ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΑΝΑΓΝΩΡΙΣΗ ΠΡΟΤΥΠΩΝ
Χειμερινό Εξάμηνο 2018-19

2η Εργαστηριακή Άσκηση:
Αναγνώριση φωνής με Κρυφά Μαρκοβιανά Μοντέλα και Αναδρομικά Νευρωνικά Δίκτυα

ΠΕΡΙΓΡΑΦΗ

Σκοπός είναι η υλοποίηση ενός συστήματος επεξεργασίας και αναγνώρισης φωνής, με εφαρμογή σε αναγνώριση μεμονωμένων λέξεων. Το πρώτο μέρος αποσκοπεί στην εξαγωγή κατάλληλων ακουστικών χαρακτηριστικών από φωνητικά δεδομένα, χρησιμοποιώντας τα κατάλληλα πακέτα python, καθώς και η ανάλυση και απεικόνισή τους με σκοπό την κατανόηση και την εξαγωγή χρήσιμων πληροφοριών από αυτά. Τα εν λόγω χαρακτηριστικά είναι στην ουσία ένας αριθμός συντελεστών cepstrum που εξάγονται μετά από ανάλυση των σημάτων με μια ειδικά σχεδιασμένη συστοιχία φίλτρων (filterbank). Η συστοιχία αυτή είναι εμπνευσμένη από ψυχοακουστικές μελέτες.

Πιο συγκεκριμένα, το σύστημα που θα αναπτύξετε αφορά σε αναγνώριση μεμονωμένων ψηφίων (isolated digits) στα Αγγλικά. Τα δεδομένα που θα χρησιμοποιήσετε περιέχουν εκφωνήσεις 9 ψηφίων από 15 διαφορετικούς ομιλητές σε ξεχωριστά .wav αρχεία. Συνολικά θα βρείτε 133 αρχεία, αφού 2 εκφωνήσεις θεωρήθηκαν προβληματικές και δεν έχουν συμπεριληφθεί. Τα ονόματα των αρχείων (π.χ. eight8.wav) υποδηλώνουν τόσο το ψηφίο που εκφωνείται (π.χ. eight), όσο και τον ομιλητή (οι ομιλητές είναι αριθμημένοι από 1-15). Οι εκφωνήσεις έχουν ηχογραφηθεί με συχνότητα δειγματοληψίας ίση με Fs = 16kHz και η διάρκειά τους διαφέρει.

ΒΙΒΛΙΟΘΗΚΕΣ PYTHON

- Διάβασμα αρχείων ήχου και εξαγωγή χαρακτηριστικών: librosa
- Αλγόριθμοι ταξινόμησης: scikit-learn
- Διαγράμματα: matplotlib, seaborn etc.
- Hidden Markov Models: pomegranate
- Νευρωνικά δίκτυα: pytorch

ΕΠΙΠΛΕΟΝ ΛΟΓΙΣΜΙΚΟ

- Praat: http://www.fon.hum.uva.nl/praat/

ΕΚΤΕΛΕΣΗ

Κατεβάστε τα δεδομένα της προπαρασκευής από τις διευκρινίσεις του mycourses.

Βήμα 1
Ανάλυση αρχείων ήχου με το Praat (το οποίο πρέπει να εγκαταστήσετε από το παραπάνω link): Ανοίξτε τα αρχεία *onetwothree1.wav* και *onetwothree8.wav* με το πρόγραμμα Praat. Τα αρχεία αυτά περιέχουν την πρόταση “one two three” από τους ομιλητές 1 και 8, οι οποίοι είναι άντρας και γυναίκα αντίστοιχα. Παρατηρήστε τις κυματομορφές και τα spectrograms και έπειτα εξάγετε τη μέση τιμή του pitch στα φωνήεντα “α”, “ου”, “ι” για τα 3 ψηφία και για κάθε ομιλητή. Έπειτα, εξάγετε τα 3 πρώτα formants του κάθε φωνήεντος. Παρουσιάστε τα αποτελέσματα και γράψτε τις παρατηρήσεις σας.

Χρησιμοποιώντας Python 3, εκτελέστε τα παρακάτω βήματα:

Βήμα 2
Φτιάξτε μία συνάρτηση (data parser) που να διαβάζει όλα τα αρχεία ήχου που δίνονται μέσα στο φάκελο *digits/* και να επιστρέφει 3 λίστες Python, που να περιέχουν: Το wav που διαβάστηκε με librosa, τον αντίστοιχο ομιλητή και το ψηφίο.

Βήμα 3
Εξάγετε με το librosa τα Mel-Frequency Cepstral Coefficients (MFCCs) για κάθε αρχείο ήχου. Εξάγετε 13 χαρακτηριστικά ανά αρχείο. Χρησιμοποιήστε μήκος παραθύρου 25 ms και βήμα 10 ms. Επίσης, υπολογίστε και την πρώτη και δεύτερη τοπική παράγωγο των χαρακτηριστικών, τις λεγόμενες deltas και delta-deltas (hint: υπάρχει έτοιμη υλοποίηση στο librosa).

Βήμα 4
Αναπαραστήστε τα ιστογράμματα του 1ου και του 2ου MFCC των ψηφίων n1 και n2 για όλες τους τις εκφωνήσεις. Πόση απόκλιση υπάρχει?
Εξάγετε για 2 εκφωνήσεις των n1 και n2 από 2 διαφορετικούς ομιλητές τα Mel Filterbank Spectral Coefficients (MFSCs), δηλαδή τα χαρακτηριστικά που εξάγονται αφού εφαρμοστεί η συστοιχία φίλτρων της κλίμακας Mel πάνω στο φάσμα του σήματος φωνής αλλά χωρίς να εφαρμοστεί στο τέλος ο μετασχηματισμός DCT (εξάγετε και πάλι χαρακτηριστικά διάστασης 13). Αναπαραστήστε γραφικά τη συσχέτιση των MFSCs για την κάθε εκφώνηση. Σε ξεχωριστά διαγράμματα πραγματοποιήστε το ίδιο για τα MFCCs. Τι παρατηρείτε? Γιατί χρησιμοποιούμε τα MFCCs αντί των MFSCs?

Βήμα 5
Μια πρώτη προσέγγιση για την αναγνώριση των ψηφίων είναι η εξαγωγή ενός μοναδικού διανύσματος χαρακτηριστικών για κάθε εκφώνηση.
Ενώστε τα mfccs – deltas – delta-deltas και έπειτα για κάθε εκφώνηση δημιουργείστε ένα διάνυσμα παίρνοντας τη μέση τιμή και την τυπική απόκλιση κάθε χαρακτηριστικού για όλα τα παράθυρα της εκφώνησης. Αναπαραστήστε με scatter plot τις 2 πρώτες διαστάσεις των διανυσμάτων αυτών, χρησιμοποιώντας διαφορετικό χρώμα και σύμβολο για κάθε ψηφίο. Σχολιάστε το διάγραμμα.

Βήμα 6
Μια καλή τακτική για απεικόνιση πολυδιάστατων διανυσμάτων είναι η μείωση των διαστάσεών τους με Principal Component Analysis (PCA). Μειώστε σε 2 τις διαστάσεις των διανυσμάτων του προηγούμενου βήματος με PCA και δημιουργείστε εκ νέου το scatter plot. Σχολιάστε και επαναλάβετε τη διαδικασία για 3 διαστάσεις και τρισδιάστατο scatter plot.
Τι ποσοστό της αρχικής διασποράς διατηρούν οι συνιστώσεις που προέκυψαν? Τι πληροφορία δίνουν αυτά τα νούμερα για τα principal components? Είναι επιτυχημένη η μείωση διαστάσεων?

Βήμα 7
Χωρίστε τα δεδομένα σε train-test με αναλογία 70%-30%. Ταξινομήστε με χρήση του Bayesian ταξινομητή της πρώτης εργαστηριακής άσκησης, καθώς και του Naive Bayes του scikit-learn. Χρησιμοποιήστε επίσης, άλλους 3 ταξινομητές της επιλογής σας. Αναφέρετε το ποσοστό επιτυχίας στο test set και συγκρίνετε τα αποτελέσματα. Σημείωση: Τα δεδομένα πριν την ταξινόμηση πρέπει να κανονικοποιηθούν.
(Bonus: Θα αυξηθεί το ποσοστό επιτυχίας αν προσθέσω επιπλέον ηχητικά χαρακτηριστικά στο διάνυσμά μου, όπως π.χ. zero-crossing rate? Χρησιμοποιήστε ελεύθερα τέτοια επιπλέον χαρακτηριστικά και εάν αυξηθεί το ποσοστό επιτυχίας αναφέρετε τη σχετική αύξηση, διαφορετικά αναφέρετε τους λόγους που απέτυχε η προσπάθειά σας)

Βήμα 8
Εξοικείωση με το PyTorch:
Δημιουργείστε ακολουθίες 10 σημείων ενός ημιτόνου και ενός συνημιτόνου με συχνότητα $f = 40$ *Hz*. Σκοπός είναι η πρόβλεψη του συνημιτόνου με δεδομένη την ακολουθία του ημιτόνου. Επιλέξτε σταθερή και μικρή

απόσταση ανάμεσα στα διαδοχικά σημεία.
Εκπαιδεύστε ένα Αναδρομικό Νευρωνικό Δίκτυο (Recurrent Neural Network – RNN), το οποίο θα δέχεται ως είσοδο τις ακολουθίες του ημιτόνου και θα πρέπει να προβλέπει τις αντίστοιχες ακολουθίες συνημιτόνου. Αντί για χρήση του απλού RNN μπορούν να χρησιμοποιηθούν και οι μονάδες LSTM και GRU (δώστε το λόγο που τις χρησιμοποιήσατε και γιατί είναι τόσο διαδεδομένες).

Τα παρακάτω βήματα δεν αποτελούν μέρος της προπαρασκευής

Για το κυρίως μέρος της άσκησης θα χρησιμοποιηθεί ένα μεγαλύτερο σετ δεδομένων, το Free Spoken Digit Dataset (FSDD), το οποίο μπορείτε να κατεβάσετε από εδώ: https://github.com/Jakobovski/free-spoken-digit-dataset. Στο βοηθητικό υλικό που θα σας δωθεί, υπάρχει υλοποιημένη η διαδικασία διαβάσματος των νέων δεδομένων, εξαγωγής των MFCCs, κανονικοποίησης και χωρισμού σε train και test set (*parser.py*). Η συνάρτηση αυτή θα πρέπει να πάρει ως όρισμα τη διεύθυνση του φακέλου *recordings*.

Βήμα 9:
Χωρίστε τα train δεδομένα σε training και validation set με ποσοστό 80%-20%. Προσέξτε να διαχωρίσετε με τέτοιο τρόπο τα δεδομένα ώστε να διατηρηθεί ίδιος ο αριθμός των διαφορετικών ψηφίων σε κάθε set (stratified split).

Βήμα 10:
Αναγνώριση ψηφίων με GMM-HMM (Gaussian Mixture Models – Hidden Markov Models).
Για το ερώτημα αυτό θα χρησιμοποιήσετε τη βιβλιοθήκη *pomegranate* της Python (εγκατάταση: *pip install pomegranate).*
Αρχικοποιήστε ένα GMM-HMM μοντέλο **για κάθε ψηφίο**. Το μοντέλο θα πρέπει να είναι της μορφής left – right. Συγκεκριμένα, αν $A=\{a_{ij}\}$ είναι ο πίνακας μεταβάσεων του μοντέλου, τότε $a_{ij}=0\,\gamma\iota\alpha\; j<i$, ενώ οι αρχικές πιθανότητες των καταστάσεων είναι:

$$\pi_i=\begin{cases}0 & i\neq 1\\ 1 & i=1\end{cases}$$

Επιπλέον επιτρέπονται μεταβάσεις μόνο μεταξύ διαδοχικών καταστάσεων, δηλαδή υπάρχει ο περιορισμός $a_{ij}=0\,\gamma\iota\alpha\; j>i+1$.
Ένα διάνυσμα ακουστικών χαρακτηριστικών, όπως αυτό εξάγεται από την επεξεργασία ενός πλαισίου φωνής, αποτελεί μια πιθανή παρατήρηση σε κάποια κατάσταση. Λόγω του ότι είναι επιτρεπτές συνεχείς μεταβολές τέτοιων παρατηρήσεων, η πιθανότητα τους μοντελοποιείται με ένα μίγμα Γκαουσιανών κατανομών (GMM).
(Δείτε το βοηθητικό κώδικα (*hmm.py*) ως πρότυπο για την υλοποίηση ενός τέτοιου μοντέλου)

Βήμα 11:
Στη φάση αυτή εκπαιδεύονται τα 10 μοντέλα με χρήση του αλγορίθμου Expectation Maximization. Ο αλγόριθμος εφαρμόζεται για καθορισμένο πλήθος επαναλήψεων N_{iter} ή εως να υπάρξει σύγκλιση. Η σύγκλιση ελέγχεται μέσω της μεταβολής του αλγορίθμου της πιθανοφάνειας (Log Likelihood, πιθανότητα των δεδομένων με γνωστό μοντέλο). Για την εκπαίδευση κάθε μοντέλου (που αντιστοιχεί σε κάποιο ψηφίο) χρησιμοποιείστε όλα τα διαθέσιμα δεδομένα για το ψηφίο αυτό.
Χρησιμοποιήστε από 1 έως 4 καταστάσεις HMM και από 1 έως 5 Γκαουσιανές κατανομές.

Βήμα 12:
Αναγνώριση μεμονωμένων ψηφίων – Testing. Ολοκληρώνοντας τη διαδικασία της εκπαίδευσης, έχετε καταλήξει στις εκτιμήσεις των παραμέτρων των 10 μοντέλων (δηλαδή ένα μοντέλο για κάθε ψηφίο). Στη συνέχεια υπολογίζεται ο λογάριθμος της πιθανοφάνειας (log likelihood) για κάθε εκφώνηση η οποία ανήκει στο σύνολο των δεδομένων για αναγνώριση. Το μοντέλο το οποίο δίνει τη μέγιστη πιθανοφάνεια είναι και το αποτέλεσμα της αναγνώρισης για τη συγκεκριμένη εκφώνηση. Τέλος, για κάθε μοντέλο (ψηφίο) υπολογίζεται το πλήθος των αποτελεσμάτων όπως αυτά κατανέμονται στις διαφορετικές κατηγορίες ψηφίων.
Θα πρέπει να πραγματοποιήσετε αυτή τη διαδικασία αρχικά μόνο στο validation set, μεταβάλλοντας τις

παραμέτρους εκπαίδευσης του μοντέλου ώστε να καταλήξετε στο καλύτερο δυνατό αποτέλεσμα. Έπειτα, με τις παραμέτρους του καλύτερου μοντέλου θα αναγνωρίσετε τα ψηφία του test set.
Γιατί πραγματοποιούμε αυτή τη διαδικασία? Σε τι μας βοηθάει και εάν δεν την κάνουμε τι κίνδυνος υπάρχει?

Βήμα 13:
Confusion Matrix: Σχηματίστε 2 πίνακες οι οποίοι θα περιέχουν τα αποτελέσματα της διαδικασίας του βήματος 12 για το validation και το test set.
Πιο συγκεκριμένα, ο Confusion Matrix πίνακας είναι της μορφής 10x10, και στις γραμμές του περιέχει τα προς ταξινόμηση ψηφία, ενώ στις στήλες του τις κλάσεις στις οποίες αυτά ταξινομήθηκαν.
Accuracy: Επίσης, υπολογίστε ένα ολικό ποσοστό αναγνώρισης ως το ποσοστό των σωστά κατηγοριοποιημένων εκφωνήσεων.

Βήμα 14:
Εκπαιδεύστε ένα Αναδρομικό Νευρωνικό Δίκτυο πάνω στο training set, χρησιμοποιώντας το validation set για τη ρύθμιση των υπερπαραμέτρων.

1. Χρησιμοποιήστε το βοηθητικό κώδικα (*lstm.py)* για την υλοποίηση του δικτύου.
2. Αρχικά υλοποιήστε ένα απλό LSTM δίκτυο.
3. Εκπαιδεύστε το δίκτυο, τυπώνοντας μόνο το training loss σε κάθε εποχή.
4. Εκπαιδεύστε το δίκτυο, τυπώνοντας το training loss και σε κάθε εποχή κάντε αποτίμηση του μοντέλου στο validation set και τυπώστε το validation loss.
5. Προσθέστε στο μοντέλο σας Dropout και L2 Regularization. Εξηγήστε τι κάνει το καθένα και σε τι βοηθάνε κατά την εκπαίδευση.
6. Υλοποιήστε Early Stopping και Checkpoints (δηλαδή αποθήκευση του καλύτερου μοντέλου σε pickle). Εξηγήστε γιατί είναι σωστό να χρησιμοποιούμε Early Stopping κατά την εκπαίδευση.
7. Εκπαιδεύστε ένα Bidirectional LSTM. Τι αλλάζει εσωτερικά στο δίκτυο και τι μας προσδίδει αυτή η αλλαγή?
8. (Bonus) Χρησιμοποιήστε το pack_padded_sequence στην εκπαίδευση του μοντέλου. Θα χρειαστεί να ταξινομήσετε τις ακολουθίες που δίνετε στη συνάρτηση αυτή του PyTorch κατά φθίνουσα σειρά μήκους (μην ξεχάσετε να κρατήσετε και τα αντίστοιχα labels όταν γίνει η ταξινόμηση). Συγκρίνετε την ταχύτητα εκπαίδευσης του δικτύου με πριν.

Παρουσιάστε το ποσοστό επιτυχίας του καλύτερου μοντέλου στο validation και στο test set, καθώς και το confusion matrix. Επίσης, δώστε το διάγραμμα του σφάλματος στο training set για κάθε εποχή εκπαίδευσης και στο ίδιο διάγραμμα με διαφορετικό χρώμα επίσης σχεδιάστε το σφάλμα στο validation set για κάθε εποχή.

Προχωρήστε σε κατ'οίκον ολοκλήρωση των βημάτων εκείνων που δεν προλάβατε κατά τη διεξαγωγή του εργαστηρίου.

ΣΗΜΕΙΩΣΗ

Τα ψηφία n1 και n2 που αναφέρονται παραπάνω, είναι το προτελευταίο και το τελευταίο ψηφίο του Α.Μ. σας αντίστοιχα. Αν κάποιο είναι το 0, τότε ορίστε το ως το προηγούμενο ή επόμενο του άλλου ψηφίου, το οποίο δεν είναι μηδενικό.

ΠΑΡΑΔΟΤΕΑ

(1) Σύντομη αναφορά (σε pdf ή jupyter notebook) που θα περιγράφει τη διαδικασία που ακολουθήθηκε σε κάθε βήμα, καθώς και τα σχετικά αποτελέσματα. Τα αποτελέσματα πρέπει να συνοδεύονται και από ερμηνεία – σχολιασμό.
(2) Κώδικας Python (συνοδευόμενος από σύντομα σχόλια). Προσπαθήστε να κάνετε vectorized υλοποιήσεις.

Συγκεντρώστε τα (1) και (2) σε ένα .zip αρχείο το οποίο πρέπει να αποσταλεί μέσω του mycourses.ntua.gr πριν από τη διεξαγωγή του εργαστηρίου.

ΠΑΡΑΡΤΗΜΑ: Οδηγίες εγκατάστασης πακέτων Python

1η εναλλακτική: Χρήση του Virtual Machine και του υλικού από το εισαγωγικό εργαστήριο Python: https://github.com/georgepar/python-lab

2η εναλλακτική: Εγκατάσταση με miniconda: https://conda.io/miniconda.html
Ο τρόπος αυτός είναι εύκολος και για Linux και για Windows χρήστες.

- Εγκαταστήστε το miniconda, σε Python 3, και έπειτα ανοίξτε ένα terminal.
- Μπορείτε να εγκαταστήσετε τα πακέτα στο default περιβάλλον του miniconda, αλλιώς μπορείτε να δημιουργήσετε ένα καινούργιο περιβάλλον με την εντολή:
  conda create -n my_env pip
  Για να ενεργοποιήσετε το περιβάλλον αυτό η εντολή είναι:
  source activate my_env
- Για χρήστες Windows εμφανίζεται ένα καινούργιο εκτελέσιμο πρόγραμμα μετά την εγκατάσταση εάν κάνετε search: Open Anaconda Prompt. Όταν ανοίξει το terminal ακολουθείτε την ίδια διαδικασία με τους χρήστες Linux.
- Για την εγκατάσταση των πακέτων που χρειάζονται για την άσκηση η εντολή είναι η εξής:
  pip install numpy matplotlib librosa scikit-learn pomegranate torch torchvision
- Για όσους επιθυμούν να δουλέψουν σε jupyter notebook θα πρέπει να τρέξουν επίσης:
  pip install jupyter
  και για να ανοίξουν τον notebook server:
  jupyter notebook