

Automatisierungstechnik

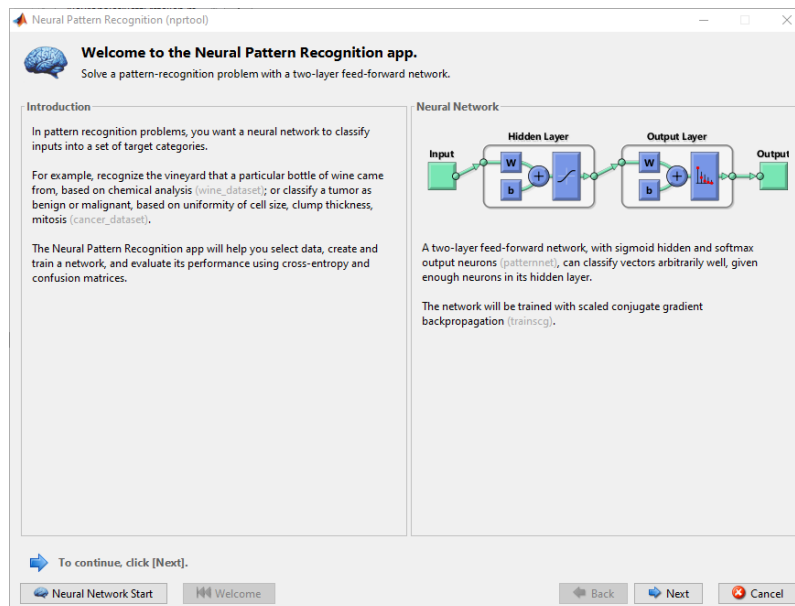
| | |
|--|----------|
| Automatisierungstechnik - Übung 4 | 2 |
| Aufgabe 1: Neural Pattern Recognition Application | 2 |
| Aufgabe 2: Erzeugen eines KNN auf der Kommandozeile..... | 3 |
| Automatisierungstechnik - Übung 5 | 4 |
| Aufgabe 3: Ablaufdiagramm Sortieranlage | 4 |

Automatisierungstechnik - Übung 4

Aufgabe 1: Neural Pattern Recognition Application

Die *Neural Pattern Recognition Application* erleichtert das Erstellen künstlicher neuronaler Netze auf Basis vorhandener Daten. Die Auswahl der Daten erfolgt dabei geführt durch die Applikation und kann auf den Workspace oder mat-Dateien zugreifen.

Von der Matlab-Kommandozeile startet man das Programm mit: `nprtool`



- Importieren Sie die Eingangs- und Zieldaten und erstellen sie ein Netz mit zehn verborgenen Neuronen. Verwenden Sie die Standardeinstellungen der Anwendung zur Aufteilung der Trainings-, Validierungs- und Test-Daten.
- Trainieren sie das Netzwerk und lassen Sie sich die Konfusionsmatrix ausgeben.
- Wie verhält sich die Konfusionsmatrix, wenn Sie den Trainingsvorgang wiederholen?
- Wie wirkt es sich aus, wenn Sie die Aufteilung der Trainings-, Validierungs- und Test-Daten oder die Anzahl der verborgenen Neuronen ändern?

Aufgabe 2: Erzeugen eines KNN auf der Kommandozeile

Neben der *Neural Pattern Recognition Application* können künstliche neuronale Netze auch von der Kommandozeile oder in Matlab-Skripten erzeugt werden. Hier stehen wesentlich mehr und flexiblere Möglichkeiten zur Verfügung.

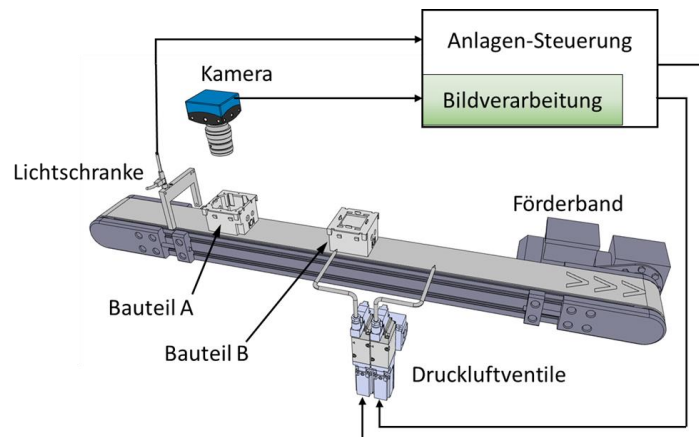
Verwenden Sie für diese Aufgabe wieder die Bilder aus `vTrainImages` und die Zieldaten `tTrain`.

- a) Erstellen Sie das künstliche Neuronale Netz auf der Kommandozeile. Recherchieren Sie dazu mit dem Befehl `doc` die Befehle `patternnet`, `train`, `view`, `net` und `plotconfusion`.
- b) Überprüfen Sie die Funktionsfähigkeit des KNN mit dem Datensatz `vTestImages` und vergleichen Sie das Ergebnis mit der Konfusionsmatrix aus dem Trainingsdurchlauf.
- c) Speichern Sie das erzeugte KNN in einer Datei. Löschen Sie das erzeugte KNN im Workspace und laden Sie es von Ihrer gespeicherten Datei. Erzeugen Sie einen Simulink-Funktionsblock mit dem Befehl `gensim`.

Automatisierungstechnik - Übung 5

Aufgabe 3: Ablaufdiagramm Sortieranlage

Folgendes Bild zeigt eine Sortieranlage für optisch unterscheidbare Bauteile:



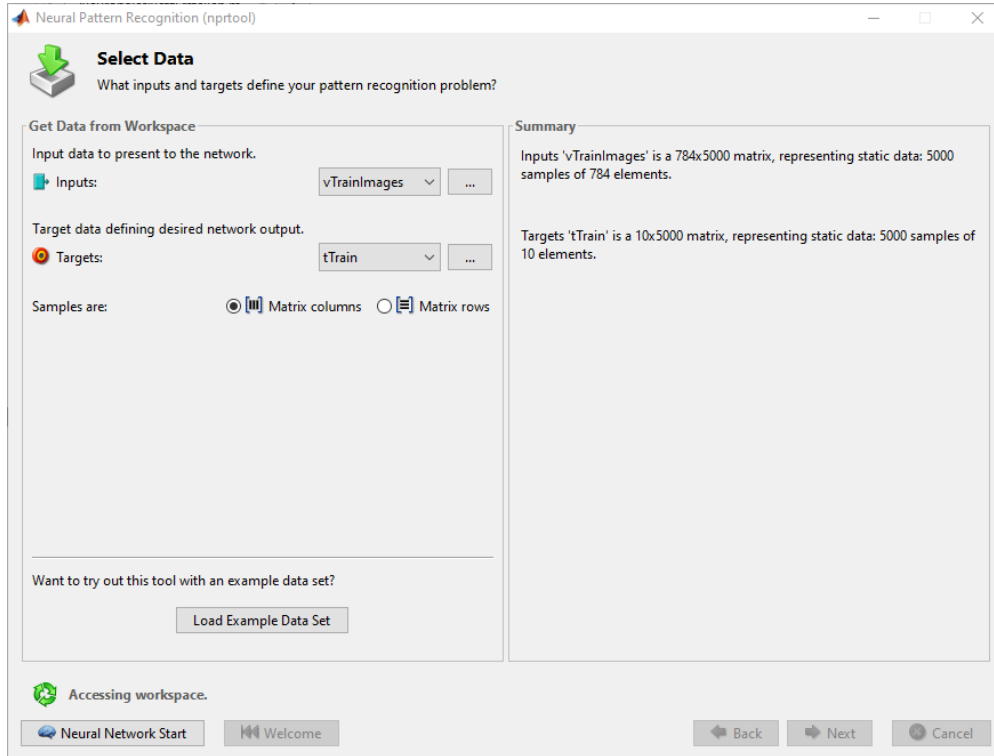
Bauteile verschiedener Art werden auf einem Förderband von links nach rechts mit gleichmäßiger Geschwindigkeit transportiert. Der Sortiervorgang soll aufgrund optisch unterscheidbarer Merkmale der Bauteile erfolgen. Die Orientierung der Bauteile auf dem Förderband ist zufällig.

Durchläuft das Bauteil die Lichtschranke, so wird die Kamera ausgelöst und ein Bild vom Bauteil an die Steuerung geschickt. Diese kann zwei verschiedene Bauteile unterscheiden. Die Bauteile werden per Druckluft vom Förderband jeweils in einen eigenen Auffangbehälter geblasen. Nicht erkannte oder fehlerhafte Bauteile fallen in einen dritten Auffangbehälter am Ende des Förderbands.

- Zeichnen Sie einen Ablaufplan, der kontinuierlich zwei Bauteile sortiert.
- Erstellen Sie eine Liste aller Ein- und Ausgänge.

Lösung 1

a) Dialoge der Anwendung bei der Erstellung des KNN



Neural Pattern Recognition (nprtool)

Select Data
What inputs and targets define your pattern recognition problem?

Get Data from Workspace

Input data to present to the network.

Inputs: vTrainImages ...

Target data defining desired network output.

Targets: tTrain ...

Samples are: ☒ Matrix columns ☐ Matrix rows

Want to try out this tool with an example data set?

Load Example Data Set

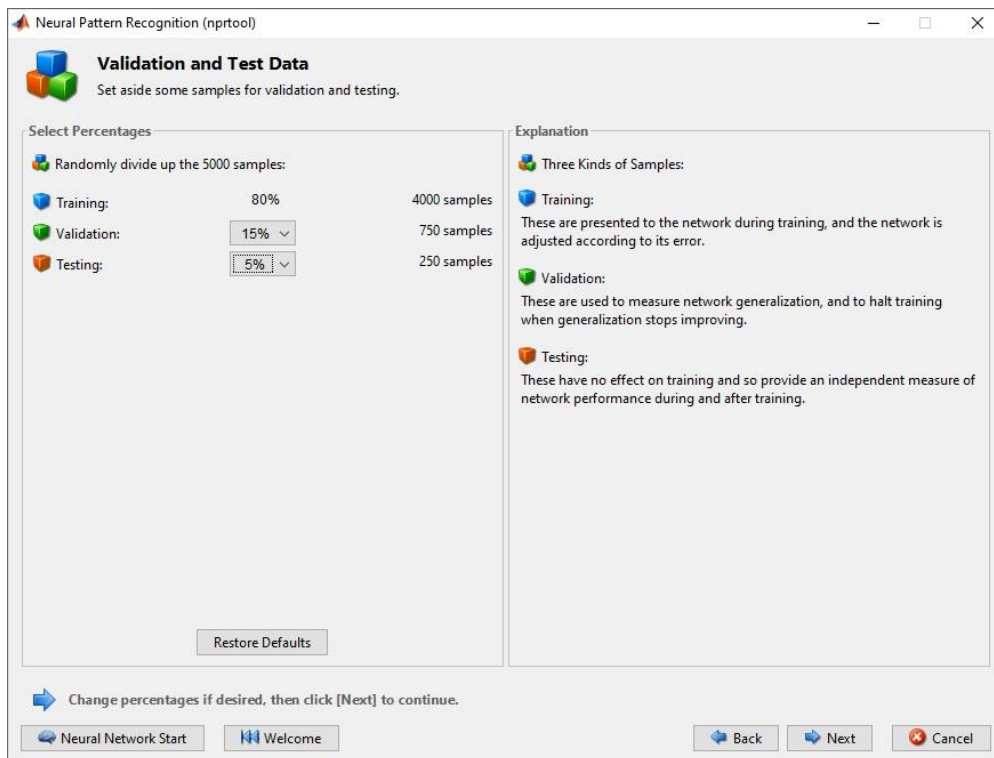
Summary

Inputs 'vTrainImages' is a 784x5000 matrix, representing static data: 5000 samples of 784 elements.

Targets 'tTrain' is a 10x5000 matrix, representing static data: 5000 samples of 10 elements.

Accessing workspace.

Neural Network Start Welcome Back Next Cancel



Neural Pattern Recognition (nprtool)

Validation and Test Data
Set aside some samples for validation and testing.

Select Percentages

Randomly divide up the 5000 samples:

| Category | Percentage | Number of Samples |
|-------------|------------|-------------------|
| Training: | 80% | 4000 samples |
| Validation: | 15% | 750 samples |
| Testing: | 5% | 250 samples |

Restore Defaults

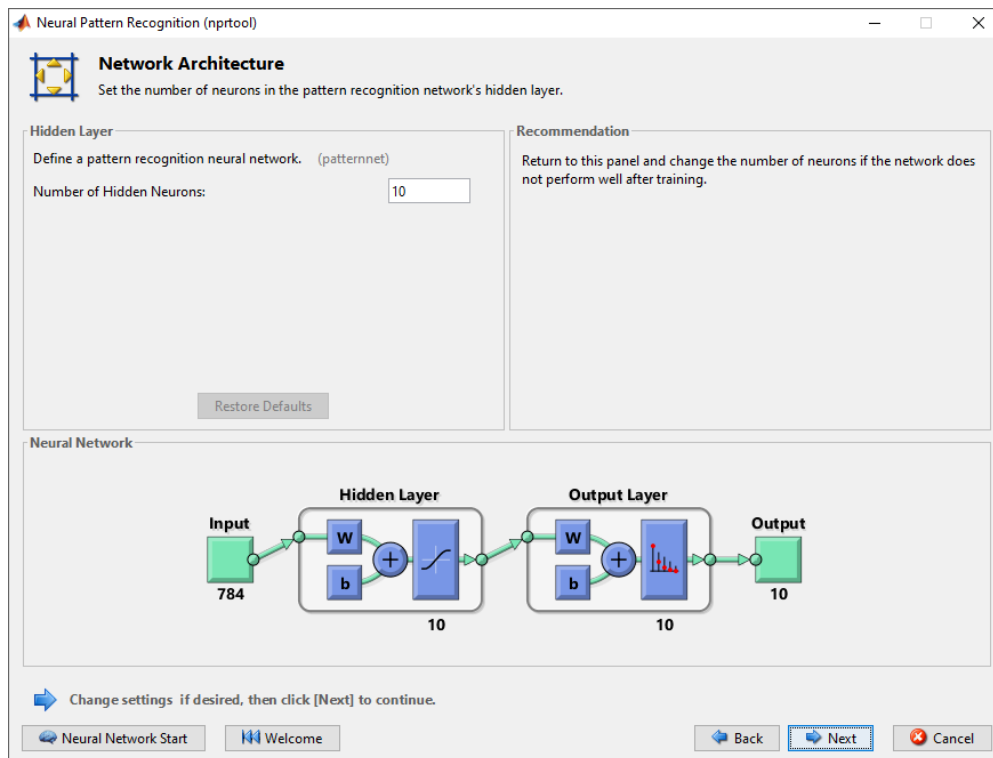
Explanation

Three Kinds of Samples:

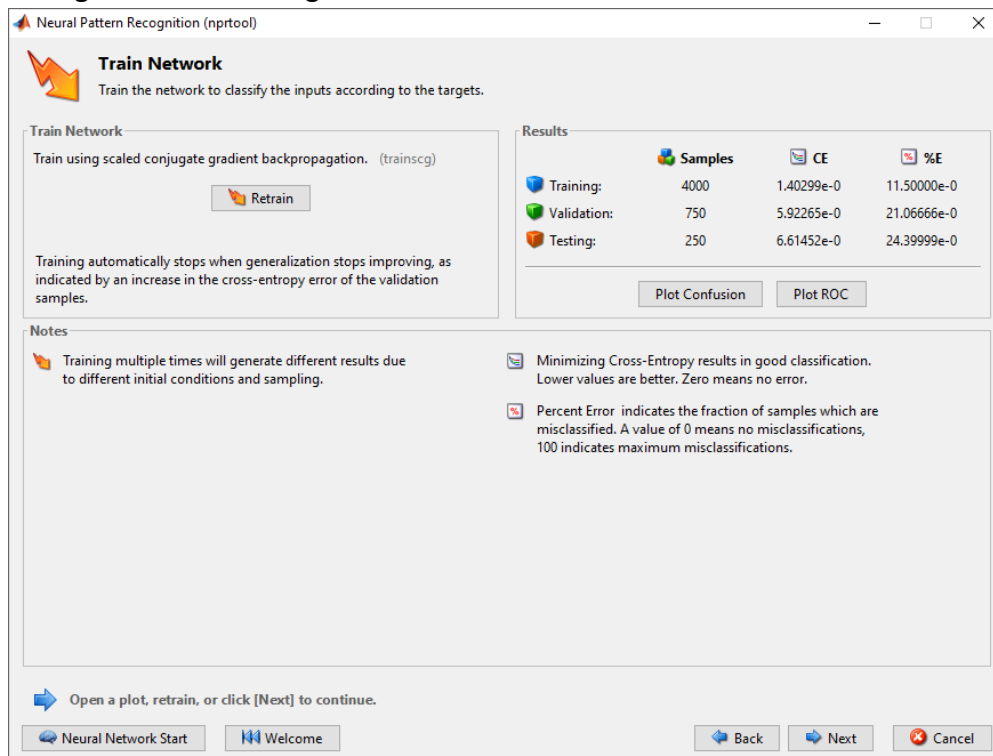
- Training:** These are presented to the network during training, and the network is adjusted according to its error.
- Validation:** These are used to measure network generalization, and to halt training when generalization stops improving.
- Testing:** These have no effect on training and so provide an independent measure of network performance during and after training.

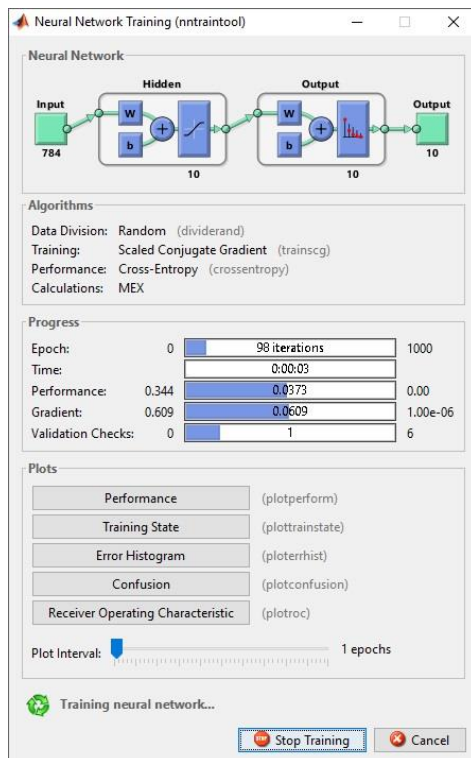
Change percentages if desired, then click [Next] to continue.

Neural Network Start Welcome Back Next Cancel



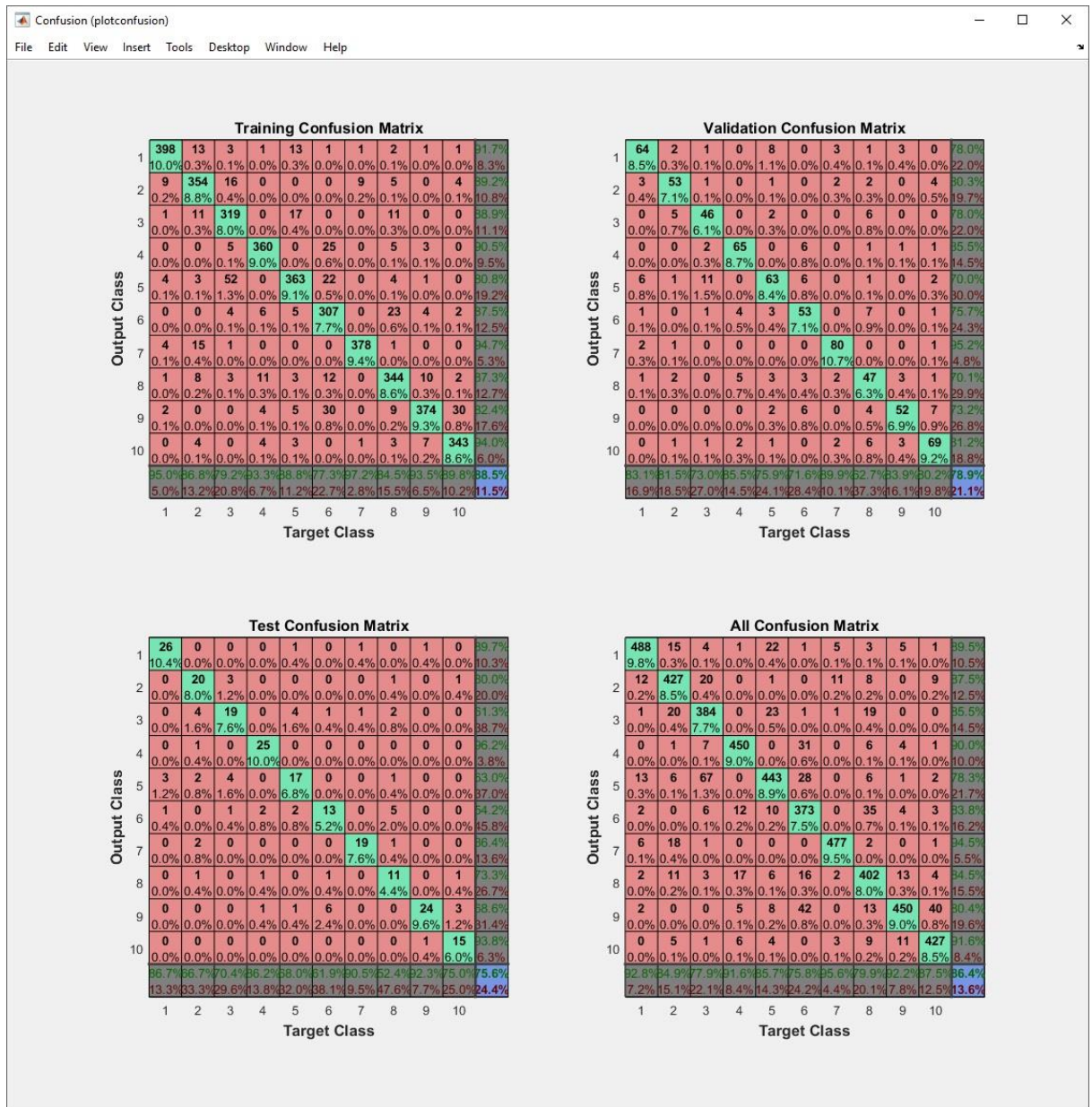
b) Dialoge der Anwendung zum Trainieren des Netzwerkes





- c) Konfusionsmatrix deren Werte im Detail immer leicht abweichen, da die Verteilung des Datensatzes auf Trainings-, Validierungs- und Testdatensätze bei jedem Durch-

lauf variiert.



Lösung 2

a) Matlab-Code

```
% define neural network
net = patternnet(10)
```

```
% train neural network with training data
net = train(net,vTrainImages,tTrain);
% display resulting network
view(net)
```



```
% plot confusion matrix  
y = net(vTrainImages);  
plotconfusion(tTrain,y,'Training Data ');
```

b) Matlab-Code

```
% test network and plot confusion matrix  
y = net(vTestImages);  
plotconfusion(tTest,y,'Testing Data ');
```

c) Matlab-Code

```
% save neural net from workspace to file  
save('net.mat', 'net');
```

```
% loading the neural net into workspace  
clear net;  
load('net.mat');
```

```
% generate Simulink block for neural network simulation  
gensim(net);
```

