ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΜΕΤΑΦΡΑΣΗ ΜΟΥΣΙΚΗΣ ΣΕ ΑΝΑΠΑΡΑΣΤΑΣΗ MIDI ΜΕΣΩ ΧΡΗΣΗΣ ΝΕΥΡΩΝΙΚΩΝ ΔΙΚΤΥΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Γεώργιος Ε. Πετρόπουλος

Επίβλεψη : Άγγελος Χ. Πικράκης, Επίκουρος Καθηγητής Πα.Πει.

Αθήνα, Σεπτέμβριος 2018

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΜΕΤΑΦΡΑΣΗ ΜΟΥΣΙΚΗΣ ΣΕ ΑΝΑΠΑΡΑΣΤΑΣΗ MIDI ΜΕΣΩ ΧΡΗΣΗΣ ΝΕΥΡΩΝΙΚΩΝ ΔΙΚΤΥΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Γεώργιος Ε. Πετρόπουλος

Επίβλεψη : Άγγελος Χ. Πικράκης, Επίκουρος Καθηγητής Πα.Πει.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την ..... Σεπτεμβρίου 2018

.....	.....	.....
Άγγελος Πικράκης	*insert name*	*insert name*
Επ. Καθηγητής Πα.Πει.	Καθηγητής	Καθηγητής

Αθήνα, Σεπτέμβριος 2018

.....
Γεώργιος Ε. Πετρόπουλος
Διπλωματούχος Πληροφορικάριος Πανεπιστημίου Πειραιώς.

ΠΕΡΙΛΗΨΗ

Σκοπός της παρούσας διπλωματικής εργασίας είναι η διατύπωση και επίλυση του προβλήματος της μετάφρασης συχνοτήτων ήχου και μουσικής σε αναπαράσταση midi. Στόχος της ανάθεσης του προβλήματος αυτού είναι η εκμάθηση του φοιτητή πάνω στα αντικείμενα ενασχόλησης της εργασίας, δηλαδή, να αποκτήσει εμπειρία πάνω σε θέματα επεξεργασίας ήχου και μηχανικής μάθησης.

Αρχικά αναλύθηκε η επεξεργασία ήχου που χρειάστηκε κατά την προεπεξεργασία των δεδομένων. Η επεξεργασία των δεδομένων συμπεριέλαβε διάφορες τεχνικές για να λυθεί το περίπλοκο πρόβλημα του ήχου ως πηγή δεδομένων. Το πρόβλημα ταξινόμησης, ανατέθηκε ως πρόβλημα κλάσεων, υπό επίβλεψη, καθώς οι κλάσεις-στόχοι ήταν γνωστοί πριν την πρόβλεψη. Η βάση δεδομένων που χρησιμοποιήθηκε παρείχε και τα χαρακτηριστικά προς εκμάθηση, αλλά και τις ταμπέλες των κλάσεων, υπό τη μορφή midi αριθμών. Χρησιμοποιήθηκε ειδική αρχιτεκτονική ενός νευρωνικού δικτύου, όπου επιτρέπει την κατανόηση χρονοσειρών ως είσοδο, ώστε να προβλέπεται η κλάση στόχος.

Στην παρούσα εργασία το προτεινόμενο μοντέλο υλοποιήθηκε στην γλώσσα προγραμματισμού Python3. Το περιβάλλον που επιλέχθηκε ήταν το Jupyter, που αξιοποιήθηκε η απλότητά του, και η εύκολη χρήση του σε debugging θέματα. Υλοποιούνται διάφορες λειτουργίες σε όλα τα στάδια της εργασίας, που μπορούν να επεκτείνουν την λειτουργία αυτού του προγράμματος. Υπάρχουν διαφοροποιήσεις μεταξύ των υπερπαραμέτρων που χρησιμοποιήθηκαν στις εκτελέσεις του προγράμματος, ώστε να γίνει κατανοητή η διαφορά απόδοσης μεταξύ αυτών.

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ

Επεξεργασία ήχου, Μετασχηματισμός Φουριέ, Νευρωνικά δίκτυα, Αναδρομικά νευρωνικά δίκτυα, Long short term memory neural network, LSTM, python, machine learning, keras, tensorflow, προσπελαση αρχειων midi, data parsing, μηχανικη μάθηση με επίβλεψη.

ABSTRACT

The scope of this diploma thesis is the formulation and solution of audio music, to midi translation. Primary goal is for the student is to get a hands on experience on concepts, theory and practice included in this challenging task, on the fields of sound processing and machine learning.

Firstly, there is a thorough explanation of the digital sound processing that was necessary for the data pre-processing part. This part included a variety of technics used to solve the hard task of sound processing as a data source. The part of classification in this thesis, was considered as a supervised learning task, as the labels where known prior the predictions. The dataset used, contained both the features that were used for training the model, and the labels of the classes, which were midi note numbers. A very specific architecture of a neural network was created, which was optimal for understanding concepts like time series, as an input.

In this diploma thesis, the suggested software environment was the use of Python3 programming language. The interface used was Jupyter, which was preferred to utilize its simplicity and easy debugging methods. There were many parts of the code that could be used to expand the capabilities of the program created. There were created many networks, with different hyper-paramerters, so to understand the various performances between them.

KEY WORDS

Audio processing, Fourier Transform, Neural networks, Recurrent neural networks, Long short term memory neural network, LSTM, python, machine learning, keras, tensorflow, midi protocol parsing, data parsing, supervised machine learning.

ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα διπλωματική εργασία εκπονήθηκε κατά το ακαδημαϊκό έτος 2017–2018 υπό την επίβλεψη του κ. Άγγελου Πικράκη, επίκουρου καθηγητή της σχολής Πληροφορικής του Πα.Πει., στον οποίο οφείλω ιδιαίτερες ευχαριστίες για την ανάθεσή της, δίνοντάς μου την ευκαιρία να ασχοληθώ με ένα τόσο ενδιαφέρον και δημιουργικό θέμα. Επίσης, θα ήθελα να τον ευχαριστήσω θερμά για την υπομονή, την πολύτιμη βοήθεια και καθοδήγηση που μου παρείχε σε όλη τη διάρκεια εκπόνησης της εργασίας, καθώς και για τον πολύτιμο χρόνο που μου αφιέρωσε.

Τέλος, θα ήθελα να ευχαριστήσω την οικογένειά μου και ιδιαίτερα τη μητέρα μου Δέσποινα, η οποία αποτελεί πηγή έμπνευσης και δύναμης σε όλη τη διάρκεια της ζωής μου, και την Έλενα, στην συμπαράσταση και την αγάπη της οποίας οφείλω το μέλλον μου.

ΠΕΡΙΕΧΌΜΕΝΑ

ΕΙΣΑΓΩΓΗ 1
1.1 Μηχανική Μάθηση σε Θέματα Ήχου 1
1.2 Περιγραφή Προβλήματος 2
1.3 Περιγραφή Εργασίας 3
1.4 Δομή Εργασίας 3
ΠΡΟΕΠΕΞΕΡΓΑΣΙΑ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΠΕΡΙΓΡΑΦΗ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ ... 5
2.1 Εισαγωγή 5
2.2 Βιβλιοθήκες 5
2.3 Ανάλυση Προεπεξεργασίας 6
2.3.1 Extract 6
2.3.2 Create_Frames 9
2.3.3 Spectrogram 10
2.3.4 Εναλλακτική Μέθοδος Τοπικών Μεγίστων 18
2.3.5 Συνάρτηση zero_pad 18
2.4 Τελική Συνάρτηση 21
ΠΡΟΕΠΕΞΕΡΓΑΣΙΑ ΚΛΑΣΕΩΝ MIDI ΝΟΤΩΝ 23
3.1 Εισαγωγή 23
3.2 Αρχείο Midiz.ipynb 23
3.3 Διαδικασία Εξαγωγής Κλάσεων 27
3.3.1 Συνάρτηση Piano_roll 29
ΑΡΧΙΤΕΚΤΟΝΙΚΉ ΚΑΙ ΕΚΠΑΊΔΕΥΣΗ ΑΝΑΔΡΟΜΙΚΟΎ ΝΕΥΡΩΝΙΚΟΎ ΔΙΚΤΥΟΥ 31
4.1 Εισαγωγή 31
4.2 Backpropagation Through Time (BPTT) 32
4.2.1 Συναρτήσεις κόστους 36
4.2.2 Long Short Term Memory 37
4.3 Γενικά χαρακτηριστικά αρχιτεκτονικής δικτύου 37
4.3.1 Δεδομένα 39
4.3.2 Εκπαίδευση Δικτύου 40
4.4 Ανάλυση Κώδικα Νευρωνικού Δικτύου 42
4.4.1 Αρχιτεκτονική και Εκπαίδευση Δικτύου 44
ΕΦΑΡΜΟΓΕΣ ΚΑΙ ΑΠΟΤΕΛΕΣΜΑΤΑ 50
5.1 Εισαγωγή 50
5.2 Εφαρμογή στο Δίκτυο 50
5.3 Τεχνική Επαλήθευσης 51

5.3.1 F-score χωρίς βιβλιοθήκη 51
5.3.2 F-score με βιβλιοθήκη 53
5.3.3 Πρόβλεψη μέγιστης πιθανότητας 54
ΣΥΜΠΕΡΑΣΜΑΤΑ 56
6.1 Σύνοψη Εργασίας και Αποτελεσμάτων της 56
6.2 Μελλοντικές Επεκτάσεις της Εργασίας 58
ΒΙΒΛΙΟΓΡΑΦΙΑ 59

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

1.1 Μηχανική Μάθηση σε Θέματα Ήχου

Την τελευταία δεκαετία η μηχανική μάθηση και η τεχνητή νοημοσύνη βρίσκεται σε ραγδαία ανάπτυξη, λόγω της εξίσου ραγδαίας ανάπτυξης των ηλεκτρονικών υπολογιστών, όσον αναφορά την επεξεργαστική τους δύναμη, αλλά και λόγω των διαθέσιμων δεδομένων. Παρόλο που η θεωρία των νευρωνικών δικτύων είχε πρωτοεμφανιστεί την δεκατία των 50', δεν ήταν δυνατή η εφαρμογή της σε περίπλοκα προβλήματα, καθώς έλλειπαν τα παραπάνω ζητούμενα.

Η θεωρία αυτή μπορεί να εφαρμοστεί όπως έχει αποδειχθεί σε οποιοδήποτε είδος προβλήματος, ενεξαρτήτως αντικειμένου. Το αντικείμενο της διπλωματικής εργασίας ασχολείται με δεδομένα ήχου, πιο συγκεκριμένα, με μουσική. Υπάρχουν πολλές προσεγγίσεις πάνω στο αντικείμενο του ήχου, όπως η αναγνώριση φωνής, αναγνώριση ηχητικών κυμάτων, για διάφορους σκοπούς. Στην περίπτωση της διπλωματικής εργασίας αυτής, ασχολούμενη με μουσική, έχει ως στόχο την αναγνώρισή της ώστε να μπορεί να υπάρξει μετάφραση σε μορφή midi νοτών. Δηλαδή, να υπάρξει αναγνώριση των χρωματικών νοτών που λαμβάνουν μέρος σε ένα αρχείο εισόδου, δημιουργώντας την κατάλλη αναπαράστασή του σε midi, ως έξοδο.

Το παρόν πρόβλημα, συγκαταλέγεται στον τομέα Music Information Retrieval. Το θέμα της διπλωματικής εργασίας επιλέχθηκε μέσω της ιστοσελίδας [1]. Καθώς επιλέχθηκαν μερικά θέματα από την ιστοσελίδα, αποφασίστηκε από κοινού με τον επιβλέπων καθηγητή Α.Πικράκη το αντικείμενο [2]. Έπειτα, συμφωνήθηκε να ακολουθηθεί μία έρευνα [3] σαν σκελετός, και κατά την πάροδο της εκπόνησης, υπήρξαν οι κατάλληλες καθοδηγήσεις από τον επιβλέπων καθηγητή που αποκλείναν από την έρευνα, δίνοντας την ευκαιρία για πειραματισμούς, ελέγχους, μεγαλύτερη ελευθερία πάνω στην εργασία, καθώς και πραγματική γνώση, έναντι τυφλής αναπαράστασης.

Τα τελευταία χρόνια έχει αναρηχηθεί στις προτεραιότητες των επιστημόνων όσον αναφορά την πρόκληση της αναγνώρισης του ήχου, καθώς η εποχή ευνοεί την επίλυση προβλημάτων αυτού του είδους, λόγω της ανάπτυξης της υπολογιστικής ισχύος και τεραστίων βάσεων δεδομένων, όπως προαναφέρθηκε παραπάνω. Υπάρχουν πολλές προσεγγίσεις, για διαφορετικές θεμιτές λειτουργίες, όπως [4], όπου αναγνωρίζει παρτιτούρες οι οποίες διαβάζονται σαν φωτογραφίες, ώστε να μεταφραστεί και αναγνωριστεί η μουσική με αυτόν τον τρόπο. Η έρευνα [5] χρησιμοποιεί άλλες τεχνικές, με τον ίδιο όμως στόχο. Χρησιμοποιεί επίσης νευρωνικά δίκτυα [6], διαφορετικού τύπου από την προσέγγιση που ακολουθήθηκε στην διπλωματική εργασία. Άλλες προσεγγίσεις πάνω στο αντικείμενο του ήχου είναι και η [7] που προτείνει την κατάλληλη μουσική για τον χρήστη, αλλά και η δημιουργία μουσικής χρησιμοποιώντας τεχνικές όπως αυτές που ακολουθήθηκαν στην εργασία [8].

Πολλές έρευνες, όπως αυτή που ακολουθήθηκε για την διπλωματική εργασία [3], προσφέρουν στην κοινότητα του χώρου κάποια πρόοδο στην επίλυση του προβλήματος

αυτού. Το πρόβλημα που πραγματεύεται η διπλωματική εργασία, βρίσκεται σε πρώιμο στάδιο ως προς την βέλτιστή του λύση. Μεγαλύτερες προσπάθειες γίνονται με την πάροδο του χρόνου [9], όπως και , δίνοντας δυνατά θεμέλια για τους ανερχόμενους επιστήμονες να συμβάλλουν με την σειρά τους, έως ότου ο τομέας της αναγνώρισης του ήχου να καλυφθεί. Το παρόν πρόβλημα θεωρείται δύσκολο σε υλοποίηση, καθώς δεν έχει υπάρξει κάποια έρευνα που να αποδεικνύει πως η προσέγγισή της είναι η καλύτερη δυνατή. Υπάρχουν, φυσικά, κάποιες τεχνικές θεωρούνται καλύτερες από άλλες, αλλά αυτό βρίσκεται στην εκτίμηση του ίδιου του επιστήμονα με βάση τα κριτήριά του.

Στην εργασία χρησιμοποιήθηκαν πολλές τεχνικές ώστε να υλοιποιηθεί το πρόγραμμα που πραγματοποιεί το ζητούμενο. Χρησιμοποιήθηκαν τεχνικές επεξεργασίας ήχου, όπως εφαρμογές παραθύρων [10] πριν την εφαρμογή.

1.2 Περιγραφή Προβλήματος

Η εργασία εκπονήθηκε σύμφωνα με την έρευνα [3], αλλά και με τις απαραίτητες διαφοροποιήσεις, για χάριν πειραματισμού, την εκμάθηση του αντικειμένου παρατηρώντας τις αλλαγές, όπως και σαφώς με την καθοδήγηση του επιβλέποντα καθηγητή.

Το πρόβλημα που τέθηκε ως αντικείμενο αυτής της διπλωματικής εργασίας, αφορά την κατανόηση ενός υπολογιστή, ως προς την συσχέτιση μεταξύ ήχου, μουσικής, και μουσικών νοτών στο πρωτόκολο midi. Μπορεί να θεωρηθεί ένας μετασχηματισμός από ήχο σε midi νότες. Στόχος ήταν η καταγραφή ήχου ως αριθμό midi νότας πάνω σε μία αναπαράσταση midi παρτιτούρας.

Αρχικά, δημιουργήθηκε το φασματογράφημα ενός αρχείου wave όπου χρησιμοποιήθηκε ως χαρακτηριστικό για την εκμάθηση του δικτύου,και ταυτόχρονα δημιουργήθηκαν οι στόχοι-κλάσεις με την αναπαράσταση του ίδιου μουσικού τραγουδιού, αρχείου σε μορφή midi.

Χρησιμοποιήθηκαν γνωστά δεδομένα (dataset [11]) το οποίο θα αναλυθεί στα παρακάτω κεφάλαια.

Η διπλωματική εργασία είχε πολλά στάδια μέχρι την τελική της υλοποίηση. Αρχικά, λύθηκε το πρόβλημα της επεξεργασίας αρχείων ήχου τύπου wave (.wav), ώστε να υπάρξει μια κατάλληλη μορφή αυτών των δεδομένων για να επεξεργαστούν από το νευρωνικό δίκτυο. Επίσης, δημιουργήθηκε πρόγραμμα το οποίο επεξεργάζεται αρχεία τύπου midi (.mid), ώστε να υπάρξουν οι ταμπέλες των δεδομένων των αντίστοιχων αρχείων wave. Έπειτα, δημιουργήθηκε λογισμικό που πραγματοποιεί την αρχιτεκτονική του νευρωνικού δικτύου. Συγκεκριμένα, επιλέχθηκε η αρχιτεκτονική ενός ειδικού αναδρομικού νευρωνικού δικτύου, όπου ονομάζεται Long Short Term Memory δίκτυο [12]. Ο λόγος επιλογής της συγκεκριμένης αρχιτεκτονικής δικτύου, έναντι του κλασικού αναδρομικού, οφείλεται στην δυνατότητα αυτού να λύνει το πρόβλημα της απόκλισης της κλίσης (Gradient Vanish/Explosion) [13]. Οι αναλύσεις περί νευρωνικού δικτύου θα αναλυθούν εκτενώς στο Κεφάλαιο 4.

1.3 Περιγραφή Εργασίας

Το προγραμματιστικό περιβάλλον που εκπονήθηκε η εργασία αυτή, έλαβε μέρος σε windows 7, χρησιμοποιώντας την προγραμματιστική γλώσσα python, έκδοση 3.6.3 [14]. Ο κώδικιας γράφτηκε πάνω στο Jupyter [15], καθώς καθίστησε την δυνατότητα του ελέγχου του προγράμματος (debugging) πολύ εύκολη, σχετικά με άλλα, όπως το περιβάλλον Pycharm. Η εκπαίδευση του δικτύου, δυστυχώς έλαβε μέρος χρησιμοποιώντας CPU του προσωπικού υπολογιστή, καθώς η κάρτα γραφικών (GPU) δεν συνίσταται να χρησιμοποιηθεί, σύμφωνα με την επίσημη ιστοσελίδα της Nvidia ([16], δίνοντας στην κάρτα του υπολογιστή GTX GeForce 607M πολύ χαμηλή βαθμολογία 2.1. Η δυνατότητα για cloud computing θεωρήθηκε περιττή.

Η εργασία απαρτίστηκε από 3 αρχεία κώδικα. Στο 1^{ο} , υλοποιήθηκαν οι μηχανισμοί προεπεξεργασίας των δεδομένων των αρχείων ήχου τύπου wave (feature extraction). Στο 2^{ο} , υλοποιήθηκαν οι μηχανισμοί για τις κλάσεις των αρχείων, αξιοποιώντας τα αρχεία τύπου midi. Να σημειωθεί πως έχει γίνει περαιτέρω επεξεργασία των αρχείων αυτών στον κώδικα αυτό, επιτρέποντας να εξαχθούν πολλές παραπάνω πληροφορίες που δεν αξιοποιήθηκαν στην συγκεκριμένη διπλωματική εργασία. Στο 3^{ο} αρχείο, υλοποιήθηκαν, η αρχιτεκτονική του δικτύου, η εκπαίδευση αυτού, καθώς και η δοκιμή των προβλέψεων αυτού. Δημιουργήθηκαν επίσης, υποφάκεφαλοι που αποθηκεύτηκαν τα κατάλληλα ονομασμένα αρχεία hdf5, όπου έχουν αποθηκευμένα τα βάρη του δικτύου, ώστε να μπορεί να φορτώσει το δίκτυο εύκολα για μια πρόβλεψη. Κάθε φάκελος, έχει ξεχωριστή αρχιτεκτονική και υπερπαραμέτρους, ρυθμισμένα βάσει την δοκιμή της εκάστοτε εκτέλεσης του προγράμματος.

Τα δεδομένα που χρησιμοποιήθηκαν, με τίτλο MAPS Dataset [11] αποθηκεύτηκαν σε εξωτερικό σκληρό δίσκο, λόγω του μεγάλου όγκου τους. Η βάση περιείχε αρχεία μεμονομένων νοτών, αλλά και μουσικών κομματιών, σε εκτελέσεις πιάνου. Χρησιμοποιήθηκαν μόνο τα μέρη της βάσης, τα οποία είναι μουσικά τραγούδια. Ο λόγος είναι ότι αυτό δίνει την δυνατότητα στο δίκτυο να δέχεται μέρη τραγουδιών ώστε να συλλάβει την έννοια της συνέχειας της μελωδίας. Περισσότερα θα αναλυθούν παρακάτω.

Δημιουργήθηκε φάκελος για την επικοινωνία των αρχείων μεταξύ τους, καθώς συναρτήσεις που δημιουργήθηκαν στην προεπεξεργασία των δεδομένων, χρειάστηκαν και στον κώδικα του νευρωνικού δικτύου, όπως και συναρτήσεις της επεξεργασίας των midi αρχείων, αντίστοιχα.

1.4 Δομή Εργασίας

Η παρούσα εργασία οργανώνεται σε επτά κεφάλαια:

- ✓ Στο **Κεφάλαιο 2** παρουσιάζεται το στάδιο της προεπεξεργασίας των δεδομένων, αρχείων wave, όπως και η περιγραφή των δεδομένων που χρησιμοποιήθηκαν.

- ✓ Στο **Κεφάλαιο 3** παρουσιάζεται το στάδιο της προεπεξεργασίας των δεδομένων, αρχείων midi.

- ✓ Στο **Κεφάλαιο 4** αναλύεται η αρχιτεκτονική του νευρωνικού δικτύου, η διαδικασία εκπαίδευσης, σε όλες τις παραδοχές που έλαβαν μέρος.

- ✓ Στο **Κεφάλαιο 5** παρουσιάζονται και σχολιάζονται τα αποτελέσματα εφαρμογής του μοντέλου που εκπαιδεύτηκε.

- ✓ Στο **Κεφάλαιο 6** γίνεται μια σύνοψη της εργασίας και προτείνονται τρόποι βελτίωσης και επέκτασής της.

- ✓ Στο **Κεφάλαιο 7** παρουσιάζεται η βιβλιογραφία που χρησιμοποιήθηκε σε όλη την έκταση της εργασίας.

ΚΕΦΑΛΑΙΟ 2

ΠΡΟΕΠΕΞΕΡΓΑΣΙΑ ΔΕΔΟΜΕΝΩΝ ΚΑΙ ΠΕΡΙΓΡΑΦΗ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

2.1 Εισαγωγή

Η προεπεξεργασία δεδομένων, είναι απαραίτητο βήμα σε κάθε επίλυση προβλήματος που συμπεριλαμβάνει νευρωνικά δίκτυα. Η επεξεργασία που πραγματοποιήθηκε ανήκει στην κατηγορία επεξεργασίας ήχου, καθώς επεξεργάζονται ασυμπίεστα αρχεία ήχου τύπου wave. Υπάρχουν πολλές προσεγγίσεις στην επεξεργασία αυτών των αρχείων, όμως, πρέπει να επιλεχθεί η κατάλληλη, με βάση τον στόχο της εκάστοτε διαδικασίας. Στην εργασία αυτή, στόχος της επεξεργασίας είναι να παράξουμε μια τράπεζα 88 κάδων, αντιπροσωπώντας τις 88 midi νότες, ώστε να παραχθεί το φασματογράφημα, το οποίο έπειτα γίνεται η είσοδος το νευρωνικό δίκτυο.

2.2 Βιβλιοθήκες

Χάρη στην γλώσσα που επιλέχθηκε, υπήρξε η δυνατότητα να χρησιμοποιηθούν βιβλιοθήκες της γλώσσας. Μία βιβλιοθήκη είναι ένα πακέτο κώδικα, όπου προσφέρει μια πληθώρα από συναρτήσεις, ώστε να διευκολύνει όσους έχουν ανάγκη τις συναρτήσεις αυτές. Μία βιβλιοθήκη μπορεί να δώσει συναρτήσεις για ένα συγκεκριμένο αντικείμενο, όπως για παράδειγμα η βιβλιοθήκη scipy [17] ,όπου χρησιμοποιήθηκε στην παρούσα εργασία. Δίνει την δυνατότητα να χρησιμοποιούνται συναρτήσεις, κερδίζοντας χρόνο αλλά και καταφέρνοντας πιο πολύπλοκες εκφράσεις μέσα στον κώδικα. Πάνω στην επεξεργασία του ήχου, απαραίτητη συνάρτηση είναι αυτή του μετασχηματισμού Φουριέ [18] η οποία μπορεί να βρεθεί έτοιμη πάνω στις βιβλιοθήκες numpy [19]. Χρησιμοποιήθηκαν πολλές βιβλιοθήκες, οι οποίες βρίσκονται στο αρχικό μέρος του αρχείου Preprocess.ipynb.

Συγκεκριμένα, με την εντολή import είναι εφικτό να ενταχθούν στον κώδικα οι βιβλιοθήκες, έτσι ώστε να καλεστεί κάποια συνάρτησή τους. Οι κύριες βιβλιοθήκες που χρησιμοποιήθηκαν ήταν οι Numpy, Scipy, Librosa, και sklearn.

Η βιβλιοθήκη Numpy χρησιμεύει στις μαθηματικές εκφράσεις που είναι απαραίτητες να γραφτούν στον κώδικα. Η Scipy βιβλιοθήκη, χρησιμοποιείται για πολλούς σκοπούς. Για την διπλωματική εργασία αξιοποιήθηκε το μέρος της που αφορά σήματα, Scipy.Signal. Η βιβλιοθήκη librosa [20], είναι μία βιβλιοθήκη, η οποία αφορά επεξεργασία ήχου αποκλειστικά, και χρησιμοποιείται ευρέως για ανάλυση ήχου και μουσικής. Αξιοποιήθηκε μόνο η δυνατότητα εισαγωγής ενός αρχείου τύπου wave, με τα κατάλληλα ορίσματά της. Η βιβλιοθήκη sklearn,

χρησιμοποιείται για μηχανική μάθηση. Στην περίπτωση της εργασίας αυτής, αξιοποιήθηκε το μέρος της προεπεξεργασίας δεδομένων, χρησιμοποιώντας τεχνικές ομαλοποίησης τιμών.

2.3 Ανάλυση Προεπεξεργασίας

Η βασική αρχή με την οποία λαμβάνει χώρα η προεπεξεργασία δεδομένων, υφίσταται από την έρευνα [3]. Δημιουργήθηκαν συναρτήσεις σε όλη την έκταση της εργασίας, ώστε να είναι δυνατό να καλεστούν όποτε θεωρηθεί απαραίτητο. Στο αρχείο του νευρωνικού δικτύου, αν κάποιος εκτελέσει την εντολή help(Preprocess), θα εμφανιστεί η παρακάτω έξοδος, απεικονιζόμενη στο σχήμα 2.1:

```
In [4]: 1 help(Preprocess)

Help on module Preprocess:

NAME
    Preprocess

FUNCTIONS
    create_frames(data, nfft_length, number_of_frames)

    extract(inputfile, small_window_size=True)

    firdif(fb)

    process(inputfile, small_window_size)

    spectro(frames, nfft_length, sr)
        Hertz vocabulary.

    wave_plot(filterbanks)

    zero_pad(filterbanks)

DATA
    interact = <ipywidgets.widgets.interaction._InteractFactory object>

FILE
    c:\users\gio\desktop\p12127\jupyter notebook home\ptuxiakh\preprocess.ipynb
```

Σχήμα 2.1 : Παρουσίαση υπάρχοντων συναρτήσεων του αρχείο Preprocess.ipynb

2.3.1 Extract

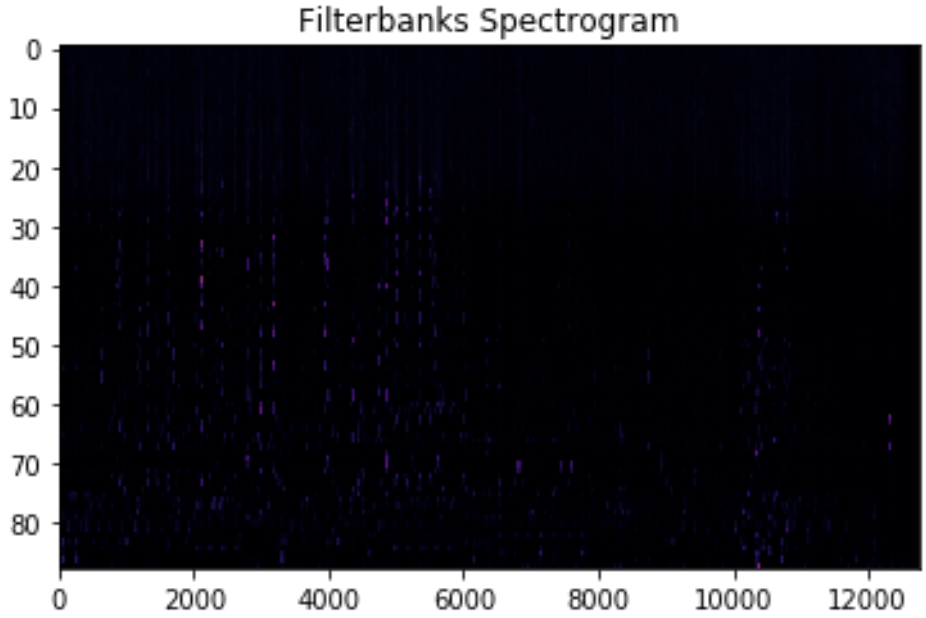
Η πρώτη συνάρτηση που δημιουργήθηκε, έχει ως αποτέλεσμα την φόρτωση ενός αρχείου wave, στο πρόγραμμα, δίνοντας στις εξόδους της απαραίτητες πληροφορίες. Συγκεκριμένα, η συνάρτηση έχει 2 ορίσματα. Το $1^{ο}$, δηλώνει το ποιο αρχείο wave πρόκειται να επεξεργαστεί, και το $2^{ο}$ τι είδους παράθυρο μετασχηματισμού Φουριέ θα υποστεί, σε επόμενα στάδια της επεξεργασίας. Καλείται η συνάρτηση από την βιβλιοθήκη librosa, φορτώνοντας το αρχείο. Έτσι, υπάρχουν τα δεδομένα και ο ρυθμός δειγματοληψίας του αρχείου αυτού, σε μεταβλητές. Έπειτα, εφαρμόζουμε ένα φίλτρο τύπου pre-emphasis [21], δίνοντας μικρότερο

εύρος τιμών μεταξύ χαμηλών και υψηλών συχνοτήτων. Αυτός ο τρόπος, παρόλο που εφαρμοζόταν κυρίως σε παλαιότερες μεθοδολογίες, και δεν αναφέρεται στην έρευνα [3], βοηθάει στο πρόβλημα που αντιμετωπίζει η έρευνα σε μεταγενέστερο επίπεδο. Το πρόβλημα αυτό, είναι η χαμηλή ανάλυση των χαμηλών τραπεζών (δηλαδή συχνοτήτων), προς τα τελικά στάδια της επεξεργασίας. Επομένως, επιλέχθηκε το φίλτρο τύπου pre-emphasis, το οποίο ορίζεται ως εξής:

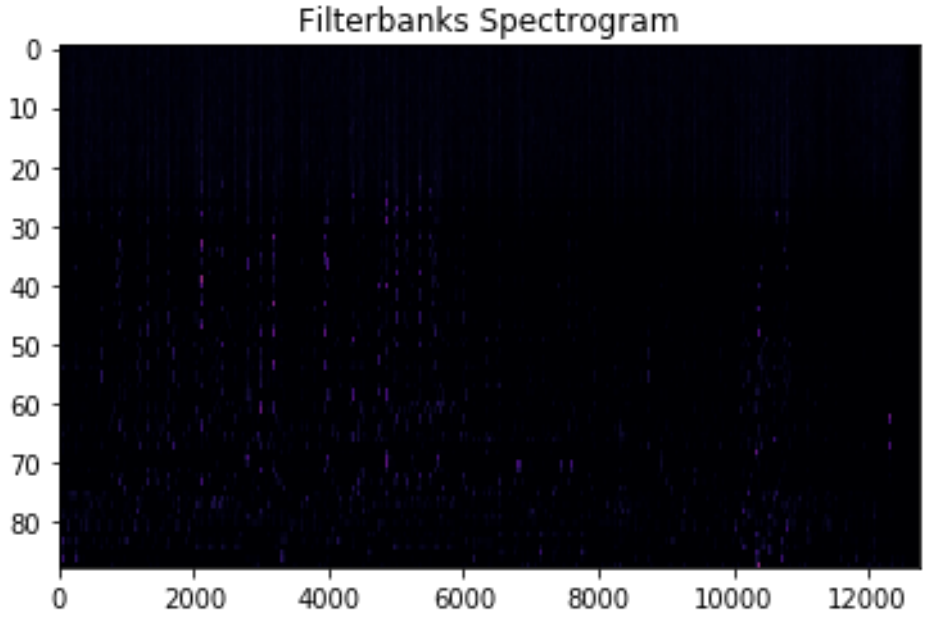
$$y(t) = x(t) - ax(t-1)$$

First Order High Pass Filter

Όπου x(t) είναι το σήμα, a είναι ο συντελεστής, με συνήθεις τιμές {0.95, 0.97}, και y(t) προφανώς το νέο σήμα. Επιλέχθηκε στο πρόγραμμα η τιμή του συντελεστή a ίση με 0.95. Παρατηρήθηκε, μετά από ελέγχους του τελικού φασματογραφήματος, μηδαμινή διαφορά μεταξύ των συντελεστών, όπως φαίνεται στο Σχήμα 2.2:

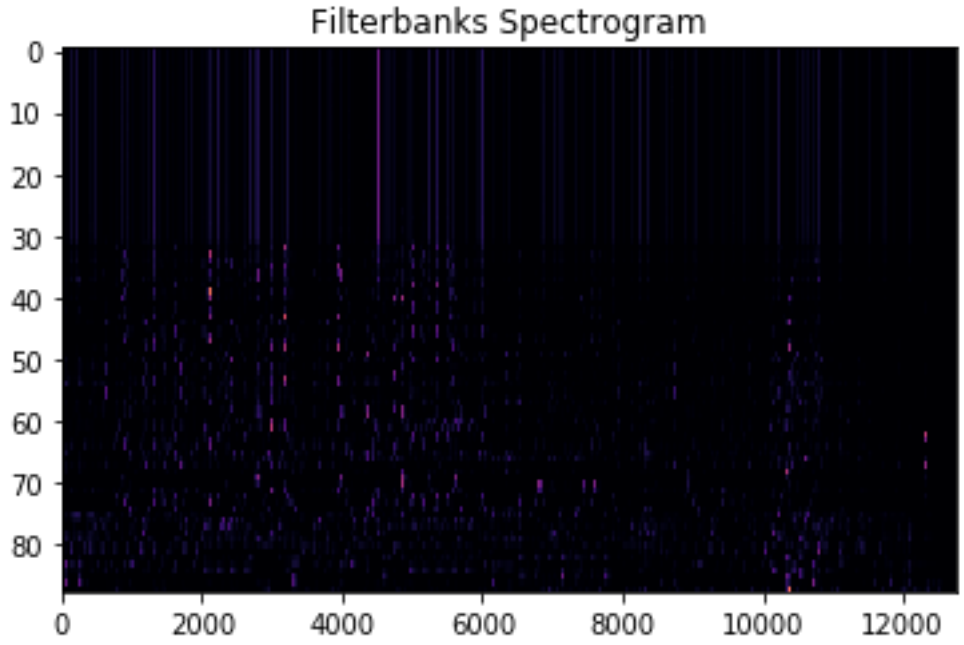


Σχήμα 2.2.1 : Συντελεστής 0.97



Σχήμα 2.2.2 : Συντελεστής 0.95

Όπως και στην περίπτωση που δεν χρησιμοποιηθεί το φίλτρο αυτό, έχουμε το παρακάτω αποτέλεσμα :



Σχήμα 2.2.3 : Φασματογράφημα χωρίς φίλτρο

Όπως είναι εμφανές στην ακριβώς παραπάνω εικόνα, οι κάθετες γραμμές δίνουν την ψευδαίσθηση των υπάρχουσων συχνοτήτων. Λόγω λοιπόν αυτής της αλλοίωσης που υπάρχει στο τελικό αποτέλεσμα χωρίς το φίλτρο, επιλέχθηκε να χρησιμοποιηθεί, ώστε να αποφευχθεί ο θόρυβος αυτός.

Το παράθυρο που προαναφέρθηκε, αναφέρεται στο εύρος παρατήρησης του μετασχηματισμού Φουριέ, που θα πραγματοποιηθεί παρακάτω. Η έρευνα [3] προτείνει 2 είδη

παραθύρων. Το 1ο, προτείνεται στα **46.4 ms**, που σημαίνει **2048 στοιχεία** προς μετασχηματισμό. Αυτό το παράθυρο, όπως συνίσταται, προσφέρει ευαίσθητες τιμές όσο αναφορά τις αλλαγές που συμβαίνουν κατα την διάσχιση των εικόνων (frames). Επομένως, προτείνεται και το ευρύτερο παράθυρο παρατήρησης, τετραπλάσιο σε μέγεθος από το προηγούμενο, δηλαδή **185.8 ms**, δίνοντας στοιχεία σε ομάδες των **8192**. Το μακρύτερο παράθυρο, όπως προτείνει η έρευνα που ακολουθείται, προσφέρει καλύτερη ανάλυση των συχνοτήτων. Επειδή λοιπόν υπάρχουν 2 πιθανές τιμές για το μέγεθος του παραθύρου, το αντίστοιχο όρισμα της συνάρτησης extract, τέθηκε σε είδος Boolean, ώστε όταν τεθεί ως Αληθής, να δώσει το μικρό παράθυρο, αλλιώς να δώσει το μεγαλύτερο. Με αυτόν τον τρόπο, εισπράττεται η τιμή του μήκους του μετασχηματισμού Φουριέ που θα ακολουθήσει. Αποθηκεύεται το παράθυρο που θα χρησιμοποιηθεί στην μεταβλητή **nfft_length**.

Στην συνέχεια της συνάρτησης, εξάγονται δεδομένα περί διαστάσεων που έχει το αρχείο που εισάχθηκε. Συγκεκριμένα, παίρνοντας το μήκος της λίστας των δεδομένων, με προτεινόμενο παράθυρο επικάλυψης 1% = 10 ms (λόγω ρυθμού δειγματοληψίας ίση με 44100 Hz), υπολογίζεται το τελικό πλήθος των στοιχείων. Το πλήθος αποθηκεύεται στην μεταβλητή **number_of_frames.**

Η συνάρτηση επιστρέφει με την εντολή **return,** τα εξής δεδομένα :

1. data
   - Οι τιμές του αρχείου, τα δεδομένα.
2. sr
   - Ο ρυθμός δειγματοληψίας του αρχείου
3. nfft_length
   - Το μήκος του παραθύρου υπολογισμού Μεταχ.Φουριέ.
4. number_of_frames
   - Πλήθος στοιχείων προς επεξεργασία.

2.3.2 Create_Frames

Στην συγκεκριμένη συνάρτηση, δημιουργούνται τα πλαίσια προς επεξεργασία. Η συνάρτηση **create_frames** δέχεται 3 ορίσματα, παραγόμενα από την συνάρτηση **extract**. Δίνονται στα ορίσματα data, nfft_length, number_of_frames, οι αντίστοιχες τιμές των μεταβλητών που δημιουργήθηκαν καλώντας την συνάρτηση **extract**. Δημιουργείται χώρος

για τα πλαίσια, ίσος με τον συνολικό αριθμό πλαισίων που έχει προϋπολογιστεί επί του μήκους παραθύρου του μετασχηματισμού Φουριέ. Χρησιμοποιείται η βιβλιοθήκη numpy ώστε να μπορεί να επεξεργαστεί ευκολότερα σε όλο το μήκος της εργασίας. Είναι απαραίτητο να δημιουργηθεί numpy μήτρα/πίνακας, καθώς οι μαθηματικές συναρτήσεις που πρόκειται να εφαρμοστούν, προέρχονται από την ίδια βιβλιοθήκη, επομένως οφείλουν να συμφωνούν όσο αναφορά την εσωτερική δομή. Γνωρίζοντας πως η επικάλυψη παραθύρου παραμένει 10ms, εύκολα υπολογίζεται σε 441 πλαίσια. Έτσι, δημιουργείται, χρησιμοποιώντας αναδρομικές αναθέσεις από την λίστα data, στην μήτρα frames. Η συνάρτηση επιστρέφει με την εντολή **return:**

1. frames

- Πλαίσια διαμορφωμένα σύμφωνα με το παράθυρο Φουριέ

2.3.3 Spectrogram

Η συγκεκριμένη συνάρτηση, περιέχει πολλά στάδια της επεξεργασίας του συνολικής διαδικασίας που ακολουθήθηκε. Η συνάρτηση **spectrogram** έχει ως στόχο να δημιουργηθούν οι τράπεζες των συχνοτήτων. Έχει 3 ορίσματα, τα πλαίσια (frames) που δημιουργήθηκαν στην συνάρτηση **create_frames**, το μήκους του παραθύρου που χρησιμοποιήθηκε για τον μετασχηματισμό Φουριέ, καθώς και τον ρυθμό δειγματοληψίας.

Αρχικά, παράγεται στην συνάρτηση μία λίστα με τις συχνότητες που αντιστοιχούν στις μουσικές νότες που συναντώνται σε ένα πιάνο. Η λίστα έχει ως 1ο στοιχείο την συχνότητα 0, και ως τελευταίο στοιχείο, την συχνότητα του Nyquist, σύμφωνα με το θεώρημά του [22],το οποίο δηλώνει πως η μέγιστη συχνότητα που μπορεί να υπολογίσει την ύπαρξή της ο μετασχηματισμός Φουριέ, είναι η μισή του ρυθμού δειγματοληψίας:

$$\max(f) = \frac{f(sample\ rate)}{2}$$

Οι συχνότητες που επιλέχθηκαν ως κέντρα των 88 κάδων, θεωρούνται οι νότες του πιάνου, σύμφωνα με την εξίσωση:

$$\sum_{i=1}^{88} 2^{\frac{(i-49)}{12}} \times 440\ Hz$$

Όπου η σταθερά 440 οφείλεται στην νότα Λα (ή A) της 4ης οκτάβας του πιάνου. 440 Hz είναι η συχνότητα δηλαδή, της νότας Λα στην 4η οκτάβα, λέγοντας ότι το πιάνο είναι κουρδισμένο με βάση αυτήν τη συχνότητα. Υπάρχουν διάφορες τιμές που μπορούν να τεθούν, στο εύρος:

$$[432{,}446] \in \mathbb{Z}: \{mod\ 2 \equiv 0\}$$

Να σημειωθεί πως, στον κώδικα, επειδή υπάρχουν ήδη οι $1^{η}$ και τελευταία τιμή, οι συνολικές τιμές που λαμβάνονται για τον διαχωρισμό των τραπεζών είναι 88 + 2 = 90. Ο λόγος είναι πως πρέπει να διαχωριστούν οι τράπεζες με κορυφές τις συχνότητες που υπάρχουν στην λίστα **hz_notes** στον κώδικα. Επίσης, λόγω της γλώσσας Python, που ξεκινάει την μέτρηση δεικτών από το 0 και όχι το 1, η εξίσωση δημιουργίας των νοτών που εφαρμόστηκε αλλάζει στον κώδικα, εφαρμόζοντας:

$$\sum_{i=1}^{88} 2^{\frac{(i-48)}{12}} \times 440\ Hz$$

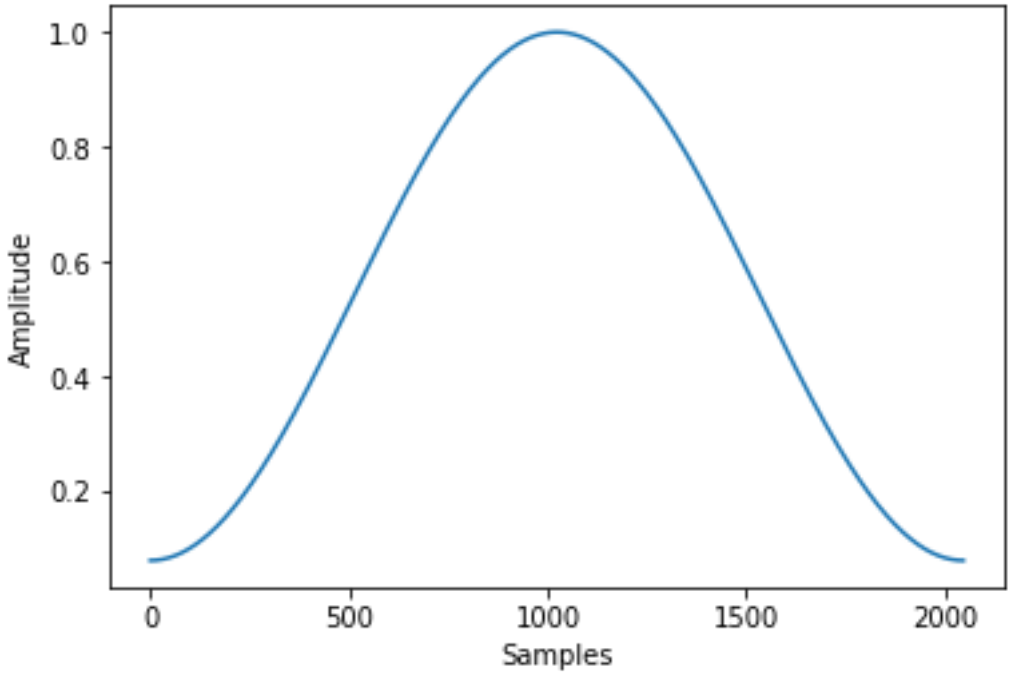
Έτσι, παίρνοντας τις 88 συχνότητες, και τις συχνότητες ίση με 0 και την Nyquist, δίνεται ο παρακάτω πίνακας, που δείχνει όλες τις συχνότητες που χρησιμοποιήθηκαν για την διαχώριση των τραπεζών:

```
Frequencies of notes that split the filterbanks:
[    0.          27.5         29.13523509    30.86770633
    32.70319566  34.64782887  36.70809599    38.89087297
    41.20344461  43.65352893  46.24930284    48.9994295
    51.9130872   55.          58.27047019    61.73541266
    65.40639133  69.29565774  73.41619198    77.78174593
    82.40688923  87.30705786  92.49860568    97.998859
   103.82617439 110.         116.54094038   123.47082531
   130.81278265 138.59131549 146.83238396   155.56349186
   164.81377846 174.61411572 184.99721136   195.99771799
   207.65234879 220.         233.08188076   246.94165063
   261.6255653  277.18263098 293.66476792   311.12698372
   329.62755691 349.22823143 369.99442271   391.99543598
   415.30469758 440.         466.16376152   493.88330126
   523.2511306  554.36526195 587.32953583   622.25396744
   659.25511383 698.45646287 739.98884542   783.99087196
   830.60939516 880.         932.32752304   987.76660251
  1046.5022612 1108.73052391 1174.65907167 1244.50793489
  1318.51022765 1396.91292573 1479.97769085 1567.98174393
  1661.21879032 1760.         1864.65504607 1975.53320502
  2093.0045224 2217.46104781 2349.31814334 2489.01586978
  2637.0204553 2793.82585146 2959.95538169 3135.96348785
  3322.43758064 3520.         3729.31009214 3951.06641005
  4186.00904481 22050.       ]
```

Σχήμα 2.3: Συχνότητες νοτών πιάνου στα 400Hz , κορυφές τραπεζών.

Στην συνέχεια της συνάρτησης, χρειάστηκε, σύμφωνα με την έρευνα [3] να εφαρμοστεί ένα παραμορφωτικό παράθυρο, πρωτού εφαρμοστεί ο μετασχηματισμός Φουριέ. Ο λόγος είναι πως τέτοια παράθυρα, εφαρμόζονται σε σήματα ήχου που βρίσκονται στο φάσμα του χρόνου, και όχι των συχνοτήτων. Αυτή η έρευνα [23] δίνει την σημαντικότητα εφαρμογής παραθύρων, αλλά και τις διαφορές μεταξύ των πολλών ειδών που υπάρχουν. Το παράθυρο που χρησιμοποιήθηκε ονομάζεται Hamming Window, μήκους ίσο με το παράθυρο εύρους του μετασχηματισμού Φουριέ. Ο λόγος για αυτήν την επιλογή μήκους, οφείλεται στο ότι ο σκοπός των παραμορφωτικών παραθύρων είναι να πολλαπλασιάσει ή να διαιρέσει το

πλάτος ενός σήματος, επομένως, θα πρέπει κάθε πλαίσιο παρατήρησης στο φάσμα του χρόνου, να γίνει αυτή η πράξη. Το αποτέλεσμα του παραθύρου, είναι να ελαχιστοποιήσει τις συχνότητες οι οποίες δεν είναι χρήσιμες για τον εκάστοτε σκοπό, αλλά και να αναδείξει τις θεμιτές. Ακόμα, ο λόγος που είναι απαραίτητο να ισούται το παράθυρο παραμόρφωσης με το μήκος πλαισίου, είναι για να αποφεχθεί, όσο είναι εφικτό, φασματική διαρροή [24]. Η τεχνική της παραθυροποίησης, δίνει στο φάσμα του χρόνου ενός σήματος, μια ομαλότητα μεταξύ των πλαισίων. Ο λόγος είναι πως, το παράθυρο, στις άκρες του, οι συντελεστές του προσεγγίζουν το 0, και σταδιακά προς το κέντρο του, αυξάνονται οι τιμές των συντελεστών. Παρακάτω φαίνεται το παράθυρο Hamming, που χρησιμοποιήθηκε στην εργασία:



Εικόνα 2.4: Hamming παράθυρο, μικρού Fourier Transform συντελεστή *2048*

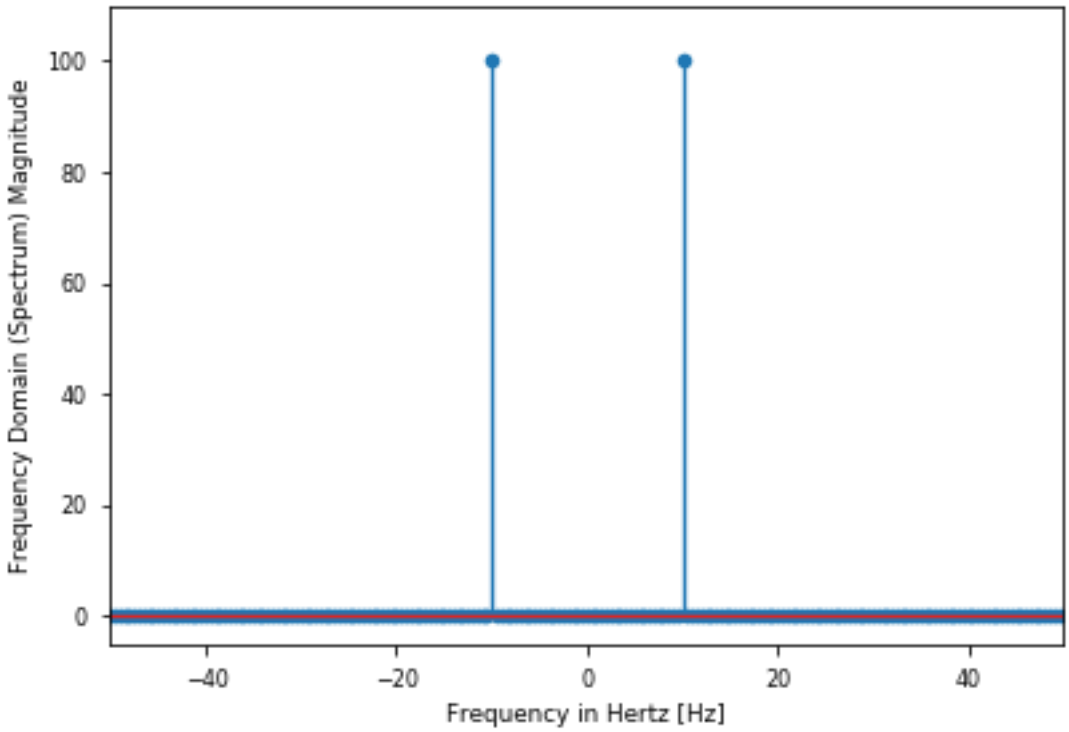
Χρησιμοποιώντας την βιβλιοθήκη scipy.signal [25], χρησιμοποιήθηκε το παράθυρο hamming, το οποίο συναρτησιακά εκφράζεται ως:

$$w(n) = 0.54 - 0.46 \cdot \cos(\frac{2\pi \cdot n}{M-1})$$

Όπου , $0 \leq n \leq M-1$, και M το πλήθος συντελεστών, δηλαδή το μέγεθος του παραθύρου για τον μετασχηματισμό Φουριέ. Στην Εικόνα 2.4, M =2048.

Έπειτα, υπολογίστηκε ο μετασχηματισμός Φουριέ. Ανάλογα το μέγεθος παραθύρου που υπιλέχθηκε, χρησιμοποιήθηκε η βιβλιοθήκη numpy, για να εφαρμοστεί ο αλγόριθμος FFT [18]. Χρησιμοποιήθηκε η συνάρτηση **rfft**, που σημαίνει πως, υπολογίστηκε ο μετασχηματισμός Φουριέ του πραγματικού μέρους του σήματος, δίνοντας έτσι σαν αποτέλεσμα, μόνο το πραγματικό μέρος του, μήκους μειωμένου δηλαδή κατά το ήμισυ. Ο τρόπος αυτός είναι δυνατός, καθώς η συνάρτης εκμεταλεύεται την συμμετρία του

μετασχηματισμού που παράγει, υπολογίζοντας μόνο τις θετικές συχνότητες που παράγονται. Παρακάτω, στην Εικόνα 2.5 ενδεικτικά φαίνεται το αποτέλεσμα του μετασχηματισμού Φουριέ, κρατώντας και τις αρνητικές συχνότητες:



Εικόνα 2.5 Αποτέλεσμα συνάρτησης FFT.

Ο αλγόρθιμος Fast Fourier Transform, έχει πολλές παραλλαγές, όμως ο πιο γνωστός είναι ο Cooley-Tukey FFT, με την έρευνά τους [26].Έτσι, όπως εξηγεί και η [27], ο FFT αλγόριθμος δίνει έναν γρήγορο τρόπο, για τον υπολογισμό μιας τέτοιας δύσκολης πράξης. Στην έρευνα [27] αναφέρεται λεπτομερώς ο χρόνος πολυπλοκότητας υπολογισμού του αλγορίθμου σε O(n log(n)), έναντι του κλασικού διακριτού μετασχηματισμού (DFT) που δίνει χρόνο O(n^2). Είναι σημαντική μείωση χρόνο εκτέλεσης, αν σκεφτεί κανείς πως αυτή η πράξη πρέπει να εφαρμόζεται για κάθε μέρος αρχείου που πρόκειται να τροφοδοτηθεί στο νευρωνικό δίκτυο.

Ο μετασχηματισμός Fourier ενός σήματος ορίζεται από την εξίσωση:

$$\mathcal{F}\{g(t)\} = G(f) = \int_{-\infty}^{\infty} g(t)e^{-2\pi i f t}\,dt$$

Όμως, επειδή τα άκρα του ολοκληρώματος αναφέρονται σε σήμα άπειρης ακρίβειας (δηλαδή, ο ρυθμός δειγματοληψίας να πλησιάζει το άπειρο), δεν είναι εφικτή η προσέγγιση μέσω αυτής της εξίσωσης. Χρειάζεται να υπάρξει διακριτό εύρος λήψης τιμών, ώστε να υπολογισθεί το φάσμα του σήματος. Η συνάρτηση που περιγράφει τον διακριτό μετασχηματισμό Φουριέ(DFT), [28] είναι η εξής:

$$X(\omega_\kappa) = \sum_{n=0}^{N-1} x(t_n) e^{-j\omega_k t_n}$$

k = {0, 1, …, N-1}

Όπου:

$\sum_{n=0}^{N-1} f(n)$	$f(0) + f(1) + \cdots + f(N-1)$
$x(t_n)$	Ένταση σήματος την χρονική στιγμή t_n (sec)
t_n	nT = n-οστό δείγμα, $n \exists \mathbb{Z} \geq 0$
T	Συχνότητα δειγματοληψίας
$X(\omega_\kappa)$	Φάσμα συχνότητας ω_κ
ω_κ	$k\Omega$ = K-οστή συχνότητα
Ω	$\frac{2\pi}{NT}$ = συχνότητα δειγματοληψίας (rad/sec)
f_s	$\frac{1}{T}$ = ρυθμός δειγματοληψίας (Hz)
N	Πλήθος δειγμάτων

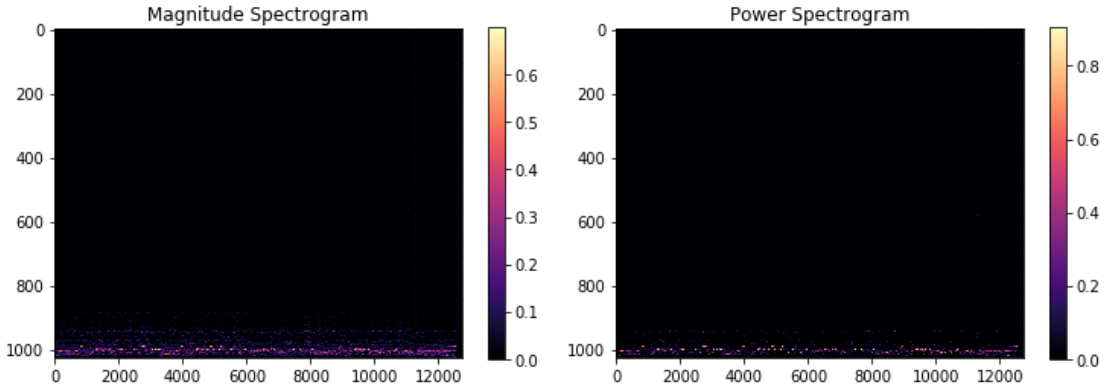
Ο αλγόριθμος FFT [28] (βλ. Σελίδα 6) εκμεταλεύεται την συμμετρία των περιττών και άρτιων τιμών του πλήθους δειγμάτων N, μέσω της συνάρτησης του DFT, δημιουργώντας πλέον 2 DFT σειρές των M=N/2 δειγμάτων η καθεμία. Αυτό, μπορεί να επαναληφθεί όσο το M παραμένει άρτιο. Χρησιμοποιώντας δηλαδή, την τεχνική Διαίρει και Βασίλευε [29], μειώνεται η πολυπλοκότητα χρόνου σε O(N logN) όπως προειπώθηκε. Αυτός είναι και ο σημαντικότερος λόγος που το μέγεθος πλαισίων που επιλέχτηκαν ήταν 2048 ή 8192. Οι 2 τιμές αυτές είναι δυνάμεις του 2, που σημαίνει διαιρούνται πλήρως, έτσι ώστε ο αλγόριθμος να είναι πλήρως αποδοτικός. Παρακάτω, υπάρχει ένας πίνακας που δείχνει την υπεροχή εκμετάλλευσης χρόνου του αλγορίθμου αυτού, έναντι του κλασικού DFT (που ονομάζεται Brute Force στον πίνακα):

Cost comparison:

N	$r = \log_2 N$	**BRUTE FORCE** $4N^2$	**FFT** $2N \log_2 N$	speedup
2	1	16	4	4
4	2	64	16	4
8	3	256	48	5
1,024	10	4,194,304	20,480	205
65,536	16	$1.7 \cdot 10^{10}$	$2.1 \cdot 10^{6}$	˜10^4

Σχήμα 2.6 [28] Σύγκριση πολυπλοκότητας απλού DFT και FFT

Επομένως, για κάθε πλαίσιο που παράχθηκε, σύμφωνα με το μήκος παραθύρου, εφαρμόστηκε το παράθυρο Hamming. Έπειτα εφαρμόσαμε τον αλγόριθμο FFT, λαμβάνοντας τις εντάσεις κάθε συχνότητας (Magnitude Εισαγωγή: [30]). Δηλαδή, εφαρμόζοντας τον αλγόριθμο FFT, το αποτέλεσμα ήταν να κρατηθεί το πραγματικό μέρος του σήματος, μήκους N/2 αν περιττός, ή N/2 +1 αν άρτιος αντίστοιχα, συντελεστών. Οι τιμές που δεν κρατήθηκαν ονομάζονται φάση του σήματος. Η φάση του σήματος, επειδή δεν χρησιμοποιήθηκε στην εργασία αυτή, δεν σημαίνει πως είναι περιττή [31]. Υπάρχουν έρευνες που αξιοποιούν την φάση ενός σήματος, ειδικά σε εικόνες, όπως για παράδειγμα [32].
Η τελική επεξεργασία που αφορά τον μετασχηματισμό, έγινε παίρνοντας τις ενέργειες του σήματος που λήφθηκε. Η έρευνα [33] (βλ. Κεφ.3.5) δίνει μια παρόμοια προσέγγιση ανάλυσης σήματος με την προεπεξεργασία της διπλωματικής εργασίας. Το φάσμα της ενέργειας του σήματος ονομάζεται Power Spectrum. Παρόλο που η έρευνα [3] προτείνει την έκφραση του σήματος μόνο στην έντασή της (Magnitude), αποφασίστηκε να χρησιμοποιηθεί για την διπλωματική εργασία, το φάσμα ενεργειών (Power Spectrum), καθώς προσφέρει μεγαλύτερη λεπτομέρια, αν τυπωθεί το αντίστοιχο φασματογράφημα:

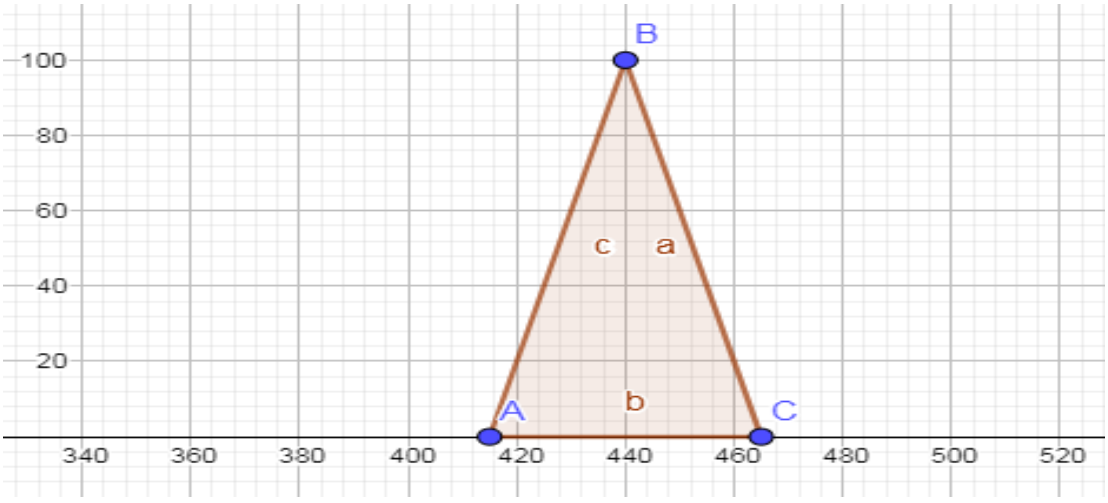


Εικόνα 2.7: Σύγκριση Φασματογραφημάτων μεταξύ Magnitude & Power Spectrums.

Εκ πρώτης όψεως, φαίνεται πιο λογικό να επιλέξει κανείς το φάσμα εντάσεων, καθώς είναι πιο διακριτές οπτικά. Όμως, παρατηρείται και μια συνέχεια (ελαφρύ μωβ = χαμηλές τιμές) σε όλην την πάροδο του χρόνου (Y άξονα). Αυτό, δημιουργούσε πρόβλημα κατά την εκπαίδευση, καθώς, αν και ομαλοποιημένες τιμές, δεν ήταν ποιοτικές αρκετά, ώστε να μάθει το δίκτυο την εξάρτηση μεταξύ των νοτών. Η έρευνα [3] χρησιμοποιεί τις εντάσεις, και όχι τις ενέργειες του μετασχηματισμού Φουριέ του σήματος. Παρά το γεγονός αυτό, αποφασίστηκε να αποκλίσει η εργασία από τα προτεινόμενα της έρευνας αυτής. Επομένως, επιλέχθηκε το φάσμα ενεργειών σαν το φασματογράφημα το οποίο, μετά από κάποια ακόμα επεξεργασία, θα τροφοδοτηθεί στο νευρωνικό δίκτυο.

Στην συνέχεια της συνάρτησης **spectrogram**, χρειάστηκε να φτιαχτούν οι 88 κάδοι των συχνοτήτων. Η λογική πίσω από αυτή την τεχνική, οφείλεται στις 88 νότες του πρωτόκολλου midi. Αυτό σημαίνει, πως, το φάσμα που θα τροφοδοτηθεί, πρέπει να αποτελείται από συγκεκριμένο αριθμό κλάσεων, ώστε να μπορεί να αντιστοιχηθεί σε κάποια νότα. Έτσι, δημιουργήθηκε μια τράπεζα συχνοτήτων, αποτελούμενη από 88 κάδους, ώστε όλες οι συχνότητες που υπάρχουν στο λαμβανόμενο σήμα, να δηλώνονται ως μία από τις 88 πιθανές προβλέψιμες νότες. Όπως είναι λογικό, στο πεδίο των συχνοτήτων, δεν υφίσταται ο όρος «νότα». Οι νότες αναφέρονται στις συχνότητες που έχουν οριστεί, όπως για παράδειγμα

η 4η οκτάβας ΛΑ αντιπροσωπεύει την συχνότητα 440 Hz. Αυτό με τη σειρά του, πρέπει να δώσει στην τράπεζα συχνοτήτων, έναν βαθμό στον κάδο που αντιστοιχεί σε αυτή την νότα. Λόγω της μεγάλης διασποράς συχνοτήτων σε ηχογραφήσεις τραγουδιών και φωνής, ενδέχεται να υπάρχουν πολλές συχνότητες, εκτός του μουσικού κουρδίσματος. Δηλαδή, ακόμα και αν είναι κουρδισμένο ένα πιάνο, μπορούν να υπάρξουν αρμονικές συχνότητες (ακέραια πολλαπλάσια των θεμελιωδών συχνοτήτων) που μπορούν να αλλοιώσουν την καθαρότητα του φασματογραφήματος. Επίσης, πολύ σημαντικό μέρος της εκπαίδευσης, αποτελεί και η εκτέλεση των μουσικών τραγουδιών, τα οποία έχουν εκτελεστεί σε διαφορετικό πιάνο. Η διαφορετική κατασκευή, παίξιμο, διαστάσεις χώρου, ποιότητας μικροφώνου ηχογράφησης, ή ακόμα θερμοκρασίας δωματίου και υγρασίας, δίνουν διαφορετικό «τόνο» στο κάθε όργανο. Επομένως, η συγκάληψη όλων των παρεμφερών συχνοτήτων, πέρα από τις αντιστοιχούμενες σε νότες, οφείλεται απαραίτητη για να συγκλίσει το πρόβλημα ταξινόμησης, και να μην υπερπροσαρμοστεί σε συγκεκριμένο ήχο το νευρωνικό δίκτυο. Το ίδιο συστήνει και η έρευνα [3].
Επομένως, για τις τράπεζες 88 θέσεων, θα ακολουθηθεί ένα τριγωνικό σχήμα (μορφή παραθύρου) για να δωσθούν όσο το δυνατό οι πιο ενδιαφέρουσες συχνότητες. Ενδιαφέρουσα συχνότητα ορίζεται αυτή που πλησιάζει κάποια συχνότητα νότας. Έχοντας την numpy μήτρα hz_notes, είναι γνωστές οι πιο ενδιαφέρουσες συχνότητες. Αυτές οι συχνότητες θα είναι η κορυφή για κάθε φίλτρο που θα χρησιμοποιήσουμε. Για να γίνει κατανοητό, παίρνοντας ένα από τα 88 τριγωνικά φίλτρα, έστω το 49ο (που αντιστοιχεί στην συχνότητα 440hz = A4) δίνεται το παρακάτω φίλτρο :



Εικόνα 2.9: Τριγωνικό φίλτρο κάδου νότας A4=440hz.

Στην **Εικόνα 2.9** το σημείο Α(415.3,0) αντιστοιχεί στην κορυφή του προηγούμενου τριγώνου, καθώς είναι 1 ημιτόνιο χαμηλότερα αυτής της Α4. Το σημείο Β(440,100) αντιστοιχεί στην νότα κορυφή του συγκεκριμένου τριγώνου. Το σημείο Γ(466.16, 0) αντιστοιχεί στην κορυφή του επόμενου τριγώνου, καθώς είναι 1 ημιτόνιο ψηλότερα αυτής της Α4. Ο οριζόντιος άξονας αντιπροσωπεύει τις συχνότητες, και ο κάθετος τον συντελεστή που θα πολλαπλασιαστεί με την συχνότητα που θα βρεθεί. Δηλαδή, αν υπήρχε ύπαρξη συχνότητας $\frac{415.3+440}{2}$, τότε θα υπήρχε ένα σημείο Δ($\frac{415.3+440}{2}$, 0.5) (ή 50 στην **Εικόνα 2.9**), δηλώνοντας το κατά πόσο συμφωνεί στην κοντινότερη νότα.

Επομένως, η κάθε κορυφή των 88 τριγώνων, αποτελεί την αντιστοιχούμενη νότα midi αριθμού, με απόκριση ίση με 1, και καθώς αποκλίνει γραμμικά από την συχνότητα, η απόκριση μηδενίζει στις γειτονικές συχνότητες που θεωρούνται νότες.

Χρειάστηκε να μετατραπούν οι συχνότητες από μονάδα μέτρησης Hertz, σε Mel-scale μονάδες. Ο λόγος μετατροπής, βρίσκεται στο γεγονός της ανθρώπινης ακοής. Η μετατροπή αυτή δύναται να παρουσιάζει νούμερα συχνοτήτων πιο όμοια με την αντίληψη ενός ανθρώπου. Αυτό σημαίνει πως, ενώ υπάρχει χαμηλότερη ανάλυση στις χαμηλότερες συχνότητες, και ψηλότερη ανάλυση στις υψηλότερες συχνότητες, στην ανθρώπινη ακοή, η μετατροπή αυτή αναιρεί την ιδιαιτερότητα αυτή. Η έρευνα [34] δίνει ένα παράδειγμα χρήσης των συχνοτήτων, όπως και η [35] κάνει εύκολα κατανοητή την λογική πίσω από αυτή τη μετατροπή. Συγκεκριμένα, οι συχνότητες [0,1000] Hz δίνονται γραμμικά από την μετατροπή, αλλά σε μεγαλύτερες συχνότητες, η αναλογία ανέρχεται σε λογαριθμική. Η συνάρτηση που χρησιμοποιήθηκε για την μετατροπή είναι η παρακάτω:

$$\mathrm{m} = 2595 \log_{10}\left(1 + \frac{f}{700}\right)$$

Η αντίστροφη συνάρτηση, ώστε να μετατραπούν οι mel συχνότητες σε Hertz:

$$f = 700\left(10^{\frac{m}{2595}} - 1\right)$$

Τελικά, λοιπόν, δίνεται η συνάρτηση που χρησιμοποιήθηκε για να φτιαχτούν και οι 88 τράπεζες φίλτρων:

$$H_m(k) = \begin{cases} 0 & \alpha\nu\ k < f(m-1)\ \text{ή}\ k > f(m-1) \\ \dfrac{k - f(m-1)}{f(m) - f(m-1)} & \alpha\nu\ f(m-1) \le k < f(m) \\ \dfrac{f(m+1) - k}{f(m+1) - f(m)} & \alpha\nu\ f(m) < k\ \le f(m+1) \\ 1 & \alpha\nu\ k = (fm) \end{cases}$$

Καθώς δημιουργήθηκαν οι τράπεζες φίλτρων πολλαπλασιάζοντας τις ενέργειες των πλαισίων με τους κάδους, χρειάστηκε μια περαιτέρω επεξεργασία. Η επεξεργασία που επιλέχθηκε αρχικά ήταν κανονικοποίηση.

Δόθηκε στις τράπεζες φίλτρων, ένας λογαριθμικός παράγοντας, ώστε οι ποσότητες των φίλτρων να είναι db (Decibel). Έπειτα, χρησιμοποιήσαμε την έτοιμη συνάρτηση της βιβλιοθήκης scipy.normalize, λαμβάνοντας κανονικοποιημένες τιμές εύρους (-1,1). Προτάθηκε από τον επιβλέποντα καθηγητή, οι αρνητικές τιμές να μετατραπούν σε 0, καθώς δεν υφίσταται αρνητική ένταση ήχου. Τέλος, οποιαδήποτε μηδενική υπήρχε στην τράπεζα, προστέθηκε μια ελάχιστη ποσότητα, ώστε να μην υπάρξει πρόβλημα στις λογαριθμήσεις, ή στην διαφορά μεταξύ διαδοχικών πλαισίων. Η συνάρτηση **spectrogram** επέστρεψε με την εντολή **return,** τα εξής δεδομένα:

1. filterbanks

- Τράπεζες φίλτρων 88 νοτών.

2.3.4 Εναλλακτική Μέθοδος Τοπικών Μεγίστων

Μια διαφορετική προσέγγιση, αποφεύγοντας την ομαλοποίηση των δεδομένων μεταξύ τους, είναι και η έυρεση τοπικών μεγίστων. Ανά πλαίσιο, στις 88 τιμές που υπάρχουν, αντιπροσωπεύοντας τις νότες, κρατήθηκαν τα τοπικά μέγιστα με πλάτος σύγκρισης ίσο με 1. Δηλαδή, αν η τιμή της $49^{ης}$ τράπεζας ήταν μεγαλύτερη της $48^{ης}$ και $50^{ης}$, η $49^{η}$ τράπεζα θα θεωρείται τοπικό μέγιστο. Σε περίπτωση ισότητας, αγνοείται. Έτσι, δημιουργήθηκε αντίστοιχος κώδικας, δίνοντας τα τοπικά μέγιστα. Αυτή η προσέγγιση, προφανώς έγινε ως αντικατάσταση της ομαλοποίησης που αναφέρθηκε στο τέλος της παραπάνω παραγράφου. Η τεχνική αυτή, παρόλο που υπάρχει απώλεια πληροφορίας, δίνει πιο κατανοητά δεδομένα στο νευρωνικό δίκτυο. Περισσότερες λεπτομέριες και συγκρίσεις, στο κεφάλαιο των αποτελεσμάτων.

2.3.5 Συνάρτηση zero_pad

Η συνάρτηση αυτή δημιουργήθηκε, ώστε να δημιουργηθούν παραπάνω πλαίσια για το αρχείο που είναι φορτωμένο. Ο λόγος είναι πως, το νευρωνικό δίκτυο, πρέπει να έχει σταθερές διαστάσεις εισόδου. Παρά το γεγονός ότι λαμβάνονται τυχαία πλάισια, είναι μια τεχνική για πιο ασφαλή προσπέλαση της εκπαίδευσης, ώστε να μην δημιουργηθούν προβλήματα ασυμβατότητας λόγω διαστάσεων. Πιο συγκεκριμένα, προστέθηκαν πλαίσια τόσα, ώστε τα συνολικά πλαίσια να είναι ακέραια πολλαπλάσια του 50, καθώς η είσοδος του νευρωνικού

δικτύου, δέχεται 50 διαδοχικά πλαίσια την φορά. Η συνάρτηση δέχεται τις τράπεζες ως είσοδο, και η εντολή **return** δίνει τα εξής δεδομένα:

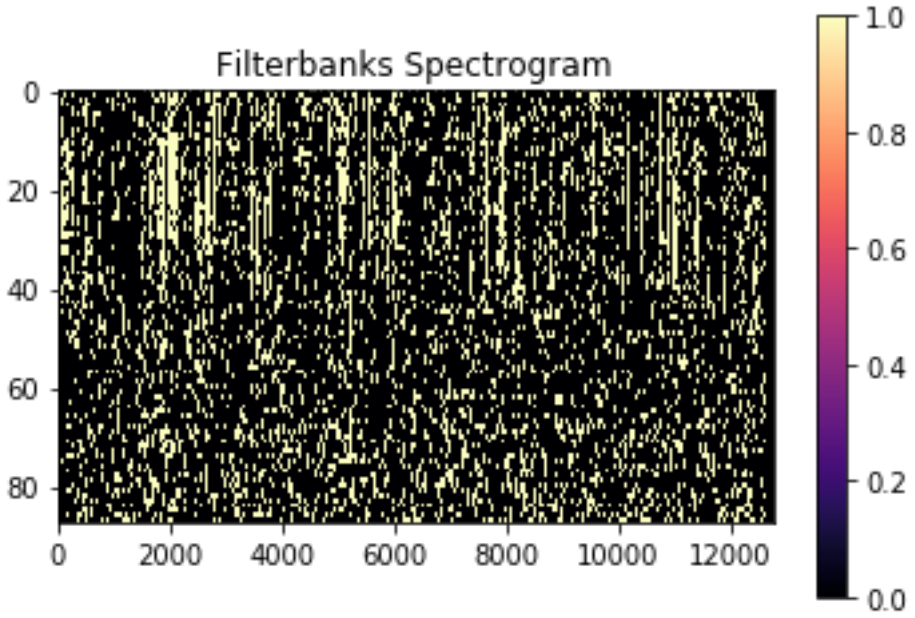
1. filterbanks

- Τράπεζες φίλτρων 88 νοτών. αριθμού πλαισίων πολλ/σιο του 50.

- **Παρουσίαση Φασματογραφήματος**

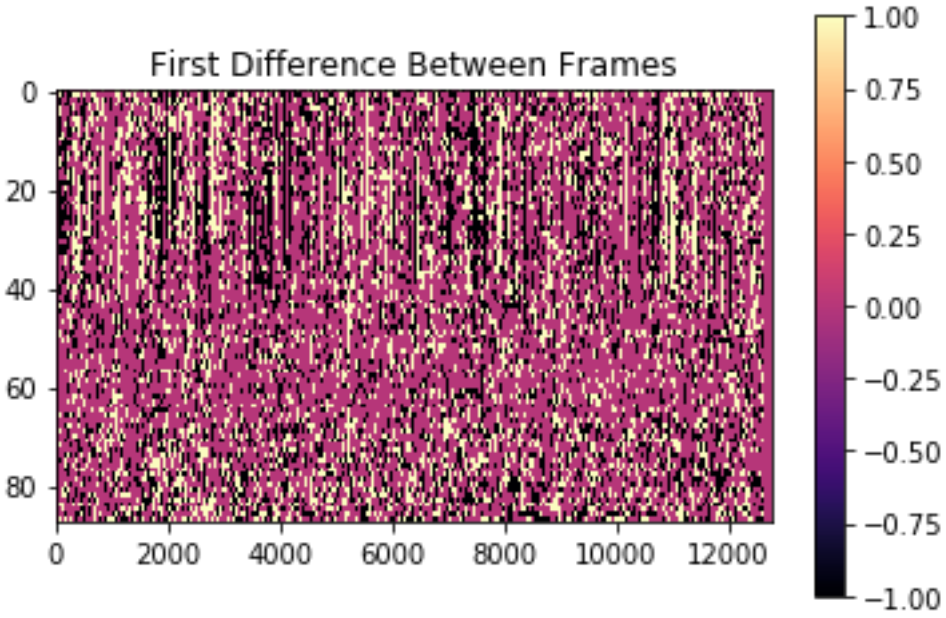
Κατά τη διαδικασία του ελέγχου κώδικα, ήταν απαραίτητο, να εμφανίζεται το τελικό αποτέλεσμα του φασματογραφήματος που δημιουργήθηκε από όλη την επεξεργασία που αναφέρθηκε παραπάνω. Υπήρχαν πάρα πολλές διαφορετικές προσεγγίσεις πάνω στην επεξεργασία, δίνοντας πολύ διαφορετικά αποτελέσματα στην οπτική των φασματογραφημάτων. Χρησιμοποιήθηκε η βιβλιοθήκη matplotlib για την παρουσίαση των φασματογραφημάτων. Δημιουργήθηκαν 2 συναρτήσεις, η μία εκ των οποίων δείχνει το φασματογράφημα του αρχείου που επεξεργάστηκε, και η άλλη, δείχνει επίσης το φασματογράφημα, αλλά με βάση τις διαφορές που υπήρξαν μεταξύ των διαδοχικών πλαισίων. Οι συναρτήσεις ονομάζονται **wave_plot** και **firdif** αντίστοιχα. Θα παρατηρηθεί προς τα τελευταία πλαίσια, μια εμφανής απώλεια τιμών, η οποία οφείλεται στην τεχνική zero padding που χρησιμοποιήθηκε ακριβώς πριν.

Στην **Εικόνα 2.10** παρουσιάζεται το φασματογράφημα του 1ο αρχείου των δεδομένων εκπαίδευσης, χρησιμοποιώντας την τεχνική τοπικών μεγίστων:

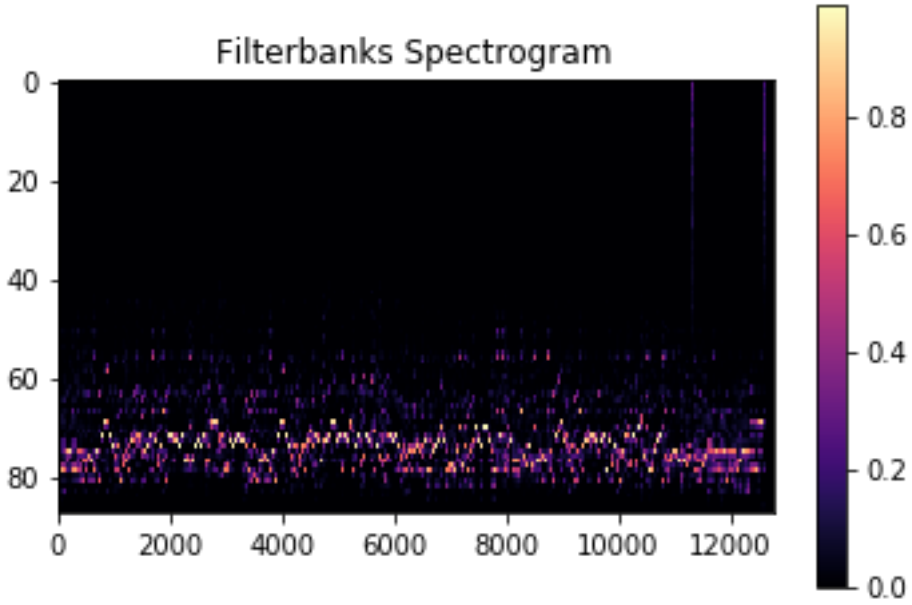


Εικόνα 2.10: Φασματογράφημα αρχείου, τεχνική τοπικών μεγίστων.

Στην **Εικόνα 2.11** παρουσιάζεται το ίδιο αρχείο, υπολογισμένα με την διαφορά πρώτης τάξης [36] :

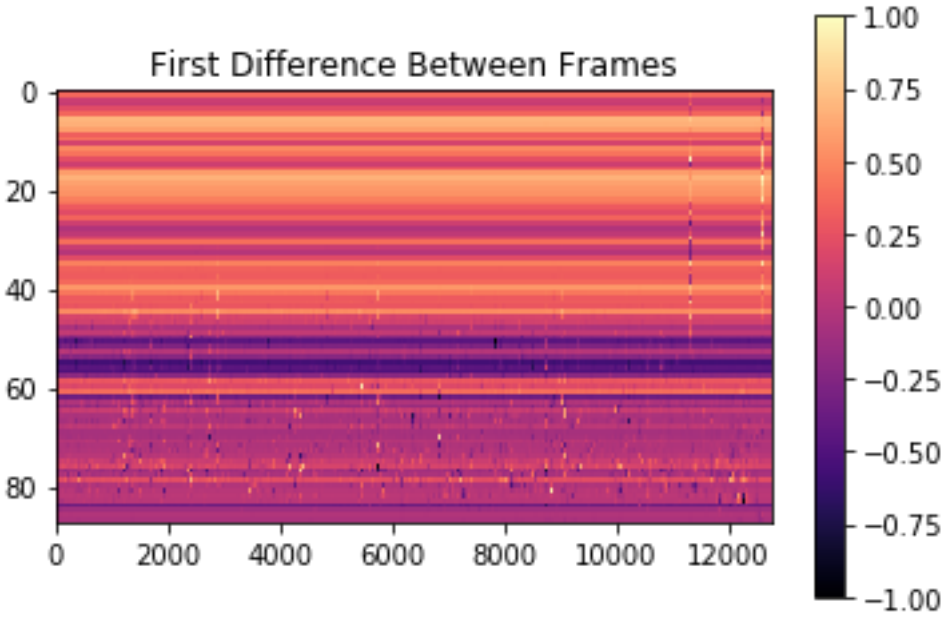


Εικόνα 2.11: Φασματογράφημα διαφορών 1ης τάξης πλαισίων, τεχνική τοπικών μεγίστων. Παρακάτω, το ίδιο αρχείο, προσαρμοσμένο με την τεχνική ομαλοποίησης:



Εικόνα 2.12: Φασματογράφημα Ομαλοποιημένο.

Και με εφαρμογή 1ης τάξης διαφοράς:

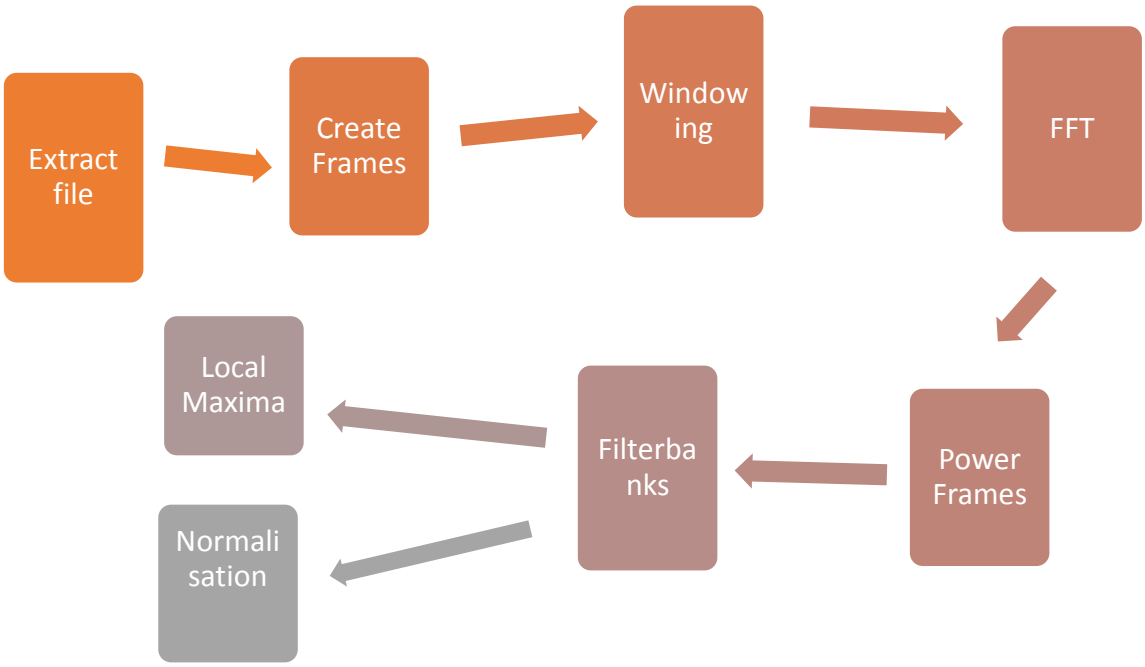


Εικόνα 2.13: Ομαλοποιμένο Φασματογράφημα διαφορών.

Να σημειωθεί πως, η 1η τάξη διαφοράς, χρησιμοποιήθηκε επιπρόσθετα μαζί με το φασματογράφημα, στο ίδιο δίκτυο, στην έρευνα [3].

2.4 Τελική Συνάρτηση

Η τελική συνάρτηση που καλέστηκε, ονομάστηκε **process.** Αυτή η συνάρτηση καλεί όλες τις προαναφερθείσες, παίρνοντας σαν είσοδο το όνομα του αρχείου και ποιο είδος παραθύρου θα εφαρμοστεί στον μετασχηματισμό, και παράγει τις τράπεζες φίλτρων.



Σχήμα 2.14 Διαδικασία παραγωγής φασματογραφήματος

Η συνάρτηση **process** χρησιμοποιεί τις παρακάτω συναρτήσεις:

1. extract
   - εξάγει τα δεδομένα του αρχείου.
2. create_frames
   - Παράγει πλαίσια βάσει παραθύρου.
3. spectrogram
   - Τράπεζες φίλτρων 88 νοτών = Έξοδος συνάρτησης.

Η έξοδος της συνάρτησης αυτής, είναι τα χαρακτηριστικά του αρχείου που θα τροφοδοτεί το νευρωνικό δίκτυο, τόσο για την εκπαίδευση, όσο και για την επαλήθευση μέτρησης σφάλματος.

ΚΕΦΑΛΑΙΟ 3

ΠΡΟΕΠΕΞΕΡΓΑΣΙΑ ΚΛΑΣΕΩΝ MIDI ΝΟΤΩΝ

3.1 Εισαγωγή

Σε αυτό το μέρος του κώδικα, **Midiz.ipynb,** δημιουργήθηκε λογισμικό τέτοιο, ώστε, διαβάζοντας ένα αρχείο .mid, να παράγονται ως έξοδοι πολλές χρήσιμες πληροφορίες. Το πρωτόκολλο midi αποτελείται από μυνήματα τα οποία έχουν σχεδιαστεί έτσι, ώστε να μπορούν ηλεκτρονικά όργανα (synthesizers, samplers, υπολογιστές) να διαβάζουν σειριακά και έπειτα να ηχούν κατάλληλα. Το ακρωνύμιο Midi, σημαίνει Musical Instrument Digital Interface. Τα midi αρχεία, δεν είναι αρχεία ήχου. Χρησιμοποιούνται για παραγωγή ήχου όμως, καθώς όργρανα όπως ηλεκτρονικά αρμόνια, ορίζοντας έναν καθορισμένο τρόπο αναπαράστασης των νοτών. Αυτό δίνει τεράστιο πλεονέκτημα από την αρχή της δεκαετίας του 1980, καθώς μέχρι τότε, το πρόβλημα της ασυμβατότητας μεταξύ μηχανών ήταν πολύ σύνηθες. Διάφορες εταιρίες όπως η Korg και η Roland, συνέβαλαν στην τελειοποίηση του πρωτοκόλλου. Χάρη σε αυτό, πλέον μπορεί κανείς να ηχογραφήσει, αποθηκεύσει και επεξεργαστεί μουσική που αποτελείται αποκλειστικά από συνθετικούς ήχους (synthesizers). Αυτό που επιτεύχθηκε στην επεξεργασία, είναι να αναπαραχθεί αρχείο .mid όπως το φασματογράφημα που αναλύθηκε στο Κεφάλαιο 2. Το μόνο που χρησιμοποιήθηκε για να υπάρξουν οι κλάσεις των πλαισίων, είναι ποια νότα υπάρχει στο κάθε πλαίσιο του midi πρωτοκόλλου.

3.2 Αρχείο Midiz.ipynb

Όπως αναφέρθηκε παραπάνω, χρειάστηκε η midi αναπαράσταση ενός μουσικού τραγουδιού, μαζί με την wave μορφή του, ώστε να «μάθει» το νευρωνικό δίκτυο την αλληλοσυσχέτιση μεταξύ των φασματογραφημάτων. Ο κώδικας του αρχείου **midiz.ipynb** δεν αρκέστηκε στην εξαγωγή πληροφοριών των δεδομένων αυτών. Δημιουργήθηκε μια εκτενής επεξεργασία midi αρχείου, δίνοντας πολλών ειδών δεδομένων. Δημιουργήθηκε επίσης συνάρτηση αναλογίας μεταξύ νοτών σε συχνότητα Hertz και αριθμού midi νότας. Καλώντας την συνάρτηση **midi_hertz()** τυπώνεται ένας πίνακας 2 διαστάσεων. Η $1^{η}$ στήλη είναι οι αριθμοί midi και η $2^{η}$ είναι οι νότες σε hertz. Είναι χρήσιμο να υπάρχει η αναλογία αυτή, καθώς υπάρχει μερική σύγχηση στο πρωτοκολλο midi, για το ποια είναι η νότα C4(Ντο της $4^{ης}$ οκτάβας). Κάποιοι θεωρούν τον αντίστοιχο αριθμό ίσο με 60, άλλοι με 72. Για να αποφευχθεί οποιαδήποτε σύγχηση λοιπόν, δόθηκε η συνάρτηση αυτή για να διευκρυνίσει οποιαδήποτε παρεξήγηση περί αντιστοιχίας νοτών. Παρακάτω το αποτέλεσμα αυτής της συνάρτησης:

Midi notes: ***Frequencies (Hertz)***

0	8.175798915643682
1	8.661957218027228
2	9.17702399741896
3	9.722718241315002
4	10.300861153527157
5	10.913382232281341
6	11.562325709738543
7	12.249857374429633
8	12.978271799373253
9	13.749999999999964
10	14.567617547440275
11	15.43385316425384
12	16.351597831287375
13	17.323914436054462
14	18.35404799483793
15	19.44543648263001
16	20.601722307054324
17	21.826764464562697
18	23.1246514194771
19	24.49971474885928
20	25.956543598746517
21	27.499999999999947
22	29.13523509488056
23	30.867706328507698
24	32.703195662574764
25	34.647828872108946
26	36.708095989675876
27	38.890872965260044
28	41.20344461410867
29	43.65352892912541
30	46.24930283895422
31	48.99942949771858
32	51.913087197493056
33	54.999999999999915
34	58.270470189761156
35	61.73541265701542
36	65.40639132514957
37	69.29565774421793
38	73.4161919793518
39	77.78174593052012
40	82.40688922821738
41	87.30705785825087
42	92.4986056779085
43	97.99885899543722
44	103.82617439498618
45	109.99999999999989
46	116.54094037952237
47	123.4708253140309
48	130.8127826502992
49	138.59131548843592
50	146.83238395870364

51	155.56349186104035
52	164.81377845643485
53	174.61411571650183
54	184.9972113558171
55	195.99771799087452
56	207.65234878997245
57	219.9999999999999
58	233.08188075904488
59	246.94165062806198
60	261.6255653005985
61	277.182630976872
62	293.66476791740746
63	311.1269837220808
64	329.62755691286986
65	349.2282314330038
66	369.99442271163434
67	391.99543598174927
68	415.3046975799451
69	440.0
70	466.1637615180899
71	493.8833012561241
72	523.2511306011974
73	554.3652619537443
74	587.3295358348153
75	622.253967444162
76	659.2551138257401
77	698.456462866008
78	739.988845423269
79	783.990871963499
80	830.6093951598907
81	880.0000000000003
82	932.3275230361803
83	987.7666025122488
84	1046.5022612023952
85	1108.7305239074892
86	1174.659071669631
87	1244.5079348883246
88	1318.5102276514808
89	1396.912925732017
90	1479.977690846539
91	1567.9817439269987
92	1661.218790319782
93	1760.000000000002
94	1864.6550460723618
95	1975.5332050244986
96	2093.0045224047913
97	2217.4610478149793
98	2349.3181433392633
99	2489.0158697766506
100	2637.020455302963
101	2793.8258514640347

102	2959.9553816930793
103	3135.963487853999
104	3322.437580639566
105	3520.0000000000055
106	3729.310092144725
107	3951.0664100489994
108	4186.009044809585
109	4434.922095629961
110	4698.636286678529
111	4978.031739553304
112	5274.040910605929
113	5587.651702928073
114	5919.910763386162
115	6271.926975708001
116	6644.875161279136
117	7040.000000000014
118	7458.620184289454
119	7902.132820098003
120	8372.018089619174
121	8869.844191259926
122	9397.272573357064
123	9956.063479106611
124	10548.081821211863
125	11175.303405856152
126	11839.82152677233
127	12543.853951416007

Σχήμα 3.1 Πίνακας αναλογίας midi / Hz

Όπως φάινεται και στον πίνακα παραπάνω, τελικά, στην εργασία αυτή χρησιμοποιείται η νότα C4 που ισούται με περίπου 261.6 Hz, ως η $60^{η}$ νότα midi.

Οι πληροφορίες των midi αρχείων που επεξεργάστηκαν και είναι εφικτό να καλεσθούν, αποτελούνται από:

- **Onsets** (Χρόνος που πατήθηκε το πλήκτρο)
- **Offsets** (Χρόνος που αφέθηκε το πλήκτρο)
- **Channels** (Αριθμός καναλιών, καθώς και ποιες νότες παράγει το καθένα)
- **Velocities** (Εντάσεις που πατήθηκε το πλήκτρο)
- **Time** (Χρόνος σε ticks. Τρόπος μέτρησης μουσικών αξιών)
- **Tempo** (Χρόνος περιγραμμένος σε χτύπους ανά λεπτό b.p.m.)
- **Notes** (Νότες σε όλο το μήκος του αρχείου midi)

Υπάρχει επίσης συνάρτηση **midi_onsets,** δίνοντας το ίδιο αποτέλεσμα με την έξοδο onsets της συνάρτησης **midi_data_extractor.**

Η εντολή **return** εξάγει:

1. onsets
   - χρόνοι ενεργοποίησης νοτών.
2. offsets
   - χρόνοι απενεργοποίησης νοτών.
3. notes
   - νότες ολόκληρου του αρχείου.
4. on_offs
   - Λίστα που δηλώνει πάτημα νότας με 1, και άφεση με 0.

Μπορούν να εξαχθούν περαιτέρω δεδομένα και χαρακτηριστικά, όπως αναφέρθηκαν παραπάνω. Η διαδικασία που ακολουθήθηκε για την επεξεργασία αυτών των δεδομένων ακολουθεί διάβασμα αρχείου σαν μεταβλητή τύπου απλού κειμένου (string) και έπειτα διασπάται ανάλογα τις θέσεις και τους διαχωρισμούς. Χάρη στο πρωτόκολλο midi, οι θέσεις των δεδομένων είναι τοποθετημένες με σειρά, δίνοντας εύκολη λειτουργία parsing.

Οι προαναφερθείσες λειτουργίες χαρακτηριστικών δεν χρειάστηκαν στην ανάπτυξη του μοντέλου, όμως θα μπορούσε να είναι πολύ χρήσιμες πληροφορίες για την μετέπειτα εξέλιξη της εργασίας σαν αντικείμενο.

3.3 Διαδικασία Εξαγωγής Κλάσεων

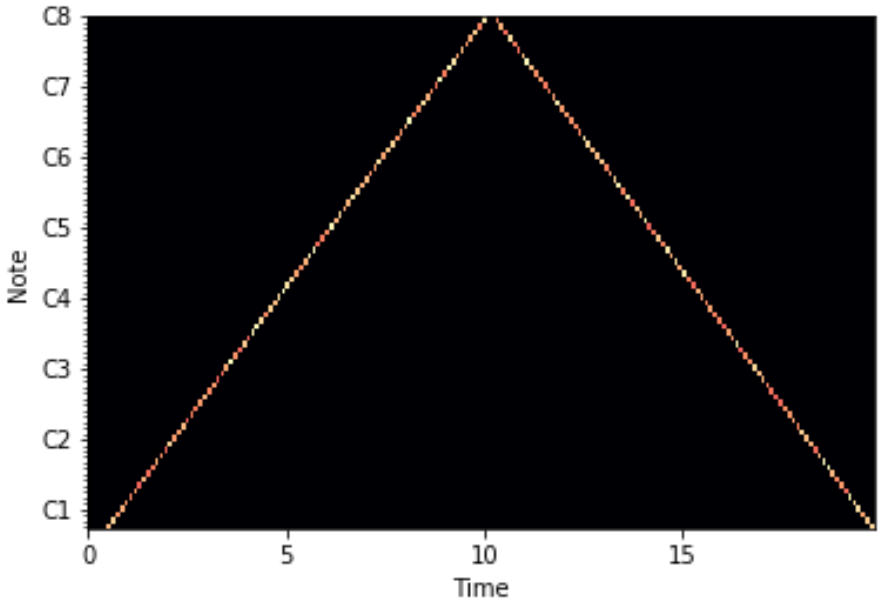
Για την ανάγνωση ενός αρχείου midi, χρειάστηκε η βιβλιοθήκη **Pretty_midi**. Συγκεκριμένα, η συνάρτηση όπου υλοποιήθηκε για την εξαγωγή των χαρακτηριστικών-στόχων για την εκπαίδευση του νευρωνικού δικτύου, ονομάστηκε **piano_roll** και χρειάστηκε τα εξής ορίσματα:

- **Midi_path :** Η τοποθεσία του αρχείου στον σκληρό δίσκο.
- **Number_of_frames :** Αριθμός πλαισίων που αποτελείται το αρχείο.
- **Start_pitch :** χαμηλότερη δυνατή νότα που θα παρθεί.
  Προκαθορισμένη τιμή: 21
- **End_pitch :** ψυλότερη δυνατή νότα που θα παρθεί.

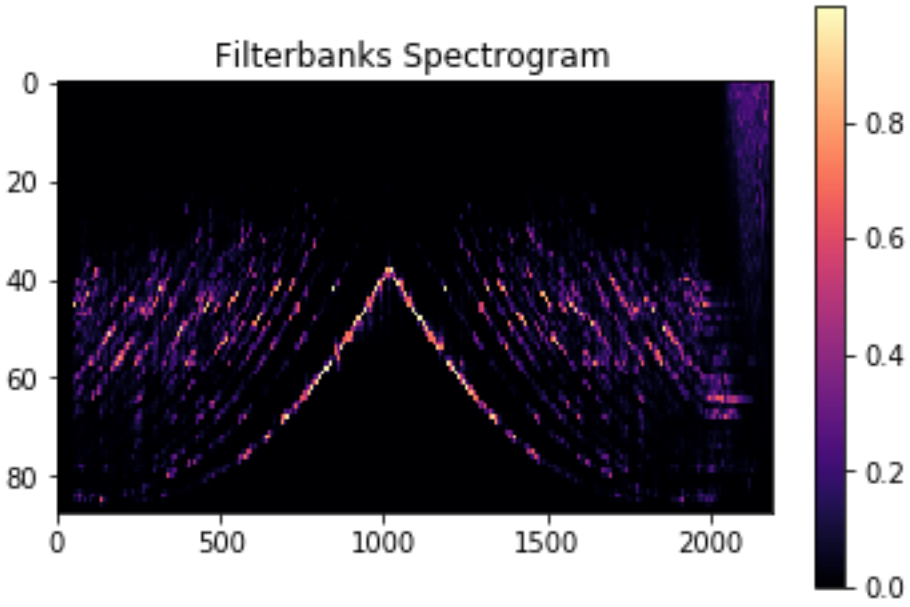
Προκαθορισμένη τιμή: 21+88 = 109

Χρειάστηκαν οι 2 τελευταίες μεταβλητές επειδή, παρατηρώντας και από τον πίνακα 3.1 πως υπάρχουν 127 νότες συνολικά. Χρειάζεται όμως, λόγω συμφωνίας διαστάσεων, να παρθούν μόνο 88 από αυτές τις νότες, καθώς οι τράπεζες φίλτρων ορίζουν τον αριθμό αυτό. Είναι προφανές πως για κάθε πιθανή κλάση πρέπει να υπάρχει αντίστοιχη τιμή στα χαρακτηριστικά προς εκπαίδευση, δίνοντας ακριβώς ίσες διαστάσεις. Άλλωστε, αντιλαμβανόμενος κανείς την έννοια της μετάφρασης, θα ήταν αδιανόητο να εκφραστεί λεξιλόγιο 88 χαρακτήρων σε 127, χωρίς συγκεκριμένη μετάθεση. Έτσι λοιπόν, αποφασίστηκε να δοθεί ως όρισμα στην συνάρτηση αυτή, σε περίπτωση που κάποιος έχει επιθυμία να μετακινήσει το παράθυρο νοτών που προβλέπονται.

Το πρόβλημα της μετάφρασης μεταξύ wave και midi αρχείων, μπορεί να γίνει ευκολότερα αντιληπτό, παρατηρώντας τα φασματογραφήματα ενός αρχείου, και στις 2 μορφές του. Παρακάτω ένα παράδειγμα του αρχείου MAPS_AkPnBcht_1\AkPnBcht\ISOL\CH\MAPS_ISOL_CH0.1_F_AkPnBcht.midi και .wav αντίστοιχα:



Εικόνα 3.1.1: Απεικόνιση νοτών midi του αρχείου.



Εικόνα 3.1.2: Φασματογράφημα ίδιου αρχείου (μορφή wave).

Οπτικά και μόνο, μπορεί να εντοπιστεί η ομοιότητα των δύο απεικονίσεων. Δίνοντας το 2ο σαν χαρακτηριστικά και το 1ο σαν ταμπέλες αντίστοιχα, καθίσταται δυνατό να εφαρμοστεί αλγόριθμος νευρωνικού δικτύου, όπως θα αναλυθεί στο επόμενο κεφάλαιο.

3.3.1 Συνάρτηση Piano_roll

Δημιουργήθηκε η συνάρτηση **piano_roll** για τον λόγο ότι υπάρχει για αντικείμενα τύπου pretty_midi, έτοιμη συνάρτηση που εξάγει τις νότες ανά χρονική στιγμή. Έτσι, λαμβάνεται ένας πίνακας, ανάλυσης 100 πλαισίων ανά δευτερόλεπτο (100 f.p.s.) όπως προτείνεται και στην [3]. Έπειτα, οι τιμές των νοτών που εμφανίζονται είναι της μορφής:

```
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0
. 0. 0. 0. 0. 0. 0. 0. 0. 0. 52. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0.]
```

Όπου είναι οι τιμές των midi νοτών που επιλέχθηκαν, συνολικού πλήθος 88. Η ύπαρξη ενός μη μηδενικού αριθμού, σηματοδοτεί την ύπαρξη νότας στην εκάστοτε τράπεζα. Επομένως, στο παράδειγμα παραπάνω, εφόσον η νότα με τον αριθμό 52 βρίσκεται στην θέση 31 (με αρχικό δείκτη = 0), θα τοποθετηθεί στην τράπεζα φιλτρων, ως ύπαρξη αυτής της κλάσης. Να σημειωθεί πως αυτός ο πίνακας αναφέρεται μόνο σε 1 πλαίσιο (frame). Επίσης, ο αριθμός κάθε νότας, δεν είναι τυχαίος. Είναι η αριθμημένη midi νότα που υπάρχει βάσει πρωτοκόλλου. Όμως, επειδή χρειάζονται 88 από τις 127 νότες, και τέθηκε όριο [21,109], δίνεται στην 31η θέση η τιμή νότες 31 + 21 = 52. Έτσι, έχοντας τις ταμπέλες για κάθε πλαίσιο, δόθηκαν όλες οι προβλεπώμενες κλάσεις για κάθε wave αρχείο χαρακτηριστικών. Όμως, για την εκπαίδευση του δικτύου, χρειάζεται να μετατραπούν όλες οι νότες των πλαισίων, σε τιμές ίσες με 1. Δηλαδή, όπου η τιμή είναι μη μηδενική (προφανώς δεν υπάρχουν αρνητικές τιμές), θα

αντικαταστείται η τιμή με την τιμή 1. Έτσι, θα υπάρχει ευκολότερη εκπαίδευση, και δίνει την δυνατότητα να συγκλίσει το δίκτυο χωρίς να υπερπροσαρμοστεί από μια συγκεκριμένη νότα. Άλλωστε, το τελικό αποτέλεσμα αποτελείται από την αναπαράσταση σε φασματογράφημα.

ΚΕΦΑΛΑΙΟ 4

ΑΡΧΙΤΕΚΤΟΝΙΚΉ ΚΑΙ ΕΚΠΑΊΔΕΥΣΗ ΑΝΑΔΡΟΜΙΚΟΎ ΝΕΥΡΩΝΙΚΟΎ ΔΙΚΤΎΟΥ

4.1 Εισαγωγή

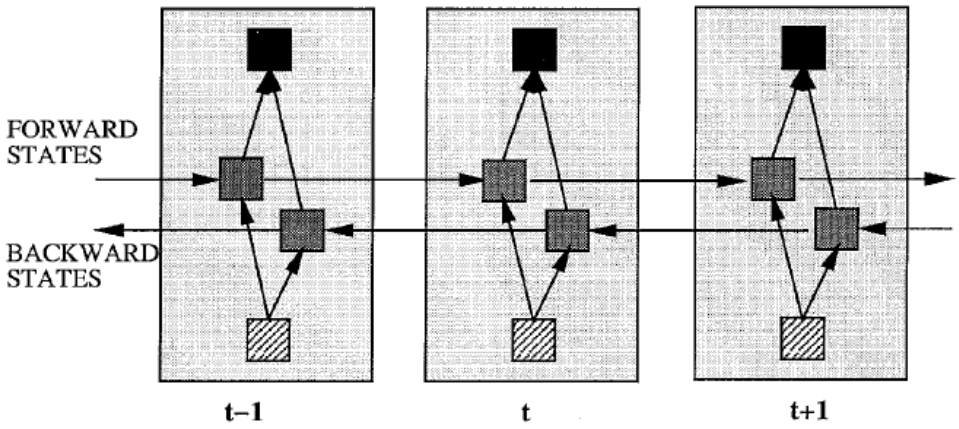
Η μηχανική μάθηση σαν αντικείμενο, την τελευταία δεκαετία έχει αναπτυχθεί με ραγδαίες ταχύτητες. Πλέον, είναι δυνατό να εφαρμοστούν αλγόριθμοι που, ήταν εφικτοί μόνο στην θεωρία. Πολλές τεχνικές και αρχιτεκτονικές των νευρωνικών δικτύων δημιουργήθηκαν τον προηγούμενο αιώνα, σε συνδυασμό με την υπολογιστική δύναμη και την άφθονη φαινομενικά ποσότητα δεδομένων, είναι πλέον εφικτό να εφαρμοστούν, ακόμα και σε προσωπικό υπολογιστή. Ειδικά στην περίπτωση των νευρωνικών δικτύων, σε αντίθεση με άλλες τεχνικές όπως Naive Bayes [37], K κοντινών γειτόνων [38], τα νευρωνικά δίκτυα απαιτούν πολυπληθείς πολλαπλασιασμούς, τάξης εκατομμυρίων, ώστε να γίνουν οι πράξεις των πινάκων τους. Έτσι, έχουν αναπτυχθεί κάρτες γραφικών δίνοντας την δυνατότητα παράλληλης επεξεργασίας αυτών των πολλαπλασιασμών. Το πλεονέκτημα είναι οτι ο χρόνος εκτέλεσης μειώνεται δραματικά, αν σκεφτεί κανείς πως, σε ένα από τα πολλά πειράματα αυτής της εργασίας, 8 ώρες εκπαίδευσης σε CPU των 2.4Khz, δίνεται σε λιγότερο από 1 ώρα μέσω GPU, μεσσαίας κατηγορίας.

Στην εργασία που υλοποιήθηκε, χρησιμοποιήθηκαν οι τεχνικές των Αναδρομικών Νευρωνικών Δικτύων. Σε αντίθεση με τα κλασικά εμπροσθοτροφοδοτούμενα δίτκυα, δίνεται η δυνατότητα να δέχεται ως είσοδο το δίκτυο, την ίδια του την έξοδο. Αυτό γίνεται διακρίνοντας χρονικές στιγμές t_i , περνόντας έτσι με αναδρομικό τρόπο την έξοδο του προηγούμενου στιγμιότυπου, στην είσοδο του επόμενου. Αυτό δίνει το πλεονέκτημα στο δίκτυο, να αποκτήσει μια «συνδυαστική σκέψη» με βάση την σειρά που τροφοδοτείται. Η δύναμη της μνήμης που δημιουργείται έτσι , περνάει και στις επόμενες τιμές που δίνονται, επηρεάζοντας την τελική τιμή εξόδου.

Τα αναδρομικά δίκτυα, όπως όλα τα δίκτυα, έχουν τις αδυναμίες τους. Συγκεκριμένα, στα αναδρομικά δίκτυα στην απλή τους μορφή, ονόματι RNN (Recurrent Neural Network) [39] , [40], αντιμετωπίζουν το πρόβλημα της εκτόξευσης ή εξαφάνισης της κλίσης (Exploding/Vanishing Gradient). Η έρευνα [41] εξηγεί αναλυτικά, με πολύ σαφή τρόπο, το πως και γιατί συμβαίνει αυτό. Περιληπτικά, αυτό που συμβαίνει είναι το εξής:
Καθώς η σειρά αναδρομής μεγαλώνει, τα βάρη W_i των εκάστοτε χρονικών στιγμών αυξάνονται σε πλήθος. Έχοντας μια τυχαία αρχικοποίηση των βαρών, δίνεται πολύ μικρή αλλαγή στα αρχικά βάρη, και ταυτόχρονα πολύ μεγάλη αλλαγή στα τελευταία επίπεδα. Έχοντας, λοιπόν, τους συντελεστές αυτούς, όταν εφαρμοστεί η οπισθοδιάδοση (backpropagation) [42] στο κόστος που υπολογίστηκε, καθώς επιστρέφει στα αρχικά επίπεδα του δικτύου. Σε περίπτωση που το μέτρο της τιμής εξόδου επί το βάρος της είναι μικρότερο από 1, υπάρχει τελικός συντελεστής στην κλίση μικρότερη του 1. Όταν αυτό επαναλαμβάνεται για πολλά επίπεδα του δικτύου, οι συνεχόμενοι πολλαπλασιασμοί με τιμές υπό του 1, παράγει

μια τελική τιμή πάρα πολύ μικρή, ανύμπορη να συγκλίνει κάπου σε μετέπειτα στάδιο. Το ίδιο συμβαίνει, αν πολλές τιμές είναι άνω του 1, δίνοντας τεράστιες τιμές, αντίστοιχα.

Στην εργασία αυτή, χρησιμοποιήθηκε και η δυνατότητα αμφίδρομης τροφοδότησης. Τέτοια δίκτυα ονομάζονται Bidirectional Neural [40]. Η αμφίδρομη τροφοδότηση χωρίζει τους νευρώνες κάθε επιπέδου, ώστε το ένα μέρος να δέχεται την χρονική σειρά των δεδομένων προς τον θετικό άξονα του χρόνου ($t_i < t_{i+1}$), και το άλλο μέρος, να δέχεται σε ανάποδη φορά. Οι έξοδοι κάθε φοράς, δεν συνδέονται ως είσοδοι στην άλλη, σε αντίθεση με το κλασικό αναδρομικό δίκτυο. Το παρακάτω σχήμα, παρμένο από [40], δείχνει την ροή ενός απλού δικτύου BRNN :



Σχήμα 4.1: Βάρη δικτύου σε αμφίδρομη ροή.

- Τα διακεκομμένα σύμβολα αποτελούν τις εισόδους δεδομένων στο δίκτυο.
- Τα μαύρα σύμβολα αποτελούν τις εξόδους του δικτύου.
- Τα ενδιάμεσα σύμβολα αποτελούν τους νευρώνες των κρυφών επιπέδων του δικτύου.

4.2 Backpropagation Through Time (BPTT)

Οι συζητήσεις περί του αλγόριθμου αυτού στα αναδρομικά νευρωνικά δίκτυα, άρχισαν στην δεκαετία του 80. Η φανταστική έρευνα [43] παρουσίασε τα προβλήματα που αντιμετωπίζει ένα τέτοιο δίκτυο. Συγκεκριμένα, το μεγαλύτερο πρόβλημά του, είναι πως χρειάζεται τεράστια ποσότητα μνήμης, ώστε να κρατηθούν όλες οι διαδοχικές αλλαγές ανά χρονική στιγμή, καθώς αλλάζουν οι τιμές των βαρών, για κάθε νευρώνα, για κάθε επίπεδο. Ειδικά στην εποχή που πρωτοεμφανίστηκαν οι τεχνικές αυτές, η υπολογιστική δύναμη που χρειαζόταν ένα τέτοιο σύστημα, ήταν, αν όχι αδύνατο, πολύ δύσκολο να υλοποιηθεί. Η

κατανάλωση μνήμης αποτελεί πρόβλημα ακόμα και την σημερινή εποχή, με προτεινόμενες λύσεις όπως [44].

Σε ένα κλασικό εμπροσθοτροφοδοτούμενο δίκτυο (feedforward net),έστω 2 κρυφών επιπέδων, η συνάρτηση που προσπαθεί να συγκλίνει στις τιμές στόχους μέσω των εισαχθέντων δίνεται από την συνάρτηση:

$$y(t) = G\left(F\left(x(t)\right)\right)$$

Όπου:

- y είναι η έξοδος,
- x το επίπεδο εισόδου,
- t το δείγμα,
- F,G τα κρυφά επίπεδα του δικτύου (συναρτήσεις ενεργοποίησης).

Στην περίπτωση ενός αναδρομικού δικτύου, υπάρχει μια βασική διαφορά. Δεν λειτουργούν μόνο με τις τιμές εισόδου, αλλά και με το τι έχουν λάβει ως είσοδο μέχρι την χρονική στιγμή που βρίσκονται. Η δεύτερη λειτουργία, δίνει την δυνατότητα μάθησης του δικτύου αξιοποιώντας τις συσχετίσεις των εισαχθέντων τιμών κατά την διάρκεια προσπέλασης των διακριτών ή και συνεχόμενων χρονικών στιγμών. Αυτό μπορεί να εκφραστεί ως εξής:

$$y(t) = G\big(s(t)\big)$$

με

$$s(t) = F\big(s(t-1), x(t)\big)$$

Όπου:
s η στάση του δικτύου την χρονική στιγμή t.

Έτσι, η οπισθοδιάδοση (Backpropagation) ενός απλού εμπροσθοτροφοδοτούμενου δικτύου, για κάθε επίπεδό του, δίνεται από την συνάρτηση:

$$net_i(t) = \sum_{i}^{n} x_i(t) n_{ji} + b_j$$

Δίνοντας:

$$y_i(t) = f\left(net_j(t)\right)$$

Όπου:

- i είναι ο κόμβος εισόδου,
- j, h ο κρυφός κόμβος,
- n, πλήθος εισόδων,
- b, σταθερά (bias),
- f, συνάρτηση εξόδου (ολοκληρώσιμη).

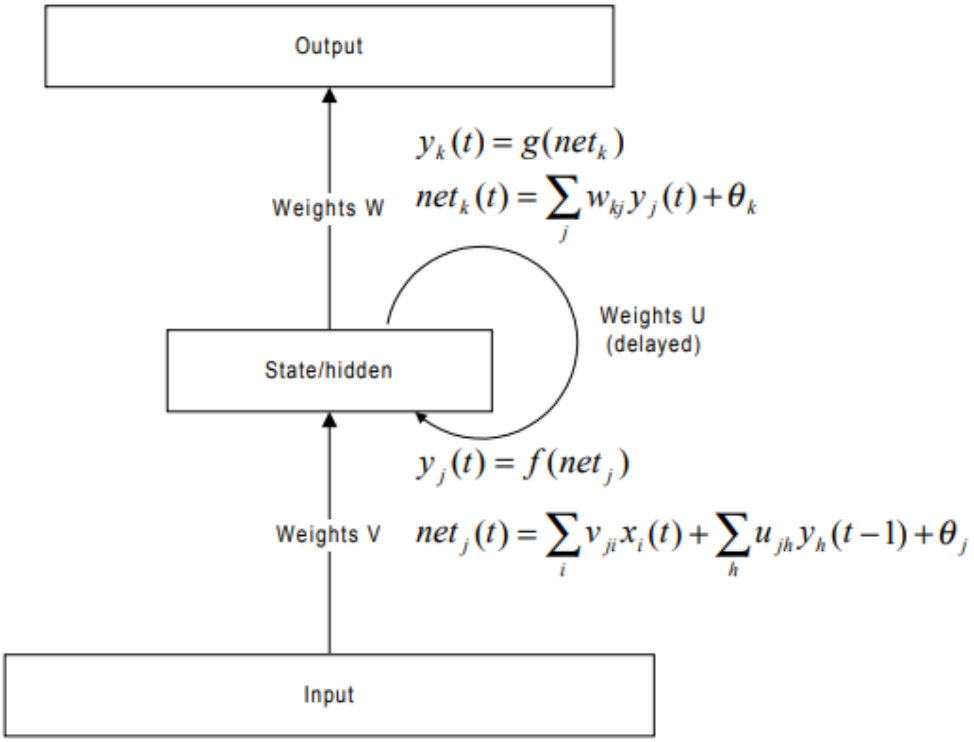
Ομοίως, ένα αναδρομικό δίκτυο, διαδίτεαι προς τα βάρη, με την προσθήκη της προηγούμενης κατάστασης στην τιμή:

$$net_k(t) = \sum_j^m y_j(t) w_{ji} + \sum_h^m y_h(t-1) u_{jh}) + b_j$$

με

$$y_i(t) = g\big(net_i(t)\big)$$

Σχηματικά, παράγεται το παρακάτω:



Σχήμα 4.2: Αναδρομικό δίκτυο.

Καθώς, λοιπόν, πρέπει να βρεθεί το τοπικό ελάχιστο του κόστους, τα βάρη πρέπει να λάβουν όσο πιο ακριβείς τιμές και διορθώσεις είναι δυνατόν. Έτσι, έχοντας ως κόστος ορισμένο μια C μεταβλητή, ο ρυθμός μεταβολής των βαρών, ακολουθεί:

$$\Delta(w) = -\eta \frac{\partial C}{\partial w}$$

Χρησιμοποιώντας τον κανόνα της αλυσίδας (chain rule), είναι εφικτό να υπολογιστούν όλα τα βάρη, σε κάθε επίπεδο του δικτύου. Ας οριστεί:

$$\delta = -\frac{\partial C}{\partial net}$$

Το διάνυσμα του κόστους, για κάθε κόμβο στο δίκτυο.
Για την έξοδο ισχύει:

$$\delta_{pk} = -\frac{\partial C}{\partial w}\frac{\partial y_{pk}}{\partial net_{pk}}$$

Για τους κρυφούς κόβμους ισχύει:

$$\delta_{pj} = -\sum_{k}^{O}\frac{\partial C}{\partial y_{pk}}\frac{\partial y_{pk}}{\partial net_{pk}}\frac{\partial net_{pk}}{\partial s_{pj}}\frac{\partial s_{pj}}{\partial net_{pj}} = \sum_{k}^{O}\delta_{pk}w_{kj}f'\left(net_{pj}\right)$$

Όπου:

- j = δείκτης κρυφού επιπέδου χρονικής στιγμής t,
- k = δείκτης εξόδου.

Εύκολα μπορεί να υπολογίσει κανείς την αλλαγή του βάρους

$$\Delta w_{ji} = \eta\sum_{p}^{n}\delta_{pk}s_{pj}$$

Όπου s(t) = κρυφό επίπεδο χρονικής στιγμής t.
Σε ένα αναδρομικό νευρωνικό δίκτυο, τα κόστη μπορούν να διαδοθούν σε μεγαλύτερο πλήθος, ώστε να κατανοήσουν πιθανές σχέσεις μεταξύ των πληροφοριών διαδοχικών πλαισίων εισόδου. Καθώς το ξεδίπλωμα (unfolding) αποτελεί τεχνική μόνο για τα δίκτυα αυτά, τα βάρη κάθε χρονικής στιγμής απλώς κλονοποιούνται, δίνοντας μια αντίστοιχη αναπαράσταση του δικτύου, ως ένα κλασικό εμπροσθοτροφοδοτούμενο δίκτυο. Έτσι, το κόστος μεταδίδεται:

$$\delta_{pj}(t-1) = \sum_{h}^{m}\delta_{ph}(t)u_{hj}f'(s_{pj}(t-1))$$

Όπου:

- h = δείκτης κρυφού κόμβου χρονικής στιγμής t,
- j = δείκτης κρυφού κόμβου χρονικής στιγμής t-1

Προφανώς, το παραπάνω άθροισμα αναφέρεται μόνο στο ξεδίπλωμα 2 χρονικών στιγμών. Σε περίπτωση που χρειαστεί μεγαλύτερη χρονική στιγμή εξαρτητοποίησης, όπως στην εργασία που χρησιμοποιούνται 50 πλαίσια, ισχύει ο τύπος αναδρομικά.
Εφόσον όλα τα σφάλματα υπολογιστούν, τα βάρη που αντιστοιχούν στον ίδιο κόμβο, αθροίζονται, δίνοντας την τελική κατάληξη του βάρους. Δηλαδή, υπολογίζεται η αλλαγή για το κάθε βάρος:

$$W_{node} = \sum_{i=1}^{T} w_{i\,,node}$$

Όπου:

- W = τελικό βάρος του κόμβου node,
- T = μέγεθος ξεδιπλώματος δικτύου (πόσες χρονικές στιγμές προσμετρώνται)

Η ενστικτώδης εξήγηση της οπισθοδιάδοσης είναι ότι, για κάθε βάρος, υπολογίζει το κόστος μέσω μιας συνάρτησης (απόκλιση μεταξύ τιμής που προβλήφθηκε και πραγματικής) σύμφωνα με το βάρος αυτό, και έπειτα αλλάζει την τιμή του βάρους κατάλληλα. Στην εργασία αυτή, χρησιμοποιήθηκαν 2 συναρτήσεις κόστους, για χάρη πειραματισμού, που θα αναλυθούν παρακάτω. Η ανάλυση παρακάτω, μπορεί να καλυφθεί και από [44], δίνοντας μια εκτενή εξήγηση.

4.2.1 Συναρτήσεις κόστους

Οποιοδήποτε δίκτυο χρησιμοποιηθεί, χρειάζεται να έχει συναρτήσεις που υπολογίζουν το σφάλμα, αλλιώς, το κόστος. Οι συναρτήσεις αυτές, ποσοτικοποιούν το μέτρο του σφάλματος, δίνοντας μια μέτρηση στο δίκτυο, ώστε να γνωρίζει κατά πόσο σωστή ήταν η πρόβλεψή του. Ο υπολογισμός κόστους, συμβαίνει στο τέλος κάθε δείγματος εκπαίδευσης, δηλαδή, αφού κάνει το δίκτυο την εμπροσθοδιάδοση (forward pass). Σε εκείνο το σημείο, υπάρχει η προβλεπόμενη τιμή βάσει των βαρών που έχουν δοθεί μέχρι στιγμής. Η προβλεπόμενη τιμή ορίζεται ως y_p και η αληθινή τιμή αυτού ως y_{true} . Προφανώς οι ονομασίες μπορούν να αλλάξουν, αποτελούν καθαρά ορισμούς για χάρη χρηστικότητας. Υπάρχουν πολλες συναρτήσεις που χρησιμοποιούνται. Οι πιο γνωστές συναρτήσεις είναι

- αυτή του μέσου τετραγωνικού σφάλματος (Mean Squared Error M.S.E.), η οποία υπολογίζει μια ευκλείδια απόσταση,
- η συνάρτηση Εντροπίας (Cross Entropy), υπολογίζοντας πιθανοτητες,
- η Hinge Loss, η οποία δίνει μια ιδιαίτερη προσέγγιση [45].

όπως και πολλές παραπάνω. Οι συναρτήσεις δοκιμάζονται εμπειρικά σε ένα δίκτυο. Σίγουρα υπάρχει κάποια προτίμηση επιλογής, σύμφωνα με το είδος προβλήματος που επιλύει το νευρωνικό δίκτυο. Στο πρόβλημα της εργασίας, χρησιμοποιήθηκαν οι 2 πρώτες αναφερόμενες συναρτήσεις. Η τετραγωνικού σφάλματος προτάθηκε από τον επιβλέπων καθηγητή, και η δυαδική Εντροπία (Binary Cross Entropy) προτάθηκε από την [3], καθώς χρησιμοποιήθηκε και εκεί.
Η συνάρτηση κόστους που ονομάζεται Mean Squared Error, ορίζεται ως εξής:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}\left(y_{true\ i} - y_{p\ i}\right)^2$$

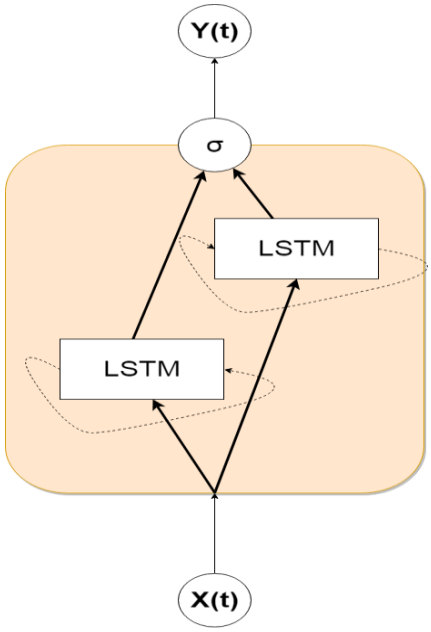
Ο υπολογισμός αυτός, στο τέλος της εμπροσθοδιάδοσης, θα δώσει την σειρά του στην οπισθοδιάδοση, υπολογίζοντας τις μερικές παραγώγους για κάθε επίπεδο του δικτύου, όπως αναλύθηκε παραπάνω.

Η συνάρτηση κόστους που ονομάζεται Cross Entropy, ορίζεται ως εξής:

$$CE = \frac{1}{N}\sum_{i=1}^{N} y_{true\,i}\,\log\left(y_{p\,i}\right) + \left(1 - y_{true\,i}\right)\log\left(1 - y_{p\,i}\right)$$

Παραδόξως, η συνάρτηση αυτή, χρησιμοποιείται και στην δυαδική της μορφή (N = 2) σε προβλήματα που αφορά 2 κλάσεις. Η έρευνα [3] χρησιμοποιεί ακριβώς αυτή την τη συνάρτηση, παρόλο που το πρόβλημα αναθέτει 88 πιθανές κλάσεις.

4.2.2 Long Short Term Memory

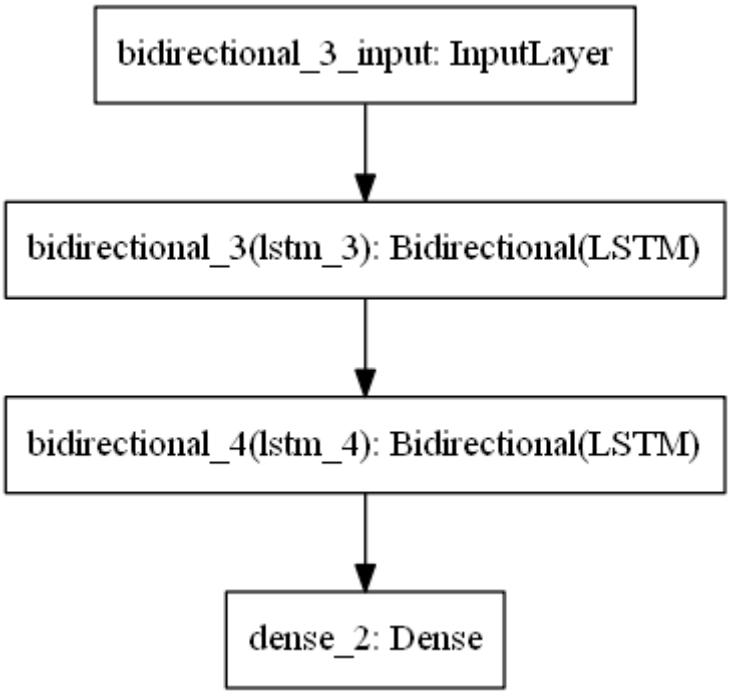


Εικόνα 4.2: BiLSTM δίκτυο, χρονικής στιγμής t.

4.3 Γενικά χαρακτηριστικά αρχιτεκτονικής δικτύου

Η έρευνα [3] ακολουθήθηκε ως προς την αρχιτεκτονική που χρησιμοποιείται. Έπειτα, με πολλές δοκιμές μέσω εκπαιδεύσεων, αποφασίστηκαν συγκεκριμένες ρυθμίσεις για το δίκτυο. Χρησιμοποιήθηκε ένα αμφίδρομο αναδρομικό δίκτυο, τύπου LSTM. Ο λόγος είναι πως, καθώς χρησιμοποιούνται σειριακά δεδομένα ως είσοδος, είναι χρήσιμη η αναδρομική λειτουργία. Επίσης, για να καταγραφηθεί η εξάρτηση των νοτών μέσα σε ένα μουσικό τραγούδι, η αμφίδρομη προσπέλλαση στο δίκτυο δίνει βαθύτερη κατανόηση στο αντικείμενο που επιλύει το δίκτυο. Τέλος, επιλέχθηκε η ιδιαίτερη αρχιτεκτονική LSTM [12] καθώς, όπως προαναφέρθηκε και παραπάνω, επιλύει το πρόβλημα της εξαφάνισης της κλίσης (Vanishing Gradient) μέσω των πυλών. Είναι απαραίτητη η χρήση ενός LSM έναντι κλασικών δικτύων, καθώς η είσοδος στο δίκτυο είναι μακροχρόνια. Συγκεκριμένα η είσοδος αποτελείται από 32 επικαλυπτώμενα(κατά 50%) πακέτα (μέγεθος ενός batch = 32), όπου το καθένα αποτελείται από 50 πλαίσια των 88 χαρακτηριστικών προς μάθηση. Αυτό, σε συνδυασμό με πολλά επίπεδα στο δίκτυο, θα ήταν αδύνατο να υπάρξει σύγκλιση στην λύση, λόγω εμφάνισης του

προβλήματος εξαφάνισης κλίσης (Vanishing Gradient). Παρακάτω, η αρχιτεκτονική του δικτύου που ακολουθήθηκε:



Εικόνα 4.3: Επίσημη δομή δικτύου, βάση keras κώδικα.

Το σχήμα που παρουσιάστηκε παραπάνω, δημιουργήθηκε αυτόματα από τον κώδικα που αναπτύχθηκε, χρησιμοποιώντας την βιβλιοθήκη keras, με την εντολή keras.utils.plot_model, η οποία δέχεται είσοδο ένα μοντέλο, και αποθηκεύει ως εικόνα το αποτέλεσμα. Χρειάζεται το λογισμικο Graphviz [46] για να σχεδιαστεί και να προσπελαστεί το πρόγραμμα δημιουργίας του γράφου αυτού.

Όπως φαίνεται, παραπάνω, η είσοδος το δίκτυο αποτελείται από χαρακτηριστικά διαστάσεων 32,50,88 που ισούνται με το μέγεθος της παρτίδας, τα πλαίσια για την χρονική κατανόηση, και οι τιμές των νοτών σύμφωνα με το φασματογράφημα του wave αρχείου, ανίστοιχα.

Έπειτα, επιλέχθηκαν διάφορες ρυθμίσεις για το δίκτυο. Αρχικά, ακολουθήθηκε η πρόταση της [3] με 2 κρυφά επίπεδα, των 100 νευρώνων το καθένα. Η απόφαση για το μέγεθος των νευρώνων, έγκειται στην ανάγκη του δικτύου να μπορέσει να «περιγράψει» το πρόβλημα προς επίλυση τόσο αναλυτικά ώστε να υπάρξει κατανόηση, αλλά και όσο το δυνατόν πιο απλοϊκά γίνεται, έτσι ώστε οι λύση να συγκλίνει. Δηλαδή, θεωρητικά, όσο περισσότερο αυξάνεται το πλήθος των νευρώνων και των κρυφών επιπέδων, αυξάνεται και η δυνατότητα ανάλυσης ενός προβλήματος, αλλά αυξάνεται και ο χρόνος σύγκλισής του. Ένα πολύ απλό δίκτυο, δεν θα μπορεί να καταλάβει μια πιο περίπλοκη ταξινόμηση, όμως ένα υπερβολικά μακροσκελές και βαθύ δίκτυο, δεν δίνει απαραίτητα πάντα καλύτερες λύσεις. Στην συνέχεια, προτάθηκε από τον επιβλέποντα καθηγητή, να χρησιμοποιηθεί μόνο 1 κρυφό επίπεδο δικτύου, και με μειωμένους νευρώνες κατά το ήμισυ. Η εκπαίδευση αυτού του δικτύου, αποφάνθηκε πιο γρήγορη προφανώς, σίγουρα όμως είχε μικρότερη ακρίβεια μετά τις

10 εποχές που εκπαιδεύτηκε. Η απόφαση των ρυθμίσεων αυτών, οφείλεται σε εμπειρικούς κανόνες, και στην αγγλική ορολογία ονομάζεται fine tuning.

Η έξοδος του δικτύου, αποτελείται από 88 νευρώνες. Ο προφανής λόγος είναι πως προβλέπονται 88 πιθανές κλάσεις, επομένως, χρειάζονται 88 διαφορετικοί νευρώνες για να ενεργοποιηθούν όποτε υπάρχει νότα για την κάθε πρόβλεψη. Η συνάρτηση ενεργοποίησης για την έξοδο του δικτύου που χρησιμοποιήθηκε, ήταν η Σιγμοϊδής συνάρτηση (Sigmoid), καθώς παράγει τιμές εύρους [0,1]. Είναι αναγκαία η χρήση αυτής της συνάρτησης, καθώς αν θυμηθεί κανείς τις ταμπέλες που δόθηκαν από τα midi αρχεία, η αναπαράσταση ύπαρξης νότας ισούται με 1, ειδαλλιώς με 0.

Χρησιμοποιήθηκε η βιβλιοθήκη Keras [47] για την εύκολη κατασκευή της αρχιτεκτονικής. Χρησιμοποιήθηκε επίσης, κανονικοποίηση των τιμών, όπως και τυχαία παύση των νευρώνων (dropout), το οποίο βοήθησε στην εκμάθηση του δικτύου, υπό την έννοια της αποτροπής της υπερπροσαρμογής (overfitting) στα δεδομένα εκπαίδευσης (training dataset).

4.3.1 Δεδομένα

Καθώς χτίστηκε η βάση του δικτύου, χρειάστηκε να εκπαιδευτεί πάνω στη βάση δεδομένων MAPS [11] . Δεν χρησιμοποιήθηκε ολόκληρη η βάση δεδομένων στην παρούσα εργασία. Τα δεδομένα έπρεπε, όπως σε κάθε νευρωνικό δίκτυο, να χωριστούν σε δεδομένα εκπαίδευσης, και δεδομένα επαλήθευσης. Φυσικά, όσα αρχεία δεν χρησιμοποιήθηκαν σε αυτές τις 2 ομάδες, χρησιμοποιήθηκαν ως δοκιμές, μετά την εκπαίδευση του δικτύου. Συγκεκριμένα, επιλέχθηκαν μόνο τα αρχεία τα οποία ήταν μουσικά τραγούδια. Υπήρχαν διάφορα άλλα αρχεία που παραήγαγαν μεμονομένες νότες ή συγχορδίες, μήκους 1-5 δευτερολέπτων (στην μορφή wave). Αυτά τα αρχεία δεν χρησιμοποιήθηκαν ούτε στην εκπαίδευση, ούτε στην επαλήθευση, καθώς νοητικά δεν συμφωνούσαν στην αναπαράστασή τους ως midi, γιατί το δίκτυο έπρεπε να εκμεταλευτεί την λογική συνέχεια ενός μουσικού τραγουδιού. Έτσι, για την εκπαίδευση επιλέχθηκαν οι φάκελοι

- MAPS_AkPnBcht_2
- MAPS_AkPnBsdf_2
- MAPS_AkPnCGdD_2
- MAPS_AkPnStgb_2
- MAPS_ENSTDkAm_2
- MAPS_ENSTDkCl_2
- MAPS_SptkBGCl_2

και μόνο τους υποφακέλους με το όνομα **MUS.** Κάθε φάκελος περιείχε 90 συνολικά αρχεία. 30 εκ των οποίων ήταν αρχεία wave, και υπήρχαν αντίστοιχα 30 midi αρχεία, όπως 30 text αρχεία που έδειχνα τις πληροφορίες των midi αρχείων. Επομένως, κάθε φάκελος περιείχε 30 τραγούδια wave προς εκπαίδευση. Συνολικά 210 μουσικά τραγούδια, χώρου περίπου 8.4 GB, χρησιμοποιήθηκαν για την εκπαίδευση. Αυτό στοίχησε περίπου 78% της βάσης δεδομένων που χρησιμοποιήθηκε συνολικά.

Το υπόλοιπο 22% , αφιερώθηκε στην επαλήθευση των δεδομένων. Η επαλήθευση γίνεται όταν περάσει 1 εποχή και πρέπει να υπολογιστεί η ακρίβεια του δικτύου. Οι εναπομήναντες φάκελοι ήταν οι

- MAPS_SptkBGAm_2
- MAPS_StbgTGd2_2

όπου επίσης χρησιμοποιήθηκαν μόνο οι υποφάκελοι **MUS**. 30 μουσικά τραγούδια υπήρχαν σε κάθε φάκελο, επομένως 60 μουσικά τραγούδια, χώρου περίπου 2.2 GB, χρησιμοποιήθηκαν για την επαλήθευση. Επομένως, συνολικά υπάρχουν 1480 πλαίσια ως δεδομένα επαλήθευσης.

Η κύρια διαφορά μεταξύ των διαχωρισμένων δεδομένων, είναι πως τα δεδομένα που ανήκουν στην ομάδα επαλήθευσης, δεν χρησιμοποιούνται για την οπισθοδιάδοση, που σημαίνει δεν βελτιώνουν με το σφάλμα που υπολογίζεται, το δίκτυο, παρά μόνο το βαθμολογούν. Χρησιμοποιώντας την εντολή .summary() μέσω της βιβλιοθήκης keras, υποδυκνείεται η αρχιτεκτονική του ορισμένου μοντέλου:

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
bidirectional_3 (Bidirection (32, 50, 200)             151200
_________________________________________________________________
bidirectional_4 (Bidirection (32, 200)                 240800
_________________________________________________________________
dense_2 (Dense)              (32, 88)                  17688
=================================================================
Total params: 409,688
Trainable params: 409,688
Non-trainable params: 0
_________________________________________________________________
```

Εικόνα 4.3.1: Αρχιτεκτονική μοντέλου BiLSTM.

4.3.2 Εκπαίδευση Δικτύου

Κατά την διαδικασία της εκπαίδευσης, υπήρξαν μικροί πειραματισμοί. Η αρχική ρύθμιση εκπαίδευσης εμπνεύστηκε από [3], καθώς και όποιες αλλαγές που διακρίθηκαν απαραίτητες, συμφωνήθηκαν μεταξύ επιβλέποντα καθηγητή και φοιτητή.

Πιο συγκεκριμένα, εκτός των ρυθμίσεων του μοντέλου που αναφέρθηκε παραπάνω (4.3), προτάθηκε στοχαστική μείωση κλίσης (Stochastic Gradient Descent) [48], με συνάρτηση κόστους δυαδική εντροπία (Binary Cross Entropy). Ο λόγος της επιλογής της συνάρτησης αυτής, προέρχεται από τις τιμές των ταμπελών, που έχουν είτε 1 σε ύπαρξη νότας, είτε 0 διαφορετικά. Προτάθηκε από τον καθηγητή διαφορετική συνάρτηση κόστους, συγκεκριμένα την Μέσων τετραγώνων (Mean Squared Error), καθώς δίνει απόκλιση μεταξύ πιθανοτήτων ύπαρξης κλάσης. Είναι σαφώς πιο ευκολονόητη η χρήση αυτής της συνάρτησης, καθώς το πρόβλημα που επιλύθηκε αποτελεί πρόβλεψη 88 κλάσεων, και όχι ύπαρξη νότας ή μη. Ο ρυθμός μάθησης τέθηκε στο 0.1, μειώνοντάς τον κατά 3 φορές όταν δεν υπήρχε βελτίωση στο ποσό του σφάλματος για 10 εποχές. Η εκπαίδευση δεν κράτησε ποτέ παραπάνω από 10 εποχές, καθώς λόγω της οριακής επεξεργαστικής δύναμης, εκπαίδευση άνω των 10 εποχών ξεπερνούσε τις 72 ώρες εκπαίδευσης.

Επιλέχθηκε, επομένως, ο βελτιστοποιητής Stochastic Gradient Descent, με συνάρτηση κόστους την Mean Squared Error. Επειδή οι εποχές που χρησιμοποιήθηκαν είναι 10, τέθηκε το όριο για την λήξη της εκπαίδευσης νωρίτερα, όταν περνούν 3 εποχές χωρίς βελτίωση. Δημιουργήθηκε ξεχωριστός φάκελος για κάθε επίσημη δοκιμή εκπαίδευσης, καθώς αλλάζανε οι ρυθμισείς ώστε να καταλήξει η τελική μορφή του μοντέλου.

Τόσο για την εκπαίδευση, όσο και για την επαλήθευση, χρησιμοποιήθηκαν generators, έναντι κλασικών συναρτήσεων, πάνω στο περιβάλλον της Python. Αυτό ήταν απαραίτητο καθώς, τα δεδομένα που τροφοδοτούνται στο δίκτυο πρέπει να φορτωθούν στην κάρτα RAM. Δηλαδή, παρόλο που το μηχάνημα που εκπαιδεύεται περιέχει 16 GB RAM, δεν θα χωρούσε να επεξεργαστεί όλη η βαση δεδομένων εκπαίδευσης. Ο λόγος που χρειάζεται μεγάλη επεξεργασία και χώρο, είναι ότι το δίκτυο, όντας αναδρομικό, αλλά και αμφίδρομο, χρειάζεται να κρατάει στην μνήμη όλην την χρονοσειρά που τροφοδοτείται, αλλά και όλες τις αλλαγές των βαρών που γίνεται κατά την οπισθοδιάδοση στον άξονα του χρόνου (BPTT). Όπως γίνεται κατανοητό, επομένως, ο μόνος τρόπος να τροφοδοτηθεί το δίκτυο και ταυτόχρονα να μην υπάρξουν προβλήματα στο ίδιο το μηχάνημα, χρησιμοποιήθηκαν generators. Η χρήση των generators δεν διαφέρουν από τις απλές συναρτήσεις στην σύνταξή τους, παρά μόνο στην εντολή εξόδου τους, δηλαδή, αντί **return,** υπάρχει η εντολή **yield**. Περισσότερα παρακάτω, στην ανάλυση κώδικα.

Στην εκπαίδευση, επιλέχθηκε αρχικά η τροφοδότηση δεδομένων με μικρό παράθυρο παρατήρησης του FFT. Μετά την πάροδο της εκπαίδευσής του, επιλέχθηκε περαιτέρω εκπαίδευση, στο ίδιο μοντέλο, με το μακρύτερο παράθυρο. Αυτό προκάλεσε μείωση της ακρίβειας, ειδικά στις πρώτες εποχές. Έπειτα, παρόλο που το κόστος μειώθηκε σε παρόμοιες τιμές με του μικρού παραθύρου, η δοκιμή σε μερικά αρχεία μετά την τελική εκπαίδευση, έδειξε ανακριβή αποτελέσματα. Περισσότερα, θα αναλυθούν στο 5ο Κεφάλαιο.

Τα δεδομένα εκπαίδευσης, καθώς επιλέγονται με τυχαίο τρόπο, θεωρητικά η πηγή έχει απεριόριστα δείγματα προς τροφοδοσία. Όμως, επειδή τα δεδομένα επαλήθευσης, αποτελούνται από 1480 δείγματα, καθώς ο generator επιλέγει σειριακά τα αρχεία.

Η πρώτη εκπαίδευση του δικτύου, έχοντας επιλέξει 2100 δείγματα εκπαίδευσης για κάθε εποχή, 600 δείγματα επαλήθευσης, έδιναν χρόνο 1 εποχής πάνω από 36 ώρες. Συγκεκριμένα, εκπαιδεύτηκε για 20 ώρες. Έχοντας αναλυτικά τις ακρίβειες, τις πρώτες 8 ώρες η ακρίβεια βρέθηκε στο 57%. Στις 15 ώρες, στο 86%, ενώ στις 20 ώρες πλησίασε το 90%. Τα αποτελέσματα δεν ήταν ικανοποιητικά όταν δοκιμάστηκε το δίκτυο, επομένως δημιουργήθηκαν καινούρια.

Με 200 δείγματα εκπαίδευσης, 150 επαλήθευσης, η 1 εποχή ανερχόταν στις 3,5 ώρες.

Η ακρίβεια σύμφωνα με την εκπαίδευση ήταν:

ΕΠΟΧΗ	**TRAINING ACC.**	**VALIDATION ACC.**
1	55%	72%
2	70%	89%
3	82%	94%
4	91%	96%
5-10	95-96.7%	96-96.5%

Αυτό σημαίνει πως το δίκτυο σύγκλινε στα βάρη του, ή βρέθηκε σε τοπικό ελάχιστο ο βελτιστοποιητής (Gradient Descent), που σημαίνει πως το πρόβλημα, σύμφωνα με αυτές τις ρυθμίσεις, έχει λυθεί. Όμως, κάνοντας μια απλή εφαρμογή του δικτύου, τα αποτελέσματα έδειξαν μικρή ακρίβεια. Επομένως, το δίκτυο σε αυτήν την διαμόρφωση, είχε υπερπροσαρμοστεί στα δεδομένα εκπαίδευσης (overfitting).

Η ανάλυση του κώδικα θα καταστήσει το μέρος της εκπαίδευσης πιο ξεκάθαρο, όπως αναλύεται παρακάτω.

4.4 Ανάλυση Κώδικα Νευρωνικού Δικτύου

Το αρχείο το οποίο αναγράφεται ο κώδικας όπου, χτίζεται, εκπαιδεύεται και ελέγχεται το δίκτυο, ονομάζεται **Architecture Sequential Model of keras.ipynb**. Όπως και στα άλλα 2 αρχεία που δημιουργήθηκαν για την εργασία αυτή, έτσι και εδώ, στην αρχή του κώδικα εισάχθηκαν όλες οι χρήσιμες βιβλιοθήκες που αξιοποιήθηκαν για να δημιουργηθεί το δίκτυο. Προστέθηκαν επίσης εντολές για debugging, όπως να δοθεί στα midi αρχεία χώρος που να ξεπερνά το αρχικό όριο που έχει τεθεί από τους δημιουργούς της pretty_midi βιβλιοθήκης. Επίσης, συνδέθηκαν με χρήση εντολής import και τα 2 αρχεία που δημιουργήθηκαν προηγουμένως, καθώς έπρεπε να καλεστούν συναρτήσεις για να δημιουργηθούν οι είσοδοι και οι ταμπέλες.

Δημιουργήθηκαν 2 generators. Συγκεκριμένα, ο πρώτος ονομάστηκε **training_generator** και ζητά 1 όρισμα:

1. window

- Το είδος παραθύρου που θα χρησιμοποιηθεί (2048/8192 samples).

Όπου, η χρήση του είναι να κατασκευάζει και να προετοιμάζει την είσοδο στο νευρωνικό δίκτυο. Πιο συγκεκριμένα, χρησιμοποιήθηκε η βιβλιοθήκη glob, ώστε να παρθούν όλοι οι υποφάκελοι στο μέρος των δεδομένων προς εκπαίδευση. Δηλαδή, δημιουργήθηκε μια λίστα με κάθε wave αρχείο που υπάρχει σε κάθε υποφάκελο του φακέλου «Training Dataset». Έτσι, υπάρχουν έτοιμα όλα τα μονοπάτια προς προσπέλαση.

Στη συνέχεια, επιλέχθηκε από αυτή την λίστα, ένα τυχαίο αρχείο. Ανεξάρτητα αν έχει επιλεχθεί ξανά ή όχι, ένα αρχείο μπορεί να επιλεχθεί όσες φορές τύχει από τον ψευδοτυχαίο αλγόριθμο που χρησιμοποιήθηκε. Ανοίγοντας το αρχείο wave που επιλέχθηκε, μετατράπηκε το μονοπάτι και στο αντίστοιχο μονοπάτι του midi αρχείου. Δηλαδή, δόθηκε το wave αρχείο, και μέσω επεξεργασίας του μονοπατιού ως κείμενο, έγινε εφικτό να επιλεχθεί και το αντίστοιχο midi αρχείο, ίδιας ονομασίας, κάθε φορά. Προφανώς χρειάζεται να επιλεχθούν, καθώς στο τέλος του generator πρέπει να εξάγει τα δεδομένα προς εκπαίδευση, αλλά και τις ταμπέλες (κλάσεις) αυτών, καθώς το πρόβλημα είναι υπό επίβλεψη (Supervised Learning). Έπειτα, καλέσθηκαν από τα προηγούμενα αρχεία, **Preprocess.ipynb** και **Midiz.ipynb,** οι 2 συναρτήσεις που εξάγουν τις απαραίτητες πληροφορίες. Καλώντας την συνάρτηση **process**, δημιουργήθηκαν οι τράπεζες φίλτρων που τροφοδοτούνται στο δίκτυο. Παρομοίως, με την συνάρτηση **piano_roll** δημιουργήθηκε η αναπαράσταση των κλάσεων-στόχων, βάσει το midi αρχείο.

Έχοντας τα απαραίτητα δεδομένα, χρειάστηκε μια περαιτέρω επεξεργασία ώστε να μπορεί να τροφοδοτηθεί στο δίκτυο χωρίς κανένα πρόβλημα. Δημιουργήθηκε χώρος, μήτρες

τύπου numpy. Για την είσοδο δημιουργήθηκε μήτρα διαστάσεων 32*50*88 και για την εξαγωγή ταμπέλων-στόχων, μήτρα διαστάσεων 32*88.

Όπως είναι φανερό από τις διαστάσεις και μόνο, το πρόβλημα θα χρειαστεί να προβλέψει 1 πλαίσιο των 88 νοτών, χρησιμοποιώντας 50 πλαίσια των 88 νοτών το καθένα. Να σημειωθεί, πως, με βάση την [3] το πλαίσιο το οποίο προβλέπεται κάθε φορά, για πλαίσια [t-50, t], δεν είναι το t+1, αλλά το t-25. Δηλαδή, προβλέπεται το μεσσαίο πλαίσιο για κάθε είσοδο. Ο λόγος είναι ότι, καθώς χρησιμοποιείται αμφίδρομο αναδρομικό δίκτυο, η πρόβλεψη των πλαισίων t-51ή t+1 δίνει μονόπλευρη άποψη, με την έννοια ότι γνωρίζει μόνο το μέλλον ή το παρελθόν αντίστοιχα. Έτσι, για να υπάρχει ακριβέστερη πρόβλεψη, γνωρίζοντας και το παρελθόν αλλά και το μέλλον του επί πρόβλεψη πλαισίου, επιλέχθηκε να προβλέπεται το μεσσαίο σαν κλάσεις-στόχοι. Εν ολίγοις, για τα 50 συνεχόμενα πλαίσια, παρά το γεγονός ότι υπάρχουν διαθέσιμες όλες οι κλάσεις στόχοι για κάθε πλαίσιο, επιλέγεται μόνο το 25ο σαν ολικός στόχος. Ο τρόπος που γίνεται αυτό, από άποψη κώδικα, θα αναλυθεί παρακάτω, όταν ορίζεται η αρχιτεκτονική του νευρωνικού δικτύου.

Οι τράπεζες και οι κλάσεις που έχουν φορτωθεί, αποτελούν ολόκληρο το αρχείο. Όμως, χρειάστηκε να επιλεχθούν μήκη ίσα με τις διαστάσεις που προαναφέρθηκαν, ώστε να συμφωνούν με την αρχιτεκτονική του δικτύου. Έτσι, επιλέχθηκε επίσης τυχαία ένα πλαίσιο από το μήκος των τραπεζών, ως χρονική στιγμή t_0 , και επιλέχθηκαν τα ακριβώς επόμενά του μέχρι την στιγμή t_0 + 50.Αυτό επαναλήφθηκε 32 φορές συνολικά. Να σημειωθεί πως, οι 32 φορές συμβάλλουν στο γεγονός ότι τα πλαίσια τα οποία διαβάζονται, περιέχουν επικάλυψη κατά 50%. Αν για batch=1, διαβαστούν τα πλαίσια 100-150, τότε για batch=2, θα διαβαστούν τα πλαίσια 125-175 κ.ο.κ.. Αυτό δίνει τεράστιο πλεονέκτημα λόγω της ιδιαίτερης αρχιτεκτονικής που επιλέχθηκε, καθώς δύναται η σύνδεση παρελθόντος και μέλοντος, διαβάζοντας τα ίδια δεδομένα, αλλά σε άλλη σχετική χρονική σειρά.

Τέλος, τοποθετήθηκαν με την σειρά αυτή στις ήδη δημιουργημένες μήτρες numpy, χρησιμοποιώντας την εντολή **yield**, επιστράφηκαν οι μήτρες αυτές. Καλώντας αυτόν τον generator, παράγει λοιπόν, τα χαρακτηριστικά προς μάθηση (Xtrain), και τις κλάσεις-στόχους για την ποσοτικοποίηση σφάλματος (Ytrain):

Η εντολή **yield** εξάγει:

1. Xtrain
   - Χαρακτηριστικά προς μάθηση, διαστάσεων (32,50,88).
2. Ytrain
   - Κλάσεις-στόχοι, διαστάσεων (32,88).

Παρομοίως, χρειάστηκε να δημιουργηθεί και generator για την επαλήθευση των δεδομένων. Ονομάστηκε **validation_generator**, περιείχε 1 όρισμα:

1. window

- Το είδος παραθύρου που θα χρησιμοποιηθεί (2048/8192 samples).

Η διαδικασία, συγκριτικά με τον generator των δεδομένων εκπαίδευσης, έχει πάρα πολλά σημεία. Χρησιμοποιήθηκε ακριβώς ο ίδιος τρόπος συλλογής των μονοπατιών, μέσω της βιβλιοθήκης glob. Δημιουργήθηκαν τα αντίστοιχα μονοπάτια και για τα midi αρχεία που αντιστοιχούσαν στο καθένα wave αρχείο. Δημιουργήθηκαν μήτρες numpy μεγέθους 32*50*88, και 32*88, για τα χαρακτηριστικά και τις κλάσεις-στόχους, αντίστοιχα. Καλέστηκαν οι συναρτήσεις **process**, **piano_roll** για την εξαγωγή των δεδομένων. Η διαφορά μεταξύ των 2 generators, ήταν ότι σε αυτόν για την επαλήθευση δεδομένων, δεν πάρθηκαν τα δεδομένα με τυχαίο τρόπο. Δηλαδή, διαβάστηκαν τα αρχεία σειριακά, σύμφωνα με το μονοπάτι που διαβάστηκε. Λόγω διαφοράς μεγέθους στα αρχεία, όμως, χρειάστηκαν κάποιες περαιτέρω διαχειρίσεις στον κώδικα, ώστε να μπορέσει το πρόγραμμα να λειτουργήσει χωρίς πρόβλημα. Δηλαδή, σε περίπτωση που ο δείκτης, και των δεδομένων αλλά και των κλάσεων-στόχων, δεν έφτανε να συμπληρώσει 50 πλαίσια, επιλεγόταν το τελευταίο πλαίσιο όσες φορές χρειαζόταν, ώστε να συμπληρωθεί. Αυτό δεν προκαλεί προβλήματα, όπως λάθος υπολογισμός εκτίμησης/ακρίβειας, καθώς οι περιπτώσεις αυτές ανέρχονται σε πάρα πολύ χαμηλό ποσοστό, όπως και δεν αποκλείεται σαν φυσικό μουσικό τραγούδι (να συνεχίζει για μεγαλύτερο χρονικό διάστημα, δηλαδή, τις ίδιες συχνότητες/νότες).

Η εντολή **yield** εξάγει:

1. Xvalid

- Χαρακτηριστικά προς εκτίμηση, διαστάσεων (32,50,88).

2. Yvalid

- Κλάσεις-στόχοι προς υπολογισμό κόστους , διαστάσεων (32,88).

4.4.1 Αρχιτεκτονική και Εκπαίδευση Δικτύου

Στην συνέχεια, δημιουργήθηκαν 2 συναρτήσεις, για την δημιουργία του μοντέλου, και την εκπαίδευσή του, αντίστοιχα. Ονομάστηκαν **note_model** και **train_note_model.** Η υλοποίηση αυτών των εννοιών, χάρη στην βιβλιοθήκη Keras κατέστησε τον κώδικα αρκετά συμπιεσμένο οπτικά. Δηλαδή, λόγω του υψηλού επιπέδου προγραματισμού της Keras, με μερικές γραμμές

κώδικα ήταν εφικτό να πραγματοποιηθεί η περίπλοκη αρχιτεκτονική του νευρωνικού δικτύου που χρησιμοποιήθηκε.

Η πρώτη συνάρτηση, δεν δέχεται κάποιο όρισμα, καθώς δεν είναι απαραίτητη κάποια είσοδος. Αρχικά ορίστηκαν σταθερές, για την διευκόλυνση των δεδομένων που τροφοδοτήθηκαν στο δίκτυο. Όπως προαναφέρθηκε, οι απαραίτητες διαστάσεις εισόδου, πρέπει να συμφωνούν με την αρχιτεκτονική του δικτύου. Έτσι, δημιουργήθηκαν κατάλληλα ονομασμένες σταθερές, ίσες των διαστάσεων τροφοδότησης του δικτύου. Καλέστηκε η εντολή **Sequential()** από την βιβλιοθήκη Keras. Η συγκεκριμένη λειτουργία, παράγει ένα αντικείμενο, το οποίο έχει ιδιότητες που χρησιμοποιήθηκαν παρακάτω. Δοκιμαστικά, είχε κατασκευαστεί η αρχιτεκτονική και με την άλλη δυνατότητα που παρέχει η Keras, ονομαζόμενο Functional API. Για λόγους ευκρίνειας του κώδικα, επιλέχθηκε το πρώτο από τα 2. Αρχικά προστέθηκε ένα κρυφό επίπεδο, τύπου αμφίδρομου, LSTM, των 100 νευρώνων. Επειδή ήταν το πρώτο κρυφό επίπεδο που ορίστηκε, έπρεπε να οριστεί η μεταβλητή batch_input_shape, η οποία ορίζει το μέγεθος εισόδου στο δίκτυο. Σε εκείνο το σημείο, χρησιμοποιήθηκαν οι 3 σταθερές που ορίστηκαν στην αρχή της συνάρτησης. Τέθηκε η μεταβλητή stateful σε Ψευδής. Η μεταβλητή αυτή παρέχει την δυνατότητα να θυμάται το δίκτυο τα προηγούμενα δεδομένα, ώστε να μπορέσει να παράξει συσχετίσεις μεταξύ των παρτίδων (batches). Επιλέγοντάς το, θα δινόταν η δυνατότητα στο δίκτυο, να κρατάει τις καταστάσεις των πυλών του, σαν αρχικές θέσεις για την επόμενη σειρά παρτιδών. Άρα, αφού χρησιμοποιήθηκαν με τυχαία σειρά οι διαδοχικές τροφοδοτήσεις, οφείλει να είναι Ψευδής η επιλογή αυτή. Η μεταβλητή **return_sequences** προσφέρει την δυνατότητα να δίνει στα μεταγενέστερα κρυφά επίπεδα ολόκληρα τα βάρη, μήκους 32*50 εισαχθέντων τιμών, αν είναι Αληθής. Στην περίπτωση που είναι Ψευδής, η μόνη έξοδος του δικτύου προς το επόμενο επίπεδο, είναι μόνο οι 32 διαστάσεις. Είναι ένας τρόπος να μειώνονται οι διαστάσεις, αλλά και να προβλέπεται μόνο 1 πλαίσιο, και όχι όλα όσα τροφοδοτούνται. Είναι μία από τις εξαιρετικά ενδιαφέρουσες ιδιότητες των αναδρομικών νευρωνικών δικτύων.

Όπως προειπώθηκε, επίσης, χρησιμοποιήθηκε ομαλοποίηση των τιμών των βαρών, τύπου L1, επειδή η γνωστή και ως Lasso Regression (Least Absolute Shrinkage and Selection Operator) [49] υπολογίζει τις απόλυτες τιμές των συσχετίσεων, έτσι δίνοντας σε ασήμαντα χαρακτηριστικά μηδενικές τιμές. Αυτό είναι πολύ χρήσιμο στην συγκεκριμένη εργασία, καθώς χρειάζεται να υπάρχει καθαρή επιλογή των κλάσεων, ανάμεσα στις 88 πιθανές κλάσεις. Βέβαια, θα ήταν δυνατό να υπολογιστεί και με διαφορετικές τεχνικές, όπως η πιο φησισμένη cross-validation, όπως σχολιάζεται και στην[50], αλλά εξαιτίας της [3], ακολουθήθηκε η ομαλοποίηση L1. Η συνάρτηση που ονομάζεται L1, προσθέτει έναν παραπάνω όρο στην συνάρτηση κόστους που χρησιμοποιείται, και είναι ο παρακάτω όρος:

$$\lambda \sum_{j=1}^{p} |w_j|$$

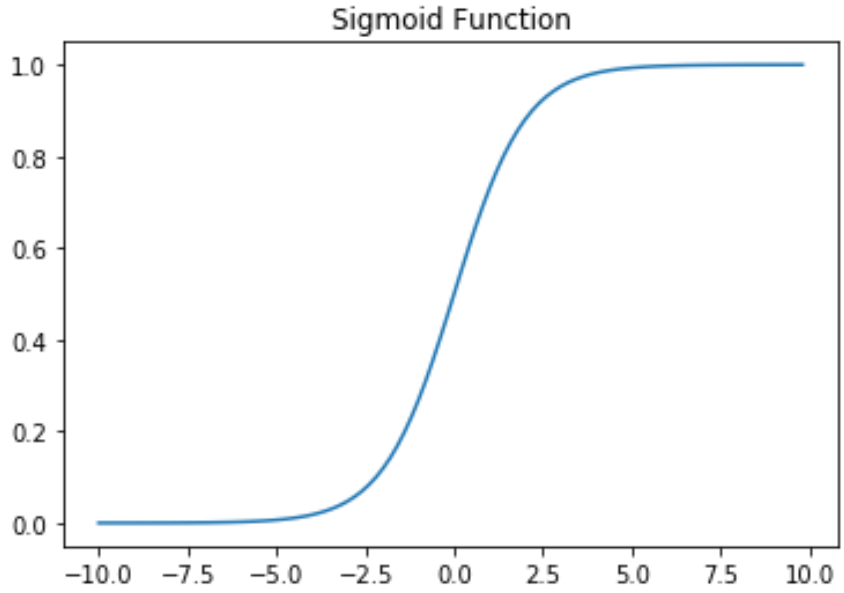
Το θετικό αυτής της συνάρτησης, είναι πως προσφέρει καλά αποτελέσματα σε δεδομένα που έχουν ακραίες τιμές (outliers). Προφανώς, υπάρχουν και μειονεκτήματα. Τα βασικότερα 2, είναι πως για μικρές αλλαγές τιμών, μπορούν να υπάρχουν τεράστιες αλλαγές στην τιμή που θα δώσει ο υπολογισμός, και πως σε αντίθεση με την ομαλοποίηση L2, η L1 έχει περιθώριο για πολλές λύσεις, επομένως μπορεί να υπάρχουν αποκλίσεις από την καλύτερη δυνατή ομαλοποίηση.

Σε συνδυασμό με αυτή την ομαλοποίηση των βαρών, χρησιμοποιήθηκε και αφαίρεση μνήμης, δηλαδή dropout [51] . Η τεχνική dropout χρησιμοποιείται για την αποφυγή υπερπροσαρμογής των δεδομένων μάθησης του δικτύου. Πρακτικά, αυτό που γίνεται, είναι να απενεργοποιεί ορισμένους νευρώνες του δικτύου, ώστε κατά την ώρα της εκπαίδευσης, να υπάρχει μια ελαστικότητα στην πιθανή αθέμιτη ομοιότητα των δεδομένων. Στην εργασία αυτή, όπως αναφέρθηκε και η [3], χρησιμοποιήθηκε dropout ίσο με 50% νευρώνων, ώστε να αποφευχθεί η υπερπροσάρμοση σε συγκεκριμένο ηχόχρωμα πιάνου, λόγω της εκτέλεσης που ηχογραφήθηκε στο wave αρχείο που τροφοδοτήθηκε. Αυτός είναι ένας πολύ αποτελεσματικός τρόπος να μάθει το δίκτυο χωρίς να υπάρχουν ευαίσθητα βάρη νευρώνων. Η επιλογή των νευρώνων που απενεργοποιούνται κάθε φόρα, γίνεται τυχαία, δίνοντας στοιβαρά αποτελέσματα. Και η ομαλοποίηση βαρών L1, αλλά και η τυχαία απενεργοποίηση των μισών νευρώνων κατά την διάρκεια της εκπαίδευσης, χρησιμοποιήθηκαν και στα 2 κρυφά επίπεδα του δικτύου.

Για το επίπεδο εξόδου, χρησιμοποιήθηκε ένα πλήρες συνδεδεμένο επίπεδο, γνωστό στην βιβλιοθήκη Keras ως **Dense Layer.** Το πλήθος των νευρώνων, έπρεπε να είναι ίσο με το πλήθος των κλάσεων, δηλαδή 88. Επίσης, δόθηκε συγκεκριμένη συνάρτηση ενεργοποίησης, η Σιγμοϊδής συνάρτηση:

$$S(x) = \frac{1}{1 + e^{-x}}$$

Όπου, σχηματικά παρουσιάζεται:



Εικόνα 4.4.1: Σιγμοϊδής συνάρτηση.

Η συνάρτηση αυτή δίνει εύρος τιμών μεταξύ [0,1], δίνοντας την πιθανότητα ύπαρξης μιας από τις 88 νότες. Έτσι γίνεται η ποσοτικοποίηση ύπαρξης μιας νότας στο τέλος του δικτύου. Στη συνέχεια του κώδικα, τυπώνεται και η αρχιτεκτονική που στήθηκε, και τέλος επιστρέφει με την εντολή **return** το μοντέλο:

1. model
   - Αρχιτεκτονική μοντέλου.

Στην συνέχεια, δημιουργήθηκε η προαναφερθείσα συνάρτηση **train_note_model**. Δέχεται 2 ορίσματα:

1. model
   - ένα μοντέλο δικτύου, αντικειμένου τύπου Sequential.
2. window
   - Παράθυρο παρατήρησης FFT (2048/8192 samples).

Αρχικά, δημιουργείται το μονοπάτι στον ηλεκτρονικό υπολογιστή, όπου θα αποθηκευτούν οι εποχές του μοντέλου κατά την εκπαίδευση. Το είδος αρχείου που δημιουργείται είναι .hdf5 και πρόκειται για μοντέλα δικτύων όπου αποθηκεύουν τα βάρη και την αρχιτεκτονική του δικτύου. Χρησιμεύει στην χρησιμοποίηση του δικτύου, καλώντας το αρχείο αυτό, χωρίς να χρειάζεται εκ νέου εκπαίδευση για να φορτωθεί στον κώδικα. Δημιούργθηκε μέσω της βιβλιοθήκης Keras, ο τρόπος εκπαίδευσης του δικτύου. Συγκεκριμένα, επιλέχθηκε Stochastic Gradient Descent και συνάρτηση κόστους η Mean Squared Error, όπως προαναφέρθηκαν. Αυτό που έλεγξε το δίκτυο για την μάθησή του, ήταν η ακρίβεια στα δεδομένα επαλήθευσης (validation dataset) όπου προσπελαυνόταν κάθε τέλος εποχής. Δημιουργήθηκε επίσης λειτουργία πρόωρου τερματισμού της εκπαίδευσης μέσω **EarlyStopping**, όπου έλεγχε το κόστος των δεδομένων επαλήθευσης, ώστε, στην περίπτωση που 3 συνεχόμενες εποχές έδιναν μικρότερη διαφορά από την ελάχιστη αποδεκτή 0.003, να διέκοπτε την μάθηση του δικτύου. Πολύ χρήσιμη τεχνική για την εξοικονόμηση χώρου και επεξεργαστικής δύναμης. Επίσης, χρησιμοποιήθηκε τεχνική όπου, με αρχικό ρυθμό μάθησης 0.1, μειώνει τον ρυθμό μάθησης με παράγοντα 0.3 (ή και 3 σε δοκιμές), μέχρι να φτάσει έως 0.001, σε διάστημα 5 εποχών, όσο δεν υπήρχε βελτίωση στο κόστος των δεδομένων επαλήθευσης. Δηλαδή, όσο δεν υπήρχε βελτίωση στην ακρίβεια,σε συνεχόμενη βάση, τόσο θα μειωνόταν ο ρυθμός μάθησης. Έπειτα καλέστηκε λειτουργία για την αποθήκευση του μοντέλου αυτού. Στο τέλος, καλέστηκε συνάρτηση του μοντέλου του Keras, η **fit_generator**. Χρειάστηκε η συγκεκριμένη συνάρτηση, καθώς χρησιμοποιήθηκαν generators, και αυτός είναι ο τρόπος που προσφέρει η βιβλιοθήκη Keras. Αρχικά, δηλώθηκε ο generator που δημιουργήθηκε συγκεκριμένα για τα δεδομένα μάθησης, επιλέχθηκε πλήθος βημάτων ανά εποχή για μάθηση, ίσο με 78, καλέστηκαν οι λετουργίες που αναφέρθηκαν παραπάνω, δόθηκε για τον generator των επαληθεύσων ο αντίστοιχος που δημιουργήθηκε, με βήματα επαλήθευσης 4, και τελικά δόθηκαν πόσες εποχές να προσπελαστούν, πειραματικά ίσες με 20. Οι τιμές που επιλέχθηκαν για τις ρυθμίσεις (hyperparameters) του δικτύου, επιλεχθήκαν μέσω της [3], την εμπειρία του επιβλέποντα καθηγητή Α.Πικράκη, αλλά και πειραματισμούς του φοιτητή.Δεν επιστρέφεται μεταβλητή από αυτή την συνάρτηση.

Έπειτα, καλέστηκε η συνάρτηση **note_model** για να δημιουργηθεί το μοντέλο που περιγράφθηκε, και ακριβώς παρακάτω του, καλέστηκε η συνάρτηση για την εκπαίδευσή του (**train_note_model**(model, window=True)). Σε αυτή την στιγμή, η εκπαίδευση του δικτύου ξεκίνησε. Οι έξοδοι της κονσόλας, είχαν την εξής μορφή:

```
Epoch 1/20
78/78 [==============================] - 2982s 38s/step - loss: 0.
8203 - acc: 0.0108 - val_loss: 0.8192 - val_acc: 0.0000e+00
```

Όπου, δίνει τον αριθμό εποχής στην οποία εκπαιδεύεται το δίκτυο εκείνη την στιγμή, δείχνει σε ποιο βήμα εκπαίδευσης βρίσκεται, πόση ώρα συνολικά διαρκεί η εποχή αλλά και το κάθε βήμα, αλλά και τις σημαντικότερες πληροφορίες περί κόστους, ακρίβειας και των δεδομένων εκπαίδευσης, και των δεδομένων επαλήθευσης. Στο τέλος της κάθε εποχής, η κονσόλα δίνει:

```
Epoch 00001: val_loss improved from inf to 0.81920, saving model t
o 24 Oct run\2x100units-loss=MSE-Model-01-0.00.hdf5
```

Δείχνοντας ποια εποχή ολοκληρώθηκε, την βελτίωση του κόστους, αλλά και που αποθηκεύτηκε το μοντέλο. Στην συνέχεια, εκπαιδεύτηκε το μοντέλο επιπλέον, χρησιμοποιώντας και το μεγαλύτερο παράθυρο FFT.

ΚΕΦΑΛΑΙΟ 5

ΕΦΑΡΜΟΓΕΣ ΚΑΙ ΑΠΟΤΕΛΕΣΜΑΤΑ

5.1 Εισαγωγή

Στο παρόν κεφάλαιο, παρουσιάζοντα τα αποτελέσματα της διαδικασίας που προηγήθηκε. Αναλύονται τα αποτελέσματα του νευρωνικού δικτύοου πάνω στα δεδομένα δοκιμής (test dataset) όπου και παράγεται το τελικό ποσοστό ακρίβειας του δικτύου. Η εκτίμηση δηλαδή, της ακρίβειας του δικτύου, δημιουργήθηκε μέσω εφαρμογής πρόβλεψης των αρχείων που δεν έχει εκπαιδευτεί το δίκτυο. Παρακάτω, παρουσιάζονται δοκιμές, τεχνικές και ποσοστά αποτελεσμάτων, με οδηγό την έρευνα [3] αλλά και φυσικά την καθοδήγηση του επιβλέποντα καθηγητή, Ά.Πικράκη.

Αναλύονται αποτελέσματα σε ποσοστά ακρίβειας με διάφορες ομαδοποιήσεις, είτε ανά μουσικού τραγουδιού, είτε μέσου όρου ανά πλαίσιο, αλλά και συνολικά το ποσοστό ακρίβειας των προβλέψεων.

5.2 Εφαρμογή στο Δίκτυο

Ο τρόπος με τον οποίο το εκπαιδευμένο δίκτυο, κατάφερε να κάνει πρόβλεψη, απλώς δημιουργήθηκαν με την ίδια προεπεξεργασία τα χαρακτηριστικά των τραπεζών φίλτρων, με επικάλυψη πλαισίων, ακρίβως όπως η είσοδος στο δίκτυο, όταν βρισκόταν υπό εκπαίδευση. Πιο συγκεκριμένα, οι διαστάσεις εισόδου του δικτύου, ανεξαρτήτως φάσης εκπαίδευσης ή δοκιμής, χρειάζεται να ισούνται με (32,50,88). Αυτό σημαίνει πως, μπορεί να ανακυκλωθεί κώδικας ώστε να τροφοδοτηθεί το δίκτυο, όπως εκτελέστηκε και στην φάση εκπαίδευσης, στο μέρος της δοκιμής. Έτσι, έχοντας ξεχωριστό generator για τα δεδομένα δοκιμής, προσπελάστηκαν τα αρχεία, σειριακά και όχι τυχαία.

Συγκεκριμένα, αφού δημιουργήθηκε το μοντέλο, χρειάστηκε η επαλήθευσή του, χρησιμοποιώντας δεδομένα που δεν έχουν χρησιμοποιηθεί στην φάση της εκπαίδευσης. Καθώς διαβάστηκαν με την σειρά τα αρχεία δοκιμής, χρησιμοποιήθηκε μια συγκεκριμένη τεχνική, η οποία υπολογίζει την ακρίβεια του δικτύου, πάνω σε δεδομένα που δεν έχει εκπαιδευτεί. Έτσι, αποκτήθηκαν ποσοστά ακρίβειας του μοντέλου στις προβλέψεις που υπολόγισε, δίνοντας μια πιο πλούσια πληροφόρηση του κατά πόσο αξιόπιστο είναι το δίκτυο που δημιουργήθηκε.

5.3 Τεχνική Επαλήθευσης

Η τεχνική που χρησιμοποιήθηκε, ακολουθήθηκε σύμφωνα την [3]. Στην έρευνα αυτή, χρησιμοποιήθηκε η τεχνική F-score (ή F-measure) [52]. Ο λόγος που χρησιμοποιήθηκε, ήταν για να είναι δυνατό να συγκριθούν τα αποτελέσματα με αντίστοιχες έρευνες που είχαν λάβει μέρος. Η τεχνική αυτή, χρησιμοποιείται συνήθως, για επαλήθευση δικτύων που αφορούν 2 κλάσεις, ή έστω, την ύπαρξη ή μη μίας κλάσης. Αυτό συμβαίνει, επειδή, υπολογίζοντας τα απαραίτητα ζητούμενα για τον υπολογισμό του F-score, χρειάστηκαν να καταμετρηθούν ορισμοί που δεν έχουν κάποια βάση σε καταστάσεις πολλών κλάσεων.

Για την εφαρμογή της τεχνικής αυτής, χρειάστηκε να υπολογιστούν ορισμένα πλήθη. Αυτά τα πλήθη δημιουργούν την μήτρα σύγχυσης, όπου παρουσιάζει πολύ εϋνοϊκά την ακρίβεια ενός μηχανισμού πρόβλεψης. Ορίζεται ως παρακάτω:

		Predicted	
		Negative	**Positive**
Actual	**Negative**	True Negative	False Positive
	Positive	False Negative	True Positive

Εικόνα 5.1: Μήτρα Σύγχυσης, Confusion Matrix.

Οι αρισμοί των ονομασιών αυτών, έχει ως εξής:

- True: Ανήκει στην κλάση, σύμφωνα με την τιμή αληθείας (Ground Truth).
- False: Δεν ανήκει στην κλάση, σύμφωνα με την τιμή αληθείας.
- Positive: Ανήκει στην κλάση, σύμφωνα με την πρόβλεψη.
- Negative: Δεν ανήκει στην κλάση, σύμφωνα με την πρόβλεψη.

Επομένως, μπορεί να σχεδιαστεί, στην περίπτωση των 2 κλάσεων (ή και 1 κλάσης, τύπου δυαδικής ταξινόμησης), αυτός ο πίνακας, δηλώνοντας στην κύρια διαγώνιο τα σωστά προβλεπόμενα χαρακτηριστικά, και στην αντίθετη διαγώνιο του πίνακα, πόσα ταξινομήθηκαν λάθος. Σε μια δυαδική κλάση, έστω 0 και 1, για παράδειγμα, αν η τιμή των False Negative για την κλάση 1, είναι υψηλή, σημαίνει πως ο ταξινομητής προέβλεψε τιμή 0, ενώ έπρεπε να είχε προβλέψει 1.

Όμως, στο πρόβλημα που είχε ανατεθεί για την εργασία αυτή, δεν υπάρχουν 2 κλάσεις, αλλά 88. Προφανώς, αυτό περιπλέκει αυτήν την τεχνική. Παρά την δυσκολία μετατροπής του προβλήματος για να εφαρμοστεί αυτή η μέτρηση ακρίβειας, υπολογίστηκε χρησιμοποιώντας την βιβλιοθήκη sklearn, αλλά και με κώδικα χωρίς την βοήθεια καμίας βιβλιοθήκης.

5.3.1 F-score χωρίς βιβλιοθήκη

Στην περίπτωση υπολογισμού της ακρίβειας χωρίς βιβλιοθήκη, απλώς υπολογίστηκαν οι τιμές των True Positive, False Positive, True Negative, False Negative. Η

συνάρτηση **evaluate** υλοποιεί αυτόν τον υπολογισμό. Χρειάζεται να είναι φορτωμένο το μοντέλο στον κώδικα, ώστε να υπάρξει η πρόβλεψη των τιμών. Τα ορίσματα που χρειάστηκαν, βρίσκονται παρακάτω:

1. threshold

- Όριο ταξινομητή, για την ελάχιστη τιμή που θα θεωρηθεί πως η νότα υπάρχει στο πλαίσιο. Προτεινόμενη τιμή : 0.5 .

Η προτεινόμενη τιμή, οφείλεται και στην έρευνα [3], αλλά φυσικά και στην παρατήρηση του φοιτητή στις τιμές που προβλήφθηκαν, καθώς όλες οι προβλεπούσες τιμές, πολύ σπάνια ξεπερνούσαν την πιθανότητα εμφάνισης της τάξης 0.50. Τιμές όπως 0.51 δίναν ελάχιστες νότες ως ύπαρξη, ενώ τιμές όπως 0.49 μπορεί να έδιναν μέχρι και τις 88 νότες σε ένα πλαίσιο, πράγμα φυσικά αδύνατο για τον άνθρωπο. Υπολογίστηκαν τα νούμερα των νοτών midi, τόσο των κλάσεων-στόχων, όσο και των προβλεπόμενων τιμών, και έγινε σύγκριση αυτών των λιστών μεταξύ τους. Η συνάρτηση αυτή, επιστρέφει την τιμή F-score υπολογίζοντας τα εξής:

Πρώτα, χρειάζεται ο υπολογισμός ακρίβειας, που προέρχεται από τον παρακάτω μαθηματικό τύπο:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

Όπως και ο υπολογισμός ανάκλησης:

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

Έχοντας αυτές τις τιμές υπολογισμένες, για κάθε πλαίσιο που διαβάζεται, υπάρχει ο υπολογισμός της συνάρτησης F1 Score:

$$F1 = 2 * \frac{Precision * Recall}{Preciison + Recall}$$

Η εντολή **return**, επιστρέφει:

1. f1

- Μέσος όρος f1 score, μεταξύ όλων των πλαισίων προς επαλήθευση (validation dataset).

Το πρόβλημα με αυτή την προσέγγιση, είναι πως, λόγω των 88 κλάσεων, οι τιμές που υπολογίζονται είναι ανακριβείς όσο αναφορά τα False Negatives. Αυτό συμβαίνει διότι, κάθε πρόβλεψη ύπαρξης νότας, προσθέτει στα ψευδή αρνητικά ποσό ίσο με 87 καθώς, θεωρητικά προβλέπεται λάθος. Αυτό είναι το μεγάλο πρόβλημα αυτής της τεχνικής, για το πρόβλημα των 88 κλάσεων. Το τελικό αποτέλεσμα που δόθηκε από τον υπολογισμό, ανέρχεται σε **2.54%** ακρίβεια του δικτύου, σύμφωνα με το validation dataset που χρησιμοποιήθηκε.

5.3.2 F-score με βιβλιοθήκη

Στην συγκεκριμένη διαδικασία, αρχικά δημιουργήθηκε συνάρτηση **sparse_pred**:

1. x

- Χαρακτηριστικά προς πρόβλεψη

1. threshold

- Όριο ταξινομητή, για την ελάχιστη τιμή που θα θεωρηθεί πως η νότα υπάρχει στο πλαίσιο. Προτεινόμενη τιμή : 0.5 .

Όπου, δημιουργεί το προβλεπόμενο πλαίσιο στην ίδια μορφή με αυτή των στόχων. Δηλαδή, έγινε αναπαράσταση των προβλεπόμενων πλαισίων σε 88 διαστάσεις, λαμβάνοντας μια αραιή μήτρα, όπου η τιμή 1 σημαίνει ύπαρξη νότας, και 0 διαφορετικά. Η συνάρτηση αυτή επιστρέφει με την εντολή **return**:

1. ypred

- Προβλεπόμενο πλαίσιο, σε αναπαράσταση αραιής μήτρας (88 διαστάσεων).

Έπειτα, δημιουργήθηκε η συνάρτηση που επικαλεί αυτή τη συνάρτηση, ώστε να υπολογιστεί ο μέσος όρος των f1-score, καθώς υπολογίζεται ανά πλαίσιο. Επίσης, υπολογίστηκε πόσα πλαίσια αποτελούν τα αρχεία επαλήθευσης, όπου ήταν 60 πλαίσια. Έτσι, για να αποφευχθεί η παρουσίαση προβλήματος του generator που χρησιμοποιήθηκε, χρησιμοποιήθηκαν 1480 πλαίσια ακριβώς, χωρίς να χρειαστεί να εξαντληθεί ο generator. Η συνάτηση **fscore** δεν δέχεται κάποιο όρισμα, όμως θεωρεί απαραίτητο την ύπαρξη μοντέλου, φορτωμένο στον κώδικα (εξαιτίας της **sparse_pred**). Στο τέλος κάθε υπολογισμού πλαισίου, εκτυπώνεται η εκτίμηση, που έχει υπολογιστεί για το εκάστοτε πλαίσιο. Οι τιμές κυμαίνονται από **0.55%**

μέχρι **11.1%**. Το τελικό αποτέλεσμα που δόθηκε από τον υπολογισμό αυτό, ανέρχεται σε **6.67%** ακρίβεια του δικτύου, σύμφωνα με το validation dataset που χρησιμοποιήθηκε, καλώντας:

1. f1

- F1-score, μέσος όρος όλων των πλαισίων από δεδομένα επαλήθευσης.

5.3.3 Πρόβλεψη μέγιστης πιθανότητας

Σε αυτό το υποκεφάλαιο, υπολογίστηκε η πιθανότητα ύπαρξης της νότας αυτής, που είχε την μεγαλύτερη πιθανότητα ύπαρξης, για κάθε πλαίσιο. Σε αυτή την περίπτωση, μόνο 1 νότα είναι δυνατόν να προβλεθφεί, επομένως θα χρειαστεί μόνο 1 διάσταση από τις 88 κάθε φόρα, καθιστώντας τον υπολογισμό F1-score πιο κατάλληλο.

Συγκεκριμένα, χρησιμοποιήθηκε η πρόβλεψη κλάσης που παρέχει η βιβλιοθήκη keras, δίνοντας για κάθε πλαίσιο, την νότα η οποία έχει την μεγαλύτερη πιθανότητα. Έτσι, για κάθε πλαίσιο, ελέγχθηκε αν η νότα αυτή, στον πίνακα κλάσεων-στόχων, έχει την τιμή 1, που συμβολίζει την ύπαρξη της νότας.

Έπειτα, αυτό εφαρμόστηκε για όλα τα πλαίσια που παρήγαγε ο generator των δεδομένων επαλήθευσης, πλήθους 1480 πλαισίων. Στο τέλος, υπολογίστηκε πόσες νότες προβλήφθηκαν σωστά. Με αυτόν τον τρόπο υπολογίστηκε το ποσοστό ακρίβειας. Δεν υπολογίστηκε, όπως προηγουμένως, το F-score, παρά μόνο ποσοστό επιτυχίας πρόβλεψης νότας, προς όλα τα πλαίσια. Με αυτόν τον τρόπο, το ποσοστό επιτυχίας ανήλθε σε **14.11%.**

ΚΕΦΑΛΑΙΟ 6

ΣΥΜΠΕΡΑΣΜΑΤΑ

6.1 Σύνοψη Εργασίας και Αποτελεσμάτων της

Στην εργασία αυτή, χρησιμοποιήθηκαν πολλές τεχνικές, πάνω σε 2 τομείς, τόσο πάνω στην επεξεργασία ήχου, όσο και στην μηχανική μάθηση. Το πρόβλημα που ανατέθηκε, αφορούσε την μετάφραση αρχείων ήχου, σε αναπαράσταση midi νοτών. Το πρόβλημα αυτό, παρουσίασε μια πληθώρα προβλημάτων και δυσκολιών ως προς την υλοποίηση. Το πρόβλημα επιλέχθηκε ώστε ο φοιτητής να μάθει αυτές τις τεχνικές, να αποκτήσει εμπειρία πάνω σε αυτά τα ζητήματα, δίνοντας προτεραιότητα στους πειραματισμούς, αφήνωντας την ποιότητα των αποτελεσμάτων σε δευτερεύοντα ρόλο, λόγω της πολυπλοκότητας του αντικειμένου.

Κατά την προεπεξεργασία, παρατηρήθηκε πως η παραμικρή αλλαγή σειράς στα βήματά της, υπήρχαν πολύ μεγάλες αλλαγές στην συμπεριφορά των δεδομένων. Συγκεκριμένα, ακόμα και η τεχνική του zero padding, έδινε άλλη έξοδο, αν εφαρμοζόταν, πριν ή μετά τον μετασχηματισμό Φουριέ. Επίσης, το παράθυρο του μετασχηματισμού, ήταν πολύ κρίσιμο σημείο, καθώς όσο μικρότερο παράθυρο, τόσο πιο λεπτομερής ανάλυση των συχνοτήτων υπήρχε, αλλά και τόσο ευαίσθητη ήταν η πληροφορία. Η ευαίσθητη πληροφορία, περιείχε τον κίνδυνο να καταγραφούν, στα διαδοχικά πλαίσια διαφορετικές συχνότητες από αυτές που πραγματικά υπήρχαν. Η τεχνική ομαλοποίησης, και η τεχνική τοπικών μεγίστων, παρουσίασαν παρόμοια αποτελέσματα όσο αναφορά την παρουσίαση των δεδομένων.

Στο στάδιο της δημιουργίας της αρχιτεκτονικής του δικτύου, παρουσιάστηκαν πολλές λειτουργίες που στην αγγλική ορολογία ονομάζονται hyperparameters. Οι επιλογές των ρυθμίσεων αυτών, πάρθηκαν με βάση την έρευνα [3] αλλά και την καθοδήγηση του επιβλέποντα καθηγητή. Το λεγόμενο κούρδισμα του δικτύου, αντιμετωπίστηκε ως πειραματισμός πάνω στο πρόβλημα που ανατέθηκε. Κατά την εκπαίδευση, δημιουργήθηκαν πολλά μοντέλα, με διαφορετικές ρυθμίσεις, ώστε να συγκριθούν μεταξύ τους, ως το κατά πόσο κατάλληλα αποτελέσματα παρήγαγαν.

Στο μέρος της επαλήθευσης του μοντέλου και των αποτελεσμάτων, δημιουργήθηκαν ποσοστά ακρίβειας, όπως το F-score, που προτάθηκε και χρησιμοποιήθηκε από την [3]. Τα ποσοστά ήταν πολύ χαμηλά, φτάνοντας έναν μέσο όρο επιτυχίας, 6.67%. Όμως, το αποτέλεσμα δεν είναι χειρότερο από μια τυχαία πρόβλεψη, καθώς δεν υπάρχουν μόνο 2 κλάσεις, αλλά 88 κλάσεις. Δηλαδή, η τυχαία πρόβλεψη, θα έδινε πιθανότητα επιτυχίας 1/88 = 1.13%. Αυτό το ποσοστό θα ήταν στην περίπτωση που προβλεπόταν μόνο 1 νότα ανά πλαίσιο. Προφανώς μεγάλο μέρος της δυσκολίας του προβλήματος ήταν η ταυτόχρονη αναγνώριση συχνοτήτων στο ίδιο πλαίσιο. Το κατώφλι που προτάθηκε, ήταν 0.5, δηλαδή όταν η πιθανότητα βρισκόταν άνω του 50% στην πρόβλεψη, έπρεπε να θεωρηθεί πως η νότα πραγματικά υπάρχει στο πλαίσιο αυτό. Αυτό, δίνει το συμπέρασμα, πως με τόσο χαμηλό κατώφλι, ακόμα και η έρευνα [3], κατάφερε πολύ μικρό ποσοστό ακρίβειας. Αυτό δηλώνει

πως, οι τεχνικές που χρησιμοποιήθηκαν σε αυτό το πρόβλημα, δεν είναι οι καλύτερες δυνατές.
Αυτό μπορεί να οφείλεται, στην προεπεξεργασία των δεδομένων, στην αρχιτεκτονική του νευρωνικού δικτύου που επιλέχθηκε, στο κούρδισμα αυτού, ή ακόμα και στα ίδια τα δεδομένα. Άλλες έρευνες έχουν παρουσιάσει καλύτερα αποτελέσματα, σύμφωνα με [3], όπου χρησιμοποιήθηκαν συνελικτικά νευρωνικά δίκτυα (Convolutional Neural Networks).

Όπως φαίνεται, η χρήση αναδρομικών νευρωνικών δικτύων, είναι ευαίσθητη και ασταθής σε πολλούς τομείς. Παρά το γεγονός ότι θεωρητικά είναι το πιο συμβατό είδος νευρωνικού δικτύου, άλλα δίκτυα φαίνεται να υπερτερούν σε αναγνώριση χρονοσειρών. Έχοντας ορισμένες συγκρίσεις μεταξύ των δικτύων που εκπαιδεύτηκαν, αλλάζοντας τις υπερ-παραμέτρους όπως το πλήθος νευρώνων, φαίνεται πως χρειάζεται μια διαφορετική προσέγγιση, ώστε να επιτευχθεί ένα ικανοποιητικό αποτέλεσμα ακρίβειας. Η έρευνα που ακολουθήθηκε, είναι πολύ πιθανό να δημοσίευσε επιλεγμένες ρυθμίσεις του προβλήματος που ασχολήθηκε. Έτσι, μόνο μια εμπειρική συμβολή πάνω στην εργασία θα μπορούσε να βελτιώσει τα αποτελέσματα, εφαρμόζοντας μόνο τις τεχνικές που τελικώς υλοποιήθηκαν σε αυτή την διπλωματική εργασία.

Τα αποτελέσματα της εργασίας, παρά το γεγονός ότι είναι πολύ χαμηλά σε ακρίβεια, η διαδικασία που ακολουθήθηκε, είναι λογικά ορθή. Δηλαδή, μπορεί να παραχθεί πανομοιότυπη έξοδος με αυτή της εισόδου, όπως ακριβώς συνέβη και στην έρευνα [3].

6.2 Μελλοντικές Επεκτάσεις της Εργασίας

Η εργασία αυτή, είναι ένα εισαγωγικό μέρος σε ένα όραμα του φοιτητή να υπάρξει ένα παγκοσμίως γνωστό πρόγραμμα που να υλοποιεί με 100% ακρίβεια το πρόβλημα που ανατέθηκε. Δηλαδή, να μπορεί κανείς να παράγει μία midi παρτιτούρα, έχοντας ένα αρχείο wave, ή και σε συμπιεσμένη μορφή, ώστε να υπάρχει μια πλήρως σωστή αναπαράσταση. Η χρήση τέτοιου εργαλείου, θα δώσει στον χρήστη τεράστια ελευθερία όσο αφορά την ανάγνωση μουσικής, αλλά και καταγραφή της, ειδικά προς άτομα που δεν έχουν βάσεις στην θεωρία μουσικής.

Η παρούσα διπλωματική εργασία μπορεί μελλοντικά να επεκταθεί ως ακολούθως:

- Βελτίωση της ακρίβειας πρόβλεψης, αλλά και ποσοστού σιγουριάς του δικτύου (confidence) στην εκάστοτε πρόβλεψη κλάσεων, κρατώντας την αρχιτεκτονική σταθερή.

- Ανάπτυξη λογισμικού, τέτοιου ώστε αυτόματα να παράγονται 2 φασματογραφήματα των midi αναπαραστάσεων, ένα των προβλεπόμενων νοτών, και ένα με την πραγματική αναπαράσταση (ground truth), δίνοντας οπτική πληροφορία, χωρίς να εμπλέκονται τεχνικές ή θεωρίες. Σκοπός του, να γίνεται αντιληπτός ο στόχος του λογισμικού που έχει αναπτυχθεί στα πλαίσια της διπλωματικής εργασίας.

- Δημιουργία λογισμικού, τέτοιου ώστε να γίνεται περαιτέρω μετάφραση του ήχου, σε πραγματική παρτιτούρα, όχι midi αναπαράστασης, αλλά γραμματικής της μουσικής θεωρίας.

- Ανάπτυξη ιστοσελίδας, όπου χρήστης θα μπορεί να ανεβάζει το αρχείο ήχου, ώστε να επεξεργάζεται μέσω της ιστοσελίδας του, και έπειτα να κατεβάζει το αρχείο midi που παράχθηκε.
- Περαιτέρω επέκταση δικτύων, όπου είναι εκπαιδευμένα πάνω σε περισσότερα δεδομένα, περισσότερων οργάνων, όπως κιθάρα, άρπα, βιολί. Ένα μεγαλύτερο εύρος οργάνων, μπορεί να δημιουργήσει ένα άρτιο λογισμικό προς αξιοποίηση μουσικών, αλλά και δισκογραφικών εταιρειών που χρειάζονται καταγραφή παρτιτούρας για την επιβεβαίωση πνευματικών δικαιωμάτων της μουσικής του κάθε καλλιτέχνη τους.

ΚΕΦΑΛΑΙΟ 7

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] MIREX, “MIREX Wiki.” 2013.

[2] K. Suzuki, “Real-time Audio to Score Alignment.” pp. 1–26.

[3] T. Kwon, D. Jeong, and J. Nam, “Audio-to-score alignment of piano music using RNN-based automatic music transcription,” pp. 380–385, 2017.

[4] D. Byrd and J. G. Simonsen, “Towards a Standard Testbed for Optical Music Recognition: Definitions, Metrics, and Page Images,” *J. New Music Res.*, vol. 44, no. 3, pp. 169–195, 2015.

[5] M. Dorfer, A. Arzt, and G. Widmer, “Towards Score Following in Sheet Music Images,” *Proc. 17th Int. Soc. Music Inf. Retr. Conf.*, no. Section 3, 2016.

[6] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio, “Object Recognition with Gradient-Based Learning,” *J. Exp. Psychol. Gen.*, vol. 136, no. 1, pp. 23–42, 2007.

[7] H.-P. Königs, “Deep content-bases music recommendation,” *IT-Risiko-Management mit Syst.*, pp. 109–121, 2006.

[8] “magenta.tensorflow.org.” .

[9] S. Böck and M. Schedl, “Enhanced Beat Tracking With Context-Aware Neural Networks,” *14th Int. Conf. Digit. Audio Eff.*, pp. 301–306, 2011.

[10] P. L. Chithra and R. Aparna, “Performance analysis of windowing techniques in automatic speech signal segmentation,” *Indian J. Sci. Technol.*, vol. 8, no. 29, 2015.

[11] V. Emiya, N. Bertin, B. David, and R. Badeau, “A piano database for multipitch estimation and automatic transcription of music,” 2010.

[12] S. Hochreiter and J. Schmidhuber, “LONG SHORT-TERM MEMORY,” vol. 9, no. 8, pp. 1–32, 1997.

[13] R. Pascanu, D. Tour, T. Mikolov, and D. Tour, “On the difficulty of training recurrent neural networks,” no. 2.

[14] “www.python.org.” .

[15] “Jupyter.Org.” .

[16] “developer.nvidia.com.” .

[17] “docs.scipy.org.” .

[18] W. T. Cochran *et al.*, “What is the Fast Fourier Transform?,” *IEEE Trans. Audio Electroacoust.*, vol. 15, no. 2, pp. 45–55, 1967.

[19] Open Source, “Overview — NumPy v1.” .

[20] B. Mcfee *et al.*, “Librosa - Audio and Music Signal Analysis in Python,” no. Scipy, pp. 18–25, 2015.

[21] E. Loweimi, S. M. Ahadi, T. Drugman, and S. Loveymi, “On the importance of Pre-emphasis and window shape in phase-based speech recognition,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7911 LNAI, pp. 160–167, 2013.

[22] B. A. Olshausen, “Aliasing,” pp. 1–6, 2000.

[23] P. Podder, T. Z. Khan, M. H. Khan, and M. M. Rahman, “Comparative Performance Analysis of Hamming, Hanning and Blackman Window,” *Int. J. Comput. Appl.*, vol. 96, no. 18, pp. 975–8887, 2014.

[24] N. Instruments, “Understanding FFTs and Windowing,” *White Pap. No. 4844*, pp. 1–11, 2015.

[25] “Scipy 0.19 Signal Hamming window.” .

[26] W.Cooley and W.Tukey, “An Algorithm for the Machine Calculation of Complex Fourier Series,” *Math. Comput.*, vol. 19, pp. 297–301, 1965.

[27] D. N. Rockmore, “The FFT: An algorithm the whole family can use,” *Comput. Sci. Eng.*, vol. 2, no. 1, pp. 60–64, 2000.

[28] P. Heckbert, “Fourier Transforms and the Fast Fourier Transform (FFT) Algorithm,” *Comput. Graph. (ACM).*, vol. 2, pp. 1–13, 1998.

[29] D. R. Smith, “The design of divide and conquer algorithms,” *Sci. Comput. Program.*, vol. 5, no. C, 1985.

[30] N. Sturmel and L. Daudet, “Signal Reconstruction from STFT Magnitude: A State of the Art,” pp. 375–386, 2011.

[31] L. Tesic, B. Bondzulic, M. Andric, and B. Pavlovic, “An experimental study on the phase importance in digital processing of speech signal,” *Acta Polytech. Hungarica*, vol. 14, no. 8, pp. 197–213, 2017.

[32] T. S.Huang, J. W.Burnett, and A. G. Deczky, “The importance of phase in image processing,” 1975.

[33] J. Saini, “Power Spectral Density Analysis of Speech Signal using Window Techniques,” vol. 131, no. 14, pp. 33–36, 2017.

[34] G. Der Wu and C. T. Lin, "Word boundary detection with mel-scale frequency bank in noisy environment," *IEEE Trans. Speech Audio Process.*, vol. 8, no. 5, pp. 541–553, 2000.

[35] "The mel frequency scale and coefficients," vol. 1, pp. 1–3, 2000.

[36] S. Kadry and A. El Hami, "Solution of Stochastic Non-Homogeneous Linear First-Order Difference Equations," no. August, pp. 245–248, 2014.

[37] H. Zhang, "The Optimality of Naive Bayes," 2004.

[38] T. M. Cover and P. E. Hart, "Nearest Neighbor Pattern Classification," vol. I, 1967.

[39] Y. Zhang and S. S. Ge, "Design and analysis of a general recurrent neural network model for time-varying matrix inversion," *IEEE Trans. Neural Networks*, vol. 16, no. 6, pp. 1477–1490, 2005.

[40] M. Schuster and K. K.Paliwal, "Bidirectional Recurrent Neural Networks," *Arch. des Mal. du Coeur des Vaiss. - Prat.*, vol. 16, no. 186, p. 7, 2010.

[41] S. Hochreiter, "The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions," *Int. J. Uncertainty, Fuzziness Knowledge-Based Syst.*, vol. 06, no. 02, pp. 107–116, 1998.

[42] Y. le Cun, "A Theoretical Framekwork for Back-Propagation." 1988.

[43] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, *Learning Internal Representations by Error Propagation*. Morgan Kaufmann Publishers, Inc., 2013.

[44] A. Gruslys, R. Munos, I. Danihelka, M. Lanctot, and A. Graves, "Memory-Efficient Backpropagation Through Time," 2016.

[45] C.Gentile and M.K.Warmuth, "Linear Hinge Loss and Average Margin," *Adv. Neural Inf. Process. Syst.*, vol. 11, pp. 225–231, 1999.

[46] "Graphviz - Graph Visualization Software." .

[47] Keras, "Keras Documentation," *Loos functions*. 2017.

[48] H. Robbins and S. Monro, "A Stochastic Approximation Method," *IEEE Trans. Syst. Man Cybern.*, vol. 1, no. 4, pp. 338–344, 1971.

[49] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Anim. Genet.*, vol. 39, no. 5, pp. 561–563, 2008.

[50] A. Krogh and J. Vedelsby, "Neural Network Ensembles, Cross Validation, and Active Learning," *Pure Appl. Chem.*, vol. 76, no. 3, pp. 453–463, 2004.

[51] N. Srivastrava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple Was to Prevent Neural Networks from Overfitting," vol. 1,

no. 60, p. 11, 2004.

[52] Y. Sasaki, "The truth of the F-measure," *Am. Rev. Respir. Dis.*, vol. 125, no. 4 II, p. 114, 1982.