1) The reference electrodes for both muscles were placed at the elbow bone (see figure 1a). This position was chosen since motion of the wrist generates limited movement of the elbow joint. To maximize the voltage potential difference between the two recording electrodes, one was placed at the center of the flexor/extensor and the second near the elbow. Finally, an automatic labeling system based on a predetermined sequence of flexion (1), no movement (0) and extension (-1) with visual validation to guide the user was implemented to ensure that the training dataset was acquired properly (see figure 1b). A long pause between flexion and/or extension was used to ensure return to baseline signal and proper separation of each movement in the sequence.

2) Channel 1 and channel 2 raw data were standardized before processing (see figure 1c), with $\overline{x}$ and $\sigma$ computed during training for each channel and then applied for each window during the validation. This ensures that both channel 1 and channel 2 means are centered to 0 and are of comparable in distribution with a standard deviation of 1. Since each window has a limited number of samples and to make live application possible, a time-domain lowpass filter with a Hanning convolution kernel was implemented. The filter was applied on each acquired window. However, since feature performances decreased on the ANOVA score (see table 1 and figure 2), the filter was removed in the final version of the code.

3) Time domain features were selected accordingly to the literature [1, 2]. No feature was extracted directly in the Fourier domain considering the low number of samples in each window. The features that were analyzed include mean absolute value (MAV), mean absolute value slope (MAVS), variance (VAR), root mean square (RMS), wavelength (WL), zero crossings (ZC) and slope sign changes (SSC). The distribution of each feature were then analyzed with respect to their label (flexion, no movement and extension) (see figure 2). For each of these features, an one-way ANOVA test was performed (see table 1). Results showed than the MAVS feature did not reached statistical significance. For this reason, the MAVS was removed from the training model. All other features had a $p<0.0001$ when tested on the ANOVA and therfore were kept into the model.

4) A multiclass logistic regression model (LRM) was implemented for the classification (see appendix for details). This solution was chosen since LRM is relatively simple to implement from scratch and allows fast prediction. To improve performance, features were all standardized before classification training, meaning each feature has a mean of 0 and a standard deviation of 1. This ensure that all features have equal contribution to the loss function and avoid possible bias caused by difference of scale in unit value of each feature. To optimize hyper-parameters, the learning rate and regularization parameters $\eta$ and $\mu$ were heuristic tested with different values (see figure 3a) and optimal values were selected a posteriori after monitoring of the loss function ( 1000 iterations with $\eta = 0.5$ and $\mu = 0$). To improve temporal prediction stability, a memory system based on a weighted average of previous and present probabilities was implemented (see appendix).

5) Since data acquisition was simple and straightforward, two separated acquisitions were performed in order to build a training and a validation set, rather than using a split system (see figure 3b). This also ensures that the offline testing uses the same training dataset dimension than for the live demonstration. Prediction was performed on the validation dataset to evaluate model performance (see 3c). The model was fined tune by monitoring the loss on the validation dataset to avoid overfitting during training (see figure 3d). A confusion matrix was computed to evaluate global performance of the model (see figure 3d). Globally, performance reached 89.9 % on the validation dataset, with higher accuracy for the no movement class ( 95.25 %), followed by extension ( 82.5 %) and flexion ( 76 %). Interestingly, no false positives were detected between flexion and extension.

6) The automatic labeling system was used to build the training dataset by using the same flexion/extension sequence used for offline training. The training dataset was also of the same dimension to ensure good reproducibility. After visual confirmation of quality of the signal acquisition and labeling, the training was executed automatically with the same pre-processing, features selection and hyper-parameters established previously. Live testing was made on successive time window with live display of the label.

7) The system performed well during the demo day with fast responsiveness and accurate distinction of flexion and extension. Some false positives were detected during the first trial at rest, which was caused by shaking the hand due to stress. Live retraining of the model while the hand was still shaking considerably reduced false detections while preserving fast responsiveness.

8) The drawback of using segmented windows for live classification is that the window length causes a delayed responsiveness in the system. To use of overlapping or sliding window system would be beneficial for a faster live response. The model used is also purely mathematics with no physiological understanding of EMG signals and mechanism of flexion and extension of the wrist. A better system would used prior information and would exclude the probability of an extension following immediately a flexion for instance.

# 1 References

[1] B. Hudgins, P. Parker, and R. N. Scott, "A new strategy for multifunction myoelectric control," *IEEE transactions on biomedical engineering*, vol. 40, no. 1, pp. 82–94, 1993.

[2] Q. Li, A. Zhang, Z. Li, and Y. Wu, "Improvement of emg pattern recognition model performance in repeated uses by combining feature selection and incremental transfer learning," *Frontiers in Neurorobotics*, vol. 15, p. 699174, 2021.

# Appendix

Table 1: ANOVA scores for the different features

| | Without filter | | With filter | |
|---|---|---|---|---|
| Features | f-value | p-value | f-value | p-value |
| MAV1 | 571.34 | <0.0001 | 305.85 | <0.0001 |
| MAV2 | 95.69 | <0.0001 | 80.65 | <0.0001 |
| MAVS1 | 1.59 | 0.20 | 0.84 | 0.43 |
| MAVS2 | 0.03 | 0.97 | 0.013 | 0.99 |
| VAR1 | 798.06 | <0.0001 | 626.86 | <0.0001 |
| VAR2 | 34.49 | <0.0001 | 21.88 | <0.0001 |
| RMS1 | 655.13 | <0.0001 | 364.07 | <0.0001 |
| RMS2 | 109.94 | <0.0001 | 88.47 | <0.0001 |
| WL1 | 1305.30 | <0.0001 | 1183.59 | <0.0001 |
| WL2 | 519.68 | <0.0001 | 397.04 | <0.0001 |
| ZC1 | 430.64 | <0.0001 | 330.46 | <0.0001 |
| ZC2 | 33.97 | <0.0001 | 18.18 | <0.0001 |
| SSC1 | 392.39 | <0.0001 | 162.18 | <0.0001 |
| SSC2 | 54.99 | <0.0001 | 19.90 | <0.0001 |

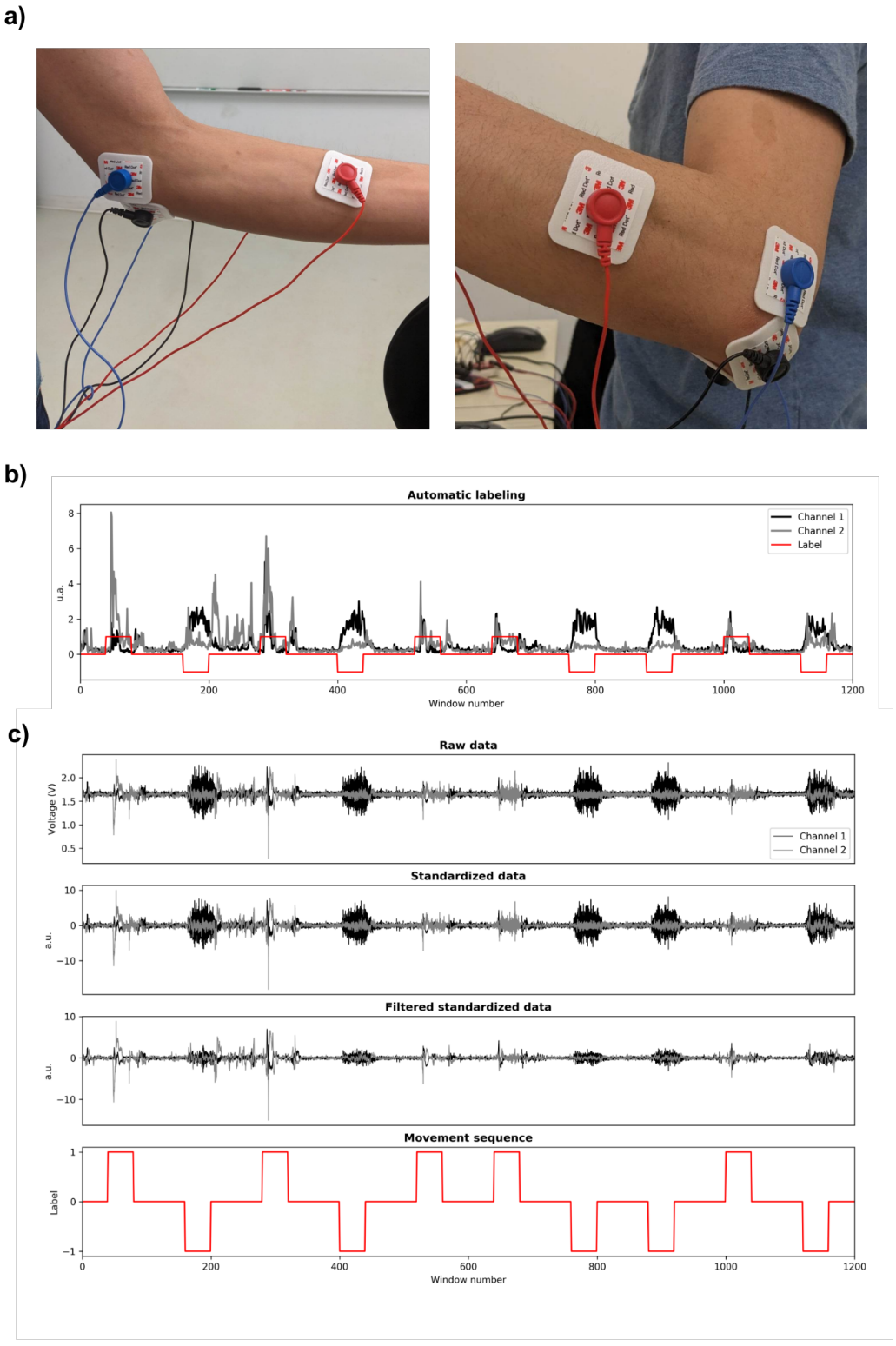


Figure 1: Setting for the EMG recordings. a) EMG electrode positions. d) Automatic labeling of the training dataset. c) Exemple of EMG recordings for raw, standardized and filtered data. Movement sequence labels are defined by flexion=1, extension = -1 and no movement =0.

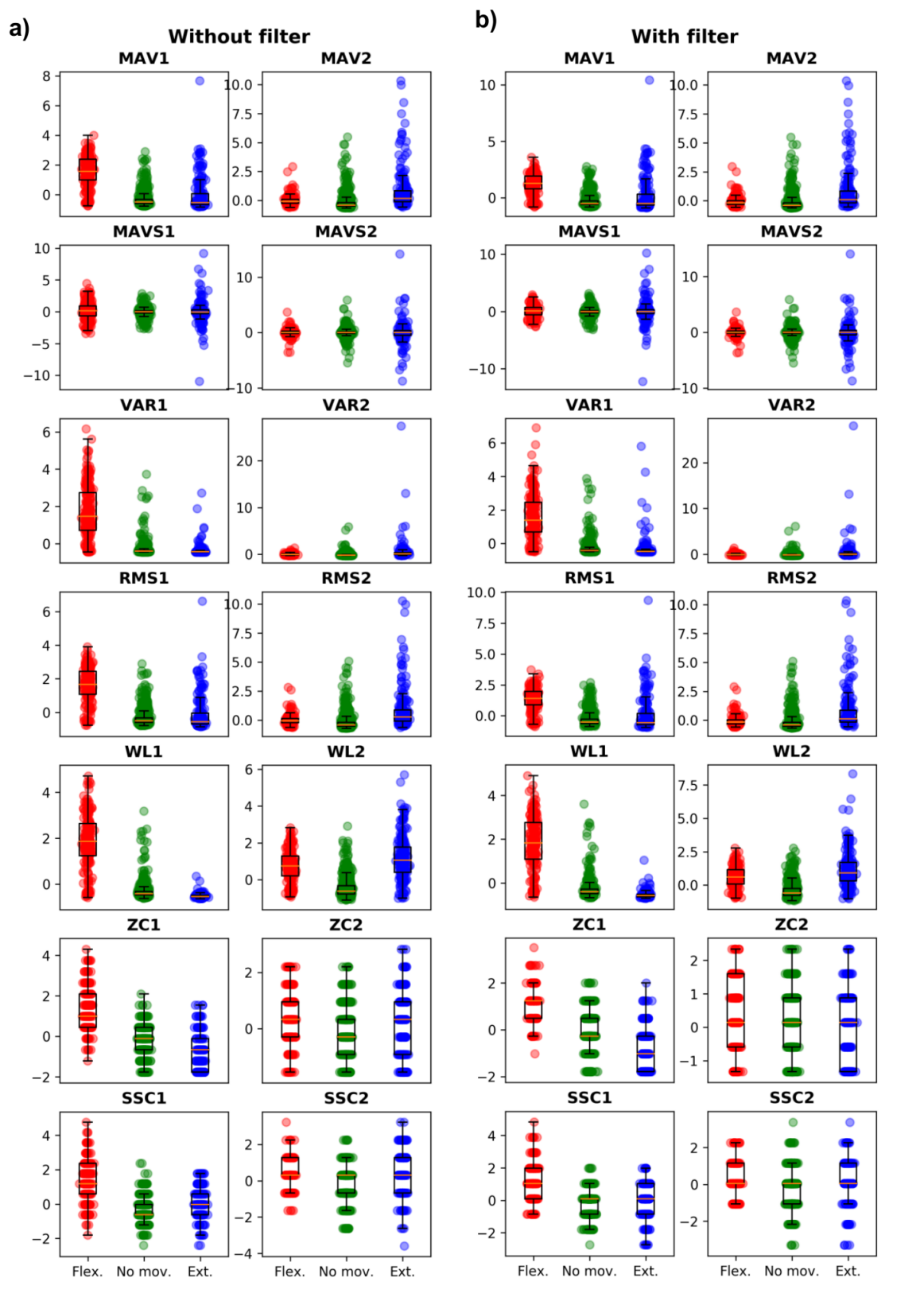


Figure 2: Feature distribution for each label without filter (a) and with filter (b). MAV : mean absolute value, MAVS: mean absolute value slope, VAR: variance, RMS: root mean square, WL: wavelength, ZC: zero crossings, SSC: slope sign changes. Number 1 and 2 indicates channel 1 and channel 2, respectively.

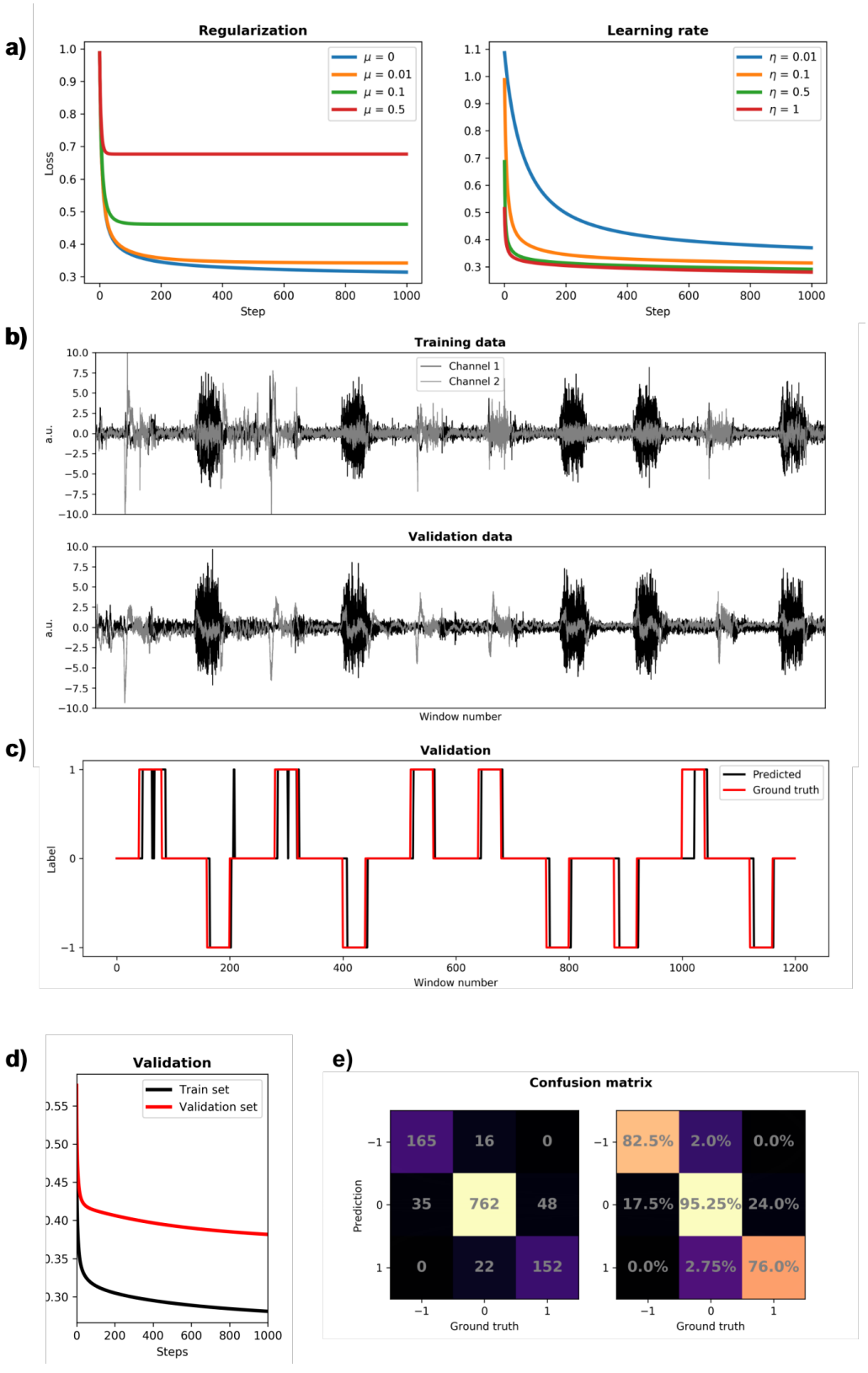


Figure 3: Model training and validation. a) Hyperparameter adjustment. (b) Dataset used for training (top) and for validation (bottom). c) Prediction of the model on the validation dataset. d) Validation loss during training. e) Confusion matrix of the validation dataset in absolute value (left) and in percentage (right).

# Multiclass logistic regression

Code implementation and theory were adapted from :

`https://sophiamyang.github.io/DS/optimization/multiclass-logistic/multiclass-logistic.html`

Multiclass logistic regression is a generalization of the logistic regression and can be used to predict more than 2 classes. A selection of features in the training data must first be performed. Each of these feature $x$ must be standardized:

$$\tilde{x} = \frac{x - \overline{x}}{\sigma}, \tag{1}$$

where $\overline{x}$ is the mean and $\sigma$ the standard deviation of $x$. This ensures that each feature has a mean of 0 and a standard deviation of 1. Given a training sample, the model computes a score $\mathbf{z}$ for each class based on a linear combination of the selected features $\mathbf{x}$ with a weight matrix $\mathbf{W}$ and bias vector $\mathbf{b}$:

$$\mathbf{z} = -(\mathbf{W}\mathbf{x} + \mathbf{b}), \tag{2}$$

$\mathbf{W}$ is of dimension $n_{classes}$ by $n_{features}$. The score matrix $\mathbf{Z}$ for all training sample can by computed by using the matrix notation:

$$\mathbf{Z} = -(\mathbf{X}\mathbf{W}^T + \mathbf{b}), \tag{3}$$

where $\mathbf{X}$ is the feature matrix of dimension $n_{samples}$ by $n_{features}$. To make implementation easier, this expression can be rewritten by using the augmented matrix:

$$\mathbf{Z} = -\left(\mathbf{X}|\mathbf{1}\right)\begin{pmatrix}\mathbf{W}\\ \mathbf{1}\end{pmatrix} := -\mathbf{X}\mathbf{W}. \tag{4}$$

The softmax function is then used to convert the score of each samples $j$ into probabilities for each class:

$$\mathbf{p}_j(\mathbf{W}) = softmax(\mathbf{z}_j) = \frac{\exp(\mathbf{z}_j(\mathbf{W}))}{\sum_i^{n_{classes}} \exp(\mathbf{z}_{ji}(\mathbf{W}))}. \tag{5}$$

Since the true class $\mathbf{y}_j$ of each training sample is known, the average likelihood between $\mathbf{P}$ and $\mathbf{Y}$ can be computed for all samples in the training dataset. Here, the negative log-likelihood is used:

$$f(\mathbf{W}) = -\frac{1}{n_{samples}} \log p(\mathbf{Y}|\mathbf{P}(\mathbf{W})) + \mu||\mathbf{W}||^2. \tag{6}$$

A regularization term $\mu$ is used here. Gradient descent can be used to adjust the values of $\mathbf{W}$ that minimize the difference between $\mathbf{P}$ and $\mathbf{Y}$ in an iterative manner:

$$\mathbf{W}_{it+1} = \mathbf{W}_{it} - \eta\nabla f(\mathbf{W}), \tag{7}$$

where $\eta$ is the learning parameter. Finally, memory can be implemented into the system by using a weighted average based on previous probabilities (previous two time steps) and present probability $\mathbf{P}_{out}$

$$\mathbf{p} = \frac{\mathbf{p}_{-1}}{6} + \frac{\mathbf{p}_0}{6} + \frac{\mathbf{p}_{out}}{3}. \tag{8}$$

The final prediction was then given by $argmax(\mathbf{P}_{out})$.