Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
ΤΟΜΕΑΣ ΣΗΜΑΤΩΝ, ΕΛΕΓΧΟΥ ΚΑΙ ΡΟΜΠΟΤΙΚΗΣ

Ταξινόμηση των Σταδίων του Ύπνου από Βιοσήματα Φορητών Συσκευών με Χρήση Βαθιών Νευρωνικών Δικτύων

Classification of Sleep Stage from Wearable-Derived Biosignals via Deep Neural Networks

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

Μυρσίνης Λεμονιάς Χρηστίδου

Επιβλέπων: Πέτρος Μαραγκός
Καθηγητής Ε.Μ.Π.
Συν-Επιβλέπων: Δρ. Αθανασία Ζλατίντση

ΕΡΓΑΣΤΗΡΙΟ ΟΡΑΣΗΣ ΥΠΟΛΟΓΙΣΤΩΝ, ΕΠΙΚΟΙΝΩΝΙΑΣ ΛΟΓΟΥ ΚΑΙ ΕΠΕΞΕΡΓΑΣΙΑΣ ΣΗΜΑΤΩΝ
Αθήνα, Ιούλιος 2023

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Σημάτων, Ελέγχου και Ρομποτικής
Εργαστήριο Όρασης Υπολογιστών, Επικοινωνίας Λόγου και Επεξεργασίας Σημάτων

Ταξινόμηση των Σταδίων του Ύπνου από Βιοσήματα Φορητών Συσκευών με Χρήση Βαθιών Νευρωνικών Δικτύων

Classification of Sleep Stage from Wearable-Derived Biosignals via Deep Neural Networks

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

Μυρσίνης Λεμονιάς Χρηστίδου

Επιβλέπων: Πέτρος Μαραγκός
Καθηγητής Ε.Μ.Π.
Συν-Επιβλέπων: Δρ. Αθανασία Ζλατίντση

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 12η Ιουλίου, 2023.

....
Πέτρος Μαραγκός
Καθηγητής Ε.Μ.Π.

....
Αθανάσιος Ροντογιάννης
Αναπληρωτής Καθηγητής Ε.Μ.Π.

....
Γεράσιμος Ποταμιάνος
Αναπληρωτής Καθηγητής Πανεπιστήμιο Θεσσαλίας

Αθήνα, Ιούλιος 2023

....
Μυρσινη Χρηστιδου Λεμονια
Διπλωματούχος Ηλεκτρολόγος Μηχανικός
και Μηχανικός Υπολογιστών Ε.Μ.Π.

Copyright © – All rights reserved Μυρσίνη Λεμονιά Χρηστίδου, 2023.
Με επιφύλαξη παντός δικαιώματος.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Περίληψη

Η έννοια του ύπνου ανέκαθεν αποτελεί ένα από τα καίρια αντικείμενα ενδιαφέροντος όσον αφορά την κατανόηση της ανθρώπινης φύσης και ύπαρξης. Παρόλο που μελετάται από την αρχαιότητα, η βαθύτερη κατανόησή του έχει αναπτυχθεί τους τελευταίους μόνο αιώνες. Η μελέτη των μοτίβων του ύπνου ενός ατόμου μπορεί να αποκαλύψει σημαντικές πληροφορίες για την γενική του υγεία, αλλά και τυχόν παθολογικές καταστάσεις. Η καταγραφή και ανάλυση του ύπνου σε πραγματικό χρόνο έχει κινήσει το επιστημονικό ενδιαφέρον, ενώ παράλληλα, οι πρόσφατες εξελίξεις στον τομέα της μηχανικής μάθησης έχουν δώσει την δυνατότητα για την εξερεύνηση ενός πολύ ισχυρού επιστημονικού πεδίου για αυτόν τον σκοπό. Η εξέλιξη της φορητής τεχνολογίας αισθητήρων έχει επίσης ωθήσει στην χρήση φορητών συσκευών για εύκολη και προσιτή καταγραφή του ύπνου δίνοντας σε ένα άτομο την δυνατότητα να παρακολουθήσει την υγεία του, καθώς και την πρόληψη πιθανών παθολογικών καταστάσεων.

Το αντικείμενο της παρούσας εργασίας είναι η μελέτη του προβλήματος της Ταξινόμησης των Σταδίων του Ύπνου μέσω νευρωνικών δικτύων, χρησιμοποιώντας δύο σύνολα δεδομένων που προέρχονται από διαφορετικές φορητές συσκευές, και περιλαμβάνουν χειροκίνητα τοποθετημένες ετικέτες ύπνου. Τα δύο σύνολα διαφέρουν ως προς το μέγεθός τους, με το πρώτο να είναι πολύ μικρότερο από το δεύτερο. Στο πρώτο σκέλος των πειραμάτων, χρησιμοποιείται μια αμφίδρομη-LSTM αρχιτεκτονική σε ένα σύνολο χαρακτηριστικών που έχουν ήδη εξαχθεί, και τα οποία είναι κοινά και για τα δύο σύνολα δεδομένων. Το μοντέλο που έχει εκπαιδευτεί με το πρώτο σύνολο δεδομένων επιτυγχάνει ανταγωνιστική επίδοση, ενώ για το δεύτερο σύνολο δεδομένων, αν και χαμηλότερη, η απόδοσή του εξακολουθεί να χαρακτηρίζεται καλή. Η προσπάθεια γενίκευσης του μοντέλου που έχει εκπαιδευτεί στο πρώτο σύνολο δεδομένων, χρησιμοποιώντας δείγματα από το δεύτερο σύνολο, δεν έχει επιτυχία. Στο δεύτερο σκέλος των πειραμάτων, δοκιμάζεται μια αυτοματοποιημένη μέθοδος εξαγωγής χαρακτηριστικών με την χρήση ενός συνελικτικού επιπέδου, το οποίο ενσωματώνεται πριν το αμφίδρομο-LSTM δίκτυο. Τα ανεπεξέργαστα δεδομένα δίνονται σαν είσοδος και το μοντέλο εκπαιδεύεται από αρχή-έως-τέλος. Το δεύτερο σύνολο δεδομένων αποδίδει πολύ καλά, φτάνοντας τις τιμές του απλού αμφίδρομου-LSTM με το πρώτο σύνολο δεδομένων, που σημαίνει ότι η συνελικτική αρχιτεκτονική μπορεί επιτυχώς να μοντελοποιήσει τις χρονικές συσχετίσεις των ανεπεξέργαστων δεδομένων. Το πρώτο σύνολο δεδομένων δεν αγγίζει τόσο υψηλή απόδοση, υποδεικνύοντας ότι δεν είναι τόσο συμβατό με την συγκεκριμένη μέθοδο εξαγωγής χαρακτηριστικών.

Τέλος, ελέγχεται η ιεραρχική αρχιτεκτονική SeqSleepNet, η οποία αρχικά προορίζεται για δεδομένα που προέρχονται από πολυσομνογραφία. Για την εκπαίδευση του μοντέλου με το πρώτο σύνολο δεδομένων, εξάγονται τα κατάλληλα σπεκτρογράμματα, όμως η απόδοση είναι φτωχή.Το δεύτερο σύνολο δεδομένων δεν μπορεί να εφαρμοστεί απευθείας στο SeqSleepNet, για αυτό προτείνονται δύο τροποποιήσεις του δικτύου, στις οποίες εκτελείται αυτόματη εξαγωγή χαρακτηριστικών μέσω ενός μηχανισμού προσοχής, πριν τα δεδομένα προχωρήσουν στο επίπεδο της ταξινόμησης. Η διαφορά στις δύο τροποποιήσεις έγκειται στο ότι η δεύτερη λαμβάνει υπόψιν την έννοια της *εποχής του ύπνου*, όπου κάθε ετικέτα ύπνου αντιστοιχεί σε ένα χρονικό παράθυρο 30-δευτερολέπτων, κάνοντας την είσοδο που δέχεται το δίκτυο πιο περίπλοκη, και οι δύο όμως έχουν εξίσου καλά αποτελέσματα. Από τα πειραματικά αποτελέσματα της εργασίας αναδεικνύεται η σημασία της προσεκτικής επιλογής χαρακτηριστικών και κατάλληλης αρχιτεκτονικής για το κάθε σύνολο δεδομένων. Μπορεί να επιτευχθεί υψηλή απόδοση των μοντέλων, η οποία υπερβαίνει άλλες πρόσφατες εργασίες, με την χρήση πιο μικρών αρχιτεκτονικών και προσεκτικά επιλεγμένων χαρακτηριστικών, είτε αυτόματα είτε χειροκίνητα, ενώ η βαθύτερη ιεραρχική αρχιτεκτονική του SeqSleepNet δεν κρίθηκε στον ίδιο βαθμό κατάλληλη για τα δύο σύνολα δεδομένων από φορητές συσκευές.

Λέξεις Κλειδιά — Ταξινόμηση Σταδίων Ύπνου, Φορητές Συσκευές, Βιοσήματα, Αμφίδρομο-LSTM, Συνελικτικά Νευρωνικά Δίκτυα, SeqSleepNet

Abstract

Sleep has always been of great interest regarding the understanding of human nature and its existence. Although it has been studied since ancient times, its deeper understanding has only flourished in the last centuries. Studying an individual's sleep patterns can reveal crucial information for their general health and it can also indicate special physical conditions. Under this perspective, monitoring and analyzing sleep in real-time has been of major scientific interest, and the recent advancements in the field of machine learning have allowed the exploitation of a very powerful scientific area for this purpose. The development of sensor technology has also boosted the use of wearable devices for easy and accessible sleep monitoring, allowing an individual's health self-supervision or prevention of special conditions.

The objective of this thesis is to study the problem of Sleep Stage Classification employing neural networks, given two datasets derived from wearable devices, which also contain manually transcribed sleep stage labels. The two datasets differ in their source of monitoring on their size, since the first one is significantly smaller than the second. In the first part of the experiments, a bidirectional-LSTM architecture is tested, trained on a set of already extracted features that are common. The model trained on the first dataset achieves competitive performance, and the one trained on the second dataset, while not reaching as high values, is still good. The generalization of the model trained on the first dataset, utilizing unseen data of the second dataset shows poor performance. In the next part of the experiments, an automated feature extraction method is proposed for training the model end-to-end, by integrating a convolutional module on the bidirectional-LSTM architecture that takes as input the raw data. The performance for the second dataset is very promising, achieving prediction values close to those reached with the bidirectional-LSTM on the first dataset. Thus, the proposed convolutional architecture can sufficiently model the temporal relationships among the raw features. The first dataset, does not perform so well in this setup, indicating that the automated feature extraction method is not adequate for it.

For the final part of the experiments a hierarchical architecture named SeqSleepNet is utilized, which was initially designed for the task of sleep stage classification with data derived from polysomnography. For the first dataset, the appropriate spectrograms are extracted for training the SeqSleepNet, however the model performs poorly. This observation suggests that the specific dataset does not align well with deeper architectures, but rather with shallower ones with carefully extracted features, either due to its small size, or due to its internal structure. The second dataset does not allow for spectrogram extraction, thus two modifications of the SeqSleepNet are proposed. Both receive the raw features of the second dataset and apply an automated feature selection with an attention mechanism, before passing the data to the classification module of the network. Their difference is that the second modification includes the concept of *sleep epoch*, where one sleep stage label is aligned to every 30-second window, adding one more dimension to the training input. The experimental results show that both of the proposed SeqSleepNet modifications achieve good performance, even though the second one handles more complex input data. Based on the experiments conducted in this thesis, we highlight the importance of a carefully chosen architecture and data handling for each dataset. We show that competitive results surpassing other recent work on the topic can be reached, either using meaningful extracted features or an automated method utilized in the shallower architecture of bidirectional-LSTM. The deeper architecture of SeqSleepNet is not appropriate for the two wearable-derived datasets.

Keywords — Sleep Stage Classification, Wearable Devices, Biosignals, Bidirectional-LSTM, Convolutional Neural Networks, SeqSleepNet

Ευχαριστίες

Η παρούσα διπλωματική εργασία εκπονήθηκε στο Εργαστήριο Όρασης Υπολογιστών, Επικοινωνίας Λόγου και Επεξεργασίας Σήματος, ολοκληρώνοντας την φοίτησή μου στην σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου. Κατά την διάρκεια της φοιτητικής μου πορείας έχω εξελιχθεί σαν άνθρωπος, γεμίζοντας με γνώσεις και εμπειρίες που δεν θα μπορούσα νωρίτερα να φανταστώ.

Θα ήθελα να ευχαριστήσω ιδιαιτέρως τον επιβλέποντα καθηγητή μου κ. Πέτρο Μαραγκό για την ευκαιρία που μου έδωσε να πραγματοποιήσω την διπλωματική μου εργασία σε αυτό το εργαστήριο, καθώς και για όλες τις γνώσεις και τα ενδιαφέρονται που έχω αποκομίσει από τα μαθήματά του, τα οποία ήταν το έναυσμα για την περαιτέρω πορεία μου τόσο στο αντικείμενο της διπλωματικής, όσο και στο επαγγελματικό επίπεδο, σαν συνέχιση των σπουδών μου. Θα ήθελα, επίσης, να ευχαριστήσω την ερευνήτρια του εργαστηρίου και επιβλέπουσά μου στην διπλωματική, Νάνσυ Ζλατίντση, για την αμέριστη υπομονή και υποστήριξή της, καθώς και τις συμβουλές της που έπαιξαν καθοριστικό ρόλο στην διεκπεραίωση αυτής της εργασίας.

Ένα ιδιαίτερα μεγάλο ευχαριστώ θέλω να πω στους φίλους μου, παλιούς και νέους, και ιδίως τις αγαπημένες μου φίλες που με ακολουθούν από την αρχή της φοιτητικής μας πορείας, καθώς και όλους όσους ήταν γύρω μου και με στήριξαν κατά την απαιτητική διαδικασία της εκπόνησης αυτής της εργασίας, χαρίζοντάς μου όμορφες στιγμές και δύναμη για να συνεχίσω το έργο μου. Θα ήθελα επίσης να αναφερθώ σε μια ιδιαίτερη ομάδα της οποίας έγινα μέλος στο πρώτο βήμα της επαγγελματικής μου ζωής, οι οποίοι κατά την διάρκεια εκπόνησης της διπλωματικής μου έχουν δείξει εξαιρετική κατανόηση και υποστήριξη με κάθε τρόπο, και θέλω να τους ευχαριστήσω από καρδιάς.

Τέλος, θέλω να ευχαριστήσω την οικογένειά μου, της οποίας είχα την αμέριστη στήριξη καθ'όλη την διάρκεια εκπόνησης αυτής της εργασίας, είτε είχαμε καθημερινή συναναστροφή, είτε πιο σποραδική επικοινωνία. Ιδιαίτερα τους γονείς μου, χωρίς τους οποίους δεν θα είχα καταφέρει τίποτα από όλα αυτά, τις αδερφές μου που ήταν πάντα δίπλα μου να με ακούσουν, τον παππού μου που έχει αποτελέσει την μεγαλύτερη πηγή έμπνευσης στην προσέγγισή μου στην επιστήμη, την γιαγιά μου που είναι ο πιο γλυκός άνθρωπος και, αν και μακριά, με έχει στηρίξει περισσότερο από όσο φαντάζεται, καθώς και τον θείο μου, που με την καθαρή του οπτική με έχει βοηθήσει να δω λογικά στις πιο ευάλωτες στιγμές μου.

Η ευγνωμοσύνη που νιώθω με την ολοκλήρωσης αυτής της εργασίας και κατά συνέπεια της σχολής μου είναι δύσκολο με λόγια να περιγραφεί, ξέρω όμως ότι ένα κομμάτι της ζωής μου έρχεται συγκινητικά στο τέλος του και ένα νέο κεφάλαιο ξεκινάει, το οποίο περιμένω με ενθουσιασμό να εξελιχθεί.

Χρηστίδου Μυρσίνη
Ιούλιος 2023

Contents

List of Figures

List of Tables

Chapter 0

Εκτεταμένη Περίληψη

0.1 Εισαγωγή

Ο ύπνος αποτελεί μια από τις πιο θεμελιώδεις ανάγκες του ανθρώπου, από την απαρχή της ύπαρξής του. Μέσω της μελέτης της λειτουργίας του ύπνου μπορεί κανείς να διακρίνει πολλές κρυφές πτυχές της φυσικής ή ψυχολογικής κατάστασης ενός ατόμου, καθώς και της γενικότερης υγείας του.

Στην παρούσα εργασία αναπτύσσεται μια προσέγγιση του προβλήματος της αυτοματοποιημένης κατηγοριοποίησης των σταδίων του ύπνου με την χρήση νευρωνικών δυκτύων, μέσω σημάτων που έχουν συλλεχθεί από ένα έξυπνο ρολόι και μια ακόμη φορητή συσκευή καταγραφής βιοσημάτων, συγκεκριμένα έναν ακτιγράφο ο οποίος εφαρμόζεται στον καρπό παράλληλα με την εκτέλεση μελέτης ύπνου.

0.1.1 Επιστημονικό Υπόβαθρο

Το ζήτημα της μελέτης και της κατανόησης του ύπνου έχει βρεθεί στο επίκεντρο του ανθρώπινου ενδιαφέροντος ακόμη από την αρχαιότητα, λαμβάνοντας πολλές διαφορετικές μορφές παράλληλα με την εξέλιξη της επιστήμης και του πολιτισμού. Ξεκινώντας από την θεώρηση του ύπνου ως μια κατάσταση μη-αντίληψης και αδράνειας, τον τελευταίο κυρίως αιώνα έχει καταστεί σαφές ότι πρόκειται για κάτι πολύ βαθύτερο από αυτό. Συγκεκριμένα, ο ύπνος αποτελεί μια διαδικασία έντονης δραστηριότητας του εγκεφάλου, κατά την διάρκεια της οποίας πραγματοποιείται η κυκλική εναλλαγή ενός συνόλου καταστάσεων, καθεμία από τις οποίες επιτελεί μια διαφορετική λειτουργία, ζωτικής σημασίας για τον άνθρωπο. Οι παρατηρήσεις αυτές απέκτησαν σαφήνεια με την ανακάλυψη του ηλεκτροεγκεφαλογραφήματος και της καταγραφής των ηλεκτρικών σημάτων των νευρώνων του εγκεφάλου, που υποδεικνύουν μια έντονη κινητικότητα κατά την διάρκεια του ύπνου [LHH35; DK57]. Μέσω αυτών των παρατηρήσεων, και σε συνδυασμό με άλλες έρευνες που πραγματοποιήθηκαν για την καταγραφή ηλεκτρικών σημάτων όπως το ηλεκτρομυογράφημα και το ηλεκτροοφθαλμογράφημα, ανακαλύφθηκαν πέντε βασικά στάδια ύπνου:

- **Στάδιο N1** - μεταβατικό στάδιο που σχετίζεται με την έναρξη του ύπνου. Ο ρόλος του είναι μεταβατικός για το άτομο, ώστε να επέλθει από κατάσταση εγρήγορσης σε κατάσταση ύπνου, και αποτελεί το 2-5% της συνολικής διάρκειας του ύπνου.
- **Στάδιο N2** - το σώμα εισέρχεται σε μια κατάσταση βαθύτερης καταστολής, στην οποία η θερμοκρασία πέφτει, ο καρδιακός παλμός και η αναπνοή σταθεροποιούνται και η κίνηση των ματιών ελαχιστοποιείται ή και σταματάει εντελώς. Αποτελεί περίπου το 45-55% της συνολικής διάρκειας του ύπνου, επομένως ένα άτομο περνάει τυπικά τον μισό του ύπνο στο στάδιο N2.
- **Στάδιο N3 + N4** (πλέον συνενωμένα ως N3) - πρόκειται για έναν βραδέων-κυμάτων ύπνο, που θεωρείται το βαθύτερο στάδιο ύπνου και εμφανίζεται κυρίως στο πρώτο ένα τρίτο της νύχτας. Ο καρδιακός παλμός και η αναπνοή πέφτουν στις χαμηλότερες τιμές τους, οι μύες χαλαρώνουν, και αποτελεί το στάδιο από το οποίο είναι δυσκολότερο να ξυπνήσει κανείς. Το στάδιο αυτό θεωρείται εξαιρετικά σημαντικό τόσο για την αποκατάσταση του σώματος από τις φυσικές φθορές, όσο και για την ανάπτυξη του ατόμου. Επιπλέον, στο στάδιο αυτό διενεργείται η επεξεργασία και παγίωση των γνωστικών αναμνήσεων.
- **Στάδιο REM** (Ταχεία Κίνηση των Ματιών) - το στάδιο αυτό συσχετίζεται με τα όνειρα. Παρατηρείται μια αυξημένη δραστηριότητα του εγκεφάλου, ο καρδιακός παλμός και η πίεση του αίματος φτάνουν σε επίπεδα που αντιστοιχούν σε ένα άτομο που περπατάει, ενώ παράλληλα συμβαίνει παράλυση των μυών, ώστε να αποφευχθεί πιθανή φυσική δραστηριότητα λόγω εξωτερίκευσης των ονείρων. Τέλος, θεωρείται ότι κατά την διάρκεια αυτού του σταδίου γίνεται η επεξεργασία των συναισθηματικών αναμνήσεων του ατόμου, γεγονός που το καθιστά ουσιώδες για την μάθηση και την δημιουργικότητα. Το στάδιο REM αποτελεί περίπου το 25% της συνολικής διάρκειας του ύπνου για έναν υγιή ενήλικα.

Έχοντας αυτά ως δεδομένα, και γνωρίζοντας πλέον την σημασία της επίδρασης του ύπνου στην γενικότερη υγεία και ευρωστία του ανθρώπου, η καταγραφή και η παρακολούθησή του κρίνονται απαραίτητα στοιχεία για την εξασφάλιση μιας υγειούς καθημερινότητας, όπως και της αντιμετώπισης διαφόρων διαταραχών σχετιζόμενων με τον ύπνο, στις οποίες επίσης δίνεται πλέον ιδιαίτερη βαρύτητα.

Μέχρι πρότινος η καταγραφή των σταδίων του ύπνου γινόταν αποκλειστικά σε ειδικά διαμορφωμένα εργαστήρια, στα οποία το άτομο περνούσε όλη την διάρκεια της νύχτας φορώντας ηλεκτρόδια για την καταγραφή των διαφόρων σημάτων υπό την συνεχή επίβλεψη ενός εξειδικευμένου επιστήμονα. Η διαδικασία αυτή ονομάζεται *μελέτη ύπνου* μέσω πολύυπνογραφικών μετρήσειων (PSG - polysomnography) και τα βασικά σήματα τα οποία καταγράφονται είναι το ηλεκτροεγκεφαλογράφημα, η κίνηση των ματιών (ηλεκτροοφθαλμογράφημα), το ηλεκτρομυογράφημα, ο καρδιακός παλμός και τα επίπεδα οξυγόνου στο αίμα. Ακολουθεί η ανάλυση των σημάτων αυτών για την τελική κατηγοριοποίηση των σταδίων του ύπνου και την εξαγωγή συμπερασμάτων. Η κατηγοριοποίηση των σταδίων του ύπνου γίνεται από έναν εξειδικευμένο επιστήμονα, ο οποίος, μελετώντας τα εξαγόμενα σήματα ανά χρονικά παράθυρα 30 δευτερολέπτων, αναθέτει μια ετικέτα σταδίου ύπνου ανά χρονικό παράθυρο, το οποίο ονομάζεται μια *εποχή*, για όλη την διάρκεια του ύπνου.

Καθώς η διαδικασία αυτή είναι ιδιαίτερα χρονοβόρα και απαιτητική, τα τελευταία χρόνια γίνονται προσπάθειες υιοθέτησης νέων προσεγγίσεων, στις οποίες έχει διαδραματίσει καίριο ρόλο η παράλληλη εξέλιξη της τεχνολογίας των αισθητήρων και των μικρο-αισθητήρων, όπως επίσης οι νέες μέθοδοι ανάλυσης δεδομένων μέσω της μηχανικής μάθησης και των νευρωνικών δικτύων.

0.1.2 Ορισμός Προβλήματος

Σε αυτό το πλαίσιο, η παρούσα εργασία κάνει χρήση δύο συνόλων δεδομένων, ενός συλλεγμένου από ένα έξυπνο ρολόι και ενός καταγεγραμμένου από μια άλλη φορητή συσκευή, με σκοπό την διερεύνηση αυτοματοποιημένων μεθόδων μέσω νευρωνικών δικτύων για την επίτευξη της κατηγοριοποίησης των σταδίων του ύπνου. Τα δεδομένα και στις δύο περιπτώσεις αποτελούνται από σήματα **επιτάχυνσης του καρπού** μέσω ενός τρισδιάστατου μικρο-επιταχυνσιόμετρου και **καρδιακών παλμών** μέσω ενός οπτικού πληθυσμογράφου, τα οποία είναι είτε και τα δύο ενσωματωμένα στο ψηφιακό ρολόι (Walch dataset [Wal19]), είτε αποτελούν δύο ξεχωριστές φορητές συσκευές, οι οποίες συλλέγουν τα δεδομένα κατά την διάρκεια του ύπνου (MESA dataset [Nat16; Zha+18a; Che+15]). Επίσης, και στα δύο σύνολα δεδομένων περιλαμβάνονται οι **ετικέτες με το σωστό στάδιο ύπνου** για κάθε μέτρηση, οι οποίες έχουν οριοθετηθεί από κάποιον εξειδικευμένο επιστήμονα, ενώ παράλληλα κατά την διάρκεια της μελέτης οι συμμετέχοντες φοράνε τις φορητές συσκευές μέτρησης.

Η δομή της εργασίας αποτελείται από δύο σκέλη, στα οποία εξετάζονται δύο διαφορετικές αρχιτεκτονικές:

1. Η πρώτη βασίζεται σε ένα ανακυκλούμενο (*recurrent*) νευρωνικό δίκτυο, και πιο συγκεκριμένα επιλέγεται η αρχιτεκτονική **δικτύου μακροπρόθεσμης μνήμης** (Long-Short Term Memory). Η επιλογή του συγκεκριμένου είδους δικτύου γίνεται λόγω της φύσης του προβλήματος, καθώς πρόκειται για εφαρμογή προβλέψεων πάνω σε χρονικές ακολουθίες. Τα RNN δίκτυα έχουν σαν ιδιαίτερο χαρακτηριστικό την δυνατότητα να διατηρούν πληροφορίες από προηγούμενες χρονικές στιγμές με την βοήθεια μιας ειδικής δομής *μνήμης*, την οποία ενσωματώνουν στην επεξεργασία για τις προβλέψεις της παρούσας χρονικής στιγμής, επομένως αποτελούν ιδανική επιλογή δικτύου για το πρόβλημα της ταξινόμησης των σταδίων του ύπνου. Στα πλαίσια αυτά, εξετάζονται δύο παραλλαγές του δικτύου, ανάλογα με τον τρόπο που δίνονται τα δεδομένα στην είσοδο για την εκπαίδευσή του:

    (a) Στην πρώτη περίπτωση, το δίκτυο δέχεται σαν είσοδο ένα σύνολο χαρακτηριστικών που έχουν εκ των προτέρων εξαχθεί από τα ανεπεξέργαστα δεδομένα με την μέθοδο που προτείνεται σε προηγούμενη εργασία, η οποία έχει προτείνει και το σύνολο δεδομένων από το έξυπνο ρολόι [Wal+19].

    (b) Στην δεύτερη περίπτωση δοκιμάζεται μια αυτοματοποιημένη μέθοδος εξαγωγής χαρακτηριστικών, μέσω μιας συνελικτικής μονάδας (Convolutional Neural Network) που προστίθεται στο αρχικό νευρωνικό δίκτυο. Με αυτόν τον τρόπο, τα σημαντικότερα χαρακτηριστικά της εσωτερικής δομής των ανεπεξέργαστων δεδομένων εξάγονται αυτόματα από το δίκτυο πριν δοθούν σαν είσοδος στο επόμενο επίπεδό του, για την εκμάθηση της πρόβλεψης των σταδίων του ύπνου της χρονοσειράς. Σε αυτήν την περίπτωση, η διαδικασία εκπαίδευσης όλων των επιπέδων του συνολικού δικτύου γίνεται ταυτόχρονα.

2. Η δεύτερη αρχιτεκτονική βασίζεται σε ένα υπάρχον μοντέλο βαθιάς μηχανικής μάθησης **SeqSleepNet** [Pha+19], στο οποίο γίνονται ορισμένες τροποποιήσεις προκειμένου να προσαρμοστεί στα δεδομένα της παρούσας εργασίας. Το μοντέλο αυτό αρχικά προτείνεται για την πρόβλεψη των σταδίων του ύπνου από σήματα που έχουν προέλθει από μελέτη ύπνου PSG, και συγκεκριμένα ηλεκτροεγκεφαλογράφημα, ηλεκτροοφθαλμογράφημα και ηλεκτρομυογράφημα. Τα δεδομένα αυτά είναι πολύ πιο πυκνά από ό,τι τα δεδομένα που προέρχονται από φορητές συσκευές και χρησιμοποιούνται στην παρούσα εργασία, επομένως ορισμένες αλλαγές γίνονται στο δίκτυο για να προσαρμοστεί η αρχιτεκτονική του κατάλληλα.

0.2 Περιγραφή Συνόλων Δεδομένων

Τα σύνολα δεδομένων που χρησιμοποιούνται στην παρούσα εργασία προέρχονται και τα δύο από φορητές συσκευές, συγκεκριμένα το ένα από έξυπνο ρολόι και το δεύτερο από τον συνδυασμό ενός επιταχυνσιόμετρου καρπού και ενός παλμικού οξύμετρου. Παράλληλα με την καταγραφή των βιοσημάτων μέσω των φορητών συσκευών, οι συμμετέχοντες και στις δύο έρευνες έχουν βρεθεί ταυτόχρονα σε κάποιο ειδικά διαμορφωμένο εργαστήριο για μελέτη ύπνου κατά τη διάρκεια μιας νύχτας, όπου και καταγράφονται και αναλύονται τα στάδια του ύπνου τους από έναν εξειδικευμένο επιστήμονα.

0.2.1 Σύνολο Δεδομένων Walch

Πρόκειται για ένα εσωτερικό σύνολο δεδομένων σχεδιασμένο και δημιουργημένο από την ομάδα της Walch et al. [Wal19] και μπορεί να βρεθεί εδώ. Το σύνολο δεδομένων αποτελείται από 31 άτομα κατά τη διάρκεια ενός 8ωρου νυχτερινού ύπνου. Κάθε συμμετέχων παρακολουθείται με PSG, μέσω του οποίου εξάγονται ετικέτες για το στάδιο του ύπνου ανά χρονικό παράθυρο 30 δευτερολέπτων, το οποίο ονομάζεται *εποχή*. Στους συμμετέχοντες δόθηκε να φορέσουν ένα έξυπνο ρολόι της Apple (Apple Watch) κατά την διάρκεια του νυχτερινού ύπνου στο εργαστήριο, το οποίο καταγράφει την **επιτάχυνση** (σε g) και τον **καρδιακό παλμό** (σε παλμούς ανά λεπτό, bpm). Συγκεκριμένα:

- **Επιτάχυνση (motion)** στον τρισδιάστατο άξονα (x, y, z), με τιμή μέτρησης σε $g(9.8m/s)$ και ρυθμό δειγματοληψίας 50Hz. Στην πραγματικότητα η συχνότητα δειγματοληψίας διέφερε ανάμεσα στους συμμετέχοντες, επομένως απαιτείται κάποια μαθηματική παρεμβολή σημείων ώστε να γίνει ακριβώς 50Hz. Επίσης, μέσα στις χρονοσειρές των χαρακτηριστικών υπάρχουν περιστασιακά μικρά χρονικά παράθυρα που τα δεδομένα τους λείπουν, πιθανώς λόγω προβλημάτων από την πλευρά του διακομιστή κατά την διάρκεια της καταγραφής του ύπνου των ατόμων.
- **Καρδιακός παλμός (heart rate)**, ο ρυθμός δειγματοληψίας του οποίου είναι κάθε μερικά δευτερόλεπτα (beats per minute) και είναι μονοδιάστατος. Για να υπάρχει ομοιόμορφος ρυθμός δειγματοληψίας ανάμεσα στα άτομα, η καλύτερη προσέγγιση είναι η επανα-δειγματοληψία κάθε 1 δευτερόλεπτο ώστε να είναι σταθερή στο 1Hz. Και στην περίπτωση του καρδιακού παλμού υπάρχουν τμήματα με ελλιπή δείγματα.
- **PSG (labeled sleep)** δεδομένα κατά την διάρκεια του ύπνου: ένα στάδιο ύπνου ορίζεται για κάθε τμήμα 30-δευτερολέπτων, που ονομάζεται αλλιώς μια *εποχή*. Χρησιμοποιούνται σαν ετικέτες για τα στάδια του ύπνου και δεν έχουν κενά ανάμεσα στις τιμές τους. Συγκεκριμένα, υπάρχουν οι ετικέτες: **wake** = 0 (ξύπνιος), **N1** = 1, **N2** = 2, **N3** = 3, **N4** = 4, **REM** = 5. Παρ'όλα αυτά περιλαμβάνονται επίσης κάποιες τιμές -1, που ορίζουν ότι δεν υπάρχει κάποια ετικέτα για την συγκεκριμένη εποχή, και οι τιμές αυτές πρέπει να χειριστούν με προσοχή.
- **Βήματα (count)**: Κατά την διάρκεια των μετρήσεων συλλέγονται επίσης τα βήματα των συμμετεχόντων από το έξυπνο ρολόι.

0.2.2 Σύνολο Δεδομένων MESA

Το δεύτερο σύνολο δεδομένων που χρησιμοποιείται σε αυτήν την εργασία είναι η *Πολυ-Εθνική Μελέτη για την Αθηροσκλήρωση* (Multi-Ethnic Study of Atherosclerosis - MESA) [Nat16; Zha+18a; Che+15], και βρίσκεται στο https://sleepdata.org/datasets/mesa. Πρόκειται για μια πολυκεντρική πολυ-εθνική έρευνα των παραγόντων που σχετίζονται με την ανάπτυξη υποκλινικής καρδιαγγειακής νόσου και την εξέλιξή της σε κλινική καρδιαγγειακή νόσο. Μεταξύ 2010-2012, 2237 άτομα συμμετείχαν σε μια εξέταση ύπνου (MESA sleep), κατά την οποία διεξήχθη ολονύχτια πολυσομνογραφία PSG, υπήρχε ένα ερωτηματολόγιο ύπνου, καθώς επίσης 7ήμερη χρήση ενός ακτινογράφου που φοριέται στον καρπό του χεριού. Στόχος της μελέτης αυτής ήταν να εξερευνηθεί το πώς οι διακυμάνσεις του ύπνου και οι υπνικές διαταραχές ποικίλλουν μεταξύ των φύλων και των εθνοτικών ομάδων όπως επίσης και η συσχέτισή τους με τις μετρήσεις της υποκλινικής αθηροσκλήρωσης.

Τα παραπάνω δεδομένα χρησιμοποιούνται όπως και εδώ [Wal+19], κατ' αρχάς για να ελεγχθεί η δυνατότητα γενίκευσης νευρωνικών μοντέλων ήδη εκπαιδευμένων, σε καινούρια δεδομένα που δεν είχαν περιληφθεί στο σύνολο δεδομένων εκπαίδευσης. Στην συνέχεια όμως, χρησιμοποιούνται ως ξεχωριστό σύνολο δεδομένων για να εκπαιδευτούν τα προτεινόμενα μοντέλα σε μεγαλύτερο όγκο δεδομένων από ό,τι αυτά της Walch, και τα οποία έχουν ληφθεί με διαφορετικά, αν και παρεμφερή, εργαλεία. Σε αυτό το πλαίσιο, επιλέγονται τα πρώτα 188 άτομα από το σύνολο δεδομένων της MESA, με παράλληλες μετρήσεις ακτιγραφίας καρπού και δεδομένων PSG και επεξεργάζονται κατάλληλα για να χρησιμοποιηθούν σαν ανεξάρτητο σύνολο δεδομένων για έλεγχο των μοντέλων. Όπως προτείνεται και στην προαναφερθείσα εργασία της Walch, υπάρχει μια άμεση συσχέτιση μεταξύ της κίνησης του καρπού και της τοπικής τυπικής απόκλισης (standard deviation) του καρδιακού παλμού του έξυπνου ρολογιού, με τα χαρακτηριστικά που εξάγονται από την ακτιγραφία καρπού και τον καρδιακό παλμό κατά την διάρκεια του PSG αντίστοιχα του συνόλου δεδομένων της MESA. Ο καρδιακός παλμός της MESA λαμβάνεται μέσω παλμικής οξυμετρίας, αυξάνοντας την συμβατότητα μεταξύ του έξυπνου ρολογιού που χρησιμοποιείται για την εκπαίδευση ορισμένων νευρωνικών δικτύων στην συνέχεια της εργασίας, και στον έλεγχό τους από τα δεδομένα της MESA.

Το σύνολο δεδομένων της MESA αποτελείται από τα παρακάτω χαρακτηριστικά:

- **Καρδιακός παλμός**: συλλέγεται μέσω μετρήσεων πολυσομνογράφου (PSG), και έχει ρυθμό δειγματοληψίας 1.
- **Ακτιγραφία**: 2237 άτομα συμμετείχαν στην έρευνα φορώντας στον καρπό τους φορητές συσκευές ακτιγραφίας (Actiwatch Spectrum, Philips Respironics), για μια εβδομάδα. Οι εγγραφές είναι αποθηκευμένες σε αρχεία ανά εποχή, όπου κάθε γραμμή σε ένα αρχείο αντιπροσωπεύει 30-δευτερόλεπτα περιληπτικών μετρήσεων από τα δεδομένα της ακτιγραφίας, για 2159 συμμετέχοντες.
- Οι **ετικέτες της πολυσομνογραφίας** (PSG), ανταποκρίνονται σε πέντε στάδια ύπνου: (**N1, N2, N3, N4, REM, Wake**).

  Το PSG πραγματοποιήθηκε εντός-οικίας με την χρήση της τεχνολογίας Compumedics Somte System (Compumedics Ltd., Abbotsford, Australia). Τα σήματα που καταγράφονται αποτελούνται από ηλεκτροεγκεφαλογράφημα, ηλετροοφθαλμογράφημα, ηλεκτρομυογράφημα σαγονιού, θωρακική και κοιλιακή αναπνευστική επαγωγική πλεγματογραφία, την ροή του αέρα της αναπνοής, ηλεκτροκαρδιογράφημα, κίνηση των ποδιών και δακτυλική παλμική οξυμετρία. Παράλληλα, εκπαιδευμένοι τεχνικοί βάζουν τις κατάλληλες ετικέτες ύπνου.

Στην Εικόνα 0.2.1 παρουσιάζονται οι κατανομές των κλάσεων, για τα διαφορετικά προβλήματα κατηγοριοποίησης του ύπνου που μελετώνται σε αυτήν την εργασία, για τα δύο σύνολα δεδομένων που περιεγράφησαν παραπάνω. Συμπεριλαμβάνεται επίσης το ιστόγραμμα της κατανομής των αρχικών έξι κλάσεων των δύο συνόλων, στις οποίες υπάρχει το στάδιο ύπνου N4. Αυτό συγχωνεύεται με το στάδιο ύπνου N3 για τις απαιτήσεις της παρούσας εργασίας.

0.3 Bidirectional LSTM Μοντέλα

Σαν πρωταρχικό πείραμα, προτείνεται ένα απλό αμφίδρομο δίκτυο μακροχρόνιας-βραχυπρόθεσμης μνήμης (bidirectional long-short term memory - bidirectional LSTM), καθώς επιλύει ακριβώς το είδος του προβλήματος που καλούμαστε να αντιμετωπίσουμε, δηλαδή την κατηγοριοποίηση χρονοσειράς. Το δίκτυο αυτό δοκιμάζεται πάνω σε χαρακτηριστικά που έχουν εξαχθεί μέσω κλασικών τεχνικών επεξεργασίας σήματος από τα δεδομένα της Walch, όπως προτείνεται στην αντίστοιχη δημοσίευση [Wal+19]. Στόχος είναι να ελεγχθεί η απόδοση των χαρακτηριστικών αυτών σε μεθόδους βαθιάς μηχανικής μάθησης μέσω νευρωνικών δικτύων, σε αντίθεση με τις κλασικές μεθόδους ταξινόμησης που χρησιμοποιούνται στην προαναφερθείσα εργασία. Στην συνέχεια, το ίδιο βασικό bidirectional-LSTM δίκτυο χρησιμοποιείται σε μια πιο εξελιγμένη αρχιτεκτονική, στην οποία γίνεται αυτοματοποιημένη εξαγωγή χαρακτηριστικών από τα ανεπεξέργαστα δεδομένα, μέσω μιας συνελικτικής αρχιτεκτονικής.

0.3.1 Προετοιμασία των Δεδομένων

Προεπεξεργασία Δεδομένων Walch

Εφόσον τα πειράματα της παρούσας εργασίας εστιάζουν στην επεξεργασία χρονοσειρών, κατ' αρχάς πρέπει να επιβεβαιωθεί ότι οι σειρές των δεδομένων που χρησιμοποιούνται είναι συνεχείς. Για να επιτευχθεί αυτό, πριν την εξαγωγή των χαρακτηριστικών της Walch, πρέπει να αντιμετωπιστούν τα σημεία στα οποία υπάρχουν κενά μεταξύ των εποχών της χρονοσειράς, κάτι το οποίο δεν λαμβάνεται υπ' οψιν στην εργασία της Walch, καθώς εκεί

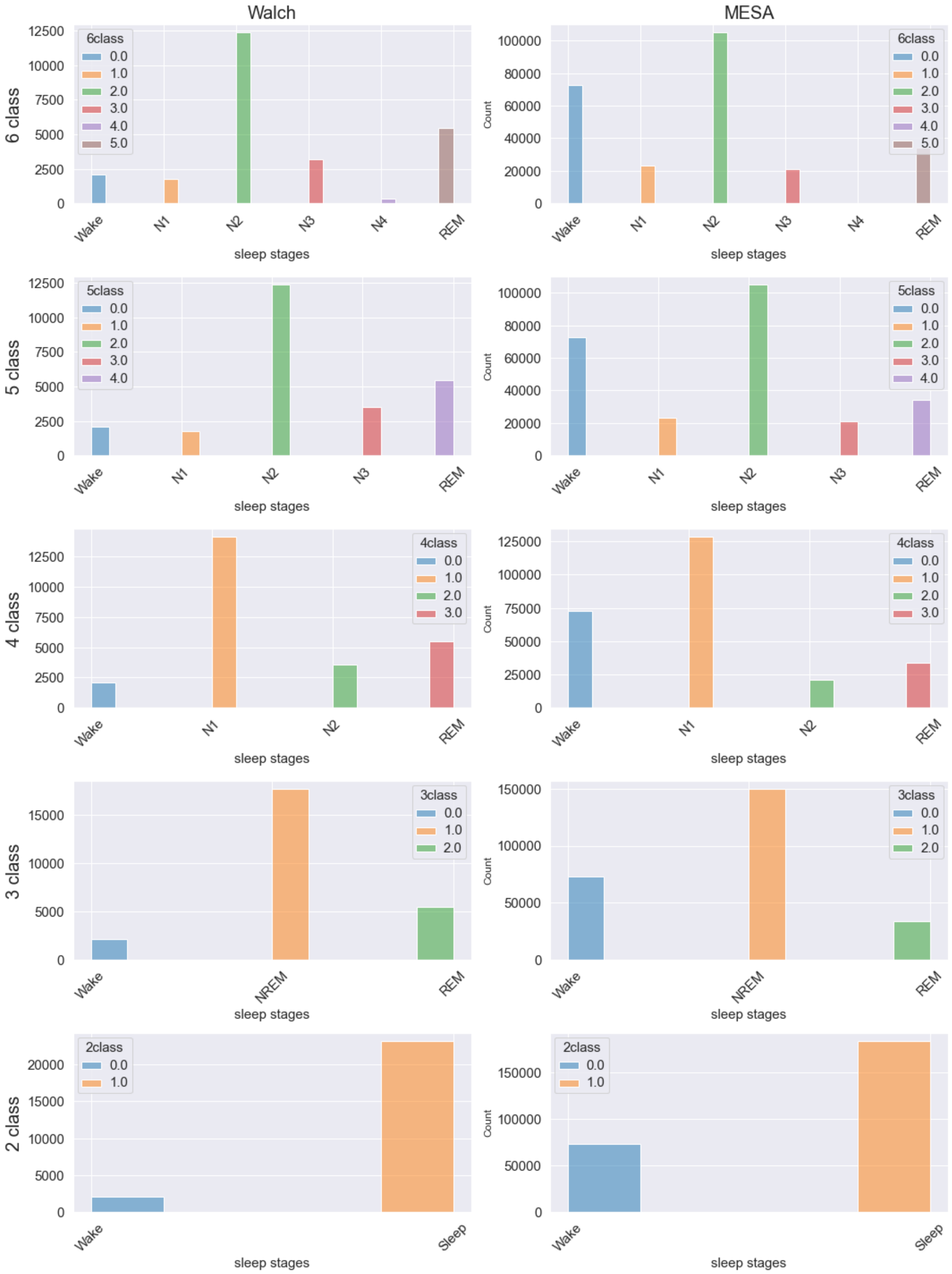


Figure 0.2.1: Τα ιστογράμματα των κατανομών των κλάσεων για τα διαφορετικά προβλήματα κατηγοριοποίησης του ύπνου. Οι έξι κλάσεις αναφέρονται στις αρχικές ετικέτες σταδίων του ύπνου που παρέχονται στα σύνολα δεδομένων, ενώ οι υπόλοιπες κατηγορίες κλάσεων μελετώνται σε αυτήν την εργασία.

τα χαρακτηριστικά αφορούν κάθε χρονική στιγμή (εποχή 30 δευτερολέπτων) χωριστά, και όχι την μεταξύ τους συσχέτιση. Προκειμένου να έχουμε συνεχή χρονικά τμήματα, τα ακατέργαστα δεδομένα σπάνε στα σημεία από τα οποία λείπουν εποχές από τουλάχιστον ένα χαρακτηριστικό, έτσι ώστε τα τελικά τμήματα των ανεπεξέργαστων δεδομένων να έχουν συνεχείς χρονικές εποχές με έγκυρες τιμές για όλα τα χαρακτηριστικά (psg, hr, κίνηση του καρπού).

Στην συνέχεια, πρέπει να αντιμετωπιστούν οι -1 τιμές που εμφανίζονται στις PSG ετικέτες, που υποδεικνύουν μη-ταξινομημένα δεδομένα.

1. Στην περίπτωση που οι τιμές αυτές εμφανίζονται στην αρχή ή στο τέλος της χρονοσειράς, τότε το κομμάτι αυτό απλά αφαιρείται από τα δεδομένα.

2. Στην περίπτωση μεμονωμένων τέτοιων τιμών ενδιάμεσα στα δεδομένα, τότε αντικαθίστανται από την μέση τιμή των δύο γειτονικών της ετικετών.

3. Τέλος, σε περίπτωση που υπάρχουν περισσότερες από μια συνεχόμενες αρνητικές τιμές εσωτερικά στην χρονοσειρά, τότε αυτή κόβεται σε δύο τμήματα στο σημείο εκείνο, και οι αρνητικές τιμές αφαιρούνται από τις δύο υπο-ακολουθίες που προκύπτουν; παρ' όλα αυτά δεν παρατηρείται η περίπτωση αυτή στο συγκεκριμένο σύνολο δεδομένων.

Εξαγωγή Χαρακτηριστικών

Τα χαρακτηριστικά που εξάγονται όπως προτείνεται στην εργασία της Walch [Wal+19], και χρησιμοποιούνται στα αρχικά μας πειράματα, είναι τα ακόλουθα:

- **Καταμέτρηση δραστηριοτήτων** (Activity counts) - εξάγονται από τα δεδομένα της κίνησης του καρπού. Μετατρέπεται από τα ανεπεξέργαστα δεδομένα της επιτάχυνσης του καρπού (σε m/s^2), μέσω της μεθόδου που προτείνεται στο [TV13] και υλοποιείται στον κώδικα που έχει εκδόσει η Walch. Το χαρακτηριστικό αυτό επιλέγεται λόγω της συμβατότητάς του με μετρήσεις που εξάγονται από άλλα εργαλεία, εφόσον αυτές μετατραπούν επίσης σε *καταμέτρηση δραστηριοτήτων (activity counts).* Για την εξαγωγή του χαρακτηριστικού αυτού, τα δεδομένα διαχωρίζονται σε χρονικά παράθυρα 10 λεπτών γύρω από κάθε εποχή. Το τελικό χαρακτηριστικό του *activity count* προκύπτει από την συνέλιξη του χρονικού παραθύρου με μια Γκαουσιανή κατανομή με $\sigma = 50$ seconds.

- **Μετασχηματισμός συνημιτόνου** (cosine transform) - αντιπροσωπεύει μια απλοποιημένη μορφή του *κιρκαδικού ρολογιού* του ατόμου. Εφαρμόζεται ένα σταθεροποιημένο συνημίτονο, το οποίο είναι κατάλληλα κλιμακωμένο και σχετικά μετατοπισμένο με τον χρόνο έναρξης της καταγραφής των σημάτων, και το οποίο αυξάνεται και ελαττώνεται κατά την διάρκεια της νύχτας, έτσι ώστε να ταυτίζεται με την φυσιολογική έκφραση του κιρκαδικού ρολογιού κατά την διάρκεια του ύπνου.

- **Χαρακτηριστικό καρδιακού παλμού** - εξάγεται ύστερα από κάποια βήματα προ-επεξεργασίας του αρχικού σήματος. Αρχικά, το σήμα $\pi\alpha\rho\epsilon\mu\beta\acute{\alpha}\lambda\lambda\epsilon\tau\alpha\iota$ με σημεία (interpolation), ώστε να αποκτήσει ρυθμό δειγματοληψίας ακριβώς 1Hz. Στην συνέχεια, εξομαλύνεται (smoothed) και φιλτράρεται μέσω της συνέλιξης με την διαφορά δύο Γκαουσιανών φίλτρων, με $sigma = 120$ seconds και $sigma = 600$ seconds. Οι μετρήσεις κάθε ατόμου κανονικοποιούνται διαιρώντας με το ενενηκοστό εκατοστημόριο της απόλυτης διαφοράς μεταξύ κάθε σημείου των δεδομένων του καρδιακού παλμού και της μέσης τιμής του καθ' όλη την διάρκεια του ύπνου. Τέλος, η τυπική απόκλιση του παραθύρου που σχηματίζεται γύρω από την κάθε εποχή χρησιμοποιείται σαν το χαρακτηριστικό που εκπροσωπεί τον καρδιακό παλμό.

- **Χαρακτηριστικό του χρόνου** - αναφέρεται στον χρόνο από την έναρξη της εγγραφής.

- **Ετικέτες PSG**

0.3.2 Απλή αμφίδρομη LSTM αρχιτεκτονική

Η πρώτη απλή αρχιτεκτονική αμφίδρομου δικτύου μακροχρόνιας-βραχυπρόθεσμης μνήμης (bidirectional long-short term memory - bidirectional LSTM) υλοποιείται με την χρήση του πακέτου **Pytorch** [Pas+19]. Το προτεινόμενο μοντέλο αποτελείται από:

- Ένα **LSTM επίπεδο**, έχοντας ορίσει την παράμετρο του αμφίδρομου σαν *σωστή*.

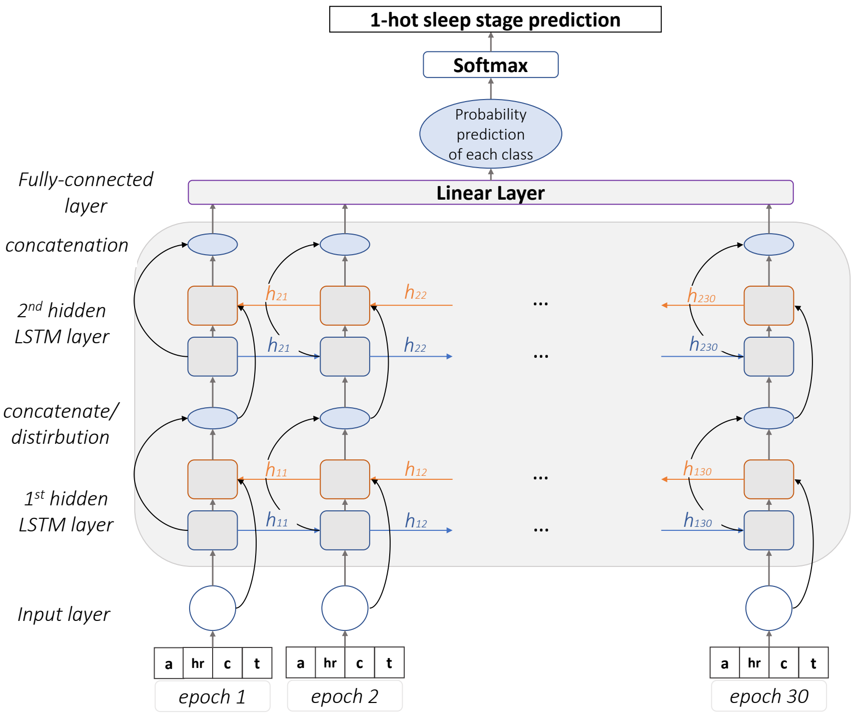


Figure 0.3.1: Η προτεινόμενη αρχιτεκτονική αμφίδρομου δικτύου μακροχρόνιας-βραχυπρόθεσμης μνήμης (bidirectional long-short term memory - bidirectional LSTM).

- Ένα **πλήρως συνδεδεμένο επίπεδο**, αλλιώς *γραμμικό επίπεδο*, στο οποίο όλοι οι κόμβοι εισόδου του νευρωνικού δικτύου είναι συνδεδεμένοι με τους κόμβους εξόδου. Δέχεται σαν είσοδο την έξοδο του BiLSTM, και επιστρέφει ένα διάνυσμα μεγέθους όσα και στα στάδια ύπνου του προβλήματος, το οποίο εκφράζει την πιθανότητα κάθε σταδίου ύπνου να είναι το σωστό.
- Το ***σφάλμα της διασταυρούμενης εντροπίας*** (cross-entropy loss), χρησιμοποιείται ως το κριτήριο του μοντέλου για την οπισθοδιάδοση (back-propagation) κατά την διάρκεια της εκπαίδευσης.

Η αρχιτεκτονική του δικτύου φαίνεται στην Εικόνα 0.3.1.

Μέγεθος Δέσμης (batch size)

Τα νευρωνικά δίκτυα είναι ιδιαίτερα απαιτητικά λόγω των βαριών υπολογιστικών πράξεων που καλούνται να πραγματοποιήσουν κατά την διάρκεια της εκπαίδευσης, επομένως η παραλληλοποίηση των υπολογισμών αυτών έχει αποτελέσει ένα κομβικό σημείο εστίασης της τεχνολογίας τους και των εξελίξεων του λογισμικού. Εφόσον τα δεδομένα που χρησιμοποιούνται στα νευρωνικά δίκτυα εκφράζονται ως πολυδιάστατοι πίνακες, η μονάδα επεξεργασίας γραφικών (graphic processing unit - GPU) χρησιμοποιείται για τους χειρισμούς τους. Οι GPU έχουν αρχικά σχεδιαστεί για την γραφική διεπαφή με τον υπολογιστή, και συγκεκριμένα για την βελτιστοποίηση της επεξεργασίας των εικόνων, οι οποίες έχουν μια παρεμφερή αναπαράσταση μέσω πινάκων, καθιστώντας τις μονάδες GPU ιδανικούς υποψηφίους για τους βαρείς υπολογισμούς των νευρωνικών δικτύων. Τα δεδομένα εκπαίδευσης τυπικά δίνονται ένα-ένα ανά σειρά σαν είσοδος στο νευρωνικό δίκτυο και στην συνέχεια εφαρμόζεται η οπισθοδιάδοση του σφάλματος για την εκπαίδευση του δικτύου μέσω της βελτίωσης των βαρών του. Όμως, η διαδικασία αυτή μπορεί να επιταχυνθεί με μια πιο προηγμένη προσέγγιση, η οποία εκμεταλλεύεται όλη την υπολογιστική δύναμη των μονάδων GPU. Έτσι, τα δεδομένα ομαδοποιούνται σε *δέσμες* ενός προκαθορισμένου μεγέθους, και μια δέσμη δίνεται σε κάθε βήμα της εκπαίδευσης του δικτύου. Κατά την διαδικασία αυτή, όλα τα δεδομένα εκπαίδευσης της δέσμης διέρχονται πρώτα από το δίκτυο και το συνολικό σφάλμα της δέσμης υπολογίζεται πριν σταλεί με την διαδικασία της οπισθοδιάδοσης σφάλματος (back-propagation) για την βελτίωση των βαρών του δικτύου. Συνεπώς, το δίκτυο "βλέπει" τα δεδομένα ολόκληρης της δέσμης προτού αλλάξει τα βάρη του, που επηρεάζουν τους υπολογισμούς του. Αυτό όμως κρύβει μια παγίδα, καθώς, σε περίπτωση που το σύνολο δεδομένων είναι σχετικά μικρό, ένα μεγαλύτερο μέγεθος δέσμης θα οδηγούσε σε μια αδυναμία σύγκλισης του μοντέλου, καθώς θα "έβλεπε" ολόκληρο το σύνολο δεδομένων σε πολύ λίγες επαναλήψεις, χωρίς να έχει αρκετό χρόνο να εκπαιδευτεί και να βελτιώσει τα βάρη του μέσω της διαδικασίας back-propagation. Επίσης, έχει παρατηρηθεί ότι δέσμες μεγάλου μεγέθους τείνουν να μην γενικεύουν τόσο καλά στα δεδομένα δοκιμής του δικτύου, καθώς εντοπίζουν οξεία ελάχιστα αντί για πιο επίπεδα ελάχιστα σημεία [Kes+16]. Αυτή η ιδιότητα του μεγέθους της δέσμης απαιτείται να ισορροπήσει με την απόδοση της εκπαίδευσης του μοντέλου και συνεπώς το μέγεθος της δέσμης εξαρτάται σε μεγάλο βαθμό από την κάθε περίπτωση δεδομένων ξεχωριστά.

Στην δική μας περίπτωση, δοκιμάζοντας μεγέθη δέσμης 8, 16, 32, 64, η καλύτερη τιμή για την απόδοση του δικτύου συνδιαστικά με τον χρόνο εκπαίδευσης είναι **32**.

Εκπαιδεύοντας το μοντέλο

Το μοντέλο δέχεται σαν είσοδο ζευγάρια σειρών συνεχών χρονικών τμημάτων ενός συγκεκριμένου μήκους και τις αντίστοιχες ετικέτες τους με το στάδιο του ύπνου στο οποίο βρίσκονται. Ύστερα από δοκιμές με διάφορα μήκη σειρών μεταξύ [5, 10, 20, 30] χρονικών βημάτων ανά τμήμα, παρατηρούμε ότι τα καλύτερα αποτελέσματα προέρχονται από χρονικά παράθυρα 30 χρονικών βημάτων (δηλαδή 15 συνεχόμενα λεπτά δειγμάτων). Στην συνέχεια, το μοντέλο προβλέπει το στάδιο ύπνου της τελευταίας εποχής της χρονοσειράς, λαμβάνοντας υπ' όψιν τις δοσμένες εποχές που προηγούνται. Κατ' αυτόν τον τρόπο, τα δεδομένα χωρίζονται σε επικαλυπτόμενα παράθυρα 30-εποχών, έτσι ώστε να περιλαμβάνουν όλες τις ετικέτες PSG για την εκπαίδευση του δικτύου, εκτός από τις πρώτες 29, οι οποίες δεν έχουν αρκετές εποχές να προηγούνται για να σχηματιστεί σωστά η χρονοσειρά. Για την προετοιμασία των δεδομένων, ύστερα από τον σχηματισμό των χρονοσειρών 30-εποχών, αυτές αναμειγνύονται και χωρίζονται σε σύνολα εκπαίδευσης, αξιολόγησης και δοκιμής, σε ποσοστό 80-10-10%. Μια συγκεκριμένη τιμή δίνεται για την συνάρτηση ανάμειξης των δεδομένων (shuffling seed), η οποία διατηρείται σταθερή κάθε φορά, διασφαλίζοντας τον ίδιο διαχωρισμό των δεδομένων για την εκπαίδευση και αξιολόγηση του κάθε μοντέλου, έτσι ώστε να υπάρχει μέτρο σύγκρισης μεταξύ των διαφορετικών μοντέλων που δοκιμάζονται. Για να εξασφαλιστεί ότι το ποσοστό κάθε σταδίου ύπνου κατανείμεται σε ισάξιες ποσότητες μεταξύ όλων των υπο-ομάδων (εκπαίδευσης, αξιολόγησης και δοκιμής) του δικτύου, χρησιμοποιούνται τα βάρη των σταδίων του ύπνου ως παράμετροι για τον ισοσταθμισμένο διαχωρισμό των δεδομένων.

Κατηγορίες πειραμάτων

Στην εργασία αυτή ελέγχονται τέσσερις κατηγορίες πειραμάτων για τα διαφορετικά στάδια ύπνου, τα οποία είναι ως εξής:

1. **Sleep - Wake**
2. **Wake - REM - NREM**
3. **Wake - Light - Deep - REM** (light = N1 & N2, deep = N3 & N4)
4. **Wake - N1 - N2 - N3 - REM** (N3 = true N3 & N4, όπως περιγράφεται στο 1.3.1)

Το στάδιο ύπνου N4 έχει ομαδοποιηθεί με το στάδιο N3 για όλα τα πειράματα, έτσι όπως είναι αποδεκτό ως σύμβαση και προτείνεται σε πολλές άλλες εργασίες, οδηγώντας σε ένα σύνολο πέντε σταδίων του ύπνου.

Παράμετροι εκπαίδευσης

Για την εκπαίδευση του νευρωνικού δικτύου LSTM πρέπει να ελεγχθούν ορισμένες παράμετροι, για τις οποίες η επιλογή κατάλληλων τιμών επηρεάζει την απόδοση του δικτύου. Ελέγχθηκαν οι παρακάτω τιμές παραμέτρων και με πιο έντονα γράμματα φαίνονται αυτές που έδιναν τα βέλτιστα αποτελέσματα για το δίκτυο στις περισσότερες περιπτώσεις:

- αριθμός επιπέδων του δικτύου: **2** ή 3
- αριθμός βημάτων χρονοσειράς: 5 - 10 - 20 - **30**
- dropout για το LSTM: **0.5** ή 0 (δηλαδή χωρίς dropout. Το dropout εκφράζει την πιθανότητα το βάρος ενός κόμβου επίτηδες να μηδενιστεί κατά την διάρκεια της εκπαίδευσης του δικτύου, αποτρέποντας την περίπτωση *overfitting*)

Table 1: Οι παράμετροι που βελτιστοποιούν το LSTM μοντέλο για τα σύνολα δεδομένων της Walch και MESA, για όλες τις κατηγορίες σταδίων του ύπνου. 2-class: Sleep-Wake, 3-class: Sleep-NREM-REM, 4-class: Sleep-Light-Deep-REM, 5-class: Wake-N1-N2-N3-REM.

	Walch				**MESA**			
Αριθμός Κλάσεων Προβλήματος	**2-class**	**3-class**	**4-class**	**5-class**	**2-class**	**3-class**	**4-class**	**5-class**
Αριθμός βημάτων χρονοσειράς	30	30	30	30	30	30	30	30
dropout	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
Ρυθμός εκμάθησης (learning rate)	0.0001	0.001	0.001	0.001	0.0001	0.0001	0.0001	0.0001
Αριθμός επιπέδων LSTM δικτύου	2	2	2	2	2	2	2	1
Μέγεθος κρυφών επιπέδων LSTM	512	512	512	512	512	512	512	512
Μέγεθος δέσμης	32	32	32	32	32	32	32	32

- ρυθμός εκμάθησης: **0.001** - 0 0001 - 0.00001
- αριθμός εποχών εκπαίδευσης: δοκιμάστηκαν έως 1000 εποχές, με πολλές περιπτώσεις να έχουν καλύτερα αποτελέσματα για το δίκτυο στις 800 εποχές.
- χρονοπρογραμματιστής ρυθμού εκμάθησης: ελαττώνοντας τον ρυθμό εκμάθησης κατά μια τάξη μεγέθους μετά από μια ορισμένη εποχή εκπαίδευσης δεν παρατηρήθηκε ότι προσφέρει κάποια ιδιαίτερη βελτίωση στο δίκτυο, επομένως δεν εισήχθη σαν παράμετρος στα τελικά μοντέλα.
- μέγεθος κρυφών επιπέδων του LSTM: 256 - **512**
- μέγεθος δέσμης: **32**
- loss_weights: όταν η συγκεκριμένη παράμετρος ορίζεται ως αληθής, τα βάρη των κλάσεων χρησιμοποιούνται σαν παράμετροι για το σφάλμα του κριτηρίου (Cross-Entropy Loss).

Οι τελικές παράμετροι που χρησιμοποιούνται στα μοντέλα της κάθε κατηγορίας σταδίων ύπνου φαίνονται αναλυτικά στον Πίνακα 1.

Πειραματικά αποτελέσματα Walch

Τα πειραματικά αποτελέσματα από την εκπαίδευση του απλού αμφίδρομου LSTM μοντέλου στα δεδομένα της Walch, φαίνονται συνοπτικά στον Πίνακα 2, συνοπτικά μαζί με τα αποτελέσματα των υπόλοιπων πειραμάτων των LSTM μοντέλων. Παρόλο που τα μοντέλα φαίνεται να εκπαιδεύονται ήδη από λίγες μόνο επαναλήψεις,η καλύτερη ακρίβεια (accuracy) επιτυγχάνεται περίπου για την επανάληψη 800. Η ακρίβεια που επιτυγχάνεται με το προτεινόμενο LSTM μοντέλο και τα δεδομένα της Walch με τα αντίστοιχα εξαχθέντα χαρακτηριστικά, υπερβαίνει τις τιμές της εργασίας της Walch, στην οποία απλές τεχνικές μηχανικής μάθησης εφαρμόζονται, χωρίς να λαμβάνεται υπ' όψιν η ιδιαίτερη φύση του προβλήματος, στο οποίο υπάρχει μια χρονολογική εξάρτηση μεταξύ των μεμονωμένων σημείων του συνόλου δεδομένων. Επίσης παρατηρείται ότι η ακρίβεια των αποτελεσμάτων υπερβαίνει αυτήν άλλων εργασιών που χρησιμοποιούν παρόμοια δεδομένα από φορητές συσκευές και έξυπνα ρολόγια. Παρόλο που δεν φτάνει το ύψος των τιμών εργασιών που χρησιμοποιούν σήματα προερχόμενα από μετρήσεις PSG, αυτό δεν είναι αποθαρρυντικό, καθώς τέτοιου είδους δεδομένα εμπεριέχουν πολύ πιο πυκνή πληροφορία και διαφορετικής μορφής, όπως το ότι χρησιμοποιούνται πολύ περισσότερα κανάλια καταγραφής σημάτων, όπως επίσης υπάρχουν περισσότεροι συμμετέχοντες αφού η τεχνολογία αυτή είναι πολύ πιο παλιά.

Αξιολόγηση του BiLSTM μοντέλου της Walch πάνω στα MESA δεδομένα

Εφόσον τα χαρακτηριστικά που εξάγονται για το σύνολο δεδομένων της Walch και της MESA εκφράζουν τις ίδιες φυσικές τιμές, το μοντέλο που έχει εκπαιδευτεί με τα δεδομένα τις Walch αξιολογείται στον τρόπο που αποδίδει στα δεδομένα της MESA, κάτι το οποίο προτείνεται ήδη στην εργασία της [Wal+19]. Κατ' αυτόν τον τρόπο ελέγχεται η δυνατότητα γενίκευσης του προτεινόμενου μοντέλου σε δεδομένα που δεν έχει χειριστεί κατά την διάρκεια της εκπαίδευσής του. Για τα πειράματα επιλέγονται οι πρώτοι 188 συμμετέχοντες, όπως γίνεται και στην προαναφερθείσα εργασία. Στον Πίνακα 2 φαίνονται συνοπτικά τα αποτελέσματα των μοντέλων για όλα τα στάδια του ύπνου πάνω στο σύνολο δεδομένων της MESA. Όπως μπορεί να παρατηρηθεί, η απόδοση των μοντέλων δεν είναι τόσο καλή σε σχέση με το σύνολο δεδομένων της Walch, καθώς τα μοντέλα έχουν ήδη χειριστεί δεδομένα του ίδιου είδους κατά την διάρκεια της εκπαίδευσής τους, που προέρχονται από τα ίδια

Table 2: Η ακρίβεια και οι τιμές των F1-scores των καλύτερων LSTM μοντέλων από όλες τις κατηγορίες σταδίων του ύπνου παρουσιάζονται συγκεντρωτικά, για τα σύνολα δεδομένων της Walch και MESA.

		Sleep-Wake	**REM-NREM**	**Light-Deep**	**All**
Walch					
BiLSTM	**F1-score**	0.75	0.81	0.74	0.69
	Accuracy	0.94	0.90	0.80	0.79
BiLSTM on MESA	**F1-score**	0.67	0.46	0.38	0.30
	Accuracy	0.76	0.60	0.48	0.41
CNN-BiLSTM	**F1-score**	0.60	0.50	0.60	0.50
	Accuracy	0.80	0.56	0.63	0.58
MESA					
BiLSTM	**F1-score**	0.73	0.60	0.54	0.44
	Accuracy	0.78	0.66	0.63	0.63
CNN-BiLSTM	**F1-score**	0.88	0.80	0.75	0.62
	Accuracy	0.88	0.82	0.80	0.73

εργαλεία μετρήσεων. Για τα προβλήματα 2 και 3 κλάσεων, τα μοντέλα αποδίδουν αρκετά καλά, επιτυγχάνοντας ακρίβεια πάνω από 60%. Όμως, όσο πιο περίπλοκο γίνεται το πρόβλημα της κατηγοριοποίησης τόσο πέφτει η απόδοση του μοντέλου σε νέα δεδομένα, που προέρχονται από άλλες πηγές καταγραφής, στην περίπτωση του συνόλου MESA από PSG και ακτιγραφία. Επίσης, μπορούμε να παρατηρήσουμε ότι οι κλάσεις που δεν έχουν τόσο έντονη παρουσία στο σύνολο δεδομένων της εκπαίδευσης, λόγω μικρότερης διάρκειάς τους κατά τον ύπνο, είναι και αυτές που επιτυγχάνουν μικρότερη ακρίβεια και τοποθετούνται εσφαλμένα στις πιο εξέχουσες κλάσεις.

Πειραματικά αποτελέσματα MESA

Επιθυμώντας να μελετήσουμε τις πλήρεις δυνατότητες του προτεινόμενου αμφίδρομου LSTM μοντέλου, το εκπαιδεύουμε σε ολόκληρο το σύνολο δεδομένων της MESA. Για να υπάρχει ένα προς ένα αντιστοιχία, εξάγονται τα ίδια χαρακτηριστικά που χρησιμοποιήθηκαν και στην προηγούμενη ακριβώς ενότητα, για τον έλεγχο της γενίκευσης του LSTM μοντέλου εκπαιδευμένου στα δεδομένα της Walch, δηλαδή **activity counts**, **cosine transform**, **heart rate feature**, **time feature** και οι **PSG ετικέτες**. Η αρχιτεκτονική του LSTM μοντέλου παραμένει ακριβώς η ίδια, όμως οι υπερπαράμετροί του ρυθμίζονται πάλι μέσω πειραμάτων, με τις καλύτερες να παρουσιάζονται στον Πίνακα 1. Είναι φανερό ότι τα καταλληλότερα μοντέλα για τα σύνολα δεδομένων της MESA και της Walch είναι πολύ παρεμφερή, παρόλες τις διαφορές ανάμεσα στα δεδομένα που χρησιμοποιούνται, συγκεκριμένα όταν πάνω από δύο επίπεδα LSTM χρησιμοποιούνται για τα MESA δεδομένα, το μοντέλο γίνεται ασταθές και *υπερπροσαρμόζεται* (overfits) στα δεδομένα. Ο ρυθμός εκμάθησης πρέπει να ελαττωθεί κατά μια τάξη μεγέθους για να αποτρέψει αυτό το φαινόμενο και ναι επιτύχει ένα πιο σταθερό σφάλμα κατά την διάρκεια της εκπαίδευσης. Αυτό οφείλεται είτε στον μεγαλύτερο όγκο δεδομένων σε σχέση με το Walch, είτε στην διαφορετική τους κατανομή ανάμεσα στις κλάσεις που απαιτεί έναν μικρότερο ρυθμό εκμάθησης σε κάθε επανάληψη. Από τα πειραματικά αποτελέσματα που φαίνονται συνοπτικά στον Πίνακα 2 παρατηρείται ότι όσο πιο πολλές κλάσεις ύπνου πρέπει να μάθει το μοντέλο, τόσο ελαττώνεται η ακρίβειά του, όπως και προηγουμένως. Αυτό θα μπορούσε να είναι δείγμα overfitting προς τις πιο ισχυρές κατηγορίες, ή γενικά μια αδυναμία του δικτύου να εντοπίσει τις πιο λεπτομερείες και αμυδρές πλευρές των δεδομένων έτσι ώστε να κατηγοριοποιήσει κάθε στάδιο ύπνου σωστά.

0.3.3 CNN - αμφίδρομη LSTM αρχιτεκτονική

Σαν συνέχεια των πειραμάτων δοκιμάζεται μια παραλλαγή του αμφίδρομου LSTM δικτύου που έχει ήδη προταθεί, έτσι ώστε αντί να δέχεται στην είσοδό του ήδη επεξεργασμένα χαρακτηριστικά, να εκπαιδεύεται στα ανεπεξέργαστα δεδομένα, ενσωματώνοντας μια μονάδα αυτοματοποιημένης εξαγωγής χαρακτηριστικών. Αυτό συνήθως επιτυγχάνεται μέσω ενός συνελικτικού δικτύου (convolutional neural network), το οποίο δέχεται σαν είσοδο τα ακατέργαστα δεδομένα, και μέσω συνελικτικών πράξεων πάνω σε αυτά, εξάγει αυτόματα ορισμένα χαρακτηριστικά, τα οποία στην συνέχεια προωθεί στο LSTM δίκτυο για να ακολουθήσει η διαδικασία της εκμάθησης των σταδίων του ύπνου. Καθώς τα δεδομένα του καρδιακού παλμού διαφέρουν από αυτά της επιτάχυνσης του καρπού, τόσο για τα Walch δεδομένα όσο και για τα MESA, εφαρμόζονται ελαφρώς διαφορετικές αρχιτεκτονικές για την αυτοματοποιημένη εξαγωγή χαρακτηριστικών για το καθένα από αυτά, όπως περιγράφεται στην συνέχεια.

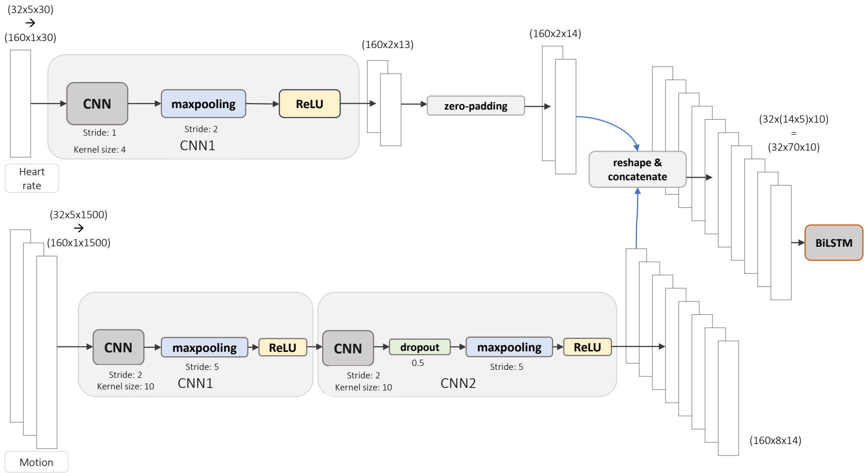


Figure 0.3.2: Η προτεινόμενη συνελικτική αρχιτεκτονική για την αυτοματοποιημένη εξαγωγή χαρακτηριστικών από τα ανεπεξέργαστα δεδομένα καρδιακού παλμού και επιτάχυνσης του καρπού των δεδομένων της Walch. Τα χαρακτηριστικά δίνονται σαν είσοδος στο υπόλοιπο αμφίδρομο δίκτυο μακροχρόνιας-βραχυπρόθεσμης μνήμης, ώστε να εκπαιδευτεί στο πρόβλημα της ταξινόμησης των σταδίων του ύπνου. Εφαρμόζεται η *μέθοδος κατανεμημένου χρόνου* (time distributed method), όπου το μήκος της χρονικής ακολουθίας ευθυγραμμίζεται με το μέγεθος της δέσμης για καλύτερη υπολογιστική απόδοση.

CNN - αμφίδρομο LSTM στα Walch δεδομένα

Ακολουθώντας τα βήματα προετοιμασίας των ανεπεξέργαστων δεδομένων που περιγράφονται στην Ενότητα 0.3.1, εξάγονται χρονικά συνεχείς σειρές για τον καρδιακό παλμό και την ταχύτητα του καρπού ξεχωριστά, που όμως έχουν ένα προς ένα αντιστοιχία ανά εποχή και PSG ετικέτα. Το μεγαλύτερο εμπόδιο στην δομή των συνελικτικών δικτύων είναι ότι, σε αντίθεση με τα *ανακυκλούμενα νευρωνικά δίκτυα* (RNN) στα οποία ανήκει και το LSTM, δέχονται είσοδο συγκεκριμένου μήκους, που ορίζεται από την κατασκευή του δικτύου. Προκειμένου να επιτευχθεί ομοιόμορφη διάσταση εισόδου για τις ακολουθίες των δεδομένων, εφαρμόζεται η μέθοδος της παρεμβολής σε αυτά, ώστε να έχουν σταθερό ρυθμό δειγματοληψίας ανά κατηγορία, όπως ορίζεται από την επίσημη περιγραφή του συνόλου δεδομένων, δηλαδή 50Hz για την επιτάχυνση του καρπού και 1Hz για τον καρδιακό παλμό. Τέλος τα δεδομένα κανονικοποιούνται ανά άτομο, έτσι ώστε να ακολουθούν κανονική κατανομή και να βρίσκονται όλα στον ίδιο χώρο για όλους τους συμμετέχοντες.

Η αρχιτεκτονική του δικτύου που χρησιμοποιείται φαίνεται στο Σχήμα 0.3.2. Δύο ξεχωριστά CNN διαφορετικών διαστάσεων χρησιμοποιούνται για τον καρδιακό παλμό και την επιτάχυνση του καρπού, και οι έξοδοί τους ενώνονται για να δοθούν μαζί σαν είσοδο στο bidirectional-LSTM επίπεδο. Μια ιδιαιτερότητα του προτεινόμενου δικτύου είναι ο τρόπος που δίνονται τα δεδομένα σαν είσοδος στα συνελικτικά δίκτυα, ώστε να είναι πιο αποδοτικά υπολογιστικά, που ονομάζεται *μέθοδος κατανεμημένου χρόνου* (time distributed method). Πρόκειται για ένα επίπεδο *περιτύλιξης* (wrapper layer), το οποίο εφαρμόζει ένα τμήμα ενός δικτύου σε κάθε χρονικό κομμάτι της χρονικής διάστασης μιας ακολουθίας εισόδου, στην περίπτωση αυτή ένα CNN.

CNN - αμφίδρομο LSTM στα MESA δεδομένα

Όμοια με τα δεδομένα της Walch, τα ανεπεξέργαστα MESA Δεδομένα υπόκεινται την ίδια διαδικασία διαχωρισμού σε συνεχή χρονικά τμήματα. Μια διαφορά ανάμεσα στα δύο σύνολα δεδομένων είναι ότι το χαρακτηριστικό της ακτιγραφίας παρέχεται σε ήδη επεξεργασμένη μορφή, καθώς οι τιμές του έχουν επισημειωθεί χειροκίνητα ανά χρονικό παράθυρο 30 δευτερολέπτων (εποχή), επομένως δεν χρειάζεται κάποια περαιτέρω εξαγωγή χαρακτηριστικών για αυτό. Ο καρδιακός παλμός έχει ήδη σταθερό ρυθμό δειγματοληψίας, επομένως εισέρχεται απευθείας σε ένα CNN δίκτυο προσαρμοσμένο στις διαστάσεις του, με την μέθοδο κατανεμημένου χρόνου όπως περιγράφεται παραπάνω, για την αυτόματη εξαγωγή χαρακτηριστικών. Η αρχιτεκτονική που χρησιμοποιείται σε αυτήν την περίπτωση φαίνεται στο σχήμα 0.3.3.

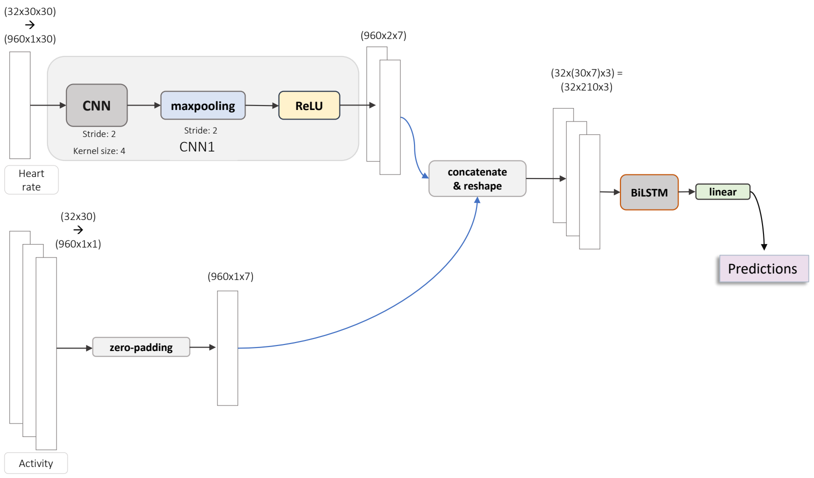


Figure 0.3.3: Η προτεινόμενη συνελικτική αρχιτεκτονική για την αυτοματοποιημένη εξαγωγή χαρακτηριστικών από τα ανεπεξέργαστα MESA δεδομένα επιτάχυνσης του καρπού. Τα χαρακτηριστικά δίνονται σαν είσοδο στο υπόλοιπο αμφίδρομο δίκτυο μακροχρόνιας-βραχυπρόθεσμης μνήμης, ώστε να εκπαιδευτεί στο πρόβλημα της ταξινόμησης των σταδίων του ύπνου. Εφαρμόζεται η *μέθοδος κατανεμημένου χρόνου* (time distributed method), όπου το μήκος της χρονικής ακολουθίας ευθυγραμμίζεται με το μέγεθος της δέσμης για καλύτερη υπολογιστική απόδοση.

Table 3: Οι παράμετροι που βελτιστοποιούν το LSTM τμήμα του μοντέλου CNN - bidirectional LSTM για τα σύνολα δεδομένων της Walch και MESA, για όλες τις κατηγορίες σταδίων του ύπνου. 2-class: Sleep-Wake, 3-class: Sleep-NREM-REM, 4-class: Sleep-Light-Deep-REM, 5-class: Wake-N1-N2-N3-REM.

	Walch				**MESA**			
Αριθμός Κλάσεων Προβλήματος	**2-class**	**3-class**	**4-class**	**5-class**	**2-class**	**3-class**	**4-class**	**5-class**
Αριθμός βημάτων χρονοσειράς	30	30	30	30	30	30	30	30
dropout	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
Ρυθμός εκμάθησης (learning rate)	0.0001	0.001	0.001	0.001	0.0001	0.0001	0.0001	0.0001
Αριθμός επιπέδων LSTM δικτύου	2	2	2	2	1	1	1	1
Μέγεθος κρυφών επιπέδων LSTM	512	512	512	512	512	512	512	512
Μέγεθος δέσμης	32	32	32	32	32	32	32	32

Οι βέλτιστες τιμές παραμέτρων για τα CNN - bidirectional LSTM δίκτυα για τα δύο σύνολα δεδομένων φαίνονται στον Πίνακα 3.

Πειραματικά αποτελέσματα

Τα πειραματικά αποτελέσματα και για τα δύο σύνολα δεδομένων, για την αμφίδρομη LSTM αρχιτεκτονική με αυτόματη εξαγωγή χαρακτηριστικών παρουσιάζονται περιληπτικά στον Πίνακα 2.

Όσον αφορά τα Walch δεδομένα, παρατηρείται ότι η τιμή της ακρίβειας παραμένει σχετικά σταθερή ανάμεσα στις διαφορετικές κατηγορίες σταδίων του ύπνου. Παράλληλα, παρατηρώντας την πιο λεπτομερή αναφορά ταξινόμησης (classification report) στον Πίνακα 4.9 φαίνεται ότι τα στάδια "wake" για όλες τις κατηγορίες είναι πιο δύσκολο να προβλεφθούν. Η συνολική τιμή της ακρίβειας (accuracy) ανά κατηγορία σταδίων του ύπνου ελαττώνεται σε σχέση με το βασικό LSTM μοντέλο που δέχεται σαν είσοδο τα ήδη εξαγμένα χαρακτηριστικά. Αυτό υποδεικνύει ότι η προτεινόμενη μέθοδος αυτοματοποιημένης εξαγωγής χαρακτηριστικών μέσω ενός συνελικτικού δικτύου υπολείπεται στην δυνατότητα του λεπτομερή εντοπισμού των σωστών χαρακτηριστικών των δεδομένων για τον διαχωρισμό μεταξύ των διαφόρων σταδίων του ύπνου, σε σχέση με την μέθοδο εξαγωγής που προτείνεται από την Walch. Μια πιθανή αιτιολόγηση της παρατήρησης αυτής είναι ότι ο τρόπος που έχουν συλλεχθεί τα δεδομένα της Walch (σε σχέση με τα όργανα που μετρήθηκαν, και τον ρυθμό δειγματοληψίας τους), αποδίδει καλύτερα στην εξαγωγή χαρακτηριστικών από κλασικές μεθόδους επεξεργασίας σήματος που λειτουργούν στοχευμένα για τον εντοπισμό χρήσιμων πληροφοριών, σε σχέση με τις μαθηματικές πράξεις που εφαρμόζονται εσωτερικά σε ένα νευρωνικό δίκτυο αυτόματης εξαγωγής χαρακτηριστικών.

Σχετικά με τα δεδομένα της MESA, υπάρχει μια εμφανής βελτίωση στις τιμές της ακρίβειας και του F1-score, σε σχέση με το LSTM μοντέλο με τα ήδη επεξεργασμένα δεδομένα. Παρατηρούμε ότι υπάρχει και πάλι μια συσχέτιση της τιμής της ακρίβειας και του F1-score με την περιπλοκότητα του προβλήματος, δηλαδή όσο περισσότερα είναι τα στάδια που πρέπει να προβλεφθούν, τόσο ελαττώνεται και η τιμή του accuracy. Οι τιμές των αποτελεσμάτων για την ακρίβεια κυμαίνονται γύρω από το 80% και είναι συγκρίσιμες με τα αποτελέσματα του απλού LSTM μοντέλου με τα ήδη επεξεργασμένα δεδομένα της Walch. Η παρατήρηση αυτή είναι ενδιαφέρουσα καθώς διαφαίνεται ότι διαφορετικά σύνολα δεδομένων αποδίδουν καλύτερα σε διαφορετικές προσεγγίσεις επεξεργασίας τους και διαφορετικές αρχιτεκτονικές δικτύων. Στην συγκεκριμένη περίπτωση, το σύνολο δεδομένων της Walch ανταποκρίνεται καλύτερα στην επεξεργασία χαρακτηριστικών μέσω κλασικών τεχνικών επεξεργασίας σήματος και στην συνέχεια εισαγωγή τους σε ένα απλό αμφίδρομο αναδρομικό δίκτυο, ενώ το σύνολο δεδομένων MESA αποδίδει καλύτερα σε μια βαθιά αρχιτεκτονική όπου η εξαγωγή χαρακτηριστικών ενσωματώνεται στο πρώτο από τα επίπεδα του δικτύου, ενώ ακολουθεί ένα παρόμοιο αμφίδρομο αναδρομικό δίκτυο για το πρόβλημα της ταξινόμησης των σταδίων του ύπνου. Πιθανότατα ρόλο σε αυτό το γεγονός διαδραματίζει και ο όγκος των δειγμάτων στα δύο σύνολα δεδομένων, καθώς τα δεδομένα της Walch είναι πολύ λιγότερα σε πλήθος και ενδεχομένως δεν επαρκούν για να εκπαιδεύσουν ένα βαθύτερο νευρωνικό δίκτυο, σε αντίθεση με τα MESA δεδομένα.

0.4 SeqSleepNet Αρχιτεκτονική

Στο δεύτερο σκέλος της εργασίας ελέγχεται μια διαφορετική αρχιτεκτονική νευρωνικών δικτύων, η οποία αρχικά εισήχθη για την ταξινόμηση σταδίων του ύπνου από σήματα που προέρχονται από πολυσομνογράφημα (PSG), και συγκεκριμένα ηλεκτροεγκεφαλογράφημα, ηλεκτρομυογράφημα και ηλεκτροοφθαλμογράφημα [Pha+19]. Η κύρια διαφορά ανάμεσα σε τέτοιου είδους σήματα και αυτά που προέρχονται από φορητές συσκευές όπως το έξυπνο ρολόι, είναι ότι τα πρώτα είναι πολύ πιο πυκνά. Συνήθως υπάρχουν πολύ περισσότεροι αισθητήρες για την καταγραφή τους, με περισσότερα κανάλια στο κάθε βιοσήμα και με πολύ μεγαλύτερο ρυθμό δειγματοληψίας, συνεπώς υπάρχει περισσότερος όγκος πληροφορίας για την εκπαίδευση του δικτύου. Επομένως, ο στόχος εδώ είναι να εξεταστεί η αποτελεσματικότητα ενός τέτοιου είδους δικτύου σε δεδομένα που προέρχονται από φορητές συσκευές, οι οποίες διαφέρουν κατά πολύ από τον εξοπλισμό και τις μετρήσεις που επιτυγχάνονται σε εργαστήριο σε μια *μελέτη ύπνου.*

0.4.1 SeqSleepNet μοντέλο αναφοράς

Στο βασικό μοντέλο, που προτείνεται στην προαναφερθείσα εργασία, η χρονική φύση των δεδομένων λαμβάνεται υπ' όψιν, θεωρώντας τα ως μια χρονική ακολουθία, και υιοθετώντας μια αρχιτεκτονική **many-to-many**. Τέτοιου είδους αρχιτεκτονικές ανήκουν στην κατηγορία των *αναδρομικών νευρωνικών δικτύων* (RNN), στα οποία τα μοντέλα λαμβάνουν μια ακολουθία εισόδων και παράγουν μια ακολουθία εξόδων, ίδιου ή διαφορετικού μήκους. Το πρόβλημα των σταδίων του ύπνου γίνεται αντιληπτό ως μια ταξινόμηση ακολουθιών, στο οποίο μια ακολουθία πολλαπλών εποχών δίνεται σαν είσοδος και όλες οι ετικέτες τους ταξινομούνται ταυτόχρονα. Στην αρχική εργασία η εκπαίδευση γίνεται με δεδομένα από 200 άτομα, τα οποία είναι διαχωρισμένα σε 180 - 10 - 10 υποσύνολα εκπαίδευσης - επαλήθευσης και δοκιμής αντίστοιχα. Τα δεδομένα αποτελούνται από ένα PSG σήμα τριών καναλιών, συγκεκριμένα EEG, EMG και EOG, με αρχικό ρυθμό δειγματοληψίας 256Hz, οποίος μέσω ***downsampling*** ελαττώνεται στα 100Hz. Από τα σήματα εξάγονται λογαριθμικά σπεκτρογράμματα ανά χρονικά παράθυρα 30 δευτερολέπτων (εποχές) ανά κανάλι, και εισάγονται στο δίκτυο για εκπαίδευση. Επομένως, η είσοδος του αρχικού μοντέλου αποτελείται από μια εικόνα χρόνου-συχνότητας τριών καναλιών.

Βασική αρχιτεκτονική

Η αρχιτεκτονική του βασικού δικτύου αποτελείται από τρεις διαφορετικές δομές, και παρουσιάζεται στην Εικόνα 0.4.1.

1. Αρχικά υπάρχει ένα επίπεδο *φίλτρων* (filterbank layer) για κάθε χαρακτηριστικό, για την εκμάθηση των διάφορων *ζωνών συχνοτήτων*. Λαμβάνοντας ως είσοδο εικόνες στον χώρο του χρόνου-συχνότητας, το δίκτυο αναμένεται να δώσει έμφαση στις πιο χρήσιμες ζώνες για την ζητούμενη εργασία και να αμβλύνει τις λιγότερο εμφανείς. Κάθε επίπεδο φίλτρων μοντελοποιείται από ένα *πλήρως συνδεδεμένο (γραμμικό)*

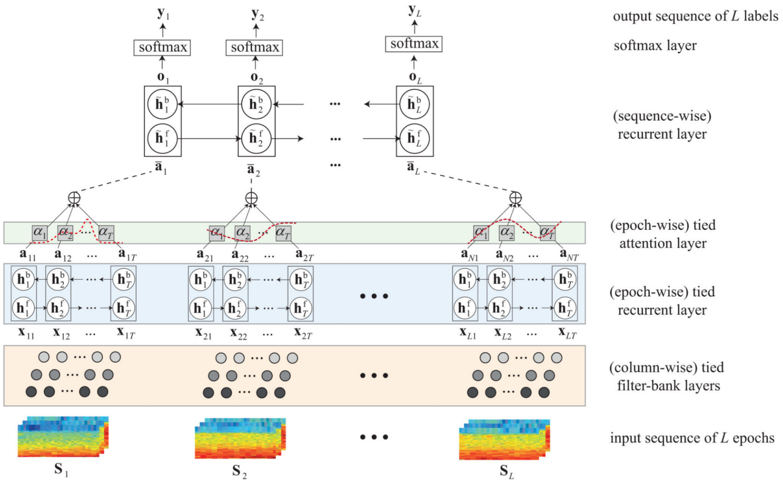


Figure 0.4.1: Η βασική αρχιτεκτονική SeqSleepNet, πρόκειται για ένα από την αρχή-εως-τέλος ιεραρχικό αναδρομικό δίκτυο για την ταξινόμηση σταδίων του ύπνου ακολουθιακά, χρησιμοποιώντας τα σπεκτροκράμματα των PSG σημάτων σαν είσοδο για το δίκτυο [Pha+19].

επίπεδο M κρυφών μονάδων, όπου το M εκφράζει τον αριθμό των φίλτρων. Η εικόνα της εξόδου είναι μικρότερη σε μέγεθος από ό,τι η εικόνα της εισόδου. Στην συνέχεια εφαρμόζεται συνένωση στο επίπεδο της συχνότητας για όλα τα χαρακτηριστικά, οδηγώντας σε μια εικόνα δύο διαστάσεων, η οποία μπορεί να προσληφθεί ως μια ακολουθία από T διανύσματα χαρακτηριστικών $\mathbf{X} = (x_1, x_2, ..., x_T)$, όπου το κάθε $(x_t$ μπορεί να θεωρηθεί ως μια εικόνα-στήλη την χρονική στιγμή t.

2. Στο δεύτερο επίπεδο του δικτύου βρίσκεται ένα αμφίδρομο αναδρομικό νευρωνικό δίκτυο (RNN), συνδυασμένο με έναν *μηχανισμό προσοχής* (attention mechanism), με σκοπό την εκμάθηση των βραχυπρόθεσμων ακολουθιακών χαρακτηριστικών που εκπροσωπούν την κάθε εποχή. Συγκεκριμένα, χρησιμοποιείται ένα δίκτυο GRU (*gated recurrent unit - επαναλαμβανόμενη μονάδα με πύλη*), καθώς περιλαμβάνει λιγότερες παραμέτρους, πράγμα που το καθιστά υπολογιστικά αποδοτικότερο. Τέλος, ο μηχανισμός *προσοχής* χρησιμοποιείται για να ενισχύσει τα πιο χρήσιμα τμήματα της ακολουθίας και να ελαττώσει αυτά που περιέχουν λιγότερη πληροφορία.

   Στον μηχανισμό προσοχής ένα σταθμισμένο διάνυσμα μαθαίνεται αυτόματα ώστε να συνδυάζει τα διανύσματα εξόδων διαφορετικών χρονικών στιγμών σε ένα μοναδικό διάνυσμα χαρακτηριστικών. Έτσι, κάθε εποχή εκπροσωπείται από ένα διάνυσμα χαρακτηριστικών το οποίο προκύπτει από το άθροισμα των διανυσμάτων της χρονοσειράς πολλαπλασιασμένων με το αντίστοιχο attention βάρος τους.

3. Το τρίτο επίπεδο του δικτύου αποτελείται από ένα αμφίδρομο αναδρομικό νευρωνικό δίκτυο, αλλά σε αυτήν την περίπτωση είναι σε επίπεδο ακολουθίας. Στόχος είναι να εντοπίσει όλες τις μακροπρόθεσμες χρονικές πληροφορίες μεταξύ των εποχών της ακολουθίας εισόδου, μοντελοποιώντας την ανα-εποχή ακολουθία των διανυσμάτων χαρακτηριστικών. Και σε αυτήν την περίπτωση χρησιμοποιείται ένα δίκτυο GRU ακολουθώντας την δομή που προτείνεται στο προηγούμενο επίπεδο του δικτύου και λαμβάνει ως είσοδο το διάνυσμα χαρακτηριστικών προσοχής, επιστρέφοντας μια νέα ακολουθία διανυσμάτων.

4. Τέλος, η έξοδος του δεύτερου επιπέδου GRU περνάει από ένα softmax επίπεδο προκειμένου να παραχθούν οι προβλέψεις μιας εποχής να ανήκει σε κάθε στάδιο ύπνου, για όλες τις εποχές της χρονικής ακολουθίας εισόδου. Έτσι, το μοντέλο δίνει σαν έξοδο μια ακολουθία ταξινομήσεων με τις πιθανοτικές κατανομές για κάθε στάδιο ύπνου, για κάθε εποχή της εισόδου.

Για την τελική πρόβλεψη του δικτύου, χρησιμοποιείται ένα σύνολο αποφάσεων και πιθανολογική συνάθροιση, όπως προτείνεται σε προηγούμενη εργασία του [Pha+18], όπου ένα σχήμα πολλαπλασιαστικής συνάθροισης αποδεικνύεται ότι είναι το αποτελεσματικότερο. Το SeqSleepNet είναι ένα δίκτυο πολλαπλών εξόδων, δίνοντας προβλέψεις για όλες τις εποχές που συμμετέχουν σε κάθε ακολουθία εισόδου. Δεδομένου ότι η ακολουθία εισόδου έχει μήκος L, τότε επαυξάνοντάς την κατά μια εποχή κατά την διάρκεια αξιολόγησης με τα δεδομένα δοκιμής, θα οδηγήσει σε ένα σύνολο αποφάσεων L σε κάθε εποχή. Η συνένωση του συνόλου των προβλέψεων για κάθε εποχή σε μια τελική απόφαση ταξινόμησης αποδίδει καλύτερα από ό,τι θα απέδιδαν οι μεμονωμένες προβλέψεις για το στάδιο ύπνου κάθε εποχής.

0.4.2 Σύνολο Δεδομένων Walch

Προετοιμασία δεδομένων

Το πρώτο πείραμα με την αρχιτεκτονική SeqSleepNet γίνεται στα δεδομένα της Walch, ακολουθώντας την ίδια προσέγγιση με την αυθεντική εργασία, δηλαδή εξάγοντας σπεκτρογράμματα από τα ακατέργαστα δεδομένα προκειμένου να δοθούν στο δίκτυο για αυτόματη εξαγωγή χαρακτηριστικών και εκπαίδευση. Προκειμένου να γίνει αυτό, πρέπει να προηγηθεί μια προετοιμασία στα ακατέργαστα δεδομένα της επιτάχυνσης του καρπού και του καρδιακού παλμού, και να εξαχθούν τα αντίστοιχα σπεκτρογράμματα.

Κατ' αρχάς, τα δεδομένα διαχωρίζονται στα σημεία που υπάρχουν κενά στην ακολουθία των εποχών σε τουλάχιστον ένα από τα δύο χαρακτηριστικά, όπως έχει περιγραφεί στην Ενότητα 0.3.1. Οι PSG ετικέτες επίσης επεξεργαζονται όπως περιγράφεται στην προαναφερθείσα ενότητα, απομακρύνοντας τις *-1* τιμές οι οποίες αντιπροσωπεύουν στάδια ύπνου χωρίς ετικέτα. Στην συνέχεια εφαρμόζεται στα δεδομένα η διαδικασία της παρεμβολής έτσι ώστε να έχουν ρυθμό δειγματοληψίας ακριβώς 50Hz για την τρισδιάστατη κίνηση του καρπού και 1Hz για τον καρδιακό παλμό. Τέλος, εξάγονται τα λογαριθμικά σπεκτρογράμματα για όλα τα χαρακτηριστικά. Προκειμένου τα σπεκρτογράμματα να μπορούν να δοθούν απευθείας στο SeqSleepNet μοντέλο, οι διαστάσεις τους πρέπει να είναι όσο και τα αρχικά του δεδομένα εκπαίδευσης, δηλαδή $(29, 129)$ στον χώρο του χρόνου και συχνότητας. Εξάγονται μέσω της βιβλιοθήκης *scipy* της python, είτε πρώτα υπολογίζοντας τον Short-Time Fourier Transform (STFT) και στην συνέχεια λαμβάνοντας το τετραγωνισμένο πλάτος του μετασχηματισμού, είτε απευθείας χρησιμοποιώντας μια συνάρτηση που παρέχεται από την βιβλιοθήκη. Και στις δύο περιπτώσεις, οι παράμετροι των συναρτήσεων επιλέγονται έτσι ώστε το μέγεθος των σπεκτρογραμμάτων να είναι το επιθυμητό.

Πειραματικά αποτελέσματα

Τα δεδομένα της Walch χωρίζονται σε δεδομένα εκπαίδευσης, επαλήθευσης και δοκιμής με ποσοστά 90 - 5 - 5%, χρησιμοποιώντας τα ίδια σύνολα δεδομένων για όλα τα πειράματα. Η επιλογή των συγκεκριμένων ποσοστών διαχωρισμού έγινε θεωρώντας ότι ένα τόσο περίπλοκο και βαθύ νευρωνικό δίκτυο χρειάζεται όσο το δυνατόν περισσότερη πληροφορία για την εκπαίδευσή του, επομένως αυτή είναι μια σύμβαση που εξασφαλίζει αυτήν την απαίτηση, σε αντίθεση με τα πειράματα της προηγούμενης ενότητας. Το SeqSleepNet δεν λαμβάνει υπ' όψιν του την ταυτότητα του ατόμου για κάθε δείγμα κατά την διάρκεια της εκπαίδευσης, είναι δηλαδή ***subject-agnostic***, ωστόσο δεν θεωρείται το μοντέλο αυτό εξ' αρχής καθολικό, ότι δηλαδή μπορεί να εφαρμοστεί επιτυχώς σε νέα άτομα τα οποία δεν έχει δει κατά την διάρκεια της εκπαίδευσης. Έτσι, στα πειράματα που πραγματοποιούνται στην παρούσα εργασία τα δείγματα που περιλαμβάνονται στο σύνολο δοκιμής προέρχονται από άτομα τα οποία περιέχονται επίσης στο σύνολο εκπαίδευσης.

Εφόσον το SeqSleepNet χρησιμοποιεί αρχικά ένα σήμα τριών καναλιών, στο πρώτο πείραμα χρησιμοποιούνται τα δεδομένα της επιτάχυνσης, τα οποία είναι επίσης τρισδιάστατα, για να εξαχθούν τα λογαριθμικά τους σπεκτρογράμματα. Επίσης, ο ρυθμός δειγματοληψίας του χαρακτηριστικού αυτού είναι μια τάξη μεγέθους μεγαλύτερος από ό,τι ο ρυθμός δειγματοληψίας του καρδιακού παλμού, που συνεπάγεται ότι περισσότερη πληροφορία θα περιλαμβάνεται στα σπεκτρογράμματα. Πραγματοποιούνται τρία διαφορετικά πειράματα, χρησιμοποιώντας σαν δεδομένα εκπαίδευσης τα i) λογαριθμικά σπεκτρογράμματα, ii) STFT και iii) τα απλά σπεκτρογράμματα. Δυστυχώς, τα πειραματικά αποτελέσματα υποδεικνύουν ότι, ενώ η ακρίβεια κατά την διάρκεια της εκπαίδευσης του μοντέλου είναι 70% ή και παραπάνω, η ακρίβεια των δεδομένων επαλήθευσης και δοκιμής είναι πολύ χαμηλή.

Στον δεύτερο κύκλο πειραμάτων το μοντέλο εκπαιδεύεται χρησιμοποιώντας τα δεδομένα του καρδιακού παλμού, με πολύ αδύναμα αποτελέσματα. Εφόσον ο ρυθμός δειγματοληψίας είναι 1Hz συγκριτικά με τα αρχικά δεδομένα εκπαίδευσης του SeqSleepNet που ήταν 200Hz, τα λογαριθμικά σπεκτρογράμματα φαίνεται να μην συλλαμβάνουν τις κατάλληλες πληροφορίες για την εκπαίδευση του μοντέλου. Η ακρίβεια κατά την διάρκεια της εκπαίδευσης του μοντέλου παραμένει γύρω στο 50%, που υποδεικνύει ότι το μοντέλο δεν είναι σε θέση να διαχωρίσει τις χρήσιμες πληροφορίες και υπάρχουν μεγάλες διακυμάνσεις που δείχνουν ότι το μοντέλο έχει πιθανώς *υπερ-εκπαιδευτεί* (over-fitted).

0.4.3 Σύνολο Δεδομένων MESA - Παραλλαγή 1

Εφόσον το σύνολο δεδομένων MESA παρέχεται σε ήδη προ-επεξεργασμένη μορφή, δεν μπορεί το SeqSleepNet να εφαρμοστεί απευθείας σε αυτό. Αντ' αυτού προτείνονται δύο παραλλαγές του δικτύου, όπου διαφορετικές μορφές εισόδου χρησιμοποιούνται, αφού τα σπεκτρογράμματα δεν μπορούν να εξαχθούν για το παρόν σύνολο δεδομένων. Η βασική ιδέα της προσέγγισης αυτής είναι ότι τα ήδη προ-επεξεργασμένα χαρακτηριστικά μπορούν απευθείας να εφαρμοστούν στο δεύτερο επίπεδο του δικτύου. Έτσι, αντί να διέλθουν τα σπεκτρογράμματα των ακατέργαστων δεδομένων μέσω του επιπέδου φίλτρων προκειμένου να εντοπιστούν χρήσιμα χαρακτηριστικά, τα ήδη προετοιμασμένα χαρακτηριστικά δίδονται απευθείας στο δεύτερο επίπεδο, που είναι αναδρομικό ανά εποχή, ακολουθώντας με τον μηχανισμό προσοχής (attention layer).

Η πρώτη από τις δύο προτεινόμενες παραλλαγές είναι μια απλοποιημένη εκδοχή του αρχικού SeqSleepNet δικτύου. Τα χαρακτηριστικά του καρδιακού παλμού και της κίνησης του καρπού διαχωρίζονται σε ακολουθίες από χρονικά συνεχή δείγματα ενός συγκεκριμένου επιθυμητού μήκους, τα οποία στην συνέχεια δίνονται απευθείας στο δεύτερο επίπεδο του δικτύου. Στο σύνολο δεδομένων MESA, οι PSG ετικέτες των σταδίων του ύπνου παρέχονται ανά 1 δευτερόλεπτο, επομένως η παρούσα προσέγγιση είναι ανεξάρτητη από τις εποχές των 30 δευτερολέπτων που τυπικά ορίζονται από μια ετικέτα σταδίου ύπνου.

Αρχιτεκτονική και εκπαίδευση δικτύου

Το χαρακτηριστικό του **καρδιακού παλμού** έχει ρυθμό δειγματοληψίας 1Hz, επομένως θεωρείται έτοιμο να δοθεί στο δίκτυο για εκπαίδευση, και δεν χρειάζεται να εφαρμοστούν περαιτέρω βήματα προεπεξεργασίας σε αυτό.

Το χαρακτηριστικό της **επιτάχυνσης** του καρπού αντιθέτως, παρέχεται με 30 δείγματα ανά μια ετικέτα ύπνου, που σημαίνει ότι ένα νευρωνικό δίκτυο επιλογής χαρακτηριστικών μπορεί να εφαρμοστεί στην ακολουθία δεδομένων που αντιστοιχεί σε κάθε ετικέτα σταδίου ύπνου. Ακολουθώντας την προϋπάρχουσα αρχιτεκτονική του SeqSleepNet, τα δεδομένα της επιτάχυνσης διέρχονται από ένα αμφίδρομο αναδρομικό επίπεδο έτσι ώστε να αποκτηθούν κάποιες ενδιάμεσες αναπαραστάσεις, οι οποίες στην συνέχεια διέρχονται από τον μηχανισμό προσοχής για να εξαχθούν τα πιο χρήσιμα χαρακτηριστικά.

Η έξοδος του μηχανισμού προσοχής συνενώνεται με το χαρακτηριστικό του καρδιακού παλμού, και, στην συνέχεια, τα χαρακτηριστικά που αντιπροσωπεύουν κάθε εποχή 30 δευτερολέπτων διέρχονται από ένα αμφίδρομο αναδρομικό δίκτυο σε επίπεδο εποχής έτσι ώστε να αποκτηθούν οι πιθανότητες για κάθε εποχή να ανήκει σε κάθε ένα από τα στάδια ύπνου.

Τέλος, οι πιθανότητες για κάθε εποχή της ακολουθίας διέρχονται από ένα πλήρως συνδεδεμένο γραμμικό επίπεδο και μια *softmax* συνάρτηση ενεργοποίησης ώστε να αποκτηθούν οι τελικές προβλέψεις των σταδίων του ύπνου.

Η αρχιτεκτονική του μοντέλου φαίνεται στο Σχήμα 0.4.2.

0.4.4 Σύνολο Δεδομένων MESA - Παραλλαγή 2

Για την δεύτερη παραλλαγή της αρχιτεκτονικής του SeqSleepNet επιθυμούμε να λάβουμε υπ' όψιν την έννοια των εποχών 30 δευτερολέπτων που τυπικά αντιστοιχούν σε κάθε ετικέτα σταδίου ύπνου. Για να επιτευχθεί αυτό, τα δεδομένα διαχωρίζονται σε χρονικά συνεχή παράθυρα 30 δευτερολέπτων, διατηρώντας το στάδιο ύπνου του τελευταίου χρονικού σημείου ως το αντιπροσωπευτικό για την εποχή, και στην συνέχεια, διαδοχικά παράθυρα (που αντιπροσωπεύουν εποχές) σχηματίζουν την είσοδο του μοντέλου για εκπαίδευση.

Η βασική αρχιτεκτονική του δικτύου παραμένει όπως και στην πρώτη παραλλαγή που παρουσιάστηκε παραπάνω, όμως ο τρόπος που γίνεται ο χειρισμός των δεδομένων αλλάζει, ώστε να εξυπηρετήσει την νέα διάσταση των εποχών 30-δευτερολέπτων για κάθε σημείο δεδομένων.

Αρχιτεκτονική και εκπαίδευση δικτύου

Τα δεδομένα του **καρδιακού παλμού** έχουν μια μοναδική τιμή για κάθε σημείο στην χρονική διάσταση εφόσον ο ρυθμός δειγματοληψίας τους είναι 1Hz, επομένως, όπως και προηγουμένως, έχουν μια διάσταση λιγότερο από ό,τι ο καρδιακός παλμός.

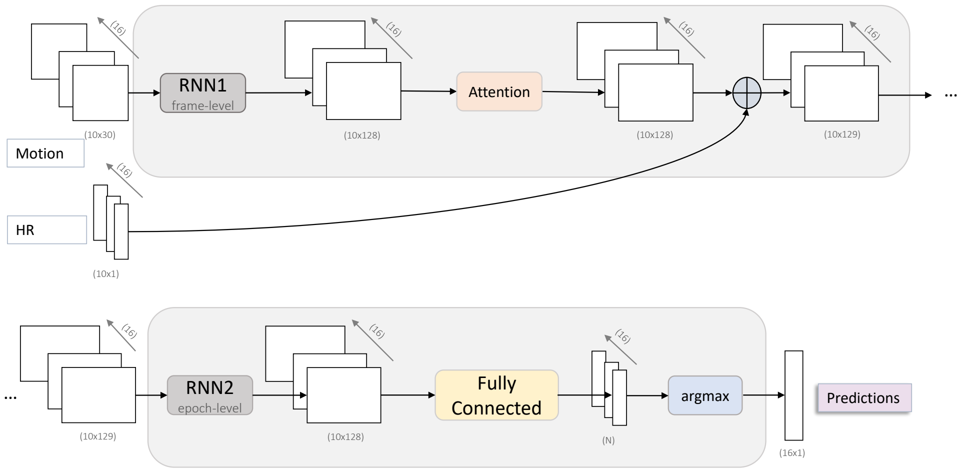


Figure 0.4.2: Η προτεινόμενη πρώτη παραλλαγή για την εφαρμογή του συνόλου δεδομένων MESA στην αρχιτεκτονική του SeqSleepNet. Μια ετικέτα σταδίου ύπνου παρέχεται ανά 1 δευτερόλεπτο, και οι προβλέψεις γίνονται σε χρονικές ακολουθίες επιθυμητού μήκους δευτερολέπτων. Το **N** αναφέρεται στον αριθμό των κλάσεων σε κάθε ένα από τα πειράματα (all: N=5, light-deep: N=4, rem-nrem: N=3, sleep-wake: N=2).

Όμοια με την προηγούμενη μέθοδο, η **επιτάχυνση** διέρχεται από ένα αμφίδρομο αναδρομικό δίκτυο έτσι ώστε να αποκτηθούν οι ενδιάμεσες αναπαραστάσεις των ακολουθιών σε επίπεδο εποχής.

Στην προκείμενη περίπτωση, όμως, εξετάζεται μια ακολουθία σε επίπεδο εποχής σε κάθε βήμα εκπαίδευσης, αποτελούμενη από διαδοχικές ακολουθίες σε επίπεδο παραθύρων 30-δευτερολέπτων. Τα τελικά χαρακτηριστικά που είναι να επιλεχθούν πρέπει να αντιπροσωπεύουν όλη την ακολουθία από N διαδοχικές εποχές. Για να επιτευχθεί αυτό, η έξοδος του πρώτου επιπέδου αμφίδρομου αναδρομικού δικτύου που περιλαμβάνει τις ενδιάμεσες αναπαραστάσεις της κίνησης του καρπού, συνενώνεται με τα αντίστοιχα δεδομένα του καρδιακού παλμού στο επίπεδο της ακολουθίας εποχών, έτσι ώστε στην συνέχεια να διέλθουν από τον μηχανισμό προσοχής. Κατ' αυτόν τον τρόπο, τα πιο σημαντικά χαρακτηριστικά της ακολουθίας σε επίπεδο εποχής κρατούνται για το επόμενο επίπεδο.

Στην συνέχεια, τα δεδομένα διέρχονται από το δεύτερο αμφίδρομο αναδρομικό δίκτυο, παρέχοντας τις τιμές πιθανοτήτων για κάθε στάδιο ύπνου της ακολουθίας, ενώ με την βοήθεια ενός γραμμικού επιπέδου, όπως και παραπάνω, καθώς και με την συνάρτηση ενεργοποίησης προκύπτουν οι τελικές προβλέψεις.

Η αρχιτεκτονική της δεύτερης παραλλαγής φαίνεται στην Εικόνα 0.4.3.

0.4.5 Πειραματικά αποτελέσματα MESA

Για την εκπαίδευση του δικτύου SeqSleepNet χρησιμοποιείται όλο το σύνολο δεδομένων της MESA, με την εξαίρεση 5 ατόμων τα οποία χρησιμοποιούνται αποκλειστικά σαν δοκιμαστικά δεδομένα, στοιχεία των οποίων το μοντέλο δεν έχει ξαναδεί. Τα δεδομένα διαχωρίζονται με τον ίδιο τρόπο σε υποσύνολα εκπαίδευσης, επαλήθευσης και δοκιμής κάθε φορά, για να υπάρχει στα πειράματα συνοχή. Η εκπαίδευση του κάθε μοντέλου είναι απαιτεί ιδιαίτερα πολύ χρόνο, και διαρκεί περισσότερο από 24 ώρες για να ολοκληρωθεί, επομένως δεν έχει εφαρμοστεί η μέθοδος του *cross-validation* (διασταυρούμενη επαλήθευση), παρόλο που αυτή θα ήταν η ορθότερη προσέγγιση επιστημονικά για τον έλεγχο των μοντέλων.

Στατιστική ανάλυση δεδομένων

Για την στατιστική ανάλυση του συνόλου δεδομένων της MESA εξάγονται η μέση τιμή και η τυπική απόκλιση ανά κατηγορία σταδίου ύπνου για κάθε άτομο, για τα χαρακτηριστικά της κίνησης του καρπού και του καρδιακού παλμού ξεχωριστά, και τα αντίστοιχα *violin plots* και *scatter plots* φαίνονται στα Σχήματα 4.8.4, 4.8.3.

Όσον αφορά το χαρακτηριστικό του **καρδιακού παλμού**, από τα *violin plots* φαίνεται ότι, πέρα από τα

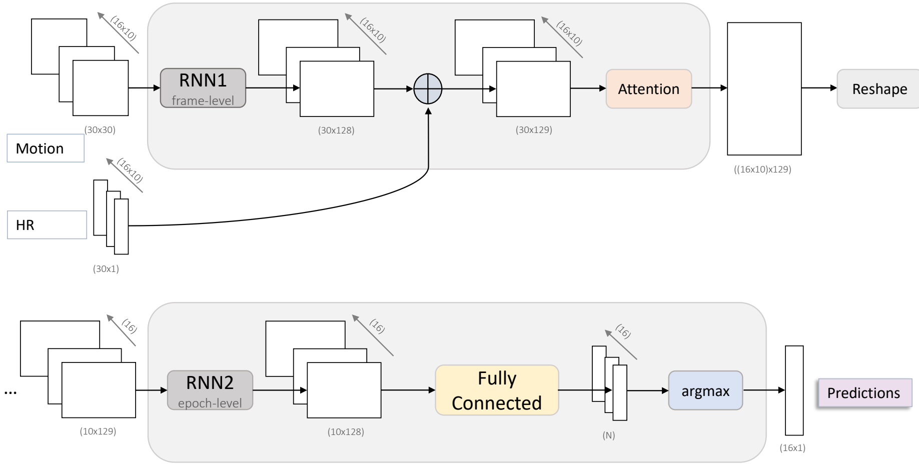


Figure 0.4.3: Η προτεινόμενη δεύτερη και πιο περίπλοκη παραλλαγή του δικτύου SeqSleepNet για την εφαρμογή του συνόλου δεδομένων MESA, όπου για κάθε εποχή 30-δευτερολέπτων θεωρείται ένα στάδιο ύπνου, και η πρόβλεψη γίνεται σε χρονικές ακολουθίες 10 δειγμάτων (εποχών). Το **N** αναφέρεται στον αριθμό των κλάσεων κάθε πειράματος (all: N=5, light-deep: N=4, rem-nrem: N=3, sleep-wake: N=2).

στάδια *wake* και *N4* τα οποία βρίσκονται λίγο πιο χαμηλά, οι μέσες τιμές των υπόλοιπων σταδίων ύπνου (δηλαδή *N1, N2, N3, REM*) έχουν διάμεση τιμή πολύ κοντά στο 60. Αυτό θα μπορούσε να προμηνύει ότι δεν είναι εύκολα διαχωρίσιμα το ένα από το άλλο. Επίσης, ακολουθούν μια παρόμοια κανονική κατανομή χωρίς πολλές παρεκκλίσεις, που σημαίνει ότι υπάρχει μεγάλη πιθανότητα η τιμή του καρδιακού παλμού να βρίσκεται κοντά στον διάμεσο για τα στάδια ύπνου *N1, N2, N3, REM*. Παρόλα αυτά, το στάδιο *wake* ακολουθεί μια σχεδόν διωνυμική κατανομή, η οποία επίσης επιμηκύνεται πολύ στον κατακόρυφο άξονα, που σημαίνει ότι υπάρχουν πολλά σημεία που αποκλίνουν από την μέση τιμή του ανά άτομο. Για το στάδιο ύπνου *N4* η μέση τιμή ανά άτομο ακολουθεί κανονική κατανομή που επίσης είναι στενή στον οριζόντιο άξονα, όμως φαίνεται να υπάρχουν πολλές τιμές που αποκλίνουν και τα δεδομένα δεν είναι συγκεντρωμένα γύρω από την διάμεση τιμή.

Παρατηρώντας την τυπική απόκλιση ανά άτομο μέσω των ***violin plots***, φαίνεται ότι οι τιμές της ακολουθούν πιο ομοιόμορφη κανονική κατανομή μεταξύ των σταδίων του ύπνου, που σημαίνει ότι οι τιμές για κάθε άτομο χωριστά δεν αποκλίνουν πολύ μεταξύ τους. Ορισμένες εξαιρέσεις εντοπίζονται στα στάδια ύπνου *N1, N2, N3, REM*. Το πιο διακριτό από όλα τα στάδια ύπνου είναι το *wake*, στο οποίο η κατανομή είναι πολύ στενή και προεκτείνεται πολύ στον κάθετο άξονα, ακολουθώντας μια σχεδόν διωνυμική κατανομή. Επομένως, για τα δείγματα του σταδίου *wake* υπάρχει μεγάλη απόκλιση μεταξύ των κατανομών ανά άτομο.

Όσον αφορά το χαρακτηριστικό της **κίνησης του καρπού**, φαίνεται ότι η μέση τιμή των σταδίων *N1, N2, N3, REM* ακολουθεί μια σχεδόν μοναδιαία κατανομή, όπου όλα τα δείγματα είναι συγκεντρωμένα πολύ κοντά σε ένα μοναδικό σημείο που είναι ο διάμεσος, και δεν υπάρχουν σχεδόν καθόλου αποκλίσεις. Αντιθέτως, τα στάδια *wake* και *N4* εκτείνονται σε ένα μεγάλο εύρος τιμών. Η σύγκλιση των τιμών των σταδίων *N1, N2, N3, REM* είναι πολύ λογική δεδομένου ότι το χαρακτηριστικό της κίνησης του καρπού είναι ήδη επεξεργασμένο όπως παρέχεται από το σύνολο δεδομένων MESA και τα ακατέργαστα χαρακτηριστικά δεν είναι δημόσια διαθέσιμα.

Η τυπική απόκλιση για το χαρακτηριστικό του καρπού επίσης ακολουθεί μια κανονική κατανομή για όλα τα στάδια ύπνου πέραν του *N4*, το οποίο φαίνεται να έχει όλες τις τιμές του μηδενικές. Υπάρχει μια μικρή ποικιλομορφία μεταξύ των *N1, N2, N3, REM*, των οποίων οι τιμές της τυπικής απόκλισης ανά άτομο είναι πολύ κοντά στο μηδέν, αναδεικνύοντας ότι το χαρακτηριστικό της κίνησης του καρπού είναι πολύ ομοιόμορφο σε κάθε άτομο. Όμως, και πάλι το στάδιο *wake* φαίνεται να περιλαμβάνει πολλές εξαιρέσεις, παρόλο που και αυτό ακολουθεί Γκαουσιανή κατανομή. Όσον αφορά το στάδιο *N4*, ύστερα από λεπτομερή αναζήτηση στο σύνολο δεδομένων MESA, φαίνεται ότι για πολλά άτομα το στάδιο αυτό δεν υπάρχει καν στις μετρήσεις τους, που συνεπάγεται η μέση τιμή να λαμβάνει την τιμή NaN (κενό) και η τυπική απόκλιση να είναι μηδενική. Αυτό δεν αποτελεί εμπόδιο για τα πειράματά μας, καθώς το στάδιο *N4* πάντα συγχωνεύεται με το *N3*, όπως ορίζεται από τις σύγχρονες συμβάσεις, ή και με περισσότερα στάδια, ανάλογα με το πρόβλημα ταξινόμησης που αντιμετωπίζεται κάθε φορά.

Table 4: Η ακρίβεια και η μετρική του Cohen's kappa για κάθε κατηγορία προβλήματος σταδίων του ύπνου, όπως έχουν εξαχθεί από το σύνολο δεδομένων δοκιμής της MESA, για τις δύο παραλλαγές της SeqSleepNet αρχιτεκτονικής. Τα καλύτερα μοντέλα κατά την διάρκεια της εκπαίδευσης αποθηκεύονται και χρησιμοποιούνται για τον σκοπό της δοκιμής της προτεινόμενης αρχιτεκτονικής.

	Modification 1			**Modification 2**		
	Accuracy	**F1-score**	**Kappa**	**Accuracy**	**F1-score**	**Kappa**
All	0.58	0.32	0.33	0.59	0.27	0.34
Light-Deep	0.63	0.35	0.37	0.64	0.35	0.38
REM-NREM	0.69	0.49	0.42	0.7	0.49	0.45
Sleep-Wake	0.78	0.78	0.54	0.8	0.77	0.56

Το *scatter plot* της Εικόνας 4.8.3 επιβεβαιώνει τις παραπάνω παρατηρήσεις, με όλες τις μέσες τιμές των σταδίων *N1, N2, N3, REM* να βρίσκονται πολύ κοντά η μια στην άλλη, ενώ το στάδιο *wake* κυμαίνεται σε ένα μεγαλύτερο εύρος τιμών και το *N4* είναι σχεδόν ανύπαρκτο.

Πειραματικά αποτελέσματα και συζήτηση

Στον Πίνακα 4 παρουσιάζονται συνοπτικά τα πειραματικά αποτελέσματα των δύο παραλλαγών του δικτύου SeqSleepNet, και συγκεκριμένα η ακρίβεια, το F1-score και το Cohen's kappa. Η μετρική Cohen's kappa εκφράζει τον βαθμό της συμφωνίας μεταξύ δύο αξιολογητών, δηλαδή την μεταξύ τους μεταβλητότητα.

Και οι δύο παραλλαγές του δικτύου SeqSleepNet δίνουν παρόμοια αποτελέσματα. Υπάρχει μια αυξανόμενη απόδοση και στα δύο μοντέλα, η οποία συσχετίζεται με την περιπλοκότητα του προβλήματος που είναι να επιλυθεί, επομένως όσο λιγότερα στάδια ύπνου πρέπει να ταξινομηθούν, τόσο καλύτερη είναι η απόδοση του μοντέλου. Είναι ενθαρρυντικό το ότι η πιο περίπλοκη αρχιτεκτονική της δεύτερης παραλλαγής οδηγεί σε προβλέψεις πολύ κοντινές σε αυτές της πρώτης παραλλαγής ανά στάδιο ύπνου. Φαίνεται ότι η εισήγηση της μιας παραπάνω παραμέτρου των εποχών 30-δευτερολέπτων αντί για σημειακά δεδομένα σε κάθε ετικέτα σταδίου ύπνου, και μια ακολουθία N-δειγμάτων από δεδομένα εποχών 30-δευτερολέπτων για την πρόβλεψη των τελικών σταδίων του ύπνου εξακολουθεί να επιτρέπει στο μοντέλο να αποτυπώνει αποδοτικά τις εσωτερικές δομές των δεδομένων για επιτυχείς προβλέψεις. Η εμφάνιση μηδενικών προβλέψεων για κάποια από τα στάδια ύπνου, όπως φαίνεται στον αναλυτικό Πίνακα 4.14, και για τις δύο παραλλαγές του βασικού δικτύου συσχετίζεται με το πόσο ισχυρή ή όχι είναι η παρουσία των σταδίων του ύπνου στο σύνολο δεδομένων. Συγκεκριμένα, τα στάδια *N1* και *N3*, όπου το πλήθος τους διαφέρει κατά μια τάξη μεγέθους από τα υπόλοιπα, φαίνεται να έχει τις περισσότερες μηδενικές τιμές στην ακρίβεια. Επίσης, η δεύτερη παραλλαγή φαίνεται να ταξινομεί λάθος περισσότερα στάδια ύπνου από ό,τι η πρώτη, η μετρική *macro-average* του F1-score είναι μικρότερη, όπως επίσης εμφανίζονται περισσότερες προβλέψεις με μηδενικές τιμές για ορισμένα στάδια (το στάδιο ύπνου *N3* λείπει και από τα δύο μοντέλα για το πρόβλημα των 5 κλάσεων, ενώ το στάδιο *N1* λείπει μόνο στα αποτελέσματα της δεύτερης παραλλαγής).

Η μετρική **Cohen's kappa** μπορεί να χρησιμοποιηθεί για την μέτρηση της ορθότητας των προβλέψεων ενός μοντέλου ταξινόμησης, δείχνοντας την συμφωνία ή την τυχαία επιλογή μεταξύ των προβλέψεων και των αρχικών ετικετών του προβλήματος, αντί να συγκρίνει τις βαθμολογίες δύο αξιολογητών. Όσο πιο κοντά στην τιμή 1 βρίσκεται η μετρική κάππα, τόσο πιο ακριβείς είναι οι προβλέψεις του μοντέλου, ενώ όσο χαμηλότερη η τιμή του κάππα, οι σωστές προβλέψεις τείνουν να γίνονται με τυχαίο τρόπο, και όχι επειδή το μοντέλο έχει πράγματι μάθει να τις ξεχωρίζει σωστά. Από τον Πίνακα 4.14 φαίνεται ότι η τιμή του κάππα είναι ιδιαίτερα χαμηλή για τα πιο περίπλοκα προβλήματα των πέντε και τεσσάρων σταδίων ύπνου, ενώ βρίσκεται κάπου κοντά στην μέση του 0.5 για τα προβλήματα των δύο και τριών κλάσεων. Η παρατήρηση αυτή μπορεί να επιβεβαιωθεί από τις τιμές των F1-score της αναφοράς ταξινόμησης (classification report), που παρουσιάζεται και πιο αναλυτικά στον Πίνακα 4.14. Στον πίνακα αυτό, οι διακυμάνσεις των τιμών της ακρίβειας και των F1-score ανά κατηγορία προβλήματος (δηλαδή αριθμό σταδίων ύπνου που πρέπει να ταξινομηθούν), ευθυγραμμίζονται σωστά με την κατανομή των δεδομένων που συζητήθηκε σε προηγούμενες παραγράφους, όπου ορισμένα από τα στάδια ύπνου είναι ιδιαίτερα υπο-εκπροσωπούμενα λόγω του φυσικού κύκλου του ανθρώπινου ύπνου. Αυτό το στοιχείο της μεγάλης ανισορροπίας των δεδομένων θα έπρεπε να μην διαδραματίζει τόσο μεγάλο ρόλο όταν αρκετός όγκος δεδομένων είναι διαθέσιμος, αλλά φαίνεται να εξακολουθεί να επηρεάζει τα προτεινόμενα μοντέλα της εργασίας.

Chapter 1

Introduction

1.1 A Brief History of Human Sleep

Sleep is one of the fundamental human functions and can indicate various hidden information for an individual's condition, either physical or psychological, as well their general health. In fact, humans spend one-third of their lives asleep. It has been of a great interest from the earlier years of human history, where the focus was mostly on the interpretation of dreams and its connection with consciousness, up to the 18th century, when the interest shifted towards a different, more thorough direction; scientists started researching sleep patterns and their corresponding physiology of the human body.

Stepping a little bit behind, in 1729, the French astrophysicist Jean-Jacques d'Ortous de Mairan, conducting research in plants, noticed that mimosa leaves open during the day and close at night, even when kept in total darkness [deM29]. Thus, a day-night cycle seemed to exist in living organisms independently of the environment, known today as circadian rhythm. In 1880, Jean Baptiste Edouard Gellineau published his landmark description of the narcolepsy syndrome [Sch+07], while some years earlier, in 1875, the Scottish physiologist Richard Caton demonstrated electrical rhythms in the brains of animals [Cat75]. However, the aforementioned scientific landmarks occurred too early to be exploited by the field of sleep medicine [Dem98]. Interestingly enough, we have to mention here that the leading sleep disorder of the 20th century, obstructive sleep apnea syndrome, was firstly described not by a scientist, but by the novelist Charles Dickens in his series of papers entitled *The Posthumous Papers of the Pickwick Club*; there, he outlines an obese boy named Joe, who was excessively somnolent, a loud snorer, and who probably had right-sided heart failure (thus earning the nickname "young dropsy") [Tho11].

Back to our narration, the greatest advancements in the field have taken place during the last century. The discovery of neurons in 1888 paved the path for a deeper understanding on how the brain communicated with the rest of the body and consequently how it induced sleep. Some years later, in 1925, the invention of electroencephalogram (EEG) lead to the observation that brain waves connected to wakefulness, differentiated during sleep. Hans Berger, a German psychiatrist, made in 1929 early descriptions of the difference between sleep and wakefulness through the record and study of brain wave patterns [Ber29]. However, at that time these observations still strengthened the notion that sleep is an inactive or "idling" state. Specifically, until the discovery of rapid eye movements (REM) by Nathaniel Kleitman, sleep was universally regarded as an inactive state of the brain, which occurred of the lack of sensory input during night-time [Dem05]. In 1935, two other researchers in parallel, Hans Kalmus and Erwin Bünning [Bün35], independently discovered that the circadian rhythm existed in fruit flies and plants, and consequently to all living creatures. Just two years later, the team of Loomis, Newton and Hobart determined the different sleep states using the newly discovered electroencephalograph; they classified sleep into five stages and named the different characteristics of brain waves: delta, alpha, theta, beta, gamma [LHH35]. In 1939, Kleitman published his monumental treatise titled "Sleep and Wakefulness", where many years of sleep research were covered, as well sleep disorders, temperature changes during sleep and sleep-wake cycles, establishing the role brain stem has in skeletal muscle relaxation during sleep, which advanced the neurophysiology of sleep.

REM sleep was firstly detected in 1953 by Kleitman and Eugene Aserinsky in a young boy, contradicting the general impression that brain activity declined during sleep [AK53]. REM is the deepest stage of sleep, during which the eyes move rapidly from side to side and most dreaming occurs. Another student of Kleitman, Dr. William Dement, for the first time documented sleep cycles in 1955. Together with Kleitman, they observed some cyclical variation of EEG patterns and found that they occurred repeatedly throughout the night at intervals [DK57]. In 1958, melatonin hormone was discovered by Aaron Lerner and proved to be the key in sleep regulation. Michael Jouvet made the crucial distinction between REM and NREM (non rapid eye movement) sleep, demonstrating the sleep-related muscle atonia in 1959 [JMC59; Lup19] as well that the brainstem serotonin-containing neurons of the raphe nuclei were important in sleep and wakefulness [Rou+67]. Jouvet found that REM was not light-sleep, but "paradoxical" sleep, where increased brain activity is accompanied by skeletal muscle inhibition, which prevents the body from actively interpreting the vivid images and sounds in the dreams experienced during REM. In parallel, during NREM sleep, the brain activity is low and inhibition at this state is not detected.

The same year of 1958, Franz Halberg is said to be introducing the word "circadian", which derives from the Latin *about* (circa) a *day* (diem). Circadian rhythm is a physiological roughly 24-hour cycle, dictating when the body feels tired, awake, and hungry. Halberg was the first to thoroughly study these rhythms in human and was named the father of chronobiology. The discovery of the circadian rhythm revolutionized how the diagnosis of sleep disorders was done.

In the 1960s, the Association for the Psychophysiological Study of Sleep started collecting all the sleep-related research findings and discussing the creation of a consistent system for sleep staging. 1960s was the decade when sleep research greatly developed, emphasizing in all-night sleep recordings and paving the path for sleep medicine, specifically its core clinical test, polysomnography. The need of a consistent sleep staging occurred, thus the newly formed Association for the Psychophysiological Study of Sleep started collecting all the sleep-related research findings for this purpose. Allan Rechtschaffen and Anthony Kales were selected as co-chairs of an expert committee, tasked to develop a scoring manual for human sleep [Kir11]. There, they defined four non-REM stages of sleep according to brain wave patters. This guide was officially used up to 2017, where the American Academy of Sleep Medicine changed it to three non-REM stages [Mos+09]. A milestone during this time, was the discovery of obstructive sleep apnea in Europe in 1965, by two independent groups; Gastaut and colleagues in France [GTD66] and Jung and Kuhlo in Germany [JK65]. However, those findings were initially ignored in the US, since there was still no tradition in observing breathing during sleep in the prominent medical communities [Dem98].

The next phase of sleep research in recent history begins in 1970 with the launching of the first sleep lab, at Stanford University, by Dr. William Dement, specifically focused on studying sleep disorders. Stanford sleep researchers formally extended the practice of sleep medicine in order to include the *sleeping patient* [Dem98], meaning that the concept of sleep disorders and people suffering of those being considered as patients, is taken into account. The foundation of the routine of recording the respiratory and cardiac variables as part of the all-night sleep test was set, later to be called *polysomnography*. The specific parameters of Obtrusive Sleep Apnea syndrome (OSA) were established in 1976 [GTD76], while a year earlier the American Sleep Disorders Association (ASDA) or Association of Sleep Disorders Centers was formed, which later became the American Academy of Sleep Medicine (AASM). Also in 1975, Dr. Dement and Dr. Mary Carskadon created the Multiple Sleep Latency Test (LSLT), which helps diagnose a variety of sleep disorders [CD79], and is the standard approach until today as a quantification for sleepiness. One final important addition in this decade was the launching of the scientific jounal *Sleep* in 1979.

The latest history of sleep is marked by the introduction of alternative treatments to chronicle tracheostomy for OSA in the 1980s. The connection between circadian rhythms and sleep duration was determined, as well with other cues [Pot+16]. Also the relationship between sleep and learning was studied and the physiological necessity of sleep to human life, both in terms of quality and existence, was confirmed. The gold standard for sleep research to date was published in 1989, entitled *Principles and Practice of Sleep Medicine* [KRD89], now being at its 6th edition.

In the 1990s, the biggest aim grew to be the acceleration in the acceptance of sleep medicine throughout the world. Together with the establishment of several Foundations and Research Centers to pursue sleep science and its normalization to the public, new theories have also been proposed. Narcolepsy was discovered to be due to orexin receptor deficiency [DGD11], while the synchronization of human's biological clock with the sun as the retinal pigment processes the light was uncovered [BM16]. In 2003, Giulio Tononi and Chiara Cirelli suggested that sleep allowed the nervous system's communication networks to increase and reduce energy levels to conserve strength [TC14]. They also found that memory was directly dependent on sleep, so was the ability of a person to make correct judgment.

1.2 Sleep monitoring methods - then and now

Coming to a conclusion, it can be safely said that sleep is one of the fundamental human functions, with a great history, which has immensely developed during the last century. Its influence applies on so many aspects of human life, that it has to be taken into serious consideration on how it affects not only the profound health related issues, but our everyday life in general. Towards this direction, the most common method for sleep monitoring has been polysomnography since its discovery in the 20th century. However, the advancements of technology have allowed for newer methods, more straightforward, which can be easily used by individuals for their self-monitoring as well.

1.2.1 Polysomnography

Polysomnography, also called a sleep study, is a comprehensive test used to diagnose sleep disorders. Polysomnography records your brain waves, the oxygen level in your blood, heart rate and breathing, as well as eye and leg movements during the study.

It is typically done in a sleep center or a sleep disorder's unit of a hospital and originally performed at night. It can be also performed during day in order to accommodate shift workers or other people with circadian rhythm sleep disorders, who habitually sleep during different times of the day. A polysomnography technologist is supervising the process and monitoring the subject the whole night. Specifically, during the night's sleep, they monitor:

- brain activity (EEG)
- eye movements (EOG)
- muscle activity or skeletal muscle activation (EMG)
- heart rhythm (ECG)
- blood oxygen level through pulse oximetry
- snoring and other noise made during sleeping

The collected data are afterwards used to determine the subject's sleep stages and cycles, and detect possible abnormalities. A sleep "scorer" analyzes the extracted data, by reviewing the study into 30-second epochs. The extracted sleep score consists of the following:

- **Sleep onset latency**, meaning the onset of sleep after the lights were turned off; typically no more than 20 minutes.
- **Sleep efficiency**, defined as the real sleep duration in minutes by the total minutes in bed; this is usually around 85% or higher.
- **Sleep stages**, there are 4 sleep stages in total, 1-3 being called non-REM, whilst 4 being the REM stage. Awake is often considered as an extra sleep stage, for consistency.
- **Breathing irregularity**
- **Arousals**, which are sudden shifts in brain wave activity.
- **Cardiac rhythm abnormality**
- **Leg movement**
- **Body position during sleeping**
- **Oxygen saturation during sleeping**

A sleep medicine physician is interpreting the sleep score together with the test recording, in order to determine the subject's health matters. In that case, any medical history, list of drugs the patient is taking, as well any other crucial information are taken into consideration. Thus, many types of sleep disorders can be diagnosed, including narcolepsy, idiopathic hypersomnia, periodic limb movement disorder (PLMD), REM behavior disorder, parasomnias, and sleep apnea, although circadian rhythm sleep disorders cannot be directly diagnosed through this process.

1.2.2 Wearable devices

With the development of sensor technology, more fine and lightweight devices are used for the purpose of monitoring the signals to perform sleep analysis afterwards, which even allow the process to take place at the individual's home. Amongst the most common techniques are *actigraphy* and *photoplethysmography.*

Actigraphy refers to the use of an actimetry sensor with an embedded accelerometer, in the form of a wristwatch-like package, in order to measure gross motor activity, for a prolonged period of time. The actigraph is typically worn on the non-dominant arm and is used to study sleep-wake patterns by detecting

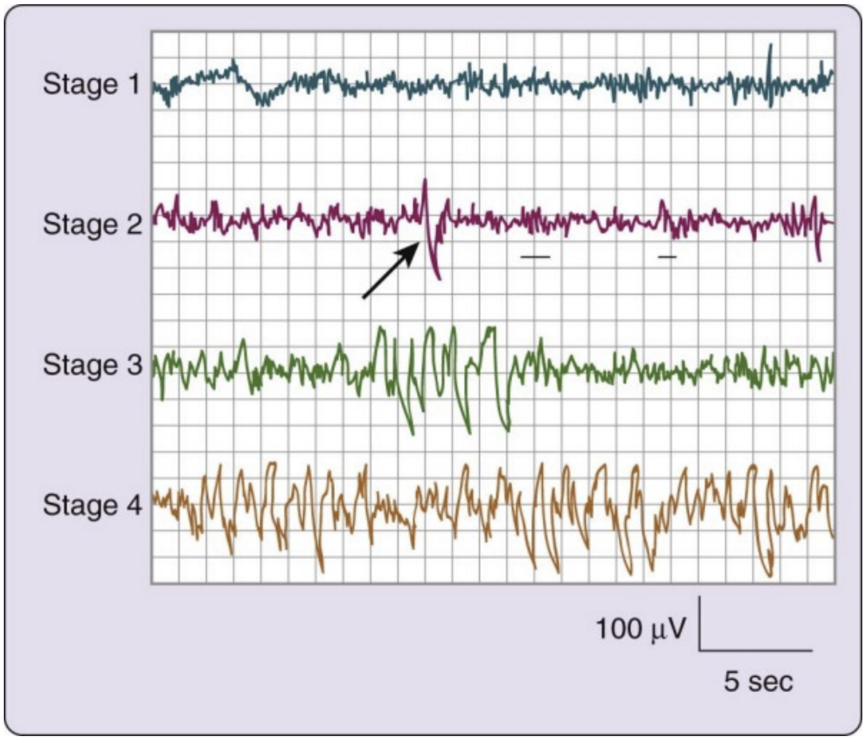


Figure 1.2.1: Characteristic EEG activity of each of the four stages of NREM sleep. On the second tracing, the arrow indicates a K-complex and the underlining shows two sleep spindles [CD+05] [01].

motion of the wrist with linear accelerometers in single or multiple axes. Based on movement-derived data, predictions of the time spent during sleep and wakefulness can be made, and even assumptions on sleep staging.

Photoplethysmography is a simple and low-cost technique for detecting blood volume changes in the microvascular bed of tissue, usually obtained with a pulse oximeter. This is a device that monitors the perfusion of blood to the dermis and the subcutaneous tissue of the skin, by illuminating the skin and measuring the changes in the light absorption. The collected measurements of a photoplethysmography are usually processed to determine heart rate and cardiac cycle. It can also be used to monitor respiration, depth of anesthesia, hypo- or hypervolemia and blood pressure.

Those advancements are increasingly being incorporated into smart wristbands, such as smartwatches or activity trackers, not only to monitor health-related issues, but also to serve as a daily lifestyle self-tracking interface, with the purpose of improving physical, mental or emotional performance.

Under this perspective, sleep tracking has been one of the main goals for wearable devices technology, due to the high impact it has on so many aspects of life. It consists of two primary parts; the first being the data monitoring and collection, and the second being the data processing to extract the sleep stages and other sleep-related factors, such as its efficiency. One of the key differences from a traditional PSG is that it does not measure sleep as defined by electroencephalography (EEG), electrooculographic (EOG), or chin electromyographic (EMG) criteria or the subjective experience of sleep (as measured by sleep logs and questionnaires). Thus, there might be differences in the estimation of e.g. sleep duration or latency between the two and the results of a wearable device's health monitoring should not be taken as proof or evidence, but as an indication for the need of possible further research with the cooperation of a professional medical expert.

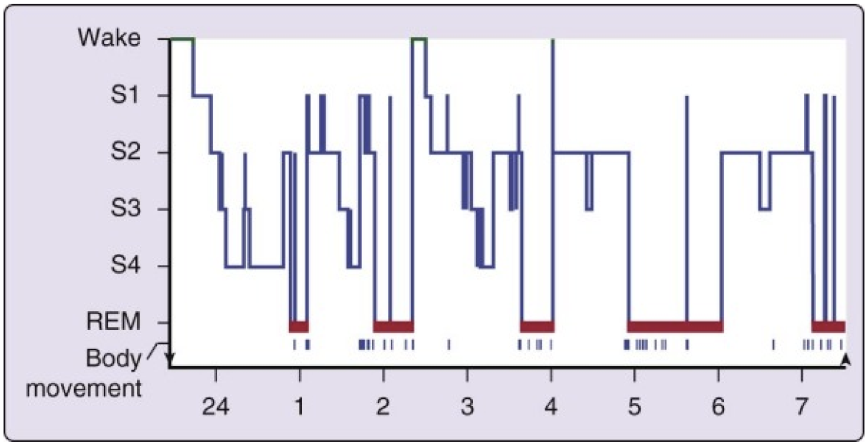


Figure 1.3.1: The progression of sleep stages across a single night in a normal young adult volunteer is illustrated in this sleep histogram. The text describes the ideal or average pattern. This histogram was drawn on the basis of a continuous overnight recording of electroencephalogram, electrooculogram, and electromyogram in a normal 19-year-old man. The record was assessed in 30-second epochs for the various sleep stages. REM, rapid eye movement [CD+05] [01].

1.3 Sleep Physiology

1.3.1 Sleep Architecture

Sleep might be considered by some as an inactive state of mind, but the truth lies far from that. In reality, some parts of the brain are quite active during sleep. There are two types of sleep, rapid eye movement (REM) and non-rapid eye movement (NREM), and the latter one consists of three different stages, named N1, N2 and N3. During the course of the night, the body goes through several rounds of the sleep cycle, with every successive REM stage increasing in duration and depth of sleep. Each sleep stage has unique characteristics, including variations in brain wave patterns, eye movements and muscle tone. The sleep-wake cycle is regulated by two internal biological mechanisms: circadian rhythm and homeostasis, and sleep patterns are dependent on age, changing over an individual's life span.

REM - NREM sleep cycles

Beginning with a short period of N1 stage from NREM category, sleep episode is processing through stages N2, N3 and REM, circulating around them several times. NREM sleep constitutes around 75-80% of the total amount of sleep, and the remaining 25-30% is REM. The first REM-NREM cycle has an average length of 70-100 minutes, but, as the night progresses, the cycles' duration increases, getting approximately at 90-120 minutes [CD+05]. REM has its longest duration at the last one-third of the sleep episode, while N2 begins to be the primary stage of NREM sleep and stages N3 and N4 might gradually disappear [MLW18].

In Figure 1.3.1 a hypnogram of a whole night's sleep is depicted, derived from a young adult without any sleep disorders.

Stage N1

The first stage of sleep serves a transitional role between wakefulness and sleep. This is the way an average individual's sleep episode begins, except for people with narcolepsy or other specific neurological disorders, as well as newborns.

- The body and brain activities start to slow down, with periods of brief movements (twitches). The body has not fully relaxed yet during this sleep stage and it can be easily interrupted by a disruptive noise. If an awakening happens, the individual might not feel as if they have slept at all.

- If no interruption occurs, N1 stage usually lasts 1-7 minutes in the initial cycle, being 2-5% of the total sleep.
- Brain activity as monitored by EEG, transitions from wakefulness, marked with rhythmic alpha waves, to low-amplitude mixed-frequency (LAMF) activity.
- Also, muscle tone is detected in the skeletal muscles and breathing still occurs at a regular rate.

Stage N2

During this stage, the body enters a more subdue state, where the body temperature goes down, heart rate and breathing regulate, and the eye movements slow or completely stop.

- Its approximate duration is 10-25 minutes in the beginning of sleep, expanding with each successive cycle, eventually being the 45-55% of the total sleep episode. Thus, a person typically spends about half their sleeping in N2 stage.
- The brain patterns as seen in an EEG also change; this stage is characterized by the presence of sleep spindles, K-complexes or both.
- Sleep spindles [SP18] are thought to be a feature of memory consolidation [And+11]. It has been shown that individuals who are in the process of learning a new task have significantly higher density of sleep spindles than those in a control group [Gai+02].
- K-complexes show a transition into deeper sleep. They are single, long delta waves only lasting for a second. As deeper sleep ensues and the individual will be passing through N3, all their brain waves will be gradually replaced by delta waves.

Stage N3

Sleep stage N3, previously known as two separate stages N3 and N4, is referred to as slow-wave sleep (SWS) and it mainly occurs during the first third of the night. It is considered the deepest stage of sleep and it is characterized by delta waves, which are high amplitude signals with much lower frequency.

- The heartbeat and breathing slow to their lowest levels during sleep. The muscles are relaxed and it is the most difficult stage to wake up from.
- Getting enough N3 NREM sleep is considered to be crucial for feeling refreshed next morning. It is believed that this stage is critical to restorative sleep, boosting bodily recovery and growth. Also, although brain activity is low, declarative memories are processed and consolidated [FD15].
- This stage is the most difficult to awaken from, and for some individuals even loud noises, over 100 decibels, will not fulfill this purpose. Awakening from N3 stage leads to a transient phase of mental fogginess, known as sleep inertia, which might take 30 minutes up to 1 hour to recover from.
- As people get older, the time spent in NREM sleep stages is shifted from N3 to N2. Typically, during the early sleep cycles, N3 stages last 20-40 minutes, while as sleep processes, these stages get shorter and more time gets spent in REM phase instead.

REM

REM stage is associated with dreaming. During REM sleep, an increased activity of the brain is observed, similar to when a person is awake. At the same time, the skeletal muscles are atonic and without movement. This temporary immobilization of the body prevents individuals from acting out their dreams [AC+06].

- The eyes move rapidly behind the closed eyelids and breathing rate becomes more erratic and irregular.
- Also, the heart rate and blood pressure increase to almost the waking levels.
- REM sleep occurs after around 90 minutes into the sleep cycle. The first REM period lasts approximately only around 1-5 minutes. However, it becomes progressively prolonged as the sleeper cycles through the stages several times before waking, gradually lasting even up to an hour in later rounds [Mon+18; Fer+17].

- The amount of REM sleep an individual gets, changes with aging. The percentage of REM sleep is highest during infancy and early childhood, declines during adolescence and young adulthood, and then lessens even more as a person gets older. In total, a healthy adult experience around 25% of REM sleep through a whole nights' sleep.
- REM is considered essential for cognitive functions such as memory [EPS06], learning and creativity [Cai+09]. Specifically, it is believed that REM sleep is the time when mostly emotions and emotional memories are processed [Glo+20].
- People suffering of sleep disorders, such as obstructive sleep apnea, often do not reach deeper sleep levels as needed, since they get frequently woken due to their condition. Their quality of life is thus affected, and might be drastically decreased, as a result of their body being unable to perform the needed actions of repairing damage, leading to an increased fatigue upon waking and throughout the day.

Once REM sleep is over, the body will normally return to N2 stage of NREM sleep, before beginning a new cycle all over again. About four or five cycles in total are passed through during a normal night's sleep, each one being around 90 - 110 minutes. For adults, it is recommended to get a total of at least 7 up to 9 hours of sleep per night.

1.3.2 Sleep - Awake Regulation

The sleep-wake system is considered to be controlled by the interplay of two separate, internal biological mechanisms; the circadian rhythm and homeostasis. These two processes are complementary, with one promoting sleep (homeostasis) and the other maintaining wakefulness (circadian rhythm) [GA05]. The need of sleep accumulates throughout the day, reaches its highest level right before bedtime, and decompresses during the night's sleep.

Circadian rhythms lead a wide range of body functions, from daily fluctuations in wakefulness, to body temperature, metabolism or even the release of hormones. They control the timing of sleep and motivate the body to get sleepy at night and wake up in the morning without the need of an alarm.

Sleep-wake homeostasis regulates an individual's need for sleep. The homeostatic sleep drive directs the body to sleep after a certain amount of time has passed, and controls the intensity of sleep. To achieve this, it strengthens with every passing hour of wakefulness, leading to longer and deeper sleep after a period of sleep deprivation.

The way circadian rhythm maintains wakefulness balances the homeostatic drive for sleep during the day, promoting alertness and vigilance. The system gradually withdraws, until bedtime, in order to enhance sleep consolidation, which deprecates through the night [AC+06; GA05]. An adequate night's sleep allows the homeostatic drive for sleep to reduce, while the circadian waking drive increases, leading to a new start of the cycle. This cycle lasts roughly 24 hours, in other words a day, and is synchronized with the environmental cues, such as light and temperature, although it can function in the absence of those cues as well. In case the process of maintaining wakefulness is lacking or deficient, total sleep time remains the same, however it is not consistent during the nighttime, but rather randomly distributed during the whole duration of the day. Thus, it is important to note that circadian rhythm serves to maintain sleep and wake states into two separate functions, which successively alternate in the course of 24-hour time periods.

1.3.3 Circadian Rhythms

Circadian rhythms collectively refer to the daily rhythms of a 24-hour time window, in physiology and behavior [AC+06]. They are generated by neural structures that lie in the hypothalamus and are functioning as a biological clock [DLD04]. As Bünning observed in 1964, plants and animals have endogenous clocks, which give rhythm to their daily behavioral and physiological functions in accordance with the external day-night cycle [Bün64]. The foundation of these clocks is believed to be the expression of a series of molecular pathways involving "clock" genes, in a nearly 24-hour basis [VPT05].

In mammals there is a closed cycle of two specific proteins' expression which bind together and travel into the nucleus, causing the activation of genes in specific areas of the DNA, among which are *Period* and

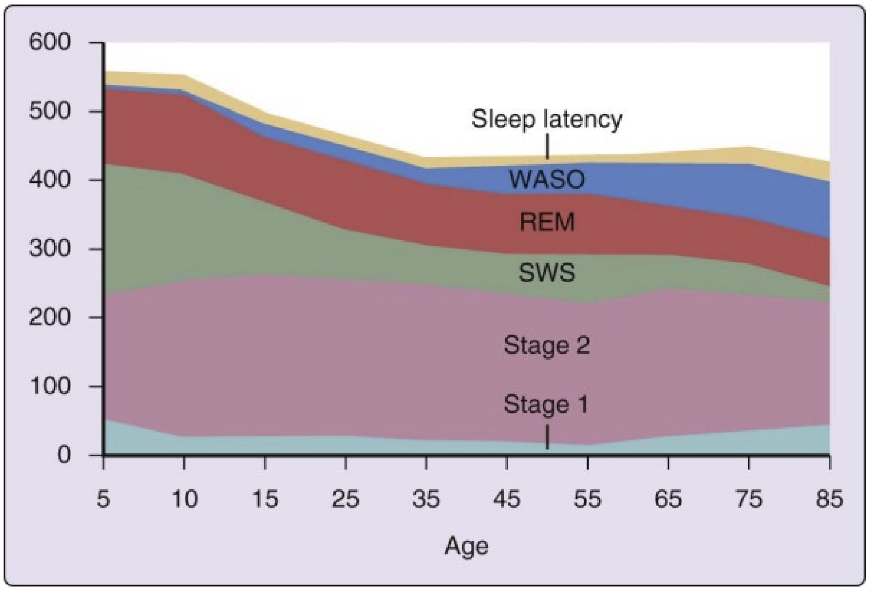


Figure 1.3.2: Changes in sleep with age. Time (in minutes) for sleep latency and wake time after sleep onset (WASO) and for rapid eye movement (REM) sleep and non-REM (NREM) sleep stages 1, 2, and slow wave sleep (SWS). Summary values are given for ages 5 to 85 years [CD+05] [01].

Cryptochrome. The products from the expression of those two genes return to the nucleus to disrupt the binding of these proteins, resulting in stopping their own synthesis. This results in a rising and falling pattern of the expression of *Period* and *Cryptochrome*, with a periodicity being approximately 24 hours. This biological process is applied to many other genes as well, which affect many tissues in the body, triggering daily patterns of activity. Those rhythmically expressed genes contribute to several parts of cellular function, pointing out the great importance of the circadian system in many central aspects of life.

The Suprachiasmatic Nucleus

Responsible for regulating all organs' circadian rhythms is the suprachiasmatic nucleus (SCN); it receives signals from a class of nerve cells in the retina, which have the role of brightness detection, and can reset the clock of SCN in a daily basis. The signals are transmitted from there to the rest of the body and brain, in order to synchronize with the external day-night cycle.

Resultingly, the sleep mechanism is also affected by the SCN; a series of relays of the environment's brightness signal is passed through the dorsomedial nucleus of the hypothalamus, where the structures generating the circadian rhythms are lying. Hence, the wake-sleep systems are forced to coordinate their activity with the day-night cycles. This can be altered independently of the day-night cycle under some circumstances in animals due to special external conditions, but they will not be discussed here.

One more important pathway that gets signals from SCN as input, is the mechanism controlling the secretion of melatonin, which is a hormone created in the pineal gland. Melatonin is mostly produced during night and further stabilizes the circadian rhythm, but its effect directly on sleep has limited range.

Sleep and Thermoregulation

Circadian system influences the body temperature regulation. In general the body temperature is higher during the day and lower at night. Together with body temperature reduction, a decrease in heat production is also observed, as well an increase of heat loss, which all lead to sleep onset and maintenance and EEG slow-wave activity. There is an opposite mechanism promoting heat to increase some hours before waking, where the brain sends signals to other parts of the body to pursue heat production and conservation, which will gradually stimulate waking.

1.3.4 Sleep Patterns and Aging

Sleep architecture is greatly dependent on age, changing continuously and considerably in parallel with aging. Starting from infancy and moving towards adulthood, prominent changes appear in how sleep is initiated and maintained, the percentage of time each sleep stage lasts, and the overall sleep efficiency. Sleep efficiency is observed to decline with age and, while the consequences of this state are well documented, its causes seem quite complex and poorly understood. The clarification of sleep characteristics by age, however, allows for a deeper understanding of human sleep and its effect on human development and successful aging [AC+06]. Thus, aging should be taken into consideration in parallel to the general health condition of a subject when conducting sleep research.

1.4 Thesis Overview

Summing up all the above, in this thesis we will examine methods for sleep stage classification using biosignals collected from a wearable device; specifically an Apple Watch. Except of the data collected through the smartwatch, one more dataset is incorporated to enhance the amount of data tested on the proposed methods. The second dataset is collected via actigraphy and polysomnography, which can be directly compared to the smartwatch-collected dataset.

The aim of the current thesis is to test several deep learning architectures for the task of sleep stage classification, using data collected from wearable devices. For this purpose, two main approaches are applied on the two aforementioned datasets separately:

1. In the first approach, a standard neural network (Bidirectional Long-Short Term Memory), appropriate for temporal and sequential classification problems, is incorporated and an architecture is designed from scratch, in order to analyze the following two cases:
   (a) Using manually extracted features to train a neural network for the sleep-stage classification task. First the same kind of features are extracted for both datasets, so that they can be directly comparable with each other. Then they are given to the proposed neural network architecture for training.

       The chosen features are derived by another work, which introduced the AppleWatch dataset and incorporated some classic machine learning algorithms for the same task of sleep stage classification. Under this scope, in this work a comparison is also made between typical machine learning approaches and neural networks for the specific task, using the same kind of features to train both system categories.

   (b) The second case is an extension of the first, where the proposed architecture is enhanced with a neural module (consisting of Convolutional Neural Networks) for automated feature extraction, taking as input the raw data provided by the wearable devices.
2. One more approach is tested, where a deep neural network initially designed for PSG (polysomnography) data, hence EEG, EMG and EOG spectrograms, is altered in order to take as input the raw data of the two wearable-derived datasets and train on those for the task of sleep-stage classification. In this case, except for testing a more complex model on the two datasets, a comparison of how wearable-derived datasets perform on an architecture designed for PSG-collected signals is presented, and the initial differences and even possible limitations of the wearable devices are highlighted.

Neural networks lie under the umbrella of machine learning, which is an advanced section of artificial intelligence. Specifically, a neural network algorithm automatically improves through experience and by the use of data in order to get the best possible results.

Hence, this work begins with a short explanation of sleep stages and the problem of sleep stage classification. It continues with an introduction to machine learning, neural networks and the techniques that are mostly used on the subject of sleep stage classification. Finally, the implemented experiments of the two main categories and datasets follow, together with the results and final thoughts.

Chapter 2

Literature Review

2.1 Introduction

Some of the most noteworthy works on the subject of sleep stage classification will be discussed next, covering several approaches to the problem, starting with the type of features to be handled. There are two main categories of data in the field of sleep stage classification; traditional data recorded during a polysomnography study in a specialized sleep center or unit and data taken from a wearable device, which can be done in the individual's home. After collecting the data, typically some preprocessing methods are used, to clean them and shape them in a useful form to be further processed. Following the data preparation, future extraction and feature selection can be applied to the processed data, to collect meaningful features that will help with the classification afterwards. It is to be noted that this is a typical pipeline of processing steps for any kind of data and application on a scientific problem and not only for sleep stage classification, which is examined here. Finally, after the data are properly prepared, a classification method is applied, which can also be divided in two main categories, as presented in Chapter 3.1. This can be either a particular machine learning algorithm or a deep learning approach with neural networks and will be further discussed in section 2.4.

2.2 Types of Data

As stated in Section 1.2 of Chapter 1, there are two types of data to be used for sleep stage classification, depending on the manner they were recorded. The first one is the traditional method of monitoring the human physiology during sleep through polysomnography, including **EEG, EOG, EMG, ECG** and **pulse oximetry** recordings for blood oxygen levels. The second category of data is acquired through a wearable device, typically consisting of actigraphy and photoplethysmography measurements, deriving **3-D acceleration** and blood volume changes thus the **heart rate** of the individual, respectively.

During the recording session of the first type of data via PSG, the labels for the sleep stages are collected as well, determined by a sleep expert who has been monitoring and supervising the whole process. Thus, the analysis of the data and the task of classification, after the manual labeling of the expert are straightforward. Since this method of sleep monitoring and data collection has been the main approach for many years now, there is a great amount of data and datasets to be used in research and other applications of sleep stage classification and sleep physiology in general. One drawback however is how demanding PSG is, as it requires the subject to be in a specified unit during the whole sleep session and a sleep expert to be in charge of the procedure the whole time.

The second type of data acquired from wearable devices are much easier to collect due to the portability of the devices and how accessible they are, which is their main advantage as discussed in section 1.2. However, in order to obtain sleep stage labels, a setup similar to PSG is required, with a sleep expert supervising and labeling the whole sleep session. Since wearable devices are a much newer approach in sleep stage monitoring, the data resources for research purposes are still limited. One of the first open datasets collected from a smartwatch and consisting of both heart rate and wrist acceleration measurements in parallel to PSG labels is developed by Walch et al. [Wal+19], which is also examined in the current thesis. One more standard dataset is the Multi-Ethnic Study of Atherosclerosis (MESA) cohort, which consists of motion data from actigraphy-derived activity counts and heart rate via pulse oximetry from co-recorded PSG [Bil+02], which is also utilized in this thesis for experimental purposes.

2.3 Data preprocessing and Feature Extraction

The collected bio-signals need to go through preprocessing, to ensure that only quality signals will be used in the next stages of the experiments [Sup+16]. This pipeline includes removing irrelevant artifacts from the signals, correcting inaccurate signals, applying interpolation or extrapolation, normalizing them into a desired range of values, or use a filter to exclude unwanted components.

After the first preprocessing steps, meaningful features need to be extracted or derived for the specific problem to be solved. The algorithms to be used for feature extraction are defined by the nature of the problem and are typically hand-engineered by experts. With the establishment of neural networks, machine learning algorithms are employed to train and construct models that understand relationships between input (i.e., extracted features) and their desired output (i.e., labels), and generalize observed data to new situations.

Towards this direction, deep learning can be employed in several applications to automate the hand-engineered process of feature extraction. Due to the NN's complexity, which consists of multiple layers of linear or non-linear processing units, meaningful representations are expected to be derived from high-dimensional data in an automated way. The data can either be given to the network having undergone a minimal preprocessing first, thus the network is considered to be taking as input the *raw data*; or a subset of extracted features is used as input to an NN model, as it can speed up the whole process, and improve the generalization of the constructed model to prevent overfitting.

The biological data used for sleep stage classification consist of time sequences, which belong to the spatio-temporal domain. A wide variety of signal processing techniques can be utilized to extract discriminative information from the collected signals; those can be grouped in the following categories [BKN17]:

- **Time domain features** - They are simply interpretable and can represent the *morphological characteristics* of a signal. They are quite suitable for real-time applications and some of the most prominent ones are the *statistical parameters.* They consist of the 1^{st} to 4^{th}-order moments of a time series, namely *mean, standard deviation, skewness and kurtosis* respectively, but also the *median* and 25^{th}*,* 75^{th} *percentile* of the signal distribution.

- **Frequency domain features** - In order to extract *spectral characteristics* of the signals, the time series should be transferred in the frequency domain first. To do so, the Fourier Transform (FT) is applied on the auto-correlation of the signal, to extract the *Power Spectral Density* (PSD).

  Fourier Transform is an extension of the Fourier series, which introduces the incorporation of complex exponential functions for expressing features, such as the amplitude and phase of a frequency component, through complex numbers. It consists of a frequency continuum of components using an infinite integral of integration, as it can be seen in Equation 2.3.1.

$$\mathbf{FT}\{x(t)\}(t,\omega) \equiv X(\omega) = \int_{-\infty}^{+\infty} x(t)e^{-j\omega t}dt \tag{2.3.1}$$

  In the discrete time domain, the estimation of PSD can be either done through parametric or non-parametric methods.

  The first ones are based on parametric models of a time series, such as autoregressive (AR), moving average (MA), and autoregressive-moving average (ARMA) models. Therefore, parametric methods are also known as model-based. To estimate the PSD of a time series with parametric methods, the model parameters of the time series need to be obtained first.

  The non-parametric methods are based on the Discrete Fourier Transform and are calculated directly from the signal samples in a given windowed signal. Amongst these methods are the periodogram, Welch, and Capon method for calculating PSD and extracting features from the signals. An easy implementation of DFT is the Fast Fourier Transform, which is widely used for this kind of calculations. The primary limitation of non-parametric methods is that the computation uses data windowing, resulting in distortion of the PSD due to window effects, but this is counterbalanced by the method's robustness.

  Higher order spectra is another way of extracting frequency domain features and represents the frequency content of higher order statics of signals (cumulants). The main advantage of high order spectra is its ability to reveal non-Gaussian and nonlinearity characteristics of the data.

- **Time-Frequency domain features** - This type of features is quite efficient in bio-signals due to their non-stationary nature. The most used category of time-frequency feature extraction is *signal decomposition*, in which signals are decomposed to a series of basis functions.

  Short-time Fourier Transform (STFT) is the most typical time-frequency analysis, and, being a Fourier-related transform, it is used to determine the sinusoidal frequency and phase content of local sections of a signal as it changes over time. Specifically, the signal is split into windows and a Fourier Transform is applied to each one separately, as shown in Equation 2.3.2.

$$\mathbf{STFT}\{x(t)\}(t,\omega) \equiv X(\tau,\omega) = \int_{-\infty}^{+\infty} x(t)w(t-\tau)e^{-i\omega t}dt \tag{2.3.2}$$

where $w(\tau)$ is the window function, commonly a Hann window or Gaussian one centered around zero, and $x(t)$ is the signal to be transformed. $X(\tau,\omega)$ is essentially the Fourier transform of $x(t)w(t-\tau)$, which is a complex function that represents the phase and the magnitude of the signal over time and frequency.

However, one pitfall of STFT is that it has a fixed resolution; the width of the window determines whether there is a good frequency or time resolution. There is an inverse relationship between the length of the window and the time resolution and also a trade-off between the time and frequency resolution in an FT. The product of standard deviation in time and frequency is lower bounded and this property is connected to Heisenberg's uncertainty principle.

This is one of the reasons for the creation of the *Wavelet Transform* (WT) and multiresolution analysis, which can give good time resolution for high-frequency events and good frequency resolution for low-frequency events. WT is a popular time-frequency transformation that applies different filters to decompose a signal into dyadic frequency scales. It is a powerful method since it describes the signal into different frequency resolutions where the decomposed signals are orthogonal in most mother wavelets. A *mother wavelet* is a function that generates the filters for the signal's decomposition.

- **Non-linear features** - Due to the non-linear characteristics and complex dynamics that comprise biological signals, there are two major non-linear methods for their analysis.

  Entropy-based methods calculate the irregularity and impurity of a signal in the time domain. Entropy indicated the amount of changing patterns inside a windowed signal; the more regular they get, the lower the entropy value and vise versa.

  A second non-linear approach for bio-signal analysis is *fractal-based*. It derives from the notion that such signals, while having a noisy behavior, are rule-based in nature. Instead of analyzing a signal in the time domain, these methods analyze its trajectory behavior in the phase space, in terms of self-similarity. Thus, the concept of fractal dimension can describe the behavior of random-like shape by determining the amount of self-similarity on that given shape or signal [BKN17].

2.4 Related Work

Based on the previously stated discrimination between the data types used for sleep stage classification, there will be two separate sections discussing each, starting with the classical PSG bio-signals, and continuing with works on signals collected via wearable devices, which is also the main focus of this thesis. The pipeline of each work will be briefly presented as it allows for a preliminary understanding of how the problem of sleep stage classification is handled under different circumstances.

2.4.1 PSG-based Works

Machine Learning

In an early work of sleep stage classification [Lia+12] employing multichannel EEG, EOG and EMG signals and PSG labels from seventeen subjects, twelve features including temporal and spectral analyses were extracted. The signals were firstly downsampled and filtered, and then segmented into 30-second epochs. For the spectral features, the Fast Fourier Transform was applied into 2-s non-overlapping segments. Then the extracted features were normalized to reduce the effects of individual variability. A hierarchical decision tree with fourteen rules was constructed for the classification of five stages, as scored by R&K rules [Kir11], with the overall agreement and kappa coefficient of the proposed model being 86.68% and 0.79, respectively.

In [Şen+14], a combination of methods is applied on a single-channel EEG signal to extract 41 features of the aforementioned categories (time, non-linear, frequency-based and entropy). For the experiments 25 individuals are selected from the [Gol+00] database. Some preprocessing filters are applied on the signal in order to smooth it and remove any noise and artifacts. The signal is split into 30-second epochs and a Hamming window is applied on each. A group of feature selection algorithms is applied on the extracted features, in order to reduce redundancy, increase the computation efficiency and keep only the useful ones. Those are the *fast correlation based filter* (FCBF) [YL03], *mRMR algorithm* [DP05], *Fisher score algorithm* (FS), *t-test* and the *ReliefF algorithm* (RF) [Kon94] and are tested with four different classification algorithms, to find the combination with the highest score. The best results for six sleep stages according to R&K rules are obtained via a hybrid approach where three or more attributes of each category were selected and the final accuracy score is 97.03% for *Random Forest* algorithm (RF), followed by *Decision Trees* with 92.35%, *Radial basis network* (RBF) with 89.45%, *Support Vector Machines* (SVM) with 93.12% and *Feed-forward neural network* with 71.88%.

An attempt to combine classical ECG measurements with thoracic respiratory effort collected via respiratory inductance plethysmography (RIP) was made in [Fon+15], for 3 and 4-sleep stage classification. The data collection is made through an unobtrusive method with the ability to analyze them digitally afterwards. In this work the dataset consist of 48 subjects acquired by the Siesta project [Klo+01] and 142 features are extracted from cardiac and respiratory activity using a sliding window centered on each 30-second epoch. The features lie on the time and frequency domain and are z-score normalized and smoothed via cubic spline-fitting, for better interpolation of missing values due to e.g. motion artifacts. A multi-class Bayesian linear classifier with time-varying prior probabilities [Lon+14] is used on a final set of 80 features, found through a wrapper feature selection method based on sequential forward selection (SFS) and using as criterion the Cohen's kappa coefficient of agreement κ. Applying 10-fold cross validation, the best accuracy for Wake, Light, Deep and REM classification was 69% and $\kappa = 0.49$, while for 3-stage classification the values increased to 80% and 0.59, respectively.

In another work using a single-channel EEG signal of 25 individuals [HB17], a newly proposed tunable-Q factor wavelet transform (TQWT) is applied, decomposing the signal segments into TQWT sub-bands. Then, a normal inverse Gaussian (NIG) PDF modeling of TQWT sub-bands is performed, wherein NIG parameters are used as features and statistical hypothesis testing is applied on those in order to extract the final feature set. Adaptive boosting [FS97] is employed for sleep stage classification, resulting in accuracy of 91.36%, 92.46%, 94.83% and 98.01% for 5-, 4-, 3- and 2-stage classification respectively.

Rahman et al. [RBH18] used single-channel EOG signals and applied DTW to decompose them into time-frequency domain components, extracting various statistical features afterwards. The discrimination ability of the features is established via One Way Analysis of Variance (ANOVA) statistical analysis and a feature reduction scheme based on Neighborhood Component Analysis (NGA) is employed to reduced their number. Finally, three different ML classification algorithms are tested, namely Random Under-Sampling Boosted Tree (RUSBoost), Random forest (RF) and Support Vector Machine (SVM), achieving state-of-the art results at the time of the publication for the 4- and 2-class sleep stage problem of 92.89% and 98.24% respectively.

Deep Learning

A single-channel EEG signal is used in [Hsu+13], but instead of classical ML methods, a recurrent neural network is utilized for the classification task. The greatest advantage of choosing NN instead of simple ML algorithms, apart from their learning capability and robustness, is their fault tolerance property, as recording phases can be partially impeded and processed data can be blurred due to undesirable or unexpected events. The data undergo some preprocessing steps and energy features are extracted. The proposed classifier is an Elman RNN, capable of dealing with time series signals and it is compared with a Feed-forward NN and a Probabilistic NN. The experimental results indicate that the classification rate for 5 sleep stages of the RNN outperforms the ones of FNN and PNN, being 87.2%, 81.1% and 81.8% respectively.

In [Don+17] a mixed neural network (MNN) is proposed, consisting of a rectified neural network, suitable for detecting naturally sparse patterns, followed by an LSTM, best handling temporal pattern recognition problems. A single-channel EEG signal is used as well, and time-frequency analysis is applied to extract spectral features. The spectral features are given as input to the rectified neural network for detecting the internal hierarchical structures and then the outputs are passed to the LSTM for sequential learning, leading to a final model's accuracy of 85.92%.

[Sor+18] also use a single-channel EEG signal, but apply a CNN architecture instead. In order to include the temporal context of the data, the network takes as input the current epoch as well as the two preceding epochs and the following one, and requires no signal preprocessing or feature extraction phase. The network is trained to learn feature detectors that are suited to the classification task at hand and are likely to perform better than hand-engineered features, while this learned sensible pattern detection can be visualized. The proposed method is competitive in terms of performance for the task of 5-sleep stage classification, with accuracy of 87% and Cohen's kappa of 0.81.

An ensemble of five CNNs is proposed in [Fer+19], to improve the results obtained by a single model: all the 5 models classify the same input and the final decision is taken using the majority of the votes. The ensemble consists of the 5 best performing hyperparameter configurations, for the task of 5-sleep stage classification. In this work multiple signals are used, namely two EEG, one EMG, and two (left and right) EOG channels, which undergo filtering first to reduce noise and artifacts. The proposed method achieves an average precision, sensitivity, and F1 score of 0.78, 0.75 and 0.76 respectively, with a kappa index value of 0.83.

[FRP20] conduct experiments to test whether a double EEG signal outperforms a model using a single signal as an input. The proposed model is CNN-based and two cases are examined; the first one is giving the two EEG signals as double input to a single model, while the second method is to construct an ensemble model as described in the previous work. Experimental results indicate that using two signals improve the result over using a single one as input to the mode, whereas the ensemble model shows no advantage with respect to the double-signal-input one. This observation lays on the incapacity of the single-signal-input architectures to identify information that was not previously captured by the double-signal-input model. The best acquired result shows an accuracy of 92.67% and a Cohen's kappa value over 0.84 compared to human experts.

A different approach is proposed in [Li20], trying to show-light the importance of data pre-processing in parallel to the choice of a simpler neural network to deliver a more robust model, outperforming traditional ML methods as well as complex DL models. For this purpose, a single-channel EEG signal of 45 individuals is chosen and the Welch method is applied to make the conversion from time domain to the frequency domain. It is observed that the raw EEG signal in time domain contains a huge amount of noise, which misleads the model during training; on the other hand, signals in the frequency domain could circumvent the presence of noises and augment the effect of valuable features for sleep stage scoring. A shallow CNN structure is utilized and the experimental results showcase that the model performs relatively well on the 5-class problem, in all stages except from stage N1. The incorporation of an LSTM layer at the end of the CNN does not make a big difference on the results, while the computational cost greatly increases. Hence, both data pre-processing and neural network structure is equally important when designing the model, especially balancing between model accuracy and efficiency.

[Zha+20] propose an orthogonal convolutional neural network (OCNN) for learning rich and effective feature representation. Using EEG signals for the experiments, it is stated that time-frequency analysis can provide a better representation of EEG waves and sleep events; thus the input signals are converted to dynamic time-frequency images via Hilbert-Huang transform (HHT), which is considered to perform better when the signal is nonlinear and non-stationary. Afterwards, an autoencoder is used to reduce the dimensionality of the extracted images, which are then given to an orthogonal CNN for sleep stage classification. Compared to vanilla CNNs, it is observed that the proposed OCNN can learn rich and diverse feature representations. Testing on two different datasets, the accuracy and kappa coefficient are formed as 88.4% and 0.82, 87.6% and 0.8, respectively.

In the same manner of automatic feature learning without utilizing any hand-engineered features, [XZY20] uses single-channel EEG signals and proposes a fast representation learning (FRL) and semantic-to-signal learning (S2SL) framework. Inspired by the use of semantic information in image classification to improve the performance of the model, this work employs S2SL, with auxiliary classifier generative adversarial network (ACGAN) as the basic structure to mine semantic features related to sleep stages. Textual information is collected from the AASM manual [Ibe07] and Wikipedia articles, describing each sleep stage class. This information is processed and used together with the EEG signals to train the GAN network and an AdaBoost model for predicting the original EEG signal features and the plausible sleep stage classes. The main model comprises a shallow CNN and a bidirectional LSTM for feature extraction. The weighted softmax loss is applied to the FRL in order to alleviate class-imbalance problems. The results show that the FRL can extract effective EEG features and achieve state-of-the-art performance on some evaluation metrics. Applying weighted model fusion to the results of FRL and S2SL enhances the performance of the proposed method even further.

In another worth-stating work [Che+20], a combined bidirectional-LSTM and CNN, named Bi-LSTM-CNN, is used to perform sleep stage classification on multi-channel PSG data, specifically EEG, EOG and EMG. Three model configurations are tested; one simple LSTM-RNN, which takes as input features extracted from the signals after some preprocessing steps. The second model tested is a bidirectional LSTM, taking the same extracted features as input. Finally, the first layer of the LSTM is replaced by a CNN; in this case the model takes as input the raw signal in order to automatically extract features and perform sleep classification afterwards. The acquired accuracies for the three models are 81.6%, 84.8% and 89.4%, respectively, showing a better performance with the Bi-LSTM-CNN and the automatically extracted features for the specific setup and dataset.

A more complicated architecture is proposed in [Pha+19], where a hierarchical recurrent neural network named SeqSleepNet tackles the task of sleep stage classification as a sequence-to-sequence problem. In the aforementioned model, data are received as a sequence of multiple epochs as input and all of their labels are classified at once. At the epoch processing level, the network consists of a filterbank layer tailored to learn frequency-domain filters for preprocessing and an attention-based recurrent layer designed for short-term sequential modelling. At the sequence processing level, a recurrent layer placed on top of the learned epoch-wise features for long-term modelling of sequential epochs. The model is trained in an end-to-end manner. Using EEG, EOG and EMG channels, the proposed method achieves a total accuracy, macro F1-score, and Cohen's kappa of 87.1%, 83.3%, and 0.815 respectively, for a 5-sleep stage classification.

An extension of SeqSleepNet is presented in [Pha+21], where the proposed model, namely XSleepNet, is capable of learning a joint representation from both raw signals and time-frequency images, getting multiple-view inputs. Different views may generalize or overfit at different rates while training, thus XSleepNet is trained such that the learning pace of each view is adapted dependent on their generalization/overfit behavior. The time-frequency stream is based on an RNN to extract epoch-wise features in the same manner as SeqSleepNet, while the raw stream is based on a CNN. The multi-view network is trained such that learning on the network stream that is generalizing well is accelerated while the overfitting one is discouraged. Using SeqSleepNet as a baseline and testing on 5-sleep stage classification task, the proposed model not only delivers more favorable results than the baseline, but also outperforms existing work on five databases of different sizes. One advantage of the proposed method is that it is generic enough to serve sleep analysis with other modalities other than EEG, EOG and EMG, especially when multimodal data are available. Also, it could be applicable on other applications where the target signals are inherently multi-view.

2.4.2 Wearable-based Works

In the field of wearable-acquired data, fewer publications exist, due to its newer nature. The two most common data used are heart rate and wrist actigraphy, derived from a wearable device. This type of data are used in [Bea+17], where a wrist-worn device is used (a *Fitbit Surge*), measuring movement through a 3D accelerometer and an optical pulse photoplethysmograph (PPG). The data are obtained during overnight recordings of 60 adult participants, wearing these devices on their right and left wrist simultaneously, in parallel to recordings for getting the scores of the sleep stages based on the standard AASM guidelines, on a 30s epoch level. The data undergo some preprocessing steps and features are extracted from the accelerometer and PPG sensors, reflecting movement, breathing and heart rate variability. A peak detector algorithm has been developed to find the peaks in the PPG signal; the time between PPG peaks (PP-interval) is taken as a surrogate for the RR intervals obtained from an ECG. The initial set of the extracted 180 features on a 30s epoch basis consists of motion-based features, heart-rate and breathing-based. The features are fed to an automated classifier together with the gold standard labels, for a 4-sleep stage classification of 'Wake', 'Light', 'Deep' and 'REM'. As a second experimental step, a standard recursive feature elimination is implemented to reduce redundancy between them, ending up with a set of 54 features, which provide good performance. Several types of classifiers are tested: linear discriminant classifiers, quadratic discriminant classifiers, random forests and support vector machine approaches. The LDA had the best performance, thus this is chosen for the final model. The overall Cohen's kappa for the leave-one-out cross validation on the left-hand recordings was 0.52 ± 0.14. This corresponds to an overall per-epoch accuracy of 69%. Interestingly, the most common misclassification errors are Light classed as REM, REM classed as Light, Deep classed as Light, and Wake classed as Light.

[Fon+17] uses heart variability measured by a photoplethysmography (PPG) combined with body movements measured with an accelerometer, to examine their accuracy in sleep stage classification compared to polysomnography (PSG) and traditional actigraphy. The study is based on three separately collected datasets, one used to train the sleep staging algorithm and two for validation. The training dataset consists of PSG measurements in parallel to a CE-marked logging device containing a PPG and three-axial accelerometer sensors, while the second validation set includes actigraphy measured with *Actiwatch Spectrum*, which uses a piezoelectric accelerometer to detect and log limb movements. Interbeat intervals are detected from PPG instead of a traditional ECG, and HRV features are extracted both on time and frequency domain. Those are combined with features related to body movements, calculated based on the three-axial accelerometer signal, while no respiratory features are available from the wrist-worn sensors. No PPG data are available in the training set, so the HRV features used to train the algorithm are estimated from ECG. Since the features are based on the distance between consecutive R-R intervals, they are essentially equivalent (in the absence of cardiovascular conditions) when computed from consecutive pulses measured from PPG. An algorithm similar to [Fon+15] is used for the evaluation of the proposed method, using linear discriminant analysis. In the final results, the sleep–wake classifier obtained an epoch-by-epoch Cohen's κ between PPG and PSG sleep stages of 0.55 ± 0.14, sensitivity to wake of 58.2 ± 17.3%, and accuracy of 91.5 ± 5.1%. κ and sensitivity were significantly higher than with actigraphy (0.40 ± 0.15 and 45.5 ± 19.3%, respectively). The 3-class classifier achieved a κ of 0.46 ± 0.15 and accuracy of 72.9 ± 8.3%, and the 4-class classifier, a κ of 0.42 ± 0.12 and accuracy of 59.3 ± 8.5%.

In [Zha+18b], a multi-learning feature technique is proposed and an RNN-based method is applied for 5-sleep stage classification. The dataset consist of 39 healthy subjects' recordings, both by a *Microsoft Band I* and the respective PSG labels. Firstly low-level features are extracted; for heart rate, temporal and frequency properties of the data are considered on a 10-epoch window. For actigraphy, the features are extracted within one epoch and the dominant cepstrum components of the first-order difference along the three axes are calculated and concatenated to form the actigraphy feature vector. The mid-level features are calculated based on the low-level ones. Mid-level feature learning pays more attention to analyzing compositions and exploring the inherent structure of signals. Sleep can be seen as comprising of different compositions, with varying weights among the different sleep stages. Thus, a bag-of-words (BOW) is incorporated for obtaining the mid-level sleep representations. K-means clustering is applied on the low-level feature set of all epochs and K cluster centers are extracted. Then, for each epoch the mid-level features are the reciprocals of the Euclidean distances from all cluster centers, expressing the weighted influence of each cluster center (sleep) composition to the current epoch. The final feature vector is formed by the concatenation of the low and mid-level features and z-score normalization is applied. A deep bidirectional-LSTM is used and 8-fold cross validation is conducted for unbiased performance of the algorithm. The best performances, weighted precision, recall, and F1 score are formed as 58.0%, 60.3%, and 58.2% in the resting group and 58.5%, 61.1%, and 58.5% in the comprehensive group using heart rate combined with actigraphy.

In [Zha+19] a new low-cost wearable multi-sensor system is presented, for acquiring the cardiorespiratory signals from subjects. The designed system measures ECG and breathing signals via three electrode patches to achieve the single ECG monitoring and a wrist oximeter, thus it does not lie in the standard wearable category of heart rate and actigraphy signals. Feature extraction is applied to acquire three novel features, being effective to detect the sudden variation of RR intervals; 152 features are extracted in total, both on time and frequency domain. A bidirectional-LSTM is used for 5-sleep stage classification, with the prediction accuracy being 80.25% on a large public dataset (417 subjects), and 80.75% on a private dataset of 32 enrolled subjects, respectively.

In [Wal+19] a mobile application is developed in order to collect raw acceleration data and heart rate from the *Apple Watch* worn by participants undergoing polysomnography, as well as during the ambulatory period preceding in lab testing. The collected data are used for extracting three crucial features, namely motion, local standard deviation in heart rate and "clock proxy", and testing them on several classifiers afterwards. The concept of circadian rhythm and how the changing circadian propensity for sleep over night influences the performance across classifiers is specifically taken into account. Sleep is governed by the well-described two-process model comprised of the circadian oscillator and homeostatic sleep drive. Additionally, an ultradian cycle of alternating non-rapid eye movement (NREM) and rapid eye movement (REM) sleep stages is superimposed on the two-process model. Thus, the "clock proxy" feature, representing simulated input to sleep from the circadian clock, is introduced alongside the traditionally incorporated measurements of motion and heart rate. To make trained classifiers backwards compatible with historical data collection methods, raw acceleration data is converted to activity counts using MATLAB. Regarding heart rate, the standard deviation in the window around the scored epoch is used as the representative feature and while this represents variation in heart rate, it is distinct from ECG-based definitions of HRV. Regarding the clock proxy feature, the well-validated mathematical model used for its computation requires light input in order to predict circadian phase; since no such data are available via the smartwatch, the steps data are imported instead, with a rationale that walking or running typically takes place in a lit environment. The imported steps data are used to infer a "typical" daily pattern of rest and activity, and are converted to estimate light via a simple steps-to-light function. The algorithms used are logistic regression, k-nearest neighbors, a random forest classifier and an MLP, which is the simplest form of neural network. Models are trained and tested using both Monte Carlo cross-validation, with parameters being tuned for each training dataset to minimize the risk of overfitting, and leave-one-out cross validation, to understand the subject variability in classifier performance. The generalization of the models is tested on data derived from the Multi-ethnic Study of Atherosclerosis (MESA), which consists of motion data from actigraphy-derived activity counts and heart rate via pulse oximetry from co-recorded PSG, achieving performance close to the wearable-based dataset. The “clock proxy” feature is derived from the ambulatory actigraphy recording for each MESA participant. The experimental results indicate that across every algorithm surveyed, performance is best when all available features—motion, heart rate, and clock proxy are used as inputs to the classifiers. The best performance is achieved using the MLP classifier, where for sleep-wake classification it scores 90% of epochs correctly, with 59.6% of true wake epochs (specificity) and 93% of true sleep epochs (sensitivity) scored correctly. Accuracy for differentiating wake, NREM sleep, and REM sleep is approximately 72% when all features are used. The generalization of the trained models on the MESA dataset indicates that it is possible to predict sleep with performance comparable to testing the private training dataset.

[Li+20] presents a novel ML unsupervised algorithm based on Hidden Markov Model (HMM) for sleep/wake classification, using only actigraphy features acquired by *Actiwatch*. PSG is used as reference for the performance evaluation. While actigraphy does not contain as rich information as PSG, it is useful when long-time and non-invasive monitoring is required. It is an individualized approach, meaning that it takes into account individual variabilities and analyzes each individual actigraphy profile separately into sleep and wake stages. Three methods are compared in this study: HMM, an unsupervised algorithm embedded into Actiwatch software and a pre-trained UCSD algorithm [Jea+01]. The Actiwatch software uses information such as 10 consecutive epochs below a pre-specified immobility threshold as the sleep start and consecutive epochs above a pre-specified mobility threshold as the wake start. All three methods tend to over-estimate sleep and under-estimate wake compared to PSG. The estimated HMM parameters can characterize individual activity patterns and sedentary tendencies that can be further utilized in downstream analysis. Actigraphy data is converted into activity counts. Assuming that the sequence of the observed activity counts is generated from an unobservable two-state Markov chain, with the two states being sleep and wake, in the sleep state the activity counts are mostly zeros with some low values, while in the wake state they are generally high with some low counts denoting sedentary behaviors. Thus, the activity counts follow different distributions under the sleep and wake states, and it is possible to infer the hidden sleep/wake states at each time point based on the observed count data. Although the activity count can be directly modeled using a Poisson or Negative Binomial distribution in the wake state and a zero-inflated Poisson distribution in the sleep state, the observed activity counts can range from 0 to 4,000 per epoch, and this large range poses both statistical and computational challenges in data analysis. For this reason, log-transformed values are considered and, empirically, HMM works well for them. Based on the obtained sequence of hidden states, the focus stays on the same-state sequences longer than 15 minutes and shorter sequences are smoothed out to ensure that it captures stable sleep durations. Using PSG as the reference, the accuracy is configured as 85.7% for HMM, 84.7% for the AS, and 85.0% for UCSD.

Chapter 3

Theoretical Background

3.1 Introduction

Machine Learning (ML) *is a branch of Artificial Intelligence (AI) and Computer Science (CS), which provides systems the ability to automatically learn and improve from experience, without being explicitly programmed* [Koz+96].

But what is really learning?

Learning has been an intimate process of human since its early years of evolution. As described in the book [Har15], "Sapiens did not forage only for food and materials. They foraged for knowledge as well". Their survival depended on their constant development and learning of crucial skills and an understanding of their surrounding environment as well themselves. This kind of learning is known as *experiential*, in other words defined as "learning through reflection on doing" [Fel11]. The notion is believed to be initially introduced by Aristotle in the *Nicomachean Ethics*, quoting "for the things we have to learn before we can do them, we learn by doing them" [Ari11]. This primitive procedure has been advancing together with human for over 2 million years now, taking different shapes and forms; leading to today's structure of learning, which is considered as "the process of acquiring new understanding, knowledge, behaviors, skills, values, attitudes, and preferences" [Gro10].

But learning has not been only a human characteristic. It is a standard behavior in animals, allowing them to adapt in a constantly changing environment, increasing their chances of survival. Oftentimes, this process of "learning by doing" even comes in contradiction to their genetically inherited, innate knowledge and can be beneficial in case their innate behavior is disadvantageous for them. A similar habit is detected in plants and how they respond to external cues, a function necessary for their endurance. In a series of experiments done on the garden pea (Pisum sativa) presented in [Gag+16], Monica Gagliano aimed in differentiating between innate phototropism behavior and associative learning behaviors, based on the observations of a plant's directional growth to maximize its capture of sunlight.

Thus learning has been a natural process for all living creatures, amongst them human, since the beginning of life. Through this, we have acquired the knowledge to reach the current levels on our primary aspects of life, but also to achieve great advancements in science and technology; and this is an ongoing procedure. It is a logical continuation to consider applying our inherent way of learning to those fields as well. But this turns out to be not-so-easy to implement.

Artificial Intelligence as an academic discipline was introduced in 1959 at a workshop at Dartmouth College. The previous years had already been an outburst in the scientific world, with the discovery of the Church-Turing thesis, indicating that digital computers can stimulate any process of formal reasoning [Ber01]. In parallel, new discoveries in neurobiology, information theory and cybernetics made the idea of building an electronic brain seem feasible for researchers. McCullouch and Pitts in 1943 designed the "Turing-complete" artificial neurons [Rus10], considered to be the first work generally recognized today as AI. However, it was not until the 1980s, after two "AI winters", when artificial intelligence was finally established.

Machine learning (ML) reorganized as a separate field in the 1990 and started to grow independently. It shifted from the logical, knowledge-driven approach inherited from AI, to a data-driven one, with algorithms incorporating probability theory and statistics in order to solve practical problems, in terms of providing services. Two major factors influenced this shift, one being Big Data; the amount of data had outgrown in such a scale that new approaches were brought to life by practical necessity of how they would be handled, rather than pure scientific interest [Fra20]. The second factor was the exponential increase of the computational efficiency with the introduction of GPUs as units, where the machine learning algorithm calculations would be made, as well the advancements in computing parallelization and better memory handling. A milestone in this new era was in 1997, when the IBM-developed Deep Blue computer won against the world's chess champion Garry Kasparov. Since then, the progress in the machine learning and AI field has been tremendous, with applications varying between computer vision, speech recognition, natural language processing, medical imaging, data analysis, fraud detection, recommendation systems and the list continuous.

3.2 Machine Learning Definition

The structure of ML in the way it has been formed today, is very close to the natural process of learning as described in the previous section.
As stated in T. Michell's book of 1997 [Mit97]:

*A computer program is said to learn from **experience E** with respect to some class of **tasks T** and **performance measure P** if its performance at tasks in **T**, as measured by **P**, improves with **experience E**.*

Thus, the main goal of ML is for a machine to achieve a good level of generalization on a specific task, based on its experience. Having a learning dataset, where the training examples come from a generally unknown probability distribution, the ML algorithm has to build a model for this space which can accurately predict any new given cases. As human observe their surrounding environment and analyze the information they perceive in order to extract conclusions about it, in the same manner an ML model discerns discrete features from the given data, in order to make a mathematical representation of them to resolve a specific task; and thus be able to correctly process unseen data afterwards, and extract results about them. Also, as human learn by repeating an action and improving from their faults, the same principle idea is applied on ML systems learning the features' representation. The main objectives ML is handling are two, the first being data classification and the second one being prediction making for future outcomes based on the input data.

Given the input signal's nature, which is passed through an ML system, and the way the "feedback" is returned for the system to learn by itself and improve its performance, ML approaches are divided in three main categories:

- **Supervised learning**: the computer is presented with the input data together with the corresponding desired outputs and the system has to learn a mapping for the inputs to the correct outputs.
- **Unsupervised learning**: in this case the computer is presented only with the input data and no labels are given for them; it has to learn an internal structure on its own.
- **Reinforcement learning**: the computer actively interacts with a constantly changing environment (dynamical), such as playing a game against an opponent or driving a vehicle, having to perform a certain goal. In this case the feedback it receives while navigating the problem's space is in the form of a reward, which it tries to maximize.

3.3 Supervised Learning

Supervised Learning (SL) is the ML task of learning a function that maps an input X to an output Y based on example input-output pairs [Rus10]. The data for training are given as a set of training examples $D = \{f(\mathbf{x_n}, \mathbf{y_n}), n = 1, ..., N\}$, and each pair has an input object x_i, typically a vector known as the *feature vector*, and its corresponding output value or *label* y_i. An *objective function* or *loss function* $L : X \times Y \rightarrow \mathbb{R}^d$ is iteratively optimized so that the SL algorithm learns a mapping function $f : X \times Y$, which can be used for predicting the correct output $\hat{\mathbf{y}} = f(\mathbf{x})$ of an instance $\boldsymbol{x}$. The term "supervised" derives from the concept of a teacher supervising the learning process. As the algorithm iteratively learns from the training data, it makes predictions and in each step it is corrected by the teacher, based on the loss function. Given a training example $(\mathbf{x_i}, \mathbf{y_i})$, the loss for predicting the value $\hat{\mathbf{y}}_\mathbf{i} = f(\mathbf{x_i})$ is calculated as $L(\hat{\mathbf{y}}_\mathbf{i}, \mathbf{y_i})$. When an acceptable level of performance is achieved and the loss reaches a desired threshold, the learning process stops. The accuracy of the SL algorithm can then be defined as its potential in making accurate predictions of inputs unseen during the training: $\tilde{\mathbf{x}} \notin D$. This is known as the *generalization ability* of the algorithm.

Having gathered a dataset of training examples, a sequence of specific steps has to be followed for the solution of an SL problem.

- The data have to be split into two separate sets, one called *training* and one *test set*, with a ratio of around 80% - 20% of the total amount of data.

- The input feature representation has to be determined, choosing a suitable feature vector which adequately describes the object and contains enough information to accurately predict the output. The *curse of dimensionality* has to be taken into consideration though, to avoid various unwanted phenomena that arise in high-dimensional spaces.
- After choosing the correct feature representation, a learning algorithm has to be determined, together with the structure of the learning function. Usually, there are several control parameters in SL algorithms that have to be adjusted as well. The *cross-validation* method is applied for this purpose, or else a subset of the training set is selected, around 10% of it, called *validation set*, and the algorithm's performance is optimized on it.
- The trained model's *accuracy* is finally measured through the evaluation of the learning function on the test set.

3.3.1 No-Free Lunch Theorem

The selection of the correct algorithm is essential for the solution of an SL problem and the No Free Lunch (NFL) theorem strongly highlights this. In the NFL theorem is stated that, in mathematical problems lying in the search space of probability density function like SL does, the computational cost of finding a solution will be the same for any solution method, when averaged over all problems in the class [Wol96]. The "cost of lunch" reflects the performance of a procedure in solving a problem and "no free lunch" means that there is no improvement in the cost when the probability distribution on problem instances is such that all problem solvers (algorithms) have identically distributed results; prior information is needed to match the procedures to the problems to achieve performance improvement. Thus, no single ML algorithm is a universal, best-performing solution for learning all possible target functions of an SL problem.

The term **inductive bias** or **learning bias** is introduced to describe the set of *assumptions* about the nature of the target function, for the learning algorithm to be able to predict a certain target output given inputs that it has not previously encountered [Mit80; GD95]. In ML, the goal is the construction of an algorithm with the ability to predict a specific type of target output. This is done by exposing the algorithm to training examples which demonstrate the intended relation between the input and output values. However, lacking the additional assumptions to be made for the data, the algorithm might not be able to solve the problem of the approximation of the correct output for examples that have not been seen during training. This is due to the probability of those situations to have an arbitrary output value.

3.3.2 Bias - Variance Tradeoff

The bias-variance tradeoff explores the relationship between two major sources of error for an SL model, namely *variance error* and *bias error*.
The expected generalization error of a learning algorithm can be decomposed into three interpretable terms [Has+09]. Considering a learning variable denoted as Y and its covariates denoted as X, there is a relationship relating one to the other so that $Y = f(X) + \varepsilon$, where ε is an error term following the normal distribution $\varepsilon \sim \mathcal{N}(0, \sigma_\varepsilon)$.

The typical ML approach of estimating the relation $f(X)$ is by creating a model $\hat{f}(X)$ of it using an SL algorithm. The most common loss function which can be used is the mean squared loss, and in this case the expected test error can be written as:

$$\begin{gathered} Err(X) = E[(\hat{f}(X) - Y)^2] = E[(\hat{f}(X) + \overline{f}(X) - \overline{f}(X) - Y)^2] \rightarrow \\ Err(X) = E[(\hat{f}(X) - \overline{f}(X))^2] + 2E[(\hat{f}(X) - \overline{f}(X))(\overline{f}(X) - Y)] + E[(\overline{f}(X) - Y)^2] \end{gathered} \tag{3.3.1}$$

$$\begin{gathered} E[(\overline{f}(X) - Y)^2] = E[\overline{f}(X) - \overline{Y} + \overline{Y} - Y)^2] = \\ E[(\overline{f}(X) - \overline{Y})^2] + E[(\overline{Y} - Y)^2] + 2E[(\overline{Y} - Y)(\overline{f}(X) - \overline{Y})] \end{gathered} \tag{3.3.2}$$

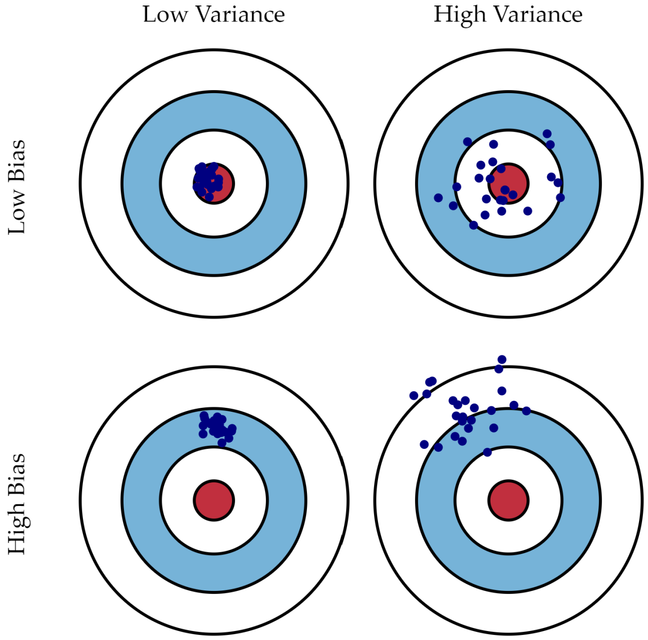


Figure 3.3.1: Graphical illustration of bias and variance [02].

The multiplying terms are zero:

$$
\begin{aligned}
2E[(\overline{Y}-Y)(\overline{f}(X)-\overline{Y})] &= 0 \\
2E[(\hat{f}(X)-\overline{f}(X))(\overline{f}(X)-Y)] &= 0 \qquad (3.3.3)
\end{aligned}
$$

Considering that $\overline{f}(X)=E[\hat{f}(X)]$, $E[f(X)]=f(X)$ since $f(X)$ is constant for a particular X, $Y=f(X)+\varepsilon$, $E[\varepsilon]=0$, $E[\varepsilon^2]=\sigma_\varepsilon^2$, and combining the Equations 3.3.2, 3.3.2, 3.3.2, we have the final expected error:

$$
\begin{aligned}
Err(X) &= E[(\hat{f}(X)-\overline{f}(X))^2]+E[(\overline{f}(X)-\overline{Y})^2]+E[(\overline{Y}-Y)^2] \rightarrow \\
Err(X) &= E[(\hat{f}(X)-E[\hat{f}(X)])^2]+(E[\hat{f}(X)]-f(X))^2+\sigma_\varepsilon^2 \rightarrow \\
Err(X) &= Variance + Bias^2 + Noise \qquad (3.3.4)
\end{aligned}
$$

- **Bias** is the *simplifying assumptions* needed to be made in order for the target function to be easier to learn. It is connected to the inability of a model to capture the true relationship f between Y and X, and it is a type of error that can be reduced. In general, a higher bias makes algorithms faster to learn and easier to understand, but less flexible, e.g. linear algorithms, which cannot perform well on more complex problems due to their simplicity.

- **Variance** refers to the amount of change, which is introduced in the estimate of the target function, when different training data are used. This is a kind of error that can also be improved. High variance implies that the trained model cannot perform well on previously unseen data (the test set), and this is an undesirable characteristic that more flexible, e.g. nonlinear models share.

- Finally, **noise**, otherwise known as **irreducible error**, is a kind of error that fundamentally cannot be reduced. It corresponds to noise of the true relationship itself, and represents ambiguity in the data distribution and feature representation.

The goal for any SL algorithm is to achieve good prediction performance through the minimization of both bias and variance, however this is a constant battle to counterweight between the two. Intuitively, bias is

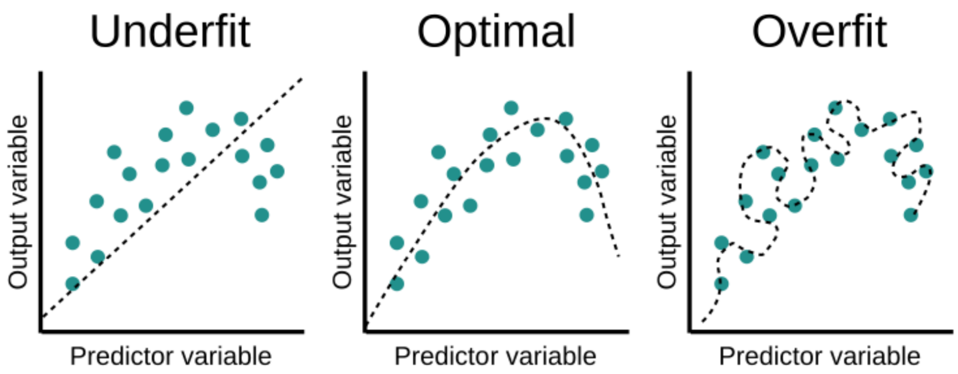


Figure 3.3.2: Example scatter plot of predictions from model being underfit, optimal and overfit [03].

reduced by using only local information, where accurate assumptions can be made, whereas variance can be reduced only by averaging over multiple observations, meaning that the data are selected from a larger region.

Figure 3.3.1 depicts a bull's eye diagram, where the center of the target indicates a model that makes perfect predictions of the correct values. Repeating the entire training of the model over a several times' course, the portion of hits during testing greatly depends on the chance variability in the training data gathered. In cases where the training data have a good distribution, the predictions are close to the bull's eye, while in cases where many outliers or non-standard values exist, the predictions tend to be further away from the center.

3.3.3 Overfitting and Underfitting

In Figure 3.3.1 can also be seen a tendency of low bias and high variance leading to predictions close to the center, but far away from each other, while high variance and low bias give predictions which are close to each other, but away from the center. Those trends are called *overfitting* and *underfitting* respectively. Another visual example of overfitting and underfitting is shown in Figure 3.3.2. In the second case the model is unable to capture the underlying pattern of the data, while at the first case it captures it in such a detail that part of the inherent noise is also contained.

If a model is too simple, without a lot of parameters, it mostly has high bias and low variance, whilst when it has a large number of parameters and is in general complex, the opposite happens. Thus bias is reduced and variance is increased in relation to model complexity. For example, as more polynomial terms are added to a linear regression, the greater the resulting model's complexity will be. In other words, as it is can be seen in Figure 3.3.3, bias has a negative first-order derivative in response to model complexity, while variance has a positive slope.

Understanding bias and variance is critical for understanding the behavior of prediction models, but in practice, what the attention has to be led on is the overall error, not the specific decomposition. The desired spot for every model is where the increase in bias is equivalent to the reduction in variance and there is not an analytical way to detect this. Instead, an accurate measure of prediction error has to be selected, and this needs to be very considerate, and, by exploring different levels of complexity to pick up the best one for the specific problem.

3.3.4 Classification

This kind of problem has discrete categories as the output variables, also called classes, such as "car" or "bicycle", and the goal for the ML algorithm is to assign the test data, each one to the correct class. For example, given retinal medical images from healthy subjects and subjects with glaucoma or diabetic retinopathy, the goal is to detect those deceases in new images and thus claim whether the subject falls under the healthy category or to one of the diseases. A common practice for classification models is to predict a

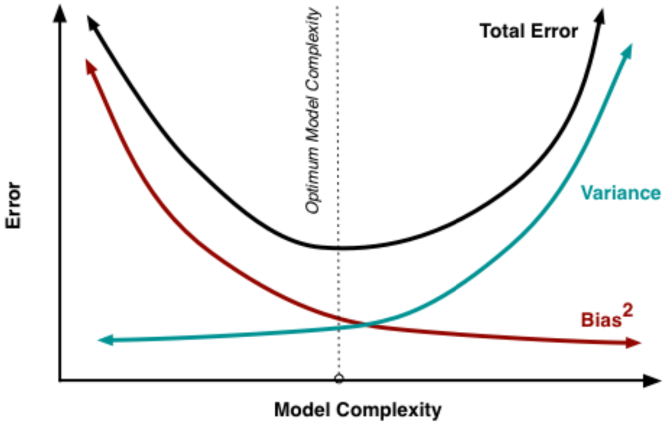


Figure 3.3.3: The variation of Bias and Variance with the model complexity. This is similar to the concept of overfitting and underfitting. More complex models overfit while the simplest models underfit [02].

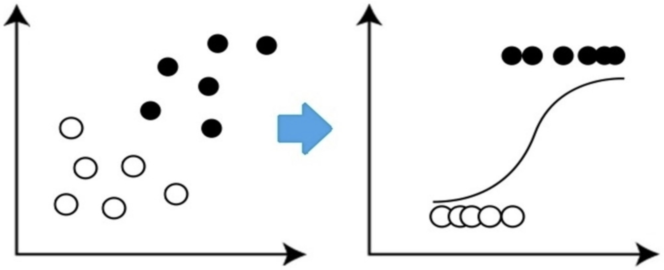


Figure 3.3.4: Logistic Regression [04].

vector of continuous values as the probabilities of a given example to belong to each of the classes. The predicted probabilities can be interpreted as the likelihood of the example to belong to each class, and the one with the highest probability is chosen as the model's prediction. Thus, in classification problems the input space is divided into classes based on different characteristics of the feature vectors of the training data and the mapping function identifies the *decision boundary*, which is the line where the different classes meet.

Some examples of standard classification algorithms are:

Linear Classifiers
In this group of algorithms, amongst others, belong *logistic regression* and *naive bayes classifier*, each one having its own characteristics.

Logistic regression is a fundamental classification algorithm, which uses one or more independent variables to determine a binary outcome. The probabilities describing the possible outcomes of a single trial are modeled using a logistic function. The coefficients of its equation are estimated by the "maximum likelihood estimation" (see Figure 3.3.4). It is a simple and efficient, with low variance, algorithm, however it has poor performance in handling large categorical data and it also assumes that there are no missing values and the predictors are independent from each other.

Naive Bayes calculates the possibility of whether a data point belongs within a certain category or does not. It is an extension of the Bayes theorem wherein each feature assumes independence. Being an easy and quick way to predict the class of the dataset, it can be used for multi-class prediction, provided the validness of the independence assumption. However this is difficult to achieve in real-life data.

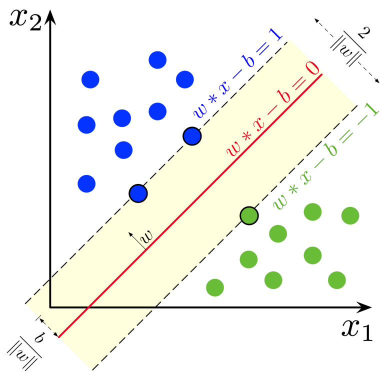


Figure 3.3.5: Support Vector Machine (SVM) [05].

Support Vector Machines

SVM are based on the concept of decision planes that define decision boundaries. A decision plane (hyperplane) separates between a set of objects having different class memberships. An SVM performs non-probabilistic classification and other tasks by finding the hyperplane (or set of hyperplanes in a high- or infinite-dimensional space), that maximizes the margin between two classes with the help of support vectors. The margin is defined as the perpendicular distance between the decision boundary and the closest of the data points, as shown in Figure 3.3.5. Maximizing the margin leads to a particular choice of decision boundary, the location of which is determined by a subset of the data points, known as *support vectors*.

In order for the SVM to perform non-linear classification, the *kernel trick* was developed, implicitly mapping the SVM inputs into high-dimensional feature spaces. Instead of using $\mathbf{x} \in \mathbb{R}^m$ to describe the inputs of an SVM algorithm, a feature map is used $\phi(\mathbf{x})^T \in \mathbb{R}^M, M \gg m$. It is observed that many ML algorithms can be written exclusively in terms of dot products between the training examples. Also, the kernel theorem directs that a feature map $\phi(.)$ can be defined using a kernel function $k(.,.)$ such as $k(\mathbf{x}, \mathbf{x}') := (\mathbf{x}^T\mathbf{x}')^2 = \phi(\mathbf{x})^T\phi(\mathbf{x}')$. Extending this idea into high-dimensional spaces, if the data is linearly separable in $\mathbb{R}^M$, then the classification problem can be solved as follows. The optimal b^* of the hyperplane is derived through:

$$b^* = t_j - \sum_{i=1}^{n} a_i^* t_i k(\mathbf{x}_i, \mathbf{x}_j), \text{ for any } a_j^* > 0 \tag{3.3.5}$$

and the prediction for a new data point $\mathbf{x}$:

$$sign(\mathbf{w}^{*T}\phi(\mathbf{x}) + b^*) = sign(\sum_{i=1}^{n} a_i^* t_i k(\mathbf{x}_i, \mathbf{x}) + b^*) \tag{3.3.6}$$

where a is a vector of coefficients.
The kernel-based learning SVM function is equivalent to transforming the data space into a linear representation by applying $\phi(\mathbf{x})$ on all the input data, and then learning a linear model on the newly transformed space.

Kernel Estimation

One typical algorithm from this group is *k-nearest neighbors* (KNN), which is non-parametric and it classifies data points based on their proximity and association prior available data. The algorithm does not attempt to construct a general internal model by making any assumptions about how the data is distributed, but instead classification is computed from a simple majority vote of the k nearest neighbors of each point, as it can be seen in Figure 3.3.6.

Decision Trees and Random Forest

Given a data of attributes together with its classes, a *decision tree* produces a sequence of rules that can be used to classify the data. Specifically, it is a form of algorithmic tree where each node represents a feature or attribute, each branch represents a decision or rule and each leaf defines an outcome, which can be either a categorical or continuous value. The idea is very straightforward to understand and little data preparation is required. However, it is prone to creating complex structures that do not generalize well.

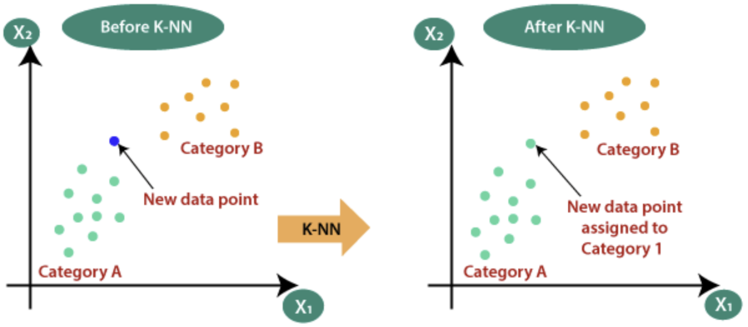


Figure 3.3.6: K-nearest neighbors (KNN) [02].

A way to improve this method's performance, is through merging together multiple uncorrelated decision trees to reduce variance and create more accurate data predictions. This structure is known as *random forest* and it facilitates the reduction of over-fitting while maintaining the intuitive nature of this form of classification approach.

3.3.5 Regression

There is one second category for SL problems, which differentiates in the sense that the output variables lie in a continuous space, or have to be real values. Those often refer to quantities such as amounts ("salary") or sizes ("weight"). In this case, the mapping function tries to find a correlation between a dependent variable (*response* variable) and one or more independent variables (*features*). The most common form of regression analysis is *linear regression*, which tries to fit the training data with the best hyper-plane that passes through the points in the input feature space, according to a specified mathematical criterion. Regression analysis is widely used for prediction and forecasting problems, such as house pricing or weather prediction. Another substantial use of regression, though not so widely spread as forecasting, is the detection of casual relationships between the independent and dependent variables, provided that this kind of relationships have a causal interpretation.

3.4 Unsupervised Learning

In Unsupervised Learning (UL), the data for training do not contain any corresponding output variables, or else labels. Thus the nature of the problem to be solved differentiates to learning internal patterns of untagged data. The algorithms are expected to learn an adequate representation of the dataset, finding its underlying structure, uncovering hidden similarities and grouping by them, as well expressing the dataset in a compressed format. Being able to handle a dataset without any labels can be very beneficial in cases of exploratory data analysis, cross-selling strategies or image recognition.

3.4.1 Clustering

For example, given a dataset consisting images of cats and dogs with no accompanying labels, and applying an unsupervised algorithm on them, it can unravel the images' hidden patterns of the two different categories (cats, dogs). The UL algorithm can thus accomplish the clustering task for those two groups based on the detected similarities between the images.

This is a representative example of to the first main category of unsupervised learning, which is **clustering**. Some typical algorithms for clustering are K-means, Principal component analysis (PCA) and hierarchical clustering.

3.4.2 Association

The second category is called **association**, and refers to the problems where the goal is to discover *association rules* that describe large portions of the data, thus finding relationships between variables in the dataset. A common use of this kind of unsupervised learning is for companies to understand consumption habits of customers, in order to improve their business strategies and recommendation engines, such as Spotify's "Discover Weekly" playlist [FS17; TY17].

3.4.3 Dimensionality Reduction

A final main application of unsupervised learning is **dimensionality reduction**. This is very useful in cases where the dataset contain a lot of redundant information with features overlapping in the input space, increasing the computational time and making the machine learning algorithms prone to overfitting. By applying dimensionality reduction as a preprocessing step, the dataset is reshaped so that the number of features is of manageable size while at the same time the integrity of the dataset is preserved as much as possible.

3.5 Reinforcement Learning

Reinforcement Learning (RL) is a type of machine learning technique that enables an agent to learn in an interactive environment by trial and error using feedback from its own actions and experiences. Though both supervised and reinforcement learning use mapping between input and output, unlike SL where the feedback provided to the agent is a correct set of actions for performing a task, RL uses rewards and punishments as signals for positive and negative behavior. Hence RL does not need labeled input/output pairs to be presented, or sub-optimal actions to be explicitly corrected. Instead, a reinforcement *agent* decides what action need to be taken for performing a task, in a game-like situation. Compared to unsupervised learning, RL differs in terms of goals. While the goal in Unsupervised Learning is to detect similarities and differences between data points, in the case of RL the goal is to find a suitable action model capable of maximizing the total cumulative reward of the agent. Due to the absence of training data, an RL model is bound to learning from its own experience. During training, the model will return a state and the agent will decide to award or punish it based on the current output, as its goal is to maximize the total reward. Since RL requires a lot of data, it is most applicable in domains where simulated data is readily available, such as game-play and robotics. Applications of RL can be found in training autonomous vehicles or to train a virtual runner for upgrading the prosthetic legs technology [Kid+18]. Other applications of RL include abstractive text summarization engines, dialog agents (text and speech), which can learn from user interactions and improve with time, learning optimal treatment policies in healthcare and RL based agents for online stock trading.

3.6 Neural Networks

Artificial Neural Networks has been a specific part of machine learning, and, while existing for already some decades now, it has gradually gained ground over the course of recent years; being one of the most drastic methods for solving ML problems nowadays, and a state-of-the-art technology with applications ranging in almost every aspect of modern life. NNs lie in the same trend of nature-inspired discoveries in last century's technological world, and, as indicated by their name, are computing systems inspired by the biological neural networks that constitute animal brains.

An ANN consists of a collection of connected computational units or nodes, which are called *artificial neurons*, and are supposed to loosely model neurons in a biological brain. Each neuron functions by receiving a collection of signals, which is actually a set of real numbers given as separate, parallel inputs, and processes it by applying a non-linear function on the sum of its inputs. Neurons are connected with each other by structures called *edges*, and like the *synapses* in a biological brain, can transmit a signal through them. Both neurons and edges have a *weight*, which adjusts during the process of *training the network*, as the learning progresses. The purpose of the weight is to increase or decrease the strength of the signal at a connection, hence to control the amount of the specific signal's impact during the non-linear function computations. It is common for neurons to also contain a threshold for the aggregated signal, after which it is allowed to be

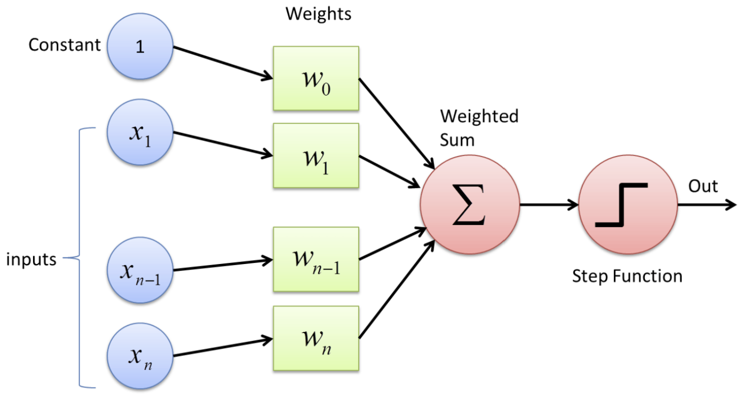


Figure 3.6.1: Perceptron visualization, where w_0 is equal to the bias [06].

forwarded to the succeeding neurons. Typically, neurons are assembled into layers, which perform different transformations on their outputs, justifying different purposes for the network. Details for the architectures that are used in this thesis will be discussed in the following sections, after firstly covering some fundamental topics of neural networks.

3.6.1 The Perceptron

The first artificial neural network named *perceptron* was introduced in 1958, by psychologist Frank Rosenblatt [RP57; Ros57], funded by the United States office of Naval Research [Ola96]. Rosenblatt heavily inspired by the biological neuron and its ability to learn, created an algorithmic implementation of it, consisting of one or more inputs, a processor and only one output. Perceptron is made up with only one node, and it belongs to the SL category of binary classifiers, meaning that it is able to solve the task of binary classification, provided linearly separable classes [Figure 3.6.1]. Although today perceptron is widely recognized as an algorithm, it was initially intended as an image recognition machine. It gets its name from performing the human-like function of perception, seeing and recognizing images.

Given a real-valued vector $\mathbf{x} \in \mathbb{R}^n$, where n is the number of inputs for the perceptron, a single binary valued output $\mathbf{y} \in \{0, +1\}$ and a real-valued vector of weights $\mathbf{w} \in \mathbb{R}^n$, the perceptron learns a threshold function which maps the input $\mathbf{x}$ to the output value as denoted below:

$$f(\mathbf{x}) = \begin{cases} 1 & \text{,if } \mathbf{x} \cdot \mathbf{w} + b > 0 \\ 0 & \text{,otherwise} \end{cases} \tag{3.6.1}$$

where $\mathbf{x} \cdot \mathbf{w}$ is the dot-product $\sum_{i=1}^{n} w_i x_i$ and b is the bias, which does not depend on any input value. The step function was historically being used as threshold and it can be considered as a very simple activation function for the dot-product term $\mathbf{x} \cdot \mathbf{w} + b$ and the bias allows shifting the activation function up or down. Thus, it is a simple mapping of summed weighted input to the output of the neuron. The term *activation function* is used because it governs the threshold at which the neuron is activated and the strength of the output signal.

3.6.2 Multi-Layer Perceptron

The idea of perceptron can be generalized into creating more complex networks consisting of groups of nodes, forming different *layers*: an *input layer*, one or more *hidden layers* and an *output layer* which gives the final result. This kind of architecture is known as *Multilayer perceptron* (MLP) and is depicted in Figure 3.6.2. One key difference from the original simple perceptron though, is that MLP utilizes non-linear activation functions for its nodes; this allows it to distinguish data that is not linearly separable, which was a necessary condition for perceptron. Also, if an MLP had linear activation functions in all of its neurons, then linear algebra shows that this architecture can be reduced to a two-layer input-output model, no matter how many hidden layers initially existed.

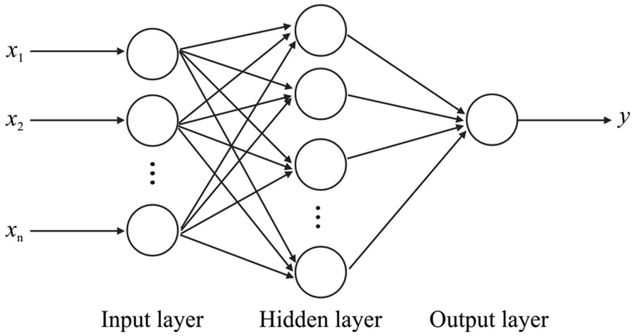


Figure 3.6.2: Multilayer Perceptron visualization, where each node, except for the input ones, is a neuron that uses a nonlinear activation function [07].

The input layer of an MLP receives the signal to be processed and does not apply any transformations to it. Provided an input vector $\mathbf{x} \in \mathbb{R}^D$ and an MLP with K hidden layers, each layer with M_K nodes, the input layer receives the vector $\mathbf{x}$, one value at each node, and distributes them to the first hidden layer, for $k = 1$. Each node of the K hidden layers applies an activation function $h(.)$ on the received input data, in a manner similar to the perceptron.
For the first hidden layer, the equations for node j, $j = 1, ..., M_{(1)}$, are formed as:

$$a_j^{(1)} = \sum_{i=1}^{D} W_{ij}^{(1)} x_i + b_j^{(1)}$$

$$z_j^{(1)} = h(a_j^{(1)}) \tag{3.6.2}$$

where $W_{ij}^{(1)}$ are the weight parameters and $b_j^{(1)}$ is the bias of the node.
Those parameters form the *weight matrix* and *bias vector* of the layer. The quantity $a_j^{(1)}$ is known as the *activation* of node j of the 1^{st} layer and is passed through the node's activation function to give the output $z_j^{(1)}$.
Thus, for the k_{th} layer, since it receives the previous layer's outputs as its input signals, the activation equations are shaped as follows:

$$a_j^k = \sum_{i=1}^{D} W_{ij}^k z_j^{k-1} + b_j^k$$

$$z_j^k = h(a_j^k) \tag{3.6.3}$$

Activation Functions

The choice of activation function depends on the nature of the problem to be solved and the assumed distribution of the training data. The linear or identity activation function, is the simplest case and is not capable in solving more complex problems; this is where nonlinear activation functions are acquired. Nonlinearity allows the model to generalize or adapt with variety of data and to differentiate between the output. Some typical activation functions can be seen in Figure 3.6.3.

When the desired prediction of the model is a probability, then usually the *sigmoid* or *logistic* activation function is used, since its output is limited in the range of $[0, 1]$. *Softmax* is a more generalized logistic activation function, which is used for multiclass classification. *Tanh* (hyperbolic tangent Activation Function) is also sigmoidal (s-shaped) but ir ranges from $(-1, 1)$. Its main advantage is that the negative inputs will be mapped strongly negative and the zero inputs will be mapped near zero in the tanh graph, and it is mainly used for binary classification problems. *ReLU* (rectified linear unit) is the most commonly used activation function today and *Leaky ReLU* is used to solve a problem known as *dying ReLU*, where all the negative values in the simple ReLU activation immediately become zero, decreasing the ability of the model to fit or train from the data properly.

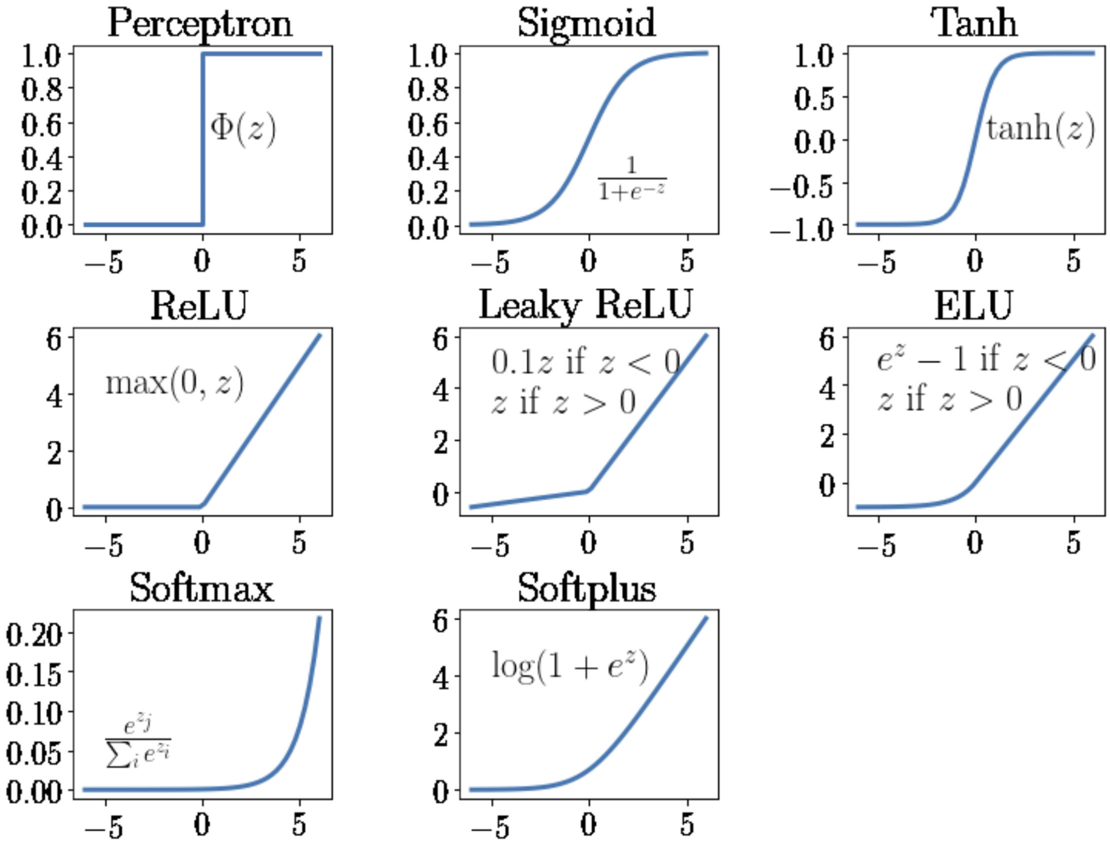


Figure 3.6.3: Common activation functions [08].

3.6.3 Training Algorithms

The multilayer perceptron lies under the category of *feed-forward algorithms*, where the information moves forward, starting from the input nodes, going through the hidden nodes -if any- and to the output ones. In order for this kind of network to learn internal representations of the training data, several techniques have been developed; and one of the most popular ones is *backpropagation* [RHW86].

Backpropagation

The idea behind this method is the backward-propagation of the error that occurs between the network's output against the true label of the input data. The goal is to optimize the network's weights, by adequately iterating through the training data and in each iteration improving the node weights based on the current error from the loss function. The problem of training the neural network would thus be considered as an optimization problem of the form $\min\limits_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$.

The term backpropagation is used under two perspectives; the first one being to describe the whole algorithm for fitting a neural network. The second refers to the computation of the derivative from the loss function, given a single input-output example, with respect to the weights of the network, with great efficiency. The *chain rule* is utilized and the gradient is computed in one layer at a time, iterating backward from the last layer, to avoid redundant calculations of intermediate terms. Using the notation from Equations 3.6.2, the chain rule formula calculating the derivative of the loss function $L(\hat{\mathbf{x}}, \mathbf{x})$ is:

$$\frac{\vartheta L}{\vartheta w_{ij}} = \frac{\vartheta L}{\vartheta z_j}\frac{\vartheta z_i}{\vartheta a_j}\frac{\vartheta a_j}{\vartheta w_{ij}} \tag{3.6.4}$$

Gradient Descent

In mathematical optimization, this kind of minimization problems is solved by algorithms using the gradient method, where the search directions are defined by the gradient of the function at the current point. **Gradient descent**, which is the most common approach, is an iterative algorithm used to find the local minima of a *differentiable* function and it originates from the fact that a function $f(\mathbf{x})$ reaches faster the local minima if $\mathbf{x}$ moves towards the direction of the negative gradient of f at $\mathbf{x}$, $-\nabla f(\mathbf{x})$. Given a neural network and its weight matrices W, the weights can be updated in each iteration as follows:

$$W = W - \gamma \nabla_W L(\mathbf{x}) \tag{3.6.5}$$

where γ is the learning rate, a tuning parameter that determines the step size at each iteration while moving toward the minimum of the loss function L.

The above Equation 3.6.5 describes vanilla gradient descent, or else **batch gradient descent** (BGD), where the calculation of the gradients is done on the whole dataset, before the update of the weights. However, this approach is very computational demanding and time consuming, making it impractical for real-life applications, where the amount of data is large, the network architectures are deep and there is need for online learning.

To overcome these limitations, a stochastic approximation of the algorithm is introduced, named **stochastic gradient descent** (SGD), where the actual gradient is replaced by an estimate thereof, calculated by a randomly selected data point. Thus, the equation for a training example x^i, y^i is formed as follows:

$$W = W - \gamma \nabla_W L(\mathbf{x}^i, \mathbf{y}^i) \tag{3.6.6}$$

The stochastic aspect of SGD makes it prone to overshooting, missing convergence to the exact minimum, while on the other hand gives the freedom of finding potentially better local minima than the standard one BGD converges to. Also, it has been shown that, when the learning rate is slowly decreased through iterations during training the network, SGD closely follows the performance of BGD, converging to a local or the global minimum, for non-convex or convex optimization spaces.

One last common approach lies in between the two aforementioned gradient descent methods, known as **mini-batch gradient descent**. It dictates for each iteration to sample a small subset instead of a single data point, called mini-batch, consisting of n data points. Then, the weight update equation becomes:

$$W = W - \gamma \nabla_W L(\mathbf{x}^{i,i+n}, \mathbf{y}^{i,i+n}) \tag{3.6.7}$$

However, the use of grouped data for the calculation of the derivative for the network's weights update has to be mindful. Since the learning rate is applied to the whole mini-batch at once, it is prone to erroneous generalizations in case the data are sparse and the features appear with varying frequencies, where there should be a different handling in each feature category. Also, the choice of a suitable learning rate value is difficult to achieve and might demand the exhaustive search to find an appropriate one.

Adaptive Moment Estimation - Adam

Adam [KA15] is a method for efficient stochastic optimization, computing adaptive learning rates for each of the network's parameters. It combines two stochastic gradient descent approaches, Adaptive Gradients, and Root Mean Square Propagation (RMSprop). In addition to storing an exponentially decaying average of past squared gradients v_t, like RMSprop, Adam also keeps an exponentially decaying average of past gradients m_t, similar to momentum. Momentum can be perceived as a ball running down a slope, whilst Adam is closer to a heavy ball with friction, preferring flat minima in the error surface [Heu+17]. The decaying averages of past gradients and past square root gradients m_t and v_t respectively, are computed as:

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t \\ v_t &= \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \end{aligned} \tag{3.6.8}$$

where m_t and v_t are the estimates of the first and second *moment* of the gradients (mean and variance respectively). Those variables are initialized as vectors of $0's$, thus have a natural tendency towards the zero value. To overcome this bias, the bias-corrected first and second moment estimations are proposed:

$$\begin{aligned} \hat{m}_t &= \frac{m_t}{1-\beta_1^t} \\ \hat{v}_t &= \frac{v_t}{1-\beta_2^t} \end{aligned} \tag{3.6.9}$$

Thus, the Adam update rule for the network's parameters is formed as:

$$w_{t+1} = w_t - \frac{\gamma}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t \tag{3.6.10}$$

3.7 Deep Learning

Deep learning is an advanced form of Artificial Neural Networks, where the term "deep" refers to the use of multiple layers in the network. It is a modern variation which is concerned with an unbounded number of layers of bounded size, permitting practical application and optimized implementation, while retaining theoretical universality under mild conditions. In deep learning the layers are also permitted to be heterogeneous and to deviate widely from biologically informed connectionist models, for the sake of efficiency, trainability and understandability, whence the "structured" part. The use of multiple layers allows the network to progressively extract higher-level features from the raw input; for example, in image processing, lower layers may identify edges, while higher layers may identify the concepts relevant to a human such as digits or letters or faces. The advancement of different network architectures has been tremendous and two of the most common ones will be discussed in the following sections, as they are used for this thesis' experiments.

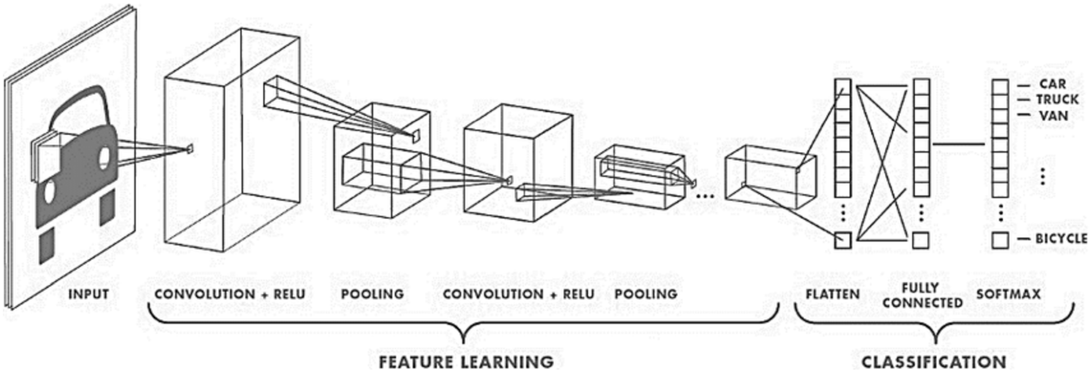


Figure 3.7.1: Convolutional neural network architecture for image classification [09].

3.7.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) is a specialized type of neural network for handling data with a known grid-like topology. This kind of data includes images, which can be perceived as a 2-D grid of pixels, but also time-series data, which can be thought of as a 1-D grid, taking samples at regular time intervals [GBC16]. The name of this network category derives from the mathematical operation of *convolution*, which is utilized in some of the network's layers in place of general matrix multiplication, as done in the simpler case of fully-connected networks.

The Convolution Operation

The mathematical operation of convolution expresses the weighted average over a period of time and it consists of an *input* argument, a *kernel* defining the weighting factor and an output, oftentimes referred to as *feature map*. The continuous and discrete time convolution is defined by the following equations respectively:

$$\begin{aligned} s(t) &= (x * w)(t) = \int x(a)w(t-a)da \\ s(t) &= (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a) \end{aligned} \tag{3.7.1}$$

In machine learning, the input is usually a multidimensional array of data, and the kernel is also a multidimensional array of parameters to be adapted by the learning algorithm. Those multidimensional arrays are known as tensors and, since they have a finite set of points for which values are stored, we can assume that the convolutional function is zero everywhere else. Thus, for a two-dimensional image I with kernel K, convolution is shaped as:

$$\begin{aligned} S(i,j) = (K * I)(i,j) = \\ = \sum_m \sum_n I(m,n)K(i-m,j-n) \\ = \sum_m \sum_n I(i-m,j-n)K(m,n) \end{aligned} \tag{3.7.2}$$

A simplified alternative of Equation 3.7.1 is used on most machine learning and deep learning implementations, known as *cross correlation*:

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i+m,j+n)K(m,n) \tag{3.7.3}$$

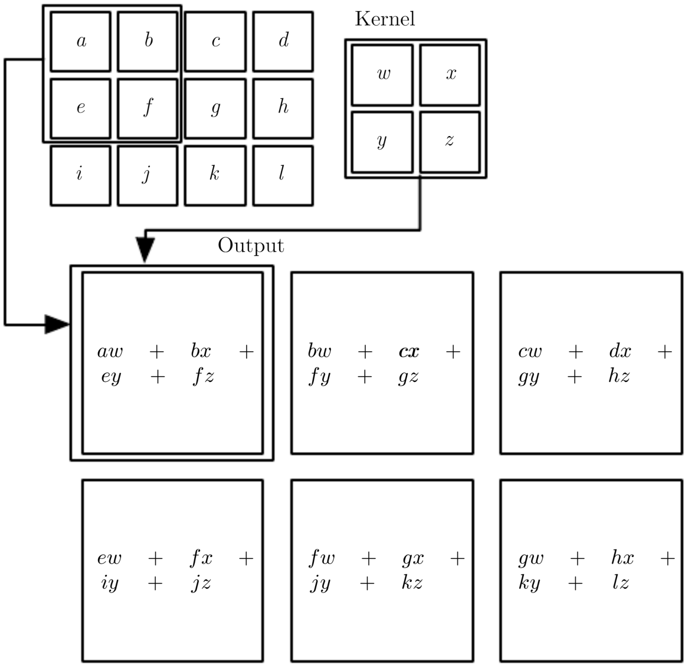


Figure 3.7.2: An example of 2-D convolution without kernel flipping, with receptive field of size $F = (2, 2)$ [10].

The difference between the two lies in whether the commutative property of convolution is applied for flipping the kernel relative to the input or not; in the context of machine learning, the algorithm will learn the appropriate values of the kernel in the appropriate place, so an algorithm based on convolution with kernel flipping will learn a kernel that is flipped relative to the kernel learned by an algorithm without the flipping.

CNN Architecture

Typically, a CNN consists of three main types of layers: a **convolutional layer**, a **pooling layer** and a **fully connected layer**. The layers are stacked to form a full convolutional network architecture (see example in Figure 3.7.1).

Convolutional Layer

The convolutional layer is the core building block of a CNN and most of the computational heavy lifting is done in this layer. The layer's parameters can be perceived as a set of learnable filters, where every filter is relatively small spatially, but extends through the full length of the input's volume. For example, for an RGB image consisting of 3 color channels, the respective filter could be of size $5 \times 5 \times 3$, where 5 pixels are for width and height, and 3 is the depth to match the image's channels. Filters represent the kernel of a convolution. During the forward pass of the data through the CNN, each filter slides, convolving across the width and height of the input volume, and computes dot products between the entries of the filter and the input at all positions. This convolutional process will produce a two-dimensional **activation map**, giving the responses of that filter at every spatial position. The activation maps from all filters of the convolutional layer are stacked together, forming the final output volume to be forwarded to the next layer of the network.

One characteristic of convolutional layers is the *local connectivity* of their neurons. This is due to the impracticality of connecting neurons to all the neurons in the previous volume when dealing with high-dimensional inputs as images. The spatial extend of this connectivity is a hyperparameter called **receptive field** of the neuron and it represents the area of the input visible to the kernel or filter at each operation.An example of a convolution with receptive field $F = (2, 2)$ applied on a 2D input can be seen in Figure 3.7.2.

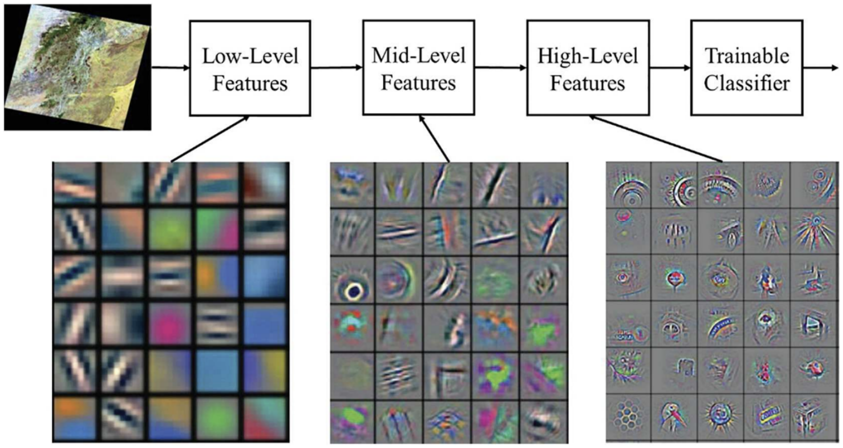


Figure 3.7.3: An example of CNN filters response in an image classification task [11].

The size of the output volume of the convolutional layer is controlled by four main parameters:

1. **Depth** corresponds to the number of filters to be used for the CNN, each one looking for a different feature in the input data, that will be learned during training. The number of filters defines the number of different feature maps yielded in the output, thus the depth of the output data. An example of the filters' response in learning different characteristics can be seen in Figure 3.7.3.
2. **Stride** is the number of pixels, or in general data points, that a filter has move towards one direction, when preparing for the next convolutional operation on the input volume. The size of stride is inversely proportional to the output volume size spatially; a larger stride will give smaller output volumes, while a stride equal to 1 is neutral, meaning the convolution slides at the exact next position.
3. **Zero-padding** refers to padding the input volume with zeros around the border. This gives the flexibility of controlling the spatial size of the output wrt the input volume. A common use of zero-padding is to preserve the size of the input volume, so that the width and height of the output remain the same. Padding can also be applied by repeating the edge values, thus instead of zero-padding, same-padding might be used as well.
4. **Dilation** forms non-contiguous filters, introducing spaces between each cell of the kernel, the size of which is defined by the dilation parameter and was more recently introduced as a CNN feature [YK15]. Denoting the kernel size as K and the dilation as D, then the size of the receptive field of the filter is $F = K \times D$. Also, when dilation is $D = 1$ the kernel cells are sequential, whilst a dilation of $D = 2$ would mean that one sample is skipped between each kernel cell. Dilation can be perceived as searching for features at different scales, with low dilation indicating scanning of local patterns and higher dilation for global ones.

In Figure 3.7.4 an example of using the above parameters is presented, where the weights of the kernel (receptive field of neuron) are shared across all neurons.

Pooling Layer

A pooling layer is commonly used in between successive convolutional layers, to reduce the dimensionality of the feature map, and hence decrease the amount of parameters and computation in the network, also preventing it from overfitting. This type of layer operates on every depth slice of the input separately, and resizes it spatially by applying a downsampling operation, usually the max mathematical operation. In addition to max pooling, the pooling units can also perform other functions, such as average pooling or even L2-norm. Average pooling was often used historically but has recently fallen out of favor compared to the max pooling operation, which has been shown to work better in practice. Pooling layer requires the definition of two hyperparameters, namely the stride of the kernel S and the receptive field F. Usually, pooling layers are not trainable, meaning that the operation they perform is fixed and they have no weights to train.

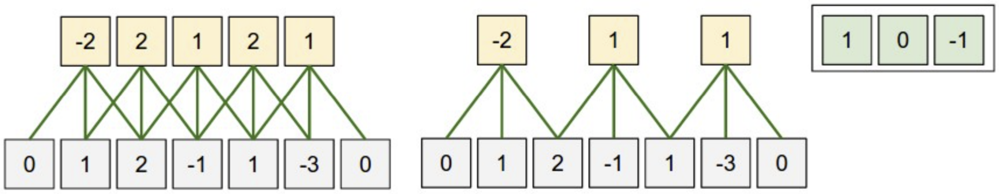


Figure 3.7.4: An example of 1-D convolution with zero-padding of $P = 1$, dilation $D = 1$ and receptive field $F = 3$. The input size is $W = 5$. **Left**: The neuron strided across the input in stride of $S = 1$, giving output of size $(5 - 3 + 2)/1 + 1 = 5$. **Right**: The neuron uses stride of S = 2, giving output of size $(5 - 3 + 2)/2 + 1 = 3$ [12].

Fully-Connected Layer

A Fully-Connected layer (FC), is the final major layer of a CNN. Neurons in an FC layer work in a similar fashion as MLP; they have full connections to all activations in the previous layer, so that each input is connected to each output with a weight. Their activations can hence be computed with a matrix multiplication followed by a bias offset. Fully-connected layers are typically used at the end of the CNN network, succeeding all the other layers, and they perform the final task, either this be e.g. a classification task or some transformation of the processed feature maps in order to discover global patterns in the data.

Motivation behind CNNs

Convolution leverages three important ideas that have motivated the machine learning community to incorporate the method into neural networks: **sparse interaction**, **parameter sharing**, and **equivariant representation**. Moreover, convolution provides a means for working with inputs of variable size, contrarily to a fully-connected network that requires a standard-size input.

First and foremost, CNNs are less computationally demanding and faster to train than their fully-connected counterparts. Due to their need for a smaller kernel than the input size, known as *sparse interactions* or *sparse connectivity*, fewer parameters have to be stored, which both reduces the memory of the model, but also improves its statistical efficiency. Given a network of m inputs and n outputs, the original matrix multiplication requires $m \times n$ parameters, and the runtime of the algorithm is $O(m \times n)$ per iteration. In the case of convolutional operations, however, the connections are limited to k, which might be even several orders of magnitude smaller than m, thus the parameters are reduced to $k \times n$ and the runtime becomes $O(k \times n)$.

Parameter sharing refers to the use of the same parameters for more that one function in the model. This is an inherent property of convolutional neural networks, as by applying the convolutional operation, the kernel passes through all the elements of the input. Hence, rather than learning a separate set of parameters for every location, the same set is applied to all.

The property of parameter sharing in CNNs induces one more fundamental property of the convolutional layers, named *equivariance of translation*. An equivariant function changes its output in the same way as the input changes; specifically, a function $f(x)$ is considered equivariant to a function $g(x)$ if $f(g(x)) = g(f(x))$. In the context of convolutions, denoting the convolutional operation as function f, this is equivariant to any function that translates the input, e.g. shifts it, which can be denoted by g.

The importance of this property can be made more clear by the following examples. In time-series data, when shifting an event later in time in the input sequence, the convolutional operation will be able to detect it and the exact same representation of it will appear in the output, just in a later time slot. This allows the identification of when different features appear in the input, hence the detection of their timeline. Also, in the spatial domain of images, convolution creates a 2-D map of where certain features appear in the input. When processing an image, it is typical to detect edges in the first layer of a CNN. Similar edges might appear in several areas of the image to be detected, and here arises the need of parameter sharing across the entire image, which is nicely handled by the equivariance property of convolutions.

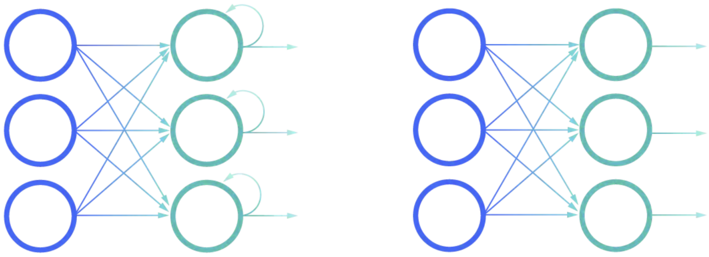


Figure 3.7.5: Recurrent neural network (left) vs. Feed-forward neural network (right) [13].

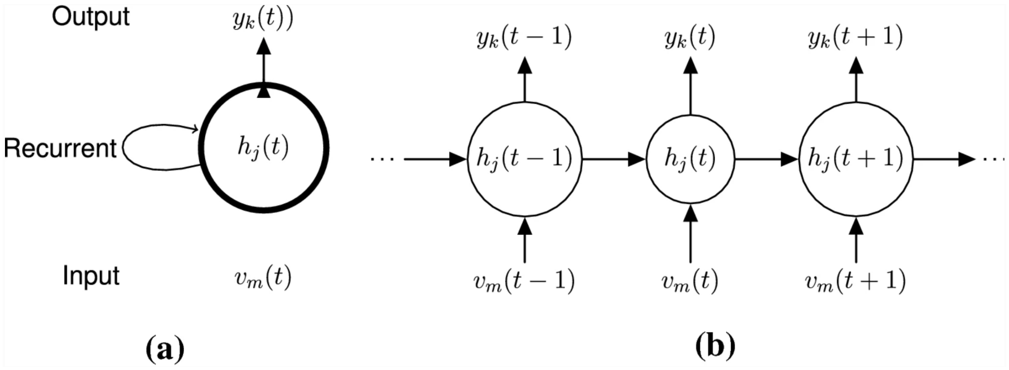


Figure 3.7.6: Unfolded RNN architecture, where the network has completed $t+1$ iterations, for $t \in [0, T]$ [14].

The neural networks that have been presented until now lie under the umbrella of feed-forward architectures, where a fixed amount of input data is given to the network to be processed simultaneously, giving the output data all together. This kind of networks is great for handling static data such as images; however in sequential data the sequence-parameter and its dependencies are not taken into account as the data points are processed each one separately, loosing possibly viable time-related or ordinal-related information. A solution to this problem is proposed by the networks that will be discussed next, namely *recurrent neural networks*.

3.7.2 Recurrent Neural Networks

Recurrent neural networks [RHW86] is a type of NN that applies on sequential or time-series data, taking into consideration the nature of their structure, and they are commonly used in temporal or ordinal problems. Specifically, they are distinguishable by their concept of "memory", where information from prior inputs is saved to be used on the current input and influence the respective output. To transition from multilayer networks to recurrent networks, the idea of *sharing parameters across different parts of the model* has to be applied. Parameter sharing makes it possible to extend and apply the model to examples of different lengths and generalize across them.

Basic RNN Structure

A simple RNN has a feedback loop as shown in Figure 3.7.5, which gives the feedback of T time steps, the number of which is defined by the architectural design, and can be unfolded an equal amount of times to the number of time steps. For simplicity, the RNNs will be referred to as operating on a sequence of vectors $\mathbf{x}^t$, where the time step index t will be ranging from $[1, T]$. The time step index does not have to literally express the passage of time in the real world, it could be just the position in the sequence. An unfolded middle stage representation of an RNN can be seen in Figure 3.7.6. This process will be repeated for T time steps, starting from $x^{<0>}$ until $x^{<T>}$, where the network gives the final output y^T. For each time step t, the activation a_t and the output y_t are expressed by the following equations:

$$h_t = a^{<t>} = g_1(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a)$$
$$y^{<t>} = g_2(W_{ya}a^{<t>} + b_y) \tag{3.7.4}$$

where $W_{ax}, W_{aa}, W_{ya}, b_a, b_y$ are the RNNs coefficients which are temporarily shared, and g_1, g_2 are the activation functions. In RNNs specifically, the activations of the middle representations of the sequence are better known as *hidden states* of the network, h_t.

Types of RNNs

Depending on how the input and the output data are related and fed to the recurrent neural network, there are four different types of RNNs:

1. **One to One** - else known as *Vanilla RNN*, is used for general ML problems and has one input and one output, as a single (x_t, y_t) pair.
2. **One to Many** - in this type of networks, a single input at x_t can produce multiple outputs e.g. (y_{t0}, y_{t1}, y_{t2}).
3. **Many to One** - those networks take many inputs from different time steps, to produce a single output. For example, (x_t, x_{t+1}, x_{t+2}) can produce the output y_t.
4. **Many to Many** - they take a sequence of inputs and generate a sequence of outputs. The input and the output can have the same or different lengths.

As it can be seen in Figure 3.7.7, the type of RNN to be chosen greatly depends on what kind of data are to be handled and the problem that has to be solved. In our work, data is in the form of time sequences, were many inputs are given and a single prediction has to be made for each sequence, thus the Many to One architecture will be used. But this will be further discussed in Chapter 4.

Bidirectional RNNs

The basic structure of RNNs that has been discussed until now is "causal", meaning that it takes into consideration only states of the past $h^{(1)}, ..., h^{(t-1)}$ and the current input $\mathbf{x}^{(t)}$ in order to make a prediction $y^{(t)}$ for the current time step. Also, some of the models allow information from past y values to affect the current state, when they are available. However, there are applications were future time steps need to be utilized in order to improve the model's accuracy; thus the whole input sequence is used to output a prediction of $y^{(t)}$. For example, in speech recognition, the correct interpretation of the current sound as a phoneme may depend not only on the previously predicted phonemes, but also on the next ones, due to co-articulation. Also, since linguistic dependencies between nearby words may exist, if there are multiple interpretations of the current word that are acoustically plausible, it is needed to look both into the preceding and following words to disambiguate them.

Bidirectional RNNs achieve the access to both past and future time steps for each prediction, by stacking an RNN that moves forward through time, beginning from the start of the sequence, with a second RNN that moves backwards through time, beginning from the last time step of the sequence. This presupposes that the prediction of $y^{(t)}$ is made after the whole sequence is passed through the network, as it is depicted in Figure 3.7.8.

Long Term Dependencies

There are cases in sequential problems, where the relevant information lie far away in the sequence, thus a larger network is needed to capture their relation. However, as that gap grows, RNNs become unable to efficiently learn to connect the information. It is difficult to capture long-term dependencies, since the multiplicative gradient during the model's training can exponentially decrease or increase with respect to the number of layers. Hence, the *vanishing* and *exploding gradient* phenomena are often encountered in the context of RNNs.

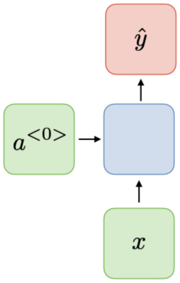
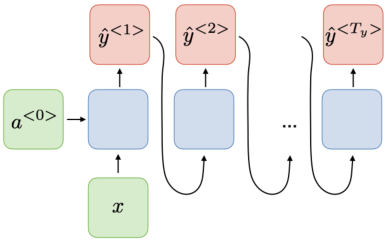
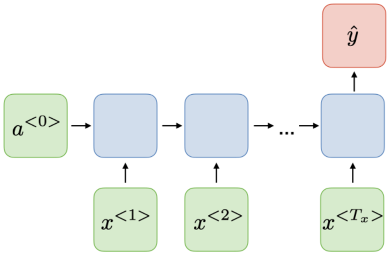
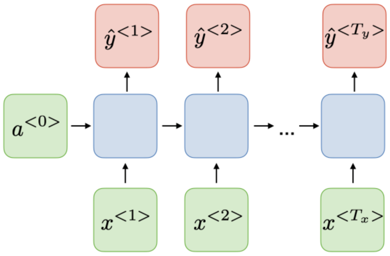
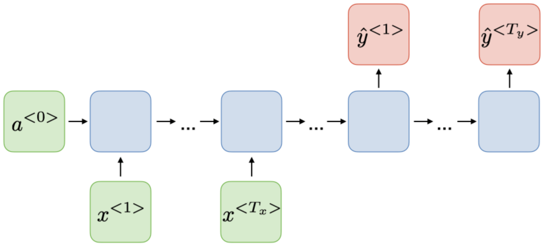
RNN Type	**Illustration**	**Example for application**
One to One, $T_x = T_y = 1$		Traditional neural network
One to Many, $T_x = 1, T_y > 1$	 $\hat{y}^{<1>}$ $\hat{y}^{<2>}$ $\hat{y}^{<T_y>}$ $a^{<0>}$ ... x	Music generation
Many to One, $T_x > 1, T_y = 1$	 $\hat{y}$ $a^{<0>}$ ... $x^{<1>}$ $x^{<2>}$ $x^{<T_x>}$	Sentiment classification
Many to Many, $T_x = T_y$	 $\hat{y}^{<1>}$ $\hat{y}^{<2>}$ $\hat{y}^{<T_y>}$ $a^{<0>}$ ... $x^{<1>}$ $x^{<2>}$ $x^{<T_x>}$	Name entity recognition
Many to Many, $T_x \neq T_y$	 $\hat{y}^{<1>}$ $\hat{y}^{<T_y>}$ $a^{<0>}$ ... ... $x^{<1>}$ $x^{<T_x>}$	Machine translation

Figure 3.7.7: The different RNN categories, depending on the relation between the length of input and output sequences, T_x, T_y [15].
The choice of the correct architecture is defined by the nature of the problem, for example: (a) Traditional neural network, (b) Music generation, (c) Sentiment classification, (d) Name entity recognition, (e) Machine translation

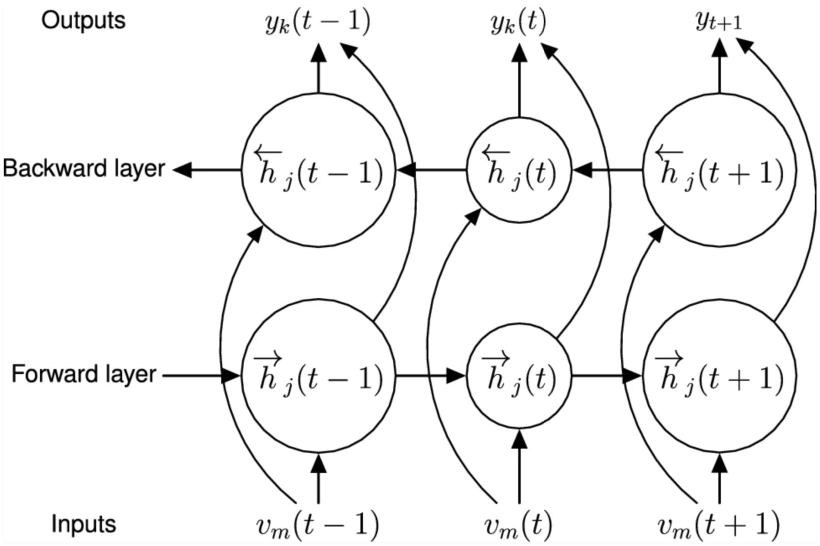

Outputs
$y_k(t-1)$
$y_k(t)$
y_{t+1}
Backward layer
$\overleftarrow{h}_j(t-1)$
$\overleftarrow{h}_j(t)$
$\overleftarrow{h}_j(t+1)$
Forward layer
$\overrightarrow{h}_j(t-1)$
$\overrightarrow{h}_j(t)$
$\overrightarrow{h}_j(t+1)$
Inputs
$v_m(t-1)$
$v_m(t)$
$v_m(t+1)$

Figure 3.7.8: Bidirectional RNN architecture [14].

As an example of long term dependencies, consider predicting the last word in the text "I grew up in Greece, [...] I speak fluent *Greek*.". Recent information suggests that the next word is probably the name of a language, but in order to narrow down which one it is, the context of Greece is needed, which appeared further back. It's entirely possible for the gap between the relevant information ("Greece" in the current example) and the point it is needed in the sequence ("I *speak* fluent ..."), to become very large.

In order to remedy the vanishing gradient problem, specific gates are used in some types of RNNs and usually have a well-defined purpose. They are normally noted as Γ and are equal to:

$$\Gamma = \sigma(Wx^{<t>} + Ua^{<t-1>} + b) \tag{3.7.5}$$

where W, U, b are coefficients specific to the gate and σ is the sigmoid function.

Two of the most well-known gated RNN architectures will be discussed next.

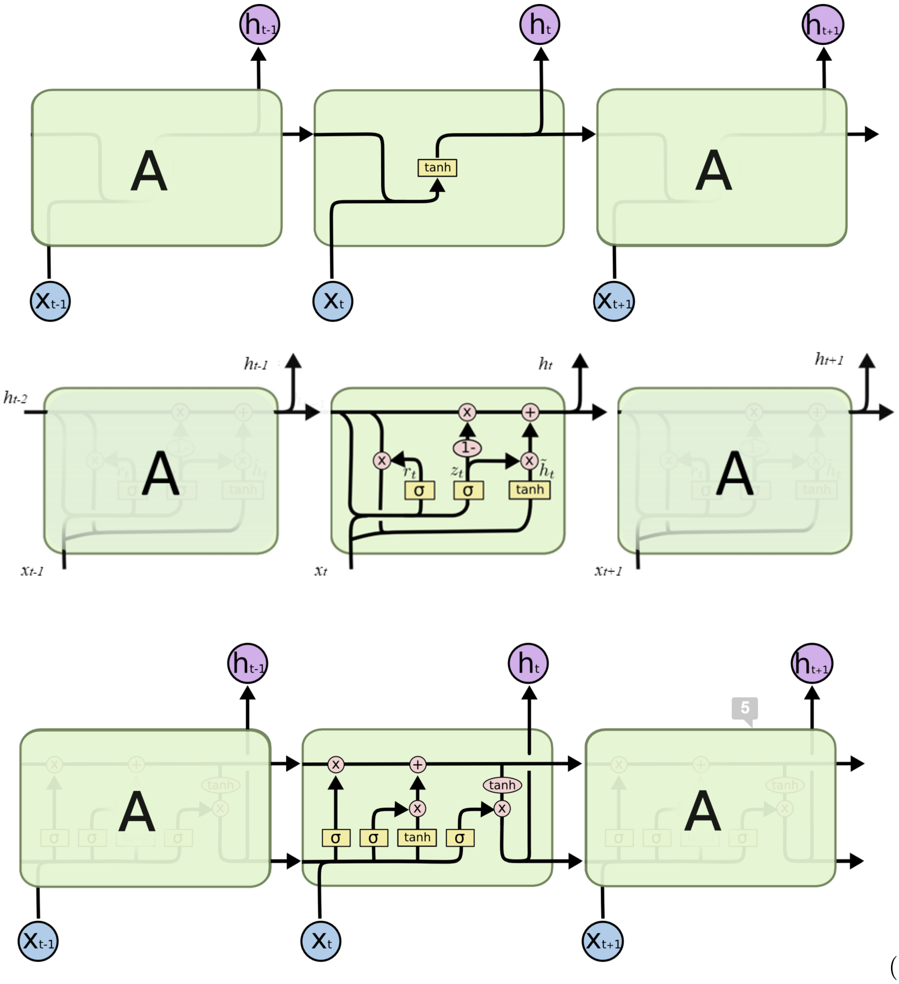
Gated Recurrent Unit - GRU

This RNN variant has two gates, a *reset* gate Γ_r and an *update* one Γ_u, which control how much and which information of the sequence to retain for future predictions [Chu+14]. Basically, these are two vectors, which decide what information should be passed to the output. They are trained together with the rest of the network, learning to keep useful information from long ago, without washing it through time, and remove information, which is irrelevant to the prediction.

The equations for time step t of the GRU module are formed as follows:

$$\begin{aligned} z_t &= \sigma(W_z x_t + U_z h_{t-1}) \\ r_t &= \sigma(W_r x_t + U_r h_{t-1}) \\ \tilde{h}_t &= tanh(W x_t + r_t \odot U h_{t-1}) \\ h_t &= z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t \end{aligned} \tag{3.7.6}$$

- The **update gate** z_t helps the model to determine how much of the past information needs to be passed along to the future. This is a powerful method of the model to decide whether to copy all the information of the previous time steps and eliminate the risk of the vanishing gradient problem.
- The **reset gate** r_t allows the model to decide how much of the past information to forget.



(a) RNN

(b) GRU

(b) LSTM

Figure 3.7.9: Different RNN variations overview: RNN, GRU and LSTM [16].

- The **current memory content** $\tilde{h}_t$ uses the reset gate to store the relevant information from the past. The element-wise multiplication between the reset gate and the previous memory product Uh_{t-1} will determine what to remove from the previous time steps.
- The **final memory** at the current time step h_t incorporates the update gate to determine what to collect from the current memory content $\tilde{h}_t$ and what from the previous time steps h_{t-1}.

The application of update and reset gates in GRUs eliminates the vanishing gradient problem since the model is not *washing out* the new input every single time (meaning that the previous input information is deleted every new iteration); instead, it keeps the relevant information and passes it down to the next time steps in the network. GRUs, if carefully trained, can perform extremely well, even in complex scenarios.

Long-Short Term Memory Network - LSTM

LSTM [HS97] is another variation of RNNs, predecessor of GRUs, and has been widely used for efficiently coping with long-term dependencies limitations of RNNs until today. At each time step, the LSTM cell takes as input three different pieces of information – the *current input data* x_t, the *short-term memory from the previous cell* h_t (similar to hidden states in RNNs) and lastly *the long-term memory* c_t. The short-term memory is commonly referred to as the **hidden state**, and the long-term memory is usually known as the **cell state**. There are three gates in LSTMs compared to the two gates GRUs have. Their purpose is also to regulate the information to be kept or discarded at each time step before passing on the long-term and short-term information to the next cell, and are known as the *input gate*, the *forget gate*, and the *output gate*.

The key to LSTMs is the cell state, which, as shown in Figure 3.7.10, runs straight through the entire chain, with only some minor, linear interactions. Information can be easily added or removed from it, regulated by the LSTM's gates.

The equations that define an LSTM are the following:

$$\begin{aligned} f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\ i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ \tilde{C}_t &= tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \\ C_t &= f_t * C_{t-1} + i_t * \tilde{C}_t \\ o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\ h_t &= o_t * tanh(C_t) \end{aligned} \tag{3.7.7}$$

where the weight matrix can be considered as $W_j = [W_j|U_j]$ for $j = f, i, C$ for consistency with the equations of GRU 3.7.2 and b_j is the bias.

- The first step of an LSTM is to apply the *forget layer* to the input cell state, as shown in Figure 3.7.11.
- The next step is to decide what new information is going to be stored in the cell state and it consists of two parts. First, the *input gate layer*, which is a sigmoid layer, decides which values to update. Then, a *tanh* layer creates a vector of new candidate values $\tilde{C}_t$, that could be added to the state, as it is shown in Figure 3.7.12.
- In the next step, these two are combined to create an update to the state, as it can be seen in Figure 3.7.13. To do so, the old state is multiplied by f_t, forgetting the things which were decided to forget earlier. Then, $i_t * \tilde{C}_t$ is added. This is the new candidate values, scaled by how much it was decided to update each state value.
- The output of the LSTM is a filtered version of the cell state. First, a sigmoid layer is applied, deciding what parts of the cell state are going to output. Then, the cell state go through *tanh* and are multiplied by the output of the sigmoid gate, so that only the parts that were decided form the output. This final step can be seen in Figure 3.7.14.

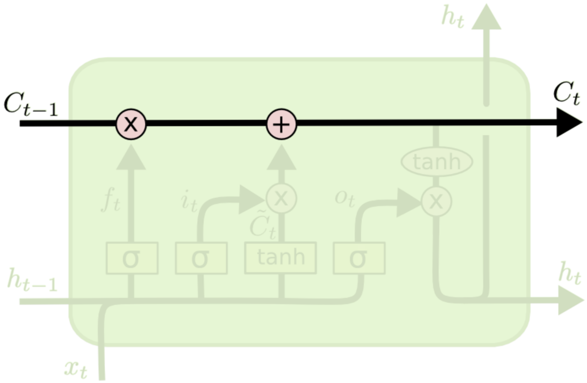


Figure 3.7.10: Cell state of an LSTM. It's very easy for information to just flow along it unchanged [16].

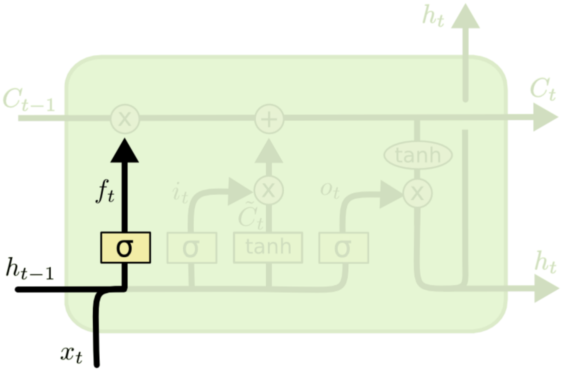


Figure 3.7.11: The Forget Gate Layer of an LSTM. In this layer, the decision of whether to keep or throw away from the cell state some information is made [16].

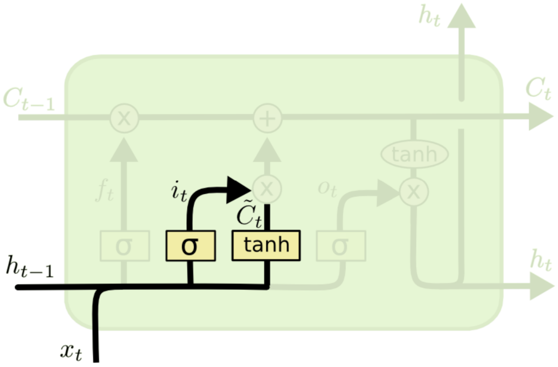


Figure 3.7.12: The Input Gate Layer of an LSTM [16].

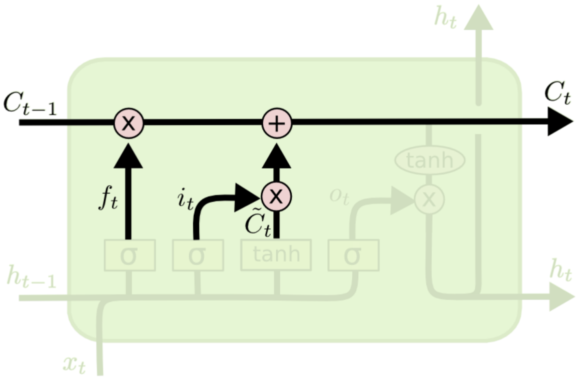


Figure 3.7.13: Updating the cell state of an LSTM [16].

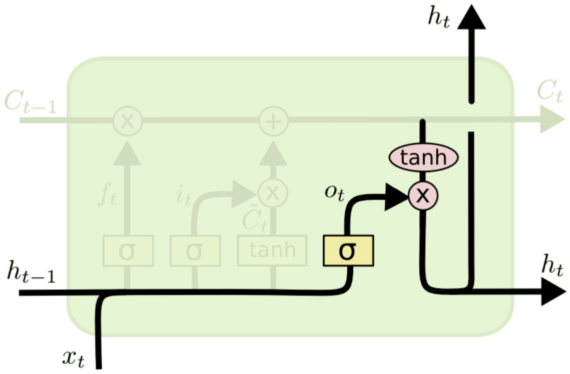


Figure 3.7.14: Output Layer of an LSTM [16].

Chapter 4

Experiments

4.1 Datasets

In our work the problem of sleep stage classification via wearable devices is examined, testing some of the previously presented methods in the literature review, as well as some new advancements on them. The data used for the experiments that follow are the two datasets employed in [Wal+19].

4.1.1 Walch dataset

The first one is an internal dataset created by the team of Walch et al. [Wal19], and can be acquired from here: https://physionet.org/content/sleep-accel/1.0.0/. The dataset consists of 31 subjects during an 8-hour night's sleep. Initially 39 subjects were recruited, but eight of them were excluded due to errors with data transmission, sleep apnea and REM sleep behavior disorder conditions. The sleep is monitored with PSG, giving labels of the sleep stage every 30-second epochs. The subjects were given an Apple Watch to wear during the night's sleep at the lab, which recorded **acceleration** (in g) and **heart rate** (in beats per minute, bpm). Specifically, the measurements consist of:

- **Acceleration** on the 3D axis. It has values in the (x,y,z) direction, and it is measured in units of $g(9.8m/s)$. Its sampling rate ranges around $50Hz$.

  By investigating the dataset, it was observed that the sampling rate greatly differed among subjects, thus some kind of interpolation should be applied to ensure a uniform sampling rate amongst all subjects.

  Also, short windows of time with missing data occasionally occurred, likely due to server-side issues during the real-time sleep night data collection.

- **Heart rate**, for which the sampling rate is every several seconds (beats per minute) and is 1-dimensional feature.

  In order to have a uniform sampling rate, the best approach is to resample every 1-second, so that it is fixed to 1Hz.

  There are missing values in some time segments during the night's sleep as well.

- **PSG** data during a night's sleep: one sleep state is annotated with a value every 30-second segments, equivalent to 1 epoch.

  They are used as labels for the data and they have no missing values.

  However, they contain -1 values which indicate unlabeled data and need to be handled carefully during the data processing.

The subject's ambulatory steps are also collected with the Apple Watch. The PSG labels consist of: **Wake, N1, N2, N3, N4, REM** as directed by the AASM method. In total, the following types of data are provided in the Walch Dataset:

- **motion (acceleration)**: Recorded from the Apple Watch and saved as .txt text files with the naming convention '[subject-id-number]_acceleration.txt'

  Each line in this file has the format: *date (in seconds since PSG start), x acceleration (in g), y acceleration, z acceleration*

- **heart rate (bpm)**: Recorded from the Apple Watch and saved as .txt files with the naming convention '[subject-id-number]_heartrate.txt'

  Each line in this file has the format: *date (in seconds since PSG start), heart rate (bpm)*

- **steps (count)**: Recorded from the Apple Watch and saved in the format '[subject-id-number]_steps.txt'

  Each line in this file has the format: *date (in seconds since PSG start), steps (total in bin from this timestamp to next timestamp)*

- **labeled sleep**: Recorded from polysomnography and saved in the format '[subject-id-number]_labeled_sleep.txt'

  Each line in this file has the format: *date (in seconds since PSG start) stage (0-5, wake = 0, N1 = 1, N2 = 2, N3 = 3, N4 = 4, REM = 5)*

4.1.2 MESA dataset

The second dataset used in our work is the Multi-Ethnic Study of Atherosclerosis (MESA) [Nat16; Zha+18a; Che+15] and can be found in here https://sleepdata.org/datasets/mesa. MESA is a multicenter longitudinal investigation of factors associated with the development of subclinical cardiovascular disease and its progression from subclinical to clinical cardiovascular disease, hence from a state where the patient has no signs and symptoms that are detectable by physical examination or laboratory test to a state where the disease can be clinically manifested. It consists of 6814 black, white, Hispanic and Chinese-American men and women initially aged between 45-84 starting at baseline in 2000-2002. Between 2010-2012, 2,237 participants also were enrolled in a Sleep Exam (MESA Sleep), which included **full overnight unattended polysomnography**, **7-day wrist-worn actigraphy**, and a **sleep questionnaire**. The objectives of the sleep study are to understand how variations in sleep and sleep disorders vary across gender and ethnic groups and relate to measures of subclinical atherosclerosis.

We use the data in the same manner as in [Wal+19], for testing purposes of how well can the pre-trained models generalize on unseen datasets. In this sense, the first 188 subjects of MESA dataset with co-recorded actigraphy and PSG data are extracted and processed for use as an independent testing set. As is done in the referring work, there is a direct correspondence between the motion and local standard deviation of heart rate features to activity counts from actigraphy and heart rate during PSG, respectively. The heart rate in the MESA dataset is derived from pulse oximetry (PPG), enhancing the comparability of the Apple Watch dataset used for training and the MESA dataset used for testing purposes.

Specifically, the dataset is formed as follows:

- **Heart rate**: it is collected via the polysomnography records, as described below.

  It is derived through the corresponding .EDF files, with sampling rate of 1Hz.

- **Actigraphy**: 2,237 participants were recruited to wear wrist-worn actigraphy devices (Actiwatch Spectrum, Philips Respironics) between 2010 and 2013. Participants were instructed to wear the watch for a week. Records were scored by a trained technician at the Boston Sleep Reading Center.

  Epoch-by-epoch (EBE) data files (CSV) have been created for 2,159 participants with actigraphy data. Each row in these files represents 30 seconds worth of summary data from the actigraphy device.

- **PSG**: Polysomnography records, corresponding to five sleep stages (**N1, N2, N3, N4, REM, Wake**).

  In-home polysomnography (PSG) was conducted using the Compumedics Somte System (Compumedics Ltd., Abbotsford, Australia). The sensors and recording montage consisted of cortical electroencephalograms (central C4-M1, occipital Oz-Cz, and frontal Fz-Cz leads), bilateral electrooculograms, chin EMG, thoracic and abdominal respiratory inductance plethysmography (by auto-calibrating inductance bands); airflow (by nasal-oral thermocouple and pressure recording from a nasal cannula); ECG; leg movements, and finger pulse oximetry. In addition to connection of sensors and electrodes, trained staff members completed signal calibrations and checked impedance. Nocturnal recordings were transmitted to the centralized reading center at Brigham and Women's Hospital and data were scored by trained technicians using current guidelines.

  Raw polysomnography data are available for 2,056 MESA Sleep participants. Each recording has a signal file (.EDF) and two versions of the event scoring and epoch staging annotations (.XML).

In Figure 4.1.1 are depicted the data distributions for both Walch and MESA datasets, for all the sleep stages tested in this work. Their initial labels distributions are also presented, which consist of six sleep stages, including stage N4. In Figure 4.1.2 and in Figure 4.1.2 can be seen the histograms of each feature separately, for every sleep stage, for the Walch and MESA dataset respectively.

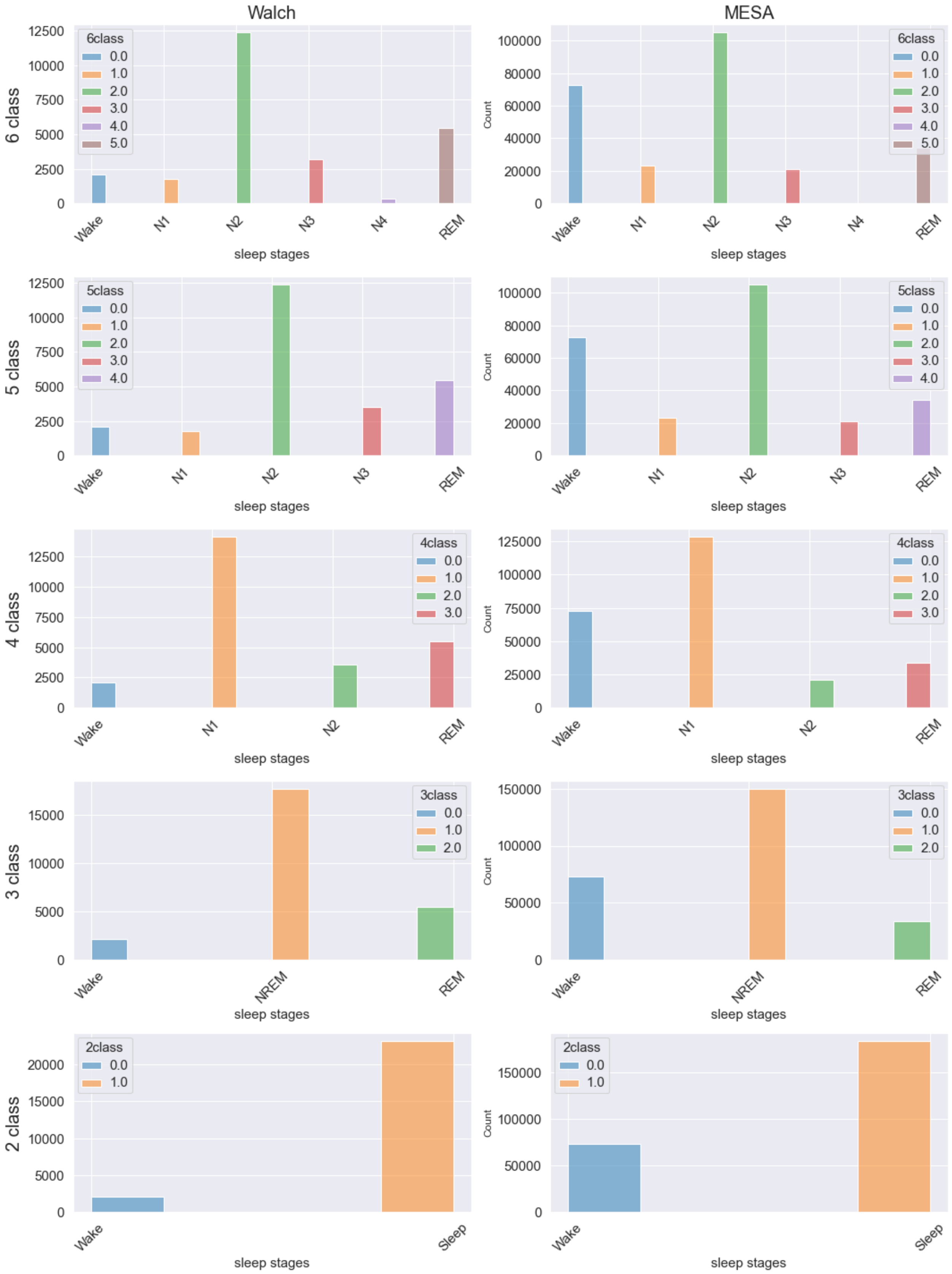


Figure 4.1.1: The histograms of class distributions both for MESA and Walch datasets, for all the sleep stage classes. The first six classes are the ones provided by the original datasets, while the rest of the classification schemes are studied in this work.

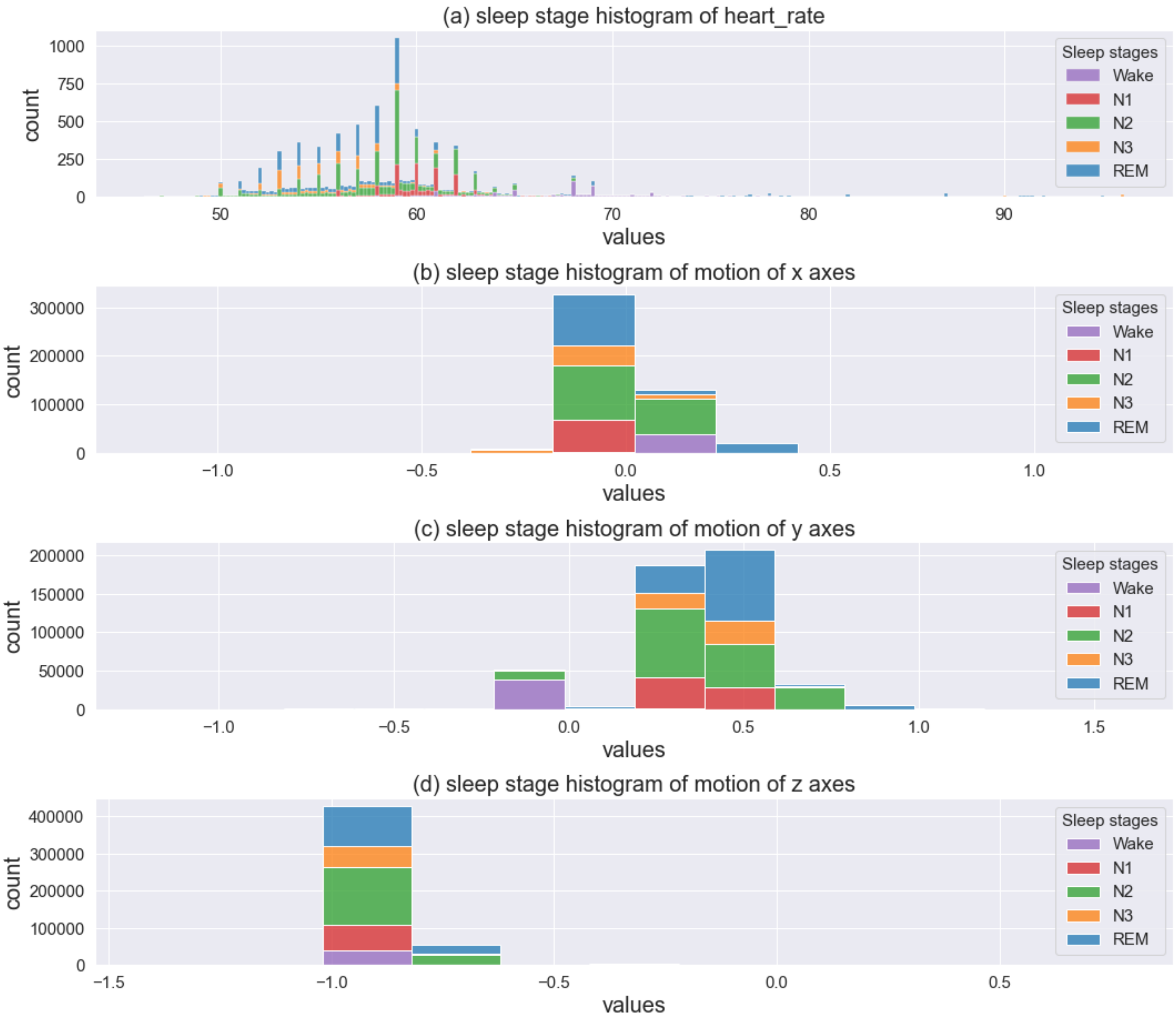


Figure 4.1.2: Histograms of the raw features from the Walch dataset, grouped by the sleep stage they belong to. (a) heart rate, (b) wrist motion of x-axes, (c) wrist motion of x-axes, (d) wrist motion of z-axes

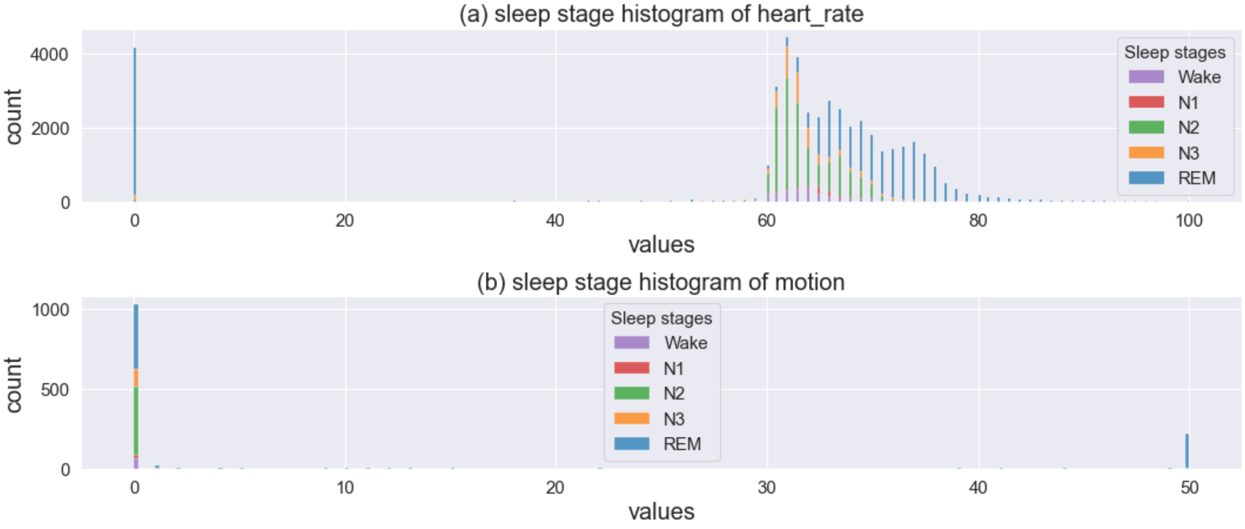


Figure 4.1.3: Histograms of the raw features from the Walch dataset, grouped by the sleep stage they belong to. (a) heart rate, (b) wrist motion

4.2 Simple Bidirectional LSTM

For the first of the proposed experiments, a simple bidirectional-LSTM architecture was tested on the Walch-designed features, as described in [Wal+19]. The source code for the feature extraction can be found in here: https://github.com/ojwalch/sleep_classifiers. In the aforementioned paper, the classification methods used lie in the area of ML algorithms, and a simple MLP deep learning approach, which do not take into consideration the temporal relations of the data. Hence, here an RNN architecture is applied on the features extracted by Walch et al. to take the time parameter into account and examine how such a model performs having those features as input. Bidirectional-LSTMs have been widely used in other works tackling sleep stage classification with wearable derived data, thus it seemed to be the best-suited option for the task. The classification task is tested in all of the 2-5 sleep stages.

4.2.1 Data Preparation

Data Preprocessing

For the data preparation, the same feature extraction method is followed as in Walch et al., as it is proposed in their given code. However, some preprocessing steps are taken first, to ensure the correctness of time alignment between the epochs:

- To begin with, there are missing epochs in Walch‘s data, which were not handled in their approach, since they did not use continuous sequential data, but separate timestamps as their data points.
  In order to get **continuous time segments**, in our work the raw data are split at the points where epochs were missing from at least one feature, so that the new segments have continuous epochs with valid values in all features (psg, heart rate, acceleration).
- Also, the -1 values needed to be removed from the PSG labels, as they indicate unsorted data. To do this, 3 cases are considered:
  1. If the values are at the beginning or the end of the sequence they are just removed, by cutting them off the sequence.
  2. If there is a single value in the middle of the sequence, it will be replaced by the round of the mean value of the two neighboring ones.
  3. Finally, in the case of a sequence of -1 values being in the middle of the data (meaning that they are not boundary values), they have to be removed and the data is split in their place; but this case is not practically encountered in the dataset.
- Then, the preprocessed raw data (psg, motion, acceleration), segmented into continuous time sequences, are given as input to the Walch code for feature extraction.
- Finally, the extracted features are given as input to the proposed bidirectional-LSTM for training.

Feature Extraction

The final features consist of:

- **Activity counts**, extracted from the motion data; it is converted from raw acceleration in m/s^2 using the method outlined in [TV13] and implemented in Walch source code. It is chosen in order to allow compatibility with measurements from other tools after they are also being converted into activity counts. For the preparation of features, the data are cropped to a local window of 10 minutes around the scored epoch. The final activity count feature is derived by convolving the window with a Gaussian ($\sigma = 50$ seconds).
- **Cosine transform**, which implements a simplified version of the circadian clock of the individual. A fixed cosine wave is used, appropriately scaled and shifted relative to the time of recording start, which rises and falls over the course of the night, so that it matches the physiological behavior of the circadian clock during a night of sleep.

- **HR feature**, which is derived after the application of some preprocessing steps on the heart rate measurements. The signal is firstly interpolated in order to have sampling rate of 1Hz exactly. Then it is smoothed and filtered by convolving with a difference of Gaussian filter ($\sigma_1 = 120$ seconds, $\sigma_2 = 600$ seconds). Normalization is applied on each individual by dividing with the 90^{th} percentile of the absolute difference between each HR data point and the mean value of heart rate over the whole sleep period. Finally, the standard deviation in the window around the scored epoch is extracted and the resulting values are used as the representative feature for heart rate.
- **Time feature**, which refers to the time since the recording offset.
- **PSG labels**

In Figure 4.2.1 are depicted the raw heart rate and 3D wrist acceleration features for subject 46343_0, after applying the preprocessing method described in Section 4.2.1.In Figure 4.2.2 are depicted the corresponding extracted features, as described above. The time feature is used for the horizontal axes of all the other subject figures, given that it expresses a linear time sequence, since the beginning of the recording. In Figure 4.2.3 are presented the histograms per extracted feature for the Walch dataset, grouped by their corresponding sleep labels. For the activity count feature, there are outliers, which are purposely collected together in the histogram for better visibility. Figure 4.2.4 shows the same extracted features for the MESA dataset.

4.2.2 Architecture

A bidirectional-LSTM architecture is implemented using **Pytorch** [Pas+19]. The model consists of:

- an **LSTM layer** with the bidirectional parameter set to true.
- a **Linear Layer**, which receives the output of the biLSTM layer and returns a vector of size equal to the number of sleep stages, which represent the probability of each sleep stage to be the correct one.
- **Cross-entropy loss** is used as the model criterion for back-propagation during training: **softmax** and **logarithmic transformation** are applied to the output of the linear layer in order to get the prediction of the class, by returning a one-hot vector with the sleep stage with the highest probability.

Batch size

The parameters determining some of the most prominent NN architectures, which are also incorporated in our work, are discussed in Chapter 3. However, some more technical details need to be explained before presenting the models designed for the purpose of the current work's experiments. Neural networks are quite time demanding due to the heavy computations that need to be performed during the training process, so parallelizing the computations has been a major focus point of the NN technological and hardware advancements. Since the data used in neural networks are interpreted as multidimensional arrays, the graphic processing unit of a computer is utilized for their handling. GPUs are originally designed for the graphic interface of the computer, to optimally process images, which have the same representation of arrays, making GPUs a perfect candidate for the heavy computations of neural networks. The standard way of passing the data through the network for training, is by giving the data points one at a time to the CPU, then applying back-propagation to calculate the corresponding loss and improve the network's weights at each iteration. However this method is quite time-consuming, hence a more advanced technique has been developed, in order to take full advantage of the computational power of the GPU. The data are split into batches of a predefined size, and one batch is given as input at each iteration during the training process. The total training loss is computed for all the samples in the batch and it is back-propagated for correcting the network weights. Once all the batches of the training data are passed through the network and back-propagated accordingly, this is called a *training epoch*. The total training of the network consists of a specified number of training epochs. In this manner, the network "sees" the data points of the whole batch before improving its weights. The choice of the batch size needs to be handled carefully though, as, in case the dataset is relatively small, a larger batch size will not allow the model to properly train. This is due to the fact that the model's weights are corrected with the aggregated loss of all samples in each batch; hence, if a dataset is relatively small and it has a large batch size, the model will be corrected just a few times in each training epoch, not capturing all the details of the dataset and not being able to converge in an optimal state. Also, it has been observed that larger batch sizes incline to generalizing worse at test data, as the model tends to find sharp minima

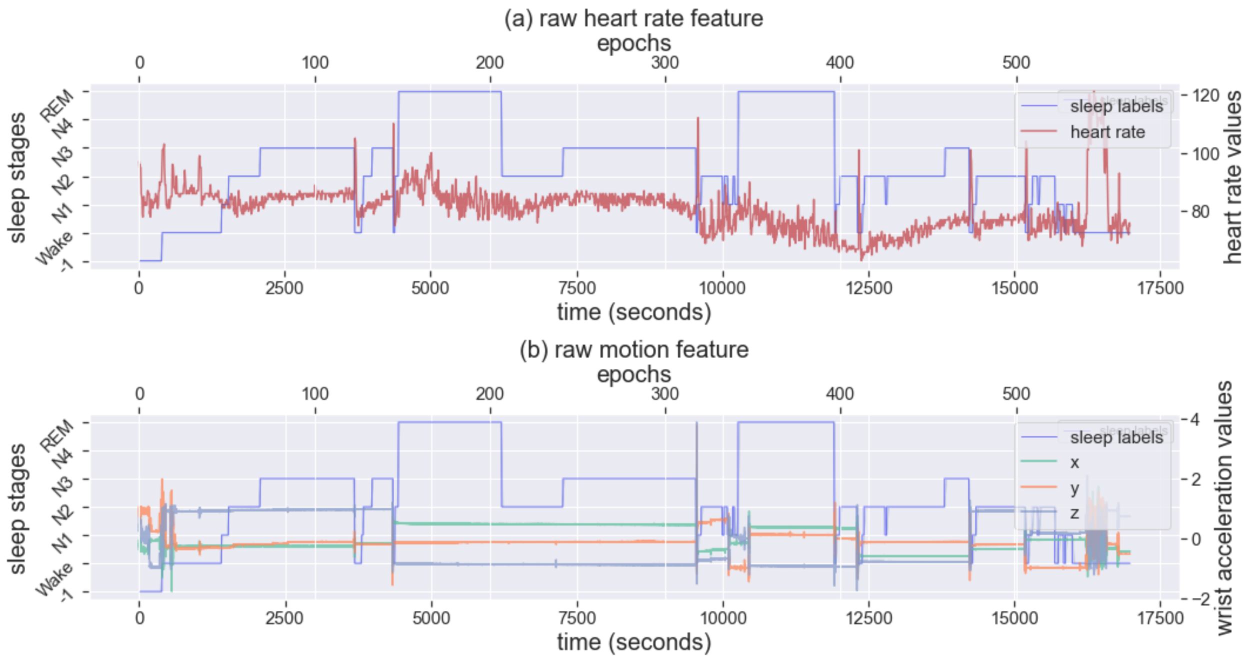


Figure 4.2.1: The raw features of subject 46343_0 from the Walch dataset, with their corresponding sleep labels. (a) raw heart rate and (b) wrist acceleration

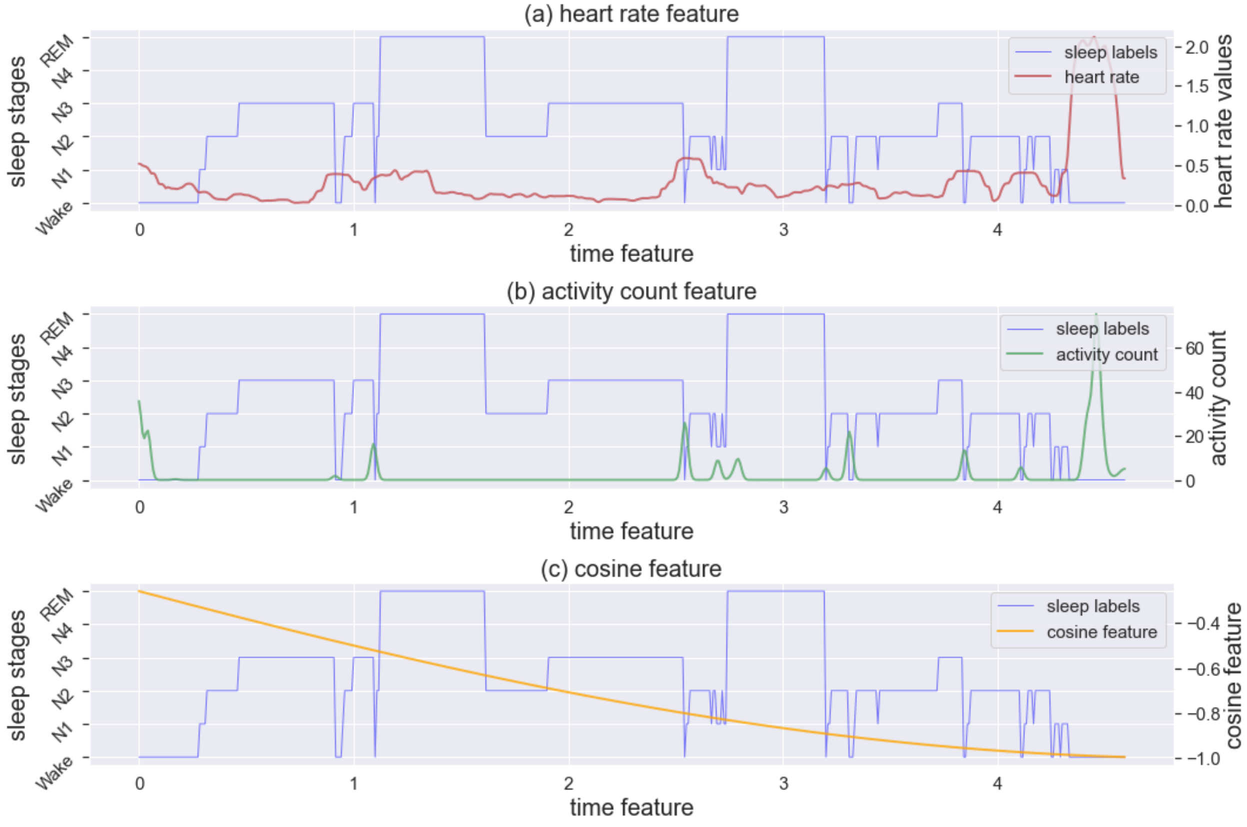


Figure 4.2.2: The extracted features with their corresponding sleep labels, for the Walch subject 46343_0. The x-axes is the time feature, which is counting since the start of the recordings. (a) heart rate feature, (b) activity count feature, (c) cosine feature

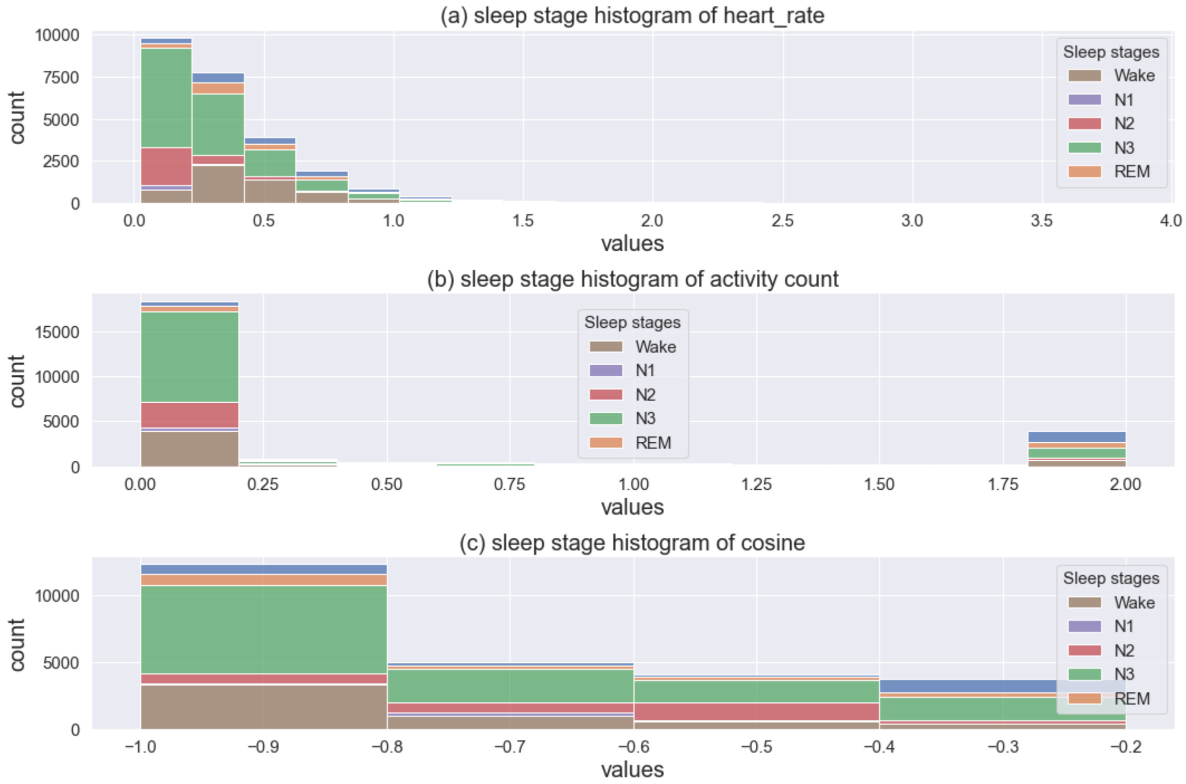


Figure 4.2.3: Histograms of the extracted features of the Walch dataset, grouped by their corresponding sleep labels. (a) heart rate, (b) activity count, (c) cosine

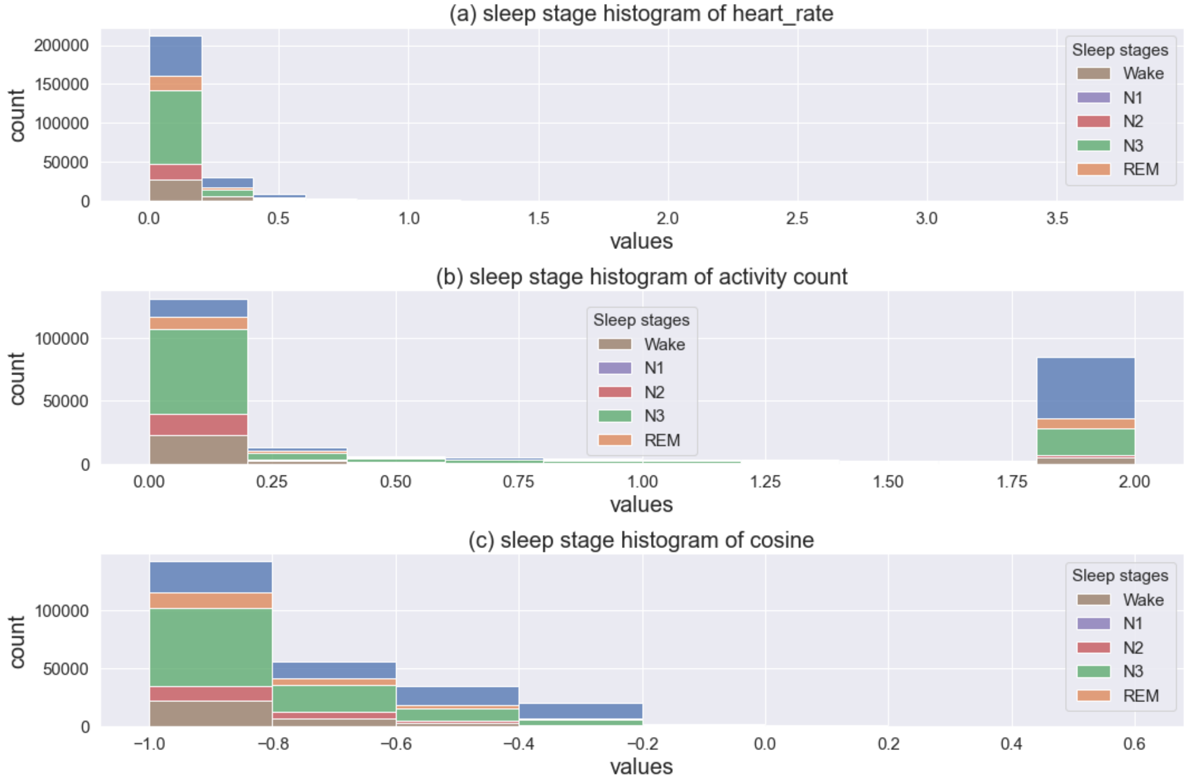


Figure 4.2.4: Histograms of the extracted features of the MESA dataset, grouped by their corresponding sleep labels. (a) heart rate, (b) activity count, (c) cosine

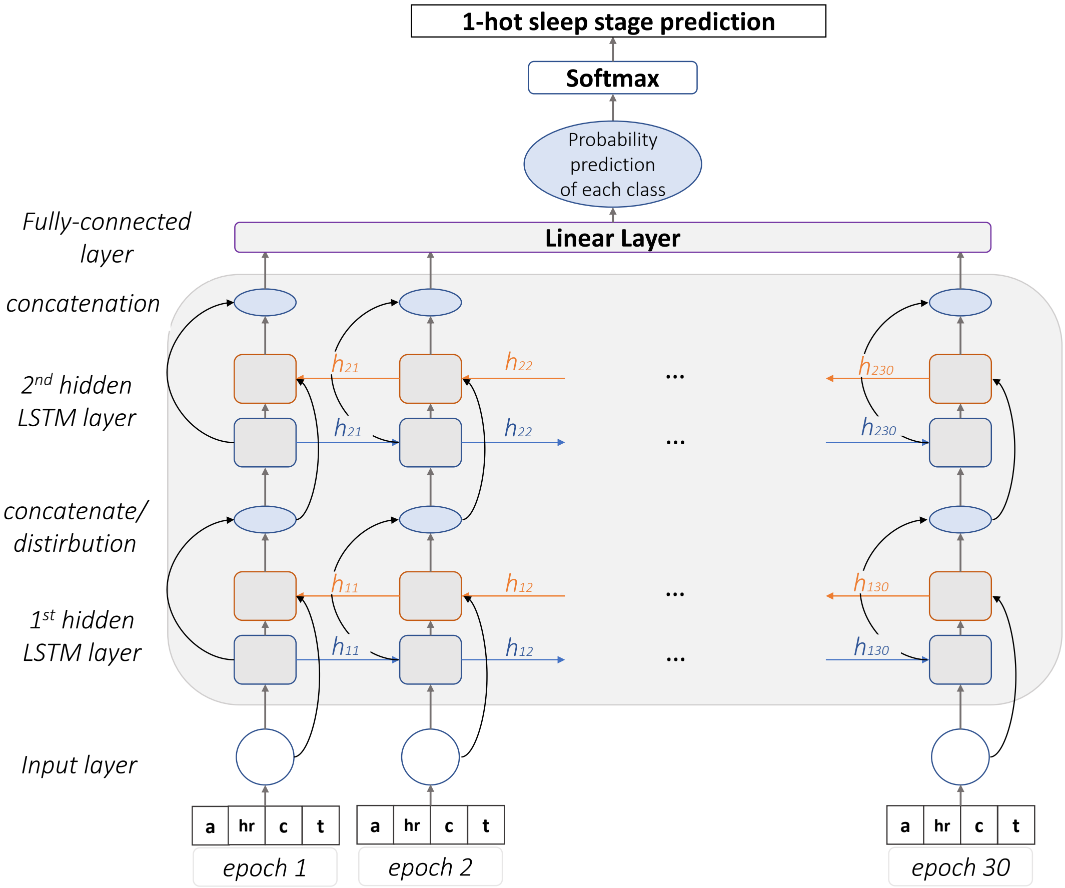


Figure 4.2.5: The proposed simple Bidirectional-LSTM architecture.

instead of flat ones [Kes+16]. The batch size and its effect on the performance of the model needs to be counterbalanced with the time efficiency that is desired for training and, hence, the choice of its size depends on each different case of training.

In our work, after experimenting with batch sizes of 8, 16, 32, 64, the best performing value for the shorter amount of time is 32. Thus, a batch size of 32 is used for the experiments that follow.

Training the model

The model takes as input pairs of sequences of continuous time segments with a specified length and their respective sleep label, which is the one defining the last epoch of each sequence. After experimenting with several sequence lengths, varying between [5, 10, 20, 30] time-steps per segment, it is shown that a 30 time-step window (15 continuous minutes of samples) gives the best results. Then, the model makes a prediction for the sleep stage of the last epoch of the segment, taking into consideration the given previous epochs as well. Thus, the data are split into 30-epoch overlapping windows, in order to incorporate all PSG labels for training. The first 29 PSG labels are not included, as there are not enough preceding data points to form the required sequence. To prepare the data, after the 30-epochs' segmentation, they are shuffled and split into train – evaluation and test sets, divided by 80-10-10% respectively. A specific seed is given to the shuffling function in order to always have the same data for training and evaluation, for the comparison of different model parameters. Sleep stages differ in their presence in the dataset, due to the natural cycle of sleep. Hence a weight is calculated for each one of them, by taking the percentage of its samples against the samples of all sleep stages in the dataset. Those weights are used for splitting in a balanced way between train, evaluation and test sets, to ensure that the amount of each sleep stage is proportionally similar between the three sets.

Experiment categories

There are four categories of sleep stage classification tested:

1. **Sleep - Wake**
2. **Wake - REM - NREM**
3. **Wake - Light - Deep - REM** (light = N1 & N2, deep = N3 & N4)
4. **Wake - N1 - N2 - N3 - REM** (N3 = true N3 & N4, as described in Section 1.3.1)

The sleep stage N4 was grouped together with N3 for all the experiments as it is proposed is several works, having a total of 5 sleep stages.

Training Parameters

The following parameters were tested, to find the combination giving the best results and the model with the highest accuracy. With bold are depicted the best resulting values in most of the cases. The final choices for training parameters for all sleep stages are presented in Table 4.1.

- number of layers for the LSTM: **2** or 3
- number of timesteps: 5 – 10 – 20 – **30**
- dropout for the LSTM: **0.5** or 0. (no dropout)
- learning rate: **0.001** – 0.0001 – 0.00001
- number of epochs for training: up to 1000 (best results for **epoch 800** in most cases)
- learning rate scheduler: reducing the learning rate by 0.1 after a checkpoint did not seem to improve the model, thus it was not applied on the final training.
- hidden size of LSTM: 256, **512**
- batch size: **32**
- loss_weights: when set to true, the class weights are used as a parameter for the criterion loss (CrossEntropy)

Table 4.1: Best model parameters for Bidirectional-LSTM.

	Sleep - Wake	**Wake - REM - NREM**	**Wake - Light - Deep - REM**	**Wake - N1 - N2 - N3 - REM**
Time sequence (time-steps)	30	30	30	30
dropout	0.5	0.5	0.5	0.5
learning rate	0.0001	0.001	0.001	0.001
Number of LSTM layers	2	2	2	2
LSTM hidden size	512	512	512	512
Batch size	32	32	32	32

4.2.3 Experimental Results

In this section the results of training the models on the Walch dataset and testing them on both the Walch and MESA datasets, are presented. The best resulting model configurations for each of the four sleep-stage categories are presented in Table 4.1.

Training and testing on Walch dataset

The results of the best-performing models on the Walch data can be seen in Table 4.2. The classification report on the test set is presented for all sleep stages. As it can be observed, the best accuracy is for the less complex problem of two sleep stage classification (Sleep - Wake), while its value degrades as the number of sleep stages to be predicted increases. Nevertheless, the accuracy is very promising for all the four categories of sleep stages, ranging between 79% for the five-class problem to 94% for the two-class one. Examining each class separately, however, the F1-score and support values indicate that the actual occurrences of each class in the dataset greatly affect the ability of the model to detect the class correctly. Thus, there is a discrimination between the total accuracy of the model and the F1-score of each class. Even though the total accuracy can be of a high value, the model might not be able to properly detect specific sleep stages, such as stage N1 in the five-class problem.

The confusion matrices of the experiments are visualized as heatmaps and can be seen in Figure 4.2.7. A confusion matrix is a class-wise distribution of the predictive performance of a classification model, where the columns represent the original or expected class distribution, and the rows represent the predicted or output distribution by the classifier. The cells with the brightest color on the heatmaps indicate that most of the predictions are collected in those categories, and the cells following the diagonal line of the map indicate correct predictions. As it can be seen from the heatmaps, the cells with the brightest color indicate that most of the predictions are consistent with the PSG labels, since the cells around the diagonal line contain most of the samples. However, there is a divergence for some sleep stages, which are misclassified with their neighboring stages in the sleep cycle. This observation is also consistent with the classification report, where the most prominent sleep stages have the highest values for F1-score, while the less-represented classes do not achieve so good classification results.

The train and validation accuracy and losses are seen in Figure 4.2.6. Although it seems like the models train quite fast from very early epochs, such as epoch 50, when inferring the sleep stages from the test data, the best accuracy is acquired at around epoch 800. Checkpoints are saved every 50 epochs, so that the model parameters can be retrieved for testing. The results compared to other works presented in Chapter 2.4 can be seen in Table 4.4. It is clear that the proposed bidirectional-LSTM model outperforms the NN utilized in the Walch paper, but it also gives a higher accuracy than the other wearable-based models. It can be seen that the works incorporating PSG signals for training the models achieve better performance, but this may be justified by several factors. The signals derived by a PSG are of greater detail than the ones coming from a wearable device, since they are typically much more dense with a higher sampling rate, but also the technology used tends to be more precise. Also, more channels are usually measured for each of the signals, such as in an EEG, making the final dataset more rich with useful information.

Table 4.2: BiLSTM - Classification report on the test set of the model trained on Walch dataset, for epoch 800, for the all the classification problems, hence All, Light-Deep, REM-NREM, Sleep-Wake.

	precision	**recall**	**f1-score**	**support**
wake	0.49	0.61	0.54	272
sleep	0.98	0.96	0.97	4420
accuracy			**0.94**	
macro avg	0.73	0.78	**0.75**	4692
weighted avg	0.95	0.94	0.94	4692

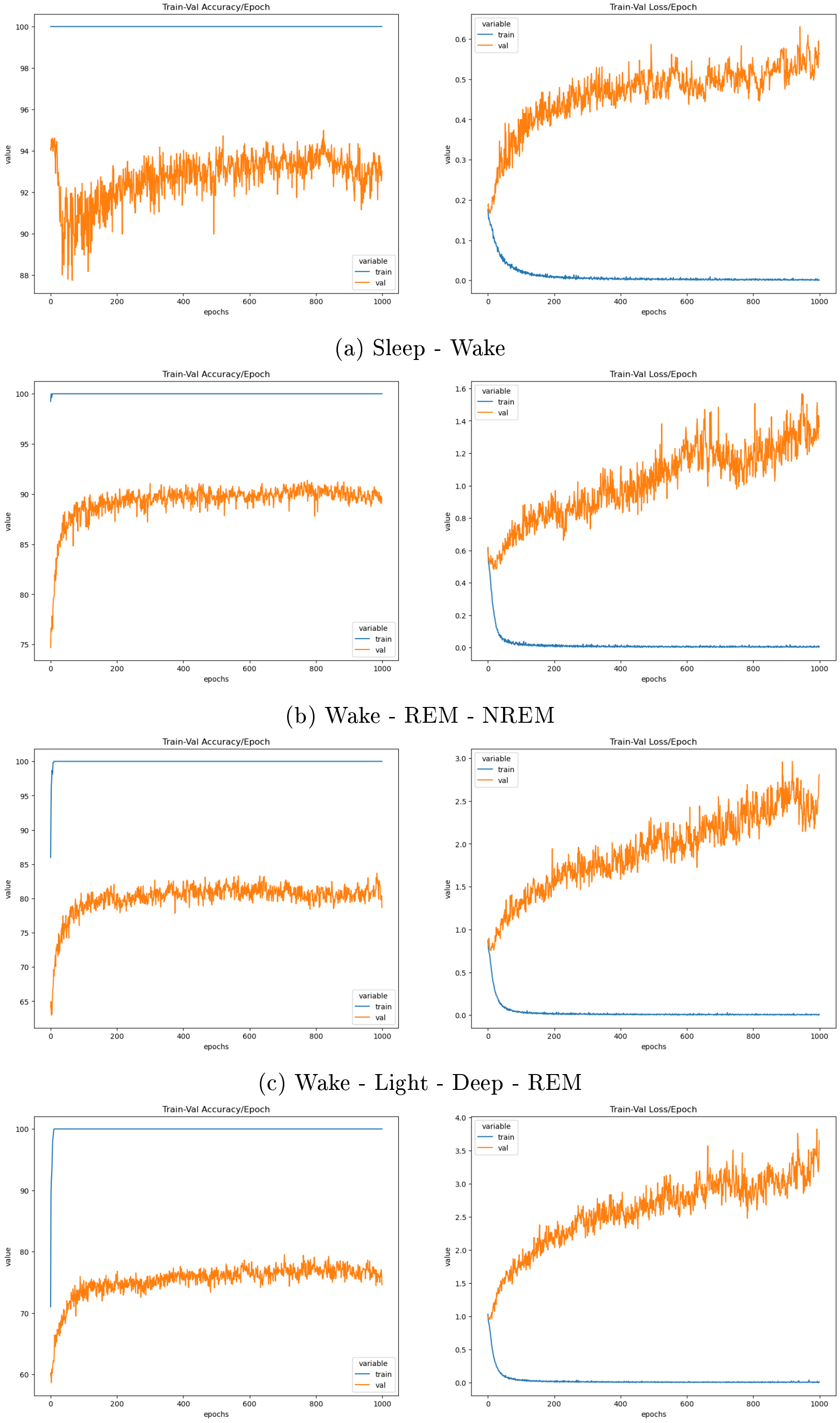
	precision	**recall**	**f1-score**	**support**
wake	0.68	0.64	0.66	272
nrem	0.93	0.94	0.93	3375
rem	0.84	0.84	0.84	1045
accuracy			**0.90**	
macro avg	0.82	0.80	**0.81**	4692
weighted avg	0.90	0.90	0.90	4692

	precision	**recall**	**f1-score**	**support**
wake	0.60	0.63	0.62	272
light	0.84	0.86	0.85	2677
deep	0.71	0.71	0.71	698
rem	0.84	0.75	0.79	1045
accuracy			**0.80**	
macro avg	0.74	0.74	**0.74**	4692
weighted avg	0.80	0.80	0.80	4692

	precision	**recall**	**f1-score**	**support**
wake	0.68	0.62	0.65	272
n1	0.46	0.34	0.39	306
n2	0.80	0.88	0.84	2371
n3	0.80	0.67	0.73	698
rem	0.85	0.84	0.85	1045
accuracy			**0.79**	
macro avg	0.72	0.67	**0.69**	4692
weighted avg	0.78	0.79	0.78	4692

Table 4.3: Simple BiLSTM - Classification report on the test set of the model trained on Walch dataset, with the criterion loss being parameterized with the class weights, for epoch 800, for the 5-sleep stage classification problem, hence All (Wake - N1 - N2 - N3 - REM).

	precision	**recall**	**f1-score**	**support**
wake	0.59	0.68	0.62	272
n1	0.34	0.43	0.38	306
n2	0.83	0.81	0.82	2371
n3	0.79	0.71	0.75	698
rem	0.84	0.83	0.83	1045
accuracy			**0.77**	
macro avg	0.68	0.69	**0.68**	4692
weighted avg	0.78	0.78	0.78	4692



(a) Sleep - Wake

(b) Wake - REM - NREM

(c) Wake - Light - Deep - REM

(d) Wake - N1 - N2 - N3 - REM

Figure 4.2.6: Train and Validation Accuracy (left column) and Loss (right column) of Bidirectional-LSTM, when training with the Walch dataset for 1000 epochs. Training data are depicted with the blue line and validation data are depicted with the orange line.

Adding class weights to criterion loss

One alteration to the baseline model is to add the class weights, as described in Section 4.2.2, as a parameter for the criterion loss. Cross-entropy is used as the criterion loss for the bidirectional-LSTM, which is a very typical choice for this type of multi-class classification problems. In the Pytorch implementation of CrossEntropyLoss, there is an optional *weight* argument, which, when provided, receives as input an 1D *Tensor* (an array visible by the GPU), that assigns a weight to each one of the classes. This asset is particularly useful when the training set is unbalanced, as the weight of each class is taken into account while training, to avoid emphasizing only on the most prominent classes. However, when adding a weight to the bidirectional-LSTM model, the performance does not improve; on the contrary the accuracy of the model even slightly decreases, as it can be seen in Table 4.3 for the case of all the 5-sleep stages problem.

Learning rate scheduler

One more alteration to the baseline BiLSTM model, is to add a learning rate scheduler during training, instead of having a permanent learning rate for the whole training session. A learning rate scheduler applies a different learning rate at each given epoch of the schedule, and the training continues with the new value, until it changes again. In the Pytorch implementation used in the current work, a parameter *gamma* is defined, with which the current learning rate is multiplied when a specified epoch is reached. Gamma is chosen to be $\gamma = 0.1$, thus the learning rate is reduced by one order of magnitude each time. This serves in letting the model learn with a slower pace after the epoch threshold, lowering the amount of updating the weights in each iteration. In our work, several combinations were tested. We have experimented using varying learning rates, in the range of $[0.01, 0.001, ...0.00001]$ and the milestone epochs for the weight update are $[300, 500, 800, ...]$. However, no improvement was detected compared to the baseline model using a standard learning rate. For instance, a model with the same starting parameters and learning rate as the baseline BiLSTM, with a value of $\gamma = 0.1$ and the epoch milestone being 300, will decrease its learning rate by an order of magnitude after epoch 300 and will learn in a slower pace. Experimental results have shown that for this model, the accuracy of the test set at checkpoint 800 is 0.77, which lies in the same range of values as the baseline and even it is slightly worse than it.

Evaluation on MESA dataset

After training the BiLSTM model on the Walch dataset, a similar approach to their proposed method is followed, testing its generalization capabilities on a subset of the publicly available MESA dataset. To do so, the same method as in the Walch approach is used for feature extraction, in order to be analogous to the features used in the training of the models. The extraction of heart rate and motion features is straightforward, while for the "clock proxy" feature, the ambulatory actigraphy recording for each MESA participant is used. Following the approach proposed in the aforementioned paper, the goal is to test the model performance to unseen individuals and whether it can generalize to a new dataset, unseen during training, which is derived from a different source. The subset of the first 188 individuals is used and the best performing model for each sleep-stage group is selected for the experiments.

In Table 4.5 are presented the results of testing the models of the four sleep-stage classification problems on the MESA subset. It can be observed that the models do not perform as well on data derived with a different monitoring method, in this case PSG and actigraphy. In the simpler problems such as 2- and 3-sleep stage classification, the model performs quite well, achieving accuracy of 60% and higher. However, focusing on the performance of each one of the sleep stages separately, it can be seen that the accuracy immensely decreases the more complex the problem gets, hence more sleep stages to be classified. Additionally, the sleep stages that have a weaker presence in the dataset, due to their shorter duration during sleep, tend to get a lower accuracy and be misinterpreted as the most prominent ones. In Figure 4.2.8 a hypnogram is depicted for the first subject of the MESA dataset, tested for 2- and 5-sleep stage classification. It can be seen that the model does not adapt so well on this sort of data.

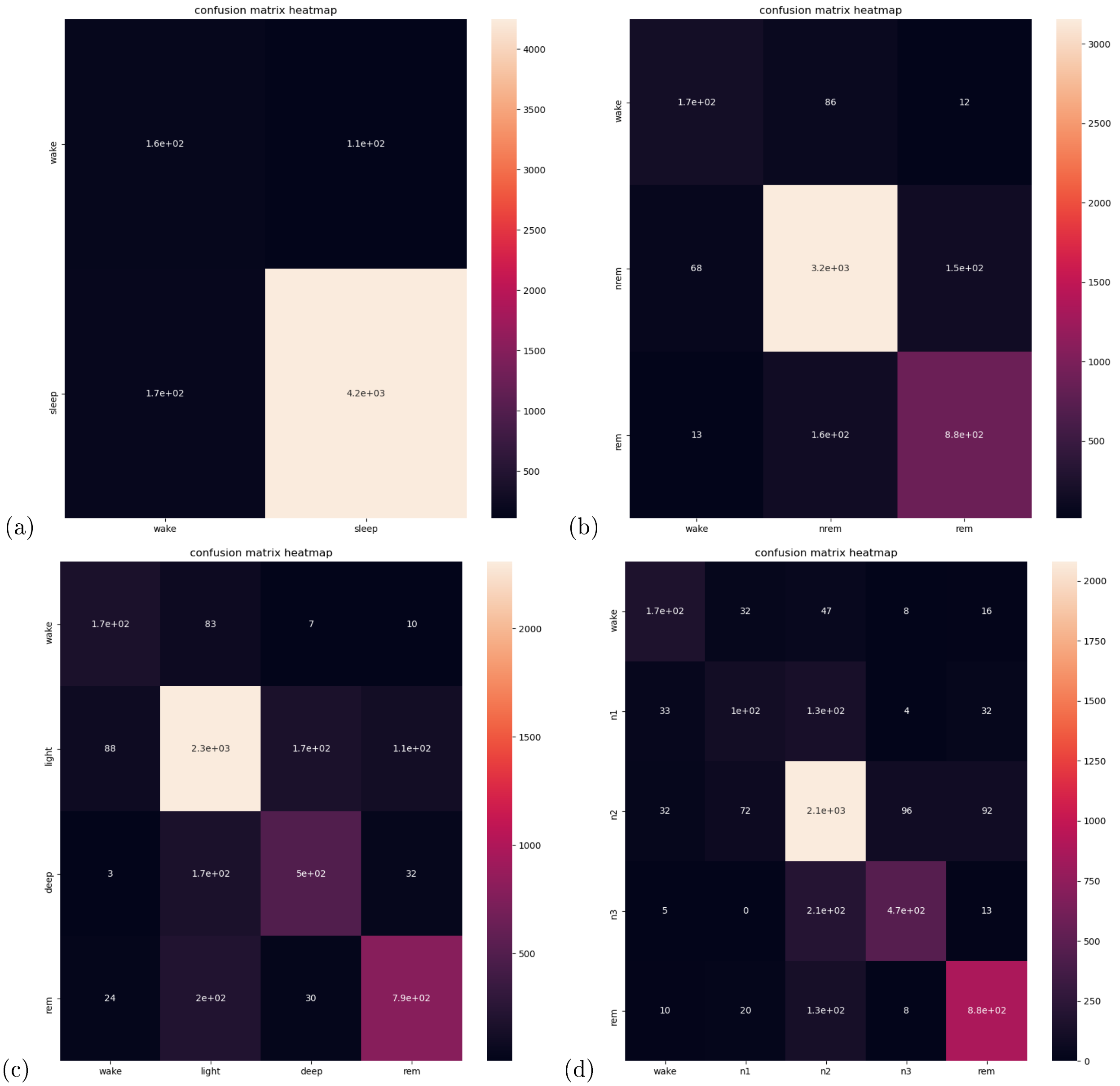


Figure 4.2.7: Confusion matrix heatmaps of the different classification schemes for the best performing BiLSTM models on the Walch-test set: (a) Sleep - Wake, (b) Wake - REM - NREM, (c) Wake - Light - Deep - REM, (d) Wake - N1 - N2 - N3 - REM.

Table 4.4: Comparison of accuracy achieved in previous works to ours, on all the four categories of classification. In yellow color are emphasized the ones using a bidirectional-LSTM architecture. In red color is highlighted the best accuracy in each classification category among all works, independently of whether the data used are derived by PSG or a wearable device. It can be seen that our proposed method incorporating the Walch-extracted features achieve the best accuracy in the wearable category.

Author	**Year**	**Data**	**Method**	**Sleep** **Wake**	**Wake** **REM** **NREM**	**Wake** **Light** **Deep** **REM**	**Wake** **N1** **N2** **N3** **REM**
Walch et al.	2019	HR PPG (Smartwatch)	Logistic Regression	0.8	0.71		
			k-nn	0.8	0.721		
			Random Forrest	0.799	0.702		
			MLP for NN	0.801	0.723		
X. Chen et al.	2020	EOG EEG EMG	Bi-LSTM-CNN				89.40%
			Bi-LSTM				84.80%
			Bi-LSTM-RNN				81.60%
P. Fonseca et al.	2017	PSG PPG Actigraphy	ML	91.50%	72.90%		59.30%
Zhang et al.	2018	Heart Rate Actigraphy	Bi-LSTM (group1)				58.20%
			Bi-LSTM (group2)				58.50%
H. Phan et al.	2021	EEG EOG	XSleepNet				80%
H. Phan et al.	2019	EEG EOG	SeqSleepNet (dataset1)				87.60%
			SeqSleepNet (dataset2)				89.10%
Z. Beattie et al.	2017	3D acc. PPG				69%	
Our results		HR PPG (Smartwatch)	BiLSTM	**94.22%**	**89.62%**	**81.30%**	**78.88%**
			CNN-BiLSTM		**0.56%**	**0.63%**	**0.58%**

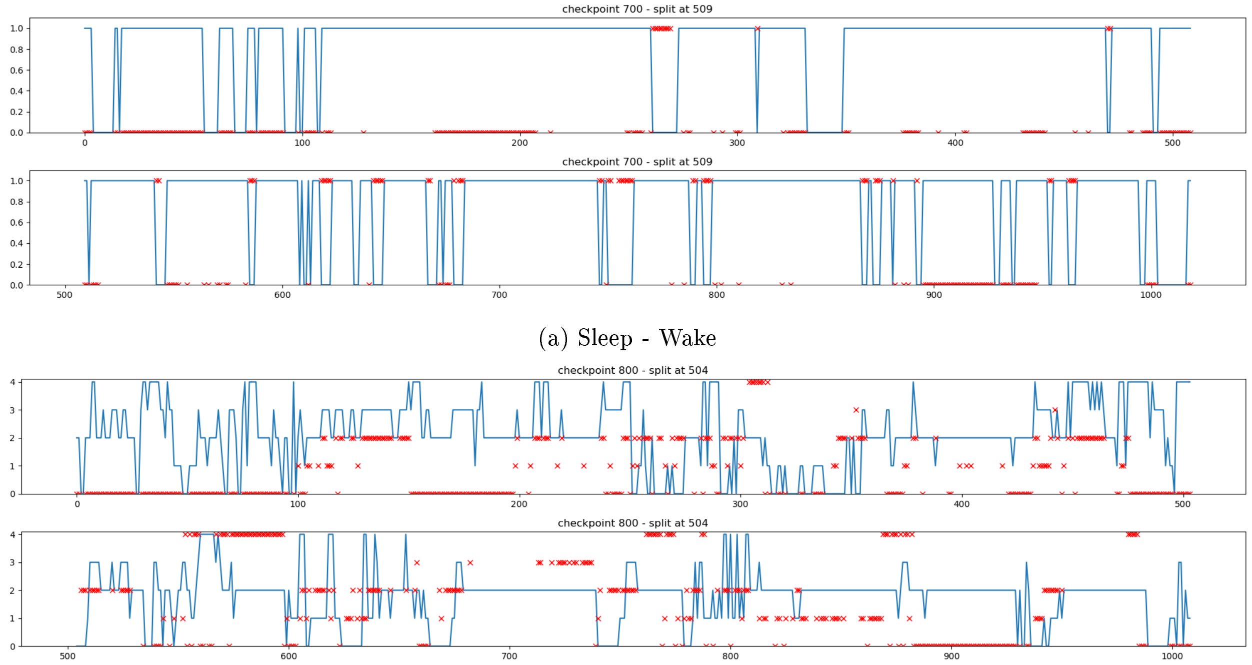
Table 4.5: BiLSTM - Model generalization: testing on MESA subset for all classification categories, hence All, Light-Deep, REM-NREM, Sleep-Wake.

	precision	**recall**	**f1-score**	**support**
wake	0.59	0.43	0.49	47536
sleep	0.80	0.89	0.84	126933
accuracy			**0.76**	
macro avg	0.70	0.66	**0.67**	174469
weighted avg	0.75	0.76	0.75	174469

	precision	**recall**	**f1-score**	**support**
wake	0.70	0.37	0.48	47536
nrem	0.66	0.82	0.73	102748
rem	0.17	0.15	0.16	24185
accuracy			**0.60**	
macro avg	0.51	0.45	**0.46**	174469
weighted avg	0.60	0.60	0.58	174469

	precision	**recall**	**f1-score**	**support**
wake	0.74	0.23	0.35	47536
light	0.54	0.75	0.63	88324
deep	0.23	0.23	0.23	14424
rem	0.15	0.14	0.14	24185
accuracy			**0.48**	
macro avg	0.41	0.34	**0.34**	174469
weighted avg	0.51	0.48	0.45	174469

	precision	**recall**	**f1-score**	**support**
wake	0.70	0.38	0.50	47536
n1	0.12	0.08	0.09	16332
n2	0.48	0.59	0.53	71992
n3	0.19	0.42	0.26	14424
rem	0.18	0.12	0.14	24185
accuracy			**0.41**	
macro avg	0.33	0.32	**0.30**	174469
weighted avg	0.44	0.41	0.40	174469



(a) Sleep - Wake

(b) Wake - N1 - N2 - N3 - REM

Figure 4.2.8: BiLSTM - Hypnograms for the subject 001 of the MESA dataset, tested on the 2- and 5-class models of the proposed bidirectional-LSTM. The blue lines are the PSG values and the red cross depicts the false predictions. Note: The hypnogram is split on the data point stated at the top of each figure, in order to better fit into the page.

4.3 Simple Bidirectional LSTM - MESA

Willing to study the full potential of our proposed simple bidirectional LSTM model, it is tested on the whole MESA dataset, which is quite bigger than the Walch dataset. In order to make an 1-1 comparison, the same features which were extracted for the first set of experiments in Section 4.2 are used. The RNN model used does not need any alterations or modifications, since the applied features are meant to express the same physical values for both Walch and MESA datasets. Thus, after the feature extraction, which is presented in [Wal+19], the bidirectional LSTM model is trained and tested on the whole MESA dataset.

4.3.1 Data preparation & Architecture

The features are extracted using the method proposed by Walch et al., giving as input the raw MESA data. The final features used for training are the same as in Section 4.2.1, hence consisting of **activity count**, **heart rate feature**, **cosine feature**, **time feature** and the corresponding **PSG labels**, per 30-second epochs.

In Figure 4.3.1 are depicted the raw heart rate and wrist activity features for the subject 0001, after cleaning the data by applying the preprocessing method described in Section 4.2.1. In Figure 4.3.2 are depicted the corresponding extracted features. The time feature is used for the horizontal axes of the rest of the feature plots, given that it expresses a linear time sequence, since the beginning of the recording.

The model architecture is the same as presented in Figure 4.2.5 and described in Section 4.2.2. The model parameters are tested and fine-tuned to improve the model performance for each one of the four sleep-categories. They are firstly initialized with the parameter values chosen for the simple BiLSTM, as derived for the Walch dataset. The best parameters are presented in Table 4.6. The experimental results are analyzed in the following section.

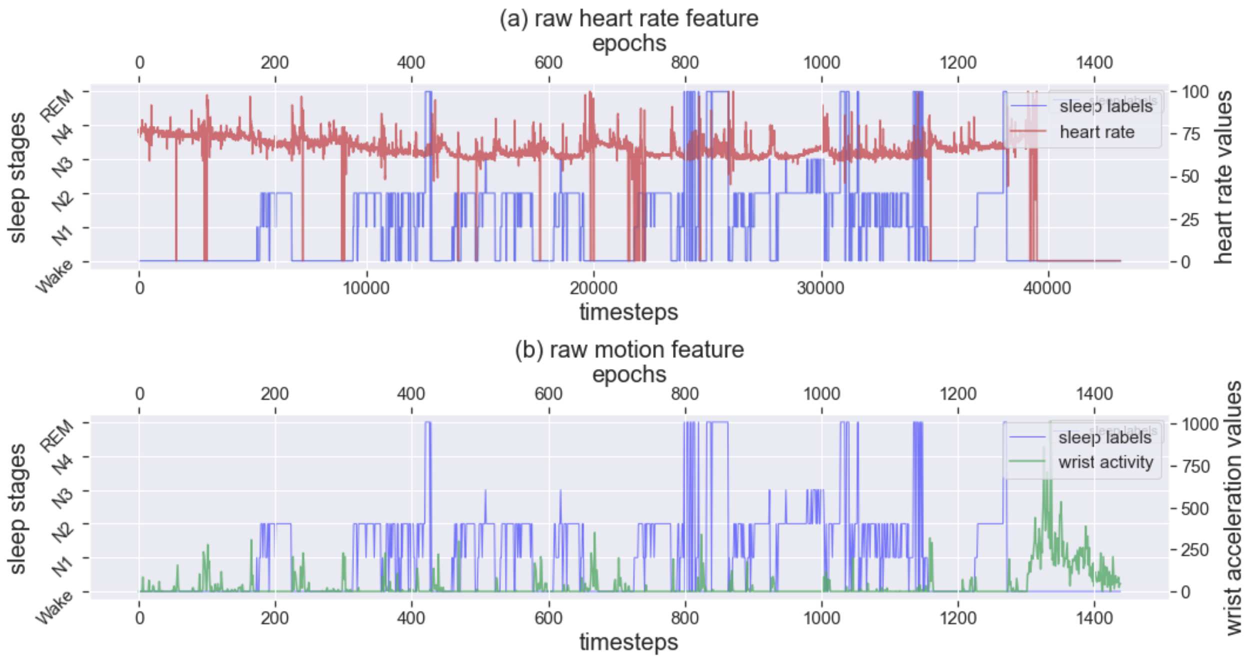


Figure 4.3.1: The raw features of subject 0001 from the MESA dataset, with their corresponding sleep labels. (a) raw heart rate and (b) wrist acceleration

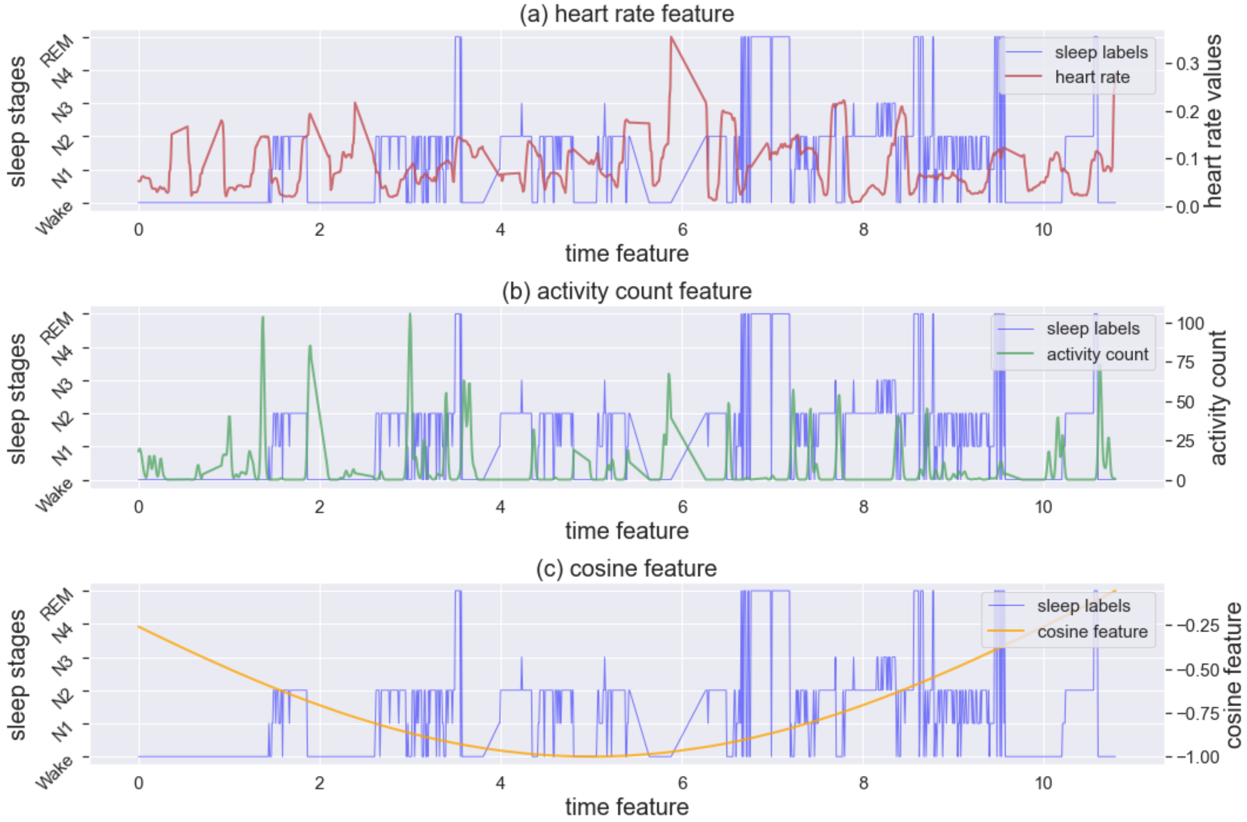


Figure 4.3.2: The extracted features with their corresponding sleep labels, for the MESA subject 0001. The x-axes is the time feature, which is counting since the start of the recordings. (a) heart rate feature, (b) activity count feature, (c) cosine feature

Table 4.6: Best model parameters for Bidirectional-LSTM on MESA.

	Sleep - Wake	Wake - REM - NREM	Wake - Light - Deep - REM	Wake - N1 - N2 - N3 - REM
Time sequence (time-steps)	30	30	30	30
dropout	0.5	0.5	0.5	0.5
learning rate	0.0001	0.0001	0.0001	0.0001
Number of LSTM layers	2	2	2	1
LSTM hidden size	512	512	512	512
Batch size	32	32	32	32

4.3.2 Experimental Results

After experimenting with different values for the network parameters, it is shown that a similar architecture to the one used for the Walch dataset is also suitable for the MESA dataset. Specifically, when more than two layers are used for the LSTM module, it tends to overfit and becomes unstable while training. Hence, for the case of the MESA dataset, two layers are also used for most of the sleep stage experiments, except for the one where all the five sleep stages are to be predicted. This classification problem is the most complex one, and the experimental results indicate that a more shallow architecture with a smaller learning rate can better capture the different class details. Furthermore, the amount of data in the MESA dataset is quite larger than in the Walch dataset, thus in each training epoch the model is exposed to more information and a slower learning pace might be needed. Reducing the learning rate by one order of magnitude prevents the model from overfitting and assures a more stable training loss. The different monitoring methods used for the MESA dataset might also affect the way the model trains, since the internal patterns might differ compared to the Walch dataset.

The classification report of the best performing model for each sleep category is presented in Table 4.7. Additionally, in Table 4.8, can be seen the classification report for the 5-class problem, where the criterion loss of the model is not parameterized by the class weights. The accuracy of the model using the class weights for the criterion loss is similar to the model with no such weighting parameters. We keep the model with the parameterized criterion loss as the main experimental architecture, since these were the initial experiments to be conducted. Observing the classification tables, the accuracy values are proportional to the complexity of the problem to be solved, hence the more sleep stages to be predicted, the lower the accuracy value is. In the case of five sleep stages, the accuracy ranges around 53%, which indicates that there is a high tendency for the correct predictions to be so by chance and the model performs relatively poor. In the rest of the classification problems (two, three and four sleep stages), the model performs better, with the accuracy ranging between 63% for the four-class and 78% for the two-class problem. The above observation is verified by the F1-score of the different classification models, as its value increases the less sleep stages to be predicted. F1-score is a weighted harmonic mean of precision and recall, between 1.0 (best model performance) and 0.0 (worst model performance). This model behavior could be a sign of overfitting towards the most prominent categories, or in general a difficulty of the network to capture some of the more detailed aspects of the data, in order to classify each sleep stage correctly.

4.4 CNN-Bidirectional LSTM

In the next stage of the experiments, a more advanced approach to the bidirectional-LSTM baseline is adopted. Instead of having as input the extracted features as proposed by Walch et al., the raw data are used. It is a common practice to incorporate a CNN architecture as an automatic feature extraction module and feed its outputs to an RNN network in the case of time sequences classification or prediction (forecasting), which has been described in detail in previous works in Chapter 2. The data used in our work are sequential, hence an 1D convolutional architecture is chosen. Since motion and heart rate raw data have different sampling rates, two separate CNN architectures are created for each one of them respectively, and by taking the concatenation of

Table 4.7: BiLSTM trained on MESA - Classification report on the test set for epoch 300 - 300 - 800 - 300 for the All, Light-Deep, REM-NREM, Sleep-Wake class problem respectively.

	precision	**recall**	**f1-score**	**support**
wake	0.57	0.69	0.62	13129
sleep	0.88	0.80	0.84	35739
accuracy			**0.78**	
macro avg	0.72	0.75	**0.73**	48868
weighted avg	0.79	0.78	0.78	48868

	precision	**recall**	**f1-score**	**support**
wake	0.53	0.79	0.64	13129
nrem	0.82	0.64	0.72	29108
rem	0.46	0.45	0.45	6631
accuracy			**0.66**	
macro avg	0.60	0.63	**0.60**	48868
weighted avg	0.69	0.66	0.66	48868

	precision	**recall**	**f1-score**	**support**
wake	0.63	0.74	0.68	13129
light	0.69	0.69	0.69	24933
deep	0.35	0.30	0.32	4175
rem	0.54	0.40	0.46	6631
accuracy			**0.63**	
macro avg	0.55	0.53	**0.54**	48868
weighted avg	0.63	0.63	0.63	48868

	precision	**recall**	**f1-score**	**support**
wake	0.58	0.73	0.65	13129
n1	0.18	0.24	0.20	4504
n2	0.62	0.55	0.59	20429
n3	0.34	0.31	0.32	4175
rem	0.56	0.38	0.45	6631
accuracy			**0.53**	
macro avg	0.94	0.44	**0.44**	48868
weighted avg	0.53	0.53	0.53	48868

Table 4.8: Simple BiLSTM - Classification report on the test set of the model trained on MESA dataset for all five sleep stage categories (Wake - N1 - N2 - N3 - REM), with the criterion loss of the model not taking the class weights as an extra parameter.

	precision	**recall**	**f1-score**	**support**
wake	0.59	0.75	0.66	13129
n1	0.19	0.2	0.19	4504
n2	0.62	0.59	0.6	20429
n3	0.34	0.32	0.33	4175
rem	0.56	0.34	0.42	6631
accuracy			**0.54**	
macro avg	0.46	0.44	**0.44**	48868
weighted avg	0.54	0.54	0.53	48868

their outputs, the extracted features are forwarded to the previously introduced simple bidirectional-LSTM, by altering some of the model parameters in order for the newly extracted features to fit to the input of the network. The total network is trained end-to-end, meaning that the data pass through the whole length of the network, both the CNN and the bidirectional-LSTM module, before applying back-propagation, hence all its components are trained simultaneously.

4.4.1 Data Preprocessing

The disadvantage of CNNs compared to RNNs is that they take a standard-sized input instead of the flexibility that an LSTM offers regarding the length of the input sequence. Thus, in order to secure this condition, the following preprocessing steps are applied on the raw Walch dataset:

1. Firstly, the same segmentation method as in the first experiment, Section 4.2, is applied on the data, in order to remove the unscored PSG labels (denoted as -1 in the dataset.) This way, continuous time sequences are created, which may be sub-sequences of the initial one.
2. The second step is to apply interpolation on both raw features, so that all 30-second segments (epochs) have the exact same number of samples. The interpolation frequency is indicated by the dataset description, and it is the same as the sampling rate of monitoring, at 50Hz for motion and at 1Hz for heart rate.
3. Finally, the interpolated data are standardized per individual, in order to have a normal distribution and lie in the same space for all participants:

$$z = \frac{X - \mu}{\sigma} \tag{4.4.1}$$

   where μ, σ are the mean and standard deviation of the individual for each feature and X is the value at each data point.

The final features consist of two sequences of data points, one for the motion feature and one for the acceleration, being 1- and 3-dimensional, respectively. Every continuous time sequence occurring from the segmentation of the whole raw data consists of a number of epochs and its length has to be equal to or greater than the chosen number of time steps that will form each input sequence. We experiment with 5, 10, 20 and 30 time steps as it was done for the simple bidirectional-LSTM baseline. Each epoch consists of exactly 1 sample for the motion feature and 50 3D samples for the acceleration feature, for axes x, y, z respectively.

4.4.2 Architecture

The design of the CNNs has to be made with great precaution, since the output dimensions of the two features (motion and acceleration) need to have identical lengths, in order to be concatenated and fed in parallel to the bidirectional-LSTM, as it was done in the baseline model. To do so, some technical characteristics of the CNNs need to be examined. Specifically, how the dimensionality of the input data in a CNN model affects the dimensionality of its output.

The output size of a CNN depends on two kinds of variables: (i) the size and dimensions of the input data and (ii) the parameters that define the network. The parameters characterizing the network are the ones controlling the convolution operation as presented in Section 3.7.1, namely **depth, stride, zero-padding** and **dilation**, also the number of filters used, each one studied separately as follows:

- The **depth** of the network differs for the heart rate feature and the motion one. To begin with, the features extracted by the CNN will be fed to the bidirectional LSTM afterwards, to continue the process of automated data analysis and pattern recognition, for the final classification of each epoch. Thus, the CNN network does not need to be very deep, in contrast a narrow architecture is preferred. Since the HR feature has a small sampling rate of 1Hz, meaning that only a short amount of data points is included in each training sequence, a single-layer CNN is chosen, with the depth parameter being equal to 1. For the acceleration feature, the sampling rate is much higher, at 50Hz, so a 2-layered network is designed to process it. The parameters of each convolutional layer are defined separately from each other and which will be presented in greater detail below.

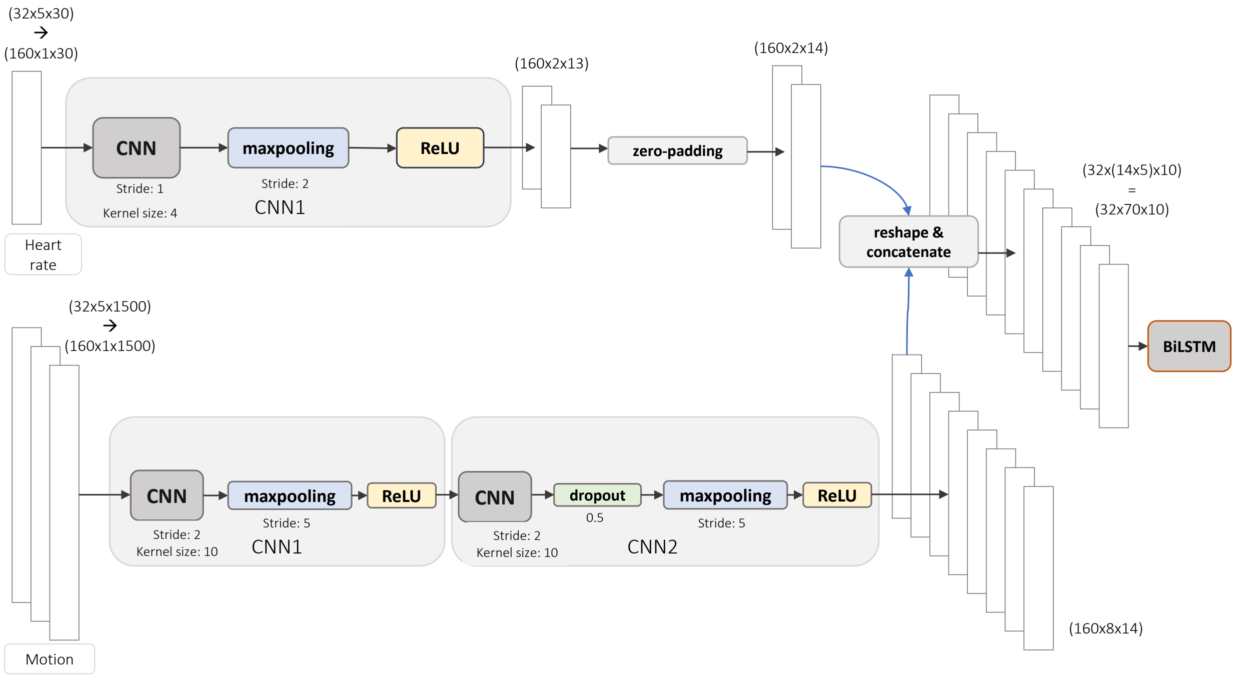


Figure 4.4.1: The proposed convolutional architecture for the automated feature extraction of the raw heart rate and acceleration data, in order to be given to the bidirectional LSTM afterwards. The **time distributed** method is adopted, where the sequence length is aligned with the batch size for computational efficiency.

- No **zero-padding** is added to neither of the two CNN architectures; both heart rate and motion sequences already have a defined and fixed length, so there is no need to adjust them again inside the CNNs. However, each heart rate feature sequence, extracted by the respective HR 1-layer CNN, is extended and reshaped afterwards, in order to match with the length of the motion feature sequence. Motion has a larger sampling rate, so it is expected to have a longer output sequence after the 2-layered CNN. Thus the HR output feature has to be zero-padded to match with the length of the motion feature sequence, in order to be then given together, as a whole input, to the bidirectional LSTM module. The process of how the motion and heart rate features are prepared and given as input to the bidirectional LSTM are described in the following Paragraph of *Time-Distributed network* 4.4.2.

- **Dilation** is omitted in the initial experiments as well. It refers to the spacing between the kernel elements and it offers a wider field of view without increasing the computational cost at the same time. Dilated convolution can expand the receptive field without pooling, allowing each convolution output to contain a wide range of information, and it is typically applied to problems that require longer sequence information dependencies such as speech and text. However, given that sampling rate is not very high neither for HR nor for the motion feature and long-term dependencies are supposed to be handled by the LSTM afterwards, dilation is not incorporated in the current CNN models.

- **Stride** refers to the stride of the cross-correlation. It determines the length of the output sequence, and for the heart rate feature at least one sample per epoch is needed. Thus, since HR has sampling rate of 1Hz, a stride of 1 is used for it, so that the convolutional operation moves one step at a time and the desired condition is satisfied. Regarding the motion feature, the sampling rate is much higher at 50Hz, thus a stride of size 2 is used for both CNNs of the motion feature network, to constrict its output length. The mentality behind this choice of stride values, is that, since the already proposed bidirectional LSTM will be incorporated for the experiments, a similar input of features would be preferred, in terms of their shape and size. The value of the stride for the two CNNs is chosen so that the output of the networks has a size close to the size of the manually extracted features, both for heart rate and for motion. The method proposed by Walch, introduces a single representative value of every feature for each epoch, so the automatically extracted features by the CNN modules try not to differ much from this form.

- The **number of filters** used in the CNN network determine how many output features will be, which are also called *channels*.
- **Batch size** does not alter the output shape of the network, but it is a factor of how the input sequences are given to the model. For this network a batch size of 32 is used.

Given an input for a CNN network of size (N, C_{in}, L_{in}), where N is the batch size, C denotes the number of channels and L is the length of the signal sequence, the output size is (N, C_{out}, L_{out}) and the output length is calculated as:

$$L_{out} = \lfloor \frac{L_{in} + 2 \times padding - dilation \times (kernel\_size - 1) - 1}{stride} + 1 \rfloor \tag{4.4.2}$$

Time Distributed network

For the design of the CNNs used in the proposed model, a specific method is applied, namely *time distributed*. The layer parameters are as described above, however the way the input sequences are given to the network slightly differs than the standard approach. *Time Distributed is a wrapper Layer that will apply a layer on every temporal slice of the temporal dimension of an input.* Originally, it was introduced in Keras [Cho+15], but a workaround can be implemented in Pytorch, as well. The goal of the method is to optimally handle the input data in order to save up computational time and memory, in the case of sequence processing.

In the specific case of our work, the input data are split into **5-timestep sequences** (for each of those sequences one sleep stage is aligned for prediction), and each sequence contains four features, the three being the 3D acceleration for motion, and the fourth is the heart rate. Each time step represents a 30-second segment and $30seconds \times 50Hz = 1500$ data points are for motion, while $30seconds * 1Hz = 30$ data points are for heart rate. Thus, in order to form the feature vector, heart rate is initially zero-padded to be of the same length as motion, thus 1500 data points. Given the batch size defined as 32, an input to the model during training is of size $(batch\_size, time\_steps, channels, data\_points) = (32, 5, 4, 1500)$. The goal is to extract for each time step some features from the raw data, in an automated way through the CNNs. Thus, the input vector is reshaped as $(batch\_size \times time\_steps, channels, data\_points) = (32 \times 5, 4, 1500)$, where the batch size is perceived as the batch samples and their number of time steps, and each input of the CNN is just the 1500 features of a single 30-second time step. This method gives robustness to the model, as it allows the incorporation of a smaller network architecture and it also takes advantage of the optimized way that Pytorch utilizes the GPU for the parallel computations of the batch samples.

As previously explained, a separate CNN architecture is used for the motion and the HR features. The input data array is split into a motion-feature array of shape $(batch\_size, time\_steps, channels, data\_points) = (32, 5, 3, 1500)$ and an HR-feature array of shape $(batch\_size, time\_steps, channels, data\_points) = (32, 5, 1, 30)$, where the zero-padding at the end of the array is removed. Then, each of these arrays is reshaped again, in order to get the desired format of $(batch\_size \times time\_steps, channels, data\_points)$, as described. The raw feature arrays are given to each of the two CNN architectures, with an output of size $(batch\_size \times time\_steps, output\_channels, output\_sequence)$, where the output sequence is derived by Equation 4.4.2.

Finally, the output arrays are reshaped again to regain their initial format $(batch\_size, time\_steps, channels, data\_points)$, and the heart rate is zero-padded, to exactly match the length of the motion array. The final arrays are concatenated in their third dimension, which corresponds to the number of channels, and are ready to be forwarded to the next layer, which is the bidirectional LSTM used in the previous experiments as well.

4.4.3 Experimental Results

The experimental results from the different classification schemes are presented in Table 4.9. The bidirectional LSTM used for those experiments applies the weighted cross-entropy loss, as it was discussed in the baseline model's section. Interestingly, the accuracy does not tremendously differ between the classification

Table 4.9: CNN-BiLSTM - Classification report on the test set of the model trained on Walch dataset, for epoch 800, for all the classification categories, hence All, Light-Deep, REM-NREM, Sleep-Wake

	precision	**recall**	**f1-score**	**support**
wake	0.22	0.53	0.31	417
sleep	0.95	0.83	0.89	4610
accuracy			**0.80**	
macro avg	0.59	0.68	**0.60**	5027
weighted avg	0.89	0.80	0.84	5027

	precision	**recall**	**f1-score**	**support**
wake	0.28	0.59	0.38	417
nrem	0.84	0.54	0.66	3525
rem	0.37	0.64	0.47	1085
accuracy		8	**0.56**	
macro avg	0.49	0.59	**0.50**	5027
weighted avg	0.69	0.56	0.59	5027

	precision	**recall**	**f1-score**	**support**
wake	0.36	0.70	0.47	417
light	0.81	0.60	0.69	2818
deep	0.52	0.64	0.57	707
rem	0.60	0.69	0.64	1085
accuracy			**0.63**	
macro avg	0.57	0.66	**0.60**	5027
weighted avg	0.69	0.63	0.65	5027

	precision	**recall**	**f1-score**	**support**
wake	0.45	0.54	0.49	417
n1	0.21	0.27	0.23	345
n2	0.71	0.64	0.67	2473
n3	0.49	0.53	0.51	707
rem	0.61	0.61	0.61	1085
accuracy			**0.58**	
macro avg	0.49	0.52	**0.50**	5027
weighted avg	0.60	0.58	0.59	5027

categories, while it can be seen that the wake stages are the most difficult to predict. Also, the total accuracy per classification category is degraded compared to the baseline bidirectional LSTM model using the manually extracted features presented in Section 4.2.1. This observation indicates that the proposed automated feature extraction through CNN modules is not capable of detecting so much in detail the correct feature characteristics for discriminating between the different sleep stages, compared to the manually extracted features.

4.5 CNN-Bidirectional LSTM - MESA

The CNN-Bidirectional LSTM model is tested on the raw MESA dataset as well. To do so, the whole MESA data is incorporated and some alterations are applied to the proposed network, since the provided raw MESA data differ from the ones collected by Walch.

4.5.1 Data preparation & Architecture

The given raw MESA data consist of **heart rate** and **activity count**. The activity count is manually labeled and has one corresponding value per 30-second epoch, thus there is no need for self-supervised feature extraction by a neural layer, as it was done with the analogous raw acceleration data of the Walch dataset. Hence, the CNN-BiLSTM model described in Section 4.4 is slightly altered. Specifically, the activity

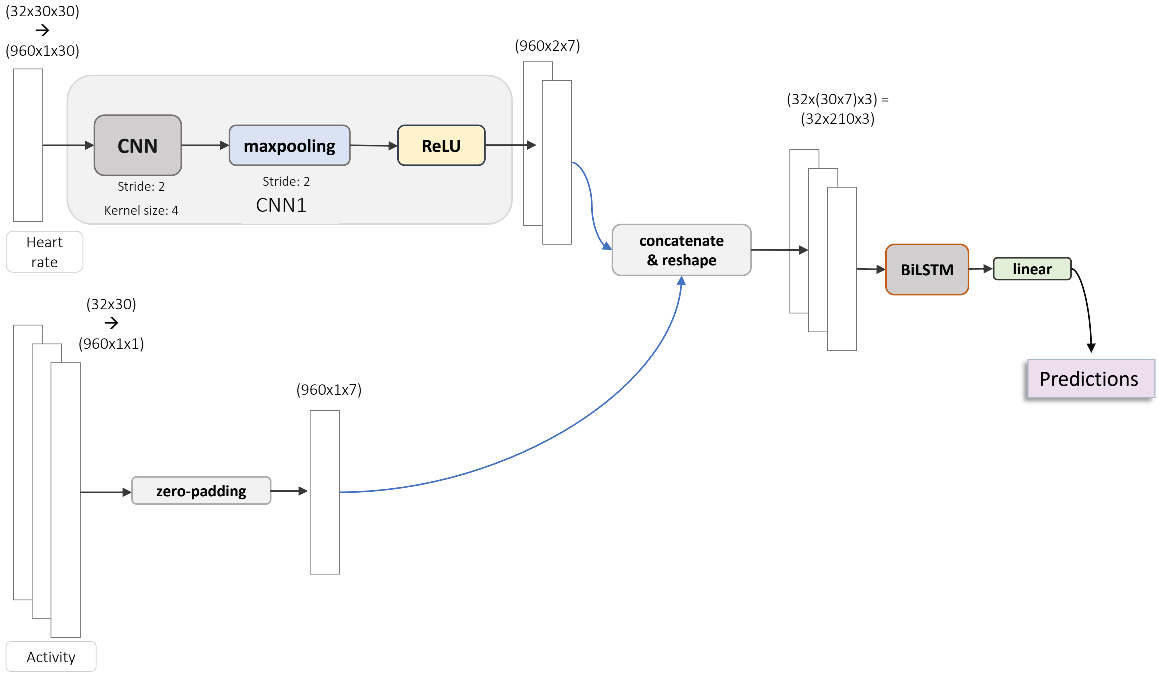


Figure 4.5.1: The proposed convolutional architecture for the automated feature extraction of the raw heart rate and acceleration data, in order to be given to the bidirectional LSTM afterwards. The **time distributed** method is adopted, where the sequence length is aligned with the batch size for computational efficiency.

(motion) CNN layer is discarded and only the CNN for the heart rate feature extraction is used in the model. The activity count feature is zero-padded in order to match the dimensions of the extracted heart rate feature. Then, the output of the heart-rate CNN is concatenated with the padded activity count feature, they are reshaped and given as input to the bidirectional LSTM layer, where the temporal information of the data is learned, as shown in Figure 4.5.1. The output of the LSTM layer is passed through a linear layer and the final predictions for each sleep stage are made.

The model is initialized with the best parameter values found in Section 4.4 and, after experimenting with a variety of values, the best model parameters for the CNN-BiLSTM on the MESA dataset can be seen in Table 4.10.

Table 4.10: Best model parameters for CNN - Bidirectional-LSTM trained on the MESA dataset.

	Sleep - Wake	**Wake - REM - NREM**	**Wake - Light - Deep - REM**	**Wake - N1 - N2 - N3 - REM**
Time sequence (time-steps)	30	30	30	30
dropout	0.5	0.5	0.5	0.5
learning rate	0.0001	0.0001	0.0001	0.0001
Number of LSTM layers	1	1	1	1
LSTM hidden size	512	512	512	512
Batch size	32	32	32	32

Table 4.11: CNN-BiLSTM - Classification report on the test set of the model trained on MESA dataset, for epoch 300, for all the classification categories, hence All, Light-Deep, REM-NREM, Sleep-Wake

	precision	**recall**	**f1-score**	**support**
wake	0.85	0.87	0.86	26518
sleep	0.91	0.89	0.9	37206
accuracy			**0.88**	
macro avg	0.88	0.88	**0.88**	63724
weighted avg	0.88	0.88	0.88	63724

	precision	**recall**	**f1-score**	**support**
wake	0.78	0.89	0.83	26518
nrem	0.87	0.77	0.82	30355
rem	0.77	0.75	0.76	6851
accuracy			**0.82**	
macro avg	0.81	0.8	**0.8**	63724
weighted avg	0.82	0.82	0.82	63724

	precision	**recall**	**f1-score**	**support**
wake	0.8	0.9	0.85	26518
light	0.81	0.76	0.78	26087
deep	0.75	0.5	0.6	4268
rem	0.79	0.77	0.78	6851
accuracy			**0.8**	
macro avg	0.79	0.73	**0.75**	63724
weighted avg	0.8	0.8	0.8	63724

	precision	**recall**	**f1-score**	**support**
wake	0.79	0.88	0.83	27034
n1	0.24	0.21	0.22	4949
n2	0.71	0.71	0.71	21674
n3	0.68	0.5	0.57	4286
rem	0.82	0.68	0.74	6998
accuracy			**0.73**	
macro avg	0.65	0.6	**0.62**	64941
weighted avg	0.72	0.73	0.72	64941

Table 4.12: CNN BiLSTM - Classification report on the test set of the model trained on MESA dataset for all five sleep stage categories (Wake - N1 - N2 - N3 - REM), with the criterion loss of the model being conditioned on the class weights.

	precision	**recall**	**f1-score**	**support**
wake	0.67	0.9	0.77	27034
n1	0.16	0.28	0.21	4949
n2	0.78	0.28	0.42	21674
n3	0.4	0.38	0.39	4286
rem	0.51	0.62	0.56	6998
accuracy			**0.58**	
macro avg	0.5	0.49	**0.47**	64941
weighted avg	0.63	0.58	0.56	64941

4.5.2 Experimental Results

The classification report of the best model for each sleep stage category is presented in Table 4.11. Additionally, in Table 4.12 can be seen the classification report of the 5-class sleep stage problem, where the criterion loss for training the model is conditioned on the class weights, due to the imbalanced nature of the dataset.

4.6 Discussion - LSTM models

Table 4.13 collectively presents the accuracy values and the macro-average F1-scores from all LSTM models tested on both Walch and MESA datasets.

Accuracy is the fraction of predictions the model got right, hence

$$accuracy = \frac{(Number \quad of \quad correct \quad predictions)}{Total \quad number \quad of \quad predictions} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.6.1}$$

The **F1-score** represents the arithmetic mean per sleep stage label, and is calculated as the weighted harmonic mean of precision and recall.

$$F1 - score = 2 \times \frac{Precision \times Recall}{Presicion + Recall} \tag{4.6.2}$$

where

$$Precision = \frac{TP}{TP + FP}, \quad Recall = \frac{TP}{TP + FN} \tag{4.6.3}$$

and $TP : True \quad Positives, TN : True \quad Negatives, FP : False \quad Positives, FN : False \quad Negatives.$

The F1-score should be used for model comparisons, since it embeds both precision (the accuracy of positive predictions) and recall (the fraction of positives that were correctly identified). In our case, the models can be compared based on their F1-score grouped by the dataset they were trained on, while accuracy can be used as a universal comparison among inter-dataset comparisons. It can be seen that the best F1-scores for the Walch dataset models occur in the simple bidirectional LSTM using the designed features of Section 4.2. On the contrary, for the MESA dataset, the model with the highest F1-score is the CNN-BiLSTM of Section 4.5, where a CNN module is used for automatic feature extraction from the raw data.

The best overall accuracy appears on the Simpe-BiLSTM model trained on the Walch dataset, but the CNN-BiLSTM trained on the MESA dataset follows quite closely, with its values ranging around $0.80 + -8\%$. It can be concluded that the internal structure of the data and their hidden patterns pay a crucial role in how a model performs on each dataset, as well the depth of the network needed for a successful classification. The Walch dataset, although quite smaller, with the correct choice of manual feature extraction can give good results with a narrow and simple, hence less computationally demanding architecture. However, we could not show that there is a good generalization capability of the pre-trained model on the Walch dataset to correctly classify samples extracted from different sources, such as the MESA dataset, even though they have undergone a similar manual feature extraction process. Regarding the MESA dataset, it has performed better in the model containing the automated feature extraction CNN module, where the raw data are given as input for training. This could be either due to the larger amount of data, being able to properly train the model, or it could be due to the different internal nature and distribution of the dataset. The main differences between the Walch and MESA datasets, except of the amount of data, are the different technology used for collecting each dataset, as well the different group of participants used for each. Specifically, in the MESA dataset description it is stated that the subjects participating in the study are aged between 45-84 years old, while the Walch dataset recruited subjects from the University of Michigan without defining their age range. There might be a need for a larger and more representative group of participants in order to achieve good generalization on unseen subjects. Also, a possible fine-tuning of a system on different datasets, by training the pre-trained model for a few iterations on the new dataset could be another approach for a better generalization.

Finally, a degradation in both accuracy and F1-score is observed the more complex the classification problem becomes, meaning the more classes to be predicted correctly. This is quite intuitive due to the increased

Table 4.13: The accuracy and F1-scores of the best model for all sleep-stage categories are collectively presented, for both the Walch and MESA dataset.

		Sleep-Wake	REM-NREM	Light-Deep	All
Walch					
BiLSTM	**F1-score**	0.75	0.81	0.74	0.69
	Accuracy	0.94	0.90	0.80	0.79
BiLSTM on MESA	**F1-score**	0.67	0.46	0.38	0.30
	Accuracy	0.76	0.60	0.48	0.41
CNN-BiLSTM	**F1-score**	0.60	0.50	0.60	0.50
	Accuracy	0.80	0.56	0.63	0.58
MESA					
BiLSTM	**F1-score**	0.73	0.60	0.54	0.44
	Accuracy	0.78	0.66	0.63	0.63
CNN-BiLSTM	**F1-score**	0.88	0.80	0.75	0.62
	Accuracy	0.88	0.82	0.80	0.73

difficulty of the task; however, the heavily imbalanced nature of the dataset should play a crucial role as well. As seen in the data distributions in Figure 4.1.1, some of the classes are greatly under-represented, being difficult for the models to correctly learn to predict them. This does not improve with increasing the amount of the training data, as we can see from the MESA CNN-BiLSTM model compared to the Walch simple-BiLSTM one, hence a more thorough study of different architectures and more defined approaches for the imbalanced dataset need to be examined.

4.7 SeqSleepNet

As a final experiment of this thesis, a model initially designed to handle PSG signals for the task of sleep stage classification was incorporated and altered in order to use the Walch and MESA datasets, to test its performance on wearable-derived data. The original work of SeqSleepNet is described in [Pha+19] and the code for the implementation can be found in https://github.com/pquochuy/SeqSleepNet. This model is implemented using **TensorFlow v1.3.0** [Mar+15]. In the proposed model, the time-sequence nature of the data is taken into account, leading to the design of a **many-to-many** architecture for taking advantage of this characteristic. The the task is perceived as a sequence-to-sequence classification problem that receives a sequence of multiple epochs as input and classifies all of their labels at once.

In the original work, the training was done with a 200 subject dataset, split into 180 train - 10 evaluation - 10 test sets. The data consists of a 3-channel PSG signal of ECG, EOG and EMG with an initial sampling rate of 256Hz, which was downsampled at 100Hz for the experiments. The log-spectrograms of the data are extracted per 30-second epoch per channel, and are used as input to the model. Since three channels are used for the initial SeqSleepNet, the model's input consists of a 3-channel time-frequency image S^C, where C is the number of the image's channels.

4.7.1 Basic Architecture

The SeqSleepNet architecture comprises of three different modules, trained in an end-to-end manner.

1. The first one is a **Filterbank layer** for learning frequency-domain filterbanks. Taking as input the extracted time-frequency images, the learned filterbank is expected to emphasize the most useful sub-bands for the required task and attenuate the less prominent ones. Also, since the nature of the PSG channels used in the initial SeqSleepNet work greatly differ from one another, a different filterbank layer is used for each one of them; so, in this case 3 separate filterbank layers are used to learn three channel-specific filterbanks.

   Each filterbank layer is modeled by a **fully connected layer of M hidden units**. The number of hidden units represent the number of filters, where $M < F$ and F if the size of the initial image in the frequency domain. Then, the weight matrix $\mathbf{M}^C \in \mathbb{B}^{F\times M}$ of the layer counterparts as the filterbank's

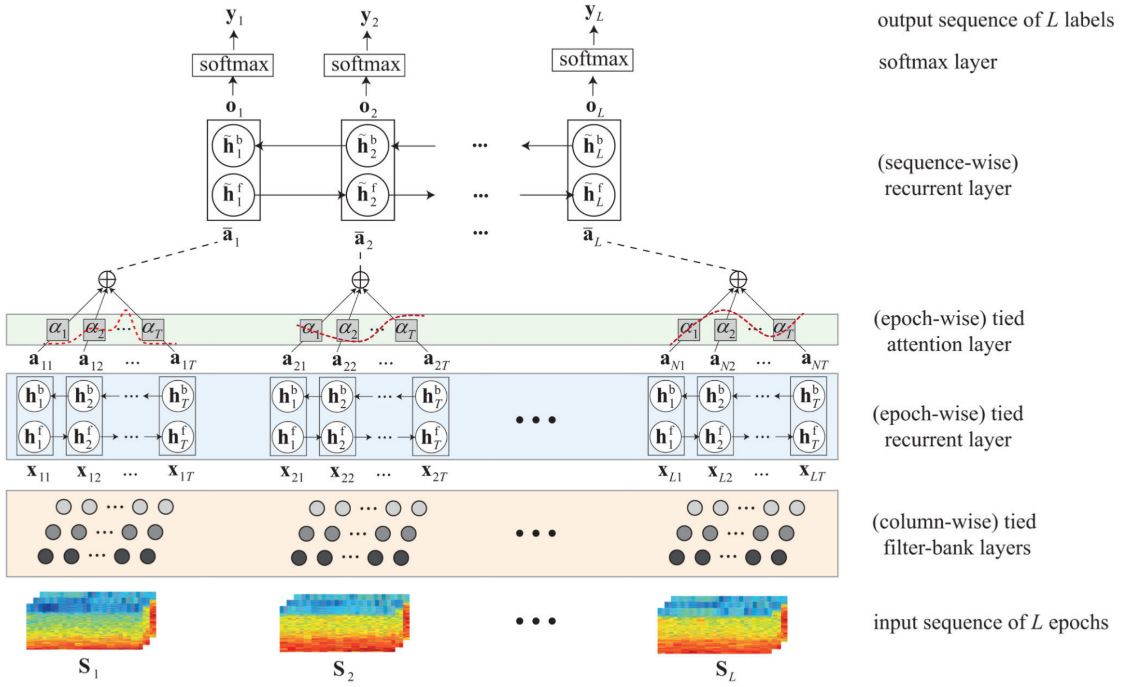

y_1
y_2
y_L
softmax
softmax
softmax
o_1
o_2
o_L
$\tilde{h}_1^b$
$\tilde{h}_1^f$
$\tilde{h}_2^b$
$\tilde{h}_2^f$
$\tilde{h}_L^b$
$\tilde{h}_L^f$
$\bar{a}_1$
$\bar{a}_2$
$\bar{a}_L$
α_1 α_2 α_T
a_{11} a_{12} ... a_{1T}
a_{21} a_{22} ... a_{2T}
a_{N1} a_{N2} ... a_{NT}
h_1^b h_2^b h_T^b
h_1^f h_2^f h_T^f
x_{11} x_{12} ... x_{1T}
x_{21} x_{22} ... x_{2T}
x_{L1} x_{L2} ... x_{LT}
S_1
S_2
S_L
output sequence of L labels
softmax layer
(sequence-wise) recurrent layer
(epoch-wise) tied attention layer
(epoch-wise) tied recurrent layer
(column-wise) tied filter-bank layers
input sequence of L epochs

Figure 4.7.1: SeqSleepNet architecture, an end-to-end hierarchical RNN for sequence-to-sequence sleep stage classification, utilizing signal spectrograms as the network's inputs [Pha+19].

weight matrix. Typically, a filterbank has the characteristics of being *non-negative*, *band-limited* and *ordered in the frequency domain*, and some constrains are enforced to the FC layer that need to be satisfied.

$$\mathbf{X}^C = \mathbf{W}_{fb}^C \top \mathbf{S}^C, \mathbf{X}^C \in \mathbb{R}^{M \times T}$$
$$\mathbf{W}_{fb}^C = f_+(\mathbf{W}) \odot \mathbf{T}, \mathbf{T} \in \mathbb{R}_+^{F \times M} \tag{4.7.1}$$

where

- f_+ is the sigmoid function, which is non-negative and will make the elements of the matrix $\mathbf{W}$ non-negative as well.
- $\mathbf{T}$ is a linear-frequency triangular filterbank matrix, in order to enforce the filters to have limited band, and
- the operator $\odot$ is the element-wise multiplication.

The output image $\mathbf{X}^C$ is smaller in the frequency dimension than the input $\mathbf{S}^C$, and concatenation is applied on the frequency dimension as well, resulting in a final image of size $MC \times T$. This image can be interpreted as a sequence of T feature vectors, $\mathbf{X} = (x_1, x_2, ..., x_T)$, where each $x_t \in \mathbb{R}^{MC}, 1 \leq t \leq T$ can be perceived as an image column at time index t.

2. For the second layer of the proposed model, a **bidirectional-RNN** coupled with an **attention mechanism** [LPM15; BCB14] is incorporated to learn short-term sequential features for epoch representation. Specifically, a **GRU** network is employed, since it has fewer parameters making it more computationally efficient. The equations describing the GRU layer are presented in Chapter 3.7.2, Equation 3.7.2. Provided that the forward and backward sequences of the hidden state vectors $\mathbf{H}^f = (\mathbf{h}_1^f, \mathbf{h}_2^f, ..., \mathbf{h}_T^f)$ and $\mathbf{H}^b = (\mathbf{h}_1^b, \mathbf{h}_2^b, ..., \mathbf{h}_T^b)$ are computed as:

$$\mathbf{h}_t^f = \mathcal{H}(\mathbf{x}_t, \mathbf{h}_{t-1}^f)$$

$$\mathbf{h}_t^b = \mathcal{H}(\mathbf{x}_t, \mathbf{h}_{t+1}^b) \tag{4.7.2}$$

where $\mathcal{H}$ is the hidden state function as described in the Equations 3.7.2, then the output sequence of the GRU $\mathbf{A} = (\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_T)$ is computed as:

$$\mathbf{a}_t = \mathbf{W}_{ha}[\mathbf{h}_t^b \oplus \mathbf{h}_t^f] + \mathbf{b}_a \tag{4.7.3}$$

where $\mathbf{b}_a$ is the bias.
Finally, an **attention mechanism** is used in this layer to strengthen the more informative parts of the sequence and lower the focus of the weaker ones. To achieve this, a weighting vector is learned through the attention mechanism in order to combine the output vectors $\mathbf{a}_t$ at different time steps into a single feature vector. The attention weight α_t at the time index t is computed as:

$$\alpha_t = \frac{exp(f(\mathbf{a}_t))}{\sum_{i=1}^{T} exp(f(\mathbf{a}_i))}$$

$$f(\mathbf{a}) = \mathbf{a}\top\mathbf{W}_{att} \tag{4.7.4}$$

where f is the scoring function of the attention layer and $\mathbf{W}_{att}$ is a trainable weight matrix, that gets improved, while training the whole model. Finally, the attention feature vector, which is used as the representation of each PSG epoch in the next level of the network, is calculated as:

$$\bar{\mathbf{a}} = \sum_{t=1}^{T} \alpha_i \mathbf{a}_t \tag{4.7.5}$$

3. The third layer of the original SeqSleepNet network consists of a sequence-level bidirectional-RNN, to capture the long-term temporal information across epochs, by modeling the sequence of epoch-wise feature vectors. Given the attentional feature vectors per epoch as calculated by Equation 4.7.5, the total attentional feature vector for a sequence of length L is represented as $\bar{\mathbf{A}} = (\bar{\mathbf{a}}_1, \bar{\mathbf{a}}_2, ..., \bar{\mathbf{a}}_L), 1 \leq l \leq L$. A **bidirectional-GRU module** is incorporated, following the same structure proposed in the second layer of the network, taking as input the attentional feature vector $\bar{\mathbf{A}}$ and returning the sequence of vectors $\mathbf{O} = (\mathbf{o}_1, \mathbf{o}_2, ..., \mathbf{o}_L), 1 \leq l \leq L$, where each output vector $\mathbf{o}_l$ is computed similarly to Equation 4.7.3:
$$\mathbf{o}_l = \tilde{\mathbf{W}}_{ho}[\tilde{\mathbf{h}}_l^b \oplus \tilde{\mathbf{h}}_l^f] + \tilde{\mathbf{b}}_o \tag{4.7.6}$$
where $\tilde{\mathbf{h}}_l^f, \tilde{\mathbf{h}}_l^b$ are the forward and backward hidden states respectively, $\tilde{\mathbf{W}}_{ho}$ is the weight matrix and $\tilde{\mathbf{b}}_o$ is the bias.

4. Finally, the output of the second GRU layer $\mathbf{O}$ is passed through a **softmax layer** in order to produce the probability predictions of one epoch belonging to each of the sleep stages, for all the epochs of the current input sequence. Thus, the model gives as an output a classification sequence $\hat{\mathbf{Y}} = (\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2, ..., \hat{\mathbf{y}}_L)$, where $\hat{\mathbf{y}}_l$ is the output probability distribution over all sleep stages for the l^{th} epoch.

The loss used for training the network needs to take into account all the predictions made for each input sequence $(\mathbf{S}_1, \mathbf{S}_2, ..., \mathbf{S}_L)$. Denoting the classification predictions for the epochs of a sequence as $(\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2, ..., \hat{\mathbf{y}}_L)$ and the ground-truth PSG labels as one-hot vectors $(\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_L)$, then the sequence loss is defined as follows:

$$E^S(\theta) = -\frac{1}{L}\sum_{l=1}^{L} \mathbf{y}_l log(\hat{\mathbf{y}}_l(\theta)) \tag{4.7.7}$$

While training the network, the backpropagation of the loss is done over N training sequences at a time, thus the final loss to be minimized at each iteration of the network over a batch of N training sequences of the data is:

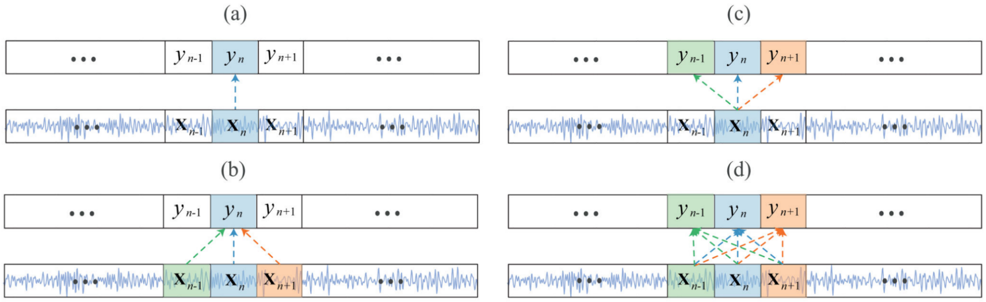


Figure 4.7.2: The many-to-many approach adopted in the SeqSleepNet model (d), compared to the other possible classification schemes: (a) one-to-one, (b) many-to-one, (c) one-to-many [Pha+19].

$$E(\theta) = \frac{1}{N}\sum_{n=1}^{N} E_n^S(\theta) + \frac{\lambda}{2}\|\theta\|_2^2 \tag{4.7.8}$$

where λ is a hyperparameter for trading off the error terms and the $\ell_2 - norm$.

For the final prediction of the network, an **ensemble of decisions** and **probabilistic aggregation** is used, as proposed in a previous work of [Pha+18], where a multiplicative aggregation scheme is shown to be efficient. SeqSleepNet is a multiple-output network, giving a prediction for all the epochs participating in each input sequence. Given that the input sequence is of length L, then advancing it by one epoch when evaluating the model on test data will result in an *ensemble* of L decisions at every epoch. Fusing the ensemble of predictions for each epoch to a final classification decision performs better than having individual predictions for the sleep stage of each epoch.

The $log-$posterior probability of each sleep stage $y_t \in \mathcal{L} = \{W, N_1, N_2, N_3, REM\}$ at time index t is calculated as follows:

$$\log P(y_t) = \frac{1}{L}\sum_{i=t-L+1}^{t} \log P(y_t|\mathcal{S}_i) \tag{4.7.9}$$

where $\mathcal{S}_i = (\mathbf{S}_i, \mathbf{S}_{i+1}, ..., \mathbf{S}_{L-1})$ is the epoch sequence starting at point i.

Then, *likelihood maximization* is applied for the final prediction of the network $\hat{y}_t$:

$$\hat{y}_t = \arg\max_{y_t} \log P(y_t), y_t \in \mathcal{L} \tag{4.7.10}$$

Training the network

For training the network, the data are split in batches that are used in the same iteration for the backpropagation of the loss, to improve the network weights. Each batch consists of S sequences, and each sequence consists of L epochs. For simplicity, the assumption of a single-channel input is made. Given this, the network uses only one filterbank layer for the first processing step of the input. Each 30-second epoch of the training data, which is assigned to one PSG label, is represented by a time-frequency image of size $T \times F$. For passing the data through the first layer of the network, each epoch image is interpreted as a sequence of T image columns. Thus, the S input sequences are unfolded to a set of $S \times L \times T$ image columns, each of size F, to be presented to the filterbank layer. The filterbank layer outputs the same form of image-columns of size $S \times L \times T$, however each column is now of size M instead of F. The output set is folded again to form a set of new feature-images of size $S \times L$, and each image is of size $T \times M$, and is given as input to the next layer. The epoch-level attention-based bidirectional-LSTM encodes each epoch's image into an attentional feature vector, resulting in a set of S sequences, each one consisting of L attentional feature vectors. This set is passed through the last layer of the sequence-level bidirectional-LSTM for the sequence-to-sequence classification and the final sleep stages prediction.

4.7.2 Walch Data Preprocessing

To prepare the Walch dataset for training and testing on SeqSleepNet, some preprocessing steps are first applied.

1. Firstly, the data are split on the timestamps, where missing epochs exist for at least one of the motion and heart rate features, in the same manner it was done for the BiLSTM and CNN-BiLSTM experiments.
2. Then, the data are **interpolated** in order to have a sampling rate of exactly 50Hz for the 3D motion feature and 1Hz for the heart rate.
3. Afterwards, the **log-spectrograms** are extracted for all the features. The extracted spectrograms are saved as *.mat* files, which is the file format used in the original SeqSleepNet work, where the PSG data are preprocessed using Matlab.
4. Additionally, regarding the PSG labels of the Walch dataset, they were prepared in the same manner that was done for the BiLSTM experiments in the preprocessing method described in Section 4.2.1, removing the -1 values indicating unlabeled sleep epochs.

The final spectrograms after the preprocessing steps, are ready for training and consist of images of size $(29, 129)$, where 29 are the dimensions in the time-axis and 129 are the dimensions in the frequency domain.

Extracting the spectrograms

The spectrograms are required to have the same size as the original PSG spectrograms that are used in the SeqSleepNet work, of dimensions (29,129) in the time and frequency domain respectively. To achieve this, some technical details on how spectrograms are extracted in python need to be addressed.

The *scipy* python library is used for this purpose [Vir+20]. The **motion feature** is examined first, since it consists of three channels, matching the 3-channel data structure of the original SeqSleepNet work. For the extraction of the motion feature's spectrograms, the *Short Time Fourier Transform* is firstly applied on the data. Given the continuous-time equation of STFT in 2.3.2, the respective discrete time equation is:

$$\mathbf{STFT}\{x[n]\}(m,w) \equiv X(m,w) = \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-jwn} \tag{4.7.11}$$

In the discrete time case, the data to be transformed are broken up into chunks or frames, which usually overlap each other, to reduce artifacts at the boundary. On each chunk, the Fourier transform is applied, and the complex result is added to a matrix, which records magnitude and phase for each point in time and frequency. $x[n]$ denotes the signal and $w[n]$ is the window, where m is discrete and ω is continuous. Typically in computer applications the STFT is performed using the Fast Fourier Transform, on which both variables are discrete and quantized, described by the following equation:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{i2\pi kn}{N}},\ k = 0, ..., N-1 \tag{4.7.12}$$

The STFT output size is defined by the window size and overlapping percentage used for the calculation. Given that the input signal's size for the motion feature is (30 seconds per epoch)×(50 samples per second) = 1500 data points, then in order to get 29 spectrograms, a *hamming window* of size 110 is chosen with 55% overlap.

The *spectrogram* is mathematically derived through the squared magnitude of the STFT of a signal:

$$\mathbf{spectrogram}\{x(t)\}(\tau,\omega) \equiv |X(\tau,\omega)|^2 \tag{4.7.13}$$

For the **heart rate** feature a similar approach is followed, based on the specific signal characteristics. The heart rate has a sampling rate of 1Hz after the interpolation, meaning that for (30 *seconds per epoch*) × (1 *sample per second*) = 30 data points are in each 1-epoch sequence. In order to get an output spectrogram of (29,129) as it was desired for the motion feature as well, the window used for the spectrogram calculation is a *hamming* window of size 100, with a 50% overlap.

Finally, as a last step for both the motion and the heart rate feature, the common logarithm with base 10 is applied, in order to convert the spectrograms to log-power spectra. Thus, those extracted features are used for testing the SeqSleepNet on the Walch -wearable derived- dataset.

4.7.3 Experimental Results - Walch

For the experiments, except for the log-spectrograms suggested in the original SeqSleepNet work, some of the other previously described spectral features were tested to examine the performance of the model on them as well. The data were split in train-eval-test sets in a manner similar to what was done for the BiLSTM experiments, in a percentage of around 90 - 5 - 5 %. Specifically, a text file is manually created defining which of the segments belong to the train - evaluation and test sets. Thus, for each training of the model, the same data are used for consistency of the different experimental setup comparison.

After splitting the data of each individual into continuous epoch segments, the experimental sets are created so that different segments of each individual exist in all data sets. The way that SeqSleepNet is designed, the model is subject agnostic, meaning that no subject identification is given to the model for each training sample. However, in the experiments conducted in our work, the individuals used for testing are also seen during training, but the segments of the data are unique in each set.

Training with motion data

Since the SeqSleepNet model originally uses a three-channel signal, the acceleration data are utilized in the first experiments, also consisting of a 3D signal, to extract the required log-spectrogram feature as described above. Also, the motion feature has a much higher sampling rate than the sampling rate of the heart rate feature, meaning that the extracted spectrogram should contain much more information. After testing with a batch size of 2, 8 and 16, *batch size* = 16 was chosen as the best for the current experimental setup.

The following experimental combinations were tested:

- **log-spectrogram**
- **stft**
- **spectrogram**

with the learning rate varying between $[0.001, 0.0001]$. The rest of the model's parameters are the same as the ones proposed in the original SeqSleepNet work.

Unfortunately, the experimental results indicate that when training is done with the acceleration data the accuracy while training can be as high as 70% or more, but the test and evaluation accuracy is pretty low at around 20% or less.

Training with heart rate data

As a second experiment, the heart rate feature was tested on the SeqSleepNet, with poor results. Due to the quite small sampling rate of only 1Hz compared to 200Hz in the original SeqSleepNet data, the log-spectrograms cannot capture proper information to be then learned by the model, as it was intended. The accuracy when training with the heart rate spectrograms stays at around 50%, which shows that the model is not capable of distinguishing useful information from training, and there are wide fluctuations indicating the model has potentially over-fitted.

Since the HR sampling rate is so low, the spectrograms do not seem like an optimal choice for handling this kind of feature, as a big amount of information seems to be lost, and by inspecting the output spectrograms from the different sleep stage categories, no consistency seems to exist.

4.8 SeqSleepNet - MESA Modifications

Due to the MESA dataset being already in a preprocessed form, the SeqSleepNet architecture can not be directly applied on it. Instead, two modifications of the network are utilized for this purpose, where a different type of input is used, since spectrograms cannot be extracted from the current form of MESA data. The main idea of the proposed modifications is that the already preprocessed features can be directly applied on the second level of data processing of the SeqSleepNet architecture. Thus, instead of passing the extracted spectrograms of the raw data through filter-bank layers in order to capture meaningful features from them, the already prepared features are directly passed through the epoch-wise recurrent layer, and then, the outputs are given to the attention layer that follows.

4.8.1 Modification 1

The first data application on the network is a simplified version of the original SeqSleepNet architecture. The activity and heart rate features are split into sequences of continuous samples with a specified length, which are then directly fed to the (column-wise) tied filter-bank layers. The model is supposed to predict the sleep stages of all the data points in the sequence, but, since it is temporal in nature, the prediction of the last data point is the most crucial, as it defines the current sleep stage of the subject.

The current approach is independent of the typical 30-second epochs defined by a single sleep stage label, since the MESA dataset version used in this work is labeled every 1 second.

- **Heart rate** feature has a sampling rate of 1Hz, meaning that it has 1 sample per second. Thus it can be considered ready for training, and no extra preprocessing steps need to be applied.
- **Wrist acceleration** on the contrary is provided with 30 values per second in the dataset, meaning that an automated feature extraction and selection approach via neural networks can be applied on the data sequence corresponding to each sleep stage label. Following the already existing architecture of SeqSleepNet, the HR sequence is passed through a bidirectional recurrent layer in order to obtain an intermediate representation, which is then passed through an attention layer so that the most meaningful features are kept.

  The output of the attention layer is concatenated with the activity feature and then the features defining each epoch are passed through an epoch-level bidirectional recurrent layer in order to obtain the probabilities for each epoch to match to each one of the existing sleep stages.

  Finally, the probabilities for each epoch of the sequence are passed through a fully-connected linear layer and a softmax activation function to get the final sleep-stage prediction.

  The model's architecture can be seen in Figure 4.8.1.

4.8.2 Modification 2

For the second modification of the SeqSleepNet architecture, we wanted to take into account the concept of a 30-second epoch corresponding to each sleep stage. To do so, the data are split into continuous 30-second frames, keeping the sleep stage of the last timestep as the correct one, and then consecutive frames form the input of the model for training. The number of consecutive frames forming each training input constitutes one of the hyperparameters to be tuned for the best performance of the model.

The basic SeqSleepNet architecture remains the same as described in Modification 1 presented in Section 4.8.1, however the way the input data is handled changes to serve the extra dimension of the 30-second epochs for each data point.

- The heart rate has a single value for each data point in the temporal dimension since it has sampling rate of 1Hz, thus it has one dimension less than the wrist activity data.
- Similarly to the previous method, the motion (acceleration feature) feature is passed through a bidirectional recurrent layer in order to obtain an intermediate representation for each frame-level sequence.
- However, since in the current method an epoch level sequence is examined at each training step, consisting of several 30-second frame-level consecutive sequences, the final features to be selected need

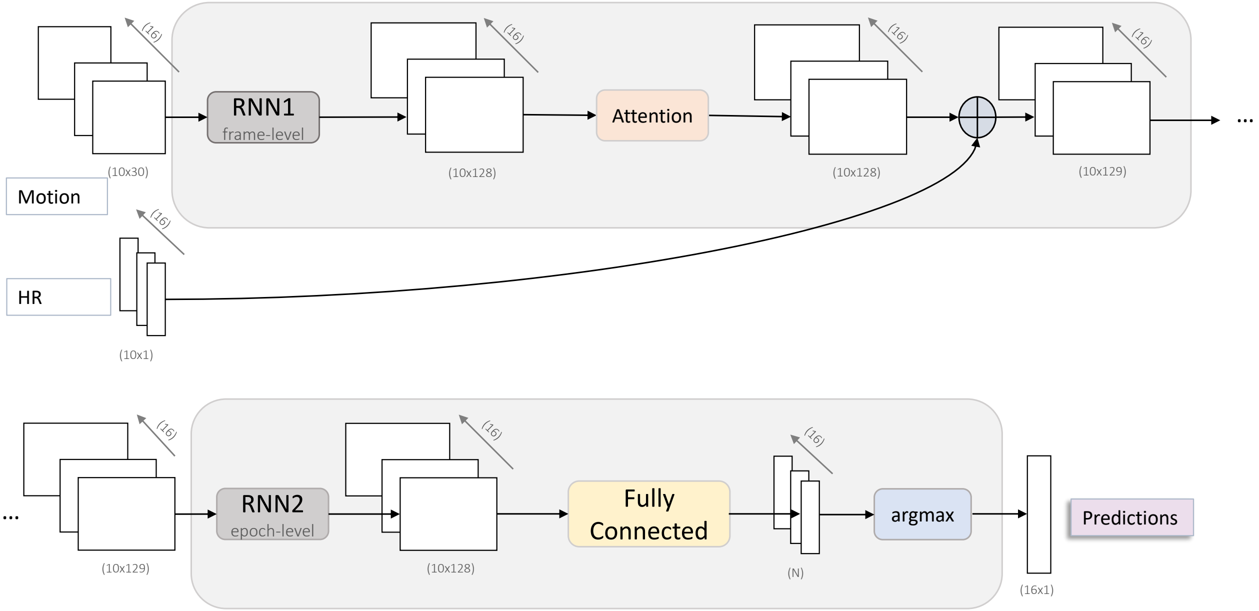


Figure 4.8.1: The proposed first modification for MESA application on SeqSleepNet architecture, where a sleep stage value is given per second, and the prediction is made for a sequence of X seconds. **N** refers to the number of classes in each experiment (all: N=5, light-deep: N=4, rem-nrem: N=3, sleep-wake: N=2).

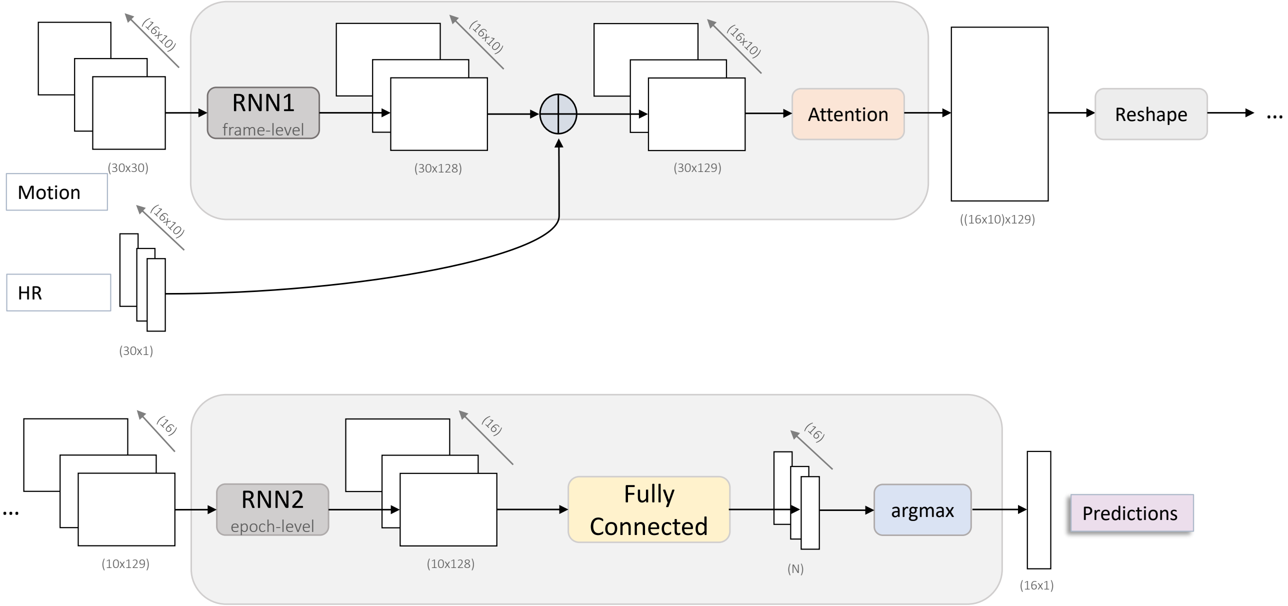


Figure 4.8.2: The proposed second and more complex modification for MESA application on SeqSleepNet architecture, where each sleep stage is considered for a 30-second epoch, and the prediction is made for a 10-sample sequence. **N** refers to the number of classes in each experiment (all: N=5, light-deep: N=4, rem-nrem: N=3, sleep-wake: N=2).

to represent the whole sequence of N consecutive epochs. Thus, the output of the first bidirectional-RNN layer containing intermediate representations of the motion feature is concatenated with the corresponding heart rate data at the epoch-sequence level, in order to be passed through the attention layer together. Consequently, the most important features at the epoch-sequence level are attended for the next neural layer.

- Afterwards, the data are passed through a second bidirectional-RNN layer giving the probability scores for each sleep stage of the sequence, followed by a linear layer and an activation function to get the final prediction.

The total architecture of the second modification can be seen in Figure 4.8.2.

4.8.3 Experimental Results - MESA

For training the SeqSleepNet models the whole MESA dataset was utilized, excluding only 5 subjects to be used for testing purposes. The data were split the same way for the train-test-evaluation subsets each time for consistency. Since the training of the models is quite time consuming and requires a lot of resources, taking more than 24 hours for each training session, no cross-validation was applied, although this would be the most scientifically correct approach. In Figure 4.8.5 and Figure 4.8.6 are depicted the hypnograms of the five excluded subjects, tested on the Modification 2 of the SeqSleepNet model, trained on the MESA dataset.

MESA statistical analysis

The statistical analysis of the raw MESA activity and heart rate features can be seen in Figure 4.8.4 and Figure 4.8.3. The mean value and standard deviation are extracted per sleep-stage category for every subject, for the activity and heart rate features separately, and the corresponding violin plots and scatter plots are presented.

Regarding the **heart rate** feature, in the violin plots it can be seen that, except of *wake* and *N4* stages which are a little bit lower, the mean values for the rest of the subjects have a median very close to 60 for all of the *N1, N2, N3, REM* sleep stages. This could indicate that they are not so easily distinguishable from one another. Also, they follow a similar normal distribution without many outliers, meaning that there is a high chance of heart rate value being close to the median for *N1, N2, N3*, and *REM*. However, *wake* sleep stage follows an almost bimodal distribution, which also elongates a lot on the vertical axis, exceeding the Tukey's fences, meaning that there are a lot of outliers on its mean value per subject. For stage *N4*, the mean-value per-subject distribution is normal and narrow, but many outliers seem to be present and the data are not being collected around the median value.

By the observation of the violin plots depicting the per-subject standard deviation of heart rate, it can be seen that the standard deviation values have a more even normal distribution between the sleep stage categories, meaning that inter-subject values for each sleep stage do not deviate much. Some outliers are present in sleep stages *N1, N2, N3*, and *REM*. The most distinct of all sleep stages is *wake*, where the distribution is very narrow and expands a lot on the vertical axis, following an almost bimodal kernel density estimation. Hence, for *wake* samples, there is a great divergence between the data distribution per subject.

Concerning the **activity** feature, it can be seen that the mean value for sleep stages *N1, N2, N3*, and *REM* follows an almost unit distribution, where all the samples are collected very close to a single point, which is the median, and there are almost no outliers. On the contrary, *wake* and *N4* span across a large range of values. The fact that the mean value per subject of the sleep stages *N1, N2, N3, REM* is converging towards the median is quite reasonable, since the activity feature provided in the MESA dataset is already preprocessed, and the raw data are not publicly available.

The standard deviation for activity feature is also a Gaussian distribution for all sleep stages except *N4*, which seem to have all its values equal to zero. There is a small diversity between *N1, N2, N3, REM* with all of the stages having their standard deviation values per subject very close to zero, showcasing that the activity feature is very uniform in each subject. However, again the *wake* stage appears to have a lot of outliers, although following a Gaussian distribution as well. Concerning stage *N4*, after finely investigating the MESA dataset, we found that for many subjects the *N4* stage is not present in their recordings, meaning a NaN mean value and a zero standard deviation. This is not a burden for our experiments, since *N4* is always merged with another sleep stage, depending on the classification task.

The scatter plot in Figure 4.8.3 verify the above observations, with all the mean values of *N1, N2, N3* and *REM* sleep stages lying very close to each other, while *wake* ranges in a big variety of values and *N4* is almost absent from the dataset.

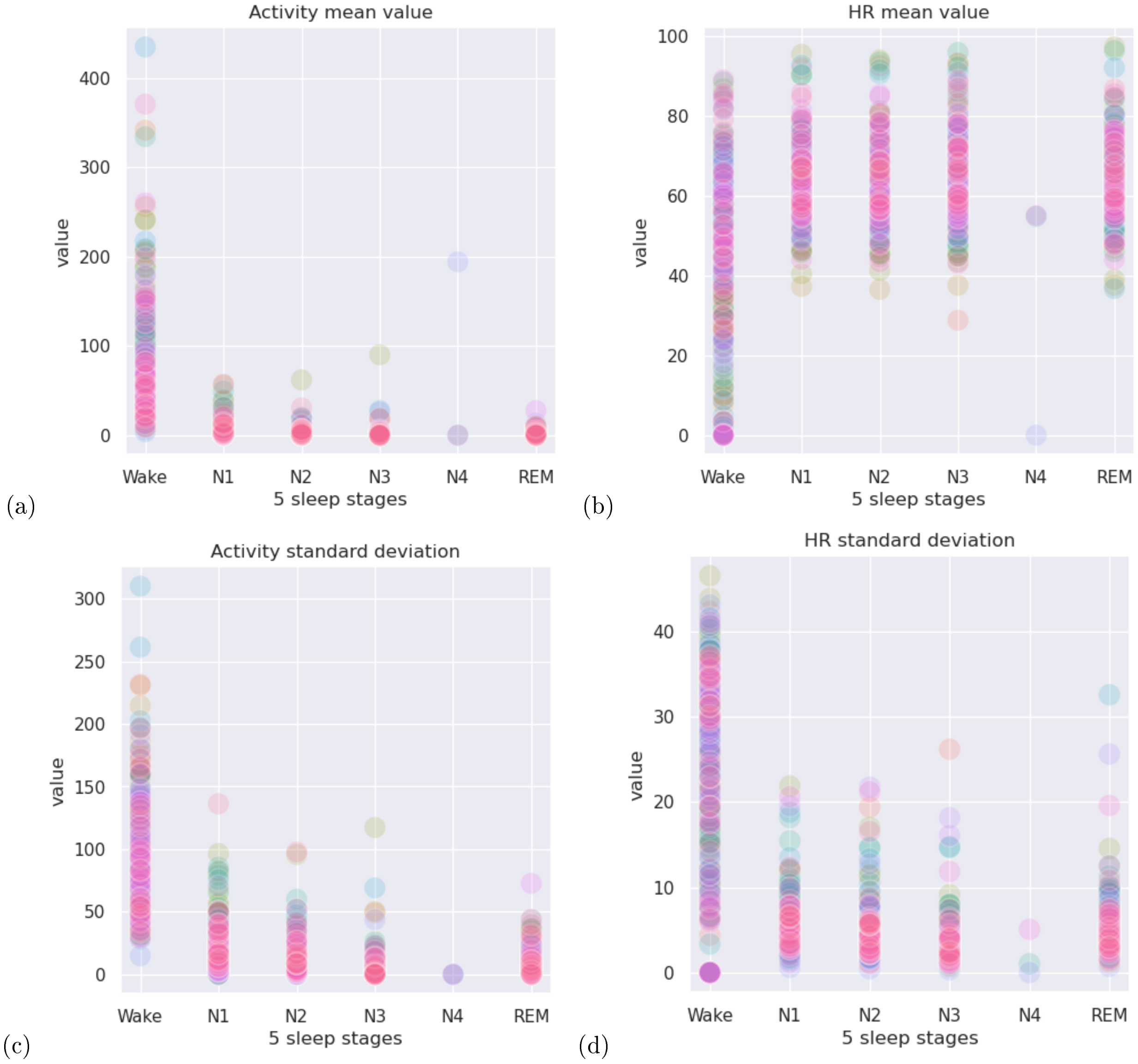
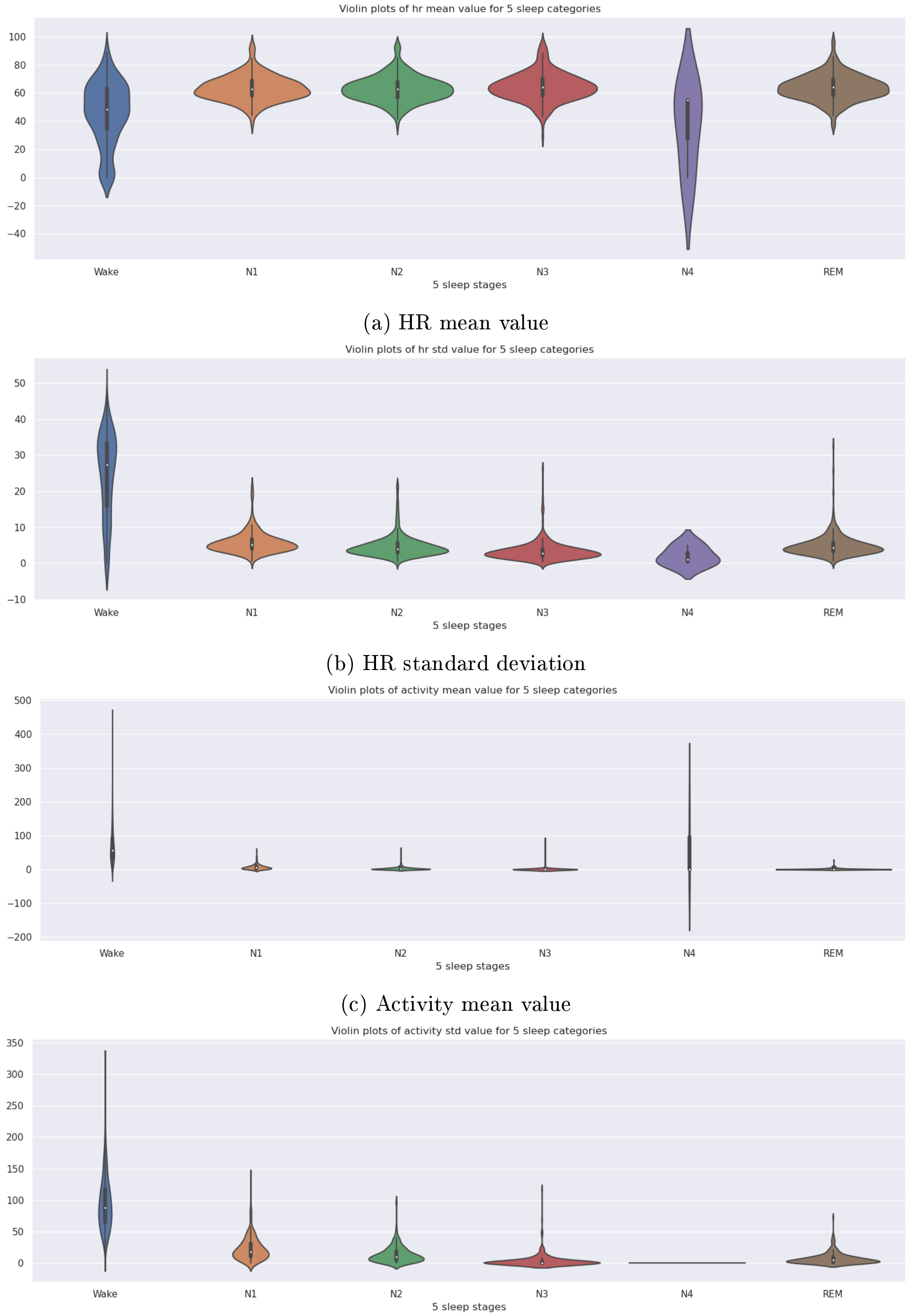


Figure 4.8.3: Scatter plot of MESA statistical features per sleep stage category: a. Activity mean value, b. Heart rate mean value, c. Activity standard deviation, d. Heart rate standard deviation. The data points correspond to every subject of the dataset, and the statistical features are extracted for every subject, for every sleep-stage category.



(a) HR mean value

(b) HR standard deviation

(c) Activity mean value

(d) Activity standard deviation

Figure 4.8.4: Violin plots of the distribution for each sleep stage, for the MESA dataset. The mean value for each sleep stage category for each subject is taken, in order to showcase the distribution between the subjects of the dataset.

Experimental results and discussions

Table 4.14 presents the classification reports of all sleep stage categories from the best model of each, for both modifications. In Table 4.15 are presented the accuracy and Cohen's kappa values of each sleep stage category for both modifications.

Cohen's Kappa measures the degree of agreement between two evaluators, hence the inter-rater variability. Provided raters 1 and 2, let:

- A: The total number of instances that both raters said were correct. The Raters are in agreement.
- B: The total number of instances that Rater 2 said was incorrect, but Rater 1 said were correct. This is a disagreement.
- C: The total number of instances that Rater 1 said was incorrect, but Rater 2 said were correct. This is also a disagreement.
- D: The total number of instances that both Raters said were incorrect. Raters are in agreement.

Then, the following probabilities can be defined:

$$\begin{aligned} P_o &= \frac{A+D}{A+B+C+D} \\ P_{(correct)} &= \frac{A+B}{A+B+C+D} \times \frac{A+C}{A+B+C+D} \\ P_{(incorrect)} &= \frac{C+D}{A+B+C+D} \times \frac{B+D}{A+B+C+D} \\ P_e &= P_{(correct)} + P_{(incorrect)} \end{aligned} \tag{4.8.1}$$

where P_o is the probability of agreement and P_e is the probability of random agreement.

Cohen's kappa is then formed as:

$$K = \frac{P_o - P_e}{1 - P_e} \tag{4.8.2}$$

Both of the modifications on the standard SeqSleepNet model give similar results. There is an increasing performance on both of the models correlated to the complexity of the problem to be handled, hence the less sleep stages to be classified the better the performance of the model is. It is encouraging that the more complex architecture of the second modification results in predictions very close to the first modification. It shows that, taking into account the extra parameter of the 30-second epoch sequences per every single sleep stage label, and an N-sampled sequence of 30-epoch data-points to predict the final sleep stage, can still sufficiently capture the internal patterns of the data. The appearance of zero-valued predictions for some of the sleep stages on the classification report for both modifications is correlated to how strong is their presence in the dataset. Specifically, this occurs on stages *N1* and *N3*, where their number of samples differ by an order of magnitude from the most prominent classes. Additionally, the second modification seem to misclassify more sleep stages than the first modification: the macro-average of the F1-score is lower, and also more zero-valued predictions for stages appear (stage N3 has no predictions for both models in the 5-class problem, while stage N1 is missing predictions only for the second modification).

Cohen's Kappa can be used to measure the correctness of predictions a classification model has made. Specifically, instead ot comparing the ratings between two evaluators, Cohen's kappa metric can be used to show the agreement or random choice between the predictions and the true class labels of a classification task. The kappa value being closer to 1 indicates that the model makes more accurate predictions, while a lower value of kappa, closer to 0, indicates that the correctly predicted samples tend to be so by chance, and the model is not properly trained. It can be seen that Cohen's kappa is quite low for the more complex problems of five and four sleep stages, while it is somewhere around the middle of 0.5 for the two and three class problem. This observation can also be confirmed by the F1-scores of the classification report. It aligns correctly with the data distribution discussed in the previous paragraphs, where some of the sleep stages

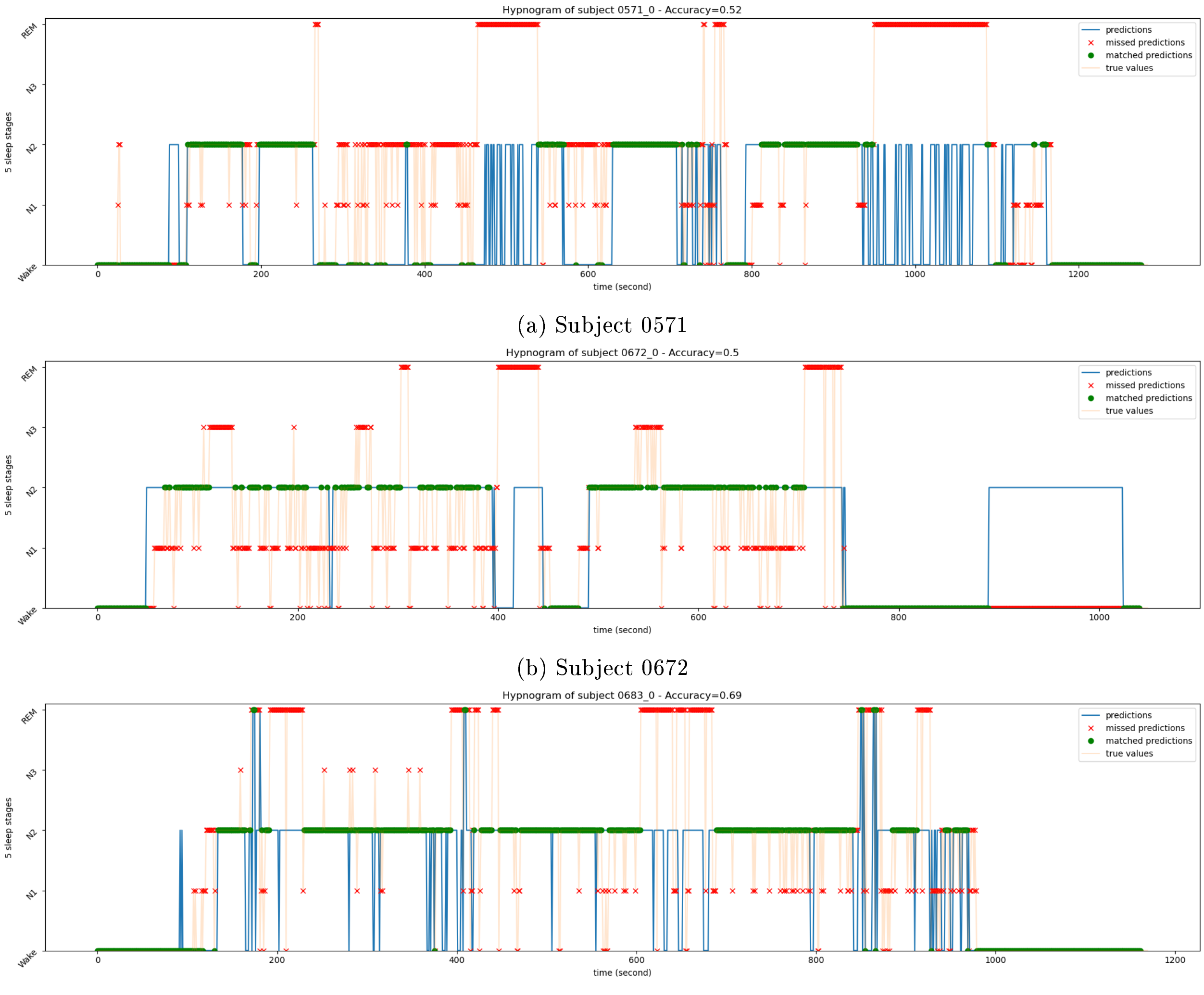
Table 4.14: The classification report of each sleep-stage category for both modifications done on the original model to apply it on MESA. The best models during training are saved and chosen for testing purposes.

	Modification1				**Modification 2**			
ALL	**precision**	**recall**	**f1-score**	**support**	**precision**	**recall**	**f1-score**	**support**
wake	0.73	0.72	0.73	28573	0.67	0.78	0.72	27124
n1	0.11	0.02	0.03	4911	0	0	0	4861
n2	0.51	0.79	0.62	23593	0.52	0.78	0.62	23388
n3	0	0	0	4260	0	0	0	4232
rem	0.47	0.15	0.23	7015	0.55	0.02	0.03	6987
accuracy			**0.59**				**0.59**	
macro avg	0.36	0.34	**0.32**	68352	0.35	0.31	**0.27**	66592
weighted avg	0.54	0.59	0.54	68352	0.51	0.59	0.52	66592
Light-Deep	**precision**	**recall**	**f1-score**	**support**	**precision**	**recall**	**f1-score**	**support**
wake	0.82	0.66	0.73	28573	0.79	0.65	0.71	27124
light	0.56	0.88	0.68	28504	0.56	0.89	0.69	28249
deep	0	0	0	4260	0	0	0	4232
rem	0	0	0	7015	0	0	0	6987
accuracy			**0.64**				**0.64**	
macro avg	0.34	0.39	**0.35**	68352	0.34	0.38	**0.35**	66592
weighted avg	0.57	0.64	0.59	68352	0.56	0.64	0.58	66592
REM-NREM	**precision**	**recall**	**f1-score**	**support**	**precision**	**recall**	**f1-score**	**support**
wake	0.82	0.63	0.71	28573	0.78	0.66	0.72	27124
nrem	0.63	0.9	0.74	32764	0.66	0.88	0.75	32481
rem	0	0	0	7015	0	0	0	6987
accuracy			**0.69**				**0.7**	
macro avg	0.48	0.51	**0.49**	68352	0.48	0.52	**0.49**	66592
weighted avg	0.65	0.69	0.65	68352	0.64	0.7	0.66	66592
Sleep-Wake	**precision**	**recall**	**f1-score**	**support**	**precision**	**recall**	**f1-score**	**support**
wake	0.85	0.63	0.73	28573	0.82	0.63	0.71	27124
sleep	0.78	0.92	0.84	39779	0.78	0.9	0.84	39468
accuracy			**0.8**				**0.79**	
macro avg	0.81	0.78	**0.78**	68352	0.8	0.77	**0.77**	66592
weighted avg	0.81	0.8	0.79	68352	0.8	0.79	0.79	66592

are quite under-represented, due to the natural cycle of human sleep. This characteristic of the dataset highlights an imbalance in its distribution between the classes. Given that an adequate number of samples is available, this property of the MESA dataset could be overcome, but, as it can be seen, it still greatly affects the proposed models. Thus, more samples of the under-represented sleep stage classes might be needed, or different architectures and ways of handling the data could be explored to surpass this occurrence.

Table 4.15: The accuracy and Cohen's kappa for each sleep-stage category extracted on the test subset of MESA, for both modifications. The best models during training are saved and chosen for testing purposes.

	Modification 1		**Modification 2**	
	Accuracy	**Kappa**	**Accuracy**	**Kappa**
All	0.58	0.33	0.59	0.34
Light-Deep	0.63	0.37	0.64	0.38
REM-NREM	0.69	0.42	0.7	0.45
Sleep-Wake	0.78	0.54	0.8	0.56

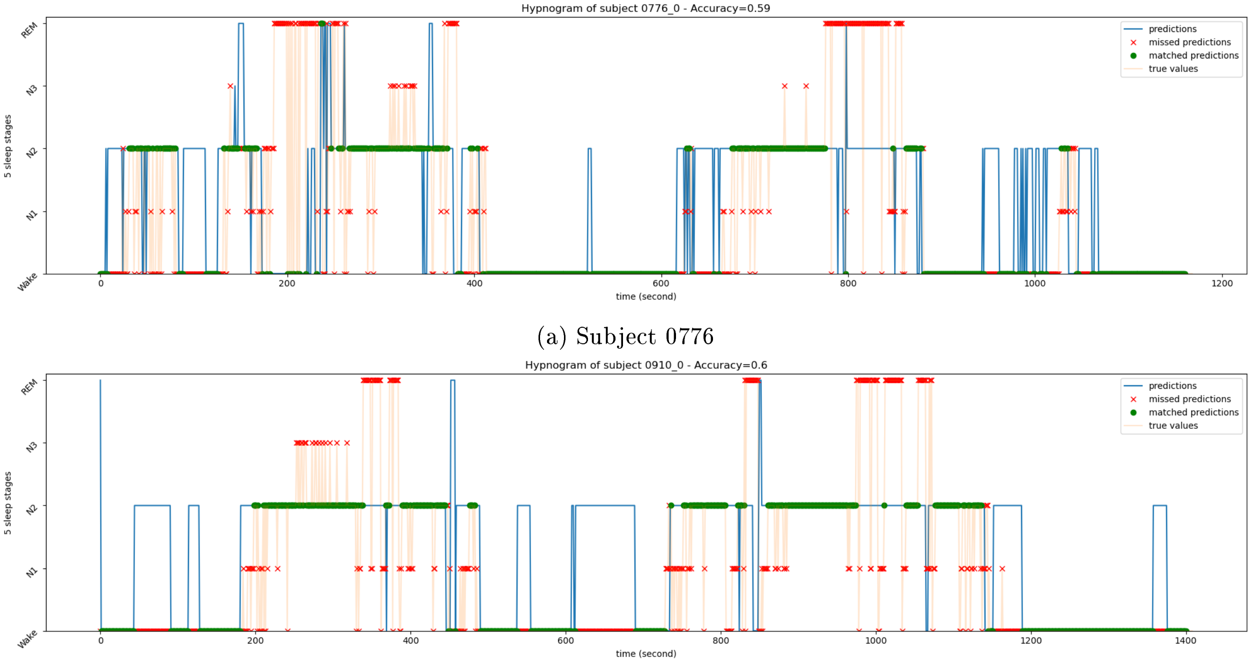


(a) Subject 0571

(b) Subject 0672

(c) Subject 0683

Figure 4.8.5: Hypnograms of five test subjects from the MESA dataset, excluded from the training and evaluation subsets. The hypnograms show the predictions for the first three subjects tested on the Modification 2 model of the SeqSleepNet architecture, trained on the MESA dataset, for the five sleep stages: Wake, N1, N2, N3, REM.



(a) Subject 0776

(b) Subject 0910

Figure 4.8.6: Hypnograms of five test subjects from the MESA dataset, excluded from the training and evaluation subsets. The hypnograms show the predictions for the last two subjects tested on the Modification 2 model of the SeqSleepNet architecture, trained on the MESA dataset, for the five sleep stages: Wake, N1, N2, N3, REM.

Chapter 5

Conclusion

5.1 Contributions, Conclusions and Future Work

5.1.1 Contributions

This Thesis has covered a thorough study on the task of sleep stage classification using data derived from wearable devices and employing neural network architectures for this purpose. Two datasets are incorporated for the experiments of this work. The first one is collected from a smartwatch (Apple Watch) [Wal19] and consists of 3D motion from wrist accelerometer and heart rate from pulse oximetry. The second dataset [Nat16; Zha+18a; Che+15] consists of motion signals derived from a wrist-worn actigraphy and heart rate derived from polysomnography. Both of them have sleep stage labels, manually aligned by an expert during a polysomnography night's sleep study. The main differentiation between the two datasets is, while the first one has much less subjects, the second dataset is not completely derived from wearable devices, since the heart rate feature is collected from the polysomnography recordings. The goal of this work is to study the sleep stage classification problem utilizing datasets derived from wearable devices with deep learning architectures. Specifically, the two datasets are tested under different neural architectures and it is investigated, how they correspond to the task of sleep stage classification, given their primary differences.

In the initial experiments, the two datasets undergo the feature extraction method firstly presented in [Wal+19], in order to obtain epoch-level features of heart rate and activity count, where each epoch is a 30-second segment with a corresponding sleep label. The extracted features represent the same physical values for both datasets, hence they are directly comparable. The aforementioned work, which firstly introduced the manually extracted features, applies traditional machine learning techniques on them for the task of sleep stage classification, using every sample separately for training. The aim of the experiments proposed in the current Thesis, is to take advantage of the temporal nature of the feature sequences per subject, and test some standard neural network architectures on them, to examine their performance compared to the classification techniques, which do not take this aspect into account. The first model to be implemented is a bidirectional LSTM network. The choice for this kind of recurrent neural network is made based on other works utilizing this architecture [Zha+18b; Zha+19], and also the bidirectional attribute is adopted, since it is more intuitive and better suits the type of temporal sequential problem. Under this perspective, at first, the smartwatch-derived dataset is tested on the proposed bidirectional LSTM model, giving very promising results and overriding the results given by the classical machine learning approaches. The already trained model on the first dataset is also tested on unseen data from the second dataset, performing poorly, which shows that the model cannot generalize to data unseen during training that are sourced in a different way. Then the same model is tested on the features derived by the second dataset. Although the results are relatively good, they are lower than the model trained on the first dataset, which indicates that the extracted features better suit the first dataset, combined with a bidirectional LSTM architecture.

As a second group of experiments, an automated feature extraction method is tested in parallel to a similar bidirectional LSTM architecture. Specifically, a CNN architecture is utilized to automatically extract features from the raw data sequences, which are then given to the bidirectional LSTM module to solve the sleep stage classification task. The use of CNNs for automated feature extraction has been previously tested on PSG-extracted data, as seen in [Che+20]. Training the CNN - bidirectional LSTM model end-to-end with the smartwatch derived dataset, leads to inferior results compared to the manually extracted features with the simple LSTM architecture. On the contrary, training the model with the raw data originated from the second dataset, leads to quite accurate predictions on the test set. The sleep stage classification capabilities of the CNN-bidirectional LSTM architecture on the second dataset are comparable to the performance of the simple bidirectional LSTM model on the manually extracted features of the first dataset. This observation leads to the conclusion that the performance of the model greatly depends on the nature of the data. Hence, in our case, the first dataset is relatively smaller, but an appropriate manual feature extraction method with a shallow network performs better, while for the second dataset an automated feature extraction module combined with the basic model leads to similar results.

As a final set of experiments, a deeper architecture is tested, which was initially proposed in [Pha+19] utilizing signals derived from a PSG study. The aforementioned architecture is a hierarchical recurrent neural network with an attention mechanism named SeqSleepNet, and handles the problem of sleep stage classification as a sequence-to-sequence task. The aim of the final experiments is to test a deeper architecture, initially designed for the more complex signals of a PSG study, on wearable derived data. The two datasets differ in their raw format in terms of sampling rate and their monitoring method, thus they are handled separately. For the first dataset, the type of raw features allow for spectrogram extraction, which is the format the original SeqSleepNet takes as input. However, the experiments give poor results, indicating that this kind of deep architecture is incompatible with the specific dataset. For the second dataset, due to the format the heart rate feature is provided, the model is altered so that it does not receive spectrogram images as input, but temporal sequential features instead. Two approaches are adopted in this case: in the first one the sequence is on second-level, meaning that there is one sleep label for every second. The second case is on epoch level, which means that one sleep label corresponds to a 30-second window (epoch), and the sequence consists of consecutive epochs, adding one more dimension to the input data. The experimental results indicate that the altered SeqSleepNet network performs relatively good for the problem with less classes to be predicted (e.g. sleep-wake), but the more complex the prediction task gets (e.g. for the five sleep stages), the model capabilities get weaker.

5.1.2 Conclusions

Given the above experiments, it must be stated that, regarding the bidirectional-LSTM models, both cases give better results than previous works on data derived from wearable devices, and especially when classical machine learning techniques are used for the task of sleep stage classification. Provided the previous work employing both datasets in [Wal+19], where the manual feature extraction method is applied in combination with typical machine learning applications for the classification task, we show that deeper neural network architectures, which take into account the temporal nature of the data sequences, lead to an improved performance. Additionally, the utilization of manual or automatic feature extraction greatly depends on the specific dataset. Finally, the choice of a shallower architecture with carefully extracted features can perform better or equally to a deeper architecture, with less computational needs. As seen from the results of the SeqSleepNet in comparison to the bidirectional-LSTM models, a deeper and more complex architecture does not guarantee better performance for the network.

5.2 Future Work

Towards this direction, other architectures, suitable for sequential data, such as Transformers [Vas+17], could be adapted on the problem of sleep stage classification. This could be an interesting examination of whether other deep applications perform better with any of the two datasets, as well the extend to which the size of the dataset affects the training of the model. Additionally, different combinations of features from different domains can be tested, compared to the features utilized in this work, in order to examine whether the bidirectional-LSTM performance can be improved. Regarding the two different datasets, and the difficulty of the bidirectional-LSTM model to generalize on unseen data from a separate source, a fine-tuning method could be tested. By firstly training the model on one dataset, fine-tuning could be then applied for a few iterations on the second one, to explore the model's capabilities. This application could be quite useful in cases where not enough data are available for fully training a model, but, given that the same kind of features can be extracted for two different datasets, the model trained on the largest dataset can be then partially trained on the fewer samples of the second dataset. Finally, all the experiments presented in this work are subject-agnostic, meaning that there is no information for the subject that the training samples belong to. Considering that each individual have their unique physical mechanisms and homeostatic patterns, a personalized model could be a next step towards optimizing the automatic sleep stage classification task. By adding a user embedding module on a model, more personalized information could be learned, thus, during the classification process, more accurate details might be retrieved, leading to more precise predictions.

Appendix A

Βιβλιογραφία

[01] 01. *Normal Human Sleep: An Overview. Principles and Practice of Sleep Medicine.* URL: https://www.researchgate.net/publication/287231408.

[02] 02. *Understanding the Bias-Variance Tradeoff.* URL: http://scott.fortmann-roe.com/docs/BiasVariance.html.

[03] 03. *Overfitting and underfitting.* URL: https://www.educative.io/edpresso/overfitting-and-underfitting.

[04] 04. URL: https://www.analyticsvidhya.com/blog/2021/05/5-classification-algorithms-you-should-know-introductory-guide/.

[05] 05. URL: https://en.wikipedia.org/wiki/Support_vector_machine.

[06] 06. URL: https://towardsdatascience.com/what-the-hell-is-perceptron-626217814f53.

[07] 07. *Implementation of multilayer perceptron (MLP) and radial basis function (RBF) neural networks to predict solution gas-oil ratio of crude oil systems.* URL: https://www.sciencedirect.com/science/article/pii/S2405656118301020.

[08] 08. URL: https://www.researchgate.net/figure/Common-activation-functions-in-artificial-neural-networks-NNs-that-introduce_fig7_341310767.

[09] 09. URL: https://www.researchgate.net/figure/CNN-architecture-7_fig1_333168248.

[10] 10. *Convolutional Networks.* URL: https://www.deeplearningbook.org/contents/convnets.html.

[11] 11. *Deep convolution neural network for image recognition.* URL: https://hal.archives-ouvertes.fr/hal-02053205/document.

[12] 12. *CS231n Convolutional Neural Networks for Visual Recognition.* URL: https://cs231n.github.io/convolutional-networks/.

[13] 13. *What are recurrent neural networks?* URL: https://www.ibm.com/cloud/learn/recurrent-neural-networks.

[14] 14. *Bidirectional internal memory gate recurrent neural networks for spoken language understanding.* URL: https://link.springer.com/article/10.1007/s10772-020-09708-9.

[15] 15. *Recurrent Neural Networks cheatsheet.* URL: https://stanford.edu/ shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks.

[16] 16. *Understanding LSTM Networks.* URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[AC+06] Altevogt, B. M., Colten, H. R., et al. *Sleep Disorders and Sleep Deprivation: An Unmet Public Health Problem.* National Academies Press, 2006. DOI: 10.17226/11617. URL:

[And+11] Andrillon, T. et al. “Sleep Spindles in Humans: Insights from Intracranial EEG and Unit Recordings”. In: *Journal of Neuroscience* 31.49 (2011). URL: https://www.jneurosci.org/content/31/49/17821, pp. 17821–17834. ISSN: 0270-6474. DOI: 10.1523/JNEUROSCI.2604-11.2011.

[Ari11] Aristotle. *Nicomachean Ethics.* Vol. Book 2. Chase translation, 1911.

[AK53] Aserinsky, E. and Kleitman, N. “Regularly Occurring Periods of Eye Motility, and Concomitant Phenomena, During Sleep”. In: *Science* 118.3062 (1953). URL:

https://www.science.org/doi/abs/10.1126/science.118.3062.273, pp. 273–274. DOI: 10.1126/science.118.3062.273.

[BCB14] Bahdanau, D., Cho, K., and Bengio, Y. "Neural Machine Translation by Jointly Learning to Align and Translate". In: *arXiv preprint arXiv:1409.0473* (2014). DOI: https://doi.org/10.48550/arXiv.1409.0473.

[Bea+17] Beattie, Z. et al. "Estimation of Sleep Stages in a Healthy Adult Population from Optical Plethysmography and Accelerometer Signals". In: *Physiological Measurement* 38.11 (2017), p. 1968.

[Ber29] Berger, H. "Über das Elektrenkephalogramm des Menschen". In: *Archiv für Psychiatrie und Nervenkrankheiten* 87 (1929). URL: https://link.springer.com/article/10.1007/BF01797193, pp. 527–570.

[Ber01] Berlinski, D. *The Advent of the Algorithm : The 300-Year Journey from an Idea to the Computer.* San Diego: Harcourt, 2001. ISBN: 978-0-15-601391-8.

[BM16] Besharse, J. C. and McMahon, D. G. "The Retina and Other Light-sensitive Ocular Clocks". In: *Journal of Biological Rhythms* 31.3 (2016). URL: https://doi.org/10.1177/0748730416642657, pp. 223–243. DOI: 10.1177/0748730416642657.

[Bil+02] Bild, D. E. et al. "Multi-Ethnic Study of Atherosclerosis: Objectives and Design". In: *American Journal of Epidemiology* 156.9 (2002), pp. 871–881.

[BKN17] Boostani, R., Karimzadeh, F., and Nami, M. "A Comparative Review on Sleep Stage Classification Methods in Patients and Healthy Individuals". In: *Computer Methods and Programs in Biomedicine* 140 (2017), pp. 77–91.

[Bün35] Bünning, E. "Zur Kenntnis der erblichen Tagesperiodiztät bei den Primarblattern von Phaseous Multiflorus". In: *Jahrb Wiss. Botan.* 81 (1935), pp. 411–418.

[Bün64] Bünning, E. "Endodiurnal Oscillations as the Principle of Many Physiological Time Measuring Processes". In: *The Physiological Clock.* Springer, 1964, pp. 4–19.

[Cai+09] Cai, D. J. et al. "REM, not Incubation, Improves Creativity by Priming Associative Networks". In: *Proceedings of the National Academy of Sciences* 106.25 (2009). URL: https://www.pnas.org/content/106/25/10130, pp. 10130–10134. ISSN: 0027-8424. DOI: 10.1073/pnas.0900271106.

[CD79] Carskadon, M. A. and Dement, W. C. "Effects of Total Sleep Loss on Sleep Tendency". In: *Perceptual and Motor Skills* 48.2 (1979). PMID: 461051. URL: https://doi.org/10.2466/pms.1979.48.2.495, pp. 495–506. DOI: 10.2466/pms.1979.48.2.495.

[CD+05] Carskadon, M. A., Dement, W. C., et al. "Normal Human Sleep: An Overview". In: *Principles and Practice of Sleep Medicine* 4.1 (2005), pp. 13–23.

[Cat75] Caton, R. "The Electric Currents of the Brain". In: *The British Medical Journal* (1875). URL: https://digilib.mpiwg-berlin.mpg.de/digitallibrary/jquery/digilib.html?fn=/permanent/vlp/lit27690/pages, p. 278.

[Che+15] Chen, X. et al. "Racial/Ethnic Differences in Sleep Disturbances: The Multi-Ethnic Study of Atherosclerosis (MESA)". In: *Sleep* (2015). URL: https://doi.org/10.5665/sleep.4732. DOI: 10.5665/sleep.4732.

[Che+20] Chen, X. et al. "Sleep Staging by Bidirectional Long Short-Term Memory Convolution Neural Network". In: *Future Generation Computer Systems* 109 (2020), pp. 188–196.

[Cho+15] Chollet, F. et al. *Keras.* URL: https://github.com/fchollet/keras. 2015.

[Chu+14] Chung, J. et al. "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling". In: *arXiv preprint arXiv:1412.3555* (2014). DOI: https://doi.org/10.48550/arXiv.1412.3555.

[DGD11] De La Herrán-Arita, A., Guerra-Crespo, M., and Drucker-Colin, R. "Narcolepsy and Orexins: An Example of Progress in Sleep Research". In: *Frontiers in Neurology* 2 (2011). URL: https://www.frontiersin.org/article/10.3389/fneur.2011.00026, p. 26. ISSN: 1664-2295. DOI: 10.3389/fneur.2011.00026.

[deM29] deMairan, J.-J. "Observation Botanique". In: *Histoire de l'Académie Royale des Sciences* (1729). URL: http://www.bibnum.education.fr/sciencesdelavie/biologie/observation-botanique, p. 35.

[DK57] Dement, W. and Kleitman, N. "Cyclic Variations in EEG During Sleep and their Relation to Eye Movements, Body Motility, and Dreaming". In: *Electroencephalography and Clinical Neurophysiology* 9.4 (1957). URL: https://www.sciencedirect.com/science/article/pii/0013469457900883, pp. 673–690. ISSN: 0013-4694. DOI: https://doi.org/10.1016/0013-4694(57)90088-3.

[Dem98] Dement, W. C. “The Study of Human Sleep: A Historical Perspective”. In: *Thorax* 53.suppl 3 (1998), pp. 2–7.

[Dem05] Dement, W. C. “History of Sleep Medicine”. In: *Neurologic Clinics* 23.4 (2005). Sleep Disorders. URL: https://www.sciencedirect.com/science/article/pii/S0733861905000587, pp. 945–965. ISSN: 0733-8619. DOI: https://doi.org/10.1016/j.ncl.2005.07.001.

[DP05] Ding, C. and Peng, H. “Minimum Redundancy Feature Selection from Microarray Gene Expression Data”. In: *Journal of Bioinformatics and Computational Biology* 3.02 (2005), pp. 185–205.

[Don+17] Dong, H. et al. “Mixed Neural Network Approach for Temporal Sleep Stage Classification”. In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 26.2 (2017), pp. 324–333.

[DLD04] Dunlap, J. C., Loros, J. J., and DeCoursey, P. J. *Chronobiology: Biological Timekeeping.* Sinauer Associates, 2004.

[EPS06] Ellenbogen, J. M., Payne, J. D., and Stickgold, R. “The Role of Sleep in Declarative Memory Consolidation: Passive, Permissive, Active or None?” In: *Current Opinion in Neurobiology* 16.6 (Dec. 2006). URL: https://doi.org/10.1016/j.conb.2006.10.006, pp. 716–722. DOI: 10.1016/j.conb.2006.10.006.

[FD15] Feld, G. B. and Diekelmann, S. “Sleep Smart—Optimizing Sleep for Declarative Learning and Memory”. In: *Frontiers in Psychology* 6 (2015). URL: https://www.frontiersin.org/article/10.3389/fpsyg.2015.00622, p. 622. ISSN: 1664-1078. DOI: 10.3389/fpsyg.2015.00622.

[Fel11] Felicia, P. *Handbook of Research on Improving Learning and Motivation through Educational Games : Multidisciplinary Approaches.* Hershey PA: Information Science Reference, 2011. ISBN: 978-1609604967.

[FRP20] Fernandez-Blanco, E., Rivero, D., and Pazos, A. “Convolutional Neural Networks for Sleep Stage Scoring on a Two-Channel EEG Signal”. In: *Soft Computing* 24.6 (2020), pp. 4067–4079.

[Fer+19] Fernández-Varela, I. et al. “A Convolutional Network for Sleep Stages Classification”. In: *arXiv preprint arXiv:1902.05748* (2019). DOI: https://doi.org/10.48550/arXiv.1902.05748.

[Fer+17] Ferri, R. et al. “REM Sleep EEG Instability in REM Sleep Behavior Disorder and Clonazepam Effects”. In: *Sleep* 40.8 (2017). URL: https://doi.org/10.1093/sleep/zsx080. DOI: 10.1093/sleep/zsx080.

[FS17] Fleischer, R. and Snickars, P. “Discovering Spotify - A Thematic Introduction”. In: *Culture Unbound* 9.2 (2017), pp. 130–145.

[Fon+15] Fonseca, P. et al. “Sleep Stage Classification with ECG and Respiratory Effort”. In: *Physiological Measurement* 36.10 (2015), p. 2027.

[Fon+17] Fonseca, P. et al. “Validation of Photoplethysmography-Based Sleep Staging Compared with Polysomnography in Healthy Middle-Aged Adults”. In: *Sleep* 40.7 (2017).

[Fra20] Fradkov, A. L. “Early History of Machine Learning”. In: *IFAC-PapersOnLine* 53.2 (2020). URL: https://doi.org/10.1016/j.ifacol.2020.12.1888, pp. 1385–1390. DOI: 10.1016/j.ifacol.2020.12.1888.

[FS97] Freund, Y. and Schapire, R. E. “A Decision-Theoretic Generalization of on-line Learning and an Application to Boosting”. In: *Journal of Computer and System Sciences* 55.1 (1997), pp. 119–139.

[Gag+16] Gagliano, M. et al. “Learning by Association in Plants”. In: *Scientific Reports* 6.1 (2016). URL: https://doi.org/10.1038/srep38427. DOI: 10.1038/srep38427.

[Gai+02] Gais, S. et al. “Learning-Dependent Increases in Sleep Spindle Density”. In: *Journal of Neuroscience* 22.15 (2002), pp. 6830–6834.

[GTD66] Gastaut, H., Tassinari, C., and Duron, B. “Polygraphic Study of the Episodic Diurnal and Nocturnal (Hypnic and Respiratory) Manifestations of the Pickwick Syndrome”. In: *Brain Research* 1.2 (1966). URL: https://www.sciencedirect.com/science/article/pii/000689936690117X, pp. 167–186. DOI: https://doi.org/10.1016/0006-8993(66)90117-X.

[GA05] Gillette, M. and Abbott, S. “Fundamentals of the circadian system”. In: *Sleep Research Society, Illinois* (2005), pp. 131–138.

[Glo+20] Glosemeyer, R. W. et al. “Selective Suppression of Rapid Eye Movement Sleep Increases Next-Day Negative Affect and Amygdala Responses to Social Exclusion”. In: *Scientific Reports* 10.1

(2020). URL: https://doi.org/10.1038/s41598-020-74169-8, p. 17325. DOI: 10.1038/s41598-020-74169-8.

[Gol+00] Goldberger, A. L. et al. "PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals". In: *Circulation* 101.23 (2000), e215–e220.

[GBC16] Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. URL: http://www.deeplearningbook.org. MIT Press, 2016.

[GD95] Gordon, D. F. and Desjardins, M. "Evaluation and Selection of Biases in Machine Learning". In: *Machine Learning* 20.1 (1995), pp. 5–22.

[Gro10] Gross, R. *Psychology : The Science of Mind and Behaviour*. London: Hodder Education, 2010. ISBN: 978-1444108316.

[GTD76] Guilleminault, C., Tilkian, A., and Dement, W. C. "The Sleep Apnea Syndromes". In: *Annual Review of Medicine* 27 (1976), pp. 465–484.

[Har15] Harari, Y. *Sapiens : A Brief History of Humankind*. New York: Harper, 2015. ISBN: 978-0062316097.

[HB17] Hassan, A. R. and Bhuiyan, M. I. H. "An Automated Method for Sleep Staging from EEG Signals using Normal Inverse Gaussian Parameters and Adaptive Boosting". In: *Neurocomputing* 219 (2017), pp. 76–87.

[Has+09] Hastie, T. et al. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Vol. 2. Springer, 2009.

[Heu+17] Heusel, M. et al. "Gans Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium". In: *Advances in Neural Information Processing Systems* 30 (2017).

[HS97] Hochreiter, S. and Schmidhuber, J. "Long Short-Term Memory". In: *Neural Computation* 9.8 (1997), pp. 1735–1780.

[Hsu+13] Hsu, Y.-L. et al. "Automatic Sleep Stage Recurrent Neural Classifier using Energy Features of EEG Signals". In: *Neurocomputing* 104 (2013), pp. 105–114.

[Ibe07] Iber, C. "The AASM Manual for the Scoring of Sleep and Associated Events: Rules". In: *Terminology and Technical Specification* (2007).

[Jea+01] Jean-Louis, G. et al. "Sleep Estimation from Wrist Movement Quantified by Different Actigraphic Modalities". In: *Journal of Neuroscience Methods* 105.2 (2001), pp. 185–191.

[JMC59] Jouvet, M., Michel, F., and Courjon, J. "On a Stage of Rapid Cerebral Electrical Activity in the Course of Physiological Sleep". In: *Comptes rendus des seances de la Societe de biologie et de ses filiales* 153 (1959), pp. 1024–1028.

[JK65] Jung, R. and Kuhlo, W. "Neurophysiological Studies of Abnormal Night Sleep and the Pickwickian Syndrome". In: *Progress in Brain Research* 18 (1965). URL: https://www.sciencedirect.com/science/article/pii/S0079612308635906, pp. 140–159. DOI: https://doi.org/10.1016/S0079-6123(08)63590-6.

[Kes+16] Keskar, N. S. et al. "On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima". In: *arXiv preprint arXiv:1609.04836* (2016). DOI: https://doi.org/10.48550/arXiv.1609.04836.

[Kid+18] Kidziński, Ł. et al. "Learning to Run Challenge Solutions: Adapting Reinforcement Learning Methods for Neuromusculoskeletal Environments". In: *The NIPS '17 Competition: Building Intelligent Systems*. URL: https://doi.org/10.1007/978-3-319-94042-7_7. Springer International Publishing, 2018, pp. 121–153. DOI: 10.1007/978-3-319-94042-7_7.

[KA15] Kingma, D. P. and Adam, J. B. "Adam: A Method for Stochastic Optimization". In: *Optimization. In, ICLR* 5 (2015). DOI: https://doi.org/10.48550/arXiv.1412.6980.

[Kir11] Kirsch, D. B. "There and Back Again: A Current History of Sleep Medicine". In: *Chest* 139.4 (2011). URL: https://www.sciencedirect.com/science/article/pii/S0012369211601980, pp. 939–946. ISSN: 0012-3692. DOI: https://doi.org/10.1378/chest.10-1235.

[Klo+01] Klosh, G. et al. "The SIESTA Project Polygraphic and Clinical Database". In: *IEEE Engineering in Medicine and Biology Magazine* 20.3 (2001), pp. 51–57. DOI: 10.1109/51.932725.

[Kon94] Kononenko, I. "Estimating Attributes: Analysis and Extensions of RELIEF". In: *European Conference on Machine Learning*. Springer. 1994, pp. 171–182.

[Koz+96] Koza, J. R. et al. "Automated Design of Both the Topology and Sizing of Analog Electrical Circuits using Genetic Programming". In: *Artificial Intelligence in Design'96*. Springer, 1996, pp. 151–170.

[KRD89] Kryger, M. H., Roth, T., and Dement, W. C. *Principles and Practice of Sleep Medicine*. Saunders, 1989.

[Li20] Li, H. "A Computationally Efficient Single-Channel EEG Sleep Stage Scoring Approach using Simple Structured CNN". In: *Journal of Physics: Conference Series*. Vol. 1678. 1. IOP Publishing. 2020, p. 012103.

[Li+20] Li, X. et al. "A Novel Machine Learning Unsupervised Algorithm for Sleep/Wake Identification using Actigraphy". In: *Chronobiology International* 37.7 (2020), pp. 1002–1015.

[Lia+12] Liang, S.-F. et al. "A Rule-Based Automatic Sleep Staging Method". In: *Journal of Neuroscience Methods* 205.1 (2012), pp. 169–176.

[Lon+14] Long, X. et al. "Analyzing Respiratory Effort Amplitude for Automated Sleep Stage Classification". In: *Biomedical Signal Processing and Control* 14 (2014), pp. 197–205.

[LHH35] Loomis, A. L., Harvey, E. N., and Hobart, G. "Potential Rhythms of the Cerebral Cortex During Sleep". In: *Science* 81.2111 (1935). URL: https://www.science.org/doi/abs/10.1126/science.81.2111.597, pp. 597–598. DOI: 10.1126/science.81.2111.597.

[LPM15] Luong, M.-T., Pham, H., and Manning, C. D. "Effective Approaches to Attention-Based Neural Machine Translation". In: *arXiv preprint arXiv:1508.04025* (2015). DOI: https://doi.org/10.48550/arXiv.1508.04025.

[Lup19] Luppi, P.-H. "Michel Jouvet, From the Discovery of Paradoxical Sleep and Muscle Atonia to the Role of Neuropeptides". In: *Biologie Aujourd'hui* 213.3-4 (2019), pp. 81–86.

[MLW18] Malik, J., Lo, Y.-L., and Wu, H.-t. "Sleep-Wake Classification via Quantifying Heart Rate Variability by Convolutional Neural Network". In: *Physiological Measurement* 39.8 (2018), p. 085004.

[Mar+15] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. URL: https://www.tensorflow.org/. 2015.

[Mit97] Mitchell, T. "Machine Learning". In: (1997).

[Mit80] Mitchell, T. M. *The Need for Biases in Learning Generalizations*. Department of Computer Science, Laboratory for Computer Science Research, 1980.

[Mon+18] Monica, C. D. et al. "Rapid Eye Movement Sleep, Sleep Continuity and Slow Wave Sleep as Predictors of Cognition, Mood, and Subjective Sleep Quality in Healthy Men and Women, Aged 20–84 Years". In: *Frontiers in Psychiatry* 9 (2018). URL: https://doi.org/10.3389/fpsyt.2018.00255, p. 255. DOI: 10.3389/fpsyt.2018.00255.

[Mos+09] Moser, D. et al. "Sleep Classification According to AASM and Rechtschaffen & Kales: Effects on Sleep Scoring Parameters". In: *Sleep* 32.2 (2009), pp. 139–149.

[Nat16] National Sleep Research Resource. *Multi-Ethnic Study of Atherosclerosis (MESA)*. URL: https://sleepdata.org/datasets/mesa. 2016. DOI: 10.25822/N7HQ-C406.

[Ola96] Olazaran, M. "A Sociological Study of the Official History of the Perceptrons Controversy". In: *Social Studies of Science* 26.3 (1996). URL: https://doi.org/10.1177/030631296026003005, pp. 611–659. DOI: 10.1177/030631296026003005.

[Pas+19] Paszke, A. et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf. Curran Associates, Inc., 2019, pp. 8024–8035.

[Pha+18] Phan, H. et al. "Joint Classification and Prediction CNN Framework for Automatic Sleep Stage Classification". In: *IEEE Transactions on Biomedical Engineering* 66.5 (2018), pp. 1285–1296.

[Pha+19] Phan, H. et al. "SeqSleepNet: End-To-End Hierarchical Recurrent Neural Network for Sequence-To-Sequence Automatic Sleep Staging". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 27.3 (2019), pp. 400–410.

[Pha+21] Phan, H. et al. "XSleepNet: Multi-View Sequential Model for Automatic Sleep Staging". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021).

[Pot+16] Potter, G. D. M. et al. "Circadian Rhythm and Sleep Disruption: Causes, Metabolic Consequences, and Countermeasures". In: *Endocrine Reviews* 37.6 (Oct. 2016). URL: https://doi.org/10.1210/er.2016-1083, pp. 584–608. ISSN: 0163-769X. DOI: 10.1210/er.2016-1083.

[RBH18] Rahman, M. M., Bhuiyan, M. I. H., and Hassan, A. R. "Sleep Stage Classification using Single-Channel EOG". In: *Computers in Biology and Medicine* 102 (2018), pp. 211–220.

[Ros57] Rosenblatt, F. *The Perceptron, a Perceiving and Recognizing Automaton Project Para.* Cornell Aeronautical Laboratory, 1957.

[RP57] Rosenblatt, F. and Papert, S. “The Perceptron”. In: *A Perceiving and Recognizing Automation, Cornell Aeronautical Laboratory Report* (1957), pp. 85–460.

[Rou+67] Roussel, B. et al. “Locus Ceruleus, Paradoxal Sleep, and Cerebral Noradrenaline”. In: *Comptes rendus des seances de la Societe de biologie et de ses filiales* 161.12 (1967), pp. 2537–2541.

[RHW86] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. “Learning Representations by Back-Propagating Errors”. In: *Nature* 323.6088 (1986), pp. 533–536.

[Rus10] Russell, S. *Artificial Intelligence : A Modern Approach.* Upper Saddle River, New Jersey: Prentice Hall, 2010. ISBN: 978-0-13-604259-4.

[Sch+07] Schenck, C. H. et al. “English translations of the first clinical reports on narcolepsy and cataplexy by Westphal and Gélineau in the late 19th century, with commentary”. In: *Journal of Clinical Sleep Medicine* 3.3 (2007), pp. 301–311.

[SP18] Schönauer, M. and Pöhlchen, D. “Sleep Spindles”. In: *Current Biology* 28.19 (Oct. 2018), R1129–R1130.

[Şen+14] Şen, B. et al. “A Comparative Study on Classification of Sleep stage Based on EEG Signals Using Feature Selection and Classification Algorithms”. In: *Journal of Medical Systems* 38.3 (2014), pp. 1–21.

[Sor+18] Sors, A. et al. “A Convolutional Neural Network for Sleep Stage Scoring from Raw Single-Channel EEG”. In: *Biomedical Signal Processing and Control* 42 (2018), pp. 107–114.

[Sup+16] Supratak, A. et al. “Survey on Feature Extraction and Applications of Biosignals”. In: *Lecture Notes in Computer Science.* URL: https://doi.org/10.1007/978-3-319-50478-0_8. Springer International Publishing, 2016, pp. 161–182. DOI: 10.1007/978-3-319-50478-0_8.

[TY17] Tang, M.-C. and Yang, M.-Y. “Evaluating Music Discovery Tools on Spotify: The Role of User Preference Characteristics”. In: *Journal of Library & Information Studies* 15.1 (2017).

[TV13] Te Lindert, B. H. and Van Someren, E. J. “Sleep Estimates using Microelectromechanical Systems (MEMS)”. In: *Sleep* 36.5 (2013), pp. 781–789.

[Tho11] Thorpy, M. J. “Chapter 1 - History of sleep medicine”. In: *Sleep Disorders Part I.* Ed. by P. Montagna and S. Chokroverty. Vol. 98. Handbook of Clinical Neurology. URL: https://www.sciencedirect.com/science/article/pii/B9780444520067000010. Elsevier, 2011, pp. 3–25. DOI: https://doi.org/10.1016/B978-0-444-52006-7.00001-0.

[TC14] Tononi, G. and Cirelli, C. “Sleep and the Price of Plasticity: From Synaptic and Cellular Homeostasis to Memory Consolidation and Integration”. In: *Neuron* 81.1 (2014). URL: https://www.cell.com/neuron/fulltext/S0896-6273(13)01186-0, pp. 12–34. DOI: https://doi.org/10.1016/j.neuron.2013.12.025.

[Vas+17] Vaswani, A. et al. “Attention is All You Need”. In: *Advances in Neural Information Processing Systems* 30 (2017). URL: https://proceedings.neurips.cc/paper$_{f}iles/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-$*Paper.pdf*.

[Vir+20] Virtanen, P. et al. “SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python”. In: *Nature Methods* 17 (2020). URL: https://rdcu.be/b08Wh, pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[VPT05] Vitaterna, M. H., Pinto, L. H., and Turek, F. W. “Molecular Genetic Basis for Mammalian Circadian Rhythms”. In: *Principles and Practice of Sleep Medicine.* Elsevier Inc, 2005, pp. 363–374. DOI: 10.1016/B0-72-160797-7/50037-9.

[Wal19] Walch, O. *Motion and Heart Rate From a Wrist-Worn Wearable and Labeled Sleep from Polysomnography.* URL: https://physionet.org/content/sleep-accel/1.0.0/. 2019. DOI: 10.13026/HMHS-PY35.

[Wal+19] Walch, O. et al. “Sleep Stage Prediction with Raw Acceleration and Photoplethysmography Heart Rate Data Derived from a Consumer Wearable Device”. In: *Sleep* 42.12 (2019), zsz180.

[Wol96] Wolpert, D. H. “The Lack of A Priori Distinctions Between Learning Algorithms”. In: *Neural Computation* 8.7 (Oct. 1996). URL: https://doi.org/10.1162/neco.1996.8.7.1341, pp. 1341–1390. DOI: 10.1162/neco.1996.8.7.1341.

[XZY20] Xiang, H., Zeng, T., and Yang, Y. "A Novel Sleep Stage Classification via Combination of Fast Representation Learning and Semantic-To-Signal Learning". In: *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2020, pp. 1–8.

[YK15] Yu, F. and Koltun, V. "Multi-Scale Context Aggregation by Dilated Convolutions". In: *arXiv preprint arXiv:1511.07122* (2015). DOI: https://doi.org/10.48550/arXiv.1511.07122.

[YL03] Yu, L. and Liu, H. "Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution". In: *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*. 2003, pp. 856–863.

[Zha+18a] Zhang, G.-Q. et al. "The National Sleep Research Resource: Towards a Sleep Data Commons". In: *Journal of the American Medical Informatics Association* 25.10 (2018). URL: https://doi.org/10.1093/jamia/ocy064, pp. 1351–1358. DOI: 10.1093/jamia/ocy064.

[Zha+20] Zhang, J. et al. "Orthogonal Convolutional Neural Networks for Automatic Sleep Stage Classification Based on Single-Channel EEG". In: *Computer Methods and Programs in Biomedicine* 183 (2020), p. 105089.

[Zha+18b] Zhang, X. et al. "Sleep Stage Classification Based on Multi-Level Feature Learning and Recurrent Neural Networks via Wearable Device". In: *Computers in Biology and Medicine* 103 (2018), pp. 71–81.

[Zha+19] Zhang, Y. et al. "Sleep Stage Classification using Bidirectional LSTM in Wearable Multi-Sensor Systems". In: *IEEE Conference on Computer Communications Workshops*. IEEE. 2019, pp. 443–448. DOI: 10.1109/INFCOMW.2019.8845115.