ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΑΝΑΓΝΩΡΙΣΗ ΠΡΟΤΥΠΩΝ

(2021-2022)

3ο Εργαστηριακό Project

Θέμα: Αναγνώριση Είδους και Εξαγωγή Συναισθήματος από Μουσική

Ονοματεπώνυμο:

- Χρήστος Τσούφης

Αριθμός Μητρώου:

- 031 17 176

Στοιχεία Επικοινωνίας:

- el17176@mail.ntua.gr

Περιγραφή

Σκοπός είναι η αναγνώριση του είδους και η εξαγωγή συναισθηματικών διαστάσεων από φασματογραφήματα (spectrograms) μουσικών κομματιών. Σας δίνονται 2 σύνολα δεδομένων, το Free Music Archive (FMA) genre με 3834 δείγματα χωρισμένα σε 20 κλάσεις (είδη μουσικής) και τη βάση δεδομένων (dataset) multitask music με 1497 δείγματα με επισημειώσεις (labels) για τις τιμές συναισθηματικών διαστάσεων όπως valence, energy και danceability. Τα δείγματα είναι φασματογραφήματα, τα οποία έχουν εξαχθεί από clips 30 δευτερολέπτων από διαφορετικά τραγούδια.

Ειδικότερα, η εργασία επικεντρώνεται στην ανάλυση των φασματογραφημάτων με χρήση βαθιών αρχιτεκτονικών με συνελικτικά νευρωνικά δίκτυα (CNN) και αναδρομικά νευρωνικά δίκτυα (RNN).

Η άσκηση χωρίζεται σε 5 μέρη:

1) Ανάλυση των δεδομένων και εξοικείωση με τα φασματογραφήματα.

2) Κατασκευή ταξινομητών για το είδος της μουσικής πάνω στη βάση δεδομένων (dataset) FMA.

3) Κατασκευή regression μοντέλων για την πρόβλεψη valence, energy και danceability πάνω στη Multitask βάση δεδομένων.

4) Χρήση προηγμένων τεχνικών εκπαίδευσης (transfer - multitask) learning για τη βελτίωση των αποτελεσμάτων.

5) (*Προαιρετικά*) Υποβολή των μοντέλων στο Kaggle competition του εργαστηρίου και σύγκριση των αποτελεσμάτων [0].

Τα δεδομένα είναι διαθέσιμα στο [1]. Μπορείτε να κάνετε χρήση των Kaggle kernels για να έχετε πρόσβαση σε δωρεάν GPUs: [2].

Τεχνολογίες & Τρόπος Εκτέλεσης εφαρμογής

Η παρούσα εργασία υλοποιήθηκε σε ένα Python περιβάλλον και συγκεκριμένα με την χρήση των Notebooks του Google Colab & Kaggle τα οποία στην συνέχεια έγιναν export σε .py αρχεία.

Βιβλιοθήκες:

librosa, numpy, pytorch, scikit-learn

Δομή:

Αποτελείται από το αρχείο `patrec_lab3.py` που περιέχει όλα τα βήματα και τις συναρτήσεις. Η εκτέλεση των αρχείων γίνεται μέσω του `Colab` ή `Kaggle` αφού πρώτα φορτωθούν τα κατάλληλα αρχεία του φακέλου (`multitask-affective-music-lab`) που θα χρησιμοποιηθούν ως input με την διαδικασία που περιγράφεται στην αρχή του κώδικα.

Περισσότερες πληροφορίες στα σχόλια στο helios.

Εκτέλεση

Η προπαρασκευή αφορά την αναγνώριση είδους μουσικής με βάση το φασματογράφημα (spectrogram). Όπως εξετάστηκε και στο εργαστήριο 2, το φασματογραφήματα είναι μια οπτική αναπαράσταση της μεταβολής του συχνοτικού περιεχομένου ενός σήματος με το χρόνο (time - frequency distribution), όπου η εξαγόμενη εικόνα αναπαριστά την ενέργεια του σήματος για διαφορετικές ζώνες συχνοτήτων και χρονικά παράθυρα.

Προπαρασκευή

Βήμα 0 (Εξοικείωση με Kaggle kernels)

Ανοίξτε ένα (private) Kaggle kernel στη σελίδα [2]. Τα δεδομένα μπορούν να φορτωθούν όπως φαίνεται από το notebook. Τρέξτε την εντολή `os.listdir("../input/data/data/")` *για να εξερευνήσετε τους υποφακέλους, δοκιμάστε να ενεργοποιήσετε και να απενεργοποιήσετε τη GPU και κάντε commit τις αλλαγές σας.*

Βήμα 1 (Εξοικείωση με φασματογραφήματα στην κλίμακα mel)

Τα δεδομένα που θα χρησιμοποιήσετε στην προπαρασκευή είναι ένα υποσύνολο του Free Music Archive (FMA) dataset. Το FMA είναι μια βάση δεδομένων από ελεύθερα δείγματα (clips) μουσικής με επισημειώσεις ως προς το είδος της μουσικής.

Έχουμε εξάγει τα φασματογραφήματα και τις επισημειώσεις τους στο φάκελο:

“`../input/data/data/fma_genre_spectrogram`*”*

Το αρχείο “`../input/data/data/fma_genre_spectrograms/train_labels.txt`*” περιέχει γραμμές του στη μορφή “*`spec_file label`*”.*

(α) *Διαλέξτε δύο τυχαίες γραμμές με διαφορετικές επισημειώσεις (labels). Τα αντίστοιχα αρχεία βρίσκονται στο φάκελο “*`./input/data/data/fma_genre_spectrograms/train`*”*

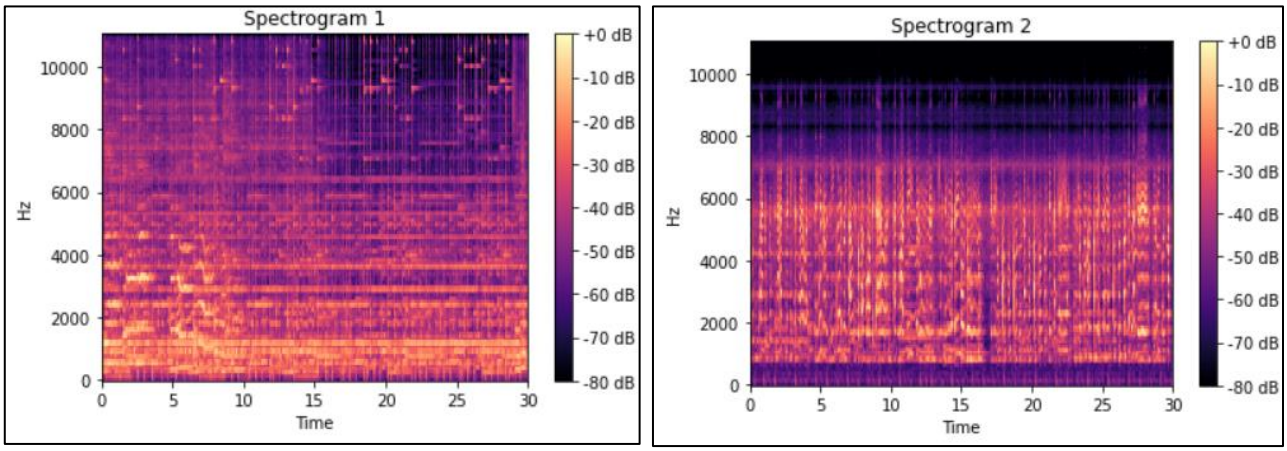
(β) *Διαβάστε τα αρχεία και πάρτε το φασματογράφημα σε κλίμακα mel σύμφωνα με τις οδηγίες [1].*

(γ) *Απεικονίστε τα φασματογραφήματα για τα διαφορετικά labels με χρήση της συνάρτησης* `librosa.display.specshow`*. Σχολιάστε τι πληροφορία σας δίνουν και τις διαφορές για δείγματα που αντιστοιχούν σε διαφορετικές επισημειώσεις. (Υπόδειξη: συχνότητα στον κατακόρυφο άξονα, χρόνος στον οριζόντιο).*

(α) Πρώτα, διαβάζονται όλα τα ονόματα αρχείων που ανήκουν στο train set και από τον κατάλογο "`fma_genre_spectrograms/train`" επιλέγονται τυχαία δύο δείγματα.

(β) Έπειτα, με βάση τις παραπάνω οδηγίες γίνεται το διάβασμα των αρχείων ώστε να προκύψουν τα mel spectrograms για τα δείγματα που επιλέχθηκαν.

(γ) Ύστερα, με χρήση της συνάρτησης `librosa.display.specshow` γίνεται η απεικόνιση ως φασματογραφήματα σε κλίμακα mel.



Σχολιασμός:

Παρατηρείται ότι τα αρχεία που έχουν διαφορετικές labels έχουν υψηλές τιμές σε διαφορετικές συχνότητες. Αυτό φαίνεται να είναι βοηθητικό για τον διαχωρισμό μεταξύ τους. Γενικότερα, τα spectrograms χρησιμεύουν για την οπτικοποίηση της έντασης κάθε συχνότητας στο χρόνο.

Βήμα 2 (Συγχρονισμός φασματογραφημάτων στο ρυθμό της μουσικής (beat-synced spectrograms)

(α) *Τυπώστε τις διαστάσεις των φασματογραφημάτων του Βήματος 1.*

- Πόσα χρονικά βήματα έχουν;

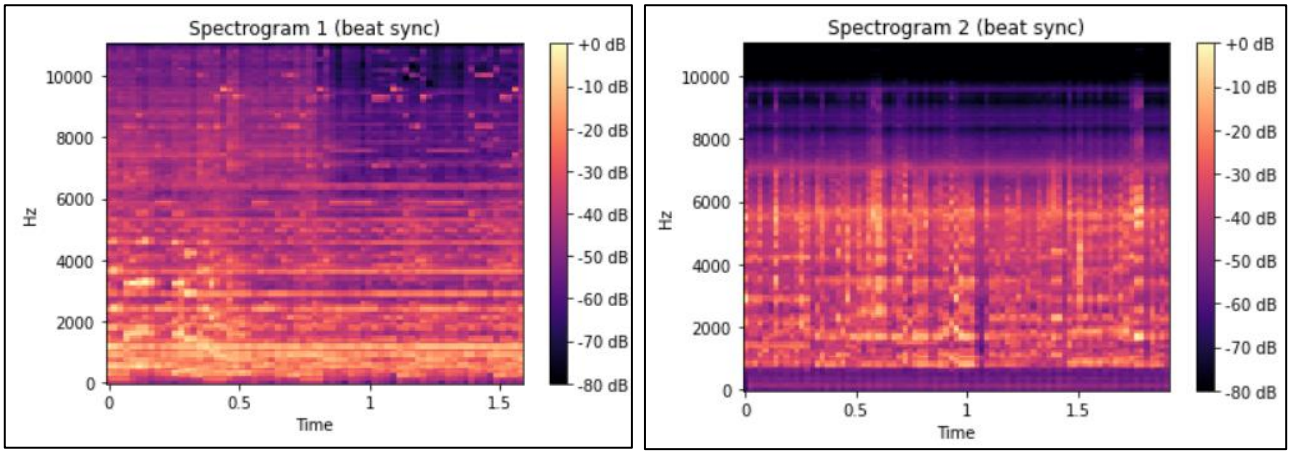
- Είναι αποδοτικό να εκπαιδεύσετε ένα LSTM πάνω σε αυτά τα δεδομένα;

- Γιατί;

(β) *Ένας τρόπος να μειώσουμε τα χρονικά βήματα είναι να συγχρονίσουμε τα φασματογραφήματα πάνω στο ρυθμό. Για αυτό το λόγο παίρνουμε τη διάμεσο (median) ανάμεσα στα σημεία που χτυπάει το beat της μουσικής. Τα αντίστοιχα αρχεία δίνονται στο φάκελο "`../input/data/data/fma_genre_spectrograms_beat`". Επαναλάβετε τα βήματα του Βήματος 1 για αντίστοιχα beat synced spectrograms και σχολιάστε τις διαφορές με τα αρχικά.*

(α) Οι διαστάσεις των φασματογραφημάτων του Βήματος 1 είναι: `(128, 1291)` & `(128, 1291)`. Όπου, ο 1ος αριθμός περιγράφει τις κλίμακες συχνότητας και ο 2ος αριθμός τα χρονικά βήματα/σκάλες.

(β) Ένας τρόπος μείωσης των χρονικών βημάτων είναι ο συγχρονισμός των φασματογραφημάτων πάνω στο ρυθμό με χρήση της διαμέσου (median) ανάμεσα στα σημεία που χτυπάει το beat της μουσικής. Για αυτό το λόγο, χρησιμοποιούνται τα data που είναι σχετικά με beat synchronized αρχεία τα οποία απεικονίζονται στην συνέχεια.



Σχολιασμός:

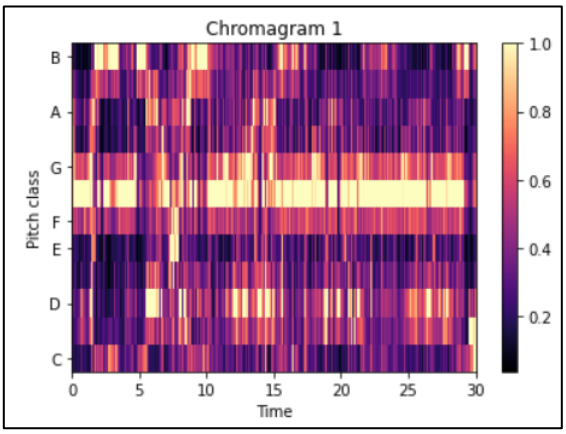
(α) Για την εκπαίδευση ενός LSTM, εξαιτίας του μεγάλου sequence των data, είναι πιθανό να προκληθεί μεγάλη αύξηση στην πολυπλοκότητα επειδή υπάρχει πολλή και επικαλυπτόμενη πληροφορία. Αφενός, αυτό δεν είναι αποδοτικό διότι τα data είναι ακόμα raw και μη επεξεργασμένα οπότε μπορεί να μην περιέχουν όλα χρήσιμη πληροφορία και αφετέρου, λόγω των μεγάλων sequence, μπορεί να προκληθούν vanishing gradients. Στην συνέχεια, θα φανεί ότι η πιο χρήσιμη πληροφορία για το classification σχετίζεται με τον ρυθμό της μουσικής.

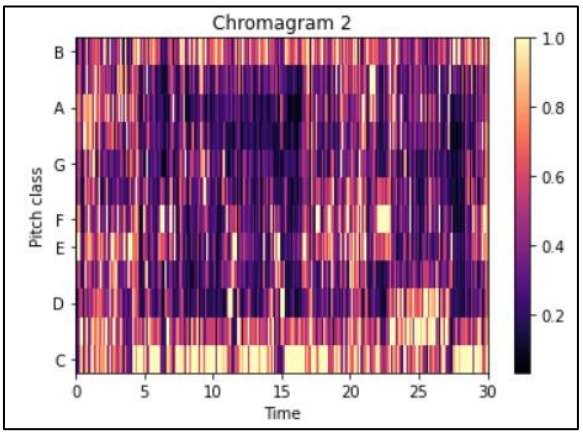
(β) Παρατηρώντας τα spectrograms φαίνεται ότι είναι σχεδόν ίδια με το Βήμα 1 όμως με μειωμένη ευκρίνεια και επιπλέον από τις διαστάσεις φαίνεται ότι οι συχνοτικές συνιστώσες παρέμειναν ίδιες σε πλήθος ενώ τα χρονικά βήματα μειώθηκαν.

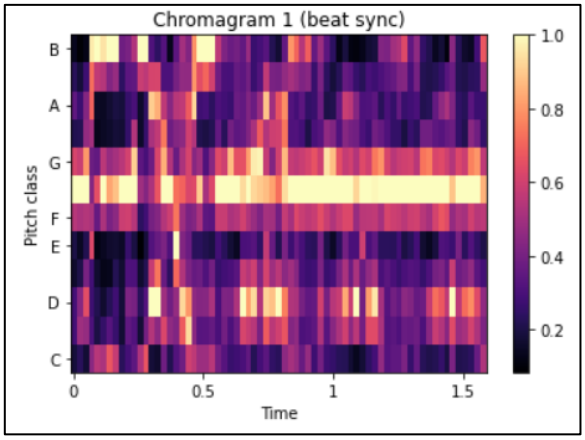
Βήμα 3 (Εξοικείωση με χρωμογραφήματα)

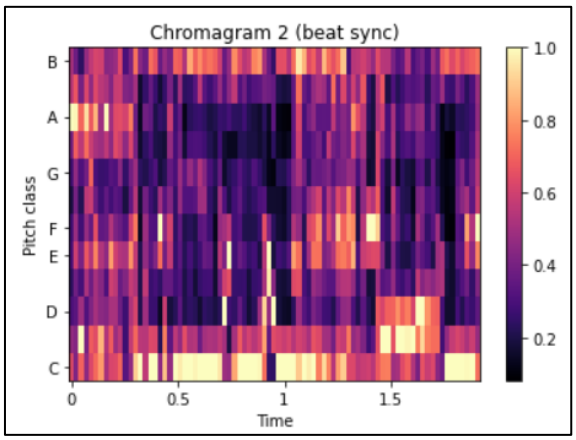
Τα χρωμογραφήματα (chromagrams) απεικονίζουν την ενέργεια του σήματος μουσικής για τις ζώνες συχνοτήτων που αντιστοιχούν στις δώδεκα διαφορετικές νότες της κλίμακας κλασικής μουσικής {C, C♯, D, D♯, E, F, F♯, G, G♯, A, A♯, B} και μπορούν να χρησιμοποιηθούν ως εργαλείο για την ανάλυση της μουσικής αναφορικά με τα αρμονικά και μελωδικά χαρακτηριστικά της ενώ επίσης είναι αρκετά εύρωστα και στην αναγνώριση των αλλαγών του ηχοχρώματος και των οργάνων (μπορεί να θεωρήσει κάποιος ότι το χρωμογράφημα είναι ένα φασματογράφημα modulo την οκτάβα). Επαναλάβετε τα υποερωτήματα από τα Βήματα 1 και 2 για τα χρωμογραφήματα των αντίστοιχων αρχείων.

Σε αυτό το βήμα γίνεται χρήση των χρωμογραφημάτων τα οποία απεικονίζουν την ενέργεια του σήματος μουσικής για τις ζώνες που αντιστοιχούν στις δώδεκα διαφορετικές νότες της κλίμακας κλασσικής μουσικής και συνάμα αποτελούν καλά χαρακτηριστικά για την αναγνώριση αλλαγής στο ηχόχρωμα και στα μουσικά όργανα. Συνεπώς, προκύπτουν τα ακόλουθα:









Σχολιασμός:

Παρατηρείται ότι τα παραπάνω chromagrams απεικονίζουν την ένταση σε κάθε κλάση μουσικών τόνων σε αντίθεση με τις συχνοτικές συνιστώσες που εξετάστηκαν στο προηγούμενο Βήμα. Πλέον, οι διαφορές μεταξύ των δειγμάτων είναι πιο ευδιάκριτες. Έτσι, στο "Classical" sample διακρίνονται οι νότες Μι και Φα, ενώ στο "Folk" sample οι νότες Ντο, Ντο# και Σι. Συνεπώς, τα samples που είναι συντονισμένα με τον ρυθμό έχουν μικρότερη διάσταση χωρίς ωστόσο να αλλοιώνουν την σημαντική πληροφορία.

Βήμα 4 (Φόρτωση και ανάλυση δεδομένων)

Χρησιμοποιήστε το βοηθητικό κώδικα [8].

(α) *Στο βοηθητικό κώδικα σας παρέχεται έτοιμη μια υλοποίηση ενός PyTorch Dataset η οποία διαβάζει τα δεδομένα και σας επιστρέφει τα δείγματα. Μελετήστε τον κώδικα και τα δείγματα που επιστρέφει και σχολιάστε τις λειτουργίες που εκτελούνται.*

(β) *Στον κώδικα που σας δίνουμε συγχωνεύουμε κλάσεις που μοιάζουν μεταξύ τους και αφαιρούμε κλάσεις που αντιπροσωπεύονται από πολύ λίγα δείγματα.*

(γ) *Σχεδιάστε δύο ιστογράμματα που θα δείχνουν πόσα δείγματα αντιστοιχούν σε κάθε κλάση, ένα πριν από τη διαδικασία του βήματος 4β και ένα μετά.*

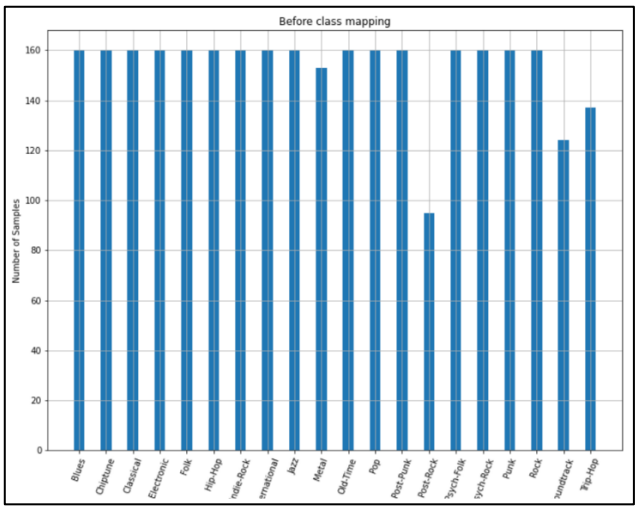
(α) Έπειτα από την μελέτη του κώδικα, προκύπτουν τα εξής πορίσματα:

- Υπάρχει ένας LabelTransformer που κάνει την μετατροπή των labels: strings ⇆ integers.
- Υπάρχει ένας PaddingTransformer που κάνει padding στα δείγματα για να έχουν όλα το ίδιο μήκος (max length) ώστε να γίνουν tensors.
- Υπάρχει το SpectrogramDataset το οποίο διαβάζει όλα τα δεδομένα ενός path, τα τροποποιεί και τελικά δημιουργεί ένα Dataset με tuples (sample, labels, length), όπου ως length ορίζεται το πραγματικό μήκος του δείγματος πριν την τροποποίηση.

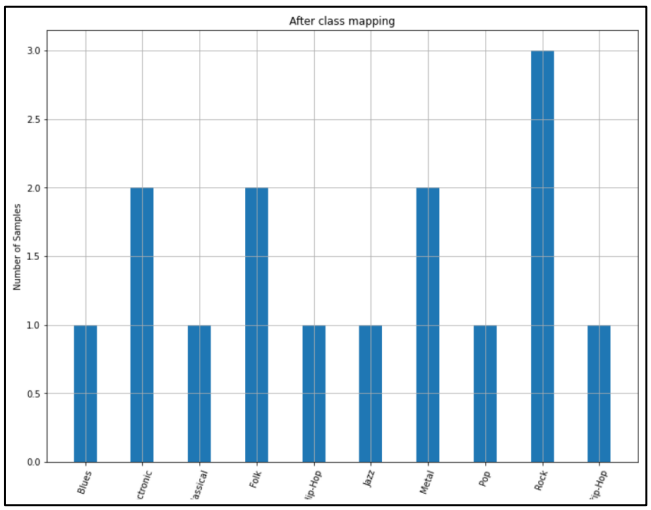
(β) Στον υπό μελέτη κώδικα παρατηρεί κανείς ότι είτε συγχωνεύονται μερικές κλάσεις με κάποιες άλλες που μοιάζουν μεταξύ τους (λ.χ. Post-rock και Rock) είτε διαγράφονται αν αντιπροσωπεύονται από πολύ λίγα δείγματα διότι το σύνολο των δεδομένων προς εκπαίδευση είναι περιορισμένο. Με αυτό τον τρόπο, οι κλάσεις που μοιάζουν μεταξύ τους και οι κλάσεις που η μια είναι υποκλάση της άλλης παύουν να δυσκολεύουν το classification και τελικά αυξάνεται το accuracy. Για την επίτευξη καλής γενίκευσης, αυτή η προεπεξεργασία είναι απαραίτητη.

(γ) Τα ιστογράμματα που δείχνουν πόσα δείγματα αντιστοιχούν σε κάθε κλάση είναι τα εξής:

Πριν το Βήμα 4β



Μετά το Βήμα 4β



Βήμα 5 (Αναγνώριση μουσικού είδους με LSTM)

Με τη βοήθεια του κώδικα που υλοποιήσατε στη δεύτερη άσκηση:

(α) *Προσαρμόστε τον κώδικα του LSTM που υλοποιήσατε στη 2η άσκηση για να δέχεται ως είσοδο τα φασματογραφήματα από το Pytorch dataset του βήματος 4.*

(β) *Για να επισπεύσετε τη διαδικασία ανάπτυξης και αποσφαλμάτωσης των μοντέλων σας, στη συνάρτηση train() που εκπαιδεύει το μοντέλο σας προσθέστε μια boolean παράμετρο overfit_batch. Όταν η overfit_batch είναι False το δίκτυο εκπαιδεύεται κανονικά. Όταν είναι True, θα πραγματοποιεί υπερεκπαίδευση του δικτύου σε ένα μικρό σύνολο από batches (3-4).*

- Υπερεκπαίδευση του δικτύου σε ένα batch: Μια καλή πρακτική κατά την ανάπτυξη νευρωνικών είναι να βεβαιωθούμε ότι το δίκτυο μπορεί να εκπαιδευτεί (τα gradients γυρνάνε πίσω κτλ). Ένας γρήγορος τρόπος για να γίνει αυτό είναι να επιλέξουμε τυχαία ένα πολύ μικρό υποσύνολο των δεδομένων (ένα batch) και να εκπαιδεύσουμε το δίκτυο για πολλές εποχές πάνω σε αυτό. Αυτό που περιμένουμε να δούμε είναι το σφάλμα εκπαίδευσης να πάει στο 0 και το δίκτυο να κάνει overfit (δείτε και τα [12], [13], [14]).

(γ) *Εκπαιδεύστε ένα LSTM [15] δίκτυο, το οποίο θα δέχεται ως είσοδο τα φασματογραφήματα του συνόλου εκπαίδευσης (train set) και θα προβλέπει τις διαφορετικές κλάσεις (μουσικά είδη) του συνόλου δεδομένων (dataset).*

(δ) *Εκπαιδεύστε ένα LSTM δίκτυο, το οποίο θα δέχεται ως είσοδο τα beat-synced spectrograms (train set) και θα προβλέπει τις διαφορετικές κλάσεις (μουσικά είδη) του συνόλου δεδομένων.*

(ε) *Εκπαιδεύστε ένα LSTM δίκτυο, το οποίο θα δέχεται ως είσοδο τα χρωμογραφήματα (train set) και θα προβλέπει τις διαφορετικές κλάσεις (μουσικά είδη) του συνόλου δεδομένων.*

(ζ) *(extra credit) εκπαιδεύστε ένα LSTM δίκτυο, το οποίο θα δέχεται ως είσοδο τα ενωμένα (concatenated) χρωμογραφήματα και φασματογραφήματα (train set) και θα προβλέπει τις διαφορετικές κλάσεις (μουσικά είδη) του συνόλου δεδομένων.*

Υπόδειξη*: Για την εκπαίδευση χρησιμοποιήστε και σύνολο επαλήθευσης (validation set).*

Υπόδειξη*: Για την εκπαίδευση ενεργοποιήστε τη GPU. Χρησιμοποιείστε Adam optimizer.*

Υπόδειξη*: Χρησιμοποιείστε την κλάση Subset του PyTorch, ώστε να μπορείτε να εκπαιδεύσετε τα μοντέλα σας σε μικρότερα υποσύνολα και να επιταχύνετε τις διαδικασίες debugging/tuning. Παράδειγμα χρήσης.*

(α) – **(β)** Για την εκπαίδευση των διαφορετικών ζητούμενων LSTM δικτύων χρησιμοποιήθηκαν τα spectrograms, τα beat-synced spectrograms και τα chromagrams των κομματιών του train set. Σκοπός της διαδικασίας αυτής είναι η πρόβλεψη του μουσικού είδους κάθε κομματιού. Για την επίσπευση της διαδικασίας ανάπτυξης και αποσφαλμάτωσης των μοντέλων, έγινε υπερεκπαίδευση των μοντέλων σε πολύ μικρά batches για πολλές epochs. Με αυτό τον τρόπο διασφαλίστηκε ότι είναι εφικτό το overfitting για ένα μικρό batch ώστε να επιτευχθεί μηδενικό error.

(γ) Spectrogram Accuracy: 12 %

(δ) Beat-Synced Spectrogram Accuracy: 40 %

(ε) Chromogram Accuracy: 22 %

Βήμα 6 (Αξιολόγηση των μοντέλων)

Αναφέρετε τα αποτελέσματα των μοντέλων από το Βήμα 5 στα ακόλουθα σύνολα αξιολόγησης (test sets)

- *"../input/data/data/fma_genre_spectrograms_beat/test_labels.txt"*
- *"../input/data/data/fma_genre_spectrograms/test_labels.txt"*

*Συγκεκριμένα **(α)** υπολογίστε το accuracy*

***(β)** υπολογίστε το precision, recall και F1-score για κάθε κλάση*

***(γ)** υπολογίστε το macro-averaged precision, recall και F1-score για όλες τις κλάσεις*

***(δ)** υπολογίστε το micro- averaged precision, recall και F1-score για όλες τις κλάσεις*

Αναφέρετε την ερμηνεία των μετρικών αυτών και σχολιάστε ποια από αυτές τις μετρικές θα επιλέγατε για την αξιολόγηση ενός ταξινομητή σε αυτό το πρόβλημα. Συγκεκριμένα εστιάστε στις ερωτήσεις

- Τι δείχνει το accuracy / precision / recall / f1-score;

- Τι δείχνει το micro / macro averaged precision / recall / f1-score;

- Πότε μπορεί να έχω μεγάλη απόκλιση ανάμεσα στο accuracy / f1-score και τι σημαίνει αυτό;

- Πότε μπορεί να έχω μεγάλη απόκλιση ανάμεσα στο micro/macro f1-score και τι σημαίνει αυτό;

- Υπάρχουν προβλήματα όπου το precision με ενδιαφέρει περισσότερο από το recall και αντίστροφα; Είναι ένα καλό accuracy / f1 αρκετό σε αυτές τις περιπτώσεις για να επιλέξω ένα μοντέλο;

***Υπόδειξη**: Χρησιμοποιήστε τη συνάρτηση `sklearn.metrics.classification_report`*

***Υπόδειξη**: Δείτε τα [9], [10], [11]*

Για την συστηματική ανάλυση των παραπάνω μοντέλων θα χρησιμοποιηθούν οι μετρικές accuracy, precision, recall και F1-score για κάθε κλάση σε κάθε LSTM δίκτυο.

Accuracy: είναι μια μετρική για την εκτίμηση ενός classification model. Ανεπίσημα, η accuracy υπολογίζεται ως ένα κλάσμα από τις προβλέψεις όπου το μοντέλο ήταν σωστό. Επίσημα, η accuracy δίνεται από το συνολικό ποσοστό σωστών απαντήσεων του δικτύου ως εξής:

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions} = \frac{TP + TN}{TP + TN + FP + FN}$$

Όπου TP = True Positives, TN = True Negatives, FP = False Positives, FN = False Negatives.

Εν προκειμένω, TP είναι το πλήθος των σωστών αναγνωρίσεων της κλάσης, TN το πλήθος των λάθος αναγνωρίσεων και αντίστοιχα για τα FP και FN.

Συνεπώς, η μετρική accuracy από μόνη της δεν δίνει την πλήρη εικόνα σε ένα class-imbalanced data set.

Η accuracy δείχνει το πλήθος των σωστών ταξινομήσεων αλλά μπορεί να είναι παραπλανητική όταν εξετάζεται μόνη της.

Precision: είναι μια μετρική που προσπαθεί να απαντήσει στο ερώτημα: "What proportion of positive identifications was actually correct". Η precision ορίζεται ως το πηλίκο του αριθμού των δειγμάτων που έγινε σωστή πρόβλεψη από το μοντέλο ότι ανήκουν σε αυτή την κλάση, προς το συνολικό αριθμό των δειγμάτων που έγινε πρόβλεψη ότι ανήκουν σε αυτή την κλάση:

$$Precision = \frac{TP}{TP + FP}$$

Δηλαδή, η precision δείχνει το ποσοστό ακρίβειας στα δείγματα που έγινε πρόβλεψη ότι ανήκαν στην κλάση. Συνήθως, χρησιμοποιείται για μοντέλα όπου αναζητούν τα FP λάθη. Λόγου χάρη, σε ένα Spam Filter, έχει σημασία να μην σημειώνονται ως spam e-mails εκείνα που δεν είναι όντως spam. Ακόμη, είναι σημαντική μετρική και πχ για τον εντοπισμό παράνομης χρήσης μια τραπεζικής κάρτας ώστε να μην γίνεται αναίτια η ακύρωσή της.

Recall: είναι μια μετρική που προσπαθεί να απαντήσει στο ερώτημα: "What proportion of actual positives was identified correctly". Η recall ορίζεται ως το πηλίκο του αριθμού των δειγμάτων του test set που έγινε πρόβλεψη ότι ανήκουν σε αυτή την κλάση και όντως ανήκαν σε αυτή, προς το συνολικό αριθμό των δειγμάτων του test set που ανήκουν στην κλάση:

$$Recall = \frac{TP}{TP + FN}$$

Δηλαδή, η recall δείχνει το ποσοστό των δειγμάτων της που έγιναν σωστά classified. Συνήθως, χρησιμοποιείται για μοντέλα που αναζητούν τα positive δείγματα, δηλαδή όταν τα FN λάθη έχουν μεγαλύτερο κόστος από τα FP λάθη. Λόγου χάρη, σε μοντέλα πρόβλεψης ασθενειών είναι σημαντικό να υπάρχει μεγάλο recall διότι έτσι ανιχνεύεται μεγάλο ποσοστό των περιπτώσεων που εντοπίζεται η ασθένεια αφού, ακόμα και αν τα FP είναι σημαντικά λάθη, δεν θα είναι κρίσιμα για τον ασθενή. Έτσι, ένας Dummy Classifier που προβλέπει διαρκώς την υπό μελέτη κλάση, θα βρίσκει recall 100.

Σημειώνεται ότι, για την εκτίμηση της πλήρους αποτελεσματικότητας ενός μοντέλου, κρίνεται σκόπιμο να εξεταστούν τόσο η precision όσο και η recall. Αυτό συμβαίνει διότι η βελτίωση της precision τυπικά δυσχεραίνει την recall και το αντίθετο.

F1-Score: είναι μια μετρική που αποτελεί τον αρμονικό μέσο μεταξύ precision και recall. Η μετρική ορίζεται ως εξής:

$$F1 - score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

Σημειώνεται ότι αυτή η μετρική είναι προτιμότερη από την accuracy όταν τα FP και FN έχουν μεγαλύτερο κόστος από τα TN και TP. Επίσης, αυτές οι μετρικές διαφέρουν και στις περιπτώσεις όπου υπάρχει imbalanced κλάσεις ή γενικά όταν το μοντέλο "πηγαίνει καλά" σε ορισμένες μόνο κλάσεις.

Η F1-score δίνει την ίδια βαρύτητα στο γινόμενο των precision και recall και χρησιμοποιείται όταν εξετάζονται οι λάθος προβλέψεις.

Εν γένει, όταν υπάρχουν πολλές κλάσεις και στόχος είναι ο υπολογισμός μιας από τις παραπάνω μετρικές, τότε προτιμάται ο υπολογισμός των μετρικών micro-average και macro-average.

Micro-average: είναι μια μετρική που αθροίζει τα μεμονωμένα TP, FP, TN, FN του συστήματος για τις διαφορετικές κλάσεις και χρησιμοποιείται στον τύπο της εκάστοτε μετρικής. Με άλλα λόγια, υπολογίζει τα επιμέρους κομμάτια που απαιτούνται αθροιστικά από όλες τις μετρικές. Δηλαδή, για micro-averaged recall ως TP προσμετράται το άθροισμα των TP από κάθε κλάση. Σημειώνεται ότι εδώ προέκυψε ίδιο με το accuracy και για αυτό απουσιάζει από τον πίνακα των αποτελεσμάτων.

Macro-average: είναι μια μετρική που υπολογίζει τον μέσο όρο από το precision και το recall του συστήματος για κάθε κλάση. Με άλλα λόγια, υπολογίζει τον μέσο όρο της μετρικής από κάθε κλάση. Δηλαδή, ως macro-averaged recall ορίζεται ο μέσος όρος των recall όλων των κλάσεων. Επομένως, αποτελεί μια ένδειξη συνολικής απόδοσης του αλγορίθμου σε πολλές κλάσεις ή σε πολλά Dataset και δεν χρησιμοποιείται για τη λήψη κάποιας απόφασης.

Εν προκειμένω, η αναλυτική απόδοση των LSTM δικτύων για τις παραπάνω μετρικές είναι η εξής:

(5γ) LSTM με είσοδο τα spectrograms του συνόλου εκπαίδευσης

	precision	recall	f1-score	support
0.0	0.00	0.00	0.00	40
1.0	0.04	0.03	0.03	40
2.0	0.16	0.29	0.21	80
3.0	0.15	0.35	0.21	80
4.0	0.00	0.00	0.00	40
5.0	0.07	0.17	0.10	40
6.0	0.00	0.00	0.00	78
7.0	0.06	0.07	0.06	40
8.0	0.31	0.08	0.12	103
9.0	0.04	0.03	0.03	34
accuracy			0.12	575
macro avg	0.08	0.10	0.08	575
weighted avg	0.11	0.12	0.10	575

(5δ) LSTM με είσοδο τα beat-synced spectrograms του συνόλου εκπαίδευσης

	precision	recall	f1-score	support
0.0	0.28	0.12	0.17	40
1.0	0.41	0.53	0.46	40
2.0	0.53	0.61	0.57	80
3.0	0.36	0.54	0.43	80
4.0	0.43	0.23	0.30	40
5.0	0.21	0.10	0.14	40
6.0	0.46	0.68	0.55	78
7.0	0.00	0.00	0.00	40
8.0	0.40	0.38	0.39	103
9.0	0.29	0.32	0.31	34
accuracy			0.41	575
macro avg	0.34	0.35	0.33	575
weighted avg	0.37	0.41	0.38	575

(5ε) LSTM με είσοδο τα chromagrams του συνόλου εκπαίδευσης

	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	0.00	0.00	0.00	80
3.0	0.21	0.61	0.32	80
4.0	0.00	0.00	0.00	40
5.0	0.00	0.00	0.00	40
6.0	0.27	0.55	0.36	78
7.0	0.00	0.00	0.00	40
8.0	0.17	0.30	0.21	103
9.0	0.00	0.00	0.00	34
accuracy			0.21	575
macro avg	0.07	0.15	0.09	575
weighted avg	0.10	0.21	0.13	575

---------- *ΤΕΛΟΣ ΠΡΟΠΑΡΑΣΚΕΥΗΣ* ----------

Βήμα 7 (2D CNN)

Ένας άλλος τρόπος για την κατασκευή ενός μοντέλου για την επεξεργασία ηχητικών σημάτων είναι να δούμε το φασματογράφημα σαν εικόνα και να χρησιμοποιήσουμε συνελικτικά δίκτυα (CNN).

(α) *Στο σύνδεσμο [19] μπορείτε να εκπαιδεύσετε απλά συνελικτικά δίκτυα και να δείτε την εσωτερική λειτουργία του δικτύου οπτικοποιώντας τις ενεργοποιήσεις (activations) των επιμέρους επιπέδων του δικτύου χωρίς προγραμματιστικό κόπο. Εκπαιδεύστε ένα δίκτυο στο MNIST και παρατηρήστε τη λειτουργία των ενεργοποιήσεων κάθε επιπέδου. Σχολιάστε τις επιμέρους λειτουργίες, τι φαίνεται να μαθαίνει το δίκτυο και δώστε κατάλληλα screenshots στην αναφορά.*

(β)*Υλοποιήστε ένα 2D CNN με 4 επίπεδα (layers) που θα επεξεργάζεται το φασματογράφημα σαν μονοκάναλη εικόνα, να το εκπαιδεύσετε στο train + validation set και να αναφέρετε τα αποτελέσματα στο test set. Κάθε επίπεδο θα πραγματοποιεί τις εξής λειτουργίες (operations) με αυτή τη σειρά:*

1) 2D convolution

2) Batch normalization

3) ReLU activation

4) Max pooling

(γ) *Εξηγήστε τη λειτουργία και τον ρόλο των convolutions, batch normalization, ReLU και Max pooling. Παραπέμπουμε στις αναφορές [16], [17], [18].*

(δ) *Πραγματοποιήστε τη διαδικασία overfit_batch του βήματος 5β για να βεβαιωθείτε ότι το δίκτυο μπορεί να εκπαιδευτεί.*

(ε) *Χρησιμοποιήστε αυτή την αρχιτεκτονική για την αναγνώριση μουσικού είδους με φασματογραφήματα και συγκρίνετε με το μοντέλο 5α.*

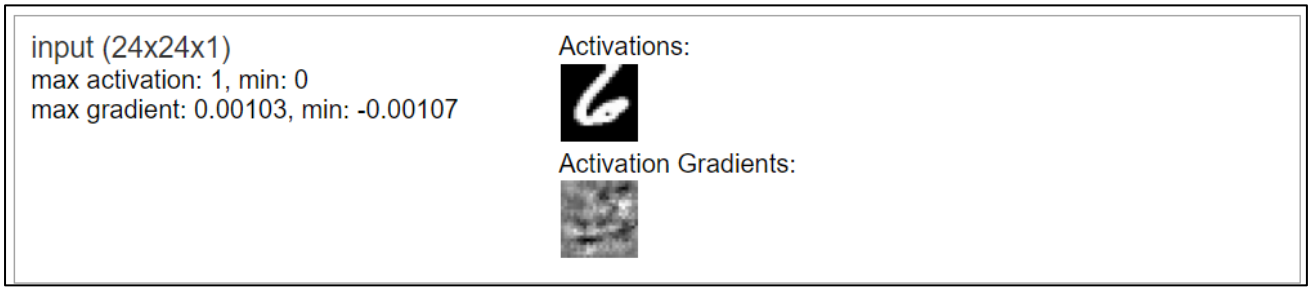
Υπόδειξη*: Εκτελέστε το μοντέλο σε διαφορετικό kernel για να αποφύγετε προβλήματα μνήμης.*

Υπόδειξη*: Ισχύουν όλες οι υποδείξεις του Βήματος 5*

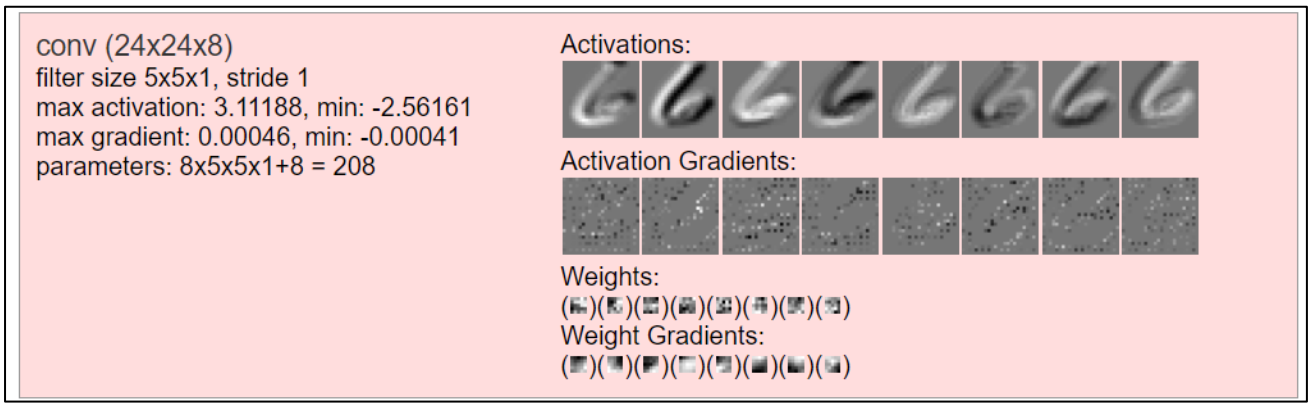
Υπόδειξη*: Μην σπαταλήσετε πολύ χρόνο στη ρύθμιση των υπερπαραμέτρων (hyperparameters) του δικτύου. Απλά δείτε κάποιες έτοιμες online υλοποιήσεις από CNNs και βάλτε κάποιες "λογικές" τιμές (πχ kernel size ~ 3 ή 5) κτλ. Αν το δίκτυο σας δε λειτουργεί όπως θα έπρεπε, είναι πιο πιθανό να οφείλεται σε κάποιο λάθος (bug) στον κώδικα από την κακή επιλογή παραμέτρων, ειδικά αν δεν αποκλίνουν πολύ τις προεπιλεγμένες (default) τιμές.*

(α) Σε αυτό το βήμα, εκπαιδεύτηκε ένα CNN πάνω στο MNIST digit Dataset. Παρατηρείται πως τα βάρη των φίλτρων και τα activations στο 1^{ο} συνελικτικό layer που επιδρούν απευθείας πάνω στα pixel των εικόνων εξάγουν χαρακτηριστικά όπως ακμές και γωνίες. Επειδή χρησιμοποιούνται 8 φίλτρα, εξάγονται 8 πίνακες χαρακτηριστικών. Με χρήση του δοσμένου κώδικα, προκύπτουν τα ακόλουθα:

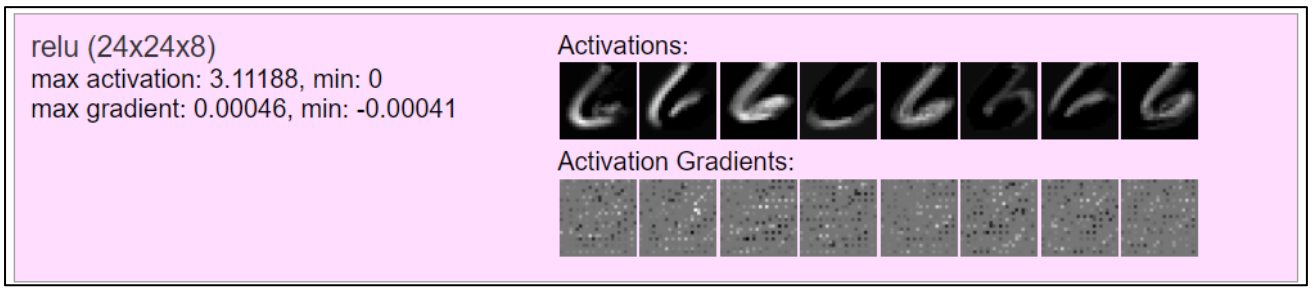
Input:



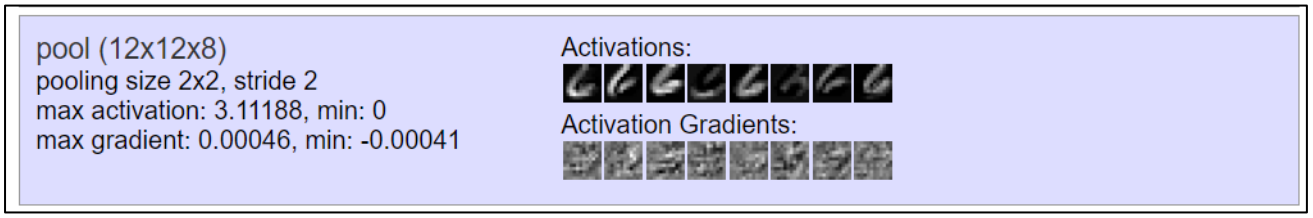
Conv:



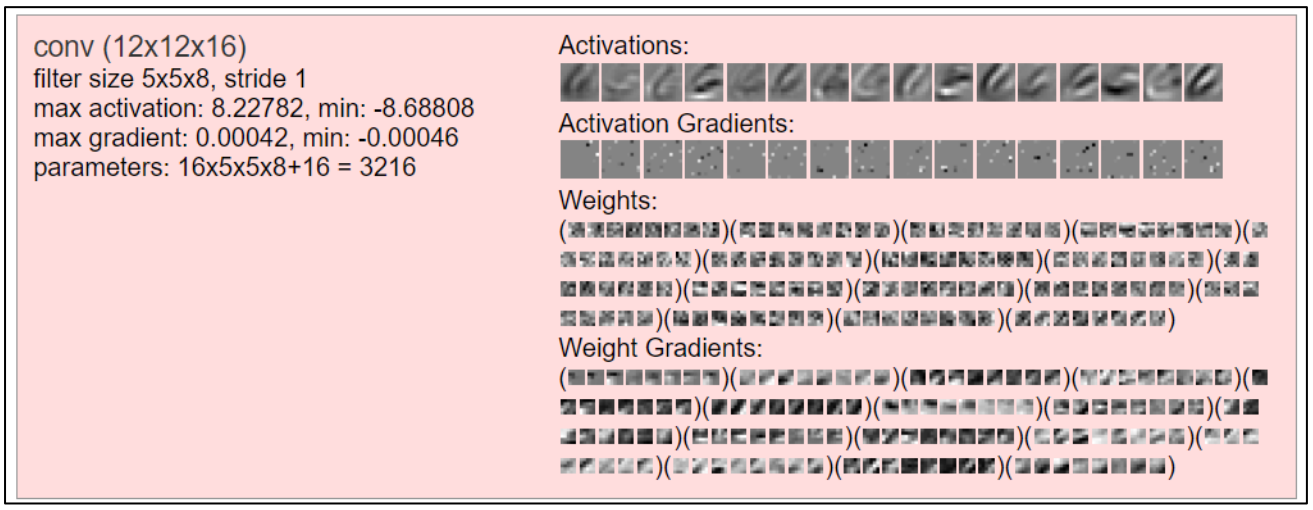
ReLU:



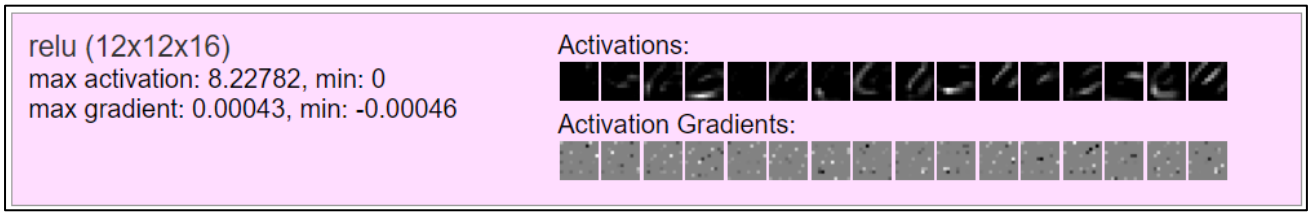
Pool:



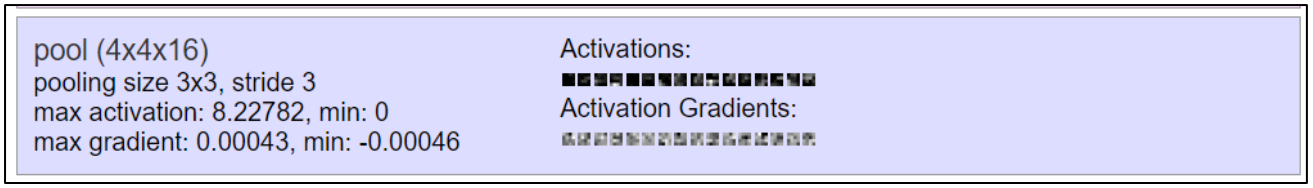
Conv:



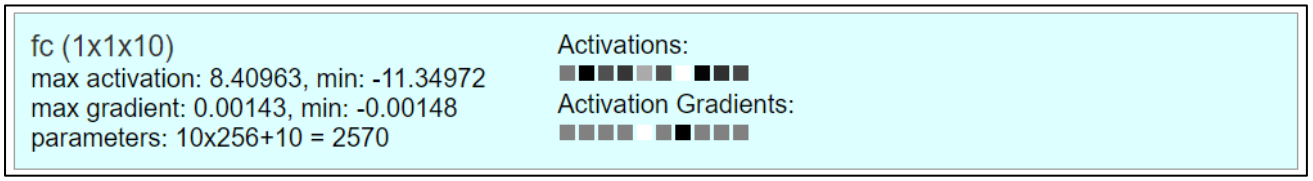
ReLU:



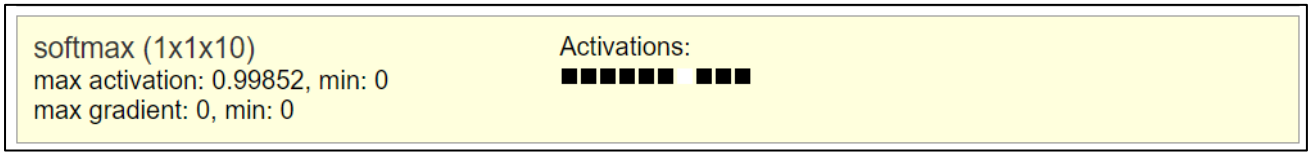
Pool:

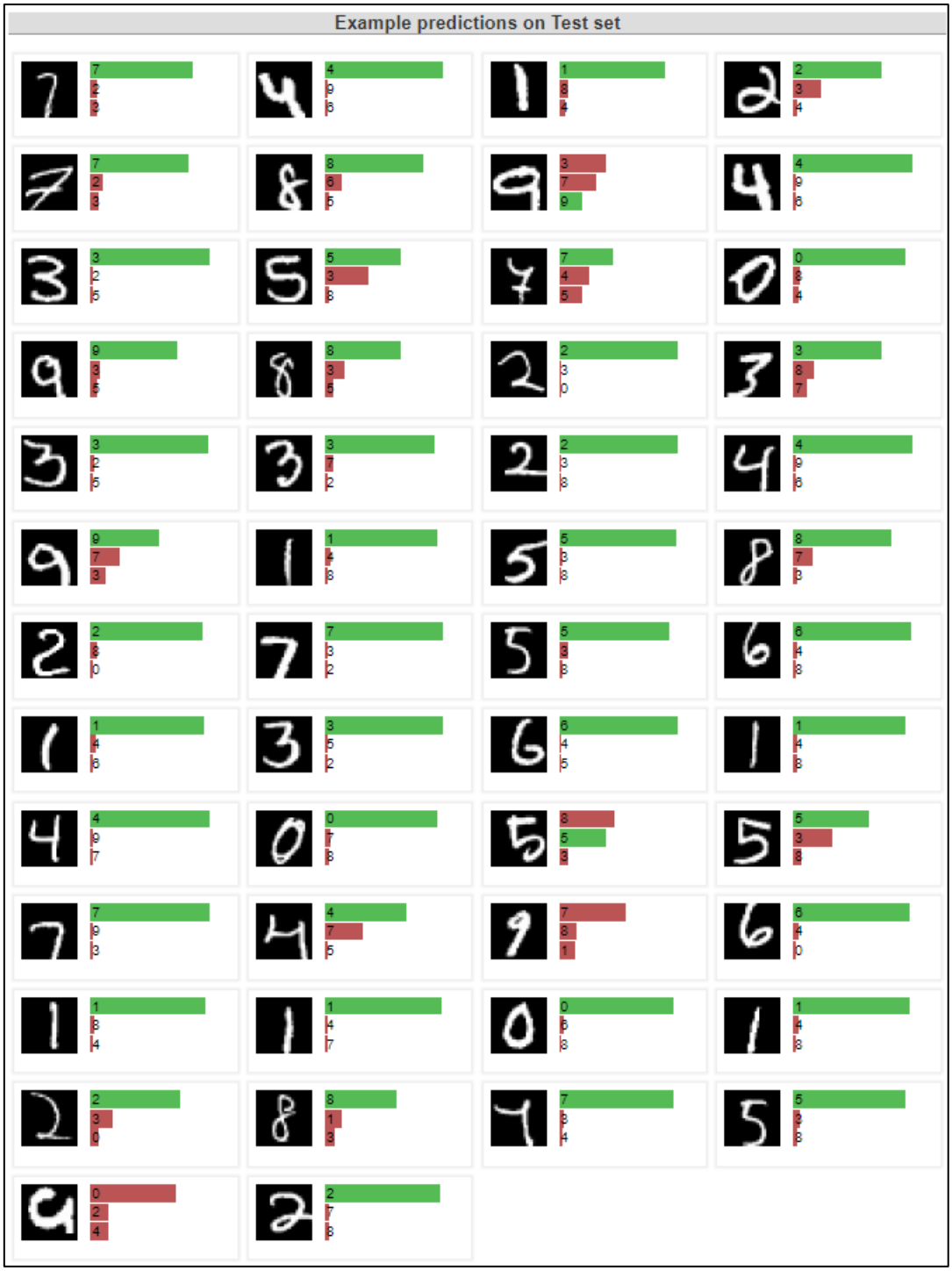


Fc:



Softmax:




Example predictions on Test set

Το CNN αποτελείται από δύο επίπεδα που εκτελούν τις λειτουργίες:

- 2D Convolution για feature extraction στις εικόνες εισόδου μέσω φίλτρων και αναγνώριση πιο υψηλού επιπέδου σχήματα ενώ κατά την εμβάθυνση του νευρωνικού, τα στοιχεία γίνονται περισσότερο δυσδιάκριτα.
- ReLU Activation για μηδενισμό των pixel με αρνητική τιμή ή απόρριψη των pixel με χαμηλές τιμές.
- Max Pooling για downsampling, μείωση της πολυπλοκότητας και σύνοψη της εισόδου μια γειτονιά 2×2 ώστε να δώσει ως έξοδο έναν πίνακα μισού μεγέθους από τον αρχικό.
- Στο πλήρως συνδεδεμένο επίπεδο, τα τελευταία στοιχεία προβάλλονται στον αριθμό των τάξεων.
- Στο Linear layer με softmax υπολογίζει τις posteriori πιθανότητες που οδηγούν στο classification των samples, δηλαδή την προβολή στις κλάσεις που ορίζονται (εδώ ψηφία 0-9).

(β) Με βάση τα χαρακτηριστικά που αναφέρονται παραπάνω ορίζεται το 2D CNN, το οποίο επιστρέφει τα ακόλουθα:

```
CNN((cnn_layers): Sequential(
(0):Conv2d(1,128,kernel_size=(5,5),stride=(1 1),padding=(2,2))
(1):BatchNorm2d(128,eps=1e-05,momentum=0.1,affine=True,track_running_stats=True)
(2):ReLU(inplace=True)
(3):MaxPool2d(kernel_size=2,stride=2,padding=0,dilation=1,ceil_mode=False)
(4):Conv2d(128,64,kernel_size=(5,5),stride=(1,1),padding=(2,2))
(5):BatchNorm2d(64,eps=1e-05,momentum=0.1,affine=True,track_running_stats=True)
(6):ReLU(inplace=True)
(7):MaxPool2d(kernel_size=2,stride=2,padding=0,dilation=1,ceil_mode=False)
(8):Conv2d(64,32,kernel_size=(5,5),stride=(1,1),padding=(2,2))
(9):BatchNorm2d(32,eps=1e-05,momentum=0.1,affine=True,track_running_stats=True)
(10):ReLU(inplace=True)
(11):MaxPool2d(kernel_size=2,stride=2,padding=0,dilation=1,ceil_mode=False)
(12):Conv2d(32,8,kernel_size=(5,5),stride=(1,1),padding=(2,2))
(13):BatchNorm2d(8,eps=1e-05,momentum=0.1,affine=True,track_running_stats=True)
(14):ReLU(inplace=True)
(15):MaxPool2d(kernel_size=2,stride=2,padding=0,dilation=1,ceil_mode=False))
(fc):Sequential((0): Linear(in_features=512,out_features=10,bias=True)))
```

(γ) Συνοπτικά, η λειτουργία και ο ρόλος:

- 2D Convolutions: εκτελούν γραμμική συνέλιξη της εικόνας εισόδου με έναν ειδικά επιλεγμένο πυρήνα που φιλτράρει όλη την εικόνα.
- Batch normalization: εκτελούν κανονικοποίηση στην μέση τιμή και στην διασπορά των δειγμάτων του κάθε batch ώστε να λάβουν τις τιμές 0 και 1 και συμβάλλει στην καλύτερη ταχύτητα, ευστάθεια και απόδοση των δικτύων.
- ReLU: μηδενίζει όλα τα pixel με αρνητική τιμή αφού αποτελεί μια activation function
- Max pooling: είναι η διαδικασία κατά την οποία σε κάθε pool επιλέγεται μόνο το pixel με την μεγαλύτερη τιμή ώστε να περάσει στο υπόλοιπο δίκτυο και έτσι μειώνεται η διάσταση που έχει αυξηθεί από τις προηγούμενες διαδικασίες.

Στην εκπαίδευση ενός CNN πρέπει να λαμβάνεται υπόψη μεταξύ άλλων και η υπολογιστική πολυπλοκότητα. Για παράδειγμα, η εκπαίδευση ενός CNN με εικόνες 8×8 και fully-connected αρχιτεκτονική είναι αρκετά straightforward και είναι υπολογιστικά εφικτό να προκύψουν χαρακτηριστικά για όλη την εικόνα. Όμως, σε μεγαλύτερες εικόνες, πχ pixels, μια τέτοια εκπαίδευση καθίσταται πολύ δύσκολη αφού για 10^4 εισόδους και για εκμάθηση 100 χαρακτηριστικών, προκύπτουν 10^6 parameters.

Οι feedforward & backpropagation algorithms είναι 10^2 φορές πιο αργοί σε σχέση με τις εικόνες.

Μια λύση για αυτό τον υπολογισμό είναι τα CNNs στα οποία τα χαρακτηριστικά ενός batch εικόνας είναι τοπικά εντοπισμένα. Επιπλέον, τα CNNs είναι ανεξάρτητα ως προς τις μετατοπίσεις της εικόνας σε διάφορες διευθύνσεις, έχουν μικρό αριθμό παραμέτρων και μεγάλο βήμα μεταξύ των batches για το pooling. Όλα αυτά συνδράμουν στην ταχύτερη εκτέλεση του αλγορίθμου.

Γενικότερα, κατά την εκπαίδευση των NN ένα σύνηθες ζήτημα καθίσταται η διαρκής μεταβολή της εισόδου των κρυφών επιπέδων. Αυτή η μεταβολή είναι γνωστή ως internal covariate shift και αποτελεί βασικό λόγο για την μείωση του ρυθμού μάθησης ώστε να συνεχίσει να μειώνεται. Ο τρόπος που αντιμετωπίζεται αυτό το φαινόμενο είναι με το batch normalization και η υλοποίησή του λαμβάνει υπόψη την μέτρηση των κατανομών όλων των ενεργοποιήσεων για κάθε batch. Έτσι, κατά το εμπρόσθιο πέρασμα γίνεται κανονικοποίηση με βάση τα στατιστικά του εκάστοτε batch για το κάθε layer ώστε ο μέσος όρος να είναι 0 και η απόκλιση ίση με 1.

Τα pooling layers στα CNNs συνοψίζουν τις εξόδους γειτονικών γκρουπ νευρώνων εντός ενός batch με μια αντιπροσωπευτική τιμή, ενώ συνήθως τα γειτονικά παράθυρα δεν επικαλύπτονται.

Τέλος, η ReLU είναι μια activation function που αντιστοιχίζει την είσοδο με μια τιμή στο [0, x], x>0. Χρησιμοποιείται ευρέως διότι δεν αντιμετωπίζει το πρόβλημα vanishing gradients και είναι υπολογιστικά πολύ αποδοτική αφού είναι εύκολα υλοποιήσιμη. Ωστόσο, δεν είναι zero centered και για αρνητικές τιμές εισόδου δεν αντιμετωπίζει πρόβλημα vanishing gradients.

(δ) – **(ε)** Για το classification σε μουσικό είδος με βάση τα spectrograms χρησιμοποιώντας το παραπάνω μοντέλο που περιγράφηκε, προκύπτουν τα ακόλουθα:

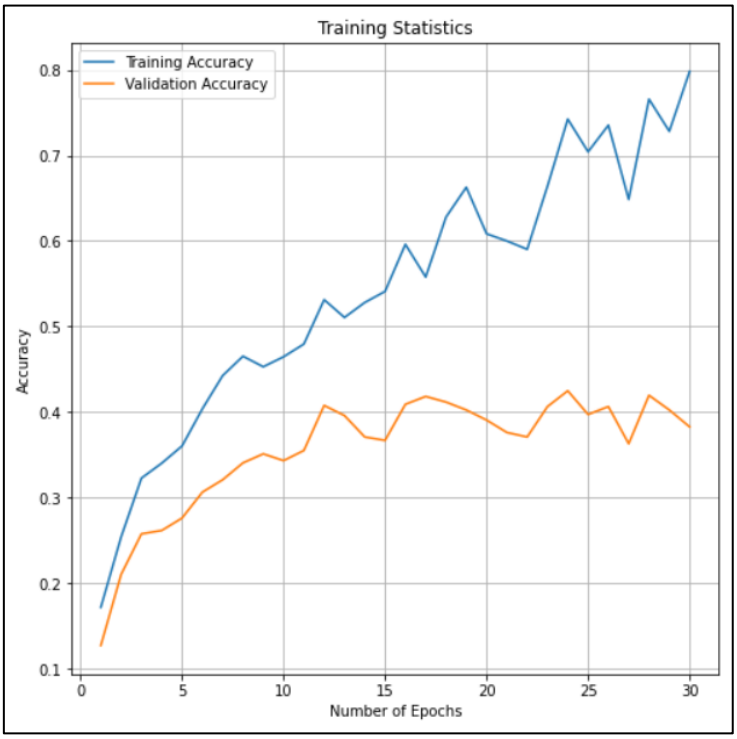
`Validation Accuracy after EPOCH:`

2	4	6	8	10	12	14	16
.2105	.2618	.3066	.3408	.3434	.4079	.3711	.4092

18	20	22	24	26	28	30
0.4118	0.3908	0.3711	.4250	.4066	.4197	.3829

Οπότε, το καλύτερο accuracy είναι της τάξης 42 %.

Μάλιστα, οπτικοποιώντας το accuracy τόσο στο train set όσο και στο validation set προκύπτει το εξής:



Σχολιασμός:

Το μοντέλο πιθανόν να έχει κάνει overfit πάνω στο train set. Ωστόσο, φαίνεται πως το CNN έχει καλύτερη απόδοση από το LSTM.

Βήμα 8 (Εκτίμηση συναισθήματος - συμπεριφοράς με παλινδρόμηση)

Σε αυτό το βήμα θα χρησιμοποιήσετε το multitask dataset

(“`../input/data/data/multitask_dataset/train_labels.txt`”).

Εδώ σας δίνονται τα φασματογραφήματα, καθώς και επισημειώσεις σε 3 άξονες που αφορούν το συναίσθημα του τραγουδιού. Οι επισημειώσεις είναι πραγματικοί αριθμοί μεταξύ 0 και 1:

- *Valence (πόσο θετικό ή αρνητικό είναι το συναίσθημα), όπου αρνητικό κοντά στο 0, θετικό κοντά στο 1.*

- *Energy (πόσο ισχυρό είναι το συναίσθημα), όπου ασθενές κοντά στο 0, ισχυρό κοντά στο 1.*

- *Danceability (πόσο χορευτικό είναι το τραγούδι), όπου μη χορευτικό κοντά στο 0, χορευτικό κοντά στο 1.*

***(α)** Προσαρμόστε το καλύτερο μοντέλο του Βήματος 5 και το μοντέλο του Βήματος 7 για παλινδρόμηση (regression) αλλάζοντας τη συνάρτηση κόστους.*

***(β)** Εκπαιδεύστε τα μοντέλα του 8α για την εκτίμηση του valence.*

***(γ)** Επαναλάβετε για την εκτίμηση του energy.*

***(δ)** Επαναλάβετε για την εκτίμηση του danceability.*

***(ε)** Η τελική μετρική είναι το μέσο Spearman correlation ανάμεσα στις πραγματικές (ground truth) τιμές και στις προβλεπόμενες τιμές για όλους τους άξονες.*

***Υπόδειξη**: Προσοχή. Σε αυτό το σύνολο δεδομένων δε σας παρέχονται οι επισημειώσεις για το test set, οπότε η εκτίμηση του πόσο καλά γενικεύει το μοντέλο θα πρέπει να γίνει παίρνοντας ένα υποσύνολο από τα δεδομένα που σας δίνονται.*

Σκοπός αυτού του βήματος είναι η προσαρμογή των καλύτερων μοντέλων από τα προηγούμενα Βήματα 5 και 7 για εκτέλεση παλινδρόμησης (regression) για 3 χαρακτηριστικά των δεδομένων εισόδου. Αυτό επιτυγχάνεται αλλάζοντας τις εξόδους του τελευταίου επιπέδου και προσθέτοντας μια sigmoid activation function. Δίνεται, ακόμη, ότι οι επισημειώσεις (labels) είναι πραγματικοί αριθμοί μεταξύ 0 και 1 και είναι τα valence, energy και danceability.

Τα αποτελέσματα είναι τα ακόλουθα:

```
CNN((cnn_layers):Sequential(
(0):Conv2d(1,128,kernel_size=(5,5),stride=(1,1),padding=(2,2))
(1):BatchNorm2d(128,eps=1e-05,momentum=0.1,affine=True,track_running_stats=True)
(2):ReLU(inplace=True)
(3):MaxPool2d(kernel_size=2,stride=2,padding=0,dilation=1,ceil_mode=False)
(4):Conv2d(128,64,kernel_size=(5,5),stride=(1,1),padding=(2,2))
(5):BatchNorm2d(64,eps=1e-05,momentum=0.1,affine=True,track_running_stats=True)
(6):ReLU(inplace=True)
(7):MaxPool2d(kernel_size=2,stride=2,padding=0,dilation=1,ceil_mode=False)
(8):Conv2d(64,32,kernel_size=(5,5),stride=(1,1),padding=(2,2))
(9):BatchNorm2d(32,eps=1e-05,momentum=0.1,affine=True,track_running_stats=True)
(10):ReLU(inplace=True)
(11):MaxPool2d(kernel_size=2,stride=2,padding=0,dilation=1,ceil_mode=False)
(12):Conv2d(32,8,kernel_size=(5,5),stride=(1,1),padding=(2,2))
(13):BatchNorm2d(8,eps=1e-05,momentum=0.1,affine=True,track_running_stats=True)
(14):ReLU(inplace=True)
(15):MaxPool2d(kernel_size=2,stride=2,padding=0,dilation=1,ceil_mode=False))
(fc):Sequential((0):Linear(in_features=384,out_features=1,bias=True))
(activation): Sigmoid())
```

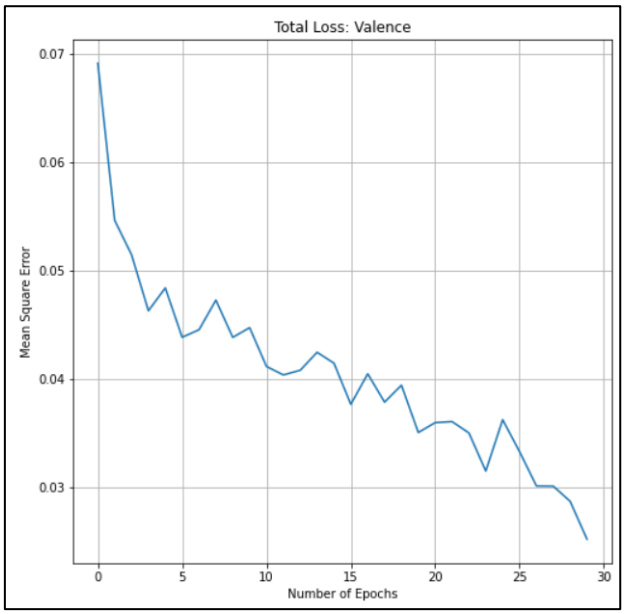
Για την προσαρμογή απαιτούνται επιπλέον οι εξής αλλαγές:

- Αλλαγή του loss function από Cross Entropy σε MSE
- Υλοποίηση της Spearman Correlation ως τελική μετρική
- Αλλαγή του SpectrogramDataset προσθέτοντας άλλη μια παράμετρο για την επιλογή του χαρακτηριστικού που θα περιλαμβάνει το Dataset όπου θα περιγράφει τα valence, energy, danceability

Τελικά, από την εκπαίδευση 3 διαφορετικών μοντέλων για 3 διαφορετικά χαρακτηριστικά, προκύπτουν τα ακόλουθα:

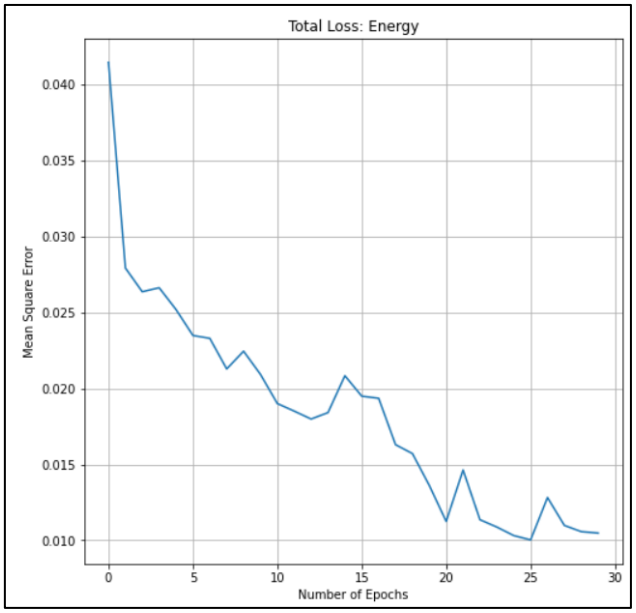
Valence: Spearman Correlation Coefficient after EPOCH:

1	2	4	5	8	9	17	20
0.1822	0.1953	0.2121	0.3331	0.4276	0.4478	0.4976	0.5093



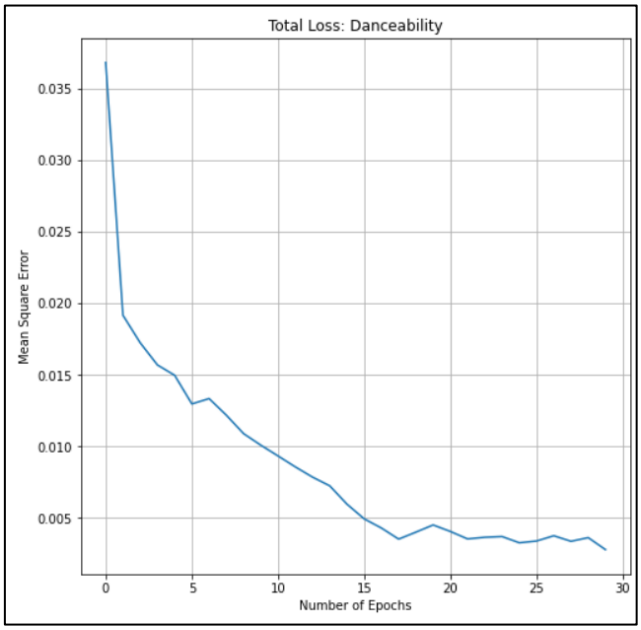
Energy: Spearman Correlation Coefficient after EPOCH:

1	3	6
0.7651	0.7791	0.7902



Danceability: Spearman Correlation Coefficient after EPOCH:

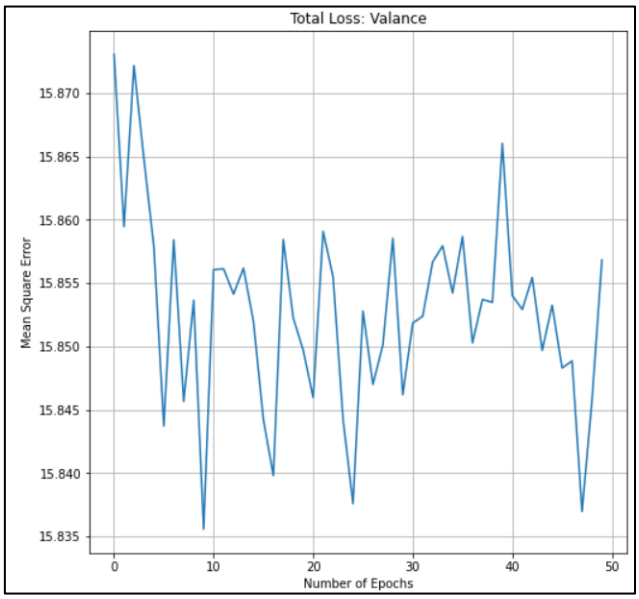
1	3	4	5	7	11	15
0.4582	0.5035	0.5296	0.5764	0.5901	0.5910	0.5925



Παρομοίως, από την εκπαίδευση 3 διαφορετικών LSTM προκύπτουν τα ακόλουθα:

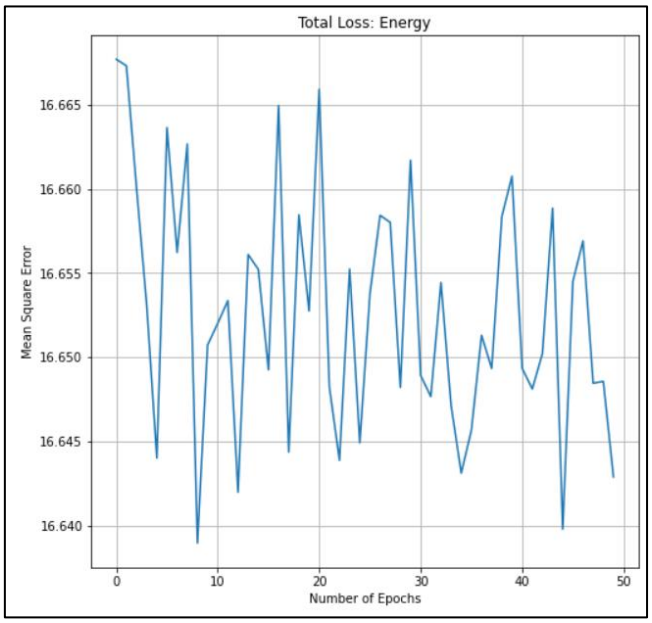
Valence: Spearman Correlation Coefficient after EPOCH:

1	2	5
0.1961	0.2268	0.2812



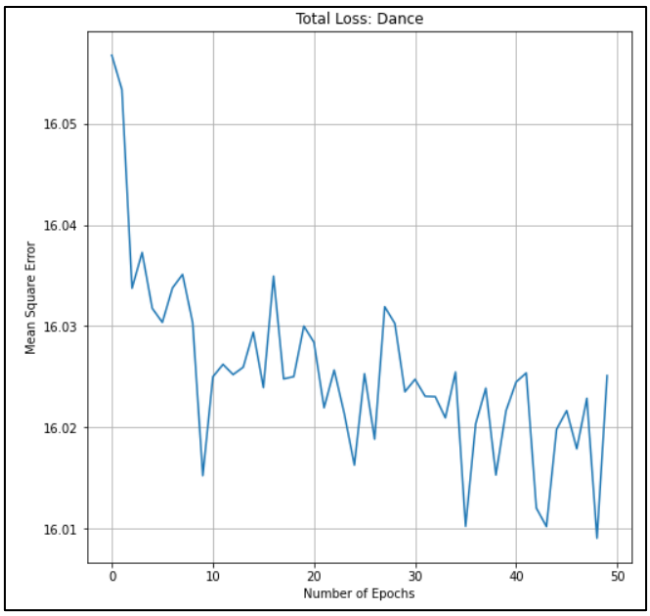
Energy: Spearman Correlation Coefficient after EPOCH:

1	2	4	9
0.4669	0.5128	0.5388	0.5402



Danceability: Spearman Correlation Coefficient after EPOCH:

1	2	3	6	7	8	10	11
0.0701	0.0723	0.1040	0.1069	0.1101	0.1174	0.1295	0.1704
18	**19**	**21**	**22**	**24**	**34**	**39**	**41**
0.1864	0.1876	0.2054	0.2064	0.2182	0.2189	0.2264	0.2327



Βήμα 9 (Μεταφορά γνώσης (Transfer Learning))

Ένας τρόπος για τη βελτίωση των βαθιών νευρωνικών όταν έχουμε λίγα διαθέσιμα δεδομένα είναι η μεταφορά της γνώσης από ένα άλλο μοντέλο, εκπαιδευμένο σε ένα μεγαλύτερο dataset. Για αυτό το λόγο

***(α)** Δείτε τα links [3], [4], [5]. Περιγράψτε με 2 προτάσεις τα βασικά συμπεράσματα του [5].*

***(β)** Επιλέξτε ένα μοντέλο από τα βήματα 5, 7. Εξηγήστε γιατί επιλέξατε αυτό το μοντέλο.*

***(γ)** Εκπαιδεύστε αυτό το μοντέλο στο `fma_genre_spectrograms dataset` και αποθηκεύστε τα βάρη του δικτύου στην εποχή που έχει την καλύτερη επίδοση (checkpoint). Εξηγείστε ποια μετρική και σε ποιο σετ επιλέγετε για να κρίνετε πιο μοντέλο έχει την καλύτερη επίδοση. Γιατί;*

***(δ)** Αρχικοποιήστε ένα μοντέλο με αυτά τα βάρη για το πρόβλημα του ερωτήματος 8 και εκπαιδεύστε το για λίγες εποχές (fine tuning) σε έναν από τους συναισθηματικούς άξονες της επιλογής σας (πχ valence).*

***(ε)** Συγκρίνετε τα αποτελέσματα με αυτά από το Βήμα 8.*

(α) Το paper μελετά το κατά πόσο έχει σημασία να χρησιμοποιηθούν βάρη από NN τα οποία είναι εκπαιδευμένα σε κάποιο task για την επίλυση ενός άλλου task και τις δυσκολίες που υπάρχουν.

Τα συμπεράσματα στα οποία καταλήγει είναι τα εξής:

- Τα layers ανάλογα με το βάθος στο οποίο βρίσκονται χωρίζονται σε γενικά, για την εξαγωγή γενικών χαρακτηριστικών, όπου είναι θεμιτή η μεταφορά τους καθώς λειτουργούν ως feature extractors, και σε ειδικά, για την εξαγωγή πληροφοριών οι οποίες είναι task-specific, όπου δεν προσφέρουν κάτι στο δίκτυο.
- Επιπλέον, παρατηρείται μείωση στην απόδοση κατά την μεταφορά γνώσης που οφείλεται στο φαινόμενο fragile co-adapted features (δηλ layers των οποίων οι τιμές διαμορφώθηκαν κατά την εκπαίδευση με βάση την αλληλεπίδραση με άλλα layers του δικτύου). Έτσι, με την αφαίρεση και μεταφορά κάποιων από αυτά τα layers σε άλλο δίκτυο, δεν προκύπτει το ίδιο καλό αποτέλεσμα εξαιτίας της απουσίας αυτής της αλληλεπίδρασης.
- Ακόμη, παρατηρείται ότι η μεταφορά βαρών από ένα άλλο δίκτυο και η μετεκπαίδευσή τους (fine tuning) βοηθάει στην γενίκευση, ανεξαρτήτως του μεγέθους του dataset διότι αυτά τα βάρη συνεχίζουν και μετά από αρκετές επαναλήψεις να διατηρούν κομμάτι της αρχικής πληροφορίας.
- Τέλος, όσο πιο ανόμοια είναι τα tasks, τόσο περισσότερο μειώνεται η επίδραση της μεταφοράς γνώσης.

(β) – **(ε)** Για την βελτίωση των προηγούμενων δικτύων έγινε μια προσπάθεια μεταφοράς γνώσης από ένα άλλο μοντέλο το οποίο είχε εκπαιδευτεί σε μεγαλύτερο Dataset. Το μοντέλο που επιλέχθηκε τελικά είναι το CNN του Βήματος 7 το οποίο με classification πετυχαίνει την καλύτερη επίδοση (αφού πρώτα γίνει save). Η μόνη τροποποίηση γίνεται στο τελευταίο layer ώστε να εκτελεί regression.

Τα αποτελέσματα είναι τα εξής:

```
CNN((cnn_layers):Sequential(
(0):Conv2d(1,128,kernel_size=(5,5),stride=(1,1),padding=(2,2))
(1):BatchNorm2d(128,eps=1e-05,momentum=0.1,affine=True,track_running_stats=True)
(2):ReLU(inplace=True)
(3):MaxPool2d(kernel_size=2,stride=2,padding=0,dilation=1,ceil_mode=False)
(4):Conv2d(128,64,kernel_size=(5,5),stride=(1,1),padding=(2,2))
(5):BatchNorm2d(64,eps=1e-05,momentum=0.1,affine=True,track_running_stats=True)
(6):ReLU(inplace=True)
(7):MaxPool2d(kernel_size=2,stride=2 padding=0 dilation=1 ceil_mode=False)
(8):Conv2d(64 32 kernel_size=(5,5),stride=(1,1),padding=(2,2))
(9):BatchNorm2d(32,eps=1e-05,momentum=0.1,affine=True,track_running_stats=True)
(10):ReLU(inplace=True)
(11):MaxPool2d(kernel_size=2,stride=2,padding=0,dilation=1,ceil_mode=False)
(12):Conv2d(32,8,kernel_size=(5,5),stride=(1,1),padding=(2,2))
(13):BatchNorm2d(8,eps=1e-05,momentum=0.1,affine=True,track_running_stats=True)
(14):ReLU(inplace=True)
(15):MaxPool2d(kernel_size=2,stride=2,padding=0,dilation=1,ceil_mode=False))
(fc):Linear(in_features=384,out_features=1,bias=True) (activation):Sigmoid())
```

Τα αποτελέσματα που προκύπτουν για το valence είναι τα ακόλουθα:

```
Spearman Correlation Coefficient after each EPOCH:
```

2	3	4	5	6	8	9
0.0736	0.1492	0.1943	0.2304	0.3140	0.3285	0.4226

Σχολιασμός:

Η μεταφορά γνώσης χρησιμοποιείται όταν δεν υπάρχουν διαθέσιμα αρκετά δεδομένα για εκπαίδευση. Εδώ, δεν παρατηρείται βελτίωση στην επίδοση συγκριτικά με το Βήμα 8 καθώς εκείνο το μοντέλο με αυτό έχουν περίπου ίδιο πλήθος δειγμάτων. Επιπλέον, σύμφωνα με το paper, τα πρώτα layers ενός δικτύου δεν είναι συγκεκριμένα για κάποιο task ή dataset σε αντίθεση με τα τελευταία layers. Έτσι, κρίνεται απαραίτητο το tuning των τελευταίων layers του pre-trained model ώστε να πετύχει καλύτερα στο εκάστοτε task που εκτελεί.

Βήμα 10 (Εκπαίδευση σε πολλαπλά προβλήματα (Multitask Learning))

Στο Βήμα 8 εκπαιδεύσαμε ξεχωριστά ένα μοντέλο για κάθε συναισθηματική διάσταση. Ένας τρόπος για να εκπαιδεύσουμε πιο αποδοτικά μοντέλα όταν μας δίνονται πολλές επισημειώσεις είναι η χρήση multitask learning.

(α) *Δείτε τα links [3], [6], [7] και περιγράψτε με 2 προτάσεις τα βασικά συμπεράσματα του [7].*

(β) *Υλοποιήστε μια συνάρτηση κόστους (σαν nn.Module) η οποία θα λαμβάνει σαν είσοδο τα logits (outputs του μοντέλου) και τα targets (πραγματικές τιμές) για το valence, arousal και danceability και θα επιστρέφει το άθροισμα των επιμέρους losses για κάθε task. Μπορείτε να χρησιμοποιήσετε βάρη για να φέρετε τα επιμέρους κόστη στην ίδια τάξη μεγέθους.*

(γ) *Εκπαιδεύστε ένα μοντέλο στο multitask dataset χρησιμοποιώντας την παραπάνω συνάρτηση κόστους.*

(δ) *Συγκρίνετε τα αποτελέσματα με αυτά από το Βήμα 8.*

(α) Στο paper εξετάζεται η δυνατότητα ορισμένων μοντέλων να αποδίδουν καλύτερα σε διαφορετικά tasks όταν η εκπαίδευση για τα εν λόγω tasks συμβαίνει ταυτόχρονα. Συγκεκριμένα, μελετά όχι μόνο με διαφορετικά tasks από το ίδιο domain αλλά και με tasks από διαφορετικά domains, όπως Image Captioning, Parsing, Machine Translation etc. Αυτό το καταφέρνει συνδυάζοντας δομικές μονάδες που είναι ευρέως χρησιμοποιούμενες στα επιμέρους tasks (convolutional layers, attention mechanisms etc) τα οποία είναι αναγκαία για το task στο οποίο απευθύνονται, ενώ δε μειώνουν την απόδοση στα υπόλοιπα tasks. Επιπλέον, χρησιμοποιεί διαφορετικά δίκτυα για το κάθε modality για να φέρει κάθε είσοδο σε έναν κοινό τρόπο αναπαράστασης. Έτσι, εκπαιδεύοντας ένα μοναδικό μοντέλο για όλα τα tasks υπάρχει δυνατότητα να αξιοποιηθούν εσωτερικές αναπαραστάσεις που πιθανώς να είναι κοινές σε κάποια tasks, σε αντίθεση με το να γινόταν εκπαίδευση ξεχωριστά για κάθε task, και τελικά να αυξάνεται η απόδοση στα επιμέρους tasks αλλά και να παρατηρείται βελτίωση στην απόδοση για τα tasks τα οποία έχουν τα λιγότερα δεδομένα.

(β) – **(δ)** Έχουν υλοποιηθεί σε κώδικα.

```
Data shape

torch.Size([32, 104, 128])

Labels

tensor([7756., 6033., 6526., 6241., 8578., 5699., 9069., 9860., 3625., 4162.,
        6941., 6895., 9627., 1422., 1475., 2811., 8973., 7473., 1430., 9859.,
        9220., 3738., 4661., 2322., 2395., 1789., 2935., 4235., 5722., 3692.,
        1052., 4389.], dtype=torch.float64)

Lengths

tensor([ 52,  56,  58,  56,  48,  62,  43,  56,  64,  14,  79,  63,  44,  55,
         42,  73,  50,  51,  49,  62,  72,  64,  48,  42, 102,  87,  50,  61,
         62,  65,  60,  76])
```

Και επιπλέον, προκύπτει:

```
CNN((cnn_layers):Sequential(
(0):Conv2d(1,128,kernel_size=(5,5),stride=(1,1),padding=(2,2))
(1):BatchNorm2d(128,eps=1e-05,momentum=0.1,affine=True,track_running_stats=True)
(2):ReLU(inplace=True)
(3):MaxPool2d(kernel_size=2,stride=2,padding=0,dilation=1,ceil_mode=False)
(4):Conv2d(128,64,kernel_size=(5,5),stride=(1,1), padding=(2,2))
(5):BatchNorm2d(64,eps=1e-05,momentum=0.1,affine=True,track_running_stats=True)
(6):ReLU(inplace=True)
(7):MaxPool2d(kernel_size=2,stride=2,padding=0,dilation=1,ceil_mode=False)
(8):Conv2d(64,32,kernel_size=(5,5),stride=(1,1),padding=(2,2))
(9):BatchNorm2d(32,eps=1e-05,momentum=0.1,affine=True,track_running_stats=True)
(10):ReLU(inplace=True)
(11):MaxPool2d(kernel_size=2,stride=2,padding=0,dilation=1,ceil_mode=False)
(12):Conv2d(32,8,kernel_size=(5,5),stride=(1,1),padding=(2,2))
(13):BatchNorm2d(8,eps=1e-05,momentum=0.1,affine=True,track_running_stats=True)
(14):ReLU(inplace=True)
(15):MaxPool2d(kernel_size=2,stride=2,padding=0,dilation=1,ceil_mode=False))
(fc):Sequential((0):Linear(in_features=384,out_features=3,bias=True))
(activation):Sigmoid())
```

Βήμα 11 ((Προαιρετικό): Υποβολή στο Kaggle)

(α) *Επιλέξτε το καλύτερο μοντέλο σας για το multitask dataset και πραγματοποιήστε προβλέψεις για το valence, energy και danceability στα test δεδομένα.*

(β) *Διαμορφώστε ένα αρχείο solution.txt στη μορφή*

Id.fused.full.npy.gz,valence,energy,danceability

123212738.fused.full.npy.gz,0.153,0.961,0.013

(γ) *Υποβάλετε το solution.txt στο διαγωνισμό στο Kaggle και δείτε τα αποτελέσματα στο leaderboard.*

(δ) *Σχολιάστε πόσο κοντά είναι τα αποτελέσματά σας με αυτά που περιμένατε.*

Σχολιασμός:

Αυτό το βήμα μπορεί να υλοποιηθεί με κατάλληλη επιλογή CNN από το Βήμα 8. Ύστερα, με την δημιουργία ενός solution.txt όπως δίνεται παραπάνω και την υποβολή του στο διαγωνισμό μπορεί να δει κανείς το σκορ. Σε περίπτωση που το σκορ είναι χαμηλό πιθανώς να απαιτείται finetuning στις parameters.

Βιβλιογραφία - Αναφορές

[0]https://www.kaggle.com/c/multitask-affective-music-lab-2022/overview

[1]https://www.kaggle.com/c/multitask-affective-music-lab-2022/data

[2]https://www.kaggle.com/c/multitask-affective-music-lab-2022/kernels

[3]https://www.coursera.org/lecture/machine-learning-projects/transfer-learning-WNPap

[4]https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a

[5]http://papers.nips.cc/paper/5347-how-transferable-are-features-in-deep-neural-networks.pdf

[6]http://ruder.io/multi-task/

[7]https://arxiv.org/pdf/1706.05137.pdf

[8]https://github.com/slp-ntua/patrec-labs/tree/main/lab3

[9]https://towardsdatascience.com/multi-class-metrics-made-simple-part-ii-the-f1-score-ebe8b2c2ca1

[10]https://becominghuman.ai/how-to-evaluate-the-machine-learning-models-part-3-ff0dd3b76f9

[11]https://towardsdatascience.com/metrics-for-imbalanced-classification-41c71549bbb5

[12]https://twitter.com/karpathy/status/1013244313327681536

[13]http://karpathy.github.io/2019/04/25/recipe/

[14]https://www.reddit.com/r/MachineLearning/comments/5pidk2/d_is_overfitting_on_a_very_small_data_set_a/

[15]https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[16]https://colah.github.io/posts/2014-07-Understanding-Convolutions/

[17]https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

[18]https://blog.xrds.acm.org/2016/06/convolutional-neural-networks-cnns-illustrated-explanation/

[19]https://cs.stanford.edu/people/karpathy/convnetjs/

[20]https://www.kaggle.com/geoparslp/data-loading-tutorial

[21]https://pytorch.org/docs/stable/data.html#torch.utils.data.Subset

[22]https://stackoverflow.com/questions/47432168/taking-subsets-of-a-pytorch-dataset

[23]https://en.wikipedia.org/wiki/Spearman%27s_rank_correlation_coefficient

[24]https://developers.google.com/machine-learning/crash-course/classification/accuracy

[25]https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall

[26]Σημειώσεις / διαφάνειες μαθήματος & παλαιότερο υλικό