Ψηφιακή Επεξεργασία και Ανάλυση Εικόνας

Εργαστηριακές Ασκήσεις – Μέρος Β
Θέμα 1: Κατηγοριοποίηση Εικόνων

ΚΟΤΣΙΝΗΣ ΔΗΜΗΤΡΙΟΣ
1059482

Περιεχόμενα

Περίληψη

Για την κατηγοριοποίηση εικόνων και για την εξαγωγή χαρακτηριστικών τους έχουν αναπτυχθεί πολλοί μέθοδοι. Στην εργασία αυτή θα κατηγοριοποιήσουμε με την χρήση συνελικτικών νευρωνικών δικτύων (CNN) και Histogram of Oriented Gradients και Support Vector Machine τις εικόνες τις βιβλιοθήκης MNIST (δεδομένα: http://yann.lecun.com/exdb/mnist/) που αναπαριστούν αριθμούς από το μηδέν μέχρι το εννιά. Θα αναλύσουμε τις τεχνικές που χρησιμοποιήσαμε και μέσω του confusion matrix θα συγκρίνουμε στο τέλος τις δύο μεθόδους.

Για την κατηγοριοποίηση θα χρησιμοποιήσουμε το περιβάλλον της MATLAB και συγκεκριμένα την βιβλιοθήκη Deep Learning Toolbox.

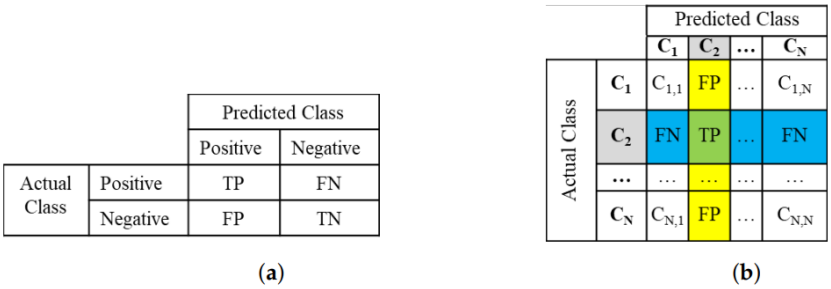
Βιβλιοθήκη MNIST

Η βιβλιοθήκη MNIST περιέχει 60000 δεδομένα εκπαίδευσης και 10000 δεδομένα δοκιμής. Κάθε δεδομένο περιέχει μία εικόνα που αναπαριστά έναν αριθμό από το μηδέν μέχρι το εννιά διάστασης 28x28 και εύρος τιμών pixel [0, 1]. Η γνώση του κάθε αριθμού της εικόνας βρίσκεται σε ένα διάνυσμα label.

Από την ιστοσελίδα, που κατεβάσαμε τις εικόνες (http://yann.lecun.com/exdb/mnist/) δημιουργούμε ένα αρχείο που βρίσκεται στο Παράρτημα Γ το οποίο ανοίγει αυτά τα αρχεία και εξάγει επίσης τα χαρακτηριστικά HOG για patch 8x8 και 4x4 για το μέρος Β.

Confusion Matrix

Το Comfussion matrix είναι ένας τετραγωνικός πίνακας όπως φαίνεται παρακάτω και δείχνει την κατηγοριοποίηση των δεδομένων δοκιμής, δηλαδή που τοποθέτησε την εικόνα το μοντέλο (Predicted Class, στήλες) και που έπρεπε να το τοποθετήσει.



		Predicted Class	
		Positive	Negative
Actual Class	Positive	TP	FN
	Negative	FP	TN

(a)

		Predicted Class			
		C_1	C_2	…	C_N
Actual Class	C_1	$C_{1,1}$	FP	…	$C_{1,N}$
	C_2	FN	TP	…	FN
	…	…	…	…	…
	C_N	$C_{N,1}$	FP	…	$C_{N,N}$

(b)

Εικόνα Ε.1: (a) Παράδειγμα confusion matrix για κατηγοριοποίηση δύο κλάσεων και (b) κατηγοριοποίηση N κλάσεων.[8]

Έτσι με τον πίνακα αυτόν μπορούμε να ορίσουμε τις εξής μετρικές [9]:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

$$Sensetivity = \frac{TP}{TP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$F\_Score = \frac{2 * Precision * Sensetivity}{Precision + Sensetivity}$$

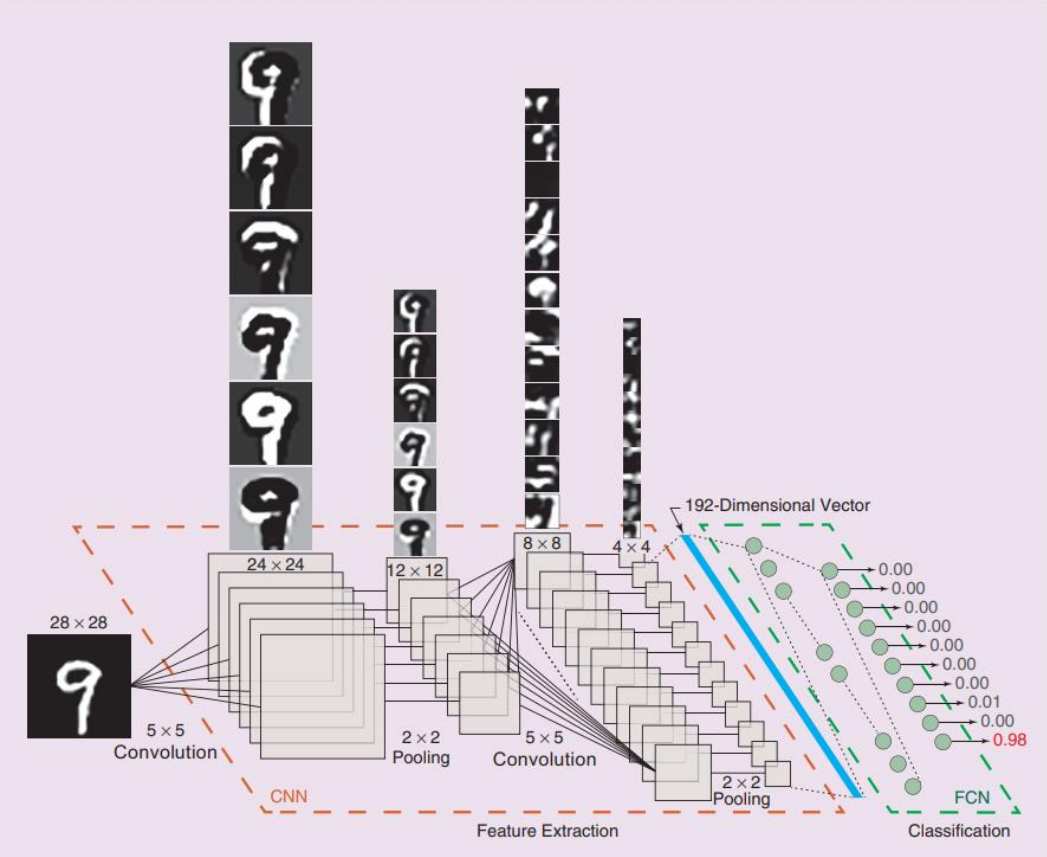
Όταν ο πίνακας είναι ν-δισδιάστατος υπολογίζουμε τις μετρικές της κάθε κλάσης και μετά υπολογίζουμε τον μέσο όρο αυτόν. (Παράρτημα Α, Precision.m, Sensetivity.m)

Μέρος Α: Κατηγοριοποίηση Εικόνων με χρήση Συνελικτικών Νευρωνικών Δικτύων (CNN)

Η αρχιτεκτονική των συνελικτικών νευρωνικών δικτύων ήταν η πρώτη ικανή μέθοδος κατηγοριοποίηση εικόνων. Σε αυτό το μέρος θα αναλύσουμε συνοπτικά την δομή, την λειτουργία και την εκπαίδευση ενός συνελικτικό νευρωνικού δικτύου και στην συνέχεια θα θέσουμε και θα σχολιάσουμε τα αποτελέσματα της άσκησης.

Δομή ενός CNN

Τα CNN, όπως αναφέρει και το όνομά τους, σε κάθε επίπεδο γίνεται συνέλιξη της εικόνας με μάσκες ή αλλιώς kernels οι οποίες εξάγουν χαρακτηριστικά της εικόνας και μειώνουν τις διαστάσεις της, όπως φαίνεται και στην παρακάτω εικόνα.



Εικόνα Α.1: Παράδειγμα αρχιτεκτονικής CNN.[1]

Στην εικόνα αυτή βλέπουμε ότι η επιλεγμένη εικόνα από την βιβλιοθήκη MNIST που αναπαριστά το νούμερο 9, στο πρώτο επίπεδο γίνεται συνέλιξή της 6 φορές με kernels διάστασης 5x5, στην συνέχεια γίνεται επιλογή της μέγιστης τιμής ή της μέσης τιμής ανά παράθυρο διάστασης 2x2 (pooling) και μετά από αυτήν την ενέργεια μπαίνει έχουμε μία συνάρτηση ενεργοποίησης. Αυτή η διαδικασία συνεχίζεται τόσες φορές ώστε να καταλήξουμε στις διαστάσεις που επιθυμούμε.

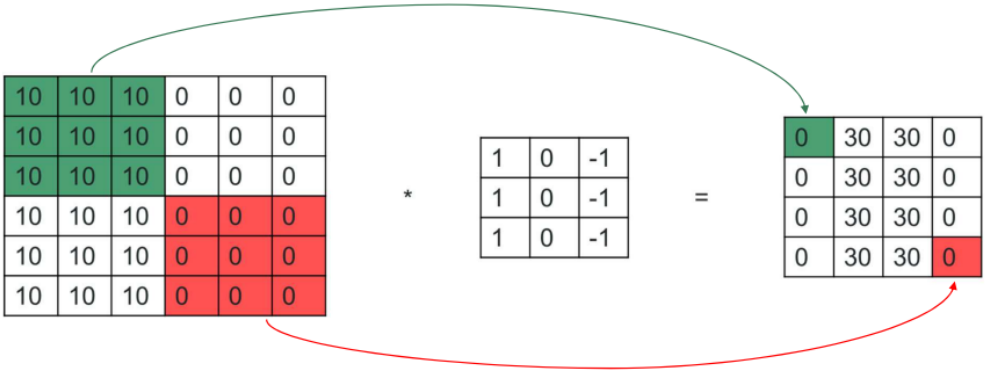
Στην έξοδο ενός CNN προκύπτει ένα διάνυσμα το οποίο μπαίνει σαν είσοδο σε μία απλή αρχιτεκτονική νευρωνικού δικτύου (FCN) που έχει σαν έξοδο την κατηγοριοποίηση των εικόνων. Παρόμοια αρχιτεκτονική θα ακολουθήσουμε και στην άσκηση.

Συνέλιξη εικόνας με kernel

Ο μαθηματικός τύπος της συνέλιξης [6] μίας εικόνας a(x, y) και ενός kernel w διαστάσεων l, k είναι:

$$w * a(x,y) = \sum_{l}\sum_{k} w_{l,k} a_{x-l,y-k}$$

Ουσιαστικά αυτό που γίνεται φαίνεται στην παρακάτω εικόνα:



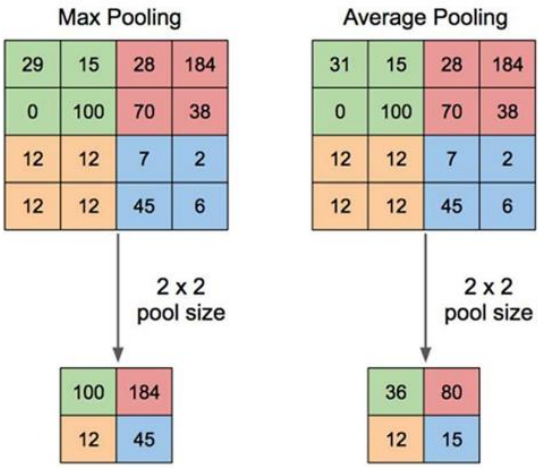
Εικόνα Α.2: Παράδειγμα συνέλιξης. [7]

Το kernel τοποθετείται στο κέντρο ενός παραθύρου της εικόνας και πολλαπλασιάζει στοιχείο προς στοιχείο, τα αθροίζει και τοποθετεί το αποτέλεσμα στο κέντρο του παραθύρου και αυτό συνεχίζει και για τα υπόλοιπα κέντρα.

Στην συνέλιξη ενός CNN στην διαδικασία αυτή ορίζουμε και δύο μεταβλητές την Stride και την Padding. Το Stride είναι ανά πόσα κέντρα το kernel κινείται προς την οριζόντια γραμμή της εικόνας και το Padding είναι με πόσα μηδενικά γεμίζουμε την περιφέρεια της εικόνας. Για αυτήν την άσκηση θα ορίσουμε Stride ίσον με ένα και Padding ίσον με το μηδέν.

Pooling

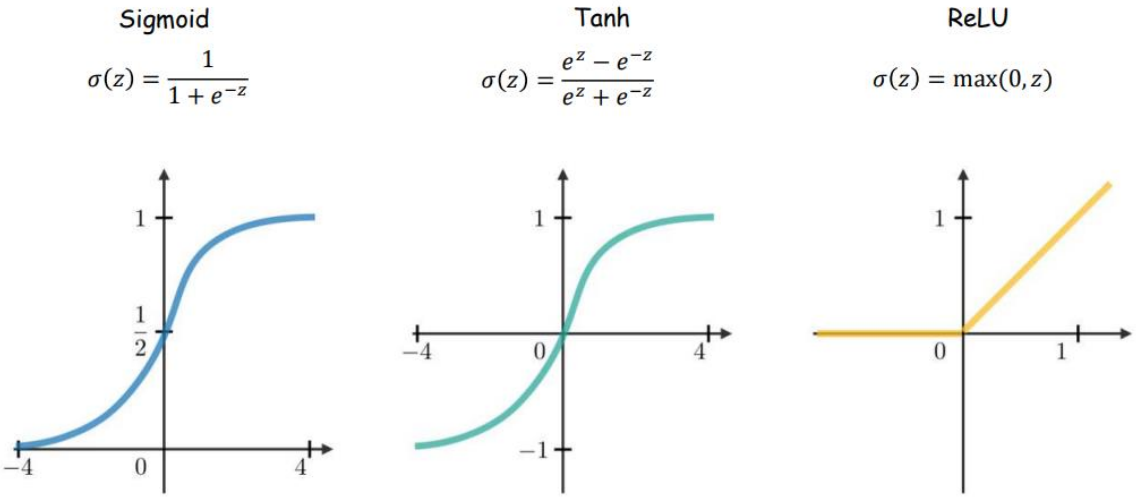
Υπάρχουν δύο είδη pooling, το average που υπολογίζει την μέση τιμή του παραθύρου που ορίζουμε και το maximum που βρίσκει την μεγαλύτερη τιμή που έχει το παράθυρο. Παρακάτω βρίσκεται ένα παράδειγμα με τις δύο μεθόδους.



Εικόνα Α.2: Παράδειγμα pooling. [7]

Συναρτήσεις Ενεργοποίησης

Οι συναρτήσεις ενεργοποίησης είναι συναρτήσεις οι οποίες μπαίνουν πριν ή μετά το pooling για να κανονικοποιήσουν τα δεδομένα ή να τα φιλτράρουν. Παρακάτω υποδεικνύονται ορισμένες από τις πιο συχνές συναρτήσεις ενεργοποιήσεις.



Εικόνα Α.3: Δείγμα συναρτήσεων ενεργοποίησης. [7]

Για την άσκηση θα χρησιμοποιήσουμε την ReLU.

Εκπαίδευση

Για την εκπαίδευση θα πρέπει επιλέξουμε την συνάρτηση κόστους που θα ελαχιστοποιήσουμε:

$$F(X, W) = \frac{1}{|D|} \sum_{i=1}^{|D|} f_i(W)$$

Όπου το $|D|$ το πλήθος των batches.

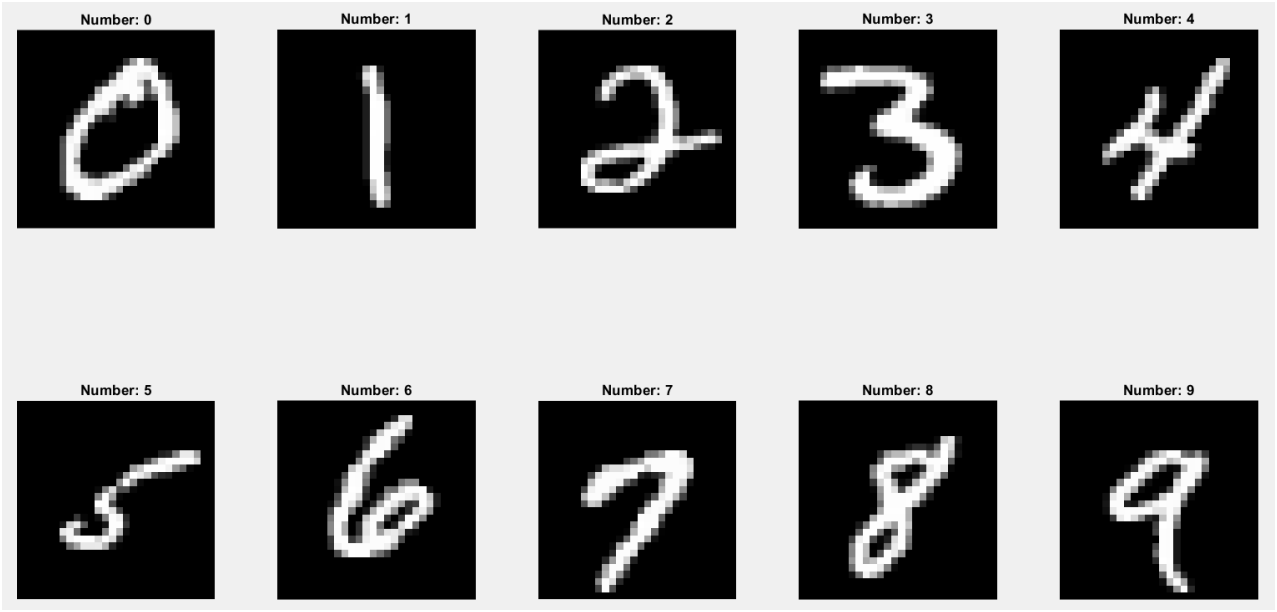
Για την ελαχιστοποίηση θα χρησιμοποιήσουμε τον Stochastic Gradients Descent:

$$W_{t+1} = W_t - \mu \nabla F(X, W)$$

Όπου το μ είναι ο ρυθμός εκπαίδευσης.

Αποτελέσματα Άσκησης

Στην παρακάτω εικόνα βλέπουμε μία εικόνα από κάθε κλάση της βιβλιοθήκης MNIST από το σύνολο δεδομένων εκπαίδευσης.

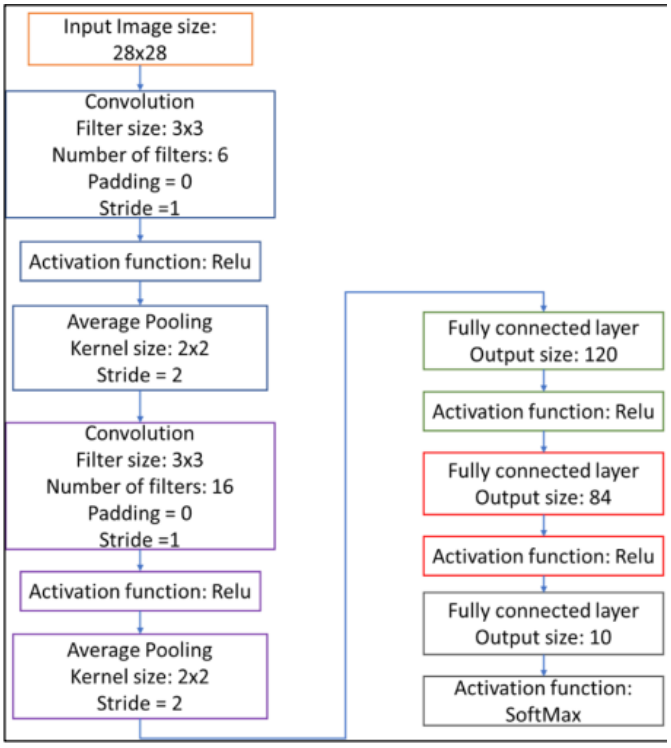


Εικόνα Α.4: Ένα δείγμα από κάθε κατηγορία της βιβλιοθήκης MNIST.

```
%% Απεικόνιση δεδομένων από κάθε κλάση (α ερώτημα)
classes = Inf * ones(1, 10); % Ποιά νούμερα έχουν τοποθετηθεί στο images
images = zeros(28, 28, 10); % Αποθήκευσει των εικόνων απεικόνισης
imgs_train = extractdata(XTrain); % Εξαγωγή των δεδομένων από το dlarray
labels = YTrain; % Οι κατηγορίες των εικονών
i = 1; % μετρητής
while 1
    number = int16(labels(i)); % Αριθμός
    % Έλεγχος αν έχουν περάσει όλες οι κατηγορίες
    if sum(ismember(classes, Inf)) == 0
        break;
    end
    img_num = imgs_train(:, :, 1, i); % Εικόνα αριθμού
    images(:, :, number) = img_num; % Αποθήκευση αριθμού
    classes(number) = number; % Αποθήκευση ότι πέρασε αυτός ο αριθμός
    i = i+1; % Αύξηση μετρητή κατά ένα
end
% Απεικόνιση δεδομένων
figure;
for count = 1 : 10
    subplot(2, 5, count);
    imshow(images(:, :, count))
    title("Number: " + string(count-1));
    count = count + 1;
end
```

Εικόνα Α.5: Κώδικας απεικόνισης των εικόνων.(Παράρτημα Α, main.m)

Η αρχιτεκτονική του συνελικτικού νευρωνικού δικτύου φαίνεται στην παρακάτω εικόνα όπως στο φυλλάδιο των ασκήσεων και ορίζουμε το κάθε επίπεδο του δικτύου μέσα σε μία λίστα στην MATLAB.



Εικόνα Α.6: Αρχιτεκτονική Συνελικτικού Δικτύου.

```
% Ορισμός επιπέδων
layers = [
    imageInputLayer([28 28 1])
    convolution2dLayer(3,6,'Padding', 0, 'Stride', 1)
    reluLayer
    averagePooling2dLayer(2,'Stride', 2)
    convolution2dLayer(3,16,'Padding', 0, 'Stride', 1)
    reluLayer
    averagePooling2dLayer(2,'Stride', 2)
    fullyConnectedLayer(120)
    reluLayer
    fullyConnectedLayer(84)
    reluLayer
    fullyConnectedLayer(10)
    softmaxLayer
    classificationLayer];
```

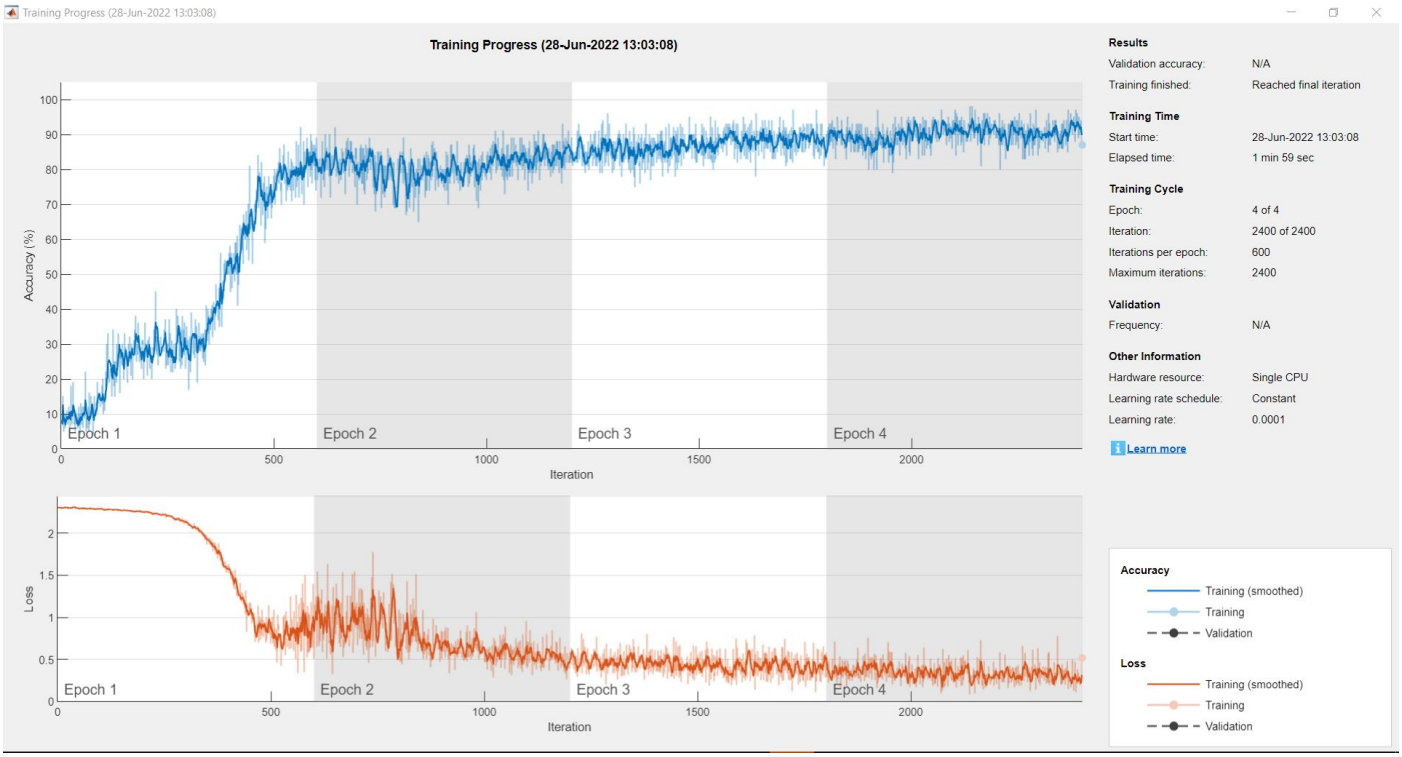
Εικόνα Α.7: Αρχιτεκτονική Συνελικτικού Δικτύου στην MATLAB. .(Παράρτημα Α, main.m)

Στο δίκτυο επίσης θα θέσουμε κάποια χαρακτηριστικά της εκπαίδευσης. Για την σύγκλιση στην βέλτιστη λύση θα χρησιμοποιήσουμε την μέθοδο Stochastic Gradients Descent με συνάρτηση κόστους την cross entropy, ρυθμό εκπαίδευσης 1e-4, εποχές 4 και μέγεθός batch 100, όπως φαίνεται και παρακάτω στον κώδικα της MATLAB.

```
% Τοποθέτηση χαρακτηριστικών του cnn
options = trainingOptions("sgdm", ...
    'Momentum', 1, ...
    'InitialLearnRate',1e-4, ...
    'Shuffle','every-epoch', ...
    'MaxEpochs', 4, ...
    'MiniBatchSize', 100, ...
    'Verbose', false, ...
    'Plots', "training-progress");
% Εκπαίδευση CNN
net = trainNetwork(XTrain,YTrain,layers,options);
```

Εικόνα Α.8: Επιλογές εκπαίδευσης δικτύου. .(Παράρτημα Α, main.m)

Όπως αναγράφεται στην εικόνα Α.8 η συνάρτηση που θα χρησιμοποιήσουμε για την εκπαίδευση του δικτύου είναι η *net=trainNetwork(XTrain,YTrain,layers,options)*, η οποία παίρνει ως εισόδους τα δεδομένα εκπαίδευσης (εικόνες της βιβλιοθήκης MNIST και την επικεφαλίδα της κάθε εικόνας), την αρχιτεκτονική και τις επιλογές εκπαίδευσης του δικτύου που ορίσαμε παραπάνω. Τα αποτελέσματα που παίρνουμε είναι τα εξής.



Εικόνα Α.9: Γραφικές της ακρίβειας του μοντέλου και την συνάρτηση κόστους σε συνάρτηση των επαναλήψεων.

Στην παραπάνω εικόνα βλέπουμε τις γραφικές παραστάσεις της ακρίβειας του μοντέλου και της συνάρτηση κόστους σε συνάρτηση των επαναλήψεων. Οι επαναλήψεις που είχαμε για τις 4 εποχές ήταν 2400.

Με το τέλος της εκπαίδευσης, κατηγοριοποιούμε τα δεδομένα δοκιμής και η ακρίβεια του μοντέλου και η τιμή της συνάρτησης κόστους για αυτήν την περίπτωση είναι τα εξής:

Ακρίβεια: 0.9356

Συνάρτηση Κόστους: 0.0865

```
58    %% Κατηγοριοποίηση των δεδομένων και ποσοστό ακρίβειας του μοντέλου
59 -  YPred = classify(net,XTest); % Κατηγοριοποίηση των δεδομένων
60 -  accuracy = sum(YPred == YTest)/numel(YTest) % Ακρίβεια μοντέλου
61 -  YPred = string(YPred);
62 -  YPred = double(YPred);
63 -  YTest = string(YTest);
64 -  YTest = double(YTest);
65 -  loss = crossentropy(YPred,YTest) % Συνάρτηση κόστου
```

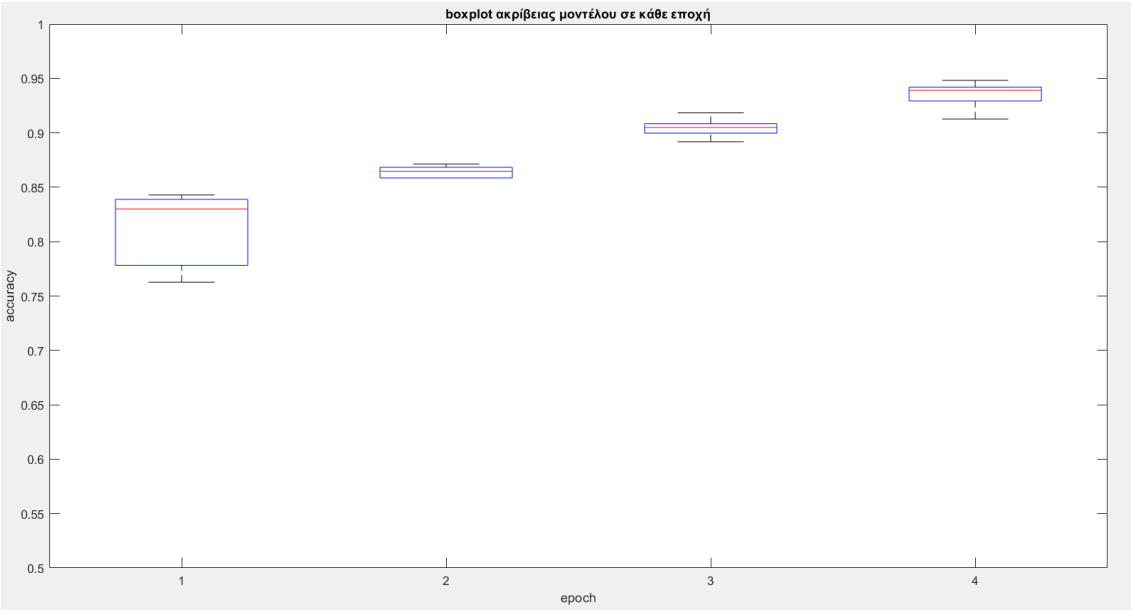
Εικόνα Α.10: Κατηγοριοποίηση δεδομένων δοκιμής.(Παράρτημα Α, main.m)

Τώρα σε κάθε εποχή κατηγοριοποιούμε τα δεδομένα δοκιμής και βρίσκουμε την ακρίβεια του μοντέλου και την τιμή της συνάρτησης κόστους. Επειδή όμως το μοντέλο για την εκπαίδευση παίρνει κάθε φορά διαφορετικά δεδομένα και έχει διαφορετικά βάρη θα έχουμε κάθε φορά διαφορετική ακρίβεια και τιμή συνάρτηση κόστους. Οπότε θα τρέξουμε το πρόγραμμα 5 φορές για κάθε εποχή και θα κρατήσουμε τις τιμές της ακρίβειας και της συνάρτηση κόστους. Παρακάτω φαίνεται ο πίνακας με τις τιμές αυτές για κάθε εποχή.

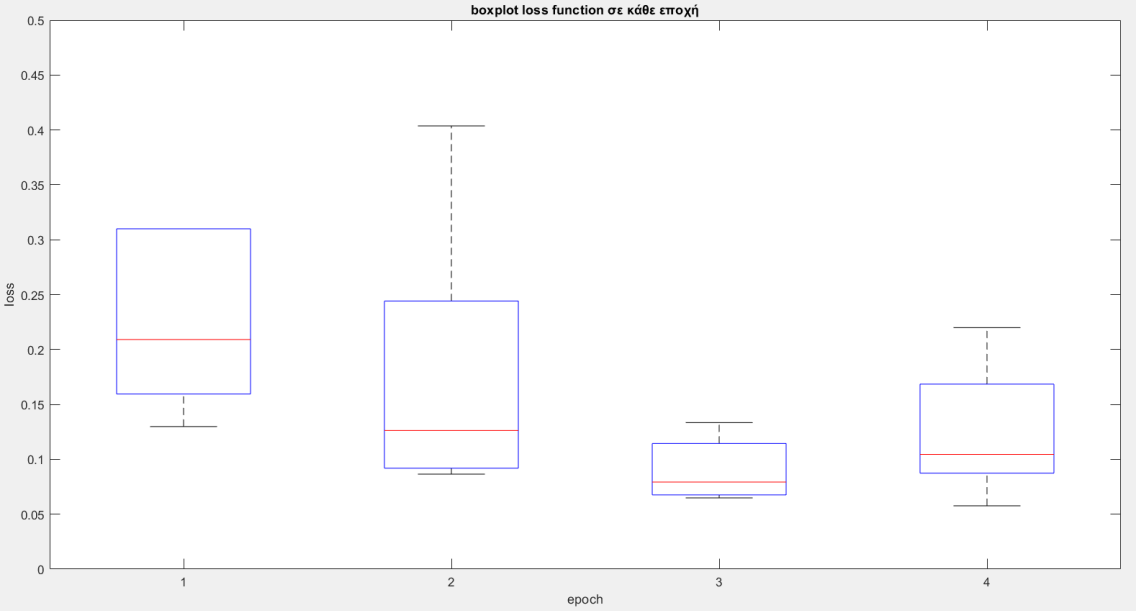
Εποχή 1	Ακρίβεια	Τιμή Συνάρτηση Κόστους
Επανάληψη 1η	0.7834	0.2199
Επανάληψη 2η	0.8300	0.1298
Επανάληψη 3η	0.7627	0.5803
Επανάληψη 4η	0.8375	0.1694
Επανάληψη 5η	0.8429	0.2091
Εποχή 2	Ακρίβεια	Τιμή Συνάρτηση Κόστους
Επανάληψη 1η	0.8674	0.4037
Επανάληψη 2η	0.8714	0.0865
Επανάληψη 3η	0.8648	0.1910
Επανάληψη 4η	0.8585	0.1262
Επανάληψη 5η	0.8585	0.0937
Εποχή 3	Ακρίβεια	Τιμή Συνάρτηση Κόστους
Επανάληψη 1η	0.9023	0.0649
Επανάληψη 2η	0.9051	0.0685
Επανάληψη 3η	0.9050	0.1081
Επανάληψη 4η	0.8919	0.1334
Επανάληψη 5η	0.9184	0.0793
Εποχή 4	Ακρίβεια	Τιμή Συνάρτηση Κόστους
Επανάληψη 1η	0.9128	0.2199
Επανάληψη 2η	0.9400	0.1514
Επανάληψη 3η	0.9348	0.0973
Επανάληψη 4η	0.9482	0.1045
Επανάληψη 5η	0.9391	0.0577

Πίνακας Α.1: Οι τιμές ακρίβειας και συναρτήσεις κόστους για κάθε εποχή.

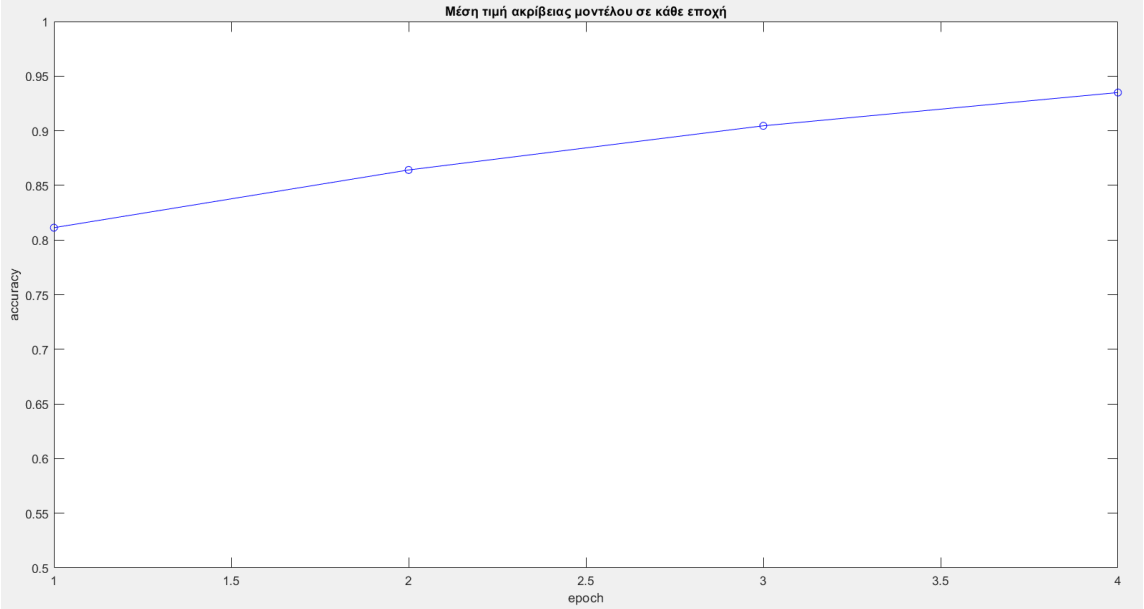
Τα δεδομένα αυτά τα αναπαριστούμε με box plot και αναπαριστούμε την μέση τιμή των τιμών αυτών για κάθε εποχή.



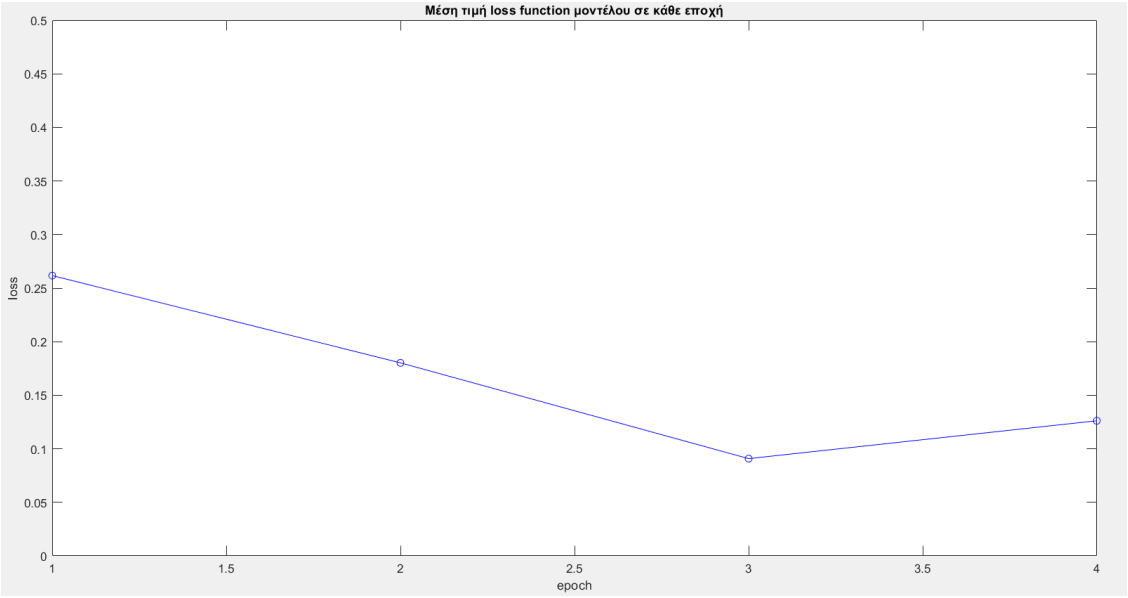
Εικόνα Α.10: boxplot με ακρίβεια μοντέλου σε κάθε επιλογή.



Εικόνα Α.11: boxplot με συνάρτηση κόστους σε κάθε εποχή.



Εικόνα Α.12: Μέση τιμή ακρίβειας σε κάθε εποχή.



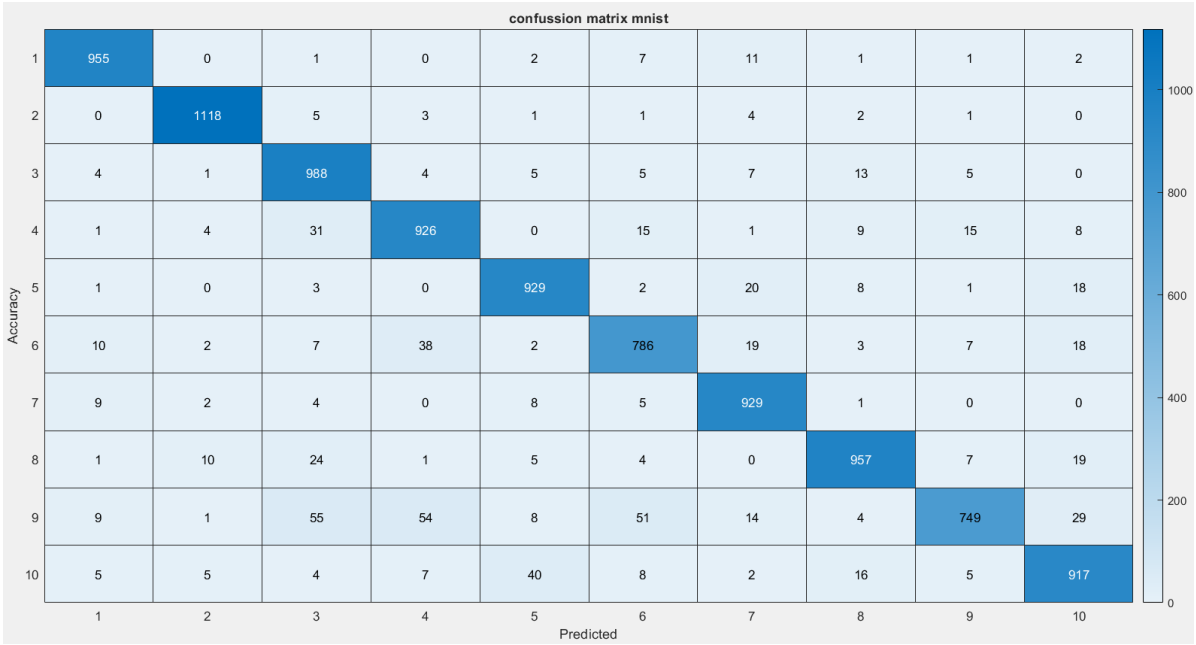
Εικόνα Α.13: Μέση τιμή συνάρτηση κόστους σε κάθε εποχή.

Μετά το τέλος της εκπαίδευσης υπολογίζουμε το confusion matrix. Η συνάρτηση που γράφουμε φαίνεται παρακάτω.

```
function [Confusion_Matrix] = createConfusionMatrix(YPred,YTest)
    % rows: πραγματικά, columns: προβλεπόμενα
    Confusion_Matrix = zeros(10, 10);
    len_Y = length(YPred); % Μέγεθος test label
    % Δημιουργεία confussion matrix
    for i = 1 : len_Y
        yp = YPred(i)+1;
        yt = YTest(i)+1;
        Confusion_Matrix(yt, yp) = Confusion_Matrix(yt, yp)+1;
    end
    % Απεικόνιση του confussion matrix
    figure;
    heatmap(Confusion_Matrix);
    xlabel('Predicted')
    ylabel('Accuracy')
    title('confussion matrix mnist')
end
```

Εικόνα Α.14: Συνάρτηση κατασκευής confusion matrix.

Τα αποτελέσματα που παίρνουμε για τις 4 εποχές είναι οι εξής:



Εικόνα Α.15: Confusion matrix.

Τα αποτελέσματα από τις μετρικές είναι:

Precision	0.9253
Sensetivity	0.9239
F-score	0.9246

Πίνακας Α.2: Ακρίβεια, ευαισθησία και F-score.

Μέρος Β: Κατηγοριοποίηση Εικόνων με χρήση Histogram of Oriented Gradients και Support Vector Machine

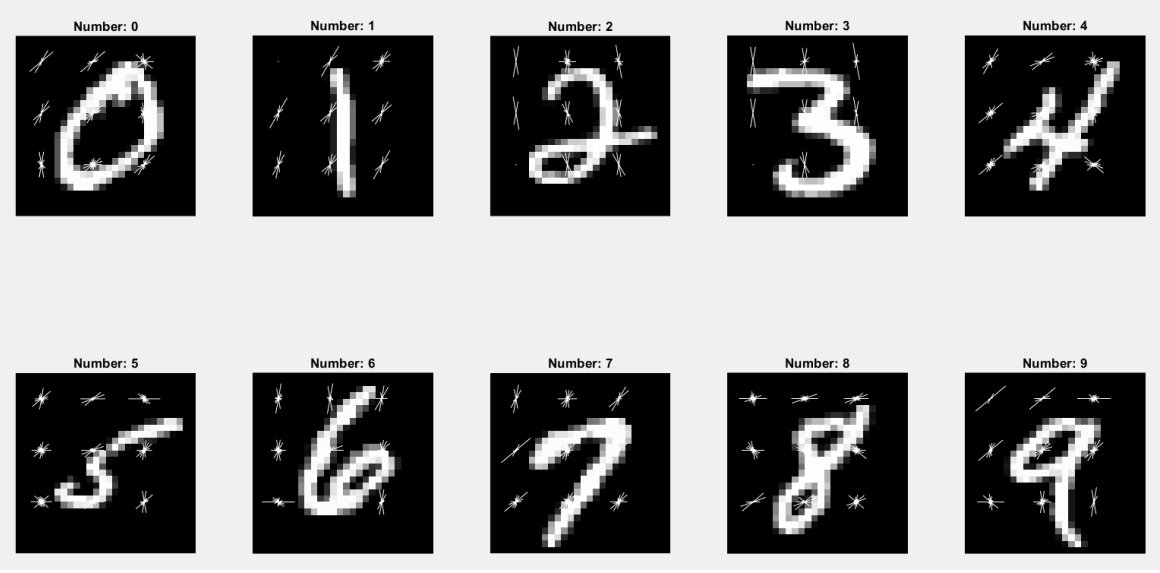
Η τεχνική HOG είναι μια σχετικά πρόσφατη μέθοδος η οποία ομαδοποιεί τις κατευθύνσεις της κλίσης της εικόνας σε μία συγκεκριμένη περιοχή, ώστε να υπάρχει καλύτερη περιγραφή του σχήματός της εικόνας. Για αυτό τον λόγο κατασκευάζεται ένα ιστόγραμμα που δείχνει το πλάτος της κάθε κατεύθυνσης. Στην άσκηση θα πάρουμε αυτές τις τιμές του ιστογράμματος και θα εκπαιδεύσουμε ένα SVM μοντέλο για την κατηγοριοποίηση των εικόνων της βιβλιοθήκης MNIST.

Εξαγωγή Χαρακτηριστικών

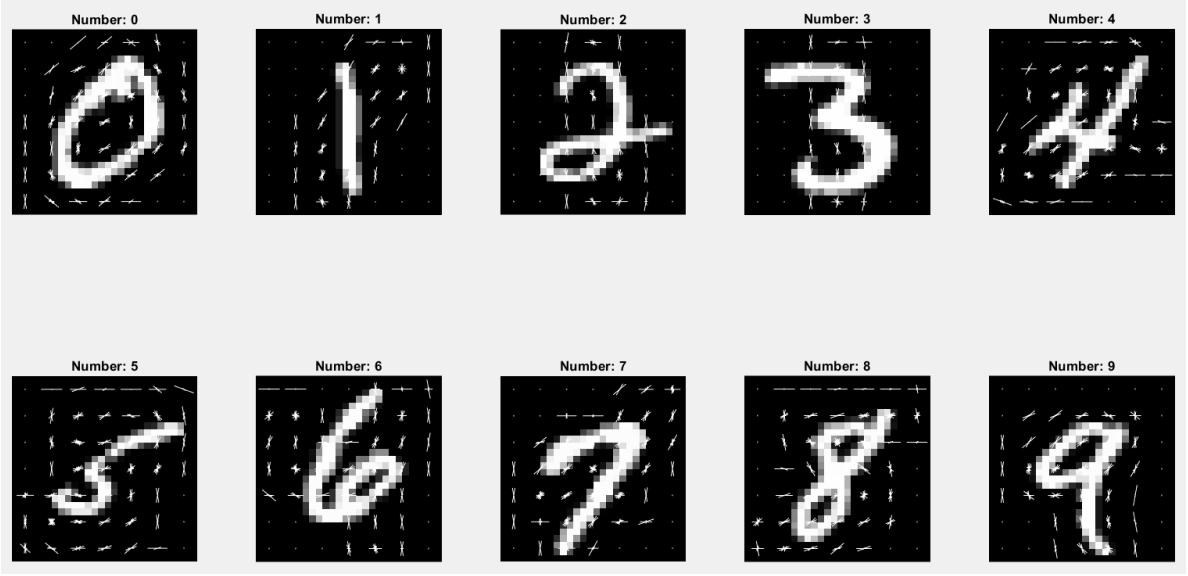
Για την εξαγωγή των χαρακτηριστικών του ιστογράμματος (HOG) θα χρησιμοποιήσουμε την συνάρτηση της MATLAB `[featureVector,hogVisualization]=extractHOGFeatures(img_num,'CellSize',[8 8])`, που παίρνει ως εισόδους την εικόνα και τον ορισμό του κάθε patch και ως εξόδους αντίστοιχα τις τιμές του ιστογράμματος και την απεικόνιση των τιμών των κλίσεων πάνω στην εικόνα.

Αποτελέσματα Άσκησης

Στις παρακάτω εικόνες βλέπουμε τα χαρακτηριστικά HOG πάνω στις εικόνες της βιβλιοθήκης MNIST για patch 8x8 και 4x4:



Εικόνα Β.1: Χαρακτηριστικά HOG των εικόνων για patch 8x8.(Παράρτημα Β, main.m)



Εικόνα Β.2: Χαρακτηριστικά HOG των εικόνων για patch 4x4.(Παράρτημα Β, main.m)

Αυτά τα χαρακτηριστικά θα τα πάρουμε και θα εκπαιδεύσουμε ένα SVM για κάθε patch με την συνάρτηση της MATLAB `classifier_to_deploy = fitcecoc(XTrainHog, trainingLabelsGrps)`, όπου παίρνει ως εισόδους τα χαρακτηριστικά HOG των δεομένων εκπαίδευσης και την κλάση αυτών και ως έξοδο έχουμε το μοντέλο SVM. Σε ξεχωριστά αρχεία εκπαιδεύουμε τα μοντέλα SVM για τα δύο patch.

Patch 8x8

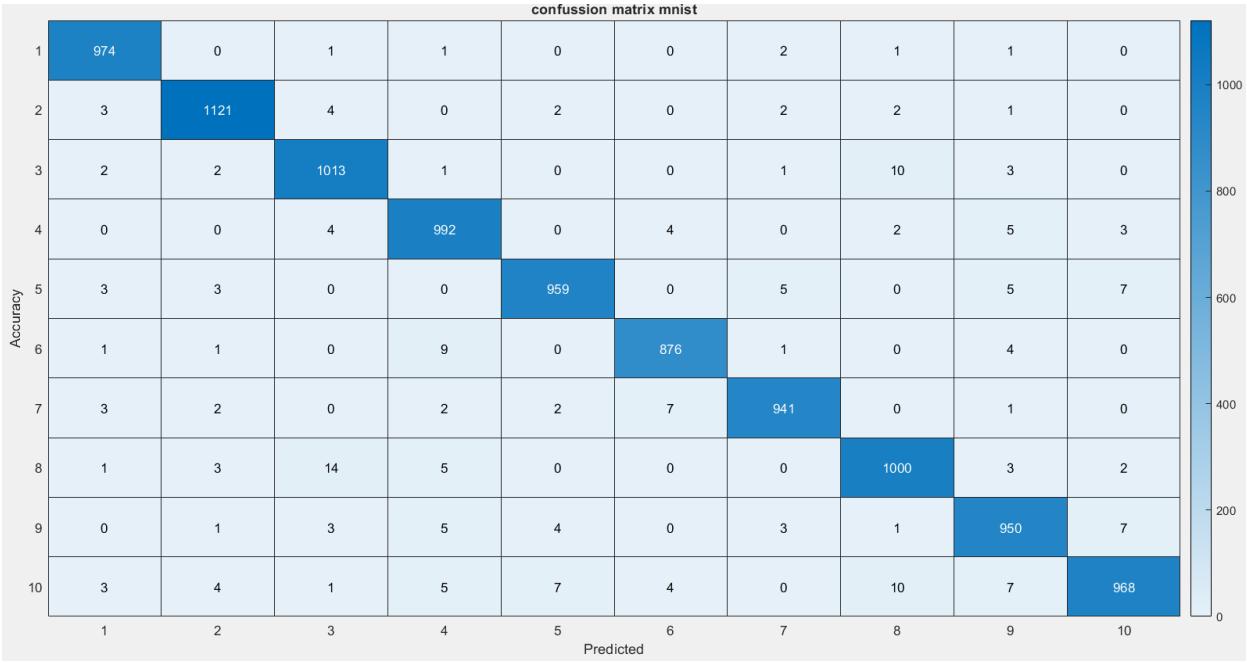
Σημείωση: Ο κώδικας βρίσκεται στο Παράρτημα Β main.m

Τα αποτελέσματα που παίρνουμε για την ακρίβεια του μοντέλου αλλά και για τις μετρικές είναι:

Accuracy	0.9794
Precision	0.9794
Sensetivity	0.9794
F-score	0.9794

Πίνακας Β.1: Ακρίβεια και μετρικές.

Και το Confusion matrix όπως ορίσαμε την συνάρτηση στην εικόνα Α.14 είναι:



Εικόνα Β.3: Confusion matrix για patch 8x8.

Patch 4x4

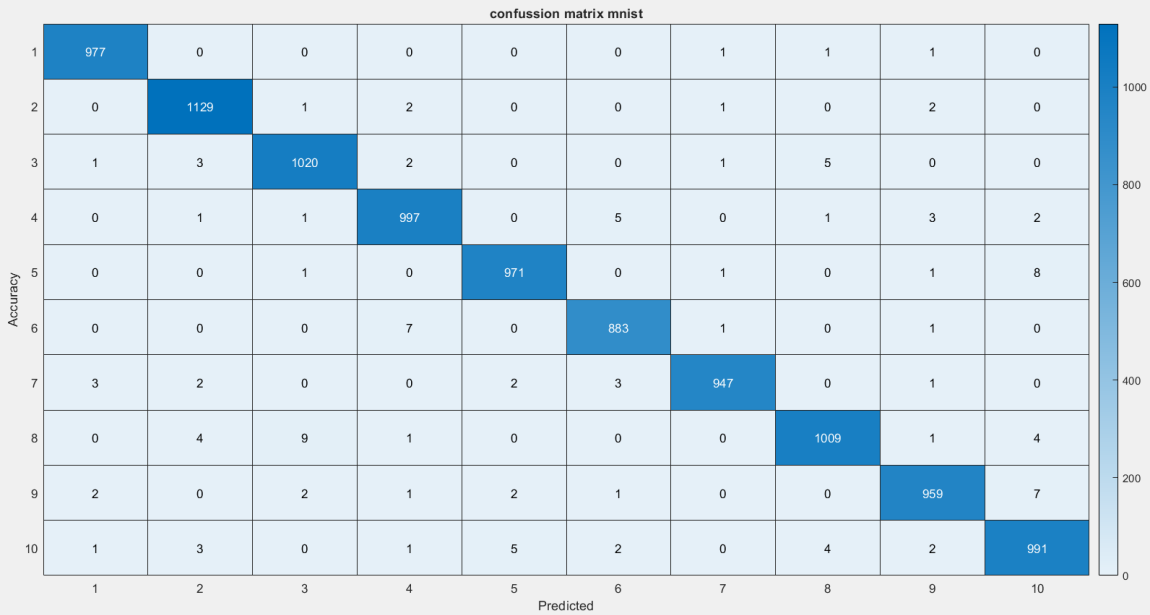
Σημείωση: Ο κώδικας βρίσκεται στο Παράρτημα Β hog4.m

Τα αποτελέσματα που παίρνουμε για την ακρίβεια του μοντέλου αλλά και για τις μετρικές είναι:

Accuracy	0.9883
Precision	0.9883
Sensetivity	0.9883
F-score	0.9883

Πίνακας Β.1: Ακρίβεια και μετρικές.

Και το Confusion matrix όπως ορίσαμε την συνάρτηση στην εικόνα Α.14 είναι:



Εικόνα Β.4: Confusion matrix για patch 4x4.

Συμπεράσματα

Παρατηρούμε από τα παραπάνω αποτελέσματα ότι το μοντέλο SVM βγάζει καλύτερα αποτελέσματα από την αρχιτεκτονική CNN που επιλέξαμε. Επίσης το μοντέλο SVM με τα χαρακτηριστικά HOG για patch 4x4 βγάζει καλύτερα αποτελέσματα από ότι για patch 8x8, επειδή έχουμε περισσότερα στοιχεία για την κλίση της κάθε υπό-εικόνας.

Βιβλιογραφία

[1] R. C. Gonzalez, “Deep Convolutional Neural Networks [Lecture Notes],” IEEE Signal Process. Mag., vol. 35, no. 6, pp. 79–87, Nov. 2018.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, Deep Learning, MIT Press, 2016, http://www.deeplearningbook.org

[3]. N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), 2005, pp. 886-893 vol. 1, doi: 10.1109/CVPR.2005.177.

[4]. P. E. Rybski, D. Huber, D. D. Morris and R. Hoffman, "Visual classification of coarse vehicle orientation using Histogram of Oriented Gradients features," 2010 IEEE Intelligent Vehicles Symposium, 2010, pp. 921-928, doi: 10.1109/IVS.2010.5547996.

[5]. X. Yuan, L. Cai-nian, X. Xiao-liang, J. Mei and Z. Jian-guo, "A two-stage hog feature extraction processor embedded with SVM for pedestrian detection," 2015 IEEE International Conference on Image Processing (ICIP), 2015, pp. 3452-3455, doi: 10.1109/ICIP.2015.7351445.

[6]: Gonzalez R.C, Woods R.E, “Ψηφιακή Επεξεργασία Εικόνας”, 4η Έκδοση, Εκδόσεις Τζιόλας 2021.

[7]: K. Berberidis, "Machine Learning & Deep learning for Image Processing & Analysis", Course: Digital Image Processing and Analysis, University of Patras.

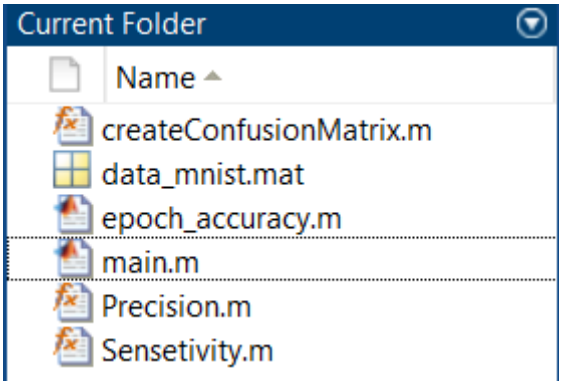
[8]: I. Markoulidakis, I. Rallis, I. Georgoulas, G. Kopsiaftis , A. Doulamis and N. Doulamis, “Multiclass Confusion Matrix Reduction Method and Its Application on Net Promoter Score Classification Problem”, Technologies, 2021, 9, 81. https://doi.org/ 10.3390/technologies9040081.

[9]: F. Yang, K. Fan, D. Song, H. Lin, “Graph-based prediction of Protein-protein interactions with attributed signed graph embedding”, BMC Bioinformatics, 2020, https://doi.org/10.1186/s12859-020-03646-8.

Παράρτημα Α

Το παράρτημα αυτό περιέχει τους κώδικες και τις συναρτήσεις που χρησιμοποιήθηκαν για το μέρος Α της άσκησης.

Η δομή του φακέλου είναι οι εξής:



Εικόνα Π.Α.1: Φάκελος Μέρος Α.

createConfusionMatrix.m

```
function [Confusion_Matrix] = createConfusionMatrix(YPred,YTest)
    % rows: πραγματικά, columns: προβλεπόμενα
    Confusion_Matrix = zeros(10, 10);
    len_Y = length(YPred); % Μέγεθος test label
    % Δημιουργεία confussion matrix
    for i = 1 : len_Y
        yp = YPred(i)+1;
        yt = YTest(i)+1;
        Confusion_Matrix(yt, yp) = Confusion_Matrix(yt, yp)+1;
    end
    % Απεικόνιση του confussion matrix
    figure;
    heatmap(Confusion_Matrix);
    xlabel('Predicted')
    ylabel('Accuracy')
    title('confussion matrix mnist')
end
```

epoch_accuracy.m

```
clc;
clear all;
%% Δεδομένα
accuracy1 = [0.7834; 0.8300; 0.7627; 0.8375; 0.8429];
accuracy2 = [0.8674; 0.8714; 0.8648; 0.8585; 0.8585];
accuracy3 = [0.9023; 0.9051; 0.9050; 0.8919; 0.9184];
accuracy4 = [0.9128; 0.9400; 0.9348; 0.9482; 0.9391];
loss1 = [0.2199; 0.1298; 0.5803; 0.1694; 0.2091];
loss2 = [0.4037; 0.0865; 0.1910; 0.1262; 0.0937];
loss3 = [0.0649; 0.0685; 0.1081; 0.1334; 0.0793];
loss4 = [0.2199; 0.1514; 0.0973; 0.1045; 0.0577];
n = [1; 2; 3; 4];
accuracies = [accuracy1, accuracy2, accuracy3, accuracy4];
losss = [loss1, loss2, loss3, loss4];
%% Box plots
figure;
boxplot(accuracies);
xlabel('epoch')
ylabel('accuracy')
title('boxplot ακρίβειας μοντέλου σε κάθε εποχή')
axis([0.5, 4.5, 0.5, 1])
figure;
boxplot(losss);
xlabel('epoch')
ylabel('loss')
title('boxplot loss function σε κάθε εποχή')
axis([0.5, 4.5, 0, 0.5])
%% Mean plots
m_acc = mean(accuracies);
m_loss = mean(losss);
figure;
plot(m_acc, '-bo');
xlabel('epoch')
ylabel('accuracy')
title('Μέση τιμή ακρίβειας μοντέλου σε κάθε εποχή')
axis([1, 4, 0.5, 1])
figure;
plot(m_loss, '-bo');
xlabel('epoch')
ylabel('loss')
title('Μέση τιμή loss function μοντέλου σε κάθε εποχή')
axis([1, 4, 0, 0.5])
```

main.m

```
clc;
clear all;
%% load data
load('data_mnist.mat', 'XTrain', 'YTrain', 'XTest', 'YTest');
%% Απεικόνιση δεδομένων από κάθε κλάση (α ερώτημα)
classes = Inf * ones(1, 10); % Ποιά νούμερα έχουν τοποθετηθεί στο images
images = zeros(28, 28, 10); % Αποθήκευση των εικόνων απεικόνισης
imgs_train = extractdata(XTrain); % Εξαγωγή των δεδομένων από το dlarray
labels = YTrain; % Οι κατηγορίες των εικόνων
i = 1; % μετρητής
while 1
    number = int16(labels(i)); % Αριθμός
    % Έλεγχος αν έχουν περάσει όλες οι κατηγορίες
    if sum(ismember(classes, Inf)) == 0
        break;
    end
    img_num = imgs_train(:, :, 1, i); % Εικόνα αριθμού
    images(:, :, number) = img_num; % Αποθήκευση αριθμού
    classes(number) = number; % Αποθήκευση ότι πέρασε αυτός ο αριθμός
    i = i+1; % Αύξηση μετρητή κατά ένα
end
% Απεικόνιση δεδομένων
figure;
for count = 1 : 10
    subplot(2, 5, count);
    imshow(images(:, :, count))
    title("Number: " + string(count-1));
    count = count + 1;
end
%% Αρχικοποίηση των παραμέτρων και των δεδομένων του CNN (β ερώτημα)
% Ορισμός επιπέδων
layers = [
    imageInputLayer([28 28 1])
    convolution2dLayer(3,6,'Padding', 0, 'Stride', 1)
    reluLayer
    averagePooling2dLayer(2,'Stride', 2)
    convolution2dLayer(3,16,'Padding', 0, 'Stride', 1)
    reluLayer
    averagePooling2dLayer(2,'Stride', 2)
    fullyConnectedLayer(120)
    reluLayer
    fullyConnectedLayer(84)
    reluLayer
    fullyConnectedLayer(10)
    softmaxLayer
    classificationLayer];
% Τοποθέτηση χαρακτηριστικών του cnn
options = trainingOptions("sgdm", ...
    'Momentum', 1, ...
    'InitialLearnRate',1e-4, ...
    'Shuffle','every-epoch', ...
    'MaxEpochs', 4, ...
    'MiniBatchSize', 100, ...
    'Verbose', false, ...
    'Plots', "training-progress");
% Εκπαίδευση CNN
net = trainNetwork(XTrain,YTrain,layers,options);
%% Κατηγοριοποίηση των δεδομένων και ποσοστό ακρίβειας του μοντέλου
YPred = classify(net,XTest); % Κατηγοριοποίηση των δεδομένων
accuracy = sum(YPred == YTest)/numel(YTest) % Ακρίβεια μοντέλου
YPred = string(YPred);
YPred = double(YPred);
```

```
YTest = string(YTest);
YTest = double(YTest);
loss = crossentropy(YPred,YTest) % Συνάρτηση κόστους
%% Confussion Matrix (δ ερώτημα)
CM = createConfusionMatrix(YPred,YTest);
%% F_Score
[pr_val] = mean(Precision(CM)) % Ακρίβεια
[sen_val] = mean(Sensetivity(CM)) % Ευαισθησία
F_Score = 2*pr_val*sen_val/(pr_val+sen_val) % F_Score
```

Precision.m

```
function [val] = Precision(CM)
    val = zeros(1, 10);
    for i = 1 : 10
        TP = CM(i, i);
        FP = sum(CM(i, :)) - CM(i, i);
        pr = TP/(TP + FP);
        val(i) = pr;
    end
end
```

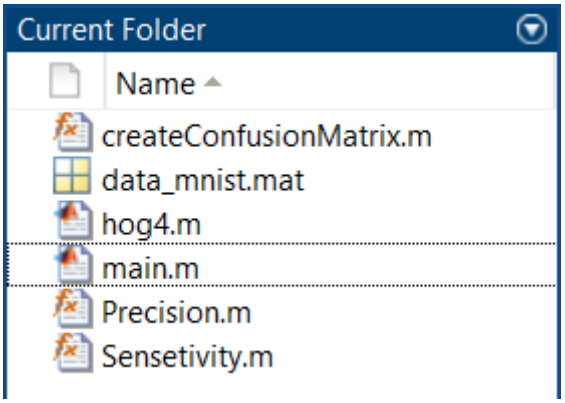
Sensetivity.m

```
function [val] = Sensetivity(CM)
    val = zeros(1, 10);
    for i = 1 : 10
        TP = CM(i, i);
        FN = sum(CM(:, i)) - CM(i, i);
        sen = TP/(TP + FN);
        val(i) = sen;
    end
end
```

Παράρτημα Β

Το παράρτημα αυτό περιέχει τους κώδικες και τις συναρτήσεις που χρησιμοποιήθηκαν για το μέρος Β της άσκησης.

Η δομή του φακέλου είναι οι εξής:



Εικόνα Π.Β.1: Φάκελος Μέρος Β.

Να σημειώσουμε ότι το createConfusionMatrix, Precision, Sensetivity είναι όπως στο παράρτημα Α.

main.m

```
clc;
clear all;
%% load data
load('data_mnist.mat', 'XTrain', 'XTrainHog8', 'YTrain', 'XTestHog8', 'YTest');
%% Απεικόνιση δεδομένων από κάθε κλάση (α ερώτημα)
classes = Inf * ones(1, 10); % Ποιά νούμερα έχουν τοποθετηθεί στο images
images = zeros(28, 28, 10); % Αποθήκευση των εικόνων απεικόνισης
images_number = []; % Αποθήκευση των εικόνων απεικόνισης
images_number_hog = [];
imgs_train = extractdata(XTrain); % Εξαγωγή των δεδομένων από το dlarray
labels = YTrain; % Οι κατηγορίες των εικόνων
i = 1; % μετρητής
while 1
    number = int16(labels(i)); % Αριθμός
    % Έλεγχος αν έχουν περάσει όλες οι κατηγορίες
    if sum(ismember(classes, Inf)) == 0
        break;
    end
    img_num = imgs_train(:, :, 1, i); % Εικόνα αριθμού
    images(:, :, number) = img_num; % Αποθήκευση αριθμού
    [featureVector,hogVisualization] = extractHOGFeatures(img_num,'CellSize',[8 8]);
    images_number = [images_number, number];
    images_number_hog = [images_number_hog, hogVisualization];
    classes(number) = number; % Αποθήκευση ότι πέρασε αυτός ο αριθμός
    i = i+1; % Αύξηση μετρητή κατά ένα
end
% Sort
for i = 1:10
    for j = i+1:10
        if images_number(i) > images_number(j)
            images_number([i, j]) = images_number([j, i]);
            images_number_hog([i, j]) = images_number_hog([j, i]);
        end
    end
end
% Απεικόνιση δεδομένων
figure;
for count = 1 : 10
    subplot(2, 5, count);
    imshow(images(:, :, count));
    hold on;
    plot(images_number_hog(count))
    title("Number: " + string(count-1));
    count = count + 1;
end
%% Εκπαίδευση SVM
trainingLabelsGrps = grp2idx(YTrain)-1;  % returns group indices  1 through 10
classifier_to_deploy = fitcecoc(XTrainHog8, trainingLabelsGrps);
%% Ταξινόμηση δεδομένων
label = predict(classifier_to_deploy,XTestHog8);
YTest = string(YTest);
YTest = double(YTest);
accuracy = sum(label == YTest)/length(label)
%% Confussion Matrix
CM = createConfusionMatrix(label,YTest);
[pr_val] = mean(Precision(CM))
[sen_val] = mean(Sensetivity(CM))
F_Score = 2*pr_val*sen_val/(pr_val+sen_val)
```

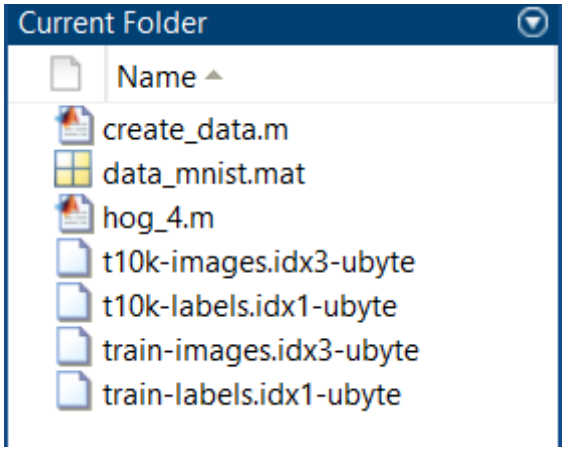
hog4.m

```
clc;
clear all;
%% load data
load('data_mnist.mat', 'XTrain', 'XTrainHog4', 'YTrain', 'XTestHog4', 'YTest');
%% Απεικόνιση δεδομένων από κάθε κλάση (α ερώτημα)
classes = Inf * ones(1, 10); % Ποιά νούμερα έχουν τοποθετηθεί στο images
images = zeros(28, 28, 10); % Αποθήκευση των εικόνων απεικόνισης
images_number = []; % Αποθήκευση των εικόνων απεικόνισης
images_number_hog = [];
imgs_train = extractdata(XTrain); % Εξαγωγή των δεδομένων από το dlarray
labels = YTrain; % Οι κατηγορίες των εικόνων
i = 1; % μετρητής
while 1
    number = int16(labels(i)); % Αριθμός
    % Έλεγχος αν έχουν περάσει όλες οι κατηγορίες
    if sum(ismember(classes, Inf)) == 0
        break;
    end
    img_num = imgs_train(:, :, 1, i); % Εικόνα αριθμού
    images(:, :, number) = img_num; % Αποθήκευση αριθμού
    [featureVector,hogVisualization] = extractHOGFeatures(img_num,'CellSize',[4 4]);
    images_number = [images_number, number];
    images_number_hog = [images_number_hog, hogVisualization];
    classes(number) = number; % Αποθήκευση ότι πέρασε αυτός ο αριθμός
    i = i+1; % Αύξηση μετρητή κατά ένα
end
% Sort
for i = 1:10
    for j = i+1:10
        if images_number(i) > images_number(j)
            images_number([i, j]) = images_number([j, i]);
            images_number_hog([i, j]) = images_number_hog([j, i]);
        end
    end
end
% Απεικόνιση δεδομένων
figure;
for count = 1 : 10
    subplot(2, 5, count);
    imshow(images(:, :, count));
    hold on;
    plot(images_number_hog(count))
    title("Number: " + string(count-1));
    count = count + 1;
end
%% Εκπαίδευση SVM
trainingLabelsGrps = grp2idx(YTrain)-1;  % returns group indices  1 through 10
classifier_to_deploy = fitcecoc(XTrainHog4, trainingLabelsGrps);
%% Ταξινόμηση δεδομένων
label = predict(classifier_to_deploy, XTestHog4);
YTest = string(YTest);
YTest = double(YTest);
accuracy = sum(label == YTest)/length(label)
%% Confussion Matrix
CM = createConfusionMatrix(label,YTest);
[pr_val] = mean(Precision(CM))
[sen_val] = mean(Sensetivity(CM))
F_Score = 2*pr_val*sen_val/(pr_val+sen_val)
```

Παράρτημα Γ

Το παράρτημα αυτό περιέχει τους κώδικες για το άνοιγμα των αρχείων που πάρθηκαν από την ιστοσελίδα http://yann.lecun.com/exdb/mnist/ και την δημιουργία των δεδομένων για τις ασκήσεις (data_mnist.mat).

Η δομή του φακέλου είναι οι εξής:



Εικόνα Π.Γ.1: Φάκελος δημιουργίας δεδομένων.

Για την δημιουργία του αρχείου data_mnist.mat τρέχουμε πρώτα το create_data.m και μετά το αρχείο hog_4.m.

create_data.m

```
clc;
clear all;
%% Αρχεία Mnist Βιβλιοθήκης
oldpath = addpath(fullfile(matlabroot,'examples','nnet','main'));
filenameImagesTrain = 'train-images.idx3-ubyte';
filenameLabelsTrain = 'train-labels.idx1-ubyte';
filenameImagesTest = 't10k-images.idx3-ubyte';
filenameLabelsTest = 't10k-labels.idx1-ubyte';
%% Πίνακες με τα δεδομένα
XTrain = processImagesMNIST(filenameImagesTrain);
YTrain = processLabelsMNIST(filenameLabelsTrain);
XTest = processImagesMNIST(filenameImagesTest);
YTest = processLabelsMNIST(filenameLabelsTest);
%% Extract hog [8, 8]
sz_Xtrain = size(XTrain);
sz_Xtest = size(XTest);
len_xtr = sz_Xtrain(4);
len_xtes = sz_Xtest(4);
XTrainHog8 = [];
XTestHog8 = [];
for i = 1 : len_xtr
    I = extractdata(XTrain(:, :, 1, i));
    XTr_featureVector = extractHOGFeatures(I,'CellSize',[8 8]);
    XTrainHog8 = [XTrainHog8; XTr_featureVector];
end
for i = 1 : len_xtes
    I = extractdata(XTest(:, :, 1, i));
    XTes_featureVector = extractHOGFeatures(I,'CellSize',[8 8]);
    XTestHog8 = [XTestHog8; XTes_featureVector];
end
%% Αποθήκευση αρχείων
save('data_mnist.mat', 'XTrain', 'YTrain', 'XTest', 'YTest', 'XTrainHog8',
'XTestHog8');
```

hog_4.m

```
clc;
clear all;
%% Αρχεία Mnist Βιβλιοθήκης
oldpath = addpath(fullfile(matlabroot,'examples','nnet','main'));
filenameImagesTrain = 'train-images.idx3-ubyte';
filenameLabelsTrain = 'train-labels.idx1-ubyte';
filenameImagesTest = 't10k-images.idx3-ubyte';
filenameLabelsTest = 't10k-labels.idx1-ubyte';
%% Πίνακες με τα δεδομένα
XTrain = processImagesMNIST(filenameImagesTrain);
YTrain = processLabelsMNIST(filenameLabelsTrain);
XTest = processImagesMNIST(filenameImagesTest);
YTest = processLabelsMNIST(filenameLabelsTest);
%% Extract hog [4, 4]
sz_Xtrain = size(XTrain);
sz_Xtest = size(XTest);
len_xtr = sz_Xtrain(4);
len_xtes = sz_Xtest(4);
XTrainHog4 = [];
XTestHog4 = [];
for i = 1 : len_xtr
    I = extractdata(XTrain(:, :, 1, i));
    XTr_featureVector = extractHOGFeatures(I,'CellSize',[4 4]);
    XTrainHog4 = [XTrainHog4; XTr_featureVector];
end
for i = 1 : len_xtes
    I = extractdata(XTest(:, :, 1, i));
    XTes_featureVector = extractHOGFeatures(I,'CellSize',[4 4]);
    XTestHog4 = [XTestHog4; XTes_featureVector];
end
%% Αποθήκευση αρχείων
save('data_mnist.mat', 'XTrainHog4', 'XTestHog4','-append');
```