ΕΠΛ 442: Μηχανική Μάθηση

ΕΡΓΑΣΙΑ 2: Letter Recognition
25/10/2023

Νικόλαος Θεοδώρου 1030496

Εισαγωγή

Σε αυτήν την αναφορά περιγράφονται οι λεπτομέρειες της εκπαίδευσης ενός νευρωνικού δικτύου για την επίλυση του προβλήματος αναγνώρισης γραμμάτων του αγγλικού αλφαβήτου. Περιλαμβάνονται οι χρησιμοποιηθείσες μεταβλητές, οι επιλογές που αφορούν τη δομή του νευρωνικού δικτύου και τα αποτελέσματα που προέκυψαν από την εκπαίδευση. Η εργασία τρέχει με την εντολή python3 Train.py και απαιτεί python 3.6 η νεότερη και τα πακέτα numpy και matplotlib και έχει δοκιμαστεί σε περιβάλλον Linux/Debian.

Προετοιμασία δεδομένων εισόδου

Αρχικα τα δεδομένα θα πρέπει να κανονιικοποιηθούν πριν μπουν στο νευρωνικο δύκτιο. Χρησιμοποίησα την μέθοδο κανονικοποίσης MinMax για να κανονικοποιήσω κάθε στύλη των δεδομένων.
Στη συνέχεια οργάνωσα τα δεδομένα ανά γράμμα που υπάρχει στην έξοδο.
Μετά, μοίρασα τα ¾ των δεδομένων στο train set και το ¼ στο test set.
Τέλος, ανακάτεψα τις εγγραφές των κάθε set δεδομένων.

Η πιο πάνω διαδικασία γίνεται με τη βοήθεια του κώδικα που έγραψα στα αρχεία FileManager.py και utility.py

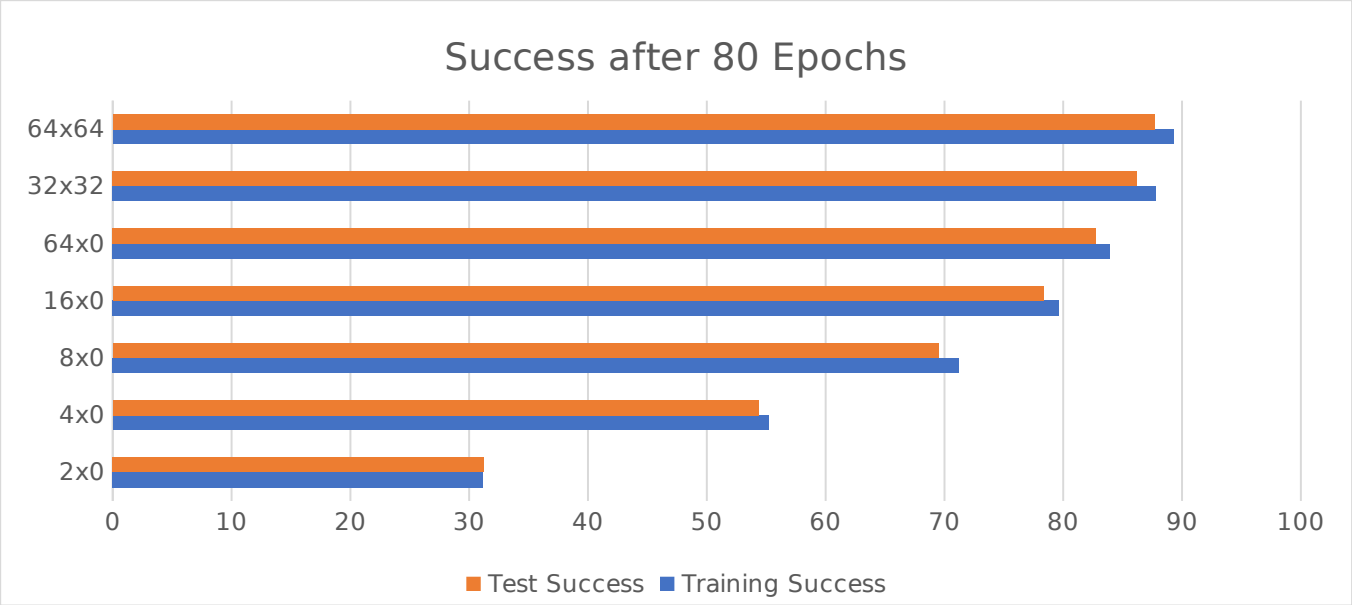
Αυτή η διαδικασία γίνεται αυτόματα με το τρέξιμο του script Trainer.py, πριν να εκκινήσει η εκπαίδευση.

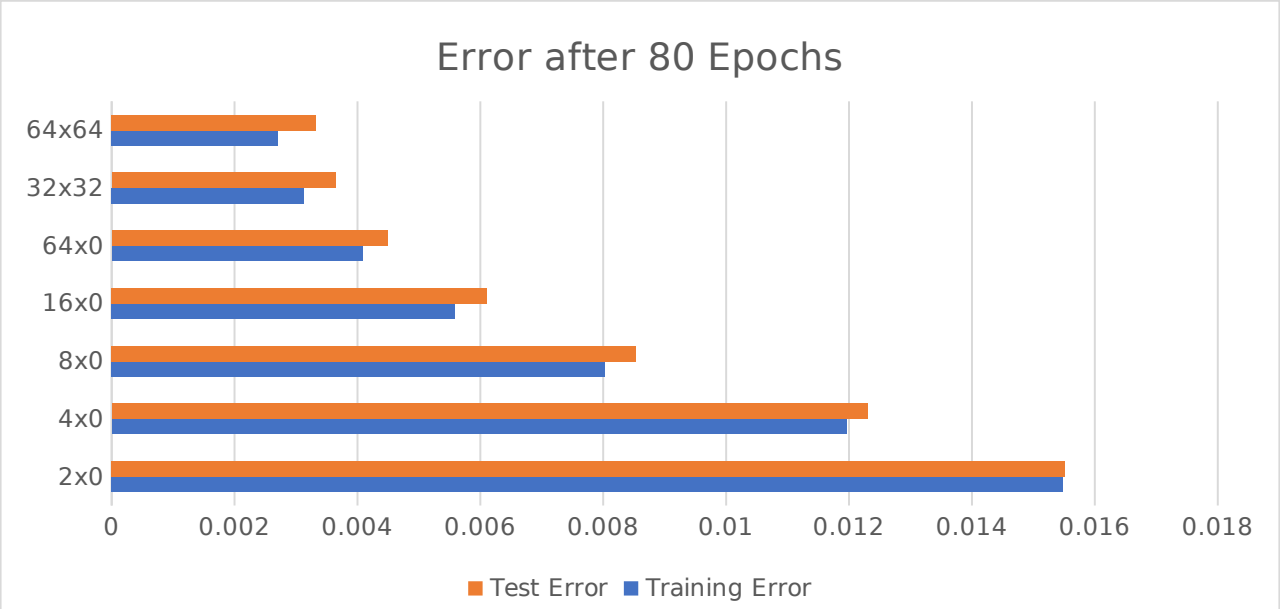
Μεταβλητές και Επιλογές για το Νευρωνικό Δίκτυο

Έγινε δοκιμή διάφορων παραμέτρων του νευρωνικου δυκτίου τόσο για την τοπολογία του όσο και για το learning rate και για το momentum. Για το learning rate επέλεξα 0.3, αφού δίνει ιδανική αναλογία χρόνου εκπαίδευσης και ακρίβειας. Για το momentum επέλεξα 0.3, αφού μας αποτρέπει από τοπικά ελάχιστα. Επίσης πρόσεξα πως με αυτές τις μεταβλητές μετά από 80 εποχές το success και error rate επιπεδόνονται.

Για την τοπολογία του δυκτίου είναι αναμφισβήτητο πως θα πρέπει να βάλουμε 16 νευρώνες εισόδου, ίσο με τον αριθμό των εισόδων που έχουμε στα δεδομένα μας και 26 εξόδους ίσο με τον αριθμό των γραμμάτων του αγγλικού αλφαβήτου.

Για την τοπολογία των κρυφών επιπέδων έγιναν δοκιμές με τοπογιες (Layer 1 Nodes x Layer 2 nodes) 2x0, 4x0, 8x0, 16x0, 64x0, 32x32, 64x64. Προέκυψαν τα πιο κάτω αποτελέσματα:





Νικητής με πιο ψηλό success rate στο test set, ήταν η διάταξη 64x64 με 87,68%. Όμως η διάταξη 32x32 χρησιμοποιόντας πολύ λιγότερους πόρους είχε success rate 86,21%. Έτσι μπορούμε να πούμε πως είναι πιο αποδοτική η διάταξη του 32x32 ως προς την ακρίβεια ανά πόρους και επιτυγχάνει πάνω από 85% ακρίβεια.

Εκπληκτικό είναι το γεγονός που η διάταξη 4x0 δίνει περισσότερο από 50% ακρίβεια (54,36%).

Τέλος σύφωνα με τα πιο πάνω αποτελέσματα, οι επιλογές που έγιναν για τη δομή του νευρωνικού δικτύου είναι οι εξής:

- Πρώτο Επίπεδο: 16 νευρώνες (16 είσοδοι)
- Δεύτερο Επίπεδο: 32 νευρώνες
- Τρίτο Επίπεδο: 32 νευρώνες
- Τέταρτο Επίπεδο (Έξοδος): 26 νευρώνες (26 γράμματα)

- Ρυθμός Μάθησης (learningRate): 0.3
- Μεταβλητή Momentum: 0.3
- Αριθμός Επαναλήψεων (maxIterations): 80

Αντιστοίχηση και επιλογή εξόδου στο επίπεδο εξόδου με γράμματα

Αντιστοίχησα κάθε νευρώνα εξόδου με κάθε γράμμα του αλφαβήτου και ως έξοδο του δυκτίου έβαλα τον νευρώνα με την πιο ψηλή τιμη εξόδου. Για την επαλήθευση και υπλογισμό σφάλματος, όρισα πως ο νευρώνας που αντιπροσοπεύει την στοχευμένη έξοδο θα πρέπει να επιστρέφει την τιμή 1, ενώ όλοι οι άλλοι 0.

Ενδεικτικά δεδομένα αρχείου all_data.txt:

Out in1 in2 in3 in4 in5 in6 in7 in8 in9 in10 in11 in12 in13 in14 in15 in16

T,2,8,3,5,1,8,13,0,6,6,10,8,0,8,0,8

I,5,12,3,7,2,10,5,5,4,13,3,9,2,8,4,10

Έξοδος

Η έξοδος γινεται στα αρχεία "*error/success (LR={learning rate}, M={momentum}, L1={hidden layer 1 nodes}, L2={hidden layer 2 nodes}).txt*" με το κάθε ένα να έχει τις ανάλογες στήλες για epoch training error/success rate και test error/success rate

Ανάλυση Αποτελεσμάτων

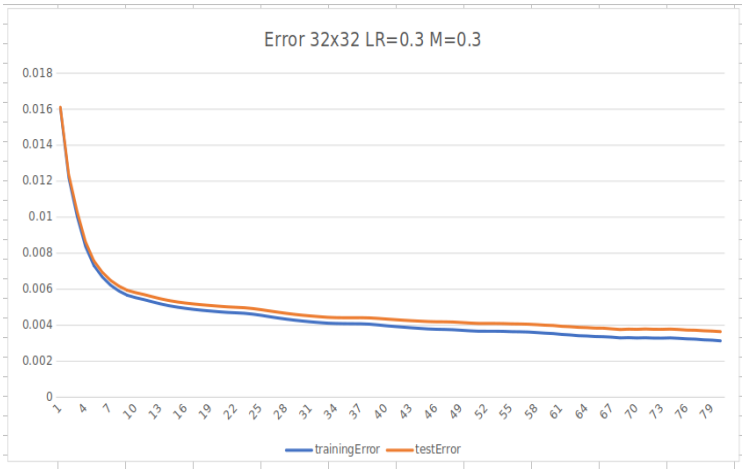
Σύμφωνα με τις πιο κάτω γραφικές, οι προβλεπόμενες εξόδοι σχετίζονται σωστά με τις εισόδους, καταφέρνοντας να επιτύχει το επιθυμητό αποτέλεσμα για όλα τα παραδείγματα του συνόλου δεδομένων.

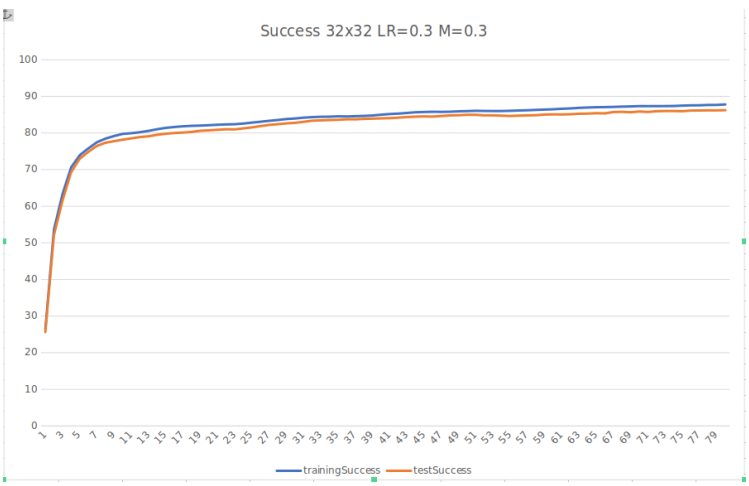
Τα αποτελέσματα δείχνουν ότι το νευρωνικό δίκτυο κατάφερε να μάθει να αναγνωρίζει τα γράμματα. Οι μεταβλητές του δικτύου ενημερώθηκαν με βάση την αλγοριθμική διαδικασία του backpropagation. Παρατηρούμε πως συνήθως μετά απο 10 εποχές εκμάθησης κατα μέσο όρο, το δύκτιο εκπαιδεύεται επαρκώς χωρίς να κάνει λάθη, τόσο στο σύνολο της εκμάθησης όσο και του ελέγχου. Δεν φαίνεται να κολλούμε σε τοπικά ελάχιστα.

Γραφική Παράσταση

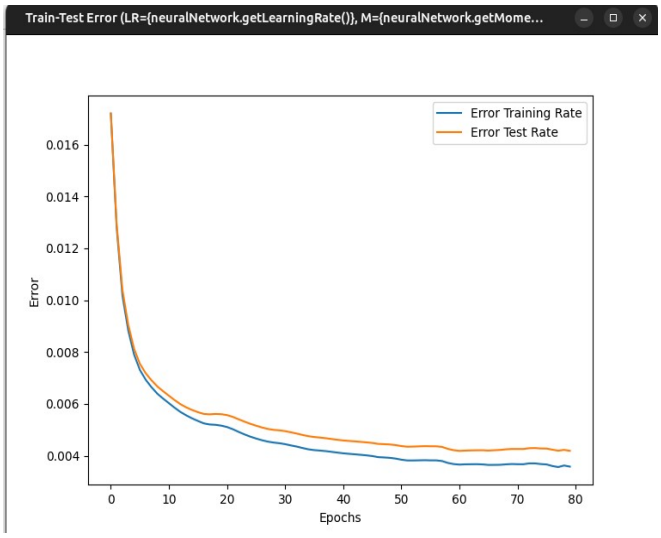
Παρακάτω παρουσιάζονται γραφικές παραστάσεις για τα αποτελέσματα της εκπαίδευσης για διάφορες δοκιμές με διαφορετικό seed κατα την αρχικοποίηση των βαρών:

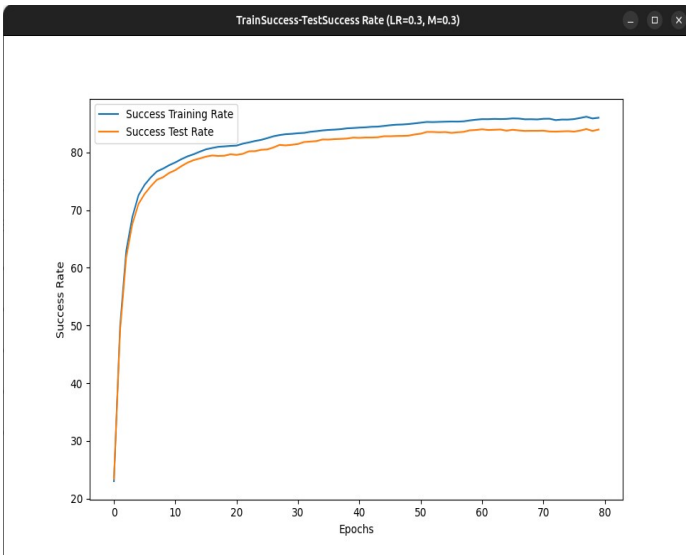
- Γραφική παράσταση του αρχείου errors.txt και success.txt για seed = 3





- Γραφική παράσταση του αρχείου errors.txt και success.txt για seed = 4





Συμπεράσματα

Η επιτυχής εκπαίδευση του νευρωνικού δικτύου να προσεγγίσει την ορθή αναγνώριση των γραμμάτων, αποδεικνύει την αποτελεσματικότητα του αλγορίθμου backpropagation και της κατάλληλης δομής δικτύου. Αποδεικτικέ πως 2 κρυφά layer με 32 νευρώνες είναι αρκετό για να λύσει το πρόβλημα με μεγάλη ακρίβεια. Η γραφική αναπαράσταση των αποτελεσμάτων ενισχύει αυτό το συμπέρασμα, αποτυπώνοντας τη βελτίωση της απόδοσης του δικτύου κατά την πρόοδο της εκπαίδευσης.

Σημείωση

Περαιτέρω τρεξίματα του δυκτίου για την διαπίστωση της τοπολογίας μπορούν να βρεθούν στο αρχείο data.xlsx.

Κλάσεις που χρησιμοποιούνται στον Κώδικα

Στον παρεχόμενο κώδικα, πολλές προσαρμοσμένες κλάσεις χρησιμοποιούνται για τη δημιουργία και τη διαχείριση ενός νευρωνικού δικτύου, την εκτέλεση feed forward και back propagation και τη διαχείριση λειτουργιών αρχείων. Παρακάτω, θα συζητήσουμε καθεμία από αυτές τις κλάσεις και τους ρόλους τους στην υλοποίηση του νευρωνικού δικτύου.

1. Neuron

Η κλάση Neuron αντιπροσωπεύει έναν νευρώνα στο νευρωνικό δίκτυο. Οι νευρώνες είναι οι βασικές μονάδες στο δίκτυο, υπεύθυνες για την επεξεργασία των εισροών, τον υπολογισμό της εξόδου και τη διάδοση του σφάλματος κατά τη διάρκεια της εκπαίδευσης. Τα βασικά χαρακτηριστικά και μέθοδοι αυτής της κλάσης περιλαμβάνουν:

- **Attributes:**
    - `id`: Identifier for the neuron.
    - `connectedFromNeurons`: Neurons connected as inputs to this neuron.
    - `connectedToNeurons`: Neurons connected as outputs to this neuron.
    - `output`: Output of the neuron after activation.
    - `delta`: Delta value used in backpropagation.
    - `weights`: Weights associated with connections to other neurons in the output of this neuron.
    - `isBias`: Boolean indicating whether the neuron is a bias neuron.
    - `isOutput`: Boolean indicating whether the neuron is an output neuron.
- **Methods:**
    - `calculateOutput()`: Calculates the output of the neuron based on its inputs and weights.
    - `calculateDelta()`: Calculates the delta value of the neuron during backpropagation.
    - `updateWeights()`: Updates the weights of the neuron based on the calculated delta using a learning rate and momentum.

2. NeuralNetwork

Η κλάση NeuralNetwork αντιπροσωπεύει ολόκληρο το νευρωνικό δίκτυο. Ενορχηστρώνει τη δημιουργία στρωμάτων, σύνδεση νευρώνων, αρχικοποίηση βάρους και παρέχει μια διεπαφή για εκπαίδευση. Οι βασικές λειτουργίες περιλαμβάνουν:

- **Attributes:**
    - `parameters`: Dictionary containing parameters for the neural network.
    - `layers`: List of layers, each represented as a list of neurons.
- **Methods:**
    - `createLayers()`: Creates the layers of the neural network based on specified parameters.
    - `connectLayers()`: Connects neurons in adjacent layers by establishing connections between them.
    - `initializeWeights()`: Initializes weights for connections between neurons.

- `getLayers()`: Returns the layers of the neural network.
- `getLearningRate()`: Returns the learning rate from the parameters.
- `getMomentum()`: Returns the momentum from the parameters.

3. FeedForward and BackPropagation

Αυτές είναι συναρτήσεις για την εκτέλεση των βημάτων feed forward και back propagation στο νευρωνικό δίκτυο. Διευκολύνουν το πέρασμα προς τα εμπρός για τον υπολογισμό της απόδοσης και το πέρασμα προς τα πίσω για την ενημέρωση των βαρών με βάση το σφάλμα.

4. FileReader and FileWriter

Αυτές οι κλάσεις χειρίζονται την ανάγνωση και εγγραφή σε αρχεία, που χρησιμοποιούνται ιδιαίτερα για την ανάγνωση δεδομένων εκπαίδευσης και δοκιμής, καθώς και παραμέτρων, και εγγραφή αρχείων καταγραφής σφαλμάτων και ποσοστού επιτυχίας.

5. Utility
Χρησιμοποιήθκηε για την οργάνωση και διαχώριση των δεδομένων