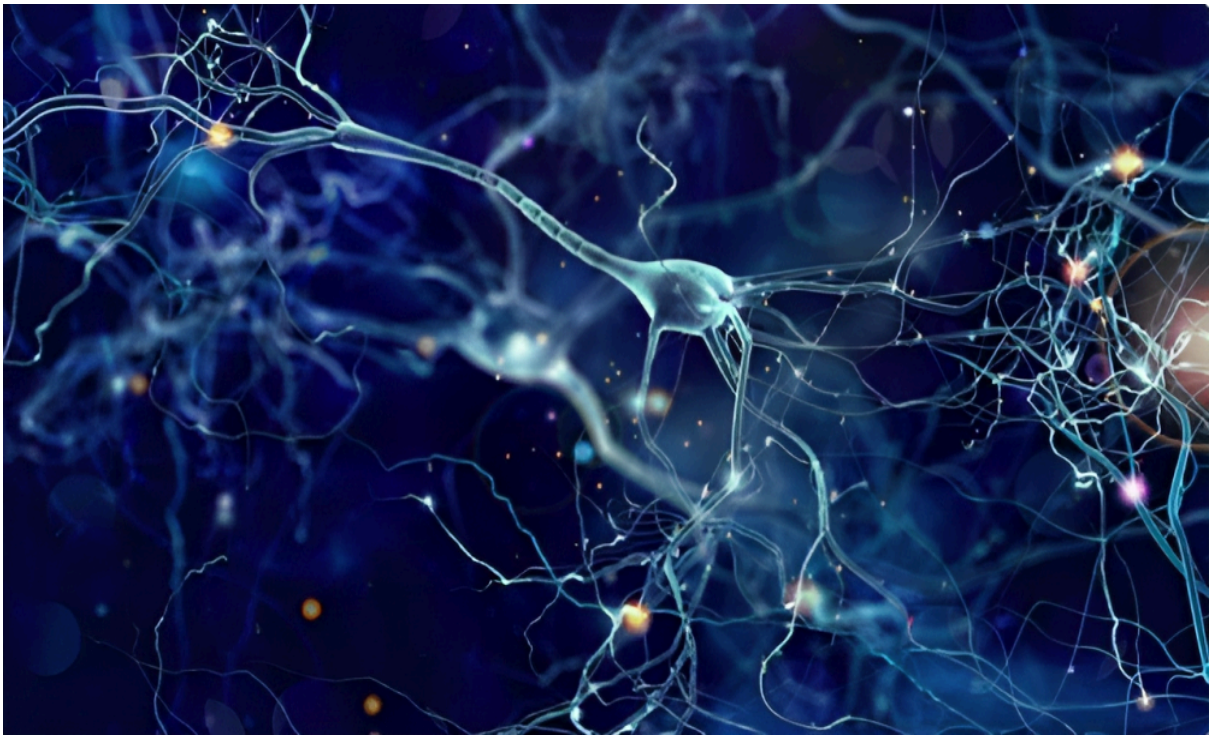
Νευρωνικά Δίκτυα - Βαθιά Μάθηση
Εργασία 3



Ονοματεπώνυμο: Μαχμουτάι Έλενα
Τμήμα: Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
ΑΕΜ: 10012

1. Εισαγωγή

Σκοπός της εργασίας είναι η υλοποίηση ενός Radial Basis Function Neural Network που θα επιλύει το πρόβλημα διαχωρισμού όλων των κλάσεων που υπάρχουν στην Cifar-10 βάση.

Για την υλοποίηση της εργασίας χρησιμοποιήθηκε η γλώσσα Python και οι βιβλιοθήκες keras, numpy, sklearn για την υλοποίηση του RBFNN και η matplotlib για την απεικόνιση των παραδειγμάτων. Επιπλέον, για τη βελτίωση της υπολογιστικής απόδοσης, χρησιμοποιείται η ανάλυση PCA για τη μείωση των διαστάσεων, βοηθώντας στην ταχύτερη εκπαίδευση και αξιολόγηση.

2. Υλοποίηση

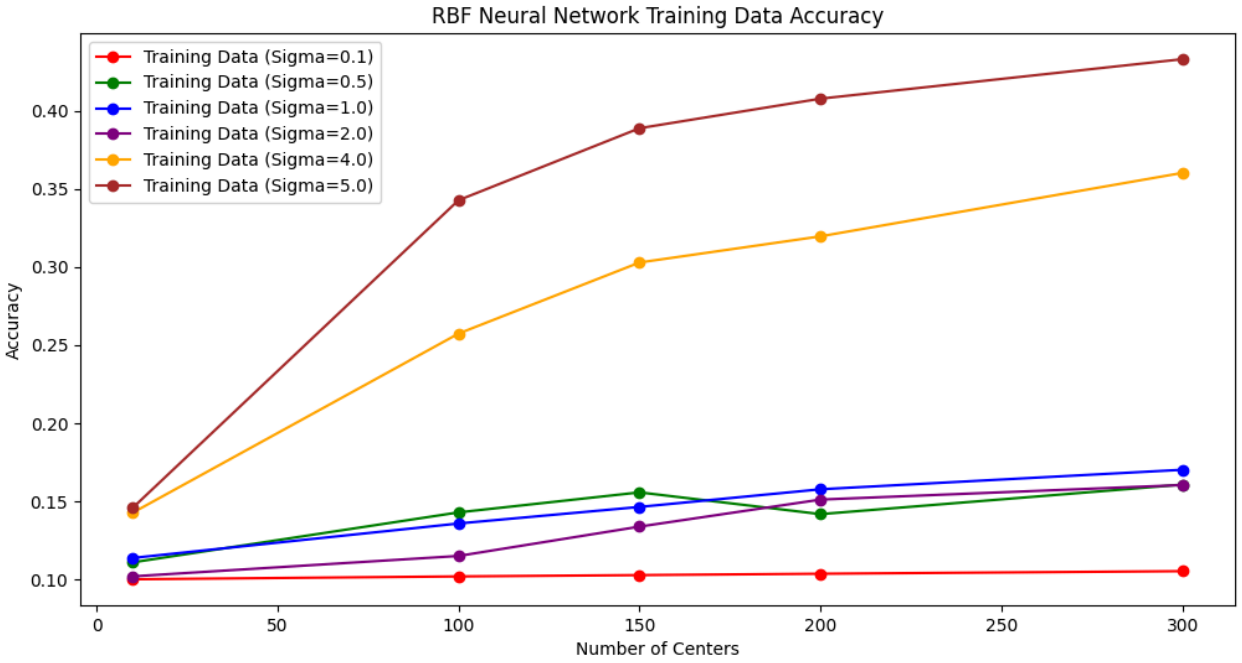
Η υλοποίηση του RBF Νευρωνικού δικτύου πραγματοποιήθηκε στο αρχείο με όνομα *radial_basis_function_nn.py* και ακολουθεί τα εξής βήματα:

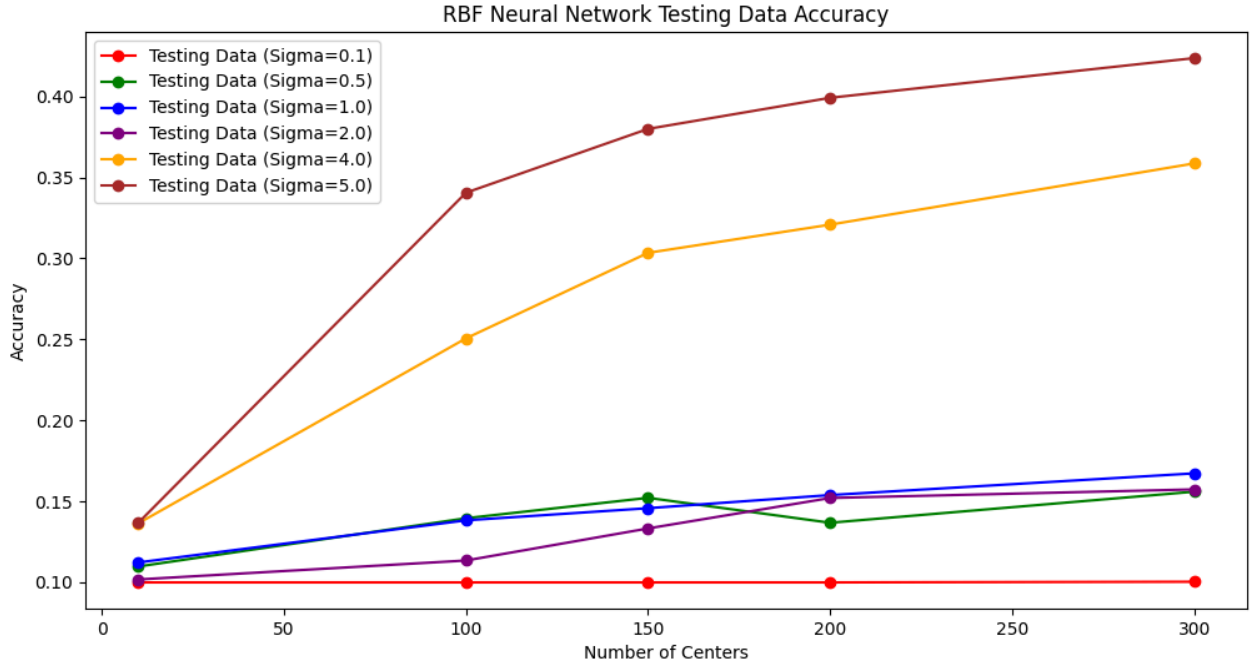
- Δημιουργία μιας *RBFNetwork* κλάσης, σχεδιασμένη για να την υλοποίηση του δικτύου. Περιέχει τις μεθόδους *__init__(self, num_centers, sigma) , _calculate_activations(self, X) , fit(self, X, y)* και *predict(self, X) .*
- Μέθοδος *__init__(self, num_centers, sigma)*: Υπεύθυνη για την αρχικοποίηση των παραμέτρων *num_centers, sigma, centers* και *weights.*
- Μέθοδος *_calculate_activations(self, X)* που υπολογίζει τις ενεργοποιήσεις RBF για κάθε σημείο δεδομένων σε σχέση με κάθε κέντρο. Η διαδικασία που ακολουθεί περιλαμβάνει τον υπολογισμό της Ευκλείδειας απόσταση μεταξύ του δείγματος και κάθε κέντρου RBF και στην συνέχεια χρησιμοποιεί τις υπολογισμένες αποστάσεις για τον υπολογισμό των τιμών ενεργοποίησης με βάση τη συνάρτηση ενεργοποίησης Gaussian RBF.
- Μέθοδος *fit(self, X, y)*: Υπεύθυνη για την εκπαίδευση του δικτύου. Επιλέγει τυχαία ένα υποσύνολο δειγμάτων εισόδου ως κέντρα RBF από το σύνολο δεδομένων X. Στη συνέχεια υπολογίζει τις ενεργοποιήσεις για όλα τα δείγματα εισόδου χρησιμοποιώντας τη μέθοδο *_calculate_activations*. Η μεταβλητή *y* μετατρέπεται σε κωδικοποίηση one-hot χρησιμοποιώντας *to_categorical*. Τέλος, υπολογίζει τα βάρη του δικτύου και τα βάρη που προκύπτουν αποθηκεύονται στο *self.weights.*
- Μέθοδος *predict(self, X):* Υπεύθυνη για τη δημιουργία προβλέψεων του μοντέλου υπολογίζοντας τις συναρτησεις activation για τα δεδομένα *X* και τα βάρη.
- Φόρτωση των δεδομένων CIFAR-10 και προετοιμασία τους (flatten) και χρήση PCA για μείωση των διαστάσεων τους.
- Βρόχος *for* για εκπαίδευση και αξιολόγηση των δεδομένων για διάφορες τιμές των *center_numbers* και *sigma_values* και τρόπους εκπαίδευσης (K-μέσους, τυχαία επιλογή κέντρων).
- Αποθήκευση των αποτελεσμάτων, συμπεριλαμβανομένου του χρόνου εκπαίδευσης και της ακρίβειας, σε αρχεία κειμένου (*rbf_results.txt* και *rbf_random_choice_results.txt*) ανάλογα με τον τρόπο εκπαίδευσης που χρησιμοποιήθηκε στην κάθε περίπτωση.

3. Αποτελέσματα

Τα δεδομένα των αρχείων αποθηκεύτηκαν στα αρχεία *rbf_results.txt* (τρόπος εκπαίδευσης K-μέσους) και *rbf_random_choice_results.txt* (τρόπος εκπαίδευσης τυχαία επιλογή κέντρων) και απεικονίστηκαν στο αρχείο *radial_basis_function_plot.py* με την συνάρτηση *plot_rbf_data(unique_sigma_values, num_centers_values, data, sigma_colors, type_of_data, y_label, title).* Τα αποτελέσματα είναι οι παρακάτω γραφικές παραστάσεις.

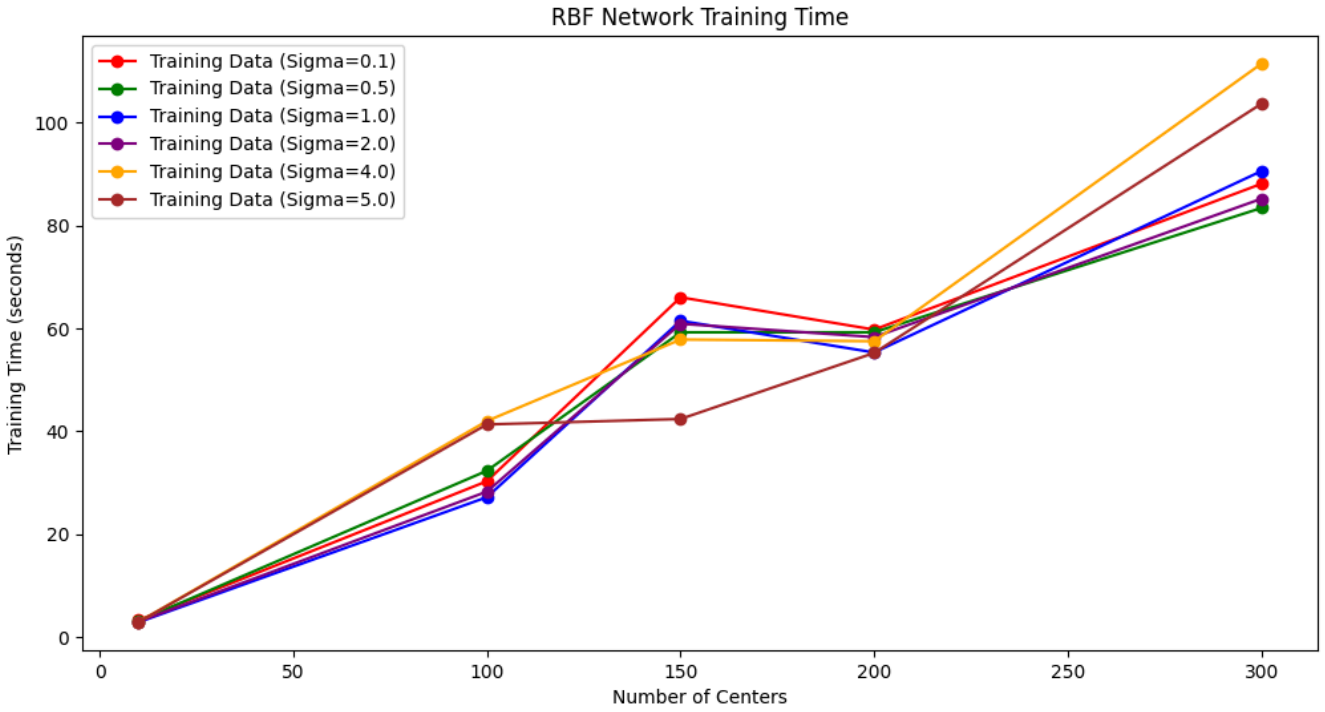
Για τον τρόπο εκπαίδευσης K-μέσοι έχουμε τις παρακάτω γραφικές παραστάσεις:





Οι παραπάνω γραφικές παραστάσεις εμφανίζουν παρόμοια αποτελέσματα για τα *training* και *testing* δεδομένα όσον αφορά τις μεταβολές με την τιμή του Accuracy στα *training data* να είναι πιο αυξημένη (όπως θα περιμέναμε).

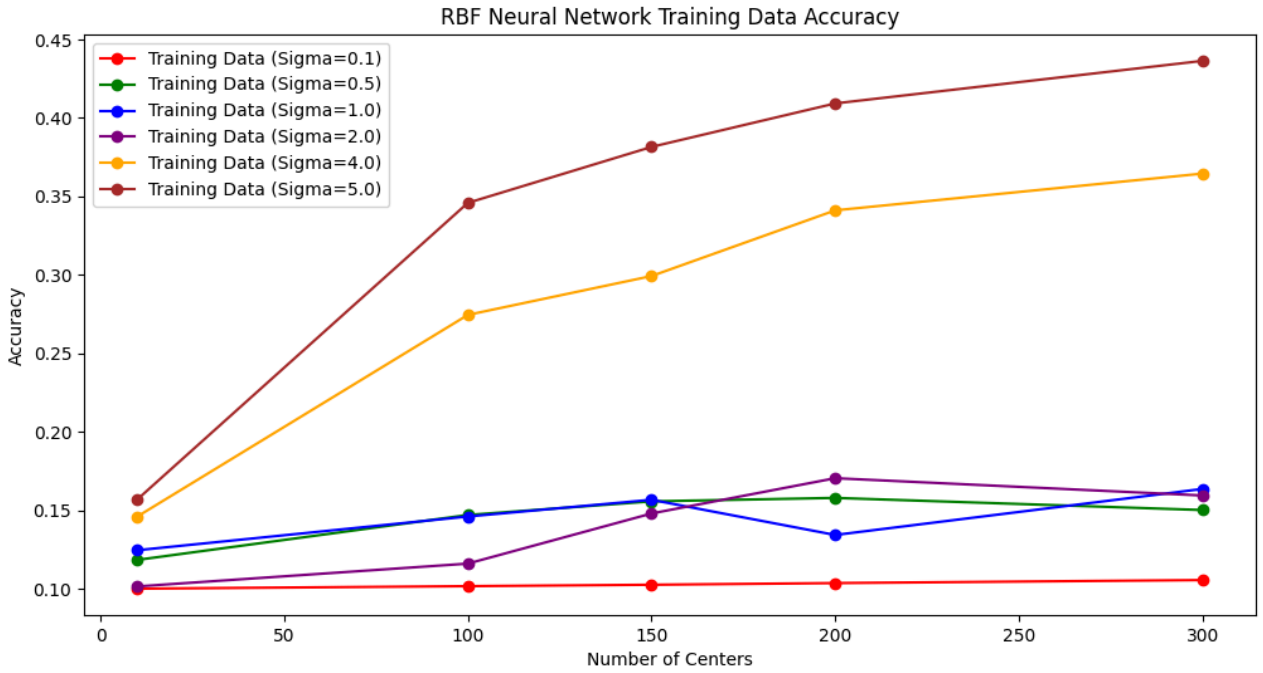
Γενικότερα, παρατηρούμε ότι με την αύξηση του αριθμού των κέντρων, η ακρίβεια αυξάνεται για όλες τις τιμές του σίγμα, αλλά με διαφορετικούς ρυθμούς. Για μικρές τιμές σίγμα (π.χ. 0.1), η ακρίβεια έχει χαμηλό ποσοστό μένει σταθερή με την αύξηση των κέντρων. Αντίθετα, για μεγαλύτερες τιμές σίγμα (π.χ. 5.0), η ακρίβεια ξεκινά από υψηλότερο σημείο και φαίνεται να αυξάνεται πιο ραγδαία. Οι καμπύλες για τις τιμές *sigma* 1.0 και 2.0 δείχνουν μέτρια αρχική ακρίβεια και μέτρια αύξηση.

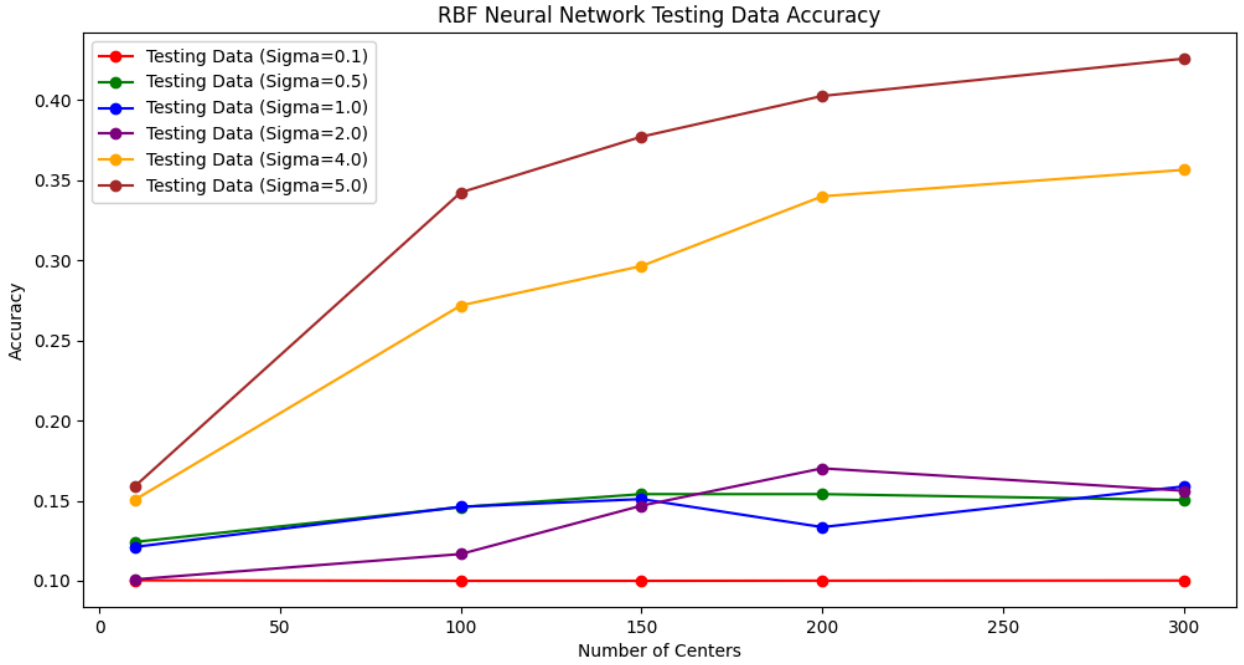


Από το παραπάνω διάγραμμα παρατηρούμε ότι γενικότερα ο χρόνος εκπαίδευσης αυξάνεται με την αύξηση του αριθμού των κέντρων, ανεξάρτητα από την τιμή του *sigma*. Η αύξηση του χρόνου εκπαίδευσης δεν είναι γραμμική και φαίνεται να υπάρχουν ορισμένες απότομες αυξήσεις σε συγκεκριμένα διαστήματα του αριθμού των κέντρων. Η τιμή του *sigma* φαίνεται να έχει λιγότερη επίδραση στον χρόνο εκπαίδευσης σε σύγκριση με την επίδραση που είχε στην ακρίβεια του πρώτου διαγράμματος. Παρόλα αυτά, μπορεί να παρατηρηθεί ότι σε υψηλότερες τιμές του *sigma* ο χρόνος εκπαίδευσης φαίνεται να είναι περισσότερα αυξημένος σε σχέση με χαμηλότερες τιμές.

Παρατηρείται μια ενδιαφέρουσα κατάσταση στο διάστημα ανάμεσα στα 100 και 200 κέντρα όπου οι καμπύλες διασταυρώνονται σε μερικά σημεία, υποδηλώνοντας ότι η συγκεκριμένη επιλογή του *sigma* μπορεί να έχει διαφορετικό αντίκτυπο στην απόδοση του χρόνου εκπαίδευσης σε διαφορετικά διαστήματα του αριθμού κέντρων.

Για τον τρόπο εκπαίδευσης με τυχαία επιλογή κέντρων έχουμε τις παρακάτω γραφικές παραστάσεις:



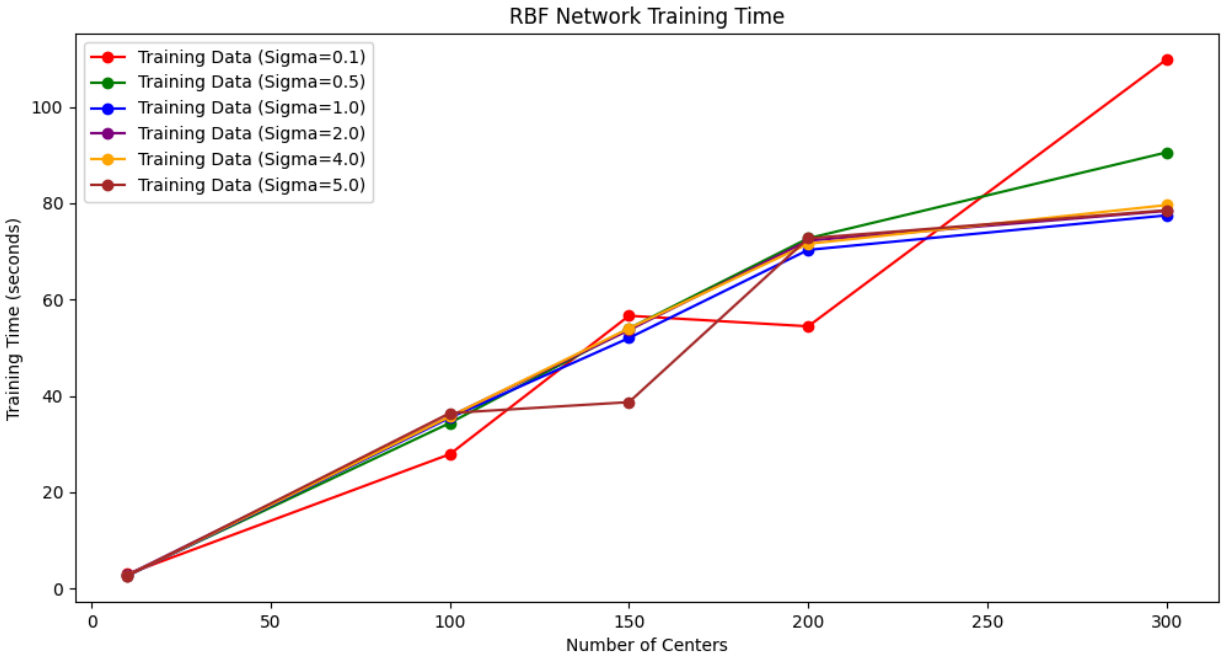


Όπως και προηγουμένως, οι παραπάνω γραφικές παραστάσεις εμφανίζουν παρόμοια αποτελέσματα για τα *training* και *testing* δεδομένα όσον αφορά τις μεταβολές με την τιμή του Accuracy στα *training data* να είναι πιο αυξημένη (όπως θα περιμέναμε). Μερικές περαιτέρω παρατηρήσεις είναι οι παρακάτω:

Η ακρίβεια για *sigma* 0.1 παραμένει σταθερή και πολύ χαμηλή σε όλο το εύρος των κέντρων. Για *sigma* 0.5 η ακρίβεια αρχικά αυξάνεται και στη συνέχεια παραμένει σχεδόν

σταθερή μετά την τιμή των κέντρων ίση με 50, ενώ για *sigma* 1.0 η ακρίβεια αυξάνεται αργά και σταθερά σε όλο το εύρος των κέντρων.

Για *sigma* 2.0 η ακρίβεια αρχίζει από ένα υψηλότερο σημείο, αυξάνεται γρήγορα μέχρι την τιμή 50 των κέντρων και μετά αυξάνεται πιο ομαλά, ενώ για *sigma* 4.0 η ακρίβεια αυξάνεται σημαντικά για τους αριθμούς των κέντρων μεταξύ 0 και 50, μετά από το οποίο η αύξηση επιβραδύνεται. Τέλος, για *sigma* 5.0 η ακρίβεια ξεκινά από το υψηλότερο αρχικό σημείο και αυξάνεται σταθερά, φτάνοντας την υψηλότερη ακρίβεια στο τέλος του διαγράμματος.

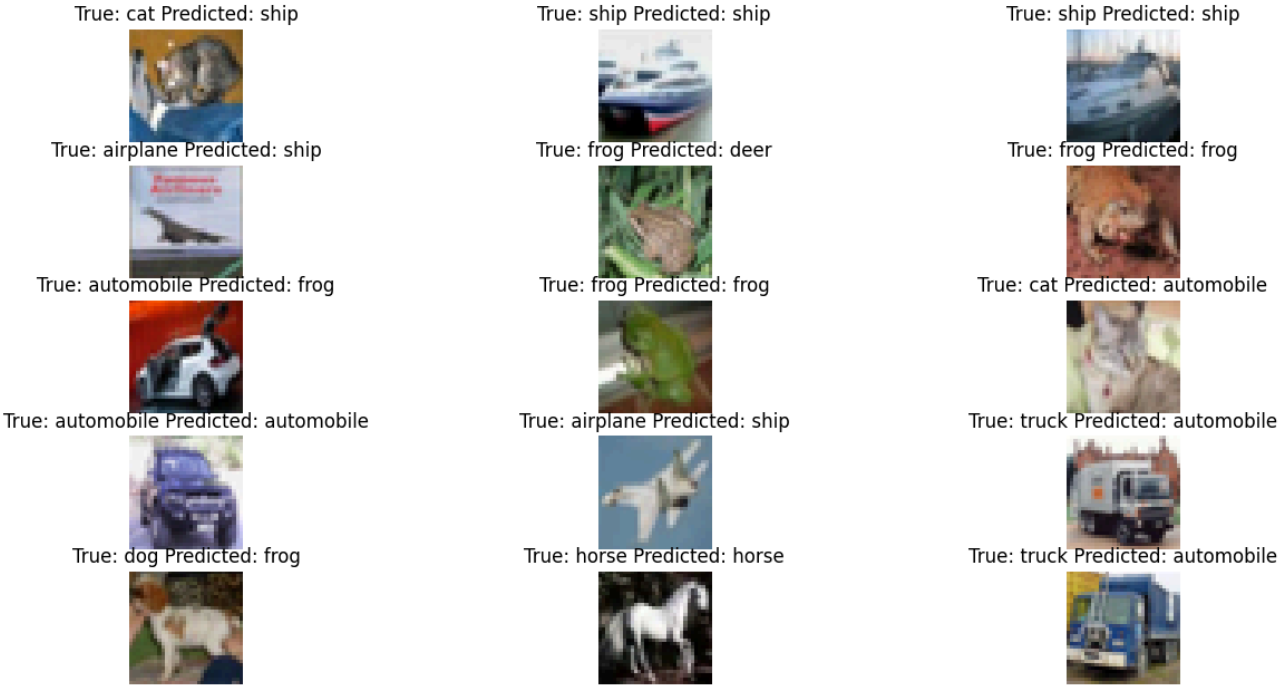


Από το παραπάνω διάγραμμα φαίνεται ότι ο χρόνος εκπαίδευσης τείνει να αυξάνεται με την αύξηση του αριθμού των κέντρων (όπως και προηγουμένως), αλλά η ακριβής μορφή της αύξησης διαφέρει ανάλογα με την τιμή του *sigma*. Υπάρχουν επίσης ορισμένα σημεία όπου οι καμπύλες φαίνεται να διασταυρώνονται ή να παρουσιάζουν μη γραμμικές αλλαγές στον χρόνο εκπαίδευσης, πράγμα που ενδέχεται να είναι αποτέλεσμα συγκεκριμένων δυναμικών του μοντέλου ή της διαδικασίας εκπαίδευσης. Πιό συγκεκριμένα:

- Για *sigma = 0.1* ο χρόνος εκπαίδευσης φαίνεται να αυξάνεται γραμμικά με την αύξηση του αριθμού των κέντρων.
- Για τις τιμές *sigma = 0.5* και *1.0* ο χρόνος εκπαίδευσης ακολουθεί παρόμοια πρότυπα, αυξάνεται γρήγορα μέχρι τα 100 κέντρα και στη συνέχεια φαίνεται να επιβραδύνει η αύξησή του.
- Για *sigma = 2.0* ο χρόνος εκπαίδευσης αυξάνεται σημαντικά μεταξύ 50 και 150 κέντρων και μετά από αυτό το σημείο συνεχίζει να αυξάνεται αλλά πιο ομαλά.
- Για *sigma = 4.0* και *5.0* ο χρόνος εκπαίδευσης ακολουθεί μια παρόμοια ανοδική τάση, με την αύξηση να είναι πιο απότομη μετά τα 100 κέντρα.

4. Παραδείγματα σωστής και λανθασμένης κατηγοριοποίησης

Ο κώδικας για την αναπαράσταση των παραδειγμάτων βρίσκεται στο αρχείο με όνομα *radial_basis_function_examples.py.* Μερικά παραδείγματα σωστής και λανθασμένης κατηγοριοποίησης είναι τα παρακάτω.



Όπως παρατηρούμε το νευρωνικό δίκτυο έχει καταφέρει να προβλέψει με επιτυχία κάποιες από τις εικόνες, ενώ έχει αποτύχει σε κάποιες άλλες.

5. Σύγκριση με KNN και Nearest Class Centroid

Συγκριτικά με τους δύο κατηγοριοποιητές μας KNN και Nearest Class Centroid που είχαμε εξετάσει στο πρώτο παραδοτέο, παρατηρούμε ότι το νευρωνικό μας δίκτυο κινείται στα ίδια επίπεδα ακρίβειας και αναλόγως με τις μεταβλητές *sigma* και *num_centers* που επιλέγουμε κάθε φορά επιτυγχάνει καλύτερα ή χειρότερα αποτελέσματα.

6. Συμπεράσματα

Η εκτέλεση και ανάλυση των διαφόρων μοντέλων μηχανικής μάθησης στο πλαίσιο της CIFAR-10 βάσης δεδομένων μας επέτρεψε να εξάγουμε σημαντικά συμπεράσματα σχετικά με την απόδοση και την πρακτικότητα των Radial Basis Function Neural Networks (RBFNNs), καθώς και τη σύγκριση τους με τα KNN και Nearest Class Centroid μοντέλα.

Η απόδοση των RBFNNs επηρεάζεται έντονα από την επιλογή των παραμέτρων *sigma* και αριθμού κέντρων. Συγκεκριμένα, υψηλότερες τιμές *sigma* φαίνεται να οδηγούν σε καλύτερη αρχική ακρίβεια, ενώ η αύξηση των κέντρων βελτιώνει γενικά την ακρίβεια. Παρά την ευελιξία των RBFNNs, η επιλογή των κατάλληλων παραμέτρων αποτελεί μια σημαντική πρόκληση, η οποία απαιτεί προσεκτική διερεύνηση και πειραματισμό.

Σε σύγκριση με τα KNN και Nearest Class Centroid μοντέλα, τα RBFNNs παρουσίασαν ανάλογα επίπεδα ακρίβειας, αν και τα αποτελέσματα διαφέρουν ανάλογα με τις επιλεγμένες παραμέτρους. Η ευελιξία των RBFNNs στην προσαρμογή τους στα δεδομένα μπορεί να προσφέρει πλεονεκτήματα σε σύνθετα προβλήματα ταξινόμησης, αλλά επίσης εισάγει πρόσθετη πολυπλοκότητα στην εκπαίδευση των μοντέλων.

Η μελέτη αυτή επιβεβαιώνει επίσης ότι η υπολογιστική απόδοση είναι κρίσιμη πτυχή, καθώς ο χρόνος εκπαίδευσης αυξάνεται με τον αριθμό των κέντρων. Τα RBFNNs απαιτούν περισσότερους πόρους και χρόνο σε σύγκριση με πιο απλά μοντέλα, ένας παράγοντας που πρέπει να ληφθεί υπόψη σε πραγματικές συνθήκες εφαρμογής.

7. Πηγές

- https://medium.com/@evertongomede/radial-basis-functions-neural-networks-unlocking-the-power-of-nonlinearity-c67f6240a5bb
- https://towardsdatascience.com/most-effective-way-to-implement-radial-basis-function-neural-network-for-classification-problem-33c467803319
- https://gamedevacademy.org/using-neural-networks-for-regression-radial-basis-function-networks/
- https://matplotlib.org/stable/gallery/subplots_axes_and_figures/gridspec_and_subplots.html#sphx-glr-gallery-subplots-axes-and-figures-gridspec-and-subplots-py