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

$\boldsymbol{while}$ TRAINING_WITHOUT_HIGHEST_ACCURACY $> 0$ **OR** $nrOfTrainings < 10$
  $CREATE\_NET()$
  $Initialize\ PatternSet$

  $TRAINER.train()$

  $\boldsymbol{for}$ each pattern
      $n = getWinningNeuron()$
      $actualClass = pattern.class()$

      $actualClass.numberOfUses() + 1$

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

          $actualSc.safeWrongPattern(pattern)$


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

  $\boldsymbol{if}\ net.currentAccuracy() > actualHighestAccuracy$
      $actualHighestAccuracy = net.currentAccuracy$
      TRAINING_WITHOUT_HIGHEST_ACCURACY $= 5$
      store the whole net
  $\boldsymbol{else}$
      TRAINING_WITHOUT_HIGHEST_ACCURACY $- 1$

  nrOfTrainings $+ 1$
$\boldsymbol{end\ while}$
```

Class().error = numberOfWrongOutput() / numberOfUses() = % indication of error frequency

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