Νευρωνικά Δίκτυα - Βαθιά Μάθηση

Τμήμα Πληροφορικής ΑΠΘ 7ο εξάμηνο

Εργασία 1 - Multilayer Perceptron

Αλεξία Νταντουρή ΑΕΜ: 3871

1η Εργασία

Να γραφεί πρόγραμμα σε οποιαδήποτε γλώσσα προγραμματισμού το οποίο να υλοποιεί ένα **νευρωνικό δίκτυο πολυστρωματικού perceptron** (το δίκτυο μπορεί να είναι πλήρως συνδεδεμένο ή συνελικτικό ή συνδυασμός) που θα εκπαιδεύεται με τον αλγόριθμο back-propagation. Το ΝΝ αυτό θα εκπαιδευτεί για να επιλύει οποιοδήποτε πρόβλημα κατηγοριοποίησης πολλών κλάσεων ΕΚΤΟΣ MNIST.

Σημείωση

Η ανάλυση του κώδικα και των αποτελεσμάτων γίνεται μέσα στο Jupyter Notebook, αλλά παρέχω κι εδώ, επιγραμματικά, κάποια σημαντικά σημεία και τέλος, γίνεται σύγκριση των KNN, NCC και MLP.

Δημιουργία μοντέλου

Χρησιμοποίησα τη συνάρτηση **MLPClassifier** της sklearn για τη δημιουργία πλήρως συνδεδεμένου νευρωνικού δικτύου πολυστρωματικού perceptron.

Data preprocessing

Αφαίρεσα το **χρώμα** από τις εικόνες και χρησιμοποίησα **PCA** (Principal Component Analysis) για να δω ποια επεξεργασμένα δεδομένα είναι καλύτερα για την εκπαίδευση του MLP (Multilayer perceptron).

	No of features	No of neurons	No of epochs	Time in seconds	Train accuracy	Test accuracy
Grayscale images without PCA	1024	(100,50)	60	206	48%	41%
Grayscale images with PCA	76	(400,100)	23	22	61%	48%
Original images without PCA	3072	(100,50)	40	241	53%	49%
Original images with PCA	99	(400,100)	18	79	**67%**	**54%**

Όπως φαίνεται από τον παραπάνω πίνακα, η καλύτερη επιλογή είναι η χρήση των αρχικών εικόνων (με χρώμα) και η εφαρμογή PCA για τη μείωση των διαστάσεων.

Ο παραπάνω συνδυασμός μας δίνει τα καλύτερα αποτελέσματα. Συγκεκριμένα, δίνει μεγαλύτερη ακρίβεια σε συνδυασμό με γρήγορο χρόνο εκτέλεσης.

Hyperparameter tuning

Χρησιμοποίησα τη συνάρτηση **RandomSearchCV** της sklearn για να βρω υπερπαραμέτρους για τη βελτίωση της ακρίβειας του μοντέλου.

```
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV

mlp = MLPClassifier(max_iter=300, early_stopping=True)

parameter_grid = {
    'hidden_layer_sizes': [(400,100), (500), (600,200)],
    'activation': ['logistic', 'tanh', 'relu'],
    'solver': ['sgd', 'adam'],
    'alpha': [0.0001, 0.001],
    'learning_rate': ['constant','adaptive'],
}

# grid_search = GridSearchCV(mlp, parameter_grid, cv=5)
grid_search = RandomizedSearchCV(mlp, parameter_grid, random_state=2, n_iter=5, cv=5, n_jobs=-1)

start = timer()

grid_search.fit(train_pca, y_train.ravel())

end = timer()
print("Time: ", end - start) # time in seconds

print("Best params:")
print(grid_search.best_params_)
```

```
Time:  2740.676056788001
Best params:
{'solver': 'adam', 'learning_rate': 'adaptive', 'hidden_layer_sizes': (600, 200), 'alpha': 0.001, 'activation': 'relu'}
```

Πειραματίστηκα, στη συνέχεια, με μοντέλα με διαφορετικό πλήθος νευρώνων και διαφορετικό αριθμό επιπέδων και δοκίμασα διαφορετικές τιμές στις υπερπαραμέτρους για να δω πως επηρεαζόταν η ακρίβεια του μοντέλου.

Εν τέλει, το μοντέλο που διάλεξα είναι το μοντέλο που πρότεινε η RandomSearchCV, το οποίο έδωσε τα παρακάτω αποτελέσματα:

```
Time:  52.91149928399997
Number of iterations with early stopping: 20
Train accuracy: 0.77094
Test accuracy: 0.5603
```

Convolutional neural network with Keras

Τέλος, αποφάσισα να δοκιμάσω να χρησιμοποιήσω ένα συνελικτικό νευρωνικό δίκτυο πολυστρωματικού perceptron με τη χρήση της βιβλιοθήκης Keras, καθώς τα Συνελικτικά Νευρωνικά Δίκτυα (CNNs) θεωρούνται πολύ αποτελεσματικά στην ταξινόμηση εικόνων.

Πράγματι, χωρίς πολλούς πειραματισμούς στην αρχιτεκτονική του CNN, το νευρωνικό έπιασε τις παρακάτω ακρίβειες:

```
Train accuracy : 0.8401600122451782
Test accuracy : 0.718999981880188
```

Comparing Results

Ο κατηγοριοποιητής πλησιέστερου γείτονα (KNN) δεν έχει φάση εκπαίδευσης, γι' αυτό στις στήλες των KNN έβαλα τους χρόνους πρόβλεψης των train και test sets.

	NCC	KNN 1	KNN 3	FC MLP	CNN
Training time	0.7	574 (train set) 129 (test set)	556 (train set) 109 (test set)	102	745
Train accuracy	**27%**	**100%**	**58%**	**76%**	**84%**
Test accuracy	**28%**	**35%**	**33%**	**55%**	**72%**

Όπως φαίνεται από τον παραπάνω πίνακα, το συνελικτικό νευρωνικό δίκτυο **(CNN)** δίνει τα καλύτερα αποτελέσματα όσον αφορά την ακρίβεια. Το πλήρως συνδεδεμένο νευρωνικό δίκτυο **(FC MLP)** είναι το 2ο καλύτερο και ακολουθεί ο κατηγοριοποιητής πλησιέστερου γείτονα με 1 γείτονα **(KNN 1)** κι έπειτα με 3 γείτονες **(KNN 3)**. Τέλος, τα χειρότερα αποτελέσματα δίνει ο κατηγοριοποιητής πλησιέστερου κέντρου κλάσης **(NCC)**.