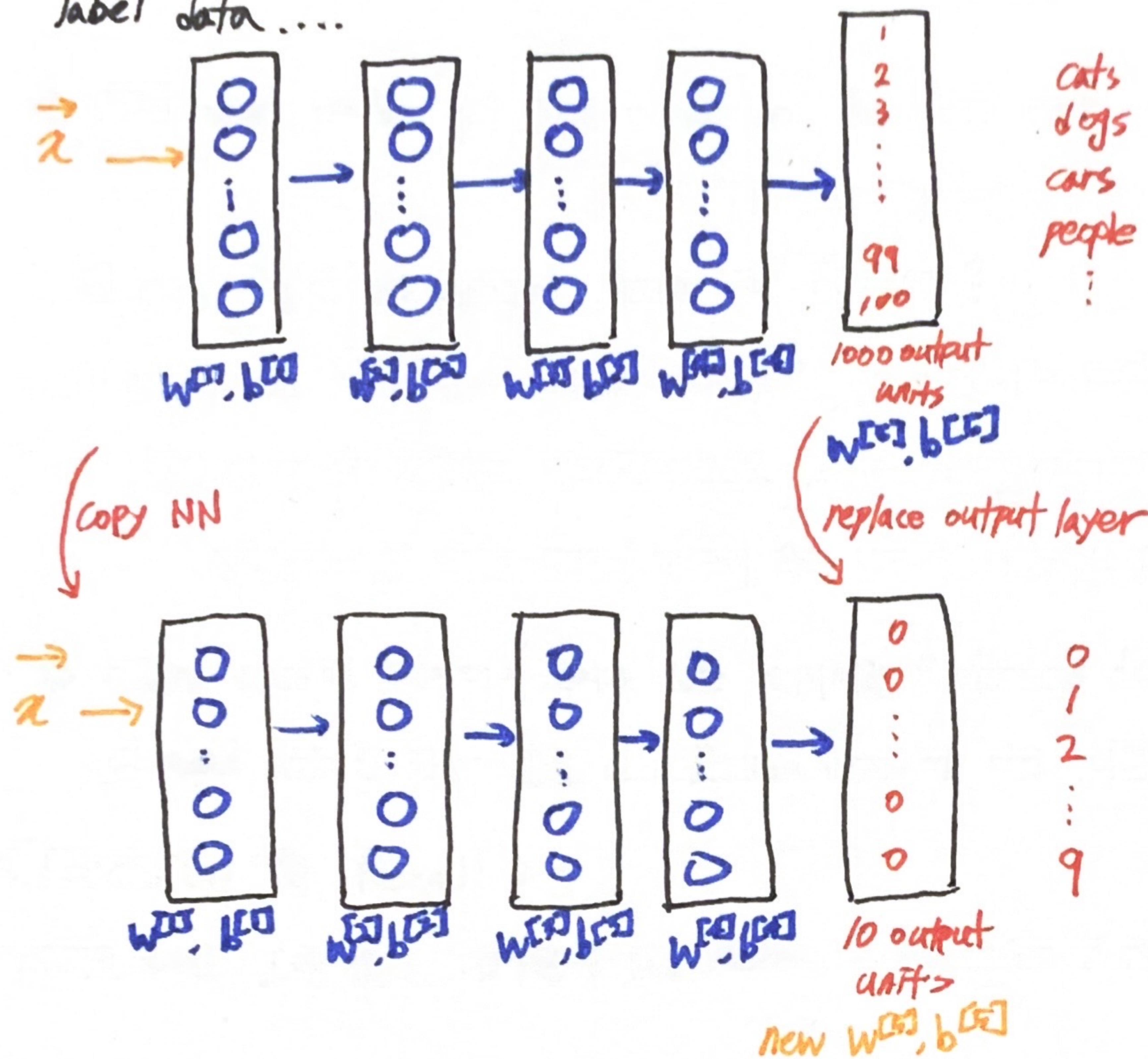⟨Machine Learning development process – Transfer learning using data from different task⟩

– Transfer learning

(☆) When you want to recognize the handwritten digits from 0~9, but don't have enough label data ....

$\vec{x}$ →

$W^{[1]}, b^{[1]}$ $W^{[2]}, b^{[2]}$ $W^{[3]}, b^{[3]}$ $W^{[4]}, b^{[4]}$ 1000 output units

$W^{[5]}, b^{[5]}$

cats
dogs
cars
people

1,000 classes   ⇐ 1 million train images

① Supervised pretraining
  – training on a large dataset
  (– not quite related task (digit recognition)

(Copy NN

replace output layer

② Fine tuning
  – take parameters that have initialized or gotten from supervised pre-training
  – then run optimization further to fine-tune to suit your specific task

$\vec{x}$ →
$W^{[1]}, b^{[1]}$ $W^{[2]}, b^{[2]}$ $W^{[3]}, b^{[3]}$ $W^{[4]}, b^{[4]}$ 10 output units →

new $W^{[5]}, b^{[5]}$

0
0
1
2
:
9

Option 1 : only train output layer parameters
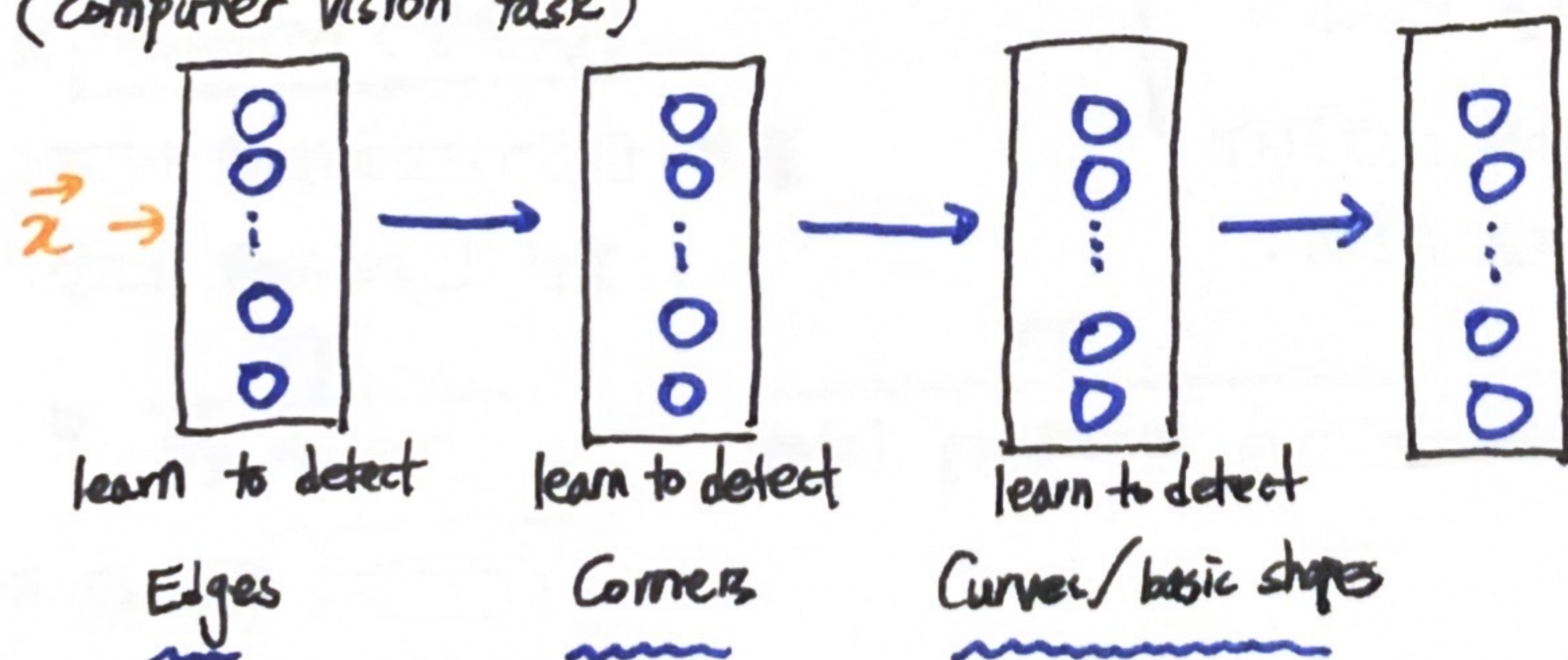 → hold $W^{[1]}, b^{[1]} \sim W^{[4]}, b^{[4]}$ fixed and don't change them
  → only update new $W^{[5]}, b^{[5]}$ to lower cost function that are used for learning to recognize 0~9 digits

Option 2 : train all parameters.
  → first four layers parameters ($W^{[1]}, b^{[1]} \sim W^{[4]}, b^{[4]}$) would be initialized using the values that you have trained on the first NN (1000 classes)

* Why does transfer learning work?
(computer vision task)

$\vec{x}$ →

learn to detect   learn to detect   learn to detect
Edges   Corners   Curves / basic shapes

* restriction *
⇒ use same input type that had used on supervised pretraining when you're going to fine-tune

⇒ By learning on detecting lots of different images, you're teaching the neural network to detect edges, corners, curves ("generic features") ⇒ helpful other computer vision task