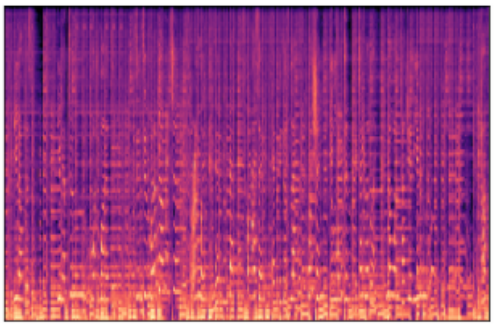
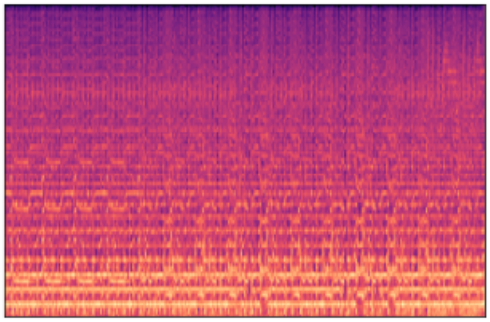
Προπαρασκευή Εργαστηρίου

Βήμα 1

Αφού φορτώσουμε τα φασματογραφήματα για τα mel coeficients δύο διαφορετικών τραγουδιών παρακάτω βλέπουμε τα φασματογραφήματά τους.



Σχήμα 1: Spectogram blues

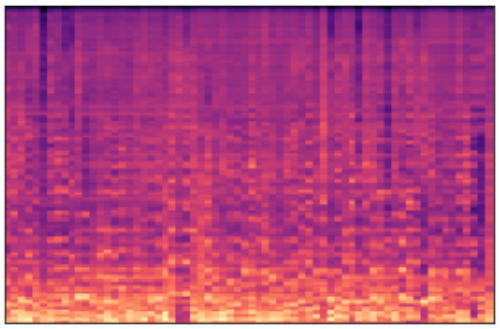


Σχήμα 2: Spectogram chiptune

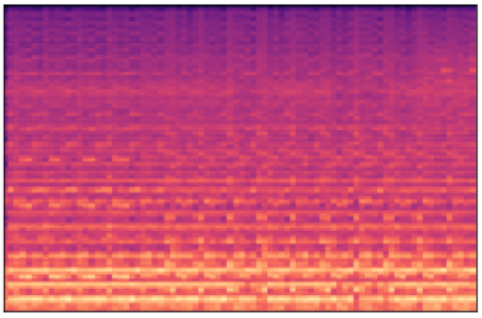
Παρατηρούμε ότι στο φασματογράφημα του είδους της chiptune μουσικής εμφανίζεται περιεχόμενο σε μεγαλύτερη μπάντα συχνοτήτων, ενώ υπάρχει και μεγαλύτερη σταθερότητα στο χρόνο, κάτι που είναι λογικό γιατί η chiptune είναι μουσική που δημιουργείται ηλεκτρονικά και δεν υπάρχουν χρονικές ασυνέχειες. Αντιθέτως, στο σπεκτόγραμμα της μπλουζ μουσικής, παρατηρούμε ότι έχει μεγαλύτερη μεταβλητότητα στο συχνοτικό της περιεχόμενο ως προς τον χρόνο.

Βήμα 2

Τυπώνοντας τις διαστάσεις των φασματογραφημάτων από το προηγούμενο ερώτημα παρατηρούμε ότι έχουν 1293 χρονικά βήματα (είναι πίνακες 128*1293). Η εκπαίδευση ενός lstm επάνω σε αυτά τα δεδομένα θεωρείται αποτελεσματική καθώς υπάρχει μεγάλος αριθμός διαφορετικών χρονικών καταστάσεων οι οποίες έχουν εξάρτηση μεταξύ τους και επομένως χρειάζεται η ιδιότητα των lstm για μνήμη από προηγούμενες και επόμενες καταστάσεις. Παρακάτω παρατίθενται τα beat spectograms για τα ίδια δείγματα με το προηγούμενο ερώτημα.



Σχήμα 3: Beat spectogram blues

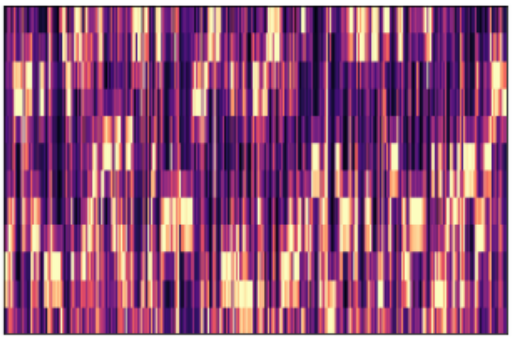


Σχήμα 4: Beat spectogram chiptune

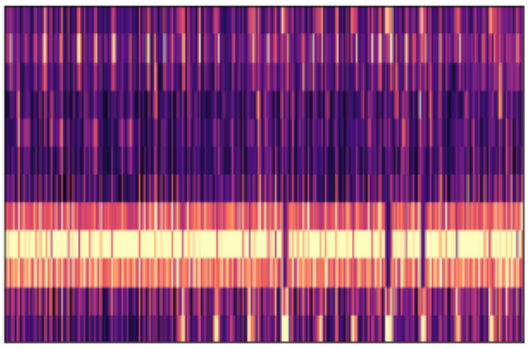
Παρατηρούμε ότι και τα δύο σπεκτογράμματα είναι πιο εξομαλυνσμένα στην εξέλιξη του χρόνου, αφού έχουμε πάρει τον median ανάμεσα στα beat της μουσικής και έτσι δεν παρατηρούνται τόσες χρονικές ασυνέχειες στο φασματικό περιεχόμενο

Βήμα 3

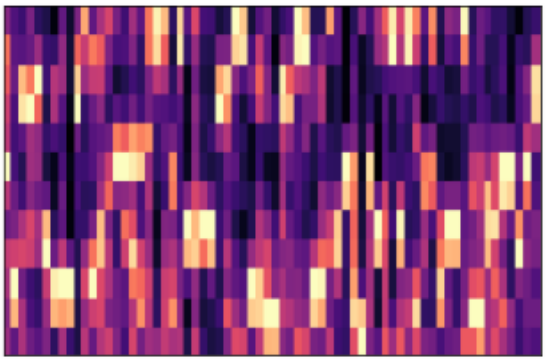
Παρακάτω επαναλαμβάνουμε την ίδια διαδικασία με τα βήματα 1 και 2 για τα χρωμογραφήματα, δηλαδή εμφανίζουμε το περιεχόμενο της ενέργειας για τις 12 νότες της κλασσικής μουσικής για κάθε μία από τις 1293 χρονικές στιγμές.



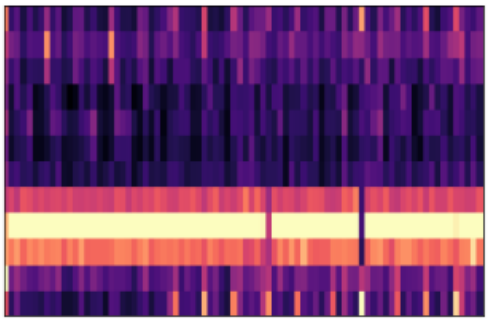
Σχήμα 5: Chromagram blues



Σχήμα 6: Chromagram chiptune



Σχήμα 7: Chromagram beat blues



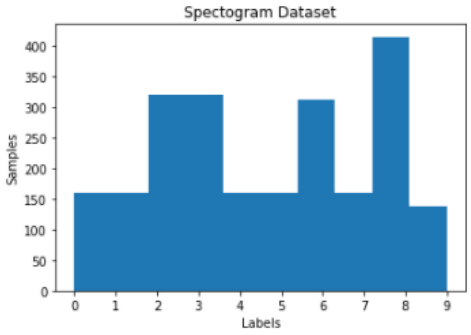
Σχήμα 8: Chromagram Beat chiptune

Από τα παραπάνω χρωμογραφήματα παρατηρούμε ότι για το είδος μουσικής blues η κατανομή της ενέργειας είναι πιο αυθαίρετη ενώ αντίθετα για το είδος chiptune υπάρχουν πιο ομοιόμορφα μοτίβα, κάτι φυσιολογικό καθώς το δεύτερο είδος παράγεται από υπολογιστή. Επίσης παρατηρούμε ότι και σε αυτή την περίπτωση τα beat chromagram παρουσιάζουν περισσότερη εξομάλυνση σε σχέση με τα αρχικά.

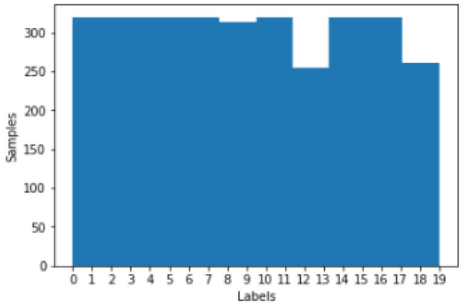
Βήμα 4

α) ο κώδικας που μας δίνεται εισάγει το dataset το οποίο περιέχει τα φασματογραφήματα και τα χρωμογραφήματα. Μέσα από αυτόν τον κώδικα εξάγωνται τα labels για κάθε είδος και επίσης χωρίζονται τα σύνολα σε σύνολα για εκπαίδευση και για αξιολόγηση του

μοντέλου κατα την εκπαίδευση(validation και όχι test). Επίσης μέσω της συνάρτησης class mapping γίνεται μια συγχώνευση στα labels έτσι ώστε κλάσεις που δεν έχουν μεγάλο αριθμό δειγμάτων ή για κάποιο λόγο θεωρούνται ιδιαίτερες, κατατάσονται σε γενικότερες κλάσεις.
γ) Παρακάτω παραθέτουμε 2 φασματογραφήματα όπου το πρώτο αναπαριστά τα δείγματα που αντιστοιχούν σε κάθε κλάση πριν από το class mapping και το δεύτερο μετά την εφαρμογή αυτού.



Σχήμα 9: Sample distribution before

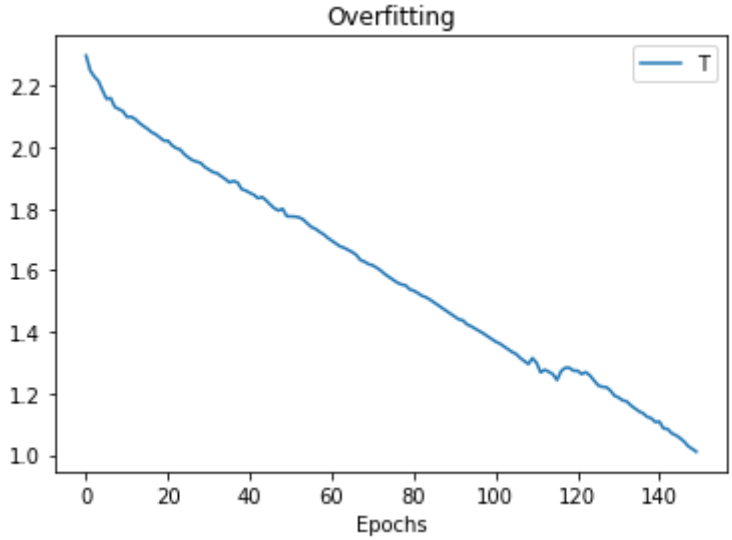


Σχήμα 10: sample distribution after

Όπως παρατηρούμε, τα δείγματα είναι πιο ομοιόμορφα κατανεμημένα μετά την εφαρμογή του class mapping.

Βήμα 5

Πριν εκπαιδεύσουμε το νευρωνικό θα κάνουμε overfit σε 1 batch για να ελέγξουμε αν μπορεί να εκπαιδευτεί. Παναμένουμε να δούμε το loss να μειώνεται διαρκώς.

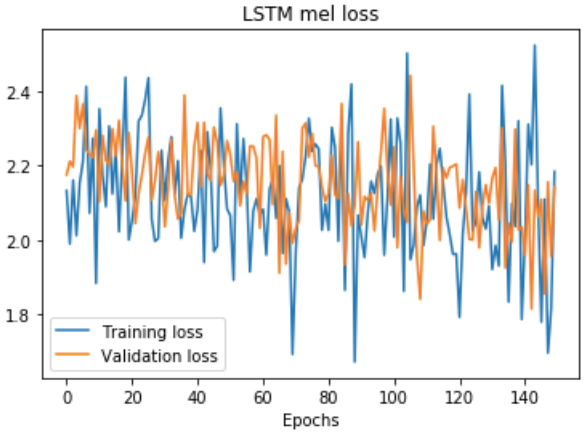


Σχήμα 11: Validation Loss overfitting

Από το παραπάνω γράφημα παρατηρούμε ότι το μοντέλο μας έχει στηθεί σωστά και

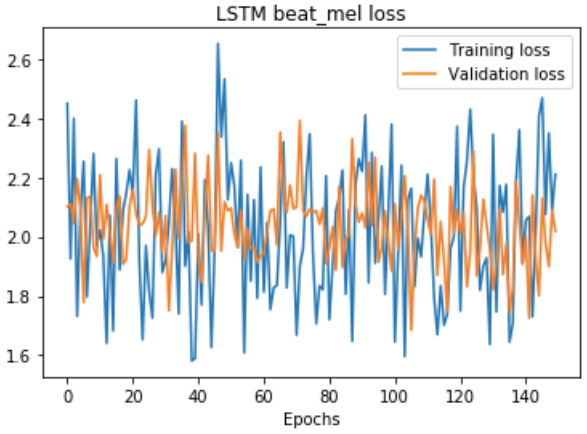
μπορεί να κάνει fit σε δεδομένα, αν και στην συγκεκριμένη περίπτωση επείδη γίνεται εκπαίδευση μόνο σε ένα batch των 32 δειγμάτων έχουμε υπερεκπαίδευση και το μοντέλο μας δεν θα γενικεύει σωστά.

γ) Για την εκπαίδευση του μοντέλου που ζητείται αρχικά ξεκινήσαμε εκπαίδευση με learning rate 0.001. Με το συγκεκριμένο ρυθμό το μοντέλο είχε την τάση να μειώνει το training και το validation loss με πολύ αργό όμως ρυθμό. Στη συνέχεια δοκιμάσαμε εκπαίδευση με lr 0.01 με σκοπό την γρηγορότερη σύγκλιση κάτι όμως που αύξησε τα σκαμπανευάσματα στο validation loss, χωρίς να υπάρχει συγκεκριμένη τάση αύξησης ή μείωσης. Παρακάτω απεικονίζεται η γραφική για την δεύτερη περίπτωση εκπαίδευσης σε 150 και με το σύνολο των δεδομένων προς εκπαίδευση (το 66% των δεδομένων είναι για training και το 33% για validation).



Σχήμα 12: Validation and training Loss mel

δ) Αντίστοιχα, παρακάτω βλέπουμε τον ρυθμό εκπαίδευσης του lstm μας επάνω στα δεδομένα τα οποία είναι συγχρονισμένα επάνω στο beat της μουσικής:

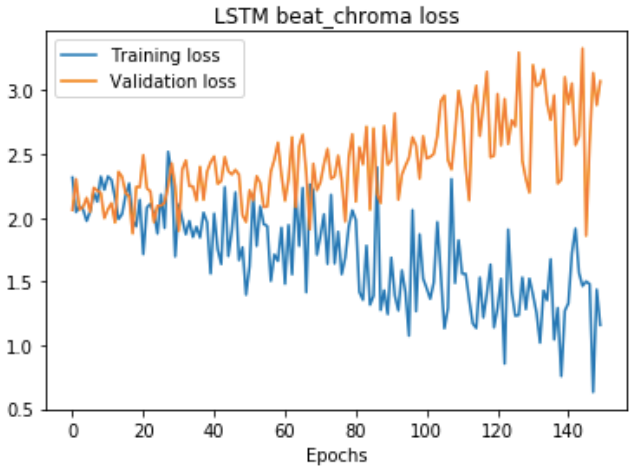


Σχήμα 13: Validation and training Loss beat mel

Είναι εμφανές ότι και σε αυτή την περίπτωση τόσο το validation όσο και το training

loss εμφανίζουν συνεχείς διακυμάνσεις και όχι μια σταθερή πορεία μείωσης.

ε) Τέλος, για την περίπτωση εκπαίδευσης μοντέλου επάνω στα δεδομένα με τα χρωμογραφήματα παρατηρούμε την εξής πορεία:



Σχήμα 14: Validation and training Loss beat mel

Το διαφορετικό στην συγκεκριμένη περίπτωση είναι ότι από ένα σημείο και πέρα το training loss εμφανίζει πτώση ενώ το validation ακολουθεί αντίθετη πορεία, κάτι που εκτιμάμε ότι οφείλεται στο ότι τα χρωμογραφήματα έχουν μόνο 12 συνιστώσες και έτσι το μοντέλο κάνει overfitting.

Βήμα 6

Στο συγκεκριμένο ερώτημα υπολογίσαμε τις μετρικές αξιολόγησης που μας ζητήθηκαν για τα μοντέλα των βημάτων **5γ, 5δ, 5ε**. Παρακάτω παραθέτουμε τα αποτελέσματα:

Accuracy: 0.2539130434782609

	precision	recall	f1-score	support
0.0	0.00	0.00	0.00	40
1.0	0.00	0.00	0.00	40
2.0	0.21	0.72	0.33	80
3.0	0.00	0.00	0.00	80
4.0	0.00	0.00	0.00	40
5.0	0.00	0.00	0.00	40
6.0	0.00	0.00	0.00	78
7.0	0.00	0.00	0.00	40
8.0	0.29	0.85	0.43	103
9.0	0.00	0.00	0.00	34
accuracy			0.25	575
macro avg	0.05	0.16	0.08	575
weighted avg	0.08	0.25	0.12	575

Σχήμα 15: metrics on model c

Accuracy: 0.08521739130434783

	precision	recall	f1-score	support
0.0	0.00	0.00	0.00	40
1.0	0.07	0.55	0.12	40
2.0	0.00	0.00	0.00	80
3.0	0.15	0.25	0.19	80
4.0	0.00	0.00	0.00	40
5.0	0.00	0.00	0.00	40
6.0	0.00	0.00	0.00	78
7.0	0.00	0.00	0.00	40
8.0	0.12	0.02	0.03	103
9.0	0.05	0.15	0.08	34
accuracy			0.09	575
macro avg	0.04	0.10	0.04	575
weighted avg	0.05	0.09	0.04	575

Σχήμα 16: metrics on model d

Accuracy: 0.20869565217391303

	precision	recall	f1-score	support
0.0	0.02	0.03	0.02	40
1.0	0.20	0.25	0.22	40
2.0	0.25	0.30	0.27	80
3.0	0.21	0.28	0.24	80
4.0	0.12	0.07	0.09	40
5.0	0.12	0.12	0.12	40
6.0	0.32	0.26	0.29	78
7.0	0.21	0.10	0.14	40
8.0	0.27	0.26	0.26	103
9.0	0.17	0.12	0.14	34
accuracy			0.21	575
macro avg	0.19	0.18	0.18	575
weighted avg	0.21	0.21	0.21	575

Σχήμα 17: metrics on model e

Στους παραπάνω πίνακες βλέπουμε τις μετρικές precision, recall και f1 ανά κλάση, συνολικά 10 είδη μουσικής, ενώ εμφανίζονται και τα weighted και macro average για κάθε μία από αυτές τις μετρικές.

Η μετρική precision για μια κλάση δείχνει πόσες από τις συνολικές προβλέψεις που έκανε το μοντέλο είναι προβλέψεις που ανήκουν στην σωστή κλάση, η μετρική recall δείχνει τι ποσοστό από τα συνολικά δείγματα μιας κλάσης έχουν επιλεχθεί από το μοντέλο μας και τέλος η μετρική f_1 score είναι ένας συνδυασμός των δύο παραπάνω και συγκεκριμένα εξάγεται με τον τύπο

$$f_1 = \frac{precision * recall}{precision + recall}$$

Επίσης η μετρική accuracy υπολογίζεται από τον εξής τύπο:

$$f_1 = 2\,\frac{TP + TN}{TP + TN + FP + FN}$$

όπου TP = true positives, FP = false positives και TN = true negatives, FN = false negatives. Η συγκεκριμένη μετρική δείχνει το ποσοστό των συνολικών σωστών προβλέψεων σε σχέση με τις συνολικές.

Η μετρική macro average για όλες τις προηγούμενες μετρικές είναι η μέση τιμή των μετρικών ανά κλάση που έχουμε . Πχ για την μετρική f_1 η macro f_1 είναι ο μέσος όλων των επιμέρους f_1 και η χρησιμότητά της είναι για να μας δείχνει μια γενική εκτίμηση του μοντέλου και όχι μόνο τα επιμέρους κάθε κλάσεις, καθώς η f_1 και οι υπόλοιπες μετρικές υπολογίζονται με την λογική one vs all the others για κάθε επιμέρους κλάση. Αντίστοιχα η weighted average αντί για τον απλό μέσο δίνει σε κάθε επιμέρους μετρική κάποιο βάρος ανάλογο του πλήθους των δειγμάτων που έχει η κάθε κλάση. Το micro average στην ουσία υπολογίζεται με την ίδια μέθοδο όπως το accuracy επομένως δεν εμφανίζεται σαν ξεχωριστή μετρική στους παραπάνω πίνακες.

Μεγάλη απόκλιση ανάμεσα στο accuracy και στο f_1 score μπορεί να έχω όταν έχω μεγάλο accuracy λόγω λίγων εκτιμήσεων αλλά επομένως μικρό recall και έτσι η μετρική f_1 να μην είναι το ίδιο υψηλή ή αντίθετα να υπάρχει πολύ μικρό accuracy αλλά μεγάλο recall λόγω πολλών εκτιμήσεων και έτσι η μετρική f_1 να είναι μεγαλύτερη από την accuracy.

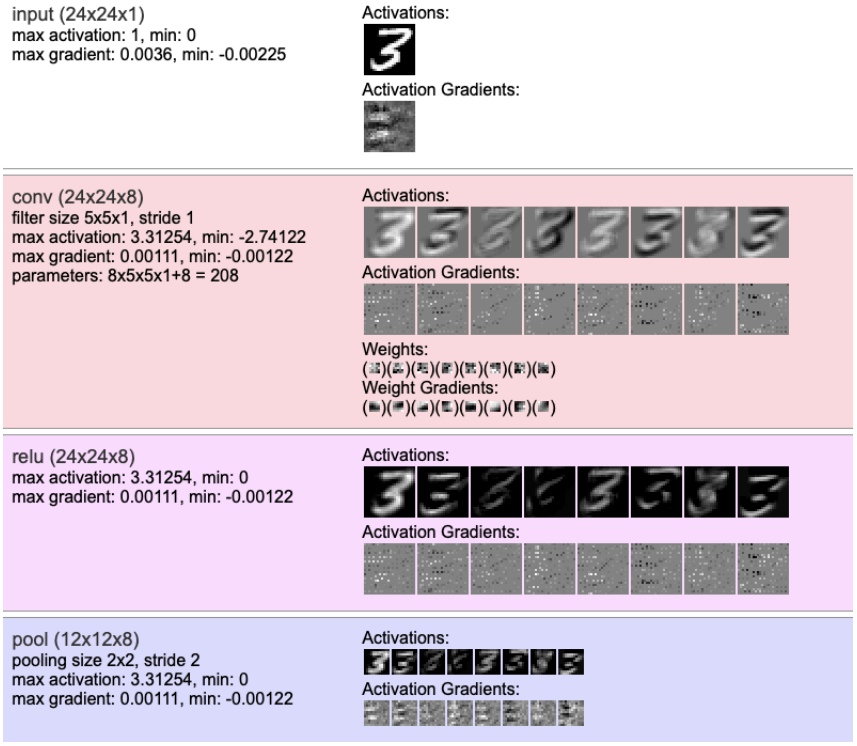
Απόκλιση στο micro/macro f_1 score μπορεί να έχω όταν τα περισσότερα δεδομένα του dataset ανήκουν σε μία μόνο κλάση και επομένως μπορεί να έχουμε καλό accuracy εάν οι προβλέψεις μας είναι καλές για αυτή την κλάση, όμως εάν δεν επιτυγχάνουμε στις υπόλοιπες κλάσεις τότε το macro f_1 δεν θα είναι υψηλό. Μία λύση για την αντιμετόπιση του συγκεκριμένου ζητήματος είναι να χρησιμοποιήσουμε το weighted average ώστε να δωθεί μεγαλύτερη βαρύτητα στην κλάση με τα περισσότερα δείγματα.

Ορισμένες φορές μας ενδιαφέρει περισσότερο είτε το precision είτε το recall. Ένα Παράδειγμα που μας ενδιαφέρει το recall είναι σε ιατρικές προβλέψεις όπου μεγαλύτερη σημασία έχει να εντοπίσουμε όλους τους ασθενείς που πάσχουν από μία ασθένεια και να τους ενημερώσουμε παρά να να γίνει και μία λάθος διάγνωση η οποία θα αναιρεθεί σε μετέπειτα εξετάσεις. Μια περίπτωση όπου το precision μας ενδιαφέρει περισσότερο είναι για παράδειγμα σε κάποιο σύστημα συστάσεων σχετικά με την επόμενη πρόταση στον χρήστη κάποιας εφαρμογής όπως πχ στο netflix. Εκεί θέλουμε περισσότερο να κάνουμε σωστές προτάσεις παρόλο που υπάρχουν και πολλές ακόμα προτάσεις τις οποίες θα μπορούσαμε να κάνουμε. Επομένως σε τέτοιου είδους περιπτώσεις θα πρέπει να εξετάζουμε την ιδιαίτερη φύση του προβλήματος και να μην αρκεστούμε σε μια πιο γενική μετρική όπως το f_1 ή το accuracy.

Κύριο Μέρος Εργαστηρίου

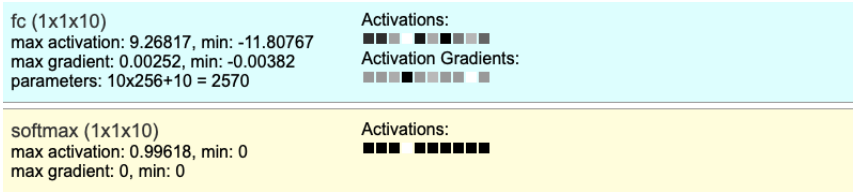
Βήμα 7

Σε αυτό το βήμα θα εκπαιδεύσουμε ένα συνελικτικό νευρωνικό δίκτυο (CNN) με είσοδο τα φασματογραφήματα, τα οποία θα επεξεργαστούμε ως εικόνες.
α) Εκπαιδεύουμε ένα CNN στο MNIST και παρατηρούμε τι μαθαίνουν τα διάφορα layers:



- Convolutional layer: Μαθαίνει low-level features όπως ακμές και blobs
- ReLU layer: Εισάγει μια μη-γραμμικότητα, μηδενίζοντας τα αρνητικά pixel και δίνει μια πιο σαφή έξοδο του Conv
- Max Pooling layer: Ρίχνει τη διάσταση του activation, με σκοπό τη μείωση των παραμέτρων εκπαίδευσης

Μετά από ένα αριθμό επαναλήψεων του παραπάνω block από layers, καταλήγουμε σε Fully Connected layer(s) και Softmax.



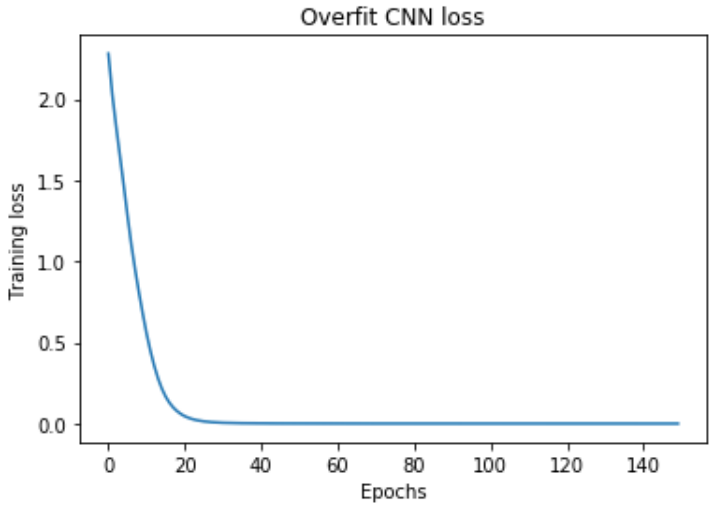
- Fully Connected layer: Μετατρέπει την πρόβλεψη σε 1D vector με μέγεθος ίσο με τον αριθμό των κλάσεων κατηγοριοποίησης. Σε αυτό το σημείο, η έξοδος είναι απλά pixels που αναπαριστούν high-level features
- Softmax layer: Μετατρέπει τον 1D-vector σε πιθανότητες και διαλέγει την πιο πιθανή κλάση

β) Υλοποιούμε ένα δικό μας CNN με τα εξής διαδοχικά layers:

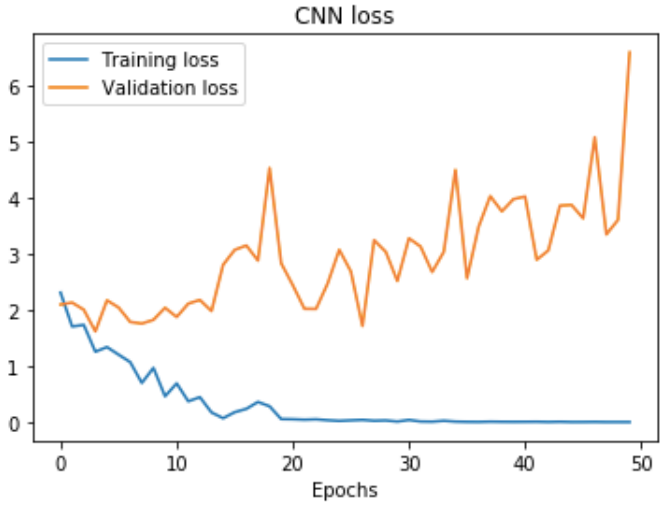
Layer	Channels	Output Channels	Kernel Size
Conv 1	1	6	5
Batch Norm 1	-	-	-
ReLU 1	-	-	-
Max Pool 1	-	-	2
Conv 2	6	16	5
Batch Norm 2	-	-	-
ReLU 2	-	-	-
Max Pool 2	-	-	2
FC 1	13456	120	-
FC 2	120	84	-
FC 3	84	10	-

γ) Το Batch Normalization layer επιταχύνει την εκπαίδευση του νευρωνικού μειώνοντας το internal covariant shift. Ουσιαστικά εφαρμόζει κανονικοποίηση σε κάθε mini-batch, μετατρέποντας τη μέση τιμή και τη διασπορά του σε επιθυμητές τιμές. Έτσι, μπορούμε να χρησιμοποιήσουμε υψηλότερο learning rate χωρίς να δώσουμε πολύ βάρος στην αρχικοποίηση.
Οι λειτουργίες των υπόλοιπων layers εξηγούνται στο ερώτημα (**α**).

δ) Παρακάτω βλέπουμε το training του δικτύου σε ένα μονο batch, κάνοντας επιτηδευμένα overfitting. Παρατηρούμε ότι το loss τείνει στο μηδέν, άρα το δίκτυο εκπαιδεύεται.



ε) Παρακάτω βλέπουμε το training του CNN μας στο dataset με τα beat mel spectograms.



Παρατηρούμε ότι το training loss μειώνεται φυσιολογικά, αλλά το validation loss αυξάνεται, γεγονός που δείχνει ότι το δίκτυο κάνει λίγο overfitting στο training set.

Παρακάτω βλέπουμε την αξιολόγησή του στο test set και τη συγκρίνουμε με την επίδοση του LSTM που εκπαιδεύσαμε στο βήμα 5α:

Accuracy: 0.3443478260869565

	precision	recall	f1-score	support
0.0	0.18	0.15	0.16	40
1.0	0.44	0.42	0.43	40
2.0	0.33	0.38	0.35	80
3.0	0.41	0.44	0.42	80
4.0	0.44	0.50	0.47	40
5.0	0.13	0.07	0.10	40
6.0	0.48	0.56	0.52	78
7.0	0.08	0.07	0.08	40
8.0	0.34	0.34	0.34	103
9.0	0.19	0.15	0.17	34
accuracy			0.34	575
macro avg	0.30	0.31	0.30	575
weighted avg	0.33	0.34	0.33	575

Επίδοση CNN στα beat synced mel spectrograms

Accuracy: 0.08521739130434783

	precision	recall	f1-score	support
0.0	0.00	0.00	0.00	40
1.0	0.07	0.55	0.12	40
2.0	0.00	0.00	0.00	80
3.0	0.15	0.25	0.19	80
4.0	0.00	0.00	0.00	40
5.0	0.00	0.00	0.00	40
6.0	0.00	0.00	0.00	78
7.0	0.00	0.00	0.00	40
8.0	0.12	0.02	0.03	103
9.0	0.05	0.15	0.08	34
accuracy			0.09	575
macro avg	0.04	0.10	0.04	575
weighted avg	0.05	0.09	0.04	575

Επίδοση LSTM στα beat synced mel spectrograms

Παρατηρούμε ότι το CNN παρουσιάζει πολύ καλύτερα αποτελέσματα στο classification report και είχε καλύτερο training, δηλαδή έπεφτε πολύ περισσότερο και πιο ομάλα το loss. Βέβαια, ήταν σχετικά υψηλό το validation loss, οπότε έγινε overfitting σε ένα βαθμό. Συμπεραίνουμε ότι το LSTM δεν είναι κατάλληλο για το συγκεκριμένο dataset, εκτός αν κάνουμε πολύ εξαντλητικό fine-tuning και εκτεταμένο training.

Βήμα 8

Στο συγκεκριμένο βήμα προσαρμώσαμε τα μοντέλα cnn και lstm έτσι ώστε να εκτιμούν τα χαρακτηριστικά valence, energy και danceability δοθέντως του συνόλου δεδομένων multitask-dataset. Για να το επιτύχουμε αυτό προσαρμώσαμε το νευρωνικό μας, ώστε στο τελικό layer να δίνει μόνο μία έξοδο (από 10 πριν) με την εκτίμηση της τιμής της παραμέτρου. Και για τις τρεις παραμέτρους εκπαιδεύσαμε τα νευρωνικά με labels τις πραγματικές τιμές αναφοράς τους.
Σαν μετρική εκτίμησης σφάλματος χρησιμοποιήσαμε το στατιστικό μέγεθος spearman correlation μεταξύ των εκτιμήσεων και των πραγματικών τιμών. Οι τιμές αυτής της μετρικής κυμαίνονται μεταξύ -1 και 1 και όταν η τιμή είναι θετική σημαίνει ότι η μεταβλητή y τείνει να αυξάνεται όταν αυξάνεται και η τιμή της x ενώ όταν η τιμή της συσχέτισης είναι αρνητική, σημαίνει ότι όταν η μεταβλητή x τίνει να αυξάνεται η μεταβλητή y τίνει να μειώνεται.

Εν προκειμένω, η μέση συσχέτιση που προκύπτει και για τις τρεις παραμέτρους με χρήση δικτύου CNN είναι 0.1633 ενώ αντίστοιχα για τα lstm 0.0342 κάτι που δείχνει ότι οι προβλέψεις από τα cnn έχουν μια τάση να βρίσκονται πιο κοντά στις πραγματικές τιμές σε σχέση με τις προβλέψεις από τα LSTM.

Βήμα 9

α) Το paper [5] από την εκφώνηση είναι η ανακάλυψη ότι τα πρώτα layers των νευρωνικών δικτύων μαθάινουν πάντα χαρακτηριστικά παρόμοια με τα Gabor φίλτρα και τα color blobs. Οπότε όταν έχουμε στη διάθεσή μας μικρό όγκο δεδομένων, μπορούμε να χρησιμοποιήσουμε transfer learning από ένα άλλο καλώς εκπαιδευμένο νευρωνικό δίκτυο. Έτσι, μπορούμε να μεταφέρουμε τα χαρακτηριστικά αυτών των layer στο δικό μας δίκτυο, αφού αν το εκπαιδεύαμε από την αρχή, θα μάθαινε χαρακτηριστικά πολύ παρόμοια σε αυτά που του μεταφέρουμε.

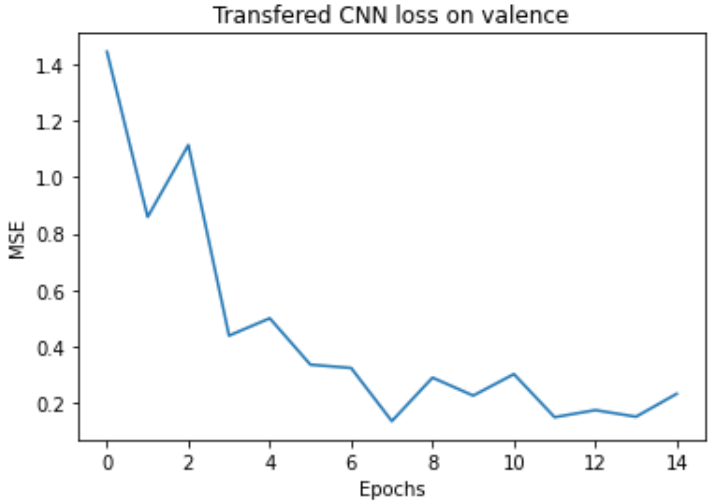
β) Θα επιλέξουμε να εκπαιδεύσουμε το μοντέλο του βήματος 7, δηλαδή το συνελικτικό δίκτυο (CNN), αφού θέλουμε ένα δίκτυο στο οποίο να ταιριάζει η παραπάνω λογική του transfer learning. Το LSTM εκπαιδεύεται σε χρονοσειρές, οπότε δεν είναι φανερό αν τα layer του, μαθαίνουν παρόμοια με τα παραπάνω χαρακτηριστικά.

γ) Θα χρησιμοποιήσουμε το εκπαιδευμένο CNN μας στο fma genre spectrograms, που παρουσίαζε ικανοποιητικά αποτελέσματα και έχει την εξής δομή:

```
(conv1): Conv2d(1, 6, kernel_size=(5, 5), stride=(1, 1))
(batch_norm1): BatchNorm2d(6, eps=1e-05, momentum=0.1, affine=True,
track_running_stats=True)
(conv2): Conv2d(6, 16, kernel_size=(5, 5), stride=(1, 1))
(batch_norm2): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True,
 track_running_stats=True)
(fc1): Linear(in_features=148480, out_features=120, bias=True)
(fc2): Linear(in_features=120, out_features=84, bias=True)
(fc3): Linear(in_features=84, out_features=10, bias=True)
```

δ) Για να κάνουμε τη μεταφορά μάθησης (transfer learning), αρχικά αλλάζουμε το τελευταίο fully connected layer ώστε να δίνει έξοδο διάστασης 1 αντί για 10, αφού τώρα το πρόβλημά μας αφορά σε regression και προσεγγίζουμε μια τιμή. Έπειτα, κάνουμε freeze όλες τις παραμέτρους των προηγούμενων layer, δηλαδή δεν τις κάνουμε train, αλλά κάνουμε μόνο στο τελευταίο layer.

Εκτελέσαμε την παραπάνω διαδικασία για τον άξονα valence και παρακάτω φαίνεται η εξέλιξη του σφάλματος κατά το training:



Το spearman correlation μεταξύ των προβλέψεων για των labels προέκυψε -0.00787. Αυτό σημαίνει ότι υπάρχει μια ελαφριά τάση των προβέψεων να παίρνουν τιμές μικρότερες καθώς οι πραγματικές τιμές αυξάνονται. Συγκρίνοντας τα αποτελέσματα του μοντέλου που εφαρμόσαμε Transfer Learning με αυτό που εκπαιδεύτηκε από την αρχή, βλέπουμε ότι με την μέθοδο Transfer Learning έχουμε μικρότερη σύγκλιση σε σχέση με το μοντέλο του βήματος 8, (η μετρική spearman correlation γίνεται αρνητική) και αυτό πιθανώς οφείλεται στο ότι η φύση του προβλήματος είναι πιο ειδική και χρειάζεται περισσότερα layers εκπαιδευμένα συγκεκριμένα επάνω σε αυτό το dataset για να κάνει καλύτερες εκτιμήσεις.

Βήμα 10

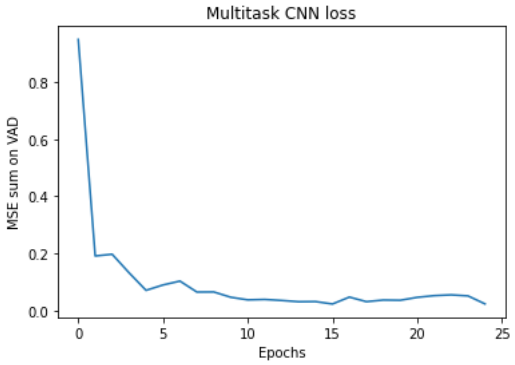
α) Το paper [7] από την εκφώνηση δείχνει τα πλεονεκτήματα του multi-task learning δηλαδή της εκπαίδευσης ενός μοντέλου σε πολλά διαφορετικά προβλήματα ταυτόχρονα. Για κάθε πρόβλημα, προκειμένου να φτιάξουμε βαθειά μοντέλα, χρειάζεται εκτεταμένη έρευνα για το fine-tuning. Εκπαιδεύοντας ένα μοντέλο στα διαφορετικά αυτά προβλήματα, δείχνουν οτι κάθε block είναι σημαντικό και βοηθάει σε κάθε πρόβλημα και αν αυτό δεν ισχύει σε κάποιες περιπτώσεις, τότε σίγουρα η παρουσία του δεν σημαίνει χειροτέρευση της επίδοσης.

β) Αρχικά, προκειμένου να επεξεργαζόμαστε δεδομένα με πολλά labels ταυτόχρονα (valence, arousal, danceability), δημιουργήσαμε μια κλάση Dataset για το φόρτωμα των δεδομένων στο Multitask πρόβλημα.

Έπειτα, δημιουργήσαμε μια κλάση για τη συνάρτηση σφάλματος **multitask_loss_function**, η οποία παίρνει τις εκτιμήσεις του CNN για τους 3 άξονες και υπολογίζει τα MSE ξεχωριστά μεταξύ των προβλέψεων και των labels. Αυτό που επιστρέφει είναι το άθροισμα των επιμέρους losses. Η κλάση κληρονομεί από την nn.Module, οπότε τα παραπάνω έγιναν μέσα σε μια μέθοδο forward, ενώ το backpropagation έγινε από την default μέθοδο της pytorch.

γ) Η αρχιτεκτονική του μοντέλου που εκπαιδεύτηκε, καθώς και η εξέλιξη του training loss, φαίνονται παρακάτω:

```
(conv1): Conv2d(1, 6, kernel_size=(5, 5), stride=(1, 1))
(batch_norm1): BatchNorm2d(6, eps=1e-05, momentum=0.1, affine=True,
 track_running_stats=True)
(conv2): Conv2d(6, 16, kernel_size=(5, 5), stride=(1, 1))
(batch_norm2): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True,
 track_running_stats=True)
(fc1): Linear(in_features=148480, out_features=120, bias=True)
(fc2): Linear(in_features=120, out_features=84, bias=True)
(fc_valence): Linear(in_features=84, out_features=1, bias=True)
(fc_arousal): Linear(in_features=84, out_features=1, bias=True)
(fc_danceability): Linear(in_features=84, out_features=1, bias=True)
```



Αξιολογούμε το μοντέλο στο test set και προκύπτει spearman correlation 0.5822 . Παρατηρούμε ότι με το multitask learning επιτυγχάνουμε τα καλύτερα αποτελέσματα για την μετρική του correlation, αφού είναι αρκετά υψηλά και θετική άρα σημαίνει ότι ακολουθεί καλύτερα τις πραγματικές και αυτό οφείλεται στο ότι το μοντέλο περιέχει περισσότερη συνδυαστική πληροφορία και έτσι το βοηθάει να αποφεύγει τα λάθη από παρόμοιες καταστάσεις. Τα πλεονεκτήματα της συγκεκριμένης αρχιτεκτονικής επιβεβαιώνονται και από το paper [7].