Towards Discriminant Analysis Classifiers Using Online Active Learning via Myoelectric Interfaces

Andrés Jaramillo-Yánez[1], Marco E. Benalcázar[2], Sebastian Sardina[1] and Fabio Zambetta[1]

[1] School of Computing Technologies, RMIT University, Melbourne, Australia
[2] Artificial Intelligence and Computer Vision Research Lab, Escuela Politécnica Nacional, Quito, Ecuador
andres.jaramillo.yanez@student.rmit.edu.au,marco.benalcazar@epn.edu.ec, sebastian.sardina@rmit.edu.au, fabio.zambetta@rmit.edu.au

Abstract

We propose a discriminant analysis (DA) classifier that uses online active learning to address the need for the frequent training of myoelectric interfaces due to covariate shift. This online classifier is initially trained using a small set of examples, and then updated over time using streaming data that are interactively labeled by a user or pseudo-labeled by a soft-labeling technique. We prove, theoretically, that this yields the same model as training a DA classifier via full batch learning. We then provide experimental evidence that our approach improves the performance of DA classifiers and is robust to mislabeled data, and that our soft-labeling technique has better performance than existing state-of-the-art methods. We argue that our proposal is suitable for real-time applications, as its time complexity w.r.t. the streaming data remains constant.

1 Introduction

This work proposes a discriminant analysis (DA) classifier (linear LDA or quadratic QDA) that uses online active learning applied to hand gesture recognition via myoelectric interfaces. By updating its parameters over time using streaming data that are interactively (pseudo-)labeled by a user or soft-labeling technique, this classifier is able to better deal with the frequent (and demanding) training effort inherent in the domain arising from covariance shift.

Myoelectric interfaces use streaming data to develop hand gesture recognition applications, for example, prosthesis control, sign language recognition, hand rehabilitation, and virtual reality (Jaramillo-Yánez, Benalcázar, and Mena-Maldonado 2020). By using surface electromyography (sEMG) sensors, these interfaces provide a non-invasive way to record the electrical activity from neurologically activated skeletal muscles. While sEMG sensors do not suffer from several common issues (e.g., measurement accumulated errors over time, movement constraints, occlusion, illumination, and focus problems) of vision, flex, and inertial sensors (Jaramillo-Yánez, Benalcázar, and Mena-Maldonado 2020), the data acquired through such sEMG sensors are *user-dependent* and *non-stationary* (Shwedyk, Balasubramanian, and Scott 1977), as they are affected by traits such as subcutaneous fat, skin impedance, and pattern of muscle synergies (Matsubara and Morimoto 2013). Not surprisingly, some works have reported *covariate shift*, a phenomenon that occurs when the probability distribution of the input variables (sEMG data) changes with time (Kaufmann, Englehart, and Platzner 2010) and across different users (Castellini, Fiorilla, and Sandini 2009). This phenomenon, in turn, causes the performance of myoelectric interfaces to deteriorate significantly over time (Patricia, Tommasit, and Caputo 2014). To minimize the covariate shift, myoelectric interfaces based on pattern recognition (called from now on myoelectric interfaces) must be re-trained every time a user interacts with them (Sensinger, Lock, and Kuiken 2009). For each training, a large set of examples must be collected to achieve high performance, which is a time-consuming and inconvenient task in terms of usability. This *demanding and frequent training* has, consequently, limited the adoption of myoelectric interfaces by industry (Farina et al. 2014; Liu et al. 2014), even though such interfaces have been investigated for several decades.

To reduce the training effort—both the training time and the training set size—, we propose a *DA classifier* (LDA/QDA) that is trained with a small set of examples and expands its knowledge over time using *online active learning*. Online active learning fits in streaming data applications as myoelectric interfaces, in which obtaining labeled data could be expensive. In this learning paradigm, a learner (e.g., a classifier) interactively query a user or teacher to label new data (Hoi et al. 2021). We use LDA, a classifier widely researched in myoelectric interfaces, and QDA because both demand low data and computational requirements at training, in which few parameters are estimated with respect to data-hungry classifiers (e.g., convolutional neural networks) (Huang et al. 2017; Jaramillo-Yánez, Benalcázar, and Mena-Maldonado 2020).

Some works have proposed LDA for dimensionality reduction, both using online supervised (Pang, Ozawa, and Kasabov 2005; Wang et al. 2017) and semi-supervised (Kim et al. 2007, 2011; Nie et al. 2009; Dhamecha, Singh, and Vatsa 2016; Wang et al. 2016) learning. These approaches are based on Fisher's linear discrimination rule (Fisher 1936; Härdle and Simar 2019), in which the within-class, between-class, and total scatter matrices are updated to compute a linear projection that maps a sample into a low dimension space. In contrast, our proposal focuses on *classification*, using the Bayesian decision rule and the Gaussian assump-

tion (Fukunaga 2013) so as to update the mean vectors and covariance matrices per class and then calculate functions that assign a class label to a given sample. Also while the dimensionality reduction techniques are evaluated in conjunction with a classifier (generally not a DA classifier), we evaluate the performance of our online DA classifier. To the best of our knowledge, LDA or QDA have not been investigated for online active learning, and our work aims to fill this gap.

Concretely, our online classifier is initially trained (in a supervised fashion) with a small dataset (one gesture per class) from a user, and then expands its knowledge using sEMG streaming data labeled by the user or pseudo-labeled by soft-labeling technique that is able to determine the degree in which the streaming data contribute to the classifier. This proposal is user-friendly and fit for real-time applications, as the classifier's parameters are continuously updated with the user interaction with the myoelectric interface and the time complexity remains constant (unlike with full batch learning). We prove, theoretically, that our online classifier yields the same model as that obtained via full batch learning—training with all data at once. Finally, we use five publicly available datasets to experimentally demonstrate that the technique proposed improves the performance of DA classifiers and is robust to mislabeled data, and that the soft-labeling mechanism used has better performance than the state-of-the-art models.

2 DA Classifier using Online Active Learning

We propose a DA classifier (LDA/QDA) that updates its parameters over time using streaming data that are interactively *labeled* by a user or *pseudo-labeled* by a soft-labeling technique. This classifier is initially trained over a small (finite) labeled dataset $\mathcal{I} \subset \{(\vec{x}, y) \mid \vec{x} \in \mathbb{R}^F, y \in \{1, \cdots, C\}\}$ of $C \in \mathbb{N}$ classes and $F \in \mathbb{N}$ features.

Based on the Bayesian decision rule and the Gaussian assumption (Fukunaga 2013), the **LDA and QDA classifiers** w.r.t. a **parameter configuration** $\Theta = \langle(\vec{\mu}_1, \mathbf{\Sigma}_1, \pi_1), \ldots, (\vec{\mu}_C, \mathbf{\Sigma}_C, \pi_C)\rangle$, where $\vec{\mu}_c$, $\mathbf{\Sigma}_c$, and π_c is the mean vector, covariance matrix, and prior probability for class c are functions defined as follows:

$$\hat{y}_L^{\Theta}(\vec{x}) = \underset{c\in\{1,\ldots,C\}}{\arg\max} \left(\vec{\mu}_c^T \mathbf{\Sigma}^{-1} \vec{x} - \frac{1}{2}\vec{\mu}_c^T \mathbf{\Sigma}^{-1} \vec{\mu}_c + \ln \pi_c\right); \quad (1)$$

$$\hat{y}_Q^{\Theta}(\vec{x}) = \underset{c\in\{1,\ldots,C\}}{\arg\max} \left(-\frac{1}{2}(\vec{x}-\vec{\mu}_c)^T \mathbf{\Sigma}_c^{-1}(\vec{x}-\vec{\mu}_c) - \frac{1}{2}\ln|\mathbf{\Sigma}_c| + \ln \pi_c\right). \quad (2)$$

That is, the LDA (QDA) classifier $\hat{y}_L^{\Theta}(\vec{x})$ ($\hat{y}_Q^{\Theta}(\vec{x})$) induced by the parameter configuration Θ *assigns* a class label $c \in \{1, \ldots, C\}$ to a given sample $\vec{x}$. The parameters Θ will, in our setup, be computed w.r.t. an initial dataset $\mathcal{I}$ containing labeled samples $(\vec{x}, c) \in \mathbb{R} \times \{1, \ldots, C\}$. Set $\mathcal{I}_c = \{(\vec{x}, c) \mid (\vec{x}, c) \in \mathcal{I}\}$ represents the initial dataset for a class c.

The pooled within-class covariance matrix $\mathbf{\Sigma}$ combines the covariance matrices of all classes as follows:

Definition 1. *A **pooled within-class covariance matrix** $\mathbf{\Sigma}$ is the weighted average of the covariance matrices $\mathbf{\Sigma}_c$ per class c, where:*

$$\mathbf{\Sigma} = \sum_{c=1}^{C} \frac{|\mathcal{I}_c| - 1}{|\mathcal{I}| - C} \mathbf{\Sigma}_c. \quad (3)$$

We note that, when the pooled within-class covariance matrix $\mathbf{\Sigma}$ or the covariance matrix $\mathbf{\Sigma}_c$ are singular, one uses their *pseudo*-inverse matrices (Moore 1920) instead of their inverse matrices $\mathbf{\Sigma}^{-1}$ and $\mathbf{\Sigma}_c^{-1}$ in Equations (1) and (2).

For simplicity and w.l.o.g., we assume prior probabilities to be equal among classes. Hence, terms $\ln \pi_c$ in Equations (1) and (2) can be dropped, and a parameter configuration Θ is just a tuple of pairs $(\vec{\mu}_c, \mathbf{\Sigma}_c)$, one per class $c \leq C$.

Next, we define the *initial classifier* as follows:

Definition 2. *A **initial classifier**, trained over an initial dataset $\mathcal{I} = \bigcup_{i=1}^{C} \mathcal{I}_c$, is the DA classifier w.r.t. parameter configuration $\Theta = \langle(\vec{\mu}_c, \mathbf{\Sigma}_c) \mid c \in \{1, \ldots, C\}\rangle$, where:*

$$\vec{\mu}_c = \frac{\sum_{\vec{x}\in\mathcal{I}_c} \vec{x}}{|\mathcal{I}_c|}; \quad (4)$$

$$\mathbf{\Sigma}_c = \frac{\sum_{\vec{x}\in\mathcal{I}_c} (\vec{x} - \vec{\mu}_c)(\vec{x} - \vec{\mu}_c)^T}{|\mathcal{I}_c| - 1}. \quad (5)$$

Traditionally, a DA classifier is trained over the entire data at once via *full batch learning*. In our approach, which uses *online active learning* instead, the parameter tuple $\Theta = \langle(\vec{\mu}_c, \mathbf{\Sigma}_c) \mid c \in \{1, \ldots, C\}\rangle$ of a DA *online* classifier is updated using streaming data that is interactively labeled by a user or by a soft-labeling technique (Hoi et al. 2021).

More concretely, the idea is to *incrementally* update the parameter configuration Θ of the initial classifier over time using streaming data represented by a sequence $X_1, \cdots, X_k, \cdots$, where $X_t \subseteq \{\boldsymbol{x} \mid \boldsymbol{x} \in \mathbb{R}^F\}$, with $t \geq 1$, is a set of samples collected at time $t \geq 1$. Importantly, we assume that all samples of a set X_t belong to a single class, as they are samples stemming from the same gesture instance interaction. Then, the parameter configuration $\Theta_t = \langle(\vec{\mu}_{(c,t)}, \mathbf{\Sigma}_{(c,t)}) \mid c \in \{1, \ldots, C\}\rangle$ of the online classifier at time t can be computed using a *label* y_t (when the set X_t is labeled) or a *pseudo-label* $\vec{\omega}_t$ (when the set X_t is unlabeled). Informally, a **pseudo-label** is a vector $\vec{\omega}_t = \langle\omega_{(c,t)} \mid c \in \{1, \ldots, C\}\rangle$ that specifies, for each class c, the "contribution" of dataset X_t to the parameter tuple Θ_t, and is meant to be calculated with a so-called *soft-labeling* technique, as the on we will present in Section 3.

With the general setup in place, we show next, in detail, our proposal on how to update the online classifier using both labels and pseudo-labels.

2.1 Updating using Labeled Data

When the set X_t is labeled with y_t, we propose to update the parameter $\Theta_t = \langle(\vec{\mu}_{(c,t)}, \mathbf{\Sigma}_{(c,t)}) \mid c \in \{1, \ldots, C\}\rangle$ of the online classifier following the spirit of the online LDA

techniques (Kim et al. 2007, 2011; Nie et al. 2009) or dimensionality reduction, albeit some differences to suit the classification task. While these approaches update the between-class and total scatter covariance matrices, under our account, which deals with classification, we update instead the mean vectors $\vec{\mu}_{(c,t)}$ and covariance matrices $\mathbf{\Sigma}_{(c,t)}$ per class c of the online classifier, as shown below in Definition 3.

Concretely, as set X_t becomes available, we incrementally update $\vec{\mu}_{(c,t)}$ and $\mathbf{\Sigma}_{(c,t)}$ (Eq. (6) and (7), resp.) in Θ_{t-1} using the mean vector $\vec{\mu}_{[X_t]}$ and covariance matrix $\mathbf{\Sigma}_{[X_t]}$ computed using the samples in the set X_t, for $c = y_t$ (the label associated to X_t). All other parameters in Θ_{t-1} (for classes different to y_t) remain unchanged. We take Θ_0 to be the parameter configuration Θ of the initial classifier.

Definition 3. *Let Θ_0 be the parameter configuration of the initial classifier when trained over dataset $\mathcal{I} = \bigcup_{i=1}^{C} \mathcal{I}_c$. The **online classifier** incrementally updated w.r.t. $t \geq 1$ data sets $X_1, \cdots, X_t$ with corresponding **labels** $y_1, \cdots, y_t$ is defined using parameter configuration $\Theta_t = \langle (\vec{\mu}_{(c,t)}, \mathbf{\Sigma}_{(c,t)}) \mid c \in \{1, \ldots, C\} \rangle$, where:*

$$\vec{\mu}_{(c,t)} = \begin{cases} \vec{\mu}_c & \text{if } t = 0 \\ \frac{N_{(c,t-1)}\vec{\mu}_{(c,t-1)} + |X_t|\vec{\mu}_{[X_t]}}{N_{(c,t-1)} + |X_t|} & \text{if } y_t = c \\ \vec{\mu}_{(c,t-1)} & \text{otherwise;} \end{cases} \quad (6)$$

$$\mathbf{\Sigma}_{(c,t)} = \begin{cases} \mathbf{\Sigma}_c & \text{if } t = 0 \\ \frac{\boldsymbol{\sigma}^B_{(c,t)} + \boldsymbol{\sigma}^A_{(c,t)}}{N_{(c,t-1)} + |X_t| - 1} & \text{if } y_t = c \\ \mathbf{\Sigma}_{(c,t-1)} & \text{otherwise,} \end{cases} \quad (7)$$

where:

$$\boldsymbol{\sigma}^A_{(c,t)} = \left[\frac{(N_{(c,t-1)}|X_t|)}{N_{(c,t-1)} + |X_t|} \times (\vec{\mu}_{[X_t]} - \vec{\mu}_{(c,t-1)})(\vec{\mu}_{[X_t]} - \vec{\mu}_{(c,t-1)})^T \right];$$

$$\boldsymbol{\sigma}^B_{(c,t)} = (N_{(c,t-1)} - 1)\mathbf{\Sigma}_{(c,t-1)} + (|X_t| - 1)\mathbf{\Sigma}_{[X_t]}$$

$$N_{(c,t)} = \begin{cases} |\mathcal{I}_c| & \text{if } t = 0 \\ N_{(c,t-1)} + |X_t| & \text{if } y_t = c \\ N_{(c,t-1)} & \text{otherwise} \end{cases}$$

The following result (Theorem 1) states that performing *incremental* (small) *updates* on labeled data arising from sequential interactions (as per Definition 3) yields the same classifier as the one obtained by full batch learning at once (over all the data collected). In what follows, we denote $\Theta^\star_t$ to the parameter configuration of the so-called *batch* classifier obtained by training it over the full dataset $\mathcal{I} \cup \{(\vec{x}, y_k) \mid \vec{x} \in X_k, 1 \leq k \leq t\}$ as per Definition 2, when y_i is the labeled of set X_i.

Theorem 1. *Let Θ_t and $\Theta^\star_t$ be the parameter configurations of the **online classifier** (as per Definition 3) and **batch classifier**, resp., w.r.t. an **initial dataset** $\mathcal{I}$ and $t \geq 1$ sets $X_1, \cdots, X_t$ with **labels** $y_1, \cdots, y_t$. Then $\Theta_t = \Theta^\star_t$.*

Therefore, the classification accuracy of these two classifiers is the same. However, at each time t, the online classifier only uses the data in the set X_t and its label y_t, whereas the batch classifier is trained with all the data.

2.2 Updating using Pseudo-Labeled Data

We now show the updating of the parameter configuration Θ_t when the a set X_t is *pseudo-labeled* with a vector $\vec{\omega}_t = \langle \omega_{(c,t)} \mid c \in \{1, \ldots, C\} \rangle$ specifying the "contribution" of X_t to the specific parameters of each class. Technically, $\omega_{(c,t)} \in [0, 1]$, where 1 means that the data in set X_t belongs to class c and 0 means it does not belong to class c.

We first define a generalization of Definition 2 for full batch classifiers in the context of pseudo-labeled data.

Definition 4. *A **weighted batch classifier** trained over the dataset $\mathcal{I} = \bigcup_{i=1}^{C} \mathcal{I}_c$ and $t \geq 1$ sets $X_1, \cdots, X_t$ with their pseudo-labels $\vec{\omega}_1, \cdots, \vec{\omega}_t$ of the form $\vec{\omega}_t = \langle \omega_{(c,t)} \mid c \in \{1, \ldots, C\} \rangle$ is defined using parameter configuration $\Theta'_t = \langle (\vec{\mu'}_{(c,t)}, \mathbf{\Sigma'}_{(c,t)}) \mid c \in \{1, \ldots, C\} \rangle$, where:*

$$\vec{\mu'}_{(c,t)} = \frac{\sum_{\vec{x} \in \mathcal{I}_c} \vec{x} + \sum_{i \leq t} \left(\sum_{\vec{x} \in X_i} \omega_{(c,i)} \vec{x} \right)}{|\mathcal{I}_c| + \sum_{i \leq t} |X_i| \omega_{(c,i)}}; \quad (8)$$

$$\mathbf{\Sigma'}_{(c,t)} = \frac{1}{|\mathcal{I}_c| + \sum_{i \leq t} |X_i| \omega_{(c,i)} - 1} \times \left[\sum_{\vec{x} \in \mathcal{I}_c} (\vec{x} - \vec{\mu}_c)(\vec{x} - \vec{\mu}_c)^T + \sum_{i \leq t} \left(\sum_{\vec{x} \in X_i} \omega_{(c,i)} (\vec{x} - \vec{\mu}_{[X_i]})(\vec{x} - \vec{\mu}_{[X_i]})^T \right) \right]. \quad (9)$$

Thus, a *weighted* batch classifier is trained, at once, over both the initial labeled dataset $\mathcal{I}$ as well as the pseudo-labeled data from $X_1, \cdots, X_t$.

We observe that whereas the covariance matrix $\mathbf{\Sigma}^\star_{(c,t)}$ of the full batch classifier is calculated w.r.t. the mean vector of all (labeled) data $\mathcal{I}_c \cup \bigcup_{i=1}^{t} X_i$, the covariance matrix $\mathbf{\Sigma'}_{(c,t)}$ of the weighted batch classifier (equation (9)) is calculated w.r.t. the mean vectors of the sets $\mathcal{I}_c, X_1, \cdots, X_t$. For example, the weighted covariance of a sample $\vec{x} \in X_t$ in $\mathbf{\Sigma'}_{(c,t)}$ is $\omega_{(c,i)}(\vec{x} - \vec{\mu}_{[X_t]})(\vec{x} - \vec{\mu}_{[X_t]})^T$, whereas the covariance of such sample in $\mathbf{\Sigma}^\star_{(c,t)}$ is $(\vec{x} - \vec{\mu}_{(c,t)})(\vec{x} - \vec{\mu}_{(c,t)})^T$.

The reason why we calculate the covariance matrix $\mathbf{\Sigma'}_{(c,t)}$ w.r.t. the mean vectors of each set $\mathcal{I}_c, X_1, \cdots, X_t$ is to reduce the between-class overlapping of the covariance matrices produced by mislabeled data. In Figure 1, we show the geometric representation (using ellipses) of the covariance matrices of the batch and modified batch classifiers in a time $t = 1$ for a binary classification problem. These covariance matrices are calculated using a set $X_{t=1}$ that is mislabeled (while it actually belongs to class $c = 2$, it is labeled as class $c = 1$). The covariance matrices $\mathbf{\Sigma}^\star_{(c=1,t=1)}$ and $\mathbf{\Sigma}^\star_{(c=1=2,t=1)}$ depicted in Figure 1.a and calculated w.r.t. the mean vector $\vec{\mu}^\star_{(c=1,t=1)}$ overlap more the covariance matrices $\mathbf{\Sigma'}_{(c=1,t=1)}$ and $\mathbf{\Sigma'}_{(c=2,t=1)}$ in Figure 1.b, which is calculated w.r.t. the mean vectors $\vec{\mu'}_{(c=1,t=0)}$ and $\vec{\mu}_{[X_{t=1}]}$.

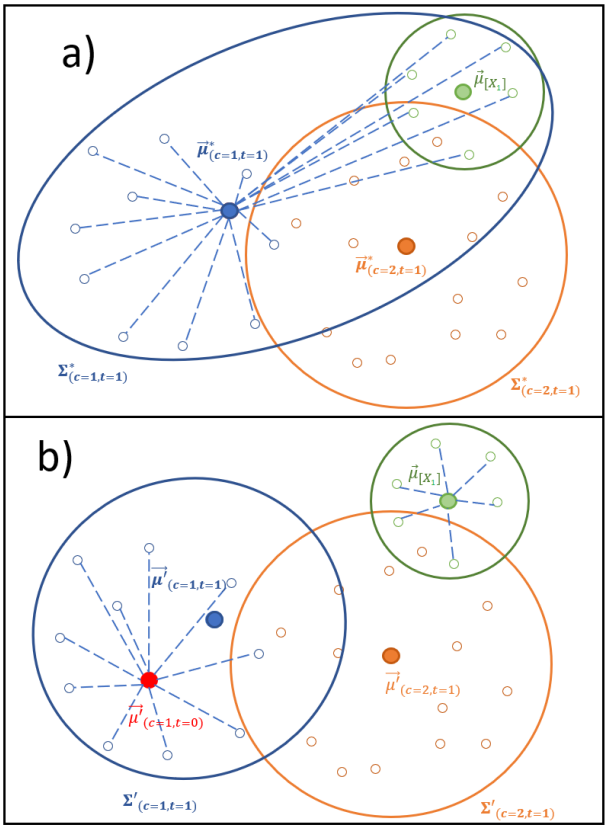

Figure 1: The geometric representation of the covariance matrices of the batch (a) and modified batch (b) classifiers in a time $t = 1$ for a binary classification problem using a mislabeled set X_1.

The fact is that this between-class overlapping can affect the performance of the DA classifiers: it is difficult to establish linear (LDA) or quadratic (QDA) decision boundaries to separate overlapped distributions (Duda, Hart et al. 2006).

Definition 5. *Let Θ_0 be the parameter configuration of the initial classifier when trained over dataset $\mathcal{I} = \bigcup_{i=1}^{C} \mathcal{I}_c$. The **online classifier** incrementally updated w.r.t. $t \geq 1$ data sets $X_1, \cdots, X_t$ with corresponding **pseudo-labels** $\vec{\omega}_1, \cdots, \vec{\omega}_t$ of the form $\vec{\omega}_t = \langle \omega_{(c,t)} \mid c \in \{1, \ldots, C\} \rangle$ is defined using parameter configuration $\Theta_t = \langle (\vec{\mu}_{(c,t)}, \mathbf{\Sigma}_{(c,t)}) \mid c \in \{1, \ldots, C\} \rangle$, where:*

$$\vec{\mu}_{(c,t)} = \begin{cases} \vec{\mu}_c & \text{if } t = 0 \\ \frac{N_{(c,t-1)}\vec{\mu}_{(c,t-1)} + |X_t|\omega_{(c,t)}\vec{\mu}_{[X_t]}}{N_{(c,t-1)} + |X_t|\omega_{(c,t)}} & \text{otherwise;} \end{cases} \quad (10)$$

$$\mathbf{\Sigma}_{(c,t)} = \begin{cases} \mathbf{\Sigma}_c & \text{if } t = 0 \\ \frac{\boldsymbol{\sigma}^A_{(c,t)}}{N_{(c,t-1)} + |X_t|\omega_{(c,t)} - 1} & \text{otherwise;} \end{cases} \quad (11)$$

where:

$$\boldsymbol{\sigma}^A_{(c,t)} = (N_{(c,t-1)} - 1)\mathbf{\Sigma}_{(c,t-1)} + \omega_{(c,t)}(|X_t| - 1)\mathbf{\Sigma}_{[X_t]}$$

$$N_{(c,t)} = \begin{cases} |\mathcal{I}_c| & \text{if } t = 0 \\ N_{(c,t-1)} + |X_t|\omega_{(c,t)} & \text{otherwise.} \end{cases}$$

Similar to Section 2.1, the parameters $\vec{\mu}_{(c,t)}$ and $\mathbf{\Sigma}_{(c,t)}$ (Equations (10) and (11)) are incrementally/sequentially updated using the mean vector $\vec{\mu}_{[X_t]}$, covariance matrix $\mathbf{\Sigma}_{[X_t]}$, and pseudo-label $\vec{\omega}_t$ of the streaming set X_t.

It turns out that the Theorem 1 result for incremental updates over labeled data can be carried out to updates over pseudo-labeled data, as Theorem 2 demonstrates. The proofs of these two theorems are in Appendix A.

Theorem 2. *Let Θ_t and Θ'_t be the parameter configurations corresponding to the **online classifier** (as per Definition 5) and the weighted batch classifier (as per Definition 4), respectively, over an **initial dataset** $\mathcal{I}$ and $t \geq 1$ **streaming sets** $X_1, \cdots, X_t$ with their corresponding **pseudo-labels** $\vec{\omega}_1, \cdots, \vec{\omega}_t$. Then, $\Theta_t = \Theta'_t$.*

That is, no accuracy is given up when the classifier is incrementally updated as streaming data is gathered.

We close this section by performing a complexity analysis on the calculations required for obtaining the parameter configurations under the various settings above. As expected, the time complexity to compute the parameter configurations $\Theta^\star_t$ (namely, $|\mathcal{I} \cup \bigcup_{i=1}^{t} X_i|(F^2 + 2F)$) and Θ'_t (namely, $|\mathcal{I} \cup \bigcup_{i=1}^{t} X_i|(F^2 + 2F + 1)$) of the batch approaches increases with t, the number of streaming "steps." In contrast, the time complexity for computing the parameter configuration Θ_t of the online classifier using labeled (namely, $|X_t|C(F^2+2F)+C(5F^2+F+9)$) and pseudo-labeled (namely, $|X_t|C(F^2+2F)+C(4F^2+6)$) data depend only on the number of samples $|X_t|$ in the streaming set X_t. Thus, provided the size of the streaming data at each step is fixed, the time complexity of the online classifier is constant (i.e., $\mathcal{O}(1)$), and hence adequate for streaming applications as hand gesture recognition via myoelectric interfaces.

With a method at hand to update the classifier using pseudo-labeled data, what remains is an actual way of obtaining those labels. This is the focus of the next section.

3 A Soft-Labeling Technique

We present here a soft-labeling technique to pseudo-label a set X_t, in which we calculate a weight $\omega_{(c,t)}$ of the pseudo-label $\vec{\omega}_t = \langle \omega_{(c,t)} \mid c \in \{1, \ldots, C\} \rangle$, which determines the contribution of the set X_t. This weight $\omega_{(c,t)}$ is a weighted average of a probability $p_{(c,t)}$, and a normalized *Matthews correlation coefficient* (MCC) $m_{(c,t)}$. This weighted average uses Shannon entropy (Shannon 1948) to decrease the contribution of the set X_t when the uncertainty of the probability $p_{(c,t)}$ and the coefficient $m_{(c,t)}$ increase.

This term $p_{(c,t)}$ is the posterior probability of the samples $x \in X_t$ belong to class c given the initial (LDA/QDA) classifier $\hat{y}^{\Theta}(\vec{x})$ trained over $\mathcal{I}$, as follows:

$$p_{(c,t)} = P(\hat{y}^{\Theta}(\vec{x}) = c \mid \vec{x} \in X_t). \quad (12)$$

MCC is a metric that determines the performance of a classifier $\hat{y}$ over a test dataset $\mathcal{T}$ (Matthews 1975), and it is calculated with true positives TP, false positives FP, true negatives TN and false negatives FN. Their interval is $[-1, +1]$, where -1 means a total disagreement and $+1$ is a total agreement between prediction and observation. We

will use 0 as the minimum value because negative values mean the performance of the classifier is not better than a random classifier, in this way:

$$mcc(\hat{y}, \mathcal{T}) = \max\{0, \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}\}. \tag{13}$$

In our approach, MCC determines the performance of a DA classifier $\hat{y}_c^{\Theta_t}(\vec{x})$ over the re-labeled training set $\bar{\mathcal{I}}_c$ that is the set $\mathcal{I}$ (initial labeled dataset) by re-labeling every non-c sample to a distinguished class 0, as follows:

$$\bar{\mathcal{I}}_c = \{(\vec{x}, y) \mid (\vec{x}, y) \in \mathcal{I}_c\} \cup \{(\vec{x}, 0) \mid (\vec{x}, y) \in \mathcal{I} \cap \mathcal{I}_c\}. \tag{14}$$

The DA classifier $\hat{y}_c^{\Theta_t}(\vec{x})$ is a variant of the initial classifier $\hat{y}^{\Theta}(\vec{x})$ at time t, where $\vec{\mu}_c$ and $\mathbf{\Sigma}_c$ are replaced with the mean vector $\mu_{[\vec{X}_t]}$ and covariance matrix $\mathbf{\Sigma}_{[\boldsymbol{X_t}]}$ from X_t respectively. So, we calculate the normalize MCC $m_{(c,t)}$ using the DA classifier $\hat{y}_c^{\Theta_t}(\vec{x})$ and the re-labeled training set $\bar{\mathcal{I}}_c$, in this way:

$$m_{(c,t)} = \frac{mcc(\hat{y}_c^{\Theta_t}(\vec{x}), \bar{\mathcal{I}}_c)}{\sum_{i=1}^{C} mcc(\hat{y}_i^{\Theta_t}(\vec{x}), \bar{\mathcal{I}}_i)}. \tag{15}$$

To determine the uncertainty of the probability $p_{(c,t)}$ and the coefficient $m_{(c,t)}$, we calculate the two Shannon entropy $H(p_{(c,t)})$ and $H(m_{(c,t)})$ as follows:

$$H(p_{(c,t)}) = -\sum_{c=1}^{C} p_{(c,t)} \log p_{(c,t)}, \tag{16}$$

$$H(m_{(c,t)}) = -\sum_{c=1}^{C} m_{(c,t)} \log m_{(c,t)}. \tag{17}$$

And, we calculate $\omega_{(c,t)}$ as the pooled average of $p_{(c,t)}$ and $m_{(c,t)}$ using their entropy, as follows:

$$\omega_{(c,t)} = \frac{(1 - H(p_{(c,t)}))p_{(c,t)} + (1 - H(m_{(c,t)}))m_{(c,t)}}{2}. \tag{18}$$

Note that when $H(p_{(c,t)})$ or $H(m_{(c,t)})$ are equal to one, the uncertainty of the probability $p_{(c,t)}$ or of the coefficient $m_{(c,t)}$ are maximum, respectively. For example, if $H(p_{(c,t)}) = 1$, then the weight $\omega_{(c,t)}$ is at most 0.5.

4 Empirical Analysis

We evaluate our online classifier through two experiments to determine their performance (Section 4.2) and their robustness to mislabeled data (Section 4.3). In these experiments, we use three feature sets and five publicly-available datasets that contain hand gestures' sEMG data. We make publicly available the code[1] (including all suplementary material) of this empirical evaluation to easily compare our approach with future approaches in this field.

[1]https://github.com/andresjarami/OnlineDAclassifier

4.1 Datasets and Feature Sets

We use five sEMG datasets that are: NinaPro5 (Atzori et al. 2014; Pizzolato et al. 2017), CapgMyo_dbb (Du et al. 2017), MyoArmband (Côté-Allard et al. 2019), Long-Term 3DC (Cote-Allard et al. 2021), and EMG-EPN-120 (Chung and Benalcázar 2019). We use one gesture per class for training and the other gestures for updating and testing of the proposed model according to the description in Table 1, which shows the characteristics of these datasets.

The data of these five datasets are pre-processed to simulate an online scenario. Based on that, we use the sliding windowing technique (Jaramillo-Yánez, Benalcázar, and Mena-Maldonado 2020) to segment the sEMG data acquired by multiple channels in windows of 290ms with an overlap of 280ms. For example, from a hand gesture that lasted 5 seconds, we get 942 windows with a sampling rate of 200Hz.

From each window observation, we get a feature vector. For this purpose, we used three feature sets, which are suitable to develop real-time interfaces due to their low calculation time (Jaramillo-Yánez, Benalcázar, and Mena-Maldonado 2020). The first feature set (**FS1**) is proposed by Hahne, Graimann, and Muller (2012), and is composed only of one feature that is the logarithm of the data variance. Hudgins, Parker, and Scott (1993) proposed the second set (**FS2**), which has four features (mean absolute value, waveform length, zero crossing, and slope sign change) and is widely used to develop myoelectric interfaces. Phinyomark, N Khushaba, and Scheme (2018) proposed the third set (**FS3**), which has four features (L-scale, maximum fractal length, mean of the square root, and Willison amplitude). For example, when the feature set has 4 functions and the sEMG has 8 channels, we get a feature vector of 32 elements.

Note that the sEMG data of a gesture can be represented as a set of feature vectors (one per window). So, we selected randomly (with a uniform distribution) a sequence of sets (gestures) to update the online classifier simulating streaming data from users. All the results reported in this paper are an average of 20 runs per user in each dataset.

4.2 EXP1: Online Classifier Performance

To evaluate the performance of our approach, we define five DA classifiers (LDA/QDA): initial (baseline), online classifier using labels and pseudo-labels through our soft-labeling, Nigam's soft-labeling (Nigam et al. 2000), and thresholding (Van Engelen and Hoos 2020; Triguero, García, and Herrera 2015) techniques.

The *initial classifier* is a DA classifier trained over a dataset $\mathcal{I}$ that has one gesture per class. The *online classifier using labels and pseudo-labels* is a DA classifier initially trained over the set $\mathcal{I}$ and sequentially updated *with labeled gestures* (shown in section 2.1) and *with pseudo-labeled gestures* by our soft-labeling technique (shown in sections 2.2 and 3), respectively. The *Nigam-based classifier* is the online classifier updated with pseudo-labeled gestures using Nigam's soft-labeling technique (Nigam et al. 2000). In this technique, a gesture is pseudo-labeled using the conditional posterior probability $p_{(c,t)}$ (shown in equation (12)) multiplied by a parameter λ that decreases the contribution of

Dataset	Number of channels	Number of users	Number of classes	Gesture per class
NinaPro5	16	10	18	1(train), 3(update), 2(test)
Côté-Allard_db	8	19	7	1(train), 8(update), 3(test)
EMG-EPN-612	8	612	6	1(train), 24(update), 25(test)
CapgMyo_dbb	128	10	8	1(train), 6(update), 3(test)
LongTermEMG_dbb	10	20	11	1(train), 1(update), 1(test)

Table 1: Datasets used in the Experiments

DA classifier	Nigam	Thresholding	Ours
LDA	2.1	**2.0***	**2.0***
QDA	2.1	2.2	**1.7***

* Best ranked classifier at $p_{value} < 0.5$.

Table 2: Friedman test's average ranks of the accuracy of the DA classifiers (LDA/QDA) that use pseudo-labeled gestures over the five datasets.

this probability to minimize the error of gestures incorrectly pseudo-labeled. The parameter λ is in the interval $[0, 1]$. The *thresholding-based classifier* is also the online classifier updated with pseudo-labeled gestures using the thresholding technique that is commonly used in self-training learning (Van Engelen and Hoos 2020; Triguero, García, and Herrera 2015). In this technique, a gesture is labeled based on the probability $p_{(c,t)}$. If this probability is greater than a threshold τ, then this pseudo-labeled gesture is used to update the classifier. In this experiment, we determine the best parameters λ and τ (shown in Appendix B) for each dataset and feature set from the set $\{0, 0.1, \cdots, 1\}$ using grid search optimization.

For the five DA classifiers, Figure 2 shows the average classification accuracy of the users in the five datasets using the three feature sets described above. To determine if the accuracy differences between the methods tested are statistically significant, we use the 2-tailed Wilcoxon signed ranks test at $p_{value} < 0.5$ (Demšar 2006). We use black arrows to indicate that the accuracy come from the same distribution (there is no statistical difference). As we excepted, the accuracy of the online classifier using labels is higher than the accuracy of the other classifiers that use pseudo-labeled gestures. Note that this accuracy is equal to the accuracy of a DA classifier trained with all data (the initial set $\mathcal{I}$ and all labeled gestures) in full batch fashion, as we established in Theorem 1. The accuracies of the online classifier using labels and pseudo-labels are higher than the accuracy of the initial classifier, so the updating proposed in sections 2.1 and 2.2 improves the performance of a DA classifier trained with few samples (one gesture per class). In contrast, the Nigam-based and thresholding-based classifiers perform worse than the initial classifier when the DA classifier is QDA as we can see, for example, in NinaPro5 and Long-Term 3DC using the feature sets FS2 and FS3, and in Capgmyo_dbb using the feature set FS2.

To evaluate the classifiers that are updated with pseudo-labeled gestures (the online classifier using pseudo-labels and the Nigam-based and thresholding-based classifiers),

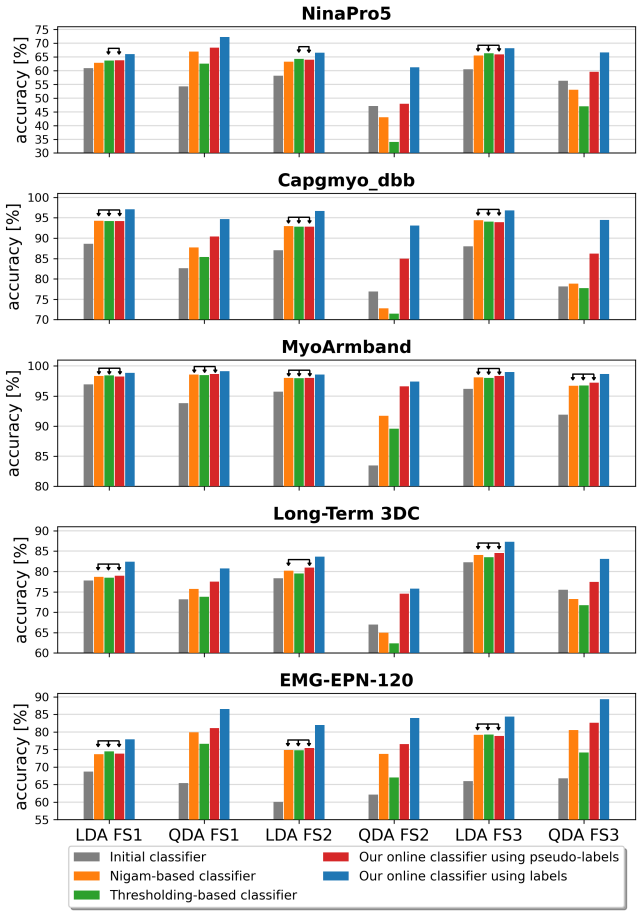

Figure 2: The classification accuracy average of the five DA classifiers in the five datasets using the three feature sets.

we rank their accuracy in the five datasets using the Friedman rank test and the Holm post-hoc test at $p_{value} < 0.5$ (Demšar 2006). Table 2 shows the Friedman test's average ranks of the accuracies of these three classifiers, where the online classifier using pseudo-labels and the thresholding-based classifier are best ranked for LDA, and our online classifier is also best ranked for QDA.

4.3 EXP2: Robustness to Mislabeled Data

In this experiment, we show the robustness to mislabeled data of our online classifier through a new way of calculating the covariance matrix (COV). Traditionally, the COV of DA classifiers is calculated w.r.t. the mean vector of all gestures,

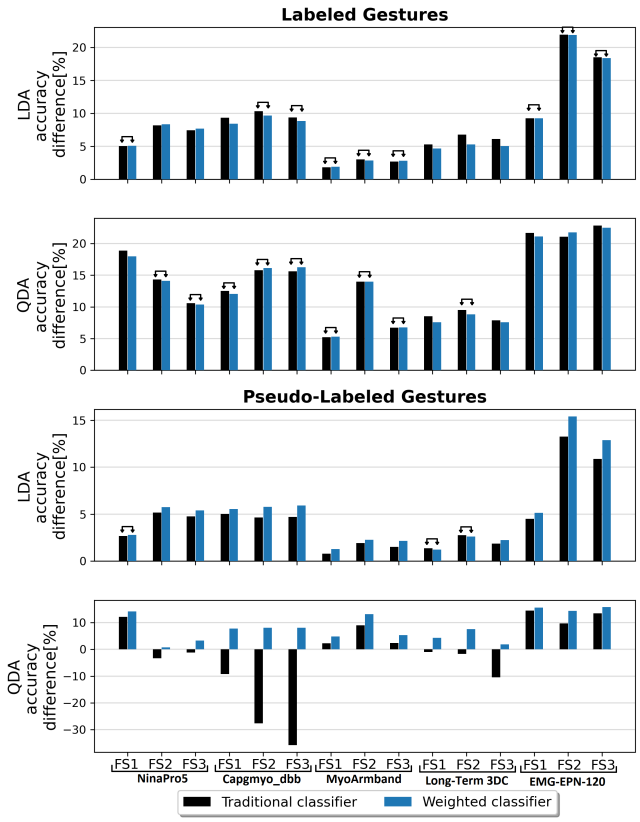

Figure 3: The accuracy difference of the two batch classifiers (traditional and weighted classifiers).

whereas we calculate w.r.t. the mean vectors of each gesture (equation (9)). Concretely, we compare the performance of the weighted batch classifier (shown in definition 4) and a version of this weighted classifier (from now on called traditional classifier), in which the covariance matrix (COV) is calculated w.r.t. the mean vector of all gestures. Note that our online classifier using pseudo-labeled gestures has the same performance as the weighted batch classifier, as shown Theorem 2.

We show the accuracy difference of these two compared batch classifiers w.r.t. the initial classifier using labeled and pseudo-labeled gestures as shown in Figure 3. When the accuracy of any of these classifiers is higher than the initial classifier, the accuracy difference is positive; otherwise, it is negative. We use the 2-tailed Wilcoxon signed ranks test at $p_{value} < 0.5$ to determine if the accuracy differences are statistically significant (Demšar 2006). When these two batch classifiers use labeled gestures, their performance is similar for LDA and QDA. In contrast, when they use pseudo-labeled gestures, the performance of the weighted batch classifier is significantly higher than the difference of the traditional classifier. In fact, the performance of the traditional classifier is worse than the performance of the initial classifier in several cases.

We also rank the accuracy differences of these two batch classifiers (with labeled and pseudo-labeled gestures) in the five datasets using the Friedman rank test and the Holm post-hoc test at $p_{value} < 0.5$ (Demšar 2006). Table 3 shows the Friedman test's average ranks of these two classifiers, where there is not a statistical difference between these two batch classifiers using labels for both LDA and QDA. However, using pseudo-labeled gestures, the weighted batch classifier is best-ranked than the traditional classifier. As we shown in Section 2.2, the performance of the traditional classifier is affected by the between-class overlapping of the covariance matrices due to mislabeled data (Figure 1). Consequently, the weighted batch classifier and the online classifier using pseudo-labels (that has the same performance as the weighted batch classifier) are robust to mislabeled gestures.

Gestures	DA	Classifier	
		Traditional	Weighted
labeled	LDA	**1.5***	**1.5***
	QDA	**1.5***	**1.5***
pseudo-labeled	LDA	1.2	**1.9***
	QDA	1.2	**1.8***

* Best ranked classifier at $p_{value} < 0.5$.

Table 3: Friedman test's average ranks of the two batch classifiers' accuracy using labeled and pseudo-labeled gestures.

5 Conclusions

We proposed a DA classifier that uses *online active learning* and a *soft-labeling* technique. The account aims to deal with the frequent and demanding training of myoelectric interfaces due to covariate shift. Specifically, we proposed to initially train an classifier with a small dataset and then use labeled/pseudo-labeled streaming data to update its parameters over time. Besides proposing an update mechanism, we also developed a soft-labeling technique. We proved, theoretically, that our online classifier yields the same model as training a DA classifier via full batch learning, that is, by training with all data at once. Importantly, unlike with full batch learning, the time complexity of our online incremental approach remains constant (provided the streaming size is fixed). Thus, we argue the approach is suitable to be implemented in streaming data applications like myoelectric interfaces. Through five publicly available datasets, we demonstrated experimentally that our proposal improves the performance of DA classifiers and reduces the updating error due to mislabeled data, and that our soft-labeling technique yields better performance than the state-of-the-art models.

Up to our knowledge, this is the first DA classifier that uses online active learning. While we evaluated the technique in the context of hand gesture classification using sEMG, we conjecture it could also be used effectively in other applications of myoelectric interfaces, and leave that for further work. Other further work remains interesting to pursue. First, an analysis of our online classifier using *imbalance-class* streaming data. Second, a limitation of our approach also worth investigating involves cases in which the small training data collected from the user is of poor quality (e.g., data from amputees).

References

Atzori, M.; Gijsberts, A.; Castellini, C.; Caputo, B.; Hager, A.; Elsig, S.; Giatsidis, G.; Bassetto, F.; and Müller, H. 2014. Electromyography data for non-invasive naturally-controlled robotic hand prostheses. *Scientific data*, 1(1): 1–13.

Castellini, C.; Fiorilla, A. E.; and Sandini, G. 2009. Multi-subject/daily-life activity EMG-based control of mechanical hands. *JNER*, 6(1): 1–11.

Chung, E. A.; and Benalcázar, M. E. 2019. Real-Time Hand Gesture Recognition Model Using Deep Learning Techniques and EMG Signals. In *Proc. of USIPCO*, 1–5.

Côté-Allard, U.; Fall, C. L.; Drouin, A.; Campeau-Lecours, A.; Gosselin, C.; Glette, K.; Laviolette, F.; and Gosselin, B. 2019. Deep Learning for Electromyographic Hand Gesture Signal Classification Using Transfer Learning. *IEEE TNSRE*, 27(4): 760–771.

Cote-Allard, U.; Gagnon-Turcotte, G.; Phinyomark, A.; Glette, K.; Scheme, E.; Laviolette, F.; and Gosselin, B. 2021. A Transferable Adaptive Domain Adversarial Neural Network for Virtual Reality Augmented EMG-Based Gesture Recognition. *IEEE TNSRE*.

Demšar, J. 2006. Statistical comparisons of classifiers over multiple data sets. *JMLR*, 7(Jan): 1–30.

Dhamecha, T. I.; Singh, R.; and Vatsa, M. 2016. On incremental semi-supervised discriminant analysis. *Pattern Recognition*, 52: 135–147.

Du, Y.; Jin, W.; Wei, W.; Hu, Y.; and Geng, W. 2017. Surface EMG-based inter-session gesture recognition enhanced by deep domain adaptation. *Sensors*, 17(3): 458.

Duda, R. O.; Hart, P. E.; et al. 2006. *Pattern classification*. John Wiley & Sons.

Farina, D.; Jiang, N.; Rehbaum, H.; Holobar, A.; Graimann, B.; Dietl, H.; and Aszmann, O. C. 2014. The extraction of neural information from the surface EMG for the control of upper-limb prostheses: Emerging avenues and challenges. *IEEE TNSRE*, 22(4): 797–809.

Fisher, R. A. 1936. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2): 179–188.

Fukunaga, K. 2013. *Introduction to statistical pattern recognition*. Elsevier.

Hahne, J. M.; Graimann, B.; and Muller, K.-R. 2012. Spatial filtering for robust myoelectric control. *IEEE TBME*, 59(5): 1436–1443.

Härdle, W. K.; and Simar, L. 2019. *Applied multivariate statistical analysis*. Springer.

Hoi, S. C.; Sahoo, D.; Lu, J.; and Zhao, P. 2021. Online Learning: A Comprehensive Survey. *Neurocomputing*.

Huang, Q.; Yang, D.; Jiang, L.; Zhang, H.; Liu, H.; and Kotani, K. 2017. A novel unsupervised adaptive learning method for long-term electromyography (EMG) pattern recognition. *Sensors*, 17(6): 1370.

Hudgins, B.; Parker, P.; and Scott, R. N. 1993. A new strategy for multifunction myoelectric control. *IEEE TBME*, 40(1): 82–94.

Jaramillo-Yánez, A.; Benalcázar, M. E.; and Mena-Maldonado, E. 2020. Real-Time Hand Gesture Recognition Using Surface Electromyography and Machine Learning: A Systematic Literature Review. *Sensors*, 20(9): 2467.

Kaufmann, P.; Englehart, K.; and Platzner, M. 2010. Fluctuating EMG signals: Investigating long-term effects of pattern matching algorithms. In *Proc. of EMBC*, 6357–6360.

Kim, T.-K.; Stenger, B.; Kittler, J.; and Cipolla, R. 2011. Incremental linear discriminant analysis using sufficient spanning sets and its applications. *IJCV*, 91(2): 216–232.

Kim, T.-K.; Wong, S.-F.; Stenger, B.; Kittler, J.; and Cipolla, R. 2007. Incremental linear discriminant analysis using sufficient spanning set approximations. In *Proc. of CVPR*, 1–8.

Liu, J.; Sheng, X.; Zhang, D.; He, J.; and Zhu, X. 2014. Reduced daily recalibration of myoelectric prosthesis classifiers based on domain adaptation. *IEEE JBHI*, 20(1): 166–176.

Matsubara, T.; and Morimoto, J. 2013. Bilinear modeling of EMG signals to extract user-independent features for multiuser myoelectric interface. *IEEE TBME*, 60(8): 2205–2213.

Matthews, B. W. 1975. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta-Protein Structure*, 405(2): 442–451.

Moore, E. H. 1920. On the reciprocal of the general algebraic matrix. *Bull. Am. Math. Soc.*, 26: 394–395.

Nie, F.; Xiang, S.; Jia, Y.; and Zhang, C. 2009. Semi-supervised orthogonal discriminant analysis via label propagation. *Pattern Recognition*, 42(11): 2615–2627.

Nigam, K.; McCallum, A. K.; Thrun, S.; and Mitchell, T. 2000. Text classification from labeled and unlabeled documents using EM. *Machine learning*, 39(2): 103–134.

Pang, S.; Ozawa, S.; and Kasabov, N. 2005. Incremental linear discriminant analysis for classification of data streams. *IEEE Transactions on SMC Part B*, 35(5): 905–914.

Patricia, N.; Tommasit, T.; and Caputo, B. 2014. Multi-source adaptive learning for fast control of prosthetics hand. In *Proc. of ICPR*, 2769–2774. IEEE.

Phinyomark, A.; N Khushaba, R.; and Scheme, E. 2018. Feature extraction and selection for myoelectric control based on wearable EMG sensors. *Sensors*, 18(5): 1615.

Pizzolato, S.; Tagliapietra, L.; Cognolato, M.; Reggiani, M.; Müller, H.; and Atzori, M. 2017. Comparison of six electromyography acquisition setups on hand movement classification tasks. *PloS one*, 12(10): e0186132.

Sensinger, J. W.; Lock, B. A.; and Kuiken, T. A. 2009. Adaptive pattern recognition of myoelectric signals: exploration of conceptual framework and practical algorithms. *IEEE TNSRE*, 17(3): 270–278.

Shannon, C. E. 1948. A mathematical theory of communication. *The Bell system technical journal*, 27(3): 379–423.

Shwedyk, E.; Balasubramanian, R.; and Scott, R. 1977. A nonstationary model for the electromyogram. *IEEE TBME*, (5): 417–424.

Triguero, I.; García, S.; and Herrera, F. 2015. Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study. *Knowledge and Information systems*, 42(2): 245–284.

Van Engelen, J. E.; and Hoos, H. H. 2020. A survey on semi-supervised learning. *Machine Learning*, 109(2): 373–440.

Wang, S.; Lu, J.; Gu, X.; Du, H.; and Yang, J. 2016. Semi-supervised linear discriminant analysis for dimension reduction and classification. *Pattern Recognition*, 57: 179–189.

Wang, Y.; Fan, X.; Luo, Z.; Wang, T.; Min, M.; and Luo, J. 2017. Fast online incremental learning on mixture streaming data. In *Proc. of AAAI*, volume 31.