

## Pseudocode Algorithm Net-Training

```
Neuron n = null
Class ClassHighError = 0; //class with the highest error
Class actualClass = 0 // actual class specified by the pattern
nrOfTrainings = 0 // number of trainings
actualHighestAccuracy = 0 // highest accuracy of a past training
Initialize input & target values

// max number of trainings without highest accuracy
TRAINING_WITHOUT_HIGHEST_ACCURACY = 5

Create Classes (–objects) with one neuron per subclass with the given target values
and map it with the neurons

while TRAINING_WITHOUT_HIGHEST_ACCURACY > 0 OR nrOfTrainings < 10
    CREATE_NET()
    Initialize PatternSet

    TRAINER.train()

    for each pattern
        n = getWinningNeuron()
        actualClass = pattern.class()

        actualClass.numberOfUses() + 1

        if n.class() = actualClass //winning neuron is in correct subclass
            if n ≠ expected Ouputneuron (n ≠ actualClass.outputneuron())
                set n as outputneuron (value 1) in target values and old outputneuron 0
        else
            actualClass.numberOfWrongOutput() + 1
            if actualClass.error > ScHighError.error
                ScHighError = actualSc

            actualSc.safeWrongPattern(pattern)

    ClassHighError.addNeuron() // wrong patterns: set old outneuron 0, added neuron 1

    if net.currentAccuracy() > actualHighestAccuracy
        actualHighestAccuracy = net.currentAccuracy
        TRAINING_WITHOUT_HIGHEST_ACCURACY = 5
        store the whole net
    else
        TRAINING_WITHOUT_HIGHEST_ACCURACY – 1

    nrOfTrainings + 1
end while
```

$\text{Class().error} = \text{numberOfWrongOutput()} / \text{numberOfUses()} = \% \text{ indication of error frequency}$

*NET\_ACCURACY()*: = percentage of correctly predicted outputs