Έκθεση τρίτης εργασίας Νευρωνικών Δικτύων - Βαθιάς Μάθησης

Ιωάννης Οικονομίδης

28 Δεκεμβρίου 2022

Περιεχόμενα

1 Εισαγωγή

1.1 Γενικά

Ονομάζομαι Ιωάννης Οικονομίδης. Στην τρίτη εργασία επέλεξα να κάνω αναγνώριση στα δεκαδικά ψηφία του Mnist dataset δημιουργώντας ένα Radial Basis Function (RBF) Neural Network. Στο συμπιεσμένο αρχείο που ανέβασα στο e-Learning υπάρχει αυτό το pdf αρχείο που είναι η έκθεσή μου γραμμένη σε Latex και το python script main.py που είναι το python πρόγραμμα που διαβάζει τα δεδομένα εκπαίδευσης και ελέγχου και εκπαιδεύει και αξιολογεί το RBF νευρωνικό δίκτυό μου που περιέχεται στο κώδικα και το συγκρίνει με τους κατηγοριοποιητές πλησιέστερου κέντρου κλάσης και 1 και 3 πλησιέστερου γείτονα.

2 Ανάλυση κώδικα

2.1 Σημειώσεις

Το κομμάτι της ανάλυσης κώδικα της έκθεσης θα συγκεντρωθεί στην εξήγηση του τι κάνει η κάθε συνάρτηση και κλάση του προγράμματος, χωρίς απαραίτητα να αναλύεται τι κάνει η κάθε γραμμή γιατί αυτό θα έπαιρνε πάρα πολλές σελίδες. Ωστόσο, στο κώδικα του προγράμματος υπάρχουν σχόλια στα αγγλικά πριν από σχεδόν κάθε γραμμή που εξηγούν τι κάνει η αντίστοιχη γραμμή. Για παραπάνω εξηγήσεις από ότι δίνει η έκθεση μπορείτε να δείτε τα σχόλια αυτά.

2.2 Ανάλυση κώδικα συναρτήσεων

2.2.1 read_normalize_flatten_mnist

Η συνάρτηση αυτή φορτώνει το Mnist dataset από το keras framework, το κανονικοποιεί διαιρώντας τις τιμές των δειγμάτων με το 255, κάνει flatten τα δείγματα εκπαίδευσης και ελέγχου από δισδιάστατους πίνακες 28 * 28 σε μονοδιάστατους πίνακες 784 στοιχείων και επιστρέφει τα δείγματα εκπαίδευσης και ελέγχου καθώς και τις αντίστοιχες ετικέτες τους.

2.2.2 ncc και knn

Αυτές οι συναρτήσεις αφορούν την μέτρηση της απόδοσης των κατηγοριοποιητών πλησιέστερου κέντρου κλάσης και k πλησιέστερου γείτονα αντίστοιχα. Και οι δύο δέχονται ως είσοδο τα δεδομένα και τις ετικέτες εκπαίδευσης (x_train, y_train) και τα δεδομένα και τις ετικέτες ελέγχου (x_test, y_test). Η knn δέχεται επίσης ως είσοδο την παράμετρο k που είναι ο αριθμός των γειτόνων του κατηγοριοποιητή k κοντινότερων γειτόνων.

Και στις δύο συναρτήσεις, στην πρώτη γραμμή τους αρχικοποιείται ο αντίστοιχος κατηγοριοποιητής που δίνεται από την βιβλιοθήκη της python sklearn. Στη συνέχεια εκπαιδεύεται ο κατηγοριοποιητής με τα δεδομένα και τις ετικέτες εκπαίδευσης x_train και y_train αντίστοιχα. Στις επόμενες γραμμές υπολογίζεται και

εμφανίζεται η απόδοση του αντίστοιχου κατηγοριοποιητή πρώτα για τα δεδομένα εκπαίδευσης και μετά για τα δεδομένα ελέγχου.

2.2.3 custom_rbf_neural_network

Αυτή η συνάρτηση δημιουργεί και εκπαιδεύει ένα RBF νευρωνικό δίκτυο χρησιμοποιώντας τη κλάση RBFNetwork του προγράμματος και επιστρέφει το δίκτυο αυτό. Επιπλέον εμφανίζει το χρόνο που διήρκησε η εκπαίδευση του δικτύου (χωρίς τον χρόνο αξιολόγησης όλων των δειγμάτων στο τέλος κάθε εποχής) και αξιολογεί τα δεδομένα εκπαίδευσης και ελέγχου. Το δίκτυο που δημιουργεί έχει 784 νευρώνες εισόδου, 1568 κρυφούς νευρώνες RBF και 10 νευρώνες εξόδου, ένα για κάθε ψηφίο του Mnist dataset. Τα χαρακτηριστικά και η μορφή του δικτύου θα αναλυθούν περαιτέρω σε επόμενο κομμάτι της έκθεσης.

2.2.4 third_project

Παίρνει ως είσοδο τα δεδομένα και ετικέτες εκπαίδευσης και ελέγχου και τις επιλογές για το ποιά μοντέλα θα ελεγχθούν (τρεις επιλογές - μία για κάθε ένα από τους δύο κατηγοριοποιητές και ακόμη μία για το RBF νευρωνικό δίκτυο). Για κάθε μοντέλο που έχει επιλεχθεί για μέτρηση της απόδοσής του, το εκπαιδεύει και εκτυπώνει στην οθόνη την επίδοσή του χρησιμοποιώντας τις αντίστοιχες συναρτήσεις που αναφέρθηκαν στις παραγράφους 2.2.2 και 2.2.3

2.2.5 calculate_accuracy

Παίρνει ως είσοδο την έξοδο του μοντέλου σε μορφή πίνακα πιθανοτήτων για την κάθε κλάση και τις πραγματικές ετικέτες και επιστρέφει την μέση ακρίβεια.

2.2.6 find_wrong_prediction

Παίρνει ως είσοδο ένα αντικείμενο της κλάσης RBFNetwork που αντιπροσωπεύει ολόκληρο το RBF νευρωνικό δίκτυο, τα δείγματα x, τις αντίστοιχες ετικέτες y και αν πρέπει να εμφανίζονται τα αποτελέσματα για κάθε δείγμα που ελέγχεται. Ελέγχει κάθε δείγμα για το αν η πρόβλεψη του μοντέλου είναι ίδια με την ετικέτα. Αν είναι ίδια, εμφανίζει την πρόβλεψη και την ετικέτα του δείγματος αν αυτό επιλέχθηκε και συνεχίζει στο επόμενο δείγμα, αλλιώς εμφανίζει τα αποτελέσματα για το δείγμα και τις πιθανότητες που έδωσε ως έξοδο το μοντέλο και επιστρέφει το δείκτη του δείγματος. Αν δεν βρεθεί δείγμα με λάθος πρόβλεψη, τότε επιστρέφει -1.

2.3 Ανάλυση κώδικα κλάσεων

2.3.1 Layer

Αυτή η κλάση υλοποιεί ένα στρώμα του νευρωνικού δικτύου. Περιέχει ένα κατασκευαστή και την μέθοδο forward_pass. Είναι κλάση που πρέπει να κληρονομείται

από τα συγκεκριμένα είδη στρωμάτων και όχι που πρέπει να χρησιμοποιείται άμεσα. Στον κατασκευαστή αρχικοποιείται η έξοδος του στρώματος στο None. Η μέθοδος forward_pass είναι κενή γιατί παρέχεται από την κλάση που κληρονομεί την κλάση αυτή.

2.3.2 RBFInputLayer

Αυτή η κλάση υλοποιεί ένα στρώμα εισόδου του RBF νευρωνικού δικτύου. Κληρονομεί την κλάση Layer που αναλύθηκε νωρίτερα. Περιέχει τις ίδιες μεθόδους με την κλάση που κληρονομεί.

Η μέθοδος forward_pass κάνει ένα forward pass δια μέσω του στρώματος εισόδου παίρνοντας ως είσοδο τις εισόδος του στρώματος που δίνονται ως παράμετρος στην μέθοδο σε μορφή δισδιάστατου πίνακα και αποθηκεύοντας τες ως έξοδο του στρώματος αφού μόνο αυτό το ρόλο παίζει το στρώμα εισόδου στα RBF νευρωνικά δίκτυα.

2.3.3 RBFHiddenLayer

Αυτή η κλάση υλοποιεί ένα κρυφό RBF στρώμα του RBF νευρωνικού δικτύου. Κληρονομεί την κλάση Layer που αναλύθηκε νωρίτερα. Στον κατασκευαστή καλείται ο κατασκευαστής της κλάσης Layer την οποία κληρονομεί, αρχικοποιούνται τα κέντρα και οι παράμετροι σίγμα των νευρώνων του κρυφού RBF στρώματος στο None και αποθηκεύεται ο αριθμός νευρώνων του στρώματος καθώς και ο τρόπος εκπαίδευσής του ("random" ή "k-means").

Η μέθοδος train εκαιδεύει το κρυφό RBF στρώμα υπολογίζοντας τα κέντρα και τις παραμέτρους σίγμα των νευρώνων του με βάση τις εισόδους του στρώματος inputs που δίνονται ως παράμετρος στη μέθοδο. Αν ο τρόπος εκπαίδευσής του κρυφού RBF στρώματος είναι "random", τότε επιλέγεται με sampling ένας αριθμός δειγμάτων που δόθηκαν ως είσοδος, ίσος με τον αριθμό των νευρώνων του στρώματος και τα sampled δείγματα αποθηκεύονται ως κέντρα των νευρώνων του στρώματος.

Αλλιώς αν ο τρόπος εκπαίδευσής του κρυφού RBF στρώματος είναι διαφορετικός από "random", τότε με τη κλάση KMeans της python βιβλιοθήκης sklearn γίνεται k-means clustering στα δείγματα εισόδου που δόθηκαν ως παράμετρος στη μέθοδο και τα κέντρα των clusters που δημιουργήθηκαν αποθηκεύονται ως τα κέντρα των νευρώνων του κρυφού RBF στρώματος. Σημειώνεται πως έχω βάλει όριο επαναλήψεων 5 στη κλάση KMeans της sklearn που χρησιμοποιείται γιατί με μεγαλύτερο όριο το k-means clustering διαρκεί πάρα πολύ ώρα με τα 60000 δείγματα εκπαίδευσης του Mnist dataset.

Στη συνέχεια υπολογίζεται η παράμετρος σίγμα των νευρώνων η οποία είναι ίδια για όλους τους νευρώνες του κρυφού RBF στρώματος χρησιμοποιώντας των παρακάτω τύπο για το σίγμα που υπάρχει στη σελίδα του μαθήματος του e-Learning στην 16η διαφάνεια των διαφάνειων του κυρίου Διαμαντάρα για τα RBFs:

$$\sigma = \frac{d}{\sqrt{2p}},$$

όπου το d είναι η μέγιστη απόσταση μεταξύ των κέντρων των νευρώνων του κρυφού RBF στρώματος και υπολογίζεται με τη συνάρτηση cdist της python βιβλιοθήκης scipy χρησιμοποιώντας την απόσταση Manhattan ή city block ώστε να γίνονται γρήγορα οι υπολογισμοί και το p είναι ο αριθμός των νευρώνων ή κέντρων του κρυφού RBF στρώματος. Μετά η τιμή του σίγμα που υπολογίστηκε αποθηκεύεται ως η τιμή του σίγμα για όλους τους νευρώνες του κρυφού RBF στρώματος.

Η μέθοδος forward_pass κάνει ένα forward pass δια μέσω του κρυφού RBF στρώματος παίρνοντας ως είσοδο τις εισόδος του στρώματος και υπολογίζοντας και αποθηκεύοντας την έξοδό του χρησιμοποιώντας τον παρακάτω τύπο για την Radial Basis Function Gauss που υπάρχει στη σελίδα του μαθήματος του e-Learning στην 3η διαφάνεια των διαφάνειων του κυρίου Διαμαντάρα για τα RBFs:

$$output = e^{-\frac{||x-c||^2}{\sigma^2}},$$

όπου το x είναι το δείγμα εισόδου για το οποίο υπολογίζεται η έξοδος, το c είναι το διάνυσμα κέντρου του νευρώνα του οποίου υπολογίζεται η έξοδος και το σ είναι η τιμή του σίγμα του νευρώνα αυτού. Ο υπολογισμός της Ευκλίδειας απόστασης ή δεύτερης νόρμας $||x - c||$ γίνεται πάλι με τη συνάρτηση cdist της python βιβλιοθήκης scipy, αλλά χρησιμοποιώντας σε αυτή τη περίπτωση την Ευκλίδεια απόσταση ή δεύτερη νόρμα ώστε να γίνονται γρήγορα οι υπολογισμοί. Οι υπολογισμοί γίνονται για όλα τα δείγματα της εισόδου inputs ταυτόχρονα σε μορφή πινάκων ώστε να διαρκούν όσο το δυνατότερο λιγότερο χρόνο οι υπολογισμοί.

2.3.4 RBFOutputLayer

Αυτή η κλάση υλοποιεί ένα στρώμα εξόδου του RBF νευρωνικού δικτύου. Κληρονομεί την κλάση Layer που αναλύθηκε νωρίτερα. Περιέχει ένα κατασκευαστή και τις μεθόδους forward_pass και back_propagate.

Στον κατασκευαστή καλείται ο κατασκευαστής της κλάσης Layer την οποία κληρονομεί και αρχικοποιούνται όλα τα μέλη της κλάσης. Τα biases αρχικοποιούνται στο 0 ενώ τα βάρη αρχικοποιούνται με τυχαίες τυμές που πολλαπλασιάζονται με το 0.01 ώστε να έχουν μικρές τιμές για να μην βγουν σε περιοχή όπου δεν θα αλλάζει η τιμή τους μετά την αρχικοποίηση. Επίσης αρχικοποιούνται στο None οι είσοδοι του στρώματος αλλά και τα διανύσματα κλίσεων ως προς τις εισόδους, τα biases και τα βάρη. Τέλος, αρχικοποιείται και το momentum κάθε bias και βάρους στο μηδέν ώστε να χρησιμοποιηθεί από τον optimizer του στρώματος εξόδου. Όλα τα μέλη της κλάσης βρίσκονται σε μορφή πινάκων ώστε να γίνεται γρήγορα η επεξεργασία τους.

Η μέθοδος forward_pass κάνει ένα forward pass δια μέσω του στρώματος παίρνοντας ως είσοδο τις εισόδος του στρώματος, αποθηκεύοντάς τες για τη χρήση τους στο back propagation και υπολογίζοντας και αποθηκεύοντας την έξοδο του στρώματος με τη χρήση του εσωτερικού γινομένου της numpy.

Η μέθοδος back_propagate κάνει back propagation δια μέσω του στρώματος παίρνοντας ως είσοδο τα διανύσματα κλίσεων ως προς τις εισόδους που επιστρέφει η συνάρτηση ενεργοποίησης του στρώματος. Υπολογίζει τα διανύσματα κλίσεων

ως προς τα biases και τα βάρη του στρώματος. Κατά τους υπολογισμούς, ο πίνακας των εισόδων που χρησιμοποιείται για τον υπολογισμό των διανυσμάτων κλίσεων ως προς τα βάρη πρώτα αναστρέφεται πριν χρησιμοποιηθεί στους υπολογισμούς ώστε να ταιριάζουν οι διαστάσεις των πινάκων κατά το εσωτερικό γινόμενο.

2.3.5 ActivationFunction

Αυτή η κλάση υλοποιεί μια συνάρτηση ενοργοποίησης. Περιέχει ένα κατασκευαστή και τις μεθόδους forward_pass και back_propagate. Είναι κλάση που πρέπει να κληρονομείται από τα συγκεκριμένα είδη συναρτήσεων ενεργοποίησης και όχι που πρέπει να χρησιμοποιείται άμεσα.

Στον κατασκευαστή αρχικοποιούνται όλα τα μέλη της κλάσης. Αρχικοποιούνται στο None οι είσοδοι και οι έξοδοι της συνάρτησης καθώς και τα διανύσματα κλίσεων ως προς τις εισόδους.

Η μέθοδοι forward_pass και back_propagate είναι κενές γιατί παρέχονται από την κλάση που κληρονομεί την κλάση αυτή.

2.3.6 SoftMax

Αυτή η κλάση υλοποιεί τη συνάρτηση ενοργοποίησης SoftMax. Περιέχει ένα κατασκευαστή και τις μεθόδους forward_pass και back_propagate. Στον κατασκευαστή καλείται ο κατασκευαστής της κλάσης ActivationFunction την οποία κληρονομεί.

Η μέθοδος forward_pass κάνει ένα forward pass δια μέσω της συνάρτησης ενεργοποίησης SoftMax και παίρνει ως είσοδο τις εισόδους της συνάρτησης ενεργοποίησης, δηλαδή τις εξόδους του στρώματος εξόδου, αποθηκεύοντάς τες για τη χρήση τους στο back propagation και υπολογίζοντας και αποθηκεύοντας την έξοδο της συνάρτησης ενεργοποίησης με βάση τον τύπο της συνάρτησης SoftMax. Συγκεκριμένα, χρησιμοποιεί τον γνωστό τύπο της συνάρτησης SoftMax, αλλά από όλες τις εισόδους αφαιρεί πρώτα τη μεγαλύτερη είσοδο ώστε όταν ανεβούν στον εκθέτη οι είσοδοι, να μη βγαίνουν υπερβολικά μεγάλες τιμές που θα ξεπερνούν το όριο των αριθμών που μπορεί να χειριστεί ο υπολογιστής.

Η μέθοδος back_propagate κάνει back propagation δια μέσω της συνάρτησης ενεργοποίησης παίρνοντας ως είσοδο τα διανύσματα κλίσεων ως προς τις εισόδους που επιστρέφει η συνάρτηση απώλειας του RBF neural network. Υπολογίζει τα διανύσματα κλίσεων ως προς τις εισόδους της συνάρτησης ενεργοποίησης κάνοντας χρήση πινάκων Τζακόμπι. Δηλαδή, για κάθε έξοδο της συνάρτησης SoftMax και για το κάθε αντίστοιχο διάνυσμα κλίσεων ως προς τις εισόδους που πήρε ως είσοδο, υπολογίζει τον αντίστοιχο πίνακα Τζακόμπι και υπολογίζει το εσωτερικό γινόμενό του με το αντίστοιχο διάνυσμα κλίσεων ως προς τις εισόδους που πήρε ως είσοδο. Αυτό αποθηκεύεται ως το αντίστοιχο διάνυσμα κλίσων ως προς τις εισόδους της συνάρτησης SoftMax για την συγκεκριμένη έξοδο. Οταν αυτό γίνει για την κάθε έξοδο της συνάρτησης SoftMax, αυτά είναι τα διανύσματα κλίσεων ως προς τις εισόδους της συνάρτησης SoftMax.

2.3.7 SgdOptimizer

Αυτή η κλάση υλοποιεί τον Stochastic Gradient Descent optimizer. Περιέχει ένα κατασκευαστή και τις μεθόδους update_learning_rate, update_layer_parameters και increment_iteration_counter.

Στον κατασκευαστή αρχικοποιούνται όλες οι παράμετροι του optimizer. Συγκεκριμένα, το learning rate, το momentum και το decay παίρνουν ως τιμές αυτές που δίνονται ως παράμετροι στον κατασκευαστή. Οι τιμές τους αν δεν δοθούν ως παράμετροι στον κατασκευαστή είναι 0.01 για το learning rate, 0.0 για το momentum και 0.0 για το decay, καθώς αυτές είναι οι τιμές που χρησιμοποιεί ως default για το Stochastic Gradient Descent του keras framework. Επίσης το τωρινό learning rate αρχικοποιείται στη τιμή του αρχικού learning rate ενώ ο μετρητής των επαναλήψεων αρχικοποιείται στο μηδέν.

Η μέθοδος update_learning_rate ενημερώνει το τωρινό learning rate αν το decay είναι διαφορετικό του μηδενός. Συγκεκριμένα, αν το decay δεν είναι μηδέν, το τωρινό learning rate υπολογίζεται από το αρχικό learning rate, το ρυθμό decay και το μετρητή των επαναλήψεων με βάση το τύπο του decay στον Stochastic Gradient Descent optimizer. Αυτή η μέθοδος πρέπει να καλείται πριν από κάθε ενημέρωση που κάνει ο optimizer στις παραμέτρους κάποιου στρώματος του μοντέλου.

Η μέθοδος update_layer_parameters δέχεται ως είσοδο ένα στρώμα και ενημερώνει τα biases και τα βάρη του. Συγκεκριμένα, πρώτα υπολογίζονται οι αλλαγές που πρέπει να γίνουν στα biases και τα βάρη με βάση τον τύπο του momentum (αν το momentum δεν είναι ενεγοποιημένο, δηλαδή είναι 0, τότε το αριστερό κομμάτι της αφαίρεσης μηδενίζεται οπότε δεν επηρεάζει τους υπολογισμούς). Στη συνέχεια αν το momentum είναι ενεργοποιημένο ενημερώνεται το αποθηκευμένο momentum για το κάθε bias και το κάθε βάρος. Τέλος, εφαρμόζονται οι αλλαγές στα biases και τα βάρη.

Η μέθοδος increment_iteration_counter αυξάνει κατά ένα το μετρητή επαναλήψεων ώστε να χρησιμοποιείται στο decay. Αυτή η μέθοδος πρέπει να καλείται μετά από κάθε ενημέρωση που κάνει ο optimizer στις παραμέτρους κάποιου στρώματος του μοντέλου.

2.3.8 Loss

Αυτή η κλάση υλοποιεί μια συνάρτηση απώλειας. Περιέχει ένα κατασκευαστή και τη μέθοδο calculate_loss. Είναι κλάση που πρέπει να κληρονομείται από τα συγκεκριμένα είδη συναρτήσεων απώλειας και όχι που πρέπει να χρησιμοποιείται άμεσα.

Στον κατασκευαστή αρχικοποιούνται τα διανύσματα κλίσης ως προς τις εισόδους της συνάρτησης απώλειας στο None.

Η μέθοδος calculate_loss δέχεται ως είσοδο την έξοδο του μοντέλου και τις αντίστοιχες ετικέτες και επιστρέφει την μέση απώλεια. Κάνει ένα forward pass δια μέσω της συνάρτησης απώλειας (η αντίστοιχη συνάρτηση forward_pass δίνεται από την κλάση που κληρονομεί τη κλάση αυτή) και επιστρέφει τη μέση τιμή της απώλειας.

2.3.9 CategoricalCrossEntropy

Αυτή η κλάση υλοποιεί τη συνάρτηση απώλειας Categorical Cross Entropy. Περιέχει τις μεθόδους forward_pass και back_propagate. Κληρονομεί τη κλάση Loss που αναλύθηκε νωρίτερα.

Η μέθοδος forward_pass κάνει ένα forward pass δια μέσω της συνάρτησης απώλειας Categorical Cross Entropy επιστρέφωντας την απώλεια για κάθε δείγμα και παίρνει ως είσοδο τις προβλέψεις του μοντέλου και τις αντίστοιχες ετικέτες. Συγκεκριμένα αποθηκεύει των αριθμό των δειγμάτων για τα οποία υπάρχουν προβλέψεις και περιορίζει τις τιμές των προβλέψεων στο διάστημα [0.0000001, 0.9999999], ώστε να μην υπάρχουν μηδενικές προβλέψεις το οποίο θα έδινε διαίρεση με το μηδέν κατά το back propagation. Στη συνέχεια δημιουργείται ένας μονοδιάσταστος πίνακας με τις προβλέψεις του μοντέλου για τις σωστές ετικέτες και επιστρέφεται ο αρνητικός λογάριθμος αυτού του πίνακα σύμφωνα με τον τύπο της Categorical Cross Entropy.

Η μέθοδος back_propagate κάνει back propagation δια μέσω του της συνάρτησης απώλειας Categorical Cross Entropy παίρνοντας ως είσοδο τα διανύσματα κλίσεων και τις αντίστοιχες ετικέτες. Αποθηκεύει τον αριθμό των δειγμάτων και των ετικετών και αν οι ετικέτες είναι αποθηκευμένες σε ένα μονοδιάστατο πίνακα, ο πίνακας αυτός μετατρέπεται σε δισδιάστατο με άσσους στην τιμή της ετικέτας που είναι σωστή και μηδέν στις άλλες θέσεις. Τέλος, υπολογίζονται τα διανύσματα κλίσης ως προς τις εισόδους με βάση τον αντίστοιχο τύπο της Categorical Cross Entropy και αφού κανονικοποιηθούν επιστρέφονται.

2.3.10 SoftmaxCategoricalCrossEntropy

Αυτή η κλάση συνδυάζει τη συνάρτηση ενεργοποίησης SoftMax και τη συνάρτηση απώλειας Categorical Cross Entropy σε μία κλάση χρησιμοποιώντας τις αντίστοιχες κλάσεις τους. Αυτή η κλάση χρειάζεται και χρησιμοποιείται αντί των μεμονομένων αντίστοιχων κλάσεων διότι όταν χρησιμοποιούσα τις μεμωνομένες κλάσεις, δημιουργόνταν διαίρεση με το μηδέν κατά το back propagation. Με τη χρήση αυτής της κλάσης, αλλάζει ο τύπος υπολογισμού των διανυσμάτων κλίσης ως προς τις εισόδους, με αποτέλεσμα να μην δημιουργείται διαίρεση με το μηδέν. Περιέχει ένα κατασκευαστή και τις μεθόδους forward_pass και back_propagate.

Στον κατασκευαστή δημιουργούνται αντικείμενα για την κλάση ενεργοποίησης SoftMax και για την κλάση απώλειας Categorical Cross Entropy. Επίσης αρχικοποιείται στο None η έξοδος της συνάρτησης και τα διανύσματα κλίσης ως προς την είσοδό της.

Η μέθοδος forward_pass κάνει πρώτα ένα forward pass δια μέσω της συνάρτησης ενεργοποίησης Soft Max και μετά κάνει ένα ακόμα forward_pass δια μέσω της συνάρτησης απώλειας Categorical Cross Entropy και παίρνει ως είσοδο τις εισόδους της συνάρτησης ενεργοποίσης και τις ετικέτες των δεδομένων. Επίσης επιστρέφει τη μέση απώλεια. Συγκεκριμένα, πρώτα γίνεται το forward pass δια μέσω της συνάρτησης ενεργοποίησης όπως αναφέρεται στην αντίστοιχη κλάση, μετά αποθηκεύεται η έξοδος της συνάρτησης ενεργοποίησης ως η έξοδος αυτής της κλάσης και επιστρέφεται η μέση απώλεια χρησιμοποιώντας την αντίστοιχη μέθοδο

της κλάσης που υλοποιεί την συνάρτηση απώλειας Categorical Cross Entropy. Κατά τον υπολογισμό της μέσης απώλειας γίνεται ταυτόχρονα και το forward pass δια μέσω της συνάρτησης απώλειας Categorical Cross Entropy όπως αναφέρεται στην μέθοδο calculate_loss της κλάσης Loss.

Η μέθοδος back_propagate κάνει back propagation συνδυάζοντας τις συναρτήσεις SoftMax και Categorical Cross Entropy παίρνοντας ως είσοδο τα διανύσματα κλίσεων και τις ετικέτες των δεδομένων. Αποθηκεύει τον αριθμό των δειγμάτων και αν οι ετικέτες είναι αποθηκευμένες σε δισδιάστατο πίνακα, τις αποθηκέυει σε μονοδιάστατο πίνακα του οποίου οι τιμές είναι η θέση της σωστής ετικέτας. Μετά δημιουργεί ένα αντίγραφο των διανυσμάτων κλίσεων ως προς τις εισόδους, αφαιρεί τον άσσο από τις θέσεις των σωστών ετικετών, κανονικοποιεί τον πίνακα και τον αποθηκεύει ως τα διανύσματα κλίσεων ως προς τις εισόδους της συνάρτησης αυτής.

2.3.11 RBFNetwork

Αυτή η κλάση υλοποιεί το RBF νευρωνικό δίκτυο. Προσπάθησα να κάνω τις συναρτήσεις της να μοιάζουν με τις συναρτήσεις της κλάσης Sequential του keras framework. Περιέχει ένα κατασκευαστή και τις μεθόδους add_layer, fit, evaluate και predict οι οποίες έχουν παρόμοια χρήση με τις αντίστοιχες μεθόδους της κλάσης Sequential του keras framework. Βέβαια η κλάση Sequential του keras framework δημιουργεί MLP νευρωνικό δίκτυο αντί για RBF, αλλά κράτησα παρόμοια τη δομή και του δικού μου RBF νευρωνικού δικτύου.

Στον κατασκευαστή αρχικοποιείται οι λιστα των στρωμάτων και η συνάρτηση ενεργοποίησης-απώλειας του στρώματος εξόδου (όπως ανέφερα και στις αντίστοιχες προηγούμενες κλάσεις, η συνάρτηση ενεργοποίησης και η συνάρτηση απώλειας υλοποιούνται ταυτόχρονα από τη κλάση SoftmaxCategoricalCrossEntropy της προηγούμενης παραγράφου). Επιπλέον, δημιουργείται το αντικείμενο του επιλεγμένου optimizer με τις παραμέτρους που δίνονται ως είσοδο στον κατασκευαστή (εδώ η μόνη επιλογή optimizer είναι ο stochasitc gradient descent optimizer).

Η μέθοδος add_layer δέχεται ως είσοδο τον αριθμό των εισόδων και νευρώνων του νέου στρώματος, το είδος του στρώματος και το τρόπο εκπαίδευσής του. Δημιουργεί και προσθέτει το αντίστοιχο στρώμα στη λίστα στρωμάτων του μοντέλου. Ο αριθμός εισόδων που δίνεται ως είσοδος έχει μόνο νόημα αν το νέο στρώμα είναι στρώμα εξόδου. Ο αριθμός νευρώνων που δίνεται ως είσοδος δεν έχει νόημα αν το νέο στρώμα είναι στρώμα εισόδου καθώς αυτού του είδους το στρώμα έχει αριθμό νευρώνων ίσο με τον αριθμό των χαρακτηριστικών των δειγμάτων εκπαίδευσης. Ο τρόπος εκπαίδευσης του νέου στρώματος που δίνεται ως είσοδος έχει μόνο νόημα αν το νέο στρώμα είναι ένα κρυφό RBF στρώμα, αλλιώς δεν αλλάζει κάτι αυτή η παράμετρος και η default τιμή της είναι "random" για τυχαία αρχικοποίηση κέντρων.

Η μέθοδος fit εκπαιδεύει το μοντέλο με τα δείγματα εκπαίδευσης x και τις αντίστοιχες ετικέτες y που δέχεται ως είσοδο και επιστρέφει τον χρόνο που διήρκησε η εκπαίδευση (χωρίς τον χρόνο αξιολόγησης όλων των δειγμάτων στο τέλος κάθε εποχής). Δέχεται επίσης ως είσοδο τον αριθμό των εποχών (default τιμή 1), το κάθε πόσες εποχές εμφανίζεται η πρόοδος (default τιμή 1) και το batch size

(default τιμή 128) τα οποία χρησιμοποιούνται κατά την εκπαίδευση του στρώματος εξόδου με Stochastic Gradient Descent.

Αρχικά, υπολογίζεται ο αριθμός των batch που θα χρειαστεί να δημιουργηθούν με βάση τα δοσμένα δείγματα εκπαίδευσης, αρχικοποιείται στο μηδέν ο συνολικός χρόνος εκπαίδευσης και αποθηκεύεται ο χρόνος αρχής της εκπαίδευσης του κρυφού RBF στρώματος. Αποθηκεύονται το στρώμα εισόδου και το κρυφό RBF στρώμα του μοντέλου και γίνεται forward pass όλων των δειγμάτων εκπαίδευσης δια μέσω του στρώματος εισόδου.

Στη συνέχεια εκπαιδεύεται το κρυφό RBF στρώμα του μοντέλου, δηλαδή υπολογίζονται τα κέντρα και οι τιμές σίγμα των νευρώνων του, και αφού ολοκληρωθεί η εκπαίδευση του κρυφού RBF στρώματος γίνεται και forward pass της εξόδου του στρώματος εισόδου δια μέσω του κρυφού RBF στρώματος. Επίσης προστίθετται στον συνολικό χρόνο εκπαίδευσης ο χρόνος που διήρκησε η εκπαίδευση του κρυφού RBF στρώματος και αρχικοποιείται στο μηδέν ο συνολικός χρόνος διάρκειας των τελευταίων epoch_print_rate εποχών.

Για κάθε εποχή αποθηκεύεται ο χρόνος αρχής της, μετά για κάθε batch αν είναι το τελευταίο αποθηκεύονται ως τωρινή είσοδος και y (ετικέτες) οι τελεταίες έξοδοι του κρυφού RBF στρώματος και οι τελευταίες ετικέτες, αλλιώς αποθηκεύονται οι έξοδοι του κρυφού RBF στρώματος και οι ετικέτες από την αρχή αυτού του batch μέχρι την αρχή του επόμενου μείον ένα. Παρακάτω, αποθηκεύεται το στρώμα εξόδου του μοντέλου, γίνεται forward pass της τωρινής εισόδου δια μέσω του στρώματος εξόδου (δηλαδή του τωρινού batch των εξόδων του κρυφού RBF στρώματος) και μετά του τωρινού batch των εξόδων του στρώματος εξόδου δια μέσω της συνάρτησης ενεργοποίησης-απώλειάς του.

Συνεχίζοντας, γίνεται πρώτα back propagation δια μέσω της συνάρτησης ενεργοποίησης - απώλειας του στρώματος εξόδου (χρησιμοποιώντας τις εξόδους του μοντέλου για το τωρινό batch ως διανύσματα κλίσης, δηλαδή τις εξόδους της συνάρτησης ενεργοποίησης-απώλειας του στρώματος εξόδου) και μετά δια μέσω του στρώματος εξόδου χρησιμοποιώντας τα διανύσματα κλίσης ως προς τις εισόδους της συνάρτησης ενεργοποίησης-απώλειάς του. Μετά γίνεται το optimization. Συγκεκριμένα, ενημερώνεται το learning rate του optimizer, βελτιστοποιείται το στρώμα εξόδου από τον Stochastic Gradient Descent optimizer και αυξάνεται ο μετρητής επαναλήψεων του optimizer.

Επιστρέφοντας τώρα στο βρόχο που περνάει από κάθε εποχή, πρώτα αποθηκεύεται ο χρόνος ολοκλήρωσης της εποχής και προστίθεται η διαφορά του με τον χρόνο αρχής της εποχής στον συνολικό χρόνο εκπαίδευσης και στον συνολικό χρόνο διάρκειας των τελευταίων epoch_print_rate εποχών. Αν η τωρινή εποχή συν ένα διαιρείται με μηδενικό υπόλοιπο από το ρυθμό εμφάνισης προόδου ή είναι η τελευταία εποχή, υπολογίζεται η μέση απώλεια και ακρίβεια του μοντέλου για όλα τα δείγματα εκπαίδευσης αξιολογώντας τα με την αντίστοιχη συνάρτηση (αυτός ο χρόνος είναι που δεν μετράται στον συνολικό χρόνο εκπαίδευσης), υπολογίζεται ο μέσος χρόνος των τελευταίων epoch_print_rate εποχών (περισσότερες λεπτομέρειες για αυτό τον υπολογισμό έχει στα σχόλια του προγράμματος), ξαναμηδενίζεται ο συνολικός χρόνος διάρκειας των τελευταίων epoch_print_rate εποχών και εμφανίζονται η απώλεια και η ακρίβεια μαζί με τη πρόοδο της εκπαίδευσης. Στο τέλος της εκπαίδευσης επιστρέφεται ο συνολικός χρόνος εκπαίδευσης χωρίς

τον χρόνο αξιολόγησης όλων των δειγμάτων στο τέλος κάθε εποχής.

Η μέθοδος evaluate δέχεται ως είσοδο δείγματα και τις αντίστοιχες ετικέτες, αν χρειάζεται να εμφανιστούν η απώλεια και η ακρίβεια και αν χρειάζεται να γίνει ξανά forward pass δια μέσω του κρυφού RBF στρώματος. Υπολογίζει και αν χρειάζεται εμφανίζει την μέση ακρίβεια και απώλεια και τις επιστρέφει. Συγκεκριμένα, πρώτα αποθηκεύονται το κρυφό RBF στρώμα και το στρώμα εξόδου του μοντέλου. Στη συνέχεια, αν επιλέχθηκε, γίνεται ξανά forward pass δια μέσω του κρυφού RBF στρώματος (αυτό πρέπει να επιλέγεται όταν τα δείγματα της evaluate είναι διαφορετικά από τα δείγματα με τα οποία εκπαιδεύτηκε το μοντέλο, για αυτό η default τιμή της αντίστοιχης παραμέτρου είναι αληθής) αποθηκεύοντας το στρώμα εισόδου του μοντέλου και κάνοντας πρώτα forward pass με τα δείγματα δια μέσω του στρώματος εισόδου και μετά με την έξοδο του στρώματος εισόδου δια μέσω του κρυφού RBF στρώματος.

Παρακάτω γίνεται forward pass με την έξοδο του κρυφού RBF στρώματος διά του στρώματος εξόδου και μετά με την έξοδο του στρώματος εξόδου διά της συνάρτησης ενεργοποίησης-απώλειας του στρώματος εξόδου το οποίο επιστρέφει και την απώλεια του μοντέλου. Μετά υπολογίζεται και αποθηκεύεται η ακρίβεια από την αντίστοιχη συνάρτηση που αναλύθηκε νωρίτερα με βάση την έξοδο της συνάρτησης ενεργοποίησης-απώλειας του στρώματος εξόδου και των δοσμένων ετικέτων, αν επιλέχθηκε εμφανίζονται η απώλεια και η ακρίβεια και επιστρέφονται.

Τέλος, η μέθοδος predict δέχεται ως είσοδο ένα δείγμα σε μορφή ενός μονοδιάστατου πίνακα ή περισσότερα δείγματα σε μορφή ενός δισδιάστατου πίνακα και επιστρέφει την έξοδο του μοντέλου για το δείγμα ή δείγματα, δηλαδή τη πιθανότητα της κάθε ετικέτας για κάθε δείγμα. Πρώτα αποθηκεύονται το στρώμα εισόδου, το κρυφό RBF στρώμα και το στρώμα εξόδου του μοντέλου. Στη συνέχεια γίνεται forward pass είτε με το δείγμα τοποθετημένο στη πρώτη θέση ενός δισδιάστατου πίνακα αν είναι αποθηκευμένο σε μονοδιάστατο πίνακα ή τα δείγματα όπως είναι αν είναι ήδη αποθηκευμένα σε δισδιάστατο πίνακα δια μέσω του στρώματος εισόδου.

Μετά γίνεται forward pass της εξόδου του στρώματος εισόδου δια μέσω του κρυφού RBF στρώματος, της εξόδου του κρυφού RBF στρώματος δια μέσω του στρώματος εξόδου και της εξόδου του στρώματος εξόδου δια μέσω της συνάρτησης ενεργοποίησης-απώλειάς του. Τέλος επιστρέφεται η έξοδος της συνάρτησης ενεργοποίησης-απώλειάς του στρώματος εξόδου που είναι η έξοδος του μοντέλου για το δοσμένο δείγμα, δηλαδή οι προβλέψεις του για το ποιές είναι οι πιθανότητες να ανήκει το κάθε δείγμα σε κάθε κλάση.

2.4 Ανάλυση κώδικα κυρίου σώματος

Πρώτα αποθηκεύονται τα κανονικοποιημένα flattened δείγματα και οι ετικέτες του Mnist dataset χρησιμοποιώντας την αντίστοιχη συνάρτηση που αναλύθηκε νωρίτερα. Μετά καλείται η συνάρτηση που τρέχει το κώδικα της τρίτης εργασίας και μετράει την απόδοση των δύο κατηγοριοποιητών και του RBF νευρωνικού δικτύου.

3 Απόδοση κατηγοριοποιητών NCC και KNN

Η απόδοση των δύο κατηγοριοποιητών είναι ίδια με την απόδοση που βρήκα στην ενδιάμεση εργασία και την οποία έβαλα και στην πρώτη εργασία αφού λύνω το ίδιο πρόβλημα με την πρώτη εργασία, αλλά εδώ δεν μετράω τον χρόνο εκπαίδευσης και κατηγοριοποίησης των κατηγοριοποιητών γιατί δεν έχει νόημα αυτό.

3.1 Απόδοση κατηγοριοποιητή πλησιέστερου κέντρου κλάσης

```
NCC calculations started...
-Train accuracy: 0.8079833333333334 (80.8%).
-Test accuracy: 0.8203 (82.03%).
```

Σχήμα 1: Αποτελέσματα προγράμματος για ncc

Όπως φαίνεται στο παραπάνω screenshot, για την κατηγοριοποίηση των ψηφίων του Mnist dataset ο κατηγοριοποιητής πλησιέστερου κέντρου κλάσης δίνει ακρίβεια 80.8% για τα δεδομένα εκπαίδευσης και 82.03% για τα δεδομένα ελέγχου.

3.2 Απόδοση κατηγοριοποιητή πλησιέστερου γείτονα

- Απόδοση για k=1:

```
KNN calculations for k=1 started...
-Train accuracy: 1.0 (100.0%).
-Test accuracy: 0.9691 (96.91%).
```

Σχήμα 2: Αποτελέσματα προγράμματος για knn (k=1)

Όπως φαίνεται στο παραπάνω screenshot, για την κατηγοριοποίηση των ψηφίων του Mnist dataset ο κατηγοριοποιητής πλησιέστερου γείτονα με k=1 γείτονα δίνει ακρίβεια 100% για τα δεδομένα εκπαίδευσης και 96.91% για τα δεδομένα ελέγχου.

- Απόδοση για k=3:

  Όπως φαίνεται στο παρακάτω screenshot, για την κατηγοριοποίηση των ψηφίων του Mnist dataset ο κατηγοριοποιητής πλησιέστερου γείτονα με k=3 γείτονες δίνει ακρίβεια 98.67% για τα δεδομένα εκπαίδευσης και 97.05% για τα δεδομένα ελέγχου.

```
KNN calculations for k=3 started...
-Train accuracy: 0.9867166666666667 (98.67%).
-Test accuracy: 0.9705 (97.05%).
```

Σχήμα 3: Αποτελέσματα προγράμματος για knn (k=3)

3.3 Συμπεράσματα απόδοσης κατηγοριοποιητών

3.3.1 Συγκεντρωτικός πίνακας αποδόσεων κατηγοριοποιητών

	NCC	**KNN (K=1)**	**KNN (K=3)**
Train acc	80.80%	100.00%	98.67%
Test acc	82.03%	96.91%	97.05%

3.3.2 Συμπεράσματα απόδοσης κατηγοριοποιητή πλησιέστερου κέντρου κλάσης

Ο κατηγοριοποιητής πλησιέστερου κέντρου κλάσης δίνει πολύ χαμηλότερη απόδοση από τους κατηγοριοποιητές πλησιέστερου γείτονα για 1 και 3 γείτονες και για τα δεδομένα εκπαίδευσης και για τα δεδομένα ελέγχου.

3.3.3 Συμπεράσματα απόδοσης κατηγοριοποιητή πλησιέστερου γείτονα για k=1

Ο κατηγοριοποιητής πλησιέστερου γείτονα για k=1 γείτονα δίνει 100% ακρίβεια για τα δεδομένα εκπαίδευσης, το οποίο είναι αναμενόμενο αφού λαμβάνει υπόψη τη τιμή του ενός πλησιέστερου γείτονα και εφόσον τα δεδομένα εκπαίδευσης έχουν ήδη αποθηκευτεί από τον κατηγοριοποιητή υπάρχει ήδη η ετικέτα για κάθε είσοδο και λαμβάνεται μόνο αυτή υπόψη. Επίσης δίνει πολύ υψηλότερη ακρίβεια ελέγχου από τον κατηγοριοποιητή πλησιέστερου κέντρου κλάσης αλλά ελάχιστα χειρότερη από τον κατηγοριοποιητή πλησιέστερου γείτονα για k=3 γείτονες.

3.3.4 Συμπεράσματα απόδοσης κατηγοριοποιητή πλησιέστερου γείτονα για k=3

Ο κατηγοριοποιητής πλησιέστερου γείτονα για k=3 γείτονες δίνει την υψηλότερη ακρίβεια από τους τρεις κατηγοριοποιητές για τα δεδομένα δεδομένα ελέγχου αλλά λίγο χαμηλότερη ακρίβεια για τα δεδομένα εκπαίδευσης από τον αντίστοιχο κατηγοριοποιητή πλησιέστερου γείτονα για k=1 γείτονα.

4 Απόδοση Radial Basis Function (RBF) νευρωνικού δικτύου

4.1 Σημειώσεις

Σε αυτό το κομμάτι της έκθεσης θα χρησιμοποιήσω RBF νευρωνικό δίκτυο που χρησιμοποιεί τις κλάσεις του προγράμματος. Στα κομμάτια όπου δεν αναφέρονται τα χαρακτηριστικά και οι παράμετροι του RBF νευρωνικού δικτύου, το RBF νευρωνικό δίκτυο έχει την εξής μορφή:

Αποτελείται από τρία στρώματα:

- Το στρώμα εισόδου που δέχεται 784 εισόδους (μία για κάθε χαρακτηριστικό των δειγμάτων του Mnist dataset) και έχει 784 νευρώνες (πάλι έναν για κάθε χαρακτηριστικό των δειγμάτων του Mnist dataset).
- Το κρυμμένο RBF στρώμα που δέχεται 784 εισόδους (μία για κάθε νευρώνα του στρώματος εισόδου) και έχει 1568 νευρώνες με RBF συνάρτηση την Gauss που αναφέρηκε στη παράγραφο 2.3.3.
- Το στρώμα εξόδου που δέχεται 1568 εισόδους (μία για κάθε νευρώνα του κρυμμένου RBF στρώματος) και έχει 10 νευρώνες με συνάρτηση ενεργοποίησης την SoftMax (ένα νευρώνα ανά κλάση του Mnist dataset, δηλάδη ανά ψηφίο).
- Η συνάρτηση απώλειας είναι η Categorical Cross Entropy και υπολογίζεται μαζί με την SoftMax σε μία κλάση που τις ενώνει όπως αναφέρθηκε στο αντίστοιχο κομμάτι της έκθεσης όπου αναλύεται ο κώδικας.

Για να αλλάξετε τη μορφή του νευρωνικού δικτύου μπορείτε να αλλάξετε την όγδοη και δέκατη γραμμή της συνάρτησης custom_rbf_neural_network, αλλάζοντας τα αντίστοιχα ορίσματα (το στρώμα εισόδου έχει πάντα όσες εισόδους και νευρώνες είναι και τα χαρακτηριστικά των δειγμάτων εκπαίδευσης):

```
8: custom_rbf_network.add_layer(784, 1568, "RBF", init="random")
10: custom_rbf_network.add_layer(1568, 10, "Output")
```

Ο optimizer είναι το Stochastic Gradient Descent και οι παράμετροί του είναι οι εξής:

- Learning rate = 9.0
- Momentum = 0.9
- Decay = 0.0001

Διάλεξα τις παραπάνω παραμέτρους διότι δουλεύουν καλά για το στρώμα εξόδου του RBF νευρωνικού δικτύου του προγράμματός μου και με το συγκεκριμένο dataset. Για να αλλάξετε τις παραμέτρους αυτές μπορείτε να αλλάξετε την δεύτερη γραμμή της συνάρτησης custom_rbf_neural_network, αλλάζοντας τα αντίστοιχα ορίσματα:

```
custom_rbf_network = RBFNetwork(learning_rate=9.0, momentum=0.9,
        decay=0.0001)
```

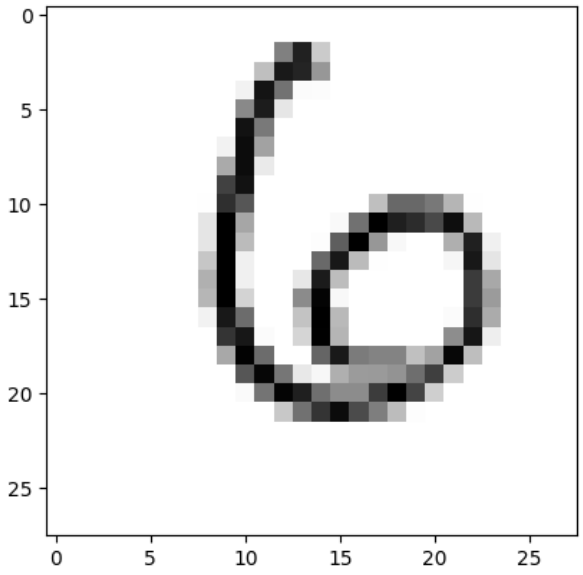
Επιπλέον, χρησιμοποιώ 50 εποχές και batch size 128. Τον αριθμό εποχών αυτό τον χρησιμοποιώ για να φαίνονται οι επιπτώσεις που έχουν οι αλλαγές των διαφόρων παραμέτρων. Το batch size αυτό το χρησιμοποιώ γιατί μου έδινε τα καλύτερα αποτελέσματα. Για να αλλάξετε τις παραμέτρους αυτές μπορείτε να αλλάξετε την δέκατη τέταρτη γραμμή της συνάρτησης custom_rbf_neural_network, αλλάζοντας τα αντίστοιχα ορίσματα:

```
training_time = custom_rbf_network.fit(x_train, y_train, epochs=50,
epoch_print_rate=1, batch_size=128)
```

Τέλος, επειδή στο τέλος κάθε εποχής όταν εμφανίζεται η πρόοδος αξιολογούνται όλα τα δείγματα για τον υπολογισμό της απώλειας και της ακρίβειας, προστίθεται αρκετός χρόνος κατά την εκπαίδευση. Οταν υπολογίζεται ο συνολικός χρόνος τις εκπαίδευσης του νευρωνικού δικτύου, δεν μετράται σε αυτόν και ο χρόνος για την αξιολόγηση όλων των δειγμάτων. Οπότε μπορεί για παράδειγμα η εκπαίδευση να πάρει πραγματικά 60 δευτερόλεπτα για να τελειώσει αν εμφανίζει την πρόοδο σε κάθε εποχή, αλλά να λέει πως πήρε συνολικά 40 δευτερόλεπτα. Αυτό συμβαίνει επειδή πήρε 20 δευτερόλεπτα η αξιολόγηση σε κάθε εποχή των δεδομένων και η εμφάνιση των αποτελεσμάτων. Επίσης αυτό σημαίνει πως αν βάλετε μεγαλύτερο epoch_print_rate ώστε να εμφανίζεται σπανιότερα η πρόοδος κατά την εκπαίδευση, θα πάρει λιγότερο πραγματικό χρόνο η εκπαίδευση αλλά δεν θα αλλάξει ο χρόνος που λέει πως διήρκησε στο τέλος.

4.2 Χαρακτηριστικά παραδείγματα ορθής κατηγοριοποίησης

4.2.1 Πρώτο παράδειγμα



Σχήμα 4: Απεικόνιση 101ου δείγματος ελέγχου

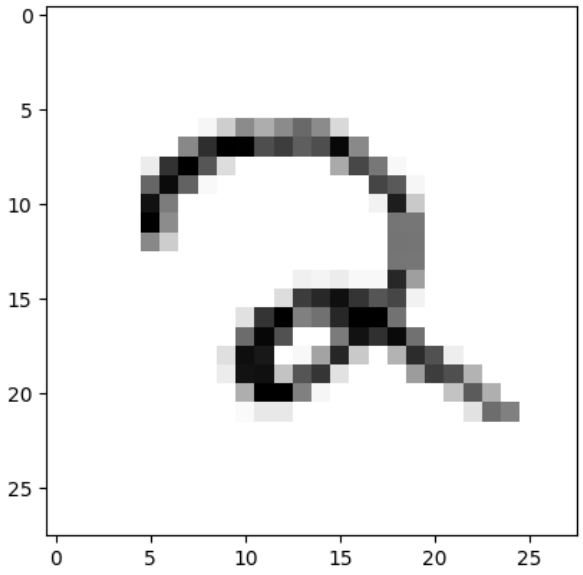
Για το 101ο δείγμα ελέγχου (αρχίζοντας από το ένα) που απεικονίζεται παραπάνω, το νευρωνικό δίκτυό μου προβλέπει σωστά την ετικέτα 6, συγκεκριμένα η έξοδός του είναι η ακόλουθη:

```
Sample 101: prediction is 6, label is 6.
Here is the model's output for sample 101:
[[4.99802394e-05 1.91394460e-05 1.55785657e-04 2.06979605e-06
  4.90580040e-06 2.05333579e-05 9.99670307e-01 8.42504252e-06
  6.83582752e-05 4.95802036e-07]]
```

Σχήμα 5: Έξοδος μοντέλου για το 101ο δείγμα ελέγχου

Όπως βλέπουμε δίνει 99.9670307% πιθανότητα να είναι 6 το δείγμα που είναι και η μεγαλύτερη από τις πιθανότητες.

4.2.2 Δεύτερο παράδειγμα



Σχήμα 6: Απεικόνιση 304ου δείγματος ελέγχου

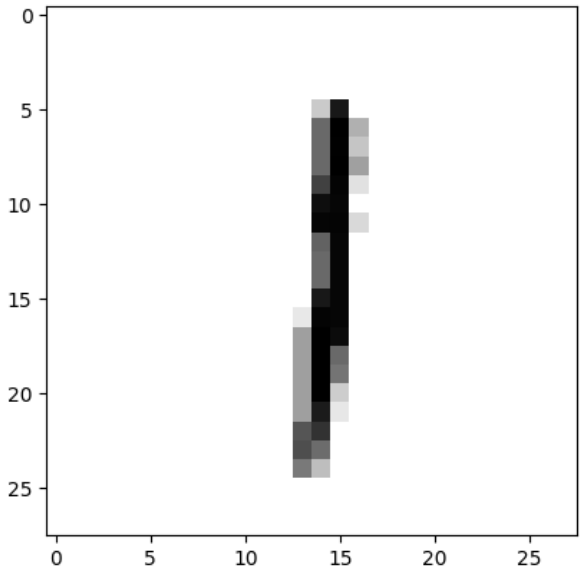
Για το 304ο δείγμα ελέγχου (αρχίζοντας από το ένα) που απεικονίζεται παραπάνω, το νευρωνικό δίκτυό μου προβλέπει σωστά την ετικέτα 2, συγκεκριμένα η έξοδός του είναι η ακόλουθη:

```
Sample 304: prediction is 2, label is 2.
Here is the model's output for sample 304:
[[8.12741075e-05 5.95811271e-07 9.92717474e-01 8.96452736e-05
  9.60116700e-06 1.01042267e-05 2.44047402e-04 5.07594840e-03
  1.70516272e-03 6.61465264e-05]]
```

Σχήμα 7: Έξοδος μοντέλου για το 304ο δείγμα ελέγχου

Όπως βλέπουμε δίνει 99.2717474% πιθανότητα να είναι 2 το δείγμα που είναι και η μεγαλύτερη από τις πιθανότητες.

4.2.3 Τρίτο παράδειγμα



Σχήμα 8: Απεικόνιση 505ου δείγματος ελέγχου

Για το 505ο δείγμα ελέγχου (αρχίζοντας από το ένα) που απεικονίζεται παραπάνω, το νευρωνικό δίκτυό μου προβλέπει σωστά την ετικέτα 1, συγκεκριμένα η έξοδός του είναι η ακόλουθη:

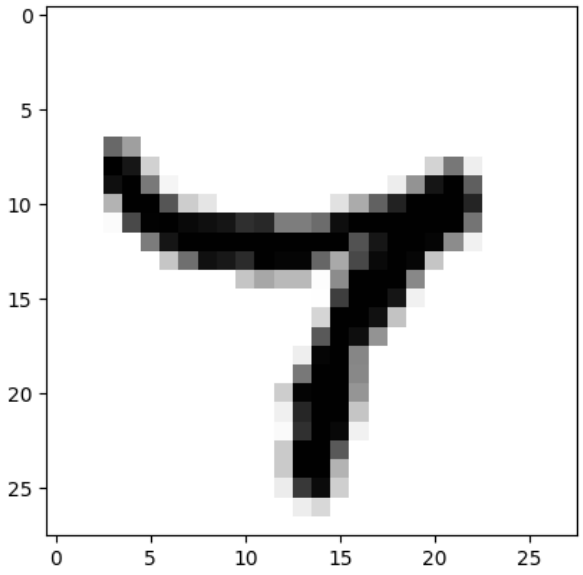
```
Sample 505: prediction is 1, label is 1.
Here is the model's output for sample 505:
[[3.21153171e-19 9.99889212e-01 8.50103617e-07 4.93121788e-08
  4.56628869e-05 8.45973560e-07 1.69002689e-07 4.07149815e-05
  3.98585660e-07 2.20973331e-05]]
```

Σχήμα 9: Έξοδος μοντέλου για το 505ο δείγμα ελέγχου

Όπως βλέπουμε δίνει 99.9889212% πιθανότητα να είναι 1 το δείγμα που είναι και η μεγαλύτερη από τις πιθανότητες.

4.3 Χαρακτηριστικά παραδείγματα εσφαλμένης κατηγοριοποίησης

4.3.1 Πρώτο παράδειγμα



Σχήμα 10: Απεικόνιση 125ου δείγματος ελέγχου

Για το 125ο δείγμα ελέγχου (αρχίζοντας από το ένα) που απεικονίζεται παραπάνω, το νευρωνικό δίκτυό μου προβλέπει εσφαλμένα την ετικέτα 4 ενώ η σωστή ετικέτα είναι το 7. Όπως βλέπουμε στην απεικόνιση του δείγματος, είναι δύσκολο ακόμη και για εμάς να αποφασίσουμε αν είναι 4 ή 7, συγκεκριμένα η έξοδός του είναι η ακόλουθη:

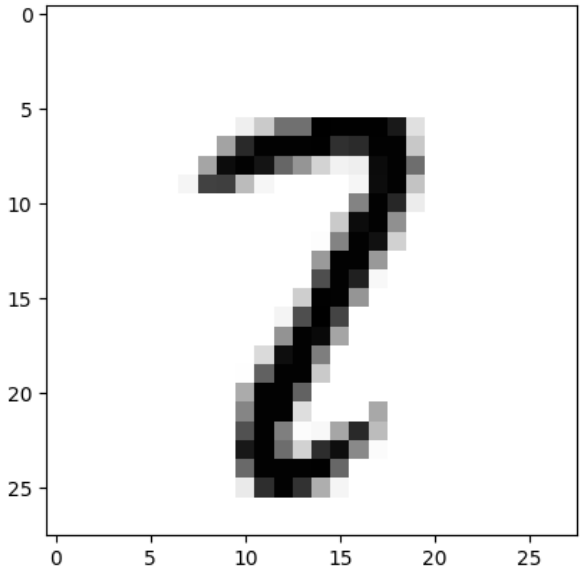
```
Sample 125: prediction is 4, label is 7.
Here is the model's output for sample 125:
[[4.68830905e-05 1.35963971e-04 6.03010027e-04 9.93843309e-05
  5.20778155e-01 3.98501266e-04 4.83967131e-05 4.19113357e-01
  4.08021153e-03 5.46961371e-02]]
```

Σχήμα 11: Έξοδος μοντέλου για το 125ο δείγμα ελέγχου

Όπως βλέπουμε στην έξοδο, η πιθανότητες να είναι 4 ή 7 είναι οι δύο μεγαλύτε-

ρες, με την πιθανότητα να είναι 4 να είναι ίση με 52.0778155% και την πιθανότητα να είναι 7 να είναι ίση με 41.9113357%.

4.3.2 Δεύτερο παράδειγμα



Σχήμα 12: Απεικόνιση 322ου δείγματος ελέγχου

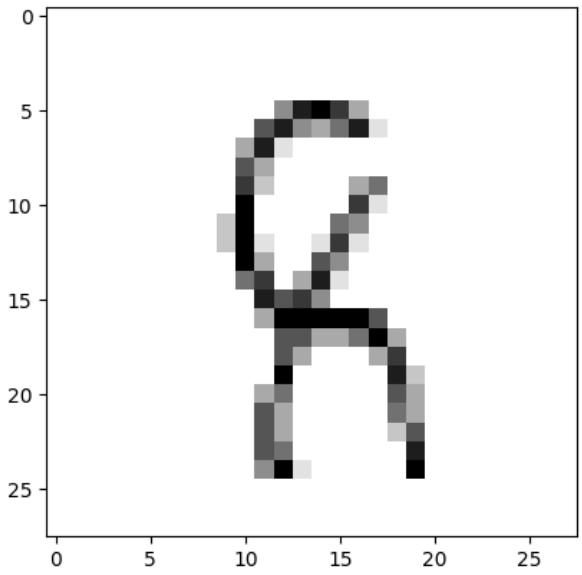
Για το 322ο δείγμα ελέγχου (αρχίζοντας από το ένα) που απεικονίζεται παραπάνω, το νευρωνικό δίκτυό μου προβλέπει εσφαλμένα την ετικέτα 7 ενώ η σωστή ετικέτα είναι το 2. Όπως βλέπουμε στην απεικόνιση του δείγματος το ψηφίο μοιάζει και με 7 και με 2 αφού η πάνω καμπύλη του δεν είναι αρκετά κυρτή για να είναι σίγουρα 2 κάνοντάς το να φαίνεται σαν 7. Συγκεκριμένα η έξοδος του δικτύου είναι η ακόλουθη:

```
Sample 322: prediction is 7, label is 2.
Here is the model's output for sample 322:
[[2.84906710e-05 1.21006620e-08 1.22023572e-01 7.11791632e-04
  2.04557863e-09 4.00945968e-05 2.59130775e-08 8.76437767e-01
  7.37225290e-04 2.10191007e-05]]
```

Σχήμα 13: Έξοδος μοντέλου για το 322ο δείγμα ελέγχου

Όπως βλέπουμε στην έξοδο, η πιθανότητες να είναι 2 ή 7 είναι οι δύο μεγαλύτερες, με την πιθανότητα να είναι 2 να είναι ίση με 12.2023572% και την πιθανότητα να είναι 7 να είναι ίση με 87.6437767%.

4.3.3 Τρίτο παράδειγμα



Σχήμα 14: Απεικόνιση 543ου δείγματος ελέγχου

Για το 543ο δείγμα ελέγχου (αρχίζοντας από το ένα) που απεικονίζεται παραπάνω, το νευρωνικό δίκτυό μου προβλέπει εσφαλμένα την ετικέτα 2 ενώ η σωστή ετικέτα είναι το 8. Όπως βλέπουμε στην απεικόνιση του δείγματος, δεν ολοκληρώνονται οι κύκλοι του 8 μπερδεύοντας το RBF νευρωνικό δίκτυο το οποίο θεωρεί πιθανό να είναι το δείγμα 2, 5 ή 8. Συγκεκριμένα η έξοδος του δικτύου είναι η

ακόλουθη:

```
Sample 543: prediction is 2, label is 8.
Here is the model's output for sample 543:
[[2.40014303e-04 9.47015647e-04 3.84617479e-01 1.93724348e-02
  2.19499404e-02 2.50692332e-01 5.38618628e-03 5.15929201e-03
  3.03629297e-01 8.00600942e-03]]
```

Σχήμα 15: Έξοδος μοντέλου για το 543ο δείγμα ελέγχου

Όπως βλέπουμε στην έξοδο, η πιθανότητες να είναι 2, 5 ή 8 είναι οι τρεις μεγαλύτερες, με την πιθανότητα να είναι 2 να είναι ίση με 38.4617479%, την πιθανότητα να είναι 5 να είναι ίση με 25.0692332% και την πιθανότητα να είναι 8 να είναι ίση με 30.3629297%.

4.4 Ποσοστά επιτυχίας στα στάδια εκπαίδευσης και ελέγχου

```
Time elapsed for training: 44.42 seconds.
Train data results:
loss: 0.0741 - accuracy: 0.9762
Test data results:
loss: 0.0969 - accuracy: 0.9687
```

Σχήμα 16: Ποσοστά επιτυχίας στα στάδια εκπαίδευσης και ελέγχου

Όπως φαίνεται στο παραπάνω screenshot, παρατηρούμε μικρή αύξηση της απώλειας και μικρή μείωση της ακρίβειας πηγαίνοντας από τα δείγματα εκπαίδευσης στα δείγματα ελέγχου. Αυτό είναι λογικό καθώς τα δείγματα ελέγχου δεν έχουν χρησιμοποιηθεί για την εκπαίδευση του μοντέλου.

4.5 Χρόνος εκπαίδευσης και ποσοστά επιτυχίας για διαφορετικούς αριθμούς νευρώνων στο κρυφό RBF στρώμα

4.5.1 Σημειώσεις

Η σύγκριση αυτών των αποτελεσμάτων θα γίνει με τα αποτελέσμα της παραγράφου 4.4 και χρησιμοποιώντας τις πραμέτρους που αναφέρονται στη παράγραφο 4.1 εκτός

και αν γράφω πως χρησιμοποιούνται άλλες τιμές για συγκεκριμένες παραμέτρους.

4.5.2 Με κρυφό RBF στρώμα 784 νευρώνων

```
Time elapsed for training: 19.44 seconds.
Train data results:
loss: 0.1181 - accuracy: 0.9627
Test data results:
loss: 0.1286 - accuracy: 0.9600
```

Σχήμα 17: Ποσοστά επιτυχίας στα στάδια εκπαίδευσης και ελέγχου με κρυφό RBFστρώμα 784 νευρώνων

Με κρυφό RBF στρώμα 784 νευρώνων και με αυξημένο decay=0.001 (αντί για 0.0001 επειδή με τόσο χαμηλό decay ταλαντώνεται η ακρίβεια στις αρχικές εποχές με 784 RBF νευρώνες επειδή είναι πολύ υψηλό το learning rate), ο χρόνος εκπαίδευσής του δικτύου είναι 19.44 δευτερόλεπτα. Δηλαδή με χρήση των μισών RBF νευρώνων από τη παράγραφο 4.4, ο χρόνος εκπαίδευσης υποδιπλασιάζεται. Ωστόσο, και η απώλεια και η ακρίβεια και για τα δεδομένα εκπαίδευσης και για τα δεδομένα ελέγχου μειώνεται αν και όχι πάρα πολύ. Αλλά παρατήρησα πως με πολλές εποχές, δηλαδή 250 και παραπάνω, η ακρίβεια δεν μπορεί να φτάσει πάνω από 0.9700 ούτε καν για τα δεδομένα εκπαίδευσης, τουλάχιστον με τις παραμέτρους που χρησιμοποίησα.

4.5.3 Με κρυφό RBF στρώμα 1176 νευρώνων

```
Time elapsed for training: 36.14 seconds.
Train data results:
loss: 0.1037 - accuracy: 0.9679
Test data results:
loss: 0.1257 - accuracy: 0.9644
```

Σχήμα 18: Ποσοστά επιτυχίας στα στάδια εκπαίδευσης και ελέγχου με κρυφό RBF στρώμα 1176 νευρώνων

Με κρυφό RBF στρώμα 1176 νευρώνων, ο χρόνος εκπαίδευσής του δικτύου είναι 36.14 δευτερόλεπτα. Δηλαδή με 0.75 φορές τον αριθμό των RBF νευρώνων από τη παράγραφο 4.4, ο χρόνος εκπαίδευσης πολλαπλασιάζεται περίπου επί 0.75 επίσης. Ωστόσο, και η απώλεια και η ακρίβεια και για τα δεδομένα εκπαίδευσης και για τα δεδομένα ελέγχου μειώνονται, βέβαια αυτό επηρεάζεται και από την τυχαία αρχικοποίηση των κέντρων του κρυφού στρώματος.

4.5.4 Με κρυφό RBF στρώμα 2352 νευρώνων

```
Time elapsed for training: 65.45 seconds.
Train data results:
loss: 0.1121 - accuracy: 0.9683
Test data results:
loss: 0.1239 - accuracy: 0.9643
```

Σχήμα 19: Ποσοστά επιτυχίας στα στάδια εκπαίδευσης και ελέγχου με κρυφό RBFστρώμα 2352 νευρώνων

Με κρυφό RBF στρώμα 2352 νευρώνων, ο χρόνος εκπαίδευσής του δικτύου είναι 65.45 δευτερόλεπτα. Δηλαδή με χρήση μιάμιση φορές τον αριθμό των RBF νευρώνων από τη παράγραφο 4.4, ο χρόνος εκπαίδευσης πολλαπλασιάζεται επί 1.5. Ωστόσο, και η απώλεια και η ακρίβεια και για τα δεδομένα εκπαίδευσης και για τα δεδομένα ελέγχου μειώνονται, βέβαια αυτό επηρεάζεται και από την τυχαία αρχικοποίηση των κέντρων του κρυφού στρώματος. Εδώ από ότι φαίνεται, θα μπορούσε να αυξηθεί και άλλο το learning rate ώστε να βελτιωθούν τα αποτελέσματα αλλά το κρατάω σταθερό για να είναι δίκαια η σύγκριση μεταξύ διαφορετικού αριθμού RBF νευρώνων.

4.5.5 Συμπεράσματα

Από ότι είδαμε στις προηγούμενες τρεις παραγράφους 4.5.2, 4.5.3 και 4.5.4, ο χρόνος εκπαίδευσης του RBF νευρωνικού δικτύου αυξάνεται γραμμικά όσο αυξάνεται και ο αριθμός των νευρώνων του κρυφού RBF στρώματος. Επιπλέον, μεγαλύτερος αριθμός RBF νευρώνων δίνει ελάχιστα καλύτερη ακρίβρεια και απώλεια για τον ίδιο αριθμό εποχών, αλλά σημαντικότερα όσο περισσότερους νεωρώνες RBF έχει το δίκτυο, τόσο καλύτερη απώλεια και ακρίβεια μπορεί να δώσει για πιο περίπλοκα προβλήματα. Για αυτό το λόγο, ως default αριθμό νεωρώνων του κρυφού RBF στρώματος επέλεξα 1568 νευρώνες που είναι διπλάσιος από τον αριθμό των χαρακτηριστικών των δειγμάτων του Mnist dataset και μπορεί να δώσει καλή ακρίβεια για την αναγνώριση των ψηφίων του Mnist dataset αλλά δεν αυξάνει απαγορευτικά τον χρόνο εκπαίδευσης.

4.6 Χρόνος εκπαίδευσης και ποσοστά επιτυχίας για διαφορετικούς τρόπους εκπαίδευσης

4.6.1 Αποτελέσματα για τυχαία επιλογή κέντρων

```
Time elapsed for training: 43.5 seconds.
Train data results:
loss: 0.0885 - accuracy: 0.9724
Test data results:
loss: 0.1024 - accuracy: 0.9673
```

Σχήμα 20: Ποσοστά επιτυχίας στα στάδια εκπαίδευσης και ελέγχου με τυχαία επιλογή κέντρων

Εδώ χρησιμοποιούνται ακριβώς οι ίδιες παράμετροι που αναφέρθηκαν στη παράγραφο 4.1 για αυτό και τα αποτελέσματα είναι σχεδόν ίδια με τα αποτελέσματα της παραγράφου 4.4. Με τυχαία επιλογή κέντρων, ο χρόνος εκπαίδευσης του δικτύου είναι 43.5 δευτερόλεπτα. Προφανώς αφού τα κέντρα επιλέγονται τυχαία από τα δείγματα εκπαίδευσης με sampling, κάθε εκπαίδευση του δικτύου ακόμα και με ίδιες παραμέτρους θα δίνει ελάχιστα διαφορετική απόδοση.

4.6.2 Αποτελέσματα για επιλογή κέντρων με K-means

```
Time elapsed for training: 87.1 seconds.
Train data results:
loss: 0.0750 - accuracy: 0.9765
Test data results:
loss: 0.0993 - accuracy: 0.9685
```

Σχήμα 21: Ποσοστά επιτυχίας στα στάδια εκπαίδευσης και ελέγχου με επιλογή κέντρων με K-means

Επιλέγοντας τα κέντρα των νευρώνων του κρυφού RBF στρώματος με κ-μέσους με 5 μέγιστες επαναλήψεις, ο χρόνος εκπαίδευσης του δικτύου είναι 87.1 δευτερόλεπτα, δηλαδή διπλάσιος από την τυχαία επιλογή κέντρων. Η απώλεια και

η ακρίβεια και για τα δεδομένα εκπαίδευσης και για τα δεδομένα ελέγχου αυξάνεται ελάχιστα.

4.6.3 Συμπεράσματα

Από ότι είδαμε στις προηγούμενες παραγράφους 4.6.1, και 4.6.2, ο χρόνος εκπαίδευσης του RBF νευρωνικού δικτύου αυξάνεται πάρα πολύ χρησιμοποιώντας κ-μέσους αντί για τυχαία επιλογή κέντρων χωρίς να βελτιώνεται ιδιαίτερα η απόδοση του δικτύου. Αυτή η αύξηση του χρόνου εκπαίδευσης είναι με μόνο 5 επαναλήψεις των κ-μέσων, οπότε αν επιτρέπονταν ακόμα περισσότερες επαναλήψεις, ο χρόνος εκπαίδευσης θα αυξανόταν ακόμα περισσότερο. Ίσως αν το πρόβλημα που θέλαμε να λύσουμε ήταν πιο περίπλοκο να βελτίωνε περισσότερο την απόδοση η χρήση κ-μέσων αντί για τυχαία επιλογή κέντρων, αλλά για την αναγνώριση ψηφίων του Mnist dataset δεν αξίζει η τεράστια αύξηση του χρόνου εκπαίδευσης για την ελάχιστη βελτίωση της απόδοσης, για αυτό και ως default τρόπο εκπαίδευσης έχω θέσει την τυχαία επιλογή κέντρων.

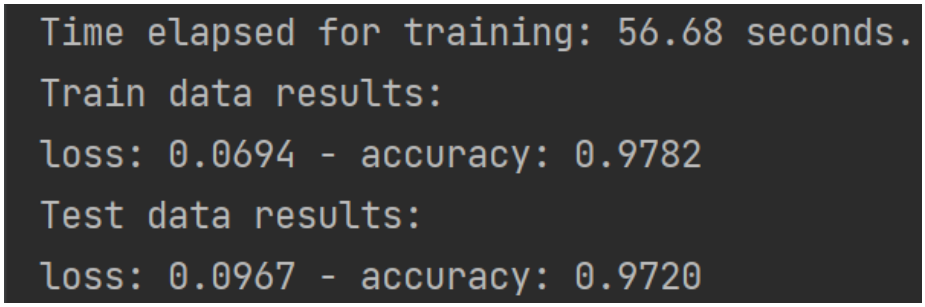
4.7 Χρόνος εκπαίδευσης και ποσοστά επιτυχίας για διαφορετικές τιμές των παραμέτρων εκπαίδευσης

4.7.1 Σημειώσεις

Οι παράμετροι εκπαίδευσης για τις οποίες θα ελεγχθούν διαφορετικές τιμές είναι το batch size που χρησιμοποείται για την εκπαίδευση του στρώματος εξόδου του RBF νευρωνικού δικτύου και οι παράμετροι του optimizer του στρώματος εξόδου του RBF νευρωνικού δικτύου. Ο optimizer αυτός είναι το Stochastic Gradient Descent. Ο χρόνος εκπαίδευσης θα αναφέρεται στα αποτελέσματα μόνο για το batch size αφού αλλάζοντας τις παραμέτρους του optimizer δεν επηρεάζεται ο χρόνος εκπαίδευσης.

Δεν θα δοκιμαστούν διαφορετικές τιμές μέγιστου αριθμού εποχών αφού είναι γνωστό πως όσο αυξάνονται οι εποχές βελτιώνεται η απώλεια και η ακρίβεια του RBF νευρωνικού δικτύου και για τα δεδομένα εκπαίδευσης και για τα δεδομένα ελέγχου μέχρι το σημείο που αρχίζει το overfitting και ενώ βελτιώνεται η απόδοση για τα δεδομένα εκπαίδευσης, χειροτερεύει για τα δεδομένα ελέγχου. Τα αποτελέσματα, δηλαδή η απώλεια, η ακρίβεια και ο χρόνος εκπαίδευσης που δίνει το κάθε μοντέλο θα συγκρίνονται με τα αποτελέσματα της παραγράφου 4.4 και φυσικά μεταξύ των μοντέλων με διαφορετικές τιμές στις ίδιες παραμέτρους τους. Οπότε όταν δεν αναφέρεται από ποιό μοντέλο είναι οι τιμές σύγκρισης, εννοείται το μοντέλο της παραγράφου 4.4.

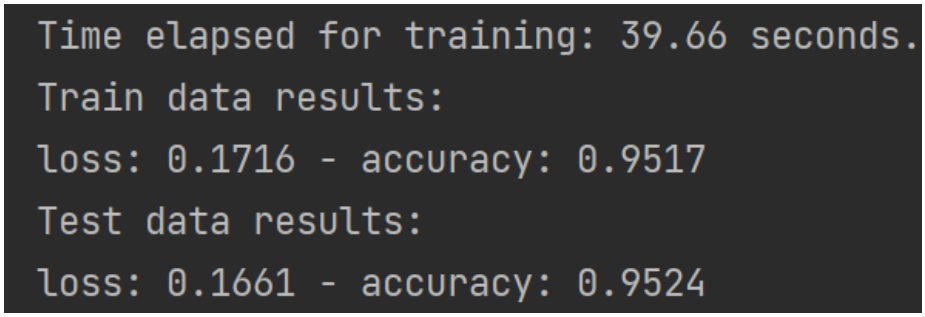
4.7.2 Αποτελέσματα για batch size=32



Σχήμα 22: Ποσοστά επιτυχίας στα στάδια εκπαίδευσης και ελέγχου με batch size=32

Με batch size 32 ο χρόνος εκπαίδευσής του δικτύου είναι 56.8 δευτερόλεπτα, όπως βλέπουμε στο παραπάνω screenshot. Με πολύ μικρό batch size αυξάνεται έντονα ο χρόνος εκπαίδευσης και η απώλεια και η ακρίβεια βελτιώνονται ελάχιστα και για τα δεδομένα εκπαίδευσης και για τα δεδομένα ελέγχου.

4.7.3 Αποτελέσματα για batch size=4096

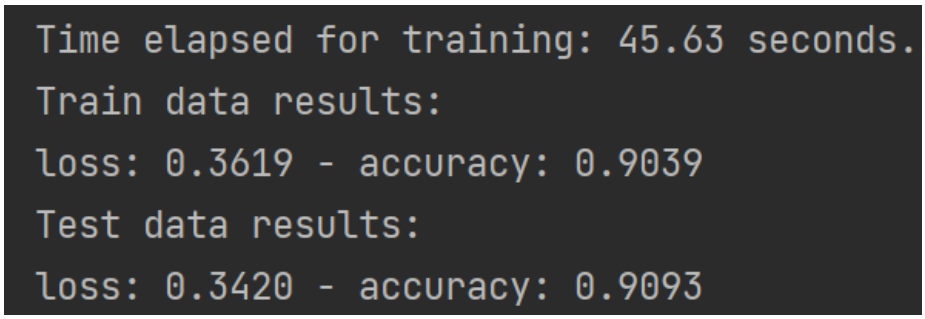


Σχήμα 23: Ποσοστά επιτυχίας στα στάδια εκπαίδευσης και ελέγχου με batch size=4096

Με batch size 4096 ο χρόνος εκπαίδευσής του δικτύου είναι 39.66 δευτερόλεπτα, όπως βλέπουμε στο παραπάνω screenshot. Με μεγαλύτερο batch size δεν αλλάζει ο χρόνος εκπαίδευσης (εδώ μειώθηκε ελάχιστα) και η απώλεια και η ακρίβεια χειροτερεύουν και για τα δεδομένα εκπαίδευσης και για τα δεδομένα ελέγχου.

4.7.4 Αποτελέσματα για batch size=60000

Με batch size 60000 ο χρόνος εκπαίδευσής του δικτύου είναι 45.63 δευτερόλεπτα, όπως βλέπουμε στο παραπάνω screenshot. Με ουσιαστικά ένα batch, δηλαδή



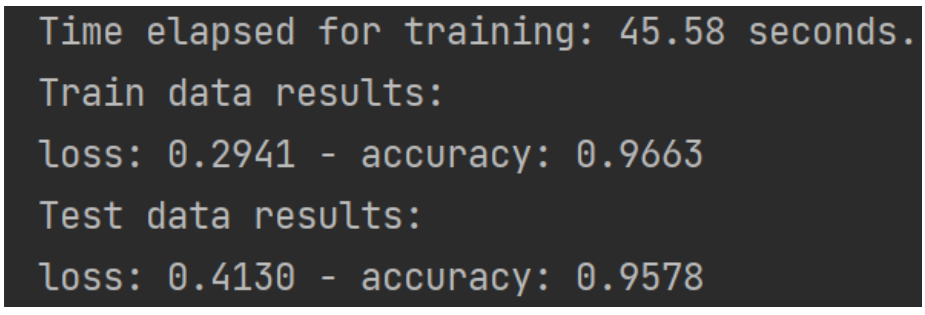
Σχήμα 24: Ποσοστά επιτυχίας στα στάδια εκπαίδευσης και ελέγχου με batch size=60000

εκπαιδεύοντας το μοντέλο με όλα τα δείγματα με ένα πέρασμα σε κάθε εποχή, αυξάνεται ελάχιστα ο χρόνος εκπαίδευσης και η απώλεια και η ακρίβεια χειροτερεύουν πάρα πολύ και για τα δεδομένα εκπαίδευσης και για τα δεδομένα ελέγχου.

Όπως φαίνεται το batch size 128 που χρησιμοποιείται ως default batch size είναι μια καλή τιμή για το batch size σε σχέση με το 32 και το 4096 αφού έχει σχεδόν ίδια ακρίβεια με το batch size=32 με αρκετά λιγότερο χρόνο εκπαίδευσης και καλύτερη ακρίβεια από το batch size=4096 για περίπου τον ίδιο χρόνο εκπαίδευσης. Το batch size=60000 δίνει πολύ κακά αποτελέσματα και δεν αποτελεί επιλογή.

Σημειώνεται πως το batch size δεν έχει τόσο μεγάλη επίδραση στον χρόνο εκπαίδευσης του RBF νευρωνικού δικτύου όσο έχει στον χρόνο εκπαίδευσης ενός MLP νευρωνικού δικτύου καθώς στον χρόνο εκπαίδευσης του RBF νευρωνικού δικτύου συμπεριλαμβάνεται και ο χρόνος εκπαίδευσης του κρυφού RBF στρώματος που ανάλογα με το πλήθος των νευρώνων του μπορεί να αποτελεί μεγάλο μέρος του συνολικού χρόνου εκπαίδευσης.

4.7.5 Αποτελέσματα για learning rate=150.0



Σχήμα 25: Ποσοστά επιτυχίας στα στάδια εκπαίδευσης και ελέγχου με learning rate=150.0

Με μεγαλύτερο learning rate 150.0 το μοντέλο ταλαντώνεται στην περιοχή της

0.9600 με 0.9700 ακρίβειας επειδή το learning rate είναι πολύ μεγάλο. Επομένως η απώλεια και η ακρίβεια χειροτερεύουν αρκετά και για τα δεδομένα εκπαίδευσης και για τα δεδομένα ελέγχου αλλά ακόμα πιο σημαντικά η περαιτέρω εκπαίδευση με περισσότερες εποχές δεν βελτιώνει την απόδοση.

4.7.6 Αποτελέσματα για learning rate=0.1

```
Time elapsed for training: 46.13 seconds.
Train data results:
loss: 0.2844 - accuracy: 0.9228
Test data results:
loss: 0.2731 - accuracy: 0.9272
```

Σχήμα 26: Ποσοστά επιτυχίας στα στάδια εκπαίδευσης και ελέγχου με learning rate=0.1

Με πολύ μικρότερο learning rate 0.1 η απώλεια και η ακρίβεια και στα δεδομένα εκπαίδευσης και στα δεδομένα ελέγχου είναι αρκετά χειρότερες από τις αντίστοιχες τιμές του μοντέλου της παραγράφου 4.4. Επομένως το μικρότερο learning rate λειτουργεί κανονικά αλλά θα μπορούσε να αυξηθεί για να γίνει γρηγορότερα η εκπαίδευση.

Επομένως τελικά πρέπει να βρεθεί μια ενδιάμεση τιμή για το learning rate που να είναι όσο μεγαλύτερη γίνεται για να βελτιώνεται γρήγορα η απόδοση του μοντέλου κατά την εκπαίδευση αλλά όχι τόσο μεγάλη που θα εμφανίζεται ταλάντωση. Δοκιμάζοντας και αρκετές άλλες τιμές για το learning rate κατέληξα στη τιμή 9.0 αφού για 10.0 και παραπάνω learning rate εμφανιζόταν ταλάντωση έστω και για λίγες εποχές. Για αυτό το λόγο και η default τιμή του learning rate είναι 9.0.

4.7.7 Αποτελέσματα για momentum=0.5

```
Time elapsed for training: 46.23 seconds.
Train data results:
loss: 0.1302 - accuracy: 0.9622
Test data results:
loss: 0.1314 - accuracy: 0.9611
```

Σχήμα 27: Ποσοστά επιτυχίας στα στάδια εκπαίδευσης και ελέγχου με momentum=0.5

Δεν θα εξετάσουμε μεγαλύτερο momentum από το 0.9 γιατί ήδη είναι μεγάλο το 0.9. Με λίγο μικρότερο momentum 0.5 η απώλεια και η ακρίβεια και στα δεδομένα εκπαίδευσης και στα δεδομένα ελέγχου είναι πάλι λίγο χειρότερες από τις αντίστοιχες τιμές του μοντέλου της παραγράφου 4.4. Επομένως το λίγο μικρότερο momentum λειτουργεί κανονικά αλλά θα μπορούσε να αυξηθεί για να βελτιώνεται γρηγορότερα η απόδοση κατά την εκπαίδευση.

4.7.8 Αποτελέσματα για momentum=0.0

```
Time elapsed for training: 46.17 seconds.
Train data results:
loss: 0.1373 - accuracy: 0.9596
Test data results:
loss: 0.1365 - accuracy: 0.9599
```

Σχήμα 28: Ποσοστά επιτυχίας στα στάδια εκπαίδευσης και ελέγχου με momentum=0.0

Με μηδενικό momentum, δηλαδή χωρίς momentum, η απώλεια και η ακρίβεια και στα δεδομένα εκπαίδευσης και στα δεδομένα ελέγχου είναι πάλι λίγο χειρότερες από τις αντίστοιχες τιμές του μοντέλου της παραγράφου 4.4. Είναι επίσης χειρότερες και από όταν είχαμε momentum=0.5. Επομένως το μηδενικό momentum λειτουργεί κανονικά αλλά θα μπορούσε να ενεργοποιηθεί για να βελτιώνεται γρηγορότερα η απόδοση κατά την εκπαίδευση.

Τελικά καταλήγουμε πως η καλύτερη επιλογή τιμής για το momentum είναι το 0.9 που είναι και η default τιμή του αφού δίνει τη καλύτερη απόδοση.

4.7.9 Αποτελέσματα για decay=0.1

```
Time elapsed for training: 45.61 seconds.
Train data results:
loss: 0.3268 - accuracy: 0.9145
Test data results:
loss: 0.3059 - accuracy: 0.9216
```

Σχήμα 29: Ποσοστά επιτυχίας στα στάδια εκπαίδευσης και ελέγχου με decay=0.1

Με μεγαλύτερο decay 0.1 η απώλεια και η ακρίβεια και στα δεδομένα εκπαίδευσης και στα δεδομένα ελέγχου είναι πολύ χειρότερες από τις αντίστοιχες τιμές του μοντέλου της παραγράφου 4.4. Το μεγαλύτερο decay έχει ως αποτέλεσμα να μικραίνει πιο γρήγορα το learning rate με αποτέλεσμα να βελτιώνεται με όλο και αργότερο ρυθμό η απόδοση του μοντέλου όσο περνάνε οι εποχές.

4.7.10 Αποτελέσματα για decay=0.000001

```
Time elapsed for training: 47.25 seconds.
Train data results:
loss: 0.1045 - accuracy: 0.9681
Test data results:
loss: 0.1300 - accuracy: 0.9641
```

Σχήμα 30: Ποσοστά επιτυχίας στα στάδια εκπαίδευσης και ελέγχου με decay=0.000001

Με μικρότερο decay 0.000001 η απώλεια και η ακρίβεια και στα δεδομένα εκπαίδευσης και στα δεδομένα ελέγχου είναι ελάχιστα χειρότερες από τις αντίστοιχες τιμές του μοντέλου της παραγράφου 4.4. Επιπλέον, παρατηρείται ταλάντωση στις εποχές 27 με 35 η οποία δεν φαίνεται εδώ.

Για αυτό το λόγο η default τιμή του decay είναι 0.0001 η οποία είναι όσο το μεγαλύτερο δυνατόν γίνεται χωρίς να εμφανίζεται ταλάντωση.

4.8 Σύγκριση απόδοσης νευρωνικού δικτύου με τους κατηγοριοποιητές

4.8.1 Συγκεντρωτικός πίνακας αποδόσεων κατηγοριοποιητών

	NCC	**KNN (K=1)**	**KNN (K=3)**	**RBF Network**
Train acc	80.80%	100.00%	98.67%	97.62%
Test acc	82.03%	96.91%	97.05%	96.87%

4.8.2 Συμπεράσματα

- Τα συμπεράσματα ισχύουν για το μοντέλο που περιγράφεται στη παράγραφο 4.4 και χρησιμοποιούν τα αντίστοιχα αποτελέσματα της παραγράφου αυτής. Οπότε επηρεάζονται από την συγκεκριμένη τυχαία αρχικοποίηση των βαρών του μοντέλου.

- Το RBF νευρωνικό δίκτυο δίνει καλύτερη ακρίβεια από τον κατηγοριοποιητή nearest class centroid και για τα δεδομένα εκπαίδευσης και για τα δεδομένα ελέγχου.

- Το RBF νευρωνικό δίκτυο δίνει ελάχιστα χειρότερη ακρίβεια από τον κατηγοριοποιητή k nearest neighbors και για τα δεδομένα εκπαίδευσης και για τα δεδομένα ελέγχου. Αυτά ισχύουν και για k=1 και για k=3.

- Ωστόσο, ο κατηγοριοποιητής k nearest neighbors θέλει πολύ περισσότερο χρόνο για τη κατηγοριοποίηση δειγμάτων από το RBF νευρωνικό δίκτυο, ειδικά αν απαιτείται κατηγοριοποίηση πολλών δειγμάτων, όπως σε αυτή τη περίπτωση που τα δεδομένα ελέγχου είναι 10000 δείγματα.

- Πρέπει να σημειωθεί πως αν αυξήσουμε τον αριθμό των εποχών σε παραπάνω από 50, για παράδειγμα 250 εποχές, πολυ πιθανώς θα ξεπεράσουμε την ακρίβεια του κατηγοριοποιητή knn για τα δεδομένα ελέγχου. Για k=1 μπορούμε στη καλύτερη περίπτωση να φτάσουμε την ακρίβεια του κατηγοριοποιητή knn για τα δεδομένα εκπαίδευσης αφού ουσιαστικά αποθηκεύει τις ετικέτες που αντιστοιχούν σε κάθε δείγμα εκπαίδεσης. Για k μεγαλύτερο του 1 μπορούμε να ξεπεράσουμε την ακρίβεια του κατηγοριοποιητή knn για τα δεδομένα εκπαίδευσης. Η ακρίβεια που θα φτάσουμε και για τα δεδομένα εκπαίδευσης και για τα δεδομένα ελέγχου εξαρτάται από το πόσο χρόνο έχουμε διαθέσιμο για την εκπαίδευση του μοντέλου καθώς και πότε θα αρχίσει το overfitting των δεδομένων εκπαίδευσης.