The Utility Metric: A Novel Method to Assess the Overall Performance of Discrete Brain-Computer Interfaces

Bernardo Dal Seno, Matteo Matteucci and Luca Mainardi

***Abstract*— A relevant issue in a brain-computer interface (BCI) is the capability to efficiently convert user intentions into correct actions, and how to properly measure this efficiency. Usually, the evaluation of a BCI system is approached through the quantification of the classifier performance, which is often measured by means of the information transfer rate (ITR). A shortcoming of this approach is that the control interface design is neglected, and hence a poor description of the overall performance is obtained for real systems. To overcome this limitation, we propose a novel metric based on the computation of BCI *Utility*. The new metric can accurately predict the overall performance of a BCI system, as it takes into account both the classifier and the control interface characteristics. It is therefore suitable for design purposes, where we have to select the best options among different components and different parameters setup. In the paper, we compute *Utility* in two scenarios, a P300 speller and a P300 speller with an Error Correction System (ECS), for different values of accuracy of the classifier and recall of the ECS. Montecarlo simulations confirm that *Utility* predicts the performance of a BCI better than ITR.**

***Index Terms*—brain-computer interface (BCI), BCI performance, P300 speller, error potential**

I. Introduction

A Brain-Computer Interface (BCI) is a direct communication pathway between the brain and an external device. It bypasses any muscle or nerve mediation and interprets human commands by picking up (and analyzing) signals generated by the brain activity [1]. A relevant issue for this kind of interface is related to the capability to convert efficiently user intentions into correct actions, and how best to assess such efficiency.

The schematic architecture of a BCI system is depicted in Fig. 1: the acquired signal (typically the EEG for non-invasive BCI) is processed, and its relevant features extracted. These features are then used to feed a classifier with the aim of discriminating among a discrete set of options. The selected option is then interpreted and the desired action generated.

Manuscript received January 9, 2009; revised June 2, 2009

This work has been partially supported by the Italian Institute of Technology (IIT), and by the grant "Brain-Computer Interfaces in Everyday Applications" from Politecnico di Milano and Regione Lombardia.

B. Dal Seno and M. Matteucci are with Politecnico di Milano, Department of Electronics and Information, IIT Unit, Piazza Leonardo da Vinci, 32 – 20133 Milano, Italy e-mail: bernardo.dalseno@polimi.it, matteucci@elet.polimi.it

L. Mainardi is with Politecnico di Milano, Department of Bioengineering, IIT Unit, Piazza Leonardo da Vinci, 32 – 20133 Milano, Italy e-mail: luca.mainardi@polimi.it

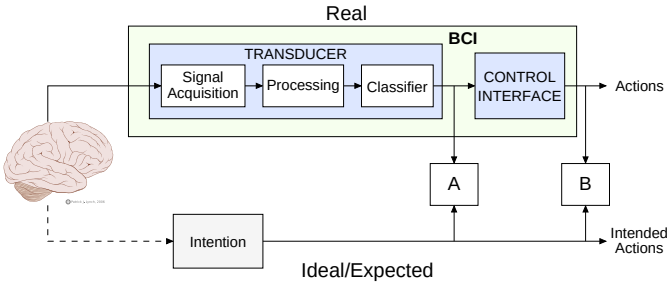


Figure 1. BCI architecture and performance measurement. Different metrics measure performance at different points.

Two functional blocks can be recognized (see also [2], [3]): the Transducer (TR) and the control interface (CI). Their outputs are evidenced by blocks 'A' and 'B', respectively. Both the TR and CI blocks contribute to the overall system performance.

The kind of recorded signal, the way it is processed, the type of classifier and the CI strategy are only a few examples of the variety of design options in the implementation of a BCI system. Evaluation criteria are therefore needed to select among available options and to identify the best design.

Usually, the evaluation of a BCI system is approached through the quantification of the classifier performance (i.e., the comparison among systems is performed at level of block 'A' in Fig. 1). A series of evaluation criteria has been proposed including the classification accuracy (or, equivalently, the error rate), Cohen's Kappa coefficient [4], and indexes based on mutual information, such as the widely-used information transfer rate (ITR) [5]. A detailed overview of these metrics can be found in [6]. Unfortunately, these metrics do not take CI into account and therefore they are poorly descriptive of the overall BCI performances. In addition, they are of limited use for design purposes, as they do not provide any clue to select among different CIs, among different error correction strategies, or to define the best combination of TR and CI.

The problem of finding an indicator able to predict the performance of the whole BCI system was recently approached in [3]. The authors illustrate that, in a copy spelling task, the same TR when combined with different CIs, may generate different performances. They also show that it is the combination of TR and CI that determines the system performance, and they propose a metric that takes into account both the TR and CI functioning. In this paper, we build upon our previous work [7] and we generalize the metric proposed in [3] by introducing the new concept of *Utility*, a quantity related to

the amount of (quantifiable) benefits obtained by the user when a given BCI system is used. As the new metric depends on both the TR performances and the CI strategy, it is a good candidate to predict the overall BCI performance and to guide the developer in the overall system optimization.

This paper is organized as follows: Section II briefly recalls the most common metrics employed for the evaluation of BCI performance, while in Section III the concept of *Utility* is introduced for a discrete-BCI system. In the next two sections, *Utility* is used to evaluate the overall performance of a P300-based speller with and without an error correction system: We show that the new metrics can predict when (and at what extent) an error-correction system improves the system performance. In Section VI we report the results of Monte-Carlo simulations, and we evidence the potentiality/advantages of the new metric compared to the ITR. Finally, some discussion and conclusions are reported in Sections VII and VIII respectively.

II. Measuring BCI Performance

A set of metrics has been proposed in the literature to asses the performance of a BCI system. Among them we recall the classification accuracy, Cohen's Kappa statistic, the information transfer rate, and the more recent efficiency.

Classification accuracy, i.e., the fraction of examples correctly classified, is the simplest performance measure used in the BCI literature. Sometimes, the error rate is used instead, but it is equivalent to accuracy, as the error rate is the fraction of examples wrongly classified. They are both easy to compute and to understand, but they present many shortcomings. They do not make any distinction between different kinds of errors, while different kinds of errors have different impact (cost) in general. Moreover, classes that appear less frequently in the data are weighted less in the accuracy computation, and this is likely to lead to biased classifiers and biased evaluations.

The *Cohen's Kappa coefficient* [4] is a way to express the agreement between two classifications. This measure takes into account the different frequencies for classes and also how errors are distributed among classes, although its meaning is not so explicit as accuracy.

Information transfer rate (ITR — sometimes simply called *bit rate*) is a performance measure widely used in the literature [5], [8]. It has many advantages: It does not depend on any particular protocol, it takes into account both the number of choices and time, it is strongly theoretically grounded, and it could be applied also to continuous ranges of choices [9].

A theoretical formula from information theory was derived in [5] to compute the (mean) number of bits transferred per trial in a BCI:

$$B = \log_2 N + p \log_2 p + (1-p) \log_2 \frac{1-p}{N-1}\,, \qquad (1)$$

where N is the number of possible choices per trial, and p is the accuracy of the BCI. When (1) is divided by the trial duration, c, the mean number of bits transferred per time unit, i.e., the Information Transfer Rate (ITR), is obtained:

$$ITR = \frac{B}{c} \qquad (2)$$

This formula is derived from Shannon's theory [10], and it represents a measure of the *mutual information* between the user's choice and the BCI selection, under the assumptions that all choices convey the same amount of information (i.e., they are chosen by the user with equal probability), that p is the same for all the possible choices, and that all the wrong choices have equal probability. In other words, a BCI system is seen as a noisy channel, in which noise is added every time the system selects the wrong option.

According to Shannon's noisy channel coding theorem [10], it is possible to achieve an arbitrarily small error probability in a communication on a noisy channel as long as the information transfer rate does not go beyond a certain limit. The maximum achievable transfer rate with no errors is the mutual information, and this seems to justify the use of mutual information in (1). The only problem is that Shannon proved his famous theorem by transferring information embedded in ever increasing blocks of bits, and in telecomunication very complex error correction schemes have been devised in order to get near Shannon's limit. For a BCI, where a human subject sits at one end of the noisy channel, it is not possible to do anything complex, and the estimate given by the mutual information can be very far from what can be achieved in practice. In other words, (1) is a theoretical figure, which may be unrealistic for measuring (or predicting) the *real* performance of practical BCI systems.

A further problem with (1) is that it does not take into account how the BCI is implemented and, in particular, how the output of the TR is interpreted by the CI. Even in an analysis like [11], where different metrics are compared through simulated experiments, or like [12], where the performance of a BCI is analyzed as a parameter varies, the CI is not modeled. There are alternative designs in which the CI can be built, but ITR does not help in selecting the best combination of TR and CI (as also underlined in [3]).

A modified version of the above relation was derived in [13] when an error correction system (ECS) based on the detection of Error Potentials (ErrP) is added to the BCI system. The resulting formula is

$$B = p_{\mathrm{t}} \cdot \left(\log_2 N + p' \log_2 p' + (1-p') \log_2 \frac{1-p'}{N-1} \right), \quad (3)$$

where $p_{\mathrm{t}} = p{\cdot}r_{\mathrm{C}} + (1-p)\,(1-r_{\mathrm{E}})$ and $p' = p{\cdot}r_{\mathrm{E}}/p_{\mathrm{t}}$, and where r_{E} is the recall for errors (the fraction of times that an actual error is recognized by the error classifier) and r_{C} is the recall for correct trials (the fraction of times that a correctly spelled letter is recognized by the error classifier). In other words, they derive the new accuracy p' for the system after discarding outcomes rejected by the ErrP detection, and use it in (1); the factor p_{t} takes into account the fact that discarded outcomes do not contribute to the information transfer. Equation (3) is subject to the same limitations of (1), though.

All the above criticisms to the ITR have been addressed in [3], where also a realistic metric measuring the efficiency of a BCI is proposed. Such efficiency represents the inverse of the expected time to issue a meaningful command through the BCI, but it does not contemplate the possibility that different

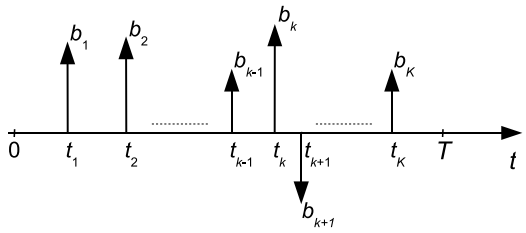


Figure 2. The benefit function for a discrete BCI system. The function is a series of delta functions, because a benefit is available only when a certain selection is made.

commands may have different benefits (or costs, if they cannot be undone).

III. Utility

We approach the quantification of a BCI system performance from a different perspective, more user-centered, and we introduce the concept of *Utility*. If a BCI is supposed to serve the user as a tool, the user should get some kind of benefit when the BCI works correctly. We can formalize this concept by defining *Utility* as the expected average benefit (for the user) over time:

$$U = \mathrm{E}\left[\lim_{T\to\infty} \frac{\int_0^T b(t)dt}{T}\right], \tag{4}$$

where $b(t)$ is a *benefit function*, which assumes positive (or negative) values depending on whether the choice at time t conforms to (or contradicts) the user intention.

Let us consider the case of a discrete BCI, i.e., a BCI system whose output is defined only at discrete time instants. In this situation, the benefit function will be discrete, and defined only in the time-instant t_k when an output is generated. In fact, a quantifiable benefit can be defined only when an output is selected (e.g., a letter printed by the speller, or a target reached on the screen). The situation is described in Fig. 2, and in mathematical terms we may write

$$b(t) = \sum_{k=1}^{K} b_k \delta(t - t_k), \tag{5}$$

where K is the number of outputs in the interval $[0, T]$. Substituting Equation (5) into Equation (4) and observing that $T = \sum_{k=1}^{K} \Delta t_k$, where $\Delta t_k = t_k - t_{k-1}$ for $k > 1$, and $\Delta t_1 = t_1$, we may write

$$U = \mathrm{E}\left[\lim_{T\to\infty} \frac{\sum_1^K \int_0^T b_k \delta(t - t_k)dt}{T}\right] = \mathrm{E}\left[\lim_{K\to\infty} \frac{\sum_1^K b_k}{\sum_{k=1}^K \Delta t_k}\right]. \tag{6}$$

For a stochastic variable n generated by an ergodic process [14], the following relation holds:

$$\lim_{K\to\infty} \frac{\sum_{k=1}^K n_k}{K} = \mathrm{E}[n]. \tag{7}$$

This can be used in (6) to eliminate the limit and obtain a formulation of *Utility* valid for a discrete-BCI system:

$$U = \mathrm{E}\left[\frac{\mathrm{E}[b_k]}{\mathrm{E}[\Delta t_k]}\right] = \frac{\mathrm{E}[b_k]}{\mathrm{E}[\Delta t_k]}. \tag{8}$$

This formulation can be easily read as the ratio between the average benefit (among all the possible discrete choices) and the average time needed to get it. Therefore *Utility* will be higher for the BCI system that makes it possible to reach the desired target (maximum benefit) in the shortest interval of time.

IV. Utility in a P300 Speller

In this section we derive an explicit formula of *Utility* for a P300 speller [15]. As our metric is dependent on the CI strategy, we have to define how the interface works.

A. The P300-Speller design

We design the interface in a very simple and natural way: At every trial the BCI selects a letter and displays it on the screen. If the letter is correct, the user moves on to the next letter, otherwise he/she has to "hit" the backspace symbol to cancel the misspelled letter. Globally, the speller has N possible selections: $N - 1$ letters plus the backspace symbol.

The following general assumptions are also considered:

- A1. the accuracy p of the system is constant over the trials, thus no time-dependency is included in the model;
- A2. the system has no memory, i.e., each trial is not influenced by the result of the previous one.

Moreover, we make use of the same assumptions underlying (1). All these assumptions are not required for the general formulation of *Utility*; they just make the following derivation simpler and let us concentrate on the main issues.

B. Utility computation

By using (8), the *Utility* for a P300 speller is obtained as the average benefit b_{L} carried by any *correctly spelled* letter divided by the expected time T_{L} required to spell it:

$$U = \frac{b_{\mathrm{L}}}{T_{\mathrm{L}}}. \tag{9}$$

We could simply set $b_{\mathrm{L}} = 1$ to assign a unitary benefit of any letter, or we can measure it in terms of information conveyed by that letter. Assuming equal probability among letters, the conveyed information is $b_{\mathrm{L}} = log_2(N - 1)\,\mathrm{bits}$. While we disregard the fact that in reality different letters appear with different frequencies, the equal-probability assumption can be useful to directly compare (9) with the ITR of (2).

To compute T_{L}, we first define c as the duration of a single trial (i.e., the time needed to spell a letter either correctly or wrongly). For each trial we have two possible cases:

1) The P300 speller selects the correct letter. This happens with some probability p, and T_{L} is the duration of a single trial.
2) The P300 speller selects the wrong letter. This happens with probability $(1 - p)$, and T_{L} is longer because one

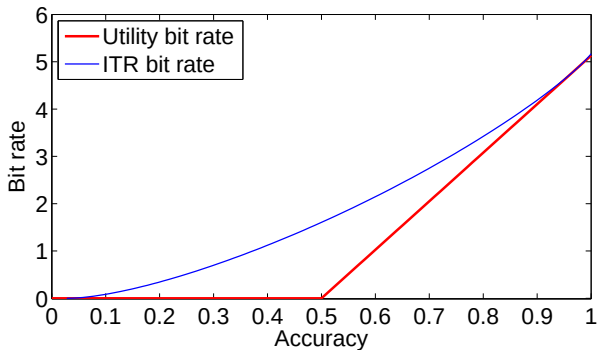


Figure 3. Comparison between utility and theoretical bit rate for a P300 speller with 36 choices.

has to add both the time needed for typing a backspace and that to respell the letter.

Globally, the *expected time* becomes

$$T_{\mathrm{L}} = p\,c + (1-p)(c + T_{\mathrm{B}} + T_{\mathrm{L}}^{(1)}) = c + (1-p)(T_{\mathrm{B}} + T_{\mathrm{L}}^{(1)})\,, \quad (10)$$

where T_{B} is the expected cost of a backspace, and $T_{\mathrm{L}}^{(1)}$ is the expected cost of respelling the letter correctly after the backspace. Similarly, the time for typing a backspace is given by:

$$T_{\mathrm{B}} = p\,c + (1-p)(c + T_{\mathrm{B}}^{(1)} + T_{\mathrm{B}}^{(2)})\,, \quad (11)$$

i.e., when a backspace is "misspelled", two more backspaces are needed. Under the assumptions A1 and A2 we obviously have $T_{\mathrm{L}}^{(1)} = T_{\mathrm{L}}$. In addition, $T_{\mathrm{B}} = T_{\mathrm{B}}^{(1)} = T_{\mathrm{B}}^{(2)}$, and by subtracting (10) and (11) we also get $T_{\mathrm{B}} = T_{\mathrm{L}}$.

Equation (10) can now be written in an iterative formulation

$$\begin{aligned} T_{\mathrm{L}} &= c + 2(1-p)T_{\mathrm{L}} = c + 2(1-p)(c + 2(1-p)T_{\mathrm{L}}) \\ &= c + 2(1-p)c + 4(1-p)^2 T_{\mathrm{L}} = \ldots \\ &= c + 2(1-p)c + 4(1-p)^2 c + 8(1-p)^3 c + \ldots \\ &= c\sum_{i=0}^{\infty}(2-2p)^i; \end{aligned} \quad (12)$$

this series converges to

$$T_{\mathrm{L}} = \frac{c}{2p-1} \quad (13)$$

if $2p-1 > 0$, i.e., $p > 0.5$. Conversely, when $p \leq 0.5$ the series does not converge, and the expected time to correctly spell a letter goes to infinite.

Putting $b_{\mathrm{L}} = log_2(N-1)$ and (13) into (9) we finally get

$$U = \frac{b_{\mathrm{L}}}{T_{\mathrm{L}}} = \frac{(2p-1)\log_2(N-1)}{c}. \quad (14)$$

As we decided to measure the benefits in terms of bits of information, the above expression has the same units of the ITR, and therefore a direct comparison is possible.

C. Comparison with ITR

Fig. 3 compares *Utility* and ITR, and shows how different they can be. The reason is that (2) measures the *capacity of a channel*, i.e., the maximum performance obtainable by a noisy channel, while (14) measures the expected performance of the same channel when information is conveyed in a specific way; in our case, the way we are using the P300 speller. As expected, the latter curve lies always below the theoretical limit, and it is equal to zero when the accuracy is too low. For high accuracy values, the two curves almost coincide, although there is a small gap due to the presence of a *backspace* symbol, which is obviously never used if no error is committed.

The bit-rate curves in Fig. 3 are consistent with the plots in [16, Chapter 3], where a somewhat similar approach to measuring a BCI performance is developed. The graph shows regions where the channel cannot be used with the described P300 speller (when $p \leq 0.5$ in our case) and also areas where the speed of the speller is very far from the theoretical limit. The reported comparison should also warn us against applying (2) blindly, because it may provide unrealistic scores.

A simple numerical example may help to stress the difference between ITR and *Utility*. Let's suppose that in the P300 speller letters are selected at a rate of about 4 per minute and suppose that a user can achieve an accuracy of $p = 45\%$, which is low, but still far better than random-level accuracy ($p = 2.7\%$ with $N = 36$). By substituting these values in (1), we get $B = 1.36$ bits. The information transfer rate for this user would be $4B = 5.4$ bits/min; this is not very fast, but, the ITR metric predicts that communication is possible. However, the above computation does not consider the way the speller works. If we include this ingredient in the recipe, we may easily observe that every displayed letter is more likely to be wrong than correct. Thus, when the user tries to correct an error by selecting backspace, another (wrong) letter is selected more likely than not and the expected time to spell a letter correctly goes to infinite! The interface cannot be used as it is and, on average, the transfer rate is exactly 0 bits/min as predicted by our *Utility* metric.

V. Utility in a P300 Speller with Error Correction

In this section we will show how Utility can be used to measure the improvement of the performance gained when an automatic error-correction system (ECS) is added to the P300 speller. We may assume that the ECS is based on the detection of error potentials [17], [18] after the presentation of a feedback, but the derived results are general and valid for any automatic correction system (e.g., T9-like systems) with known performances. In the case of an ECS based on error potentials, the TR recognizes both P300 and error potentials (in different time periods), while the CI handles the responses to errors.

A. The P300 Speller with an Error-Correction System

This modified speller basically works as described in the previous section: It selects a letter by means of P300 detection and displays it on the screen; if the automatic ECS detects an

error, the latest letter is automatically canceled, while if no error is detected, the letter is kept. The user can then decide if the letter is correct or not, and in the latter case (when the ECS has taken the wrong decision) he has to select the backspace symbol to remove the wrong letter.

B. Utility Computation

The *Utility* of a speller has been already defined by (9); in the new system, we have to compute the expected time per letter T_{L} in terms of the performance of both the P300 detector and the ECS. While the former can be expressed by the single parameter p, which defines the goodness of the classification, we characterize the latter by two parameters: 1. the recall for errors (r_{E}), and 2. the recall for correct trials (r_{C}). We assume that r_{E} and r_{C} are constant and do not depend on the actual letter.

For each trial, we have to deal with four possible cases now:

1) The P300 speller selects the correct letter, and the ECS correctly recognizes it. This happens with probability $p_1 = p \cdot r_{\mathrm{C}}$.
2) The P300 speller selects a wrong letter, and the ECS does not recognize the error. This happens with probability $p_2 = (1-p) \cdot (1-r_{\mathrm{E}})$, and the user has to "spell" a backspace and then the letter again.
3) The P300 speller selects the correct letter, and the ECS wrongly detects an error. This happens with probability $p_3 = p \cdot (1-r_{\mathrm{C}})$, and the user has to respell the letter (which has been canceled by the ECS).
4) The P300 speller selects a wrong letter, and the ECS recognizes the error. This happens with probability $p_4 = (1-p) \cdot r_{\mathrm{E}}$, and the user has to respell the letter; the wrong letter is canceled by the system.

By following a procedure similar to the one in IV-B, we can derive[1]

$$T_{\mathrm{L}} = \frac{c}{p\, r_{\mathrm{C}} + (1-p)\, r_{\mathrm{E}} + p - 1}\,. \tag{15}$$

The *Utility* is obtained by substituting T_{L} in (9):

$$U = \frac{b_{\mathrm{L}}}{T_{\mathrm{L}}} = \frac{log_2(N-1)\,(p\, r_{\mathrm{C}} + (1-p)\, r_{\mathrm{E}} + p - 1)}{c}\,. \tag{16}$$

In analogy with (13), this formula can be obtained as the limit of a series. This limit exists only if the denominator in (15) is positive, i.e., when

$$p\, r_{\mathrm{C}} > (1-p)\,(1-r_{\mathrm{E}})\,, \tag{17}$$

Fig. 4(a) shows the boundaries defined by (17) for different values of p; the inequality is satisfied for the points lying above the lines, and only in these cases the time for spelling a letter is finite (i.e., the P300 speller can be useful). It can be noticed that the constraint becomes tighter as p diminishes, with recall of errors becoming more and more important.

The left and right sides of (17) are the probabilities p_1 and p_2 (defined above), respectively; in other words, the speller can be used as long as the number of correct selections surpasses the number of wrong letters. The number of letter canceled by the error detection system affects the speed of the speller, but it does not affect the fact that the right letter is, sooner or later, spelled.

[1]A complete derivation can be found in [19].

C. Utility with and without Error Correction

The really interesting question, though, is: when does error detection give any improvement to the P300 speller? The answer can be found by comparing (13) and (15). A first observation is that, as expected, (13) is a particular case of (15) with $r_{\mathrm{C}} = 1$ and $r_{\mathrm{E}} = 0$, i.e., no error is ever corrected.

For $p \leq 0.5$, (13) has no sense, but, as shown in Fig. 4(a), it is possible to operate a P300 speller even with such a high error rate as long as the error detection is sufficiently accurate. Actually, it could be argued that this is an unlikely scenario, where incorrect P300 detection is done after many repetitions, while the ECS perfectly works in a single trial; yet, this is a piece of the whole picture.

It is more interesting to investigate the situations in which $p > 0.5$, when (13) and (15) can be compared directly. In order to have an improvement, the expected time, T_{L}, should be lower when error correction is used; thus

$$p\, r_{\mathrm{C}} + (1-p)\, r_{\mathrm{E}} + p - 1 > 2p - 1\,, \tag{18}$$

i.e.,

$$p\, r_{\mathrm{C}} + (1-p)\, r_{\mathrm{E}} > p\,. \tag{19}$$

Fig. 4(b) shows the boundaries defined by (19) for different values of p (for $p < 0.5$ the comparison has no sense); points above the lines represent values of r_{C} and r_{E} for which the presence of the ECS is advantageous. In this case, as p grows the area defined by (19) shrinks; this happens, because as p grows the performance of the P300 speller gets better and better, and it becomes harder and harder for the ECS to make any difference. The left side of (19) is equal to $p_1 + p_4$ (previously defined); this means that an ECS is advantageous when the overall accuracy of the P300 and the ECS is better than the accuracy of the P300 system alone.

We are now able to compute the gain, g, of introducing an ECS in a P300 speller. This can be obtained as the ratio of the time lengths given by (13) and (15):

$$g = \frac{p\, r_{\mathrm{C}} + (1-p)\, r_{\mathrm{E}} + p - 1}{(2p-1)}\,. \tag{20}$$

A value of $g > 1$ means that the introduction of ECS is advantageous (the value of T_{L} is reduced, and Utility increased), while introducing the ECS is counter-productive when $g < 1$. Equation (20) is subject to the constraints that both the numerator and the denominator are positive, i.e., (17) and $p > 0.5$. If only the denominator is negative, it means that the P300 speller cannot work without the ECS, and hence g should be considered infinite. If both the numerator and the denominator are negative, it means that the P300 speller cannot work, with or without ECS, and hence g is indefinite. If only the numerator is negative, it means that introducing the ECS renders the speller unusable, and hence $g = 0$.

Fig. 4(c) summarizes the first two graphs in Fig. 4, and shows the values of r_{C} and r_{E} for which the ECS is advantageous for the whole range of p. As before, the part of the

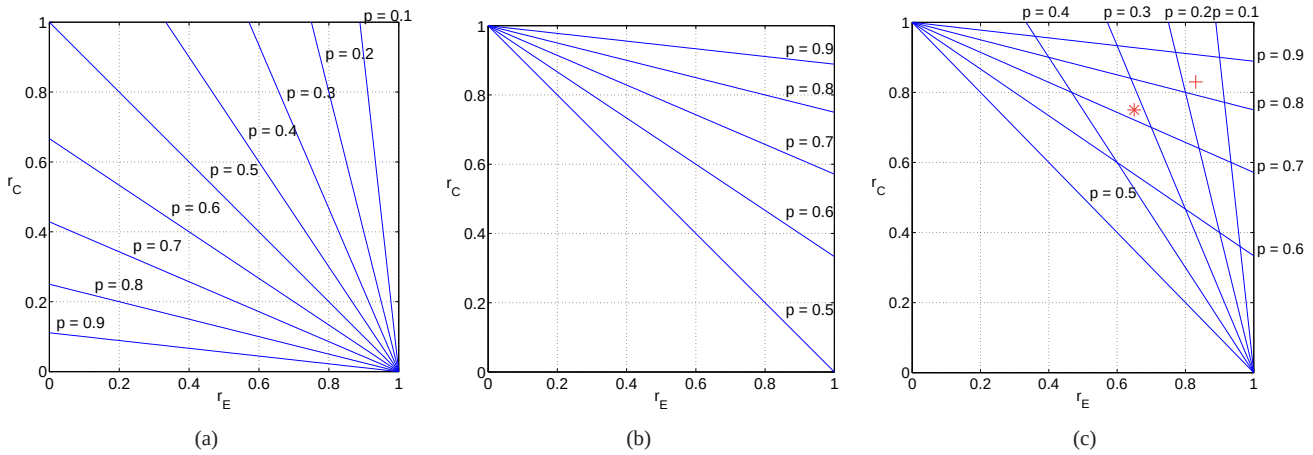


Figure 4. (a) Condition for the usability of a P300 speller with an ECS. (b) Comparison between two P300 spellers, with and without an ECS. (c) When an ECS improves the performance of a P300 speller.

plane above the lines is the useful part; values below the lines are either of no interest or counterproductive. Fig. 4(c) can be used as a guide to decide to bias the ECS either toward correct or erroneous epochs, depending on the value of p.

Some practical examples may help to better understand the above ideas. Let say that for a particular user the P300 speller reaches 90% accuracy without error correction, and the error detection reaches $r_{\mathrm{C}} = r_{\mathrm{E}} = 83\%$. This situation corresponds to the cross in Fig. 4(c). As the cross lies below the line $p = 90\%$, for this particular user the automatic error correction system is counterproductive. Another user's performance may be expressed by $p = 70\%$, $r_{\mathrm{E}} = 65\%$, $r_{\mathrm{C}} = 75\%$ (the asterisk in Fig. 4(c)); the asterisk lies above the line $p = 70\%$, and therefore the automatic error correction system should help this user.

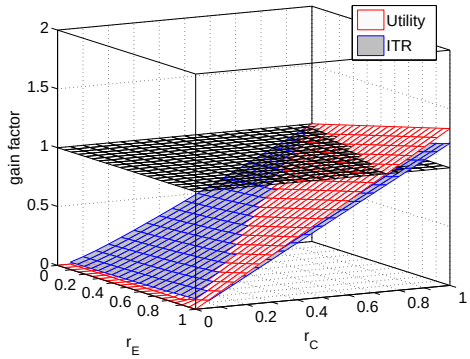


Figure 5. Comparison of performance improvement in a P3 speller (N=36 symbols, p=80%) obtained according to different formulas

D. Comparison with Modified ITR

It is interesting to compare the gain in adding an ECS computed with the approach based on utility and the gain computed with an approach based on the theoretical bit rate in (3). Fig. 5 shows the performance gain factor obtained by applying the channel-capacity approach (darker surface) and our utility-based approach (lighter surface). The two surfaces show the gain obtained by introducing an ECS in a P300 speller ($p = .8$ and $N = 36$ symbols) as a function of the recalls r_{C} and r_{E}. The graph shows the regions in the r_{C},r_{E} plane corresponding to points for which an ECS is advantageous (areas where $g > 1$). Both the regions and the gains obtained by the two metrics are different. For other values of p the graphs are qualitatively the same, but while the difference between the two approach is small for high values of p, it grows as the value of p gets smaller. The approach based on the information theoretical channel capacity seems to underestimate the contribution of ErrPs; this is due to its tendency to underestimate the cost of errors, as already shown in Fig. 3.

VI. Simulations and Evaluations

In order to validate our proposed metric, we have run a number of simulations mimicking the use of the P300 speller, with and without an ECS, and with different values for the accuracies of the component of the BCI.

A random string made of 1000 letters (symbols) and spaces is chosen in every simulation; all the symbols are used with equal probability, in order to be consistent with the assumptions of the ITR metric. A combination of the parameters p, r_{E}, and r_{C} is chosen, and the interaction of the interface is simulated, taking into account errors, wrong decisions, and backspaces; the simulation ends when the whole string is correctly spelled. A mean bit rate is computed by dividing the information contained in the string ($1000\, b_L$) by the time (in trials, i.e., selections) taken to spell the string. This value is compared with the predictions by the *Utility* (Equation (16)) and by the theoretical bit rate (Equation (3)).

Fig. 6 shows the results obtained in 100 000 simulations. The difference between the mean bit rate computed according to the formulas and the one obtained by the simulations is shown versus the three parameters that characterize the speller; since the three parameters plus the bit rate fill a 4-D space, three 2-D projections are shown. A point is plotted for every simulation, and darker regions have a higher point density; also, a line representing the average error is plotted. The ranges for the parameters have been chosen so as to be realistic (e.g., p is roughly in the same range as in [20], [21]). The

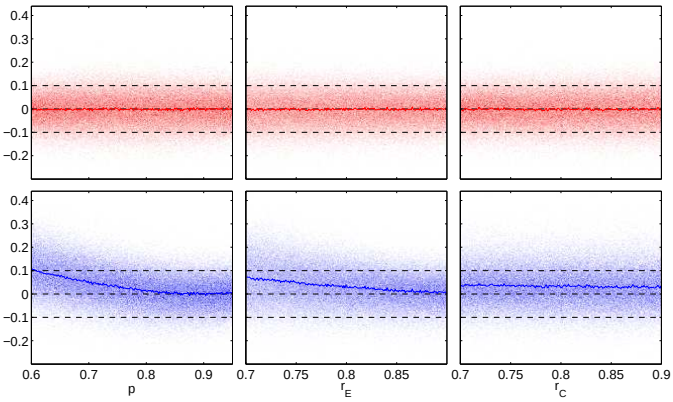


Figure 6. Difference of predicted and observed bit rates in simulations of a P300 speller with automatic error correction. Upper row: *Utility*; lower row: theoretical bit rate.

upper row contains the results for the *Utility* metric, while the lower row those for the theoretical bit rate. It is easy to observe that the error made by the *Utility* is always around zero (with fluctuations due to the intrinsic randomness of the simulations), while the error made by the theoretical bit rate is biased for a wide range of parameter values.

VII. A Possible Generalization

It is worth noting that we have derived the closed formulas for *Utility* under some simplifying assumptions. In fact, we have hypothesized equal probability of letter occurrence and equal accuracy for each selection; thus, no confusion matrix is taken into account in this paper. These assumptions have helped us to keep the formulas simpler and the exposition easier to follow, but they are not a limiting factor for our results. It is possible to derive a formula for *Utility* in a more general case, as it has already done for the ITR formula (1) [22, Appendix B]. For the P300 speller, this requires the computation of the expected time T_{L} needed to spell a letter when different letters have different frequencies f_i ($i = 1 \dots N$), and the confusion matrix (i.e., the probability p_{ji} of spelling letter j when aiming at letter i) is arbitrary:

$$T_{\mathrm{L}} = \frac{c + T_{\mathrm{Del}}\, P_{\mathrm{X|L}}}{P_{\mathrm{L|L}} - P_{\mathrm{Bs|L}}}\,. \tag{21}$$

In (21) T_{Del} is the expected time required to recover from a misspelled letter, $P_{\mathrm{X|L}}$ is the probability of spelling a letter different from the one the user is interested in, $P_{\mathrm{L|L}}$ is the probability of spelling the correct letter, and $P_{\mathrm{Bs|L}}$ the probability of spelling the backspace instead of the letter the user would like to spell. The above terms can be derived with techniques applied in Section IV and they result to be: $T_{\mathrm{Del}} = \frac{c}{2p_{n\,n}-1}$ (where n is the index of the backspace command), $P_{\mathrm{L|L}} = \sum_{i\neq n} f_i p_{ii}$, $P_{\mathrm{X|L}} = \sum_{i\neq n} f_i \sum_{j\neq n,i} p_{ji}$, and $P_{\mathrm{Bs|L}} = \sum_{i\neq n} f_i p_{ni}$. Although the derivation and analysis of this *Utility* formulation based on a full confusion matrix is out of the scope of this paper, it is worth noting that this result requires $p_{n\,n} > 0.5$ and $P_{\mathrm{L|L}} > P_{\mathrm{Bs|L}}$ for all letters, i.e., $p_{ii} > p_{ni}$ where $i \neq n$. Since the generalized ITR formula from [22] has the same shortcoming of (1) (i.e., it treats the BCI as a communication channel with no reference to the way the channel is actually used), we expect that a comparison of *Utility* and ITR based on the full confusion matrix would lead to the same result we have claimed in this paper, and some very preliminary work confirmed already our expectation.

VIII. Conclusions

In this paper we have introduced a general metric for the evaluation of the overall performance of a BCI system based on the computation of BCI *Utility*. The new metric has been also derived in closed form for two discrete-BCI systems: a P300-speller with and without ECS. Though we have discussed in details only BCIs based on P300, there is nothing in our approach that prevents the application to BCIs based on other potentials or protocols, as long as they are discrete. For example, it should be possible to apply it to self-paced protocols by using a statistical model that captures the timing produced by the particular TR employed.

Utility is a generalization of the efficiency proposed in [3]; it takes into account the time needed to produce an output in a BCI, as in the efficiency, but also the different benefits that different outputs may have and the cost connected to issuing commands whose effects cannot be undone. We also tried to maintain a simple notation, where the contribution of the different parameters of the system on the performance is clear, and thus it can help with tuning and design choices. For example, the predicted performance of two different classifiers can be compared to select the best one; or, if a binary BCI is used to select commands through a binary selection process, the utility of different dispositions of the choices can be computed without time-consuming experiments.

Using simulations of the above mentioned spellers, we have compared the performance of our metric against ITR in predicting the overall behavior of a BCI, and we have demonstrated a superior performance of our *Utility* metric. We have also shown that ITR, intended as the channel capacity of the BCI classifier, can provide unreliable results if employed to evaluate (and predict) the behavior of the whole BCI system.

This result is not completely surprising. In fact, ITR is basically a *theoretical* measure which does not take into account how the system works in practice. It is focused on the evaluation of the TR and its classification strategy, but it does not consider how this classification is further processed by the system. Conversely, *Utility* is a task-oriented metric which can take into account all the components of a BCI device as well as how the user interacts with the system. The possibility to factor both the TR and the CI characteristics into *Utility* is useful for design purposes, where there is the need to choose among different classifiers, different correction strategies, and different CIs in order to select the optimal combination.

Our simulations have shown that the use of a task-oriented metric allows realistic observations about the usefulness of an ECS and to identify optimal parameters and operating settings. It is important to notice that although we have computed the joint performance of the TR and the CI, we have modeled the two subsystems separately. This separation permits to study many combinations of TRs and CIs efficiently. For example,

one experimental session with a subject can be analyzed offline to evaluate the accuracy of some classifiers, and the performance of alternative CIs can be modeled in terms of the accuracy of such classifiers. Exploring the use of a new TR or a new CI requires only one time-consuming step, i.e., the modeling of the single new component; the combination with the other components requires only the substitution of some numerical values in a formula.

While we have applied the proposed approach to two specific cases, the same approach can be extended to study other kinds of BCIs as well as the impact of the modification of other design parameters. We believe that such an approach should lead to a better comparison between different protocols.

REFERENCES

[1] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan, "Brain-computer interfaces for communication and control," *Clin. Neurophysiol.*, vol. 113, pp. 767–791, 2002.

[2] S. G. Mason and G. E. Birch, "A general framework for brain-computer interface design," *IEEE Trans. Neural Syst. Rehab. Eng.*, vol. 11, no. 1, pp. 70–85, Mar. 2003.

[3] L. Bianchi, L. R. Quitadamo, G. Garreffa, G. C. Cardarilli, and M. G. Marciani, "Performances evaluation and optimization of brain computer interface systems in a copy spelling task," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 15, no. 2, pp. 207–216, Jun. 2007.

[4] J. Cohen, "A coefficient of agreement for nominal scales," *Educ. Psychol. Meas.*, vol. 20, pp. 37–46, 1960.

[5] J. R. Wolpaw, N. Birbaumer, W. J. Heetderks, D. J. McFarland, G. S. P. Hunter Peckham, E. Donchin, L. A. Quatrano, C. J. Robinson, and T. M. Vaughan, "Brain-computer interface technology: A review of the first international meeting," *IEEE Trans. Rehabil. Eng.*, vol. 8, no. 2, pp. 164–173, Jun. 2000.

[6] A. Schlögl, J. Kronegg, J. E. Huggins, and S. G. Mason, "Evaluation criteria for BCI research," in *Towards Brain-Computer Interfacing*, G. Dornhege, J. d. R. Millán, T. Hinterberger, D. J. McFarland, and K.-R. Müller, Eds. MIT Press, 2007, pp. 327–342.

[7] B. Dal Seno, L. Mainardi, and M. Matteucci, "Assessing the performance of a BCI: A task-oriented approach," in *Proc. the 4th International Brain-Computer Interface Workshop & Training Course*. Graz, Austria: Technischen Universität Graz, Sep. 2008, pp. 274–279.

[8] B. Obermaier, C. Neuper, C. Guger, and G. Pfurtscheller, "Information transfer rate in a five-classes brain-computer interface," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 9, no. 3, pp. 283–288, Sep. 2001.

[9] A. Schlögl, C. Keinrath, R. Scherer, and G. Pfurtscheller, "Information transfer of an EEG-based brain computer interface," in *Proc. 1st International IEEE EMBS Conference on Neural Engineering*, 2003, pp. 641–644.

[10] C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication*. The University of Illinois Press, 1949.

[11] J. Kronegg, S. Voloshynovskiy, and T. Pun, "Analysis of bit-rate definitions for brain-computer interfaces," in *2005 Int. Conf. on Human-computer Interaction (HCI'05)*, Las Vegas, Nevada, USA, Jun. 2005.

[12] G. Dornhege, B. Blankertz, G. Curio, and K.-R. Müller, "Increase information transfer rates in BCI by CSP extension to multi-class," in *Advances in Neural Information Processing Systems (NIPS 03)*, S. Thrun, L. Saul, and B. Schölkopf, Eds., vol. 16. Cambridge, MA, USA: MIT Press, 2004.

[13] P. W. Ferrez and J. d. R. Millán, "You are wrong!—automatic detection of interaction errors from brain waves," in *Proc. 19th International Joint Conference on Artificial Intelligence*, 2005, pp. 1413–1418.

[14] P. Peebles, *Probability, Random Variables, and Random Signal Principles*. McGraw-Hill Companies, 1980.

[15] L. A. Farwell and E. Donchin, "Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials," *Electroenceph. Clin. Neurophysiol.*, vol. 70, no. 6, pp. 510–523, Dec. 1988.

[16] G. Dornhege, "Increasing information transfer rates for brain-computer interfacing," Ph.D. dissertation, University of Potsdam, Germany, 2006.

[17] P. W. Ferrez and J. d. R. Millán, "Error-related EEG potentials generated during simulated brain-computer interaction," *IEEE Trans. Biomed. Eng.*, vol. 55, no. 3, pp. 923–929, Mar. 2008.

[18] G. Visconti, B. Dal Seno, M. Matteucci, and L. Mainardi, "Automatic recognition of error potentials in a P300-based brain-computer interface," in *Proc. 4th International Brain-Computer Interface Workshop & Training Course*. Graz, Austria: Technischen Universität Graz, Sep. 2008, pp. 238–243.

[19] B. Dal Seno, "Toward an integrated P300- and ErrP-based brain-computer interface," Ph.D. dissertation, Politecnico di Milano, Milan, Italy, 2009.

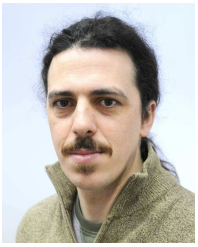
[20] E. W. Sellers, D. J. Krusienski, D. J. McFarland, T. M. Vaughan, and J. R. Wolpaw, "A P300 event-related potential brain-computer interface (BCI): The effects of matrix size and inter stimulus interval on performance," *Biol. Psychol.*, vol. 73, no. 3, pp. 242–252, Oct. 2006.

[21] D. J. Krusienski, E. W. Sellers, D. J. McFarland, T. M. Vaughan, and J. R. Wolpaw, "Toward enhanced P300 speller performance," *J. Neurosci. Methods*, vol. 167, no. 1, pp. 15–21, Jan. 2008.

[22] T. Nykopp, "Statistical modelling issues for the adaptive brain interface," Master's thesis, Helsinki University of Technology, Department of Electrical and Communications Engineering, 2001.

Bernardo Dal Seno received a *laurea* degree in computer engineering and a Ph.D. degree in information engineering from Politecnico di Milano, in 2001 and 2009, respectively.

He is currently a Temporary Researcher at the Department of Electronics and Information, Politecnico di Milano. He previously worked as a software engineer and system administrator. His research interests include brain-computer interfaces, machine learning, information theory.



Matteo Matteucci received a "laurea" degree from the Politecnico di Milano T.U. in 1999, a Master of Science in Knowledge Discovery and Data Mining at Carnegie Mellon University in 2002, and a Ph.D. in Computer Engineering and Automation from the Politecnico di Milano T.U. in 2003. He is now an Assistant Professor at Politecnico di Milano. His current research interests are focused on Robotics and Machine Learning, mainly applying, in a practical way, techniques for adaptation and learning to autonomous systems. His research is in brain-computer interface, adaptive color models, robust tracking for video surveillance, reactive robot control, behavior modeling, intrusion detection, autonomous robots and learning machines in general (i.e., neural networks, decision trees, mixture models, etc.). He is a member of the IEEE.

Luca T. Mainardi received the M.Sc. ("laurea") degree in electrical engineering and the Ph.D. degree in biomedical engineering from Politecnico di Milano, Milano, Italy, in 1990 and 1997, respectively. Since 2001 he is Assistant Professor in the Department of Bioengineering, Politecnico di Milano. He is the Head of the *Biomedical Signal, Data and Image Processing* Laboratory. He is also among the founders of the *Laboratory of Advanced Radiological Analysis* (LARA) and member of its scientific committee. His current research activity is in the field of biomedical signal and image processing and biomedical system modeling with particular applications to the cardiovascular systems. He is studying and developing methods for time-frequency analysis of nonstationary signals, including time-frequency distributions, wavelets, and recursive parametric identification. He is also interested in advanced image processing techniques for features extraction and image registration problems. Dr. Mainardi was a member of the Board of the Italian National Group of Bioengineering (GNB) from 2004 to 2007.