

Titre: Apprentissage profond pour fins de commande d'une prothèse
Title: myoélectrique transhumérale

Auteur: Olivier Barron
Author:

Date: 2021

Type: Mémoire ou thèse / Dissertation or Thesis

Référence: Barron, O. (2021). Apprentissage profond pour fins de commande d'une prothèse
Citation: myoélectrique transhumérale [Thèse de doctorat, Polytechnique Montréal].
PolyPublie. <https://publications.polymtl.ca/6576/>

 **Document en libre accès dans PolyPublie**
Open Access document in PolyPublie

URL de PolyPublie: <https://publications.polymtl.ca/6576/>
PolyPublie URL:

Directeurs de recherche: Maxime Raison, & Sofiane Achiche
Advisors:

Programme: Génie mécanique
Program:

POLYTECHNIQUE MONTRÉAL

Affiliée à l'Université de Montréal

**Apprentissage profond pour fins de commande d'une prothèse myoélectrique
transhumérale**

OLIVIER BARRON

Département de génie mécanique

Thèse présentée en vue de l'obtention du diplôme de *Philosophiae Doctor*

Génie mécanique

Mai 2021

POLYTECHNIQUE MONTRÉAL

Affiliée à l'Université de Montréal

Cette thèse intitulée :

**Apprentissage profond pour fins de commande d'une prothèse myoélectrique
transhumérale**

présentée par **Olivier BARRON**

en vue de l'obtention du diplôme de *Philosophiae Doctor*

a été dûment acceptée par le jury d'examen constitué de :

Nikola STIKOV, président

Maxime RAISON, membre et directeur de recherche

Sofiane ACHICHE, membre et codirecteur de recherche

Abolfazl MOHEBBI, membre

Rozenn DAHYOT, membre externe

DÉDICACE

*À ma famille, Louis, Patricia, Etienne et Christina,
et ma conjointe, Laurence.*

REMERCIEMENTS

Mon doctorat aura été une épreuve d'endurance qui, grâce à l'implication de plusieurs personnes importantes, aura été couronnée de succès. J'offre mes premiers remerciements à mes directeurs de recherche, Prof. Sofiane Achiche et Prof. Maxime Raison, pour leur support continu, leur savoir inestimable et leur aide précieuse à travers chaque nouvelle étape. Leur encadrement m'a non seulement permis de persévérer jusqu'à la ligne d'arrivée, mais aussi de me dépasser dans les objectifs que j'avais établis. Ils m'ont aussi donné l'opportunité d'enseigner; une passion qui continue encore de me captiver.

Un merci tout spécial à mes collègues du laboratoire CoSIM à Polytechnique avec qui j'ai passé la majeure partie de mon temps pour les 5 dernières années. Leur attitude positive et motivante jumelée à leur sens de l'humour hors-pair ont donné une ambiance de travail fantastique qui m'a donné envie de venir chaque jour au A529, soit le local avec une fenêtre qui donne sur une fenêtre. Entre nos parties de cartes au diner et nos échanges fructueux à propos de nos projets, tous et toutes sont maintenant des amis que j'espère fréquenter pour encore longtemps.

Finalement, merci à mes parents, mon frère et ma sœur qui m'ont propulsé dans cette aventure depuis mon entrée à la maternelle. Vous êtes la fondation de mon succès, que ce soit par vos encouragements ou par une bonne tourtière réconfortante. Merci à Laurence Lafortune, ma conjointe, pour ton support inconditionnel à chaque matin. Nous savons tous les deux que si je franchis la ligne d'arrivée, c'est parce que tu as été à mes côtés tout au long de cette épreuve. Merci à mes amis de longues dates, pour votre support, mais aussi pour m'avoir offert des exutoires et les distractions nécessaires pour ne pas devenir fou. Et pour ceux qui me poseront la question: je sais que j'avais dit la même chose à propos du doctorat, mais non, je ne ferai pas de post-doctorat.

RÉSUMÉ

La perte d'autonomie causée par une amputation du membre supérieur peut être restaurée partiellement ou presque entièrement par l'utilisation d'une prothèse myoélectrique adaptée à l'utilisateur. Ce dispositif utilise l'activité électrique des muscles du moignon au moyen de l'électromyographie de surface (sEMG) pour activer les divers mouvements. Il s'agit de la meilleure solution pour offrir le contrôle naturel recherché par les amputés, soit un contrôle qui requiert peu d'effort physique ou mental de la part de son utilisateur. Particulièrement pour les amputés transhuméraux (TH – amputation au-dessus du coude), les prothèses actuellement sur le marché n'offrent pas un tel contrôle alors que leur utilisation est frustrante et mène souvent à l'abandon du dispositif. Dans l'objectif de rendre la commande plus naturelle, la classification des intentions de mouvement au moyen de méthodes d'apprentissage machine (ML) classiques a été investiguée durant les dernières décennies. Malgré des études qui rapportent des performances élevées, le besoin pour un contrôle naturel est toujours présent et les méthodes ML classiques ne semblent pas être capable d'améliorer davantage la précision de classification. Afin d'améliorer les méthodes de classifications de mouvement, un nombre significatif d'études se tournent vers l'apprentissage profond (DL).

Grâce à leur capacité d'abstraction plus élevée que les méthodes ML classiques, les méthodes DL sont potentiellement mieux adaptées à la complexité entourant la classification des intentions de mouvements chez un amputé TH. En effet, les plus récentes applications de DL impliquant des signaux provenant des muscles démontrent des résultats très prometteurs et qui surpassent ceux obtenus avec des méthodes ML classiques. Dans le but de satisfaire le besoin des amputés TH pour une prothèse myoélectrique avec un contrôle naturel, cette thèse introduit une méthode DL capable de reconnaître les intentions de mouvements à partir de signaux sEMG. Les objectifs de 85 % en précision de classification et d'un temps de prédiction en dessous de 250 ms doivent être atteints pour obtenir un contrôle en temps réel qui soit satisfaisant tel que perçu par l'amputé TH.

La méthode DL sélectionnée est le réseau de neurones récurrents à longue mémoire courte (RNN-LSTM) pour sa capacité à utiliser des observations passées afin d'influencer sa décision. Cette capacité lui permet de profiter de l'aspect séquentiel des données et d'être mieux adapté à travailler avec des signaux sEMG. Dans un premier article, une méthode RNN-LSTM est implémentée pour classer six intentions de mouvements à partir de signaux sEMG. Une précision de classification

moyenne de 79.7% est obtenue pour trois amputés TH avec un temps de prédiction de 6.7 ms. La méthode RNN-LSTM est statistiquement plus précise et plus rapide que les deux meilleures méthodes ML classiques de la littérature.

Dans le but d'améliorer la précision de classification tout en maintenant un temps de prédiction faible, un deuxième article aborde l'optimisation de huit hyperparamètres de la méthode RNN-LSTM. La précision de classification moyenne augmente à 93.20 % avec un temps de prédiction de 10.8 ms. À l'issue de cet article, les trois amputés TH atteignent les objectifs de 85 % de précision de classification tout en maintenant un faible temps de prédiction de 10.8 ms.

L'optimisation des hyperparamètres est un processus qui nécessite un temps de calcul, une puissance de traitement et une mémoire qui sont considérables afin de trouver la meilleure configuration possible. Sans expertise préalable, il est possible de gaspiller ces ressources de calcul en investiguant des hyperparamètres qui ont peu d'impact sur la précision de classification. Afin d'aider les futures études impliquant un tel processus, l'article 2 propose aussi une configuration de référence et 12 recommandations pour l'obtention d'une optimisation efficiente des hyperparamètres d'une méthode RNN-LSTM pour la reconnaissance de mouvement avec des signaux sEMG.

ABSTRACT

The loss of autonomy caused by upper limb amputation can be partially or almost entirely restored using a myoelectric prosthesis adapted to its user. This device uses the electric activity of the stump muscles through surface electromyography (sEMG) to activate various movements. This is the best solution to provide the natural control sought by the amputees, i.e. a control that requires little physical or mental effort from its user. Especially for transhumeral amputees (TH – above elbow amputation), the prostheses currently on the market do not offer such control. Their use can be frustrating which often leads to the rejection of the device. To make their control more natural, movement intent classification using classical machine learning (ML) methods have been investigated during the last decades. Despite studies reporting high performance, the need for natural control is still present in the literature and classical ML methods do not seem to be the right solution to further increase classification accuracy. To improve motion classification, a significant number of studies are turning to deep learning (DL).

With a higher abstraction capacity than classical ML methods, DL methods are potentially better suited to the complexity of classifying movement intentions in a TH amputee. Indeed, the most recent DL applications involving muscle signals show very promising results that surpass those obtained with classical ML methods. To satisfy the need of TH amputees for a myoelectric prosthesis with natural control, this thesis introduces a DL method capable of recognizing movement intentions from sEMG signals. The objectives of 85 % classification accuracy and a prediction time below 250 ms must be achieved to obtain a real-time control that is satisfactory as perceived by the TH amputee.

The selected DL method is the Recurrent Neural Network Long Short-Term Memory (RNN-LSTM) for its ability to use past observations to influence its decision. This ability allows it to take advantage of the sequential aspect of the data and is better suited to work with sEMG signals. In a first paper, a RNN-LSTM method is implemented to classify six motion intentions from sEMG signals. An average classification accuracy of 79.7 % is obtained for three TH amputees with a prediction time of 6.7 ms. Additionally, the RNN-LSTM is statistically more accurate and faster than the two best classical ML methods in the literature.

To improve the classification accuracy while maintaining a low prediction time, a second paper discusses the optimization of eight hyperparameters of the RNN-LSTM method. The average

classification accuracy increases to 93.20 % with a prediction time of 10.8 ms. At the end of this article, the three TH amputees achieve the objectives of 85 % classification accuracy while maintaining a low prediction time of 10.8 ms.

Hyperparameter optimization is a process that requires considerable computing time, processing power, and memory to find the best possible configuration. Without prior expertise, it is possible to waste these computational resources by investigating hyperparameters that have little to no impact on classification accuracy. To help future studies involving such a process, article 2 also proposes a reference configuration and 12 guidelines for efficient hyperparameters optimization of a RNN-LSTM method for gesture recognition with sEMG signals.

TABLE DES MATIÈRES

DÉDICACE.....	III
REMERCIEMENTS	IV
RÉSUMÉ.....	V
ABSTRACT	VII
TABLE DES MATIÈRES	IX
LISTE DES TABLEAUX.....	XIII
LISTE DES FIGURES.....	XV
LISTE DES SIGLES ET ABRÉVIATIONS	XVII
LISTE DES ANNEXES.....	XVIII
CHAPITRE 1 : INTRODUCTION.....	1
1.1 Problématique.....	1
1.2 Structure de la thèse	2
CHAPITRE 2 : REVUE DE LITTÉRATURE	4
2.1 Prothèses myoélectriques	4
2.2 Détection de mouvement avec signaux sEMG chez un amputé	6
2.3 Apprentissage machine avec des sEMG	10
2.3.1 Tâche	11
2.3.2 Données utilisées en entrée	14
2.3.3 Algorithme d'apprentissage	15
2.4 Apprentissage profond	21
2.4.1 Algorithmes d'apprentissages	22
2.4.2 Fonctionnement d'un RNN	24

2.4.3	Optimisation des hyperparamètres	26
CHAPITRE 3 : OBJECTIFS		29
CHAPITRE 4 : ARTICLE 1 : RECURRENT NEURAL NETWORK FOR ELECTROMYOGRAPHIC GESTURE RECOGNITION IN TRANSHUMERAL AMPUTEES		31
4.1	Introduction	32
4.2	Methods	35
4.2.1	sEMG data collection	35
4.2.2	sEMG data processing	36
4.2.3	Classifier overview	40
4.2.4	Classifier comparison	41
4.3	Experimental results	42
4.3.1	Difference amongst subjects	42
4.3.2	Impact of the STFT in the input data	43
4.3.3	Classification Accuracy	46
4.3.4	Processing time	48
4.4	Conclusion	49
4.5	Acknowledgment	50
CHAPITRE 5 : ARTICLE 2 : IMPACT OF HYPERPARAMETERS ON THE PERFORMANCE OF A LSTM NETWORK FOR GESTURE RECOGNITION USING SURFACE ELECTROMYOGRAPHY		51
5.1	Introduction	52
5.2	Selected Hyperparameters	54
5.2.1	Architecture	54
5.2.2	Batch Normalization	55

5.2.3	Bidirectional LSTM	55
5.2.4	Dropout.....	56
5.2.5	Learning Rate	57
5.2.6	Self-Attention Layer.....	57
5.2.7	Window Size and Window Increment	58
5.3	Methods.....	59
5.3.1	LSTM-network for sEMG gesture recognition.....	59
5.3.2	Benchmark task.....	61
5.3.3	Impact evaluation	62
5.3.4	Hyperparameter optimal set for the benchmark task	65
5.4	Results	66
5.4.1	Guidelines and recommendation.....	66
5.4.2	10 best configurations	67
5.5	Discussion	68
5.5.1	Main contribution.....	69
5.5.2	Architecture	69
5.5.3	Batch normalization	70
5.5.4	Bidirectional layer	70
5.5.5	Dropout option	71
5.5.6	Learning rate	71
5.5.7	Self-Attention layer	72
5.5.8	Window increment	72
5.5.9	Window size.....	73
5.5.10	Best configuration for the benchmark task	73

5.6	Conclusion.....	74
5.7	Acknowledgement.....	75
CHAPITRE 6 : DISCUSSION GÉNÉRALE		76
6.1	Méthode RNN-LSTM optimisée.....	76
6.2	Contributions de recherche.....	78
6.3	Implémentations logicielles.....	80
6.4	Limitations	81
6.5	Travaux futurs	82
CHAPITRE 7 : CONCLUSION ET RECOMMANDATIONS		84
RÉFÉRENCES.....		86
ANNEXES		110

LISTE DES TABLEAUX

Tableau 2.1 : Caractéristiques des signaux sEMG du moignon.....	9
Tableau 2-2 : Features utilisés dans Hudgins & al. [76].	16
Tableau 2-3 : Features utilisés dans Jarrassé & al. [2].	19
Tableau 2-4 : Features utilisés dans Gaudet & al. [4].	20
Table 4-1 : Mean time (over 50 samples) required by each approach to accomplish the feature calculation and the prediction. Processing time for the RNN is shown with the use of the STFT (w/ STFT) and without the STFT (w/o STFT).	48
Table 5-1 : Search space.....	63
Table 5-2 : Study space	63
Table 5-3 : Impact of each studied hyperparameter with their according guidelines and recommended configuration.....	66
Table 5-4 : Hyperparameters of the baseline (B), the recommended configuration (R) and the 10 best configurations found. (“w/”: with the option. “w/o”: without the option. “Bahd.”: Bahdanau self-attention.)	67
Table 5-5 : Performance of the baseline (B), the recommended configuration (R) and the 10 best configurations found for every subject (S). The best accuracy (Acc.) for each subject is in bold.....	68
Tableau 6-1 : Description de la méthode RNN-LSTM optimisée.	77
Tableau 6-2 : Objectifs spécifiques de recherche couverts dans cette thèse	78
Tableau A.1 : Résultats de la revue de littérature sur les méthodes utilisées pour la classification des intentions de mouvement chez les amputés TH.....	110
Tableau B.2 : Architecture impact. The greyed box indicates the best performance. A bold p-value indicates the difference with the best median is statistically meaningful ($p < 0.01$). In each box, the first line indicates the median, the second line is the difference in median and the last line is the p-value of that difference.	112

Tableau B.3 : Batch Normalization impact. The greyed boxes indicate the best performance. A bold p-value indicates the difference with the best median is statistically meaningful ($p < 0.01$).....	113
Tableau B.4 : Bidirectional layer impact. A bold p-value indicates the difference with the best median is statistically meaningful ($p < 0.01$).	114
Tableau B.5 : Dropout impact. A bold p-value indicates the difference with the best median is statistically meaningful ($p < 0.01$).....	115
Tableau B.6 : Learning rate impact. A bold p-value indicates the difference with the best median is statistically meaningful ($p < 0.01$).	116
Tableau B.7 : Self-Attention layer impact. A bold p-value indicates the difference with the best median is statistically meaningful ($p < 0.01$).	117
Tableau B.8 : Window increment impact. A bold p-value indicates the difference with the best median is statistically meaningful ($p < 0.01$).	118
Tableau B.9 : Window size impact. A bold p-value indicates the difference with the best median is statistically meaningful ($p < 0.01$).....	119

LISTE DES FIGURES

Figure 2-1 : Niveau d'amputation (Adaptée de [24]).....	6
Figure 2-2 : Étapes de résolution d'un problème ML. (D : Données d'entraînement. Algorithme ML : Algorithme d'apprentissage. M : Modèle. x : nouvelle observation. y : nouvelle prédiction.)	10
Figure 2-3 : prothèse biofidèle développée par Ramananarivo et al. [27].	11
Figure 2-4 : Revue des méthodes d'apprentissage machine pour la reconnaissance de mouvement pour les prothèses TH entre janvier 2013 et décembre 2020 (adaptée de [24]). Les études qui concernent uniquement les prothèses TR ou dont les signaux sEMG ont été acquis sur l'avant-bras ont été écartées de cette revue. a) Évolution du nombre de publications pour la période visée. b) Distribution des méthodes d'apprentissage machine utilisées. c) Précision de classification en fonction du nombre de degrés de liberté utilisés dans l'étude (seuls les résultats impliquant un ou des amputés TH ont été retenus – ceux issus d'une expérimentation sur des sujets sains n'ont pas été comptabilisés).	17
Figure 2-5 : Fonctionnement d'un réseau de neurones (x_i : donnée en entrée. θ_i : poids du réseau. ϕ : fonction d'activation. x_i' : sortie de la couche cachée. y : sortie du réseau.)	24
Figure 2-6 : Représentation condensée d'un réseau de neurones (x : données en entrée. h : couche cachée (hidden layer). y : sorties).	25
Figure 2-7 : Représentation d'un RNN (x_i : données en entrée au moment i. h_i : activations des neurones de la couche cachée au moment i. y_i : sortie au moment i.)	25
Figure 4-1 : Input data structure.	38
Figure 4-2 : Example of training and validation split based on the repetition. Rep.: Repetition; NM: No movement; EF: Elbow Flexion; EE: Elbow extension; FP: Forearm pronation; FS: Forearm supination; WF: Wrist flexion; WE: Wrist extension; HO: Hand open; HC: Hand close.	39
Figure 4-3 : Filtered sEMG signal for every subject (S) during an Elbow Flexion (EF). The gray areas represent the sEMG activation automatically detected using TKEO on the offline data (here, the two EFs).	42

Figure 4-4 : Sensitivity study between the time window size for extracting the STFT and the Cross-Validation Accuracy obtained with the RNN.	44
Figure 4-5 : STFT Impact. Cross-Validation Accuracy based on whether STFTs were concatenated to the input data or not for the RNN.....	45
Figure 4-6 : Classification performance. The Cross-Validation Accuracy of each classification approach is reported for the 5 subjects, as well as the mean of all the subjects and the mean of the subject 1,3,5. (RNN without the use of the STFT.).....	46
Figure 5-1 : LSTM network full build. (“S”: sample size. “ws”: window size. “ch”: number of channels. “N”: Number of LSTM units. “d”: dropout rate. “L”: number of layers.).....	60
Figure B.7-1 : Batch normalization impact. The dashed line corresponds to the median of the same colored distribution.	113
Figure B.7-2 : Bidirectional layer impact. The dashed line corresponds to the median of the same colored distribution.	114
Figure B7-3 : Dropout impact. The dashed line corresponds to the median of the same colored distribution.	115
Figure B.7-4 : Learning rate (lr) impact. The dashed line corresponds to the median of the same colored distribution.	116
Figure B.7-5 : Self-Attention layer impact. The dashed line corresponds to the median of the same colored distribution.	117
Figure B.7-6 : Window increment impact. The dashed line corresponds to the median of the same colored distribution.	118
Figure B.7-7 : Window size impact. The dashed line corresponds to the median of the same colored distribution.	119

LISTE DES SIGLES ET ABRÉVIATIONS

Abréviation	Définition (Français)	Définition (Anglais)
ANN	Réseau de neurones artificiel	Artificial neural network
CNN	Réseau de neurones convolutifs	Convolutional Neural Network
DL	Apprentissage profond	Deep learning
EE	Extension du coude	Elbow extension
EEG	Électroencéphalographie	Electroencephalography
EF	Flexion du coude	Elbow flexion
FMG	Forcemyographie	Forcemyography
FP	Pronation de l'avant-bras	Forearm pronation
FS	Supination de l'avant-bras	Forearm supination
HC	Main fermée	Hand close
HD-sEMG	Électromyographie surfacique en haute définition	High-definition surface electromyography
HO	Main ouverte	Hand open
KNN	K plus proches voisins	K-nearest neighbors
LDA	Analyse discriminante linéaire	Linear discriminant analysis
ML	Apprentissage machine	Machine learning
NIR	Proche infra-rouge	Near infra-red
NM	Aucun mouvement	No movement
PLM	Mouvement du membre fantôme	Phantom limb movement
QDA	Analyse discriminante quadratique	Quadratic discriminant analysis
RNN	Réseau de neurones récurrents	Recurrent Neural Network
RNN-LSTM	Réseau de neurones récurrents à longue mémoire courte.	Recurrent neural network – long short-term memory.
sEMG	Électromyographie de surface	Surface electromyography
SVM	Machine à vecteurs de support	Support vector machine

LISTE DES ANNEXES

ANNEXE A – TABLEAU RÉSUMÉ DE LA REVUE DE LITTÉRATURE	110
Annexe B – Article 2 : Individual Hyperparameter Impact study	112

CHAPITRE 1 : INTRODUCTION

1.1 Problématique

L'amputation d'un membre réduit la capacité du patient à effectuer des tâches de la vie quotidienne. Il est possible de remédier à cette perte d'autonomie en redonnant certaines fonctionnalités du membre amputé au moyen d'une prothèse myoélectrique. Ce dispositif exécute divers mouvements en fonction de l'activité électrique musculaire de son usager. Il a été démontré qu'une grande majorité des amputés sont capables de volontairement générer des patrons d'activation musculaire qui diffèrent selon le mouvement qu'il souhaite effectuer [1]–[4]. De plus, pour des actionnements et des mouvements reproductibles du membre supérieur, ces patrons d'activations sont aussi reproductibles [5], [6]. Ils constituent ainsi un moyen fiable et intuitif de générer des mouvements avec une prothèse [1], [3]. Il s'agit de la meilleure approche [7] pour offrir un contrôle naturel recherché par ces amputés [8], soit un contrôle qui demande peu d'efforts physiques ou mentaux. Les prothèses existantes n'offrent pas ce contrôle naturel, ce qui explique le faible taux d'adhésion des amputés à ces solutions [8]–[10].

Le développement entourant le contrôle des prothèses myoélectriques du membre supérieur est en plein essor, surtout avec l'avènement de l'apprentissage machine. Cependant, il existe une disparité entre les études pour les amputés transradiaux (TR – en dessous du coude) et pour les transhuméraux (TH – au-dessus du coude). Bien que leur population soit semblable [11], [12], la littérature pour les amputés TR est plus développée, les solutions proposées sont plus abondantes [13] et des bases de données publiques sont disponibles pour encourager la coopération entre les équipes de recherche [14]–[16]. Ce n'est pas le cas pour les amputés TH pour qui les solutions disponibles sont plus limitées. Le développement de leurs prothèses est plus ardu à cause du nombre plus élevé de mouvements à rétablir et de l'accès plus limité aux muscles responsables de ces mouvements, entre autres les muscles de l'avant-bras responsables des mouvements de la main ne sont plus disponibles.

Les prothèses pour amputés TH utilisent actuellement des algorithmes d'apprentissage machine (Machine Learning – ML) classique tels que l'analyse discriminante linéaire (Linear Discriminant Analysis - LDA), le réseau de neurones (Artificial Neural Network – ANN) ou la machine à vecteurs de support (Support Vector Machine – SVM). Grâce à leur combinaison de

caractéristiques extraites de l'activité musculaire, deux études utilisant respectivement un LDA [3] et un ANN [4] ont démontré l'efficacité de leur méthode sur 5 amputés TH et constituent les meilleures solutions de la littérature. Toutefois, malgré ces performances élevées et des décennies de recherche [7], [17]–[20], la plupart des méthodes ML classiques atteignent des performances similaires [21] et le besoin pour une méthode de contrôle intuitive est toujours présent dans la littérature [19]. Afin d'améliorer les méthodes de classifications de mouvement, un nombre significatif d'études se tournent vers l'apprentissage profond (DL) [22].

Les études entourant les prothèses TR utilisent déjà du DL afin d'améliorer leur contrôle, tandis que cette piste n'a pas encore été explorée pour les amputés TH. Pourtant, le niveau d'abstraction plus élevé des méthodes de DL est potentiellement mieux adapté à la complexité de leur problème et pourrait être l'élément clé afin d'améliorer la reconnaissance de mouvement pour cette population d'amputés. Les plus récentes applications de DL impliquant des signaux provenant des muscles démontrent des résultats très prometteurs qui surpassent ceux obtenus avec des méthodes ML classiques [23].

L'objectif principal de cette thèse est de développer une méthode d'apprentissage profond qui reconnaît les intentions de mouvement d'un amputé TH pour la commande d'une prothèse myoélectrique. Une méthode de DL est proposée, optimisée, testée puis comparée aux deux meilleures méthodes ML classique. De plus, des lignes directrices sur l'optimisation efficiente de cette méthode sont proposées.

1.2 Structure de la thèse

Le chapitre 1 présente la problématique visée par cette thèse et détaille la structure du document.

La revue de littérature du chapitre 2 développe la problématique et explore le domaine des prothèses myoélectriques plus en détail. Le fonctionnement des prothèses myoélectrique est expliqué avec un accent sur la reconnaissance d'intentions de mouvements chez son usager. Ensuite, le ML est introduit comme la méthode pour accomplir cette reconnaissance et les meilleures méthodes ML classiques de la littérature sont décrites. La méthode ML à résoudre est alors reformulée sous forme d'une tâche, de données en entrée et d'un algorithme d'apprentissage. Le DL est introduit comme une alternative aux méthodes ML classiques et les principales méthodes sont discutées en lien avec la problématique ML. Finalement, le fonctionnement de la méthode DL

choisie est décrit avant d'expliquer comment optimiser ses hyperparamètres en vue d'obtenir la meilleure performance possible.

Une fois que les solutions actuelles et potentielles sont exposées, le chapitre 3 rappelle l'objectif principal et décrit les trois objectifs spécifiques qui en découle.

Le chapitre 4 décrit d'abord l'implémentation de la méthode DL, et compare ensuite la précision de classification obtenue sur 5 amputés TH et son temps de prédiction aux deux meilleures méthodes de la littérature qui utilise de l'apprentissage machine classique.

Le chapitre 5 se concentre sur l'optimisation des hyperparamètres afin de maximiser la précision de classification pour la reconnaissance de mouvement. L'étude de l'impact de huit hyperparamètres sur la performance mène à des recommandations sur la manière d'optimiser la méthode DL de façon efficiente pour cette tâche. Un ensemble d'hyperparamètres est aussi identifié pour maximiser la performance de la méthode DL sur trois amputés TH.

Le chapitre 6 discute de la contribution de ce projet de recherche et de comment l'objectif principal, de même que les trois sous-objectifs, ont été rencontrés à travers les articles scientifiques publiés (ou soumis pour publication).

Finalement, le chapitre 7 fait un résumé de la thèse et porte un regard critique sur ses résultats afin d'en identifier les limitations. De plus, des perspectives de recherche sont proposées.

CHAPITRE 2 : REVUE DE LITTÉRATURE

Le développement d'une nouvelle méthode pour la commande d'une prothèse myoélectrique transhumérale implique la combinaison de plusieurs disciplines comme la physiologie humaine, le traitement de signal et l'apprentissage machine. Afin de comprendre comment il est possible d'intégrer de l'intelligence artificielle à une prothèse, le présent chapitre traite d'abord de la prothèse myoélectrique transhumérale en élaborant sur sa définition et en expliquant son fonctionnement au moyen de la reconnaissance des signaux sEMG. Ensuite, une revue de littérature présente l'étendu des applications intégrant des algorithmes d'apprentissage machines classiques à la détection de mouvement chez les amputés du membre supérieur. Finalement, l'apprentissage profond est présenté comme une alternative aux algorithmes classiques pour répondre au besoin toujours présent pour une prothèse avec une commande naturelle.

Une revue de littérature sur le développement des prothèses myoélectriques transhumérales a déjà été rédigée par l'auteur de cette thèse (O. Barron, M. Raison, et S. Achiche. Control of transhumeral prostheses based on electromyography pattern recognition: from amputees to deep learning [24]). Ainsi, le Chapitre 2 : Revue de littérature s'inspire de cette précédente publication.

2.1 Prothèses myoélectriques

La perte d'un membre supérieur entraîne une perte d'autonomie importante pour les amputés. Une des solutions pour pallier cette situation est l'utilisation d'une prothèse. Elle peut être passive, voire esthétique, n'ayant comme objectif que de ressembler le plus possible à bras humain. Autrement, elle peut être active en accomplissant certaines fonctions qui viennent aider l'amputé dans sa vie quotidienne. La conception d'une prothèse active qui conviendrait à la majorité des usagers est ardue puisque chacun possède une amputation différente. L'hétérogénéité et le faible nombre d'amputés, surtout TH, expliquent entre autres pourquoi les chercheurs ont encore du mal à concevoir un nouveau dispositif rapidement qui rejoint les besoins des amputés.

D'un point de vue mécanique, les prothèses actives existantes sont bien développées, mais leur contrôle est souvent rudimentaire et rend leur utilisation difficile [25], [26]. Afin d'augmenter le taux d'adhésion à cette solution, le contrôle d'une prothèse active se doit d'être naturel [8], [26], [27], soit un contrôle qui demande peu d'efforts physiques ou mentaux de la part de son usager. La

prothèse myoélectrique est la solution actuellement sous investigation pour répondre à ce besoin précis [7].

La prothèse myoélectrique utilise l'activité électrique musculaire de son usager pour exécuter des mouvements spécifiques [27]. L'électromyographie de surface (Surface Electromyography - sEMG) permet l'acquisition de cette activité à l'aide de sondes posées sur la peau au-dessus des muscles [5]. Dès que l'amputé souhaite effectuer un mouvement, son intention se transforme en impulsion nerveuse qui voyage naturellement de son cerveau aux muscles de son moignon. En s'activant, ces derniers produisent des signaux électriques qui sont perçus par les sondes. La prothèse peut alors utiliser ce signal pour effectuer le mouvement correspondant. C'est cette utilisation de signaux sEMG générés naturellement qui permet aux prothèses myoélectriques d'offrir un contrôle plus intuitif [7], [20], [21].

Pour le membre supérieur, il existe plusieurs niveaux d'amputations dépendamment d'où se situe le sectionnement sur le bras du patient comme on peut l'observer à la Figure 2-1. Notamment, les patients transradiaux (TR) et transhuméraux (TH) constituent les deux groupes d'amputation les plus communs [28] et ont une amputation en dessous et au-dessus du coude respectivement. L'amputation TH, qui fait l'objet d'étude pour cette thèse, est effectuée dans la majorité des cas à la suite d'un trauma et affecte souvent une population plus adulte [29]. La proportion d'amputation TR par rapport au nombre total d'amputation du membre supérieur est similaire à celle des amputés TH [11], [12]. Cependant, la littérature concernant la conception de prothèses TR est plus développée et utilisent des technologies plus récentes pour améliorer le contrôle [20]. Il existe des bases de données publiques pour encourager la coopération entre les équipes de recherche et pour mieux comparer les méthodes entre elles [30]. C'est pourquoi on peut trouver de nombreux produits sur le marché qui couvrent une variété de besoins pour ce niveau d'amputation [13]. Ce n'est pas le cas pour les amputés TH pour qui les avancées technologiques sont plus espacées temporellement et les produits disponibles sont mal adaptés [24]. Cette population d'amputé fait rarement partie des études qui explorent de nouvelles solutions; la majorité de ces études utilisent des sujets sains ou des amputés TR [7], [23], [24].

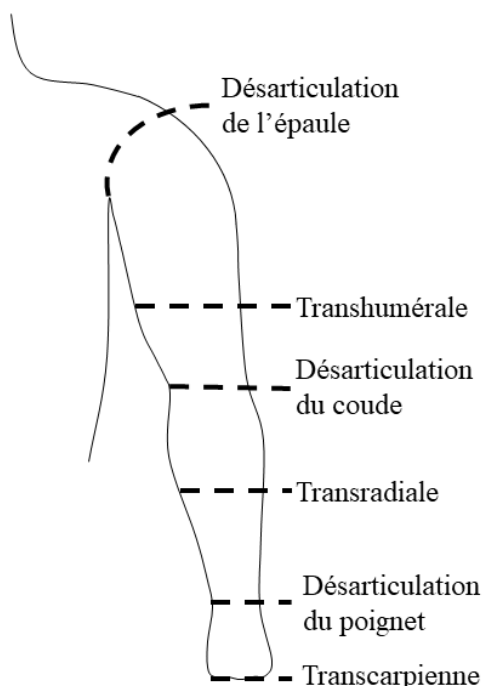


Figure 2-1 : Niveau d'amputation (Adaptée de [24])

Bien qu'à la base, la problématique soit la même pour les deux populations, le contexte et les ressources disponibles sont différents. Chez un amputé TR, les muscles résiduels de l'avant-bras peuvent être utilisés pour rétablir les mouvements de l'avant-bras, du poignet et de la main. Étant donné que ces muscles étaient en majorité responsables de ces mouvements, les signaux mesurés au moignon ont une correspondance plus directe avec les mouvements que l'amputé souhaite effectuer. Chez un amputé TH, seuls les muscles résiduels du bras sont disponibles pour rétablir les mêmes mouvements, ainsi que ceux du coude [20]. Ces muscles n'étant pas naturellement impliqués dans les mouvements du poignet et de la main, le lien entre ces mouvements et les signaux mesurés est plus ambigu. Cette complexité et le nombre plus élevé de mouvements à rétablir expliquent pourquoi le développement de prothèses myoélectriques est plus ardu pour les amputés TH qu'il ne l'est pour les amputés TR.

2.2 Détection de mouvement avec signaux sEMG chez un amputé

La sEMG est la modalité d'acquisition la plus populaire pour identifier les mouvements qu'un amputé souhaite effectuer [20], [31]. Depuis 1980, la demande grandissante pour des capteurs corporelles pour le suivi de la santé ou la robotique médicale a donné lieu à l'investigation de

plusieurs autres méthodes d'acquisition qui peuvent compléter ou remplacer la sEMG [32]–[35]. La forcemyographie (Forcemyography – FMG) utilise des jauges de contrainte ou des capteurs de pression à la surface de la peau pour mesurer la déformation du muscle [36], [37]. À l'opposé des signaux sEMG qui demandent une fréquence d'échantillonnage dans les kiloHertz, la FMG peut s'effectuer en dessous de 100 Hz et nécessite ainsi moins de ressources informatiques. Cette méthode non invasive suscite de plus en plus d'intérêt dans le domaine des prothèses, notamment pour les amputés TR, et est considérée comme une alternative aux signaux sEMG [38]–[40]. Un premier brassard avec des jauges de contraintes a été conçu en 2017 et utilisé pour identifier des mouvements chez un amputé TH [31]. En plus de la FMG, les ultrasons [41] et les spectres dans l'infrarouge proche (Near-Infrared – NIR) [12] sont considérés comme une autre alternative, ainsi que l'électroencéphalographie (Electroencephalography – EEG) qui peut être combinée avec des signaux sEMG [32] ou utilisée seule pour commander un degré de liberté [42]. Cette thèse s'intéresse à la reconnaissance des intentions de mouvements au moyen de la modalité la plus utilisée, soit les signaux sEMG [7], [20].

Initialement, les prothèses myoélectriques n'utilisaient que l'amplitude du sEMG [20], [21]. L'algorithme de commande activait le dispositif seulement si l'amplitude mesurée dépassait un seuil préétabli [43] et ne contrôlait qu'un seul degré de liberté. Pour augmenter le nombre de fonctions possibles, des commandes à deux et à trois états ont été développées en intégrant d'autres seuils à l'algorithme de décision. Quant à la vitesse de mouvement, elle était proportionnelle à l'amplitude moyenne du signal [44] ou constante tout au long du mouvement [45]. Cette stratégie de contrôle ne convient pas aux amputés et est encore largement utilisée dans l'industrie [9].

Les groupes de recherche ont rapidement réalisé que les signaux sEMG contiennent plus d'informations qu'une simple amplitude. Notamment, il est possible d'estimer les efforts musculaires durant un mouvement au moyen d'une fonction coût [46] ou en résolvant mathématiquement le problème de co-activation des muscles de chaque articulation [47]. Des travaux ont utilisé plusieurs sondes sur le moignon pour capter une plus grande partie de l'activité myoélectrique du bras [7]. La disposition de ces électrodes a longtemps été débattue dans la littérature, à savoir entre placer les électrodes au-dessus des muscles en utilisant des repères anatomiques ou placer les électrodes à équidistances comme une grille. Des recherches ont démontré que lorsque six sondes ou plus sont utilisées, la disposition de ces dernières n'a pas d'impact significatif sur l'algorithme de décision [7]. Aussi, des équipes de recherche investiguent

l'utilisation de sEMG à haute densité (High Definition Surface Electromyography – HD-sEMG) afin de capturer encore plus de détails de l'activité musculaire [21]. Cette technique utilise une matrice composée de nombreuses électrodes rapprochées qui captent les signaux sEMG d'une zone définie [48]. D'un côté, cette technique offre plus d'informations à l'algorithme de décision en passant de signaux 1D à une image 2D de l'activité myoélectrique [49]. De l'autre, cette technique exige plus de ressources informatiques pour traiter l'information et l'équipement d'acquisition s'intègre mal à une prothèse.

Pour comprendre comment les amputés sont capables de produire des signaux myoélectriques utiles pour le contrôle d'une prothèse, il faut savoir comment le corps humain fait face à la perte d'un membre. Après une amputation, le corps de l'amputé tente de s'adapter aux tissus sectionnés et aux nerfs endommagés. Il n'est pas rare qu'après ce réarrangement neuromusculaire, l'amputé ait l'impression que son membre est toujours présent après l'opération. Ce phénomène est connu sous le nom de membre fantôme et peut durer toute une vie, quoiqu'il a tendance à s'estomper avec le temps [50].

Une étude de 1998 [51] révèle que les amputés peuvent ressentir une variété de sensations à partir de leur membre fantôme. Ces sensations incluent des spasmes musculaires, des changements de températures ou même de la douleur. De plus, une majorité des amputés sont capables d'effectuer des mouvements volontaires de leur membre fantôme (Phantom Limb Movement – PLM) à répétition [52]. Les amputés rapportent que la sensation du mouvement et la façon dont le membre fantôme bouge sont différentes d'avant leur amputation. Le PLM produit aussi un signal sEMG qui est différent avec le signal normalement observé chez un sujet sain [53] d'où l'importance d'effectuer des essais avec des sujets amputés. Chaque amputé possède des patrons d'activation uniques qui dépendent de comment leur plaie a été refermée et de leur réorganisation neuromusculaire. Ces patrons sEMG sont spécifiques au PLM effectué par l'amputé, c.-à-d. le patron sEMG sera sensiblement le même pour chaque répétition du même PLM [5], mais sera différent pour les autres mouvements. De plus, ces patrons sEMG sont reproductibles d'une session d'acquisition à l'autre [5]. Ces caractéristiques du signal sEMG provenant du moignon ont été résumées dans le Tableau 2.1.

Tableau 2.1 : Caractéristiques des signaux sEMG du moignon.

Signaux sEMG du moignon	Patrons différents de ceux observés chez un sujet sain.
	Patrons spécifiques à chaque amputé.
	Un patron spécifique pour chaque PLM.
	Patrons reproductibles intra-session et intersession.

Il existe des méthodes invasives qui tentent de capter l'impulsion nerveuse du mouvement à son origine directement aux nerfs périphériques [54] ou au cerveau [55]. La technique qui a révolutionné le domaine du contrôle des prothèses est la réinnervation musculaire ciblée (Targeted Muscle Reinnervation – TMR) développée par Kuiken & al. [56], [57]. Au cours de cette opération, le chirurgien attache les nerfs sectionnés du bras à des muscles sains du patient comme les muscles dorsaux ou pectoraux. Chaque nerf est fixé à un groupe musculaire distinct de sorte que sa réponse puisse être mesurée plus clairement à un endroit précis du corps humain. Comme démontré par la prothèse développée à John Hopkins [58], la TMR permet un contrôle de la prothèse très fiable et précis qui surpasse certaines méthodes de contrôle non invasives [59]. Cependant, il existe des risques à l'opération qu'il faut prendre en considération. La TMR peut entraîner la paralysie du muscle ciblé, le développement de neurinome douloureux et des douleurs récurrentes provenant du membre fantôme [60]. Des études récentes ont démontrées qu'il n'était pas nécessaire d'avoir recours à la TMR pour obtenir un contrôle adéquat de la prothèse myoélectrique [2]–[4].

Les caractéristiques énoncées dans le Tableau 2.1 sont la base du fonctionnement derrière les prothèses myoélectriques [61]. Ces patrons étant spécifiques et reproductibles [2], [4], [5], il est possible d'acquérir les signaux sEMG d'un amputé et de les associer au mouvement souhaité. La prothèse peut alors être activée selon l'intention de mouvement de son usager. Toutefois, les signaux sEMG sont stochastiques et composés d'éléments bioélectriques non stationnaires et non linéaires qui rendent impossible de reconnaître les patrons sEMG à l'œil nu [19]. L'apprentissage machine est la solution utilisée pour analyser ces signaux et obtenir une prédiction quant au mouvement que l'amputé souhaite effectuer [20], [21].

2.3 Apprentissage machine avec des sEMG

La classification est une branche de l'intelligence artificielle, plus précisément de ML, qui associe un ensemble de données à des classes prédéfinies en fonction des formes présentes dans le signal. En utilisant des notions de probabilité, de la théorie de l'information et de calcul numérique, l'algorithme ML construit un modèle mathématique qui accomplit cette tâche à partir d'un ordinateur ou d'un microcontrôleur [62]. Par exemple, à partir des signaux sEMG et données cinématiques du mouvement d'un patient, il est possible d'utiliser une méthode nommée K-moyennes pour construire un modèle mathématique capable d'attribuer le bon index d'handicap d'un enfant atteint d'une infirmité motrice spastique unilatérale [63].

La résolution d'un problème d'apprentissage machine se fait en deux étapes. Comme sur la Figure 2-2, l'algorithme de ML s'entraîne d'abord sur une base de données D afin de créer le modèle mathématique M. Ensuite, le modèle mathématique peut être utilisé pour résoudre la tâche en prenant de nouvelles observations x et en émettant des prédictions y. À noter que les observations x doivent avoir le même format que les données contenues dans D pour le modèle M fonctionne. L'étape d'entraînement demande plus de temps et de ressources informatiques que d'utiliser le modèle mathématique fixe (souvent constitué d'une simple matrice de poids). Il est possible d'améliorer le modèle mathématique en refaisant son entraînement au moyen d'une base de données plus à jour.

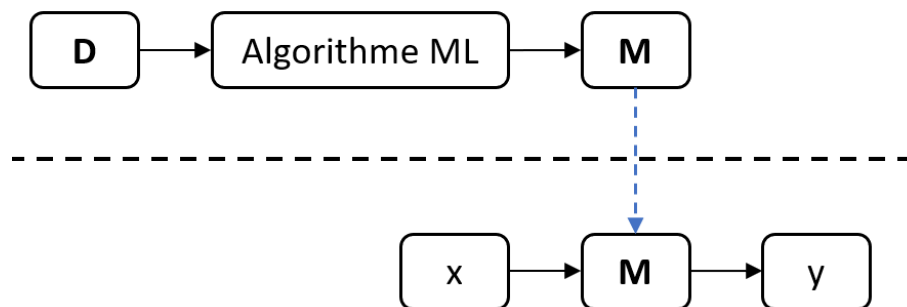


Figure 2-2 : Étapes de résolution d'un problème ML. (D : Données d'entraînement. Algorithme ML : Algorithme d'apprentissage. M : Modèle. x : nouvelle observation. y : nouvelle prédiction.)

Afin de développer une solution ML, trois éléments sont définis : la tâche à résoudre, les données utilisées en entrée et la méthode d'apprentissage. Une fois entraîné, le modèle mathématique sera spécifique à ces trois éléments. Si un de ces derniers doit changer, le processus d'entraînement du modèle doit être refait.

2.3.1 Tâche

Pour le contrôle d'une prothèse myoélectrique, la tâche générale à résoudre est la suivante :

Tâche	Classifier les intentions de mouvements de son utilisateur.
--------------	---

Les degrés de liberté d'une prothèse limitent quels mouvements peuvent être effectués. Il est donc attendu que les intentions de mouvements de l'amputé fassent partie d'une classe prédéfinie de mouvement selon la prothèse utilisée. Précisément pour cette thèse, une prothèse développée conjointement au laboratoire CoSIM et à la chaire de recherche RECAP est utilisée comme étude de cas [27]. Cette prothèse biofidèle (Figure 3) reproduit le profil et les mouvements d'un bras humain pour offrir une utilisation plus naturelle aux amputés [27]. Elle a été conçue au moyen de mesures cinématiques du bras humain [64], [65] et d'un modèle raffiné du membre supérieur capable de recréer le mouvement ostéoarticulaire [66]. Un premier moteur en rotation au coude exécute le mouvement de flexion ou extension du coude. Un deuxième moteur active le mouvement de l'avant-bras au moyen d'un système d'engrenages et de tiges conçus pour répliquer le mouvement de pronation et supination de l'avant-bras le plus fidèlement possible [27]. Finalement, un moteur active le mouvement de fermeture et d'ouverture de la main au moyen de câbles et d'élastiques. Ce mécanisme sous-actionné permet notamment un contrôle plus facile et qui s'adapte facilement à plusieurs objets que l'on souhaiterait prendre dans une main.



Figure 2-3 : prothèse biofidèle développée par Ramananarivo et al. [27].

La prothèse possède 3 degrés de liberté lui permettant d'effectuer les 6 mouvements suivants : la flexion (Elbow Flexion – EF) pour rapprocher l'avant-bras du bras; l'extension du coude (Elbow Extension – EE) pour éloigner l'avant-bras du bras; la pronation de l'avant-bras (Forearm Pronation – FP) pour tourner la paume de la main vers le bas; la supination de l'avant-bras (Forearm Supination – FS) pour tourner la paume de la main vers le haut; l'ouverture de la main (Hand Open – HO) et la fermeture de la main (Hand Close – HC). La classe « aucun mouvement » (No Movement - NM) est ajoutée pour immobiliser la prothèse lorsqu'aucun mouvement n'est souhaité.

Pour mesurer le taux de succès du modèle à résoudre la tâche, une métrique de performance doit être sélectionnée. Pour un problème multi-classes, la précision de classification (classification accuracy) est la métrique la plus utilisée [67], particulièrement pour la classification d'intentions de mouvement [7], [23], [24]. Elle se définit comme suit :

$$\text{précision de classification} = \frac{\text{Nombre de bonnes prédictions}}{\text{Nombre de prédictions total}}$$

Un défaut de la précision de classification est qu'elle donne une représentation biaisée du taux de réussite du modèle lorsque les classes ne sont pas balancées (c.-à-d. le nombre d'exemples pour chaque classe n'est pas égale dans les données tests) [67]. Le taux de succès de la classe surreprésentée aura une plus grande pondération sur la précision de classification rendant difficile de conclure sur la réussite du modèle à reconnaître tous les mouvements impliqués. Une première solution pour remédier à ce problème est de balancer le nombre d'occurrences de chaque classe en éliminant des exemples des classes surreprésentées. Toutefois, lorsque la base de données d'entraînement est déjà petite, il n'est pas souhaitable de la réduire encore plus. La deuxième solution est alors de calculer une précision de classification balancée (balanced accuracy), c.-à-d. de calculer la précision de classification pour chacune des classes individuellement et d'en faire la moyenne. De cette façon, chaque classe a la même pondération dans le calcul de la métrique de performance. Les précisions de classifications discutées dans cette thèse peuvent toutes être considérées comme balancées.

Dans le domaine de la classification d'intentions de mouvements, la spécificité des études rend leur comparaison difficile. La provenance des données d'entraînement et la façon dont la précision de classification est calculée ne sont pas toujours claires et peuvent varier d'un article à l'autre. C'est pourquoi il est suggéré de créer des bases de données qui seraient publiques et qui serviraient à

évaluer les différentes méthodes de classifications [23]. Des bases de données sont actuellement utilisées pour les amputés TR [14], mais aucune n'existe pour la population TH.

Un autre problème lors de la comparaison des études est le type de population utilisé pour tester les modèles entraînés. Une majeure partie des études sur les prothèses myoélectriques sont menées sur des sujets sains [7], [23] et très peu sur des amputés TH [4]. Des revues de littérature rapportent des précisions de 90 à 95% pour plusieurs études, des taux de succès qui justifieraient des essais cliniques pour ces systèmes de contrôle. Toutefois, la plupart n'utilisent aucun amputé pour valider leur méthode. Pour celles utilisant principalement des sujets amputés, la précision de classification diminue entre 80 à 90 %, voire moins. Les sujets sains semblent être utilisés dans ces études pour les conditions de test plus favorables au développement de ces dispositifs et pour pallier la difficulté de recruter des sujets amputés [7].

Pour le développement d'une méthode ML, plusieurs itérations de tests doivent avoir lieu pour obtenir un modèle performant. Celles-ci sont effectuées sur des bases de données puisqu'il serait difficile d'avoir accès à des participants amputés à l'occasion de chaque test. Puisqu'il n'est pas possible de valider en différé la fonctionnalité perçue par l'utilisateur de la prothèse, il faut effectuer le développement ML et l'évaluation de cette fonctionnalité en deux temps. Certaines études notent qu'une précision de classification en dessous de 85% conduit à une utilisation frustrante de la prothèse et qu'un tel seuil devrait être atteint avant de poursuivre à des tests en temps réel [3], [7]. De plus, la latence maximale pour obtenir la prédiction du mouvement à effectuer devrait être comprise entre 100 et 250 ms [68], [69].

En prenant compte ces derniers détails, il est possible de préciser la tâche à résoudre comme suit :

Tâche

Classifier les intentions de mouvement d'un amputé transhuméral en respectant les objectifs suivants :

- La précision de classification balancée devrait être maximisée et devrait atteindre un minimum de 85 %.
 - Le temps nécessaire pour obtenir une prédiction devrait être en-dessous de 250 ms.
 - Les 7 classes à l'étude sont : flexion (EF) et extension (EE) du coude, pronation (FP) et supination (FS) de l'avant-bras, ouverture (HO) et fermeture (HC) de la main.
-

2.3.2 Données utilisées en entrée

Quant aux données utilisées comme entrée pour le modèle, il s'agit de signaux sEMG récoltés à la surface du moignon de l'amputé. Dans chaque muscle, des unités motrices sont responsables d'activer un ou plusieurs fibres musculaires. À la réception de l'influx nerveux provenant du cerveau, ces unités motrices génèrent un potentiel d'action qui provoque un échange électro-chimique entre les fibres musculaires, ce qui permet alors la contraction des fibres musculaires. L'électromyographie de surface utilise des électrodes apposées sur la surface de la peau pour mesurer le potentiel électrique créé par les unités motrices sous-jacentes [71]. Deux électrodes bipolaires non-invasives sont apposées directement au-dessus du muscle pour mesurer la différence de potentiel lors de la contraction [70]. Les signaux sEMG sont très utiles pour comprendre comment le système neuromusculaire fonctionne et sont notamment populaires dans les domaines du sport et de la réadaptation [70].

L'amplitude du signal se situe en dessous de 5 mV et oscille autour de l'axe des zéros [71]. La fréquence du signal se situe autour de 500 Hz et nécessite donc minimalement une fréquence d'acquisition au-dessus de 1 kHz pour respecter le théorème de Shannon [31]. Le traitement du signal implique normalement un filtre passe-bande avec des fréquences de coupures autour de 20 Hz et de 500 Hz [71]. Il est possible d'utiliser plusieurs canaux d'acquisitions simultanément pour capturer l'activité électrique musculaire de plusieurs sites différents. De plus, une étude a démontré que les signaux sEMG recueillis pour un même mouvement peuvent présenter de 20 à 30% d'erreurs entre les acquisitions [5]. Toutefois, la capacité d'une méthode ML à classifier les mouvements semble indépendante à ce manque de reproductibilité puisque plusieurs études parviennent à obtenir des précisions de classifications au-delà de 90% [3], [4], [72]–[74]. L'algorithme d'apprentissage s'adapte au fait que les signaux sEMG varient et parvient tout de même à accomplir sa tâche.

Il peut être difficile de récolter une grande quantité de signaux sEMG puisque la fatigue musculaire qui s'accumule durant l'acquisition altère le signal [5]. Le manque de données est encore plus présent avec des sujets amputés qui sont difficiles à recruter à la base et qui peuvent éprouver des difficultés avec leur membre fantôme (p. ex. baisse de l'amplitude de mouvement) durant l'acquisition [5].

Il est important de noter que les valeurs instantanées d'un signal sEMG sont généralement inutiles pour l'algorithme d'apprentissage en raison de leur nature aléatoire. Il est préférable d'utiliser des fenêtres de temps, soit l'accumulation de données sEMG sur un laps de temps, pour prendre avantage de l'information contenue dans le signal [21]. Il est possible de fournir des fenêtres de temps successivement ou de les chevaucher pour obtenir plus d'exemples. Une taille de fenêtre entre 150 et 250 ms avec un incrément entre 50 et 80 ms est optimale pour avoir des délais de prédiction acceptable en échange d'une meilleure précision de classification [20], [75].

Les signaux sEMG utilisés dans le cadre de ce projet de recherche proviennent de 5 sujets amputés ayant participé à une étude faite conjointement au laboratoire CoSIM et à la chaire de recherche RECAP [4]. Le protocole d'acquisition et les instruments des mesures utilisés sont décrits dans la section 4.2.1.

**Données
utilisées en
entrée**

Plusieurs canaux de signaux sEMG divisés en fenêtre de temps et provenant de sondes apposées sur le moignon de l'amputé transhuméral.

2.3.3 Algorithme d'apprentissage

Il reste donc l'algorithme d'apprentissage à définir. Ces méthodes sont divisées en deux groupes : les méthodes de DL et les méthodes ML classique, soit les méthodes de ML utilisées avant l'avènement du DL. Pour définir une méthode de ML classique, il faut rassembler deux éléments soit l'ensemble de caractéristiques utilisés et l'algorithme d'apprentissage.

Les algorithmes de ML classique n'ont pas la capacité d'analyser un signal sEMG complet pour émettre leur prédiction. Un ensemble de valeur plus petit doit être fourni [21], [62]. C'est pour cette raison qu'on utilise des caractéristiques, ou features, pour représenter le signal. Un feature est une valeur calculée à partir du signal qui fournit une information nécessaire à l'accomplissement de la tâche. Pour les signaux sEMG, on divise d'abord le signal en fenêtre de temps avant de calculer des features sur chaque fenêtre et chaque canal d'acquisition.

En 1993, Hudgins & al. [76] ont démontré la possibilité d'identifier une intention de mouvement en utilisant un signal sEMG une méthode de ML classique. Les features utilisés sont :

Tableau 2-2 : Features utilisés dans Hudgins & al. [76].

Nom du feature	Abréviation	Équation
Valeur moyenne absolue (Mean Absolute Value)	MAV	$\frac{1}{N} \sum_{i=1}^N x_i $
Pente de la valeur absolue moyenne (Mean Absolute Value Slope)	MAVS	$MAV_{j+1} - MAV_j$
Nombre de changement de pentes (Slope Sign Change)	SSC	$\sum_{i=2}^{N-1} [f((x_i - x_{i-1}) \cdot (x_i - x_{i+1}))]$ $f(x) = \begin{cases} \text{si } x \geq 0, & 0 \\ \text{si } x < 0, & 1 \end{cases}$
Longueur de la forme d'onde (Waveform length)	WL	$WL = \sum_{i=1}^{N-1} x_{i+1} - x_i $
Nombre de passage par zéro (Zero Crossing)	ZC	$\sum_{i=1}^{N-1} [f(x_i \cdot x_{i+1}) \cap x_i - x_{i+1} \geq \epsilon]$ $f(x) = \begin{cases} \text{si } x \geq 0, & 0 \\ \text{si } x < 0, & 1 \end{cases}$

Où i est le pas de temps, x_i est la données sEMG au temps i , N est le nombre de pas de temps total, ϵ est le seuil d'activité du signal sEMG.

Ces features sont encore considérés comme un standard pour la classification d'intentions de mouvement [20], [21], [32] et ont favorisé le développement d'une multitude d'autres méthodes de ML classique. En plus des features du domaine temporel, des features fréquentiels [77] et des features temporel-fréquentiels, comme les transformées en ondelettes [78], ont été proposés pour représenter le signal sEMG [9], [20]. Il existe plus de 50 features qui ont été testés pour la reconnaissance de mouvement [32], [79] et plusieurs combinaisons différentes sont exploitées. Depuis quelques années, des études vont même jusqu'à créer leurs propres features basées sur des observations du signal sEMG et des performances obtenues pour la détection de mouvement, une

pratique connue sous le nom d'ingénierie de caractéristiques (feature engineering) [9], [21], [80]–[82].

Quant aux algorithmes ML classique, il existe plusieurs candidats qui ont été testés à travers les années. Une revue de littérature des années 2013 à 2018 a été conduite par l'auteur de cette thèse pour étudier les méthodes d'apprentissage utilisées pour la classification de signaux sEMG au moyen de prothèses TH [24]. Elle a été complétée pour les 2 dernières années en utilisant la même méthode d'investigation et couvre ainsi les années 2013 à 2020. La Figure 2-4 présente les résultats de cette revue.

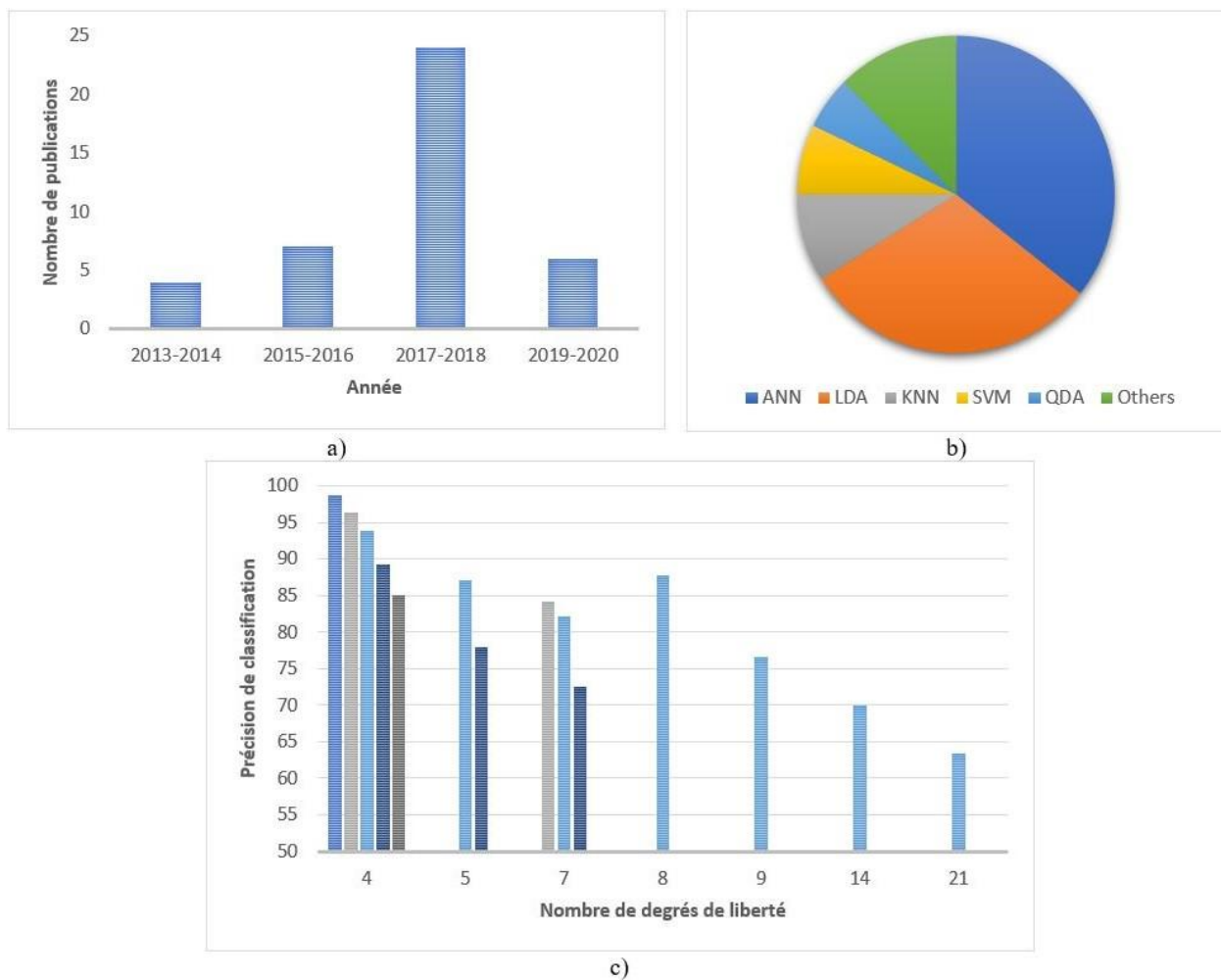


Figure 2-4 : Revue des méthodes d'apprentissage machine pour la reconnaissance de mouvement pour les prothèses TH entre janvier 2013 et décembre 2020 (adaptée de [24]). Les études qui concernent uniquement les prothèses TR ou dont les signaux sEMG ont été acquis sur l'avant-bras ont été écartées de cette revue. a) Évolution du nombre de publications pour la période visée. b)

Distribution des méthodes d'apprentissage machine utilisées. c) Précision de classification en fonction du nombre de degrés de liberté utilisés dans l'étude (seuls les résultats impliquant un ou des amputés TH ont été retenus – ceux issus d'une expérimentation sur des sujets sains n'ont pas été comptabilisés).

Comme le montre la Figure 2-4a, l'intérêt pour le ML a eu une grande croissance depuis 2013 reflétant la même tendance que dans le domaine TR. L'émergence récente de technologies basées sur la reconnaissance de forme a propulsé la recherche de dispositifs avec un contrôle naturel [19]. On remarque une baisse de publications les années 2019-2020 pour le domaine TH, ce qui n'est pas le cas pour le domaine TR qui présente plus de 200 nouveaux articles. La revue de littérature indique également que le DL n'a pas été utilisé pour les amputations TH (Figure 2-4b), une autre différence avec le domaine TR.

Bien que deuxième dans notre revue de littérature, la méthode d'êtes la plus populaire est le LDA [20], [21], [32] pour la simplicité de son implémentation et de son entraînement [7], [20]. La deuxième méthode la plus référencée est le ANN qui est plus apte à modéliser des phénomènes non linéaires. Le LDA et le ANN sont dominants dans le domaine des prothèses TH avec plus de 50 % de la distribution des méthodes ML classiques. Autrement, le SVM, les k plus proches voisins (K-Nearest Neighbors – KNN) et l'analyse discriminante quadratique (Quadratic Discriminant Analysis – QDA) sont également fréquents parmi les solutions étudiées pour les méthodes exploitant des ensembles de features [83]. Les méthodes de classifications linéaires, tel que le LDA, ont de meilleures performances en temps réel puisqu'elles sont plus simples à implémenter, mais elles n'intègrent pas le comportement actif et non linéaire des signaux sEMG issus des muscles [32]. La séparation entre les classes de mouvement étant fortement non linéaire, il est nécessaire d'utiliser une méthode d'apprentissage adapté, tel que le ANN, pour obtenir la meilleure précision de classification possible [12]. Il faut toutefois s'assurer que cette méthode permette une classification en temps réel des mouvements, soit une prédiction à chaque 250 ms ou moins. C'est entre autres ce qui explique que le LDA soit la méthode ML classique la plus utilisée pour des systèmes de classifications embarqués [20].

En observant le nombre de degré de liberté à la Figure 2-4c, on remarque que les précisions de classification décroissent selon le nombre de mouvements intégrés à la prothèse ce qui donnent lieu à des solutions peu performantes, exacerbant ainsi le problème de manque d'adhésion de

l'amputé. Jarrassé & al. [2] présente la seule étude avec une précision de classification supérieure à 85% pour six degrés de liberté ou plus. Ils présentent des résultats atteignant 88.5% et 86.9% avec deux amputés TH en utilisant un LDA avec des features du domaine temporel reportées dans le Tableau 2-3. Dépassant le seuil de 85% pour passer à une étape de validation, ils ont également présenté des résultats prometteurs lors du test de l'épingle (clothespin test) [2] qui mesure le temps pris par un amputé muni d'une prothèse pour déplacer une épingle entre une tige horizontale et une tige verticale. Cette étude inclut une étape préliminaire pour sélectionner les participants avec le meilleur contrôle de leur membre fantôme, ce qui peut expliquer leurs résultats plus élevés par rapport à ceux de la littérature qui se basent sur un recrutement souvent sans prérequis.

Tableau 2-3 : Features utilisés dans Jarrassé & al. [2].

Nom du feature	Abréviation	Équation
Moyenne quadratique (Root Mean Square)	RMS	$\sqrt{\frac{1}{N} \sum_{i=1}^N x_i^2}$
Nombre de passage par zéro (Zero Crossing)	ZC	$\sum_{i=1}^{N-1} [f(x_i \cdot x_{i+1}) \cap x_i - x_{i+1} \geq \epsilon]$ $f(x) = \begin{cases} si \ x \geq 0, & 0 \\ si \ x < 0, & 1 \end{cases}$
Entropie de l'échantillon (Sample Entropy)	SampEn	Équations disponibles dans [84].
Coefficients auto-régressifs de 4 ^e ordre (4th order Auto-Regressive coefficients)	AR	$x_i = \sum_{p=1}^4 a_p x_{i-p} + w_i$

Où i est le pas de temps, x_i est la donnée sEMG au temps i , N est le nombre total de pas de temps, ϵ est le seuil d'activité du signal sEMG, a_p est le coefficient autorégressif d'ordre p et w_i est un bruit blanc.

Gaudet et al. [4] présentent une autre étude avec des précision de classification qui varient entre 60.9% et 93.0% avec 5 amputés TH pour 9 degrés de liberté. Ils utilisent un ANN avec des features du domaine temporel présentés dans le Tableau 2-4. De plus, ils ont investigué l'utilisation de l'information cinématique du bras comme un feature pour augmenter la précision de classification du bras. Ils ont pris comme hypothèse que le positionnement du bras sain est similaire à l'information qu'il serait possible d'obtenir d'un accéléromètre sur la prothèse. Leur feature KIN est alors un regroupement d'angles articulaires du bras sain associé au mouvement et permet d'augmenter la précision de classification de 4.8% en moyenne.

Tableau 2-4 : Features utilisés dans Gaudet & al. [4].

Nom du feature	Abréviation	Équation
Valeur moyenne absolue (Mean Absolute Value)	MAV	$\frac{1}{N} \sum_{i=1}^N x_i $
Amplitude de Willison (Willison Amplitude)	WAMP	$WAMP = \sum_{i=1}^{N-1} [h(x_i - x_{i+1})]$ $h(x) = \begin{cases} 1 & \text{si } x \geq \epsilon \\ 0 & \text{si } x < \epsilon \end{cases}$
Entropie de l'échantillon (Sample Entropy)	SampEn	Équations disponibles dans [84].
Coefficients cepstraux de 4 ^e ordre (4th order Cepstral coefficients)	CC	$c_1 = -a_1$ $c_p = -a_p - \sum_{l=1}^{p-1} \left(1 - \frac{l}{p}\right) a_p c_{p-l}$

Où i est le pas de temps, x_i est la donnée sEMG au temps i , N est le nombre de pas de temps total, ϵ est le seuil d'activité du signal sEMG, a_p est le coefficient autorégressif d'ordre p (voir Tableau 2-3) et c_p est le coefficient cepstral d'ordre p .

Malgré les précisions de classification élevées obtenues par ces études [2], [4], le besoin pour une méthode de contrôle intuitive et fiable est toujours d'actualité [19]. Après des décennies de recherche à combiner des algorithmes ML classiques avec des ensembles de features [7], [17]–[20], plusieurs études comparatives s'accordent à dire qu'avec un ensemble de caractéristiques qui suit les tendances de la littérature et un nombre suffisant de canaux sEMG, la plupart des classificateurs ML classiques atteindraient une précision de classification relativement similaire [21]. La sélection et l'utilisation de features ne semblent pas être la meilleure solution pour augmenter la précision de classification [19], [20] et atteindre l'objectif de 85% pour passer à des essais cliniques. Il existe toutefois une autre branche de l'apprentissage machine qui pourrait profiter de la complexité des données sEMG [85] et qui a surpassé les méthodes ML classiques dans plusieurs domaines [86]–[88], soit l'apprentissage profond.

2.4 Apprentissage profond

Le DL a connu un développement rapide en raison des récents progrès en matière de puissance de calcul et de l'accès à de grandes bases de données (big data). Ce type de méthodes ML permet de résoudre des tâches complexes en utilisant plusieurs couches de neurones avec leur objectif respectif. Le DL peut être considéré comme une fonction mathématique composée de plusieurs fonctions plus simples reliées entre elles (couches). Chaque fois que les données passent par une couche, il en résulte une nouvelle représentation du problème qui est introduite dans la couche suivante [62]. De plus, les méthodes de DL ont tendance à s'adapter plus facilement aux variations présentes dans les données et éliminent la nécessité de choisir un ensemble de features comme c'est le cas pour les algorithmes de ML classiques [89]. En effet, les méthodes de DL peuvent utiliser le signal sEMG directement en entrée puisqu'il a la capacité d'apprendre ses propres features nécessaires à l'accomplissement de la tâche [17], [33]. L'absence de features est d'ailleurs un avantage pour les applications en temps réel puisque leurs calculs ajoutent des délais notables sur le temps de prédiction d'une fenêtre de temps.

Ce passage de l'ingénierie de features à l'apprentissage automatique de features explique en partie le succès du DL dans les dernières années [17]. Le DL a déjà révolutionné la vision par ordinateur en obtenant des erreurs de classification inférieures aux standards fixés par les méthodes ML classiques [86] et en surpassant en 2015 les cinq derniers records d'erreurs avec une différence de

plus de 10% [90]. On relate aussi plusieurs succès dans le domaine de la reconnaissance audio [87], [91] et en bio-informatique [88], [92].

2.4.1 Algorithmes d'apprentissages

Les développements récents concernant les prothèses TR s'intéressent actuellement à l'utilisation du DL afin d'établir de nouveaux standards en classification de signaux sEMG. Des publications montrent déjà que des ANN profonds, des réseaux de neurones de convolution (Convolutional Neural Network - CNN) et des réseaux de neurones récurrents (Recurrent Neural Network - RNN) avec des architectures simples dépassent les précisions de classification des méthodes ML classiques (ANN peu profond, LDA, SVM, etc.) [23]. Toutefois, l'utilisation du DL pour la classification de signaux sEMG ne s'est pas encore rendue jusqu'au domaine des prothèses TH comme en témoigne notre revue de littérature. Les ANN utilisés ne comportent qu'une seule couche cachée, ce qui n'est pas considéré comme du DL. Ainsi, les prochaines observations ci-bas sont uniquement basées sur la littérature des prothèses TR et permettront de cibler une méthode DL pour une application avec des données sur des amputés TH.

En 2016, Atzori & al. [26] ont rapporté qu'un seul article de conférence employant du DL pour des applications de prothèses myoélectriques. Ils présentent une étude préliminaire sur la classification des signaux sEMG pour les prothèses de la main avec un CNN. Leurs résultats montrent la possibilité qu'en peu de temps, il est possible d'atteindre une précision de classification moyenne de 63.43 % sur des sujets sains du même ordre que celle obtenue avec les méthodes ML classiques (61.17 %) sur les mêmes bases de données [14]. Également, leur algorithme effectue la classification en moins de 1 ms ce qui augure bien pour un système de contrôle en temps réel. Ils soulignent que ces résultats ont été obtenus sans aucune optimisation et ils identifient également des aspects de leur CNN qui pourraient être modifiés afin d'augmenter rapidement leur résultat.

Dès 2018, deux revues de littérature [22], [23] rapportent 18 études associant du DL à la classification de signaux EMG. à des signaux physiologiques. Parmi les 23 bases de données utilisées dans ces études, 20 utilisent des sujets sains pour leurs expérimentations et 3 utilisent des sujets amputés TR. Néanmoins, ces études rapportent des résultats prometteurs avec des précisions de classifications allant au-dessus de 90 % et 10 des 18 études rapportent avoir obtenu une meilleure précision de classification avec des méthodes DL lorsque comparé avec une ou plusieurs méthodes ML classiques.

Toujours selon ces revues de littérature, le CNN est la méthode DL la plus utilisée (> 50%) pour les signaux sEMG ou autres signaux physiologiques [22], [23]. Elle reste l'approche la plus populaire en raison de ses résultats reconnus et du nombre élevé d'applications disponibles dans la littérature [23], [26], [93]–[98]. Il existe une autre méthode DL, le RNN, capable de surpasser la performance des algorithmes ML classiques avec des écarts de précision de classification allant de 1.4 % à 7.5% [99]. Une autre étude rapporte aussi que le RNN atteindrait une précision de classification supérieure de 5.07 % à celle d'un CNN lorsque testé sur la même base de données avec des sujets amputés [100]. Bien que moins populaire, plusieurs exemples d'applications RNN pour la classification des signaux sEMG ont été développés avec succès [99]–[104]. Ces études ont été effectuées pour le contrôle myoélectrique TR, mais aucune implémentation n'a été accomplie avec des amputés TH.

Toutefois, les méthodes CNN ne sont pas les mieux adaptés à travailler avec des signaux sEMG. Ces derniers sont des signaux temporels dont l'information nécessaire à l'identification du mouvement est encodée dans une fenêtre de temps et non dans chaque valeur instantanée [21]. La plupart des méthodes de ML, y compris le CNN, se concentrent principalement sur la corrélation spatiale des données sEMG et n'utilisent pas la forte dépendance temporelle de la contraction musculaire [103]. Dans ces méthodes, la classification d'un pas de temps n'est pas influencée par le contenu des observations précédentes et une partie de l'information clé est alors perdue [19].

De son côté, le RNN a fait ses preuves en matière d'efficacité dans le traitement des séries séquentielles [105] grâce à sa capacité à apprendre la dynamique non linéaire et à long terme du signal [103]. Il utilise l'information provenant des observations précédentes pour influencer sa décision [19]. Cette méthode DL a été développée dans le but de mieux exploiter les séries temporelles comme les signaux sEMG et présente un réel potentiel pour les prothèses myoélectriques [19], [105]–[108].

**Méthode
d'apprentissage**

Algorithme de DL : RNN

2.4.2 Fonctionnement d'un RNN

Pour comprendre le fonctionnement d'un RNN, il faut se rappeler rapidement le fonctionnement d'un réseau de neurones de base.

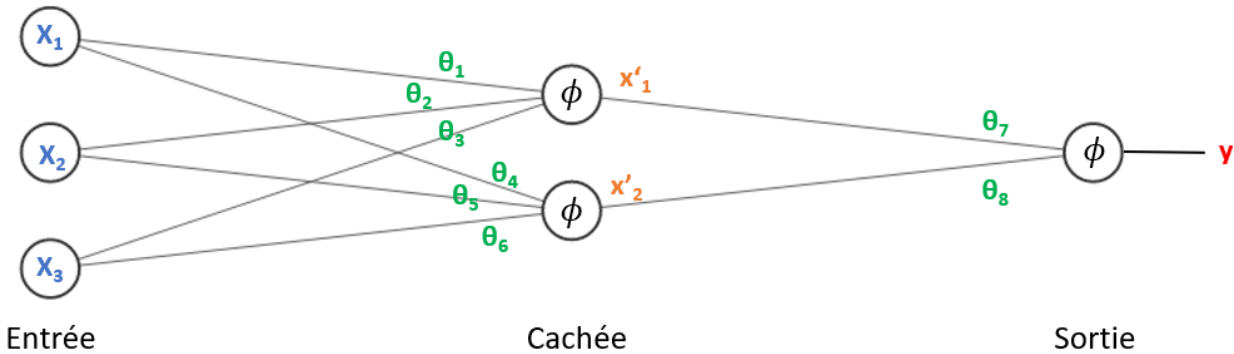


Figure 2-5 : Fonctionnement d'un réseau de neurones (x_i : donnée en entrée. θ_i : poids du réseau. ϕ : fonction d'activation. x'_i : sortie de la couche cachée. y : sortie du réseau.)

Comme à la Figure 2-5, chaque neurone de la couche cachée effectue d'abord la somme des entrées x_i pondéré par des poids θ_i . Ces poids sont les paramètres du modèle qui est développé durant l'entraînement. Une fonction d'activation ϕ préalablement sélectionnée est alors appliquée sur la somme afin d'obtenir les sorties x'_i de la couche cachée. La même procédure est alors faite pour la couche de sortie afin d'obtenir y , qui peut être une valeur continue dans le cas d'une régression ou une prédiction dans le cas d'une classification. Ainsi, pour calculer la sortie y à partir des données en entrée x_i de la Figure 2-5, on utilise le système d'équation suivant :

$$\begin{cases} x'_1 = \phi(\theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3) \\ x'_2 = \phi(\theta_4 x_1 + \theta_5 x_2 + \theta_6 x_3) \\ y = \phi(\theta_7 x'_1 + \theta_8 x'_2) \end{cases} \quad \text{Éq. 1}$$

avec $\phi(x) = \frac{1}{1+e^{-x}}$ si on sélectionne une sigmoïde comme fonction d'activation.

Afin de mieux se représenter un RNN, il est possible de condenser la représentation du réseau de neurones en cachant le nombre de neurones présents sur chacune des couches comme à la Figure 2-6.

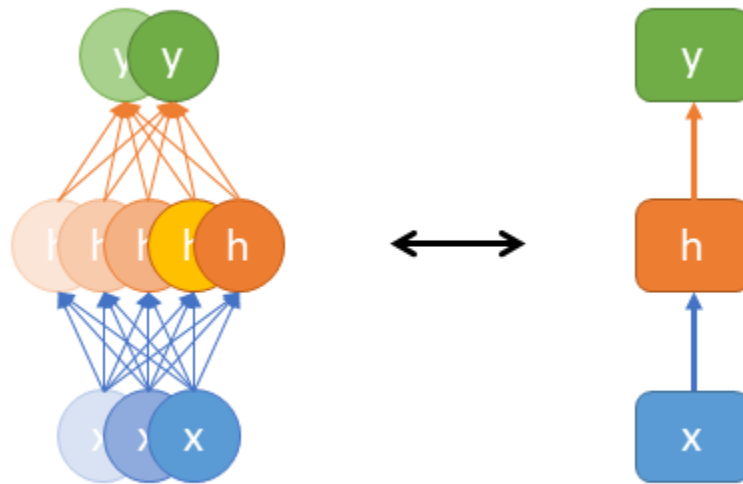


Figure 2-6 : Représentation condensée d'un réseau de neurones (x : données en entrée. h : couche cachée (hidden layer). y : sorties).

Un RNN est un type de réseau de neurones qui utilise l'information des observations précédentes pour influencer ses prochaines décisions. Il utilise un mécanisme de loupe afin de permettre à l'activation de la couche cachée de circuler d'un pas de temps vers un autre (flèche noire sur la Figure 2-7). Ce vecteur d'information qui circule à travers chaque pas de temps se nomme l'état caché (hidden state) [62] et fournit un contexte basé sur les décisions du passé. Ainsi, la sortie finale y_t du RNN est calculée à partir de la sortie de la couche cachée h_t qui utilise non seulement les données en entrée x_t , mais aussi le vecteur d'état caché qui lui fournit un contexte.

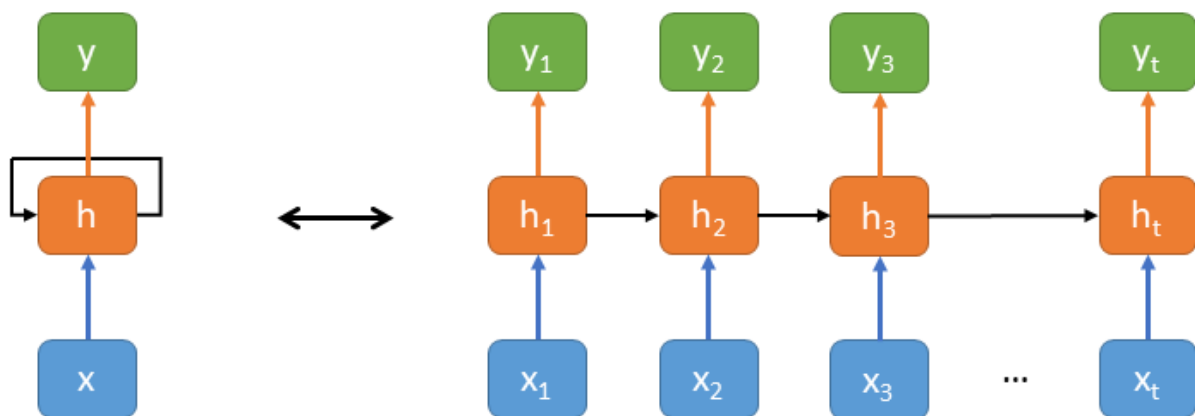


Figure 2-7 : Représentation d'un RNN (x_i : données en entrée au moment i. h_i : activations des neurones de la couche cachée au moment i. y_i : sortie au moment i.)

Un désavantage notable du RNN de base est son incapacité à gérer convenablement les dépendances à long terme (Long-Term Dependencies) d'une séquence [62], [109]. Effectivement, dans le vecteur d'état caché, le RNN accorde plus d'importance aux informations récentes qu'aux informations plus éloignées dans le temps. Ces derniers perdent progressivement de leur importance à mesure qu'on avance dans la séquence. Cet effet est problématique lorsqu'une prédiction nécessite un contexte à long terme, puisque l'apport en information des plus vieux pas de temps est minime, voire quasi inexistant.

Le RNN-Long Short-Term Memory (RNN-LSTM) a été introduit afin de résoudre ce problème [110]. Il s'agit d'une version du RNN de base qui vient modifier comment le vecteur d'état caché et les sorties sont calculés en utilisant des mécanismes de portes (gates) et des opérations tensorielles [62] (détails des équations mathématiques dans [110]). La porte d'entrée (input gate) est responsable de déterminer quelles données en entrée devraient être utilisées pour modifier la mémoire du réseau. La porte de l'oubli (forget gate) détermine quant à elle quelle information est inutile à la tâche et devrait donc être retirée, c.-à-d. oubliée. Finalement, la porte de la sortie (output gate) détermine quelle information devrait être transmise à la prochaine cellule RNN en se basant sur l'entrée et la mémoire présente dans cette cellule. À l'aide de ces mécanismes de porte, le RNN-LSTM est capable d'apprendre des dépendances à long terme plus facilement que la version RNN de base [62], [110] et est mieux adapté pour la classification de longue séquence comme les signaux sEMG. Ainsi, lors d'une transition de mouvement avec la prothèse, le RNN-LSTM est mieux conçu pour prendre avantage des signaux sEMG précédents le changement et qui contiennent de l'information annonciatrice des mouvements à venir.

Méthode d'apprentissage	Algorithme de DL : RNN-LSTM
------------------------------------	-----------------------------

2.4.3 Optimisation des hyperparamètres

En modifiant les hyperparamètres d'une méthode DL, il est possible de maximiser la précision de classification obtenue par le modèle. On appelle hyperparamètre tout paramètre qui dicte la structure du modèle ou qui dicte la façon dont les poids du modèle seront optimisés durant son entraînement. Prenons l'exemple simple du réseau de neurones à la Figure 2-5. Lors de l'entraînement, l'algorithme d'apprentissage tente de résoudre le problème d'optimisation suivant :

$$\min_{\theta} J(\theta) = E(M_{\theta}(x_j), y_j) \quad \text{Éq. 2}$$

où x_j et y_j sont des observations paires dans l'ensemble d'entraînement, θ_i sont les poids du modèle, M_{θ} est le modèle qui calcule la prédiction $\hat{y}_j$ en fonction de x_j et de θ_i , $E(M_{\theta}(x_j), y_j)$ est une fonction erreur qui calcule l'écart entre $\hat{y}_j$ et y_j , et $J(\theta)$ est la fonction coût à minimiser.

Cette optimisation se fait automatiquement par les implémentations logicielles des méthodes ML et DL. Ils utilisent divers algorithmes de descente de gradient pour y arriver comme l'optimiseur Adam [111] qui adapte la grandeur du changement exercé sur les poids θ_i à chaque itération en se basant sur une moyenne exponentielle des gradients précédents et sur l'estimation de premier et deuxième degré du moment du gradient.

Les hyperparamètres sont des paramètres qui ne figurent pas dans le modèle M_{θ} , mais qui ont une influence sur sa structure et son processus d'entraînement. Pour donner quelques exemples, le taux d'apprentissage α (learning rate) altère le gradient exercé sur les poids θ_i du modèle afin d'aider à converger vers un optimum; l'optimiseur (optimizer) détermine quelle stratégie d'optimisation est utilisée pour modifier les poids du modèle, le nombre de couches détermine la capacité du modèle à représenter un phénomène complexe et le nombre de neurones par couche augmente la quantité de paramètres θ_i du modèle. Il est alors possible de modifier le problème d'optimisation en incluant celui des hyperparamètres comme suit :

$$\min_{\epsilon} \min_{\theta} J(\theta, \epsilon) = E(M_{\theta, \epsilon}(x_j), y_j) \quad \text{Éq. 3}$$

où x_j et y_j sont des observations paires dans l'ensemble d'entraînement, θ_i sont les poids du modèle, ϵ est l'ensemble des hyperparamètres sélectionnés, $M_{\theta, \epsilon}$ est le modèle qui calcule la prédiction $\hat{y}_j$ en fonction de x_j , θ_i et ϵ , $E(M_{\theta, \epsilon}(x_j), y_j)$ est une fonction erreur qui calcule l'écart entre $\hat{y}_j$ et y_j , et $J(\theta, \epsilon)$ est la fonction coût à minimiser.

Pour améliorer les performances d'une méthode RNN-LSTM, il est possible de modifier ses hyperparamètres. Leur optimisation peut être la différence entre une performance médiocre ou qui définit de nouveaux standards dans la littérature [112], [113]. Cependant, il n'existe aucun gradient pour effectuer cette optimisation puisque la fonction entre la performance et les hyperparamètres n'est pas connue. Il faut alors tester de façon itérative des ensembles d'hyperparamètres afin

d'établir la meilleure configuration possible. Dans Éq. 3, les deux problèmes d'optimisation sont maintenus séparés puisque pour tester un ensemble d'hyperparamètres, il faut refaire l'entraînement du modèle (Éq. 2). Il existe plusieurs méthodes d'optimisation pour y arriver comme la recherche manuelle [62], [112], la recherche aléatoire [114], l'optimisation bayésienne [115], les algorithmes évolutifs [116]–[118], l'optimisation à essaim de particules (particle swarm optimization) [119], [120] et l'optimisation à colonie de fourmis (ant colony optimization) [116], [117], [121].

Bien que ces techniques d'optimisation arrivent à maximiser la performance avec un haut taux de succès, elles ne parviennent pas toujours à minimiser la quantité de ressources informatiques nécessaires pour y arriver [112], [122, p. 3-6]. En effet, de nombreuses de ces ressources peuvent être gaspillées à explorer des ensembles d'hyperparamètres qui ont très peu d'impact sur la performance ou qui l'empire [113]. Une expertise de la part de l'utilisateur est nécessaire pour identifier les hyperparamètres qui méritent d'être optimisés, de même qu'un espace de recherche approprié pour chacun [112]. Afin d'éviter d'avoir recours à un expert, des recommandations devraient être émises afin de guider les utilisateurs à travers une optimisation efficiente des hyperparamètres pour une méthode DL avec une tâche donnée. Ces lignes directrices serviraient alors à identifier les hyperparamètres qui ont un grand impact sur la performance mesurée et à définir un meilleur point de départ pour l'optimisation, de même qu'un espace de recherche plus petit.

CHAPITRE 3 : OBJECTIFS

L'objectif principal de ce projet de recherche est : **Développer une méthode d'apprentissage profond, soit le RNN-LSTM, qui reconnaît les intentions de mouvement d'un amputé transhuméral pour la commande d'une prothèse myoélectrique.**

De cet objectif découlent trois objectifs spécifiques :

OS1. Implémenter une méthode RNN-LSTM capable de reconnaître les intentions de mouvements avec une précision de classification d'au moins 85% et un temps de prédiction en dessous de 250 ms. Les sept intentions de mouvement étudiées sont la flexion et l'extension du coude, la pronation et supination de l'avant-bras, l'ouverture et la fermeture de la main, et l'absence de mouvement (repos).

Pour être considérée comme une solution viable, la performance de la méthode doit être égale ou meilleure aux métriques énoncées à la section 2.3.1 pour la précision de classification et le temps de prédiction énoncées. Pour être considérée comme une meilleure solution que les algorithmes d'apprentissage machine classique, la performance de la méthode devrait minimalement être supérieure pour l'une des deux métriques et égale ou supérieure pour l'autre. Autrement, il n'y a pas d'avantages à utiliser l'apprentissage profond par rapport aux méthodes ML classiques déjà utilisées dans la littérature, d'autant plus qu'elles sont plus simples à implémenter.

OS2. Identifier les hyperparamètres à optimiser pour l'obtention de la meilleure précision de classification avec la méthode RNN-LSTM.

Il existe une multitude d'hyperparamètres pouvant être optimisés pour améliorer une méthode d'apprentissage profond. Dans le but de ne pas gaspiller des ressources informatiques, il est important de cibler les hyperparamètres qui auront le plus gros impact sur la performance. Des recommandations propres au RNN-LSTM pour la reconnaissance de mouvement au moyen de sEMG serait bénéfique pour le domaine biomédical.

OS3. Optimiser la précision de la méthode RNN-LSTM en respectant l'objectif d'un temps de prédiction en dessous de 250 ms.

L'étape d'optimisation des hyperparamètres permet de mesurer le plein potentiel d'un algorithme d'apprentissage machine. Elle est plus cruciale pour les algorithmes d'apprentissage profond qui possèdent un plus grand nombre d'hyperparamètres pouvant influencer sa performance.

CHAPITRE 4 : ARTICLE 1 : RECURRENT NEURAL NETWORK FOR ELECTROMYOGRAPHIC GESTURE RECOGNITION IN TRANSHUMERAL AMPUTEES

Olivier Barron^a, Maxime Raison^a, Guillaume Gaudet^a, Sofiane Achiche^a

^a Department of mechanical engineering, Polytechnique Montréal, 2900 Boul. Édouard-Montpetit, Montréal, Québec, Canada H3T 1J4

Published in *Journal of Applied Soft Computing*. Vol. 96, November 2020.

Highlight

- A RNN approach is proposed for gesture recognition in transhumeral amputees.
- Learning ability of the RNN is compared with conventional methods; a LDA and a MLP.
- Electromyographic signals of five transhumeral amputees are used for comparison.
- The RNN performance is found to be more accurate than current methods.
- The RNN is better suited for real-time control due to its featureless prediction.

Abstract

Gesture recognition is a key aspect of myoelectric control of upper-limb prostheses and is rather complex to achieve for transhumeral amputees. The prosthesis control of upper arm movements must rely only on the arm muscles, which were not involved in these gestures before the amputation. For decades, machine learning has been used in research for upper-limb gesture recognition. However, reported classification accuracies for transhumeral amputees have not improved significantly since the 1990s. Latest developments in deep learning suggest it can outperform classical machine learning both in accuracy and processing time. This study aims to determine if a deep learning approach, specifically a Recurrent Neural Network (RNN), could better recognize the movement intents in transhumeral amputees. To do so, the classification accuracy and the processing time of the RNN were measured and compared to two state-of-the-art approaches that use a linear discriminant analysis (LDA) and a multilayer perceptron (MLP) respectively. All three approaches were used to classify the signals of five transhumeral amputees between 6 upper-limb gestures. For subjects 1, 3 and 5, the classification accuracy was significantly

higher ($p = 0.0002$) for the RNN (79.7%) compared to the LDA (67,1%) and the MLP (74,1%). Additionally, the RNN had a much smaller processing time, under 7 ms, compared to 385 ms and 377 ms for the LDA and the MLP respectively. Consequently, the RNN is better suited for a real-time prosthesis control that occurs between 100–250 ms. Results suggest deep learning as a viable solution for gesture recognition in transhumeral amputees.

Keyword: Classification; Recurrent Neural Network; Surface electromyography (sEMG); Phantom limb movements; Transhumeral amputee

4.1 Introduction

To recover part of their lost autonomy, upper limb amputees are presented with myoelectric prostheses able to restore specific movements using their muscle electrical activity. Transhumeral (TH — above the elbow) amputees suffer a bigger loss of functionality than transradial (TR — below the elbow) because TH amputation involves a higher number of gestures to restore. Moreover, as the forearm muscles are not present anymore, a prosthesis cannot rely on those to control the movements of the wrist and the hand. It must use arm muscles, which are not naturally involved in these gestures. These issues, specific to this level of amputation, summarize why designing novel TH prosthesis is more challenging than with TR prosthesis and why it remains particularly complex despite recent advances in technology. To achieve a functional TH prosthesis, further developments are needed regarding its control, which remains unnatural, difficult to learn by the user making the prosthesis frustrating to use [8] resulting in a higher rate of rejection when compared to TR prosthesis [2], [123]. Control possibilities need to be extended [1] to resolve the growing gap between the available state-of-the-art hardware and the control schemes limiting their actual use [2].

To work around the above-mentioned issues, classification approaches based on phantom limb movements (PLM) have been investigated [3], [4]. The phantom limb effect [50] is the result of a neuromuscular rearrangement in the amputee's limb which makes him/her feel like the arm is still attached. Some are also able to willingly move their phantom limb [52]. Interestingly, these voluntary PLM [2] generate surface ElectroMyoGraphy (sEMG) patterns that vary based on the gesture being performed [1], [3], [53], [124]. The capacity to execute PLMs has been reported extensively among amputees and constitutes an intuitive and reliable way to generate useful sEMG

signals [1], [3]. However, due to its stochastic nature, the sEMG signal displays a chaotic behavior and thus cannot be directly associated to a specific movement precisely [125]. Indeed, classification techniques, namely machine learning, are required to exploit the sEMG signal and achieve the desired classification of the gesture. This combination of PLM studies with the rapid evolvement of machine learning techniques has improved myoelectric control by attempting to associate sEMG signals to the movement intent of the amputee [3], [4], [8], [126], [127]. To achieve a successful classification, the input data needs to be described by relevant features. These characteristics must portray the data distribution in an instructive, yet discriminating manner [62]. The choice of those features represents a crucial step in designing a machine learning application and heavily influences its performance [128]. Since the first feature set for sEMG has been proposed in 1993 [76], many studies have investigated alternatives hoping to achieve new standards [7]. Recently, feature engineering has been used [23], where new features are designed to achieve an even more precise representation of the signal based on prior knowledge.

Research over the years in myoelectric control has established important insights that need to be considered. First, to capture the motor information from the sEMG signal, features require both temporal and spectral content to accurately illustrate the interaction between firing rate and motor unit recruitment in the muscle [89]. Then, although able-bodied and amputee subjects mirror the same finding across different classification schemes [129], there is a significant ($>10\%$) decrease in the classification accuracy for amputees [7]. Additional considerations such as scar tissue, condition of the residual muscles and stump morphology, need to be considered when working with TH amputees [4]. Studies using able-bodied participants might allow for easier and larger recruitment of participants but does not carry the same significance towards the development of novel prosthesis. Additionally, no established relationship exists between offline and online classification results [17], [20]–[23]. In other words, the real-time performance from a myoelectric control system cannot be predicted from its performance with a static data set. However, offline classification accuracies falling below roughly 85% have been reported to lead to a frustrating usage of the online system [7]. Finally, to achieve a real-time application of a myoelectric control system, the maximum input latency was initially 300 ms [76], but recent findings suggest values between 100–250 ms to achieve better usability [68], [69].

Machine learning applications for myoelectric control have been a research focus for decades [7], [18], [130], yet reported classification accuracies for transhumeral amputees have not significantly

improved since the 1990s. Given the importance of choosing the right features for sEMG classification, it might be more advantageous to design a system able to extract and learn its own relevant features from the sEMG signal to accomplish a certain task [130]. This shift from feature engineering to feature learning is the foundation of deep learning (DL) [95], [128] and is spreading across several fields of application [62]. While it is yet unknown how features learned by DL compared to those currently used, it has shown great potential at extracting and utilizing information [89] from stochastic biological signals such as sEMG [85]. In the recent years, DL has outperformed machine learning in various fields and is sparking interest for applications involving complex data; tremendous results have been reported in computer vision [86], audio recognition [87] and bioinformatics [88]. DL has gained popularity due to its ease of application as it eliminates the need for a feature set and can directly use the raw input signal, such as sEMG [89]. The Convolutional Neural Network (CNN), already widely used in other fields, was the first DL algorithm to be used in myoelectric control [26]. It remains the most popular feature learning approach because of its renowned results and its high number of applications available in the literature [23], [26], [93]–[98]. The Recurrent Neural Network (RNN) is another type of DL that could be effective at learning patterns from sEMG. While the CNN is specialized in processing data in a grid-like topology, such as images, the RNN is specialized in processing a sequence of values [62]. The ability of the RNN to learn from previous time steps allows it to take into account shifting behaviors instead of relying only on a static representation, like images for the CNN or features for machine learning. As the sEMG signal is a time series and contains vast and complex information [23], the RNN is considered a prime candidate for sEMG classification [102]. Examples of successful RNN applications in sEMG have been reported [99]–[102], yet these solutions have only been implemented for TR myoelectric control and not TH cases.

To our knowledge, DL has never been used to classify movement intents in TH amputees. So far, both state-of-the-art classifiers use machine learning to classify the sEMG signal coming from the stump. Jarrassé et al. have employed a Linear Discriminant Analysis (LDA) using four features and reported a classification accuracy averaging 86.4% for three degree-of-freedom (DOF) [2]. Gaudet et al. used a Multilayer Perceptron (MLP) also with four features and has reported an average accuracy of 81.1% for four DOF and 84.1% for three DOF [4]. This study aims to determine if a DL approach, specifically a RNN, could be used to better recognize the movement intents in TH amputees. To do so, the classification accuracy and the processing time of the RNN

is compared to the two state-of-the-art approaches using the sEMG signals from 5 TH amputees. Also, the impact of adding a spectral representation to the input data of the RNN was evaluated.

4.2 Methods

4.2.1 sEMG data collection

4.2.1.1 Participants

Five participants (age ranging from 21 to 59) with a unilateral traumatic TH amputation volunteered to participate in this study. All participants provided informed consent, which was approved by the Research Ethics Board of Ste-Justine University Hospital Center, Montreal, Canada. While this population might seem small, TH amputees are rare, which makes their recruitment in large numbers difficult. Similar studies rarely achieved a population higher than 5 [4].

4.2.1.2 Electrodes Placement

Six pairs of disposable sEMG electrodes (Covidien H135SG Ag/AgCl with a 10 mm recording diameter and a 24 mm center-to-center distance) were equally spaced around the stump starting with the first one in the center of the biceps brachii. Before placing the electrodes, the skin was scrubbed using an alcohol pad (70% isopropyl).

4.2.1.3 sEMG Signal Acquisition

The sEMG signal was recorded at 16-bit resolution using the wireless BTS FreeEMG300 (Gain: 476.5; CMRR >110 dB; input impedance > 10 GOhm; High-pass pre-filtered 7.32 Hz at -20 dB · s⁻¹) with a sampling frequency of 1 kHz.

4.2.1.4 Protocol

From a seating position, participants were asked to produce a specific upper limb phantom movement with a medium constant contraction. For each trial, a phantom movement was selected amongst the 6 followings: Elbow Flexion (EF), Elbow Extension (EE), Forearm Pronation (FP), Forearm Supination (FS), Hand Open (HO) and Hand Close (HC). The tag “No Movement” (NM) was used to represent when the participant was resting.

To help them attempt the gesture with their residual limb, participants were instructed to perform mirror movements with their intact limb. No prior training was conducted before this experiment.

The participant was instructed to perform the movement for 4 s, followed by a resting period of 3 s and to repeat this sequence one more time. An initial resting period of 3 s was also included resulting in the following sequence: NM (3s) – Mov. (4s) – NM (3s) – Mov. (4s) – NM (3s).

Each trial, i.e. one sequence, was repeated 3 times, cumulating in 6 repetitions (24s) per phantom movement and 24 occurrences of NM (72 s). Pauses, from 30 s to 5 min, were proposed to the participant between trials to prevent muscle fatigue.

4.2.2 sEMG data processing

4.2.2.1 sEMG signal segmentation

Once raw sEMG data was acquired for a participant, the signal is first scaled using a min/max scaler:

$$\mathbf{s} = \frac{\mathbf{d} - d_{min}}{d_{max} - d_{min}} \quad (1)$$

where $\mathbf{d}$ is the raw data vector and $\mathbf{s}$ is the scaled data. d_{max} and d_{min} represent the maximum and the minimum raw sEMG value recorded throughout the whole acquisition. As the sEMG signal was never a flat line during the whole acquisition, d_{max} cannot equal d_{min} .

As sEMG signals naturally oscillate across the X-axis, this behavior could be indicative of certain patterns. For instance, the frequent use of the Zero Crossing (ZC) in sEMG classification [7] seems to support the impact of such behavior. Hence, the range of the scaler is set to [-1, 1] as followed:

$$\mathbf{s}' = \mathbf{s} \times ([1] - [-1]) + [-1] \quad (2)$$

where $\mathbf{s}'$ is the scaled data with a range of [-1, 1].

This scaling step was necessary since raw sEMG data often ranges between 10^{-7} and 10^{-1} V in amplitude. Without any scaling, the DL algorithm did not seem to learn anything, i.e. the classification accuracy remained flat because the gradients were too small.

The scaled signal was then digitally filtered using a 30–300 Hz 4th order Butterworth bandpass filter and a 60 Hz notch filter with a Quality factor of 35. The first filter, proposed by [4], removed

the DC offset, low frequency noises, often associated with movement artifact and electrode shifting and high frequency noises, while the second cuts off the interference coming from the surrounding electrical network (60 Hz in Canada).

Afterward, to label each trial, the phantom movement must be segmented from the resting periods. The Teager–Kaiser Energy Operator (TKEO) [133] was used to highlight high sEMG activity, i.e. a movement, and determine its onset and offset time. The TKEO was calculated as followed:

$$TKEO(x(n)) = (x(n))^2 - x(n+1)x(n-1) \quad (3)$$

where $x(n)$ is the n^{th} value of the filtered signal.

The TKEO was then rectified and filtered using a 50 Hz 4th order Butterworth lowpass filter. Finally, a threshold was computed from the mean (μ) and standard deviation (σ) of the signal as followed:

$$threshold = \mu + h\sigma \quad (4)$$

The factor h served as a tuning variable to include or exclude more signal from the sEMG activity region.

Finally, depending on the movement and the segmented sEMG signal, each time steps were labeled according to the 7 possible classes, i.e. 6 phantom movements and NM.

4.2.2.2 Sliding window

Based on previous work [4], the labeled sEMG signal was divided into time windows of 250 ms with a 50 ms overlap. Time windows containing a transient portion of the contraction were not discarded as Hargrove et al. reported that, while the classification accuracy might decrease, their inclusion increases the real-time performance of the classifier [131]. Furthermore, while those results were achieved with classical machine learning approaches, the DL algorithm might further take advantage of those transient contractions leading to a lesser decrease in the classification accuracy.

After the time window transformation, the sEMG signal resulted in around 5000 samples per patient.

4.2.2.3 Short-Time Fourier Transform

Temporal features are popular and easy to implement but lack robustness to a sudden change in the signal [95]. In 2001, it was concluded that time–frequency domain features were better suited for sEMG classification, especially for a myoelectric prosthesis [134]. Since then, this feature domain is still employed by several research teams [7], [89], [93], [95], [134]. Hence, for the RNN, the Short-Time Fourier Transform (STFT) of each time window was computed and concatenated to the input data as a complimentary representation. While this addition is expensive computationally, it is expected that the increase in classification accuracy might be worth the increase in processing time.

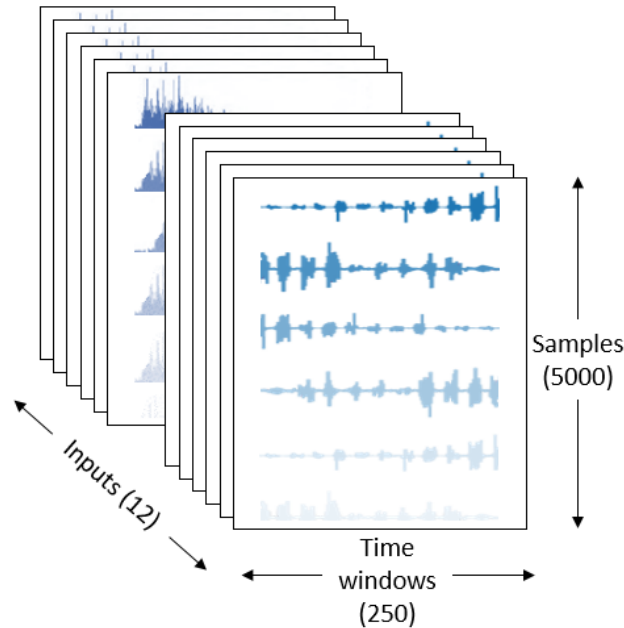


Figure 4-1 : Input data structure.

Hence, the input data for the classifier, as seen in Figure 4-1, is a 3D matrix with ~ 5000 samples $\times$ 250 ms time windows $\times$ 12 inputs, where the first 6 inputs are the filtered sEMG signal channel and the last 6 inputs are the associated STFTs.

4.2.2.4 Data split and metrics

The input data was divided into training and validation sets. To measure the performance of the classification method without any bias, the same data distribution must be present in both sets.

While 5-fold or 10-fold cross-validation is commonly used, a 6-fold cross-validation was performed because the data set was already divided into six repetitions for each movement. Hence, for each fold, a fold-accuracy is computed where five of those repetitions serve as the training set and the repetition left is used as the validation set (as described in [62]). As seen in Figure 4-2, the validation set is guaranteed to contain one of each complete movement associated with its preceding rest period, and thus have a complete representation of the data. Furthermore, each fold-accuracy was computed using a balanced accuracy, i.e. using the mean of each individual class accuracy. This ensures that the fold-accuracy represents each class equally and is not biased towards the NM class that has three times more examples than the movement classes.

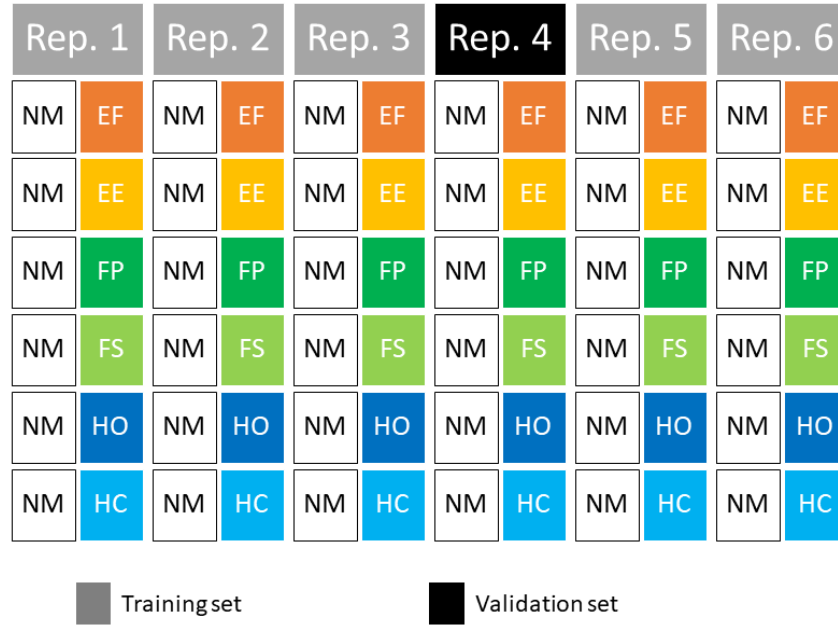


Figure 4-2 : Example of training and validation split based on the repetition. Rep.: Repetition; NM: No movement; EF: Elbow Flexion; EE: Elbow extension; FP: Forearm pronation; FS: Forearm supination; WF: Wrist flexion; WE: Wrist extension; HO: Hand open; HC: Hand close.

Due to its very deep computational graph and repetitive operations at each time step, the RNN is known for its difficulties during training [62]. For instance, after the error back-propagation, the network needs to be updated using a gradient. When this gradient propagates through the network,

its value spreads exponentially [62]. Hence, when a small gradient propagates, its value becomes so small that it does not have any effect on the learning weights anymore. This effect is known as the vanishing gradient and has been reported previously [135]. To avoid this issue, one could fix the initial state of the network, normally determined randomly, to ensure a proper gradient propagation and constant results. Although, this random initialization is a key factor for a computationally faster training of the model [62], [136]. Hence, to keep a random initialization, multiple RNN networks were trained to obtain instances without a vanishing gradient. By monitoring the gradient through the epochs, training displaying a vanishing gradient were automatically discarded.

To train a sufficient number of different models (48 for this study), 8 experiments were performed. An experiment consisted of a 6-fold cross-validation procedure where each of the 6 fold-accuracies are saved. To implement the RNN approach on an embedded platform, the best model would be selected while the rest would be discarded.

4.2.3 Classifier overview

4.2.3.1 RNN-based approach

To classify between the 7 classes (EF-EE-FP-FS-HO-HC-NM), a Recurrent Neural Network-Long Short-Term Memory [110] was chosen for its ability to learn long term dependencies easily over a data sequence [62]. The classifier and its metrics were implemented in Python 3.6 using Keras open-source library (v.2.2.4) with Tensorflow-gpu (v.1.12.0).

The RNN-based approach was built with one layer of 50 LSTM units followed by a dense layer of 7 units (one for each class) with a softmax activation. ADAM [111] was used with a learning rate of 0.01 to optimize the model. The accuracy and the loss function, i.e. the categorical cross-entropy, were computed to monitor the performance of the classifier. The validation loss was monitored to prevent overfitting. Finally, the model was fit on the training data with a batch size of 256 samples and 100 epochs. These hyperparameters were not optimized in this study. Also, the RNN was not pre-trained and was generated for each subject.

4.2.3.2 Classic machine learning approaches

In other machine learning implementations, features from the sEMG signal first have to be extracted to achieve the classification. Many features have been investigated [79], [137] and some have proven to be robust and efficient for sEMG classification. The proposed LDA [2] uses the Root Mean Square (RMS), Zero Crossing (ZC), Sample Entropy (SampEn) and the first 4 Auto-Regressive coefficients (AR). The proposed MLP [4] uses the Mean Average Value (MAV), Willison Amplitude (WAMP), SampEn and the first 4 Cepstral Coefficients (CC). The computation of these features was not optimized and was either implemented as defined [4] or using a Python library (sampen 0.0.15 and statsmodels v0.10.1).

Both approaches were implemented in Python 3.6 using the Scikit-learn library (v.0.19.1). The MLP used one hidden layer of 50 units and an output layer of 7 units. Both layers used a hyperbolic tangent sigmoid transfer function, which approximates the sigmoid function.

4.2.4 Classifier comparison

4.2.4.1 Statistics

To compare the classification accuracy between two different schemes of classification, a Shapiro–Wilk test was first performed to ensure the data has a normal distribution. As the results obtained from the RNN were not ($p < 0.05$), a one-tail Wilcoxon signed-rank test ($p < 0.05$) was conducted. To improve the power of the test, it was carried out over each of the fold-accuracies ($n = 30$) instead of the Cross-Validation means ($n=5$).

The same procedure was followed when evaluating the impact of the STFT in the input data of the RNN.

4.2.4.2 Processing time measurement

To compare the potential of the three approaches for real-time applications, two processing times were measured. The first was the calculation time of the features; it is the time between when the input data are obtained and when the features are computed. The second processing time was the prediction delay; it is the time between obtaining the features and displaying the prediction. The computer used is an Intel i7-7820X CPU @ 3.60 GHz with 48 Go of RAM at 2400 MHz and a NVIDIA GeForce GTX 1050 Ti GPU with 4 Go of RAM.

4.3 Experimental results

4.3.1 Difference amongst subjects

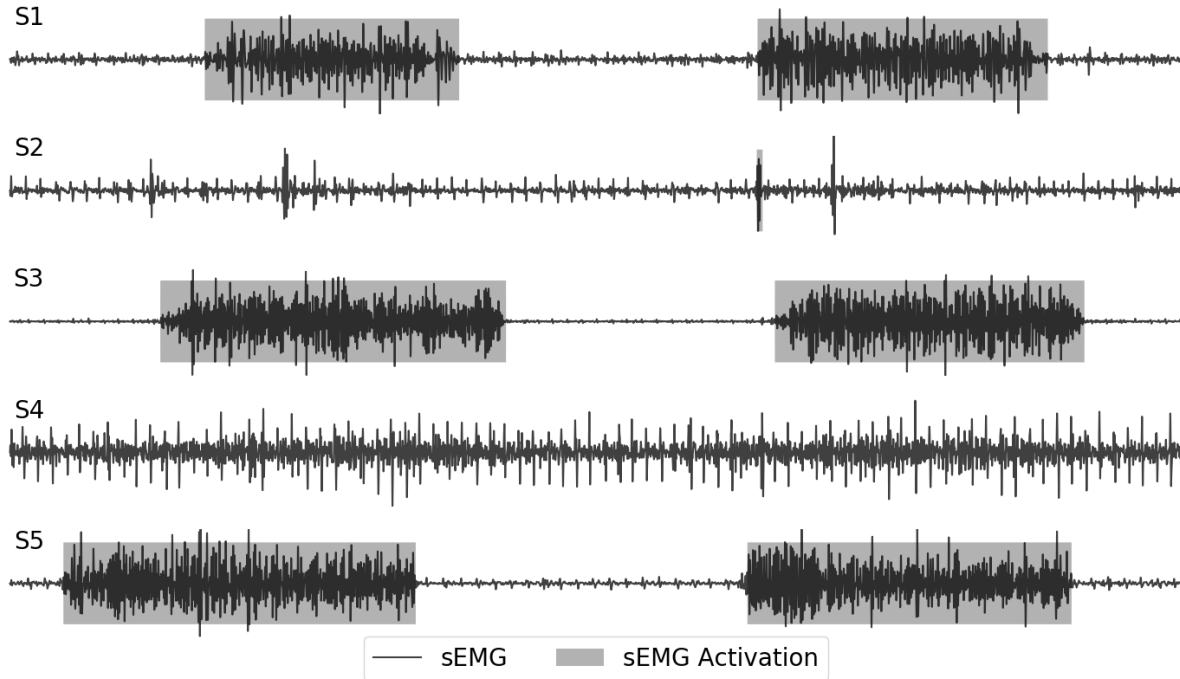


Figure 4-3 : Filtered sEMG signal for every subject (S) during an Elbow Flexion (EF). The gray areas represent the sEMG activation automatically detected using TKEO on the offline data (here, the two EFs).

A sample of the filtered sEMG signal is shown in Figure 4-3 for every subject (S). For S1, S3, and S5, the two repetitions of the gesture (EF) can be correctly identified as the sEMG signal displays two time periods with high sEMG activity. For S2 and S4, their sEMG signal shows no sustained activations throughout the acquisition. These absences of activation were observed across all trials, except for the movement EE with S2 where the sEMG signal showed activations.

When S2 and S4 performed the movements, their residual muscles generated little to no sEMG activity, which indicates they might not have the capacity to produce useful sEMG signals when performing a PLM. This capacity varies from one amputee to another and it is still unknown why some individuals lose it overtime [1]. Neuromuscular reorganization after the amputation and the use of a myoelectric prosthesis are known to have an impact on such capacity [4].

Without any observable activations, the sEMG signal is limited in terms of patterns. Therefore, the ability of the RNN to classify the signal is compromised. The classification accuracy for S2 and S4 barely improved by 10% during training the RNN, compared to 25%–40% for the other subjects. It seems that not all amputees are fit for sEMG classification based on PLM.

A preliminary evaluation of each amputee is preferable to ensure the classification will lead to satisfactory results. As presented in Jarrassé et al. through a questionnaire and a physical evaluation, this preliminary step can investigate the level of amputation, the ability to voluntarily move their phantom limb, the quality of the produced sEMG signal, the absence of phantom and residual pain, and the use of a myoelectric prosthesis [2].

For S2 and S4, the segmentation algorithm (TKEO) could not consistently identify time periods when the sEMG was activated compared to the baseline. Manual interventions were then necessary to complete the segmentation and label the data accordingly. Furthermore, if a sEMG signal has no observable activations, its manual segmentation requires other stimuli such as time flags used during acquisition or the kinematic from the able arm. Unfortunately, TH amputees are not able to perform the PLM in total synchronization with these stimuli, which incorporates a time bias between the segmentation and the activation of the muscles. Hence, it is expected that S2 and S4 may have lower classification accuracies because the RNN is not able to recognize the beginning and the end of a gesture properly, making false predictions when transitioning between gestures.

4.3.2 Impact of the STFT in the input data

The STFT was added to the input data following the recommendation of machine learning applications in myoelectric control [7], [89], [93], [134]. Although, it is not established if such practice is advantageous in the case of the RNN.

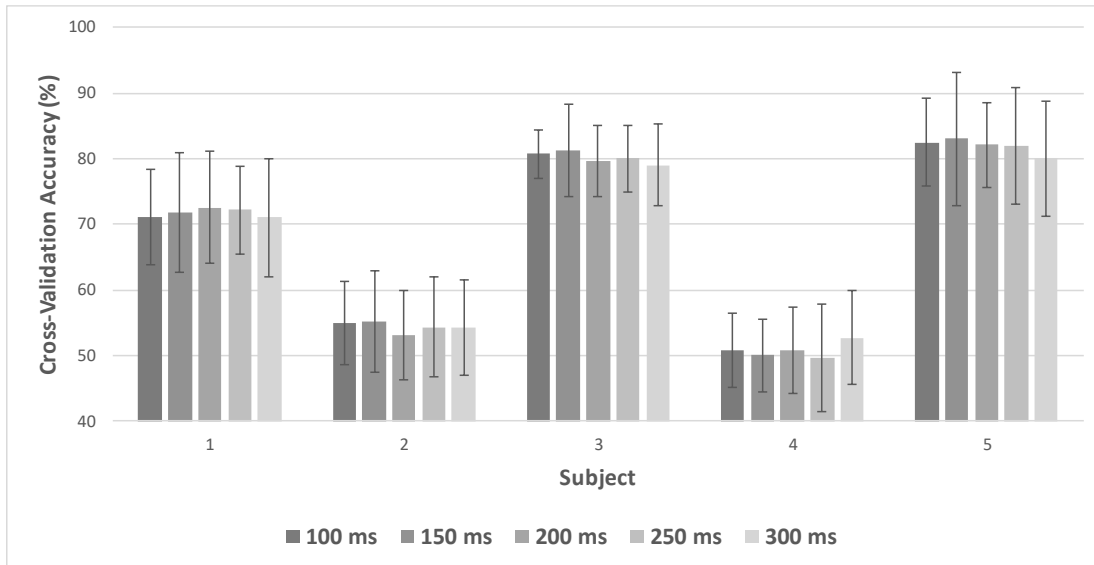


Figure 4-4 : Sensitivity study between the time window size for extracting the STFT and the Cross-Validation Accuracy obtained with the RNN.

Before evaluating the impact of using the STFT, it is necessary to establish if the classification accuracy of the RNN is sensitive to the time window size used for extracting the STFT. Thereby, for every subject, five models were trained using time windows of 100, 150, 200, 250 and 300 ms respectively. Across all models in Figure 4-4, no statistically significant differences ($p < 0.05$) were detected suggesting that the time window size used has no impact on the performance of an RNN using STFT. Thus, a time window of 250 ms is used in this study to conform with our previous work [4].

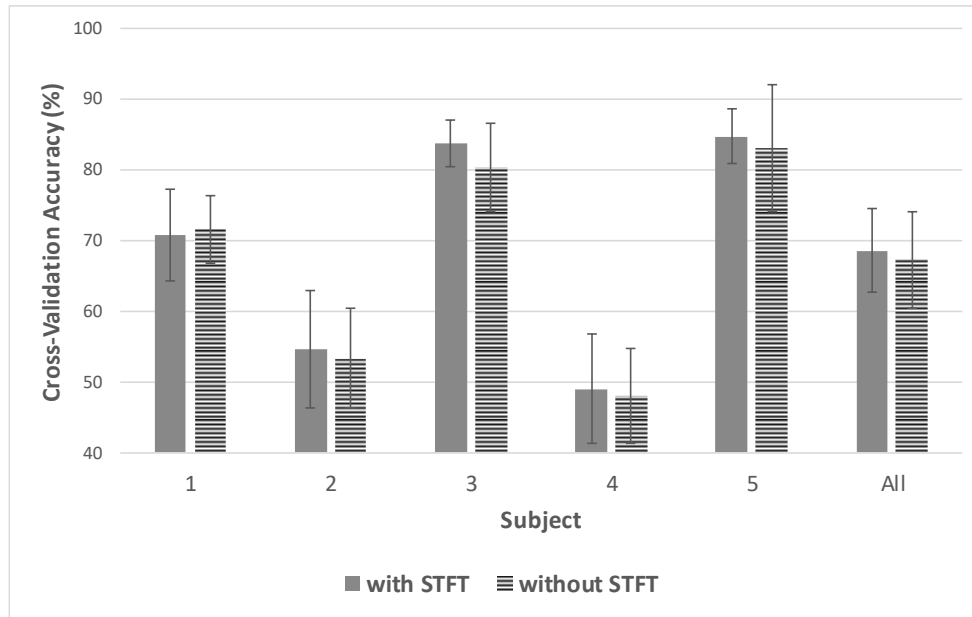


Figure 4-5 : STFT Impact. Cross-Validation Accuracy based on whether STFTs were concatenated to the input data or not for the RNN.

To measure whether the STFT of the sEMG signal has an impact on the classification performance of the RNN, two models (with or without STFT — time window of 250 ms) were trained for every subject following the Cross-Validation procedure. The accuracies obtained are shown in Figure 4-5. Even though the results for subject 2,3,4 and 5 seems to improve when the STFT is used, no statistically significant differences were detected. The same conclusion was observed in a study using EMG-based ConvNet [95] where using the STFTs achieved the same results as using the raw sEMG alone. Considering the added time needed to process the STFT for each sEMG channel, there are no benefits in adding this modality to the input data. The RNN might be able to extract relevant frequency-based features without the need to directly provide a STFT as an input as it was observed in [130].

4.3.3 Classification Accuracy

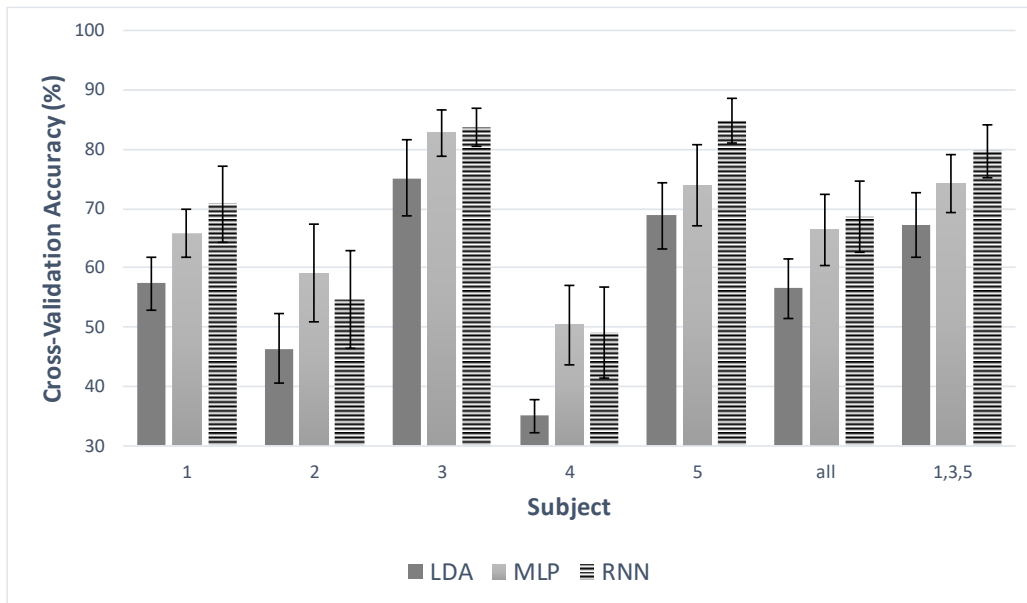


Figure 4-6 : Classification performance. The Cross-Validation Accuracy of each classification approach is reported for the 5 subjects, as well as the mean of all the subjects and the mean of the subject 1,3,5. (RNN without the use of the STFT.)

The Cross-Validation accuracies of each approach are shown in Figure 4-6. Overall, these results are lower than what can be found in the literature [3], [4]. First, this difference can be explained by the prior training that some studies allow [2], [89] to familiarize the participants with the experiment, which was not the case in this study. Then, there is the presence of transient contractions, i.e. sEMG signal recorded while transitioning between gestures, in the input data. Some studies remove these segments to have time windows containing only one class of gestures [95], [138]. However, it has been reported that including those transient contractions in the model improves the control of the amputee over the prosthesis, even if it does lower the classification accuracy [131]. Transient contractions were then kept in this study because they might contain critical information relative to shifting from one gesture to another. The RNN was specifically chosen for its ability to take into account time-dependent behaviors and could perhaps profit from these segments.

Across all subjects, the LDA obtained a significantly lower performance ($p < 10^{-6}$) than the two other approaches. As for the RNN and the MLP, they do not show a significant difference ($p = 0.07$) in the accuracies registered.

When training the three approaches for subjects 2 and 4, little to no improvement was observed in the learning curve throughout the 100 epochs. This leads us to think that, apart from catching obvious differences in the sEMG signal, e.g. NM vs EF, the classification algorithms are not able to learn well enough from the input data. Even though the MLP seems better fitted for these subjects, it fails to get over 60%, which is far too low for myoelectric control.

Otherwise, when subjects with poor sEMG activation (i.e. subject 2 and 4) are discarded, the RNN presents a significantly higher classification accuracy ($p = 0.0002$) than the MLP. These results show that the RNN is more accurate than current machine learning approaches when it comes to gesture recognition. Furthermore, subject 3 and 5 were able to achieve a satisfactory result, i.e. roughly 85%, that could lead to online testing. We believe the results of the RNN could improve even more with an appropriate optimization of the hyperparameters. More subjects would then reach the 85% mark and further surpass the current methods sEMG classification in TH amputees. It could lead to a more intuitive control of myoelectric prostheses, but also to a more robust control scheme that might be able to overcome changes in sEMG such as the electrode shift [139], the intersession variation [5] and the limb position effect [140]. These results not only confirm the potential of using a RNN but also suggest DL might be the privileged approach to use for myoelectric control.

4.3.4 Processing time

Table 4-1 : Mean time (over 50 samples) required by each approach to accomplish the feature calculation and the prediction. Processing time for the RNN is shown with the use of the STFT (w/ STFT) and without the STFT (w/o STFT).

	Feature calculation (ms)	Prediction delay (ms)	Total (ms)
LDA	385.1	0.3	385.4
MLP	376.6	0.1	376.7
RNN (w/ STFT)	0.2	6.2	6.4
RNN (w/o STFT)	0.0	6.7	6.7

The processing time for each approach is displayed in Table 4-1. From the time the algorithm receives the input data to obtain a prediction from the model, both LDA and MLP had delays higher than 250 ms when tested. These are mainly due to the feature calculation (~99.9% of the total time). The RNN, with or without the use of the STFT, is between 55 and 60 times faster than the tested machine learning methods. Its total processing time is under 7 ms, way under the 100–250 ms range for real-time prosthesis control. The calculation of the features is reduced to around 0.2 ms and the prediction delay is increased to over 6 ms, which is around 32 times longer than the LDA and MLP.

These results highlight a major difference between classical machine learning techniques and DL models. The former is often a smaller architecture but compels the user to provide a suitable representation of the input data using features. These, engineered or not, often require extensive computational power. In particular, the sampled entropy (SampEn), necessary for both LDA and MLP, takes 372.2 ms (98.8% of the feature calculation time). The feature calculation carried out in this paper was not optimized in terms of time delay and could, therefore, be faster, even reach under 250 ms. Although, its processing will always be an extra step to achieve a prediction, giving the advantage to DL.

As for the RNN, the algorithm uses feature learning to learn a suitable representation of the data. Hence, there is no need for extensive feature calculation other than complimentary modality like the STFT. The RNN complexity resides in the architecture and therefore takes more time to provide a prediction. Nevertheless, the RNN is proving to be suitable for real-time applications with a low processing time (< 7 ms), outperforming the LDA and MLP.

The processing times above were obtained using a desktop computer. In future works, they should be measured on an embedded platform to have a better estimate of the final processing time. Meeting the time constraint for real-time prosthesis control will revolve around several factors such as the processing power of the controller, the coding language, the parallelization of certain processes, and more. Rezk et al. have reviewed existing implementations of RNN models on embedded platforms (one study reports a real time response of 16 ms) and have discussed strategies to obtain efficient and performant implementations [141]. Moreover, for prosthesis control, it is not mandatory to train the network in real time on the device. The training can be done in advance on a computer and the model is then transferred to the controller. Taking into account that no feature needs to be computed, the training can be done offline and optimization strategies exist for RNN implementation [141], we believe that our RNN approach could meet the time constraint of 100–250 ms for real-time prosthesis control.

4.4 Conclusion

The aim of this study was to determine whether a deep learning approach, specifically a Recurrent Neural Network (RNN), could be used as an alternative to recognize movement intents in transhumeral amputees. To do so, the classification accuracy and the processing time of the RNN were measured and compared to two state-of-the-art approaches that use a linear discriminant analysis (LDA) and a multilayer perceptron (MLP) respectively. The RNN was able to achieve a mean accuracy of 79.7% over 3 subjects compared to 74.2% for MLP and 67.1% for LDA, which indicates a significant increase ($p = 0.0002$) for the RNN. Additionally, the RNN was able to both process the signal and predict the gesture much faster, under 7 ms, compared to 385 ms for the LDA and 377 ms for the MLP. As such, the RNN is better suited for a real-time prosthesis control that occurs between 100–250 ms. Although, the optimization of feature calculation for the LDA and MLP was not done and could result in a faster processing time using real-time computing. Both the classification accuracy and the processing time show the high potential of using a RNN, but

also that it is more accurate and faster than the state-of-the-art methods. It thus suggests deep learning as a viable solution for gesture recognition in transhumeral amputees.

Future work will also focus on optimizing the hyperparameters, which has not been done in this work and experimenting on different DL architectures to further improve the classification accuracy. By achieving a satisfactory accuracy, i.e. 85% for most of the subject, it could lead this control system to online testing where its real-time application shall be validated.

4.5 Acknowledgment

This research was supported by the Fonds de recherche du Québec - Nature et Technologies (FRQNT) and the Canada First Research Excellence Fund. The authors would like to thank the patients who participated in this study along with the team from Institut de Réadaptation Gingras-Lindsay de Montréal (IRGLM) who helped with the recruitment of participants.

CHAPITRE 5 : ARTICLE 2 : IMPACT OF HYPERPARAMETERS ON THE PERFORMANCE OF A LSTM NETWORK FOR GESTURE RECOGNITION USING SURFACE ELECTROMYOGRAPHY

Olivier Barron^a, Maxime Raison^a, Sofiane Achiche^a

^a Department of mechanical engineering, Polytechnique Montréal, 2900 Boul. Édouard-Montpetit, Montréal, Québec, Canada H3T 1J4

Submitted to *Journal of Applied Soft Computing* on March 2021.

Abstract

The hyperparameter optimization of a network is crucial, as it can improve or worsen the performance for a given task. This step commonly requires important processing power, computation time, and memory cost to explore the search space for the best hyperparameter configuration. To save those computational resources, the optimization should focus on hyperparameters that have a high impact on the performance and avoid those that are insignificant. This study aims to determine the impact of eight hyperparameters on the performance of a Long Short-Term Memory (LSTM) network solving a gesture recognition task using surface electromyography (sEMG). To this end, the control of a transhumeral myoelectric prosthesis is used as the benchmark task. Its sensibility to classification accuracy, where a small gain can be the difference between a frustrating or an intuitive device, makes it a good candidate for hyperparameter optimization. Hence, to perform the impact analysis, sEMG signals from a transhumeral amputee were used as the benchmark data. Every hyperparameter value or option was tested on 60 random network configurations with their performance recorded. The results allowed to establish an impact, a guideline and recommended value or options for each hyperparameter. These recommendations are the main contribution of this paper and should help optimize the hyperparameters for gesture recognition more efficiently by providing a better starting point and a narrower search space while using less resources for each hyperparameter. Using the recommended LSTM network set, the classification accuracy was improved by 5.56 % over the baseline set performance. Additionally, an optimal hyperparameter set was suggested for solving the

benchmark task. For three transhumeral amputees, the optimal set classification accuracy showed an average improvement of 7.16 % over the baseline set performance.

Keyword: Hyperparameter optimization; Recurrent Neural Network; Long Short-Term Memory; Gesture recognition; Transhumeral amputee;

5.1 Introduction

To further improve the performance of a LSTM network, its hyperparameters should be optimized. We define hyperparameters as any parameter that affects the behavior of the network while learning. It can be a parameter specific to LSTM, e.g. the learning rate, the number of layers, or the number of units per layer. It can also be an extra process added to the network, sometimes called extensions [112], such as dropout, a layer of batch normalization, or a layer of self-attention. The hyperparameter space is the set of values or options that the hyperparameter can take. The search space is then the set of all the hyperparameter spaces used for the optimization.

The hyperparameter optimization is central to machine learning applications, as it can be the difference between a great and a mediocre performance for a given task [62], [112]. Gradients cannot be used to perform the optimization as the function between the performance, e.g. the validation loss or the validation accuracy, and the hyperparameters is not known. Instead, the best hyperparameter configuration is found using various optimization techniques. Starting with manual search, this basic strategy involves the user in searching for a high-performance region in the hyperparameter space by manually tuning every hyperparameter. While it can be effective for a small number of hyperparameters, this technique is often too challenging to pursue on large sets. A more systematic approach would be the grid search [62]. Each hyperparameter space is divided into discrete values creating a grid of possible configurations. Then, every combination is tested to identify the best configuration to use. However, this technique needs to be completed in full to give the expected result. This can be computationally expensive as the number of configurations to test is exponential to the number of discrete hyperparameter values involved [62]. Random search [114] was introduced as an alternative to grid search and it was shown to be more efficient theoretically and empirically. With this technique, instead of dividing each hyperparameter space in discrete values, the space itself, e.g. a range of value or option, is input into the algorithm. It then randomly chooses a value for every hyperparameter before testing the configuration. This way, more regions

of the search space are explored in the same number of iterations as a grid search. Random searches can identify configurations at least as good as a grid search [114]. Adding the fact that the process can be stopped and resumed at any time, random search proves to be a less computationally expensive, easier to achieve, and at least as effective as a grid search. The disadvantage with a grid search and random search is that the algorithm is independent of the previous findings, hence it cannot adapt its process while exploring the search space. In contrast, Bayesian optimization is updated after each trial and uses a probability surrogate model to map the relationship between the tested performance and the hyperparameters used [115]. Bayesian optimization was found to outperform both grid search and random search by finding the best configuration in fewer trials [112], [115]. Although, it is more complex to use, more difficult to parallelize the process, and when the number of studied hyperparameter increases, its efficiency is on par with random search [142]. There exist many other effective hyperparameter optimization techniques such as bio-inspired metaheuristics, e.g. evolutionary algorithms [116]–[118], particle swarm optimization (PSO) [119], [120] and ant colony [116], [117], [121] to name a few.

While these techniques present effective ways to achieve a hyperparameter optimization, their success still relies on the user’s expertise. A selection of the right hyperparameters to optimize must be done, along with choosing a suitable search space for each. Otherwise, many resources can be wasted on investigating insignificant options. To avoid this issue, the impact of the hyperparameters on the network performance should be evaluated to gather some insights on their optimization. Impact studies of hyperparameters have been conducted on a variety of machine learning approaches [113], [143]–[149] and two were carried out on a LSTM network [112], [150]. These papers did not use sEMG signals as input data, but instead used datasets of sentences to accomplish various NLP tasks [112], sound recordings for music classification or speech recognition, and datasets of images for classification purposes [150]. As LSTM networks learn different type of features based on its input and the task it needs to solve [62], the impact of the hyperparameters affecting this learning process is also expected to differ. Therefore, an impact study is done for our specific task, i.e. gesture recognition using sEMG signals. The aim of this study is to evaluate the impact of eight hyperparameters on the performance of a LSTM network solving a gesture recognition task. Guidelines and a recommended configuration will be provided for each studied hyperparameter.

Serving as the benchmark task, the control of a transhumeral myoelectric prosthesis is a great example of a gesture recognition application where a small gain in accuracy is crucial to the functionality of the device. Results showed that a classification accuracy below roughly 85% resulted in a prosthesis frustrating to use for the amputee while a classification accuracy of 90% was a reasonable objective for an intuitive control [7]. To give more context, transhumeral amputees are able to recover part of their lost autonomy by using myoelectric prostheses [24]. These devices use the muscle electrical activity of its user to restore specific gestures [27]. This activity can be measured on the surface of the skin and is referred to as surface electromyographic signal (sEMG) [5]. Interestingly, sEMG displays different patterns based on the gesture performed by the user [76]. Gesture recognition strategies such as machine learning take advantage of those patterns to activate myoelectric prostheses according to the user intent [2]–[4]. In this era of big data and deep learning, these technologies are now more accessible, mature, and their use have further increased the performance of gesture recognition techniques using sEMG [23]. Recent studies have shown the potential of the LSTM for gesture recognition using sEMG with transradial amputees [151], [152] and transhumeral amputees [153]. Along with the impact study, this paper will also identify an optimal set of hyperparameters that increases the performance of the LSTM network compared to the default values as in [153].

5.2 Selected Hyperparameters

The following section presents the hyperparameters that will be used in this paper. They have been selected based on their known impact on the performance of a LSTM or in gesture recognition. This impact is described in each sub-section to highlight the benefits of either including the process or fine-tuning a certain value.

5.2.1 Architecture

The architecture of the LSTM network refers to its number of layers, number of LSTM units per layer, and their arrangement. According to the universal approximation theorem [154], a network with a single layer can represent any functions given it has enough hidden units. However, this number of hidden units may be excessively high and the network is not assured to generalize well or even be able to learn [62]. By adding additional layers, it is possible to decrease the number of hidden units required and to achieve a good generalization [62].

Adding LSTM units on the hidden layers increases the width of the model, which increases its capacity. The capacity of a model is defined as its ability to solve a task with a given complexity [62]. A model with low capacity cannot solve complex tasks and struggles to fit the training set (underfit). A model with too high capacity is long to train and may memorize properties from the training set alone that does not generalize to the test set (overfit) [62].

Adding layers increases the depth of the network, which increases its level of abstraction. A deep network can be seen as a complex mathematical function where each layer is a simple function with a purpose [24]. The output of the previous layer becomes the features of the next, thus providing a new representation of the input vector [62]. It does influence the capacity of the model by decreasing the number of LSTM units required as the number of layers increases.

The right architecture for a task cannot be easily predetermined and requires testing multiple configurations to find the right number of layers and LSTM units that offers the right capacity [62]. In the gesture recognition literature, the number of units can vary from 25 to 512 [107], [112], [155], [156] while the number of layers is usually kept between one and three [112], [155], [156].

5.2.2 Batch Normalization

Batch normalization was introduced to fix the internal covariate shift [157]. This issue is caused by the changes in the distribution of every layer's input as parameters are updated throughout the network. Batch normalization solves this issue by standardizing the output of each layer, i.e. rescales the units to have a mean of zero and a standard deviation of one, after each mini-batch. This stabilizes and speeds up the learning while still allowing the relationship between the units to be updated [62]. Batch normalization can also be used as a regularization tool, often replacing dropout [62], [157].

5.2.3 Bidirectional LSTM

The basic LSTM is unidirectional, meaning it passes through the input sequence in the forward direction only [62]. Given an input vector $\mathbf{x}$ of length m , a forward layer processes the vector from x_1 to x_m . For an index n , the LSTM layer would remember, through its hidden states, the past elements X_1 to X_{n-1} but would know nothing about the future ones, X_{n+1} to X_m .

It is possible to use a LSTM layer in the backward direction [62]. The layer would simply read the input vector in reverse from X_m to X_1 , or future to past. The backward sweep better use the information from the future, i.e. the end of the input, which can help to understand the context contained in the input [107], [112].

Bidirectional LSTM was introduced by Schuster *et al.* and is composed of two parallel LSTM layers, one forward and one backward, that are trained simultaneously [158]. They both take the same input vector and their output is concatenated, giving a more complete representation of the input vector. Gesture recognition could benefit from bidirectional LSTM, as the backward layer would extract new information from the sEMG signal that could further help to dissociate the gestures from one another. The downside of bidirectional LSTM would be that for each bidirectional layer, two distinct layers need to be trained, the forward and the backward, which increases the training time of the network. The same can be said for predicting new gestures in real time, as the network would possess twice as many parameters than a unidirectional LSTM, which increases the memory required for the network and the processing time to get the final prediction.

5.2.4 Dropout

Dropout is a regularization technique commonly used to prevent overfitting in the model [62]. For each batch, it builds a binary mask that randomly turns off, or drops, input, and hidden units by multiplying their output by zero [159]. The dropout rate is set between 0 and 1 and represents the ratio of units that will be deactivated. This process forces the network to make the best use of every unit because they are not able to co-adapt to each other. It thus improves the generalization of the model as it cannot rely on specific inputs or specialized hidden unit to do its prediction [145].

Standard dropout, as mentioned above, causes performance issues with RNN because it hinders its memory ability by dropping recurrent connections between time steps [160]. To avoid this issue, Zaremba *et al.* proposed to apply dropout only on the input and the output connections; this specific technique is known as naive dropout [161]. While naive dropout can be effective alone, certain RNN models still fail at preventing overfit [161]. Gal *et al.* then introduced variational dropout where three different masks are applied to the input, output, and recurrent connections, but the masks stay the same between timesteps. Further details can be found in [161].

5.2.5 Learning Rate

The learning rate is a positive scalar, often between 0 and 1, determining the step size taken during the optimization of the model. A small learning rate requires more epoch to achieve an optimal solution and may cause the training to be stuck with poor performance. A large learning rate is faster but may not be able to converge to an optimal minimum that leads to high training error. It is referred to as the most important hyperparameter to optimize [62], [150] and displays a U-shaped curve when monitoring the training error. Hence, an optimal value can be achieved for a given model.

5.2.6 Self-Attention Layer

Attention is what allows humans to focus on relevant details while ignoring others. In deep learning, this mechanism was mimicked to enable networks to focus on key aspects while scanning through a sequence. Attention searches for positions in the sequence where the most task-related information is concentrated and cumulates these importance weights in a vector [162]. It also assesses how strongly a certain position correlates with the previous ones. It is mostly useful for tasks using data with a context such as words in a sentence, time steps in a sequence, or even particular details in an image.

Attention was created for neural machine translation (NMT) using RNN encoder-decoder [163], [164] to solve the issue regarding long sentences [162]. The encoder is tasked with summarizing the input sequence in a context vector, e.g. reading a French sentence and memorizing the meaning. The decoder then uses the context vector to generate the desired output, e.g. translating the context vector in a new language such as English. The success of this translation partly relies on the encoder creating the best context vector possible. Although, it is more difficult when the sentence gets longer, as the encoder must summarize all the information in the fixed-length context vector. The encoder sometimes forgets information or idly summarizes it. By adding attention to the RNN encoder-decoder, the network can correlate words between them in the sentence and to focus on the relevant parts to better reflect the meaning of the sentence. There are multiple forms of attention proposed in the literature that varies in terms of how the alignment score is computed. The additive attention [162] and dot-product attention [165] are the most commonly used. It is possible to use attention without using an RNN encoder-decoder. Self-attention [166], [167], sometimes referred

to as intra-attention, uses the last layer hidden states to correlate each element of the sequence with the previous ones.

The self-attention layer uses the last layer hidden states $\mathbf{h}_i$ of length n as input and returns a context vector $\mathbf{c}_j$ of the same length n as output. If the previous layer is a bidirectional LSTM layer, $\mathbf{h}_i$ corresponds to the concatenation of both the forward and the backward hidden states. First, the alignment score is computed based on the chosen form:

$$score(\mathbf{h}_j, \mathbf{h}_i) = \begin{cases} \mathbf{v}_a^T \tanh(\mathbf{W}_a[\mathbf{h}_j; \mathbf{h}_i]) & \text{additive [162]} \\ \mathbf{h}_j^T \mathbf{h}_i & \text{dot - product [165]} \end{cases} \quad (1)$$

where $i = [1, n]$, $j = [1, n]$ and both $\mathbf{v}_a$ and $\mathbf{W}_a$ are weights that will be learned during training. Next, the attention weights $\alpha_{j,i}$ are obtained:

$$\alpha_{j,i} = softmax(score(\mathbf{h}_{j-1}, \mathbf{h}_i)) = \frac{e^{score(\mathbf{h}_{j-1}, \mathbf{h}_i)}}{\sum_{i'=1}^n e^{score(\mathbf{h}_{j-1}, \mathbf{h}_{i'})}} \quad (2)$$

Finally, the context vector $\mathbf{c}_j$ is computed by the sum of the last layer hidden states $\mathbf{h}_i$, and the attention weights multiplied together:

$$\mathbf{c}_j = \sum_{i=1}^n \alpha_{j,i} \mathbf{h}_i \quad (3)$$

Self-attention can be beneficial for gesture recognition [106], [107], [168] as it allows the network to focus on sEMG time steps that expose behaviors related to certain movements.

5.2.7 Window Size and Window Increment

To predict a gesture with a LSTM, a time window, i.e. consecutive sEMG data points of a certain time length, needs to be provided. It is possible to input these time windows one after the other, but using a sliding(overlapping) window allows for better use of the continuous data both in training and real time [80]. This technique first captures a time window of a certain size at a given position in time. It then increases this position (slides the window) by a value inferior to the window size, effectively overlapping the new window with the previous one.

To use sliding windows, two parameters need to be defined. First, the window size specifies the length of the window in milliseconds. This parameter dictates one dimension of the LSTM input

vector and thus has an impact on the quantity of information that is used for the prediction. Giving more information to the model would normally result in better classification accuracy, but the LSTM can struggle with too large time windows because of its problem with long-term dependencies [62]. Second, the window increment specifies how much time does the time window slides through the stream of data. In other words, it defines how many of the oldest data will be replaced by new incoming data. This parameter has an impact on both the training and the real time testing. It controls how to divide the continuous stream, which determines how many examples will be provided to the LSTM to train on. It also determines at which frequency predictions will be made. This has a direct impact on real-time performance as the system needs to acquire a new signal for a time equal to the window increment before providing the LSTM with a new input vector. Decreasing the window increment would provide more examples to the LSTM for training and would increase the prediction frequency, which could lead to faster control of the prosthesis. Although too small a window increment would result in similar time windows, i.e. examples that look alike, that could push the network towards overfitting. Additionally, increasing the prediction frequency would add further challenges in terms of real-time computing. A review of existing methods in EMG pattern recognition [169] shows that studies use a window size between 100 and 300 ms (majorly of 150, 200, or 250 ms). As for the window increment, it reveals that studies use a larger variety of values between 5 and 175 ms, with 50 and 100 ms being more popular.

5.3 Methods

The following section presents the methodology used to investigate the influence of every studied hyperparameter. First, the process to build the LSTM-network is explained based on the hyperparameters involved. Next, the gesture recognition task that will serve as our benchmark test is presented. Finally, the evaluation methodology is explained and how the different results are obtained.

5.3.1 LSTM-network for sEMG gesture recognition

The LSTM classifier is implemented in Python 3.6.10 using Tensorflow-GPU 2.1.0. The network is not pre-trained, and a new model is created for each configuration. As seen in Figure 5-1, the subject data input in the network is a three-dimensional matrix. It contains S samples, with a length equal to the window size ws and a dept equaled to the number of channels ch .

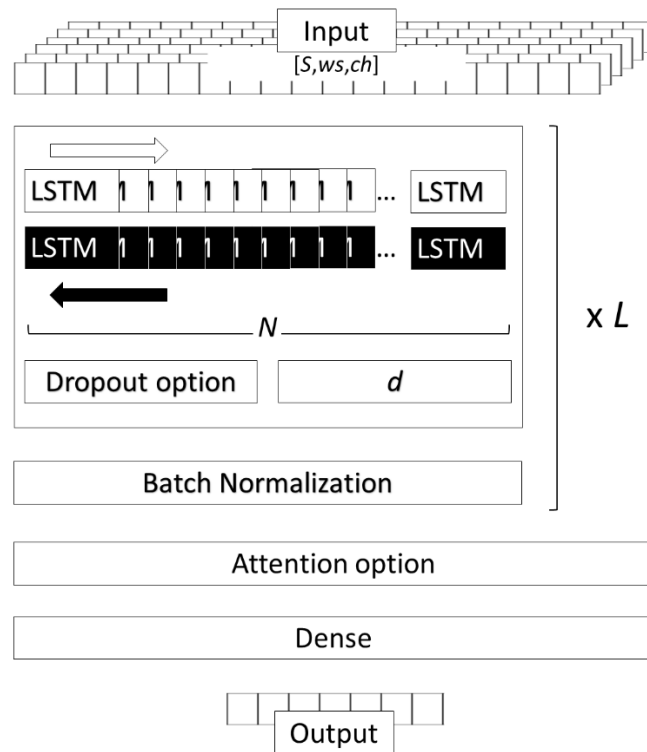


Figure 5-1 : LSTM network full build. (“S”: sample size. “ws”: window size. “ch”: number of channels. “N”: Number of LSTM units. “d”: dropout rate. “L”: number of layers.)

A LSTM layer is then added with N units. If a bidirectional LSTM is selected, the Tensorflow *Bidirectional()* wrapper is applied to the LSTM layer, which adds a backward layer as seen in black in Figure 5.1. If a naive dropout is selected, a dropout with a rate d is applied to the input vector and the output of every LSTM layers. If a variational dropout is selected, the same procedure with naive dropout is applied, except a dropout of the same rate d will also be added on the recurrent connections. These dropout options are implemented using the attribute “dropout” and “recurrent_dropout” of the Tensorflow LSTM layer. If a batch normalization is selected, the Tensorflow *BatchNormalization()* layer is added right after the LSTM layer. Both the LSTM layer, with its selected hyperparameters, and the batch normalization are repeated L times for the number of selected layers as shown in Figure 5.1.

If a self-attention option is selected, either the Tensorflow *Attention()* layer for Luong’s attention or the Tensorflow *AdditiveAttention()* layer for Bahdanau’s attention is added. The output of the previous layer and the self-attention layer are then concatenated. If a dropout option is selected, either naive or variational, the Tensorflow *Dropout()* layer with a rate d is added. The final layer

is a fully-connected with seven units, associated with the seven classes, and uses softmax activation.

The model is trained using the selected learning rate lr with the ADAM optimizer [111]. It uses a balanced accuracy where the mean of each class accuracy is computed as a metric. This ensures that no bias is added because of inequality in the number of samples for each class. The categorical cross-entropy, as implemented in Tensorflow, is used for the loss function.

The model is fit using a batch size of 32 samples and 300 epochs. A delayed early stopping with a patience of 25 epochs is used to prevent overfitting. Preliminary tests with multiple configurations showed instances where the LSTM took between 40 and 80 epochs before significantly decreasing its training loss. To consider this behavior, a delay of 100 epochs was allowed before applying early stopping to the training.

5.3.2 Benchmark task

The classification of seven gesture intents in transhumeral amputees is chosen as the benchmark task for this study. The sEMG data come from three participants (age ranging from 21 to 46) with a unilateral traumatic TH amputation that volunteered to take part in a previous study [4]. Informed consent was provided to the participant and approved by the Research Ethics Board of Ste-Justine University Hospital Center, Montreal, Canada. Details surrounding the sEMG signal acquisition, i.e. the electrodes placement, the equipment, and the protocol, can be found in [4], [153]. According to the protocol, the signal for every gesture is already divided into six repetitions containing the same data representation. Chosen randomly, one repetition serves as the validation set, another serves as the test set, and the rest is used as the training set.

The raw sEMG data was processed for classification as in [153]. After the acquisitions, the sEMG signal was first segmented between showing sEMG activity or not, using the Teager-Kaiser Energy Operator (TKEO) [133]. The segmented signal was then labeled at every time step with the six gestures, or a no movement label, based on the order the gestures were performed by the participant. The labeled signal was filtered using a 30-300 Hz 4th order Butterworth bandpass filter and a 60 Hz notch filter with a quality of 35. The filtered signal was scaled within a range of [-1, 1] using a min/max scaler. The scaled data was saved and ready to be used by the RNN. The last step before

inputting the training set into the model would be to divide the data into time windows following the selected window size ws and window increment wi .

5.3.3 Impact evaluation

To evaluate the impact of every selected hyperparameter on the classifier performance, an empirical random research is conducted. First, all the hyperparameters are chosen randomly among a defined search space except for the studied hyperparameter H . Then, a LSTM network is trained on the benchmark task for every studied value or option of H while the other hyperparameters remain the same to allow a fair comparison. This process is repeated 60 times for H to detect statistically significant differences in test accuracies between H tested values or options. Randomly changing the other hyperparameters allow the results to be independent of the configuration. Depending on whether there is a statistically significant difference or not, the hyperparameter impact can then be determined and a recommendation is made. The experiment is restricted to testing 60 random configurations for each hyperparameter on one subject (S1). This decision was made because evaluating a single configuration for a given value or option can take between 10 and 30 minutes, resulting in an experiment requiring months of testing and high computational resources. Other subjects (S2-S3) are used further to evaluate the best hyperparameter set to accomplish the benchmark task with an LSTM network.

First, the search space is created by defining each hyperparameter space. These spaces represent all the possible values or options that can be selected when randomly choosing the configuration of the LSTM network. For the architecture, i.e. the number of LSTM units N and the number of layers L , and the window size ws , a range of integer values is defined with a minimum and a maximum. For the window increment wi , only a minimum value is provided. Its maximum possible value is defined by the selected ws as the window increment cannot be bigger (otherwise, the time window would skip data when progressing). For the dropout rate d and the learning rate lr , a range of continuous values is defined by a maximum and minimum. For the batch normalization bn and the bidirectional layer bl , a binary space defines whether the LSTM network is built with or without the option. For the dropout option do and the self-attention layer sa , three categories defines if the LSTM network uses one of the two forms or none. Table 5-1 shows the resulting search space.

Table 5-1 : Search space

Hyperparameter		Symbol	Search space
Architecture	Number of LSTM units	N	[20, 512]
	Number of layers	L	[1, 3]
Batch Normalization		bn	[with, without]
Bidirectional Layer		bl	[with, without]
Dropout option		do	[None, Naive, Variational]
Dropout rate		d	[0.05, 0.5]
Learning rate		lr	[0.00005, 0.05]
Self-attention		sa	[None, Luong, Bahdanau]
Window increment		wi	[10, wS]
Window size		ws	[52, 300]

Next, a study space is defined for each hyperparameter. This space represents every hyperparameter value or option that will be tested to evaluate its impact. For categorical hyperparameters, the study space is the same as the search space in Table 5-1. For the other hyperparameter involving a continuous space, five discrete values are chosen to be tested based on the range defined in Table 5-1. For the number of layers L , the window increment wi , and the window size ws , evenly spaced values are drawn from a linear interval defined by the minimum and maximum. For the number of LSTM units N and the learning rate lr , evenly spaced values are drawn from a logarithmic interval with a base 10 and 2 respectively. A logarithmic scale is used as it better highlights the order of magnitude where the classifier performs the best rather than identifying single values. It is then easier to make recommendations and engage in fine-tuning steps by reducing the search space around the best order. Table 5-2 shows the resulting study space.

Table 5-2 : Study space

Hyperparameter		Symbol	Study space
Architecture	Number of LSTM units	N	[32, 64, 128, 256, 512]
	Number of layer	L	[1, 2, 3]
Batch Normalization		bn	[with, without]
Bidirectional Layer		bl	[with, without]
Dropout option		do	[None, Naive, Variational]
Learning rate		lr	[0.05, 0.00889, 0.00158, 0.00028, 0.00005]
Self-attention		sa	[None, Luong, Bahdanau]
Window increment		wi	[10, 50, 90, 130, 170]
Window size		ws	[52, 114, 176, 238, 300]

The algorithm 1 below describes how the impact of each hyperparameter is tested using Table 5-1 and Table 5-2. Each evaluation of the network is done three times to account for the random initialization of the LSTM network that can have a negative effect on its overall performance. This is done with the loop controlled by k . At the end of algorithm 1, a table is saved with the performance of the m values or options tested over the 60 random configurations.

Algorithm 1 Hyperparameter Impact Testing

Select H , the hyperparameter for which the impact is tested.

Set $\mathbf{u}$, the study vector of length m , equal to H 's study space defined by the Table 5-2.

for $i = 1$ to 60 do

 Set $\mathbf{s}$, the search vector, by randomly choosing a value or option from Table 5-1 for every hyperparameter except H . Special cases:

 if $H = \text{"Window increment"}$ do

 Change the lower limit of the interval for "Window size" from Table 5-1 to the maximum value from $\mathbf{u}$.

 end if

 if $H = \text{"Window size"}$ do

 Change the upper limit of the interval for "Window increment" from Table 5-1 to the minimum value from $\mathbf{u}$.

 end if

 for $j = 1$ to m do

 for $k = 1$ to 3 do

 Build the LSTM model based on $\mathbf{s}$ and $\mathbf{u}_j$.

 Fit the model on the training set.

$\mathbf{b}_k \leftarrow$ Evaluate the model balanced accuracy on the test set.

 end for

$\mathbf{a}_j \leftarrow$ Save the best value between $\mathbf{b}_1$, $\mathbf{b}_2$ and $\mathbf{b}_3$.

 end for

 Save every $\mathbf{a}_j$

end for

A Shapiro-Wilk test is then performed to assess if the results have a normal distribution. As it is not the case ($p < 0.01$) for any hyperparameter results, the non-parametric one-tail Wilcoxon signed-rank test is performed to detect significant differences ($p < 0.01$) among the values or options of H . The test is carried out over the 60 recorded performances.

Next, the impact of H is defined using the following scale: *low* if its values or options show no statistically significant difference; *medium* if some values or options show no significant difference

while others should be avoided; *high* if a value or option is significantly better than the others. Using this scale, *low* impact hyperparameter should not be optimized. *Medium* impact hyperparameter should be optimized until the performance starts to plateau, without searching further. *High* impact hyperparameter should be fine-tuned to find the best value or otherwise, the LSTM network might not reach its full potential. Specific guidelines can then be made for each hyperparameter while also providing a recommended configuration R based on the impact study.

5.3.4 Hyperparameter optimal set for the benchmark task

While not completely random (one hyperparameter out of nine was fixed to discrete values for each test), the previous hyperparameter impact study resembles a random search. Counting the 60 random configurations that are tested on every value or option, a total of 2400 performances were recorded. Just like in a random search [114], it would be possible to use these 2400 performances to identify an optimal set of hyperparameters for solving the benchmark task using a LSTM network.

When optimizing the hyperparameters of a network, the search space is defined by the number of hyperparameters and the possible values or options for each. A point within this search space is the global maximum and an optimal subspace of 1% around this point can be considered. In other words, each random tested configuration has a 1% chance to be drawn from this optimal subspace. Using a binomial distribution, it is possible to show that the probability that the 10 best configurations among the 2400 tested are drawn from this optimal subspace is superior to 99.9%.

Once selected, these 10 best configurations are tested on three TH amputees. They are compared to the previous work baseline B that used default hyperparameter values [153] and to the recommended configuration R from the impact study. To allow a fair comparison between the configurations, each set is tested 10 times to account for the random initialization of the LSTM network which can influence the classification accuracy. Each subject's best performance is then recorded. It must be noted that the baseline performance differs from the previous article [153] because the network was trained using a batch size of 256 instead of 32 as in this article. The prediction time of all 12 configurations is also measured to observe the impact of the configuration on the real-time performance. The computer used for this calculation is a Intel i7-7820X CPU @ 3.60GHz with 48 GB of RAM at 2400 MHz and a NVIDIA GeForce GTX 1050 Ti GPU with 4 GB of RAM.

5.4 Results

The results from the impact study for every hyperparameter are in appendix B. A graph and a table summarize the performance of each hyperparameter value or option and highlights the differences that may exist. The following section presents the findings from the impact study and the 10 best configurations gathered through the random search.

5.4.1 Guidelines and recommendation

The following table summarizes the findings from the impact determination of each hyperparameter. The impact is defined as low, medium or high based on the definition given in section 5.3.3. The impacts for window increment and window size were altered afterward for reasons explained in the discussion sections 5.5.8 and 5.5.9. Guidelines are formulated based on the results from the impact study in appendix A and further explained in sections 5.5.1 to 5.5.9. The recommended configuration is composed of the best value or option for each hyperparameter.

Table 5-3 : Impact of each studied hyperparameter with their according guidelines and recommended configuration.

Hyperparameter	Impact	Guideline	Recommended Configuration
Architecture (N and L)	Medium	- The number of units should be kept under 128 units. - The number of layers should be kept under 2.*	N : 32 units L : 1 layer
Batch Normalization (bn)	High	- Batch normalization should be added to the network.	with Batch Normalization
Bidirectional Layer (bl)	Low	- Bidirectional layers should not be used in the network as the performance does not change significantly.	without a Bidirectional Layer
Dropout option (do)	Medium	- Adding variational dropout should be avoided. - Naive dropout can be added without the performance changing significantly.	Naive
Learning rate (lr)	High	- The learning rate should be set to a value with an order of magnitude around 0.00158 to begin with. - A fine-tuning step should be performed once the rest of the configuration is settled.	0.00158

Self-Attention Layer (<i>sa</i>)	Medium	- A self-attention layer should be added using either implementation: [165] or [162].	Luong
Window Increment (<i>wi</i>)	Medium	- The window increment should be around 10 ms or the lowest possible.*	10 ms
Window Size (<i>ws</i>)	Low	- The window size should be kept over 52 ms. - The performance does not change significantly for window size between 114 and 300 ms.	176 ms

*This study results do not highlight a clear relationship between this hyperparameter and the classification accuracy on the benchmark test. Therefore, this guideline is not as definitive as for other studied hyperparameters.

5.4.2 10 best configurations

The following tables present the configuration and the performance of the baseline, the recommended configuration, and the 10 configurations found in the random search. They are used to identify an optimal hyperparameter set, but also to confirm the impact, the guideline, and the recommended value or option for each hyperparameter in section 5.5.

Table 5-4 : Hyperparameters of the baseline (B), the recommended configuration (R) and the 10 best configurations found. (“w/”: with the option. “w/o”: without the option. “*Bahd.*”: Bahdanau self-attention.)

	B	R	1	2	3	4	5	6	7	8	9	10
<i>N</i>	50	32	30	256	24	32	256	84	256	30	38	144
<i>L</i>	1	1	1	2	2	1	3	3	3	1	1	2
<i>bn</i>	w/o	w/	w/	w/	w/	w/	w/	w/	w/	w/	w/	w/
<i>bl</i>	w/o	w/o	w/o	w/o	w/o	w/o	w/o	w/o	w/o	w/	w/o	w/o
<i>do</i>	None	Naive	Naive	Naive	Naive	Naive	Naive	None	Naive	Naive	None	None
<i>lr</i> (E-2)	1.0	0.158	1.592	0.087	0.540	0.971	0.244	0.104	0.229	1.592	1.084	0.889
<i>sa</i>	None	Luong	Bahd.	Luong	Bahd.	Luong	Luong	Luong	Luong	Bahd.	Bahd.	Luong
<i>wi</i> (ms)	50	10	28	74	28	172	257	130	38	28	170	145
<i>ws</i> (ms)	250	176	278	254	300	237	282	279	264	278	225	285

Matches R

Matches guidelines

Table 5-4 displays the configuration of the baseline, the recommended configuration, and the 10 best configurations found among the 2400 tested in this paper. All the colored boxes, light or dark gray, are the hyperparameters that follow the study guidelines (Table 5-3). The dark gray boxes also match the recommended configuration. Among the 10 best configurations, 74.4 % of the

hyperparameter follows the guidelines (67 out of 90). The architecture, the learning rate, and the window increment are the only hyperparameters that do not follow the guidelines with respectively five, seven, and six deviating configurations. As for the baseline, the architectures, the bidirectional layer, the dropout, and the window size value or option follow the guidelines. The window increment is not far either with 50 ms compared to the recommended 10 ms.

Table 5-5 : Performance of the baseline (B), the recommended configuration (R) and the 10 best configurations found for every subject (S). The best accuracy (Acc.) for each subject is in bold.

	B	R	1	2	3	4	5	6	7	8	9	10
Acc. S1 (%)	92.80	96.10	96.31	95.51	96.00	95.00	94.49	95.28	95.51	96.37	95.45	95.98
Acc. S2 (%)	87.76	93.75	95.63	95.28	97.13	93.08	93.68	97.10	94.21	94.30	90.25	96.74
Acc. S3 (%)	77.57	83.39	83.10	85.60	86.47	86.84	85.33	86.65	85.61	84.50	81.78	84.41
Mean Acc.(%)	86.04	91.60	91.68	92.13	93.20	91.64	91.17	93.01	91.78	91.72	89.16	92.38
Prediction time (ms)	6.7	7.2	8.1	19.0	10.8	7.7	27.6	13.6	26.3	9.1	7.4	13.2

Table 5-5 first shows the accuracies obtained for all three subjects after training the baseline, the recommended configuration, and the 10 best configurations. The best configuration for subject 1 is configuration 8. Except for the learning rate, this configuration matches the study guidelines. It is worth to note that the configuration 1, which has the same hyperparameters except for the bidirectional layer, is the next best set for subject 1 with a slight difference of 0.06% in accuracy showing again the low impact of the bidirectional layer. Otherwise, configuration 3 and 4 are the best hyperparameter set for subjects 2 and 3 respectively. Configuration 3 has the highest mean of 93.20 % across all subjects. As for the prediction time, the shortest is 6.7 ms for the baseline, which is more than four times faster than the longest time of 27.6 ms for configuration 5.

5.5 Discussion

In the following section, the main contribution of this paper is explained. Then, the results for each hyperparameter are discussed based on the impact study and the 10 best configurations. The guidelines from Table 5-3 are explained along with the recommended value or options. Finally, an optimal set is identified based on maximizing the accuracy and minimizing the prediction time with a LSTM network.

5.5.1 Main contribution

The main contribution of this study is the guidelines contained in Table 5-3. They can be used to optimize the hyperparameters of a LSTM network without wasting important computational resources. The recommended configuration R is meant to be used as a starting point for the optimization. The guidelines set boundaries to the search space to further decrease its size. The impacts serve as an indicator of how much resources should be invested to optimize this hyperparameter. Using this information, the computing time and power necessary to achieve the optimization can be decrease. Furthermore, the indications from Table 5-3 are task-specific and only apply for gesture recognition using sEMG. Another impact study using a LSTM network reported different results than this study as they investigated a different task, i.e. linguistic sequence tagging using sentences [112].

5.5.2 Architecture

As shown in Tableau B.2, the impact study reveals that less complex architectures, i.e. with fewer units and layers, perform the best to solve the benchmark task. There is no significant difference in terms of accuracy between architecture with less than 128 units and two or fewer layers. It is important to find a suitable architecture that fits with the task because the accuracy can decrease by over 30%. Although, there is no need to finely tune this hyperparameter, hence the medium impact in Table 5-3.

On the other hand, five out of the 10 best configurations did not agree with the impact study. Those five architectures used either two layers with 256 units or more, or three layers. While looking at 10 configurations might not allow to draw definitive conclusions as using the impact study, it points out that the relation between the architecture and the performance is not as clear as for other hyperparameters. Bigger architectures are indeed able to achieve great accuracy. Thus, the impact might sit between low and medium.

As the benchmark task is the classification of gestures in amputees, a sub-objective along with maximizing the accuracy is to minimize the prediction time for a real-time application. The architecture should then be the least complex possible as it decreases the prediction time. For this reason, the recommended architecture is one layer with 32 units and it even scored the best accuracy

in the impact study. While not definitive, the proposed guideline follows the results impact study and suggest only upper bounds to encourage small architectures.

Further testing could bring more insights on the relation between the architecture and the performance on the benchmark task. These tests could include more configurations evaluated along with different training parameters, such as a longer patience in the early stopping to allow more complex architectures more time to train.

5.5.3 Batch normalization

The impact study shows that using a layer of batch normalization significantly increases the accuracy by 6.47 % over not using it. Furthermore, as seen in Figure B.7-1, the distribution of accuracy for the networks with batch normalization varies a lot less than those without. Indeed, batch normalization helps to achieve a higher accuracy more consistently across random configurations. This is due to its ability to stabilize the learning by mitigating the internal covariate and to serve as a regularization tool to counter overfitting [62], [157]. The 10 best configurations from Table 5-4 all include batch normalization in their hyperparameter further emphasizing its contribution to the accuracy of the LSTM network.

These results highlight a clear relation between using a batch normalization layer and the accuracy, thus the high impact in Table 5-3. The recommended configuration and the guideline both suggest adding batch normalization to the network to achieve greater performance.

5.5.4 Bidirectional layer

As seen in Figure B.7-2 and Tableau B.4, the bidirectional layers have little to no effect on the accuracy. With a difference of 1.32 % between using it or not using, there is no significant difference detected and its impact is set to low. Gesture recognition does not seem to benefit from reading the sEMG signal backward. Furthermore, using bidirectional layers doubles the number of layers (one forward, one backward) which increases the number of weights to memorize and the time to get a prediction.

Therefore, the recommended configuration and the guideline both suggest not to use bidirectional layers in the network as its impact is insignificant.

5.5.5 Dropout option

As seen in Tableau B.5 and Figure B7-3, variational dropout shows a decrease of more than 3 % in accuracy when compared to using naive dropout or none. The variational dropout shows a significant difference with naive dropout indicating it does not perform better on our benchmark task as it was intended [161]. As for the naive dropout, it is not significantly better than not using any dropout even though there is a difference of 1.17 % between both. In some configurations, the tested networks might have been prone to overfitting and the naive dropout helped to prevent it [62]. In others, naive dropout might have been redundant given the regularization effect of the batch normalization. Both techniques do not operate in the same way and might interfere positively or negatively with one another [145]. In Table 5-4, seven configurations out of the 10 best use naive dropout even though batch normalization is involved.

The impact is set to medium as variational dropout should be avoided, but naive dropout could be beneficial for situations where overfitting occurs. As it is generally the case, the recommended configuration suggests using naive dropout. The dropout rate d was not part of the studied hyperparameter in this article and was set randomly along with the other hyperparameters. Now that naive dropout has been identified as beneficial for the LSTM network, future work should investigate the impact of the rate and if a range of values works better than the rest.

5.5.6 Learning rate

As shown in Tableau B.6, the results from the impact study indicate that a learning rate with an order of magnitude around 0.00158 performed significantly better than other values. The difference in terms of accuracy ranged from 2.01 % to 36.56 % which is important. The learning has a high impact on the training of the network and should be fine-tuned.

On other hand, Table 5-4 shows that seven out of the 10 best configurations use a learning rate different from 0.00158, ranging from 0.000877 to 0.01520. These results do not agree with the impact study in Tableau B.6 and reveal a problem when trying to identify a general learning rate that will fit any LSTM network. The learning rate that maximizes the accuracy is specific to the hyperparameter set of the network [62]. There exists such value, but it differs from one configuration to another. This fact is further emphasized by looking at Figure B.7-4. The tested learning rate distributions spread over a large range of accuracy. For example, the learning rate of

0.00158 has the best median and possesses a second low peak between 50 % and 70 %. This indicates that a certain rate might work well with certain configurations and not at all with others. Perhaps, 0.00158 should rather be considered as a good starting value, seeing it performed well the most often.

Therefore, the recommended configuration and the guideline suggest 0.00158 as a starting value. The guideline further states that the learning rate should be fine-tuned once the rest of the hyperparameter set is settled. This way, it is possible to find the learning rate that achieves the best accuracy with this specific LSTM network.

5.5.7 Self-Attention layer

The impact study in Tableau B.7 reveals that not using any self-attention layer is significantly worse by 2.01 % than using Luong self-attention. Gesture recognition can indeed benefit from the attention mechanism. The LSTM network is able to use the context vector to focus on the most task-related information in the sEMG signal. There is no significant difference between which attention form is used and the difference in accuracy is only 0.83 %. The 10 best configurations in Table 5-4 all use one form, mostly Luong, and Figure B.7-5 further shows that using a self-attention layer allows to more consistently achieve good accuracies as their distributions are narrower around 90 %.

The impact is set to medium as there is a difference between using or not using a self-attention layer, but that difference is rather small. The guideline proposes to use either [165] or [162] self-attention layer. Luong form was chosen for the recommended configuration as it was the best in the impact study, and it does not require to learn any additional parameters as it is the case with Bahdanau ($\mathbf{v}_a$ and $\mathbf{W}_a$ in equation 1).

5.5.8 Window increment

The impact study shows that the lower is the window increment, the better is the accuracy. This behavior is understandable as a low window increment leads to a higher number of examples for a given sEMG signal. With more examples, a more complete representation of the phenomenon is provided to the network which might then reach higher accuracy. Also, small window increments did not seem to push the network towards overfitting as the test set accuracy was still able to reach over 90 %.

As seen in Tableau B.8, a window increment of 10 ms is significantly better than other increments and the difference in accuracy ranges between 1.72 % and 13.86 %. On the other hand, the distributions in Figure B.7-6 are widespread indicating that the window increment alone does not necessarily lead to good accuracy. This is further demonstrated in Table 5-4 where six out of the 10 best configurations have a window increment between 74 and 257 ms. Like for the architecture, this points out that the relation between the window increment and the performance is not as clear as with other hyperparameters.

While the impact study would suggest the impact to be high, the distributions and the 10 best configurations show that the window increment might not have an important effect on the performance. Thus, the impact is set to medium as it is believed that no fine-tuning step is necessary. The guideline proposes instead to aim for the smallest value possible, with 10 ms as the lower limit. When using the LSTM network on an embedded device, e.g. the controller of a prosthesis, the window increment will be further limited by the processing capacity of the device.

5.5.9 Window size

As seen in Tableau B.9 and Figure B.7-7, the window size has little to no effect on accuracy. If the size is not too small, i.e. under 52 ms in the impact study, there is no significant difference between the tested sizes. All the 10 best configurations use a window size over 225 ms. These results indicate that the LSTM network does not necessarily benefit from bigger time windows. There were also no long-term dependencies issues [62] for the tested sizes between 52 and 300 ms as the performances are about the same.

Therefore, the guideline suggests keeping the window size above 52 ms and the recommended size is 176 ms as it performed the best in Tableau B.9. While there is a value to avoid, the impact is set to low as the difference between all the other values is not significant.

5.5.10 Best configuration for the benchmark task

As seen in Table 5-5, the recommended configuration and the 10 best configurations all surpass the baseline in terms of accuracy. None of the subjects share the same best configuration suggesting it might be better to personalize the network, i.e. to optimize the hyperparameters, for each one. For gesture recognition with amputee subjects, opting for personalized solutions makes sense as each individual produces different sEMG patterns based on various physiological factors [24].

Based on Table 5-5, configuration 8 should be used for subject 1, configuration 3 for subject 2, and configuration 4 for subject 3.

In the case where one configuration should be selected to fit the three subjects of this study, the configuration 3 (see hyperparameters values and options in Table 5-4) has the highest accuracy mean and each subject accuracy is within 0.37 % from their best performance (identified in bold in Table 5-5). Except for the learning rate, configuration 3 follows all the guidelines and achieves a prediction in 10,8 ms, about 20.5 % slower than the fastest prediction time of 6.7 ms obtained with the baseline configuration. It increases the mean of every subject accuracy by 7.16 % over the Baseline configuration previously tested in [153]. Furthermore, all three subjects are above an accuracy of 85 %, which indicates the LSTM network is accurate enough to start testing the system in real time. Indeed, offline accuracies roughly below this threshold have been reported to lead to a frustrating use of the prosthesis in real time [24].

5.6 Conclusion

To save important time and computational resources, the hyperparameter optimization should focus on those that have an important impact on the performance and avoid those that are insignificant. The objective of this study was to evaluate the impact of eight hyperparameters on the performance of a Long Short-Term Memory (LSTM) network solving a gesture recognition task. To do so, every hyperparameter value or option was tested on 60 random network configurations. The impact study provided guidelines to maximize classification accuracy and proposed a recommended configuration. These indications from Table 5-3 are the main contribution of this paper and should help to optimize the hyperparameters more efficiently by providing a starting point, a narrower search space and how much resources should be invested for each hyperparameter. Using the recommended LSTM network set, the classification accuracy was improved by 5.56 % over the baseline performance. Additionally, this study was able to identify an optimal set of hyperparameters for its benchmark task, i.e. the control of a transhumeral myoelectric prosthesis. While each amputee subject had its own best hyperparameter set, a configuration was suggested based on maximizing the classification accuracy across all subjects and minimizing the prediction time. This optimal set increased the performance of the LSTM network by 7.16% over the baseline set performance [153].

As all three subject accuracies are above 85 % with the optimized hyperparameter set, future work will focus on testing the LSTM network classifier in real time with a transhumeral amputee. By validating its online potential for prosthesis control, the control system could then be implemented in a device and used by an amputee to evaluate its capacity to accomplish everyday tasks.

5.7 Acknowledgement

This research was supported by the Fonds de recherche du Québec - Nature et Technologies (FRQNT) and the Canada First Research Excellence Fund. The authors would like to thank the patients who participated in this study along with the team from Institut de Réadaptation Gingras-Lindsay de Montréal (IRGLM) who helped with the recruitment of participants.

CHAPITRE 6 : DISCUSSION GÉNÉRALE

6.1 Méthode RNN-LSTM optimisée

À travers la revue de littérature du chapitre 2, l'utilisation du DL a été identifiée comme alternative aux méthodes ML classiques pour améliorer la précision de classification d'intentions de mouvement chez un amputé TH a été proposée. L'article 1 du chapitre 4 présente une première implémentation de la méthode RNN-LSTM sans optimisation de ses hyperparamètres. Au moyen d'une base de données sEMG provenant de 5 amputés TH, la méthode est comparée aux deux meilleures méthodes de la littérature utilisant respectivement un LDA [3] et un NN [4]. La méthode RNN-LSTM surpasse ces deux méthodes ML classiques en obtenant une précision de classification significativement ($p < 0.001$) plus élevée (RNN-LSTM : 79.7%. LDA : 67.1%. NN : 74.1%) et un temps de prédiction ~54 fois plus faible. Sans cette supériorité, il n'y aurait aucun avantage à utiliser la méthode RNN-LSTM qui est plus compliquée à implémenter et à utiliser. Toutefois, la précision de classification du RNN-LSTM sans optimisation n'atteint pas l'objectif 85% de précision de classification pour être considérée viable. En effet, une précision de classification sous 85% résulte en une prothèse qui est frustrante à utiliser selon les amputés [3], [7].

Pour augmenter la précision de classification de la méthode RNN-LSTM, une optimisation de ses hyperparamètres est effectuée dans l'article 2 du chapitre 5. Au moyen de l'étude sur l'impact des hyperparamètres de ce chapitre, la méthode RNN-LSTM est optimisée en minimisant les ressources informatiques et le temps nécessaire à ce processus qui peut s'étirer sur plusieurs semaines. À l'issue de cette optimisation, la configuration d'hyperparamètres de la table 6.1 est proposée pour maximiser la performance de la méthode RNN-LSTM. Une précision de classification moyenne de 93.20 % est obtenue pour 3 amputés TH validant ainsi que cette solution est viable ($> 85\%$) pour des fins de contrôle d'une prothèse myoélectrique. Bien que le temps de prédiction passe de 6.7 ms à 10.8 ms, cette augmentation reste négligeable par rapport à l'objectif de temps de prédiction (< 250 ms). Ce temps de prédiction changera une fois que le modèle du RNN-LSTM sera implémenté sur un dispositif embarqué. Il est difficile de prévoir l'ordre de grandeur de ce changement puisque celui-ci dépendra entre autres du processeur du dispositif, du langage de code utilisé et de la parallélisation de certains processus [153]. En prenant compte qu'il existe déjà des stratégies d'implémentation de RNN-LSTM sur microcontrôleur qui permettent un

temps de prédiction de 16 ms [141], nous croyons que notre approche de classification pourra satisfaire l'objectif d'un temps de prédiction.

Tableau 6-1 : Description de la méthode RNN-LSTM optimisée.

Pré-traitement des sEMG	Filtre	<i>Butterworth</i> passe-bande d'ordre 4. 30-300 Hz.
	Filtre	<i>Notch</i> avec un facteur de qualité de 35. 60 Hz.
	Mise à l'échelle	MinMaxScaler
RNN-LSTM (Hyperparamètres)	<i>Window size</i>	300 ms
	<i>Window increment</i>	28 ms
	<i>Learning rate</i>	0.5401×10^{-2}
	<i>Optimizer</i>	ADAM
	<i>Dropout</i>	Naive, c.-à-d. sur la sortie de chaque couche.
	Architecture du réseau	Couche 1 : Entrée des données Couche 2 : 24 nœuds RNN-LSTM Couche 3 : 24 nœuds RNN-LSTM Couche 4 : Bahdanau's <i>self-attention</i> Couche 5 : <i>Batch Normalization</i> Couche 6 : 7 nœuds dense avec une activation <i>softmax</i> .

L'utilisation de la méthode RNN-LSTM optimisée du tableau 6-1 permettra une classification des intentions de mouvements en temps réel plus précise que les méthodes ML classiques présentement utilisées dans la littérature. Une fois validée en temps réel avec des sujets amputés, la combinaison de cette méthode de classification avec la prothèse biofidèle développée conjointement à la chaire de recherche RECAP et au laboratoire CoSIM [27] pourra être effectuée. Ce dispositif de nouvelle

génération répondrait alors au besoin des amputés pour une prothèse myoélectrique au contrôle naturel et devrait augmenter le taux d'adhésion à cette solution.

6.2 Contributions de recherche

Les contributions de cette recherche ont permis l'atteinte de l'objectif principal de recherche et de ses objectifs spécifiques du Tableau 6-2. Elles ont été publiées (ou soumises pour publication) dans des journaux révisés par des pairs.

Tableau 6-2 : Objectifs spécifiques de recherche couverts dans cette thèse

	Implémenter une méthode RNN-LSTM capable de reconnaître les intentions de mouvements d'un amputé TH.
OS1	<i>Une précision de classification d'au moins 85% et un temps de prédiction en dessous de 250 ms devrait être atteint.</i>
	<i>Les six intentions de mouvement étudiés sont la flexion et l'extension du coude, la pronation et supination de l'avant-bras et l'ouverture et la fermeture de la main.</i>
OS2	Identifier les hyperparamètres à optimiser pour l'obtention de la meilleure précision de classification avec la méthode RNN-LSTM.
OS3	Optimiser la précision de classification de la méthode RNN-LSTM.
	<i>Respecter l'objectif de temps de prédiction en-dessous de 250 ms.</i>

La liste de la page suivante associe chaque contribution de recherche avec l'objectif spécifique qu'il atteint suivi de la publication associée.

Contributions	Objectif spécifique	Publications associées
Nouvelles observations : une revue de littérature du développement des prothèses myoélectriques pour un amputé TH et des méthodes de classifications actuelles a été effectuée. Elle révèle notamment que :		
- Des méthodes DL ont surpassé des méthodes ML classiques dans plusieurs domaines et pourrait s'avérer être la solution pour augmenter les performances des méthodes pour fins de contrôle de prothèse myoélectrique TH. - Aucune méthode DL n'a été implémentée pour fins de contrôle d'une prothèse myoélectrique TH.	-	A
Nouvelle connaissance : la classification des intentions de mouvement au moyen de sEMG ne fonctionne pas avec tous les amputés TH. Des activations du signal sEMG créées par le mouvement sont nécessaires au bon fonctionnement de la méthode.		
	-	B
Nouvelle connaissance : l'utilisation du spectre fréquentiel (STFT) comme features additionnels n'augmente pas la précision de classification du RNN-LSTM de façon statistiquement significative.		
	-	B
Nouvelle méthode : RNN-LSTM optimisée pour classifier les 6 intentions de mouvement (EF, EE, FP, FS, HO, HC) chez 3 amputés TH.		
- Précision de classification moyenne : 93.20% (> 85% : OS1) - Temps de prédiction: 10.8 ms (< 250 ms : OS2) - Autant pour la précision de classification et le temps de prédiction, le RNN-LSTM était statistiquement supérieur aux deux meilleures méthodes de la littérature [3], [4].	OS1 OS3	B, C

Nouvelles observations : quantification de l'impact de 8 hyperparamètres sur la précision de classification d'un RNN-LSTM pour la reconnaissance de mouvement.	OS2	C
--	-----	---

Nouvelles connaissances : mise en œuvre d'une configuration de référence et de 12 recommandations pour accomplir une optimisation qui gaspille peu de ressources informatiques avec un RNN-LSTM pour la reconnaissance de mouvement.	OS2	C
--	-----	---

Publications associées :

- A. Barron, O., Raison, M., Achiche, S., 2020. Control of transhumeral prostheses based on electromyography pattern recognition: from amputees to deep learning., in: Powered Prostheses, Elsevier, pp. 1–21.
- B. Barron, O., Raison, M., Gaudet, G., Achiche, S., 2020. Recurrent Neural Network for electromyographic gesture recognition in transhumeral amputees. Appl. Soft Comput. 96, 106616.
- C. Barron, O., Raison, M., Achiche, S., 2020b. Impact of hyperparameters on the performance of a LSTM network for gesture recognition using surface electromyography. Soumis au Journal of Appl. Soft Comput. en Mars 2021.

6.3 Implémentations logicielles

L'implémentation du traitement des signaux sEMG, de la classification des intentions de mouvements et de l'optimisation des hyperparamètres ont été réalisées en Python 3.6.10. Les données sEMG et cinématiques utilisées pour l'entraînement des modèles de ML et de DL ont été acquises durant un projet de recherche précédent [4]. Cette base de données n'est pas disponible publiquement pour des raisons de confidentialités envers les participants.

6.4 Limitations

Cette thèse introduit une méthode RNN-LSTM avec un meilleur taux de succès que les méthodes offertes dans la littérature. La solution proposée est assujettie à certaines restrictions établies par notre contexte de recherche.

Une première limitation est le nombre peu élevé de participants pour tester la méthode RNN-LSTM. Bien que les données sEMG de cinq amputés ont été obtenues dans une étude précédente [4], deux de ces participants ne présentaient pas des signaux adaptés à la classification avec du ML. Ainsi, seulement trois participants ont pris part aux évaluations de la méthode. L'objectif n'est certes pas de trouver une solution qui conviendrait à tous les amputés TH, mais ce faible nombre de participants ne permet pas de dire si notre solution est généralisable à la majorité de cette population.

Une deuxième limitation est la quantité de données pour chacun des participants. Les données provenant des 3 participants ont été acquises durant une seule séance, ce qui donne un faible nombre d'exemples à fournir à une méthode DL (5000 exemples après l'utilisation d'une fenêtre de temps glissante). Les résultats obtenus au chapitre 4 montrent que la méthode DL est capable d'accomplir la classification avec cette quantité de données sans *overfitting*. Toutefois, un nombre plus élevé d'exemples écarterait cette situation avec plus de certitudes et pourrait augmenter davantage la précision de classification.

La variété des données constitue aussi une autre limitation. Outre la fatigue musculaire qui peut survenir durant la séance d'acquisition, peu de facteurs de variations ont été introduits dans les signaux sEMG. Dans un contexte où un amputé doit enlever et remettre sa prothèse chaque jour, où les électrodes peuvent bouger durant la journée et où le comportement des signaux sEMG peut changer selon le type d'activités effectuées, il n'est pas possible de déterminer si notre méthode RNN-LSTM est robuste à ces changements.

La sEMG est la modalité d'acquisition la plus populaire pour la classification de mouvements. Cette modalité possède toutefois certaines limites qui doivent être prises en considération. Un signal sEMG ne constitue pas une mesure directe du mouvement, mais plutôt une mesure de l'activité musculaire. La détection du mouvement est alors indirecte et nécessite de prendre en considération que la co-activation des muscles d'une même articulation peut complexifier le signal

sEMG résultant [170]. De plus, l'acquisition d'un signal sEMG requiert une fréquence d'échantillonnage élevée (> 1 kHz) ce qui peut occasionner des défis supplémentaires pour l'acquisition en temps réel. Finalement, ce signal possède une amplitude de l'ordre du mV rendant difficile la détection d'activations musculaires sans l'utilisation de méthodes dédiées à cet effet (p. ex. l'opérateur d'énergie Teager-Kaiser – TKEO [133]). Il pourrait être possible de développer une méthode RNN-LSTM similaire en utilisant une modalité plus facile à utiliser (p. ex. la FMG).

De plus, la revue de littérature du chapitre 2 identifie le RNN-LSTM comme la solution avec le plus de potentiel parmi les méthodes DL. Un objectif implicite de cette thèse était de déterminer si une méthode DL serait supérieure aux méthodes ML classiques déjà utilisées dans la littérature. Une seule méthode DL a donc été implémentée et optimisée exhaustivement afin d'évaluer son potentiel. Cette thèse ne se prononce pas sur quelle méthode DL est la meilleure pour la classification d'intentions de mouvement au moyen de signaux sEMG. Le populaire CNN pourrait s'avérer davantage efficace pour cette tâche.

Finalement, aucune phase de validation en temps réel n'a été effectuée. Puisque la précision de classification n'est pas entièrement garante du contrôle naturel de la prothèse, il est nécessaire de faire des essais cliniques avec des amputés TH afin de valider la fonctionnalité du système de contrôle. Autrement, bien que la méthode RNN-LSTM soit plus précise que ses autres concurrents avec des données différés, son utilisation en temps réel pourrait être frustrante pour l'utilisateur ce qui l'écarterait des solutions considérées. De plus, une implémentation temps réel de la méthode RNN-LSTM permettrait d'avoir le temps de prédiction de la prothèse plutôt que d'un ordinateur. Pour la méthode RNN-LSTM optimisée à l'issue du chapitre 5, le temps de prédiction est de 10.8 ms, bien en dessous de l'objectif de 250 ms pour le temps réel. Son implémentation sur un système embarqué fera assurément augmenter ce temps de prédiction. Le facteur de cette augmentation est difficile à établir puisque plusieurs éléments peuvent avoir un impact sur la performance temps réel, p. ex. le langage de programmation, la parallélisation de certains processus, le processeur utilisé, etc.

6.5 Travaux futurs

Dans le contexte technologique actuel, la puissance des processeurs, les infrastructures pour contenir de données et les techniques d'apprentissage machine permettent d'exploiter une grande

quantité de données pour résoudre plusieurs problématiques différentes. Puisque l'accès à des données provenant d'amputés est très restreint, il est suggéré de bâtir une banque de données qui pourrait servir à plusieurs projets de recherche entourant le développement des prothèses myoélectriques TH. Il n'existe aucune banque de données publiques actuellement dans la littérature pour cette population d'amputé. En prenant exemple sur Ninapro [14], cette banque de données pourrait être formée de façon incrémentale en ajoutant continuellement de petites acquisitions pour prendre en compte la rareté de ce type de participants. Une telle banque de données permettrait d'une part de faciliter la comparaison de méthodes de classification entre les études puisqu'elles seraient comparées avec les mêmes signaux. D'autre part, une banque de données offrirait plus de sujets à tester avec plus d'exemples par sujet et différentes variations présentes dans les signaux sEMG. Ceci permettrait alors de résoudre les trois premières limitations de la section 7.2 et de garantir que la performance obtenue avec la méthode de classification est généralisable.

Il est aussi suggéré d'implémenter une méthode CNN et de comparer sa performance avec celle du RNN-LSTM. Il existe d'ailleurs déjà plusieurs implémentations différentes dans le domaine TR [26], [93]–[98] qui pourrait servir de point de départ au modèle pour les amputés TH. Il existe aussi des implémentations CNN-RNN qui pourrait s'avérer encore plus efficace puisqu'elle exploite leurs forces respectives [106], [171]. Une investigation des différentes possibilités en DL pourrait mettre en évidence une meilleure option qu'une méthode RNN-LSTM.

Finalement, une implémentation de la méthode de classification en temps réel devrait avoir lieu pour déterminer son temps de prédiction réel et valider la fonctionnalité du système de contrôle sur une prothèse. Il est suggéré d'avoir recours à quelqu'un avec de l'expérience en programmation temps réel et en systèmes embarqués afin d'obtenir un contrôle qui satisfait le seuil de < 250 ms pour un emploi temps réel de la prothèse. De plus, pour la validation de la prothèse, le test TAC [172] ou une approche employant la loi de Fitt [129], [173], [174] est à favoriser afin de quantifier la fonctionnalité du dispositif en plus de recueillir les commentaires des amputés lors de son usage.

CHAPITRE 7 : CONCLUSION ET RECOMMANDATIONS

Depuis des décennies, des méthodes ML classiques sont utilisées pour classifier les signaux sEMG des amputés TH pour la commande d'une prothèse myoélectrique. Or, aucune solution n'a été adoptée à grande échelle d'un point de vue clinique et les dispositifs actuellement sur le marché sont mal adaptés à cette population d'amputé. Afin de satisfaire leur besoin pour une prothèse myoélectrique avec un contrôle naturel, cette thèse introduit une méthode d'apprentissage profond capable de reconnaître les intentions de mouvement d'un amputé TH avec un meilleur taux de succès que ce qu'on retrouve dans la littérature. Premièrement, la revue de littérature du chapitre 2 a été effectuée pour identifier les défis entourant le développement des prothèses myoélectriques et les méthodes actuelles de classification de mouvement. En investiguant la littérature entourant l'apprentissage machine et les pratiques employées dans le domaine des prothèses TR, le DL est identifié comme une solution pour augmenter la précision de classification et améliorer le contrôle naturel de ces dispositifs. Le RNN-LSTM est choisi comme la méthode DL pour sa capacité à utiliser des observations passées pour influencer sa décision. Cette capacité permet au RNN-LSTM de profiter de l'aspect séquentiel des données et d'être mieux adapté à travailler avec des signaux sEMG.

Le premier sous-objectif (OS1) est alors partiellement accompli au chapitre 4 en implémentant une méthode RNN-LSTM pour la classification de 6 intentions de mouvement du bras chez 5 amputés TH. Une précision de classification moyenne de 79.7 % est obtenue pour 3 des 5 participants avec un temps de prédiction de 6.7 ms. Lorsque la méthode RNN-LSTM est comparée aux deux meilleures méthodes de la littérature utilisant un ANN [4] et un LDA [3], elle est statistiquement supérieure autant pour la précision de classification que pour le temps de prédiction. Les signaux sEMG des deux participants restants ne présentaient pas d'activations musculaires significatives lors des mouvements ce qui ne permettait pas d'effectuer une classification des mouvements, autant pour le RNN-LSTM que pour le ANN et le LDA.

Le premier sous-objectif (OS1) spécifie qu'une précision de classification d'au moins 85 % devrait être obtenue pour éviter de générer de la frustration chez l'utilisateur lors de l'utilisation de la prothèse et ainsi justifier de passer à des essais cliniques. Or les résultats du chapitre 4 n'atteignent pas cet objectif. Afin de satisfaire ce critère et d'accomplir le deuxième sous-objectif (OS2), une optimisation des hyperparamètres de la méthode RNN-LSTM est effectuée au chapitre 5.

L'optimisation de 8 hyperparamètres permet alors d'augmenter la précision de classification à 93.20 % avec un temps de prédiction de 10.8 ms. La précision de classification des trois amputés TH est au-dessus de 85% tout en maintenant un faible temps de prédiction, ce qui vient compléter le premier sous-objectif (OS1).

Sans expertise préalable, l'optimisation des hyperparamètres d'une méthode DL peut s'avérer coûteuse en termes de ressources informatiques. Afin d'aider les futures équipes de recherche du domaine des prothèses myoélectriques et de satisfaire au dernier sous-objectif (OS3), le chapitre 5 présente des recommandations pour être efficient dans l'optimisation des hyperparamètres d'une méthode RNN-LSTM pour la reconnaissance de mouvement au moyen de signaux sEMG. Pour y parvenir, l'impact de 8 hyperparamètres sur la précision de classification est d'abord mesuré afin de déterminer quels hyperparamètres ont un apport significatif sur la métrique. Ensuite, une configuration de référence et 12 lignes directrices sont proposées pour réduire l'espace de recherche et converger plus rapidement vers une configuration qui optimise la précision de classification.

RÉFÉRENCES

- [1] A. Touillet, L. Peultier-Celli, C. Nicol, N. Jarrassé, I. Loiret, N. Martinet, J. Paysant, et J. B. De Graaf, « Characteristics of phantom upper limb mobility encourage phantom-mobility-based prosthesis control », *Sci. Rep.*, vol. 8, n° 1, p. 15459, oct. 2018, doi: 10.1038/s41598-018-33643-0.
- [2] N. Jarrassé, E. de Montalivet, F. Richer, C. Nicol, A. Touillet, N. Martinet, J. Paysant, et J. B. De Graaf, « Phantom-Mobility-Based Prosthesis Control in Transhumeral Amputees Without Surgical Reinnervation: A Preliminary Study », *Front. Bioeng. Biotechnol.*, vol. 6, 2018, doi: 10.3389/fbioe.2018.00164.
- [3] N. Jarrassé, C. Nicol, A. Touillet, F. Richer, N. Martinet, J. Paysant, et J. B. De Graaf, « Classification of Phantom Finger, Hand, Wrist, and Elbow Voluntary Gestures in Transhumeral Amputees With sEMG », *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 25, n° 1, p. 71-80, janv. 2017, doi: 10.1109/TNSRE.2016.2563222.
- [4] G. Gaudet, M. Raison, et S. Achiche, « Classification of Upper limb phantom movements in transhumeral amputees using electromyographic and kinematic features », *Eng. Appl. Artif. Intell.*, vol. 68, p. 153-164, févr. 2018, doi: 10.1016/j.engappai.2017.10.017.
- [5] G. Gaudet, M. Raison, F. D. Maso, S. Achiche, et M. Begon, « Intra- and Intersession Reliability of Surface Electromyography on Muscles Actuating the Forearm During Maximum Voluntary Contractions », *J. Appl. Biomech.*, vol. 32, n° 6, p. 558-570, déc. 2016, doi: 10.1123/jab.2015-0214.
- [6] L. Ballaz, M. Raison, C. Detrembleur, G. Gaudet, et M. Lemay, « Joint torque variability and repeatability during cyclic flexion-extension of the elbow », *BMC Sports Sci. Med. Rehabil.*, vol. 8, avr. 2016, doi: 10.1186/s13102-016-0033-1.
- [7] E. Scheme et K. Englehart, « Electromyogram pattern recognition for control of powered upper-limb prostheses: state of the art and challenges for clinical use », *J. Rehabil. Res. Dev.*, vol. 48, n° 6, p. 643-659, 2011.
- [8] C. L. Pulliam, J. M. Lambrecht, et R. F. Kirsch, « EMG-Based Neural Network Control of Transhumeral Prostheses », *J. Rehabil. Res. Dev.*, vol. 48, n° 6, p. 739-754, 2011.

- [9] S. I. Benchabane, N. Saadia, et A. Ramdane-Cherif, « Novel algorithm for conventional myocontrol of upper limbs prosthetics », *Biomed. Signal Process. Control*, vol. 57, p. 101791, mars 2020, doi: 10.1016/j.bspc.2019.101791.
- [10] L. Resnik, F. Acluche, M. Borgia, G. Latlief, et S. Phillips, « EMG Pattern Recognition Control of the DEKA Arm: Impact on User Ratings of Satisfaction and Usability », *IEEE J. Transl. Eng. Health Med.*, vol. 7, déc. 2018, doi: 10.1109/JTEHM.2018.2883943.
- [11] W. R. Frontera et J. K. Silver, *Fondamenti di medicina fisica e riabilitativa*. Verduci, 2004.
- [12] E. Nsugbe, C. Phillips, M. Fraser, et J. McIntosh, « Gesture recognition for transhumeral prosthesis control using EMG and NIR », *IET Cyber-Syst. Robot.*, vol. 2, n° 3, p. 122-131, 2020, doi: 10.1049/iet-csr.2020.0008.
- [13] A. Calado, F. Soares, et D. Matos, « A Review on Commercially Available Anthropomorphic Myoelectric Prosthetic Hands, Pattern-Recognition-Based Microcontrollers and sEMG Sensors used for Prosthetic Control », in *2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, avr. 2019, p. 1-6, doi: 10.1109/ICARSC.2019.8733629.
- [14] M. Atzori et H. Müller, « The Ninapro database: A resource for sEMG naturally controlled robotic hand prosthetics », in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, août 2015, p. 7151-7154, doi: 10.1109/EMBC.2015.7320041.
- [15] M. Han, S. Y. Günay, G. Schirner, T. Padır, et D. Erdoğan, « HANDS: a multimodal dataset for modeling toward human grasp intent inference in prosthetic hands », *Intell. Serv. Robot.*, vol. 13, n° 1, p. 179-185, janv. 2020, doi: 10.1007/s11370-019-00293-8.
- [16] V. H. Cene, M. Tosin, J. Machado, et A. Balbinot, « Open Database for Accurate Upper-Limb Intent Detection Using Electromyography and Reliable Extreme Learning Machines », *Sensors*, vol. 19, n° 8, avr. 2019, doi: 10.3390/s19081864.
- [17] A. Ameri, M. A. Akhaee, E. Scheme, et K. Englehart, « Real-time, simultaneous myoelectric control using a convolutional neural network », *PLOS ONE*, vol. 13, n° 9, p. e0203835, sept. 2018, doi: 10.1371/journal.pone.0203835.

- [18] P. Parker, K. Englehart, et B. Hudgins, « Myoelectric signal processing for control of powered limb prostheses », *J. Electromyogr. Kinesiol. Off. J. Int. Soc. Electrophysiol. Kinesiol.*, vol. 16, n° 6, p. 541-548, déc. 2006, doi: 10.1016/j.jelekin.2006.08.006.
- [19] A. Olsson, N. Malešević, A. Björkman, et C. Antfolk, « Exploiting the Intertemporal Structure of the Upper-Limb sEMG: Comparisons between an LSTM Network and Cross-Sectional Myoelectric Pattern Recognition Methods », in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, juill. 2019, p. 6611-6615, doi: 10.1109/EMBC.2019.8856648.
- [20] N. Parajuli, N. Sreenivasan, P. Bifulco, M. Cesarelli, S. Savino, V. Niola, D. Esposito, Tara. J. Hamilton, G. R. Naik, U. Gunawardana, et G. D. Gargiulo, « Real-Time EMG Based Pattern Recognition Control for Hand Prostheses: A Review on Existing Methods, Challenges and Future Implementation », *Sensors*, vol. 19, n° 20, oct. 2019, doi: 10.3390/s19204596.
- [21] O. W. Samuel, M. G. Asogbon, Y. Geng, A. H. Al-Timemy, S. Pirbhulal, N. Ji, S. Chen, P. Fang, et G. Li, « Intelligent EMG Pattern Recognition Control Method for Upper-Limb Multifunctional Prostheses: Advances, Current Challenges, and Future Prospects », *IEEE Access*, vol. 7, p. 10150-10165, 2019, doi: 10.1109/ACCESS.2019.2891350.
- [22] O. Faust, Y. Hagiwara, T. J. Hong, O. S. Lih, et U. R. Acharya, « Deep learning for healthcare applications based on physiological signals: A review », *Comput. Methods Programs Biomed.*, vol. 161, p. 1-13, juill. 2018, doi: 10.1016/j.cmpb.2018.04.005.
- [23] A. Phinyomark, E. Scheme, A. Phinyomark, et E. Scheme, « EMG Pattern Recognition in the Era of Big Data and Deep Learning », *Big Data Cogn. Comput.*, vol. 2, n° 3, p. 21, août 2018, doi: 10.3390/bdcc2030021.
- [24] O. Barron, M. Raison, et S. Achiche, « Control of transhumeral prostheses based on electromyography pattern recognition: from amputees to deep learning. », in *Powered Prostheses*, Elsevier., 2020, p. 1-21.
- [25] M. Sharma, H. Malhotra, N. Kumar, et J. Yadav, « Deep Learning with Convolutional Neural Network for Controlling Human Arm Prosthesis using sEMG signals », in *2019 International Conference on Computing, Power and Communication Technologies (GUCON)*, sept. 2019, p. 796-803.

- [26] M. Atzori, M. Cognolato, et H. Müller, « Deep Learning with Convolutional Neural Networks Applied to Electromyography Data: A Resource for the Classification of Movements for Prosthetic Hands », *Front. Neurorobotics*, vol. 10, sept. 2016, doi: 10.3389/fnbot.2016.00009.
- [27] M. Ramanananarivo, M. Raison, O. Barron, et S. Achiche, « Biofidelic design of the forearm of a myoelectric prosthesis with maximum functional volume », in *Proceedings of the 21st International Conference on Engineering Design (ICED 17)*, Vol 1: Resource Sensitive Design, Design Research Applications and Case Studies, Vancouver, Canada, 21-25.08.2017
- [28] N. A. Alshammary, S. A. Dalley, et M. Goldfarb, « Assessment of a multigrasp myoelectric control approach for use by transhumeral amputees », in *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, août 2012, p. 968-971, doi: 10.1109/EMBC.2012.6346094.
- [29] A. Esquenazi, « Amputation rehabilitation and prosthetic restoration. From surgery to community reintegration », *Disabil. Rehabil.*, vol. 26, n° 14-15, p. 831-836, juill. 2004, doi: 10.1080/09638280410001708850.
- [30] M. Atzori, A. Gijsberts, I. Kuzborskij, S. Elsig, A. G. M. Hager, O. Deriaz, C. Castellini, H. Müller, et B. Caputo, « Characterization of a Benchmark Database for Myoelectric Movement Classification », *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 23, n° 1, p. 73-83, janv. 2015, doi: 10.1109/TNSRE.2014.2328495.
- [31] C. Zizoua, M. Raison, S. Boukhenous, M. Attari, et S. Achiche, « Development of a Bracelet With Strain-Gauge Matrix for Movement Intention Identification in Traumatic Amputees », *IEEE Sens. J.*, vol. 17, n° 8, p. 2464-2471, avr. 2017, doi: 10.1109/JSEN.2017.2666784.
- [32] S. M. Khan, A. A. Khan, et O. Farooq, « Selection of features and classifiers for EMG-EEG-based upper limb assistive devices-a review », *IEEE Rev. Biomed. Eng.*, vol. 13, p. 248-60, 2020, doi: 10.1109/RBME.2019.2950897.
- [33] A. Mohebbi, « Human-Robot Interaction in Rehabilitation and Assistance: a Review », *Curr. Robot. Rep.*, vol. 1, n° 3, p. 131-144, sept. 2020, doi: 10.1007/s43154-020-00015-4.
- [34] S. Hernandez, M. Raison, A. Torres, G. Gaudet, et S. Achiche, « From on-body sensors to in-body data for health monitoring and medical robotics: A survey », in *2014 Global Information*

Infrastructure and Networking Symposium (GIIS), sept. 2014, p. 1-5, doi: 10.1109/GIIS.2014.6934279.

[35] A. Duivenvoorden, K. Lee, M. Raison, et S. Achiche, « Sensor fusion in upper limb area networks: A survey », in *2017 Global Information Infrastructure and Networking Symposium (GIIS)*, oct. 2017, p. 56-63, doi: 10.1109/GIIS.2017.8169802.

[36] M. Wininger, N.-H. Kim, et W. Craelius, « Pressure signature of forearm as predictor of grip force », *J. Rehabil. Res. Dev.*, vol. 45, n° 6, p. 883-892, 2008.

[37] C. Zizoua, M. Raison, S. Boukhenous, M. Attari, et S. Achiche, « Detecting muscle contractions using strain gauges », *Electron. Lett.*, vol. 52, n° 22, p. 1836-1838, 2016.

[38] E. Cho, R. Chen, L.-K. Merhi, Z. Xiao, B. Pousett, et C. Menon, « Force Myography to Control Robotic Upper Extremity Prostheses: A Feasibility Study », *Front. Bioeng. Biotechnol.*, vol. 4, mars 2016, doi: 10.3389/fbioe.2016.00018.

[39] X. Jiang, L.-K. Merhi, Z. G. Xiao, et C. Menon, « Exploration of Force Myography and surface Electromyography in hand gesture classification », *Med. Eng. Phys.*, vol. 41, p. 63-73, mars 2017, doi: 10.1016/j.medengphy.2017.01.015.

[40] A. Radmand, E. Scheme, et K. Englehart, « High-density force myography: A possible alternative for upper-limb prosthetic control », *J. Rehabil. Res. Dev.*, vol. 53, n° 4, p. 443-456, 2016, doi: 10.1682/JRRD.2015.03.0041.

[41] D. Sierra González et C. Castellini, « A realistic implementation of ultrasound imaging as a human-machine interface for upper-limb amputees », *Front. Neurorobotics*, vol. 7, oct. 2013, doi: 10.3389/fnbot.2013.00017.

[42] T. Beyrouthy, S. K. A. Kork, J. A. Korbane, et A. Abdulmonem, « EEG Mind controlled Smart Prosthetic Arm », in *2016 IEEE International Conference on Emerging Technologies and Innovative Business Practices for the Transformation of Societies (EmergiTech)*, août 2016, p. 404-409, doi: 10.1109/EmergiTech.2016.7737375.

[43] R. N. Scott et P. A. Parker, « Myoelectric prostheses: state of the art », *J. Med. Eng. Technol.*, vol. 12, n° 4, p. 143-151, août 1988.

- [44] H. H. Sears et J. Shaperman, « Proportional myoelectric hand control: an evaluation », *Am. J. Phys. Med. Rehabil.*, vol. 70, n° 1, p. 20-28, févr. 1991.
- [45] A. Fougner, O. Stavdahl, P. J. Kyberd, Y. G. Losier, et P. A. Parker, « Control of upper limb prostheses: terminology and proportional myoelectric control-a review », *IEEE Trans. Neural Syst. Rehabil. Eng. Publ. IEEE Eng. Med. Biol. Soc.*, vol. 20, n° 5, p. 663-677, sept. 2012, doi: 10.1109/TNSRE.2012.2196711.
- [46] J. Wen, M. Raison, et S. Achiche, « Using a cost function based on kinematics and electromyographic data to quantify muscle forces », *J. Biomech.*, vol. 80, p. 151-158, oct. 2018, doi: 10.1016/j.jbiomech.2018.09.002.
- [47] M. Raison, « On the quantification of joint and muscle efforts in the human body during motion », PhD thesis, Universite catholique de Louvain (UCL), 2009.
- [48] G. Drost, D. F. Stegeman, B. G. M. van Engelen, et M. J. Zwarts, « Clinical applications of high-density surface EMG: A systematic review », *J. Electromyogr. Kinesiol.*, vol. 16, n° 6, p. 586-602, déc. 2006, doi: 10.1016/j.jelekin.2006.09.005.
- [49] R. Díaz-Amador, C. A. Ferrer-Riesgo, et J. V. Lorenzo-Ginori, « Using Image Processing Techniques and HD-EMG for Upper Limb Prosthesis Gesture Recognition », in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, vol. 11401, R. Vera-Rodriguez, J. Fierrez, et A. Morales, Éd. Cham: Springer International Publishing, 2019, p. 913-921.
- [50] L. G. Cohen, S. Bandinelli, T. W. Findley, et M. Hallett, « Motor reorganization after upper limb amputation in man. A study with focal magnetic stimulation », *Brain J. Neurol.*, vol. 114 (Pt 1B), p. 615-627, févr. 1991.
- [51] V. S. Ramachandran et W. Hirstein, « The perception of phantom limbs. The D. O. Hebb lecture », *Brain J. Neurol.*, vol. 121 (Pt 9), p. 1603-1630, sept. 1998.
- [52] C. M. Kooijman, P. U. Dijkstra, J. H. Geertzen, A. Elzinga, et C. P. van der Schans, « Phantom pain and phantom sensations in upper limb amputees: an epidemiological study », *Pain*, vol. 87, n° 1, p. 33-41, juill. 2000.

- [53] K. T. Reilly, C. Mercier, M. H. Schieber, et A. Sirigu, « Persistent hand motor commands in the amputees' brain », *Brain J. Neurol.*, vol. 129, n° Pt 8, p. 2211-2223, août 2006, doi: 10.1093/brain/awl154.
- [54] M. G. Urbanchek, Z. Baghmanli, J. D. Moon, K. B. Sugg, N. B. Langhals, et P. S. Cederna, « Quantification of Regenerative Peripheral Nerve Interface Signal Transmission », *Plast. Reconstr. Surg.*, vol. 130, n° 5S-1, p. 55, nov. 2012, doi: 10.1097/01.prs.0000421762.53265.54.
- [55] C. A. Chestek, V. Gilja, P. Nuyujukian, J. D. Foster, J. M. Fan, M. T. Kaufman, M. M. Churuchland, Z. Rivera-Alvidrez, J. P. Cunningham, Stephen I. Ryu, et K. V. Shenoy, « Long-term stability of neural prosthetic control signals from silicon cortical arrays in rhesus macaque motor cortex », *J. Neural Eng.*, vol. 8, n° 4, p. 045005, août 2011, doi: 10.1088/1741-2560/8/4/045005.
- [56] K. D. O'Shaughnessy, G. A. Dumanian, R. D. Lipschutz, L. A. Miller, K. Stubblefield, et T. A. Kuiken, « Targeted reinnervation to improve prosthesis control in transhumeral amputees. A report of three cases », *J. Bone Joint Surg. Am.*, vol. 90, n° 2, p. 393-400, févr. 2008, doi: 10.2106/JBJS.G.00268.
- [57] J. E. Cheesborough, L. H. Smith, T. A. Kuiken, et G. A. Dumanian, « Targeted Muscle Reinnervation and Advanced Prosthetic Arms », *Semin. Plast. Surg.*, vol. 29, n° 1, p. 62-72, févr. 2015, doi: 10.1055/s-0035-1544166.
- [58] M. S. Johannes, J. D. Bigelow, J. M. Burck, S. D. Harshbarger, et T. Van Doren, « An overview of the developmental process for the modular prosthetic limb. », *John Hopkins APL Tech. Dig.*, vol. 30, n° 3, p. 207-216, 2011.
- [59] L. J. Hargrove, L. A. Miller, K. Turner, et T. A. Kuiken, « Myoelectric Pattern Recognition Outperforms Direct Control for Transhumeral Amputees with Targeted Muscle Reinnervation: A Randomized Clinical Trial », *Sci. Rep.*, vol. 7, n° 1, Art. n° 1, oct. 2017, doi: 10.1038/s41598-017-14386-w.
- [60] T. A. Kuiken, L. A. Miller, R. D. Lipschutz, B. A. Lock, K. Stubblefield, P.D. Marasco, P. Zhou, et G. A. Dumanian, « Targeted reinnervation for enhanced prosthetic arm function in a woman with a proximal amputation: a case study », *Lancet Lond. Engl.*, vol. 369, n° 9559, p. 371-380, févr. 2007, doi: 10.1016/S0140-6736(07)60193-7.

- [61] P. A. Parker, K. B. Englehart, et B. S. Hudgins, « Control of Powered Upper Limb Prostheses », in *Electromyography: Physiology, Engineering, and Non-invasive Applications*, John Wiley&Sons., Hoboken, New Jersey: R. Merletti et P. Parker, 2004, p. 453-475.
- [62] I. Goodfellow, Y. Bengio, et A. Courville, *Deep learning*. Cambridge, Massachusetts: The MIT Press, 2016.
- [63] S. Raouafi, S. Achiche, M. Begon, A. Sarcher, et M. Raison, « Classification of upper limb disability levels of children with spastic unilateral cerebral palsy using K-means algorithm », *Med. Biol. Eng. Comput.*, vol. 56, n° 1, p. 49-59, janv. 2018, doi: 10.1007/s11517-017-1678-y.
- [64] A. Sarcher, M. Raison, F. Leboeuf, B. Perrouin-Verbe, S. Brochard, et R. Gross, « Pathological and physiological muscle co-activation during active elbow extension in children with unilateral cerebral palsy », *Clin. Neurophysiol. Off. J. Int. Fed. Clin. Neurophysiol.*, vol. 128, n° 1, p. 4-13, janv. 2017, doi: 10.1016/j.clinph.2016.10.086.
- [65] A. Sarcher, M. Raison, L. Ballaz, M. Lemay, F. Leboeuf, K. Trudel, et P. A. Mathieu, « Impact of muscle activation on ranges of motion during active elbow movement in children with spastic hemiplegic cerebral palsy », *Clin. Biomech. Bristol Avon*, vol. 30, n° 1, p. 86-94, janv. 2015, doi: 10.1016/j.clinbiomech.2014.10.009.
- [66] M. Laitenberger, M. Raison, D. Périé, et M. Begon, « Refinement of the upper limb joint kinematics and dynamics using a subject-specific closed-loop forearm model », *Multibody Syst. Dyn.*, vol. 33, n° 4, p. 413-438, avr. 2015, doi: 10.1007/s11044-014-9421-z.
- [67] M. Hosin, et M.N. Sulaiman, « A Review on Evaluation Metrics for Data Classification Evaluations », *Int. J. Data Min. Knowl. Manag. Process*, vol. 5, n° 2, p. 01-11, mars 2015, doi: 10.5121/ijdkp.2015.5201.
- [68] T. R. Farrell et R. F. Weir, « The Optimal Controller Delay for Myoelectric Prostheses », *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 15, n° 1, p. 111-118, mars 2007, doi: 10.1109/TNSRE.2007.891391.
- [69] L. H. Smith, L. J. Hargrove, B. A. Lock, et T. A. Kuiken, « Determining the Optimal Window Length for Pattern Recognition-Based Myoelectric Control: Balancing the Competing Effects of Classification Error and Controller Delay », *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 19, n° 2, p. 186-192, avr. 2011, doi: 10.1109/TNSRE.2010.2100828.

- [70] A. D. Vigotsky, I. Halperin, G. J. Lehman, G. S. Trajano, et T. M. Vieira, « Interpreting Signal Amplitudes in Surface Electromyography Studies in Sport and Rehabilitation Sciences », *Front. Physiol.*, vol. 8, janv. 2018, doi: 10.3389/fphys.2017.00985.
- [71] H. J. Hermens, B. Freriks, R. Merletti, D. Stegeman, J. Blok, G. Rau, C. Disselhorst-Klug, et G. Hägg, *European Recommendations for Surface Electromyography: Results of the SENIAM Project*. Roessingh Research and Development, 1999.
- [72] N. Y. Sattar, U. A. Syed, S. Muhammad, et Z. Kausar, « Real-Time EMG Signal Processing with Implementation of PID Control for Upper-Limb Prosthesis », in *2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, juill. 2019, p. 120-125, doi: 10.1109/AIM.2019.8868796.
- [73] Y. S. Neelum, Z. Kausar, et S. A. Usama, « Reference position estimation for prosthetic elbow and wrist using EMG signals », *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 635, n° 1, p. 012031, oct. 2019, doi: 10.1088/1757-899X/635/1/012031.
- [74] K. Veer et T. Sharma, « Electromyographic classification of effort in muscle strength assessment. », *Biomed. Tech. (Berl)*, vol. 63, n° 2, p. 131-137, 2018, doi: 10.1515/bmt-2016-0038.
- [75] R. Menon, G. D. Caterina, H. Lakany, L. Petropoulakis, B. A. Conway, et J. J. Soraghan, « Study on Interaction Between Temporal and Spatial Information in Classification of EMG Signals for Myoelectric Prostheses », *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 25, n° 10, p. 1832-1842, oct. 2017, doi: 10.1109/TNSRE.2017.2687761.
- [76] B. Hudgins, P. Parker, et R. N. Scott, « A new strategy for multifunction myoelectric control », *IEEE Trans. Biomed. Eng.*, vol. 40, n° 1, p. 82-94, janv. 1993, doi: 10.1109/10.204774.
- [77] K. Englehart, B. Hudgins, P. A. Parker, et M. Stevenson, « Classification of the myoelectric signal using time-frequency based representations », *Med. Eng. Phys.*, vol. 21, n° 6-7, p. 431-438, sept. 1999.
- [78] A. Phinyomark, C. Limsakul, et P. Phukpattaranont, « Application of Wavelet Analysis in EMG Feature Extraction for Pattern Classification », *Meas. Sci. Rev.*, vol. 11, n° 2, p. 45-52, 2011, doi: 10.2478/v10048-011-0009-y.

- [79] A. Phinyomark, F. Quaine, S. Charbonnier, C. Serviere, F. Tarpin-Bernard, et Y. Laurillau, « EMG feature evaluation for improving myoelectric pattern recognition robustness », *Expert Syst. Appl.*, vol. 40, n° 12, p. 4832-4840, sept. 2013, doi: 10.1016/j.eswa.2013.02.023.
- [80] O. W. Samuel, H. Zhou, X. Li, H. Wang, H. Zhang, A. K. Sangaiah, et G. Li., « Pattern recognition of electromyography signals based on novel time domain features for amputees' limb motion classification », *Comput. Electr. Eng.*, vol. 67, p. 646-655, avr. 2018, doi: 10.1016/j.compeleceng.2017.04.003.
- [81] S. Guo, M. Pang, B. Gao, H. Hirata, et H. Ishihara, « Comparison of sEMG-Based Feature Extraction and Motion Classification Methods for Upper-Limb Movement », *Sensors*, vol. 15, n° 4, p. 9022-9038, avr. 2015, doi: 10.3390/s150409022.
- [82] S. Bhagwat et P. Mukherji, « Temporal Feature Extraction for Improving Myoelectric based Recognition of Prosthetic Hand », in *2020 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*, août 2020, p. 67-71, doi: 10.1109/WiSPNET48689.2020.9198476.
- [83] J. Roy, Md. A. Ali, Md. R. Ahmed, et K. Sundaraj, « Machine Learning Techniques for Predicting Surface EMG Activities on Upper Limb Muscle: A Systematic Review », in *Cyber Security and Computer Science*, Cham, 2020, p. 330-339, doi: 10.1007/978-3-030-52856-0_26.
- [84] X. Zhang et P. Zhou, « Sample entropy analysis of surface EMG for improved muscle activity onset detection against spurious background spikes », *J. Electromyogr. Kinesiol.*, vol. 22, n° 6, p. 901-907, déc. 2012, doi: 10.1016/j.jelekin.2012.06.005.
- [85] Y. Guo, G. R. Naik, S. Huang, A. Abraham, et H. T. Nguyen, « Nonlinear multiscale Maximal Lyapunov Exponent for accurate myoelectric signal classification », *Appl. Soft Comput.*, vol. 36, p. 633-640, nov. 2015, doi: 10.1016/j.asoc.2015.07.032.
- [86] A. Krizhevsky, I. Sutskever, et G. E. Hinton, « ImageNet classification with deep convolutional neural networks », *Commun. ACM*, vol. 60, n° 6, p. 84-90, mai 2017, doi: 10.1145/3065386.
- [87] O. Abdel-Hamid, A. Mohamed, H. Jiang, L. Deng, G. Penn, et D. Yu, « Convolutional Neural Networks for Speech Recognition », *IEEEACM Trans. Audio Speech Lang. Process.*, vol. 22, n° 10, p. 1533-1545, oct. 2014, doi: 10.1109/TASLP.2014.2339736.

- [88] E. Choi, A. Schuetz, W. F. Stewart, et J. Sun, « Using recurrent neural network models for early detection of heart failure onset », *J. Am. Med. Inform. Assoc. JAMIA*, vol. 24, n° 2, p. 361-370, mars 2017, doi: 10.1093/jamia/ocw112.
- [89] A. Ameri, M. A. Akhaee, E. Scheme, et K. Englehart, « Regression convolutional neural network for improved simultaneous EMG control », *J. Neural Eng.*, vol. 16, n° 3, p. 036015, avr. 2019, doi: 10.1088/1741-2552/ab0e2e.
- [90] K. He, X. Zhang, S. Ren, et J. Sun, « Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification », févr. 2015, Consulté le: oct. 01, 2018. [En ligne]. Disponible sur: <https://arxiv.org/abs/1502.01852>.
- [91] L. Deng, G. Hinton, et B. Kingsbury, « New types of deep neural network learning for speech recognition and related applications: an overview », in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, mai 2013, p. 8599-8603, doi: 10.1109/ICASSP.2013.6639344.
- [92] A. W. Senior, R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Židek, A. W. R. Nelson, A. Bridgland, H. Penedones, S. Petersen, K. Simonyan, S. Crossan, P. Kohli, D. T. Jones, D. Silver, K. Kavukcuoglu, et D. Hassabis, « Improved protein structure prediction using potentials from deep learning », *Nature*, vol. 577, n° 7792, p. 706-710, janv. 2020, doi: 10.1038/s41586-019-1923-7.
- [93] U. C. Allard, F. Nougrou, C. L. Fall, P. Giguère, C. Gosselin, F. Laviolette, et B. Gosselin, « A convolutional neural network for robotic arm guidance using sEMG based frequency-features », in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, oct. 2016, p. 2464-2470, doi: 10.1109/IROS.2016.7759384.
- [94] S. F. Chevtchenko, R. F. Vale, V. Macario, et F. R. Cordeiro, « A convolutional neural network with feature fusion for real-time hand posture recognition », *Appl. Soft Comput.*, vol. 73, p. 748-766, déc. 2018, doi: 10.1016/j.asoc.2018.09.010.
- [95] U. Côté-Allard, C. L. Fall, A. Drouin, A. Campeau-Lecours, C. Gosselin, K. Glette, F. Laviolette, et B. Gosselin, « Deep Learning for Electromyographic Hand Gesture Signal Classification Using Transfer Learning », *ArXiv180107756 Cs Stat*, janv. 2018, Consulté le: oct. 03, 2019. [En ligne]. Disponible sur: <http://arxiv.org/abs/1801.07756>.

- [96] Y. Du, W. Jin, W. Wei, Y. Hu, et W. Geng, « Surface EMG-Based Inter-Session Gesture Recognition Enhanced by Deep Domain Adaptation », *Sensors*, vol. 17, n° 3, févr. 2017, doi: 10.3390/s17030458.
- [97] K. H. Park et S. W. Lee, « Movement intention decoding based on deep learning for multiuser myoelectric interfaces », in *2016 4th International Winter Conference on Brain-Computer Interface (BCI)*, févr. 2016, p. 1-2, doi: 10.1109/IWW-BCI.2016.7457459.
- [98] M. Zia ur Rehman, A. Waris, S. Gilani, M. Jochumsen, I. Niazi, M. Jamil, D. Farina, et E. Kamavuako, « Multiday EMG-Based Classification of Hand Motions with Deep Learning Techniques », *Sensors*, vol. 18, n° 8, p. 2497, août 2018, doi: 10.3390/s18082497.
- [99] P. Koch, H. Phan, M. Maass, F. Katzberg, et A. Mertins, « Recurrent Neural Network Based Early Prediction of Future Hand Movements », in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, juill. 2018, p. 4710-4713, doi: 10.1109/EMBC.2018.8513145.
- [100] R. Laezza, « Deep Neural Networks for Myoelectric Pattern Recognition - An Implementation for Multifunctional Control. », Master's Thesis, Chalmers University of Technologies, Gothenburg, Sweden, 2018.
- [101] D. Suri et E. Yang, « Control of a Prosthetic Arm Using EMG Signals », *Interdiscip. Des. Sr. Theses*, juin 2019, [En ligne]. Disponible sur: https://scholarcommons.scu.edu/idp_senior/49.
- [102] T.-A. Teban, R.-E. Precup, E.-I. Voisan, T. E. A. de Oliveira, et E. M. Petriu, « Recurrent dynamic neural network model for myoelectric-based control of a prosthetic hand », in *2016 Annual IEEE Systems Conference (SysCon)*, avr. 2016, p. 1-6, doi: 10.1109/SYSCON.2016.7490531.
- [103] M. Jabbari, R. N. Khushaba, et K. Nazarpour, « EMG-Based Hand Gesture Classification with Long Short-Term Memory Deep Recurrent Neural Networks », in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine Biology Society (EMBC)*, juill. 2020, p. 3302-3305, doi: 10.1109/EMBC44109.2020.9175279.
- [104] H. Dantas, D. J. Warren, S. M. Wendelken, T. S. Davis, G. A. Clark, et V. J. Mathews, « Deep Learning Movement Intent Decoders Trained With Dataset Aggregation for Prosthetic

Limb Control », *IEEE Trans. Biomed. Eng.*, vol. 66, n° 11, p. 3192-3203, nov. 2019, doi: 10.1109/TBME.2019.2901882.

[105] C. R. Day, E. K. Chadwick, et D. Blana, « A comparative evaluation of time-delay, deep learning and echo state neural networks when used as simulated transhumeral prosthesis controllers », in *2020 International Joint Conference on Neural Networks (IJCNN)*, juill. 2020, p. 1-7, doi: 10.1109/IJCNN48605.2020.9206772.

[106] Y. Hu, Y. Wong, W. Wei, Y. Du, M. Kankanhalli, et W. Geng, « A novel attention-based hybrid CNN-RNN architecture for sEMG-based gesture recognition », *PLOS ONE*, vol. 13, n° 10, p. e0206049, oct. 2018, doi: 10.1371/journal.pone.0206049.

[107] A. Samadani, « Gated Recurrent Neural Networks for EMG-Based Hand Gesture Classification. A Comparative Study », in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Honolulu, HI, juill. 2018, p. 1-4, doi: 10.1109/EMBC.2018.8512531.

[108] C. Ma, C. Lin, O. W. Samuel, L. Xu, et G. Li, « Continuous estimation of upper limb joint angle from sEMG signals based on SCA-LSTM deep learning approach », *Biomed. Signal Process. Control*, vol. 61, p. 102024, août 2020, doi: 10.1016/j.bspc.2020.102024.

[109] Y. Bengio, P. Simard, et P. Frasconi, « Learning long-term dependencies with gradient descent is difficult », *IEEE Trans. Neural Netw.*, vol. 5, n° 2, p. 157-166, mars 1994, doi: 10.1109/72.279181.

[110] S. Hochreiter et J. Schmidhuber, « Long Short-Term Memory », *Neural Comput.*, vol. 9, n° 8, p. 1735-1780, nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[111] D. P. Kingma et J. Ba, « Adam: A Method for Stochastic Optimization », *ArXiv14126980 Cs*, déc. 2014, Consulté le: oct. 08, 2019. [En ligne]. Disponible sur: <http://arxiv.org/abs/1412.6980>.

[112] N. Reimers et I. Gurevych, « Optimal Hyperparameters for Deep LSTM-Networks for Sequence Labeling Tasks », *ArXiv170706799 Cs*, août 2017, Consulté le: févr. 27, 2020. [En ligne]. Disponible sur: <http://arxiv.org/abs/1707.06799>.

[113] F. Hutter, H. Hoos, et K. Leyton-Brown, « An Efficient Approach for Assessing Hyperparameter Importance », in *International Conference on Machine Learning*, janv. 2014, p.

754-762, Consulté le: sept. 03, 2020. [En ligne]. Disponible sur: <http://proceedings.mlr.press/v32/hutter14.html>.

[114] J. Bergstra et Y. Bengio, « Random search for hyper-parameter optimization », *J. Mach. Learn. Res.*, vol. 13, n° null, p. 281-305, févr. 2012.

[115] T. Yu et H. Zhu, « Hyper-Parameter Optimization: A Review of Algorithms and Applications », *ArXiv200305689 Cs Stat*, mars 2020, Consulté le: sept. 01, 2020. [En ligne]. Disponible sur: <http://arxiv.org/abs/2003.05689>.

[116] W. Gómez-Flores et J. Hernández-López, « Automatic adjustment of the pulse-coupled neural network hyperparameters based on differential evolution and cluster validity index for image segmentation », *Appl. Soft Comput.*, p. 105547, juin 2019, doi: 10.1016/j.asoc.2019.105547.

[117] A. Martín, V. M. Vargas, P. A. Gutiérrez, D. Camacho, et C. Hervás-Martínez, « Optimising Convolutional Neural Networks using a Hybrid Statistically-driven Coral Reef Optimisation algorithm », *Appl. Soft Comput.*, vol. 90, p. 106144, mai 2020, doi: 10.1016/j.asoc.2020.106144.

[118] T. Vuolio, V.-V. Visuri, A. Sorsa, S. Ollila, et T. Fabritius, « Application of a genetic algorithm based model selection algorithm for identification of carbide-based hot metal desulfurization », *Appl. Soft Comput.*, vol. 92, p. 106330, juill. 2020, doi: 10.1016/j.asoc.2020.106330.

[119] L. A. Passos, D. R. Rodrigues, et J. P. Papa, « Fine Tuning Deep Boltzmann Machines Through Meta-Heuristic Approaches », in *2018 IEEE 12th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, mai 2018, p. 000419-000424, doi: 10.1109/SACI.2018.8440959.

[120] L. A. Passos et J. P. Papa, « A metaheuristic-driven approach to fine-tune Deep Boltzmann Machines », *Appl. Soft Comput.*, p. 105717, août 2019, doi: 10.1016/j.asoc.2019.105717.

[121] A. ElSaid, F. El Jamiy, J. Higgins, B. Wild, et T. Desell, « Optimizing long short-term memory recurrent neural networks using ant colony optimization to predict turbine engine vibration », *Appl. Soft Comput.*, vol. 73, p. 969-991, déc. 2018, doi: 10.1016/j.asoc.2018.09.013.

[122] F. Hutter, L. Kotthoff, et J. Vanschoren, Éd., *Automated Machine Learning: Methods, Systems, Challenges*. Cham: Springer International Publishing, 2019.

- [123] E. Biddiss, D. Beaton, et T. Chau, « Consumer design priorities for upper limb prosthetics », *Disabil. Rehabil. Assist. Technol.*, vol. 2, n° 6, p. 346-357, nov. 2007.
- [124] E. Raffin, P. Giraux, et K. T. Reilly, « The moving phantom: motor execution or motor imagery? », *Cortex J. Devoted Study Nerv. Syst. Behav.*, vol. 48, n° 6, p. 746-757, juin 2012, doi: 10.1016/j.cortex.2011.02.003.
- [125] M. Asghari Oskoei et H. Hu, « Myoelectric control systems—A survey », *Biomed. Signal Process. Control*, vol. 2, n° 4, p. 275-294, oct. 2007, doi: 10.1016/j.bspc.2007.07.009.
- [126] M. Atzori, A. Gijsberts, C. Castellini, B. Caputo, A.-G. M. Hager, S. Elsig, G. Giatsidis, F. Bassetto, et H. Müller, « Effect of clinical parameters on the control of myoelectric robotic prosthetic hands », *J. Rehabil. Res. Dev.*, vol. 53, n° 3, p. 345-358, 2016, doi: 10.1682/JRRD.2014.09.0218.
- [127] J. Gade, R. Hugosdottir, et E. N. Kamavuako, « Phantom movements from physiologically inappropriate muscles: A case study with a high transhumeral amputee », in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, août 2015, p. 3488-3491, doi: 10.1109/EMBC.2015.7319144.
- [128] Y. Bengio, A. Courville, et P. Vincent, « Representation Learning: A Review and New Perspectives », *ArXiv12065538 Cs*, juin 2012, Consulté le: oct. 08, 2019. [En ligne]. Disponible sur: <http://arxiv.org/abs/1206.5538>.
- [129] J. W. Robertson, K. B. Englehart, et E. J. Scheme, « Effects of Confidence-Based Rejection on Usability and Error in Pattern Recognition-Based Myoelectric Control », *IEEE J. Biomed. Health Inform.*, vol. 23, n° 5, p. 2002-2008, sept. 2019, doi: 10.1109/JBHI.2018.2878907.
- [130] A. Ameri, E. J. Scheme, E. N. Kamavuako, K. B. Englehart, et P. A. Parker, « Real-Time, Simultaneous Myoelectric Control Using Force and Position-Based Training Paradigms », *IEEE Trans. Biomed. Eng.*, vol. 61, n° 2, p. 279-287, févr. 2014, doi: 10.1109/TBME.2013.2281595.
- [131] L. Hargrove, Y. Losier, B. Lock, K. Englehart, et B. Hudgins, « A Real-Time Pattern Recognition Based Myoelectric Control Usability Study Implemented in a Virtual Environment », in *2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, août 2007, p. 4842-4845, doi: 10.1109/IEMBS.2007.4353424.

- [132] B. A. Lock, K. Englehart, et B. Hudgins, « Real-time Myoelectric Control in a Virtual Environment to Relate Usability vs. Accuracy », Fredericton. Canada., 2005, Consulté le: mai 14, 2018. [En ligne]. Disponible sur: <https://dukespace.lib.duke.edu/dspace/handle/10161/2721>.
- [133] S. Solnik, P. Rider, K. Steinweg, P. DeVita, et T. Hortobágyi, « Teager–Kaiser energy operator signal conditioning improves EMG onset detection », *Eur. J. Appl. Physiol.*, vol. 110, n° 3, p. 489-498, oct. 2010, doi: 10.1007/s00421-010-1521-8.
- [134] K. Englehart, B. Hudgin, et P. A. Parker, « A wavelet-based continuous classification scheme for multifunction myoelectric control », *IEEE Trans. Biomed. Eng.*, vol. 48, n° 3, p. 302-311, mars 2001, doi: 10.1109/10.914793.
- [135] R. Pascanu, T. Mikolov, et Y. Bengio, « On the Difficulty of Training Recurrent Neural Networks », in *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, Atlanta, GA, USA, 2013, p. III-1310-III-1318, Consulté le: sept. 17, 2019. [En ligne]. Disponible sur: <http://dl.acm.org/citation.cfm?id=3042817.3043083>.
- [136] S. Scardapane et D. Wang, « Randomness in neural networks: an overview », *Wiley Interdiscip Rev Data Min Knowl Discov*, vol. 7, 2017, doi: 10.1002/widm.1200.
- [137] A. Phinyomark, P. Phukpattaranont, et C. Limsakul, « Feature reduction and selection for EMG signal classification », *Expert Syst. Appl.*, vol. 39, n° 8, p. 7420-7431, juin 2012, doi: 10.1016/j.eswa.2012.01.102.
- [138] R. Crepin, C. L. Fall, Q. Mascaret, C. Gosselin, A. Campeau-Lecours, et B. Gosselin, « Real-Time Hand Motion Recognition Using sEMG Patterns Classification », in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, juill. 2018, p. 2655-2658, doi: 10.1109/EMBC.2018.8512820.
- [139] S. Muceli, N. Jiang, et D. Farina, « Extracting Signals Robust to Electrode Number and Shift for Online Simultaneous and Proportional Myoelectric Control by Factorization Algorithms », *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 22, n° 3, p. 623-633, mai 2014, doi: 10.1109/TNSRE.2013.2282898.
- [140] E. Scheme, A. Fougner, Ø Stavdahl, A. D. C. Chan, et K. Englehart, « Examining the adverse effects of limb position on pattern recognition based myoelectric control », août 2010, p. 6337-6340, doi: 10.1109/IEMBS.2010.5627638.

- [141] N. M. Rezk, M. Purnaprajna, T. Nordström, et Z. Ul-Abdin, « Recurrent Neural Networks: An Embedded Computing Perspective », *IEEE Access*, vol. 8, p. 57967-57996, 2020, doi: 10.1109/ACCESS.2020.2982416.
- [142] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, et A. Talwalkar, « Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization », *ArXiv160306560 Cs Stat*, juin 2018, Consulté le: sept. 03, 2020. [En ligne]. Disponible sur: <http://arxiv.org/abs/1603.06560>.
- [143] M. Balazinski, S. Achiche, et L. Baron, « Influence of Optimization and Selection Criteria on Genetically-Generated Fuzzy Knowledge Bases », présenté à International Conference on Advanced Manufacturing Technology, Johor Bahru, Malesia, août 2000.
- [144] D. Coyle, G. Prasad, et T. M. McGinnity, « Faster Self-Organizing Fuzzy Neural Network Training and a Hyperparameter Analysis for a Brain–Computer Interface », *IEEE Trans. Syst. Man Cybern. Part B Cybern.*, vol. 39, n° 6, p. 1458-1471, déc. 2009, doi: 10.1109/TSMCB.2009.2018469.
- [145] C. Garbin, X. Zhu, et O. Marques, « Dropout vs. batch normalization: an empirical study of their impact to deep learning », *Multimed. Tools Appl.*, vol. 79, n° 19, p. 12777-12815, mai 2020, doi: 10.1007/s11042-019-08453-9.
- [146] P. Schratz, J. Muenchow, E. Iturritxa, J. Richter, et A. Brenning, « Hyperparameter tuning and performance assessment of statistical and machine-learning algorithms using spatial data », *Ecol. Model.*, vol. 406, p. 109-120, août 2019, doi: 10.1016/j.ecolmodel.2019.06.002.
- [147] R. Souza, L. Neves, L. Azeredo, R. Luiz, E. Tady, P. Cavalin, et M. Mattoso, « Towards a Human-in-the-Loop Library for Tracking Hyperparameter Tuning in Deep Learning Development », p. 5, 2018.
- [148] J. N. van Rijn et F. Hutter, « Hyperparameter Importance Across Datasets », in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, London United Kingdom, juill. 2018, p. 2367-2376, doi: 10.1145/3219819.3220058.
- [149] S. Zhou et W. Song, « Deep learning-based roadway crack classification using laser-scanned range images: A comparative study on hyperparameter selection », *Autom. Constr.*, vol. 114, p. 103171, juin 2020, doi: 10.1016/j.autcon.2020.103171.

- [150] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, et J. Schmidhuber, « LSTM: A Search Space Odyssey », *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, n° 10, p. 2222-2232, oct. 2017, doi: 10.1109/TNNLS.2016.2582924.
- [151] J.-L. Ren, Y.-H. Chien, E.-Y. Chia, L.-C. Fu, et J.-S. Lai, « Deep Learning based Motion Prediction for Exoskeleton Robot Control in Upper Limb Rehabilitation », in *2019 International Conference on Robotics and Automation (ICRA)*, mai 2019, p. 5076-5082, doi: 10.1109/ICRA.2019.8794187.
- [152] T.-A. Teban, R.-E. Precup, E.-C. Lunca, A. Albu, C.-A. Bojan-Dragos, et E. M. Petriu, « Recurrent Neural Network Models for Myoelectricbased Control of a Prosthetic Hand », in *2018 22nd International Conference on System Theory, Control and Computing (ICSTCC)*, oct. 2018, p. 603-608, doi: 10.1109/ICSTCC.2018.8540720.
- [153] O. Barron, M. Raison, G. Gaudet, et S. Achiche, « Recurrent Neural Network for electromyographic gesture recognition in transhumeral amputees », *Appl. Soft Comput.*, vol. 96, n° 106616, nov. 2020, doi: 10.1016/j.asoc.2020.106616.
- [154] K. Hornik, M. Stinchcombe, et H. White, « Multilayer feedforward networks are universal approximators », *Neural Netw.*, vol. 2, n° 5, p. 359-366, janv. 1989, doi: 10.1016/0893-6080(89)90020-8.
- [155] A. Karpathy, J. Johnson, et L. Fei-Fei, « Visualizing and Understanding Recurrent Networks », *ArXiv150602078 Cs*, nov. 2015, Consulté le: avr. 07, 2020. [En ligne]. Disponible sur: <http://arxiv.org/abs/1506.02078>.
- [156] I. Ketyko, F. Kovacs, et K. Z. Varga, « Domain Adaptation for sEMG-based Gesture Recognition with Recurrent Neural Networks », in *2019 International Joint Conference on Neural Networks (IJCNN)*, Budapest, Hungary, juill. 2019, p. 1-7, doi: 10.1109/IJCNN.2019.8852018.
- [157] S. Ioffe et C. Szegedy, « Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift », *ArXiv150203167 Cs*, mars 2015, Consulté le: juin 10, 2020. [En ligne]. Disponible sur: <http://arxiv.org/abs/1502.03167>.
- [158] M. Schuster et K. K. Paliwal, « Bidirectional recurrent neural networks », *IEEE Trans. Signal Process.*, vol. 45, n° 11, p. 2673-2681, nov. 1997, doi: 10.1109/78.650093.

- [159] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, et R. Salakhutdinov, « Dropout: A Simple Way to Prevent Neural Networks from Overfitting », *J. Mach. Learn. Res.*, vol. 15, n° 56, p. 1929-1958, 2014.
- [160] W. Zaremba, I. Sutskever, et O. Vinyals, « Recurrent Neural Network Regularization », *ArXiv14092329 Cs*, févr. 2015, Consulté le: juin 23, 2020. [En ligne]. Disponible sur: <http://arxiv.org/abs/1409.2329>.
- [161] Y. Gal et Z. Ghahramani, « A Theoretically Grounded Application of Dropout in Recurrent Neural Networks », *ArXiv151205287 Stat*, oct. 2016, Consulté le: juin 23, 2020. [En ligne]. Disponible sur: <http://arxiv.org/abs/1512.05287>.
- [162] D. Bahdanau, K. Cho, et Y. Bengio, « Neural Machine Translation by Jointly Learning to Align and Translate », *ArXiv14090473 Cs Stat*, mai 2015, Consulté le: juin 05, 2020. [En ligne]. Disponible sur: <http://arxiv.org/abs/1409.0473>.
- [163] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, et Y. Bengio, « Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation », *ArXiv14061078 Cs Stat*, sept. 2014, Consulté le: juin 09, 2020. [En ligne]. Disponible sur: <http://arxiv.org/abs/1406.1078>.
- [164] I. Sutskever, O. Vinyals, et Q. V. Le, « Sequence to Sequence Learning with Neural Networks », *ArXiv14093215 Cs*, déc. 2014, Consulté le: juin 09, 2020. [En ligne]. Disponible sur: <http://arxiv.org/abs/1409.3215>.
- [165] M.-T. Luong, H. Pham, et C. D. Manning, « Effective Approaches to Attention-based Neural Machine Translation », *ArXiv150804025 Cs*, sept. 2015, Consulté le: juin 05, 2020. [En ligne]. Disponible sur: <http://arxiv.org/abs/1508.04025>.
- [166] J. Cheng, L. Dong, et M. Lapata, « Long Short-Term Memory-Networks for Machine Reading », *ArXiv160106733 Cs*, sept. 2016, Consulté le: juin 05, 2020. [En ligne]. Disponible sur: <http://arxiv.org/abs/1601.06733>.
- [167] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, et I. Polosukhin, « Attention Is All You Need », *ArXiv170603762 Cs*, déc. 2017, Consulté le: juin 10, 2020. [En ligne]. Disponible sur: <http://arxiv.org/abs/1706.03762>.

- [168] S. Song, C. Lan, J. Xing, W. Zeng, et J. Liu, « An End-to-End Spatio-Temporal Attention Model for Human Action Recognition from Skeleton Data », *ArXiv161106067 Cs*, nov. 2016, Consulté le: avr. 07, 2020. [En ligne]. Disponible sur: <http://arxiv.org/abs/1611.06067>.
- [169] N. Parajuli, N. Sreenivasan, P. Bifulco, M. Cesarelli, S. Savino, V. Niola, D. Esposito, Tara. J. Hamilton, G. R. Naik, U. Gunawardana, et G. D. Gargiulo, « Real-Time EMG Based Pattern Recognition Control for Hand Prostheses: A Review on Existing Methods, Challenges and Future Implementation », *Sensors*, vol. 19, n° 20, Art. n° 20, janv. 2019, doi: 10.3390/s19204596.
- [170] M. Raison, C. Detrembleur, P. Fisette, et J.-C. Samin, « Assessment of Antagonistic Muscle Forces During Forearm Flexion/Extension », in *Multibody Dynamics: Computational Methods and Applications*, K. Arczewski, W. Blajer, J. Fraczek, et M. Wojtyra, Éd. Dordrecht: Springer Netherlands, 2011, p. 215-238.
- [171] Y. Wu, B. Zheng, et Y. Zhao, « Dynamic Gesture Recognition Based on LSTM-CNN », in *2018 Chinese Automation Congress (CAC)*, Xi'an, China, nov. 2018, p. 2446-2450, doi: 10.1109/CAC.2018.8623035.
- [172] A. M. Simon, L. J. Hargrove, B. A. Lock, et T. A. Kuiken, « Target Achievement Control Test: evaluating real-time myoelectric pattern-recognition control of multifunctional upper-limb prostheses », *J. Rehabil. Res. Dev.*, vol. 48, n° 6, p. 619-627, 2011.
- [173] E. J. Scheme et K. B. Englehart, « Validation of a Selective Ensemble-Based Classification Scheme for Myoelectric Control Using a Three-Dimensional Fitts' Law Test », *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 21, n° 4, p. 616-623, juill. 2013, doi: 10.1109/TNSRE.2012.2226189.
- [174] E. J. Scheme, B. S. Hudgins, et K. B. Englehart, « Confidence-Based Rejection for Improved Pattern Recognition Myoelectric Control », *IEEE Trans. Biomed. Eng.*, vol. 60, n° 6, p. 1563-1570, juin 2013, doi: 10.1109/TBME.2013.2238939.
- [175] I. Ruhunage, S. Mallikarachchi, D. Chinthaka, J. Sandaruwan, et T. D. Lalitharatne, « Hybrid EEG-EMG Signals Based Approach for Control of Hand Motions of a Transhumeral Prosthesis », in *2019 IEEE 1st Global Conference on Life Sciences and Technologies (LifeTech)*, mars 2019, p. 50-53, doi: 10.1109/LifeTech.2019.8883865.

- [176] Y. Omama, C. Haddad, M. Machaalany, A. Hamoudi, M. Hajj-Hassan, M. A. Ali, et L. Hamawy, « Surface EMG Classification of Basic Hand Movement », in *2019 Fifth International Conference on Advances in Biomedical Engineering (ICABME)*, oct. 2019, p. 1-4, doi: 10.1109/ICABME47164.2019.8940352.
- [177] L. He et P. A. Mathieu, « Biceps Brachii Muscle Synergy and Target Reaching in a Virtual Environment », *Front. Neurorobotics*, vol. 13, 2019, doi: 10.3389/fnbot.2019.00100.
- [178] K. Bakshi, R. Pramanik, M. Manjunatha, et C. S. Kumar, « Upper Limb Prosthesis Control: A Hybrid EEG-EMG Scheme for Motion Estimation in Transhumeral Subjects », in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, Honolulu, HI, juill. 2018, p. 2024-2027, doi: 10.1109/EMBC.2018.8512678.
- [179] H. I. Aly, S. Youssef, et C. Fathy, « Hybrid Brain Computer Interface for Movement Control of Upper Limb Prostheses », in *2018 International Conference on Biomedical Engineering and Applications (ICBEA)*, Funchal, juill. 2018, p. 1-6, doi: 10.1109/ICBEA.2018.8471729.
- [180] D. S. V. Bandara, J. Arata, et K. Kiguchi, « Towards Control of a Transhumeral Prosthesis with EEG Signals », *Bioengineering*, vol. 5, n° 2, p. 26, juin 2018, doi: 10.3390/bioengineering5020026.
- [181] S. Herle, « Movement intention detection from SEMG signals using time-domain features and discriminant analysis classifiers », in *2018 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, Cluj-Napoca, mai 2018, p. 1-6, doi: 10.1109/AQTR.2018.8402774.
- [182] Y. Xu, D. Zhang, Y. Wang, J. Feng, et W. Xu, « Two ways to improve myoelectric control for a transhumeral amputee after targeted muscle reinnervation: a case study », *J. NeuroEngineering Rehabil.*, vol. 15, n° 1, p. 37, mai 2018, doi: 10.1186/s12984-018-0376-9.
- [183] K. R. Lyons et S. S. Joshi, « Upper Limb Prosthesis Control for High-Level Amputees via Myoelectric Recognition of Leg Gestures », *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 26, n° 5, p. 1056-1066, mai 2018, doi: 10.1109/TNSRE.2018.2807360.
- [184] Y. Ogiri, Y. Yamanoi, W. Nishino, R. Kato, T. Takagi, et H. Yokoi, « Development of an upper limb neuroprosthesis to voluntarily control elbow and hand », in *2017 26th IEEE*

International Symposium on Robot and Human Interactive Communication (RO-MAN), août 2017, p. 298-303, doi: 10.1109/ROMAN.2017.8172317.

[185] D. Bai, S. Chen, J. Yang, C. Xia, B. Sun, Y. Jiang, et H Yokoi, « Intelligent prosthetic arm force control based on sEMG analysis and BPNN classifier », *2017 IEEE Int. Conf. Cyborg Bionic Syst. CBS*, p. 108-113, 2017, doi: 10.1109/CBS.2017.8266078.

[186] J. Wang et W. Wichakool, « Artificial elbow joint classification using upper arm based on surface-EMG signal », in *2017 IEEE 3rd International Conference on Engineering Technologies and Social Sciences (ICETSS)*, août 2017, p. 1-4, doi: 10.1109/ICETSS.2017.8324198.

[187] K. Veer et R. Vig, « Analysis and recognition of operations using SEMG from upper arm muscles », *Expert Syst.*, vol. 34, n° 6, p. e12221, 2017, doi: 10.1111/exsy.12221.

[188] N. Jose, R. Raj, P. K. Adithya, et K. S. Sivanadan, « Classification of forearm movements from sEMG time domain features using machine learning algorithms », in *TENCON 2017 - 2017 IEEE Region 10 Conference*, nov. 2017, p. 1624-1628, doi: 10.1109/TENCON.2017.8228118.

[189] N. Jarrassé, C. Nicol, F. Richer, A. Touillet, N. Martinet, J. Paysant, et J. B. De Graaf, « Voluntary phantom hand and finger movements in transhumeral amputees could be used to naturally control polydigital prostheses », in *2017 International Conference on Rehabilitation Robotics (ICORR)*, juill. 2017, p. 1239-1245, doi: 10.1109/ICORR.2017.8009419.

[190] L. He et P. A. Mathieu, « Muscle synergy of biceps brachii and online classification of upper limb posture », in *2017 International Conference on Virtual Rehabilitation (ICVR)*, juin 2017, p. 1-6, doi: 10.1109/ICVR.2017.8007484.

[191] X. Li, O. W. Samuel, X. Zhang, H. Wang, P. Fang, et G. Li, « A motion-classification strategy based on sEMG-EEG signal combination for upper-limb amputees », *J. NeuroEngineering Rehabil.*, vol. 14, janv. 2017, doi: 10.1186/s12984-016-0212-z.

[192] M. Salama et A. Bakr, « Six Prosthetic Arm Movements Using Electromyogram Signals: A Prototype », in *2016 UKSim-AMSS 18th International Conference on Computer Modelling and Simulation (UKSim)*, avr. 2016, p. 37-42, doi: 10.1109/UKSim.2016.19.

[193] T. Lenzi, J. Lipsey, et J. W. Sensinger, « The RIC Arm—A Small Anthropomorphic Transhumeral Prosthesis », *IEEEASME Trans. Mechatron.*, vol. 21, n° 6, p. 2660-2671, déc. 2016, doi: 10.1109/TMECH.2016.2596104.

- [194] A. Ehrampoosh, A. Yousefi-koma, et M. Ayati, « Development of myoelectric interface based on pattern recognition and regression based models », in *2016 Artificial Intelligence and Robotics (IRANOPEN)*, avr. 2016, p. 145-150, doi: 10.1109/RIOS.2016.7529505.
- [195] C. W. Antuvan, F. Bisio, F. Marini, S.-C. Yen, E. Cambria, et L. Masia, « Role of Muscle Synergies in Real-Time Classification of Upper Limb Motions using Extreme Learning Machines », *J. NeuroEngineering Rehabil.*, vol. 13, août 2016, doi: 10.1186/s12984-016-0183-0.
- [196] K. Veer, T. Sharma, et R. Agarwal, « A neural network-based electromyography motion classifier for upper limb activities », *J. Innov. Opt. Health Sci.*, vol. 09, n° 06, p. 1650025, nov. 2016, doi: 10.1142/S1793545816500255.
- [197] Q. Wu, J. Shao, X. Wu, Y. Zhou, F. Liu, et F. Xiao, « Upper Limb Motion Recognition Based on LLE-ELM Method of sEMG », *Int. J. Pattern Recognit. Artif. Intell.*, vol. 31, n° 06, p. 1750018, déc. 2016, doi: 10.1142/S0218001417500185.
- [198] P. Azaripasand, A. Maleki, et A. Fallah, « Classification of ADLs using muscle activation waveform versus thirteen EMG features », in *2015 22nd Iranian Conference on Biomedical Engineering (ICBME)*, nov. 2015, p. 189-193, doi: 10.1109/ICBME.2015.7404140.
- [199] Q. Zhang, C. Zheng, et C. Xiong, « EMG-based estimation of shoulder and elbow joint angles for intuitive myoelectric control », in *2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, juin 2015, p. 1912-1916, doi: 10.1109/CYBER.2015.7288239.
- [200] C. W. Antuvan, F. Bisio, E. Cambria, et L. Masia, « Muscle synergies for reliable classification of arm motions using myoelectric interface », in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, août 2015, p. 1136-1139, doi: 10.1109/EMBC.2015.7318566.
- [201] K. Veer, « A technique for classification and decomposition of muscle signal for control of myoelectric prostheses based on wavelet statistical classifier », *Measurement*, vol. 60, p. 283-291, janv. 2015, doi: 10.1016/j.measurement.2014.10.023.
- [202] A. Suberbiola, E. Zulueta, J. M. Lopez-Guede, I. Etxeberria-Agiriano, et M. Graña, « Arm Orthosis/Prosthesis Movement Control Based on Surface EMG Signal Extraction », *Int. J. Neural Syst.*, vol. 25, n° 03, p. 1550009, mai 2015, doi: 10.1142/S0129065715500094.

- [203] R. Almada-Aguilar, L. M. Torres-Treviño, et G. Quiroz, « Establishing a Simplified Functional Relationship between EMG Signals and Actuation Signals Using Artificial Neural Networks », in *2014 13th Mexican International Conference on Artificial Intelligence*, nov. 2014, p. 128-132, doi: 10.1109/MICAI.2014.26.
- [204] D. C. Tkach, A. J. Young, L. H. Smith, E. J. Rouse, et L. J. Hargrove, « Real-time and offline performance of pattern recognition myoelectric control using a generic electrode grid with targeted muscle reinnervation patients », *IEEE Trans. Neural Syst. Rehabil. Eng. Publ. IEEE Eng. Med. Biol. Soc.*, vol. 22, n° 4, p. 727-734, juill. 2014, doi: 10.1109/TNSRE.2014.2302799.
- [205] L. J. Hargrove, B. A. Lock, et A. M. Simon, « Pattern recognition control outperforms conventional myoelectric control in upper limb patients with targeted muscle reinnervation », in *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, juill. 2013, p. 1599-1602, doi: 10.1109/EMBC.2013.6609821.

ANNEXE A – TABLEAU RÉSUMÉ DE LA REVUE DE LITTÉRATURE

Tableau A.1 : Résultats de la revue de littérature sur les méthodes utilisées pour la classification des intentions de mouvement chez les amputés TH.

Year	Input	Features	Class	Dataset	Model	Results	Ref.
2020	sEMG/ NIR	T.D. (11)	8	12 AB	MLPNN/LDA/ QDA	84% (MLPNN) 80% (LDA) 82% (QDA)	[12]
2019	sEMG	T.D. (10)	4	10 AB	ANN/SVM	94% (ANN) 85% (SVM)	[72]
2019	sEMG	-	2	5 AB	ANN	87.0%	[175]
2019	sEMG	T.D. (5)	4	10 AB	ANN	91.9%	[73]
2019	sEMG	T.D. (20)	7	-	ANN	-	[176]
2019	sEMG	Muscle synergy	5/8	12 AB	KNN	82% (5 classes) 72% (8 classes)	[177]
2018	sEMG/ EEG	T.D. (15) F.D. (5)	21	5 AB/ 1 TH	KRLST- LSSVM	68.0% (AB)/ 63.3% (TH)	[178]
2018	sEMG	T.D. (4)	8	3 AB/ 2 TH	LDA	87.7% (TH)	[2]
2018	sEMG/ EEG	T.D. (4)	4	4 TH	QDA	98.8%	[179]
2018	EEG	T.D. (4)	3	5 AB	ANN/KNN	72.5% (NN)/ 59.6% (KNN)	[180]
2018	sEMG	T.D. (8)	4	5 AB	LDA/QDA	87.5% (LDA)	[181]
2018	sEMG	T.D. (10)	7	5 TH/ 5 TR	LDA/ANN	82.2% (TH)/ 84.0% (TR)	[80]
2018	sEMG	T.D. (5)	4	TH	ANN	96.4%	[74]
2018	sEMG+ TMR	T.D. (4)	5	1 TH	LDA	-	[182]
2018	sEMG	T.D. (4)	9	17 AB	LDA	82.8 %	[183]
2018	sEMG/ KIN	T.D. (10)	9	5 TH	ANN	76.6%	[4]
2017	sEMG/ KIN	T.D. (4)	14	5 TH	LDA	70%	[3]
2017	sEMG+ TMR/E EG	T.D. (1) F.D.(1)	7	1 TH- TMR/ 1 TH	ANN	97.6% (TH- TMR)/ 72.5% (TH)	[184]
2017	sEMG	T.D. (1) F.D. (1)	5	5 AB	ANN/LDA/ KNN	72% (NN)/ 64% (LDA)/ 75% (KNN)	[185]
2017	sEMG	T.D. (4) F.D. (1)	4	8 AB	ANN	96.3%	[186]
2017	sEMG	T.D. (8)	4	7 TH	PCA	89.3%	[187]

2017	sEMG	T.D. (5)	2	30 AB	MLPNN/RF	91.6% (MLPNN)/ 97.7% (RF)	[188]
2017	sEMG	T.D. (3)	5	3 TH	LDA	78%	[189]
2017	sEMG	Muscle synergy	2	2 AB	KNN	94.9%	[190]
2017	sEMG/ EEG	T.D. (4)	5	4 TH	LDA	87.0%	[191]
2017	sEMG	Wavelet transform	2	10 AB	ANN	99.7%	[192]
2017	sEMG	T.D. (5)	11	1 TH	LDA	-	[193]
2017	sEMG	T.D. (5)	2	1 AB	TDANN/LDA	95.1%	[194]
2017	sEMG	T.D. (2)	6	7 AB	ANN	92.1%	[195]
2017	sEMG	T.D. (6)	2	5 AB	ANN	97.%	[196]
2016	sEMG	T.D. (6)	6+	-	LLE-ELM	97.2%	[197]
2015	sEMG	T.D. (13)	3	6 AB	SVM/DT/ KNN/LDA/ Bayes	78.9% (SVM)/ 90.0% (DT)/ 92.22 (KNN)/ 96.67% (LDA)/ 52.22 (Bayes)	[198]
2015	sEMG/ KIN	T.D. (2)	4	4 AB	LS-SVM+PCA	92%	[199]
2015	sEMG	T.D. (5)	4	1 TH	LDA	93.8%	[127]
2015	sEMG	T.D. (2) Synergy	6	4 AB	ELM	76.0% (T.D.)/ 95.8% (Synergy)	[200]
2015	sEMG	T.D. (5)	4	10 TH	LDA	85%	[201]
2015	sEMG	T.D. (1)	3	-	ANN	91%	[202]
2014	sEMG	F.D. (1)	2	1 AB	ANN	82.9%	[203]
2014	sEMG+ TMR	T.D.(5)	29	1 TH	LDA	88.5%	[204]
2013	sEMG+ TMR	T.D.	7	1 SD/ 3 TH	LDA	83.7%	[205]
2013	Intramuscular EMG	T.D. (4)	4	5 AB	TDANN	Joint angle pred. RMSE: 15.7-24.9 °	[8]

ANNEXE B – ARTICLE 2 : ÉTUDE DE L'IMPACT DE CHAQUE HYPERPARAMÈTRE

The following appendix presents the impact study for every studied hyperparameter. A graph shows the probability density of the classification accuracy for each value or option of a given hyperparameter. This allows to see the distribution of the 60 random configurations tested. The graph for the architectures is not shown as the 15 probability densities make it difficult to read. Furthermore, a table summarizes the performance of each value or option and highlights the differences that may exist. The median value is displayed for all tested values or options. Then, the best median box is greyed out. Finally, the difference between the other values or options and the best median is given, along with the p-value. A bold p-value indicates that the difference is statistically significant ($p < 0.01$).

Architecture

Tableau B.2 : Architecture impact. The greyed box indicates the best performance. A bold p-value indicates the difference with the best median is statistically meaningful ($p < 0.01$). In each box, the first line indicates the median, the second line is the difference in median and the last line is the p-value of that difference.

	32 units	64 units	128 units	256 units	512 units
1 layer	90.93 %	89.83 %	90.19 %	87.73 %	78.30 %
	-	-1.1 %	-0.75 %	-3.21 %	-12.63 %
	-	0.621	0.042	0.006	< 0.001
2 layers	90.55 %	90.50 %	89.60 %	81.17 %	65.76 %
	-0.39 %	-0.43 %	-1.33 %	-9.76 %	-25.17 %
	0.512	0.131	0.013	< 0.001	< 0.001
3 layers	87.23 %	88.39 %	83.97 %	70.02 %	56.65 %
	-3.70 %	-2.55 %	-6.96 %	-20.91 %	-34.28 %
	0.002	0.003	< 0.001	< 0.001	< 0.001

The best values for the architecture was one layer ($L = 1$) of 32 units ($N = 32$), which obtained a median of 90.93%. Although, there is no significant difference ($p < 0.01$) between architectures with $N \leq 128$ and $L \leq 2$, and the maximum median difference with the best performance is 1.33 %. For $L = 3$ or $N \geq 256$, the difference is significant and can range from 3.21 % to 34.28%. For these architectures, the more complex (i.e. bigger N and L) they get, the bigger is the difference with the

best architecture tested. Additionally, all trainings did not reach the maximum of epochs (300) and were ended by the early stopping with a patience of 25 epochs.

Batch normalization

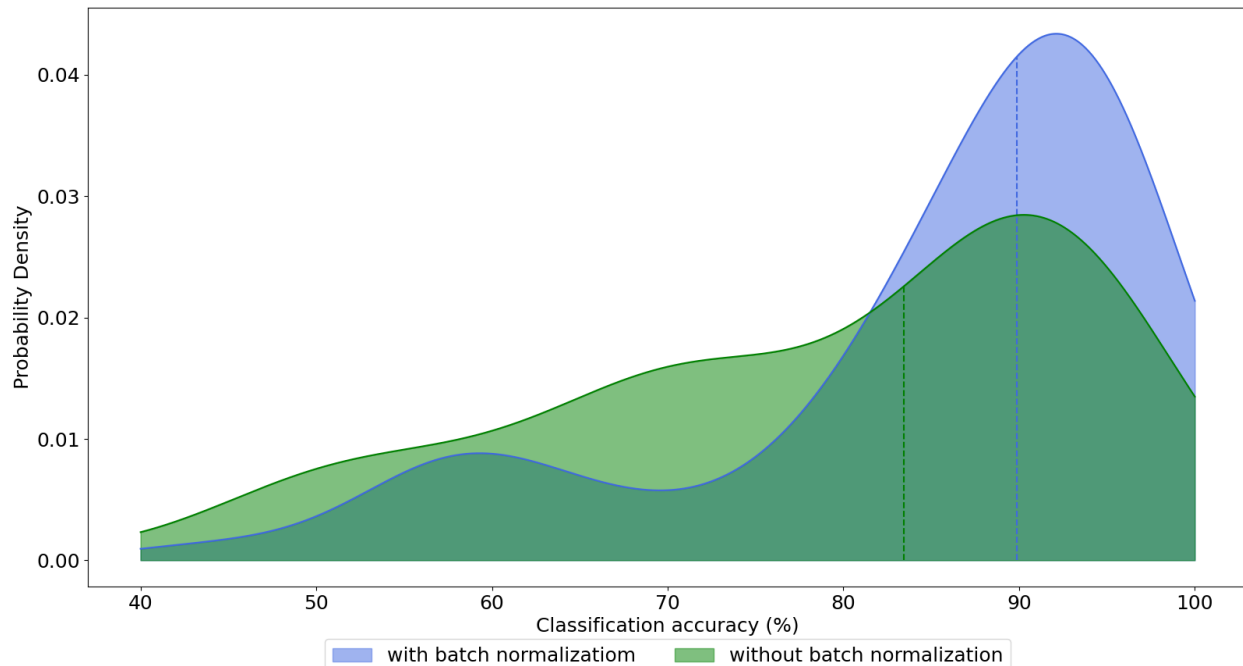


Figure B.7-1 : Batch normalization impact. The dashed line corresponds to the median of the same colored distribution.

As seen in Figure B.7-1, a LSTM network without batch normalization has an accuracy that varies a lot. Using a batch normalization layer narrows the distribution of the accuracy around a higher median.

Tableau B.3 : Batch Normalization impact. The greyed boxes indicate the best performance. A bold p-value indicates the difference with the best median is statistically meaningful ($p < 0.01$).

	with Batch Normalization	without Batch Normalization
Median	89.89 %	83.2 %
Difference in median	-	-6.47 %
Statistical difference (p-value)	-	< 0.001

The best option is a network with batch normalization as it shows a better median of 89.89 % than not using it. The difference is significant and the gain in accuracy is considerable (6.47 %).

Bidirectional LSTM

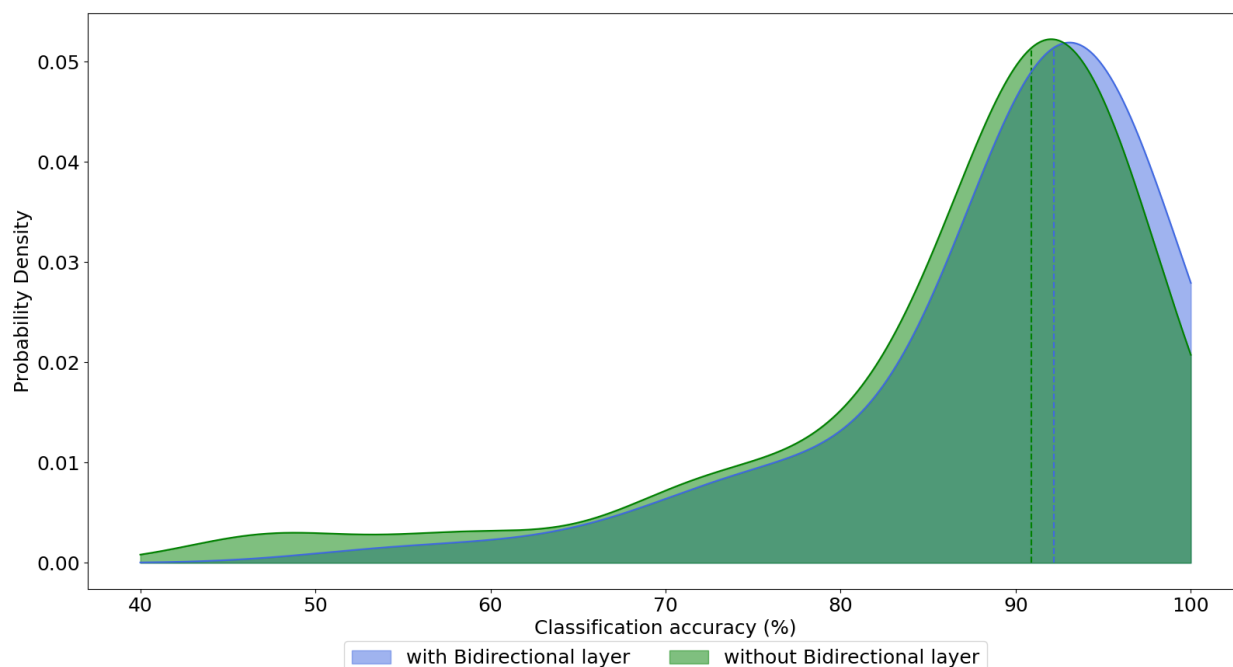


Figure B.7-2 : Bidirectional layer impact. The dashed line corresponds to the median of the same colored distribution.

Using a LSTM network with or without a bidirectional layer shows a similar distribution in the accuracy. The distribution is narrow around a median higher than 90 %.

Tableau B.4 : Bidirectional layer impact. A bold p-value indicates the difference with the best median is statistically meaningful ($p < 0.01$).

	with Bidirectional layer	without Bidirectional layer
Median	92.19 %	90.87 %
Difference in median	-	-1.32 %
Statistical difference (p-value)	-	0.013

No significant difference is detected between the two options. Both displayed a similar median with a difference of 1.32 % in accuracy.

Dropout option

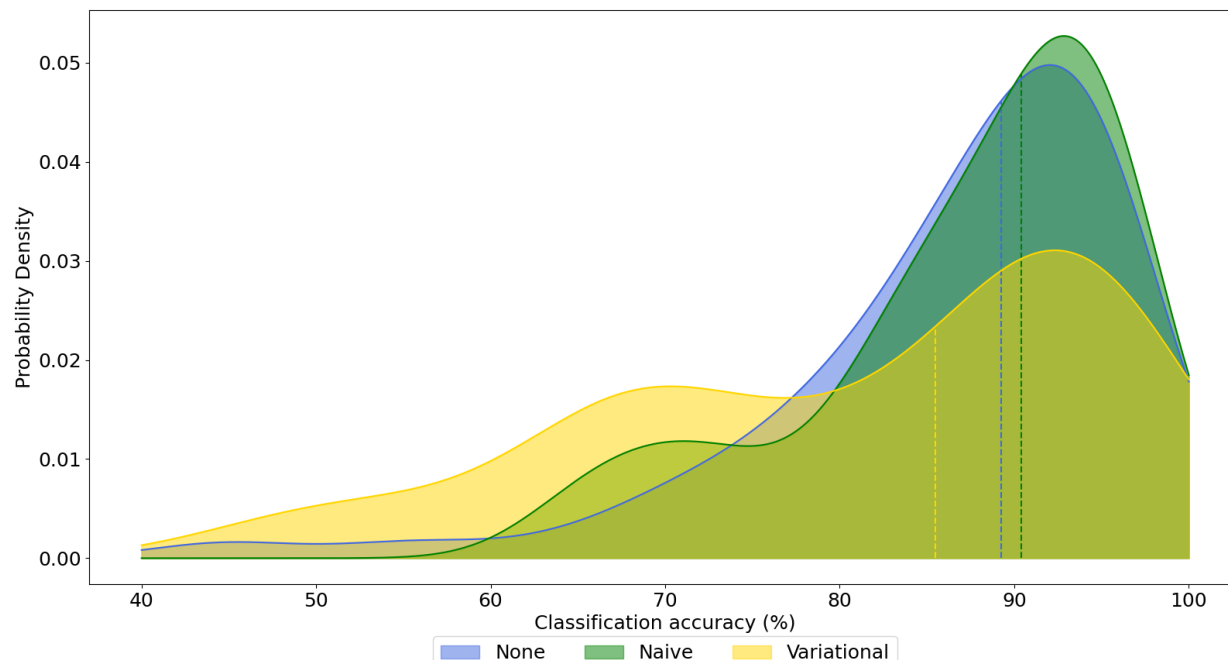


Figure B7-3 : Dropout impact. The dashed line corresponds to the median of the same colored distribution.

Both using naive dropout, or none shows a narrower distribution around a median of ~90 %. The naive dropout has more instances where the accuracy was under 70 %. As for the variational dropout, the accuracy is further spread out resulting in a lower median.

Tableau B.5 : Dropout impact. A bold p-value indicates the difference with the best median is statistically meaningful ($p < 0.01$).

	None	Naive	Variational
Median	89.23 %	90.4 %	85.49 %
Difference in median	-1.17 %	-	-4.91 %
Statistical difference (p-value)	0.860	-	< 0.001

The best performance of 90.4 % is from using naive dropout. There is no significant improvement over not using any dropout and the gain in accuracy is 1.17 %. Otherwise, when compared to naive dropout, variational dropout shows a decrease in accuracy of 4.91 % and there is a significant difference.

Learning rate

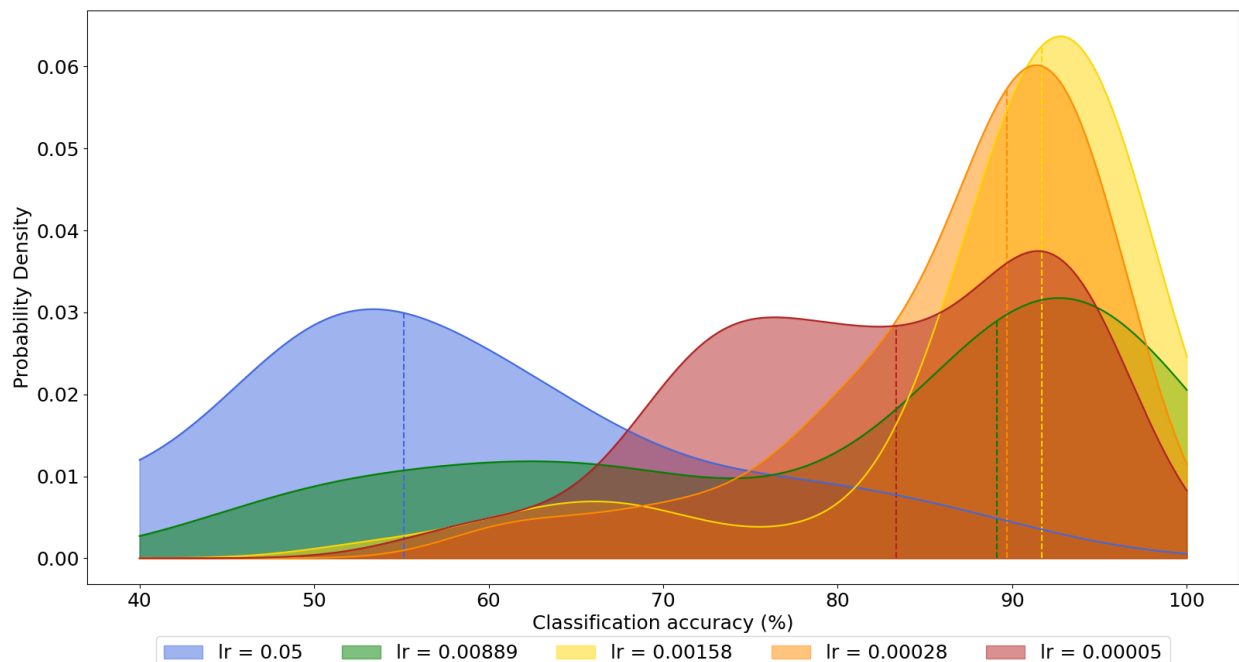


Figure B.7-4 : Learning rate (*lr*) impact. The dashed line corresponds to the median of the same colored distribution.

The learning rate displays different probability densities based on the selected value. Both $lr = 0.00158$ and $lr = 0.00028$ shows a narrow distribution around a median of ~90 % with some instances of configuration performing under 70 %. The $lr = 0.00889$ shows the widest distribution even though its median is close to $lr = 0.00028$. Both $lr = 0.05$ and $lr = 0.00005$ displays a wide distribution around 55 % and 83 % respectively.

Tableau B.6 : Learning rate impact. A bold p-value indicates the difference with the best median is statistically meaningful ($p < 0.01$).

Learning rate	0.05	0.00889	0.00158	0.00028	0.00005
Median	55.14 %	89.12 %	91.70 %	89.69 %	83.35 %
Difference in median	-36.56 %	-2.59 %	-	-2.01 %	-8.35 %
Statistical difference (p-value)	< 0.001	0.002	-	< 0.001	< 0.001

The best learning rate value is 0.00158 with a median of 91.70 %. All other tested values show a significant difference that ranges from 2.01 % to 36.56 % in accuracy.

Self-Attention layer

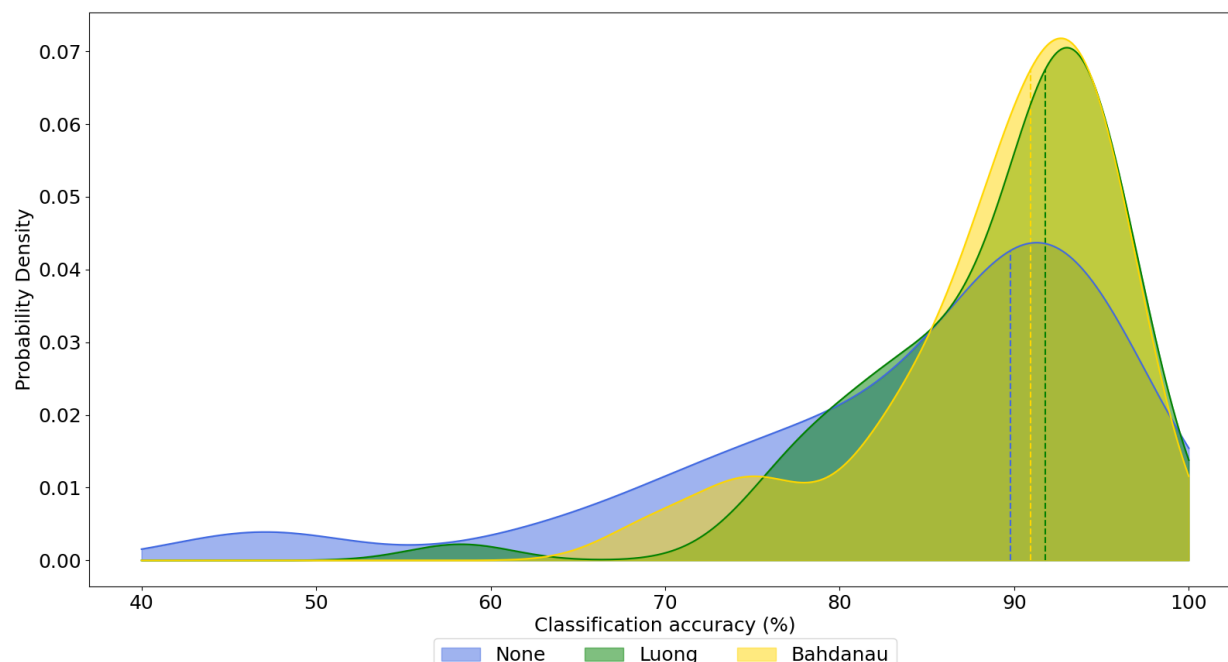


Figure B.7-5 : Self-Attention layer impact. The dashed line corresponds to the median of the same colored distribution.

All three options for the self-attention layer have a similar median around 90 %. When not using any self-attention layer, the distribution of accuracy is wide and contained over ~60%. The distributions of both self-attention methods are mostly contained over 70 % with a narrower peak around 90 %.

Tableau B.7 : Self-Attention layer impact. A bold p-value indicates the difference with the best median is statistically meaningful ($p < 0.01$).

	None	Luong	Bahdanau
Median	89.78 %	91.78 %	90.95 %
Difference in median	-2.01 %	-	-0.83 %
Statistical difference (p-value)	< 0.001	-	0.556

The best self-attention method is Luong's with a median of 91.78 %, but it is not significantly better than Bahdanau's self-attention. Although, not using any self-attention layer decreases the accuracy by 2.01 % and is significantly worse than Luong.

Window increment

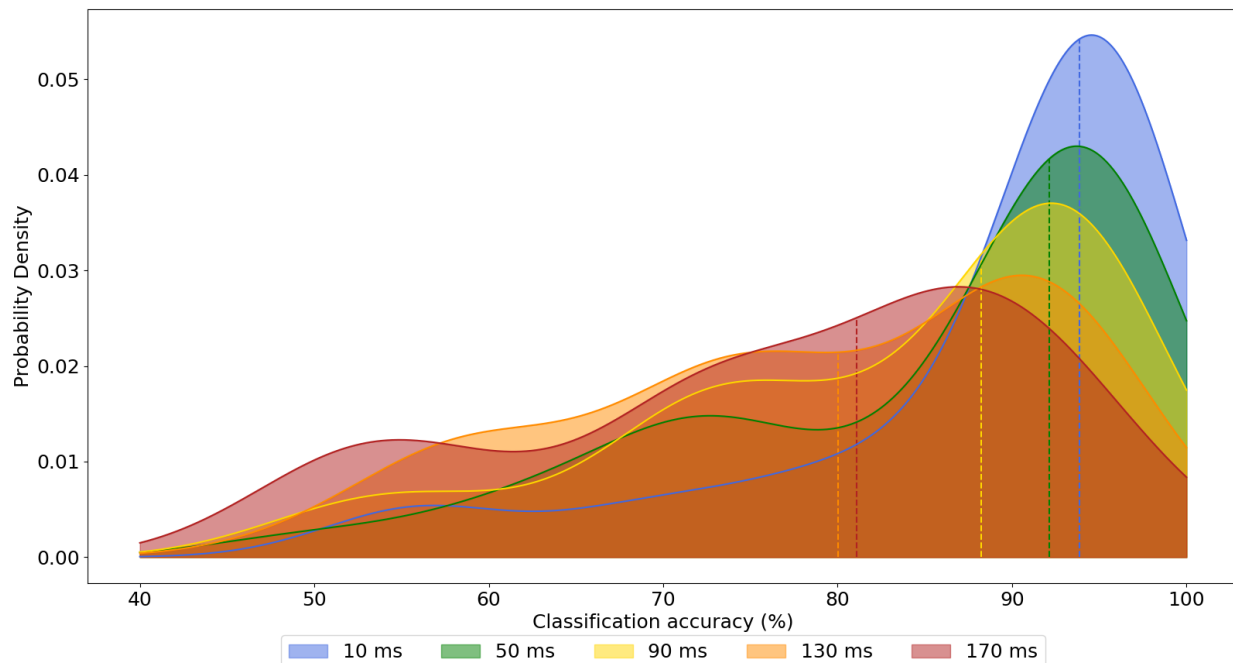


Figure B.7-6 : Window increment impact. The dashed line corresponds to the median of the same colored distribution.

Overall, the distribution of all the tested values for the window increment is widespread over the accuracy. Although it looks like as the window increment decrease, the narrower is the peak in the higher accuracies.

Tableau B.8 : Window increment impact. A bold p-value indicates the difference with the best median is statistically meaningful ($p < 0.01$).

	10 ms	50 ms	90 ms	130 ms	170 ms
Median	93.87 %	92.15 %	88.23 %	80.01 %	81.10 %
Difference in median	-	-1.72 %	-5.64 %	-13.86 %	-12.77 %
Statistical difference (p-value)	-	0.002	< 0.001	< 0.001	< 0.001

The best window increment value is 10 ms with a median of 93.87 %. It is significantly better than the other values and the gain in accuracy ranges from 1.72 % to 13.86 %.

Window size

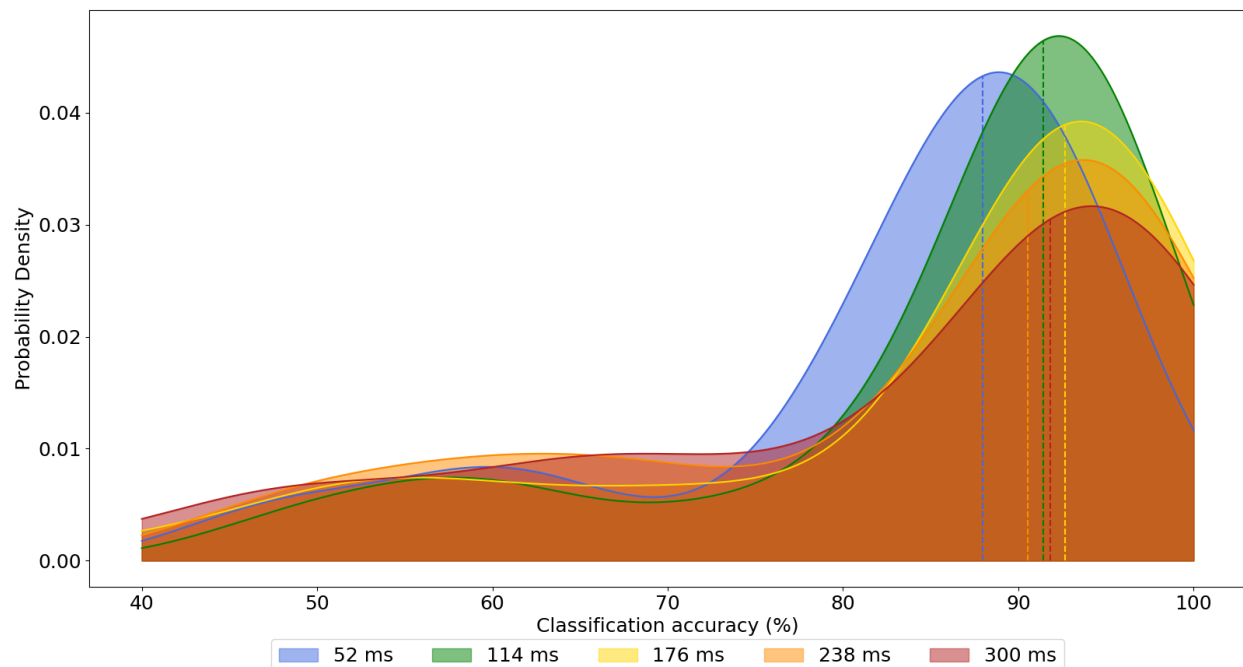


Figure B.7-7 : Window size impact. The dashed line corresponds to the median of the same colored distribution.

The distribution of all the tested value for the window size is quite similar. They all display instances of low accuracy, but overall, a narrow peak can be seen around 90 %. The window size of 52 ms has its peak performance shifted to lower accuracy.

Tableau B.9 : Window size impact. A bold p-value indicates the difference with the best median is statistically meaningful ($p < 0.01$).

Window increment	52 ms	114 ms	176 ms	238 ms	300 ms
Median	87.94 %	91.43 %	92.69 %	90.56 %	91.83 %
Difference in median	-4.74 %	-1.26 %	-	-2.13 %	-0.83 %
Statistical difference (p-value)	< 0.001	0.596	-	0.746	0.780

While the window size with the best performance is 176 ms, there are no significant differences with a window size of 114, 238, and 300 ms. Only the lowest window size of 52 ms is 4.74 % lower in accuracy and displays a significant difference.