Problem Set 4

Τρίμας Χρήστος
ΑΜ:2016030054
Στατιστική Μοντελοποιήση και Αναγνώριση Προτύπων
ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

July 7, 2020

Ερώτηση 1

Feature Selection - Classification - Cross Validation - Overfitting:

Στην 1η άσκηση του τελευταίου σέτ ασκήσεων για το μάθημα, πραγματοποιήθηκε ένα πείραμα για να δοθεί η απάντηση στο εξής ερώτημα: "Σε ποιο σημείο, κατά την διαδικασία της ταξινόμησης, είναι ιδανικό να εφαρμοστεί επιλογή των χαρακτηριστικών (feature selection)". Για τα τεχνητά δεδομένα προσομομοίωσης του πειράματος μας (ασθενείς αν έχουν ή οχι αυτισμό), θα δοκιμαστούν 3 τεχνικές:

1) Leave One Out.

2) Τροποποιημένος Leave One Out, με χρήση επιλογής χαρακτηριστικών, χρησιμοποιώντας έναν αλγόριθμο σύγκρισης, βασισμένο στον συντελεστή συσχέτισης.

3) Πάλι τροποποιημένο Leave One Out, μόνο που αυτή την φορά η επιλογή των χαρακτηριστικών, γίνεται μια φορά στην αρχή.

Πιο αναλυτικά, στην 1η περίπτωση, κάνουμε ταξινόμηση των δεδομένων μας, με την χρήση SVM, και με την κλασσική τεχνική Leave One Out (ένα δεδομένο test, και τα υπόλοιπα δεδομένα εκπαίδευσης). Αρκετά αποτελεσματική και εύκολη στην υλοποιήση μέθοδος, πετυχαίνει ακρίβεια κοντά στο 50%.

Στην 2η περίπτωση τώρα, σε κάθε επανάληψη του LOO, εφαρμόζεται feature selection, με την χρήση ενός αλγορίθμου σύγκρισης. Και αυτή η τεχνική, πετυχαίνει ακρίβεια κοντά στο 50%, όπως και αμέσως προηγούμενη, κάτι το οποίο το περιμέναμε, καθώς τα τεχνητά μας δεδομένα, στην πράξη δεν δίνουν πληροφορίες για τους ασθενείς (αν έχουν ή όχι αυτισμό).

Στην 3η και τελευταία τεχνική, όπως εξηγήθηκε παραπάνω, η επιλογή των χαρακτηριστικών έγινε μια και μόνο φορά, ενώ κατά τα άλλα χρησιμοποιήθηκε κλασσικός LOO αλγόριθμος. Δυστυχώς, σε αυτή την περίπτωση, έχουμε φαινόμενο overfitting (έχει εξηγηθεί σε προηγούμενο σετ ασκήσεων) και για αυτό τον λόγο έχουμε μονίμως ακρίβεια 1, δηλαδή 100%.

Ερώτηση 2

Υλοποίηση ενός απλού νευρωνικού δικτύου:

Σε αυτή την άσκηση, σκοπός ήταν η δημιουργία του πιο απλού Νευρωνικού Δικτύου, που μπορεί να υπάρξει. Δηλαδή ένα δίκτυο με μόνο είσοδο και έξοδο, χωρίς κανένα κρυφό επίπεδο. Σαν συνάρτηση ενεργοποίσης του δικτύου χρησιμοποιήθηκε η sigmoid συνάρτηση, η οποία ορίζεται ως:

$$f(z) = \frac{1}{1 + e^{-z}}$$

Για τον υπολογισμό του σφάλματος ανάμεσα στην πρόβλεψη και στην πραγματική τιμή χρησιμοποιήθηκε η συνάρτηση cross-entropy, η οποία ορίζεται ως εξής:

$$J(y^{(i)}, \hat{y}^{(i)}; W, b) = -y^{(i)} ln(\hat{y}^{(i)}) - (1 - y^{(i)}) ln(1 - \hat{y}^{(i)})$$

Για τα ερωτήματα α,β,γ δείτε το σκαναρισμένο.

Για το forward propagation του NN, χρείαστηκε να υπολογιστεί η είσοδος πολ/νη με τα βάρη και να προστεθεί το bias. Στη συνέχεια, το αποτέλεσμα αυτό, δίνεται ως είσοδος στην sigmoid function και βρίσκεται κατά αυτόν τον τρόπο η έξοδος του δικτύου μας.

Επόμενο βήμα είναι η αξιολόγηση του αποτελέσματος μας, και αύτο γίνεται με την cross-entropy function, όπως προαναφέρθηκε.

Για να ελαχιστοποιηθούν αυτές οι απώλειες και να μπορέσουν να ανανεωθούν οι τιμές των βαρών και το bias του δικτύου, γίνεται μια "ανατροφοδότηση" της προβλεπόμενης εξόδου στο δίκτυο. Αυτή η διαδικασία ονομάζεται backward propagation, και στην παρούσα εργασία υλοποιήθηκε με αλγόριθμο gradient descent.

Με την εύρεση των επιθυμητών παραγώγων, γίνεται ανανέωση των βαρών και του bias, όπως φαίνεται στον κώδικα.

Για να δούμε τώρα πόσο καλά αποδίδει το δίκτυο, αλλάχτηκε ο αριθμός των epochs, με την χρήση του kubeflow της google, σε 500, έτσι ώστε το epoch loss να είναι το ελάχιστο δυνατό για accuracy του μοντέλου 90%.

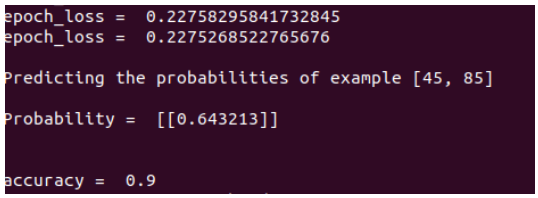

Figure 1: Accuracy and epoch loss

Στη συνέχεια ακολουθεί το αρχικό dataset το οποίο γνωρίζαμε από την αρχή τα labels, και η πρόβλεψη μας με την χρήση του μοντέλου που υλοποιήθηκε.

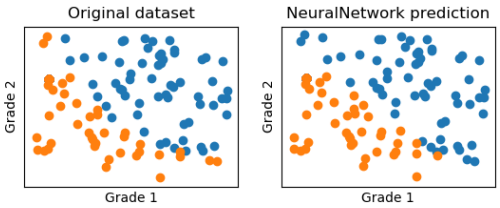

Figure 2: Original vs NN

Τέλος, για χάρη πληρότητας, έγινε σύγκριση του μοντέλου μας, με την απλή λογιστική παλινδρόμηση η οποία παρέχεται από το scikit-learn API. Συγκεκριμένα, το αποτέλεσμα είναι αρκετά κοντά, και μάλιστα για όσο το δυνατόν μικρότερο epoch loss, τόσο καλύτερα αποδίδει το μοντέλο μας από την απλό ταξινόμητη που παρέχεται.

Ερώτηση 3

Convolutional Neural Networks for Image Recognition:

Στην τελευταία άσκηση του σέτ, θα γίνει η κατασκευή ενός Deep Convolutional Neural Network, με σκόπο την αναγνώριση φωτογραφιών ειδών μόδας.

Αρχικά, θα παρακολουθήσουμε την εξέλιξη των δεικτών accuracy (επί των δεδομένων εκπαίδευσης) και τον δείκτη validation accuracy (επί των δεδομένων επικύρωσης).
Για αριθμό epochs=400, οι κλιμάκωση των δεικτών φαίνεται στην παρακάτω εικόνα:

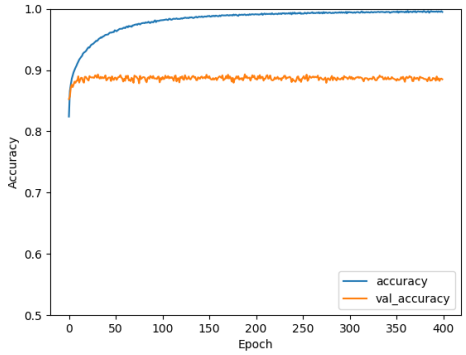

Figure 3: Accuracy and validation accuracy for adam optimizer(400 epochs).

Παρατηρείται, ότι το accuracy, τείνει στην μονάδα, σε αντίθεση με το validation accuracy, το οποίο διατηρεί σταθερή μέση τιμή κοντά στο 0.9 ή 90%.

Στο ακόλουθο γράφημα, φαίνεται η πρόβλεψη του μοντέλου.

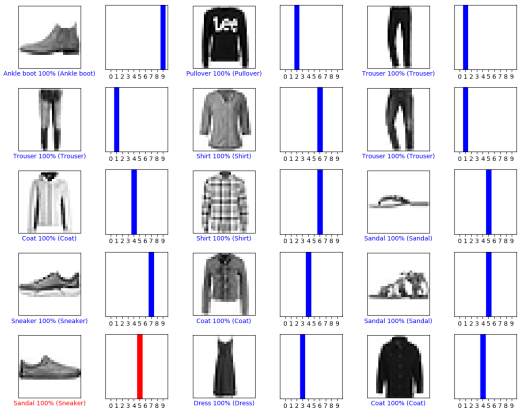

Figure 4: Probabilities of each class

Για το επόμενο ερώτημα, μειώθηκε ο αριθμός των epochs στα 50 και δοκιμάτηκαν διάφοροι optimizers, οι οποίοι παρέχονται από το keras API.

Συγκεκριμένα, οι αλδόριθμοι που δοκιμάστηκαν είναι οι εξής:

1) **adam.**
2) **sgd.**
3) **rmsprop.**
4) **nadam.**
5) **adamax.**
6) **ftrl.**

Στις εικόνες που ακολουθούν, παρουσιάζονται τα accuracy και validation accuracy για κάθε έναν από τους προαναφερθέντες αλγορίθμους, καθώς επίσης και οι προβλέψεις τους.

Με παρατήρηση όλων των παρακάτω καταλήγει κανείς εύκολα στο συμπέρασμα, ότι ο adam optimizer, είναι ο ιδανικότερος για το πείραμα μας, καθώς έχει τόσο μεγαλύτερο accuracy, όσο και validation accuracy.

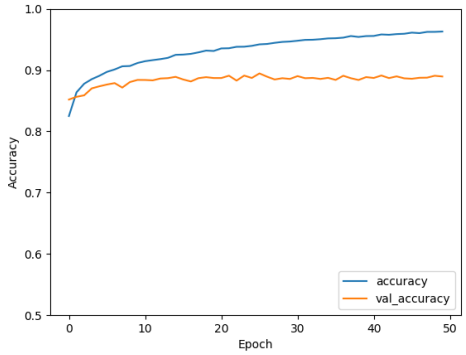

Figure 5: Adam accuracy

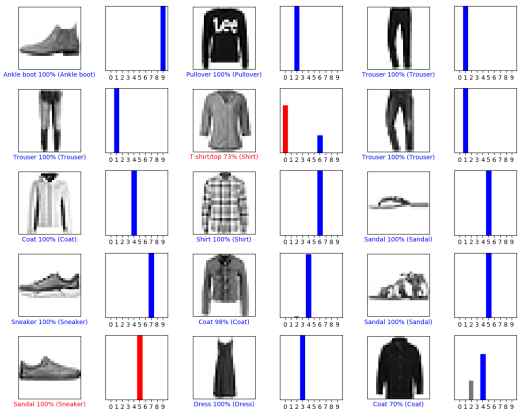

Figure 6: Adam probabilities

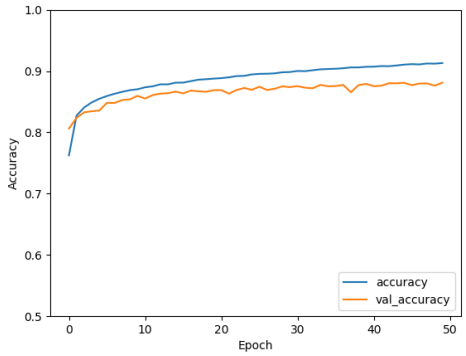

Figure 7: Sgd accuracy

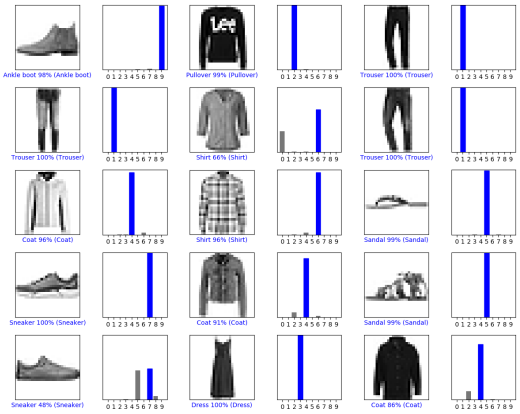

Figure 8: Sgd probabilities

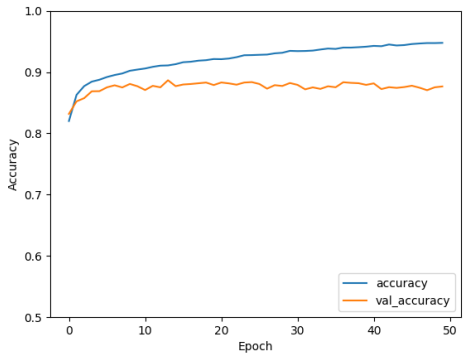

Figure 9: Rms accuracy

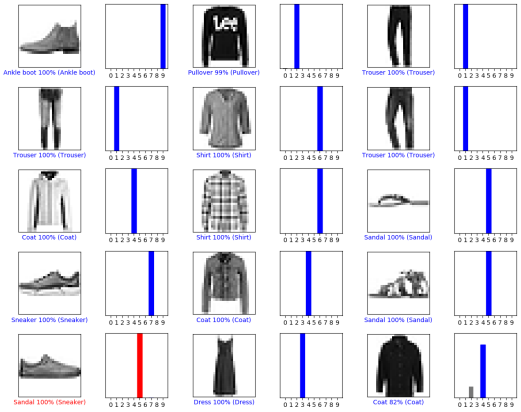

Figure 10: Rms probabilities

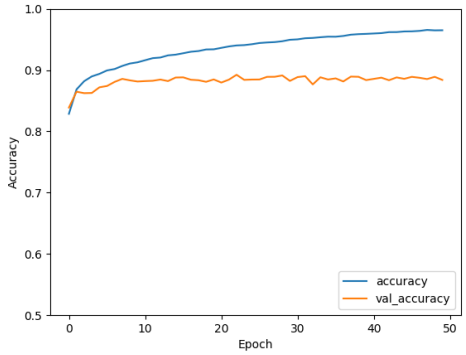

Figure 11: Nadam accuracy

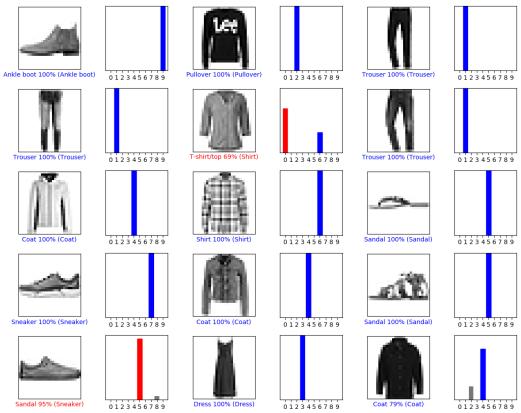

Figure 12: Nadam probabilities

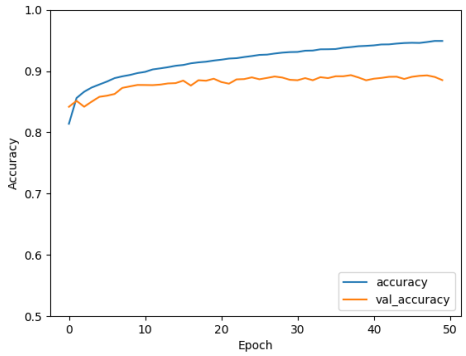

Figure 13: Adamax accuracy

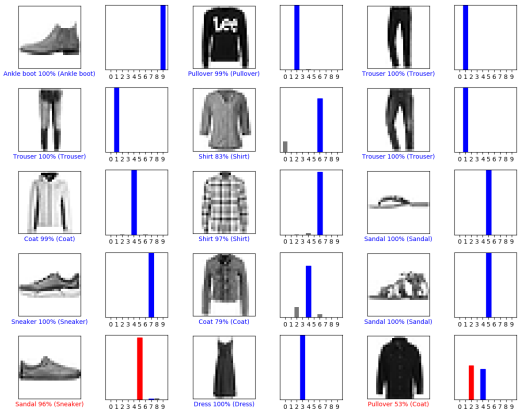

Figure 14: Adamax probabilities

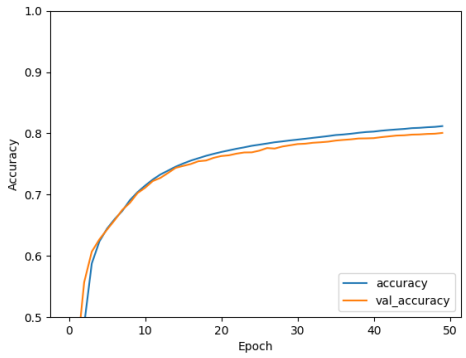

Figure 15: Ftrl accuracy

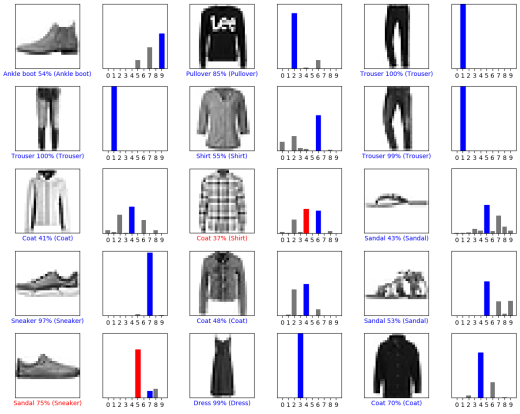

Figure 16: Ftrl probabilities

Με ότι ασχοληθήκαμε μέχρι στιγμής, δόθηκε έτοιμό από την άσκηση. Στη συνέχεια, κατασκευάστηκε ένα NN, σύμφωνα με την εκφώνηση και τις οδηγίες. Τα αποτελέσματα που δίνονται παρακάτω, μπορεί κανείς να τα δει και στα notebooks που επισυνάπτονται. Ο λόγος για τον οποίο παραδίδονται σε αυτή την μορφή είναι γιατί έγινε χρήση GPU από το google colab.

Αρχικά, έγινε η υλοποίηση του μοντέλου χωρίς batch normilization, και τα αποτελέσματα τόσο της ακρίβειας, όσο και της πρόβλεψης του μοντέλου φαίνονται παρακάτω:

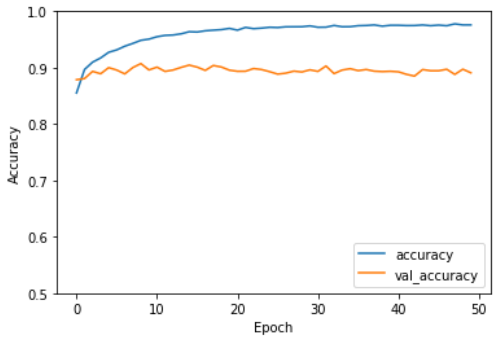

Figure 17: CNN 1st accuracy

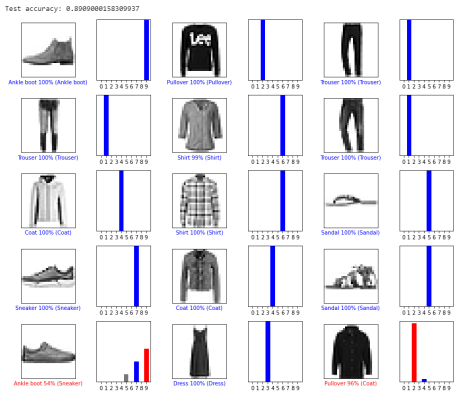

Figure 18: CNN 1st probabilities

Η ακόλουθη υλοποίηση, περιλαμβάνει και batch normalization, σύμφωνα πάντα με τις οδηγίες της εκφώνησης:

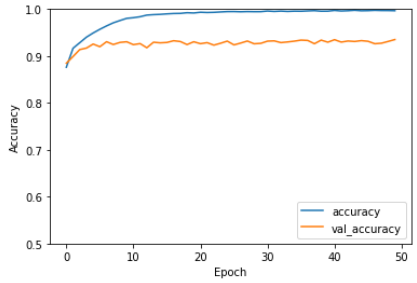

Figure 19: CNN 2nd accuracy

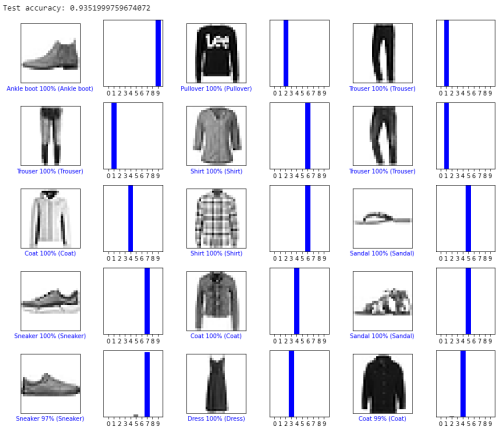

Figure 20: CNN 2nd probabilities

Τέλος, ακολουθεί το ολοκληρωμένο μοντέλο πάλι συμφωνα με τις οδηγίες της εκφώνησης:

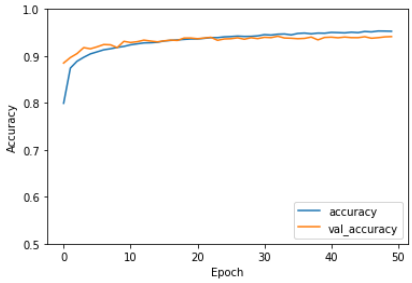

Figure 21: CNN 3rd accuracy

Φαίνεται πως το validation accuracy, αυξάνεται με την προσθήκη επιπλέον επιπέδων στο μοντέλο, και τείνει το accuracy και το validation accuracy, να ταυτιστούν.

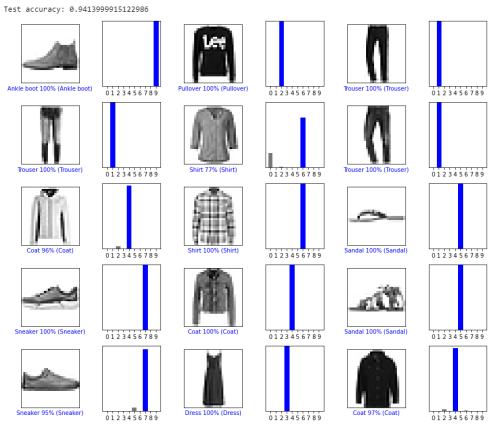

Figure 22: CNN 3rd probabilities