Εργασία 1η

Ονοματεπώνυμο : Απόστολος Γιαννουλίδης

ΑΕΜ : 2906

Η υλοποίηση της εργασίας έγινε στην γλώσσα προγραμματισμού Python 3 , και χρησιμοποιήθηκε η βιβλιοθήκη tensorflow . Η εργασία αφορά την εκπαίδευση νευρωνικού δικτιού με την Mnist βάση δεδομένων.

Επιλογή Νευρωνικού δικτύου :

Στην συγκεκριμένη εργασία αποφασίστηκε να γίνει επιλογή ενός απλού νευρωνικού δικτύου όπου η είσοδος του είναι ένα διάνυσμα με 784 στοιχεία (28χ28) ,με ένα κρυφό layer , το οποίο αποτελείται από 256 νευρώνες και τέλος το επίπεδο εξόδου όπου αποτελείται από 10 νευρώνες . Τα επίπεδα είναι πλήρος συνδεμένα μεταξύ τους (Dense) και χρησιμοποιούν “bias” . Επίσης η εσωτερική συνάρτηση των νευρώνων του κρυφού επιπέδου είναι η “relu” , ενώ του επιπέδου εξόδου είναι η “softmax” όπου μετατρέπει την έξοδο σε μορφή πιθανότητας . Κατά την εκπαίδευση του δικτύου χρησιμοποιείται ο “Stochastic Gradient Descent” βελτιστοποιητής με βήμα ίσο με 0.001 .Τέλος κλιμακώνουμε τα δεδομέα έτσι ώστε να βρίσκονται στο διάστιμα [0,1].
(αρχείο με τον κώδικα : NN.py)

Στην εργασία θα παρουσιαστούν οι διαφορές στην απόδοση του δικτύου σε σχέση με την επιλογή του “momentum” κάθε φορά και εν τέλη θα συγκριθεί η απόδοση του σε σχέση με τον κατηγοριοποιητή πλησιέστερου γείτονα .

Αλλαγές στην παράμετρο “momentum” και διαφορά στην απόδοση :

Αρχικά γίνεται επιλογή του **momentum** ίση με 0 ,
Προπονούμε το δίκτυο για 10 εποχές :

```
Epoch 1/10
 - 2s - loss: 2.2477 - accuracy: 0.2170 - val_loss: 2.1109 - val_accuracy: 0.3650
Epoch 2/10
- 2s - loss: 2.0069 - accuracy: 0.4638 - val_loss: 1.8919 - val_accuracy: 0.5502
Epoch 3/10
 - 2s - loss: 1.8090 - accuracy: 0.5938 - val_loss: 1.7016 - val_accuracy: 0.6483
Epoch 4/10
 - 2s - loss: 1.6340 - accuracy: 0.6657 - val_loss: 1.5330 - val_accuracy: 0.7052
Epoch 5/10
 - 2s - loss: 1.4795 - accuracy: 0.7083 - val_loss: 1.3860 - val_accuracy: 0.7424
Epoch 6/10
 - 2s - loss: 1.3456 - accuracy: 0.7382 - val_loss: 1.2600 - val_accuracy: 0.7675
Epoch 7/10
 - 2s - loss: 1.2314 - accuracy: 0.7602 - val_loss: 1.1537 - val_accuracy: 0.7864
Epoch 8/10
 - 2s - loss: 1.1349 - accuracy: 0.7761 - val_loss: 1.0643 - val_accuracy: 0.8001
Epoch 9/10
 - 2s - loss: 1.0536 - accuracy: 0.7897 - val_loss: 0.9892 - val_accuracy: 0.8110
Epoch 10/10
 - 2s - loss: 0.9850 - accuracy: 0.8004 - val_loss: 0.9256 - val_accuracy: 0.8205
```

Σφάλμα και ακρίβεια μετά από τις 10 εποχές : val_loss: 0.9256 , val_accuracy: 0.8205

Συνεχίζουμε με επιλογή του **momentum** ίση με 0.5,
Προπονούμε το δίκτυο για 10 εποχές :

```
Epoch 1/10
 - 2s - loss: 2.0523 - accuracy: 0.3770 - val_loss: 1.8258 - val_accuracy: 0.5715
Epoch 2/10
 - 2s - loss: 1.6591 - accuracy: 0.6435 - val_loss: 1.4824 - val_accuracy: 0.7091
Epoch 3/10
 - 2s - loss: 1.3644 - accuracy: 0.7309 - val_loss: 1.2246 - val_accuracy: 0.7663
Epoch 4/10
 - 2s - loss: 1.1481 - accuracy: 0.7754 - val_loss: 1.0398 - val_accuracy: 0.7999
Epoch 5/10
 - 2s - loss: 0.9936 - accuracy: 0.8026 - val_loss: 0.9088 - val_accuracy: 0.8204
Epoch 6/10
 - 2s - loss: 0.8825 - accuracy: 0.8183 - val_loss: 0.8136 - val_accuracy: 0.8347
Epoch 7/10
 - 2s - loss: 0.8003 - accuracy: 0.8300 - val_loss: 0.7421 - val_accuracy: 0.8442
Epoch 8/10
 - 2s - loss: 0.7374 - accuracy: 0.8388 - val_loss: 0.6865 - val_accuracy: 0.8523
Epoch 9/10
 - 2s - loss: 0.6879 - accuracy: 0.8462 - val_loss: 0.6426 - val_accuracy: 0.8569
Epoch 10/10
 - 2s - loss: 0.6480 - accuracy: 0.8515 - val_loss: 0.6065 - val_accuracy: 0.8636
```

Σφάλμα και ακρίβεια μετά από τις 10 εποχές : val_loss: 0.6065 - val_accuracy: 0.8636
Παρατηρούμε πως με την χρήση momentum έχουμε καλύτερα αποτελέσματα

Συνεχίζουμε αυξάνοντας το **momentum** και θέτουμε την τιμή του ίση με 1,
Προπονούμε το δίκτυο για 10 εποχές :

```
Epoch 1/10
 - 2s - loss: 0.9504 - accuracy: 0.7308 - val_loss: 0.5932 - val_accuracy: 0.8819
Epoch 2/10
 - 2s - loss: 0.7013 - accuracy: 0.8864 - val_loss: 0.7120 - val_accuracy: 0.8839
Epoch 3/10
 - 2s - loss: 0.6913 - accuracy: 0.8876 - val_loss: 0.6563 - val_accuracy: 0.9047
Epoch 4/10
 - 2s - loss: 0.7200 - accuracy: 0.9133 - val_loss: 0.7014 - val_accuracy: 0.9249
Epoch 5/10
 - 2s - loss: 0.8346 - accuracy: 0.9196 - val_loss: 1.0865 - val_accuracy: 0.9176
Epoch 6/10
 - 2s - loss: 0.8863 - accuracy: 0.9374 - val_loss: 0.9990 - val_accuracy: 0.9399
Epoch 7/10
 - 2s - loss: 0.8404 - accuracy: 0.9457 - val_loss: 1.0200 - val_accuracy: 0.9299
Epoch 8/10
 - 2s - loss: 0.6914 - accuracy: 0.9461 - val_loss: 0.8034 - val_accuracy: 0.9399
Epoch 9/10
 - 2s - loss: 0.6691 - accuracy: 0.9409 - val_loss: 0.8644 - val_accuracy: 0.9332
Epoch 10/10
 - 2s - loss: 0.6153 - accuracy: 0.9395 - val_loss: 0.7412 - val_accuracy: 0.9375
```

Σφάλμα και ακρίβεια μετά από τις 10 εποχές : val_loss: 0.7412 - val_accuracy: 0.9375
Παρατηρούμε ότι η ακρίβεια αυξάνεται ακόμα πιο πολύ με την αύξηση του **momentum.**

Συνεχίζουμε αυξάνοντας το **momentum** και θέτουμε την τιμή του ίση με 2,
Προπονούμε το δίκτυο για 10 εποχές :

Epoch 1/10
 - 2s - loss: nan - accuracy: 0.1662 - val_loss: nan - val_accuracy: 0.0980
Epoch 2/10
 - 2s - loss: nan - accuracy: 0.0987 - val_loss: nan - val_accuracy: 0.0980
Epoch 3/10
 - 2s - loss: nan - accuracy: 0.0987 - val_loss: nan - val_accuracy: 0.0980
Epoch 4/10
 - 2s - loss: nan - accuracy: 0.0987 - val_loss: nan - val_accuracy: 0.0980
Epoch 5/10
 - 2s - loss: nan - accuracy: 0.0987 - val_loss: nan - val_accuracy: 0.0980
Epoch 6/10
 - 2s - loss: nan - accuracy: 0.0987 - val_loss: nan - val_accuracy: 0.0980
Epoch 7/10
 - 2s - loss: nan - accuracy: 0.0987 - val_loss: nan - val_accuracy: 0.0980
Epoch 8/10
 - 2s - loss: nan - accuracy: 0.0987 - val_loss: nan - val_accuracy: 0.0980
Epoch 9/10
 - 2s - loss: nan - accuracy: 0.0987 - val_loss: nan - val_accuracy: 0.0980
Epoch 10/10
 - 2s - loss: nan - accuracy: 0.0987 - val_loss: nan - val_accuracy: 0.0980

Σφάλμα και ακρίβεια μετά από τις 10 εποχές : val_loss: nan - val_accuracy: 0.0980

Παρατηρούμε ότι η περεταίρω αύξηση του **momentum** όχι μόνο δεν αυξάνει την ακρίβεια αλλά δεν αφήνει το νευρωνικό να προπονηθεί πάνω στα δεδομένα **.**

Συμπέρασμα , επιλογή χρήσης **momentum** μπορεί να βοηθήσει σημαντικά στο πόσο γρήγορα και αποτελεσματικά θα συγκλίνει το νευρωνικό δίκτυο σε μια καλή κατάσταση , καθώς στο συγκεκριμένο παράδειγμα είχαμε βελτίωση στην ακρίβεια κατά 10% .Παρόλα αυτά χρειάζεται προσοχή και στην επιλογή της σωστής τιμής για την επίτευξη του (τοπίκου) βέλτιστου νευρωνικού δικτύου .
Για την συνέχιση της εργασίας επιλέχθηκε **momentum** ίσο με 0.95 μετά από σύγκριση αυτού με **momentum** ίσο με 1 σε διάστιμα 50 εποχών.

momentum ίσο με 1

Epoch 1/50
 - 2s - loss: 0.9175 - accuracy: 0.7471 - val_loss: 0.6072 - val_accuracy: 0.8827
Epoch 2/50
 - 2s - loss: 0.7022 - accuracy: 0.8870 - val_loss: 0.6935 - val_accuracy: 0.8844
Epoch 3/50
 - 2s - loss: 0.7387 - accuracy: 0.8793 - val_loss: 0.7493 - val_accuracy: 0.9015
Epoch 4/50
 - 2s - loss: 0.9913 - accuracy: 0.9025 - val_loss: 0.8745 - val_accuracy: 0.9204
Epoch 5/50
 - 2s - loss: 1.0284 - accuracy: 0.9222 - val_loss: 1.0755 - val_accuracy: 0.9270
.
.
.
Epoch 44/50
 - 2s - loss: 0.2525 - accuracy: 0.9749 - val_loss: 2.7787 - val_accuracy: 0.9514
Epoch 45/50
 - 2s - loss: 0.2486 - accuracy: 0.9738 - val_loss: 2.8691 - val_accuracy: 0.9502
Epoch 46/50
 - 2s - loss: 0.2587 - accuracy: 0.9730 - val_loss: 2.9769 - val_accuracy: 0.9486
Epoch 47/50
 - 2s - loss: 0.2569 - accuracy: 0.9748 - val_loss: 3.0269 - val_accuracy: 0.9503
Epoch 48/50
 - 2s - loss: 0.2638 - accuracy: 0.9742 - val_loss: 3.2057 - val_accuracy: 0.9480
Epoch 49/50
 - 2s - loss: 0.2579 - accuracy: 0.9738 - val_loss: 3.2249 - val_accuracy: 0.9504
Epoch 50/50
 - 2s - loss: 0.2494 - accuracy: 0.9758 - val_loss: 3.3209 - val_accuracy: 0.9511

momentum ίσο με 0.95

Epoch 1/50
 - 2s - loss: 1.2924 - accuracy: 0.6717 - val_loss: 0.6473 - val_accuracy: 0.8532
Epoch 2/50
 - 2s - loss: 0.5506 - accuracy: 0.8637 - val_loss: 0.4467 - val_accuracy: 0.8873
Epoch 3/50
 - 2s - loss: 0.4336 - accuracy: 0.8853 - val_loss: 0.3813 - val_accuracy: 0.8987
Epoch 4/50
 - 2s - loss: 0.3836 - accuracy: 0.8957 - val_loss: 0.3467 - val_accuracy: 0.9056
Epoch 5/50
 - 2s - loss: 0.3535 - accuracy: 0.9026 - val_loss: 0.3245 - val_accuracy: 0.9106
.
.
.
Epoch 44/50
 - 2s - loss: 0.1425 - accuracy: 0.9606 - val_loss: 0.1459 - val_accuracy: 0.9580
Epoch 45/50
 - 2s - loss: 0.1405 - accuracy: 0.9613 - val_loss: 0.1434 - val_accuracy: 0.9580
Epoch 46/50
 - 2s - loss: 0.1385 - accuracy: 0.9615 - val_loss: 0.1421 - val_accuracy: 0.9589
Epoch 47/50
 - 2s - loss: 0.1364 - accuracy: 0.9625 - val_loss: 0.1415 - val_accuracy: 0.9586
Epoch 48/50
 - 2s - loss: 0.1346 - accuracy: 0.9629 - val_loss: 0.1394 - val_accuracy: 0.9599
Epoch 49/50
 - 2s - loss: 0.1329 - accuracy: 0.9635 - val_loss: 0.1377 - val_accuracy: 0.9601
Epoch 50/50
 - 2s - loss: 0.1310 - accuracy: 0.9639 - val_loss: 0.1363 - val_accuracy: 0.9605

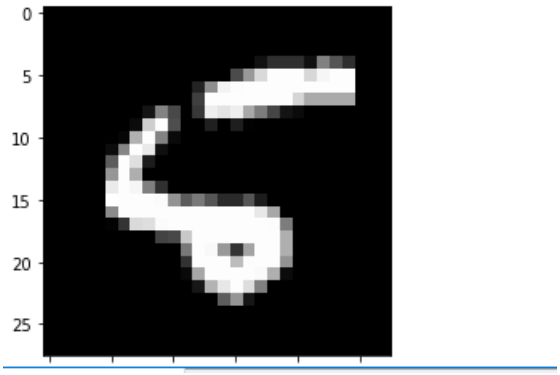
Το νευρωνικό δίκτυο είναι αποθηκευμένο στο αρχείο my_modeel_1.

Παραδείγματα ορθής και εσφαλμένης πρόβλεψης :

Στο αρχείο failprediction.py παρουσιάζεται μια λανθασμένη πρόβλεψη του δικτύου .

Παράδειγμα εκτέλεσης :

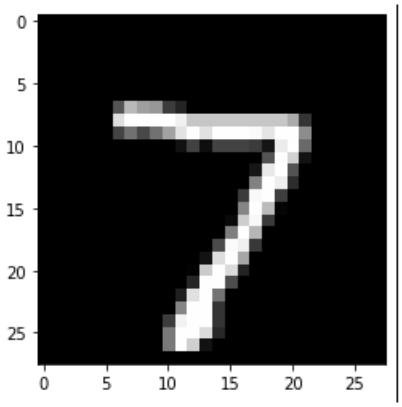
```
Prediction :  6
Real value :  5
```



Στο αρχείο correctprediction.py παρουσιάζεται μια σωστή πρόβλεψη του δικτύου .

Παράδειγμα εκτέλεσης :

```
Prediction :  7
Real value :  7
```



Σύγκριση δικτύου με κατηγοριοποιητή πλησιέστερου γείτονα με έναν και τρείς γείτονες :

Αφότου έγινε η επιλογή του **momentum** ίσο με 0.95 θα γίνει σύγκριση του δικτύου με τον αλγόριθμο του κατηγοριοποιητή πλησιέστερου γείτονα με έναν και τρείς γείτονες .

Από την ενδιάμεση εργασία έχουμε τα εξής στοιχεία για αυτούς τους δύο αλγορίθμους :

Εφαρμογή σε όλη την βάση δεδομένων μετά από εφαρμογή αλγορίθμου PCA για την μείωση των διαστάσεων των δεδομένων , λόγο ύπαρξης περιορισμένης υπολογιστικής δύναμης .
Σύγκριση με το νευρωνικό δίκτυο σε όλη την βάση δεδομένων χωρίς μείωση των διαστάσεν.

κατηγοριοποιητής πλησιέστερου γείτονα με 3 γείτονες :

Train acc = 0.9853666666666666
Test acc = 0.9687
Time : 89.62629370000013

κατηγοριοποιητής πλησιέστερου γείτονα με 1 γείτονα :

Train acc = 1.0
Test acc = 0.9638
Time : 13.796097

Τα αποτελέσματα μας δείχνουν πως οι δύο προσεγγίσεις είναι ισοδύναμες όσον αφορά την ακρίβεια , με το νευρωνικό δίκτυο να είναι πιο αργό (χρόνος εκτέλεσης:79.01678739999988) από τον κατηγοριοποιητή πλησιέστερου γείτονα με 1 γείτονα αλλα πιο γρήγορο από τον κατηγοριοποιητή πλησιέστερου γείτονα με 3 γείτονες.

Επιλογή του 1/7 της βάσης για αλγόριθμο του πλησιέστερου γείτονα:

Διαστάσεις εισόδων : (10000, 784) (10000,) (2000, 784) (2000,)

κατηγοριοποιητής πλησιέστερου γείτονα με 3 γείτωνες :

Train acc = 0.9755
acc = 0.9235
Time : 222.16224620000003

κατηγοριοποιητής πλησιέστερου γείτονα με 1 γείτωνα :

Train acc = 1.0
Test acc = 0.9225
Time : 142.87078640000001

Δεδομένου των υπολογιστικών πόρων του συστημάτος αντιλαμβανόμαστε πως η υλοποίηση του αλγορίθμου πλησιέστερου γείτονα σε όλα τα δεδομένα της βάσης χωρίς να γίνει μείωση της διάστασης της εισόδου είναι σχεδόν αδύνατο , κάτι το οποίο γίνεται με την υλοποίηση του νευρωνικού δικτύου.