Wang, A. Rosa, P. Chen, T.-P. Jung, and F. Wan, "Online adaptation boosts ssvep-based bci performance," *IEEE Transactions on Biomedical Engineering*, pp. 1–1, 2021.

[46] K. F. Lao, C. M. Wong, Z. Wang, and F. Wan, "Learning prototype spatial filters for subject-independent ssvep-based brain-computer interface," in *IEEE International Conference on Systems, Man, and Cybernetics*, pp. 485–490, 2018.

[47] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan, "Brain–computer interfaces for communication and control," *Clinical Neurophysiology*, vol. 113, no. 6, pp. 767 – 791, 2002.

[48] O. B. Guney, M. Oblokulov, and H. Ozkan, "A deep neural network for ssvep-based brain-computer interfaces," *IEEE Transactions on Biomedical Engineering*, vol. 69, no. 2, pp. 932–944, 2022.

[49] N. Gordon, J. Hohwy, M. J. Davidson, J. J. van Boxtel, and N. Tsuchiya, "From intermodulation components to visual perception and cognition," *NeuroImage*, vol. 199, pp. 480–494, 2019.

[50] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv*, 2014.

[51] N. Alp, P. J. Kohler, N. Kogo, J. Wagemans, and A. M. Norcia, "Measuring integration processes in visual symmetry with frequency-tagged eeg," *Scientific Reports*, vol. 8, pp. 1–11, 2018.

[52] A. Delorme and S. Makeig, "Eeglab: an open source toolbox for analysis of single-trial eeg dynamics including independent component analysis," *Journal of Neuroscience Methods*, vol. 134, no. 1, pp. 9 – 21, 2004.

[53] J. J. Podmore, T. P. Breckon, N. K. N. Aznan, and J. D. Connolly, "On the relative contribution of deep convolutional neural networks for ssvep-based bio-signal decoding in bci speller applications," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 27, no. 4, pp. 611–618, 2019.

[54] X. Zheng, G. Xu, Y. Zhang, R. Liang, K. Zhang, Y. Du, J. Xie, and S. Zhang, "Anti-fatigue performance in ssvep-based visual acuity assessment: A comparison of six stimulus paradigms," *Frontiers in Human Neuroscience*, vol. 14, 2020.

[55] J. Liang, D. Hu, and J. Feng, "Domain adaptation with auxiliary target domain-oriented classifier," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16627–16637, 2021.

[56] D.-H. Lee, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Workshop on challenges in representation learning, ICML*, vol. 3, p. 896, 2013.

[57] Y. Grandvalet and Y. Bengio, "Semi-supervised learning by entropy minimization," in *Advances in Neural Information Processing Systems*, vol. 17, MIT Press, 2004.

[58] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using em," *Machine learning*, vol. 39, no. 2, pp. 103–134, 2000.

[59] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014.

[60] B. Kulis, "Metric learning: A survey," *Foundations and trends in machine learning*, vol. 5, no. 4, pp. 287–364, 2012.

[61] A. K. Massimino and M. A. Davenport, "As you like it: Localization via paired comparisons," *arXiv*, 2018.

[62] M. Gönen and E. Alpaydın, "Multiple kernel learning algorithms," *The Journal of Machine Learning Research*, vol. 12, pp. 2211–2268, 2011.

[63] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987.

[64] J. Liang, D. Hu, and J. Feng, "Domain adaptation with auxiliary target domain-oriented classifier," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (Los Alamitos, CA, USA), pp. 16627–16637, IEEE Computer Society, jun 2021.

[65] Y. Zou, Z. Yu, X. Liu, B. Kumar, and J. Wang, "Confidence regularized self-training," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, (Los Alamitos, CA, USA), pp. 5981–5990, IEEE Computer Society, nov 2019.