ΕΡΓΑΣΙΑ ΣΤΟ ΜΑΘΗΜΑ ΝΕΥΡΩΝΙΚΑ ΔΙΚΤΥΑ-ΒΑΘΙΑ ΜΑΘΗΣΗ

1η εργασία

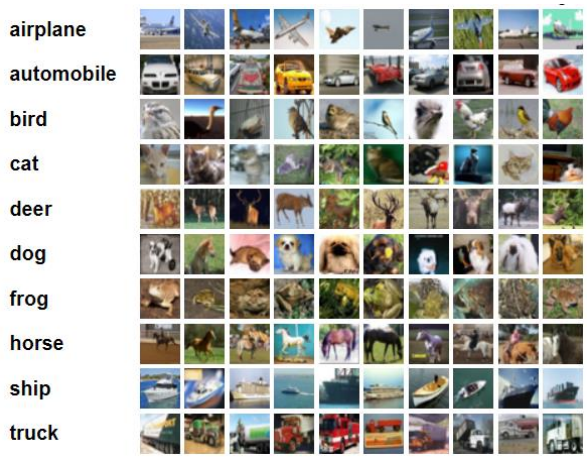
MLP Backpropagation

Ηλιάνα Κόγια 10090 ilianakogia@ece.auth.gr

Νοέμβριος 2023

Classification

Επιλέχθηκε για το πρόβλημα της ταξινόμησης η βάση δεδομένων **Cifar-10**, η οποία αποτελείται από 60000 RGB-εικόνες διάστασης 32x32 σε 10 κλάσεις με 6000 εικόνες/κλάση. Στους παρακάτω αλγορίθμους χρησιμοποιούνται όλα τα δεδομένα εκπαίδευσης (50000 εικόνες) και όλα τα δεδομένα ελέγχου (10000 εικόνες) του συγκεκριμένου dataset. Κάθε γραμμή είναι ένα δείγμα-εικόνα και αποτελείται από 3072 στοιχεία (features), τα πρώτα 1024 είναι για το red-channel, τα επόμενα 1024 για το green-channel και τα υπόλοιπα 1024 για το blue-channel.



1.1 K-NN classifier

Κεντρικός στόχος αυτού του αλγορίθμου είναι να προβλέψει την κλάση στην οποία ανήκει μια παρατήρηση (εδώ μια εικόνα) με βάση τους k πλησιέστερους γείτονες – παρατηρήσεις στο χώρο των features. Η παράμετρος *k αποτελεί υπερ-παράμετρο* του αλγορίθμου. Στην ανάλυση που έγινε επιλέχθηκαν οι τιμές **k = 1** και **k = 3**.

Η απόδοση στις 2 περιπτώσεις είναι:

```
------ k-NN Classifier------
Accuracy for k = 1 : 35.39%
Model classification report for k = 1 :
              precision    recall  f1-score   support

           0       0.42      0.48      0.45      1000
           1       0.65      0.22      0.33      1000
           2       0.24      0.38      0.30      1000
           3       0.29      0.24      0.26      1000
           4       0.25      0.46      0.32      1000
           5       0.36      0.29      0.32      1000
           6       0.33      0.35      0.34      1000
           7       0.56      0.29      0.39      1000
           8       0.40      0.62      0.49      1000
           9       0.61      0.20      0.30      1000

    accuracy                           0.35     10000
   macro avg       0.41      0.35      0.35     10000
weighted avg       0.41      0.35      0.35     10000
```

```
Accuracy for k = 3 : 33.03%
Model classification report for k = 3 :
              precision    recall  f1-score   support

           0       0.32      0.57      0.41      1000
           1       0.58      0.24      0.34      1000
           2       0.20      0.45      0.28      1000
           3       0.26      0.23      0.24      1000
           4       0.25      0.44      0.32      1000
           5       0.43      0.21      0.28      1000
           6       0.36      0.23      0.28      1000
           7       0.73      0.20      0.31      1000
           8       0.44      0.61      0.51      1000
           9       0.73      0.12      0.21      1000

    accuracy                           0.33     10000
   macro avg       0.43      0.33      0.32     10000
weighted avg       0.43      0.33      0.32     10000
```

1.2 Nearest Centroid classifier

Κατά την εκπαίδευση υπολογίζεται το κέντρο της κάθε κλάσης με βάση τα δείγματα που ανήκουν σε αυτή την κλάση. Το κέντρο αποτελεί έναν μέσο όρο των παρατηρήσεων της εκάστοτε κλάσης. Κατά την πρόβλεψη για ένα νέο δείγμα υπολογίζεται η απόσταση από τα κέντρα των κλάσεων και το δείγμα ανατίθεται στην

κλάση από το κέντρο της οποίας απέχει την μικρότερη απόσταση. Είναι αποδοτικός όταν οι κλάσεις έχουν σαφώς διαχωρισμένα κέντρα.

Η απόδοση για το συγκεκριμένο dataset είναι:

```
------ Nearest Centroid Classifier ------
Test-set score/Accuracy: 27.74%
Model classification report :
              precision    recall  f1-score   support

           0       0.27      0.54      0.36      1000
           1       0.28      0.19      0.22      1000
           2       0.28      0.11      0.16      1000
           3       0.27      0.06      0.09      1000
           4       0.28      0.12      0.17      1000
           5       0.27      0.29      0.28      1000
           6       0.22      0.54      0.31      1000
           7       0.27      0.17      0.20      1000
           8       0.42      0.37      0.39      1000
           9       0.33      0.41      0.36      1000

    accuracy                           0.28     10000
   macro avg       0.29      0.28      0.25     10000
weighted avg       0.29      0.28      0.25     10000
```

<u>Σύγκριση</u>

accuracy

k = 1 nn	**35.39%**
k = 3 nn	**33.03%**
nc	**27.74%**

Επιτυγχάνουμε μεγαλύτερη ακρίβεια με τον 1-NN, ενώ παρατηρείται η μικρότερη ακρίβεια με τον ταξινομητή nearest centroid, διότι αποτελεί μια πιο απλή προσέγγιση. Ακόμη, στο συγκεκριμένο dataset οι εικόνες έχουν χαμηλή ανάλυση καθιστώντας το πρόβλημα σχετικά δύσκολο. Γνωρίζουμε ότι η απόδοση των αλγορίθμων εξαρτάται από τις ιδιαιτερότητες του dataset, όπως και από την επιλογή των υπερπαραμέτρων.

1.3 Dense neural network – MLP Backpropagation

Τα features για κάθε εικόνα του dataset είναι 3072, οπότε χρησιμοποιούνται 3072 νευρώνες εισόδου, + 1 για το bias. Όσον αφορά το output layer καθώς έχουμε 10 κλάσεις χρησιμοποιούνται 10 νευρώνες εξόδου.

Στη συγκεκριμένη υλοποίηση χρησιμοποιούμε πράξεις πινάκων για την εύρεση των W_i που αντιπροσωπεύουν τα βάρη μεταξύ των νευρώνων δύο layer. Για παράδειγμα, έστω ότι έχουμε το νευρωνικό δίκτυο με L = 3, δηλαδή 2 hidden layer (L = 0 input layer), τότε θα έχουμε αντίστοιχα τους πίνακες των βαρών W_0 (input layer – 1st hidden), W_1 (1st hidden – 2nd hidden) και W_2 (2nd hidden – output layer).

Οι γραμμές των W_i είναι ίσες με τον αριθμό των νευρώνων του (i+1) layer και οι στήλες του layer (i). Επίσης, στους πίνακες των βαρών ενσωματώνουμε τα bias σε κάθε layer προσθέτοντας μια επιπλέον στήλη στο W, η οποία δεν χρησιμοποιείται στο backward computation, αλλά μόνο στο forward.

Loss function:

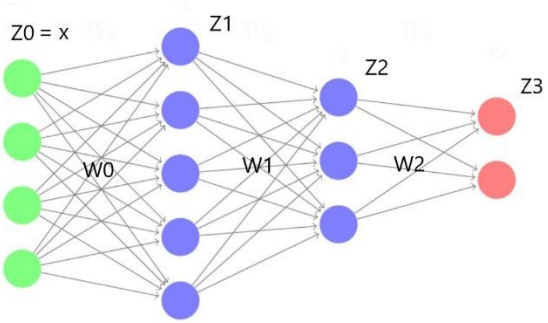
1) Μέσο τετραγωνικό σφάλμα:

$$J = \frac{1}{P}\sum_{p=1}^{P}\left\|\mathbf{d}^{(p)} - \mathbf{y}^{(p)}\right\|^2 = \frac{1}{P}\sum_{p=1}^{P}\sum_{i=1}^{m}[d_i^{(p)} - y_i^{(p)}]^2$$

P : δείγματα εικόνων, m: 10 classes
d, y: one hot encoding

Forward Pass:

$$Z_i = f\,(W_{i-1}\, Z_{i-1}) \text{ , αρχικοποίηση } Z_0 : \text{input x}$$



(Σημείωση: μεταξύ των layers i = 0 και i = 1 αντιστοιχεί ο πίνακας βαρών W_0 κτλ)

Backward Pass:

$$\delta_L = (d - y_{Lout}) \circ f'_L(W_{L-1}\, Z_{L-1})$$

$$\delta_i = W_i^T\, \delta_{i+1} \circ f'(W_{i-1}\, Z_{i-1})$$

(Σημείωση: εκτός της στήλης των bias στα W)

Update:

$$W_i = W_i + \delta_{i+1}\, Z_i^T$$

Συναρτήσεις ενεργοποίησης: sigmoid

2) Categorical Cross Entropy loss:

$$J = -\frac{1}{P}\sum_{p=1}^{P}\sum_{m=1}^{\mathrm{m}} d\,\ln\left(y_{L_{\mathrm{out}}}\right)$$

Συναρτήσεις ενεργοποίησης: sigmoid, output layer: softmax

**Η μόνη διαφορά στις παραπάνω εξισώσεις όταν χρησιμοποιούμε το cross entropy loss είναι στο τελευταίο layer που προκύπτει:*

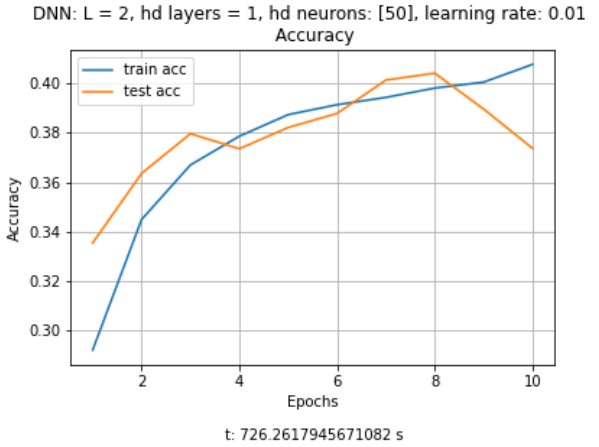
$\delta_L = (d - y_{Lout})$

SGD : μετά από κάθε sample (image) ανανεώνουμε τους πίνακες των βαρών, όπως αναφέρθηκε παραπάνω.

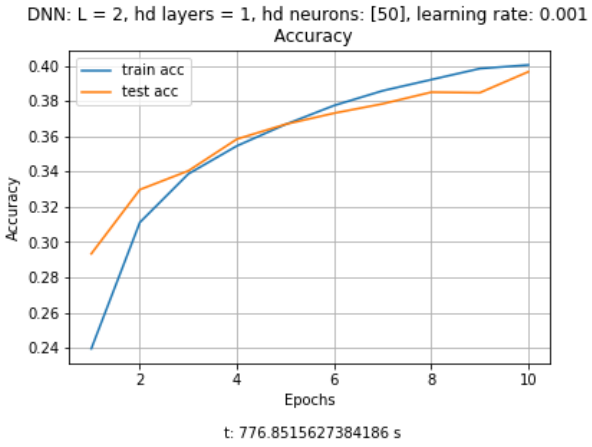
Mini-batch gradient descent: ανανεώνουμε τα βάρη μετά από κάθε mini batch, το οποίο περιλαμβάνει ορισμένο αριθμό δειγμάτων, ώστε να μειώσουμε τον χρόνο εκτέλεσης. Χρησιμοποιείται ο μέσος όρος των gradient των δειγμάτων για την ανανέωση των βαρών σε κάθε batch.

Αποτελέσματα – SGD

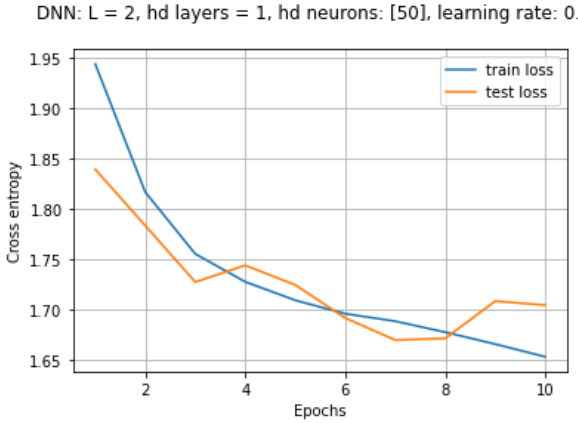
Δοκιμάζουμε ένα κρυφό layer με 50 νευρώνες για a = 0.01, 0.001, 0.0001

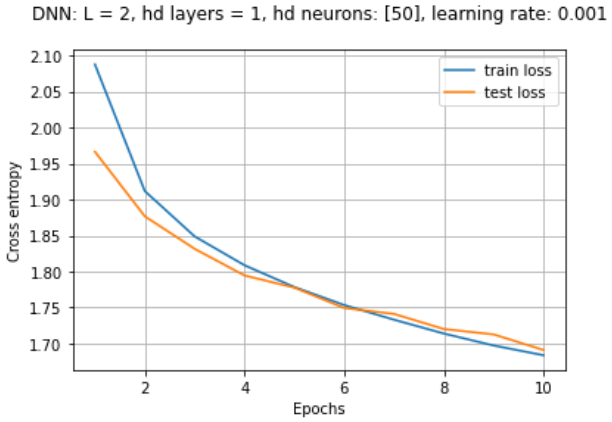


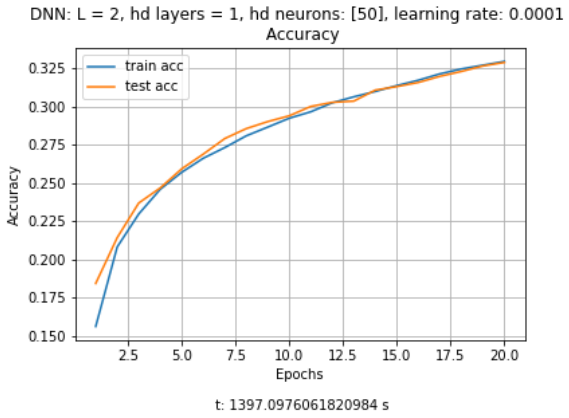
t: 726.2617945671082 s

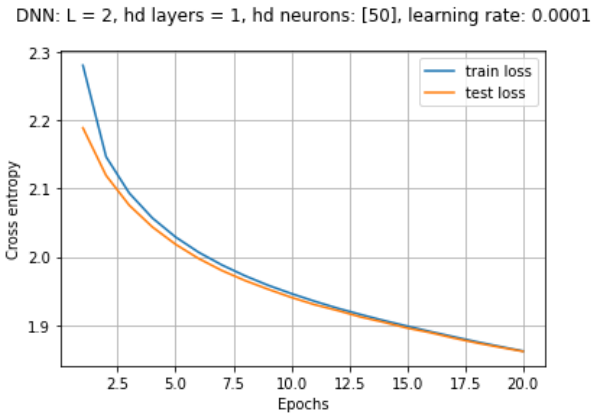


t: 776.8515627384186 s

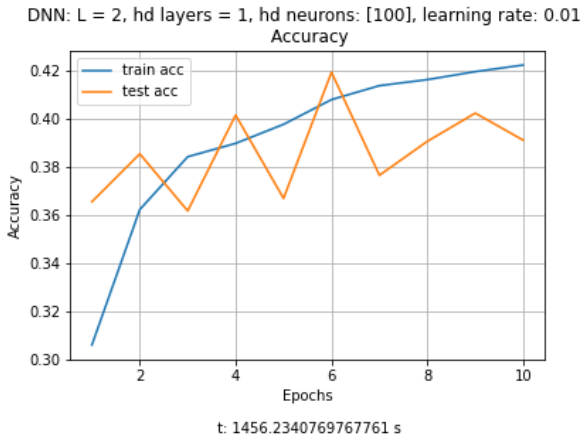


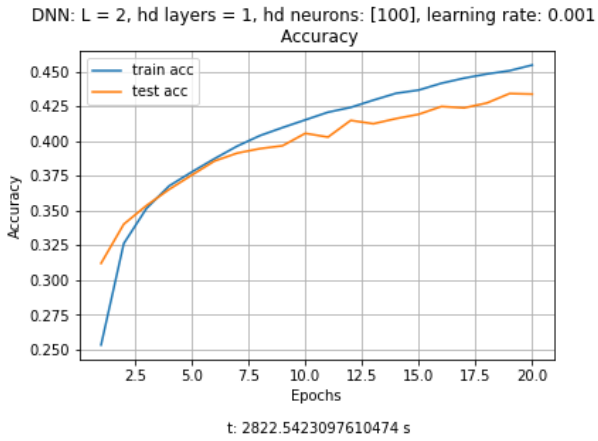


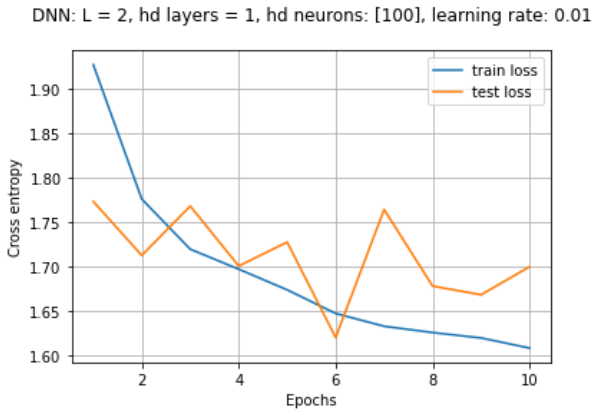


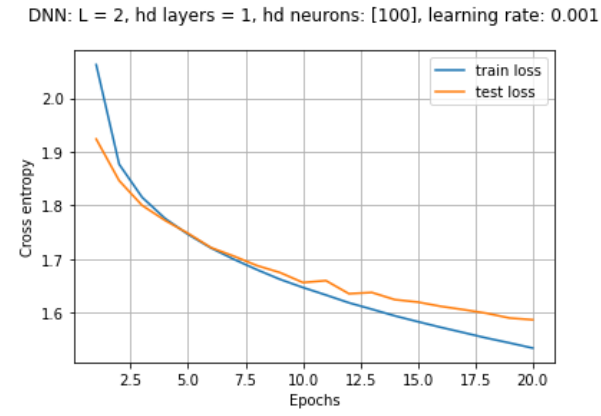


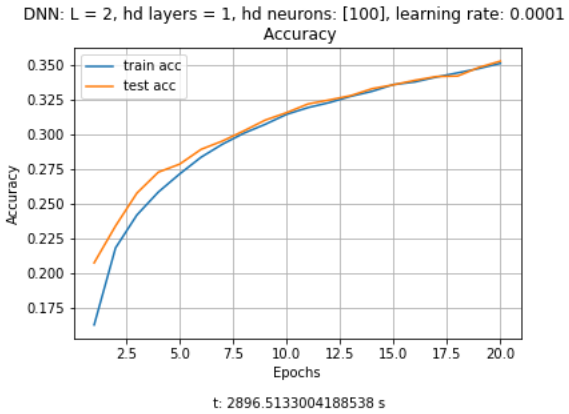
Αυξάνουμε τον αριθμό των νευρώνων σε 100 και δοκιμάζουμε τις ίδιες τιμές στο a:

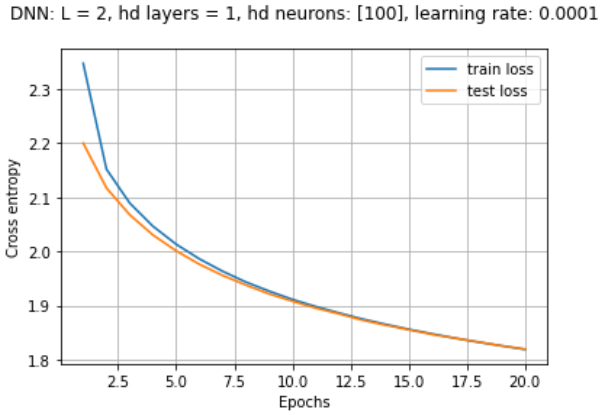












~Για ίδιο αριθμό νευρώνων όσο μειώνουμε το learning rate πετυχαίνουμε μεγαλύτερο stability και καλύτερο generalization

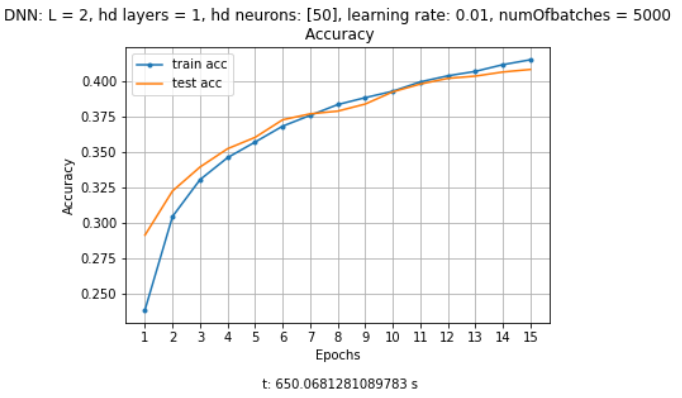
~Για ίδιο learning rate, αυξάνοντας τον αριθμό των νευρώνων του κρυφού στρώματος από 50 σε 100 αυξάνεται και το accuracy για ίδιο αριθμό εποχών

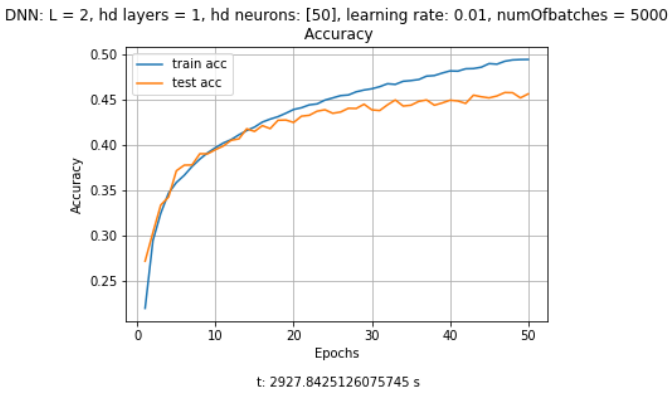
Αποτελέσματα – mini-batch gradient descent

Δοκιμάστηκαν διάφοροι συνδυασμοί των παραμέτρων του νευρωνικού δικτύου. Παρατίθενται τα learning curves:

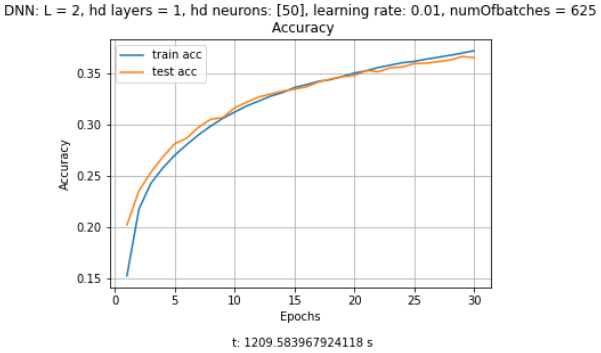
<u>Cross Entropy loss – softmax</u>

Για 50 νευρώνες, a = 0.01, και 5000 mini-batches των 10 samples η καμπύλη συγκλίνει περίπου στις 50 εποχές, ωστόσο το gap είναι της τάξης των 0.05:

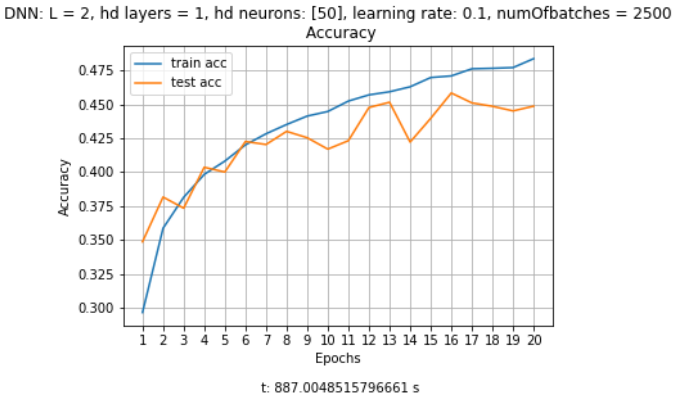


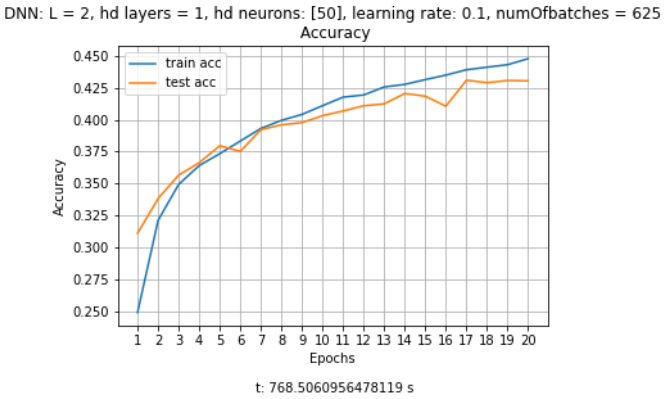


Για ίσο αριθμό νευρώνων στο κρυφό στρώμα και ίδιο a, αν μειώσουμε τον αριθμό των batches 5000 -> 625 δηλαδή αυξάνοντας τα δείγματα που υπάρχουν σε κάθε batch το accuracy στις 30 εποχές είναι μικρότερο, δηλαδή πιο αργή σύγκλιση, ωστόσο η καμπύλη στο test ακολουθεί καλύτερα την καμπύλη εκπαίδευσης:

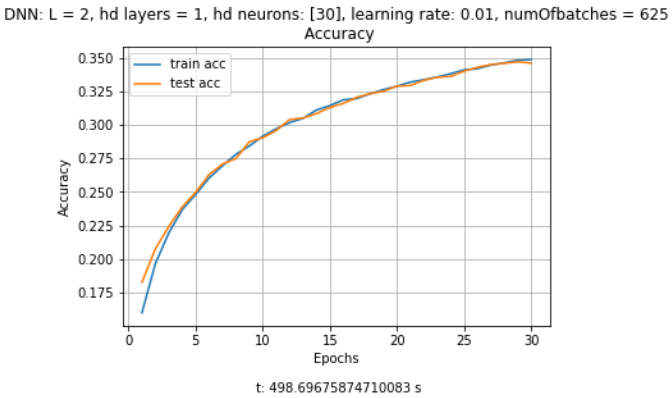


Δοκιμάζοντας learning rate 0.1 και αριθμό mini-batches 2500 (20 samples) και 625 (80 samples) παρατηρούμε ότι ενώ με περισσότερα και μικρότερα batches, (που σημαίνει και περισσότερα updates των βαρών σε κάθε εποχή), μπορεί να πετυχαίνουμε ακρίβεια 0.475, ενώ με λιγότερα και μεγαλύτερα mini-batch 0.45, καλύτερο generalization και μικρότερο overfitting παρατηρείται στα 625 minibatches συγκριτικά με τα 2500:

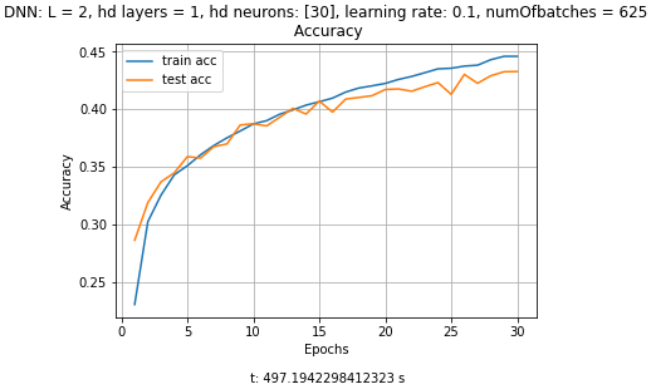




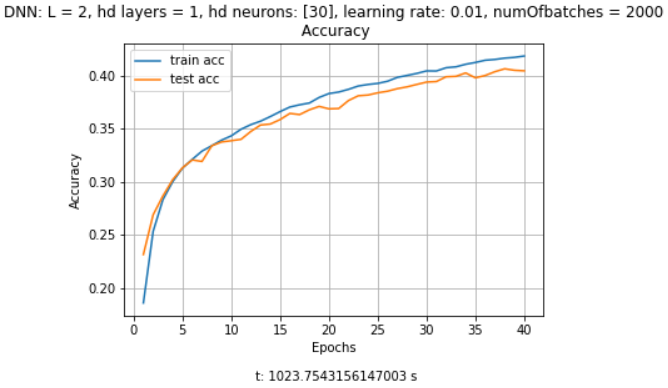
Μειώνοντας τους νευρώνες από 50 σε 30 δεν βλέπουμε μεγάλη διαφορά στο accuracy στις 30 εποχές, ενώ ο χρόνος εκτέλεσης μειώθηκε αρκετά από 498 sec στα 1209 για a = 0.01 και 625 mini-batches:



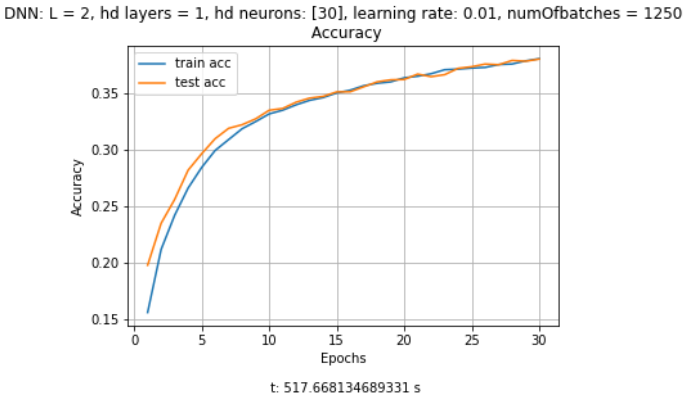
Για learning rate 0.1, 30 νευρώνες και 625 mini batch παρατηρούμε ότι στις 30 εποχές έχουμε ακρίβεια 0.45 δηλαδή συγκλίνει ταχύτερα, αλλά το gap των καμπυλών μεγαλώνει:

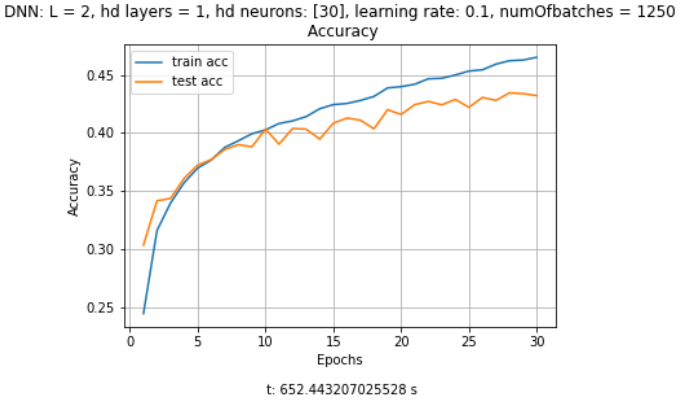


Για learning rate 0.01 και 2000 mini-batch, η ταχύτητα σύγκλισης αυξάνεται συγκριτικά με τα 625:

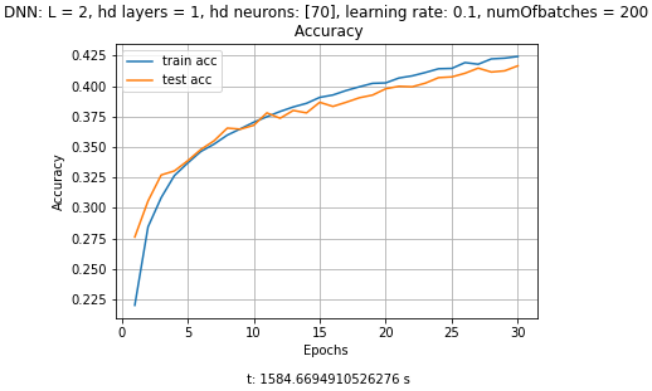


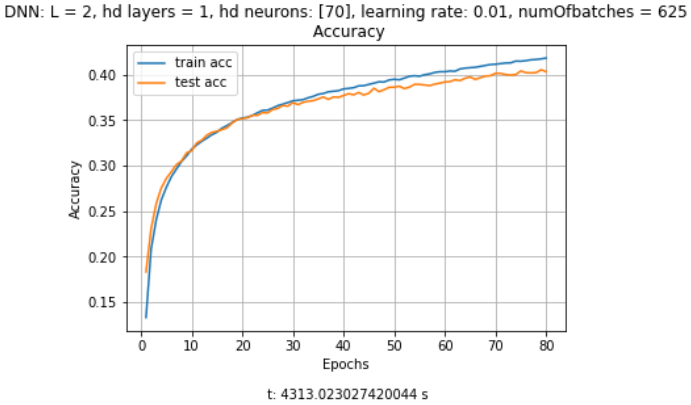
Για 1250 mini-batches, αλλάζοντας το learning rate 0.01->0.1 πετυχαίνουμε μεγαλύτερο accuracy στις 30 εποχές όμως παρατηρείται αύξηση του gap:





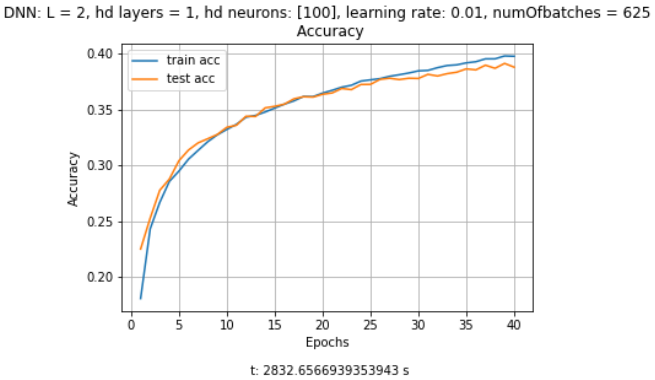
Για 70 νευρώνες δοκιμάζουμε:

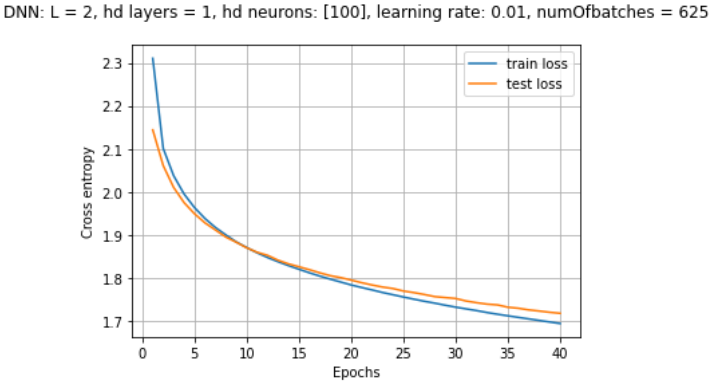




Παρατηρούμε ότι στους 70 νευρώνες επιτυγχάνουμε σχετικά μικρό gap στις 2 καμπύλες και φτάνουμε accuracy 0.42. Μπορούμε να το θεωρήσουμε ικανοποιητικό σχετικά μοντέλο

Για 100 νευρώνες και a = 0.01 και 625 minibatches στις 40 εποχές φτάνουμε 0.4 accuracy, δηλαδή επιτυγχάνουμε καλύτερο performance από τους 70 νευρώνες στις 40 εποχές:





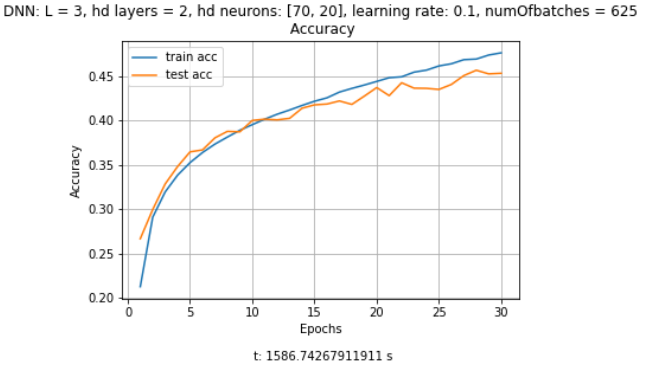
Συνολικά:

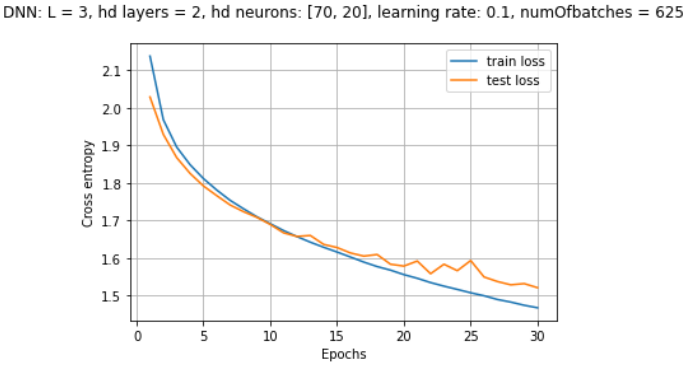
~ Παρατηρούμε ότι για περισσότερους νευρώνες στο κρυφό layer και ίδιες τις άλλες παραμέτρους πετυχαίνουμε καλύτερη ακρίβεια, ωστόσο οι τιμές των άλλων παραμέτρων μπορεί να επηρεάσουν τελικά το performance και να επιτύχεις καλύτερο accuracy με λιγότερους νευρώνες και διαφορετικά learning rate και mini-batch.

~ Για το alpha και τον αριθμό των mini-batch παρατηρούμε ότι: για ίδιο learning rate και αριθμό νευρώνων με περισσότερα mini-batch (με λιγότερα δλδ δείγματα) έχουμε ταχύτερη σύγκλιση, ενώ για ίδιο αριθμό mini-batch το gap προκύπτει μεταξύ των καμπυλών και το αν επιτυγχάνεται καλή γενίκευση, και όχι overfitting, εξαρτάται από το learning rate.

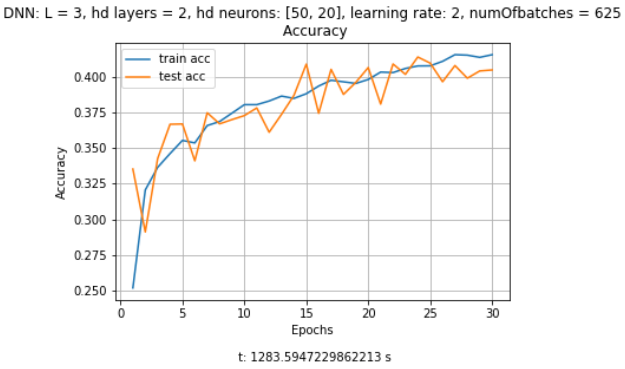
Για 2 hidden layer:

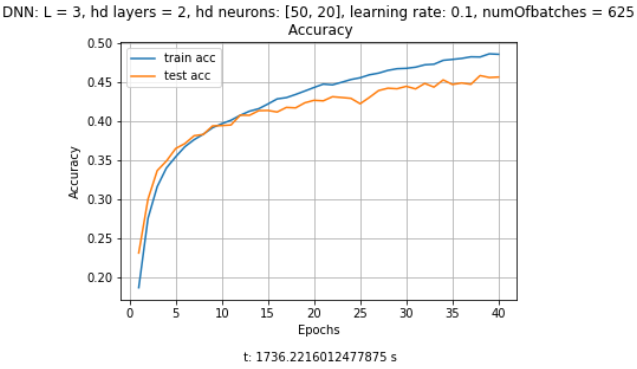
Για 70 και 20 νευρώνες στο πρώτο και δεύτερο hidden layer:

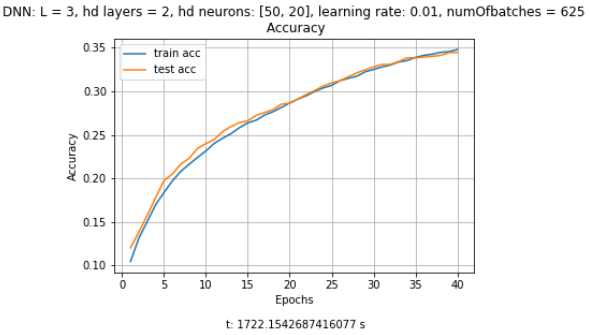




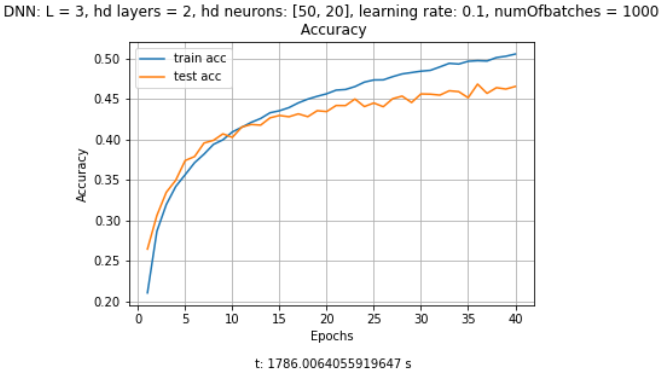
Για 50 και 20 νευρώνες στο πρώτο και δεύτερο hidden layer και για αριθμό mini-batch 625 δοκιμάζουμε διάφορα learning rate:

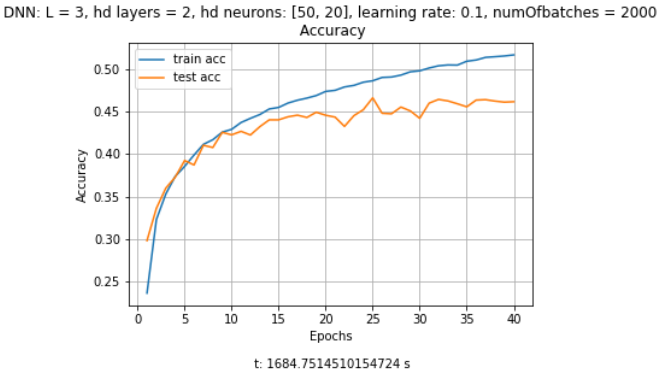




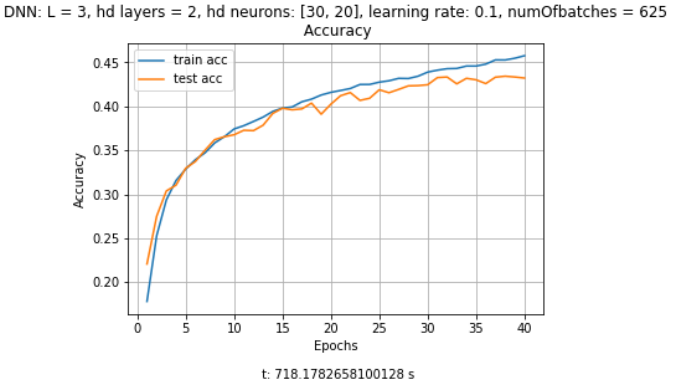


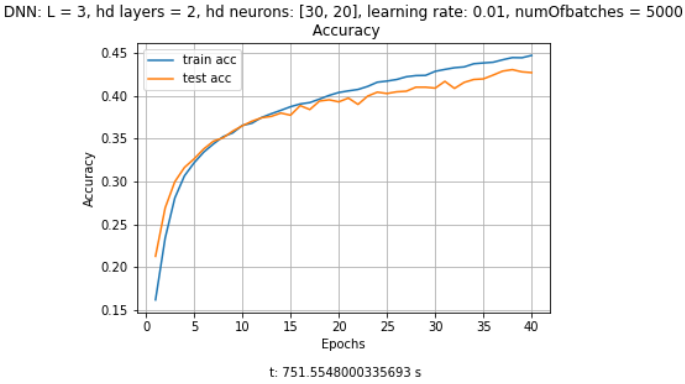
Για 50 και 20 νευρώνες και learning rate = 0.1 βλέπουμε ότι αυξάνοντας τον αριθμό των mini-batch στις ίδιες εποχές έχουμε ίδιο accuracy αλλά μεγαλύτερο gap για περισσότερα mini-batch



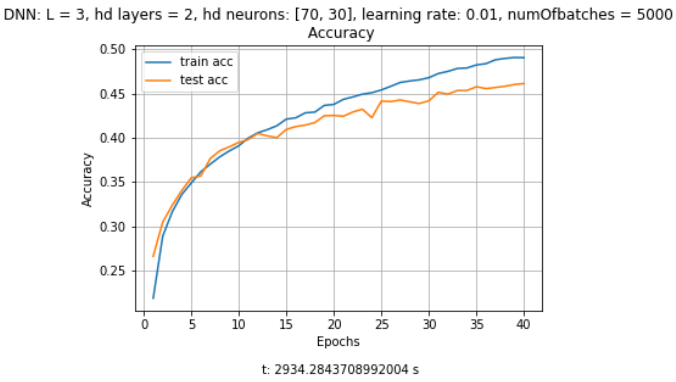


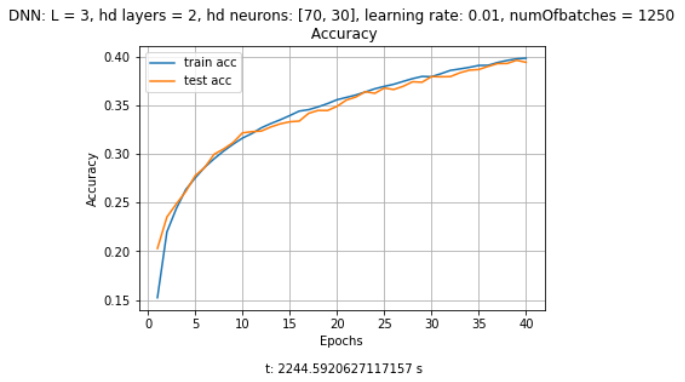
Για 30 και 20 νευρώνες βλέπουμε ότι επιτυγχάνεται παρόμοιο performance για τα ζεύγη {0.1, 625} και {0.01, 5000} :



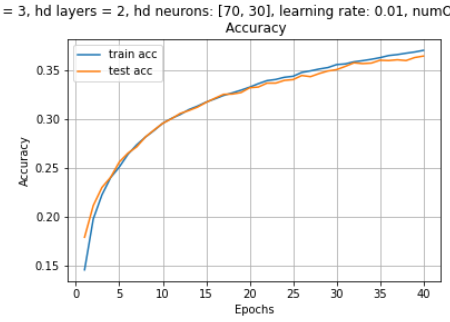


Για 70 και 30 νευρώνες και για learning rate = 0.01:





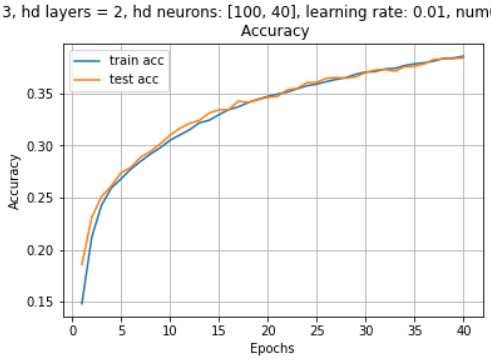
DNN: L = 3, hd layers = 2, hd neurons: [70, 30], learning rate: 0.01, numOfbatches = 625



t: 2126.139003753662 s

Αυξάνουμε τους νευρώνες στα hidden layer στους 100 και 40 νευρώνες και για learning rate = 0.01 και 625 mini-batches προκύπτει παρόμοιο performance με τους 70 και 30 νευρώνες στις 40 εποχές χωρίς κάποια βελτίωση:

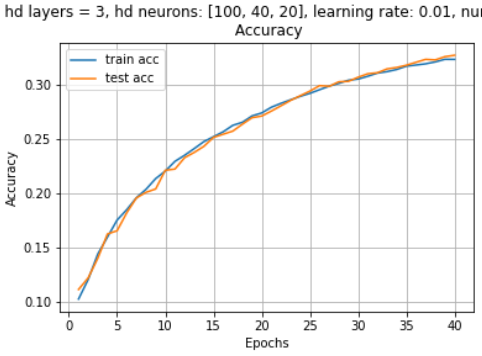
DNN: L = 3, hd layers = 2, hd neurons: [100, 40], learning rate: 0.01, numOfbatches = 625



t: 2894.7861111164093 s

Για 3 hidden layer πιο αργή σύγκλιση:

DNN: L = 4, hd layers = 3, hd neurons: [100, 40, 20], learning rate: 0.01, numOfbatches = 625



t: 3050.790228843689 s

~Παρατηρούμε ότι δεν έχουμε καταφέρει αυξάνοντας την πολυπλοκότητα του νευρωνικού δικτύου να επιτύχουμε καλύτερο accuracy στην ταξινόμηση μας.

~Συγκρίνοντας την απόδοση με τους ταξινομητές k-NN και Nearest Centroid πετυχαίνουμε στο καλύτερο μοντέλο που βρέθηκε παραπάνω περίπου 5-10% καλύτερο accuracy (0.45). Αν χρησιμοποιούσαμε συνελικτικό νευρωνικό δίκτυο μπορεί να επιτυγχάναμε καλύτερη ακρίβεια για το δεδομένο dataset.