ΕΠ08 Αναγνώριση Προτύπων – Μηχανική Μάθηση

3η Εργασία

Τύπος εργασίας: ***Ατομική***

Ημερομηνία παράδοσης: ***Κυριακή 04/09/2022, 23:55 (Δεν θα δοθεί παράταση)***

Τρόπος παράδοσης: ***Αποκλειστικά μέσω του eclass***

Σύνολο βαθμών: **100** (Προσθετικά μέχρι 3 βαθμούς στον τελικό βαθμό του μαθήματος μόνο εφόσον ο βαθμός στη τελική εξέταση (Ιουνίου ή Σεπτεμβρίου) είναι μεγαλύτερος ή ίσος του 4 στα 10).

Στην παρούσα εργασία θα ασχοληθείτε με την πρόβλεψη μουσικού είδους από σήματα μουσικής με τη χρήση νευρωνικών δικτύων. Η εργασία είναι ατομική και έχει προαιρετικό χαρακτήρα, δίνοντας σας τη δυνατότητα να βελτιώσετε τον τελικό σας βαθμό μέχρι και 30%, εφόσον ο βαθμός στη τελική εξέταση είναι μεγαλύτερος ή ίσος του 4 στα 10. Καθώς η παρούσα εργασία είναι προαιρετική **δεν** θα υπάρχει υποστήριξη από πλευράς διδασκόντων ούτε στο χώρο συζητήσεων του eclass ούτε μέσω email.

Θα πρέπει να υποβάλετε **ένα μόνο αρχείο Notebook IPython (Jupiter notebook) μέσω του εργαλείου εργασίες του eclass**, ακολουθώντας την εξής σύβαση ονομασίας για το αρχείο σας: *Επώνυμο_ΑριθμόςΜητρώου.ipynb* Μπορείτε να χρησιμοποιήσετε κελιά επικεφαλίδας για να οργανώσετε περαιτέρω το έγγραφό σας. Το notebook που θα παραδώσετε θα πρέπει βεβαιωθείτε ότι ανοίγει και να εκτελείται στο Google Colab. Αν συμβουλευτείτε ή/και χρησιμοποιήσετε οποιοδήποτε υλικό ή/και κώδικα που είναι διαθέσιμος στο διαδίκτυο, πρέπει να αναφέρεται σωστά τη πηγή ή/και το σύνδεσμο στην ιστοσελίδα που αντλήσατε πληροφορίες. Σε κάθε περίπτωση, η αντιγραφή τμήματος ή του συνόλου της εργασίας δεν είναι αποδεκτή και στη περίπτωση που διαπιστωθεί αντιγραφή θα μηδενιστούν στο μάθημα όλα τα εμπλεκόμενα μέρη.

Πιο συγκεκριμένα, ο στόχος της εργασία είναι να ταξινομήσουμε 1 δευτερόλεπτο μουσικού σήματος στα εξής είδη: κλασσική μουσική, ποπ, ροκ, και μπλουζ. Για κάθε 1 δευτερόλεπτο σας παρέχονται δύο ειδών αναπαραστάσεις του ηχητικού σήματος: (i) MFCCs, και (ii) mel-spectograms.

Τα MFCCs είναι συντελεστές του φάσματος ισχύος μετασχηματισμένοι με βάση την κλίμακα mel, μία κλίμακα που είναι κοντά στον τρόπο που αντιλαμβάνεται ο άνθρωπος τα ηχητικά σήματα μέσω της ακοής. Στην δική μας περίπτωση χρησιμοποιούμε 13 συντελεστές οι οποίοι υπολογίζονται για κάθε 50 msec και επομένως για κάθε μουσικό κομμάτι του dataset προκύπτει μία ακολουθία από 20 feature vectors διάστασης 13. Για να αναπαραστήσουμε αυτή την πληροφορία μέσω ενός στατικού διανύσματος, το οποίο είναι ευκολότερο στην χρήση, υπολογίζουμε για κάθε έναν από τους 13 συντελεστές την μέση τιμή και την τυπική του απόκλιση από την ακολουθία των 20 χρονικών στιγμών. Καταλήγουμε λοιπόν με ένα διάνυσμα 26 χαρακτηριστικών για κάθε μουσικό κομμάτι του dataset.

Το φασματογράφημα (spectrogram), που είναι ο δεύτερος τρόπος αναπαράστασης που θα χρησιμοποιήσουμε, είναι μία δισδιάστατη αναπαράσταση που δείχνει την χρονική εξέλιξη του φάσματος συχνοτήτων. Εάν στο spectrogram εφαρμόσουμε την κλίμακα mel, παίρνουμε το mel-spectrogram ή melgram με το οποίο και θα δουλέψουμε στην παρούσα εργασία. Υπολογίζοντας και αντιστρέφοντας τους άξονες χρόνου και συχνότητας, προκύπτει για κάθε στοιχείο του συνόλου δεδομένων ένας πίνακας 21 (χρόνος) x 128 (συχνότητα).

Τα δεδομένα που θα χρησιμοποιήσετε βρίσκονται εδώ και είναι χωρισμένα στα σύνολα training (3200 δείγματα), validation (800 δείγματα) και test (1376 δείγματα) sets, τα οποία θα χρησιμοποιηθούν για εκπαίδευση, εύρεση υπερπαραμέτρων και αξιολόγηση της ικανότητας γενίκευσης αντίστοιχα.

Ακολουθήστε τις οδηγίες των παρακάτω ερωτημάτων και ετοιμάστε τις απαντήσεις σας τρέχοντας τον κώδικά σας στο Google Colab. Το framework που θα πρέπει να χρησιμοποιηθεί για τον προγραμματισμό των νευρωνικών είναι **υποχρεωτικά** το Pytorch.

[Ερώτημα 1: Feedforward Neural Network] (20 βαθμοί) 0.75

Βήμα 1: Φόρτωση δεδομένων (mfccs)

Ξεκινάμε φορτώνοντας τα mfcc δεδομένα για train, validation και test μέσω των αντίστοιχων numpy αρχείων X.npy και labels.npy. Στην συνέχεια μετασχηματίζουμε τα labels από strings (classical, blues etc) σε ακέραιους αριθμούς από 0 μέχρι 3, κρατώντας το αντίστοιχο mapping από τα ονόματα των κλάσεων στους ακεραίους. Τέλος φορτώνουμε τα δεδομένα μας σε 3 Pytorch dataloaders (ένα για κάθε σύνολο δεδομένων) με batch size 16, ώστε να μπορούν να χρησιμοποιηθούν στα μοντέλα μας. Δώστε επίσης το όρισμα shuffle=True στους train και validation dataloaders.

Βήμα 2: Ορισμός Νευρωνικού Δικτύου

Ορίστε μία κλάση πλήρως συνδεδεμένου Νευρωνικού Δικτύου (fully connected neural nework) το οποίο να αποτελείται από τα 4 επίπεδα με αριθμούς νευρώνων 26, 128, 32 και 4 αντίστοιχα, όπου 26 είναι η διάσταση της εισόδου και 4 ο αριθμός των κλάσεων που θα προβλεφθούν.

Βήμα 3: Ορισμός διαδικασίας εκπαίδευσης

Ορίστε μία συνάρτηση που θα είναι αρμόδια για την εκπαίδευση του δικτύου. Συγκρεκιμένα, δεδομένου ενός αριθμού εποχών, ενός optimizer, ενός dataloader, μιας συνάρτησης κόστους και ενός νευρωνικού θα περνάει κάθε batch από το νευρωνικό, θα υπολογίζει και θα τυπώνει το loss και θα ενημερώνει τα βάρη, ενώ θα τερματίζει επιστρέφοντας το νευρωνικό δίκτυο, όταν ο αριθμός των εποχών επιτευχθεί.

Βήμα 4: Ορισμός διαδικασίας αξιολόγησης

Ορίστε αντίστοιχα μία συνάρτηση αξιολόγησης, η οποία θα περνάει όλα τα batches ενός dataloader από το μοντέλο παίρνοντας τις προβλέψεις του και χωρίς να ενημερώνει τα βάρη.

Μέσω των προβλέψεων θα υπολογίζει και θα επιστρέφει (i) το loss, (ii) το f1 macro averaged, (iii) το accuracy, και (iv) confusion matrix.

Βήμα 5: Εκπαίδευση δικτύου

Εκπαιδεύστε το νευρωνικό δίκτυο στο training set χρησιμοποιώντας τα εξής:

- optimizer: stochastic gradient descent
- learning rate: 0.002
- loss function: cross-entropy loss
- αριθμός εποχών: 30

Στην συνέχεια χρησιμοποιήστε την συνάρτηση αξιολόγησης του προηγούμενου ερωτήματος για να υπολογίσετε τις επιδόσεις του εκπαιδευμένου μοντέλου στο test set. Τι επιδόσεις πετυχαίνετε;

Βήμα 6: Εκπαίδευση δικτύου με GPU

Επαναλάβετε το βήμα 5, αλλά αυτή την φορά να έχετε αρχικά μεταφέρει τα δεδομένα και το αρχικοποιημένο νευρωνικό σας δίκτυο στην GPU του colab. Βεβαιωθείτε ότι η εκπαίδευση τρέχει στην GPU και τυπώστε τις διαφορές στους χρόνους εκτέλεσης σε GPU και CPU. Βεβαιωθείτε ότι το colab session σας περιλαμβάνει χρήση GPU - η οποία είναι δωρεάν.

Βήμα 7: Επιλογή μοντέλου

Κατά την διάρκεια εκπαίδευσης (30 εποχές) προκύπτουν διαφορετικά στιγμιότυπα του νευρωνικού μας, δηλαδή μοντέλα που έχουν διαφορετικά βάρη. Κατά την διαδικασία βελτιστοποίησης, δεν γνωρίζουμε ποιο στιγμιότυπο του μοντέλου μας έχει την καλύτερη δυνατότητα γενίκευσης. Για τον λόγο αυτό θα χρησιμοποιήσουμε το validation set στο τέλος κάθε εποχής ώστε να αξιολογούμε τα στιγμιότυπα του μοντέλου. Αποθηκεύστε το μοντέλο που έχει την καλύτερη επίδοση στην μετρική f1 για το validation set και χρησιμοποιήστε το για να μετρήσετε την απόδοση στο test set. Σχολιάστε τα αποτελέσματα.

Για τα επόμενα βήματα της εργασίας θα πρέπει να εργάζεστε με τον ίδιο τρόπο χρησιμοποιώντας το validation set για να βρείτε το κατάλληλο στιγμιότυπο

[Ερώτημα 2: Convolutional Neural Network] (20 βαθμοί) .75

Στο ερώτημα αυτό θα χρησιμοποιήσουμε τα mel-spectrograms σαν εισόδους σε Συνελικτικά Νευρωνικά Δίκτυα με σκοπό την ταξινόμηση μουσικού είδους.

Βήμα 1: Φόρτωση δεδομένων (spectrograms)

Ακολουθήστε την διαδικασία του ερωτήματος 1.1, αλλά αυτή τη φορά για τα melgrams. Οπτικοποιείστε ένα τυχαίο melgram απο κάθε κλάση.

Βήμα 2: Ορισμός Νευρωνικού Δικτύου

Ορίστε ένα Συνελικτικό Νευρωνικό Δίκτυο το οποίο να αποτελείται από

- Ακολουθία τεσσάρων συνελικτικών επιπέδων, με kernel size 5, ώστε να επιτυγχάνεται η εξής ακολουθία καναλιών: 1, 16, 32, 64, 128
- Η έξοδος του τελευταίου συνελικτικού επιπέδου να εισέρχεται σε ένα πλήρως συνδεδεμένο νευρωνικό δίκτυο 5 επιπέδων με αριθμό νευρώνων: x (διάσταση εξόδου συνελικτικού δικτύου), 1024, 256, 32, out_dim

Βήμα 3: Εκπαίδευση δικτύου

Αλλάξτε όπου χρειάζεται, και τρέξτε, την διαδικασία εκπαίδευσης και αξιολόγησης ώστε να μπορεί να εκπαιδευτεί και το νέο νευρωνικό δίκτυο.

Τι παρατηρείτε; Μπορεί να εκπαιδευτεί το δίκτυο;

Βήμα 4: Pooling and padding

Ενσωματώστε στα συνελικτικά επίπεδα max pooling με kerne size 2, και padding μεγέθους 2. Σχολιάστε την χρησιμότητα των δύο αυτών στοιχείων. Τι επίδοση πετυχαίνετε;

Βήμα 5: Αλγόριθμοι βελτιστοποίησης

Υπάρχουν διαφορετικοί αλγόριθμοι βελτιστοποίησης ενός Νευρωνικού δικτύου. Δοκιμάστε ένα σύνολο από optimizers που αναφέρονται εδώ, και φτιάξτε ένα πινακάκι που στις στήλες θα περιέχει τους αλγόριθμους και στις γραμμές τιτς μετρικές accuracy και f1. Τι διαφορές παρατηρείτε στην επίδοση;

[Ερώτημα 3: Improving Performance] (30 βαθμοί) 1.125

Σε αυτό το ερώτημα θα προσπαθήσουμε να χρησιμοποιήσουμε τεχνικές και εργαλεία της Βαθιάς Μάθησης για να βελτιώσουμε την επίδοση του Συνελικτικού Νευρωνικού ΔΙκτύου.

Βήμα 1: Reproducibility

Για να βελτιώσουμε την απόδοση του δικτύου θα πρέπει να δοκιμάσουμε διάφορες τεχνικές. Για να είμαστε σίγουροι αν μια τεχνική βελτιώνει την απόδοση θα πρέπει να εκπαιδεύουμε το δίκτυο υπό ακριβώς τις ίδιες συνθήκες. Αυτό σημαίνει πως η αρχικοποίηση των βαρών και η σειρά των δεδομένων θα πρέπει να είναι κάθε φορά ίδια, ώστε η επίδοση του δικτύου να μην εξαρτάται από κάποιο καλύτερο σημείο αρχικοποίησης ή κάποια ευνοϊκότερη διαχείριση των δεδομένων.

Για τον λόγο αυτό θα πρέπει να κάνετε “seed” όλες τις απαραίτητες βιβλιοθήκες και αλγορίθμους. Συμβουλευτείτε το ακόλουθο notebook και κάντε τις απαραίτητες αλλαγές στον κώδικά σας. Δοκιμάστε να τρέξετε 2 φορές την ίδια ακριβώς διαδικασία εκπαίδευσης. Θα πρέπει να πετυχαίνετε ακριβώς το ίδιο loss σε κάθε εποχή του train και τις ίδιες επιδόσεις στο test set.

Βήμα 2: Activation functions

Το Νευρωνικό Δίκτυό που έχουμε φτιάξει μέχρι στιγμής εφαρμόζει αποκλειστικά γραμμικούς μετασχηματισμούς στα δεδομένα. Για να μπορέσει το δίκτυο να "μάθει" πιο σύνθετες συσχετίσεις στα δεδομένα θα πρέπει να εισάγουμε μη γραμμικές συναρτήσεις ενεργοποίησης (non-linear activation functions). Την συμπεριφορά αυτή θα μας την δώσουν οι μη γραμμικές συναρτήσεις ενεργοποίησης.

Δοκιμάστε διάφορες συναρτήσεις ενεργοποίησης από αυτήν την λίστα, τοποθετώντας τες σε κάθε συνελικτικό επίπεδο (μετά την πράξη της συνέλιξης και πριν το pooling), και σε κάθε είσοδο των γραμμικών επιπέδων. Να φτιάξετεε ένα πινακάκι που να περιέχει στις γραμμές τα activation functions που δοκιμάσατε και τις μετρικές f1 και accuracy στις στήλες.

Βήμα 4: Learning rate scheduler

Το learning rate είναι μία κρίσιμη μεταβλητή του αλγόριθμου βελτιστοποίησης η οποία επηρεάζει άμεσα το στιγμιότυπο στο οποίο τελικά θα καταλήξει το μοντέλο μας. Μέχρι στιγμής έχουμε χρησιμοποιήσει ένα στατικό learning rate. Παρ' όλα αυτά το learning rate που έχουμε επιλέξει δεν είναι η καλύτερη επιλογή για όλα τα βήματα εκτέλεσης, αφού πιθανόν να είναι μικρό για τα αρχικά βήματα στα οποία επιθυμούμε μεγαλύτερες αναπροσαρμογές των βαρών, και μεγάλο για τα τελικά βήματα στα οποία έχουμε προσεγγίσει ένα καλό στιγμιότυπο του μοντέλου και δεν θέλουμε να απομακρυνθούμε αρκετά από αυτό.

Υπάρχει η δυνατότητα να χρησιμοποιήσουμε learning rate schedulers οι οποίοι αναπροσαρμόζουν δυναμικά το learning rate ανάλογα με το ιστορικό των losses. Εισάγετε στην διαδικασία εκπαίδευσής σας διαφορετικούς schedulers (από εδώ), δίνοντας στους schedulers σαν όρισμα verbose=True ώστε να έχετε εποπτεία των δυναμικών αλλαγών που γίνονται. Τι αποδόσεις πετυχαίνετε;

Βήμα 5: Batch Normalization

Τα νευρωνικά δίκτυα λειτουργούν καλύτερα εάν τα δεδομένα έχουν συγκεκριμένες στατιστικές ιδιότητες. Για να πετύχουμε αυτές τις ιδιότητες συνήθως πραγματοποιούμε normalization στο input. Παρ' όλα αυτά καθώς εφαρμόζουμε περισσότερα επίπεδα του δικτύου και καθώς ενημερώνονται τα βάρη, ενδέχεται οι στατιστικές ιδιότητες του input κάθε layer να διαφέρουν από το ένα batch στο άλλο. Αυτό δεν βοηθά τον αλγόριθμο εκπαίδευσης αφού προσπαθεί να μάθει από δεδομένα των οποίων αλλάζει λίγο η κατανομή μεταξύ των batches. Μία λύση σε αυτό το πρόβλημα είναι να εφαρμόζουμε, σε κάθε layer, normalization σε όλα τα στοιχεία του batch ώστε να πετύχουμε ίδιες ιδιότητες.

Εισάγετε λοιπόν στην αρχιτεκτονική σας BatchNorm2d layers πριν απο κάθε συνάρτηση ενεργοποίησης όλων των συνελικτικών επιπέδων.

Βήμα 6: Regularization

Δοκιμάστε να αμβλύνετε τη διαφορά του train loss απο το validation loss βάζοντας και δοκιμάζοντας διαφορετικές τιμές: (i) weight_decay στον optimizer και (ii) dropout στα linear layers. Αυξήστε τον αριθμό των εποχων από 30 σε 60 και δοκιμάστε τα (i) και (ii) μαζί και ξεχωριστά. Τι επίδοση πετυχαίνετε στο test set;

Βήμα 7: Training efficiency

Batch size
Μία μεταβλητή εκπαίδευσης που μπορεί να επηρεάσει την τελική απόδοση του μοντέλου, αλλά και τον χρόνο εκπαίδευσης είναι το batch size. Δοκιμάστε να θέσετε ως batch size τις 7 πρώτες δυνάμεις του 2 και να τυπώσετε τόσο την απόδοση όσο και τον χρόνο εκτέλεσης. Σχολιάστε τα αποτελέσματά σας.

Early stopping
Κατά την διάρκεια των πειραμάτων της έργασιας αυτής παρατηρούμε ότι αρκετές φορές, το καλύτερο στιγμιότυπο του μοντέλου πετυχαίνεται αρκετά πριν τον αριθμό εποχών που έχουμε ορίσει. Επομένως η συνέχιση της εκπαίδευσης μέχρι να φτάσουμε τον τελικό αριθμό εποχών κάνει την διαδικασία και περισσότερο χρονοβόρα αλλά προκαλεί και κατασπατάληση πόρων, κάτι το οποίο είναι μη επιθυμητό ειδικά σε σερβερς που πολλοι χρήστες επιθυμούν να έχουν πρόσβαση στις GPU. Για τον λόγο αυτό ο αριθμός εποχών που θέτουμε θα πρέπει να θεωρείται ως “ο μέγιστος αριθμός εποχών”. Επομένως, ενσωματώστε στην διαδικασία εκπαίδευσης και το early stopping, δηλαδή μία συνθήκη τερματισμού της εκπαίδευσης σε περίπτωση που η επίδοση του μοντέλου στο validation set για ένα συνεχόμενο (patience) αριθμό εποχών (π.χ. 7) δεν βελτιώνεται. Παρατηρείτε διαφορά στον χρόνο εκτέλεσης για διαφορετικές τιμές του patience;

[Ερώτημα 4: Testing] (10 βαθμοί) 0.375

Με την αξιολόγηση του ταξινομητή στο test set προσπαθούμε να αποκτήσουμε μια εικόνα της ικανότητας γενίκευσής του σε δεδομένα που δεν έχει χρησιμοποιήσει κατά την εκπαίδευση. Για να δούμε πόσο κοντά είναι αυτή η εικόνα στην πραγματικότητα, θα παράξουμε προβλέψεις για δεδομένα από τον πραγματικό κόσμο (youtube videos).

Βήμα 1: Inference

Εδώ θα χρειαστεί να φτιαχτεί μία συνάρτηση που θα παίρνει ένα σύνολο δεδομένων (dataloader με shuffle=False) και ένα εκπαιδευμένο Συνελικτικό Δίκτυο και θα επιστρέφει μία λίστα με τις προβλέψεις του μοντέλου

Βήμα 2: Κατέβασμα μουσικής

Εγκαταστήστε το youtube-dl στο colab χρησιμοποιώντας την παρακάτω ακολουθία εντολών:

```
!sudo apt-get update
```

!sudo curl -L https://yt-dl.org/downloads/latest/youtube-dl -o /usr/local/bin/youtube-dl

!sudo chmod a+rx /usr/local/bin/youtube-dl

Το αρχείο youtube.py που σας έχει δοθεί μαζί με τα δεδομένα περιέχει συναρτήσεις οι οποίες, δεδομένου ενός youtube url, κατεβάζουν τον ήχο και υπολογίζουν μία ακολουθία από mel spectrograms (ένα για κάθε δευτερόλεπτο). Η συνάρτηση youtube_to_melgram αποθηκεύει στο αρχείο melgrams.npy την ακολουθία melgram ενός δεδομένου url. Χρησιμοποιήστε την με τουλάχιστον 1 url απο κάθε μουσικό είδος που περιέχεται στο σύνολο δεδομένων μας. Είστε ελεύθεροι να χρησιμοποιήσετε όποιο url θέλετε. Σας δίνονται ενδεικτικά:

- κλασσική μουσική:https://www.youtube.com/watch?v=9E6b3swbnWg
- ποπ: https://www.youtube.com/watch?v=EDwb9jOVRtU
- ροκ: https://www.youtube.com/watch?v=OMaycNcPsHI
- μπλουζ: https://www.youtube.com/watch?v=l45f28PzfCI

Βήμα 3: Προβλέψεις

Για κάθε ένα από τα μουσικά είδη χρησιμοποιήστε την συνάρτηση του βήματος 1 ώστε να παράξετε προβλέψεις. Τυπώστε ένα διάγραμμα όπου στον κατακόρυφο άξονα θα βρίσκονται οι μουσικές κλάσεις και στον οριζόντιο τα timestamps (που αντιστοιχούν σε δευτερόλεπτα). Βγάζουν τα αποτελέσματά σας νόημα αν τα συγκρίνετε με τον ήχο των youtube videos; Εποπτικά σχολιάστε πόσο κοντά είναι η απόδοση του ταξινομητή σας στα youtube videos (σε προβλέψεις τους ενός δευτερολέπτου) σε σχέση με την απόδοση στο test set.