Sequence metric learning : Application to human activity recognition

Paul Compagnon

N°d'ordre NNT : 2021LYSEI033

THESE de DOCTORAT DE L'UNIVERSITE DE LYON
opérée au sein de
INSA de Lyon

Ecole Doctorale N° 512 InfoMaths

Spécialité : Informatique

Soutenue publiquement le 27/05/2021, par :
Paul Compagnon

Sequence Metric Learning : Application to Human Activity Recognition

---

Devant le jury composé de :

Oukhellou, Latifa Directrice de recherche, Université Gustave Eiffel, Rapporteure
Thome, Nicolas, Professeur des universités, Conservatoire National des Arts et Métiers, Rapporteur
Chateau, Thierry, Professeur des Université, Université de Clermont Auvergne, Examinateur
Douzal-Chouakria, Ahlame, Maitre de conférence HDR, Université Grenoble Alpes, Examinatrice
Habrard, Amaury, Professeur des universités, Université Jean Monnet, Examinateur
Garcia, Christophe, Professeur des université, INSA de Lyon, Directeur de thèse
Duffner, Stefan, Maitre de conférence HDR, INSA de Lyon, Co-directeur de thèse
Lefebvre, Grégoire, Chercheur, Orange Labs, Co-directeur de thèse

Département FEDORA – INSA Lyon - Ecoles Doctorales

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
CHIMIE	**CHIMIE DE LYON** **https://www.edchimie-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage secretariat@edchimie-lyon.fr	**M. Stéphane DANIELE** C2P2-CPE LYON-UMR 5265 Bâtiment F308, BP 2077 43 Boulevard du 11 novembre 1918 69616 Villeurbanne **directeur@edchimie-lyon.fr**
E.E.A.	**ÉLECTRONIQUE, ÉLECTROTECHNIQUE, AUTOMATIQUE** **https://edeea.universite-lyon.fr** Sec. : Stéphanie CAUVIN Bâtiment Direction INSA Lyon Tél : 04.72.43.71.70 secretariat.edeea@insa-lyon.fr	**M. Philippe DELACHARTRE** INSA LYON Laboratoire CREATIS Bâtiment Blaise Pascal, 7 avenue Jean Capelle 69621 Villeurbanne CEDEX Tél : 04.72.43.88.63 **philippe.delachartre@insa-lyon.fr**
E2M2	**ÉVOLUTION, ÉCOSYSTÈME, MICROBIOLOGIE, MODÉLISATION** **http://e2m2.universite-lyon.fr** Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 secretariat.e2m2@univ-lyon1.fr	**M. Philippe NORMAND** Université Claude Bernard Lyon 1 UMR 5557 Lab. d'Ecologie Microbienne Bâtiment Mendel 43, boulevard du 11 Novembre 1918 69 622 Villeurbanne CEDEX **philippe.normand@univ-lyon1.fr**
EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTÉ** **http://ediss.universite-lyon.fr** Sec. : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 secretariat.ediss@univ-lyon1.fr	**Mme Sylvie RICARD-BLUM** Institut de Chimie et Biochimie Moléculaires et Supramoléculaires (ICBMS) - UMR 5246 CNRS - Université Lyon 1 Bâtiment Raulin - 2ème étage Nord 43 Boulevard du 11 novembre 1918 69622 Villeurbanne Cedex Tél : +33(0)4 72 44 82 32 **sylvie.ricard-blum@univ-lyon1.fr**
INFOMATHS	**INFORMATIQUE ET MATHÉMATIQUES** **http://edinfomaths.universite-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage Tél : 04.72.43.80.46 infomaths@univ-lyon1.fr	**M. Hamamache KHEDDOUCI** Université Claude Bernard Lyon 1 Bât. Nautibus 43, Boulevard du 11 novembre 1918 69 622 Villeurbanne Cedex France Tél : 04.72.44.83.69 **hamamache.kheddouci@univ-lyon1.fr**
Matériaux	**MATÉRIAUX DE LYON** **http://ed34.universite-lyon.fr** Sec. : Yann DE ORDENANA Tél : 04.72.18.62.44 yann.de-ordenana@ec-lyon.fr	**M. Stéphane BENAYOUN** Ecole Centrale de Lyon Laboratoire LTDS 36 avenue Guy de Collongue 69134 Ecully CEDEX Tél : 04.72.18.64.37 **stephane.benayoun@ec-lyon.fr**
MEGA	**MÉCANIQUE, ÉNERGÉTIQUE, GÉNIE CIVIL, ACOUSTIQUE** **http://edmega.universite-lyon.fr** Sec. : Stéphanie CAUVIN Tél : 04.72.43.71.70 Bâtiment Direction INSA Lyon mega@insa-lyon.fr	**M. Jocelyn BONJOUR** INSA Lyon Laboratoire CETHIL Bâtiment Sadi-Carnot 9, rue de la Physique 69621 Villeurbanne CEDEX **jocelyn.bonjour@insa-lyon.fr**
ScSo	**ScSo*** **https://edsciencessociales.universite-lyon.fr** Sec. : Mélina FAVETON INSA : J.Y. TOUSSAINT Tél : 04.78.69.77.79 melina.faveton@univ-lyon2.fr	**M. Christian MONTES** Université Lumière Lyon 2 86 Rue Pasteur 69365 Lyon CEDEX 07 **christian.montes@univ-lyon2.fr**

*ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

ABSTRACT

This thesis aims at proposing new neural network approaches to retrieve the habitual behaviors of fragile people in order to provide them with a monitoring at home while respecting their privacy and avoiding stigmatization. In this perspective, we concentrate on the exploitation of wearable motion sensor data (accelerometer, gyrometer, magnetometer, barometer etc.) which are nowadays easily embedded into smartphones and smartwatches. In a first contribution, we propose to employ few-shot learning with an architecture called matching network [217] to learn a personalized and flexible activity recognition model. This model learn to recognize a new class from just one or few new samples since it matches rather than classify. Therefore this model allows to better handle the large variety of activities one can do in one day while alleviating the burden of data labeling. In a second part, we advocate for a change of perspectives by proposing to retrieve recurrent unlabeled activity patterns called routines instead of precise activities. We propose a formalization of the concept of routine with the notion of almost-periodic functions [26] which prompts us to employ sequence metric learning. We propose a neural network architecture based on robust sequence representation learning with a Sequence-to-Sequence model [197] and metric learning with a siamese network [30]. No activity labels are used to train the model by setting up an equivalence constraint with the data timestamps. We propose to identify the routines with a spectral clustering and to evaluate the whole routine retrieval process with information-theoretic clustering scores [215]. The last contribution of this thesis is a new neural network model for sequence metric learning called Coupled Gated Recurrent Unit. This model has been conceived by taking inspiration from the dynamical system theory and notably the concept of synchronization. We propose to improve the siamese gating

recurrent unit architecture by implementing a coupling which should allow it to better process the hard samples. We finally experiment this architecture to recognize activities and retrieve routines.

RÉSUMÉ

Cette thèse a pour objet d'étude la conception de modèle d'apprentissage automatiques pour la reconnaissance d'activités et plus particulièrement la reconnaissance de comportements habituels des usagers (routines), ceci à des fins de suivi médical à leur domicile des personnes fragiles (actimétrie) en respectant leur vie privée et en évitant la stigmatisation. Pour cela, nous nous concentrons sur l'exploitation des données produites par des capteurs de mouvement portés (accéléromètre, gyroscope magnétomètre, etc.) présents dans les téléphones mais aussi dans les montres connectées, des objets de la vie quotidienne. Nous proposons dans cette thèse des modèles de réseaux de neurones pour la reconnaissance personnalisée d'activités ou de routines qui ont pour point commun d'utiliser l'apprentissage de métrique. L'apprentissage de métrique peut la plupart du temps être réalisé sans étiquette de classe mais plutôt grâce à des contraintes d'équivalence qui définissent les échantillons similaires et dissimilaires. Cela permet d'envisager des modèles flexibles voire semi-supervisés capables de s'adapter facilement à la vie quotidienne de l'utilisateur.

Dans un premier chapitre, nous nous intéressons à la reconnaissance d'activité personnalisée car celle-ci, si elle promet des performances plus élevées déterminantes pour les applications médicales, souffre également de reporter tout le poids de la production des données et de leur étiquetage sur un seul utilisateur. Pour palier à ce problème, nous proposons d'utiliser un modèle de *few-shot learning* appelé *matching network* [217], pour apprendre un modèle personnalisé et flexible de reconnaissance d'activités à partir de peu de séquences inertielles. Ce modèle s'adapte à partir de quelques exemples à une nouvelle classe et permet donc de mieux gérer les activités très variées qu'une personne peut

réaliser dans une journée tout en étant rapidement déployable. Grâce à quelques modifications du modèle initial pour l'adapter aux séquences notamment par le biais d'un modèle *Sequence to Sequence* [197], nous testons cette architecture en reconnaissance d'activités sur deux datasets contenant respectivement 6 et 12 activités. Ces tests sont réalisés sur des données provenant de classes sur lesquelles le modèle n'a pas été entraîné. Sur le deuxième dataset, si les résultats se montrent plus faibles que l'état de l'art, ils ont été obtenus avec une quantité extrêmement limitée de données pour chaque utilisateur (une ou deux séquences sur chaque classe) ce qui montre la pertinence de notre approche pour mettre en place rapidement un service d'actimétrie.

Dans une seconde partie, nous portons notre attention sur le concept de routines, c'est à dire sur les activités habituelles de l'utilisateur que nous pensons pouvoir être détectées sans étiquette, uniquement grâce à leur récurrence. Nous proposons une formalisation mathématique du concept de routine à partir des fonctions presque-périodiques [26] et nous en déduisons une méthode de reconnaissance de celles-ci grâce à l'apprentissage de métrique. Nous réalisons ensuite un état de l'art de l'apprentissage de métrique dans lequel nous nous attardons notamment sur les approches pour les séquences et sur les fonctions de coût qui peuvent être utilisées pour entraîner des réseaux de neurones à cette tâche. Nous proposons un modèle qui combine apprentissage de représentation avec un modèle *Sequence to Sequence* [197] et apprentissage de métrique sous la forme d'un réseau de neurones siamois [30], le tout entraînable de bout en bout. Plusieurs fonctions de coût peuvent être utilisées pour apprendre la métrique et notamment celle nommée *KISSME* [69]. Le modèle est appris sans étiquette d'activité uniquement grâce à l'horodatage des données qui constitue la contrainte d'équivalence: les échantillons acquis aux mêmes heures mais des jours différents sont considérés comme similaires. Nous proposons ensuite d'identifier les routines grâce à un *clustering* spectral. Nous testons cette approche sur un dataset contenant trois jours de données continues d'un utilisateur dans son environnement habituel. La validation du modèle se fait grâce à des scores de *clustering* issus de la théorie de l'information [215] et une analyse visuelle qui semblent montrer la pertinence de la modélisation et de la procédure de reconnaissance de routines proposée ainsi que de l'architecture, même si la version non-entraînée de bout en bout atteint de meilleures performances.

Dans un troisième chapitre, nous proposons un nouveau modèle

de réseau de neurones siamois récurrents dit couplé appelé GRU couplé. Ce modèle a été conçu en s'inspirant de la théorie des systèmes dynamiques et notamment du concept de synchronisation. En effet, les deux réseaux d'un modèle siamois récurent s'apparentent à deux systèmes dynamiques identiques. La théorie des systèmes dynamiques nous enseigne alors qu'ils peuvent toujours être synchronisés, c'est à dire forcés à évoluer de façon similaire [70] si un couplage, un échange d'information, est appliqué entre les deux. Cet état s'apparente dans le cas de l'apprentissage de métrique à une faible distance entre les deux séquences de sortie d'un réseau siamois récurent. Nous étudions l'intégration d'un couplage dans le but d'améliorer l'architecture *siamese GRU*, notamment en ce qui concerne les exemples difficiles (*hard positive/negative samples*). Cette intégration se fait par l'intermédiaire d'une nouvelle porte dite de couplage dans l'architecture *siamese GRU* [46] qui vise à contrôler l'échange d'information entre les deux parties du réseau. Nous montrons ensuite que cette architecture est dérivable et qu'elle peut donc être entraînée par descente de gradient. Nous expérimentons ce modèle en reconnaissance d'activités sur deux datasets où elle atteint des performances plus élevées que sa contrepartie sans couplage. Nous la testons également pour ce qui est de la reconnaissance de routines avec des résultats plus mitigés.

Dans le dernier chapitre, nous concluons cette thèse par une synthèse des contributions avant de relever quelques limitations du travail présenté. Nous proposons des pistes d'amélioration notamment pour l'architecture GRU couplé basées sur l'apprentissage de métrique virtuel [161] et les métriques de synchronisation généralisée (*generalized synchronization* [170]). Ces métriques sont non-dérivables car faisant intervenir explicitement les *timestamps* et nous proposons d'utiliser le mécanisme d'attention pour y remédier [212].

REMERCIEMENTS

Je remercie Dr. Latifa Oukhellou, HDR, et Pr. Nicolas Thome d'avoir accepté d'être les rapporteurs de ce manuscrit ainsi que Pr. Thierry Chateau, Dr. Ahlame Douzal Chouakria, HDR, et Pr. Amaury Habrard, les examinateurs, pour l'intérêt qu'ils ont porté à mes travaux. Merci pour votre implication dans le jury de cette thèse.
J'aimerais avoir, pour mes trois encadrants, Christophe, Grégoire et Stefan des remerciements chaleureux et sincères pour leur confiance, leurs enseignements et leur soutient dans les bons comme les mauvais jours. J'espère avoir de nouveau l'occasion de travailler avec vous!
Je souhaite ensuite remercier mes collègues d'Orange. Les membres de mon équipe et en particulier Erwan, Hervé, Pierre-Yves ainsi que Thomas. Mes collègues chercheurs avec qui j'ai échangé et débattu: Dihia, Guillaume, Louis-Adrien, Mahdi, Naji, Pauline, Quentin et Wissal. Un merci à Julien et Samuel dont les thèses m'ont servi de références. Enfin comment oublier Esteban et Maël : nous serons bientôt tous les trois *Young Doctors*!
Des remerciements émus pour ma famille qui m'a accompagné depuis toujours jusqu'à cet ultime étape de ma scolarité, mon frère Maxence, mes parents Marie-Pierre et Philippe, mes grands-parents André, Bernadette, Michel et Simone, ainsi que Manou, Yves, Anne, Nino, Mikaël et Michel. J'ai une pensée particulière pour mon grand-père André, emporté pendant l'écriture de ce manuscrit.
Pour les membres du Chalet, Alexy, Amaury, Carmine et Thibaut, des remerciements magiques pour notre amitié qui, la science l'affirme, devrait maintenant durer toute la vie. J'aimerais avoir un mot particulier pour Amaury dont l'amitié fraternelle a contribué à l'écriture de cette thèse de façon plus indirecte que ses relectures.
Un merci différent pour mes amis de l'INSA. En particulier à Adrien,

mon compagnon de voyage et de casino. A Ludo et Nadia qui sont toujours présents, pour les bons et les mauvais moments. A Orianne pour son soutien et sa motivation communicative. A TongJia, pour toujours mon premier coloc. A Vinciane, pour une amitié stimulante et sans cesse renouvelée. Des smart-remerciements pour Côme, Rani et Théo qui ont assisté à la naissance des réseaux de neurones Compagnons.
Enfin, un merci pour mes amis photographes Nicolas et Rémi qui ont artistiquement animé ma vie grenobloise.
Merci à tous ceux qui ont contribué à l'écriture de ce manuscrit.

CONTENTS

ACRONYMS

k-NN k-Nearest-Neighbors. 36–38, 75, 77, 81

ADL Activities of Daily Living. 27, 28, 30, 31, 35, 41, 53, 56, 67

AI Artificial Intelligence. 25, 26, 28–30, 33

AMI Adjusted Mutual Information. 96, 98, 99, 129, 135, 137, 138

ANN Artificial Neural Network. 37, 44

ARMA AutoRegressive-Moving-Average. 113

AROMA human Activity RecognitiOn using deep Multi-tAsk learning. 39

BPTT Back Propagation Through Time. 45, 119

CGRU Coupled Gated Recurrent Unit. 117, 118, 121–131, 135–138, 141

CNN Convolutional Neural Networks. 38, 39, 48, 54, 79, 85, 91

CRF Conditional Random Field. 39

CRL Cosine Reconstruction Loss. 94, 98–100

DECADE Deep ExpeCted Alignment DistancE. 84

DTW Dynamic Time Warping. 75, 80–84, 90, 99–101, 105, 126, 129, 134

ESN Echo State Network. 83, 84, 136

GDPR General Data Protection Regulation. 29

GPU Graphical Processor Unit. 28, 37, 128, 129

GRU Gated Recurrent Unit. 42, 46, 47, 49, 51, 55, 56, 59–61, 64, 108, 112–114, 116–118, 130, 135

HAR Human Activity Recognition. 31, 35–38, 40–42, 48, 55, 69

HMM Hidden Markov Models. 31, 54, 113

IMU Inertial Measurement Unit. 52, 53

IoT Internet of Things. 28–30

KISSME Keep It Simple and Straightforward MEtric learning. 88–90, 93, 97, 99–101, 104–106, 134

LMNN Large Margin Nearest Neighbors. 77

LSTM Long-Short Term Memory neural network. 38, 39, 46, 54, 79, 84, 91, 93, 112, 113, 116, 117, 130, 134

LTMM Long-Term Movement Monitoring. 96, 98–101, 103, 105, 122, 129, 130, 136, 142, 143

MAP Mean Average Precision. 126, 127

MFNN Mutual False Nearest Neighbor. 139

MIST MIxed hiSTory. 54, 63

MLP MultiLayer Perceptron. 42, 43

MSE Mean Squared Error. 43, 47, 57, 58, 94, 98–100, 106

NLP Natural Language Processing. 71, 72, 85

NMI Normalized Mutual Information. 95, 96, 98, 99, 106, 129, 135, 137, 138

ODE Ordinary Differential Equations. 112

OPW Order-Preserving Wasserstein. 82, 83, 126

PCA Principal Component Analysis. 38, 89

PMI Periodic Mutual Information. 138

PSD Positive Semi-Definite. 22, 77, 83

RBM Restricted Boltzmann Machine. 38

ReLU Rectified Linear Unit. 44

RNN Recurrent Neural Network. 38, 42, 44–47, 54, 73, 83, 90, 106, 107, 111, 112, 114–116, 118

RVSML Regressive Virtual Sequence Metric Learning. 126, 127

Seq2Seq Sequence-to-Sequence. 35, 56, 70, 82, 91–93, 106, 134

SGRU Siamese Gated Recurrent Unit. 114, 118, 121–131, 135, 136

SHL University of Sussex-Huawei Locomotion. 135

SLSTM Siamese Long-Short Term Memory. 99

SS2S Siamese Sequence-to-Sequence. 91–93, 99–101, 105, 106, 129, 134, 135

SSMN Sequence-to-Sequence Matching Network. 42, 50, 51, 57, 58, 60, 62, 64–68, 134, 135

STE Symbolic Transfer Entropy. 137

SVM Support Vector Machine. 37, 38, 41

MATHEMATICAL NOTATIONS

Notation	Definition
net_W	a neural network parameterized by W
S	sequence or time series of length T
$S(t)$	point at time t
$\hat{S}$	output sequence
σ	sigmoid function
h_t	hidden state of RNN
x	input vector
y	target output vector or representation
$\hat{y}$	predicted output
l	class label or similar/dissimilar label
$\circ$	Hadamard product
$\mathcal{B}$	batch inputs
$\mathcal{P}$	ensemble of positive/similar pairs of samples
$\mathcal{P}_c$	ensemble of positive/similar pairs of samples of the same class c
$\mathcal{N}$	ensemble of negative/dissimilar pairs of samples
$\mathcal{L}$	a loss function
m	margin
$[.]_+$	Hinge loss
a, p^+, p^-	anchor, sample similar to anchor and sample dissimilar to the anchor.
d	a distance, whatever the type if not precised

M	a positive semi-definite matrix defining a Mahalanobis metric, by definition $M = L^T L$
$\mathbb{S}_+^n$	set of Positive Semi-Definite (PSD) matrices of order n
ϕ	dynamical system evolution rule
$\varphi(t_0)$	trajectory from initial condition t_0
Z	a dynamical system composed of two subsystems: X and Y

PUBLICATIONS ASSOCIATED WITH THIS THESIS

Publication in Peer-Reviewed Journals

Compagnon, P., Lefebvre, G., Duffner, S., and Garcia, C. "Learning personalized ADL recognition models from few raw data". In: *Artificial Intelligence in Medicine* vol. 107 (2020), p. 101916

Publications in Peer-Reviewed Conferences

Compagnon, P., Lefebvre, G., Duffner, S., and Garcia, C. "Personalized posture and fall classification with shallow gated recurrent units". In: *2019 IEEE 32nd International Symposium on Computer-Based Medical Systems (CBMS)*. IEEE. 2019, pp. 114–119

Compagnon, P., Lefebvre, G., Duffner, S., and Garcia, C. "Routine modeling with time series metric learning". In: *International Conference on Artificial Neural Networks*. Springer. 2019, pp. 579–592

Submission in Peer-Reviewed Conferences

Compagnon, P., Lefebvre, G., Duffner, S., and Garcia, C. "Sequence Metric Learning as Synchronization of Recurrent Neural Networks". In: *Submitted to IJCNN* (2021)

Patent Applications at the French National Institute for Intellectual Property

Lefebvre, G. and Compagnon, P. “Procédé d’évaluation de l’activité corporelle d’un utilisateur”. 2019

Compagnon, P. and Lefebvre, G. “Procédé d’alerte sur l’autonomie d’une personne par l’analyse de ses changements de routines”. 2019

Lefebvre, G. and Compagnon, P. “Reconnaissance de modes de déplacement par capteurs de mouvement”. 2020

CHAPTER 1

INTRODUCTION

The optimistic predictions made by Marvin Minsky on the future advances of Artificial Intelligence (AI) have sometimes been treated with sarcasms among the scientific community. Therefore beginning a thesis on one of his quotations may be considered hazardous, at least daring. Nevertheless, in a 1991 paper for AI Magazine [141], he wrote the following statement:

> Why can we build robots that compete with highly trained workers to assemble intricate machinery in factories but not robots that can help with ordinary housework? It is because the conditions in factories are constrained, and the objects and activities of everyday life are too endlessly varied to be described by precise, logical definitions and deductions.

Minsky refers in the last sentence to the symbolic approach of AI, also named by Haugeland *Good Old-Fashioned Artificial Intelligence* [87]. Examples of such AI are expert systems which reason using *if...then* rules applied to knowledge bases [221]. Recent AI are supposedly the opposite of symbolic: they are connectionist, or to say it straight in 2021, based on neural networks which learn correlations and representations from data. However, we argue that they still suffer the same limitations as symbolic approaches, at least in supervised settings and that what Minsky wrote 30 years ago on that matter still holds. How indeed handle the endless variations of the daily life of an individual with neural networks trained on clean laboratory data to classify a dozen of activities ? The purpose of this thesis is to present new neural network approaches able to bridge this gap, designed to work in daily open environments. As a consequence, we advocate for a switch of perspectives from recognizing

precise labeled activities to retrieve routines: unlabeled recurrent data patterns associated with physical movements. The proposed algorithms are to be applied to medical issues, especially long-term monitoring of fragile people.

1.1 Home Medical Services for Fragile People with Actigraphy

The AI models presented in this thesis are to be used for eHealth services. EHealth can be defined as an "emerging field in the intersection of medical informatics, public health and business, referring to health services and information delivered or enhanced through the Internet and related technologies" [68]. Its applications, which are often seen as beneficial for both patients and medical staff [154], encompass: basic prevention with mobile apps, computerized medical files, doctor's office management software, hospital information systems, computer assisted surgery, machine learning assisted diagnostic but also home support and particularly with actigraphy.

1.1.1 The Growing Need for Medical Assistance at Home

As life expectancy increases, more and more elderly people show difficulties in their every day life, and allowing them to stay at home is a social and public health issue[1]. By the year 2050, 2 billion people will be aged 60 and older, a number doubled compared to 2020. Many elderly even struggle performing basic need activities. They are also particularly exposed to chronic diseases: diabetes, cancer, psychological and cognitive disorders, heart diseases, Parkinson and Alzheimer, etc. [27]. They are finally vulnerable in simple daily life activities where they could fall, make a wrong move or loose attention[2]. Furthermore, over 1 billion people in the world live with a form of handicap, between around 4% suffering severe disabilities[3]. In addition to these difficulties, these people, particularly elderly people, can also suffer from loneliness and abandonment. Finally, home convalescence is an interesting option in numerous cases to

[1] https://www.who.int/health-topics/ageing
[2] https://www.who.int/news-room/fact-sheets/detail/falls
[3] https://www.who.int/news-room/fact-sheets/detail/disability-and-health

improve the recovery of the patient which requires, nevertheless, some precautions [237].

One eHealth service that could benefit all these populations is daily activity monitoring [162]. The process of recording the every day life activities of a subject using sensors (inertial sensors, for instance) is called actigraphy. Instead of organizing regular visits at the hospital, the patient can be monitored in his/her house with several upsides: it improves the quality of life of the patient and shortens hospital stays while facilitating the diagnosis as important medical data can be collected in the usual environment of the patient. Clinical visits are obviously indispensable but can only take a snapshot of the patient's condition and may occur too late during the disease development [9]. Among applications of actigraphy we can list: monitoring and diagnosis (prevention of emergencies, assistance for people with cognitive disorders, people with chronic condition, etc.), as already mentioned but also rehabilitation, correlation between movement and emotion, child and elderly care. Moreover, several symptoms that appear on elderly persons can be observed with an actigraphy system, for instance: bone fragility, difficulty to stand or to make efforts, sensibility to infection and medicine, etc. [55].

In this work, we are particularly interested in providing long-term monitoring of autonomy for semi-dependent people. To measure the activity of a person, energy expenditure is widely recognized as the best way [119] but it is in practice difficult to set up. It can be approximated by a physical activity measurement which gives a good idea of the autonomy. This type of monitoring could be performed with actigraphy. Another way to evaluate autonomy is by looking at several criteria related to the Activities of Daily Living (ADL) [108] (e.g. having lunch, watching television etc.) or the Instrumental ADL [122] (e.g. using of the telephone, food preparation, household, etc.). This evaluation appears as a pertinent factor for the clinical surveillance of fragile people but requires, more than just actigraphy, activity recognition which has become incredibly easier to perform during the last decade.

1.1.2 New Possibilities

Two major digital revolutions greatly facilitate the setup of actigraphy services. Firstly, the advent of 5G telecommunication networks will complete the third web revolution and allow it to become the "web

4.0", the Web or Internet of Things (IoT)[4]. It can be seen as a new paradigm which can also take different names: ubiquitous computing, pervasive computing or ambient intelligence. The "things", or objects which should become omnipresent, are more and more diverse: we naturally find sensors for temperature, light, pollution, sound level, presence, etc. but also smart assistants disguised as music speaker and daily life objects enhanced with numeric capacities (e.g. fridge automatically buying some products when there are no more). Among fields particularly impacted by internet of things, we can list: eHealth, automation, sport, etc. and more globally, one can talk about Industry 4.0, smart home, smart city, smart transport, etc. [228]. Not only the value of an individual object can be greatly increased by miniaturized and reliable microprocessors, memory units or power supplies but also several objects can communicate and share data to produce innovative services.

Among those things, we especially bear interest for embedded motion sensors (accelerometer, gyrometer, magnetometer, barometer, among some others less common) whose precision has greatly increased during these last two decades. Nowadays, smartphone sensors are perfectly adapted for medical use [142]. Smartwatches constitute an interesting alternative to collect biometric data, including cardiac frequency, and motion data: this is testified by the ubiquity of sport watches [242] which are conceived to monitor the person the whole day and during the night. These daily life connected objects are judged less intrusive at home and can be carried without getting the person stigmatized [36]. They offer the upside compared to smart home technologies also permitted by IoT of targeting precisely the wearer of the sensors, even outside her/his house.

With abundance of data coming from more and more diverse sources also come the second revolution: (better) AI. The acquired data can be processed with machine learning algorithms, especially deep learning [123] algorithms, to perform ADL or posture classification and prediction. This way, actigraphy monitoring of ADL and thus autonomy evaluation could be performed automatically. To be able to eventually equip people with such systems, high classification accuracy is required, particularly for critical events such as falls. This has been rendered possible by the progresses made to train deep neural networks with very large amount of data notably by running them on Graphical Processor Unit (GPU)

[4]The web 1.0 is the static web connecting information. The web 2.0 is the social web, connecting people. The web 3.0 is the semantic web or web of data.

[115]. Obviously, with these two game-changing technologies come new concerns which should also be addressed.

1.1.3 New Issues to Overcome

It is from now on impossible to ignore the immense ethical and moral challenges posed by the massive data collecting permitted by IoT and the exploitation of these data with AI algorithms. Those challenges include privacy preservation, biased algorithms toward certain populations (e.g. crime prediction algorithms biased toward Afroamerican populations[139]), security risks inherent to distributed infrastructures but also political concerns due to government being technically able to control its whole population at any time [129]. Thus, pervasive computing systems must be associated to strong security and privacy preserving measures. This is of course even more true for eHealth innovations as personal healthcare information are particularly sensitive [132]. Moreover, compliance with legal regulations, such as the General Data Protection Regulation (GDPR)[5] in Europe, is a necessity.

Apart from these ethical considerations, the place of the human in the caring process is primordial, especially because some fragile people may already suffer from extreme loneliness. The acceptability by the user of these new technologies should thus be discussed. A study conducted by Tuisku et al. [206] to evaluate the level of acceptability of a robot by fragile people reveals that a majority of them has negative views of it notably because of the fear to see their nurse replaced by a robot. On the other hand, from the caregiver perspective, it seems that they are perceived as useful tools notably because the number of semi-dependent people increases rapidly. These observations are further confirmed in another study by Melkas et al. [140] who studied the impact of the implementation of a care robot. Overall, they concluded that people seemed attracted and stimulated by the novelty of the robot. Regarding wearable sensor based systems, Talukder et al. [200] studied the factors favoring their adoption by elderly and concluded that the two most important were the expectancy of the technology to be beneficial and the influence of the social environment. Against intuition, they observed that ease of use did not have a significant effect. We conclude from these studies that it is overall important to design eHealth services which work together with the medical staff to improve the service provided to the user while preserving what is often his/her only social environment.

[5] https://eugdpr.org/the-regulation/

We would like finally to write some words about the environmental impacts of those technologies which have the potential to become disastrous. IoT in itself is a massive environmental threat as it promises to connect billions of objects, thus consuming enormous amounts of resources to manufacture them [180]. However, the resources on the earth being limited, the absolute necessity of those objects to improve the quality of life is to be questioned. The same reasoning goes for new complex AI models which sometimes require very large amount of energy just to be trained [19, 189] but provide a very dispensable service. Today, many (if not most) AI models are trained to choose which advertising to show to a specific user: this situation probably requires efforts from the AI research community to focus on more useful topics such as healthcare or climate change and environment preservation [168].

1.2 The Contributions of this Thesis within the Framework of Human Activity Recognition

1.2.1 Human Activity Recognition: a Broad Science Field

Human Activity Recognition is a very broad computer science field which aims at recognizing what a person is doing by analyzing data related to this person recorded from various sensors or instruments. It has numerous applications: from crime detection on video surveillance images to gesture recognition when performing a physical activity. It can be performed in several contexts and Lara et al. [120] listed seven of them: ambulation, transportation, phone usage, exercise/fitness, military, upper body gestures, for instance involved in human-computer interactions. For each of these contexts, different types of data can be considered. Video and sound data are useful for solving crimes and video surveillance but poses several ethical problems notably regarding privacy and democracy. Another application is related to the fitness watches which use inertial motion data to monitor the user during a physical effort but also during its daily life to help him/her improve her/his way of living and therefore her/his performances. This setup can also be used to monitor the ADL or self-care activities. In the context of eHealth, automatically recognizing ADL can thus help improving the quality of life of elderly persons by ensuring they still can perform

them and automatically assess their autonomy. As a consequence, many Human Activity Recognition (HAR) researches focus on recognizing ADL.

The term "activity" can thus be associated with a broad range of motion data produced by physical movements, from the simplest ones: gestures, postures, falls, ADL, etc. to more complex tasks: sports, daily household tasks, transportation and locomotion, routines etc. In this thesis, the word "activity" will therefore be employed generically along with more precise terms when appropriated. Similarly, we will used the words "recognition" or "classification" to speak of supervised classification (of activities, most of the time). To better distinguish the two tasks, the word "retrieval" will be associated with the identification of routines.

Different categories of algorithms have been considered to tackle HAR. Early approaches consisted in (manually) defining expert rules and thresholds for the sensor values leading to posture recognition and then activities [136, 138, 145]. However those rules can only be defined relatively to one user and these approaches cannot therefore be generalized easily. Other types of approaches include time series analysis, notably with statistical and parametric models such as Hidden Markov Models (HMM) or Bayesian networks [109, 146, 203, 207]. Nowadays, machine learning models can automatically learn to classify these data. Moreover, neural network models are able to automatically extract the relevant features to process and are able to adapt easily to new activities and new users [201].

1.2.2 The Steps to Perform Activity Recognition

Whatever the conditions, the data and the approaches, the process of activity recognition can globally be summarized in five steps [9]:

1. The preprocessing step: motion sensor data can be noisy and imprecise and they therefore require cleaning, notably to remove gravity influence (i.e. Fourier transform, filters, resampling, standardization etc).

2. The segmentation step: sensors data are continuous data and this requires, before associating an activity to a part of the data to segment the time series into parts using (overlapping) sliding windows for example.

3. The feature extraction step: most of the approaches cannot deal with data under the form of time series and demand vector as input. Feature extraction approaches aims at producing such vector input from the data of a segment.

4. The dimensionality reduction step: to avoid the curse of dimensionality and speed up computation, it is sometimes necessary to project the feature vectors into a lower dimensional space.

5. The classification step: finally, the labeled feature vectors are used to train the model. Then, this model can be used to recognize new postures, actions or activities from unlabeled data.

The main contributions of this thesis subvert this classical workflow in different ways. More precisely, the feature extraction and dimensionality reduction steps can be performed simultaneously by a neural network from the raw data. We are interested in modifying the classification step to be able to retrieve routines without activity label supervision instead of classifying labeled activities. This follows our initial discussion about the limitations of supervised activity recognition for real daily life applications and we will now precise the scope of this thesis.

1.2.3 Scope of this Thesis and Contributions

Conscious of the privacy concerns arisen by machine learning technologies, we choose to restrain the data from which we train our models to inertial wearable sensor data. These data are easy to gather with a watch or a smartphone. They are associated with only one individual and can be recorded outside the home (unlike smart home sensors). Finally, they are generally hard to interpret by a human being (unlike video data) which further guarantee privacy preservation.

In order to be able to work in a less supervised framework, we will also tackle activity recognition by considering rhythms. These can be circadian rhythms [5], basically when people sleep or not or more generally routines: successions of recurrent activities. From a psychological point of view, routines are a type of behavior that can be defined this way: "behavioral patterns, based on learned context-behavior associations, that are elicited automatically upon encountering associated contexts" [76]. A routine is a more abstract concept as it is not necessary related to a particular label and can just be associated to a pattern in the data. The major contribution of this manuscript is to

propose new neural network approaches to retrieve the routines from the motion data of a single individual.

The work of this manuscript also integrates itself inside a research effort by Orange on the subjects of eHealth and especially home monitoring of fragile people through actigraphy with inertial sensors and AI. This subject has several components: posture recognition [136], indoor localization [3], routine retrieval with the present work but also emotion recognition, etc.. All these directions have a single objective: produce relevant autonomy indicators to monitor the user on the long-term.

The contributions of this manuscript are therefore threefold and presented in three separated chapters:

In the chapter 2, we start by dealing with personalized supervised activity recognition. We make the observation that classical supervised machine learning approaches are not adapted to handle the high variability of activities performed by the user in its daily life. Moreover, if personalized models should achieve higher accuracy scores, they also impose to the user to provide much more data compared to the case when data from several users are available. To overcome these issues, we propose a few-shot sequence classification model based on matching networks [217]. This model is able to learn to recognize new activities at test time from just one new labeled example which makes it very versatile and flexible for daily life environment.

In the chapter 3, we tackle the issue of routine retrieval. We start by exploring the literature to better frame the idea of routine, we then propose a mathematical definition of this concept which relies on metric learning. After reviewing the main approaches for (sequence) metric learning, we propose an architecture to jointly learning a metric and a representation of sequential data. We also propose a method to retrieve routines using information theoretic clustering scores, the clustering being performed thanks to the learned metric.

In the chapter 4, we propose a new architecture for sequence metric learning, inspired from dynamical system theory and more precisely, synchronization. Two identical chaotic dynamical systems can always be synchronized if a sufficiently strong coupling is applied between them. By drawing a parallel between synchronization and metric on one hand, and identical dynamical system and siamese recurrent neural network on the other hand, we propose a

new model for sequence metric learning which implements a mechanism of coupling between the two networks. We hope with this architecture to improve the performances over the classical siamese model, notably for routine retrieval.

The chapter 5 is dedicated to concluding this manuscript and drawing some perspectives from the presented work. Envisaged future works include designing a new information theoretic temporal dependent loss [186] to be used for routine clustering but also improving the architecture described in the chapter 4. Improvement tracks on this matter encompass studying more complex synchronization metrics [175] to learn or improving the learning time with virtual metric learning [161].

CHAPTER 2

PERSONALIZED ACTIVITY RECOGNITION

If deep learning algorithms have become a primary tool to build activity recognition models, they still require in most situations large amounts of training data. Indeed, the generalization capacities of the model to several users and environments directly depend on the available data. One of the implied risk is of course overfitting: when the model is adapted too much to the training data and is unable to generalize to new ones. This usually happens when the training set is too small and therefore, not representative. On the contrary, everybody has its own specific way to perform postures or ADL, and personalized models, trained and applied to a single user's data should thus achieve a better accuracy. Data gathering is rendered much easier when several sources (users) are available, however obtaining enough data from only one user to properly train a neural network is constraining due to the large time investment required from the user which also delays setting up the service. On the other hand, many users can also implies privacy concerns, especially for healthcare services.

We propose in this chapter to employ few-shot learning to address this problem and present a neural network architecture able to one-shot learn to recognize activities from sequential inertial data. Our approach combines a Sequence-to-Sequence (Seq2Seq) model [197] for robust representation learning of the sequences and the matching network architecture [217] which produces, at test time, a one (or few) shot classifier. This chapter is organized as follows. We summarize in Section 2.1 previous work on HAR tackled with machine learning and then focus on personalized activity recognition. We then introduce classical neural

network models for sequence processing from the literature. We then describe our approach for few-shot personalized activity recognition based on matching network in Section 2.2. In the Section 2.3, we report the results of several experiments on the *MobiAct* V2 Dataset [41] and the UCI HAR dataset [7]: we assess the utility of the different components of the model and its capacity to predict classes that have not been used for training. Finally, conclusions and perspectives are drawn in Section 2.4.

2.1 Related Work

2.1.1 Supervised learning models for HAR

Supervised learning algorithms comprise data-driven approaches where a classification or regression model is learned from labeled data. In this way, instead of having to set up expert rules and thresholds, the model learns directly from the data the *rules* for associating a data sample to an activity performed by a user. Overall, machine learning approaches have the advantage of being able to adapt easily to new situations and new users thanks to their generalization capacity if they are trained on a representative set of data. Numerous research works on HAR using supervised machine learning have been proposed in the literature, notably featuring the following approaches:

Logistic Regression [100] is, despite the name, the counterpart of linear regression for classification. The linear regression models the relationship between the outcome and the features linearly, logistic regression uses the property of the sigmoid function, which squeezes the values between 0 and 1, to transform this linear relationship into a class probability. The parameters are generally obtained by maximum likelihood estimation.

Decision Trees are models that learn to realize several successive splits in the data based on the features (e.g. acceleration on x-axis is higher than some value) until the splits contain only samples from the same class (samples from the same class can be dispatched in several splits). Random Forests [93] are an ensemble method which combines multiple *weak* decision trees to reduce overfitting.

***k*-Nearest-Neighbors (*k*-NN)** classification is a non-parametric method which determines the class of a new sample by considering it the same as the class of its first or more neighbors. If the neighbors

are from different classes, the majority rule is applied sometimes weighted by the distance. Various metrics can be used to improve the results depending on the type of data [222] and *k*-NN, are therefore particularly exploited by metric learning, which we develop in the next chapter.

Support Vector Machine (SVM) [54] is a theoretically grounded algorithm which aims at finding the best separating hyperplane between training samples of different classes, the one from which the points are the farthest: this distance directly impacts the generalization capacity of the model. SVM are able to deal with non-linear decision boundaries by the means of various kernels. SVM were widely adopted by the HAR community notably through works on feature extraction/selection and on hardware embedding [6, 45, 89].

Numerous papers have used and compared these methods as testified by the following reviews [8, 9, 120, 184]. For example, Ravi et al. [165] compared several variants of naïve Bayes classifiers, SVM, *k*-NN, and decision trees to classify 8 common activities (walking, climbing stairs, brushing teeth *etc*) from the data of a triaxial accelerometer. They extracted the following features from 5.12 seconds windows with 50% overlapping: mean, standard deviation, energy, correlation between the axis. They also experimented meta classifiers, classifiers which combine the results of several classifiers to improve the performances and found that they obtained the best results using the plurality voting rule. The choice of the features is indeed a determinant factor to achieve the best performance. Casale et al. [39] conducted a study to find which 319 features they extracted were the most useful leveraging the properties of a Random Forests model. They observed that some axis were more relevant than others and that the root mean squared value of the integrated acceleration in a window was very discriminant on each axis. However, this process of computing numerous different features before selecting the most relevant can be spared by employing a machine learning approach of another kind.

Among the existing machine learning algorithms, Artificial Neural Network (ANN) have become ubiquitous. The advances made during the last 10 years to improve the training of deep neural networks, notably for image recognition [115], popularized the use of deep learning approaches for various applications, among them HAR. The GPU computation considerably accelerated the training time of deep networks on

large datasets. In the past, those approaches were also deemed to be difficult to implement due to the necessity to differentiate the equations and therefore their implementations. Nowadays, the advances made in algorithmic differentiation [16] allow to easily set up deep learning algorithms since the differentiation is rendered automatic. Neural networks are able to automatically extract features from the data: while learning how to solve the task they are trained upon, they learn to extract the most relevant features to do so. As an example for such approach, Zeng et al. [239] used a Convolutional Neural Networks (CNN) to extract features from sequential mobile sensor data and perform activity classification using a *k*-NN. A CNN is a type of network adapted to deal with matricial data such as images which uses a succession of convolutional filters to extract local features and pooling layers to condense the information. They showed that the CNN could outperform other approaches for feature extraction such as Principal Component Analysis (PCA) and Restricted Boltzmann Machine (RBM). However they did not confront their approach with non-linear dimensionality reduction approaches such as autoencoder and Laplacian Eigenmaps [17]. In fact, Wang et al. [219], in a recent review dealing with deep learning approaches for HAR, mostly discussed CNN-based approaches. However, sensor data are sequential which makes Recurrent Neural Network (RNN) *a piori* more suited for this task. For example, Lefebvre et al. [125] used a bidirectional Long-Short Term Memory neural network (LSTM) [177] to classify 14 gestures performed by 22 different individuals. In this work, no CNN is used and the LSTM is directly trained to extract the features from the data coming from inertial sensors and to classify the gestures. They achieved up to 95.57% of accuracy and showed that the combination of accelerometer and gyroscope data allows to produce better results. Murad et al. [144] proposed to compare several architectures of LSTM, unidirectional, bidirectional and cascading [230] (a combination of the first two) to SVM, *k*-NN and CNN for HAR. On several datasets, variants of the RNN outperformed the other approaches: according to the authors, this is due to the ability of the RNN to extract relevant temporal features. Finally, Guan et al. [85] improved once more the RNN model by designing an ensemble architecture combining several LSTM and merging their decisions. The multiple LSTM are trained a bit differently to allow the model to overcome the real-world HAR challenges of noisy sensor data and class imbalance. To do so, the LSTM are trained with different sampling starting points, different batch sizes and different sequence lengths. After decision fusion, the authors observed an increase of performance over a single LSTM. With the same purpose of increasing the accuracy in real

environments, we now focus on personalized models which should better take into account the specificity of each user but also bring challenges regarding the quantity of data.

2.1.2 Personalized HAR models

The question of personalized models is essential to achieve the best performances: indeed, most of the time, people perform activities in a very personal way. In 2004, Bao et al. [14] already observed that user-specific models could lead to better results while classifying vectors of handcraft features with a decision tree. Since then, several other papers came to the same conclusion that personalized models are indeed more accurate, and we present in the following three approaches to learn personalized models: multitask learning, active learning and few-shot learning.

2.1.2.1 Multitask learning

Multitask Learning aims at improving the generalization of the model by learning several tasks more or less similar in parallel while sharing a intermediate representation, a shared hidden layer in the case of neural networks [38]. Then the losses computed for all tasks are combined and used to update the whole network. This approach has been used by Sun et al. [193, 194] to tackle online personalized activity recognition. In their model, to each person corresponds a task, a Conditional Random Field (CRF) to learn. CRF are a category of statistical modeling methods which uses a graphical model to take into account the relationship between the predictions. The parameters w of the model can be learned using gradient descent. Here, for each task, the loss of all the tasks are summed using a specific weighting which is computed by measuring a distance between the tasks thanks to a kernel on the parameters w of each task. In another publication, Peng et al. [160], while not proposing a specific approach for personalized activity recognition, achieved the best result on the Ubicomp 08 Dataset [101] which contain a unique user data, compared to other classical neural network approaches. Their approach, human Activity RecognitiOn using deep Multi-tAsk learning (AROMA), consists in separating the activities to classify into two categories: simple (postures, etc.) and complex (eating, working, other routines etc.), the first being the components of the second. Simple activities are recognized using a CNN, complex one with a LSTM: complex activities should be

a sequence of simple ones. The model are both multitask learned by backpropagating a sum of the two losses into the networks.

2.1.2.2 Active learning

Active learning assumes that getting the labels for a lot of data points is very expensive (requires a lot of time or expert knowledge etc.). Therefore the goal of an active learning algorithm is to select the most efficiently as possible a subset of the unlabeled data points to learn from, the most informative for the training, then an oracle is queried to get the labels [178]. Several selection strategies exist. Stikic et al. [187] obtained better results with a low confidence strategy for an activity recognition task on sensor data. When using the low confidence strategy, a label request is triggered when the classifier outputs for a sample nearly the same classification probability for each class. Abdallah et al. [1] proposed an approach to recognize activities from streaming mobile data. They used active learning to limit the number of samples to annotate, the selection strategy employed here is to request a label to the user when the learning classifier has a low confidence in its classification. This way, the classifier gets a lot of data to be trained with limited interaction with the user. The learning is done in two steps: first a clustering model is made with the labeled samples, it is then improved with the unlabeled ones. In another publication, Longstaff et al. [133] proposes to improve the training of already deployed models with specific-user data to obtain a more adapted model. They succeeded in improving the performance of a decision tree by at least 10% with various methods notably, active learning with the low confidence strategy. However, they remarked that the better the base model was the lower the observed difference, notably for a base model exhibiting accuracy over 90%. Though being used in various publication, the low confidence strategy can be criticized. Settles et al. [179] remarked that samples getting a low confidence are not necessary representative of the class because too close to the decision boundary. They recommend on the contrary to weight how informative a sample is by its similarity to other samples: the more similar it is, the more susceptible it is to get a label request.

2.1.2.3 Few-shot learning

Personalized HAR, while theoretically promising better performances, is in practice confronted to at least one technical issue: the burden of data gathering lies on each individual user. However, deep neural networks

typically require large amounts of data to be trained which, therefore requires lots of time investment from the user and delays the availability of the service: this can be critical for medical application. To conclude, personalized models should be trainable from few data. We are helped for this purpose by the fact that not that much data may be necessary due to the fact that less generalization is required from the model (one user in most of the time in one environment, e.g. his/her home). This issue of data shortage is not really tackled by both previous approaches. In this context, machine learning models especially conceived to be trained from few data, so-called few-shot learning models, could provide a way around. For example, Nguyen et al. [149] built a model able to learn to detect never seen activities from very few wearable-sensor data. The model consists in a feature-based part associated by decision fusion with rules connecting the new activity to the others already learned by the model. This way, the model is able to prevent the degradation of performances resulting from adding a class but also not to overfit too much by learning from only 2 or 3 sequences. In another work, Wu et al. [229] presented an approach for one-shot classification of gestures from videos data. They used classical video features and tried to match test samples with one unique training example using the maximum correlation coefficient. Matching networks [217] (see Section 2.2.1 for details) are also based of this idea of matching samples with references but use metric learning instead [18]. Following this idea, Sani et al. [174] proposed to use matching nets to produce personalized models from accelerometer data. Their method achieved 79% of F1-score on 9 classes, outperforming most standard approaches trained on every user data.

2.1.3 Synthesis and discussion

We discussed the application of data-driven models to HAR, particularly to personalized HAR. On one side, classical machine learning approaches such as SVM or random forests necessitate to extract handcrafted features from the raw data, whereas neural networks are able to learn several layers of representations, extracting more and more abstract and specific features each time while pursuing the tasks of classification or regression. Postures, ADL or falls exhibit a dynamic characteristic signature (walking, running, going upstairs etc.) or are, on the contrary, more static (lying, sitting, standing etc.). Some complex activities are composed of multiple simpler ones and postures. A classical approach when working with sequences is to extract several signal feature vectors from subsequences of the signal in order to build a classifier.

This approach is efficient in numerous cases but, as the window size is limited, it cannot exploit long term dependencies. Therefore we propose to employ RNN, more precisely Gated Recurrent Unit (GRU) to be able to learn long-term dependencies trained to automatically extract features from long sequences, when this is possible, depending on the dataset. We also describe 3 approaches employed in the literature to tackle personalized HAR. From them, few-shot learning is especially adapted to data shortage situation, which we considered could occur when learning one model for one user. The paper by Sani et al. [174] presented earlier proposes a few-shot learning approach, based on neural network and seems like a good starting point but can be pushed further. First, they seem not to have used the full-context embedding (see 2.2.1) which could provide slight improvements. Then, they trained and tested the model on every class at the same time whereas matching networks are conceived to work independently of the classes they are trained on. Finally, they used discrete cosine transform coefficients as features whereas we wish to extract features automatically with a neural network. We also propose to use few-shot learning to tackle the issue of personalized activity recognition from sequential sensor data, and propose a pure neural network approach to do so, namely Sequence-to-Sequence Matching Network (SSMN). However, before introducing it, we need to describe with further details how neural networks work, especially RNN which are conceived for sequential data.

2.1.4 Sequence Processing with Recurrent Neural Networks

We describe in this subsection how feedforward neural networks and recurrent neural networks work. We then introduce more complex models able to deal with sequential data.

2.1.4.1 Feedforward Neural Networks

Neural networks designate a category of algorithms inspired by the brain function. Neural networks map an input to an output by previously learning a model. They are basically composed of units (neurons) organized as layers which compute the weighted sum of the inputs before passing it to every unit of the next layer. The first layer is called the input layer, the last the output layer and layers in the middle are called hidden layers (see Fig 2.1). This architecture is called a MultiLayer Perceptron (MLP) and the network is said to be feedforward because

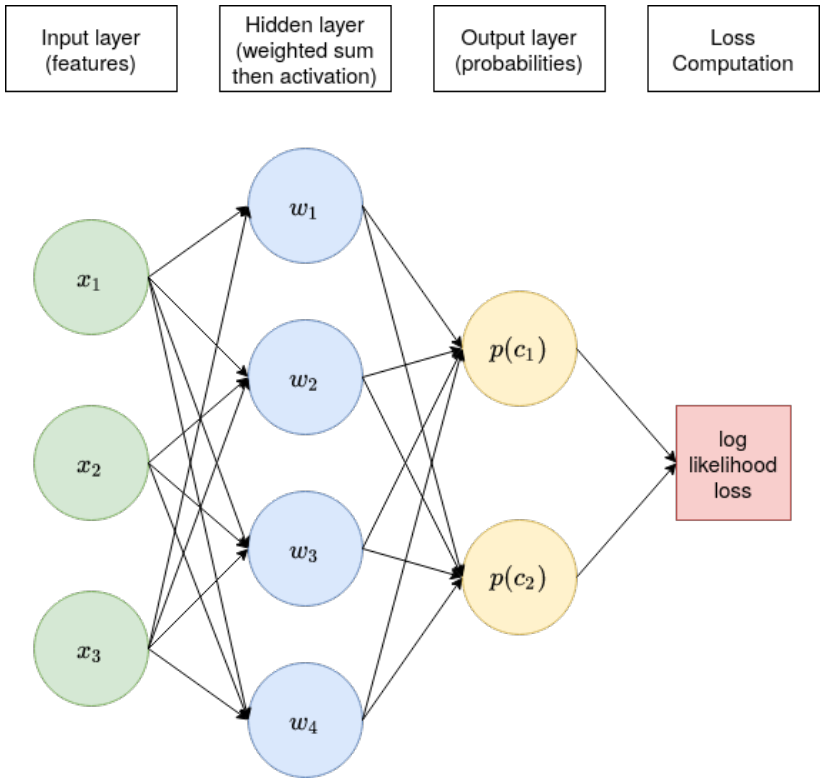


Figure 2.1 – Schema of a neural network, a MLP conceived for binary classification. In green, neurons of the input layer. In blue, neurons of the single hidden layer. In yellow, neurons of the output layer: the output vector $\hat{y}$ is equal to $\{p(c_1), p(c_2)\}$.

the information only goes forward inside. Generally, neural networks use supervised learning algorithms for training, and more particularly backpropagation [90]: the output computed by the neural network is compared to the desired output with the help of differentiable similarity function, which can be, in the simplest cases, the cross entropy loss (2.3) for classification or the Mean Squared Error (MSE) (2.9) for regression. Finally, the gradient of error is backpropagated through the network to modify the weights. The goal of training is so to find a global error minimum on the training set, or on the validation set to assert good generalization capacities to the model.

We will now present the equations governing a feed forward neural network. Firstly the signal is propagated forward, each neuron computes its activation.

Definition 2.1. The activation a_i^l of the ith-neuron of lth-layer is the value computed as follows:

$$z_i^l = w_i^l * a^{l-1} + b_i^l, \tag{2.1}$$

$$a_i^l = f(z_i^l), \tag{2.2}$$

where b is called a bias and $a^{l-1} \in \mathbb{R}^d$ is the activation vector coming from layer $l-1$ comprising d neurons and f an activation function.

The activation function is used to introduce a non linearity in the neuron activation. In the case of feedforward neural networks, the Rectified Linear Unit (ReLU) function $\max(x, 0)$ [72] has been shown great interest during last decade becoming the most used activation function.

At the end of the network, an output $\hat{y}$ associated to the input x is computed and compared to the desired output y using, for example, the cross-entropy loss:

$$\mathcal{L}(\hat{y}) = -\sum_{i=0}^{c} y_i \log(\hat{y}_i). \tag{2.3}$$

This loss is used for multiclass classification tasks ensuring that the y are one-hot encoded class labels, that the output layer of the network has a size equal to the number of classes and that the coordinates of the $\hat{y}$ represent probabilities, by using a softmax for example. This error can be computed for one input (online) or several and then averaged (batch). The use of online or batch training mainly depends on the application domain and on the quantity of data as for batch training, one needs to have all the data available at once. Then the gradient descent algorithm is used to backpropagate the error to correct the weights and biases and make the network "learn" by being driven to produce outputs leading to a global minimum of error. Using the gradient descent algorithm we can find how to update the weights w_i and biases b_i of each neuron of each layer using the derivative chain rule.

2.1.4.2 Recurrent Neural Networks

ANN have been conceived to work with fixed-size vectors however, they can be slightly modified to deal with sequences. Indeed, RNN possess recurrent connections which give the ability to map an input sequence to an output sequence while at each step taking the information of previous steps into account. This enables the network to not only extract inter-signal but also intra-signal correlations and thus to detect more complex

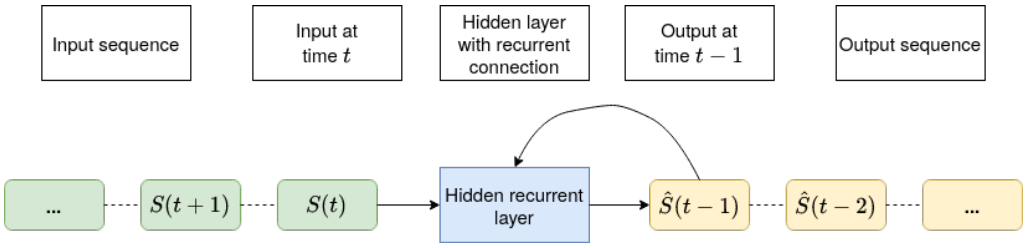


Figure 2.2 – Schema of a Recurrent Neural Network (RNN) with input and output sequences.

patterns. Let S be the sequence of inputs of the network and $S(t)$ be the input at time t. A RNN computes $\hat{S}$, the sequence of outputs (see Figure 2.2) following this equation:

$$\hat{S}_t = f(W_x S_t b_x + W_h h_{t-1} + b_h), \tag{2.4}$$

where W_x and W_h are respectively the input and hidden synaptic weight matrices, b_x and b_h the biases, h_{t-1} the hidden state of the previous step and f is an activation function, generally hyperbolic tangent. The back propagation of the error is performed by *going back in time* and taking successively into account every previous time steps: this is the Back Propagation Through Time (BPTT) [224]. In fact, when a RNN is *unrolled*, it behaves nearly as a feedforward network. When one runs BPTT, this requires to store every activations since the beginning of the sequence and therefore it can become computationally expensive. In practice, we can define a truncated depth which indicates the number of previous steps for which we compute BPTT each time we run it.

Two major difficulties have been identified when training a RNN: the vanishing and exploding gradient problems [21]. When the gradient vanishes, the network basically stops learning, when it explodes, it can cause weights to oscillate between different values [94]. These two phenomenons have a similar origin. When applying back propagation on deep networks, one must concatenate more and more multiplications of activations as it goes back in the network. These activations are bounded, in the case of the sigmoid function, between $[0,1]$ and this cause the gradient signal to vanish. This problem appears in every deep network although simple solutions have been found for feed forward networks such the use of the ReLU [82] instead of the sigmoid function and the introduction of *skip connections* in so-called Residual Networks (ResNet) [88]. Conversely, if the activations are greater than 1, they

can grow and cause the gradient to explode. Training a RNN on long sequences to learn long-term dependencies necessitates a deep unrolled recurrent network which inevitably suffer from the vanishing/exploding gradient problems, preventing it to learn correctly. An experiment made by Bengio et al. [21] gives a better appreciation of the magnitude of this phenomenon. They tried to make a single recurrent neuron learn 1 bit of information: the dependence between the first input of the sequence and the last (basically positive or negative in both cases). They discovered that the longer the sequence the more difficulties the neuron had to predict the last input, even if only the first and last input were significant. RNN really struggles with long-term dependencies and this had to be somehow overcome to make RNN relevant for a large number of applications.

2.1.4.3 Gated Recurrent Units

Hochreiter and Schmidhuber [95] proposed a way around the vanishing/exploding gradient problems to allow RNN to learn long-term dependencies by introducing a gating mechanism: the LSTM. It possesses a *cell state*, a sort of internal memory in which new content is added and old content is *forgotten* at each time step by the mean of gates. Later, GRU [46] were introduced as a simplified version of the LSTM approach, while showing similar performance on most sequential data analysis tasks [48] (e.g. speech and music modeling). We detail in the following the equations controlling the information flow in a GRU network. First, it computes the vector h_t as follows:

$$h_t = (1 - z_t) \circ \tilde{h}_t + z_t \circ h_{t-1}. \tag{2.5}$$

The hidden state h_t is updated by forgetting the old content and directly adding some new. The update gate z_t is computed according to the following equation:

$$z_t = \sigma(W_{iz} x_t + b_{iz} + W_{hz} h_{t-1} + b_{hz}). \tag{2.6}$$

We also have the following relation for the new hidden state $\tilde{h}_t$:

$$\tilde{h}_t = \tanh(W_{i\tilde{h}} x_t + b_{i\tilde{h}} + r_t \circ (W_{h\tilde{h}} h_{t-1} + b_{h\tilde{h}})), \tag{2.7}$$

where finally r_t is the reset gate computed similarly as z_t:

$$r_t = \sigma(W_{ir} x_t + b_{ir} + W_{hr} h_{t-1} + b_{hr}). \tag{2.8}$$

With this mechanism, a GRU network can retain information and learn intra-sequence correlations on the long term.

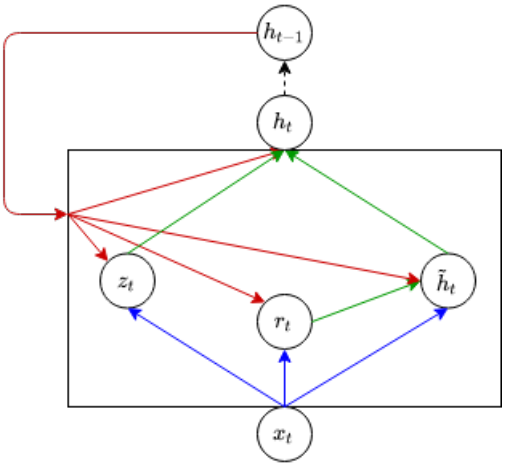


Figure 2.3 – Schema of a GRU with the three gates inside the rectangle controlling the information flow coming from the input and from the hidden state.

2.1.4.4 Sequence-to-Sequence model

Finally, one interesting way of using RNN and GRU is to learn vector representations in a similar fashion as auto-encoders [113] by connecting two RNN: an encoder and a decoder. The so-called Sequence-to-Sequence models [197] (see Figure 2.4) learn how to reconstruct the input sequence from the last output of the encoder. To measure the reconstruction error and train the network, the MSE is generally used:

$$\text{MSE}(S, \hat{S}) = \frac{1}{l} \sum_{t=1}^{l} (S(t) - \hat{S}(t))^2, \tag{2.9}$$

where $\hat{S}$ designates here the output sequence of the decoder. That way, a robust representation of the sequence can be learned. Adding noise (often putting 30 or 50% of the values of the sequence to zero) to the input sequence is a good way to improve the generalization capacities of the model [214]. The decoder is therefore trained to reconstruct the non-noisy original sequence, forcing the encoder to produce more robust features. The learned representations can then be used by other neural networks as input.

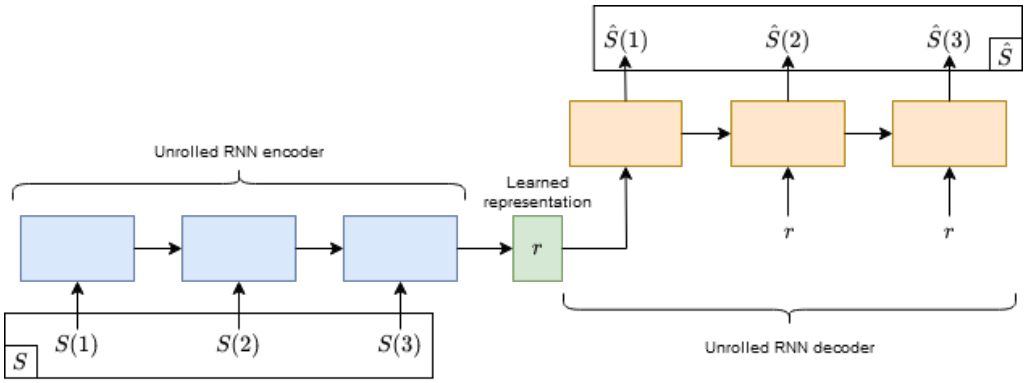


Figure 2.4 – Sequence-to-Sequence Model [197].

2.2 Few-shot Personalized ADL Classification

Neural networks are able to automatically extract the relevant features they need to diminish the error on they make the task they are learning. To do so they use a hierarchy of features and the deeper the network the more abstract representations the network is able to build [20]. However, this requires many layers, and consequently many more parameters which can not only lead to the exploding or the vanishing gradient issue (see Section 2.1.4.2) but also to the overfitting problem if not enough data is available to train those numerous parameters [185]. These theoretical issues converge with the HAR problematic of learning models for a specific user with potentially a low amount of user-specific data. A way around would be to train the model on data from several users and then to perform a fine tuning of the last layer(s) with user-specific data. This typically works well for image classification [199] and allows to reuse classical CNN architectures trained on large datasets (e.g. Imagenet[1] [62]) on more specific tasks. These types of models typically manipulate deep representations, the first ones extracting low-level features common to many images, the last more high-level and specific patterns. However, this process works well with images but is more difficult to adapt for sequences of sensor data. Another approach is to train models specifically conceived to deal with few data: few-shot learning models, for example Matching Networks [217]. An additional advantage of these models is their flexibility in terms of classes they can recognize. We will first introduce

[1] http://www.image-net.org/

the Matching Network architecture before describing adaptations made to it for personalized activity recognition from sequential sensor data. This model makes use of some form of transfer learning to pretrain a Seq2Seq model 2.2.2. This work has been published in [50].

2.2.1 Few-shot Learning with Matching Networks

Vinyals et al. [217] developed a model called matching networks based on metric learning and attention to efficiently learn to perform few-shit classification. This model is not actually trained to classify but rather to match samples with other examples that are part of a support set we call $\mathbb{S}$: it learns to produce a nearest-neighbor classifier. It allows the model to work at test time with some classes never seen during training and therefore to perform few-shot and even one-shot learning. $\mathbb{S}$ contains N labeled support examples, one or several for each of the C classes. The model is described in Figure 2.5 and is composed of 4 parts. The first is an encoder, a neural network trained to produce a vector representation y of the example sequence S to be matched and of each sequence in the support set called $\mathbb{S}_{\text{emb}}$ in the following. The second and third parts are called the context embedding and are used to adapt the representations y and $\mathbb{S}_{\text{emb}}$ relatively to each other. Thus, the second part, the bidirectional GRU, will produce $\mathbb{S}'_{\text{emb}}$, representations of each element in $\mathbb{S}_{\text{emb}}$ relatively to each other support example. This component processes $\mathbb{S}_{\text{emb}}$ taken here as a sequence in both directions and aggregates the results according to the following equation:

$$\mathbb{S}'_{\text{emb}} = \overrightarrow{h} + \overleftarrow{h} + \mathbb{S}_{\text{emb}} , \tag{2.10}$$

where $\overrightarrow{h}$ and $\overleftarrow{h}$ are the sequences of hidden states produced by the bidirectional GRU in each direction, respectively which therefore contain the same number of vectors than $\mathbb{S}_{\text{emb}}$. The vectors are aggregated with a component-wise addition. Another level of sophistication is the third part which consists in transforming the embedding y according to $\mathbb{S}'_{\text{emb}}$, that is, make it closer in the latent space to the embedding of the support elements it could match. This is done thanks to an Attention GRU model, detailed in Algorithm 1, in order to avoid the new representation y' depending on the order of the vectors in $\mathbb{S}'_{\text{emb}}$ [216]. The parameter p, the number of processing steps, is the number of times y' will be passed through the GRU with r as hidden state. This model is an attention model [212]: it selects elements in $\mathbb{S}'_{\text{emb}}$ according to their similarity with y at the first processing step and y' at the subsequent steps. The

similarity is here measured with a dot product (see line 9 in Algorithm 1). Finally, an *answer* is computed by doing the weighted sum of the support examples (line 11). Once the representations are produced, the last part computes the distances between y' and each vector in $\mathbb{S}'_{\text{emb}}$ with, for instance, the Euclidean distance or the cosine similarity. A softmax function allows then to get probabilities of matching for each sample of S which correspond to the probability for x to belong to the same class as the sample. Finally, the highest probability determines the class of the input sample. We will now describe an adaptation of this architecture for personalized activity recognition.

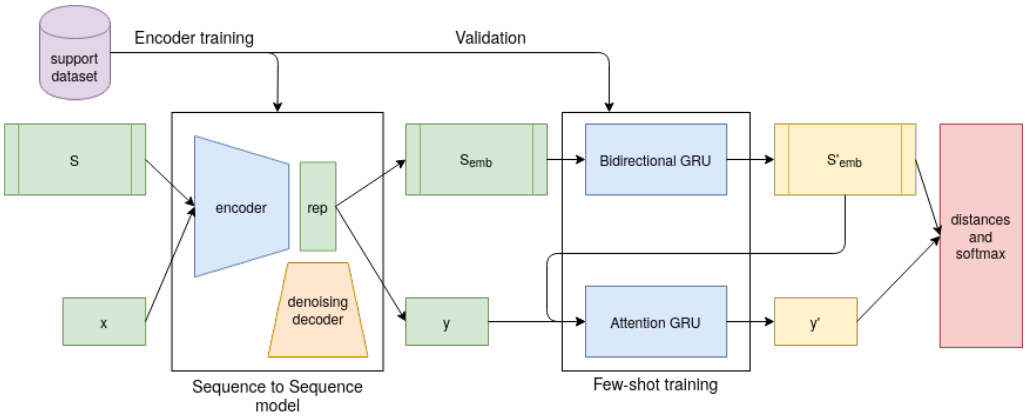


Figure 2.5 – Overview of the SSMN architecture adapted to few-shot sequence classification

2.2.2 Few-shot Learning for Personalized ADL Classification

Matching networks, which are able to generalize new classes from just one example, present several attracting properties to build personalized activity recognition models notably in a real context where people do a large variety of activities every day and sometimes completely new ones which would thus be recognizable without retraining the whole model. The general architecture described in the previous section can be used with any type of data by choosing the right encoder for those data. In their original paper, Vinyals et al. [217] used classical convolutional models pretrained on large datasets as encoders (such as VGG [181] or Inceptions [198] for image processing tasks) then they perform or not a finetuning of this encoder at training time on the specific

Algorithm 1 GRU with read attention [216]

procedure ATTENTIONGRU($y', \mathbb{S}'_{emb}, p$)
  $h \leftarrow 0_{1,n}$
  $h \leftarrow \text{GRU}(y', h)$
  $i \leftarrow 0$
  for $i = (2..p)$ **do**
    $h \leftarrow h + y'$
    $a \leftarrow 0_{1,N}$
    for $j = (0..N-1)$ **do**
      $a_{1,j} \leftarrow h\mathbb{S}'_{emb,j}$
    $a \leftarrow \text{Softmax}(a)$
    $r \leftarrow \sum_{n=0}^{N-1} a_{1,n}\mathbb{S}'_{emb,n}$
    $h \leftarrow \text{GRU}(h, r)$
    $i \leftarrow i + 1$
  return $h \leftarrow h + y'$

small datasets. We propose to train a GRU encoder for sequences as a sequence-to-sequence model [195] from which we kept only the encoder part (Sequence-to-Sequence model frame on Figure 2.5). We therefore call our approach SSMN. We leverage the properties of matching nets previously exposed by proposing to use another dataset to train the sequence-to-sequence model. This dataset must contain very similar data (*i.e.* inertial data) to the dataset we are trying learn from but can be taken from the literature and can have completely different classes. This could be assimilated to some form of transfer learning [155] but the data actually stay very similar, and we set aside the question of using completely different sequential data (music, text, etc.) to perform this pretraining. This can however raise sampling and data standardization issues because the sensors employed to record the data were not the same. Another way to exploit this property is to also use this similar dataset as a validation set to avoid overfitting during the training of the matching nets which is done with very few data (2 or 5 sequences for each activity plus one in the support set). Thus more data can be kept to train and test the model. We call this other dataset, a support dataset: it is not strictly necessary but could greatly increase the performances of SSMN.

In the next section, we experimentally verify the benefit of using a support dataset together with the property of matching networks to be

independent of the classes they are trained on by presenting only test results on classes not used for training and recognized from just one support example (*i.e.* performing one-shot learning).

2.3 Experiments

2.3.1 Datasets

The previous works mentioned in Section 2.1 tested their approaches on different datasets and the lack of reference datasets is a known issue among the activity recognition community. It is due to the variety of contexts in which it can be performed and also the variety of data that can be used. In the remaining of this chapter, we chose to concentrate on three recent datasets on which results have already been published in order to be able to perform comparisons. The main features of each dataset are summed up in Table 2.2.

2.3.1.1 Postures

The first dataset is called *Postures* and was created by Quach [164]. The data have been acquired using a 9-axes Inertial Measurement Unit (IMU) (accelerometer, gyroscopes and magnetometer, IMU) on 9 subjects executing the same activity scenario several times: 5 times for each user apart from user 2 who did 10. The scenario is composed of five postures, repeated several times: walking, sitting, lying, standing and transfer and have been conceived to reflect a daily routine of 24h on a 12 minutes activity sequence. "Transfer" represents the transition between two postures. Overall, there are about 358k labeled 9D vectors and 1938 segments of scenario related to one activity with variable length (see Table 2.1 for details). The sampling frequency is around 10 Hz. The IMU was a *Shake SK6* [226] with the following range and precision for each sensor. The range of the triple axes accelerometer is at most ±6g with a precision of 1mg. The range of the triple axes gyroscope is of ±500 °/s with a precision of 0.1 deg/second. The triple axes magnetometer has a range of ±2 Gauss and a precision of 1 mGauss.

This dataset was used in a previous publication by Makni et al. [136] where two attitude estimation algorithms were compared: Kalman filter and Complementary filter to estimate the individual postures.

	walking	sitting	lying	standing	transfer	Total
# segments	454	280	138	280	751	1938
# vectors	63863	118802	149602	19555	28425	380247

Table 2.1 – Summary of *Postures* dataset activity sequences

2.3.1.2 MobiAct V2 Dataset

The second dataset is accessible online. The MobiAct V2 [41] is an inertial dataset created to support research in ADL recognition. It includes 15 different activity labels: 4 falls and 11 ADL. The activities were recorded following a realistic scenario: a typical day of work by 67 subjects. In total, the dataset is constituted of about 3200 trial sequences. Data were acquired using a smartphone which the user could place anywhere. The IMU is a *LSM330DLC* itself composed of a tri-axes gyroscope and a tri-axes accelerometer. The measurement range of the accelerometer can be selected between ±2g, ±4g, ±8g or ±16g. The measurement range of the gyroscope can be selected between ±250°/s, ±500°/s or ±2000°/s. The orientation features combine data from the accelerometer and the magnetometer of the phone. We considered only the 19 users[2] for our experiments who have performed the 15 activities with at least two trials for 12 activities out of the 15 and one trial for "walking", "sitting" and "standing". Each user has around 53/54 trial sequences in total. No preprocessing was applied to the trial sequences apart from a resampling to a length of 50. Some trial sequences contain only one activity, others have several (for example, standing and lying before and after a fall). We treated each trial as one activity, the one mentioned in the name of the file containing the trial, and we labeled the resampled sequence. We did the same on the *Postures* dataset, after segmenting the dataset to train the encoder. In a real environment, this segmentation could be replaced by resampled sliding windows of the signal.

We now detail some published results on this dataset and the methodologies employed. To begin with, the authors of the dataset used it for activity and fall recognition with the goal to develop the most effective pipeline in terms of accuracy. To do so, the authors conducted an exhaustive study to find the best features (for example, they found that the spectral centroid was quite essential). They achieved an accuracy score of 97% with an instance-based k-nearest neighbor classifier. On the contrary, Di Pietro et al. [65] proposed a new neural network model called

[2]The selected users are the following : 1, 2, 3, 5, 6, 12, 20, 45, 53, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67.

MIxed hiSTory (MIST) RNN which they tested on *MobiAct* V2, among others. They did not select features and learn directly from the raw data. They separated users into fixed train, validation and test groups and repeated the same experiment 50 times and kept the 5 best results. Their approach requires less computation and produces better results than a LSTM Neural Network by achieving 71% of accuracy. Finally, Tsinganos [205] proposed a threshold-based approach combined with a nearest-neighbor algorithm and a mechanism of adaptation to the user of the feature vectors to classify falls. Their approach achieved above 90% of accuracy score on *MobiAct* V2. Contrary to those approaches, we aim at building personalized models on the *MobiAct* V2 but each user has at most around 50 sequences, which is not enough to train a classical neural network model able to automatically extract and process the relevant features. We thus propose to use matching networks instead to build our architecture.

2.3.1.3 UCI HAR dataset

The last dataset is called UCI HAR [7]. It provides the data of 30 users who performed 6 activities: "walking", "walking upstairs", "walking downstairs", "sitting", "standing" and "lying". It contains 10299 sequences of length 128 which are fixed-width sliding windows of 2.56 sec with a 50% overlap. The sensors are a tri-axes accelerometer and a tri-axes gyroscope recording at 50 Hz. The acceleration has been processed to separated the total acceleration and the body acceleration for a total of 9 features. We applied no additional preprocessing.

Numerous results have since been published regarding this dataset. The authors themselves notably achieved 96% accuracy with a multiclass SVM. Zhao et al. [240] proposed to use a residual bidirectionnal LSTM recurrent neural network to classify the sequences of the dataset and obtained an accuracy of 91.1%. Jiang et al. [106] proposed an approach based on CNN. They designed an architecture able to achieve high accuracy with a low computational cost and got around 95% test accuracy. CNN notably require a large quantity of data to be properly trained which is more difficult to obtain in a personalized setting. San-Segundo et al. [173] proposed to use HMM but with user adaptation which resulted in an error rate of 2% and a recall of 95.3%. Their model is first trained on the data of every users before being tuned for one specific user with Bayesian adaptation. Though being personalized and using finetuning, the approach presented in this chapter presents

Dataset	Postures	Mobiact V2	UCI HAR
Features	accelerometer, gyrometer, magnetometer	accelerometer, gyrometer, orientation	total and body acceleration, gyrometer
# Classes	6	15	6
original sampling	10Hz	maximum	50Hz
sequence length	50 or 128	50	128

Table 2.2 – Main information regarding the three datasets used in this chapter.

several differences with theirs. First, their approach actually needs to be first trained with the whole dataset before being finetuned for one user whereas our Matching Networks-based approach can be finetuned with any inertial dataset from the literature. Then, their model is composed of 6 HMM, one for each class which makes the addition of a new class more difficult to set up whereas, once trained, Matching Networks can start recognizing a new class with just one new sequence. Overall, we observe that several models are able to achieve an accuracy score of around 95% on this dataset.

2.3.2 Preliminary Experiments

2.3.2.1 Personalized Postures Classification with a GRU

We first propose a preliminary experiment on the dataset called *Postures* (see Section 2.3.1.1) where personalized models are learned on inertial data with a standard GRU only. This dataset will be used afterwards as support dataset as explained in section 2.2.2.

The personalized posture GRU-based models were trained on four sequences (nine for user #2) during 150 epochs and were tested on one randomly chosen sequence. Based on preliminary experiments on the full dataset, we use personalized GRU models with two hidden layers of size 8 and found that this small architecture was sufficient to achieve good performances. The process was reproduced excluding a different sequence each time (5 times for each user except for user 2, 10 times). We thus performed a k-folds leave-one-out test of our architecture where k is the number of sequences associated to one user. Moreover, regularization is introduced in the training by using dropout [185] and weight decay

Table 2.3 – Results for posture classification on *Postures* with GRU. (SD: Standard Deviation)

Method	f1-score	SD	Accuracy	SD
Makni et al [136]	-	-	0.807	0.024
GRU [8,8] (all users)	0.553	0.068	0.742	0.056
GRU [8,8] (user 2)	0.705	0.059	**0.874**	0.049

[116]. Finally, each training starts with a learning rate of 0.01 which decreases by a factor 10 if the loss does not diminish during 10 epochs.

A comparison with the results of Makni et al. [136] is presented in Table 2.3. On average, the GRU accuracy of 0.742 is lower than the 0.807 achieved in [136]. Nevertheless, the model achieved with the data of user #2 a better accuracy score of 0.874. This is mainly due to the fact that the user provides twice as much data as other users. Consequently, asking people to collect only 10 sequences for building a shallow GRU model shows promising results with an acceptable user effort, in practice.

This preliminary experiment shows the benefit to learn personalized models with GRU even from only 5 continuous sequences of activities. We will now push this approach further on a larger dataset with more users and activities but less data per activities. Based on the results of this preliminary experiment, we will test our approach SSMN by performing a pretraining of the encoder as a Sequence-to-Sequence model on *Postures*.

2.3.2.2 Pretraining of Sequence-to-Sequence model on Postures

We propose in this second preliminary experiment to observe how the training of the Seq2Seq model on *Postures* generalizes to the validation set (the first user) of each dataset. First, we detail the similarities and differences of each dataset with *Postures* in order to better understand the obstacles to use an encoder trained on *Postures* on another dataset.
The *Postures* and *MobiAct V2* datasets have three postures in common: walking, standing and sitting. The lying posture is not independent and always concatenated with a fall. The posture "transfer" in the Postures dataset can be assimilated to "stand to sit" ADL in the *MobiAct V2* dataset. However, "transfer" is very diverse and less characterized and we did not consider those two as strictly similar. Moreover, the activities "walking", "sitting" and "standing" in *MobiAct V2* only have one trial sequence available per user which is not enough to test our algorithm: we need at least one sequence to be the support example and another to

match with it. Thus, we removed "walking", "sitting" and "standing" from the *MobiAct V2* dataset, to have both datasets having strictly disjoint sets of activities[3]. Features in both datasets are globally the same apart from the magnetometers which is rather an orientation computed from the other features in *MobiAct V2*. The learned encoder is therefore not exactly adapted to the features of *MobiAct V2*. We hereafter name this modified version of *MobiAct V2* dataset *MiniMobiAct*.

Regarding UCI HAR, four classes out of 6 are common to both datasets. The main difficulty resides in the organization of the dataset under the form of sliding windows. To stay coherent with what has been done with *MiniMobiAct*, we chose to simply resampled the sequences of *Postures* to a length of 128. Finally, regarding the features, UCI HAR presents no magnetometer data, which could seriously deteriorate the quality of the representation when used later with SSMN.

We finally give some considerations on the standardization of the datasets. Normally, the standardization is done on the training set and the same parameters are then applied to standardize the validation and testing sets. However, we have seen that the datasets do not share exactly the same features, they have also not been recorded using the same sensors. Thus, we chose to standardize each dataset regarding to itself.

To observe how the encoder behaves on another dataset, we propose to observe the evolution of the performances depending on the encoding size. We expect the MSE to decrease for the validation set of *Postures* used here (user 9) with the increasing of the encoding size (up to some point). The MSE on the other dataset (user 1 in each case) should follow a similar evolution. The results are presented on Figure 2.6. Due to the different scaling of each features of each dataset, absolute comparison of MSE between datasets are irrelevant.

We observe that the three curves evolve similarly with a rapid decrease in MSE with encoding sizes inferior to 50. The minimum on *Postures* is achieved for a size of 100, after, the MSE stagnates near the minimum. For *MiniMobiAct*, the minimum is also achieved around the same size. For UCI HAR, the results are more difficult to interpret as the decrease is less marked, probably due to the differences mentioned earlier. This experiment helped us apprehend the relevance of

[3] Actually, there remain some "standing" or "lying" parts in some sequences, for example in fall sequences. However, they are not labeled as such and thus will not be learned as such by the model. Potentially, this could be a factor of confusion for the model, but, in view of the experimental results, our approach still focuses on the relevant parts of the sequence.

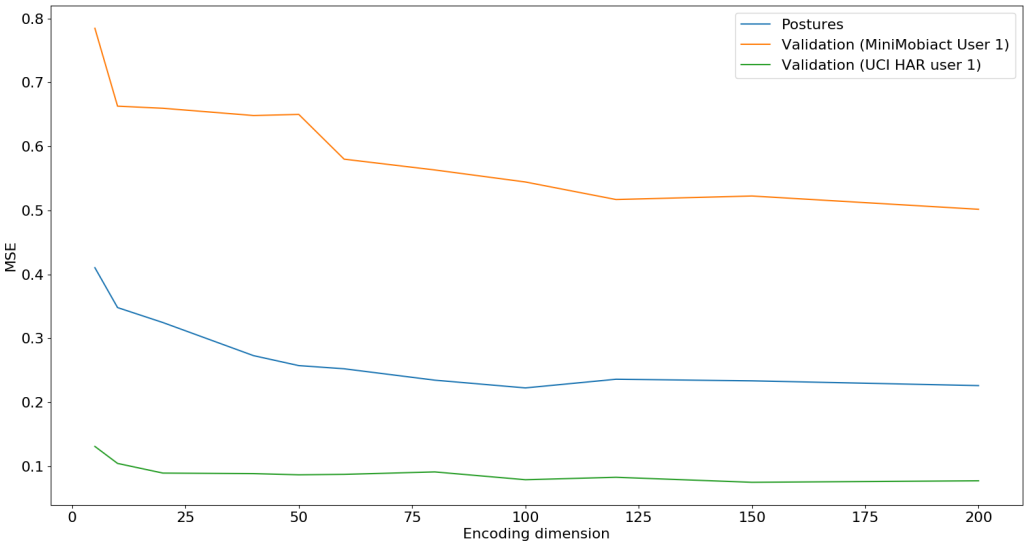


Figure 2.6 – Comparisons of the MSE achieved on *Postures* validation set (user 9) and the validation set of each other dataset for different encoding dimensions.

pretraining an encoder with another dataset. We will reuse this approach when testing SSMN in the next section.

2.3.3 Personalized Activity Recognition

After these two preliminary experiments, we are ready to test the SSMN architecture on *MiniMobiAct* and UCI HAR.

2.3.3.1 Training Strategies and Protocol Details

The protocol to train matching networks differs significantly from a classical machine learning protocol. It is not trained to classify but to produce a one-shot learning classifier at test time by matching the test samples with one or several support examples per class. The training of such algorithm needs therefore to be independent from the classes and from the support examples: thus, classes and not instances are sampled. For example, for *MiniMobiAct*, nine classes and their corresponding sequences are randomly chosen to be part of the training set, the remaining three are part of the test set (see Table 2.4 for a summary of the main parameters for both datasets). During the training, the batches are composed of disjoint classes and not of a specific number of sequences. We chose a batch size of three to match the number of

classes to predict at test time (respectively, 2 for UCI HAR). In a batch, one instance of each class is randomly selected to be a support example. After each batch, the parameters of the model are updated by computing the log likelihood loss on the remaining sequences of the batch classified with the support examples. At test time, one support example from each test class is chosen to be a support example, the other instances are classified. Therefore, at test time, Ş is a one-shot classifier (one labeled support example per class) which does not require any more training and every validation or test results subsequently presented are one-shot learning results on classes never seen at training.

To train this model, on both datasets, we used a learning rate of 0.001 which was divided by 2 every 10 epochs without decreasing of the training loss. The value of the gradient was also clipped to 6 to avoid exploding gradient in accordance with [156]. The GRU were initialized with orthogonal weight matrices scaled to have a spectral radius of 1.1 in accordance with [196]. We used the *Postures* dataset as a support dataset to improve the performance of one-shot activity classification. A sequence-to-sequence model is trained on it and only the encoding part is kept. For this model, we used a learning rate of 0.01 and a batch size of 10. To improve generalization capacity, we trained the decoding part as a denoiser by randomly setting to 0 the values in the input sequence with probability of 0.3 [214] which produces more robust features. A similar initialization as mentioned above is also done. In the validation phase, three classes of *Postures* are randomly selected to match the batch size.

We report in the following the accuracy and f1-score[4] by concatenating the results of several train-then-test phases and computing a global score after 40 iterations (thus, on about 400 classifications, depending on the sampling for *MiniMobiAct*). The evaluation scores are computed in two different ways. In the *partial* setting, the models are evaluated only on never seen classes: the model is given one new support example (one-shot learning) of the never-seen classes and tries to classify test instances into these new classes only (following the batch size, this is therefore a 3-way accuracy for *MiniMobiAct* and a 2-way accuracy for UCI HAR). In the *full* setting, the model is given support examples for the new classes and also the classes used at training (randomly selected, once again) and tries to classify the same test instances but can choose between all the classes, not only the new ones (this is therefore a 12-way accuracy for *MiniMobiAct* and a 6-way accuracy for UCI HAR). Nevertheless, the pre-

[4]Accuracy corresponds to f1-score micro-averaged for multiclass classification so we report f1-score macro-averaged, which does not take into account class imbalance.

Dataset	Mobiact V2	UCI HAR
Total #classes	12	6
#Training classes	9	4
Batch size (classes)	3	2
Initial learning rate	0.001	0.001
#Test users	18	29

Table 2.4 – Summary of the main model parameters for SSMN.

sented scores are always for all the classes due to the 40 test iterations being concatenated and the selected test classes being different each time.

2.3.3.2 Validation of SSMN Components for *MiniMobiAct*

We subsequently validate each component or the SSMN architecture on the first user of *MiniMobiAct*. We first decide if fine-tuning is effective and the number of processing steps of the Attention GRU to be performed. We begin with only one processing step and keep the best configuration after each stage. Then, we validate the interest of using the *Postures* dataset as a support dataset.

Finetuning

First, we evaluate the interest of finetuning a linear layer after the output of the pretrained encoder of our SSMN architecture. The encoder provides representations of size 100: this size was chosen according to the reconstruction error achieved on the *Postures* dataset during the pretraining. When using this encoder, the representation can either be used as is or passed through a linear layer which will be finetuned with the training data coming from the user we are learning a model for. We tested two output layer sizes, 20 (a small version) and 100 (the same output size). The results are presented in Table 2.5a. We observe that the architectures labeled `a1` for cosine distance with 0.856% of 3-way accuracy and `a2` for euclidean distance with 0.825% of 3-way accuracy, got the best results and we therefore select them for the following experiments. We observe that in both cases, the small size got the worst results with significant degradation indicating that this representation size does not seem to preserve enough information.

Number of attention GRU processing steps

Distance	Finetuned layer size	3-way accuracy	3-way f1-score
cosine	no finetuning	0.812	0.79
cosine	20	0.819	0.763
cosine (a1)	100	**0.856**	**0.823**
euclidean (a2)	no finetuning	**0.825**	**0.81**
euclidean	20	0.750	0714
euclidean	100	0.793	0.726

(a) With pretraining on the *Postures* dataset and different finetuned layer sizes.

Distance	Network architecture	3-way accuracy	3-way f1-score
cosine	[100]	0.745	0.696
cosine	[100, 20]	0.694	0.676
cosine	[100, 100]	0.733	0.709
euclidean	[100]	0.8	0.758
euclidean	[100, 20]	0.763	0.736
euclidean	[100, 100]	0.784	0.754

(b) Without pretraining on the *Postures* dataset and different architectures.

Table 2.5 – Classification accuracy on *MiniMobiAct*, user1, with different matching network architectures.

Next, we investigate the parameter p, *i.e.* the number of processing steps of the attention GRU. We tested four values of processing steps between 1 and 10, the largest value experimented in [216]. The results are presented in Table 2.6. We observe that for the cosine distance 0.858% of 3-way accuracy could be achieved (b1) with 10 processing steps and that for the euclidean distance, a maximum of 0.839% could be achieved (b2). We globally notice that better values are achieved with more processing steps as in [216]. Also as in [217], this is a very slight improvement. The batch size used here is only three, and a more important impact should be expected when trying to work with more classes. We kept these parameters for the remaining experiments.

Impact of pretraining

Now that components of the architectures and their parameters have been validated, we aim at measuring the exact impact of using the *Postures* dataset to pretrain our model. We thus propose two experiments. First, the same GRU encoder is trained by only using the *MiniMobiAct* training data (so not as a sequence-to-sequence model). The results are

Distance	Processing steps	3-way accuracy	3-way f1-score
cosine (a1)	1	0.856	0.823
cosine	3	0.793	0.802
cosine	5	0.779	0.783
cosine (b1)	10	**0.858**	**0.826**
euclidean (a2)	1	0.825	**0.81**
euclidean	3	0.777	0.759
euclidean (b2)	5	**0.839**	0.805
euclidean	10	0.814	0.809

Table 2.6 – Classification accuracy of SSMN variants on *MiniMobiAct*, user 1, with different numbers of processing steps.

shown in Table 2.5b. The proposed architectures correspond to the ones experimented for the finetuning where a layer of size 100 had already been learned and frozen. We observe that none of those architectures could outperform the results obtained by the best ones for each metric (b1 and b2). The difference is more flagrant for the cosine metric where a more than 10% improvement could be achieved with pretraining on the Postures dataset. These results show the benefit of pretraining an encoder as a sequence-to-sequence model, on a different dataset containing similar inertial data even if both have no activities in common.

Impact of early stopping

The other interesting property of matching networks is that they can recognize new classes using just one new sequence. In situations of very few available training data, there may not even be enough data to perform a proper validation or "early stopping" to prevent overfitting. In those conditions, with SSMN, another dataset can be used as validation set, here the *Postures* dataset. We trained several models (using the same parameters as b1 and b2) with fixed number of epochs to compare the results with those previously obtained with early stopping based on the performance on the *Postures* dataset. The results are presented in Table 2.7.

While the advantage for the cosine metric seems not significant, even non-existent, we remark that the models using the Euclidean distance clearly overfit and results are worse than those obtained with early stopping (on Table 2.6 and decreasing over 50 epochs. The use of the *Postures* dataset as a validation set improves the training in this case since

Distance	Max epochs	3-way accuracy	3-way f1-score
SSMN b1	20	0.845	0.795
SSMN b1	50	0.791	0.762
SSMN b1	100	0.855	0.833
SSMN b2	20	0.705	0.668
SSMN b2	50	0.753	0.709
SSMN b2	100	0.645	0.606

Table 2.7 – SSMN early stopping comparison on *MiniMobiAct*, user 1.

Algorithms	3-way accuracy	12-way accuracy
SSMN b1	**0.755±0.084**	**0.533±0.103**
SSMN b2	0.742±0.085	0.505±0.099
	3-way f1-score	12-way f1-score
SSMN b1	**0.741±0.087**	**0.522±0.099**
SSMN b2	0.734±0.081	0.494±0.094

Table 2.8 – Test scores on *MiniMobiAct*, average of 18 users.

stopping the training at the exact right moment improves the validation results.

2.3.3.3 Test Results of all users on *MiniMobiAct*

We now apply b1 and b2 using a pretrained encoder on *Postures* dataset and early stopping of the training also on *Postures* dataset to every other users of *MiniMobiAct* V2 having more than 50 sequences (18). To recall some results from the state of the art, Chatzaki et al. [41] achieved 97% accuracy with an instance-based k-nearest neighbor classifier and heavily relying on signal processing techniques and manual feature selection. DiPietro et al. [65] achieved 71% accuracy with their MIST approach. It is not possible to directly compare these results to those obtained with matching nets since the protocols are completely different: at test time, matching net learns in one shot to predict three new classes whereas the other models were trained on every class. Thus, these numbers are only a rough indications. The test results on 18 users are presented in Table 2.8.

The best results are achieved by the cosine metric with 75.5% 3-way accuracy even if the gap with the Euclidean distance can be considered as non-significant regarding the standard deviations. To make comparisons with the 71% achieved by DiPietro et al. [65], a normally trained neural

network model, our model achieved a maximum 12-way accuracy of 0.533 with the cosine. Apart from the amount of data, this difference may also be explained by the large difference of the support set size between the training and the testing (three to twelve) for which the context embedding may not be really adapted.

We see on Figure 2.7a that about 25% of the users having more than 80% accuracy and about 50% more than 75%. On the 12-way classification, our model achieved more mixed results as could be expected. However, we see on Figure 2.7b that two users achieved at least 70% 12-way accuracy with the cosine metric. These results demonstrate the capacity of SSMN to efficiently classify new classes from only examples. The worst performing user for the Euclidean distance is actually always the same across the figures. These performances seem inherent to the quality of data gathered for this user which indicates that, although SSMN can work with few data, it still requires good quality data. Concerning the difference between cosine metric and Euclidean distance, the results seem in contradiction with what was observed by Snell et al. [182] [5] which they explained by Euclidean distance being a Bregman divergence [12] contrary to the cosine metric. However, this is coherent with what we observed with other metric learning architectures making use of GRU [52] where cosine metric systematically outperformed the Euclidean distance.

On Figure 2.8, we show the confusion matrices obtained by the best users for each metric. We observe difficulties for similar classes[6] namely STN, JOG or CSI but the errors are not the same. For example, STN is confused with JOG or JUM. In the case of the cosine metric, 9 classes out of 12 achieved more than 90% 3-way prediction accuracy.

Finally, we propose to analyze the results as a binary classification of falls vs. non falls similarly as in [51] where 0.808 of f1-score and 0.878 of Accuracy could be achieved on personalized fall detection. The results are presented in Table 2.9. We observe gains for both models on both metrics, the most important being the cosine metric with a gain of 3% accuracy but also 11% of f1-score (macro) which means a great

[5] Snell et al. also proposed a model for few-shot learning called prototypical networks. Actually, their approach coincides with matching networks in the one-shot scenario which we performed here.

[6] Class legend, JOG: Jogging ; JUM: Jumping ; STU: Stairs up ; STN: Stairs down ; SCH: Stand to sit ; CHU: Sit to stand ; CSI: Car-step in ; CSO: Car-step out ; FOL: Forward-lying ; FKL: Front-knees-lying ; BSC: Back-sitting-chair ; SDL: Sideward-lying

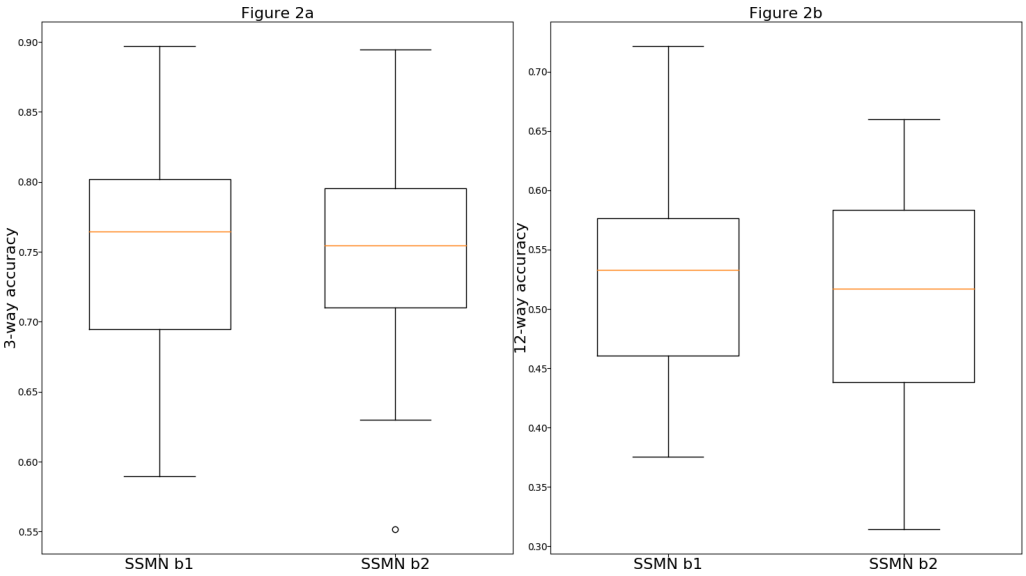
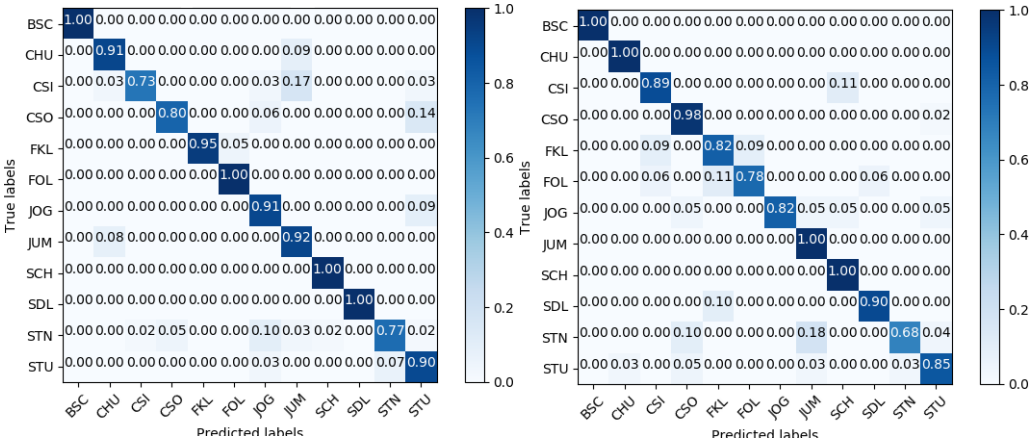


Figure 2.7 – Results per user dispersion for all activities for 3-way accuracy (a) and 12-way accuracy (b).



(a) SSMN `b1`, User 5 with 89.7% 3-way accuracy

(b) SSMN `b2`, User 6 with 89.4% 3-way accuracy

Figure 2.8 – Confusion matrix of best users for both metrics.

Algorithms	3-way fall accuracy	12-way fall accuracy
SSMN `b1`	0.928±0.053	**0.922±0.057**
SSMN `b2`	0.927±0.049	0.896±0.0876
	3-way fall f1-score	12-way fall f1-score
SSMN `b1`	0.891±0.071	**0.879±0.076**
SSMN `b2`	0.885±0.071	0.848±0.109

Table 2.9 – Test scores for fall detection on *MiniMobiAct*, average of 18 users.

Distances	Finetuning	# Training epochs
Cosine (`c1`)	20	100
Euclidean (`c2`)	20	50

Table 2.10 – Parameters validated on User 1 of UCI HAR dataset.

improvement of the recognition of falls, even in a one-shot learning setting. We can also compare those results to the work from Tsinganos et al. [205] who achieved a recall of 97.53% and a specificity of 94.89%. Our one-shot learning approach reached only slightly inferior results compared to this general approach.

2.3.3.4 Test Results of all Users on UCI HAR Dataset

We finally present test results on 29 users from the UCI HAR dataset. This dataset has less classes than *MiniMobiAct*, they also largely intersect the classes of *Postures*. To be concise, we validated the parameters reported in Table 2.10 on the data of user 1 for both distances. These architectures are hereafter named `c1` and `c2`. AttentionGRU (cf. Algorithm 1) was not used for this dataset since it lead to weaker results. This may be due to the low number of vectors in the support set with a batch size of 2 which provokes overfitting since we observed slightly better results when trying to classify one element in the 6 classes (*i.e.* using a support set of size 6). Similarly, early stopping based on the *Postures* dataset leads to slightly inferior results for user 1 and was therefore replaced by a fixed number of training epochs. We report, in Table 2.11, the test results averaged over the 29 other users.

We observe that the Euclidean distance got slightly better results than the cosine metric, SSMN achieved an average 2-way accuracy of 0.8 with 16 users out of the 29 over 0.8 and one user over 0.9. For the

Algorithms	2-way accuracy	6-way accuracy
SSMN c1	0.755±0.096	0.669±0.07
SSMN c2	**0.80±0.081**	**0.706±0.65**
	2-way f1-score	6-way f1-score
SSMN c1	0.725±0.093	0.63±0.08
SSMN c2	**0.789±0.072**	**0.665±0.077**

Table 2.11 – Test scores on UCI HAR, average of 29 users.

6-way scores, they culminate at 0.706. Best state-of-the-art approaches achieve around 95% accuracy on the complete dataset. San-Segundo et al. [173] achieved this with a user adaptation approach. We think that this dataset may actually bring three obstacles for our approach. Firstly, the sequences are short (128 points, less than 3 seconds) which decreases the quantity of discriminant information in each sequence and therefore makes matching more difficult. Secondly, some classes can be very difficult for SSMN to learn to differentiate from just one or few sequences as the data may look very similar: e.g. "walking upstairs" and "walking downstairs", "sitting" and "lying". This is particularly visible on the 6-way score values. Thirdly, the dataset has no magnetometer data, which is not a problem in itself but could create compatibility problems with *Postures*.

2.4 Conclusions and Perspectives

We presented in this chapter an approach for personalized ADL classification called SSMN based on matching networks combined with sequence-to-sequence pretraining. This approach presents two major advantages which make it very relevant to be implemented for real applications. First, it addresses the problem of limited training data that is encountered when learning personalized models by being able to learn from just a few examples. Second, it is very versatile regarding each new activity a user could perform by being able to learn it just from one example. When properly trained, SSMN is indeed independent from the classes it was trained on and only relies on the provided support set. As a consequence, its performances can be boosted with any dataset from the literature which possesses similar characteristics even if the features are not exactly the same and the activities different. This poses several difficulties regarding the sampling rate and the sequence length (or the sliding window length) but also the standardization of the features and the features

themselves which were not exactly the same (in this regard, UCI HAR posed the most compatibility problem as it has no magnetometer data). Nevertheless, this proves efficient to improve the results in some extend. With this model, we achieved over 75% of 3-way accuracy, a performance comparable to those obtained by classical neural network models trained on the whole *MiniMobiAct* dataset. Although the 12-way accuracy was actually much lower, it still proves the interest of this approach to quickly deploy actigraphy systems based on activity recognition. Those results were particularly good for "fall" classes with over 90% 3-way accuracy and 12-way accuracy, meaning SSMN is a relevant approach to detect any kind of falls. With a second experiment, we showed that this approach could be extended to another dataset but also that it presents some limits despite its great flexibility.

We tried in this chapter to propose a model adaptable enough to be quickly tested and deployed in real environments with few labelisation efforts. The next step would therefore be to test it in a realistic context and see if it keeps its promises. However, this quest for flexibility, versatility and adaptability to users and environment can be pushed further: is it possible to give up completely on the labels and to transition from a system monitoring what the person is doing to a system monitoring if the person does what he/she usually does? We tackle this problematic in the next chapter by the mean of metric learning.

CHAPTER 3

ROUTINE RETRIEVAL WITH SEQUENCE METRIC LEARNING

HAR is a key part of several intelligent systems interacting with humans: smart home services [56], actigraphy and telemedecine [151], sport applications [9], etc. It is particularly useful to develop eHealth services and to monitor a person in its everyday life. It has been so far mainly performed in supervised contexts with data annotated by experts or with the help of video recordings (e.g. [41]). Not only is this approach time consuming, but it also restricts the number of activities that can be recognized. It is associated with scripted datasets where subjects are asked to perform actions or activities. This scenario is thus unrealistic and difficult to set up for real environments where people do a vast variety of specific activities everyday and can diverge from a pre-established behavior in many different ways (e.g. falls, accidents, contingencies of life, etc.). In the previous chapter, we proposed to employ few-shot learning to overcome these issues by building highly adaptable models. We will now try to make another step toward less supervision with metric learning - which was, in some way, already a component of the Matching network architecture - and information theory.

In this chapter, we advocate for the modeling and detection of routines instead of classical supervised activity classification, and we propose a machine learning model able to identify routines in the daily recorded motion data of a person. Routines do not need to be semantically characterized, and the model does not have to use any activity labels but rather timestamp metadata. To address this problematic, we first observe routines may present characteristics of

almost-periodic functions, periodic similarity, regarding a certain metric which we propose to learn. To do so, we built on the siamese neural network architecture proposed by Bromley et al. [30] and propose to jointly learn a representation and a metric for time series using a siamese Seq2Seq model. We propose experiments to evaluate the quality of the learned metric on the problem of routine retrieval based on clustering and information theoretic scores.

The remainder of the chapter is organized as follows. Section 3.1 is dedicated to the formalization of the routine retrieval problem from both literature and theoretical perspectives. Section 3.2 gives an overview of metric learning with a focus on sequences and compatible neural network loss functions. The proposed neural network model to recognize routines is presented in Section 3.3 and Section 3.4 presents experimental protocols and results. Finally, conclusions and perspectives are drawn in the last section.

3.1 Tackling Routines instead of Activities

What are routines ? Which data can be used to build a machine learning model tackling them ? And which machine learning tools are adapted to recognize them ? We will first try to answer these questions by studying the literature and we then propose a mathematical formalization of this concept.

3.1.1 Routines in the Literature

Everybody presents some kind of recurrent behaviors in their daily life, called *routines* or at a more general level, *habits*: the time they go to sleep, morning rituals before going to work, meal times, etc. This was notably studied by Wood et al. [227] who observed that between a third and a half of the behaviors they studied were reported as habits. Results coming from behavioral psychology show that habits are hard and long to form but also hard to break when well installed [118]. For psychologists, habits do not designate the behavior itself but rather the link in the memory between a context and a response: habitual behaviors are triggered by certain time and places [75]. Habits are therefore automatic responses that can be triggered on the long-term and should produce distinguishable patterns in the data collected by wearable sensors.

From a data-driven perspective, Gonzalez et al. [83] observed the high regularity of human trajectories thanks to localization data and

concluded that "humans follow simple reproducible patterns". Furthermore, the literature suggests the use of different types of data to deal with habits. Banovic et al. [13] used location data labeled thanks to interviews with the users and proposed a decision-theoretic approach to model the causality between routines and detect variations. Xiong et al. [234] used location data to retrieve the routines by measuring the similarity in terms of correlation between sequences of locations. Qin et al. [163] also proposed to mine routines from location data obtained from a smartphone. They decomposed the human daily behavior into three levels: routine[1] patterns (the "theme of the day": day of work, weekend), one-day patterns (variations of the same routine) and trajectories that are effectively recorded. Each day is sliced into time slots containing trajectories which then constitute the one-day pattern. This model is then transform into three probability distributions which are learned with maximum likelihood. Huỳnh et al. [101] used the data recorded by two accelerometers during 16 days in order to search for activity patterns in the data. The data were annotated at two levels: routines and detailed activities, and the authors used topic modeling[2] to retrieve the routines here considered as topics after detecting the activities with a supervised classifier.

In terms of modeling of routines, we remark that several approaches adopted a hierarchical view which fit the idea that routine behaviors can appear at several temporality levels (e.g. every week, every day, at each time a specific activity is performed, etc.). According to us, this way of doing lacks proper mathematical foundations which could indicate a natural way to retrieve routines. It can also seem arbitrary since various level organizations can be imagined. Finally, it nearly forces to separate preobserved behaviors which can vary depending on the user and appears not very flexible. Regarding the data, using the localization seems rather intrusive and limited, particularly when people are at home which moreover requires specific approaches to be accurate ([238], [3]). On the other hand, wearable inertial sensors data have proven useful to recognize activities, especially when they are embedded into smartphones [218]. On the labelization, carrying on interviews with the people on their whole daily life or watching

[1]Their use of the word routine is actually much more general than ours since we designates as routines daily behaviors and not the whole type of day.

[2]Topic Modeling is a Natural Language Processing (NLP) approach used to identify the subjects of several documents from the words it contains [97] with latent Dirichlet allocation for example [23].

videos appears also disrespectful of privacy and time consuming to set up while the time regularity alone of the patterns could allow to retrieve them without the need for semantic information. This opens the gate for semi/unsupervised machine learning algorithms instead of supervised classification. Routines produce a relevant signature of the daily life of a person which can then be used to monitor him/her without knowing what he or she is exactly doing. The detection of routines with semi/unsupervised machine learning seems therefore a relevant approach to conceive privacy preserving eHealth services.

3.1.2 Formalization of the Concept of Routine

To tackle routines with machine learning, we propose a starting principle similar to the one used in NLP: *similar words appear in similar contexts* [35]. The context surrounding a word designates the previous and following words of the sentence, for example. The context of a routine corresponds here to the moment of the day or the week, etc. it generally happens.

Principle 3.1. *Similar routines occur at similar moments, almost periodically.*

From this principle, we seek now to propose a mathematical formulation of routines which would include the notions of periodicity and similarity. The almost periodic functions defined by Bohr [26] show similar properties:

Definition 3.1. Let $f : \mathbb{R} \to \mathbb{C}$ be a continuous function. f is an almost-periodic function with respect to the uniform norm if $\forall \epsilon > 0, \exists t > 0$ called an ϵ-almost period of f so that:

$$\sup |f(x+t) - f(x)| \leqslant \epsilon. \tag{3.1}$$

Obviously, the practical scenario of routine retrieval presents several divergences from this canonical definition: data are discrete time series and the periodicity of activities cannot be evaluated point-wise. Nevertheless, it is possible to adapt it to our problem. Let $S : \mathbb{N} \to \mathbb{R}^n$ be an ordered discrete sequence of vectors of dimension n. If the frequency of S is sufficiently high, it is possible to get a continuous approximation of it, by interpolation for example. We now consider a function f_S of the following form with a fixed interval length v:

$$\begin{aligned} f_S : \mathbb{R}_+ &\to \mathbb{R}^{n \times l} \\ x &\mapsto [S(x) : S(x+v)[, \end{aligned} \tag{3.2}$$

where $[S(x) : S(x+v)[$ is the set of vectors between $S(x)$ and $S(x+v)$ sampled at a certain frequency from the continuous approximation. The interval v is typically one or several hours: a sufficiently long period of time to absorb the little changes from one day to another (e.g. waking up a little earlier or later, etc.). The objective is to define almost-periodicity with respect to a distance d between sequences, such that $\forall \epsilon > 0, \exists t > 0$:

$$\begin{aligned} & d(f_S(x), f_S(x+t)) \leqslant \epsilon \\ \iff & d([S(x) : S(x+v)[, [S(x+t) : S(x+v+t)[\leqslant \epsilon. \end{aligned} \tag{3.3}$$

The parameter t can be a day, a week or a sufficiently long period of time to observe repetitions of behavior. The chosen metric d must be able to handle the high variability of activities which can be similar but somewhat different in their execution while exhibiting similar patterns (e.g. cooking every day). We therefore postulate that d may be learned for a specific user from its data and we will now show that f_S respects the condition established in Equation (3.3) with respect to d. To learn d if pairs of similar and dissimilar sequences are known, a RNN encoder parameterized by W, called net_W, can encode the sequences into vector representations and the contrastive loss [86] can be used to learn the metric from pairs of sequence encodings:

$$\begin{aligned} \mathcal{L}_{\text{contrastive}}(S_1, S_2) = (1-l)\frac{1}{2} d(\text{net}_W(S_1), \text{net}_W(S_2))^2 \\ + l\frac{1}{2}\max(0, m^- - d(\text{net}_W(S_1), \text{net}_W(S_2))^2, \end{aligned} \tag{3.4}$$

where l is equal to zero or one depending if the sequences are respectively similar or not and $m^- > 0$ a margin that defines the minimal distance between dissimilar samples. Several justifications arise for the use of a margin in metric learning. It is necessary to prevent flat energy surface, according to energy-based learning theory [124], a situation where the energy is low for every input/output associations, not only those in the training set (see Section 3.2.1.1). It also insures that metric learning models are robust to noise and a large margin is correlated with better generalization capacity of the model [222]. As the learning process aims to minimize the distances between similar sequences which are, by definition, shifted by a period t, we get, for a fixed $t > 0$ and $\forall x \in \mathbb{R}_+$:

$$d(\text{net}_W(f_S(x)), \text{net}_W(f_S(x+t))) \leqslant m^+. \tag{3.5}$$

The margin m^+ is not always involved directly into the loss (see the Section 3.2 about metric learning) but can be theoretically chosen as

close to zero as possible by increasing m^- and thus Equation (3.5) identifies itself with Equation (3.3). In practice, this optimization is only possible up to some point: if the model possesses a sufficient number of parameters, the training set will be perfectly learned but a generalization error is to be expected on the validation and testing sets [209]. This argumentation suggests the interest of modeling routines with metric learning as, in this case, the main property of almost-periodic functions is fulfilled. We will now dig into the literature of metric learning, especially dealing with sequence metric learning in order to develop an architecture implementing this formalization.

3.2 Metric Learning State of the Art with a Focus on Sequences

3.2.1 What is Metric Learning

3.2.1.1 Generalities

Metric learning is a field of machine learning which aims to produce algorithms able to learn distances between data samples, here taken in a broad sense (vectors, images, sequences, etc.). The metric can then be used to classify or cluster new data samples. This can be achieve with a large number of approaches using representation learning [20], convex optimization [222], eigenvalues based methods [15], neural networks [30], etc.. In fact, these domains are highly connected. One could argue that neural networks only perform representation learning tasks [123]. Dimensionality reduction algorithms (Principal Component Analysis, Autoencoder, Laplacian Eigenmaps [17], *etc.*) which aim to produce low vector representations of larger ones, can also be considered representation learning algorithms. In that way, learning a metric is both choosing or learning the best distance formula but also choosing or learn the best representation for the samples. Most of the time, this representation is a n-dimensional vector even if certain distances like edit distances for strings [166] or graphs [74] can be used directly on the original form of the data. While encompassing a variety of data structures and approaches, distance is a well-defined concept of mathematics:

Definition 3.2. Let $\mathbb{E}$ be a set of data samples, let $d : \mathbb{E} \times \mathbb{E} \to \mathbb{R}^+$ and let $(x_1, x_2, x_3) \in \mathbb{E}^3$. The function d is a distance function (or metric) if it

satisfies the three following properties:

$$\text{Identity: } d(x_1, x_2) = 0 \iff x_1 = x_2 \tag{3.6}$$
$$\text{Symmetry: } d(x_1, x_2) = d(x_2, x_1) \tag{3.7}$$
$$\text{Triangular inequality: } d(x_1, x_2) + d(x_2, x_3) \geqslant d(x_1, x_3). \tag{3.8}$$

This definition is canonical but some "metrics" we will come across do not satisfy all the properties: e.g. Dynamic Time Warping (DTW) [171] used for sequences does not respect triangular inequality. These properties offer guaranties and can be used to perform optimizations such as the ball-tree algorithm used to accelerate k-NN [71] which requires triangular inequality. We nevertheless call them metrics for the sake of simplicity. Three traditional metrics forms respecting the previous definition can be used on vector representations: Euclidean, cosine and Mahalanobis.

Through two decades of research, multiple views of the metric learning problem have been developed, not mutually exclusive but each introducing new concepts and ideas. If metric learning could be seen at the beginning as a variant of a classification problem where elements of the same classes were made close, some advances would transform it into what can be seen as a semi supervised approach: if supervision is still necessary at some point, it is relaxed in many ways compared to a true multiclass classification. This was notably experimented by Xing et al. [232] who proposed to label the data with equivalence constraints instead of classes, that is to say that groups of samples are qualified as similar, others (not mutually exclusive) are said to be dissimilar which allows to build dataset with rather blurry notions of similarity like metadatas, information theory or other more simple metrics, etc.. This labeling can therefore be more easily obtained without human expertise. Pushing further into this direction, Perrot et al. [161] proposed an alternative to equivalence constraints which they called virtual metric learning. Instead of making close pairs of similar samples, they propose to push toward a single virtual point every similar samples. This optimization problem has the advantages to be solvable with a closed-form and it also diminishes the number of constraints to be applied to the samples. Several virtual point construction procedures can be envisaged: with optimal transport, good representative sample of the class, one-hot vectors, etc..

Finally, another interesting framework called energy-based learning [124] highlights the interest of a central concept in metric learning:

margin. In this framework, an energy function associates an energy level to each configuration of the inputs and outputs of the model which we denote in the following by $E(x, y)$. This energy function is not to be confounded with the loss function even if the most simple loss function that can be derived from the energy function is itself (energy loss). Correct associations are associated with low levels of energy and conversely. Thus, when trying to process new inputs, we should simply pick the $\hat{y}$ value associated with lowest energy to x. However, to be effective, not only must the loss associate low levels of energy to correct associations but also creates an energy gap between the good and bad combinations. This way, bad combinations are associated with relatively higher energy levels than the good ones. This gap can be created by including a *margin* into the loss and we will now illustrate its utility by applying energy-based learning to metric learning. One loss designed with energy-based learning principles will prove highly influential for metric learning: the contrastive loss designed by Hadsell et al. [86] (see Equation 3.4). This loss incorporates a contrastive term to separate dissimilar examples under the form of a Hinge loss:

$$\mathcal{L}_{\text{Hinge}}(\mathcal{B}) = \sum_{(x_i, y_i) \in \mathcal{B}} [m + E(x_i, y_i) - E(x_i, \hat{y}_i)]_+ \tag{3.9}$$

$$= \sum_{(x_i, y_i) \in \mathcal{B}} \max(0, m + E(x_i, y_i) - E(x_i, \hat{y}_i)), \tag{3.10}$$

where $\hat{y}_i$ is produced by the model from x. By using a distance d as energy function, and dissimilar input pairs, we can rewrite the Hinge loss for metric learning as a contrastive term:

$$\mathcal{L}_{\text{Hinge}}(\mathcal{N}) = \sum_{(x_i, x_j) \in \mathcal{N}} [m - d(x_i, x_j)]_+. \tag{3.11}$$

Without the contrastive term, the loss would just impose that similar pairs get low energies (distances). But, when learning from pairs of inputs, nothing would prevent the model from just ignoring the inputs and learning to output always the same value, resulting in a low distance whatever the inputs. This justifies to also constraint the energies (distances) between dissimilar inputs to be higher than the margin.

Following this introduction, we will now describe two ways to learn metric making use of equivalence constraints and margins: constraint optimization and siamese neural networks.

3.2.1.2 Constrained Optimization

Constrained optimization[3] is a field of Mathematics which aims at optimizing parameters under some condition(s). One way to pose the metric learning problem to introduce parameters inside the distance formula is to consider a metric of this form, called the Mahalanobis distance [135]:

$$d_\Sigma(x,y) = \sqrt{(x-y)^T \Sigma^{-1}(x-y)}, \tag{3.12}$$

where Σ is the covariance matrix of the point distribution but by the mean of optimization, can be any PSD matrix M (to get a positive value under the square root). It is therefore possible to employ, most of the time, semidefinite programming algorithms [210] to solve this problem under some appropriate conditions. Xing et al. [232] proposed this formulation based on their previously introduced equivalence constraint framework for the data:

$$\min_M \sum_{(x_i,x_j)\in\mathcal{P}} d_M(x_i,x_j)^2 \tag{3.13}$$

$$\sum_{(x_i,x_j)\in\mathcal{N}} d_M(x_i,x_j) \geqslant 1 \tag{3.14}$$

$$M \succcurlyeq 0. \tag{3.15}$$

The second condition (Equation 3.14) defines a margin and ensures with the third (Equation 3.15) that that M does not collapse into the null matrix. Depending on the form of M (diagonal or full), the problem can be easy or hard to solve using semidefinite programming [210]. Weinberger et al. [222] proposed a convex formulation to learn M so to maximize the margin between the samples from different classes: Large Margin Nearest Neighbors (LMNN). A condition on the k-NN accuracy is imposed on the loss since batches are formed locally and only local dissimilar are repulsed from each other. The large margin ensures that the model is robust and generalizes well. The objective function to

[3] These constraints, used here to optimize an objective function are not to be confounded with the equivalence constraints which constraint the dataset itself. We will use the term *conditions* in the following to designate optimization constraints to distinguish them from equivalence constraints.

minimize is therefore the following:

$$\mathcal{L}_{\text{LMNN}}(\mathcal{B}) = \min_{M}\Bigg((1-\lambda)\sum_{(i,j)\in\mathcal{P}} d_M(x_i,x_j) + \lambda\sum_{(i,j)\in\mathcal{P}}\sum_{(i,k)\in\mathcal{N}}[m + d_M(x_i,x_j) - d_M(x_i,x_l)]_+\Bigg). \quad (3.16)$$

This loss function is very similar to Equation 3.4 but cannot be optimized as such with semi-definite programming. It requires to transform the Hinge loss into an inequality by introducing a variable which indicates by how much the margin between similar and dissimilar distances is violated.

Globerson et al. [81] proposed to learn a Mahalanobis distance by trying to project similar points into a single points and dissimilar infinitely far away. Using a softmax, it is possible to construct for each pair of points, the following conditional distribution of probabilities:

$$p_M(x_j|x_i) = \frac{\exp(-d_M(x_i,x_j))}{\sum_{k\neq i\in\mathcal{B}}\exp(-d_M(x_i,x_k)}. \quad (3.17)$$

Ideally, if the distribution corresponds to the geometrical interpretation described above, this probability should be one if x_i and x_j have the same label, 0 otherwise. The authors therefore propose to minimize the Kullback-Leibler divergence between this ideal distribution and the distribution generated by the Mahalanobis matrix M to learn. This problem is convex relatively to M and can thus be solved using various methods [29].

Davis et al. [59] formulated the Mahalanobis distance optimization problem with information-theoretic tools. They used the bijection existing between a Mahalanobis distance and a Gaussian distribution to write the distance between two Mahalanobis distances with the Kullback-Leibler divergence. As a consequence, it allows to optimize M to be close to a predefined Mahalanobis distance M_0 while respecting the metric learning conditions:

$$\min_{M}(\text{KL}(p(x;M_0)||p(x;M))) \quad (3.18)$$

$$d_M(x_i,x_j) \leqslant m^+,\ (i,j)\in\mathcal{P} \quad (3.19)$$

$$d_M(x_i,x_j) \leqslant m^-,\ (i,j)\in\mathcal{N}. \quad (3.20)$$

The regularizing matrix M_0 can be for example a covariance matrix [112]. More conditions can be added to this formulation and it allows also to employ Logdet optimization to improve the performances compared to semidefinite programming as no eigenvalue decomposition is necessary [117].

3.2.1.3 Siamese Networks

Siamese networks were introduced by Bromley et al. [30] and are the basis for many deep metric learning approaches. They consist in two (or more) neural networks sharing the same weights and trained for metric learning tasks (see figure 3.1). The inputs, labeled pairs, are processed using the same function and so the measured distance is symmetric. The weights of each network are updated the same way using metric learning loss functions which are typically of the form of the contrastive loss (Equation 3.4). Therefore, this setup quite differs from the classification setup presented in Section 2.1.4.1 with Equation 2.3. More complex losses are developed in the next sections.

Siamese networks can be trained with back propagation by differentiating the selected metric loss. More formally, the problem modeled by Siamese networks is to find a network parameterized by W such that the energy (the distance) is small if the inputs have the same label, large otherwise [47]. The choice of the distance is an important parameter to allow the model to learn efficiently. Chopra et al. [47] showed that a L_1 metric was better than a L_2 one as the latter could produce plateaus which could slow the training.

Regarding the inputs, they can be vectors of handcrafted features. However, as already stated in Chapter 2 (Section 2.1.4.1), neural networks are able to automatically extract the most relevant features from the raw data and siamese networks are no exception. In fact, metric learning provides powerful objective functions to learn representations [153]. Indeed, this architecture has been applied with great successes combined with CNN to image processing tasks such as person re-identification [236] or object tracking [22]. As an example of enhancement of such approach, Varior et al. [211] proposed a siamese convolutional architecture for person re-identification from video data with a differentiable gated mechanism inspired from LSTM to link parallel layers. The network is this way able to accentuate the common patterns between both representations. This leads to representations that are more suited to distinguish some pairs of similar or dissimilar images.

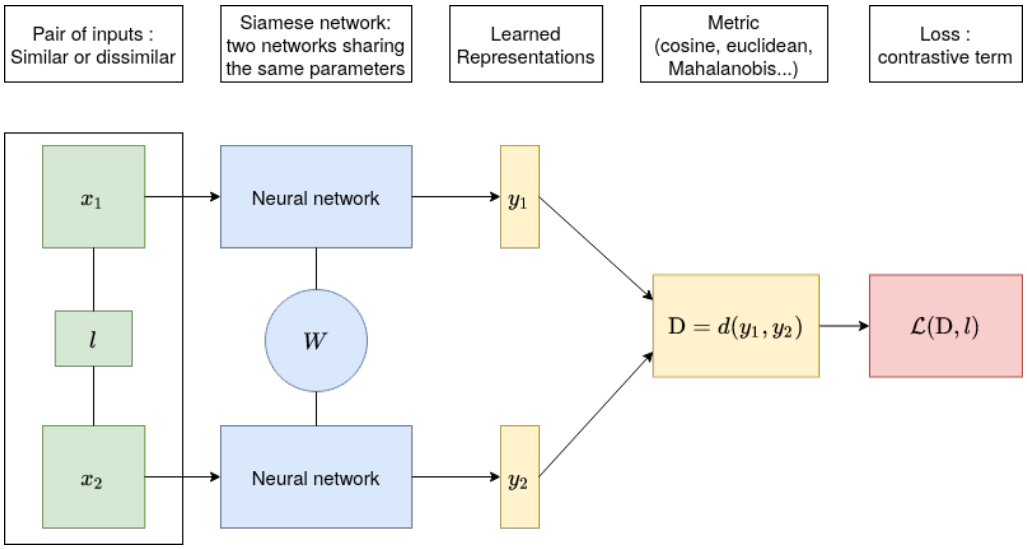


Figure 3.1 – Schema of a Siamese Network [30].

3.2.2 Sequence Metric Learning

We described the essential basic components of metric learning algorithms. However they were either very general or conceived to deal with vectors. The problem formalization presented in Section 3.1.2 involve a distance between sequences. The most straightforward way to compute one is to aggregate Euclidean distances between vectors with the same timestamps. As a consequence, it can only be applied as such on sequences with the same length Therefore, we will now introduce several metric learning algorithms adapted to deal with sequences or time series.

3.2.2.1 DTW and Related Approaches

The traditional approach to compute distances between sequences (or time series, or trajectories) is to perform a DTW [171], an algorithm introduced in 1978. DTW computes the lowest distance between two univariate or multivariate time series while trying to match them (or align them). Each point of each sequence must be matched with one or several closest points of the other sequence monotonically, that is to say by always going forward or matching at the same position. The matching is usually searched in an area (or radius) of a definite length: the Figure 3.2 shows the variations observed in the alignment depending on the size of the radius. The choice of this radius and possibly other

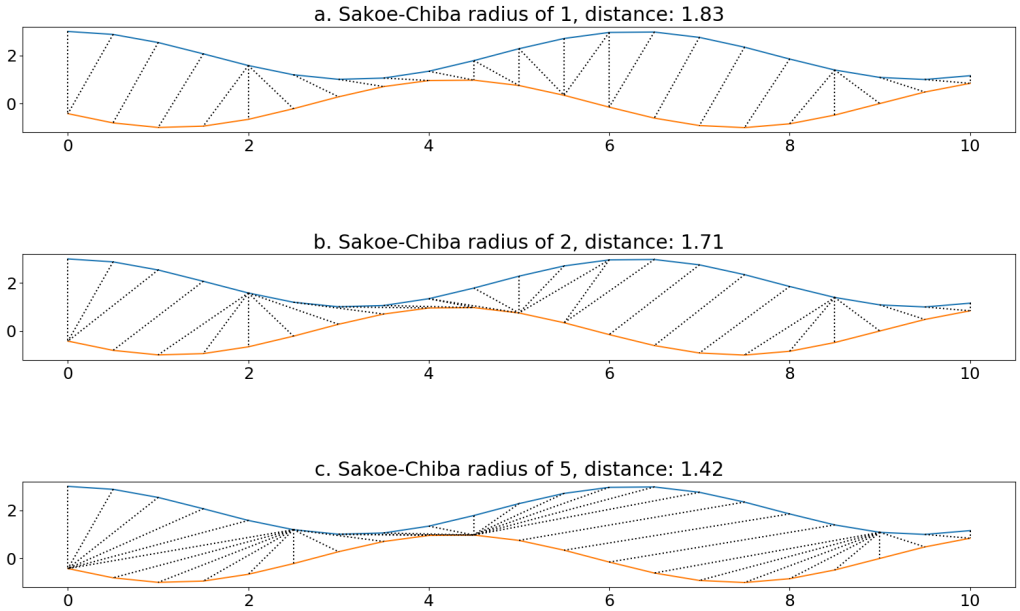


Figure 3.2 – Examples of alignments of two cosine signals shifted by a phase of 2 produced by DTW using 3 different Sakoe-Chiba radius. The distance between the two signals varies depending on the radius but a low distance does not necessarily indicates a meaningful alignment (i.e. respecting the semantic interpretation of the data) and therefore a correct radius value to achieve high classification performances.

various constraints insures the accuracy of the classification and speed up computation. The optimal radius for the dataset is generally cross-validated and then k-NN can be performed using appropriate references [231] to classify test samples. Actually, DTW is considered one of the best metric to use for sequence classification combined with k-NN [231]. Clustering can also be performed notably by computing the affinity matrix of the test set.

DTW is what is called a shape-based distance, as opposed to feature-based and model based approaches, because it computes a distance directly based on the forms of the sequences without any further abstraction. Since then, several improvements of the original algorithm have been published, notably a quasi linear complexity version by Salvador et al. [172]. DTW can also be improved by learning as stated by Nicolae et al. [150] who proposed to learn a parameterized alignment cost function. They were also able to provide theoretical guarantees of generalization for their metric. However, the integration of DTW inside deep architectures has been rendered difficult by its non-differentiability and its

theoretical quadratic time complexity which badly suits the equivalence constraint framework and some associated more complex losses [153, 183, 235] (See Section 3.2.3.2). Recent works mitigate these drawbacks notably with virtual metric learning [161, 191] and soft versions of DTW [37, 58]. Abid et al. [2] defined a family of smooth warping distances, which includes of course DTW but also Euclidean and edit distances, parameterized by only three parameters. They then propose to learn the parameters which allow to reproduce the Euclidean distances between the training sequence latent representations obtained with a Seq2Seq model [197]. They designed an objective function using a clustering score called betaCV which normally is used to compare different cluster assignments with a constant distance: it is the ratio between the average distance between the samples belonging to the same clusters and the average distance between all the samples. Here, the cluster assignment is obtained by setting thresholds on the distances in the latent space and betaCV should be minimum for the distance family parameters corresponding to the same cluster assignment in the sequence space. Finally, the optimization is carried out with gradient descent. This work is a great example of interaction between alignment approach, efficient to measure distances on sequences but originally non-smooth, and a fully-differentiable neural network architecture learning abstract latent representations.

3.2.2.2 Optimization Approaches

Much like for vector data, constrained optimization methods can be used to learn metric for sequences, often Mahalanobis metrics. Sun et al. [192] proposed to learn a Mahalanobis metric on vectors of signal processing features extracted from physiological sequential data. They optimized the ratio between the distances of the k closest similar samples and k closest dissimilar ones, for each sample with a variant of the NewtonRaphson method [105]. Garreau et al. [77] proposed to learn a Mahalanobis metric to be used inside a DTW in place of the Euclidean distance. However, the learning of this matrix is done from groundtruth alignments, which is most of the time, not available. In contrast, Su et al. [190] proposed to employ an alternative to DTW, Order-Preserving Wasserstein (OPW) distance by viewing the problem of metric learning between sequences as an optimal transport problem: the Wasserstein distance (or Earth Movers distance) is the cost of displacing the distribution of the set of points of the first sequence toward the

distribution of the elements of the second sequence. However, doing so, the temporal relationship, the *order*, is lost. Therefore, the authors propose a regularization to preserve this temporal relationship between the samples. The overall problem is solved with the matrix scaling algorithm. Then, in another publication [191], they reformulated the DTW and OPW distances as parameterized meta-metrics of a single ground metric. The ground metric is the only parameter of an optimal transport problem, it is defined by a matrix which indicates the distances between the features to be transported [57]. Such as the Mahalanobis distance, this matrix needs to be PSD so to make optimal transport a distance, and in fact the authors propose here to learn this ground metric as a Mahalanobis distance. Since the optimal transport between each sequence pair depends on the ground metric, the learning process needs to update the metric and the latent alignment separately. To reduce the number of constraints of a problem which would be otherwise intractable, the authors proposed to employ virtual metric learning [161]. Not only this approach speeds up training but it also outperforms several other metric learning approaches, notably approaches conceived for vectors generalized to sequences.

3.2.2.3 Deep Learning and RNN-Based Approaches

Sequences are 2-dimensional structures for which feed forward networks are not adapted: this justifies the extraction of high-level features to produce a vector representation of the structure and the semantic of the data [131]. These vectors can be build with features extracted by various methods such as discrete Fourier or Wavelet transforms and signal processing, etc.. However, this approach loses temporal dependency information inside the sequence and alignment information between the sequences which is very relevant to measure distances as demonstrated by the efficiency of DTW. Sequence metric learning can therefore largely benefits from the automatic feature extraction ability of RNN. Chen et al. [44] proposed an approach based on Echo State Network (ESN) to learn a Mahalanobis metric to classify labeled sequences. ESN [103] are a special kind of RNN composed of a reservoir, a large sparse generally randomly generated matrix which recurrently processes the input sequence and acts as a dynamical temporal filter [204], and a linear readout that can be trained with ridged regression to sequence modeling tasks. Here the authors used a cyclic reservoir with jumps [167] whose parameters are learned via gradient descent. For the metric learning part, they adopted the same view as Globerson et al. [81]: the readouts of the same

classes must collapse in a single same point. The readout parameters can be learned using ridge regression as for a normal ESN while gradient descent allows to learn alternatively the parameters of the reservoir and the Mahalanobis matrix.

Che et al. [42] proposed a deep learning approach called Deep ExpeCted Alignment DistancE (DECADE) able to learn end-to-end a valid metric between sequences with alignments which is normally not the case due to DTW infringing the triangular inequality. Their approach is composed of a feed forward neural network to learn a local representation between the points of the sequences to be used inside the alignment. However, the distance is not computed by using the single best alignment only such as in DTW but rather the expected alignment, that is to say the average of all possible alignment distances between the two sequences which the authors demonstrate is a valid metric regarding the properties stated in Definition 3.2. Thought the number of possible alignments grows exponentially with the length of the sequence, the authors designed sampling strategies to efficiently approximate this average in a polynomial number of steps.

Müller et al. [143] presented a siamese recurrent neural network approach to learn sentence similarities as a l_1-norm. In their method, the LSTM network combines the embeddings of the words of the sentence to learn a distance between representations of sentences. This strategy does not require to compute alignments and following this perspective we introduce in the next section several metric learning losses to be used on learned representations of sequences.

3.2.3 Metric Learning Losses

To train a neural network, a loss function quantifying the error made by the network on the batch is needed. Starting close to the classification problematic, metric learning literature introduced several new own ideas such as complex strategies for batch sampling. These loss functions can be linked with the property of the learned metric and we therefore organized this section following three genealogies of losses for three different metrics: cosine, Euclidean and Mahalanobis.

3.2.3.1 Cosine-Based Loss

Cosine similarity is at first look just a variation of the Euclidean distance. However, the normalization formally transforms a measure of distance into a measure of angle. This particularity was especially leveraged in the

field of NLP to compare vectors of word counting in documents. This counting obviously depend on the length of the documents and cosine similarity allows in this case to make comparisons independently from the length, solely on the content.

Definition 3.3. Let $\overrightarrow{x_1}, \overrightarrow{x_2} \in \mathbb{R}^n \times \mathbb{R}^n$ and let $\theta \in \mathbb{R}$ be the associate angle, the cosine similarity between $\overrightarrow{x_1}$ and $\overrightarrow{x_2}$ is defined by:

$$\cos(\theta) = \frac{\overrightarrow{x_1} \cdot \overrightarrow{x_2}}{||\overrightarrow{x_1}|| ||\overrightarrow{x_2}||}. \tag{3.21}$$

Cosine similarity has been used since the introduction of siamese networks [30] but is still used nowadays notably for face verification and person re-identification. It provides the advantages over the Euclidean distance to be bounded which could lead to more stable learning. Leveraging this property, Nguyen et al. [148] proposed a simple loss to learn an embedding and a cosine similarity for images:

$$\mathcal{L}_{\text{CSML}}(\mathcal{B}) = \sum_{(i,j)\in\mathcal{P}} \cos(Wx_i, Wx_j) + m \sum_{(i,j)\in\mathcal{N}} \cos(Wx_i, Wx_j), \tag{3.22}$$

where W defines a projection and the factor m insures a margin between the positive and negative samples. The authors proposed to solve this problem by optimizing W with the conjugate gradient method [91]. Yi et al. [236] also followed this path but combined the cosine similarity with a CNN architecture to learn the image representations, the whole could therefore be optimized by gradient descent. They expressed their loss using a log-likelihood expression:

$$\begin{aligned}\mathcal{L}_{\text{deviance}}(\{(\mathcal{B})) = \frac{1}{|\mathcal{P}|} \sum_{(i,j)\in\mathcal{P}} \log(1 + \exp(-m_1[cos(\hat{y}_i, \hat{y}_j) - m_2])) \\ + \frac{1}{|\mathcal{N}|} \sum_{(i,j)\in\mathcal{N}} \log(1 + \exp(m_1[cos(\hat{y}_i, \hat{y}_j) - m_2])),\end{aligned} \tag{3.23}$$

where m_1 and m_2 are margin parameters. This loss was chosen by the author because, according to them, it concentrates the learning on the samples close to the decision boundary therefore improving generalization. In another publication, Zheng et al. [241] started from another geometrical interpretation of the cosine similarity by defining the quantity $c_{ij} = \hat{y}_i + l_{ij}\hat{y}_j$, which is the diagonal of the parallelogram

formed by the projections $\hat{y}_i$ and $\hat{y}_j$ to be learned from the inputs x_i and x_j. We thus get, using the triangular inequality:

$$\|\hat{y}_i\|+\|\hat{y}_j\|-\|c_{ij}\|> 0. \tag{3.24}$$

If l_{ij} equals 1 or -1 following if the inputs are similar or dissimilar. Minimizing this quantity makes $\hat{y}_i$ and $\hat{y}_j$ colinear if they are similar, orthogonal otherwise which corresponds to the extreme values of the cosine similarity. As the norms can degenerate to zero, the authors propose to minimize the following objective which makes the norm approach 1 instead:

$$\mathcal{L}_{\text{triangular}}(\{(\mathcal{B})) = \frac{1}{|\mathcal{B}|}\sum_{(i,j)\in\mathcal{B}}(\frac{1}{2}\|\hat{y}_i\|^2+\frac{1}{2}\|\hat{y}_j\|^2-\|c_{ij}\|+1). \tag{3.25}$$

Although not performed by the authors, this loss is smooth and can perfectly be used to train a neural network to extract the representations of the inputs.

3.2.3.2 From Contrastive to Structural Loss

Lots of improvements have been brought to the original contrastive loss designed to learn a Euclidean distance from similar and dissimilar pairs [47]. A straightforward upgrade consists in working with triplets instead of pairs. While this idea was more or less already present in earlier works (e.g. [222]), it was first proposed by Chechik et al. to solve ranking problems [43], then introduced again for facial similarity by Lefebvre et al. [128] and finally fully adapted to neural networks by Hoffer et al. [96]. Triplet loss uses three samples: an *anchor* a, a sample similar to the anchor p^+ and another one dissimilar p^-. This way, the problem is not to make a distance between two samples smaller or larger than a margin but to have the distance with the positive sample inferior to the distance with the negative sample, by at least a margin:

$$\mathcal{L}_{\text{triplet}}(a,p^+,p^-) = [m + d(a,p^+) - d(a,p^-)]_+. \tag{3.26}$$

We now introduce the notion of hard samples: from a theoretical point of view, this designates the samples that are close to the decision boundary. When learning a metric, the "difficulty" of the pair can be assessed with the distance: relatively large distances for positive pairs or small ones for negative pairs. Easy pairs and triplets slow the convergence and should therefore be avoided [176]. It is therefore primordial to

emphasize the learning of the model on hard samples to improve the convergence, the performance and and ultimately the generalization. Building on the triplet loss, Balntas et al. [10] proposed to *swap* the anchor and the other positive sample in order to eventually get a more difficult negative sample. However, the exploration of hard sample mining allowed the introduction of more complex batch sampling strategies. Notably, Oh Song et al. [153] proposed a lifted structured loss which constraints all the distances between all the samples of the batch:

$$\mathcal{L}_{\text{lifted structured}}(f(\mathcal{B})) = \frac{1}{2|\mathcal{P}|} \sum_{(i,j)\in\mathcal{P}} \Bigg[\log\Bigg(\sum_{(i,n)\in\mathcal{N}} \exp(m - d(\hat{y}_i, \hat{y}_n)) + \sum_{(j,n)\in\mathcal{N}} \exp(m - d(\hat{y}_j, \hat{y}_n)) \Bigg) + d(\hat{y}_i, \hat{y}_j) \Bigg]_+^2 . \quad (3.27)$$

The constitution of the batches was also biased toward including hard negative samples relative to the positive ones present in the batch. Sohn [183] proposed a slightly different version with the hope to improve smoothness by replacing the Hinge loss with a log probability, where each sample is once the anchor and is compared to every negative samples:

$$\mathcal{L}_{\text{N-pair}}(f(\mathcal{B})) = \frac{1}{|\mathcal{B}|} \sum_{(\hat{y}_i, \hat{y}_j)\in\mathcal{B}} \Bigg[\log\Bigg(1 + \sum_{n\in\mathcal{B} \mid l_n \neq l_i, l_j} \exp(d(\hat{y}_i, \hat{y}_n) - d(\hat{y}_i, \hat{y}_j)) \Bigg) \Bigg]. \quad (3.28)$$

This loss implies that each batch contains N classes all negative to one another. Therefore, to speed-up the batch sampling, the authors propose to constitute batches of hard negative classes instead of samples by analyzing just a few samples of each class. Finally, prolonging those ideas, Yang et al. [235] proposed a hardness-aware structural loss, composed of two terms a local one and a global one. The first one reassemble very much the N-pair loss (see Equation 3.28) but combines to it a system of pair weighting to emphasize the learning on hard-positive samples (see Equation 3.31 and 3.32): all positive distances above a certain class threshold τ_c will contribute more to the learning than the others (Equations 3.31 and 3.32). The second term, the global one is used to prevent the similar samples to be disseminated in different places inside the feature space and ultimately improve regularization and generalization (Equations 3.33, 3.34, 3.35 and 3.36). To do so, this

term acts on the second order statistics of the distances to diminish the variance.

$$\mathcal{L}_{\text{structural}}(f(\mathcal{B})) = \frac{1}{B}\sum_{(i,j)\in\mathcal{P}} \beta_{ij}\log\left(1+\sum_{n=1}^{|\mathcal{N}|}\exp(d(\hat{y}_i,\hat{y}_j)-d(\hat{y}_i,\hat{y}_n)+m)/\eta\right) + \frac{\lambda}{2}([\sigma_p^2-m^+]_+ + [\sigma_n^2-m^-]_+) \quad (3.29)$$

$$B = \sum_{(i,j)\in\mathcal{P}} \beta_{ij} \quad (3.30)$$

$$\beta_{ij} = \exp(d(\hat{y}_i,\hat{y}_j)) - \tau_c \quad (3.31)$$

$$\tau_c = \frac{2}{|\mathcal{P}_c|}\sum_{(i,j)\in\mathcal{P}_c} d(\hat{y}_i,\hat{y}_j) - \min_{(i,j)\in P_c}(d(\hat{y}_i,\hat{y}_j)) \quad (3.32)$$

$$\sigma_p^2 = \frac{1}{|\mathcal{P}|}\sum_{(i,j)\in\mathcal{P}} (d(\hat{y}_i,\hat{y}_j)-\mu_{\mathcal{P}}^{\mathcal{B}_t})^2 \quad (3.33)$$

$$\sigma_n^2 = \frac{1}{|\mathcal{N}|}\sum_{(i,n)\in\mathcal{N}} (d(\hat{y}_i,\hat{y}_n)-\mu_{\mathcal{N}}^{\mathcal{B}_t})^2 \quad (3.34)$$

$$\mu_{\mathcal{P}}^{\mathcal{B}_t} = \gamma\mu_{\mathcal{P}}^{\mathcal{B}_{t-1}} + (1-\gamma)\mu_{\mathcal{P}}^{\mathcal{B}_t} \quad (3.35)$$

$$\mu_{\mathcal{N}}^{\mathcal{B}_t} = \gamma\mu_{\mathcal{N}}^{\mathcal{B}_{t-1}} + (1-\gamma)\mu_{\mathcal{N}}^{\mathcal{B}_t}. \quad (3.36)$$

In these equation, m, m^+ and m^- are margin parameters. The values $\mu_{\mathcal{P}}^t$ and $\mu_{\mathcal{N}}^t$ are the average distances between respectively the positive and negative samples, γ controls the smoothness of the evolution of theses values between the previous batch $\mathcal{B}_{t-1}$ and the present one $\mathcal{B}_t$. Finally, η is a scaling parameters and λ controls the magnitude of the global loss term in order to avoid it outweighing the local one. We will make use of this last loss to train the model presented in Chapter 4.

3.2.3.3 Mahalanobis Metric Learning and KISSME Loss

Following the works presented in Section 3.2.1.2, several approaches bridged the gap between constraint formulation of Mahalanobis metric learning and neural network implementation. This can be done notably by implementing the projection as the last layer of an encoding neural network. It was notably achieved for the Keep It Simple and Straightforward MEtric learning (KISSME) approach [112] by Faraki et al. [69]. The

starting point for the original idea proposed by Koestinger et al. [112] is to make the hypothesis that a certain pair (x_i, x_j) of inputs is dissimilar (H0) or similar (H1). One way to model the validation of H0 is therefore to write the following quantity which should be high if H0 is validated:

$$\delta(x_i - x_j) = \log\left(\frac{p(x_i - x_j|H0)}{p(x_i - x_j|H1)}\right). \tag{3.37}$$

By modeling these probabilities with Gaussian distributions, we are allowed to rewrite the problem under this form:

$$\delta(x_i - x_j) = \log\left(\frac{\frac{1}{\sqrt{2\pi|\Sigma_{\mathcal{N}}|}}\exp(-1/2(x_i - x_j)^T\Sigma_{\mathcal{N}}^{-1}(x_i - x_j)}{\frac{1}{\sqrt{2\pi|\Sigma_{\mathcal{P}}|}}\exp(-1/2(x_i - x_j)^T\Sigma_{\mathcal{P}}^{-1}(x_i - x_j)}\right). \tag{3.38}$$

The symmetry of the pairwise difference implies that both distributions have zero mean for both distributions, with covariances Σ of the following form:

$$\Sigma_{\mathcal{P}} = \sum_{(i,j)\in\mathcal{P}} (x_i - x_j)(x_i - x_j)^T, \tag{3.39}$$

and symmetrically for the dissimilar samples with a sum on $\mathcal{N}$. We therefore obtain after reduction and removing the constants:

$$\delta(x_i, x_j) = (x_i, x_j)^T(\Sigma_{\mathcal{P}}^{-1} - \Sigma_{\mathcal{N}}^{-1})(x_i, x_j). \tag{3.40}$$

By seeing δ as our distance and by identification and projection onto the positive semi-definite cone $\mathbb{S}_+^n$, we get $M = \text{Proj}_{\mathbb{S}_+^n}(\Sigma_{\mathcal{P}}^{-1} - \Sigma_{\mathcal{N}}^{-1})$. This projection can be done by eigen-decomposing the matrix to project and, for all its eigen values λ_i, by taking $\max(0, \lambda_i)$ [92].

KISSME was intended to learn metric from large datasets: it does not perform optimization procedures and is therefore very time efficient. However, as such, it does not perform any kind of subspace or representation learning and relies on other approaches to do so such as PCA. It is moreover very sensitive to this subspace dimension [233]. To improve the performances, Faraki et al. [69] proposed to learn the subspace along with the metric (Joint Dimensionality Reduction-KISSME). We first define a mapping $h : \mathbb{R}^n \to \mathbb{R}^m$, to be applied to all samples:

$$h(x) = W_p^T x, \tag{3.41}$$

with $W_p \in \mathbb{R}^{n \times m}$. This makes Equation 3.40 becomes:

$$\delta(x_i, x_j) = (x_i, x_j)^T (W_p(\Sigma_{\mathcal{P}}^{-1} - \Sigma_{\mathcal{N}}^{-1}) W_p^T)(x_i, x_j). \tag{3.42}$$

The mapping matrix W_p can be embedded inside a neural network as a last linear layer and the network be trained with the KISSME loss:

$$\mathcal{L}_{\text{KISSME}}(\mathcal{B}) = \sum_{(i,j)\in\mathcal{B}} l_{i,j} \log(\delta(h(x_i), h(x_j))), \tag{3.43}$$

with $l_{i,j} = 1$ if $(i, j) \in \mathcal{P}$, $l_{i,j} = -1$ otherwise, and:

$$\begin{aligned} \log(\delta(h(x_i), h(x_j))) = &\frac{1}{2}(\text{logdet}(W_p^T \Sigma_{\mathcal{P}} W_p) - \text{logdet}(W_p^T \Sigma_{\mathcal{N}} W_p) \\ &+ (x_i - x_j)^T W_p M W_p^T (x_i - x_j)). \end{aligned} \tag{3.44}$$

Logdet is a perfectly smooth operation and it is therefore possible to train a model with this loss using backpropagation. The learning phase is decomposed into two steps to be repeated alternatively until convergence: learn W_p and the encoder while keeping M fixed. Then update M after some epochs with the following closed-form integrating the projection:

$$M = \text{Proj}_{\mathbb{S}_+^m}((W_p^T \Sigma_S W_p)^{-1} - (W_p^T \Sigma_D W_p)^{-1}). \tag{3.45}$$

3.2.4 Synthesis and Discussion

We presented in this section an overview of metric learning and we then focused on sequence metric learning and deep metric learning losses. Indeed, the problem formalization of routine retrieval (see Section 3.1.2) makes use of a learned distance between two sequences. In this category, DTW can be considered one of the best metric for sequence classification [63] but is difficult to integrate inside learning algorithms. Moreover, it is very sensible to noise [67] which is inherent to sensor data. Finally, DTW works locally while routines can be shifted or not always realized in the exact same order in a sufficiently large time frame. The optimization approaches we also discussed make use of alignment strategy which either suppose the knowledge of groundtruth alignment [77] or are very long to compute [191]. We therefore think that feature-based approaches are better suited for this task notably if the appropriate representation of the data is automatically extracted accordingly to the problem, by a RNN for example. On that matter, we have seen that

metric learning losses were quite suited to learn good representations for images with CNN. The Seq2Seq model used by Abid et al. [2] to learn robust representation is attractive. However, like in several other publications ([44] in Section 3.2.2.3, [191] in Section 3.2.2.2, etc.), the two steps learning process seems sub-optimal and it would be interesting to learn the whole thing end-to-end such as in [42]. This last approach however requires to compute several alignments between the pairs of sequences which can become computationally very expensive, moreover with complex sampling strategies. Finally, deep learning approaches also present the upside of being compatible with the three forms of metrics we highlighted in this section. It allows to combine one architecture with different losses while avoiding to compute costly alignments.

In the following section, we propose a novel Siamese Sequence-to-Sequence (SS2S) neural network architecture to learn to model routines without label supervision. The model effectively combines automatic feature extraction and a similarity metric by jointly learning a robust projection of time series in a metric space. This approach is able to deal with long sequences by using LSTM networks and does not necessitate to choose a model to fit or features to extract. It can be combined with the various metric losses presented earlier.

3.3 Routine Retrieval with Siamese Sequence-to-Sequence Model

The routine retrieval problem formulated as in Section 3.1.2 prompts us to learn a metric on sequential data with a margin loss. We therefore propose in this section a neural network architecture, Siamese Sequence-to-Sequence (SS2S), to jointly learn a robust representation of the sequences and metric between those sequences. We then employ information-theoretic clustering scores to assess the routine retrieval capacities of the model. The contributions of this chapter have been first published in [52].

3.3.1 Architecture Overview and Training

The SS2S architecture (see Figure 3.3) is composed of two components which we deal into details in the next sections: a Seq2Seq model and a metric learning model combined into one architecture that can be learned end-to-end.

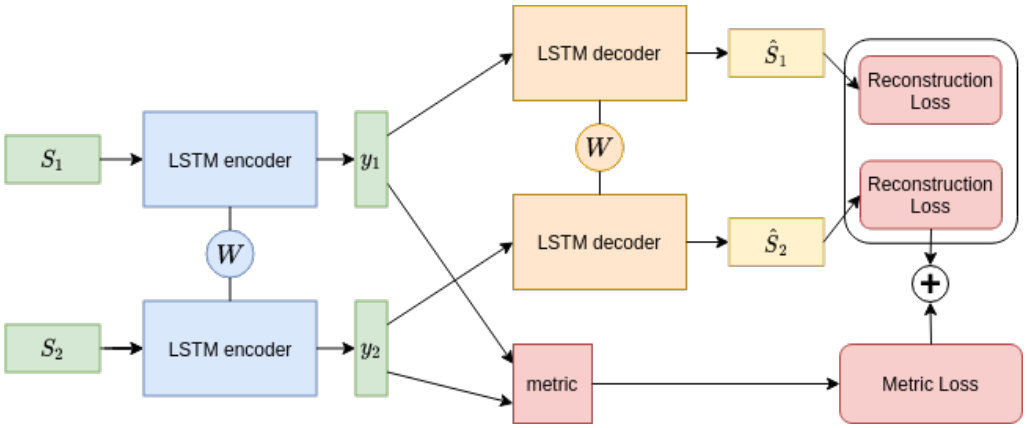


Figure 3.3 – Proposed SS2S architecture.

Two training processes can be considered to learn the parameters of this architecture. Train the Seq2Seq model parameters and then "freeze" the network to learn the metric if it is parametric (i.e. Mahalanobis metric). Or, add the metric loss to the reconstruction loss and learn jointly both tasks. This form of training can be assimilated to some form of multitask learning (see 2.1.2.1) where y_1 and y_2 constitute intermediate representations shared by two tasks: learning the metric and learning to reconstruct the input sequence. In this case, several difficulties could appear. Both losses must have similar magnitudes to have a similar influence on the training process and thus the use of a balancing parameter $\lambda < 1$ is necessary in the loss equation:

$$\mathcal{L}_{\text{SS2S}}(\mathcal{B}) = \lambda \mathcal{L}_{\text{reconstruction}}(\mathcal{B}) + (1-\lambda)\mathcal{L}_{\text{metric}}(\mathcal{B}). \tag{3.46}$$

The interaction between the two must also be considered which we also deepen in the next section. Despite the possible issues, we hope that learning both tasks jointly should lead to the learning of more appropriate representations and thus to better results. We formulate the following hypothesis which we will test in the experimental section:

Hypothesis 3.1. *Jointly learning a metric and a representation with a sequence to sequence model gives better results than learning both separately.*

3.3.2 Metric Learning Specifications

Our architecture is a siamese network [30], that is to say it is constituted of two subnetworks sharing the same parameters W (see Figure 3.3). It takes pairs of similar or dissimilar sequences as input constituted with

an equivalence constraint which we explicit later. Since three metric forms can generally be considered (Euclidean and cosine which are non parametric and Mahalanobis which implies learning the associated matrix), one different metric loss is proposed to learn each metric form. The first is the contrastive loss [86] (see Equation (3.4)) to learn an euclidean distance. The second is a cosine loss to learn a cosine distance (see Equation 3.22). To learn a Mahalanobis metric, we propose to use the KISSME approach (see Section 3.2.3.3) integrated into a neural network. One argument to employ this loss is that its two steps learning process (called "pairwise+KISSME" by Faraki et al. [69]) seems to fit well with our architecture, especially when the learning is disjoint since the author observed that separating the learning of the projection and the learning of the metric leads to better performance and stability. This setting also challenges the previously made hypothesis. The projection matrix W_p is integrated into the network as a linear layer (just after the recurrent encoding layers in SS2S). We also propose a modified version of the KISSME loss from [69] which we found was easier to train based on the contrastive loss (Equation 3.4):

$$\mathcal{L}_{\text{contrastive KISSME}}(\mathcal{B}) = \frac{1}{|\mathcal{B}|}\Big(\sum_{(i,j)\in\mathcal{P}} (y_1 - y_2)M(y_1 - y_2)^T + \sum_{(i,j)\in\mathcal{N}} [m - (y_1 - y_2)M(y_1 - y_2)^T]_+ \Big). \quad (3.47)$$

This loss also integrates a margin which allows to fit the problem formulation. The matrix M is updated at regular intervals, every few epochs following Eqation 3.45. In the experiment section, we use pairs of inputs to learn the network however this architecture is perfectly adaptable to triplet inputs or more complex batch sampling strategies.

3.3.3 Feature Extraction Specifications

The time series data obtained from inertial sensors may be very noisy and certainly vary for the same general activity (e.g. cooking). Robust feature representations of time series should therefore be learned to be used with the metric, justifying the inclusion of a denoising Seq2Seq model (see Section 2.1.4.4). The sequence is given as input to the first LSTM network (the encoder) to produce an output sequence, the last output vector is considered as the learned representation. This representation is then fed as an input sequence to the second LSTM (the decoder) which tries

to reconstruct the input sequence. Typically, an autoencoder is trained to reconstruct the original sequence with the MSE (reworked here for sequences [197]):

$$\mathcal{L}_{\text{MSE}}(\mathcal{B}) = \frac{1}{|\mathcal{B}|} \sum_{S_i \in \mathcal{B}} \frac{1}{T} \sum_{t=0}^{T-1} (S_i(t) - \hat{S}_i(t))^2, \tag{3.48}$$

where $\hat{S}$ is produced by the decoder. Symmetrically, since both tasks could have eventually divergent or not completely compatible objectives, we propose a new reconstruction loss based on cosine similarity, the Cosine Reconstruction Loss (CRL):

$$\mathcal{L}_{\text{CRL}}(\mathcal{B}) = \frac{1}{|\mathcal{B}|} \sum_{S_i \in \mathcal{B}} \left(T - \sum_{t=0}^{T-1} \cos(S_i(t), \hat{S}_i(t)) \right). \tag{3.49}$$

CRL is close to 0 if the cosine similarity between each pair of vectors is close to one which happens when the vectors are collinear. We propose this loss with the *a priori* that it should better interact with the learning of a cosine metric than MSE due to the two having similar forms. This leads to our second hypothesis:

Hypothesis 3.2. *Learning a cosine distance along a representation with CRL gives better results than with MSE.*

3.3.4 Routine Retrieval

To train the previous architecture, we need to define an equivalence constraint to form similar and dissimilar pairs. Based on the problem formulation, routines should appear at relatively similar moments. To stay in a weakly supervised setting, we chose to treat as similar samples recorded during the same hour across different days. All other combinations are considered dissimilar. This labeling is fully automatic and based on metadata, it does not use any semantic activity labels. However, while practice for training this approach reveals complex to evaluate: how to retrieve routines and not just timestamps we already have, even for test data? With minimal supervision, we are required to exploit the learned metric inside a clustering algorithm. We propose to use spectral clustering [147] which can be performed on the affinity matrix of the test set and is therefore independent of the form of the distance. It requires however to transform distance values into kernel values. For the cosine distance,

we simply add one. For the Euclidean and Mahalanobis distances, we use the following equation of a Gaussian kernel:

$$K_{\text{Gaussian}}(\hat{y}_1, \hat{y}_2) = \exp\left(-\frac{d(\hat{y}_1, \hat{y}_2)^2}{n}\right), \tag{3.50}$$

where n is the dimension of $\hat{y}_1$ and $\hat{y}_2$. Obviously, there should be much less *main* routines than hours in the day: for example the sleeping routine spans generally over 6 to 9 hours [163]. It prompts us to select a low number of clusters to be formed by the clustering algorithm (we used 5 in the experimental section). These clusters, if they correspond to routines, should present some kind of *temporal coherence* which can be assessed by using the hour labeling as "ground truth". At a first level, the clusters should gather samples recorded at the same times, which is exactly what is evaluated by a metric called *Completeness* [169][4]. At a second level, their should exist some kind of information dependency between the time slots and the clustering assignment: the observation of the first should inform us on the second and conversely. That is to say, routines (clusters) should not be independent from the time division (the labeling), which is coherent with the problem formulation. This can be measured thanks to information theory, it corresponds to a metric called NMI (see Figure 3.4) widely use for clustering scoring. It is computed as follows for two label assignments U and V:

$$\text{MI}(U, V) = \sum_{i=1}^{|U|} \sum_{j=1}^{|V|} P(i,j) \log\left(\frac{P(i,j)}{P(i)P(j)}\right) \tag{3.51}$$

$$\text{H}(U) = -\sum_{i=1}^{|U|} P(i) \log(P(i)) \tag{3.52}$$

$$\text{NMI}(U, V) = \frac{\text{MI}(U, V)}{(H(U) + H(V))/2}. \tag{3.53}$$

However, the NMI has a flaw despite the normalization: its value is not independent from the number of clusters. To "repair" it, a so-called adjustment against chance is necessary [215]. It is realized by introducing the expected value of the mutual information: $E(\text{MI}(U, V))$. The new

[4]This is one of the two components of the classical metric called V-measure. However, the other, homogeneity, measures the exact inverse of what we seek to assess: that each cluster contains samples of only one hour slot. This justifies why we do not report it. V-measure is reported as it is the same as Normalized Mutual Information (NMI).

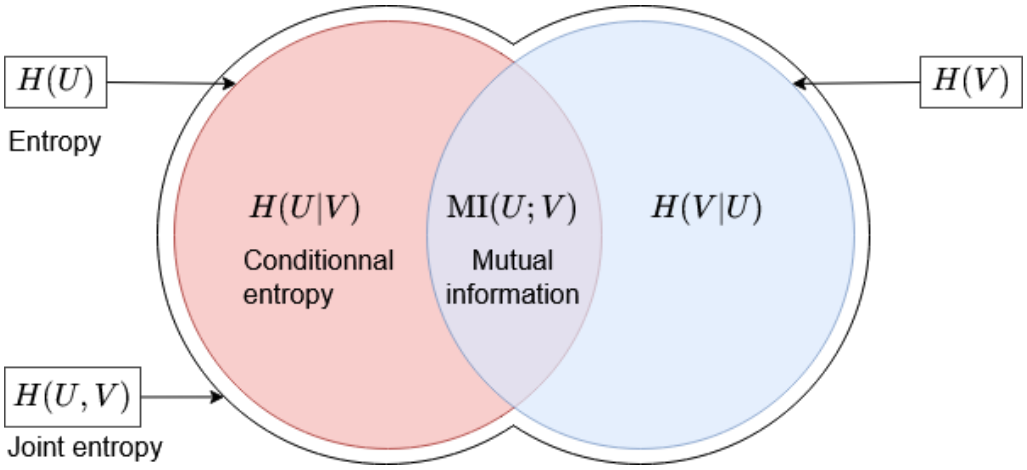


Figure 3.4 – Relationships between the main information theoretic metrics. U and V are two label assignments (random variables): mutual information at the center corresponds to the information shared by the two variables.

score, Adjusted Mutual Information (AMI), is computed as follows:

$$\mathrm{AMI}(U,V) = \frac{\mathrm{MI}(U,V) - E(\mathrm{MI}(U,V))}{(H(U)+H(V))/2 - E(\mathrm{MI}(U,V))}. \tag{3.54}$$

We report both NMI and AMI in the experimental section. Finally, the *Silhouette* score is used to describe the overall quality of the clustering, if the clusters are dense and well-separated then the value is close to 1. To conclude the exposition of our architecture, we propose a final third hypothesis:

Hypothesis 3.3. *Completeness, NMI, AMI and silhouette scores allow to asses the capacity of the propose architecture to model and retrieve routines.*

3.4 Experiments

3.4.1 Dataset Presentation and Experimental Setup

3.4.1.1 Long-Term Movement Monitoring Dataset

Long-term unscripted data from wearable sensors are difficult to gather. The dataset presented here, Long-Term Movement Monitoring (LTMM)[5],

[5] https://www.physionet.org/physiobank/database/ltmm

has been recorded by Weiss et al. [223] to investigate the correlation between gait quality and fall risks. This dataset contains recordings of 71 elderly people which have worn an accelerometer and a gyroscope during three days with no instructions. This dataset contains no labels. Figure 3.5a presents two days of data coming from one axis of the accelerometer: similar profiles can be observed at similar moment. Figure 3.5b presents the autocorrelation of the accelerometer signal: the maximum of 0.4 is reached for a phase of 24h. These figures show the interest of this dataset as the data show periodic nature while presenting major visual differences. That said, the definition of periodicity that our algorithm is made to achieve is stronger as it is based on a metric between automatically extracted feature vectors, not just correlations of signal measurements.
To constitute our dataset, we selected in the original dataset a user who did not remove the sensor during the 3 days to avoid missing values. We set up a data augmentation process to artificially increase the quantity of data while preserving its characteristic structure. The dataset is sampled at 100 Hz and thus, to multiply the number of days by 10, each vector measurement at the same index modulo 10 will be affected to a new day (the order is respected). This new dataset has a sampling rate of 10 Hz which means that one hour of data is a sequence of size 36000. We consider only non overlapping sequences. Thus, to make the computation more tractable, each sequence of one hour is resampled to a size of 100.

3.4.1.2 Model Parameters and Training Details.

We describe here the hyperparameters used to train the models. After preliminary studies, the autoencoders are constituted of one layer of 100 LSTM neurons for the encoder and the decoder. For the KISSME version, the encodings are then projected into a 50-dimensional space, and the distance matrix, which thus has also dimension 50, was updated with the closed-form every 30 epochs. The balancing parameter λ was fixed to 0.5. These parameters were determined after preliminary tests where deeper architectures and higher dimensional spaces were tested. Models are trained with 20 similar pairs for each time slot and the same total number of dissimilar pairs for a total of 960 training pairs coming from 12 different days of data. The training was stopped based on the loss computed on the validation set which contains three days of data (i.e. 72 sequences). The testing set is composed of 15 days or 360 sequences. The features in the training set were standardized to have a mean of 0 and a

standard deviation of 1, the same parameters were applied on the testing set. A learning rate of 0.001 was used and divided by 10 if the loss did not decrease anymore during 10 epochs. A batch size of 50, a margin of 1 for the contrastive loss and of 0.5 for the cosine loss were chosen. We also observed that changing to zero 30% of the values of the training sequences slightly improved the results as suggested in [213].

3.4.2 Experimental Results and Discussion

Since the model is not trained with semantic labels but timestamp metadata and to keep minimal supervision, the evaluation scores rely on clustering (see Section 3.3.4). We report average values on 20 tests for the 4 clustering evaluation metrics mentioned earlier: completeness, silhouette, NMI and AMI. A spectral clustering into 5 clusters is performed with the goal not to find the precise number of clusters maximizing the scores but to choose a number which could make appear coherent and interpretable routines of the day, namely sleep moments, meals and other daily activities performed every day. We first evaluate CRL alone before presenting the complete test results.

3.4.2.1 Evaluation of Cosine Reconstruction Loss.

The performance of the CRL on LTMM is first evaluated by jointly training models for Euclidean or cosine distances with CRL or MSE.

The results are reported in Table 3.1. An asterisk means that the average results are significantly higher according to a Welch's test with a threshold of 5%. The results demonstrate a significant improvement of the proposed CRL over MSE when trained with the cosine similarity for Completeness, NMI and AMI. For the Silhouette score, better results are obtained with the MSE. However, the standard deviations are large, and this improvement is thus not significant. With the Euclidean distance, the same improvement is not realized with a slight advantage of MSE over CRL. These results confirm our hypothesis 3.2 that it is more efficient to jointly learn a cosine distance with CRL. They also suggest a positive interaction between the two as the same effect could not be observed with the Euclidean distance. This result is also interesting in the context of multitask learning: the learned common representation seems more adequate for both tasks. We then use CRL in the remaining of the chapter.

metric \ reconstruction loss	CRL	MSE
Cosine	**0.714*** ± 0.048	0.666 ± 0.066
Euclidean	0.609 ± 0.042	0.635 ± 0.064

(a) Completeness

metric \ reconstruction loss	CRL	MSE
Cosine	0.618 ± 0.105	**0.667** ± 0.144
Euclidean	0.402 ± 0.05	0.408 ± 0.042

(b) Silhouette

metric \ reconstruction loss	CRL	MSE
Cosine	**0.449*** ± 0.032	0.397 ± 0.040
Euclidean	0.419 ± 0.033	0.434 ± 0.047

(c) NMI

metric \ reconstruction loss	CRL	MSE
Cosine	**0.253*** ± 0.03	0.205 ± 0.033
Euclidean	0.255 ± 0.027	0.264 ± 0.038

(d) AMI

Table 3.1 – Comparison of CRL and MSE on LTMM dataset.

3.4.2.2 Evaluation of the SS2S Architecture

Next, we investigated the benefit of the SS2S architecture over DTW and Siamese Long-Short Term Memory (SLSTM) [143] as well as the interest of jointly learning the encoder-decoder and the metric on the LTMM dataset. The results are presented in Table 3.2. To test the DTW, the better radius was selected on the validation set and the spectral clustering was performed using the DTW distance under a kernel form (see Equation 3.50). Although Completeness, NMI and AMI are higher than every SS2S architectures except one, we observe a very low silhouette score. Concerning the encoding architecture, SS2S gives overall better results than SLSTM and the best results are achieved by using the disjoint version of KISSME with a completeness of 0.983 and an NMI of 0.619. These results are not surprising as KISSME uses a parametric distance which can therefore be more adapted to the data. For the silhouette score, cosine distances performed best, that is to say they learned more com-

pact and well-defined clusters. We also note that disjoint versions of the architectures performed better than the joint versions, thus invalidating our hypothesis 3.1.

Metric	Model	Joint	Completeness	Silhouette	NMI	AMI
DTW [172]	x	x	0.804	0.213[6]	0.528	0.32
Euclidean	SLSTM	x	0.616 ± 0.032	0.427 ± 0.053	0.414 ± 0.022	0.246 ± 0.019
Cosine	SLSTM	x	0.617 ± 0.06	0.572 ± 0.143	0.372 ± 0.052	0.192 ± 0.046
Euclidean	SS2S	no	0.674 ± 0.04	0.528 ± 0.07	0.458 ± 0.03	0.28 ± 0.027
Euclidean	SS2S	yes	0.635 ± 0.064	0.408 ± 0.042	0.434 ± 0.047	0.264 ± 0.038
Cosine	SS2S	no	0.71 ± 0.05	**0.756*** ± 0.089	0.467 ± 0.028	0.275 ± 0.024
Cosine	SS2S	yes	0.714 ± 0.048	0.618 ± 0.105	0.449 ± 0.032	0.253 ± 0.03
KISSME	SS2S	no	**0.983*** ± 0.016	0.439 ± 0.077	**0.619*** ± 0.035	**0.363*** ± 0.046
KISSME	SS2S	yes	0.667 ± 0.021	0.316 ± 0.039	0.446 ± 0.012	0.266 ± 0.012

Table 3.2 – Evaluations on LTMM dataset of the SS2S architecture (x means non applicable).

3.4.2.3 Reconstruction Error on the Validation Set

We want investigate the reasons why the joint training performed lesser than the disjoint version and we make the hypothesis that this could due to the autoencoder not being trained properly. Table 3.3 reports the average best reconstruction errors achieved on the validation set. The lowest errors are systematically achieved when the encoder is learned alone before the metric therefore supporting the hypothesis that learning the metric prevents the autoencoder from being trained at its full potential. It may explain why the joint learning does not perform best. For the CRL, results are closer than for MSE suggesting why this reconstruction loss is easier to learn jointly. Regarding KISSME, this was also suggested by the original two step learning process proposed by the authors

3.4.2.4 Average Distance to Nearest Neighbors

To better investigate the various proposed distances, we display such as in [2] the average distance to a variable number of nearest neighbors for the cosine and KISSME metrics and both learning variants on Figure 3.6. For the cosine similarity, the joint variant display a curve growing very slowly ; for the disjoint one, the average distance increases faster and

[6]The silhouette value for DTW presented in [52] was wrong and resulted from an implementation error. We display here the correct value.

Metric	Average reconstruction error
Euclidean	0.707 ± 0.112
KISSME	0.736 ± 0.099
Disjoint	**0.55*** ± 0.083

(a) MSE

Metric	Average reconstruction error
Cosine	0.339 ± 0.036
Disjoint	**0.298*** ± 0.03

(b) CRL

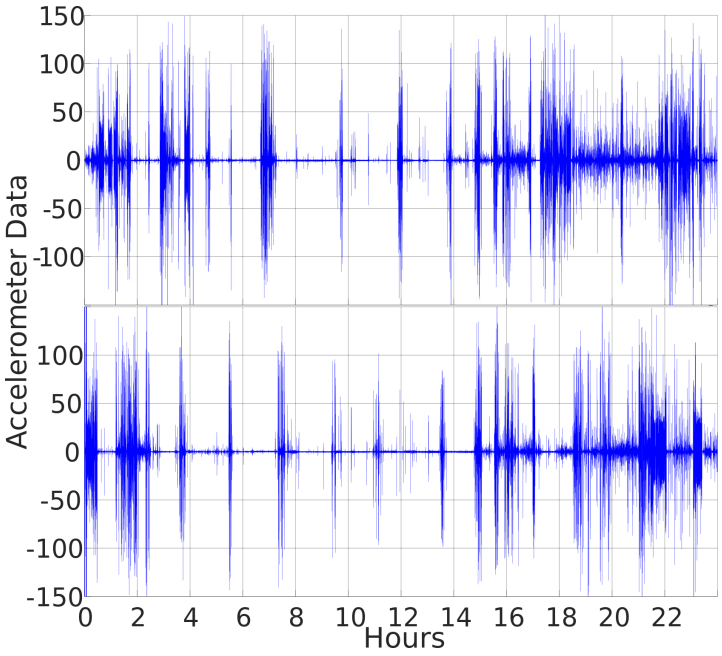
Table 3.3 – Average reconstruction errors on the validation set of LTMM.

faster with the number of neighbors particularly beyond 150. For the KISSME metric, both curves follow a similar pattern by growing quickly at the beginning before reaching a plateau. These curves are actually in line with the silhouette scores presented in Table 3.2: lower scores for the KISSME loss can be explained by a rapidly growing distance producing sparse clusters. Conversely, we observe that both cosine approaches produce distances growing much slower at the very beginning, thus the more compact clusters.

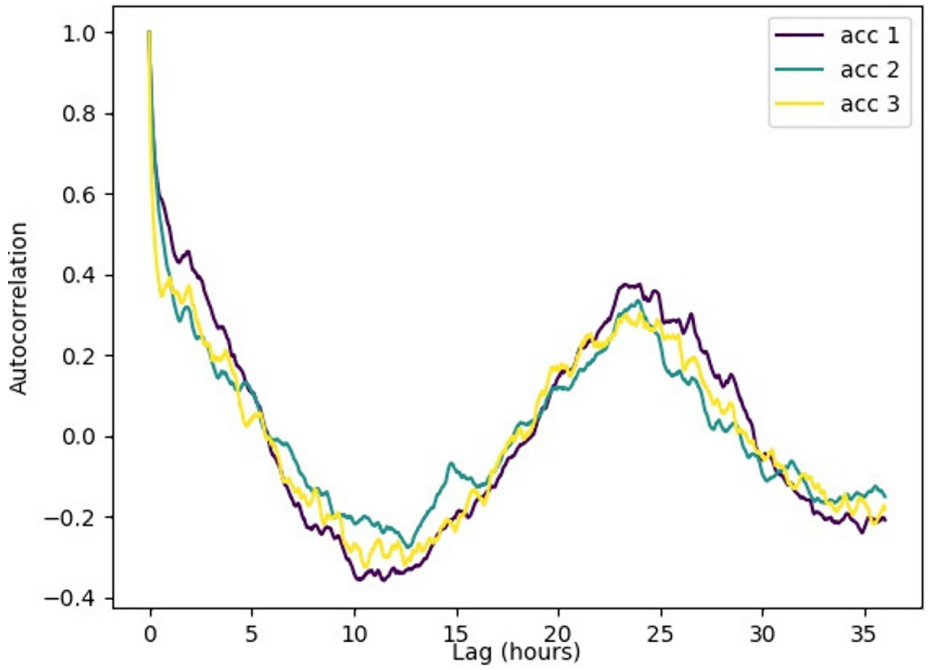
3.4.2.5 Clustering Visualization

Finally, Figure 3.7 shows clustering representations for two approaches DTW and disjoint KISSME. The clustering assignments reflect the sequences of one hour that were found similar across the days on the testing set. If these sequences are at the same hour or cover the same time slots, we can argue it is a recurrent activity (or succession of activities) and therefore a routine. The disjoint KISSME version exhibits more coherent discrimination of routines, which, according to the 4 evaluation metrics reported was predictable. More what seems to be missclassified situations appear for the DTW however, without labeling we cannot know for sure. High regularities can be observed, and it is actually possible to make interpretations: yellow probably corresponds to sleeping moments or nights and purple to diverse activities during the day. This visual interpretation seems to go in the same direction as Hypothesis 3.3 although further tests on the long term are of course necessary. Other clusters seem to correspond to activities at the evening or during meal time. Consequently, in this example, the SS2S architecture was able to learn a metric which cluster and produce a modeling of the daily rou-

tines of the person without labels. In this example, the clusters are coarse, the granularity of this analysis may be improved simply by working with sequences of half an hour or even shorter and produce more clusters.



(a) Two days of accelerometer data from Long-Term Movement Monitoring (LTMM).



(b) Input signal autocorrelation for accelerometer data.

Figure 3.5 – LTMM dataset used to evaluate routine retrieval procedure.

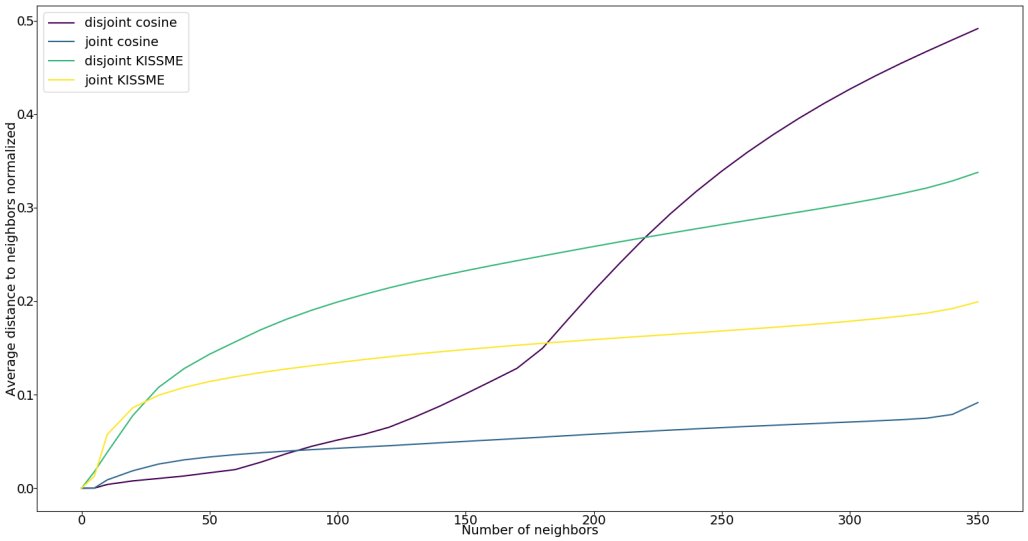
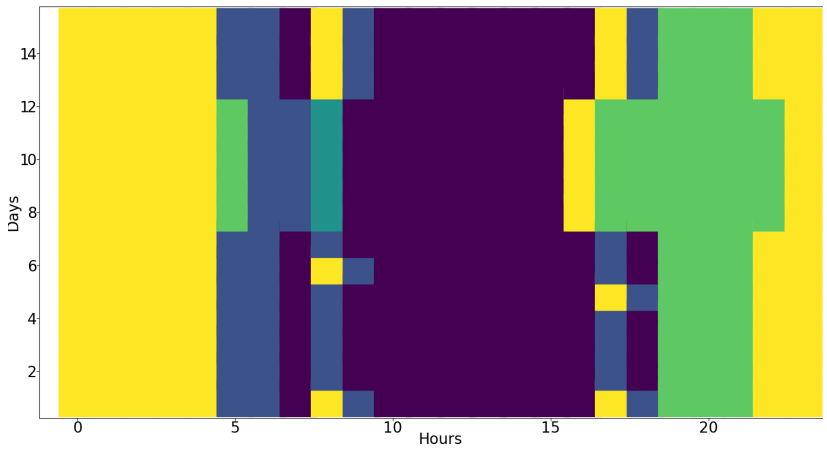
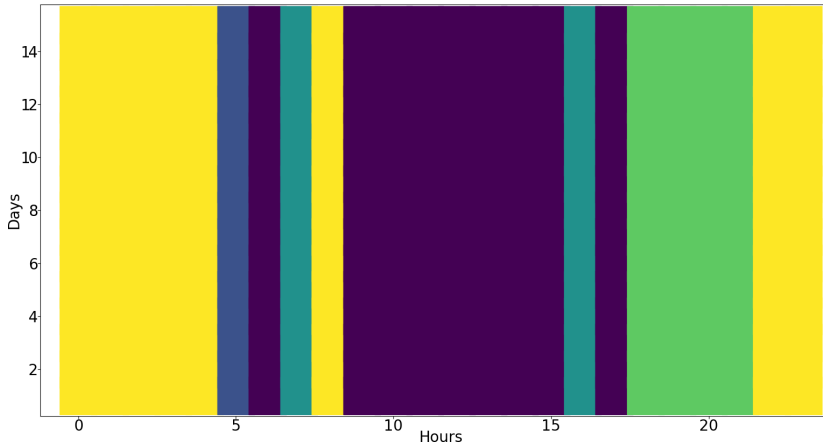


Figure 3.6 – Average normalized distance to nearest neighbors for the cosine and KISSME metrics for the joint and disjoint learning variants.



(a) DTW [172].



(b) SS2S and KISSME, disjoint learning.

Figure 3.7 – Examples of clustering assignments obtained on LTMM.

3.5 Conclusions and Perspectives

In this chapter, we tackled the issue of routine retrieval as an alternative to classical supervised activity recognition. We first proposed a mathematical formalization of the concept of routine based on almost-periodic functions. This formalization lead us to explore the literature of sequence metric learning before proposing a Siamese Sequence-to-Sequence model to jointly learn a good robust representation of the sequences and a metric. The relevance of the joint learning was to be evaluated by experiments. Our proposed architecture relies on no activity labels and is learned only from time slots. The objective of routine retrieval was performed by using the learned metric to cluster new sequences and by computing information theoretic scores. Our SS2S architecture with KISSME and disjoint learning process achieved great results with 0.983 of completeness and 0.619 of NMI. A visual evaluation allows to interpret the what seems to be recurrent behaviors discovered by the architecture (Hypothesis 3.3). However, the results showed that combining metric learning and sequence-to-sequence learning did not allow to achieve better performances (Hypothesis 3.1). We therefore proposed an explanation based on the reconstruction error achieved on the validation set at the end of training which was higher in the case of joint learning, likely indicating a lower quality of representation. A new reconstruction loss was also proposed to be learned jointly with a cosine metric and it showed better results than MSE in this case which constitute an interesting result in the context of multitask learning (Hypothesis 3.2).

There is however, plenty of room for improvement in this routine retrieval pipeline, especially concerning the learning model. One of them is the intermediate representation: to process sequences with this architecture, we need to pass through a representation of the sequences. This was further confirmed by the fact that Seq2Seq models produce better distances than simple RNN (even if the last output used to compute the metric can be considered a representation). However we have seen that this architecture is better train in two steps rather than end-to-end. Moreover, approaches relying on alignment work without producing an intermediate representation, on the sequence directly. We therefore seek to propose an intermediate approach, a neural network specifically conceived to output distance between sequences. We propose to tackle this issue with dynamical system synchronization theory in the next chapter.

CHAPTER 4

SEQUENCE METRIC LEARNING AS SYNCHRONIZATION OF RECURRENT NEURAL NETWORKS

As we have seen in the previous chapter, sequence metric learning is most the time tackled with sequence alignment approaches or representation learning. Among those approaches, less work has been devoted to conceive a specific sequence metric learning architecture with recurrent neural networks despite the simplicity of the siamese architecture [30]. One way to tackle this problematic can be to take a more theoretical look at it. RNN exhibit a temporal dynamic behavior allowing them to deal with temporal correlations. This feature also allows to study RNN with dynamical system theory. This has been notably done in the literature to get a better understanding of the issues encountered when trying to train a RNN [21, 66, 94] (between others).

In this fourth chapter, we propose to use dynamical system theory in order to design a neural network architecture specifically adapted to sequence metric learning. We draw the analogy between synchronized trajectories produced by dynamical systems and the distance between similar sequences processed by a siamese recurrent neural network. Indeed, a siamese recurrent network comprises two identical sub-networks, two identical dynamical systems which can theoretically always achieve complete synchronization if a coupling is introduced between them. We therefore propose a new neural network model that implements this

coupling with a new gate integrated into the original siamese GRU architecture. This model is thus able to simultaneously learn a similarity metric and the synchronization of unaligned multi-variate sequences in a weakly supervised way. Our experiments show that introducing such coupling improves the performance of the siamese GRU architecture on an activity recognition dataset and on a transportation recognition dataset. We also used this architecture to perform routine retrieval with more mixed success.

This chapter is organized as follows: Section 4.1 presents generalities and basics of dynamical system theory, Section 4.2 outlines the state-of-the-art approaches in sequence metric learning, Section 4.3 describes our framework and our new siamese architecture, Section 4.4 shows our experimental results to assess the performances of our approach compared on an activity recognition dataset, on a transportation recognition dataset and on the routine retrieval task. Finally, Section 4.5 presents our conclusions and perspectives.

4.1 Basics of dynamical system theory and synchronization

In this section, we introduce general notions and definitions from dynamical system theory and synchronization. For further details on dynamical system theory, the reader can refer to the book by Strogatz [188].

4.1.1 Generalities

Dynamical system theory describes physical systems which evolve from a starting point of the space (the phase space) depending solely on the time. This definition can be more formally written this way:

Definition 4.1. A dynamical system is a triplet (T, M, ϕ) where $\phi : M \times T \to M$ is called the evolution rule, M is the phase space, a vectorial space of dimension n and $T = [t_0, +\infty[$ is a set of times.

To start studying dynamical systems, one first major difference can be made, some are discrete others are continuous. They are respectively defined by the two following evolution rules, called a map in the discrete case and a flow in the continuous case:

$$\text{map} : x(t) = \phi^t(x(0)), t \in \mathbb{N} \tag{4.1}$$

$$\text{flow} : \frac{dx(t)}{dt} = \phi(x(t), t), t \in \mathbb{R}_+. \tag{4.2}$$

The point $x(0)$ is the initial condition at $t = 0$ from which the system will evolve. The set of points from M reached during the evolution is called the trajectory. A dynamical system may have one or several attractors, a point in the phase space toward each trajectory starting in a certain region called the basin of attraction, converges and will stay close even if it is slightly perturbed. When changing the parameters of the evolution rule smoothly, it is expected that the topology of the phase space (typically, the number and emplacements of the attractors) remains qualitatively the same. The phenomenon called *bifurcation* occurs at certain values of the parameters and describes sudden changes in the topology of the phase space, for example, the apparition of a new attractor.

4.1.2 Chaos and Lyapunov exponents

Apart from discrete and continuous, another major category regroups so-called chaotic systems. Chaotic systems produce resulting trajectories which exponentially diverge from infinitesimally close initial conditions. This practically means that the behavior of such systems can become rapidly unpredictable solely due to the approximations made in the measures of the current state (or due to numerical approximations). The Lyapunov exponents [134] quantify the sensibility of a system to the initial conditions and therefore the divergence rate of trajectories starting from infinitesimally close starting points. Formally, Lyapunov exponents of a map are defined with the following equation:

$$\lambda(x(0)) = \lim_{t \to +\infty} \frac{1}{t} \sum_{i=0}^{t-1} \ln(|\phi'(x(i))|). \tag{4.3}$$

This definition depends on the initial conditions but in practice, a Lyapunov exponent is unique to a given attractor. If the system has at least one positive Lyapunov exponent, the predictability of its behavior becomes impossible beyond a certain time limit (horizon) and it is thus qualified as chaotic [188]. For discrete systems, analytically obtaining the exact values of the Lyapunov exponents of an arbitrary dynamical system can be hard in practice but they can be estimated by various numerical methods and also with machine learning [158]. Lyapunov exponents can be used to measure the stability of a system with high Lyapunov exponents indicating higher sensibility to small perturbations.

This has notably found applications in the assessment of gait stability and falling risks [33, 64].

4.1.3 Coupling

Dynamical systems can be coupled, that it to say constituted of two subsystems which exchange information by the mean of a coupling:

$$Z = \begin{cases} X : \frac{\mathrm{d}x(t)}{\mathrm{d}t} = \phi_1(x(t);t) + C(y(t) - x(t))^T \\ Y : \frac{\mathrm{d}y(t)}{\mathrm{d}t} = \phi_2(y(t);t) + C(x(t) - y(t))^T. \end{cases} \tag{4.4}$$

In this system, $x(t)$ and $y(t)$ are in $\mathbb{R}^n$ and X and Y are bidirectionally coupled by the mean of C, the coupling matrix in $\mathbb{R}^{n\times n}$. This type of coupling is called *diffusive* because it will dissipate the dynamics of each sub-system [24]. The coupling can also be unidirectional: such systems are qualified as drive-response system since one subsystem *drives* the behavior of the other following the strength of the coupling.

4.1.4 Synchronization of dynamical systems

The concept of synchronization is generally well-understood for time-periodic dynamical systems: this phenomenon is called *phase synchronization*. However, it is less-known that it is a special case of a more general view on synchronization where it can also occur for chaotic dynamical systems [159]. To formalize the concept of synchronization for chaotic systems, Brown et al. [31] proposed the following general definition of it:

Definition 4.2. Let Z be a dynamical system composed of two subsystems X and Y such that:

$$Z = \begin{cases} X : \frac{\mathrm{d}x(t)}{\mathrm{d}t} = \phi_1(x(t), y(t); t) \\ Y : \frac{\mathrm{d}y(t)}{\mathrm{d}t} = \phi_2(y(t), x(t); t), \end{cases} \tag{4.5}$$

where $x(t) \in \mathbb{R}^{d_1}$ and $y(t) \in \mathbb{R}^{d_2}, \forall t \in \mathbb{R}_+$ and $d_1, d_2 \in \mathbb{N}^*$. Let $\varphi(z_0)$ be a trajectory of Z with initial conditions $z_0 = [x(0), y(0)] \in \mathbb{R}^{d_1} \times \mathbb{R}^{d_2}$. Finally let $g : \mathbb{R}^{d_1}$ (resp. $\mathbb{R}^{d_2}$) $\times \mathbb{R}_+ \to \mathbb{R}^k$ with $k \in \mathbb{N}$, be a measurable property of the subsystems. They are synchronized on the trajectory $\varphi(z_0)$ with respect to the property g if there is a time independent function $h : \mathbb{R}^k \times \mathbb{R}^k \to \mathbb{R}^k$ such that:

$$\|h(g(x), g(y))\| = 0, \tag{4.6}$$

where $\|\cdot\|$ is a norm.

4.1.5 Complete Synchronization

A special case of Definition 4.2 is when ϕ_1 and ϕ_2 are the same function, the dynamical systems share the same parameters, and they are said to be *identical*. Therefore, consider now the following two identical systems:

$$Z = \begin{cases} X : \frac{\mathrm{d}x(t)}{\mathrm{d}t} = \phi(x(t);t) + C(y(t) - x(t))^T \\ Y : \frac{\mathrm{d}y(t)}{\mathrm{d}t} = \phi(y(t);t) + C(x(t) - y(t))^T, \end{cases} \tag{4.7}$$

Fujisaka et al. [70] showed that the system described by Equation 4.7 can achieve complete synchronization if C is a multiple of the Identity matrix and a constant c which verifies the following condition:

$$c > \frac{1}{2}\lambda_L, \tag{4.8}$$

where λ_L is the largest Lyapunov exponent of the system.

From the definition 4.2, it is possible to derive several ways to measure synchronization between two trajectories, namely a synchronization error. Brown et al. [31] report several slightly different formulas for the synchronization error with the following being the most used, according to them, for identical systems:

$$h(g(x), g(y)) = \lim_{t \to +\infty} (g(x(t)) - g(y(t))), \tag{4.9}$$

where h and g are the same as in Definition 4.2.

4.2 Related Work

Before introducing our new architecture, we review some works studying the behavior of RNN with dynamical system theory. We will also discuss a category of sequence metrics we set aside in the previous chapter, model-based distance and their generalization to metric on dynamical systems.

4.2.1 Study of Recurrent Neural Networks with Dynamical System Theory.

To begin with, much similarly to the universal approximation theorem for feedforward networks [99], RNN also have approximation capabili-

ties. Indeed, Funahashi et al. [73] proved that RNN can approximate any finite-time trajectory of a dynamical system.

Training RNN as always been an intense subject of research due to their depth when unrolled (see 2.1.4.2) but also to dynamical phenomenons. When changing the parameters of the system (the network) little by little, for example by learning with gradient descent, the properties of the vector field, notably the positions of the attractor points are expected to move smoothly, unless if a bifurcation happens (see 4.1.1). The implications of this phenomenon on the training of RNN were studied by Doya [66]. He indeed argued that bifurcations can cause the learning equation to become unstable which prevents the gradient to work well. Bengio et al. [21] used dynamical system theory to theoretically tackle the problem of learning long-term dependencies (see 2.1.4.2). They observed that to store one bit of information on the long-term, the trajectory of network viewed as a dynamical system should stay in the same basin of attraction of a given attractor point. However, this is precisely where the conditions for the gradient to vanish are reunited. Pascanu et al. [156], extending Doya's work [66], showed that two types of events particularly cause drastic gradient perturbations: crossing a boundary between two basins of attraction and crossing a bifurcation boundary. Following their analysis, they devised solutions to prevent the gradient from exploding, notably gradient clipping, which is used during the experiments of the present thesis.

Another recent approach by Chang et al. [40] studied the trainability of RNN models and established a connection with discretized Ordinary Differential Equations (ODE) stability (Equation 4.2 is an example of ODE). They identified a criterion (the real part of the eigenvalues of Jacobian matrix of ϕ are approximately 0) to guarantee that the system can preserve long-term dependencies. They remarked that antisymmetric matrices (matrices which have their transpose equal their opposite) only have imaginary eigenvalues. They proposed a new version of the RNN equation which guarantees that the hidden weights are antisymmetric, thus making the RNN respecting the criterion. Laurent et al. [121] studied the dynamics of LSTM Neural Networks and GRU and observed that it is chaotic in the absence of input data. They designed a chaos-free RNN architecture having a more predictable behavior by bringing a simple modification to the gating architecture:

$$\text{Hidden state: } h_t = f_t \circ \tanh(h_{t-1}) + n_t \circ \tanh(W x_t) \tag{4.10}$$

$$\text{Forget: } f_t = \sigma(W_{if} x_t + W_{hf} h_{t-1} + b_f) \tag{4.11}$$

$$\text{Input: } n_t = \sigma(W_{in}x_t + W_{hn}h_{t-1} + b_n). \tag{4.12}$$

Laurent et al. describe the behavior of this architecture as predictable since without input data the network state goes toward zero, therefore, with input data, its state is fully determined by those data. Their approach shows similar performances compared to their chaotic counterparts and Laurent et al. logically conclude that chaos cannot explain the performances of GRU and LSTM.

4.2.2 Model-based distances and metrics on dynamical systems

In this chapter, we aim at proposing a sequence metric learning neural network architecture inspired from dynamical system theory. However, sequence metrics and dynamical systems are also related through a category of distances which was voluntary left aside in the previous chapter: model-based distances. This type of metrics makes the assumption that an underlying statistical or dynamical model has generated the sequence [4, 130]. So the first step consists in selecting a type of model or a mixture of models that could fit the data. Among the most popular, we can mention: HMM, linear dynamical systems [78], AutoRegressive-Moving-Average (ARMA) and its extensions (ARMAX, ARIMA, VARIMA, etc. [98]), etc.. Then for each sequence, it is necessary to find the parameters of the model that generate the sequence. Here again, different estimation methods can be considered such as the Box-Jenkins method [28] for ARMA-like models which uses maximum likelihood estimation. These methods have been shown great interest by the activity recognition community since several aspects of motion can be modeled with dynamical models, for example gait [225] and other body movements [207]. Once it is done, various methods allow to compute distances on the specific parameters to compare two time-series generated by a model. For example, different approaches [61, 107, 137] to compute a metrics for ARMA-like processes use cepstral analysis [25][1] and principal angles between subspaces. Finally, a recent work by, Ishikawa et al. [102] proposed a general metric on nonlinear-dynamical systems based on the transfer operator. Their metric generalizes most of the previous approaches.

[1] the cepstrum is obtained by computing the inverse Fourier Transform of the spectrum of the signal

4.2.3 Synthesis and discussion

The papers mentioned above demonstrate that dynamical system theory is a fertile soil to study and conceive new RNN models. It gives, for examples, insights on how RNN learn and what makes them difficult to train. Therefore, reformulating problems containing RNN within dynamical theory coordinates may provide innovative solutions. These solutions can span up to profound architecture modifications as it was exploited by Chang et al. [40] to limit the vanishing gradient problem. Laurent et al. [121] also successfully modified the gating architecture of the GRU to remove chaos. Another example of successful tuning with gates for Siamese architectures is the approach by Varior et al. [211] (see Section 3.2.1.3) which added a gate providing an information exchange mechanism.

Regarding model-based metric approaches for time series, they reassemble our approach since they are defined as distances on dynamical systems parameters. They are useful in the context of activity recognition notably to model the dynamical components of motion, however their efficiency is limited for complex daily activities [207]. Moreover, they suffer from scaling issues as it is necessary to find the specific set of parameters for each sequence [130]. Nevertheless, it is possible to compute metrics on the model parameters fitted for one sequence to perform classification or clustering. However, our approach is not exactly the same, as we propose to use dynamical system synchronization theory to improve metric learning on any type of sequential data, whereas these methods have been conceived to work more specifically with structural data.

In the next section, we propose to enhance the classical siamese RNN by studying this model from a dynamical system theory point of view, as it has been already done for standard RNN.

4.3 Synchronizing GRU Siamese Networks

In this section, we first draw a parallel between the concept of synchronization for dynamical systems and the task of sequence metric learning with siamese RNN. We then theoretically justify the introduction of coupling inside the siamese architecture. We finally introduce the major contribution of this chapter, a modified Siamese Gated Recurrent Unit (SGRU) model implementing this coupling and it is trainable by gradient descent.

4.3.1 Synchronization and Sequence Metric Learning

In the Siamese network architecture, both subnetworks share the same parameters [30] and produce two output trajectories from the pair of inputs. We will in the following study the conditions for those two trajectories to be synchronized and what it means in this case.

First, it is necessary to determine what type of system we have to deal with. The RNN have the same parameters and are therefore identical, the system as a whole can therefore always achieve complete synchronization if a coupling is applied [70]. To simplify, we will first study the case where the dynamics of the RNN are solely driven by its initial condition (the initial hidden state) and where no input sequence is given. We obtain what is called the *dynamical system induced* by the RNN. In this case, only the initial conditions differ and the sub-networks are identical dynamical systems. According to experiments conducted by Laurent et al. [121], dynamical systems induced by RNN exhibit a chaotic behavior: complete synchronization is only possible if a sufficiently strong coupling is applied between the systems (see Section 4.1.5). But the trajectories of RNN are most of the time also influenced by external inputs: the input sequence. In this case, the dynamics of the RNN are mostly driven by these external inputs [121] and RNN starting from different initial conditions but given identical input sequence will see their trajectories synchronize, i.e. the hidden states become the same after a few steps. Coupling is in this case not absolutely necessary to achieve synchronization: regarding metric learning, siamese LSTM actually works without coupling [143], the model achieves low distances for similar inputs and therefore synchronization. However, coupling could allow to enforce lower distances with sequences that have similar dynamics but are composed of quite different data, i.e. so-called hard positive samples, and even to force the synchronization regardless of the input pair.

If the trajectories are synchronized, it means that their synchronization error is equal to zero. We will now derive from Equation 4.9 a metric formula adapted to our context. We start by adapting it to discrete systems with finite trajectories by replacing the limit by a comparison of the last element of each trajectory (output sequences) $\hat{S}_1$ and $\hat{S}_2$ of length T:

$$h(g(\hat{S}_1), g(\hat{S}_2)) = \hat{S}_1(T) - \hat{S}_2(T)\,, \tag{4.13}$$

with g being here a function returning the coordinates of the points. We then define d as a distance on discrete dynamical system trajectories derived from the synchronization error by replacing the simple difference

with the Euclidean norm (according to definition 4.2, any norm can be used) to get only positive values:

$$d(\hat{S}_1, \hat{S}_2) = \|\hat{S}_1(T) - \hat{S}_2(T)\|_2. \tag{4.14}$$

According to this derivation, trying to synchronize the output sequences of the two sub-networks of the siamese network is thus equivalent to having a low Euclidean distance between the last outputs, a configuration we are aiming to learn for similar input sequence pairs.

While being intuitive and suitable for metric learning, the metric of Equation 4.14 measures synchronization only at one point in time, which forces the system to achieve synchronization at this precise point. This is called dead-beat synchronization, synchronization in a finite number of steps [60]. Even if at first glance, this seems not really different from computing distance on input sequence representations, synchronization could actually be assessed at several samples of the sequence and even continuously. Moreover, complete synchronization is a special case of more general synchronization notions such as the so-called *generalized* synchronization [170] for which other errors and metrics are associated (for example, mutual interdependence [175]). The same derivation could thus be made for these synchronization errors, leading to different metrics. We will in the experimental section use Equation 4.14 as our metric to learn, but this is here the simplest case of a framework from which more complex sequence metrics to be learned with Siamese RNN models, can be obtained.

By trying to understand under what conditions output sequences of RNN synchronized and interpreting this phenomenon within metric learning, we motivated the implementation of a coupling mechanism inside the siamese RNN architecture. Indeed, the induced dynamics of GRU and LSTM are chaotic and, in this case, coupling is mandatory. When given input sequences, the dynamics of GRU and LSTM are mostly driven by these external inputs. In this other case, while not being critical to achieve synchronization (and therefore low distances between similar elements), coupling could facilitate bringing similar inputs closer, particularly for hard positive pairs.

4.3.2 Coupled GRU

We now present a new neural network model that directly implements coupling within a siamese RNN architecture. From a machine learning perspective, this coupling needs to be trainable such that the network

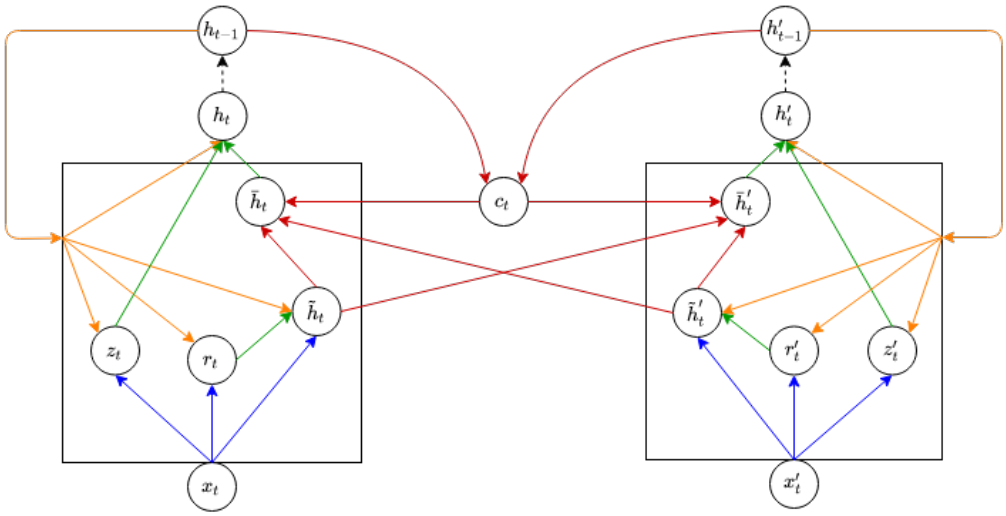


Figure 4.1 – Schema of the CGRU architecture. Blue arrows represent information coming from the input at time t, orange ones are for the hidden states and green ones for transmissions between the gates. Red arrows correspond to equations 4.17 and 4.18.

learns to achieve synchronization for similar inputs and stay desynchronized for different ones. We propose to apply the coupling by the means of two new gates inside the GRU architecture which we call in the following Coupled Gated Recurrent Unit (CGRU).

We chose to use GRU and not LSTM [95] because the operation of the GRU is defined by fewer equations while showing comparable performance in general [48]. The following equations describe the modifications brought to the architecture. Update, Reset and New gates are not modified. Let us notate h'_{t-1} and $\tilde{h}'_t$ the states coming from the second sub-network (see Figure 4.1):

$$\text{Hidden state: } h_t = (1 - z_t) \circ \bar{h}_t + z_t \circ h_{t-1} \tag{4.15}$$

$$\text{Update: } z_t = \sigma(W_{iz} x_t + b_{iz} + W_{hz} h_{t-1} + b_{hz}) \tag{4.16}$$

$$\text{Coupled New State: } \bar{h}_t = (1 - c_t) \circ \tilde{h}_t + c_t \circ \tilde{h}'_t \tag{4.17}$$

$$\text{Coupling: } c_t = \sigma(W_{hc}(h_{t-1} + h'_{t-1}) + b_{hc}) \tag{4.18}$$

$$\text{New State: } \tilde{h}_t = \tanh(W_{i\tilde{h}} x_t + b_{i\tilde{h}} + r_t \circ (W_{h\tilde{h}} h_{t-1} + b_{h\tilde{h}})) \tag{4.19}$$

$$\text{Reset: } r_t = \sigma(W_{ir} x_t + b_{ir} + W_{hr} h_{t-1} + b_{hr}). \tag{4.20}$$

The Coupling gate c_t (see Equation 4.18) serves the same purpose as z_t and r_t, controlling the information flow and is thus computed in a similar manner, but only from the hidden states. This forces the model to

apply the coupling on the new content to be added at time t solely based on the previous inputs. Then, in Equation 4.17, $\tilde{h}_t$ and $\tilde{h}'_t$ are combined similarly as $\tilde{h}_t$ and h_{t-1} are combined in the original GRU architecture. This prevents $\bar{h}_t$ and subsequently h_t from exploding and saturating the gates. Finally, in Equation 4.15, $\bar{h}_t$ replaces $\tilde{h}_t$: the New state has been replaced by a coupled version of both New states of the siamese GRU. Several possibilities exist to implement this coupling. The idea behind this proposal is to alter as little as possible the GRU architecture (see Section 2.1.4.3) and to stay close to the original purpose of each equation. Indeed, RNN can be delicate to train and the addition of the coupling already greatly modifies the information flow inside the GRU and the gradient flow during training. Therefore, by staying relatively close to the original model, a rigorous comparison is more effective, and the impact of the actual coupling can be studied more reliably. In fact, if c_t is a matrix of norm equal to zero, each sub-network is exactly a GRU. This suggests to initialize the coupling weights with very small values and to accentuate the decay. In this way, an increase of the norm of W_{hc} during training would signify that coupling is useful. Another interesting configuration of the coupling weights is when they are all equal to 0.5: in this configuration, the Coupled New States are the same, and the distance between the outputs will become null. That means, theoretically, this approach can make close any pair of input sequences, especially hard-positive samples.

We make the following hypothesis regarding the behavior of CGRU which we validate through experimentation (see Section 4.4). The first one follows the choice of implementation made for the coupling as a new gate able to block coupling if the norm of its parameters is zero. If coupling is useful, this norm should increase:

Hypothesis 4.1. *The coupling weight norm increases during training thus driving away the behavior of CGRU from SGRU.*

The second one proceeds from the analysis of coupling in siamese RNN (see Section 4.3.1):

Hypothesis 4.2. *CGRU is able to produce lower distances between hard positive samples.*

Lastly we propose an hypothesis regarding a general gain of coupling:

Hypothesis 4.3. *CGRU achieves higher performances as SGRU.*

4.3.3 Differentiation

We will now demonstrate that this architecture can be trained by BPTT with gradient descent by doing the complete differentiation of it using the chain rule. We admit we know $\frac{\partial E}{\partial h_t}$, the partial derivative of the loss with respect to the hidden state, and we call dx derivatives of the form $\frac{\partial E}{\partial x}$ with the pair of inputs concatenated (named hereafter left and right inputs). It is sometimes necessary to use separately the left and right sides of the derivative, in which case respectively indicated by $\leftarrow$ and $\rightarrow$ over the indices. Finally, $[x, y, ...]$ designates a concatenation along the suitable axis. Post-synaptic potentials (before activation function) are represented by the letter a with indices i for input and h for hidden and the associated gate letter. We seek to compute:

- dx_t,
- dh_{t-1},
- dW_i where $W_i = [W_{iz}, W_{ir}, W_{in}]$,
- dW_h where $W_h = [W_{hz}, W_{hr}, W_{hn}, W_{hc}]$,
- db_i where $b_i = [b_{iz}, b_{ir}, b_{in}]$,
- db_h where $b_h = [b_{hz}, b_{hr}, b_{hn}, b_{hc}]$.

$$
\begin{aligned}
dh_{t-1,o} &= dh_t \circ z_t && (4.21)\\
dz_t &= dh_t \circ (h_{t-1} - \bar{h_t}) && (4.22)\\
d\bar{h}_t &= dh_t \circ (1 - z_t) && (4.23)\\
dc_t &= d\bar{h}_{\overleftarrow{t}} \circ (\tilde{h}_{\overrightarrow{t}} - \tilde{h}_{\overleftarrow{t}}) + d\bar{h}_{\overrightarrow{t}} \circ (\tilde{h}_{\overleftarrow{t}} - \tilde{h}_{\overrightarrow{t}}) && (4.24)\\
d\tilde{h}_t &= d\bar{h}_t + [c_t, c_t] \circ ([d\bar{h}_{\overrightarrow{t}}, d\bar{h}_{\overleftarrow{t}}] - d\bar{h}_t) && (4.25)\\
da_z &= dz_t \circ ((1 - \sigma(a_z)) \circ \sigma(a_z)) && (4.26)\\
da_n &= d\tilde{h}_t \circ (1 - \tanh(a_{n,i} + r_t \circ a_{n,h})^2) && (4.27)\\
da_{n,h} &= da_n \circ r_t && (4.28)\\
dr_t &= da_n \circ a_{n,h} && (4.29)\\
da_r &= dr_t \circ ((1 - \sigma(a_r)) \circ \sigma(a_r)) && (4.30)\\
da_c &= dc_t \circ ((1 - \sigma(a_{\overleftarrow{c}} + a_{\overrightarrow{c}})) \circ \sigma(a_{\overleftarrow{c}} + a_{\overrightarrow{c}})) && (4.31)\\
da_{\overleftarrow{i}} &= [da_{\overleftarrow{z}}, da_{\overleftarrow{r}}, da_{\overleftarrow{n}}] && (4.32)\\
da_{\overrightarrow{i}} &= [da_{\overrightarrow{z}}, da_{\overrightarrow{r}}, da_{\overrightarrow{n}}] && (4.33)
\end{aligned}
$$

$$da_{\overleftarrow{h}} = [da_{\overleftarrow{z}}, da_{\overleftarrow{r}}, da_{\overleftarrow{n,h}}, da_c] \tag{4.34}$$
$$da_{\overrightarrow{h}} = [da_{\overrightarrow{z}}, da_{\overrightarrow{r}}, da_{\overrightarrow{n,h}}, da_c] \tag{4.35}$$
$$dW_{iz} = da_{\overleftarrow{z}} x_{\overleftarrow{t}} + da_{\overrightarrow{z}} x_{\overrightarrow{t}} \tag{4.36}$$
$$dW_{ir} = da_{\overleftarrow{r}} x_{\overleftarrow{t}} + da_{\overrightarrow{r}} x_{\overrightarrow{t}} \tag{4.37}$$
$$dW_{in} = da_{\overleftarrow{n}} x_{\overleftarrow{t}} + da_{\overrightarrow{n}} x_{\overrightarrow{t}} \tag{4.38}$$
$$dW_i = [dW_{iz}, dW_{ir}, dW_{in}] \ \square \tag{4.39}$$
$$dW_{hz} = da_{\overleftarrow{z}} h_{\overleftarrow{t}} + da_{\overrightarrow{z}} h_{\overrightarrow{t}} \tag{4.40}$$
$$dW_{hr} = da_{\overleftarrow{r}} h_{\overleftarrow{t}} + da_{\overrightarrow{r}} h_{\overrightarrow{t}} \tag{4.41}$$
$$dW_{hn} = da_{\overleftarrow{n,h}} h_{\overleftarrow{t}} + da_{\overrightarrow{n,h}} h_{\overrightarrow{t}} \tag{4.42}$$
$$dW_{hc} = da_c (h_{\overleftarrow{t}} + h_{\overrightarrow{t}}) \tag{4.43}$$
$$dW_h = [dW_{hz}, dW_{hr}, dW_{hn}, dW_{hc}] \ \square \tag{4.44}$$
$$db_i = da_{\overleftarrow{i}} \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} + da_{\overrightarrow{i}} \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \ \square \tag{4.45}$$
$$db_h = da_{\overleftarrow{h}} \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} + da_{\overrightarrow{h}} \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \ \square \tag{4.46}$$
$$dx_t = [da_{\overleftarrow{i}} W_i, da_{\overrightarrow{i}} W_i] \ \square \tag{4.47}$$
$$dh_{t-1} = dh_{t-1,o} + [da_{\overleftarrow{h}} W_h, da_{\overrightarrow{h}} W_h] \ \square \tag{4.48}$$

4.4 Experiments

4.4.1 Datasets and Experimental Setup

4.4.1.1 SHL Dataset

This dataset was the object of a challenge in 2018 [220], the Sussex-Huawei Locomotion and Transportation (SHL) Dataset [79]. The data were recorded by a single individual during a 4 month period for a total of 82 days: 62 for training, 20 for testing. Up to 8 hours of data are recorded each day. This dataset proposes 8 locomotion transportation modes: car, bus, train, subway, walk, run, bike and standing still. It contains 20 features : accelerometer, gyroscope, magnetometer, gravity and linear acceleration on three axes, orientation on 4 axes and pressure.

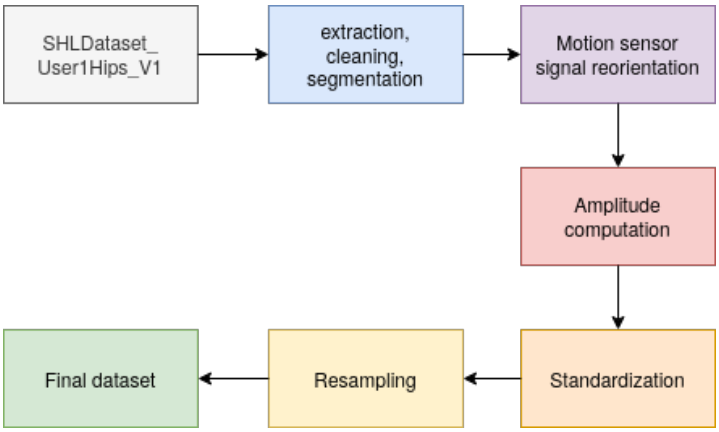


Figure 4.2 – Preprocessing steps applied to SHL dataset, we globally followed the process proposed by Janko et al. [104].

The raw dataset is sampled at a frequency of 100 Hz, that is to say, one hour sequences have a length of 360000. To use it for our experiments, we roughly followed the same preprocessing procedure as in [104], with some more steps (see Figure 4.2): all signals (except pressure) were re-oriented to the North-East-Down axes convention and magnitudes were computed for the accelerometer, the gyroscope and the magnetometer. Orientations were converted to Euler angles. This process extends the number of features in the dataset from 20 to 36. Then, the dataset was standardized according to the training set to have, for a every feature, a mean of zero and a standard deviation of 1. Finally, to speed up computations, the 1 minute sequences we used of a length of 6000 were resampled to a length of 300. The challenge summary paper [220] reports that the best test accuracy result of 93.9% was achieved by Gjoreski et al. [80] with a deep learning approach.

4.4.1.2 Other Datasets used in this Chapter

Two other datasets are used in this chapter. The first is the UCI HAR dataset (see Section 2.3.1.3) to perform activity recognition. Regarding the UCI HAR dataset, several activities in this dataset should look very similar (e.g. three variants of walking or standing and sitting), and it should make the dataset harder to process for metric learning algorithms. Finally, *walking* or *running* exhibits dynamic components which could be differently processed by CGRU and SGRU. No further preprocessing has been applied. We kept the train-test split proposed by the authors of the

dataset: there are 21 users in the training set and we therefore performed a 7-fold validation, leaving each time 3 different users out. Finally, the training set contains 7352 sequences and the testing set 2947.

The second dataset is the LTMM dataset (see Section 3.4.1.1) to perform routine retrieval following the same process presented in Section 3.3.4.

4.4.1.3 Experimental Setup for classification

We compared SGRU and CGRU on learning the metric describes in Equation 4.14 using the same hyperparameters for both models, those parameters for each datasets are displayed on Table 4.1. The training is stopped based on the accuracy on the validation set (early stopping). We chose to use the structural loss [235] (see Section 3.2.3.2) to train the model since it is a loss working on distances and not embeddings. It combines a local term similarly to the n-pair loss [183] (see Section 3.2.3.2) but emphasizes the weights on the hard positive samples, and a global term to improve the generalization. We used the same hyperparameters[2] for the loss as in the original paper and adapted the batch sizes accordingly to each dataset. The gradient was clipped according to [156] to a norm of 6. We applied a general weight decay with a factor of 10^{-4}. A stronger weight decay was applied on the coupling by adding 1% of the coupling weight norm to the loss, which seems to slightly improve the performances. The coupling weights were also initialized with a normal distribution having a mean of 0 and standard deviation of 0.1. The purpose of this initialization is to make the model start its training close to the behavior of an SGRU, with a very weak coupling and to let it increase during training. We explore this feature in the preliminary experiments which follow. Our implementation was made in Python and CUDA with Pytorch [157].

4.4.2 Preliminary experiments on UCI HAR

We present some preliminary experiments on the UCI HAR dataset to study the properties of the CGRU architecture.

[2]Namely (see section 3.2.3.2 for the exact role of each hyperparameter): $m0.2$, $m^+ = 0.01$, $m^- = 0.1$, $\gamma 0.95$, $\eta = 0.05$, $\lambda = 0.5$.

Hyperparameters	UCI HAR	SHL
Training set size	7352	15660
Testing set size	2947	5472
Architecture	[20]	[100, 100]
Batch size	36	40
Initial learning rate	0.001	0.001

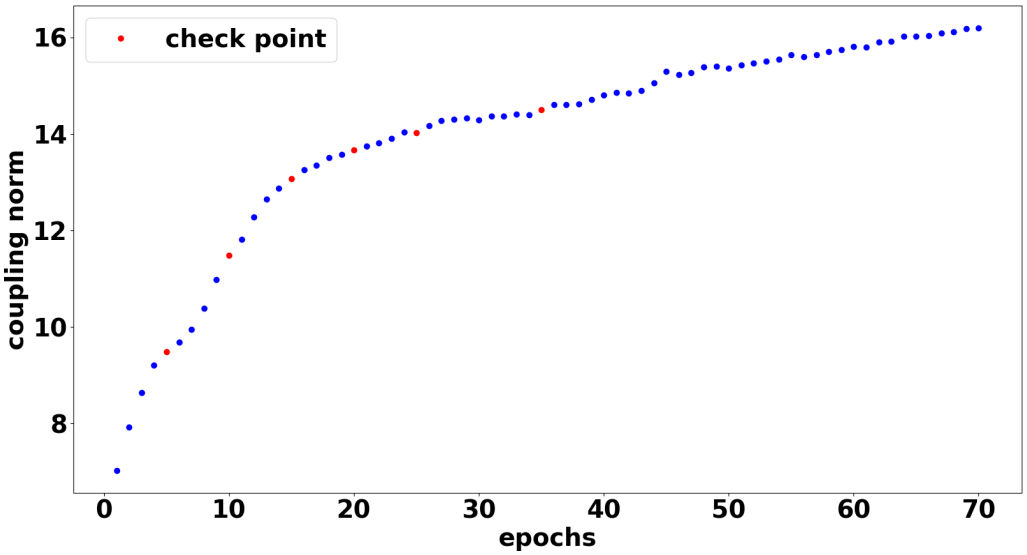
Table 4.1 – Main hyperparameters used to train our models on both datasets. These parameters are exactly the same for SGRU and CGRU.

4.4.2.1 Study of the coupling weight norm

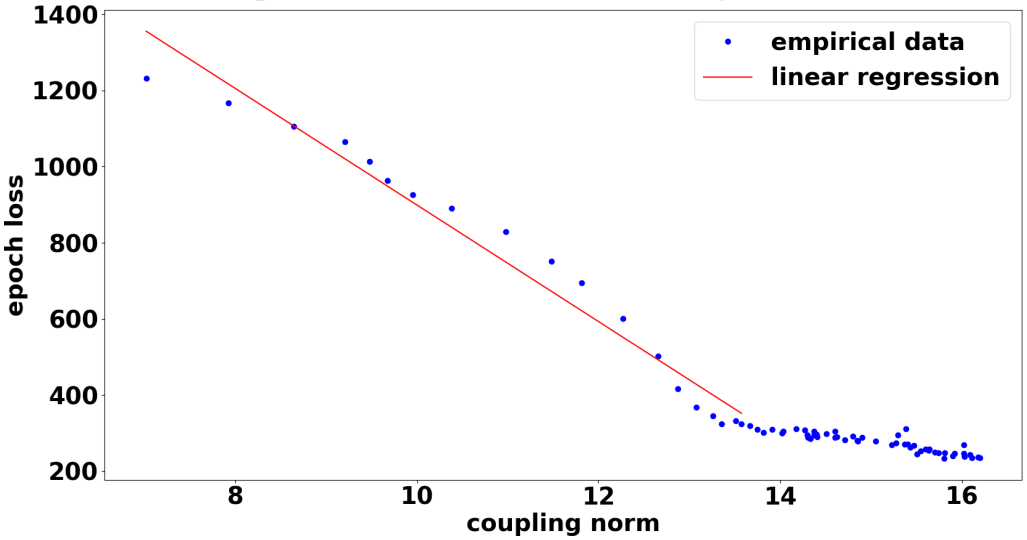
We first analyze the evolution of the coupling weight norm during training. The coupling is initialized very low which theoretically makes it behave nearly as a SGRU at the beginning of training. We can observe on Figure 4.3a that the norm increases quickly during the first 20 epochs and more than doubles despite the stronger weight decay. Thus, the model is driven away from the behavior of a SGRU. Red points indicate the iterations where validation accuracy increased. This correlation thus indicates that the overall generalization is improving with the increase of coupling strength. On Figure 4.3b, we present the evolution of the loss on the training set in terms of the coupling weights on which we compute a linear regression of the first part. We can observe an almost linear relationship between the increase of the norm and the decrease of the loss suggesting again that the coupling is helpful to the model and validating hypothesis 4.1. Those figures also confirm that the convergence during training is rather smooth.

4.4.2.2 Performances on hard positive samples

We now seek to validate hypothesis 4.2 that CGRU could perform better on hard positive samples due to coupling theoretically being able to bring close any input pair of sequences: with enough coupling, both networks can output the same sequence whatever of the input pair. We propose to verify this by observing the evolution of the average distance between the positive samples and between the negative samples when white noise is gradually added to the feature sequences of the testing set. The standard deviation of the applied noise goes from 0 to 2. We also note that both models were trained up to comparable validation accuracy. The results are presented on the Figure 4.4. We observe that the curves for both models evolve similarly with positive distances



(a) Evolution of the norm of the coupling gate weights during training on UCI HAR dataset, a red point indicates an increased accuracy for the validation set.



(b) Epoch loss in terms of coupling norm during training on UCI HAR dataset. The first part (rapid decrease) has been approximated with a linear regression. The correlation coefficient is -0.984.

Figure 4.3 – Experiments with the coupling norm.

gradually increasing with the noise. The negative and positive curves join the moment too much noise is added and the sequences become indistinguishable. CGRU produces slightly higher distance averages than SGRU but is able to maintain higher margins more longer: up to

1.75 units of standard deviation compared to about 1.25 for SGRU. This seems to show that, as theoretically possible with the coupling, CGRU better discriminate the hard samples (hypothesis 4.2).

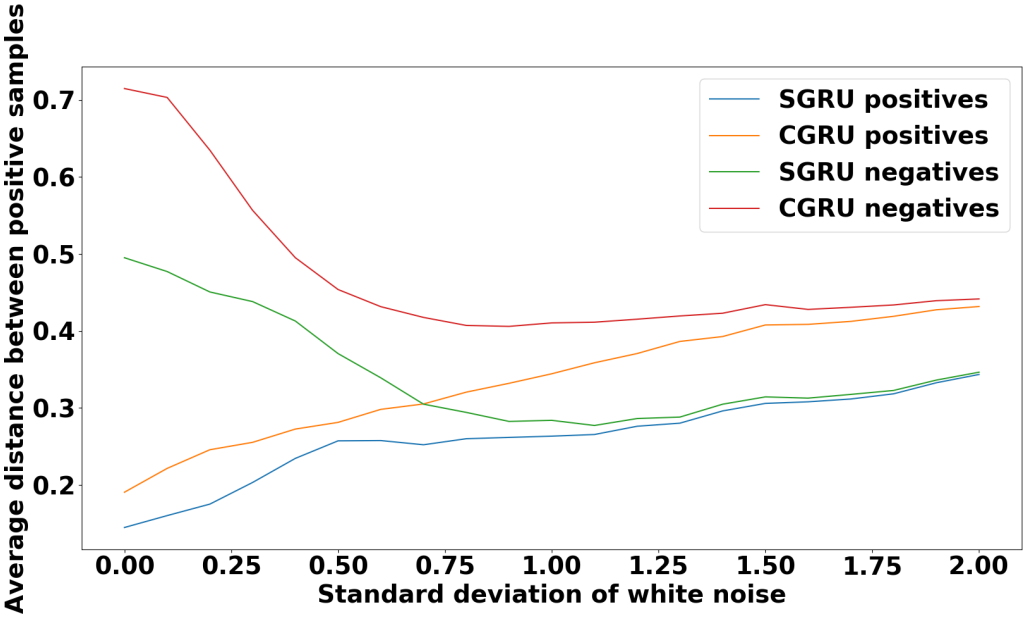


Figure 4.4 – Evolution of the average distance between the positive/negative samples for SGRU and CGRU on more and more noisy test sets.

4.4.2.3 Average normalized distance to nearest neighbors

We present in Figure 4.5 the evolution of the average distance to the nearest neighbors depending on the number of nearest neighbors. Each distance has been scaled between 0 and 1 prior to the average. For SGRU, we observe that the average distance grows slowly at first until some point between 150 and 200 (around the average number of samples per class in this validation set, which is about 165) and from there starts to grow faster. This seems to indicate that the training of SGRU has allowed the model to create an important margin between similar and dissimilar samples. The curve for CGRU is very different with a very smooth progression and no apparent margin. This can be explained by the coupling which always combines the New States in some extend even if the input are dissimilar and therefore produces closer outputs. Despite the absence of apparent margin, we shall see now if CGRU achieves lower classification performances than SGRU.

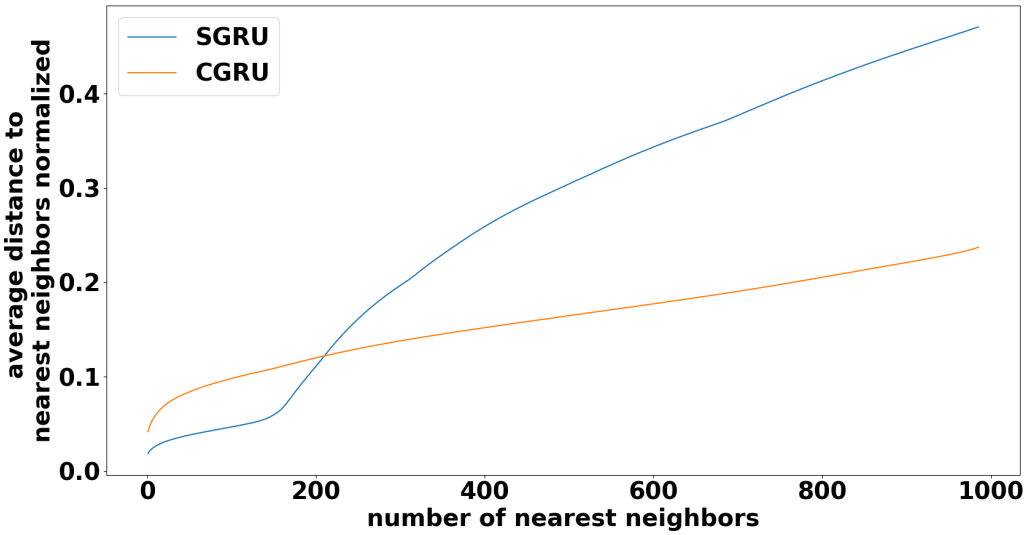


Figure 4.5 – Average normalized distance to nearest neighbors for SGRU and CGRU computed on the validation set (users 1, 3 and 5).

4.4.3 Classification performances

To assess the classification performances, we propose to use three scores: accuracy, F1 score Macro averaged and Mean Average Precision (MAP), similarly to [191]. The first two are computed by classifying the test samples using 1-nearest neighbor from training samples. The MAP is computed by querying the training set with a test sequence to retrieve all training samples of the same class. The value is averaged for all test sequences. This metric shows the ability of the algorithm to bring close every sequence of each class and not just few references to be used as nearest neighbors.

4.4.3.1 Activity recognition on UCI HAR

We now present the classification results on UCI HAR. The validation results are presented in Table 4.3a. We observe an improvement of CGRU over SGRU of about 8% for accuracy and F1-score, and an improvement of 19% points for the MAP. On the test set, we compared our approach with Regressive Virtual Sequence Metric Learning (RVSML) [191] (see Section 3.2.2.2), with OPW and DTW distances. The test results are presented in Table 4.3b. Here again we observe that CGRU outperformed SGRU with a notable improvement of 10% points of the accuracy and F1-score. Both neural network approaches clearly outperformed RVSML, especially the OPW variant. This can be, among other factors, attributed

Algorithms	Accuracy	F1 score Macro	MAP
Siamese GRU	0.835±0.068	0.827±0.074	0.711±0.016
Coupled GRU	**0.913±0.055***	**0.916±0.054***	**0.900±0.043***

(a) Validation results (21 fold average). An asterisk means a significant result with a threshold of 1%.

Algorithms	Accuracy	F1 score Macro	MAP
RVSML (OPW) ([191])	0.597	0.568	0.438
RVSML (DTW) ([191])	0.698	0.687	0.437
Siamese GRU	0.782±0.041	0.781±0.044	0.633±0.152
Coupled GRU	**0.885±0.014***	**0.887±0.014***	**0.899±0.01***

(b) Test results, average of 5 run for the neural networks approaches.

Table 4.2 – Results on UCI HAR

to a weak capacity to distinguish similar activities such as *standing* and *sitting*. On the other hand, *standing* was recognized perfectly by both RVSML variants. We also remark that they reach MAP values of the same order as in the original paper (around $0.4 \sim 0.45$) despite the fact that the data are of a completely different nature (signal instead of images). This could suggest that these approaches reached some kind of saturation whereas CGRU and SGRU are able to achieve much higher values. The best approaches on UCI HAR (see Section 2.3.1.3) achieved above 95% of test accuracy: CGRU achieved results around 7% lower.

4.4.3.2 Transportation Recognition on SHL Dataset

The second experiment is made on a larger dataset with twice as much data and a longer sequence size. The validation results are presented in the Table 4.2a. Both architectures achieved results above 90% for each metric but we observe a slight improvement of CGRU over SGRU for the considered architecture. These results are further confirmed on the test set (see Table 4.2b) with even larger improvements: 2,5% for accuracy, 2.2% for F1-score and 1.5% for MAP, in favor of CGRU. These results confirm the interest of our proposed approach CGRU compared to SGRU. However, compared to the overall results of the challenges [220], both approaches achieved similar validation results but clearly do not demonstrate the same generalization capacities on this dataset as the best ad-hoc approaches.

Algorithms	Accuracy	F1 score Macro	MAP
Siamese GRU	0.947±0.006	0.951±0.006	0.955±0.007
Coupled GRU	**0.965±0.05**	**0.967±0.006**	**0.97±0.004**

(a) Validation results (10 fold average).

Algorithms	Accuracy	F1 score Macro	MAP
Siamese GRU	0.731±0.023	0 .75±0.018	0.786±0.02
Coupled GRU	**0.747±0.019**	**0.765±0.016***	**0.798±0.018**

(b) Test results (10 fold average). An asterisk means a significant result with a threshold of 5%.

Table 4.3 – Results on SHL dataset.

Algorithms	UCI HAR	SHL
SGRU	44	45
CGRU	32	350

(a) Time to perform one epoch of training.

Algorithms	UCI HAR	SHL
SGRU	24	83
CGRU	409	16392

(b) Test computation time.

Table 4.4 – Computation times (in seconds) of SGRU and CGRU for one epoch and processing the test set on UCI HAR and SHL.

4.4.3.3 Computation times comparison

To give an appreciation of the difference regarding the complexity between SGRU and CGRU, we report in Table 4.4 the computation times for one epoch and to process the test set on both datasets. These times were obtained by processing the networks on an NVidia P6000 GPU with 24GB of video memory. We observe a large difference between the two variants, CGRU being more computationally expensive, notably for tests: for SHL dataset, the difference is of several orders of magnitude. One notable exception is to compute on epoch for UCI HAR dataset with a slight advantage of CGRU over SGRU.

Two factors can explain this difference. Firstly, coupling implies that the output sequences for each pair are dependent which forces to pass each sequence with each other. This corresponds to $|\mathbb{B}|^2/2$ passages in the network, where b is the batch size, and to the same number of distance

computations. For SGRU, where the outputs are independent, only $|\mathbb{B}|$ passages in the network are necessary. The second factor lies directly into the GPU implementation of CGRU: it seems to require 2 kernel calls to be executed. A kernel is a function to be mapped on each thread of the GPU, each time with different indices which often correspond to values in matrices. This allows to perform element-wise matrix operations in parallel but comes with a time overhead. CGRU seems to require two calls because certain values with different indices needs to be known to compute the coupling. One way to be sure of that is to finish one kernel and then launch another. SGRU should require only one kernel call in comparison. Finding an implementation of CGRU which makes only one kernel call would greatly improve the performances.

4.4.4 Routine Retrieval on LTMM

We finally test the CGRU architecture on the routine retrieval problem we presented in chapter 3. We used here neural networks with one layer of 100 neurons otherwise the other hyperparameters are the same as for UCI HAR (see 4.4.1.3). The results presented in Table 4.5 below are directly comparable with the results of Table 3.2. Compared to the previous results, we observe that both approaches are outperformed considering Completeness, Silhouette and NMI but clearly outperform the other ones for AMI. As a reminder, AMI corrects the NMI by rendering it independent from the number of clusters (see Section 3.3.4). We observe that the gap between those two scores is narrower for SGRU and CGRU than for SS2S variants which makes their NMI scores more reliable although inferior. Concentrating now only on the results from Table 4.5, we see that SGRU, slightly but significantly, outperforms CGRU on LTMM dataset for all metrics except Silhouette. We attribute the slight gain on silhouette to the coupling being able to bring closer hard-positive samples, producing more compact clusters as a result. This is another hint confirming Hypothesis 4.2. Regarding Hypothesis 4.3, if it is validated for the classification experiments, SGRU seems to conserve the superiority on routine retrieval.

To conclude the experiments on CGRU, we propose on Figure 4.6 the visualization of a clustering obtained with CGRU. The results look less clean than for SS2S and DTW (see Figure 3.7) even if the same regularities are visible. We also observe a very clear shifting pattern repeated itself across the day (one hour early) around the virtual days nine and ten which could suggest a change of schedule for the last recorded day of the three.

Model	Completeness	Silhouette	NMI	AMI
SGRU	**0.882±0.052***	0.263±0.024	**0.568±0.035***	**0.54±0.037***
CGRU	0.843± 0.043	**0.286± 0.025***	0.545±0.029	0.515± 0.031

Table 4.5 – Comparison on LTMM dataset of CGRU and SGRU, averages on 20 tests. An asterisk means a significant result with a threshold of 5%.

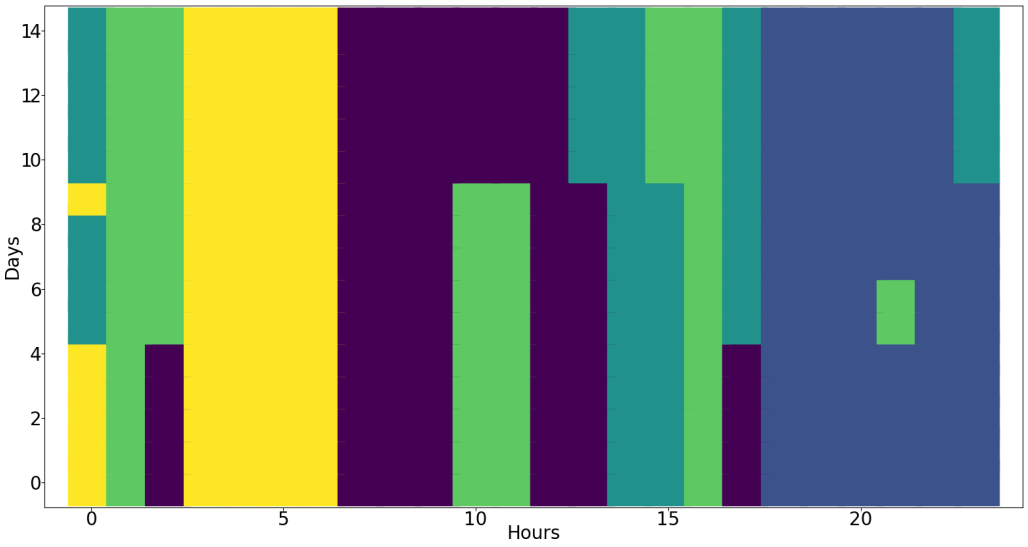


Figure 4.6 – Example of a clustering obtained with CGRU on LTMM.

4.5 Conclusion and Perspectives

We presented a new framework for sequence metric learning based on dynamical system synchronization theory. We drew a parallel between synchronized trajectories and output sequences of siamese recurrent neural networks produced from similar input pairs. We showed the contribution of introducing coupling inside the siamese architecture to balance the chaotic behavior of GRU and LSTM and to increase the capacity of the network to bring closer the embedding of some similar input pairs, especially hard positive samples. This coupling was implemented through a new gate inside the SGRU architecture which allows the network to mix the new content of both sides of the siamese network. Our experiments showed that the SGRU architecture benefits from the coupling, can be smoothly trained with it and fits well with recent complex metric learning losses such as structural loss. CGRU proved to be outperforming SGRU with the same architecture on two datasets for activity and transportation recognition. It was also able to

maintain a higher margin between hard samples in conformity with what we supposed when justifying the introduction of the coupling. We also tested this architecture to retrieve routines with more mitigated results although CGRU achieved a higher silhouette score than SGRU further confirming the hypothesis that it is able to enforce lower distances.

The study of sequence metric learning with synchronization opens several perspectives especially to design new forms of metrics by taking inspiration from the literature of synchronization criteria [114, 175]. However, the issue to overcome in numerous cases is non-smoothness, notably when those metrics use mutual neighbors which are computed using explicitly the time steps. The attention mechanism [212] can constitute here an interesting research perspective. The coupling itself could be tuned with theoretical contributions [32]. One drawback of the proposed architecture is that each pair has to be passed through the network instead of just computing once each representation and then the distance for each pair. This could be balanced by the use of virtual metric learning during training. Finally the coupling allows to bring any pair of inputs close to one another if sufficiently strong and could be use as an indicator in weakly supervised settings to invert the equivalence constraint of some pairs dynamically if the network is forcing the synchronization too much. We further develop these ideas in the perspective section of the manuscript.

CHAPTER 5

CONCLUSIONS, LIMITATIONS, PERSPECTIVES

This last chapter will be dedicated to conclude the manuscript by making a synthesis of the contributions. We will then highlight some limitations of the proposed approaches before suggesting perspectives to overcome these issues.

5.1 Conclusion

The objective of this thesis was to propose new neural network architectures conceived to overcome the challenges of real life activity recognition and monitoring notably for eHealth services. This task requires not only high performances but also flexibility and privacy due to the sensibility of the processed data.

5.1.1 Flexible Activity Recognition with Few-Shot Learning

A first step was to work with personalized models, more respectful of privacy and supposedly able to achieve higher performances. However, this has the downside of putting the burden of providing labels on one single user, retarding the moment when the actigraphy system would properly work. To alleviate this burden, we propose to employ few-shot learning. Few-shot learning models are able to recognize new classes from just one or few new training samples. We put to contribution this learning paradigm by suggesting modifications of the

architecture by Vinyals et al. called Matching Networks [217] to adapt it for sequence classification and to leverage its generalization capabilities. The SSMN architecture, rather than directly classifying samples, learns to match them with already known support examples of different classes. When the model is trained, it can start recognizing new classes by just giving it one new support example. We tested this architecture on two activity recognition datasets to perform personalized activity recognition with interesting results notably to detect falls and on UCI HAR. On the *MiniMobiact* dataset, a dataset with more classes, the results were more mitigated for the 12-way classification but the training data were extremely limited which suggests that nevertheless, a real actigraphy system could work day one with acceptable performances.

5.1.2 Weakly-Supervised Routine Retrieval with Metric Learning

Pursuing our quest for flexibility and ease of use for the user, we then proposed to completely dropped class labels in favor of retrieving recurrent activity patterns we called routines. We started by proposing a mathematical formulation of this concept based on pseudo-periodic functions [26] and we derived from here an equivalence with sequence metric learning. To fulfill this framework, we proposed an architecture combining efficient representation learning with a Sequence-to-Sequence model [197] and metric learning with a siamese architecture [30]. This architecture implements some form of multitask learning for which the intermediate representation is the one learned by the Seq2Seq model. Different metric losses can be plugged in to perform the learning of the metric. We also proposed a routine retrieval learning and testing process based on clustering and information theoretic scores by using data recording time as labels. We put into application this process on a continuous 3-day dataset of the activity of a fragile person in its living environment which we artificially extended to 30 days. We observed that the SS2S model achieved good results on the routine retrieval task compared to DTW and siamese LSTM, especially with the KISSME loss. A visual observation of the results seems to confirm that routines could be retrieved in this case.

5.1.3 Coupled GRU, a New Sequence Metric Learning Architecture

In the last chapter, we presented a new neural network architecture for sequence metric learning called CGRU. This architecture takes inspiration from the dynamical system theory concept called synchronization. We incorporate deep metric learning on sequences inside the framework of synchronization by viewing siamese recurrent neural network as two identical dynamical systems. As a consequence, the output sequences with a low distance between them can be viewed as synchronized. Dynamical system theory guarantees that identical dynamical systems can always achieve synchronization if a sufficiently strong coupling is applied between them. This theoretical argument gives incentives to implement a coupling inside the siamese architecture, notably to bring closer hard positive samples. Precisely, this coupling was implemented by the mean of a new gate inside the GRU neuron to be used in a siamese architecture. By doing so, we tried to stay close to the original behavior of the GRU so as to get a functional network, which we proved was perfectly differentiable. We compared this architecture to SGRU on two classification tasks: activity recognition on UCI HAR dataset and transportation recognition on the University of Sussex-Huawei Locomotion (SHL) dataset. We observed that in both cases, CGRU outperformed SGRU. We then compared the two on the routine retrieval tasks: both achieved interesting NMI and AMI values compared to SS2S with a slight advantaged for SGRU. However, the better silhouette value of CGRU gives an additional hint that it better processes the hard positive samples than SGRU.

5.2 Limitations

5.2.1 Limitations with the SSMN architecture

The proposed SSMN architecture present several limitations. First, its performances in the full setting (when the model is given the choice between all the classes) were quite lower than the state of the art. This could probably be improved by using larger batch sizes during training thus making the model used to select between a lot of classes. On this matter also, the pretraining of the encoding part seemed to help the model very marginally while we thought otherwise. One thing that we can doubt is that the encoding part (see Figure 2.5) really acts as a good feature extractor: it could act like a discriminator or a kind

of filter instead. This guess is reinforced by the fact that Matching Networks [217] do not properly classify features and would just need the representations to be well separated rather than properly characteristic. Moreover, training the encoder with a very different dataset, with a different sampling rate, etc. should suffice to produce such filter while it seems unlikely it could produce a good representation. This intuition could lead us to rethink this part of the architecture and to improve it by using a (randomly generated) reservoir as a better temporal filter [204] such as for the ESN [103].

5.2.2 Limits in the Validation of the Routine Retrieval Process

Although the proposed approaches aimed at being implemented for real environments, they were not tested in real conditions. This is particularly illustrated with the LTMM dataset (see Section 3.4.1.1) which although being perfect regarding quality and realism stays very short (three days). We needed to artificially expand this dataset to experiment on it which rises questions on the performances the algorithm should achieve with more real days. Alternatives are not easy to find. The extrasensory dataset [208] constitutes an interesting option although no more than about ten incomplete days are available for a single user, at most. The most efficient approach would probably to equip oneself with a watch to record the data.
The principle itself behind the concept of routine we employed (see Section 3.1.2) and the labelling process made from the timestamp are subject to caution. A first critic could be that this principle does not take into account context although if we restrict ourselves to wearable motion sensor data, context information are rather limited. Similarly for the labelling, its ability to capture the routines in a weakly supervised way seems convenient and elegant but is to be experimentally confirmed on larger datasets.

5.2.3 Limitations of CGRU

The proposed CGRU architecture is not exempt of drawbacks despite being theoretically grounded and showing interesting results compared to its counterpart without coupling. We already experimentally tackled in Section 4.4.3.3 the computation time comparison with SGRU which is largely in disfavor of the coupled version. It can be explained by the dependent relationship between the output sequences and what seems

to be implementation necessities. This has to be combined with the scalability problems inherent to metric learning due to pairwise (or triplet wise, or even more complex with structural loss [235], etc.) comparisons. Another limitation comes under the form of the learned distance with CGRU which is basically an Euclidean one. The framework of dynamical system synchronization seems able to offer us more powerful distances, potentially acting on a larger part of the sequences and not just the last outputs, but they seem difficult to setup due to being non-smooth.

5.3 Perspectives

To finally close this manuscript, we will now develop three perspectives on the presented work which address some of the limitations we just mentioned.

5.3.1 Temporal Dependent Information Theoretic Score for Clustering

Information theoretic metrics are classical approaches to evaluate clustering algorithms [215]. We explored in chapter 3 the use of NMI and AMI to evaluate metric learning algorithms on the routine retrieval task. A a recall, the Mutual Information can be expressed the following way with Kullback-Leibler divergence (see Equation 3.51 for "sum" formula):

$$\mathrm{MI}(U, V) = KL(P(U, V)||P(U)P(V)), \tag{5.1}$$

where U and V are two discrete random variables. However, these scores make the assumption that the samples are independent of each other whereas in the case of routines, a temporal dependency should exist between the samples. Actually, two kinds of dependencies should appear. The first one, which we call cyclic dependency is directly linked to the way we formulated routines initially (see Section 3.1.2): across the days, similar activities should be performed roughly at the same time making the probability to put the samples in the same cluster higher. This dependency is already measured in some extend by the completeness. The second dependency that should appear is between consecutive routines: e.g. people eat their breakfast after sleeping. This makes both routine occurrences correlated. In this regard, we are particularly interested by the Symbolic Transfer Entropy (STE) proposed by Staniek et al. [186] which takes itself inspiration from the permutation entropy

[11]:

$$\mathrm{STE}_{Y\to X} = \mathrm{KL}(P(U_{n+1}|U_n, V_n)\|P(U_{n+1}|V_n)). \tag{5.2}$$

Taking inspiration from this expression, we define a Periodic Mutual Information (PMI):

$$\mathrm{PMI} = \mathrm{KL}(P(U_{n+p}|U_n, V_{n+p}|V_n)\|P(U_{n+p}|U_n)P(V_{n+p}|V_n)), \tag{5.3}$$

with a period p which can be 24 hours, for examples. With a lower period, we can measure dependency between consecutive routines. If V is here the hour assignment, PMI should measure the difference between a perfectly timely correlated assignment and the routine assignment U. This score could then be integrated in a metric learning loss in a similar manner as in [152]. It can also undergo similar normalization and adjustment as NMI and AMI [215].

5.3.2 Virtual Coupled GRU

One major drawback of the CGRU approach is its scalability. To mitigate this downside, we propose to employ virtual metric learning [161] (see Section 3.2.1.1). Instead of having to make all the similar elements close to each other during training, the model will be trained to make them close to one virtual element. Following the work of Su et al. [191] (see Section 3.2.2.2), in the case of sequences virtual metric learning, virtual elements can be sequences of unitary vectors with a size equal to the number of classes, which are one-hot encoded. To match the number of input features, the virtual sequences can simply be projected with a (learnable) linear layer.

If we combine this approach with the structural loss of Yang et al. [235] (see Section 3.2.3.2), it requires that every positive elements of each class be compared with each other and every positive elements with each of the other negative elements of the batch. More formally, let n be the number of elements per class of the batch with batch size $|\mathcal{B}|$ and c classes. This leads for the first part to $(n^2-n)/2$ comparisons, assuming the metric is symmetric. For the second part, to $n(|\mathcal{B}|-n)/2$ comparisons. This process must be repeated c times. An adapted version of the structural loss with virtual metric learning would only require to compare each element of the batch with each virtual sequence, that is to say, only $c|\mathcal{B}|$ comparisons are required, to be arranged properly for structural loss.

However, this approach also poses a couple of issues when combined with coupling. Firstly, it creates an asymmetry during the learning if the sample and the virtual sequence are always presented respectively to

the same side of the network. The risk here is to completely deteriorate the performances since at test time, samples will be presented instead of virtual sequences. One way to potentially prevent this phenomenon would be to randomly exchange the sample and the virtual sequence. Secondly, the sample sequence lacks a proper counterpart to be coupled with due to the virtual sequences being one-hot encoded and therefore not reassembling signal data. This is also an issue at test time here again because it is supposed that true samples are compared. A way out of this impasse would be to use "realistic" virtual sequences, produced by a generative model such as a variational autoencoder [110] or a generative adversarial model [84]. In the case of the variational autoencoder, the virtual sequences could be decoded class centroids.

5.3.3 Make CGRU Achieve Generalized Synchronization by Learning with Smooth Mutual Interdependence

The literature presents several other synchronization metrics notably to detect another type of synchronization called *generalized synchronization* which identical synchronization is a special case of [111]. In the case of mutually coupled systems, generalized synchronization ensures the existence of an invertible function χ such that:

$$y(t) = \chi(x(t)), \tag{5.4}$$

and conversely [170]. This definition implies the design of more intricate synchronization error metrics such as the Mutual False Nearest Neighbor (MFNN) Parameter [170] or the Mutual Interdependence [175]. Both metrics are based on the concept of mutual neighbors. Mutual neighbors are points with the same time indices which are close for both trajectories to a reference point (see Figure 5.1).

The MFNN Parameter is defined with the following equation:

$$\mathrm{MFNN}(S_1(t), S_2(t)) = \frac{|S_2(t) - S_2(t_{nnS_1(t)})|}{|S_1(t) - S_1(t_{nnS_1(t)})|} \cdot \frac{|S_1(t) - S_1(t_{nnS_2(t)})|}{|S_2(t) - S_2(t_{nnS_2(t)})|}, \tag{5.5}$$

where $nnS_1(t)$ designates the nearest neighbor of $S_1(t)$ and conversely. The value of this parameter is between 0 and $+\infty$, synchronization is achieved if the value is closed to 1. However, two issues arise from this metric: it is non smooth due to the intervention of time indices and its minimum around one is difficult to reach since the value cannot

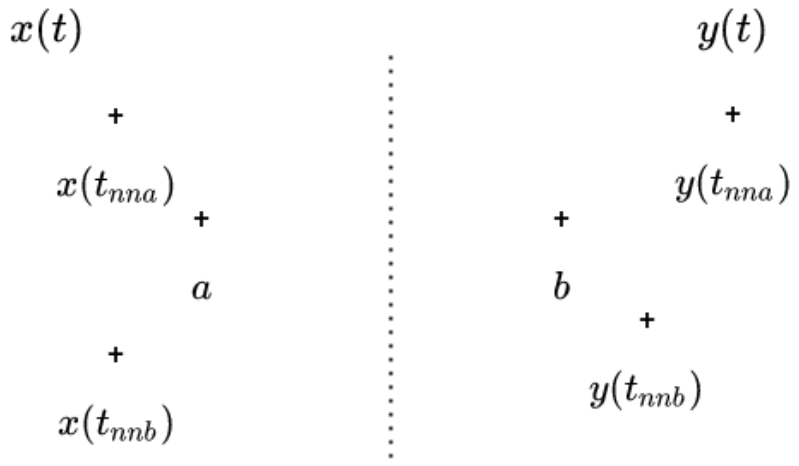


Figure 5.1 – We seek to compare the trajectories $x(t)$ and $y(t)$ around the points a and b at the same timestamp. We search for both trajectories the nearest neighbor of each point: $x(t_{nna})$ for a and $y(t_{nnb})$ for b. The mutual neighbors are the points with the same timestamp in the other trajectory.

go toward negative infinity. Therefore building a loss function around it to train a network with gradient descent appears difficult. Another metric using mutual neighbors proposed by Schiff et al. [175] seems more promising: mutual interdependence. It is based on the idea that generally synchronized sequences should be mutually predictable from the mutual neighbors after a translation. Its value is between 0, no synchronization and 1, totally synchronized. The reconstruction error is then divided by the reconstruction error achieved when averaging all the elements of sequence:

$$\tau(S_1(t)) = \frac{|S_1(t+H) - \frac{1}{k}\sum_{i=1}^{k} S_1(i_{nnS_2(t)} + H)|}{|S_1(t+H) - \frac{1}{T}\sum_{i=1}^{T} S_1(i)|}, \tag{5.6}$$

with H a translation horizon, k a number of mutual neighbors and T a sequence size. The mutual predictability $\tau(S_1(t))$ is of course non-smooth under this form since it requires to explicitly use the timestamps of the mutual neighbors. We thus propose the following smoothed version to be used on output sequences which uses a mutual translated neighborhood weighted by the softmax of the distances instead of the mutual nearest neighbors:

$$\tilde{S}_1(t+H) = \sum_{i=-k}^{+k} \frac{\exp(d(\hat{S}_2(t), \hat{S}_2(t+i)))}{\sum_{i=-k}^{+k} \exp(d(\hat{S}_2(t), \hat{S}_2(t+i)))} \hat{S}_1(t+i+H), \tag{5.7}$$

which gives within the full equation:

$$\tau(\hat{S}_1(t)) = \frac{|\hat{S}_1(t+H) - \tilde{S}_1(t+H)|}{|\hat{S}_1(t+H) - \frac{1}{T}\sum_{i=1}^{T}\hat{S}_1(i)|}. \tag{5.8}$$

The minimum of this metric being at 0, more conventional losses can be used to train it by adding $\tau(\hat{S}_1(t))$ and $\tau(\hat{S}_2(t))$. Regarding the classical properties of distances (see Section 3.2), this metric seems to respect triangular inequality under this form according to numerical tests we performed but not identity. Respecting identity can be achieved by subtracting the mutual independence of the sequences to themselves but triangular inequality is violated in this case. Finally, soft-ranking functions [34, 202] can constitute an alternative to attention to smoothly compute mutual neighbors.

These perspectives end the work presented in this thesis. We hope with the CGRU architecture to have open a framework from which a better understanding of sequence metric learning will be acquired and new neural network models based on synchronization will be designed.

LIST OF FIGURES

LIST OF TABLES

BIBLIOGRAPHY

[1] Abdallah, Z. S., Gaber, M. M., Srinivasan, B., and Krishnaswamy, S. "Adaptive mobile activity recognition system with evolving data streams". In: *Neurocomputing* vol. 150 (2015), pp. 304–317.

[2] Abid, A. and Zou, J. "Autowarp: learning a warping distance from unlabeled time series using sequence autoencoders". In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. 2018, pp. 10568–10578.

[3] Abid, M. and Lefebvre, G. "Improving indoor geomagnetic field fingerprinting using recurrence plot-based convolutional neural networks". In: *Journal of Location Based Services* (2020), pp. 1–27.

[4] Aghabozorgi, S., Shirkhorshidi, A. S., and Wah, T. Y. "Time-series clustering–A decade review". In: *Information Systems* vol. 53 (2015), pp. 16–38.

[5] Ancoli-Israel, S., Cole, R., Alessi, C., Chambers, M., Moorcroft, W., and Pollak, C. P. "The role of actigraphy in the study of sleep and circadian rhythms". In: *Sleep* vol. 26, no. 3 (2003), pp. 342–392.

[6] Anguita, D., Ghio, A., Oneto, L., Parra, X., and Reyes-Ortiz, J. L. "Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine". In: *International workshop on ambient assisted living*. Springer. 2012, pp. 216–223.

[7] Anguita, D., Ghio, A., Oneto, L., Parra, X., and Reyes-Ortiz, J. L. "A Public Domain Dataset for Human Activity Recognition using Smartphones." In: *ESANN*. 2013.

[8] Attal, F., Mohammed, S., Dedabrishvili, M., Chamroukhi, F., Oukhellou, L., and Amirat, Y. "Physical human activity recognition using wearable sensors". In: *Sensors* vol. 15, no. 12 (2015), pp. 31314–31338.

[9] Avci, A., Bosch, S., Marin-Perianu, M., Marin-Perianu, R., and Havinga, P. "Activity recognition using inertial sensing for healthcare, wellbeing and sports applications: A survey". In: *Architecture of computing systems (ARCS), 2010 23rd international conference on*. VDE. 2010, pp. 1–10.

[10] Balntas, V., Riba, E., Ponsa, D., and Mikolajczyk, K. "Learning local feature descriptors with triplets and shallow convolutional neural networks." In: *BMVC*. Vol. 1. 2. 2016, p. 3.

[11] Bandt, C. and Pompe, B. "Permutation entropy: a natural complexity measure for time series". In: *Physical review letters* vol. 88, no. 17 (2002), p. 174102.

[12] Banerjee, A., Merugu, S., Dhillon, I. S., and Ghosh, J. "Clustering with Bregman divergences". In: *Journal of machine learning research* vol. 6, no. Oct (2005), pp. 1705–1749.

[13] Banovic, N., Buzali, T., Chevalier, F., Mankoff, J., and Dey, A. K. "Modeling and understanding human routine behavior". In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM. 2016, pp. 248–260.

[14] Bao, L. and Intille, S. S. "Activity recognition from user-annotated acceleration data". In: *International Conference on Pervasive Computing*. Springer. 2004, pp. 1–17.

[15] Bar-Hillel, A., Hertz, T., Shental, N., and Weinshall, D. "Learning a mahalanobis metric from equivalence constraints". In: *Journal of Machine Learning Research* vol. 6, no. Jun (2005), pp. 937–965.

[16] Baydin, A. G., Pearlmutter, B. A., Radul, A. A., and Siskind, J. M. "Automatic differentiation in machine learning: a survey". In: *The Journal of Machine Learning Research* vol. 18, no. 1 (2017), pp. 5595–5637.

[17] Belkin, M. and Niyogi, P. "Laplacian eigenmaps for dimensionality reduction and data representation". In: *Neural computation* vol. 15, no. 6 (2003), pp. 1373–1396.

[18] Bellet, A., Habrard, A., and Sebban, M. "Metric learning". In: *Synthesis Lectures on Artificial Intelligence and Machine Learning* vol. 9, no. 1 (2015), pp. 1–151.

[19] Bender, E. M., Gebru, T., McMillan-Major, A., and Shmitchell, S. "On the dangers of stochastic parrots: Can language models be too big". In: *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency; Association for Computing Machinery: New York, NY, USA*. 2021.

[20] Bengio, Y., Courville, A., and Vincent, P. "Representation learning: A review and new perspectives". In: *IEEE transactions on pattern analysis and machine intelligence* vol. 35, no. 8 (2013), pp. 1798–1828.

[21] Bengio, Y., Simard, P., and Frasconi, P. "Learning long-term dependencies with gradient descent is difficult". In: *IEEE transactions on neural networks* vol. 5, no. 2 (1994), pp. 157–166.

[22] Bertinetto, L., Valmadre, J., Henriques, J. F., Vedaldi, A., and Torr, P. H. "Fully-convolutional siamese networks for object tracking". In: *European conference on computer vision*. Springer. 2016, pp. 850–865.

[23] Blei, D. M., Ng, A. Y., and Jordan, M. I. "Latent dirichlet allocation". In: *the Journal of machine Learning research* vol. 3 (2003), pp. 993–1022.

[24] Boccaletti, S., Kurths, J., Osipov, G., Valladares, D., and Zhou, C. "The synchronization of chaotic systems". In: *Physics reports* vol. 366, no. 1-2 (2002), pp. 1–101.

[25] Bogert, B. P. "The quefrency alanysis of time series for echoes; Cepstrum, pseudo-autocovariance, cross-cepstrum and saphe cracking". In: *Time series analysis* (1963), pp. 209–243.

[26] Bohr, H. "Zur theorie der fastperiodischen funktionen". In: *Acta Mathematica* vol. 46, no. 1-2 (1925), pp. 101–214.

[27] Boult, C., Kane, R. L., Louis, T. A., Boult, L., and McCaffrey, D. "Chronic conditions that lead to functional limitation in the elderly". In: *Journal of gerontology* vol. 49, no. 1 (1994), pp. M28–M36.

[28] Box, G. E., Jenkins, G. M., Reinsel, G. C., and Ljung, G. M. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

[29] Boyd, S. and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

[30] Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., and Shah, R. "Signature verification using a "siamese" time delay neural network". In: *Advances in Neural Information Processing Systems*. 1994, pp. 737–744.

[31] Brown, R. and Kocarev, L. "A unifying definition of synchronization for dynamical systems". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* vol. 10, no. 2 (2000), pp. 344–349.

[32] Brown, R. and Rulkov, N. F. "Designing a coupling that guarantees synchronization between identical chaotic systems". In: *Physical review letters* vol. 78, no. 22 (1997), p. 4189.

[33] Bruijn, S. M., Bregman, D. J., Meijer, O. G., Beek, P. J., and Dieën, J. H. van. "Maximum Lyapunov exponents as predictors of global gait stability: a modelling approach". In: *Medical engineering & physics* vol. 34, no. 4 (2012), pp. 428–436.

[34] Burges, C. J., Ragno, R., and Le, Q. V. "Learning to rank with nonsmooth cost functions". In: *Advances in neural information processing systems*. 2007, pp. 193–200.

[35] Burgess, C., Livesay, K., and Lund, K. "Explorations in context space: Words, sentences, discourse". In: *Discourse Processes* vol. 25, no. 2-3 (1998), pp. 211–257.

[36] Butler, R. N. "Ageism: A foreword". In: *Journal of social issues* vol. 36, no. 2 (1980), pp. 8–11.

[37] Cai, X., Xu, T., Yi, J., Huang, J., and Rajasekaran, S. "DTWNet: a Dynamic Time Warping Network". In: *Advances in Neural Information Processing Systems*. 2019, pp. 11636–11646.

[38] Caruana, R. "Multitask learning". In: *Machine learning* vol. 28, no. 1 (1997), pp. 41–75.

[39] Casale, P., Pujol, O., and Radeva, P. "Human activity recognition from accelerometer data using a wearable device". In: *Iberian Conference on Pattern Recognition and Image Analysis*. Springer. 2011, pp. 289–296.

[40] Chang, B., Chen, M., Haber, E., and Chi, H. "AntisymmetricRNN: A Dynamical System View on Recurrent Neural Networks". In: *International Conference on Learning Representations*. 2018.

[41] Chatzaki, C., Pediaditis, M., Vavoulas, G., and Tsiknakis, M. "Human daily activity and fall recognition using a smartphone's acceleration sensor". In: *International Conference on Information and Communication Technologies for Ageing Well and e-Health*. Springer. 2016, pp. 100–118.

[42] Che, Z., He, X., Xu, K., and Liu, Y. "DECADE: a deep metric learning model for multivariate time series". In: *KDD workshop on mining and learning from time series*. sn. 2017.

[43] Chechik, G., Sharma, V., Shalit, U., and Bengio, S. "Large Scale Online Learning of Image Similarity Through Ranking." In: *Journal of Machine Learning Research* vol. 11, no. 3 (2010).

[44] Chen, H., Tang, F., Tino, P., Cohn, A. G., and Yao, X. "Model metric co-learning for time series classification". In: *Twenty-Fourth International Joint Conference on Artificial Intelligence*. 2015.

[45] Chen, Z., Zhu, Q., Soh, Y. C., and Zhang, L. "Robust human activity recognition using smartphone sensors via CT-PCA and online SVM". In: *IEEE Transactions on Industrial Informatics* vol. 13, no. 6 (2017), pp. 3070–3080.

[46] Cho, K., Merrienboer, B. van, Gülçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation". In: *EMNLP*. 2014.

[47] Chopra, S., Hadsell, R., and LeCun, Y. "Learning a similarity metric discriminatively, with application to face verification". In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 1. IEEE. 2005, pp. 539–546.

[48] Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. "Empirical evaluation of gated recurrent neural networks on sequence modeling". In: *NIPS 2014 Workshop on Deep Learning, December 2014*. 2014.

[49] Compagnon, P. and Lefebvre, G. "Procédé d'alerte sur l'autonomie d'une personne par l'analyse de ses changements de routines". 2019.

[50] Compagnon, P., Lefebvre, G., Duffner, S., and Garcia, C. "Learning personalized ADL recognition models from few raw data". In: *Artificial Intelligence in Medicine* vol. 107 (2020), p. 101916.

[51] Compagnon, P., Lefebvre, G., Duffner, S., and Garcia, C. "Personalized posture and fall classification with shallow gated recurrent units". In: *2019 IEEE 32nd International Symposium on Computer-Based Medical Systems (CBMS)*. IEEE. 2019, pp. 114–119.

[52] Compagnon, P., Lefebvre, G., Duffner, S., and Garcia, C. "Routine modeling with time series metric learning". In: *International Conference on Artificial Neural Networks*. Springer. 2019, pp. 579–592.

[53] Compagnon, P., Lefebvre, G., Duffner, S., and Garcia, C. "Sequence Metric Learning as Synchronization of Recurrent Neural Networks". In: *Submitted to IJCNN* (2021).

[54] Cortes, C. and Vapnik, V. “Support-vector networks”. In: *Machine learning* vol. 20, no. 3 (1995), pp. 273–297.

[55] Couturier, P. “Place de l’actimétrie dans la gestion médicale du sujet âgé fragile”. In: *Gérontologie et société* vol. 28, no. 2 (2005), pp. 13–23.

[56] Cumin, J., Lefebvre, G., Ramparany, F., and Crowley, J. L. “Human activity recognition using place-based decision fusion in smart homes”. In: *International and Interdisciplinary Conference on Modeling and Using Context*. Springer. 2017, pp. 137–150.

[57] Cuturi, M. and Avis, D. “Ground metric learning”. In: *The Journal of Machine Learning Research* vol. 15, no. 1 (2014), pp. 533–564.

[58] Cuturi, M. and Blondel, M. “Soft-dtw: a differentiable loss function for time-series”. In: *International Conference on Machine Learning*. PMLR. 2017, pp. 894–903.

[59] Davis, J. V., Kulis, B., Jain, P., Sra, S., and Dhillon, I. S. “Information-theoretic metric learning”. In: *Proceedings of the 24th international conference on Machine learning*. ACM. 2007, pp. 209–216.

[60] De Angeli, A., Genesio, R., and Tesi, A. “Dead-beat chaos synchronization in discrete-time systems”. In: *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* vol. 42, no. 1 (1995), pp. 54–56.

[61] De Cock, K. and De Moor, B. “Subspace angles between ARMA models”. In: *Systems & Control Letters* vol. 46, no. 4 (2002), pp. 265–270.

[62] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[63] Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., and Keogh, E. “Querying and mining of time series data: experimental comparison of representations and distance measures”. In: *Proceedings of the VLDB Endowment* vol. 1, no. 2 (2008), pp. 1542–1552.

[64] Dingwell, J. B. and Cusumano, J. P. “Nonlinear time series analysis of normal and pathological human walking”. In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* vol. 10, no. 4 (2000), pp. 848–863.

[65] DiPietro, R., Rupprecht, C., Navab, N., and Hager, G. D. "Analyzing and exploiting NARX recurrent neural networks for long-term dependencies". In: *6th International Conference on Learning Representations, ICLR 2018*. 2018.

[66] Doya, K. "Bifurcations of recurrent neural networks in gradient descent learning". In: *IEEE Transactions on neural networks* vol. 1 (1993), pp. 75–80.

[67] Esling, P. and Agon, C. "Time-series data mining". In: *ACM Computing Surveys (CSUR)* vol. 45, no. 1 (2012), p. 12.

[68] Eysenbach, G. "What is e-health?" In: *Journal of medical Internet research* vol. 3, no. 2 (2001).

[69] Faraki, M., Harandi, M. T., and Porikli, F. "Large-Scale Metric Learning: A Voyage From Shallow to Deep". In: *IEEE transactions on neural networks and learning systems* vol. 29, no. 9 (2018), pp. 4339–4346.

[70] Fujisaka, H. and Yamada, T. "Stability theory of synchronized motion in coupled-oscillator systems". In: *Progress of theoretical physics* vol. 69, no. 1 (1983), pp. 32–47.

[71] Fukunaga, K. and Narendra, P. M. "A branch and bound algorithm for computing k-nearest neighbors". In: *IEEE transactions on computers* vol. 100, no. 7 (1975), pp. 750–753.

[72] Fukushima, K. and Miyake, S. "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition". In: *Competition and cooperation in neural nets*. Springer, 1982, pp. 267–285.

[73] Funahashi, K.-i. and Nakamura, Y. "Approximation of dynamical systems by continuous time recurrent neural networks". In: *Neural networks* vol. 6, no. 6 (1993), pp. 801–806.

[74] Gao, X., Xiao, B., Tao, D., and Li, X. "A survey of graph edit distance". In: *Pattern Analysis and applications* vol. 13, no. 1 (2010), pp. 113–129.

[75] Gardner, B. "A review and analysis of the use of habit in understanding, predicting and influencing health-related behaviour". In: *Health psychology review* vol. 9, no. 3 (2015), pp. 277–295.

[76] Gardner, B., Abraham, C., Lally, P., and Bruijn, G.-J. de. "Towards parsimony in habit measurement: Testing the convergent and predictive validity of an automaticity subscale of the Self-Report Habit Index". In: *International Journal of Behavioral Nutrition and Physical Activity* vol. 9, no. 1 (2012), pp. 1–12.

[77] Garreau, D., Lajugie, R., Arlot, S., and Bach, F. "Metric learning for temporal sequence alignment". In: *Advances in neural information processing systems*. 2014, pp. 1817–1825.

[78] Ghahramani, Z. and Hinton, G. E. *Parameter estimation for linear dynamical systems*. Tech. rep. Technical Report CRG-TR-96-2, University of Totronto, Dept. of Computer Science, 1996.

[79] Gjoreski, H., Ciliberto, M., Wang, L., Morales, F. J. O., Mekki, S., Valentin, S., and Roggen, D. "The University of Sussex-Huawei locomotion and transportation dataset for multimodal analytics with mobile devices". In: *IEEE Access* vol. 6 (2018), pp. 42592–42604.

[80] Gjoreski, M., Janko, V., Rei, N., Mlakar, M., Lutrek, M., Bizjak, J., Slapniar, G., Marinko, M., Drobni, V., and Gams, M. "Applying multiple knowledge to Sussex-Huawei locomotion challenge". In: *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*. 2018, pp. 1488–1496.

[81] Globerson, A. and Roweis, S. T. "Metric learning by collapsing classes". In: *Advances in neural information processing systems*. 2006, pp. 451–458.

[82] Glorot, X., Bordes, A., and Bengio, Y. "Deep sparse rectifier neural networks". In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. 2011, pp. 315–323.

[83] Gonzalez, M. C., Hidalgo, C. A., and Barabasi, A.-L. "Understanding individual human mobility patterns". In: *nature* vol. 453, no. 7196 (2008), p. 779.

[84] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. "Generative adversarial nets". In: *Advances in neural information processing systems*. 2014, pp. 2672–2680.

[85] Guan, Y. and Plötz, T. "Ensembles of deep lstm learners for activity recognition using wearables". In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* vol. 1, no. 2 (2017), pp. 1–28.

[86] Hadsell, R., Chopra, S., and LeCun, Y. "Dimensionality reduction by learning an invariant mapping". In: *Computer vision and pattern recognition, 2006 IEEE computer society conference on*. Vol. 2. IEEE. 2006, pp. 1735–1742.

[87] Haugeland, J. *Artificial intelligence: The very idea*. MIT press, 1989.

[88] He, K., Zhang, X., Ren, S., and Sun, J. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[89] He, Z. and Jin, L. "Activity recognition from acceleration data based on discrete consine transform and SVM". In: *2009 IEEE International Conference on Systems, Man and Cybernetics*. IEEE. 2009, pp. 5041–5044.

[90] Hecht-Nielsen, R. "Theory of the backpropagation neural network". In: *Neural networks for perception*. Elsevier, 1992, pp. 65–93.

[91] Hestenes, M. R., Stiefel, E., et al. "Methods of conjugate gradients for solving linear systems". In: *Journal of research of the National Bureau of Standards* vol. 49, no. 6 (1952), pp. 409–436.

[92] Higham, N. J. "Computing a nearest symmetric positive semidefinite matrix". In: *Linear algebra and its applications* vol. 103 (1988), pp. 103–118.

[93] Ho, T. K. "Random decision forests". In: *Proceedings of 3rd international conference on document analysis and recognition*. Vol. 1. IEEE. 1995, pp. 278–282.

[94] Hochreiter, S., Bengio, Y., Frasconi, P., Schmidhuber, J., et al. *Gradient flow in recurrent nets: the difficulty of learning long-term dependencies*. 2001.

[95] Hochreiter, S. and Schmidhuber, J. "Long short-term memory". In: *Neural computation* vol. 9, no. 8 (1997), pp. 1735–1780.

[96] Hoffer, E. and Ailon, N. "Deep metric learning using triplet network". In: *International Workshop on Similarity-Based Pattern Recognition*. Springer. 2015, pp. 84–92.

[97] Hofmann, T. “Probabilistic latent semantic indexing”. In: *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. 1999, pp. 50–57.

[98] Holan, S. H., Lund, R., Davis, G., et al. “The ARMA alphabet soup: A tour of ARMA model variants”. In: *Statistics Surveys* vol. 4 (2010), pp. 232–274.

[99] Hornik, K., Stinchcombe, M., White, H., et al. “Multilayer feedforward networks are universal approximators.” In: *Neural networks* vol. 2, no. 5 (1989), pp. 359–366.

[100] Hosmer Jr, D. W., Lemeshow, S., and Sturdivant, R. X. *Applied logistic regression*. Vol. 398. John Wiley & Sons, 2013.

[101] Huynh, T., Fritz, M., and Schiele, B. “Discovery of activity patterns using topic models”. In: *Proceedings of the 10th international conference on Ubiquitous computing*. ACM. 2008, pp. 10–19.

[102] Ishikawa, I., Fujii, K., Ikeda, M., Hashimoto, Y., and Kawahara, Y. “Metric on nonlinear dynamical systems with perron-frobenius operators”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 2856–2866.

[103] Jaeger, H. “The "echo state" approach to analysing and training recurrent neural networks-with an erratum note”. In: *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report* vol. 148, no. 34 (2001), p. 13.

[104] Janko, V., Reçiç, N., Mlakar, M., Drobni, V., Gams, M., Slapniar, G., Gjoreski, M., Bizjak, J., Marinko, M., and Lutrek, M. “A New Frontier for Activity Recognition: The Sussex-Huawei Locomotion Challenge”. In: *Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers*. 2018, pp. 1511–1520.

[105] Jia, Y., Nie, F., and Zhang, C. “Trace ratio problem revisited”. In: *IEEE Transactions on Neural Networks* vol. 20, no. 4 (2009), pp. 729–735.

[106] Jiang, W. and Yin, Z. “Human activity recognition using wearable sensors by deep convolutional neural networks”. In: *Proceedings of the 23rd ACM international conference on Multimedia*. 2015, pp. 1307–1310.

[107] Kalpakis, K., Gada, D., and Puttagunta, V. "Distance measures for effective clustering of ARIMA time-series". In: *Proceedings 2001 IEEE international conference on data mining*. IEEE. 2001, pp. 273–280.

[108] Katz, S., Ford, A. B., Moskowitz, R. W., Jackson, B. A., and Jaffe, M. W. "Studies of illness in the aged: the index of ADL: a standardized measure of biological and psychosocial function". In: *Jama* vol. 185, no. 12 (1963), pp. 914–919.

[109] Kim, E., Helal, S., and Cook, D. "Human activity recognition and pattern discovery". In: *IEEE Pervasive Computing* vol. 9, no. 1 (2010).

[110] Kingma, D. P. and Welling, M. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[111] Kocarev, L. and Parlitz, U. "Generalized synchronization, predictability, and equivalence of unidirectionally coupled dynamical systems". In: *Physical review letters* vol. 76, no. 11 (1996), p. 1816.

[112] Koestinger, M., Hirzer, M., Wohlhart, P., Roth, P. M., and Bischof, H. "Large scale metric learning from equivalence constraints". In: *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE. 2012, pp. 2288–2295.

[113] Kramer, M. A. "Nonlinear principal component analysis using autoassociative neural networks". In: *AIChE journal* vol. 37, no. 2 (1991), pp. 233–243.

[114] Kreuz, T., Mormann, F., Andrzejak, R. G., Kraskov, A., Lehnertz, K., and Grassberger, P. "Measuring synchronization in coupled model systems: A comparison of different approaches". In: *Physica D: Nonlinear Phenomena* vol. 225, no. 1 (2007), pp. 29–42.

[115] Krizhevsky, A., Sutskever, I., and Hinton, G. E. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[116] Krogh, A. and Hertz, J. A. "A simple weight decay can improve generalization". In: *Advances in neural information processing systems*. 1992, pp. 950–957.

[117] Kulis, B., Sustik, M., and Dhillon, I. "Learning low-rank kernel matrices". In: *Proceedings of the 23rd international conference on Machine learning*. 2006, pp. 505–512.

[118] Lally, P., Van Jaarsveld, C. H., Potts, H. W., and Wardle, J. "How are habits formed: Modelling habit formation in the real world". In: *European journal of social psychology* vol. 40, no. 6 (2010), pp. 998–1009.

[119] Laporte, R. E., Montoye, H. J., and Caspersen, C. J. "Assessment of physical activity in epidemiologic research: problems and prospects." In: *Public health reports* vol. 100, no. 2 (1985), p. 131.

[120] Lara, O. D. and Labrador, M. A. "A survey on human activity recognition using wearable sensors." In: *IEEE Communications Surveys and Tutorials* vol. 15, no. 3 (2013), pp. 1192–1209.

[121] Laurent, T. and Brecht, J. von. "A recurrent neural network without chaos". In: *arXiv preprint arXiv:1612.06212* (2016).

[122] Lawton, M. P. and Brody, E. M. "Assessment of older people: self-maintaining and instrumental activities of daily living". In: *The gerontologist* vol. 9, no. 3_Part_1 (1969), pp. 179–186.

[123] LeCun, Y., Bengio, Y., and Hinton, G. "Deep learning". In: *nature* vol. 521, no. 7553 (2015), p. 436.

[124] LeCun, Y., Chopra, S., Hadsell, R., Ranzato, M., and Huang, F. "A tutorial on energy-based learning". In: *Predicting structured data* vol. 1, no. 0 (2006).

[125] Lefebvre, G., Berlemont, S., Mamalet, F., and Garcia, C. "BLSTM-RNN based 3D gesture classification". In: *International Conference on Artificial Neural Networks*. Springer. 2013, pp. 381–388.

[126] Lefebvre, G. and Compagnon, P. "Procédé d'évaluation de l'activité corporelle d'un utilisateur". 2019.

[127] Lefebvre, G. and Compagnon, P. "Reconnaissance de modes de déplacement par capteurs de mouvement". 2020.

[128] Lefebvre, G. and Garcia, C. "Learning a bag of features based non-linear metric for facial similarity". In: *Advanced Video and Signal Based Surveillance (AVSS), 2013 10th IEEE International Conference on*. IEEE. 2013, pp. 238–243.

[129] Liang, F., Das, V., Kostyuk, N., and Hussain, M. M. "Constructing a data-driven society: China's social credit system as a state surveillance infrastructure". In: *Policy & Internet* vol. 10, no. 4 (2018), pp. 415–453.

[130] Liao, T. W. "Clustering of time series data a survey". In: *Pattern recognition* vol. 38, no. 11 (2005), pp. 1857–1874.

[131] Lin, J. and Li, Y. "Finding structural similarity in time series data using bag-of-patterns representation". In: *International conference on scientific and statistical database management*. Springer. 2009, pp. 461–477.

[132] Lin, X., Lu, R., Shen, X., Nemoto, Y., and Kato, N. "SAGE: a strong privacy-preserving scheme against global eavesdropping for ehealth systems". In: *IEEE Journal on Selected Areas in Communications* vol. 27, no. 4 (2009), pp. 365–378.

[133] Longstaff, B., Reddy, S., and Estrin, D. "Improving activity classification for health applications on mobile devices using active and semi-supervised learning". In: *2010 4th International Conference on Pervasive Computing Technologies for Healthcare*. IEEE. 2010, pp. 1–7.

[134] Lyapunov, A. M. "The general problem of the stability of motion". In: *International journal of control* vol. 55, no. 3 (1992), pp. 531–534.

[135] Mahalanobis, P. C. "On the generalized distance in statistics". In: National Institute of Science of India. 1936.

[136] Makni, A. and Lefebvre, G. "Attitude Estimation for Posture Detection in eHealth Services". In: *2018 IEEE 31st International Symposium on Computer-Based Medical Systems (CBMS)*. IEEE. 2018, pp. 310–315.

[137] Martin, R. J. "A metric for ARMA processes". In: *IEEE transactions on Signal Processing* vol. 48, no. 4 (2000), pp. 1164–1170.

[138] Mathie, M., Coster, A., Lovell, N., and Celler, B. "Detection of daily physical activities using a triaxial accelerometer". In: *Medical and Biological Engineering and Computing* vol. 41, no. 3 (2003), pp. 296–301.

[139] Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., and Galstyan, A. "A survey on bias and fairness in machine learning". In: *arXiv preprint arXiv:1908.09635* (2019).

[140] Melkas, H., Hennala, L., Pekkarinen, S., and Kyrki, V. "Impacts of robot implementation on care personnel and clients in elderly-care institutions". In: *International Journal of Medical Informatics* vol. 134 (2020), p. 104041.

[141] Minsky, M. L. "Logical versus analogical or symbolic versus connectionist or neat versus scruffy". In: *AI magazine* vol. 12, no. 2 (1991), pp. 34–34.

[142] Mourcou, Q., Fleury, A., Franco, C., Klopcic, F., and Vuillerme, N. "Performance evaluation of smartphone inertial sensors measurement for range of motion". In: *Sensors* vol. 15, no. 9 (2015), pp. 23168–23187.

[143] Mueller, J. and Thyagarajan, A. "Siamese recurrent architectures for learning sentence similarity". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 30. 1. 2016.

[144] Murad, A. and Pyun, J.-Y. "Deep recurrent neural networks for human activity recognition". In: *Sensors* vol. 17, no. 11 (2017), p. 2556.

[145] Najafi, B., Aminian, K., Paraschiv-Ionescu, A., Loew, F., Bula, C. J., and Robert, P. "Ambulatory system for human motion analysis using a kinematic sensor: monitoring of daily physical activity in the elderly". In: *IEEE Transactions on biomedical Engineering* vol. 50, no. 6 (2003), pp. 711–723.

[146] Nazerfard, E. and Cook, D. J. "Using Bayesian Networks for Daily Activity Prediction." In: *AAAI workshop: plan, activity, and intent recognition*. 2013.

[147] Ng, A. Y., Jordan, M. I., Weiss, Y., et al. "On spectral clustering: Analysis and an algorithm". In: *Advances in neural information processing systems* vol. 2 (2002), pp. 849–856.

[148] Nguyen, H. V. and Bai, L. "Cosine similarity metric learning for face verification". In: *Asian conference on computer vision*. Springer. 2010, pp. 709–720.

[149] Nguyen, L. T., Zeng, M., Tague, P., and Zhang, J. "Recognizing new activities with limited training data". In: *Proceedings of the 2015 ACM International Symposium on Wearable Computers*. ACM. 2015, pp. 67–74.

[150] Nicolae, M.-I., Gaussier, É., Habrard, A., and Sebban, M. "Similarity learning for time series classification". In: *arXiv preprint arXiv:1610.04783* (2016).

[151] Nweke, H. F., Teh, Y. W., Al-Garadi, M. A., and Alo, U. R. "Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges". In: *Expert Systems with Applications* vol. 105 (2018), pp. 233–261.

[152] Oh Song, H., Jegelka, S., Rathod, V., and Murphy, K. "Deep metric learning via facility location". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 5382–5390.

[153] Oh Song, H., Xiang, Y., Jegelka, S., and Savarese, S. "Deep metric learning via lifted structured feature embedding". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 4004–4012.

[154] Oh, H., Rizo, C., Enkin, M., and Jadad, A. "What is eHealth?: a systematic review of published definitions". In: *World Hosp Health Serv* vol. 41, no. 1 (2005), pp. 32–40.

[155] Pan, S. J. and Yang, Q. "A survey on transfer learning". In: *IEEE Transactions on knowledge and data engineering* vol. 22, no. 10 (2009), pp. 1345–1359.

[156] Pascanu, R., Mikolov, T., and Bengio, Y. "On the difficulty of training recurrent neural networks". In: *International Conference on Machine Learning*. 2013, pp. 1310–1318.

[157] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. "Pytorch: An imperative style, high-performance deep learning library". In: *Advances in neural information processing systems*. 2019, pp. 8026–8037.

[158] Pathak, J., Lu, Z., Hunt, B. R., Girvan, M., and Ott, E. "Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* vol. 27, no. 12 (2017), p. 121102.

[159] Pecora, L. M. and Carroll, T. L. "Synchronization in chaotic systems". In: *Physical review letters* vol. 64, no. 8 (1990), p. 821.

[160] Peng, L., Chen, L., Ye, Z., and Zhang, Y. "Aroma: A deep multi-task learning based simple and complex human activity recognition method using wearable sensors". In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* vol. 2, no. 2 (2018), pp. 1–16.

[161] Perrot, M. and Habrard, A. "Regressive virtual metric learning". In: *Advances in neural information processing systems*. 2015, pp. 1810–1818.

[162] Pigot, H., Lefebvre, B., Meunier, J.-G., Kerhervé, B., Mayers, A., and Giroux, S. "The role of intelligent habitats in upholding elders in residence". In: *5th international conference on Simulations in Biomedicine*. 2003, pp. 497–506.

[163] Qin, T., Shangguan, W., Song, G., and Tang, J. "Spatio-temporal routine mining on mobile phone data". In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* vol. 12, no. 5 (2018), pp. 1–24.

[164] Quach, K. A. "Extraction de caracteristiques de l'activite ambulatoire du patient par fusion d'informations de centrales inertielles". PhD thesis. Universite Claude Bernard Lyon 1, 2012.

[165] Ravi, N., Dandekar, N., Mysore, P., and Littman, M. L. "Activity recognition from accelerometer data". In: *Aaai*. Vol. 5. 2005. 2005, pp. 1541–1546.

[166] Ristad, E. S. and Yianilos, P. N. "Learning string-edit distance". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* vol. 20, no. 5 (1998), pp. 522–532.

[167] Rodan, A. and Tio, P. "Simple deterministically constructed cycle reservoirs with regular jumps". In: *Neural computation* vol. 24, no. 7 (2012), pp. 1822–1852.

[168] Rolnick, D., Donti, P. L., Kaack, L. H., Kochanski, K., Lacoste, A., Sankaran, K., Ross, A. S., Milojevic-Dupont, N., Jaques, N., Waldman-Brown, A., et al. "Tackling climate change with machine learning". In: *arXiv preprint arXiv:1906.05433* (2019).

[169] Rosenberg, A. and Hirschberg, J. "V-measure: A conditional entropy-based external cluster evaluation measure". In: *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*. 2007, pp. 410–420.

[170] Rulkov, N. F., Sushchik, M. M., Tsimring, L. S., and Abarbanel, H. D. "Generalized synchronization of chaos in directionally coupled chaotic systems". In: *Physical Review E* vol. 51, no. 2 (1995), p. 980.

[171] Sakoe, H. and Chiba, S. "Dynamic programming algorithm optimization for spoken word recognition". In: *IEEE transactions on acoustics, speech, and signal processing* vol. 26, no. 1 (1978), pp. 43–49.

[172] Salvador, S. and Chan, P. "Toward accurate dynamic time warping in linear time and space". In: *Intelligent Data Analysis* vol. 11, no. 5 (2007), pp. 561–580.

[173] San-Segundo, R., Montero, J. M., Moreno-Pimentel, J., and Pardo, J. M. "HMM adaptation for improving a human activity recognition system". In: *Algorithms* vol. 9, no. 3 (2016), p. 60.

[174] Sani, S., Wiratunga, N., Massie, S., and Cooper, K. "Personalised human activity recognition using matching networks". In: *International Conference on Case-Based Reasoning*. Springer. 2018, pp. 339–353.

[175] Schiff, S. J., So, P., Chang, T., Burke, R. E., and Sauer, T. "Detecting dynamical interdependence and generalized synchrony through mutual prediction in a neural ensemble". In: *Physical Review E* vol. 54, no. 6 (1996), p. 6708.

[176] Schroff, F., Kalenichenko, D., and Philbin, J. "Facenet: A unified embedding for face recognition and clustering". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 815–823.

[177] Schuster, M. and Paliwal, K. K. "Bidirectional recurrent neural networks". In: *IEEE Transactions on Signal Processing* vol. 45, no. 11 (1997), pp. 2673–2681.

[178] Settles, B. *Active learning literature survey*. Tech. rep. University of Wisconsin-Madison Department of Computer Sciences, 2009.

[179] Settles, B. and Craven, M. "An analysis of active learning strategies for sequence labeling tasks". In: *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*. 2008, pp. 1070–1079.

[180] Shaikh, F. K., Zeadally, S., and Exposito, E. "Enabling technologies for green internet of things". In: *IEEE Systems Journal* vol. 11, no. 2 (2015), pp. 983–994.

[181] Simonyan, K. and Zisserman, A. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[182] Snell, J., Swersky, K., and Zemel, R. "Prototypical networks for few-shot learning". In: *Advances in Neural Information Processing Systems*. 2017, pp. 4077–4087.

[183] Sohn, K. “Improved deep metric learning with multi-class n-pair loss objective”. In: *Advances in Neural Information Processing Systems*. 2016, pp. 1857–1865.

[184] Sousa Lima, W., Souto, E., El-Khatib, K., Jalali, R., and Gama, J. “Human activity recognition using inertial sensors in a smartphone: An overview”. In: *Sensors* vol. 19, no. 14 (2019), p. 3213.

[185] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. “Dropout: A simple way to prevent neural networks from overfitting”. In: *The Journal of Machine Learning Research* vol. 15, no. 1 (2014), pp. 1929–1958.

[186] Staniek, M. and Lehnertz, K. “Symbolic transfer entropy”. In: *Physical Review Letters* vol. 100, no. 15 (2008), p. 158101.

[187] Stikic, M., Van Laerhoven, K., and Schiele, B. “Exploring semi-supervised and active learning for activity recognition”. In: *2008 12th IEEE International Symposium on Wearable Computers*. IEEE. 2008, pp. 81–88.

[188] Strogatz, S. H. “Nonlinear Dynamics and Chaos with Student Solutions Manual: With Applications to Physics, Biology”. In: *Chemistry, and Engineering. CRC Press* (2018).

[189] Strubell, E., Ganesh, A., and McCallum, A. “Energy and Policy Considerations for Deep Learning in NLP”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2019, pp. 3645–3650.

[190] Su, B. and Hua, G. “Order-preserving wasserstein distance for sequence matching”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 1049–1057.

[191] Su, B. and Wu, Y. “Learning Distance for Sequences by Learning a Ground Metric”. In: *International Conference on Machine Learning*. 2019, pp. 6015–6025.

[192] Sun, J., Sow, D., Hu, J., and Ebadollahi, S. “Localized supervised metric learning on temporal physiological data”. In: *2010 20th International Conference on Pattern Recognition*. IEEE. 2010, pp. 4149–4152.

[193] Sun, X., Kashima, H., Tomioka, R., Ueda, N., and Li, P. “A new multi-task learning method for personalized activity recognition”. In: *2011 IEEE 11th International Conference on Data Mining*. IEEE. 2011, pp. 1218–1223.

[194] Sun, X., Kashima, H., and Ueda, N. "Large-scale personalized human activity recognition using online multitask learning". In: *IEEE Transactions on Knowledge and Data Engineering* vol. 25, no. 11 (2012), pp. 2551–2563.

[195] Sutskever, I. "Training recurrent neural networks". In: *University of Toronto, Toronto, Ont., Canada* (2013).

[196] Sutskever, I., Martens, J., Dahl, G., and Hinton, G. "On the Importance of Initialization and Momentum in Deep Learning". In: *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*. ICML'13. Atlanta, GA, USA: JMLR.org, 2013.

[197] Sutskever, I., Vinyals, O., and Le, Q. V. "Sequence to sequence learning with neural networks". In: *Advances in neural information processing systems*. 2014, pp. 3104–3112.

[198] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. "Going deeper with convolutions". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9.

[199] Tajbakhsh, N., Shin, J. Y., Gurudu, S. R., Hurst, R. T., Kendall, C. B., Gotway, M. B., and Liang, J. "Convolutional neural networks for medical image analysis: Full training or fine tuning?" In: *IEEE transactions on medical imaging* vol. 35, no. 5 (2016), pp. 1299–1312.

[200] Talukder, M. S., Sorwar, G., Bao, Y., Ahmed, J. U., and Palash, M. A. S. "Predicting antecedents of wearable healthcare technology acceptance by elderly: A combined SEM-Neural Network approach". In: *Technological Forecasting and Social Change* vol. 150 (2020), p. 119793.

[201] Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., and Liu, C. "A survey on deep transfer learning". In: *International conference on artificial neural networks*. Springer. 2018, pp. 270–279.

[202] Taylor, M., Guiver, J., Robertson, S., and Minka, T. "Softrank: optimizing non-smooth rank metrics". In: *Proceedings of the 2008 International Conference on Web Search and Data Mining*. 2008, pp. 77–86.

[203] Thome, N., Miguet, S., and Ambellouis, S. "A real-time, multiview fall detection system: A LHMM-based approach". In: *IEEE transactions on circuits and systems for video technology* vol. 18, no. 11 (2008), pp. 1522–1532.

[204] Tino, P. "Dynamical Systems as Temporal Feature Spaces." In: *Journal of Machine Learning Research* vol. 21, no. 44 (2020), pp. 1–42.

[205] Tsinganos, P. and Skodras, A. "A smartphone-based fall detection system for the elderly". In: *Proceedings of the 10th International Symposium on Image and Signal Processing and Analysis*. IEEE. 2017, pp. 53–58.

[206] Tuisku, O., Pekkarinen, S., Hennala, L., and Melkas, H. "Robots do not replace a nurse with a beating heart". In: *Information Technology & People* (2019).

[207] Turaga, P., Chellappa, R., Subrahmanian, V. S., and Udrea, O. "Machine recognition of human activities: A survey". In: *IEEE Transactions on Circuits and Systems for Video technology* vol. 18, no. 11 (2008), pp. 1473–1488.

[208] Vaizman, Y., Ellis, K., and Lanckriet, G. "Recognizing detailed human context in the wild from smartphones and smartwatches". In: *IEEE Pervasive Computing* vol. 16, no. 4 (2017), pp. 62–74.

[209] Valiant, L. G. "A theory of the learnable". In: *Communications of the ACM* vol. 27, no. 11 (1984), pp. 1134–1142.

[210] Vandenberghe, L. and Boyd, S. "Semidefinite programming". In: *SIAM review* vol. 38, no. 1 (1996), pp. 49–95.

[211] Varior, R. R., Haloi, M., and Wang, G. "Gated siamese convolutional neural network architecture for human re-identification". In: *European conference on computer vision*. Springer. 2016, pp. 791–808.

[212] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser,., and Polosukhin, I. "Attention is all you need". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 2017, pp. 6000–6010.

[213] Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. "Extracting and composing robust features with denoising autoencoders". In: *Proceedings of the 25th international conference on Machine learning*. ACM. 2008, pp. 1096–1103.

[214] Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., and Manzagol, P.-A. "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion". In: *Journal of machine learning research* vol. 11, no. Dec (2010), pp. 3371–3408.

[215] Vinh, N. X., Epps, J., and Bailey, J. "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance". In: *Journal of Machine Learning Research* vol. 11, no. Oct (2010), pp. 2837–2854.

[216] Vinyals, O., Bengio, S., and Kudlur, M. "Order matters: Sequence to sequence for sets". In: *arXiv preprint arXiv:1511.06391* (2015).

[217] Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. "Matching networks for one shot learning". In: *Advances in Neural Information Processing Systems*. 2016, pp. 3630–3638.

[218] Wang, A., Chen, G., Yang, J., Zhao, S., and Chang, C.-Y. "A comparative study on human activity recognition using inertial sensors in a smartphone". In: *IEEE Sensors Journal* vol. 16, no. 11 (2016), pp. 4566–4578.

[219] Wang, J., Chen, Y., Hao, S., Peng, X., and Hu, L. "Deep learning for sensor-based activity recognition: A survey". In: *Pattern Recognition Letters* vol. 119 (2019), pp. 3–11.

[220] Wang, L., Gjoreskia, H., Murao, K., Okita, T., and Roggen, D. "Summary of the sussex-huawei locomotion-transportation recognition challenge". In: *Proceedings of the 2018 ACM international joint conference and 2018 international symposium on pervasive and ubiquitous computing and wearable computers*. 2018, pp. 1521–1530.

[221] Waterman, D. *A guide to expert systems*. Addison-Wesley Pub. Co., Reading, MA, 1986.

[222] Weinberger, K. Q. and Saul, L. K. "Distance metric learning for large margin nearest neighbor classification". In: *Journal of Machine Learning Research* vol. 10, no. Feb (2009), pp. 207–244.

[223] Weiss, A., Brozgol, M., Dorfman, M., Herman, T., Shema, S., Giladi, N., and Hausdorff, J. M. "Does the evaluation of gait quality during daily life provide insight into fall risk? A novel approach using 3-day accelerometer recordings". In: *Neurorehabilitation and neural repair* vol. 27, no. 8 (2013), pp. 742–752.

[224] Werbos, P. J. "Backpropagation through time: what it does and how to do it". In: *Proceedings of the IEEE* vol. 78, no. 10 (1990), pp. 1550–1560.

[225] West, B. J. and Scafetta, N. "Nonlinear dynamical model of human gait". In: *Physical review E* vol. 67, no. 5 (2003), p. 051917.

[226] Williamson, J., Murray-Smith, R., and Hughes, S. "Shoogle: excitatory multimodal interaction on mobile devices". In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM. 2007, pp. 121–124.

[227] Wood, W., Quinn, J. M., and Kashy, D. A. "Habits in everyday life: Thought, emotion, and action." In: *Journal of personality and social psychology* vol. 83, no. 6 (2002), p. 1281.

[228] Wortmann, F. and Flüchter, K. "Internet of things, technology and value added". In: *Business & Information Systems Engineering* vol. 57, no. 3 (2015), pp. 221–224.

[229] Wu, D., Zhu, F., and Shao, L. "One shot learning gesture recognition from RGBD images". In: *Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE. 2012, pp. 7–12.

[230] Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. "Google's neural machine translation system: Bridging the gap between human and machine translation". In: *arXiv preprint arXiv:1609.08144* (2016).

[231] Xi, X., Keogh, E., Shelton, C., Wei, L., and Ratanamahatana, C. A. "Fast time series classification using numerosity reduction". In: *Proceedings of the 23rd international conference on Machine learning*. ACM. 2006, pp. 1033–1040.

[232] Xing, E. P., Jordan, M. I., Russell, S. J., and Ng, A. Y. "Distance metric learning with application to clustering with side-information". In: *Advances in neural information processing systems*. 2003, pp. 521–528.

[233] Xiong, F., Gou, M., Camps, O., and Sznaier, M. "Person reidentification using kernel-based metric learning methods". In: *European conference on computer vision*. Springer. 2014, pp. 1–16.

[234] Xiong, Y. and Lin, H. "Routine based analysis for user classification and location prediction". In: *2012 9th International Conference on Ubiquitous Intelligence and Computing and 9th International Conference on Autonomic and Trusted Computing*. IEEE. 2012, pp. 96–103.

[235] Yang, X., Zhou, P., and Wang, M. "Person reidentification via structural deep metric learning". In: *IEEE Transactions on Neural Networks and Learning Systems* vol. 30, no. 10 (2018), pp. 2987–2998.

[236] Yi, D., Lei, Z., Liao, S., and Li, S. Z. "Deep metric learning for person re-identification". In: *2014 22nd International Conference on Pattern Recognition*. IEEE. 2014, pp. 34–39.

[237] Young, J., OConnell, B., and McGregor, S. "Day surgery patients' convalescence at home: does enhanced discharge education make a difference?" In: *Nursing & Health Sciences* vol. 2, no. 1 (2000), pp. 29–39.

[238] Zafari, F., Gkelias, A., and Leung, K. K. "A survey of indoor localization systems and technologies". In: *IEEE Communications Surveys & Tutorials* vol. 21, no. 3 (2019), pp. 2568–2599.

[239] Zeng, M., Nguyen, L. T., Yu, B., Mengshoel, O. J., Zhu, J., Wu, P., and Zhang, J. "Convolutional neural networks for human activity recognition using mobile sensors". In: *Mobile Computing, Applications and Services (MobiCASE), 2014 6th International Conference on*. IEEE. 2014, pp. 197–205.

[240] Zhao, Y., Yang, R., Chevalier, G., Xu, X., and Zhang, Z. "Deep residual bidir-LSTM for human activity recognition using wearable sensors". In: *Mathematical Problems in Engineering* vol. 2018 (2018).

[241] Zheng, L., Idrissi, K., Garcia, C., Duffner, S., and Baskurt, A. "Triangular similarity metric learning for face verification". In: *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*. Vol. 1. IEEE. 2015, pp. 1–7.

[242] Zhuang, Z. and Xue, Y. "Sport-Related Human Activity Detection and Recognition Using a Smartwatch". In: *Sensors* vol. 19, no. 22 (2019), p. 5001.

FOLIO ADMINISTRATIF

THESE DE L'UNIVERSITE DE LYON OPEREE AU SEIN DE L'INSA LYON

NOM : COMPAGNON DATE de SOUTENANCE 27/05/2021:

Prénoms : Paul, Nicolas

TITRE : Sequence Metric Learning, Application to Human Activity Recognition

NATURE : Doctorat Numéro d'ordre : 2021LYSEI033

Ecole doctorale : Infomaths

Spécialité : Informatique

RESUME : This thesis aims at proposing new neural network approaches to retrieve the habitual behaviors of fragile people in order to provide them with a monitoring at home while respecting their privacy and avoiding stigmatization. In this perspective, we concentrate on the exploitation of wearable motion sensor data (accelerometer, gyrometer, magnetometer, barometer etc.) which are nowadays easily embedded into smartphones and smartwatches. In a first contribution, we propose to employ few-shot learning with an architecture called matching network to learn a personalized and flexible activity recognition model. This model learn to recognize a new class from just one or few new samples since it matches rather than classify. Therefore this model allows to better handle the large variety of activities one can do in one day while alleviating the burden of data labeling. In a second part, we advocate for a change of perspectives by proposing to retrieve recurrent unlabeled activity patterns called routines instead of precise activities. We propose a formalization of the concept of routine with the notion of almost-periodic functions which prompts us to employ sequence metric learning. We propose a neural network architecture based on robust sequence representation learning with a Sequence-to-Sequence model and metric learning with a siamese network. No activity labels are used to train the model by setting up an equivalence constraint with the data timestamps. We propose to identify the routines with a spectral clustering and to evaluate the whole routine retrieval process with information-theoretic clustering scores. The last contribution of this thesis is a new neural network model for sequence metric learning called Coupled Gated Recurrent Unit. This model has been conceived by taking inspiration from the dynamical system theory and notably the concept of synchronization. We propose to improve the siamese gating recurrent unit architecture by implementing a coupling which should allow it to better process the hard samples. We finally experiment this architecture to recognize activities and retrieve routines.

MOTS-CLÉS : Deep Learning, Metric Learning, Time Series, Dynamical Systems, Human Activity Recognition, Wearable Sensors, EHealth

Laboratoire (s) de recherche : LIRIS

Directeur de thèse: Christophe Garcia

Composition du jury : Latifa Oukhellou, Nicolas Thome, Thierry Chateau, Ahlame Douzal-Chouakria, Amaury Habrard, Stefan Duffner, Christophe Garcia, Grégoire Lefebvre