Perceptron

ក្នុងអត្ថបទនេះយើងនឹងលើកយក Perceptron Algorithm មកបង្ហាញ។ Perceptron គឺជាAlgorithm មួយដែលត្រូវបានណែនាំជាដំបូងដោយអ្នកស្រាវជ្រាវអាមេរិកគឺលោក Alexander Rosenblatt នៅឆ្នាំ១៩៥៧។ ទោះបីជាPerceptron Algorithm ជាវិធីសាស្ត្រចាស់ក្តី ប៉ុន្តែជាត្រូវបានគេស្គាល់ថាជាប្រភពដើមនៃវិធីសាស្ត្រគណនាក្នុងម៉ូឌែលNeural Network ឬ Deep Learningដែលកំពុងរីកដុះដាលយ៉ាងសកម្មនាសម័យនេះ។ ហេតុនេះ ការសិក្សាអំពីPerceptron Algorithm អាចជួយឱ្យយើងងាយស្រួលក្នុងការឈានទៅសិក្សាអំពីNeural Network ឬ Deep Learning។

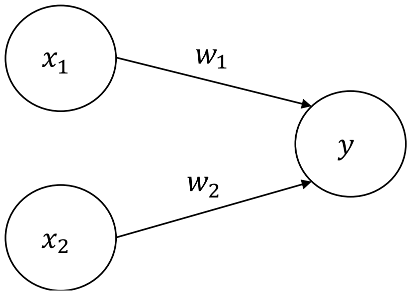
1. អំពីPerceptron

Perceptron ឬហៅថាម៉ូឌែលណឺរ៉ូនសិប្បនិម្មិត(artificial neuron) ជាម៉ូឌែលគណិតវិទ្យាមួយដែលទទួលសញ្ញាណ(signal)ឬ ជាតុចូល(input)ច្រើន និងផ្តល់នូវលទ្ធផល(output)មួយ។ សញ្ញាណដែលទទួលនៅទីនេះអាចប្រៀបបានជាចរន្តអគ្គិសនីឬសញ្ញាណព័ត៌មានដូចដែលណឺរ៉ូននៃប្រព័ន្ធប្រសាទរបស់ភាវរស់ទទួលដែរ។ ប៉ុន្តែសញ្ញាណឬជាតុចូលក្នុងPerceptron កត់យកតម្លៃធម្មតាពោលគឺបញ្ជូនបន្តឬមិនបញ្ជូនបន្តដោយតម្លៃ(1 ឬ 0)។

រូបទី១ខាងក្រោមបង្ហាញអំពីPerceptronដែលទទួលសញ្ញាណឬជាតុចូល(input) 2។ រង្វង់ដែលមានក្នុងរូបហៅថាណឺរ៉ូន(neuron) ឬnode។ x_1, x_2 ជាសញ្ញាណឬជាតុចូល ឯy គឺជាលទ្ធផលនៃណឺរ៉ូននោះ។ w_1, w_2 ជាទម្ងន់ផ្ទាល់នៃណឺរ៉ូនចំពោះជាតុចូលនិមួយៗ ពោលគឺតម្លៃដែលបង្ហាញនូវកម្រិតឥទ្ធិពលនៃជាតុចូលនិមួយៗទៅលើលទ្ធផល។ តម្លៃនៃទម្ងន់កាន់តែធំបង្ហាញពីកម្រិតសំខាន់នៃជាតុចូលនោះ។

នៅពេលដែលណឺរ៉ូនមួយទទួលបាននូវសញ្ញាណឬជាតុចូល នោះផលគុណរវាងតម្លៃនៃជាតុចូលនោះនិងតម្លៃនៃទម្ងន់ផ្ទាល់របស់ណឺរ៉ូននោះ(w_1x_1, w_2x_2)ត្រូវបានគណនា។ លទ្ធផលដែលណឺរ៉ូននោះត្រូវផ្តល់គឺអាស្រ័យនឹងផលបូកនៃគ្រប់ជាតុចូលទាំងអស់ ដោយកំណត់តាមលក្ខខណ្ឌខាងក្រោម។ លក្ខខណ្ឌនេះគឺ ណឺរ៉ូននឹងបញ្ចេញលទ្ធផល 1 បើផលបូកនៃផលគុណជាតុចូលនិងទម្ងន់របស់វាធំជាងតម្លៃនៃកម្រិតកំណត់របស់ណឺរ៉ូន ឬ បញ្ចេញលទ្ធផល 0 បើតូចជាង។ តម្លៃនៃកម្រិតកំណត់របស់ណឺរ៉ូនហៅថា Threshold។ នៅទីនេះ θ ជាតម្លៃThreshold ។

$$y = \begin{cases} 1 & (w_1x_1 + w_2x_2 > \theta) \\ 0 & (w_1x_1 + w_2x_2 \leq \theta) \end{cases}$$



រូបទី១ Perceptronដែលទទួលសញ្ញាណឬធាតុចូល(input) 2

ក្នុងការអនុវត្តភាគច្រើន គេច្រើនបម្លែងទម្រង់ខាងលើជាទម្រង់ដែលមានកម្រិតកំណត់នៃណឺរ៉ូនស្មើ0 ដោយហៅតម្លៃ θ ដែលត្រូវបានបញ្ជូនទៅអង្គទី១($b = -\theta$) ដោយ bias ។

$$y = \begin{cases} 1 & (b + w_1x_1 + w_2x_2 > 0) \\ 0 & (b + w_1x_1 + w_2x_2 \leq 0) \end{cases}$$

2. បង្ហាញសៀគ្វីឡូស៊ីកងាយៗដោយម៉ូឌែលPerceptron

ដើម្បីស្វែងយល់អំពីដំណើរការរបស់Perceptron នៅទីនេះយើងលើកយកសៀគ្វីឡូស៊ីក (Logic circuits) ងាយៗដូចជា AND gate, OR gate មកបង្ហាញដោយប្រើម៉ូឌែលPerceptron ។

2.1. AND gate

ដូចដែលអ្នកបានដឹង AND gate ផ្តល់នូវលទ្ធផលអាស្រ័យនឹងតម្លៃភាពពិតនៃធាតុចូលរបស់វាដូចក្នុងតារាងខាងក្រោម។

តារាងទី១ តម្លៃភាពពិតនៃAND gate

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	1

ក្នុងករណីនេះ បើយើងយកPerceptronមួយដែលទទួលធាតុចូលពីរនិងមានតម្លៃទម្ងន់និងកម្រិតកំណត់(threshold)៖ $(w_1, w_2, \theta) = (0.5, 0.5, 0.8)$ នោះយើងអាចបង្ហាញAND gateដោយPerceptronបាន។ ឧទាហរណ៍ ករណី $(x_1, x_2) = (1,0)$ នោះ $w_1x_1 + w_2x_2 = 0.5 < 0.8$ ហេតុនេះ $y = 0$ ។ ករណី $(x_1, x_2) = (1,1)$ នោះ $w_1x_1 + w_2x_2 = 1 > 0.8$ ហេតុនេះ $y = 1$ ។

2.2. OR gate

OR gate ផ្ដល់នូវលទ្ធផលអាស្រ័យនឹងតម្លៃភាពពិតនៃធាតុចូលរបស់វាដូចក្នុងតារាងខាងក្រោម។

តារាងទី២ តម្លៃភាពពិតនៃOR gate

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	1

ក្នុងករណីនេះ បើយើងយកPerceptronមួយដែលទទួលធាតុចូលពីរនិងមានតម្លៃទម្ងន់និងកម្រិតកំណត់(threshold): $(w_1, w_2, \theta) = (0.5, 0.5, 0.2)$ នោះយើងអាចបង្ហាញOR gateដោយPerceptronបាន។ ឧទាហរណ៍ ករណី $(x_1, x_2) = (1,0)$ នោះ $w_1x_1 + w_2x_2 = 0.5 > 0.2$ ហេតុនេះ $y = 1$។ ករណី $(x_1, x_2) = (0,0)$ នោះ $w_1x_1 + w_2x_2 = 0 < 0.2$ ហេតុនេះ $y = 0$។

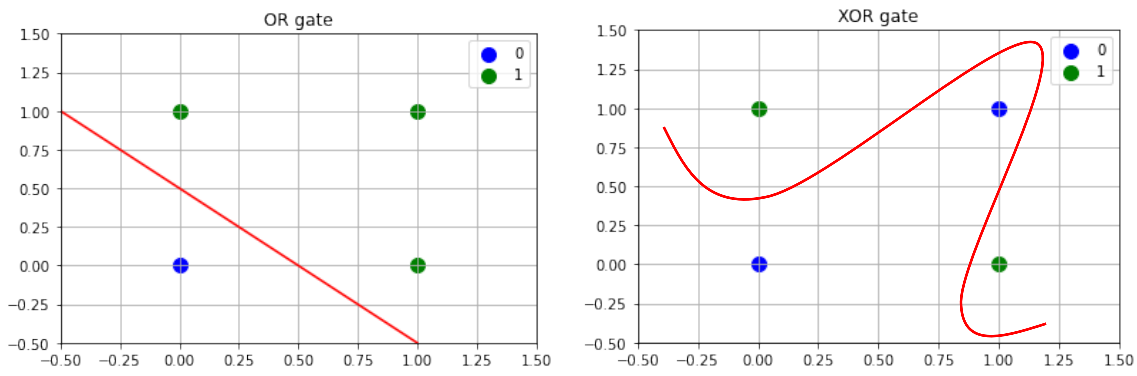
ដូចដែលបង្ហាញខាងលើ ដោយប្រើPerceptronយើងអាចបង្ហាញសៀគ្វីឡូស៊ីកងាយៗបាន។ ចំណុចសំខាន់នៅទីនេះគឺ ចំពោះAND gate, OR gate ដែលមានទម្រង់សៀគ្វីឡូស៊ីកផ្សេងៗគ្នាក្ដី Perceptronដែលយើងប្រើមានគោលគំនិតឬទម្រង់តែមួយមិនប្រែប្រួលឡើយ។ អ្វីដែលខុសគ្នាគឺតម្លៃនៃប៉ារ៉ាម៉ែត្រ(ទម្ងន់និងកម្រិតកំណត់របស់ណឺរ៉ូន)តែប៉ុណ្ណោះ។ ពោលគឺដោយប្រើទម្រង់នៃម៉ូឌែលតែមួយយើងអាចបង្ហាញទម្រង់នៃសៀគ្វីឡូស៊ីកដែលជាគ្រឹះនៃសៀគ្វីអេឡិចត្រូនិចនានាបានដោយគ្រាន់តែកែសម្រួលតម្លៃនៃប៉ារ៉ាម៉ែត្ររបស់វាតែប៉ុណ្ណោះ។

3. ព្រំដែនសមត្ថភាពនៃPerceptron

យើងឃើញថាPerceptron អាចបង្ហាញ AND gate, OR gateបានយ៉ាងងាយ។ បន្តទៅនេះយើងនឹងពិនិត្យលើករណីនៃ XOR gate។

តារាងទី៣ តម្លៃភាពពិតនៃXOR gate

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0



រូបទី២ ប្រៀបធៀបករណីOR gate និង XOR gate

មុននឹងឈានទៅមើលករណីXOR gate យើងបង្ហាញព្រមគ្នាជាមួយករណីOR gateដោយប្រើក្រាបដូចរូបទី២ ។ ក្នុងករណី OR gate យើងអាចកំណត់តម្លៃប៉ារ៉ាម៉ែត្ររបស់Perceptronបានដូចឧទាហរណ៍ក្នុងចំណុច2. ដែលលទ្ធផល1 ឬ 0 អាចបែងចែកដាច់ពីគ្នាបានដោយបន្ទាត់ត្រង់មួយបាន ។ ផ្ទុយពីនេះ ករណី XOR gate យើងមិនអាចកំណត់បន្ទាត់ត្រង់ដើម្បីបែងចែកករណីលទ្ធផល1 ឬ 0 បានឡើយ ។ ពោលគឺមិនអាចកំណត់ប៉ារ៉ាម៉ែត្រណាដែលអាចឱ្យPerceptronបង្ហាញXOR gate បានទេ ។

ទិន្នន័យដូចក្នុងករណីOR gate ហៅថាទិន្នន័យដែលអាចបែងចែកលីនេអ៊ែរបាន(linear seperable) ឯទិន្នន័យដូចក្នុងករណីXOR gate ហៅថាទិន្នន័យដែលមិនអាចបែងចែកលីនេអ៊ែរបាន (linear non-seperable) ។ ហេតុនេះ យើងអាចនិយាយបានថា Perceptronដែលមានទម្រង់ដូចណែនាំក្នុងចំណុច1. មិនអាចប្រើជាម៉ូឌែលសម្រាប់ទិន្នន័យមិនអាចបែងចែកលីនេអ៊ែរបានល្អឡើយ ។

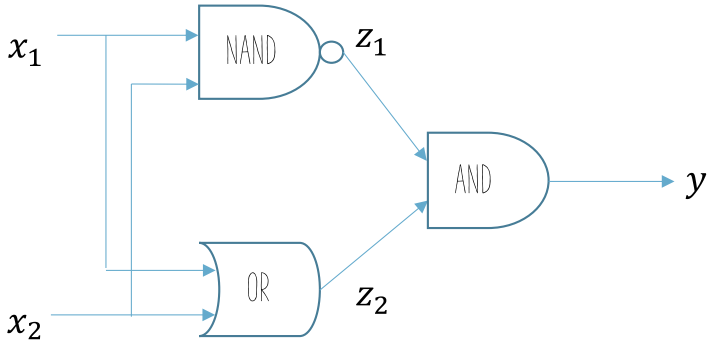
4. Perceptronច្រើនថ្នាក់ (Multilayer Perceptron)

ដូចដែលបានពិនិត្យខាងលើ Perceptronទម្រង់ធម្មតាមិនអាចធ្វើការពណ៌នាXOR gate បានប្រសើរឡើយ ។ ដើម្បីស្វែងយល់ពីដំណោះស្រាយតាមរយៈការបង្កើនចំនួនថ្នាក់នៃPerceptron យើងពិនិត្យលើការបង្ហាញទម្រង់XOR gate ដោយប្រើបង្គុំនៃសៀគ្វីឡូស៊ីកមូលដ្ឋានAND gate, OR gate, NAND gate ។

4.1. បង្ហាញXOR gate ដោយប្រើ AND gate, OR gate, NAND gate

នៅទីនេះយើងនឹងមិនធ្វើការបកស្រាយលំអិតអំពីលក្ខណៈនៃសៀគ្វីឡូស៊ីកឡើយ ប៉ុន្តែតាមពិតទៅXOR gate អាចបង្ហាញដោយប្រើ AND gate, OR gate, NAND gateបានដោយធ្វើតំណភ្ជាប់ដូចរូបទី៣ ។ ដោយសារតែAND gate, OR gate, NAND gate អាចបង្ហាញដោយប្រើ

Perceptron ទម្រង់ធម្មតាដូចក្នុងចំណុច1. 2.ខាងលើបាន ហេតុនេះ គំនិតសំខាន់ដែលយើងអាចសិក្សាពីចំណុចនេះគឺថា យើងអាចផ្គុំPerceptronធម្មតាជាច្រើនថ្នាក់ដើម្បីបង្ហាញXOR gateបាន ។



រូបទី៣ ការបង្ហាញXOR gate ដោយប្រើAND gate, OR gate, NAND gate

4.2. ការបង្កើតPerceptronងាយៗនិងច្រើនថ្នាក់ដោយប្រើPython

```python
import numpy as np
```

4.2.1. ករណីទម្រង់ធម្មតា(មិនប្រើទម្រង់Bias)

```python
def AND(x1, x2):
  (w1,w2,theta) = (0.5, 0.5, 0.8)
  t = w1*x1 + w2*x2
  y = 1 if t > theta else 0
  return y
```

```python
def OR(x1, x2):
  (w1,w2,theta) = (0.5, 0.5, 0.2)
  t = w1*x1 + w2*x2
  y = 1 if t > theta else 0
  return y
```

```python
def NAND(x1, x2):
  (w1,w2,theta) = (-0.5, -0.5, -0.8)
  t = w1*x1 + w2*x2
  y = 1 if t > theta else 0
  return y
```

4.2.2. ករណីទម្រង់ប្រើBias

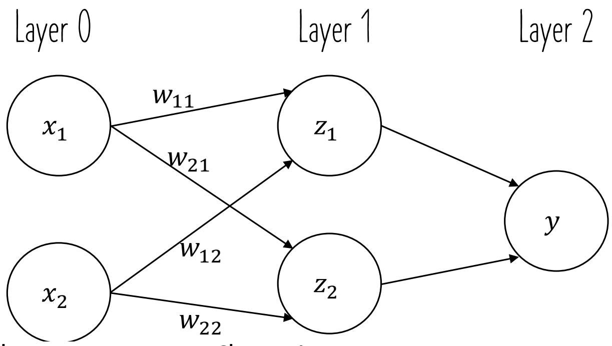
```
def AND(x1, x2):
  x_vec = np.array([x1,x2])
  w = np.array([0.5,0.5])
  b = -0.8
  t = np.sum(w*x_vec) + b
  y = 1 if t > 0 else 0
  return y
```

```
def OR(x1, x2):
  x_vec = np.array([x1,x2])
  w = np.array([0.5,0.5])
  b = -0.2
  t = np.sum(w*x_vec) + b
  y = 1 if t > 0 else 0
  return y
```

```
def NAND(x1, x2):
  x_vec = np.array([x1,x2])
  w = np.array([-0.5,-0.5])
  b = 0.8
  t = np.sum(w*x_vec) + b
  y = 1 if t > 0 else 0
  return y
```

4.2.3. XOR gate

```
def XOR(x1,x2):
  s1 = NAND(x1,x2)
  s2 = OR(x1,x2)
  y = AND(s1,s2)
  return y
```



រូបទី៤ Perceptronច្រើនថ្នាក់ (Multilayer Perceptron)

ទម្រង់នៃXOR gateដោយបង្គុំនៃAND gate, OR gate, NAND gateដូចក្នុងរូបទី៣ អាចប្រដូចបានជាករណីមួយនៃករណីទូទៅរបស់បង្គុំនៃPerceptronទម្រង់ធម្មតាច្រើនបញ្ចូលគ្នាដូចក្នុងរូបទី៤។ ទម្រង់បែបនេះហៅថា Multilayer Perceptron ដែលក្នុងអត្ថបទនេះនិងបន្តបន្ទាប់យើងកំណត់ហៅថា Perceptronច្រើនថ្នាក់។

ដូចក្នុងករណីXOR gate ដែរ ការប្រើMultilayer Perceptronអាចឱ្យយើងបង្ហាញទិន្នន័យដែលមិនអាចបែងចែកលីនេអ៊ែរដោយម៉ូឌែលPerceptronបាន។ ការបង្កើនចំនួនថ្នាក់(Layer)ក្នុងPerceptronនឹងជួយបង្កើនសមត្ថភាពរបស់វាក្នុងការពណ៌នាលក្ខណៈរបស់ទិន្នន័យដែលកាន់តែស្មុគស្មាញបានដែលនេះជាគោលគំនិតគ្រឹះក្នុង Artificial Neural Network ឬ Deep Learningដែលយើងនឹងលើកមកសិក្សាក្នុងអត្ថបទក្រោយៗ។

5. ការកំណត់ប៉ារ៉ាម៉ែត្រនៃPerceptron ១ថ្នាក់(Perceptron Algorithm)

ចំពោះPaceptronទម្រង់ធម្មតា១ថ្នាក់ យើងអាចកំណត់តម្លៃនៃប៉ារ៉ាម៉ែត្រពីទិន្នន័យដែលមានបានដោយអនុវត្តតាមវិធីសាស្ត្រខាងក្រោម។នៅទីនេះ$\boldsymbol{w}^{(t)}$សម្គាល់តម្លៃនៃប៉ារ៉ាម៉ែត្រនៅដំណាក់កាលផ្លាស់ប្តូរទីt ,η សម្គាល់តម្លៃនៃកម្រិតផ្លាស់ប្តូរប៉ារ៉ាម៉ែត្រដែលហៅថា learning rate។

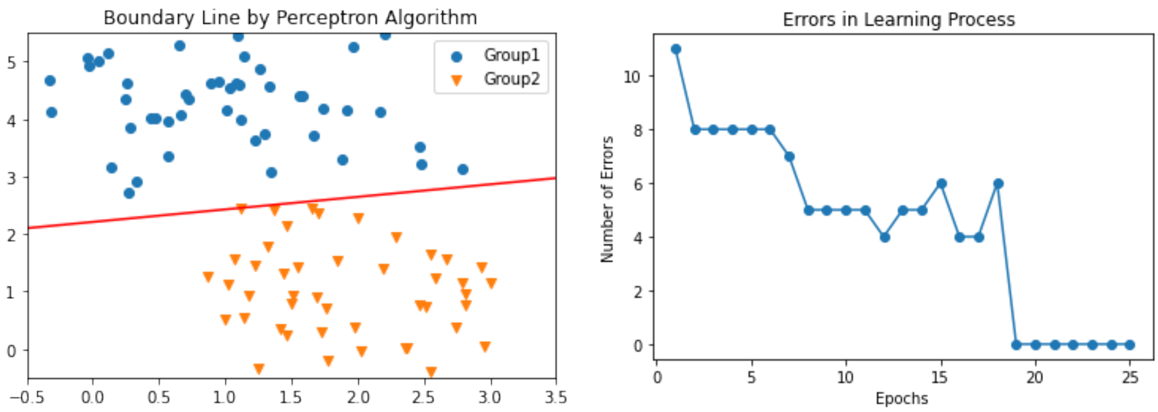
(***ជំហានទី១***) កំណត់តម្លៃដើមនៃប៉ារ៉ាម៉ែត្រ $\boldsymbol{w}$ ដោយ 0 ឬតម្លៃពីបំណែងចែកចៃដន្យ

(***ជំហានទី២***) ចំពោះទិន្នន័យ(training data) $(\boldsymbol{x}_i, y_i)$ អនុវត្តជំហានខាងក្រោមរហូតដល់គ្មានកំហុសក្នុងការប៉ាន់ស្មានតម្លៃ ពោលគឺ $\boldsymbol{\Delta w}^{(t)} = \boldsymbol{0}$

ក. គណនាលទ្ធផលនៃណឺរ៉ូន $\hat{y}_i = \boldsymbol{x}_i^\top \boldsymbol{w}^{(t)}$ ដោយប្រើតម្លៃប៉ារ៉ាម៉ែត្របច្ចុប្បន្ន

ខ. ផ្លាស់ប្តូរតម្លៃនៃប៉ារ៉ាម៉ែត្រ

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} + \boldsymbol{\Delta w}^{(t)}$$
$$\boldsymbol{\Delta w}^{(t)} = \eta(y_i - \hat{y}_i)\boldsymbol{x}_i$$



រូបទី៥ ការធ្វើចំណាត់ថ្នាក់ក្រុមទិន្នន័យ២ក្រុមដោយPerceptron Algorithm

```
import numpy as np
class Perceptron(object):
  def __init__(self, learning_rate=0.01, iteration_number=100, random_state=1):
    self.eta = learning_rate
    self.n_iteration = iteration_number
    self.random_state = random_state

  def fit(self, X, y):
    rand_gen = np.random.RandomState(self.random_state)
    self.w = rand_gen.normal(loc=0.0, scale=0.1, size=1+X.shape[1])
    self.errors = []
    XP = np.ones((X.shape[0],X.shape[1]+1))
    XP[:,:-1]=X

    for t in range(self.n_iteration):
      error = 0
      for xi,yi in zip(XP,y):
        delta = self.eta * (yi - self.predict(xi))
        self.w += delta * xi
        error += int(delta != 0.0)
      self.errors.append(error)
    return self

  def predict(self, X):
    z = X@self.w
    return np.where(z >= 0.0, 1, 0)
```

Feedforward Neural Network (FNN)

ក្នុងអត្ថបទមុន យើងបានសិក្សាអំពីPerceptron ដែលជាម៉ូឌែលអាចបែងចែកទិន្នន័យបំណែងចែកលីនេអ៊ែរបានយ៉ាងងាយដោយការកំណត់តម្លៃប៉ារ៉ាម៉ែត្រសមស្រប។ លើសពីនេះ ក្នុងករណីទិន្នន័យមិនអាចបែងចែកលីនេអ៊ែរ ការបង្កើនចំនួនថ្នាក់នៃPerceptronត្រូវបានប្រើប្រាស់។ ម៉ូឌែលបែបនេះហៅថា Multilayer Perceptron។ ដោយការភ្ជាប់ណឺរ៉ូន (node) ច្រើនបន្តគ្នាជាច្រើនថ្នាក់ដែលស្រដៀងគ្នានឹងទម្រង់នៃប្រព័ន្ធប្រសាទរបស់ភាវៈរស់ផងនោះម៉ូឌែលបែបនេះក៏ត្រូវបានគេហៅថា Aritificial Neural Network ផងដែរ។

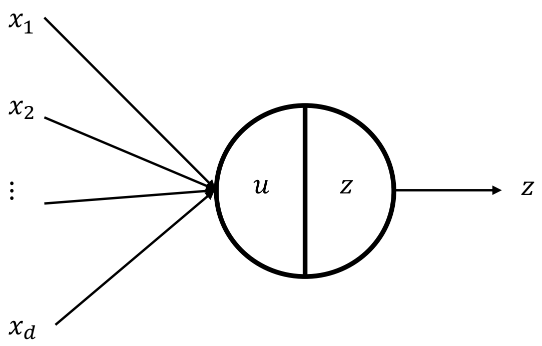
1. លទ្ធផលបញ្ជូនបន្តនៃណឺរ៉ូន

Feedforward Neural Network (FNN) គឺជាទម្រង់មួយនៃArtificial Neural Network ដែលមានណឺរ៉ូន(node)ច្រើនតម្រៀបគ្នាជាថ្នាក់និងភ្ជាប់គ្នានិងគ្នារវាងថ្នាក់នៅជាប់បន្តបន្ទាប់គ្នាដោយទម្ងន់ផ្ទាល់ខ្លួន។ សញ្ញាណឬធាតុចូលនៃFNNត្រូវបានបញ្ជូនពីផ្នែកថ្នាក់ធាតុចូល(input layer) ទៅកាន់ផ្នែកនៃថ្នាក់លទ្ធផល(output layer)តាមទិសតែមួយ។ លទ្ធផលដែលបញ្ចេញដោយណឺរ៉ូននៅថ្នាក់លទ្ធផលត្រូវបានគណនាដូចក្នុងករណីPerceptronទម្រង់ធម្មតាដែរ ប៉ុន្តែនៅទីនេះលទ្ធផលមិនគ្រាន់តែប្រៀបធៀបផលបូកនៃធាតុចូលនិងកម្រិតកំណត់(threshold)នៃណឺរ៉ូនប៉ុណ្ណោះទេ តែអនុគមន៍មិនលីនេអ៊ែរត្រូវបានអនុវត្តលើលទ្ធផលនៃផលបូកនោះដើម្បីកំណត់នូវលទ្ធផលដែលត្រូវបញ្ជូនបន្ត។ អនុគមន៍ដែលអនុវត្តលើលទ្ធផលនៃផលបូកធាតុចូលនេះហៅថា អនុគមន៍សកម្ម(activation function)។ យើងនឹងធ្វើការបកស្រាយលម្អិតអំពីអនុគមន៍សកម្មនៅចំណុចបន្ទាប់។

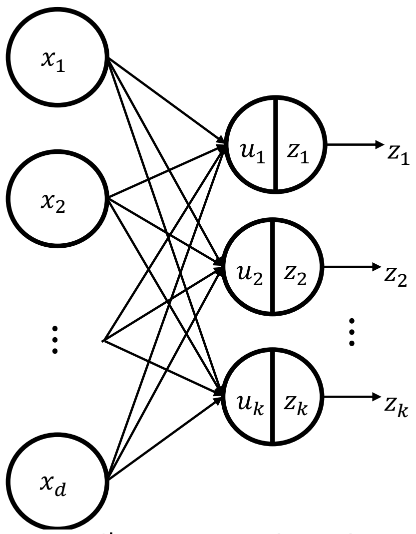
នៅពេលដែលធាតុចូល $x_1, x_2, \dots, x_d$ ត្រូវបានបញ្ជូនមកកាន់ណឺរ៉ូន(រូបទី១) ដែលមានទម្ងន់ផ្ទាល់នៃធាតុចូលរៀងគ្នា $w_1, w_2, \dots, w_d$ នោះ ផលបូកនៃធាតុចូលសរុបកំណត់ដោយ u និងលទ្ធផលបញ្ជូនបន្តនៃណឺរ៉ូននោះត្រូវបានកំណត់ដោយ z ដូចទម្រង់ខាងក្រោម។ នៅទីនេះ b ហៅថា bias ។

$$u = w_1x_1 + w_2x_2 + \cdots + w_dx_d + b$$

$$z = f(u) = f(w_1x_1 + w_2x_2 + \cdots + w_dx_d + b)$$



រូបទី១ ធាតុចូលនិងលទ្ធផលនៃណឺរ៉ូនមួយ



រូបទី២ FNNទម្រង់2ថ្នាក់

ក្នុងករណីទម្រង់2ថ្នាក់ដូចក្នុងរូបទី២ សញ្ញាណត្រូវបានបញ្ជូនបន្តបន្ទាប់។ សន្មតថានៅថ្នាក់ទី១មានណឺរ៉ូនចំនួន d និង នៅថ្នាក់ទី២មានណឺរ៉ូនចំនួន k នោះលទ្ធផលបញ្ជូនបន្តនៃណឺរ៉ុននៅថ្នាក់លទ្ធផលទី២ត្រូវបានបង្ហាញក្នុងទម្រង់ខាងក្រោម $(i = 1,2, \dots, k)$។

$$u_i = \sum_{j=1}^{d} w_{ij} x_j + b_i$$

$$z_i = f(u_i)$$

យើងក៏អាចបង្ហាញជាទម្រង់វ៉ិចទ័រនិងម៉ាទ្រីសដូចខាងក្រោមផងដែរ។

$$\boldsymbol{u} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}$$

$$\boldsymbol{z} = \boldsymbol{f}(\boldsymbol{u})$$

$$\boldsymbol{u}=\begin{bmatrix}u_1\\ \vdots\\ u_k\end{bmatrix},\boldsymbol{x}=\begin{bmatrix}x_1\\ \vdots\\ x_d\end{bmatrix},\boldsymbol{b}=\begin{bmatrix}b_1\\ \vdots\\ b_k\end{bmatrix},\boldsymbol{z}=\begin{bmatrix}z_1\\ \vdots\\ z_k\end{bmatrix},\boldsymbol{f}(\boldsymbol{u})=\begin{bmatrix}f(u_1)\\ \vdots\\ f(u_k)\end{bmatrix}$$

$$\boldsymbol{W}=\begin{bmatrix}w_{11} & \cdots & w_{1d}\\ \vdots & \ddots & \vdots\\ w_{k1} & \cdots & w_{kd}\end{bmatrix}$$

2. អនុគមន៍សកម្ម Activation Function

អនុគមន៍សកម្ម(activation function) គឺជាអនុគមន៍ដែលគ្រប់គ្រងលើកម្រិតនៃការបញ្ជូនបន្តនូវលទ្ធផលរបស់ណឺរ៉ូននិមួយៗ។ ជាទូទៅអនុគមន៍សកម្មមានទម្រង់ជាអនុគមន៍មិនលីនេអ៊ែរកើនដាច់ខាត។ មានទម្រង់ជាច្រើនត្រូវបានប្រើជាអនុគមន៍សកម្មដូចជា អនុគមន៍Sigmoid,អនុគមន៍Tanh,អនុគមន៍Softmax,អនុគមន៍ReLU(rectified linear function)ជាដើម។

អនុគមន៍Sigmoid $\sigma(\cdot)$ មានដែនកំណត់លើសំណុំចំនួនពិត$(-\infty,\infty)$ និងយកសំណុំរូបភាពលើចន្លោះបើក$(0,1)$។ អនុគមន៍បែបនេះអាចផ្តល់នូវលទ្ធផលដែលយើងអាចបកស្រាយបានជាតម្លៃប្រូបាបនៃការបញ្ជូនបន្តឬមិនបញ្ជូនបន្ត(1/0)។

$$f(u)=\sigma(u)=\frac{1}{1+e^{-u}}$$

អនុគមន៍Tanh $\tanh(\cdot)$ មានដែនកំណត់លើសំណុំចំនួនពិត$(-\infty,\infty)$ និងយកសំណុំរូបភាពលើចន្លោះបើក$(-1,1)$។ អនុគមន៍បែបនេះមានលក្ខណៈស្រដៀងនឹងអនុគមន៍Sigmoidដែរ ដែលអាចផ្តល់នូវលទ្ធផលដែលមានបម្រែបម្រួលតិចតួចក្បែរតម្លៃថេរនៅពេលដែលតម្លៃនៃធាតុចូលធំខ្លាំងដល់កម្រិតណាមួយ និងប្រែប្រួលខ្លាំងនៅពេលដែលធាតុចូលមានតម្លៃក្បែរ0។

$$f(u)=\tanh(u)=\frac{e^{u}-e^{-u}}{e^{u}+e^{-u}}$$

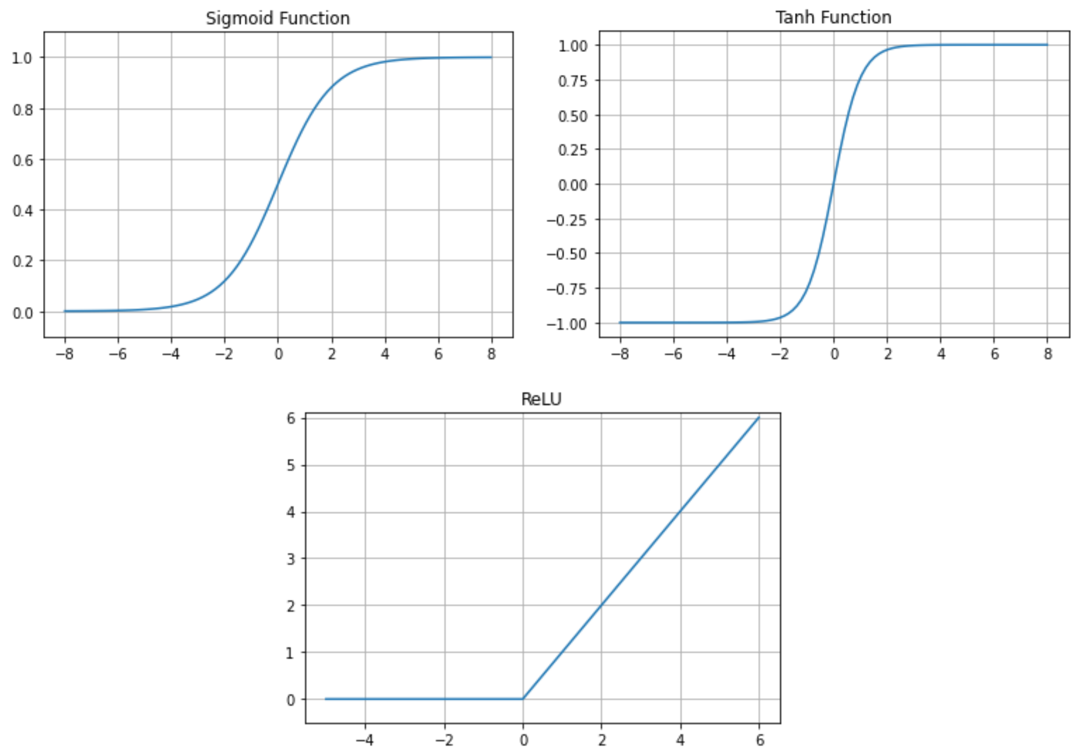
អនុគមន៍Softmax $\mathrm{softmax}(\cdot)$ មានដែនកំណត់លើវ៉ិចទ័រចំនួនពិត$\mathbb{R}^d$ និងយកសំណុំរូបភាពលើចន្លោះបើក$(0,1)^d$ដែលមានផលបូកគ្រប់កំប៉ូសង់ស្មើ 1។ អនុគមន៍បែបនេះអាចផ្តល់នូវលទ្ធផលដែលយើងអាចបកស្រាយបានជាតម្លៃប្រូបាបនៃលទ្ធផលដែលអាចចេញជាdប្រភេទផ្សេងៗគ្នាបាន។ ក្នុងករណី$d=2$ អនុគមន៍នេះសមមូលនឹងអនុគមន៍Sigmoid។ចំពោះ $\boldsymbol{u}=(u_1 \quad \cdots \quad u_d)^\top$

$$f(\boldsymbol{u})=\mathrm{Softmax}(\boldsymbol{u})=(\mathrm{Softmax}(u_1) \quad \cdots \quad \mathrm{Softmax}(u_d))^\top$$

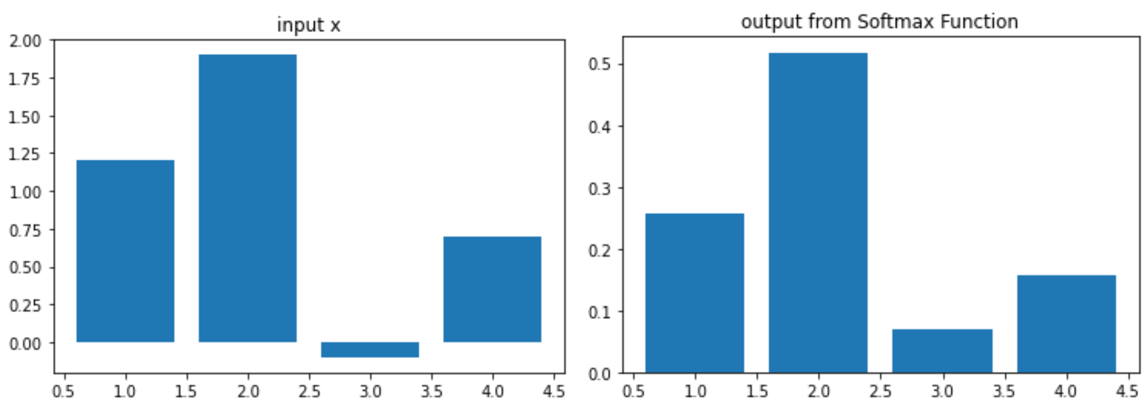
$$\mathrm{Softmax}(u_i)=\frac{e^{u_i}}{\sum_{j=1}^{d}e^{u_j}}$$

អនុគមន៍rectifier linear: $\mathrm{ReLU}(\cdot)$ មានដែនកំណត់លើសំណុំចំនួនពិត$(-\infty,\infty)$ និងយកសំណុំរូបភាពលើចន្លោះបើក$(0,\infty)$។ នៅពេលដែលធាតុចូលមានតម្លៃតូចជាងឬស្មើសូន្យ លទ្ធផលបញ្ជូនបន្តត្រូវបានកំណត់ដោយ 0 និងបញ្ជូនបន្តនូវតម្លៃដូចធាតុចូលដដែលបើធាតុចូលមានតម្លៃធំជាងឬស្មើសូន្យ។អនុគមន៍បែបនេះមានលក្ខណៈស្រដៀងនឹងអនុគមន៍ដឺក្រេទី១(លីនេអ៊ែរ)ដែរ ដែលអាចប្រើក្នុងករណីប៉ាន់ស្មានទាំងម៉ូឌែលលីនេអ៊ែរនិងមិនលីនេអ៊ែរបានល្អ។

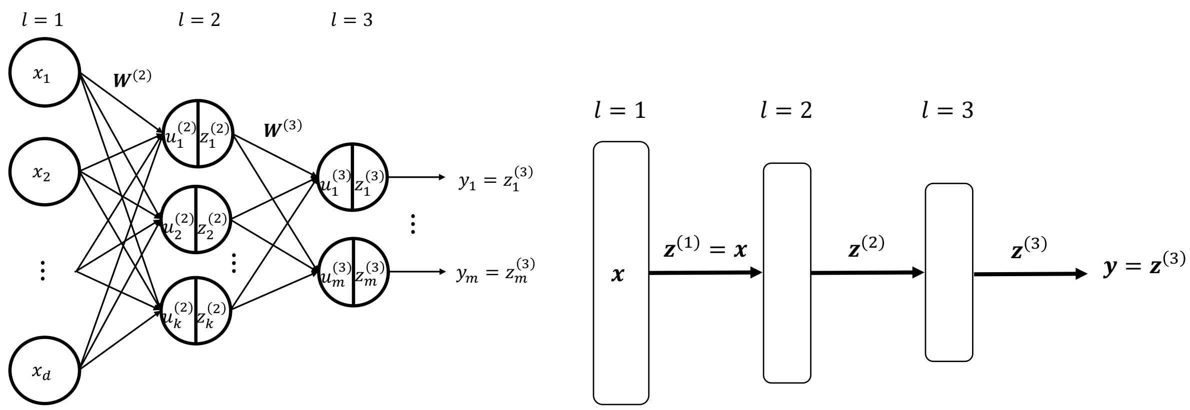
$$f(u) = \mathrm{ReLU}(u) = \max(u, 0)$$



រូបទី៣ ក្រាបតាងអនុគមន៍សកម្ម sigmoid, tanh, ReLU



រូបទី៤ ទំនាក់ទំនងរវាងធាតុចូលនិងលទ្ធផលដោយ Softmax Function



រូបទី៥ Multilayer Network

3. បណ្ដាញច្រើនថ្នាក់ (Multilayer Network)

នៅចំណុចនេះយើងពិនិត្យលើករណីម៉ូឌែលដែលមានច្រើនថ្នាក់ដូចក្នុងរូបទី៥។ ព័ត៌មាន (សញ្ញាណឬធាតុចូល)ត្រូវបានបញ្ជូនតាមលំដាប់លំដោយពីថ្នាក់នៅខាងឆ្វេងទៅស្តាំ។ នៅទីនេះយើងកំណត់ហៅថ្នាក់និមួយៗដោយ$l = 1,2,3,\ldots$។ ក្នុងរូបខាងលើថ្នាក់ $l = 1$ពោលគឺថ្នាក់នៅខាងឆ្វេងបំផុតហៅថាថ្នាក់នៃធាតុចូល(input layer), ថ្នាក់$l = 2$ ហៅថាថ្នាក់នៃធាតុកណ្ដាល(internal layer, hidden layer), ថ្នាក់$l = 3$ ពោលគឺថ្នាក់នៅខាងស្តាំបំផុត ហៅថាថ្នាក់នៃលទ្ធផល(output layer) ។ លទ្ធផលនៅថ្នាក់និមួយៗអាចសរសេរជាទម្រង់ដូចខាងក្រោម។

$$\boldsymbol{u}^{(2)} = \boldsymbol{W}^{(2)}\boldsymbol{x} + \boldsymbol{b}^{(2)} \quad , \quad \boldsymbol{z}^{(2)} = \boldsymbol{f}\left(\boldsymbol{u}^{(2)}\right)$$

$$\boldsymbol{u}^{(3)} = \boldsymbol{W}^{(3)}\boldsymbol{z}^{(2)} + \boldsymbol{b}^{(3)} \quad , \quad \boldsymbol{z}^{(3)} = \boldsymbol{f}\left(\boldsymbol{u}^{(3)}\right)$$

ជាទូទៅ លទ្ធផលនៅថ្នាក់កណ្ដាលកំណត់ដោយ

$$\boldsymbol{u}^{(l+1)} = \boldsymbol{W}^{(l+1)}\boldsymbol{z}^{(l)} + \boldsymbol{b}^{(l+1)} \quad , \quad \boldsymbol{z}^{(l+1)} = \boldsymbol{f}\left(\boldsymbol{u}^{(l+1)}\right)$$

និង លទ្ធផលនៅថ្នាក់លទ្ធផលចុងក្រោយកំណត់ដោយ

$$\boldsymbol{y} \equiv \boldsymbol{z}^{(l+1)}$$

ដូចដែលបានឃើញក្នុងទម្រង់គណនាខាងលើ ក្នុងFNN សញ្ញាណត្រូវបានបញ្ជូនបន្តបន្ទាប់ដោយការគណនាក្នុងរបៀបដូចគ្នាពីមួយថ្នាក់ទៅមួយថ្នាក់។ នេះគឺជាប្រភពដែលម៉ូឌែលនេះត្រូវបានហៅថា feedforward ។ ទម្ងន់ផ្ទាល់នៃណឺរ៉ូនចំពោះធាតុចូលតាមថ្នាក់និមួយៗ$\boldsymbol{W}^{(l)}$ និង bias $\boldsymbol{b}^{(l)}$ ត្រូវបានហៅជារួមថាជា ប៉ារ៉ាម៉ែត្រនៃបណ្ដាញ។ ក្នុងអត្ថបទនេះនិងបន្តបន្ទាប់យើងកំណត់ហៅដោយងាយនូវ បណ្ដាញដែលមានប៉ារ៉ាម៉ែត្រ(ហៅជារួម)$\boldsymbol{w}$ និងធាតុចូល $\boldsymbol{x}$ ដោយ $\boldsymbol{y}(\boldsymbol{x};\boldsymbol{w})$។

4. ការកំណត់ទម្រង់ណឺរ៉ូននៅថ្នាក់លទ្ធផលនិងអនុគមន៍លម្អៀង

4.1. ប្រភេទនៃបញ្ហានិងវិធីសាស្ត្ររៀន (Learning)

ដូចដែលបានរៀបរាប់ពីអត្ថបទមុននិងចំណុចខាងលើ បណ្តាញដែលបង្ហាញដោយទម្រង់នៃអនុគមន៍ច្រើនអថេរ $\boldsymbol{y}(\boldsymbol{x};\boldsymbol{w})$នឹងប្រែប្រួលនៅពេលដែលប៉ារ៉ាម៉ែត្ររបស់វាត្រូវបានផ្លាស់ប្តូរ។ ការជ្រើសរើសប៉ារ៉ាម៉ែត្របានល្អ នឹងធ្វើឱ្យបណ្តាញ (network)អាចបង្ហាញនូវអនុគមន៍ឬបញ្ហាដែលមានបានល្អប្រសើរ។

សន្មតថា អនុគមន៍ឬបញ្ហាជាគោលដៅដែលយើងចង់បង្ហាញដោយNeural Network មិនប្រែប្រួលសណ្ឋានខាងក្នុងរបស់វាឡើយ ហើយទទួលជាតុចូល $\boldsymbol{x}$ និងបញ្ជូនចេញនូវលទ្ធផល$\boldsymbol{t}$ ។ គូនៃទិន្នន័យបែបនេះជាច្រើនត្រូវបានផ្តល់ឱ្យ $\{(\boldsymbol{x}_1,\boldsymbol{t}_1),\ldots,(\boldsymbol{x}_N,\boldsymbol{t}_N)\}$ ។ នៅក្នុងអត្ថបទនេះ និងអត្ថបទបន្តបន្ទាប់ គូនិមួយៗហៅថាជាគម្រូសម្រាប់រៀន (training sample) ហើយសំណុំទាំងមូលហៅថាសំណុំទិន្នន័យសម្រាប់រៀន (training data) ។

ដោយធ្វើការកែសម្រួលនិងកំណត់នូវតម្លៃប៉ារ៉ាម៉ែត្រ យើងអាចធ្វើការបង្ហាញទំនាក់ទំនងដែលមានក្នុងទិន្នន័យឡើងវិញបានដោយបណ្តាញ (network)របស់យើង។ ពោលគឺចំពោះគម្រូសម្រាប់រៀន$(\boldsymbol{x}_n,\boldsymbol{t}_n)$និមួយៗ យើងចង់កំណត់នូវប៉ារ៉ាម៉ែត្រណាដែលធ្វើឱ្យយើងអាចទទួលបាន $\boldsymbol{y}(\boldsymbol{x}_n;\boldsymbol{w})$ ដែលមានតម្លៃជិតបំផុតនៅនឹង$\boldsymbol{t}_n$ ។ ដំណើរកំណត់រកនូវប៉ារ៉ាម៉ែត្រដោយប្រើសំណុំទិន្នន័យសម្រាប់រៀនបែបនេះសន្មតហៅថាជាដំណើរការរៀន (learning process) ។

ហេតុនេះការប្រៀបធៀបរវាងតម្លៃលទ្ធផល$\boldsymbol{y}(\boldsymbol{x}_n;\boldsymbol{w})$ ដែលផ្តល់ដោយបណ្តាញនិងតម្លៃ$\boldsymbol{t}_n$ ត្រូវបានធ្វើឡើង។ ដើម្បីបង្ហាញពីកម្រិតជិតគ្នានៃតម្លៃទាំងពីរយើងកំណត់រង្វាស់សម្រាប់វាស់កម្រិតនេះ។ រង្វាស់នេះយើងសន្មតហៅថាជា អនុគមន៍កម្រិតលម្អៀង (error function,loss function) ។ ការកំណត់ប្រភេទនៃអនុគមន៍កម្រិតលម្អៀងខុសគ្នាទៅតាមប្រភេទចំណោទបញ្ហាដែលយើងចង់ដោះស្រាយ។

តារាងទី១ ប្រភេទនៃអនុគមន៍សកម្មនិងកម្រិតលម្អៀងតាមប្រភេទចំណោទ

ប្រភេទចំណោទបញ្ហា	អនុគមន៍សកម្មនៅថ្នាក់លទ្ធផល	អនុគមន៍កម្រិតលម្អៀង
តម្រែតម្រង់ (regression)	Identity function $y(x)=x$	ផលបូកការេនៃលម្អៀង
ចំណាត់ថ្នាក់២ក្រុម	Sigmoid function	Cross entropy (2 classes)
ចំណាត់ថ្នាក់ច្រើនក្រុម	Softmax function	Cross entropy

4.2. ចំណោទតម្រែតម្រង់

ចំណោទតម្រែតម្រង់ (Regression) គឺជាប្រភេទចំណោទដែលធ្វើការកំណត់នូវអនុគមន៍ដើម្បីបង្ហាញនូវទំនាក់ទំនងរវាងធាតុចូលនិងលទ្ធផលដែលមានទម្រង់ជាអថេរជាប់។ ហេតុនេះក្នុង

ករណីនៃចំណោទតម្រែតម្រង់ យើងកំណត់យកអនុគមន៍សកម្ម(activation function)នៅថ្នាក់លទ្ធផលនៃបណ្ដាញ(FNN)ដោយអនុគមន៍ដែលផ្ដល់នូវសំណុំរូបភាពដូចដែននៃលទ្ធផលរបស់សំណុំទិន្នន័យសម្រាប់រៀន។ ឧទាហរណ៍ ក្នុងករណីដែលសំណុំទិន្នន័យសម្រាប់រៀនមានតម្លៃនៃលទ្ធផលលើចន្លោះ[-1,1]នោះយើងជ្រើសយកអនុគមន៍tanhសម្រាប់ជាអនុគមន៍សកម្ម។ ក្នុងករណីដែលសំណុំទិន្នន័យសម្រាប់រៀនមានតម្លៃនៃលទ្ធផលលើចន្លោះ$(-\infty, \infty)$នោះយើងជ្រើសយកអនុគមន៍ដែលផ្ដល់តម្លៃដូចជាតុចូល Identity function សម្រាប់ជាអនុគមន៍សកម្ម។

ក្នុងករណីចំណោទតម្រែតម្រង់នេះដើម្បីប្រៀបធៀបកម្រិតជិតគ្នារវាងលទ្ធផលនៃបណ្ដាញនិងតម្លៃលទ្ធផលនៃគម្រូសម្រាប់រៀន យើងប្រើផលបូកការេនៃតម្លៃលម្អៀង(sum of squared residuals) រវាងតម្លៃ $\boldsymbol{y}(\boldsymbol{x}_n;\boldsymbol{w})$ និង$\boldsymbol{t}_n$ចំពោះគ្រប់គម្រូសម្រាប់រៀនក្នុងសំណុំទិន្នន័យសម្រាប់រៀនទាំងអស់។ ពោលគឺអនុគមន៍កម្រិតលម្អៀងសម្រាប់សំណុំទិន្នន័យសម្រាប់រៀនកំណត់ដោយ ។

$$E(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N}\|\boldsymbol{t} - \boldsymbol{y}(\boldsymbol{x}_n;\boldsymbol{w})\|^2$$

នៅទីនេះការដាក់មេគុណ $\frac{1}{2}$ គឺដើម្បីសម្រួលដល់ការគណនាដល់ការធ្វើដេរីវេនាពេលខាងមុខ។

គោលដៅរបស់យើងគឺកំនត់ប៉ារ៉ាម៉ែត្រនៃបណ្ដាញយ៉ាងណាដើម្បីឱ្យអនុគមន៍កម្រិតលម្អៀងខាងលើនេះមានតម្លៃអប្បបរមា។

4.3. ចំណោទធ្វើចំណាត់ថ្នាក់២ក្រុម

ក្នុងចំណោទធ្វើចំណាត់ថ្នាក់២ក្រុម ទិន្នន័យធាតុចូល $\boldsymbol{x}$ នឹងត្រូវបែងចែកទៅក្នុងក្រុមមួយក្នុងចំណោមពីរក្រុម។ ឧទាហរណ៍ករណីធាតុចូលជារូបថតមួយសន្លឹក។ នៅពេលនោះក្រោយពីទាញយកលក្ខណៈសម្គាល់របស់រូបថតនោះជាទម្រង់វ៉ិចទ័ររួច បណ្ដាញនឹងទទួលយកវ៉ិចទ័រនោះជាធាតុចូលរួចធ្វើការបែងចែកថាជារូបមុខមនុស្សឬមិនមែន។ ក្នុងករណីនេះ លទ្ធផលនៃទិន្នន័យសម្រាប់រៀន t យកតម្លៃជាអថេរដាច់{1(មុខមនុស្ស) , 0(មិនមែនមុខមនុស្ស)}។ បែបនេះចំណោទចំណាត់ថ្នាក់២ក្រុមក៏ជាចំណោទដែលទទួលធាតុចូល $\boldsymbol{x}$ និងប៉ាន់ស្មានលទ្ធផល t ដូចតម្រែតម្រង់ដែរគ្រាន់តែប្រភេទនៃតម្លៃលទ្ធផលជាអថេរដាច់។

ក្នុងករណីនេះ ដើម្បីប៉ាន់ស្មានតម្លៃលទ្ធផល យើងសិក្សាលើម៉ូឌែលប្រូបាប ពោលគឺសិក្សាលើប្រូបាបដែលថាលទ្ធផល $t = 1$នៅពេលធាតុចូល $\boldsymbol{x}$ ត្រូវបានទទួល $p(t = 1|\boldsymbol{x})$ ។ គំនិតនៅទីនេះគឺថា បើប្រូបាបនេះមានតម្លៃធំជាង0.5 នោះយើងសន្និដ្ឋានថាលទ្ធផលគឺ $t = 1$ និង សន្និដ្ឋានថាលទ្ធផលគឺ $t = 0$ ក្នុងករណីផ្ទុយពីនេះ។

ដោយពិនិត្យលើគំនិតបែបនេះ អ្នកអាចនឹកឃើញដល់លក្ខណៈនៃអនុគមន៍Sigmoid ដែលបានបង្ហាញខាងលើ។ បើយើងប្រើអនុគមន៍Sigmoid ជាអនុគមន៍សកម្មសម្រាប់បណ្ដាញ FNN $\boldsymbol{y}(\boldsymbol{x};\boldsymbol{w})$ នោះយើងអាចបង្ហាញម៉ូឌែលប្រូបាបខាងលើដោយFNNបាន។

$$p(t=1|\boldsymbol{x}) \approx \boldsymbol{y}(\boldsymbol{x};\boldsymbol{w})$$

ហេតុនេះដើម្បីកំណត់ប៉ារ៉ាម៉ែត្រនៃFNN យើងអាចសិក្សាពីសំណុំអថេរសម្រាប់រៀន (training data) $\{(\boldsymbol{x}_n, t_n)\}_{n=1}^{N}$បានដោយកំណត់យកប៉ារ៉ាម៉ែត្រដែលធ្វើឱ្យបំណែងចែកប្រូបាប $p(t|\boldsymbol{x};\boldsymbol{w})$ មានភាពប្រហាក់ប្រហែលគ្នាបំផុតជាមួយនឹងរបាយនៃសំណុំអថេរសម្រាប់រៀន។ ក្នុងការកំណត់ប៉ារ៉ាម៉ែត្រនៃម៉ូឌែលប្រូបាបបែបនេះ យើងហៅថា ការប៉ាន់ស្មានកម្រិតសាកសមបំផុតនៃទិន្នន័យ (Maximum Likelihood Estimation, MLE)។

ដោយប្រើតម្លៃនៃ $p(t=0|\boldsymbol{x};\boldsymbol{w}), p(t=1|\boldsymbol{x};\boldsymbol{w})$ យើងអាចបង្ហាញ $p(t|\boldsymbol{x};\boldsymbol{w})$ ជារួមតាមទម្រង់ខាងក្រោម។

$$p(t|\boldsymbol{x};\boldsymbol{w}) = p(t=1|\boldsymbol{x};\boldsymbol{w})^t p(t=0|\boldsymbol{x};\boldsymbol{w})^{1-t}$$

ដោយការសន្មតខាងលើ $p(t=1|\boldsymbol{x}) = y(\boldsymbol{x};\boldsymbol{w})$ នោះ $p(t=1|\boldsymbol{x}) = 1 - y(\boldsymbol{x};\boldsymbol{w})$ ។ ក្រោមការសន្មតនៃម៉ូឌែលបែបនេះ ការប៉ាន់ស្មានកម្រិតសាកសមបំផុតនៃទិន្នន័យ MLE គឺជាការកំណត់នូវកម្រិតសាកសមនៃទិន្នន័យ(likelihood)សម្រាប់រៀនរបស់ប៉ារ៉ាម៉ែត្រ $\boldsymbol{w}$ និងជ្រើសយកតម្លៃនៃ $\boldsymbol{w}$ ណាដែលធ្វើឱ្យកម្រិតសាកសមនោះមានតម្លៃអតិបរមា។ កម្រិតសាកសមនៃទិន្នន័យសម្រាប់រៀន (training data) របស់ប៉ារ៉ាម៉ែត្រ $\boldsymbol{w}$ ត្រូវបានកំណត់ដូចទម្រង់ខាងក្រោម។

$$L(\boldsymbol{w}) = \prod_{n=1}^{N} p(t_n|\boldsymbol{x};\boldsymbol{w}) = \prod_{n=1}^{N} \{y(\boldsymbol{x}_n;\boldsymbol{w})\}^{t_n}\{1 - y(\boldsymbol{x}_n;\boldsymbol{w})\}^{1-t_n}$$

ដើម្បីសម្រួលដល់ការធ្វើបរមាកម្ម យើងអនុវត្តអនុគមន៍លោការីតលើកន្សោមខាងលើ។ ការធ្វើបែបនេះមិនប៉ះពាល់ដល់អថេរភាព(ភាពកើនចុះ)នៃអនុគមន៍ឡើយ។ ក្នុងករណីនេះ អនុគមន៍កម្រិតលម្អៀងនៃចំណោទចំណាត់ថ្នាក់២ក្រុមកំណត់ដោយ $E(\boldsymbol{w})$ ដូចខាងក្រោម។

$$E(\boldsymbol{w}) = -\log L(\boldsymbol{w}) = -\sum_{n=1}^{N}\{t_n \log y(\boldsymbol{x}_n;\boldsymbol{w}) + (1-t_n)\log\big(1 - y(\boldsymbol{x}_n;\boldsymbol{w})\big)\}$$

ដូចដែលបានបង្ហាញខាងលើ អនុគមន៍Sigmoidត្រូវបានប្រើសម្រាប់ជាអនុគមន៍សកម្មក្នុងថ្នាក់លទ្ធផលនៃFNN។ ចំណុចនេះអាចបកស្រាយដូចខាងក្រោម។

ប្រូបាប $p(t=1|\boldsymbol{x})$ អាចសរសេរជាទម្រង់ប្រូបាបមានលក្ខខណ្ឌដូចខាងក្រោម។

$$p(t=1|\boldsymbol{x}) = \frac{p(\boldsymbol{x};t=1)}{p(\boldsymbol{x};t=0)+p(\boldsymbol{x};t=1)}$$

ដោយយក

$$u \equiv \log\frac{p(\boldsymbol{x};t=1)}{p(\boldsymbol{x};t=0)}$$

នោះយើងបាន

$$p(t=1|\boldsymbol{x}) = \frac{1}{1+\exp(-u)} = \sigma(u)$$

ពោលគឺ ម៉ូឌែលប្រូបាប $p(t=1|\boldsymbol{x})$ ដែលសិក្សាខាងលើសមមូលទៅនឹងអនុគមន៍Sigmoid។

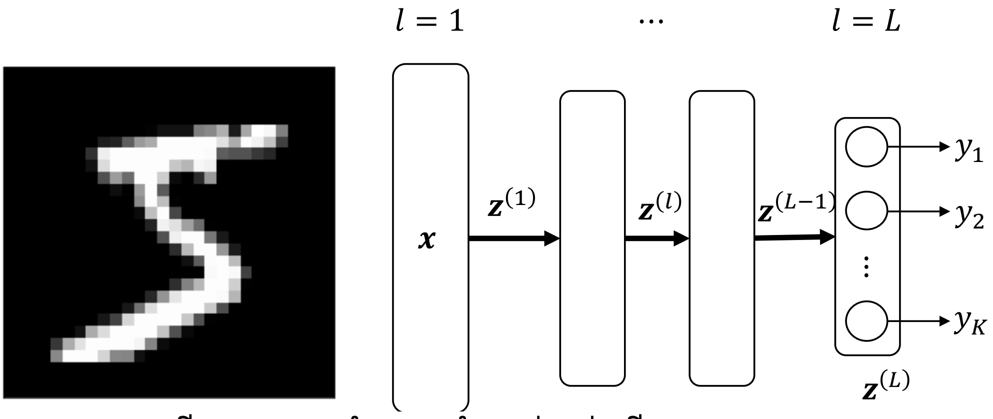
4.4. ចំណោទធ្វើចំណាត់ថ្នាក់ច្រើនក្រុម

ក្នុងចំណោទធ្វើចំណាត់ថ្នាក់ច្រើនក្រុម ទិន្នន័យធាតុចូល $\boldsymbol{x}$ នឹងត្រូវបែងចែកទៅក្នុងក្រុមមួយក្នុងចំណោមក្រុមមានកំណត់ច្រើន។ ឧទាហរណ៍ករណីធាតុចូលជារូបថតនៃលេខសរសេរដោយដៃមួយសន្លឹក។ នៅពេលនោះក្រោយពីទាញយកលក្ខណៈសម្គាល់របស់រូបថតនោះជាទម្រង់វ៉ិចទ័ររួចបណ្ដាញនឹងទទួលយកវ៉ិចទ័រនោះជាធាតុចូលរួចធ្វើការបែងចែកថាជាលេខណាមួយក្នុងចំណោម 0 ដល់ 9។ ក្នុងករណីនេះ លទ្ធផលនៃទិន្នន័យសម្រាប់រៀន t យកតម្លៃជាអថេរដាច់ {0,1, … , 9}។

ក្នុងករណីនេះ ការប្រើបណ្ដាញFNN សម្រាប់ចំណាត់ថ្នាក់ច្រើនក្រុម អាចធ្វើបានដោយកំណត់យកថ្នាក់លទ្ធផលមានចំនួនណឺរ៉ូនស្មើនឹងចំនួននៃក្រុមដែលត្រូវបែងចែក។ សន្មតថាចំនួនក្រុមដែលត្រូវបែងចែកក្នុងចំណោទមានK ។ នៅទីនេះ យើងប្រើបណ្ដាញFNNដែលមានLថ្នាក់និងចំនួនណឺរ៉ូននៅថ្នាក់លទ្ធផលមានចំនួនK។ លទ្ធផលដែលផ្ដល់ដោយណឺរ៉ូននិមួយៗក្នុងថ្នាក់លទ្ធផលអាចសរសេរដោយទម្រង់ខាងក្រោម។

$$y_k \equiv z_k^{(L)} = \frac{\exp\left(u_k^{(L)}\right)}{\sum_{i=1}^{K}\exp\left(u_i^{(L)}\right)}$$

ដូចដែលបានបកស្រាយក្នុងចំណុចអនុគមន៍សកម្មខាងលើយើងអាចប្រើអនុគមន៍Sofmax ដើម្បីបម្លែងលទ្ធផលជាប្រូបាបនៃករណីបែងចែកចូលក្នុងក្រុមនិមួយៗ។ យើងនឹងពិនិត្យលើភាពត្រឹមត្រូវនៃការសន្មតនេះដោយប្រើម៉ូឌែលប្រូបាបដូចករណី២ក្រុមដែរ។



រូបទី៦ FNN ក្នុងចំណោទចំណាត់ថ្នាក់ច្រើនក្រុម(K-ក្រុម)

សន្មតថាថ្នាក់និមួយៗនៃចំណោទខាងលើគឺ $C_1, C_2, \dots, C_K$ លទ្ធផលនៃណឺរ៉ូនkនៅថ្នាក់លទ្ធផលចុងក្រោយនៃបណ្តាញFNN កំណត់ដោយ $y_k\left(= z_k^{(L)}\right)$ ជាប្រូបាបនៃព្រឹត្តិការណ៍ដែលធាតុចូល $\boldsymbol{x}$ ត្រូវកំណត់ថានៅក្នុងក្រុម C_k ។

$$p(C_k|\boldsymbol{x}) = y_k = z_k^{(L)}$$

ជាលទ្ធផល ធាតុចូល $\boldsymbol{x}$ ត្រូវកំណត់ថានៅក្នុងក្រុម C_kបើតម្លៃប្រូបាប $p(C_k|\boldsymbol{x})$ មានតម្លៃធំជាងគេក្នុងចំណោមក្រុមទាំងអស់។

ក្នុងករណីចំណាត់ថ្នាក់ច្រើនក្រុមនេះ យើងកំណត់សរសេរលទ្ធផលពិត$\boldsymbol{t}_n$នៃអថេរ $\boldsymbol{x}_n$ដោយទម្រង់វិចទ័រ(one-hot vector) $\boldsymbol{t}_n = [t_{n1} \quad \cdots \quad t_{nk}]^\top$ ។

ក្នុងករណីចំណាត់ថ្នាក់រូបថតលេខសរសេរដោយដៃខាងលើនោះ$K = 10$។

បើ $\boldsymbol{x}_n$ ជាលេខ0 ដែលស្ថិតនៅក្នុងក្រុមC_1 នោះ $\boldsymbol{t}_n = [1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]^\top$ និង

បើ $\boldsymbol{x}_n$ ជាលេខ5 ដែលស្ថិតនៅក្នុងក្រុមC_6 នោះ $\boldsymbol{t}_n = [0 \ 0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0]^\top$។

ដោយសន្មតសរសេរបែបនេះ យើងអាចបង្ហាញ $p(\boldsymbol{t}|\boldsymbol{x})$ ជារួមតាមទម្រង់ខាងក្រោម។

$$p(\boldsymbol{t}|\boldsymbol{x}) = \prod_{k=1}^{K} p(C_k|\boldsymbol{x})^{t_k}$$

ហេតុនេះចំពោះសំណុំទិន្នន័យសម្រាប់រៀន $\{(\boldsymbol{x}_n, \boldsymbol{t}_n)\}_{n=1}^{N}$ កម្រិតសាកសមនៃទិន្នន័យសម្រាប់រៀន(training data) របស់ប៉ារ៉ាម៉ែត្រ $\boldsymbol{w}$ ត្រូវបានកំណត់ដូចទម្រង់ខាងក្រោម។

$$L(\boldsymbol{w}) = \prod_{n=1}^{N} p(\boldsymbol{t}_n|\boldsymbol{x}_n; \boldsymbol{w}) = \prod_{n=1}^{N}\prod_{k=1}^{K} p(C_k|\boldsymbol{x})^{t_{nk}} = \prod_{n=1}^{N}\prod_{k=1}^{K} \left(y_k(\boldsymbol{x}_n; \boldsymbol{w})\right)^{t_{nk}}$$

ដើម្បីសម្រួលដល់ការធ្វើបរមាកម្ម យើងអនុវត្តអនុគមន៍លោការីតលើកន្សោមខាងលើ។ ក្នុងករណីនេះ អនុគមន៍កម្រិតលម្អៀងនៃចំណោទចំណាត់ថ្នាក់ច្រើនក្រុមកំណត់ដោយ $E(\boldsymbol{w})$ ដូចខាងក្រោម។ អនុគមន៍កម្រិតលម្អៀងបែបនេះហៅថា cross entropy ។

$$E(\boldsymbol{w}) = -\log L(\boldsymbol{w}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk}\log y_k(\boldsymbol{x}_n;\boldsymbol{w})$$

ការប្រើអនុគមន៍Softmax សម្រាប់ជាម៉ូឌែលនៃបំណែងចែកច្រើនថ្នាក់នេះអាចបកស្រាយដូចខាងក្រោម។

ប្រូបាបដែលធាតុចូល $\boldsymbol{x}$ ត្រូវកំណត់ថានៅក្នុងក្រុម C_k អាចគណនាដោយ

$$p(C_k|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|C_k)}{\sum_{i=1}^{K} p(\boldsymbol{x}|C_k)}$$

នៅទីនេះដោយតាង $u_k = \log\big(p(\boldsymbol{x}, C_k)\big)$ នោះ $p(\boldsymbol{x}, C_k) = \exp(u_k)$ ហេតុនេះ

$$p(C_k|\boldsymbol{x}) = \frac{\exp(u_k)}{\sum_{i=1}^{K}\exp(u_i)}$$

កន្សោម $p(C_k|\boldsymbol{x})$ ដែលទាញបានខាងលើនេះដូចគ្នាទៅនឹងអនុគមន៍Softmaxដែរ។ ដូច្នេះភាពត្រឹមត្រូវនៃការប្រើប្រាស់អនុគមន៍Softmax ជាអនុគមន៍សកម្មសម្រាប់ចំណោទចំណាត់ថ្នាក់ច្រើនក្រុមដោយFNNត្រូវបានផ្ទៀងផ្ទាត់។

Learning Process in Neural Network

ក្នុងអត្ថបទមុន យើងបានដឹងរួចមកហើយថាដោយការផ្លាស់ប្តូរតម្លៃនៃប៉ារ៉ាម៉ែត្រនៃFNN (Feedforward Neural Network) អាចឱ្យយើងបង្ហាញពីទំនាក់ទំនងផ្សេងៗនៃទិន្នន័យដែលយើងមាន។ ចំណុចសំខាន់ក្នុងការកំណត់តម្លៃនៃប៉ារ៉ាម៉ែត្រគឺការស្វែងរកតម្លៃណាដែលសាកសមបំផុតក្នុងការបង្កើតបានជាបណ្តាញFNNដែលអាចពណ៌នាទំនាក់ទំនងក្នុងសំណុំទិន្នន័យសម្រាប់រៀន។ ការកំណត់តម្លៃនៃប៉ារ៉ាម៉ែត្រដោយផ្អែកលើសំណុំទិន្នន័យសម្រាប់រៀន ហៅថា ដំណើរការរៀន(Learning Process)។ មានវិធីសាស្ត្រច្រើនដែលត្រូវបានប្រើក្នុងដំណើរការរៀននៃFNN។ ក្នុងអត្ថបទនេះ យើងនឹងណែនាំវិធីសាស្ត្រកំណត់តម្លៃនៃប៉ារ៉ាម៉ែត្រដោយវិធីគណនាច្រំដែលលើតម្លៃលេខតាមប្រមាណវិធីងាយៗគឺ Stochastic Gradient Descent (SGD)។ ដើម្បីងាយស្រួលស្វែងយល់អំពីSGD ជាដំបូងយើងនឹងណែនាំអំពីគំនិត និងការគណនាក្នុងវិធីសាស្ត្រ Gradient Descent ជាមុន។

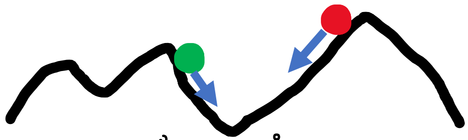
1. វិធីសាស្ត្រ Gradient Descent

ដូចដែលបានបង្ហាញក្នុងអត្ថបទមុន ប៉ារ៉ាម៉ែត្រដែលសាកសមបំផុតក្នុងការបង្កើតបានជាបណ្តាញFNNដែលអាចពណ៌នាទំនាក់ទំនងក្នុងសំណុំទិន្នន័យសម្រាប់រៀន គឺជាតម្លៃណាដែលធ្វើឱ្យអនុគមន៍កម្រិតលម្អៀងរវាងលទ្ធផលពីFNNនិងទិន្នន័យក្នុងសំណុំសម្រាប់រៀនតូចបំផុត។ ហេតុនេះគោលដៅរបស់យើងក្នុងដំណាក់កាលរៀននៃNeural Network គឺចង់កំណត់ប៉ារ៉ាម៉ែត្រនៃFNNដែលធ្វើឱ្យអនុគមន៍កម្រិតលម្អៀងមានតម្លៃតូចបំផុត ពោលគឺតម្លៃផលបូកការេនៃលម្អៀងក្នុងករណីនៃចំណោទតម្រែតម្រង់ ឬ Cross Entropyក្នុងករណីចំណោទចំណាត់ថ្នាក់ក្រុម មានតម្លៃតូចបំផុត។

$$\text{Regression}: \ E(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N}\|\boldsymbol{t} - \boldsymbol{y}(\boldsymbol{x}_n; \boldsymbol{w})\|^2$$

$$\text{Classification}: \ E(\boldsymbol{w}) = -\sum_{n=1}^{N}\left\{y(\boldsymbol{x}_n; \boldsymbol{w})^{t_n} + \left(1 - y(\boldsymbol{x}_n; \boldsymbol{w})\right)^{1-t_n}\right\}$$

គោលគំនិតក្នុងGradient Descent គឺផ្លាស់ប្តូរតម្លៃនៃប៉ារ៉ាម៉ែត្របន្តិចម្តងៗទៅតាមទិសដៅដែលធ្វើឱ្យតម្លៃនៃអនុគមន៍កម្រិតលម្អៀងមានការថយចុះ។ អ្នកអាចធ្វើការប្រដូចវិធីនេះទៅនឹងការចុះជំរាលឬចុះពីទីភ្នំដោយរំកិលខ្លួនអ្នកបន្តិចម្តងៗទៅកាន់ទីដែលទាបជាងកន្លែងដែលអ្នកនៅ។ ពេលដែលអ្នករំកិលខ្លួនដល់ទីដែលលែងមានបម្រែបម្រួលនៃរយៈកម្ពស់ អ្នកអាចសន្និដ្ឋានបានថាអ្នកដល់ទីដែលទាបបំផុតហើយ។ ដូចគ្នានេះដែរ នៅក្នុងវិធីសាស្ត្រGradient Descent តាមលក្ខណៈគណិតវិទ្យានៃ gradient (តម្លៃដេរីវេនៃអនុគមន៍ត្រង់ចំនុចណាមួយ) តម្លៃgradientត្រង់ចំណុចណាមួយគឺជាតម្លៃមេគុណប្រាប់ទិសនៃខ្សែកោងត្រង់ចំណុចនោះហើយក៏ជាតម្លៃធំបំផុតនៃបម្រែបម្រួលតម្លៃអនុគមន៍ពេលអ្នកធ្វើបម្រែបម្រួលលើអថេរមិនអាស្រ័យ។



រូបទី១ គំនិតក្នុង Gradient Descent

ពេលនេះ យើងពិនិត្យលើការគណនាក្នុងវិធីសាស្ត្រ Gradient Descent។ យើងសិក្សា លើករណីសំណុំទិន្នន័យសម្រាប់រៀន $\mathcal{D} = \{(\boldsymbol{x}_1, \boldsymbol{t}_1), \dots, (\boldsymbol{x}_N, \boldsymbol{t}_N)\}$ និងអនុគមន៍ កម្រិត លម្អៀង $E(\boldsymbol{w})$ ពោលគឺ ក្នុងករណីនៃចំណោទតម្រែតម្រង់

$$E(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N} \|\boldsymbol{t} - \boldsymbol{y}(\boldsymbol{x}_n; \boldsymbol{w})\|^2$$

និងក្នុងករណីនៃចំណោទចំណាត់ថ្នាក់ច្រើនក្រុម

$$E(\boldsymbol{w}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \log y_k(\boldsymbol{x}_n; \boldsymbol{w})$$

។ គោលដៅរបស់យើងគឺកំណត់តម្លៃនៃប៉ារ៉ាម៉ែត្រ $\boldsymbol{w}$ ដែលធ្វើអប្បបរមាកម្មលើ $E(\boldsymbol{w})$ ។

$$\widehat{\boldsymbol{w}} = \underset{\boldsymbol{w}}{\operatorname{argmin}}\, E(\boldsymbol{w})$$

សន្មតថាអនុគមន៍នេះយកតម្លៃអប្បបរមាត្រង់ចំណុច $\boldsymbol{w}^* \in \mathbb{R}^M$ ។វិធីសាស្ត្រ Gradient Descent អាចឱ្យយើងគណនាតម្លៃ(ប្រហែល)នៃ $\boldsymbol{w}^*$បានដោយចាប់ផ្តើមពីតម្លៃ $\boldsymbol{w}^{(0)}$ ណាមួយ រួចធ្វើការផ្លាស់ប្តូរតម្លៃនេះតាមការគណនាដូចខាងក្រោម។

$$\nabla E(\boldsymbol{w}) \equiv \frac{\partial E(\boldsymbol{w})}{\partial \boldsymbol{w}} = \begin{bmatrix} \frac{\partial E}{\partial w_1} & \cdots & \frac{\partial E}{\partial w_M} \end{bmatrix}^{\mathrm{T}}$$

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta_t \nabla E(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^{(t)}}$$

នៅទីនេះ$t = 0,1,\dots$ គឺជាលេខរៀងនៃការផ្លាស់ប្តូរតម្លៃអថេរ$\boldsymbol{x}$។ $\nabla E(\boldsymbol{w})$ គឺជាដេរីវេដោយផ្នែកនៃអនុគមន៍E ធៀបនឹងអថេរ $\boldsymbol{w}$ ឬហៅថា gradient ។ η_t គឺជាកម្រិតនៃការផ្លាស់ប្តូរតម្លៃអថេរដោយគ្រប់គ្រងលើឥទ្ធិពលនៃតម្លៃgradient។ η_t ត្រូវបានហៅថាជា អត្រារៀនឬ learning rate។ជាទូទៅតម្លៃនៃη_t ត្រូវបានកំណត់យកចន្លោះ០និង១ដោយតម្លៃយ៉ាងតូច។

យើងអាចកំណត់លក្ខខណ្ឌសម្រាប់បញ្ចប់ការផ្លាស់ប្តូរតម្លៃនៃអថេរបាន ដោយយកពេលដែលតម្លៃដាច់ខាតនៃ gradient យកតម្លៃសូន្យឬក្បែរសូន្យ។

ពិនិត្យលើករណីគម្រូងាយមួយ $f(x) = x^2 - 2x - 3$ ។ ករណីនេះយើងដឹងច្បាស់ថាតម្លៃអប្បបរមានៃអនុគមន៍គឺ -4 នៅពេលដែល $x^* = 1$។ យើងនឹងផ្ទៀងផ្ទាត់ជាមួយតម្លៃដែលគណនាតាមរយៈGradient Descent។

ដំបូងយើងគណនាអនុគមន៍ដេរីវេ $\frac{df(x)}{dx} = 2x - 2$ និង កំណត់យកអត្រា $\eta = 0.1$ ថេរ។ យើងចាប់ផ្តើមពីចំណុច $x^{(0)} = 0$,$f(x^{(0)}) = -3$ ។ ដោយផ្លាស់ប្តូរតម្លៃអថេរតាមរយៈGradient Descent ខាងលើយើងបានបម្រែបម្រួលនៃតម្លៃអថេរនិងតម្លៃអនុគមន៍ដូចតារាងខាងក្រោម។

តារាងទី១ បម្រែបម្រួលនៃតម្លៃអថេរនិងអនុគមន៍តាម Gradient Descent

t	$x^{(t)}$	$\frac{df(x)}{dx}$	$f(x)$
0	0.00	−2.00	−3.00
1	0.20	−1.60	−3.36
2	0.36	−1.28	−3.59
⋮	⋮	⋮	⋮
44	0.999946	−0.000109	−4.00
45	0.999956	−0.000087	−4.00

2. វិធីសាស្រ្ត Stochastic Gradient Descsent (SGD)

ការធ្វើបរមាកម្មលើតម្លៃអនុគមន៍ដោយប្រើ Gradient Descent តម្លៃអនុគមន៍កម្រិតលម្អៀងនៃគ្រប់ទិន្នន័យទាំងអស់ $E(\boldsymbol{w})$ ក្នុងសំណុំទិន្នន័យសម្រាប់រៀន(training data) ត្រូវបានធ្វើអប្បបរមាកម្ម។ ទាំងក្នុងករណីចំណោទតម្រែតម្រង់(Regression) និងចំណោទចំណាត់ថ្នាក់ក្រុមទិន្នន័យ(Classification) អនុគមន៍កម្រិតលម្អៀងនៃគ្រប់ទិន្នន័យអាចសរសេរបានជាផលបូកនៃគ្រប់តម្លៃកម្រិតលម្អៀងក្នុងករណីគម្រូសម្រាប់រៀននិមួយៗ$E_n(\boldsymbol{w})$។

$$E(\boldsymbol{w}) = \sum_{n=1}^{N} E_n(\boldsymbol{w})$$

ការផ្លាស់ប្តូរតម្លៃនៃប៉ារ៉ាម៉ែត្រដូចបានបង្ហាញក្នុងGradient Descent ដោយប្រើអនុគមន៍កម្រិតលម្អៀងនៃគ្រប់ទិន្នន័យទាំងអស់ $E(\boldsymbol{w})$ ហៅថា ការរៀនជាក្រុម/ជាបាច់(batch learning)។ ផ្ទុយពីនេះ វិធីសាស្រ្តនៃការធ្វើអប្បបរមាអនុគមន៍កម្រិតលម្អៀងដោយធ្វើការផ្លាស់ប្តូរតម្លៃនៃប៉ារ៉ាម៉ែត្រដោយប្រើគម្រូសម្រាប់រៀនម្តងមួយៗនិងតម្លៃអនុគមន៍កម្រិតលម្អៀងលើគម្រូនោះ $E_n(\boldsymbol{w})$ហៅថា stochastic gradient descent(SGD)។ ក្នុងវិធីSGD គម្រូទិន្នន័យសម្រាប់រៀន(training sample) ម្តងមួយៗ ត្រូវបានជ្រើសយកដោយចៃដន្យដើម្បីគណនា gradient នៃអនុគមន៍$E_n(\boldsymbol{w})$ រួចធ្វើការផ្លាស់ប្តូរតម្លៃប៉ារ៉ាម៉ែត្រតែម្តង ដោយមិនចាំបាច់ធ្វើការបូកសរុបគ្រប់ទិន្នន័យដែលមាននោះឡើយ។

$$\nabla E_n(\boldsymbol{w}) \equiv \frac{\partial E_n(\boldsymbol{w})}{\partial \boldsymbol{w}} = \begin{bmatrix} \frac{\partial E_n}{\partial w_1} & \cdots & \frac{\partial E_n}{\partial w_M} \end{bmatrix}^\top$$

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta_t \nabla E_n(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^{(t)}}$$

ដូចដែលអ្នកអាចធ្វើការកត់សម្គាល់បាន ដោយប្រើgradient descent ពេលខ្លះយើងអាចនឹងទទួលបានតម្លៃប៉ារ៉ាម៉ែត្រដែលធ្វើឱ្យតម្លៃអនុគមន៍កម្រិតលម្អៀងធ្លាក់ចុះទៅក្នុងទីតាំងដែលជាបរមាធៀបតែមិនមែនជាកន្លែងអប្បបរមាពិតប្រាកដប្រសិនបើទីតាំងនៃការចាប់ផ្តើមរបស់អ្នកមិនប្រសើរ។ ប៉ុន្តែជាមួយ SGD ដោយសាររាល់ការផ្លាស់ប្តូរទិន្នន័យ គម្រូទិន្នន័យនិមួយៗត្រូវបានជ្រើសរើសដោយចៃដន្យ ហេតុនេះភាពប្រថុយប្រថាននៃការធ្លាក់ចូលទៅក្នុងអប្បបរមាធៀបអាចត្រូវបានដោះស្រាយមួយកម្រិត។

ផ្ទុយពី ការរៀនជាក្រុម/ជាបាច់(batch learning) ការរៀនដោយប្រើវិធីសាស្រ្តSGD ត្រូវបានហៅថា ការរៀនអនឡាញ(online learning)។ ក្រៅពីនេះ ការរៀនដោយប្រើវិធីសាស្រ្តផ្លាស់ប្តូរប៉ារ៉ាម៉ែត្រដូចSGD ប៉ុន្តែជំនួសការរើសយកគម្រូទិន្នន័យម្តងមួយៗ ការរើសយកទិន្នន័យមួយចំនួន($< N$) សម្រាប់ប្រើគណនាgradient ក៏ត្រូវបានអនុវត្តដែរ។ ក្នុងករណីនេះគេហៅថា ការរៀនជាក្រុមតូច/ជាបាច់តូច(minibatch learning)។ ក្នុងការរៀនជាក្រុមតូច ការផ្លាស់ប្តូរប៉ារ៉ាម៉ែត្រត្រូវធ្វើឡើងដូចខាងក្រោម។

$$E_t(\boldsymbol{w}) = \frac{1}{|\mathcal{D}_t|} \sum_{n \in \mathcal{D}_t} E_n(\boldsymbol{w}) \ , \mathcal{D}_t \subset \mathcal{D}$$

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta_t \nabla E_t(\boldsymbol{w})|_{\boldsymbol{w}=\boldsymbol{w}^{(t)}}$$

3. សមត្ថភាពទូទៅ(Generalization Performance) និងស្ថានភាពរៀនហួសកំរិត(Overfitting)

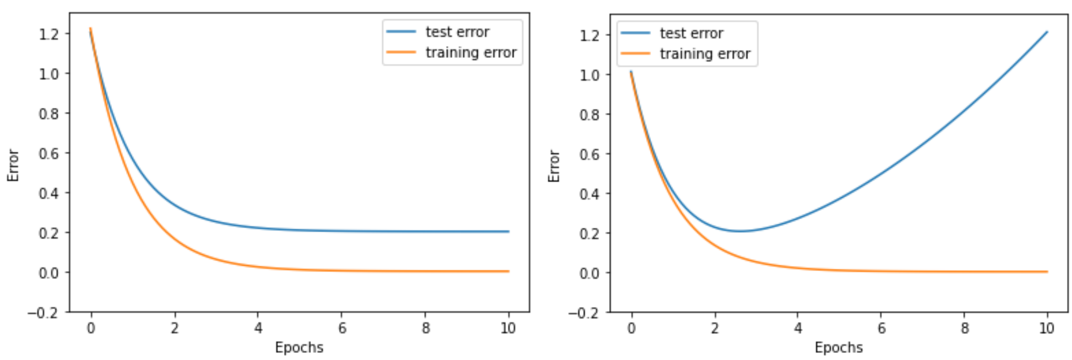
តាមការរៀបរាប់ពីដំណើរការរៀនក្នុងចំណុចខាងលើ យើងបានសិក្សាពីការកំណត់ប៉ារ៉ាម៉ែត្រណាដែលធ្វើឱ្យតម្លៃនៃអនុគមន៍កម្រិតលម្អៀងចំពោះសំណុំទិន្នន័យសម្រាប់រៀនមានតម្លៃតូចបំផុត។ ប៉ុន្តែតាមពិតទៅក្រៅពីសំណុំទិន្នន័យសម្រាប់រៀន យើងក៏ចង់បានបណ្តាញFNNដែលធ្វើឱ្យកម្រិតលម្អៀងក្នុងករណីទិន្នន័យផ្សេងក្រៅពីទិន្នន័យសម្រាប់រៀនមានតម្លៃតូចផងដែរ។

កម្រិតលម្អៀងចំពោះសំណុំទិន្នន័យសម្រាប់រៀនហៅថា កម្រិតលម្អៀងពេលរៀន(training error) និងតម្លៃសង្ឃឹមគណិតនៃកម្រិតលម្អៀងចំពោះសំណុំទិន្នន័យទាំងអស់(population)ទាំងទិន្នន័យដែលបានរៀននិងទាំងទិន្នន័យដែលមិនមានក្នុងសំណុំសម្រាប់រៀនហៅកម្រិតលម្អៀងទូទៅ(generalization error)។ គោលដៅយើងចង់បានបណ្តាញដែលផ្តល់ឱ្យនូវកម្រិតលម្អៀងទូទៅតូច

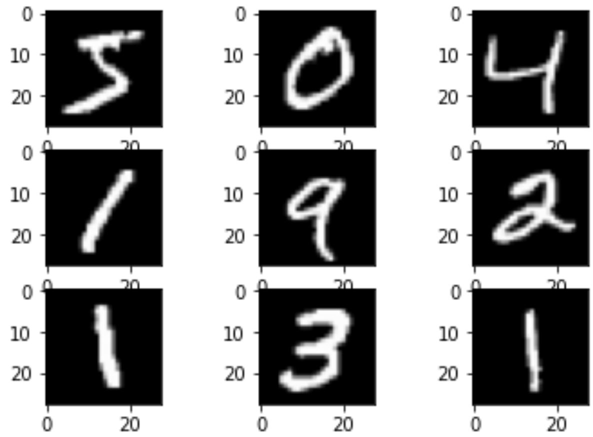
បំផុត ប៉ុន្តែដោយសារកម្រិតលម្អៀងទូទៅជាតម្លៃសង្ឃឹមគណិតលើសំណុំទិន្នន័យទាំងអស់ នោះការគណនាដោយផ្ទាល់មិនអាចធ្វើបានឡើយ។ ហេតុនេះ ជំនួសឱ្យការសិក្សាលើសំណុំទិន្នន័យទាំងអស់ គេសិក្សាលើសំណុំគម្រូទិន្នន័យមួយផ្នែកដែលមិនមានក្នុងសំណុំសម្រាប់រៀន។ សំណុំនេះហៅថា សំណុំទិន្នន័យសម្រាប់វាយតម្លៃ(test data set) ហើយកម្រិតលម្អៀងនៃសំណុំទិន្នន័យសម្រាប់វាយតម្លៃនេះហៅថា កម្រិតលម្អៀងពេលវាយតម្លៃ(test error)។ ពោលគឺជំនួសឱ្យគោលដៅនៃអប្បបរមាកម្មលើកម្រិតលម្អៀងទូទៅយើងសិក្សាលើអប្បបរមាកម្មលើកម្រិតលម្អៀងពេលវាយតម្លៃវិញ។

រូបទី២ខាងក្រោមបង្ហាញពីបម្រែបម្រួលនៃអនុគមន៍កម្រិតលម្អៀងនៅពេលប៉ារ៉ាម៉ែត្រត្រូវបានផ្លាស់ប្តូរក្នុងដំណើរការរៀន។ ក្រាបប្រភេទនេះហៅថាខ្សែកោងបង្ហាញដំណើរការរៀន(learning curve)។ ជាទូទៅអនុគមន៍កម្រិតលម្អៀងពេលរៀនថយនៅពេលដំណើរការរៀនត្រូវបានអនុវត្ត ។ ផ្ទុយទៅវិញ អនុគមន៍កម្រិតលម្អៀងពេលវាយតម្លៃថយចុះដូចអនុគមន៍កម្រិតលម្អៀងពេលរៀនដែរ នៅដំណាក់កាលដំបូង ប៉ុន្តែនៅត្រង់ចំណុចណាមួយ វាក៏ចាប់ផ្តើមមានគម្លាតរវាងករណីរៀននិងការវាយតម្លៃ។ ក្នុងករណីដែលមិនល្អ នៅពេលដែលដំណើរការរៀនបន្តទៅមុខ អនុគមន៍កម្រិតលម្អៀងពេលរៀនថយ ឯកម្រិតលម្អៀងវាយតម្លៃចាប់ផ្តើមកើនឡើង។ ករណីបែបនេះបង្ហាញក្នុងរូបទី២ ផ្នែកខាងស្តាំ។ ស្ថានភាពដែលកម្រិតលម្អៀងពេលរៀនថយចុះទាបខ្លាំងខណៈកម្រិតលម្អៀងវាយតម្លៃកើនធំខ្លាំងហៅថា ស្ថានភាពរៀនហួសកំរិត។ ស្ថានភាពបែបនេះអាចនិយាយបែបងាយបានថា បណ្តាញអាចផ្តល់លទ្ធផលបានល្អចំពោះតែទិន្នន័យណាដែលខ្លួនធ្លាប់បានរៀន រីឯទិន្នន័យដែលមិនមានក្នុងសំណុំសម្រាប់រៀន បណ្តាញមិនអាចផ្តល់លទ្ធផលបានល្អឡើយ។

ក្នុងការដោះស្រាយបញ្ហាបែបនេះមានតិចនិចជាច្រើនត្រូវបានស្រាវជ្រាវនិងអនុវត្ត។ យើងនឹងណែនាំលម្អិតក្នុងអត្ថបទខាងក្រោយ ប៉ុន្តែដំណោះស្រាយងាយគឺការបញ្ឈប់ដំណើរការរៀនមុនកំណត់(early stopping) ដោយបញ្ឈប់ដំណើរការរៀនមុនពេលកម្រិតលម្អៀងវាយតម្លៃកើនឡើង។



រូបទី២ អនុគមន៍កម្រិតលម្អៀងពេលរៀននិងពេលវាយតម្លៃក្នុងករណីធម្មតានិងករណីOverfitting



រូបទី៣ គម្រូទិន្នន័យក្នុងDataset(28 x 28)

4. សិក្សាឧទាហរណ៍៖ ការសម្គាល់លេខសរសេរដោយដៃដោយប្រើFNN

4.1. អំពីសំណុំទិន្នន័យនិង លក្ខណៈសម្គាល់នៃទិន្នន័យ

យើងប្រើសំណុំទិន្នន័យ MNIST[1] សម្រាប់ចំណោទធ្វើចំណាត់ថ្នាក់ក្រុមលេខសរសេរដោយដៃទៅតាមក្រុម(0ដល់9)ដោយប្រើFNN។ គម្រូទិន្នន័យមានបង្ហាញក្នុងរូបទី៣។ ទិន្នន័យជារូបនៃលេខសរសេរដោយដៃទាំងនេះមានទំហំ28x28។ នៅទីនេះយើងបំលែងវាជាទម្រង់វ៉ិចទ័រដែលមានទំហំ784(=28x28) និង បំលែងលេខសំគាល់ថ្នាក់របស់វាជាទម្រង់one-hot vector។

ចំពោះដំណើរការរៀននិងវាយតម្លៃ យើងបែងចែក60000គម្រូទិន្នន័យសម្រាប់ដំណើរការរៀន(training data)និង10000គម្រូទិន្នន័យសម្រាប់ការវាយតម្លៃ(test data)។

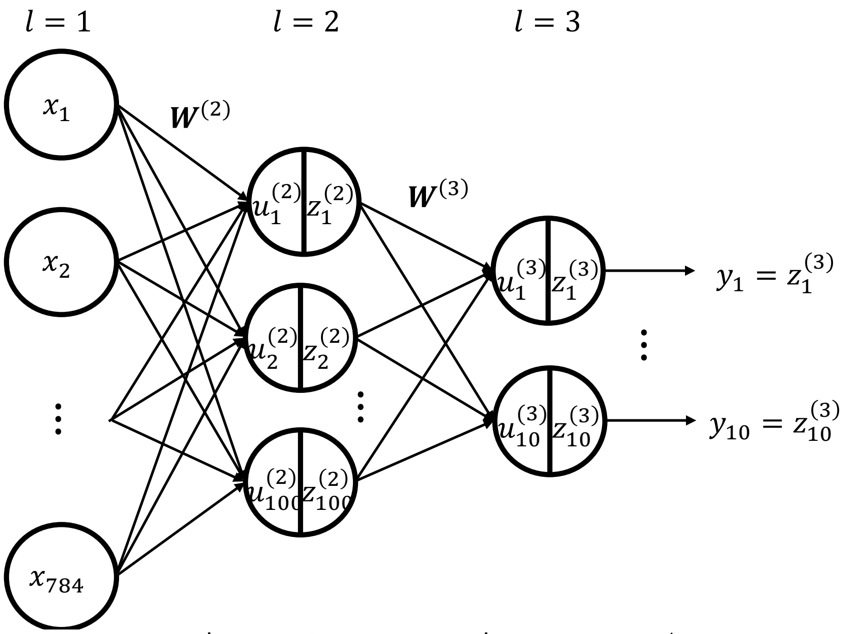
4.2. អំពីបណ្ដាញFNN

យើងកំណត់យកFNN៣ថ្នាក់សម្រាប់ប្រើនៅទីនេះ ពោលគឺ input layer- hidden layer-output layer។ ចំនួនណឺរ៉ូននៅinput layer គឺស្មើនឹងទំហំវ៉ិចទ័ររូបដែលជាធាតុចូល(784)ឯចំនួនណឺរ៉ូននៅoutput layerគឺស្មើនឹងចំនួនក្រុម(10)។ ចំពោះចំនួនណឺរ៉ូននៅhidden layerយើងកំណត់យកស្មើនឹង 100។

ចំពោះអនុគមន៍សកម្ម(activation function) យកកំណត់ប្រើ sigmoid function ចំពោះលទ្ធផលពីinput layerនិងsoftmax functionសម្រាប់លទ្ធផលចុងក្រោយ។

ចំពោះដំណើរការរៀនយើងនឹងប្រើ SGD (minibatch)។

[1] http://yann.lecun.com/exdb/mnist/ , https://keras.io/api/datasets/



រូបទី៤ ទម្រង់FNNដែលប្រើក្នុងឧទាហរណ៍

4.3. អំពីដំណើរការរៀនជាក្រុមតូច(minibatch learning)

Step-1 : ជ្រើសរើស minibatch

ជ្រើសយកគម្រូទិន្នន័យពីសំណុំទិន្នន័យសម្រាប់រៀនមួយផ្នែក(minibatch)ដោយចៃដន្យ។ គោលដៅយើងគឺធ្វើអប្បបរមាតម្លៃនៃអនុគមន៍កម្រិតលម្អៀងចំពោះទិន្នន័យទាំងនេះ។

Step-2: គណនា gradient

ដើម្បីអនុវត្តGradient Descent (minibatch) យើងគណនាតម្លៃgradientធៀបនឹងប៉ារ៉ាម៉ែត្រនិមួយៗ។

Step-3: ផ្លាស់ប្តូរតម្លៃប៉ារ៉ាម៉ែត្រ

ផ្លាស់ប្តូរតម្លៃនៃប៉ារ៉ាម៉ែត្រតាមទំនាក់ទំនងដែលបានបង្ហាញនៅចំណុច2. ខាងលើ។

Step-4: អនុវត្ត Step-1, Step-2, Step-3 ម្តងហើយម្តងទៀតរហូតដល់ចំនួនដងដែលបានកំណត់ជាមុន(Iteration number)

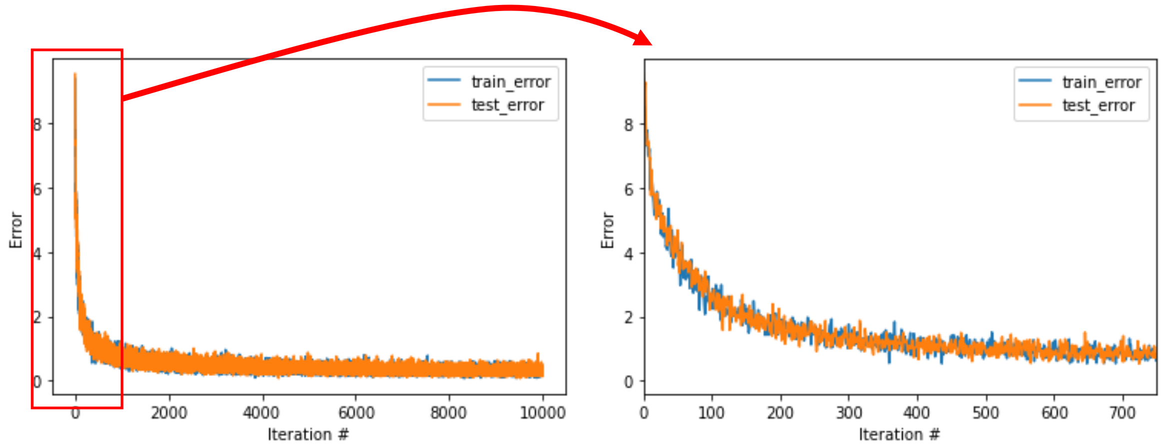
នៅទីនេះយើងកំណត់យក អត្រារៀន(learning rate) $\eta = 0.1$, ទំហំក្រុមតូច (minibatch) $|\mathcal{D}_t| = 100$ និងចំនួនដងក្នុងដំណើរការរៀន $\text{Iteration number} = 10000$។

4.4. ការបង្កើតបណ្ដាញFNNដោយប្រើPython

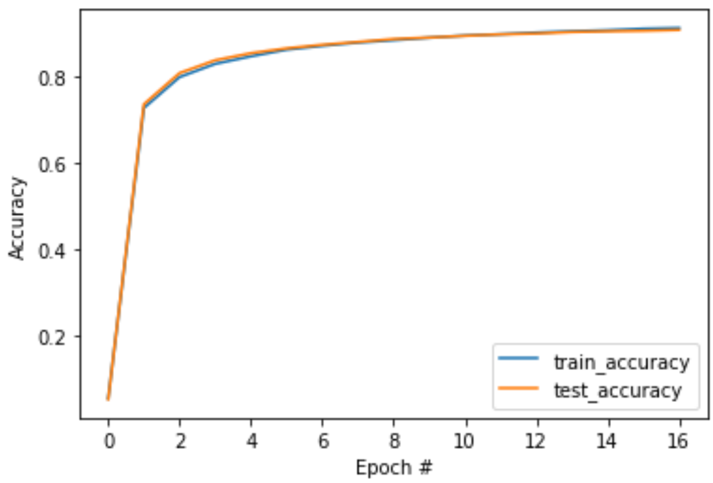
```
import numpy as np
```

```
class SimpleFNN:
  def __init__(self, input_size, hidden_size, output_size):
    self.params = {}
    self.params['W1'] = np.random.randn(input_size, hidden_size)
    self.params['b1'] = np.random.randn(hidden_size)
    self.params['W2'] = np.random.randn(hidden_size, output_size)
    self.params['b2'] = np.random.randn(output_size)

  def predict(self, x):
    W1, W2 = self.params['W1'], self.params['W2']
    b1, b2 = self.params['b1'], self.params['b2']
    u1 = np.dot(x,W1) + b1
    z1 = sigmoid(u1)
    u2 = np.dot(z1,W2) + b2
    y = softmax(u2)
    return y

  def loss(self, x, t):
    y = self.predict(x)
    return cross_entropy_error(y,t)

  def accuracy(self, x, t):
    y = self.predict(x)
    y = np.argmax(y, axis=1)
    t = np.argmax(t, axis=1)
    accuracy = np.sum(y == t)/float(x.shape[0])
    return accuracy

  def gradient(self, x, t):
        W1, W2 = self.params['W1'], self.params['W2']
        b1, b2 = self.params['b1'], self.params['b2']
        grads = {}
        batch_num = x.shape[0]
        a1 = np.dot(x, W1) + b1
        z1 = sigmoid(a1)
        a2 = np.dot(z1, W2) + b2
        y = softmax(a2)
        dy = (y - t) / batch_num
        grads['W2'] = np.dot(z1.T, dy)
        grads['b2'] = np.sum(dy, axis=0)
        dz1 = np.dot(dy, W2.T)
        da1 = sigmoid_grad(a1) * dz1
        grads['W1'] = np.dot(x.T, da1)
        grads['b1'] = np.sum(da1, axis=0)
        return grads
```

```
def sigmoid(x):
    return 1 / (1 + np.exp(-x))

def sigmoid_grad(x):
    return (1.0 - sigmoid(x)) * sigmoid(x)

def softmax(x):
    x = x - np.max(x, axis=-1, keepdims=True)   # To deal with overflow
    return np.exp(x) / np.sum(np.exp(x), axis=-1, keepdims=True)

def cross_entropy_error(y, t):
    batch_size = y.shape[0]
    t = t.argmax(axis=1)
    return -np.sum(np.log(y[np.arange(batch_size), t] + 1e-7)) / batch_size
```



រូបទី៥ កម្រិតលម្អៀង(Cross-entropy)ក្នុងដំណើរការរៀន



រូបទី៦ អត្រាត្រឹមត្រូវនៃការប៉ាន់ស្មានក្រុមចំពោះទិន្នន័យសម្រាប់រៀននិងទិន្នន័យវាយតម្លៃ

4.5. លទ្ធផល

រូបទី៥បង្ហាញពីបម្រែបម្រួលនៃអនុគមន៍កម្រិតលម្អៀងក្នុងដំណើរការរៀនចំពោះទិន្នន័យសម្រាប់រៀននិងទិន្នន័យវាយតម្លៃ។ រូបទី៦បង្ហាញពីអត្រាត្រឹមត្រូវនៃការធ្វើចំណាត់ថ្នាក់ក្រុមលើលេខសរសេរដៃដោយប្រើFNN ចំពោះទិន្នន័យសម្រាប់រៀននិងទិន្នន័យសម្រាប់វាយតម្លៃ។

ក្នុងអត្ថបទក្រោយយើងនឹងណែនាំអំពីវិធីសាស្រ្តកំណត់ប៉ារ៉ាម៉ែត្រក្នុងដំណើរការរៀនផ្សេងទៀតក្រៅពីSGD ដែលមានប្រសិទ្ធភាពក្នុងការគណនាចំពោះទម្រង់នៃNeural Network។

Backpropagation(BP)

ក្នុងអត្ថបទមុនយើងបានសិក្សាអំពីដំណើរការរៀនដើម្បីកំណត់តម្លៃប៉ារ៉ាម៉ែត្រនៃFNNតាមរយៈវិធីសាស្ត្រ stochastic gradient descend(SGD)។ ដូចដែលអ្នកអាចចាប់អារម្មណ៍បានក្នុង SGD ការគណនាgradientឬដេរីវេនៃអនុគមន៍កម្រិតលម្អៀងត្រូវបានធ្វើឡើង។ ជាទូទៅការគណនានេះអាចធ្វើបានតាមរយៈការគណនាតម្លៃប្រហែលនៃដេរីវេដោយប្រើតម្លៃអនុគមន៍ផ្ទាល់។ ប៉ុន្តែការគណនាបែបនេះចំណាយពេលច្រើន។ ដើម្បីគណនាgradientឬដេរីវេនៃអនុគមន៍ប្រកបដោយប្រសិទ្ធភាព វិធីសាស្ត្រគណនាgradientដោយប្រើ backpropagation ត្រូវបានប្រើប្រាស់ជាទូទៅ។

1. ភាពលំបាកក្នុងការគណនា gradient

នៅក្នុង stochastic gradient descend ការគណនាតម្លៃ gradient នៃអនុគមន៍កម្រិតលម្អៀង ($\nabla E(\boldsymbol{w}) = \partial E(\boldsymbol{w})/\partial \boldsymbol{w}$)គឺជាដំណាក់កាលដ៏សំខាន់។ ចំពោះFNNច្រើនថ្នាក់ ការគណនា gradient ចំពោះប៉ារ៉ាម៉ែត្រមានភាពស្មុគស្មាញខ្លាំង។

ជាឧទាហរណ៍តម្លៃកម្រិតលម្អៀងចំពោះគម្រូទិន្នន័យសម្រាប់រៀន $\boldsymbol{x}_n$ នៃចំណោទតម្រែតម្រង់(regression) កំណត់ដោយ $E_n = \frac{1}{2}\|\boldsymbol{y}(\boldsymbol{x}_n) - \boldsymbol{t}_n\|^2$។ យើងសាកល្បងគណនាដេរីវេធៀបនឹងប៉ារ៉ាម៉ែត្រទំងន់ផ្ទាល់ $w_{ji}^{(l)}$ នៃថ្នាក់ទី l ។

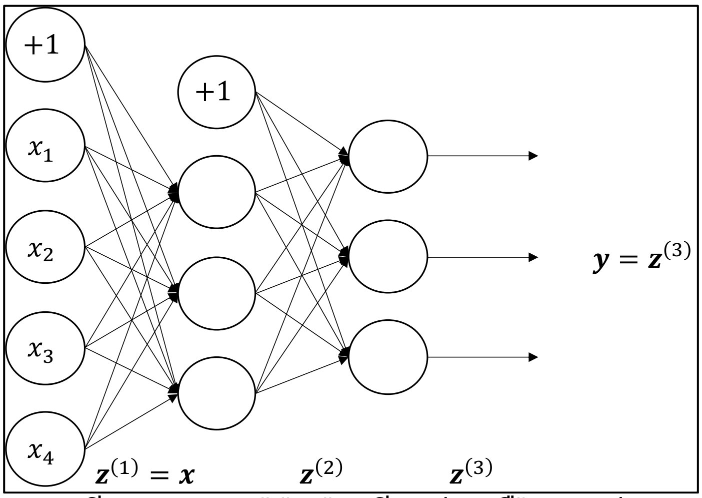
ដំបូងយើងពិនិត្យឃើញថា

$$\frac{\partial E_n}{\partial w_{ji}^{(l)}} = (\boldsymbol{y}(\boldsymbol{x}_n) - \boldsymbol{t}_n)^{\top} \frac{\partial \boldsymbol{y}}{\partial w_{ji}^{(l)}}$$

បន្ទាប់មកទៀតយើងត្រូវគណនាដេរីវេ $\frac{\partial \boldsymbol{y}}{\partial w_{ji}^{(l)}}$ ។ ដោយលទ្ធផលនៃFNN $\boldsymbol{y}(\boldsymbol{x})$ កំណត់ដោយទម្រង់ខាងក្រោម នោះយើងអាចមើលឃើញបានថាការគណនាដេរីវេតាមវិធីបែបនេះមិនមានប្រសិទ្ធភាពឡើយ ពោលគឺត្រូវចំណាយពេលច្រើនក្នុងការគណនាដោយប្រើProgramming។

$$\begin{aligned}
\boldsymbol{y}(\boldsymbol{x}) &= \boldsymbol{f}\left(\boldsymbol{u}^{(L)}\right) \\
&= \boldsymbol{f}\left(\boldsymbol{W}^{(L)}\boldsymbol{z}^{(L-1)} + \boldsymbol{b}^{(L)}\right) \\
&= \boldsymbol{f}\left(\boldsymbol{W}^{(L)}\boldsymbol{f}\left(\boldsymbol{W}^{(L-1)}\boldsymbol{z}^{(L-2)} + \boldsymbol{b}^{(L-1)}\right) + \boldsymbol{b}^{(L)}\right) \\
&= \boldsymbol{f}\left(\boldsymbol{W}^{(L)}\boldsymbol{f}\left(\boldsymbol{W}^{(L)}\boldsymbol{f}(\cdots \boldsymbol{f}(\boldsymbol{W}^{(l)}\boldsymbol{z}^{(l-1)} + \boldsymbol{b}^{(l)}\right) \cdots)) + \boldsymbol{b}^{(L)}\right)
\end{aligned}$$

វិធីសាស្ត្រ backpropagation អាចជួយដោះស្រាយបញ្ហាគណនាដេរីវេនៃអនុគមន៍បណ្តាក់ច្រើនជាន់បែបនេះបាន។ នៅក្នុងការបង្ហាញខាងក្រោម ដើម្បីសម្រួលដល់ការសរសេរ យើងកំណត់សរសេរតួ bias ជាផ្នែកមួយនៃប៉ារ៉ាម៉ែត្រទំងន់ផ្ទាល់នៃណឺរ៉ូនដែរ ពោលគឺ $w_{0j}^{(l)} = b_j^{(l)}$។ ហេតុនេះដោយកំណត់ណឺរ៉ូនទី0នៃថ្នាក់($l-1$) ឱ្យបញ្ចេញនូវលទ្ធផល $z_0^{(l-1)} = 1$ជានិច្ចនោះយើងអាចសរសេរលទ្ធផលនៃណឺរ៉ូនដោយទម្រង់ខាងក្រោម។



រូបទី១ FNNមានប៉ារ៉ាម៉ែត្រពីរថ្នាក់(ណឺរ៉ូន៣ថ្នាក់)

$$u_j^{(l)} = \sum_{i=1}^{k} w_{ji}^{(l)} z_i^{(l-1)} + b_j = \sum_{i=0}^{k} w_{ji}^{(l)} z_i^{(l-1)}$$

2. ការគណនាតាម backpropagation ករណីFNNមានប៉ារ៉ាម៉ែត្រពីរថ្នាក់(ណឺរ៉ូន៣ថ្នាក់)

រូបទី១បង្ហាញទម្រង់នៃFNNមានប៉ារ៉ាម៉ែត្រពីរថ្នាក់(ណឺរ៉ូន៣ថ្នាក់)នៃចំណោទតម្រែតម្រង់ (regression)។ អនុគមន៍សកម្ម(activation function) នៃថ្នាក់លទ្ធផលចុងក្រោយកំណត់ដោយ អនុគមន៍identity($f(x) = x$)។

សន្មតធាតុចូលនៃបណ្តាញនេះដោយ $\boldsymbol{x} = [x_1\ x_2\ x_3\ x_4]^{\top}$។ លទ្ធផលនៃថ្នាក់ណឺរ៉ូនទីមួយ ពោលថ្នាក់ធាតុចូលគឺ $z_i^{(1)} = x_i$ និងលទ្ធផលនៃថ្នាក់កណ្តាល$z_j^{(2)}$ ព្រមទាំងលទ្ធផលនៃថ្នាក់លទ្ធផលចុងក្រោយ$y_j(\boldsymbol{x}) = z_j^{(3)}$ កំណត់ដោយទម្រង់ខាងក្រោម។

$$z_j^{(2)} = f\left(u_j^{(2)}\right) = f\left(\sum_i w_{ji}^{(2)} z_i^{(1)}\right)$$

$$y_j(\boldsymbol{x}) = z_j^{(3)} = u_j^{(3)} = \sum_i w_{ji}^{(3)} z_i^{(2)}$$

សន្មតយកផលបូកការេនៃលម្អៀងជាអនុគមន៍កម្រិតលម្អៀងនៃបណ្តាញនេះ។

$$E_n = \frac{1}{2}\|\boldsymbol{y}(\boldsymbol{x}_n) - \boldsymbol{t}_n\|^2 = \frac{1}{2}\sum_j \left(y_j(\boldsymbol{x}_n) - \boldsymbol{t}_n\right)^2$$

ពេលនេះយើងពិនិត្យលើការគណនាដេរីវេនៃអនុគមន៍នេះធៀបនឹងប៉ារ៉ាម៉ែត្ររបស់វា។ ដំបូងយើងគណនាដេរីវេធៀបប៉ារ៉ាម៉ែត្រនៃផ្នែកថ្នាក់លទ្ធផលចុងក្រោយនៃបណ្តាញ $\frac{\partial E_n}{\partial w_{ji}^{(3)}}$។

$$\frac{\partial E_n}{\partial w_{ji}^{(3)}} = \frac{\partial}{\partial w_{ji}^{(3)}}\left\{\frac{1}{2}\|\boldsymbol{y}(\boldsymbol{x}_n) - \boldsymbol{t}_n\|^2\right\} = (\boldsymbol{y}(\boldsymbol{x}_n) - \boldsymbol{t}_n)^\top \frac{\partial \boldsymbol{y}}{\partial w_{ji}^{(3)}}$$

ដោយ

$$y_j(\boldsymbol{x}) = z_j^{(3)} = u_j^{(3)} = \sum_i w_{ji}^{(3)} z_i^{(2)}$$

នោះ

$$\frac{\partial \boldsymbol{y}}{\partial w_{ji}^{(3)}} = \left[0 \cdots 0 \; z_i^{(2)} \; 0 \cdots 0\right]^\top$$

ដូច្នេះ

$$\frac{\partial E_n}{\partial w_{ji}^{(3)}} = (\boldsymbol{y}(\boldsymbol{x}_n) - \boldsymbol{t}_n)^\top \frac{\partial \boldsymbol{y}}{\partial w_{ji}^{(3)}} = \left(y_j(\boldsymbol{x}_n) - t_{nj}\right) z_i^{(2)}$$

បន្ទាប់ពីនេះ យើងពិនិត្យលើថ្នាក់កណ្ដាល $\frac{\partial E_n}{\partial w_{ji}^{(2)}}$ ។

$$\frac{\partial E_n}{\partial w_{ji}^{(2)}} = \frac{\partial E_n}{\partial u_j^{(2)}} \frac{\partial u_j^{(2)}}{\partial w_{ji}^{(2)}}$$

ដោយ

$$u_j^{(2)} = \sum_i w_{ji}^{(2)} z_i^{(1)}$$

នោះយើងបាន

$$\frac{\partial u_j^{(2)}}{\partial w_{ji}^{(2)}} = z_i^{(1)}$$

ម្យ៉ាងទៀត បើយក k ជាចំនួនណឺរ៉ូននៅ ថ្នាក់លទ្ធផលចុងក្រោយនោះ E_n មាន$u_1^{(3)}, \cdots, u_k^{(3)}$ជាអថេរ ដែលយើងអាចសរសេរអនុគមន៍ដេរីវេដូចខាងក្រោម ។

$$\frac{\partial E_n}{\partial u_j^{(2)}} = \sum_k \frac{\partial E_n}{\partial u_k^{(3)}} \frac{\partial u_k^{(3)}}{\partial u_j^{(2)}}$$

$$\begin{aligned}
\frac{\partial E_n}{\partial u_k^{(3)}} &= \frac{\partial}{\partial u_k^{(3)}}\left\{\frac{1}{2}\sum_j \left(y_j(\boldsymbol{x}_n) - \boldsymbol{t}_n\right)^2\right\} \\
&= \frac{\partial}{\partial u_k^{(3)}}\left\{\frac{1}{2}\sum_j \left(u_j^{(3)} - \boldsymbol{t}_n\right)^2\right\} = u_k^{(3)} - t_{nk}
\end{aligned}$$

ដោយ

$$\frac{\partial u_k^{(3)}}{\partial u_j^{(2)}} = \frac{\partial}{\partial u_j^{(2)}}\left\{\sum_i w_{ki}^{(3)} z_i^{(2)}\right\}$$

$$= \frac{\partial}{\partial u_j^{(2)}} \left\{ \sum_i w_{ki}^{(3)} f\left(u_j^{(2)}\right) \right\}$$
$$= w_{kj}^{(3)} f'\left(u_j^{(2)}\right)$$

ហេតុនេះ យើងបាន

$$\frac{\partial E_n}{\partial w_{ji}^{(2)}} = \left(f'\left(u_j^{(2)}\right) \sum_k w_{kj}^{(3)} \left(u_k^{(3)} - t_{nk}\right) \right) z_i^{(1)}$$

3. ករណីទូទៅ៖ FNNច្រើនថ្នាក់

ដំបូង យើងពិនិត្យលើប៉ារ៉ាម៉ែត្រ $w_{ji}^{(l)}$ នៃថ្នាក់ទី l ។

$$\frac{\partial E_n}{\partial w_{ji}^{(l)}} = \frac{\partial E_n}{\partial u_j^{(l)}} \frac{\partial u_j^{(l)}}{\partial w_{ji}^{(l)}}$$

ដោយពិនិត្យរូបទី២(ឆ្វេង) បម្រែបម្រួលនៃ$u_j^{(l)}$ ជះឥទ្ធិពលលើតម្លៃនៃE_n តាមរយៈតម្លៃនៃ$z_j^{(l)}$ និង តម្លៃលទ្ធផលនៃណឺរ៉ូននៅថ្នាក់ទី$(l+1)$។ ហេតុនេះយើងបាន

$$\frac{\partial E_n}{\partial u_j^{(l)}} = \sum_k \frac{\partial E_n}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial u_j^{(l)}}$$

។ ដោយពិនិត្យលើកន្សោមខាងលើ យើងឃើញថា $\partial E_n/\partial u_j^{(\cdot)}$បង្ហាញនៅអង្គទាំងសង្ខាង។ នៅទីនេះ យើងសន្មតតាង

$$\delta_j^{(l)} \equiv \frac{\partial E_n}{\partial u_j^{(l)}}$$

។ ដោយប្រើទំនាក់ទំនង $u_k^{(l+1)} = \sum_j w_{kj}^{(l+1)} z_j^{(l)} = \sum_j w_{kj}^{(l+1)} f\left(u_j^{(l)}\right)$ យើងបាន $\partial u_k^{(l+1)}/\partial u_j^{(l)} = w_{kj}^{(l+1)} f'\left(u_j^{(l)}\right)$ ។ ហេតុនេះ

$$\delta_j^{(l)} = \frac{\partial E_n}{\partial u_j^{(l)}} = \sum_k \frac{\partial E_n}{\partial u_k^{(l+1)}} \frac{\partial u_k^{(l+1)}}{\partial u_j^{(l)}} = \sum_k \delta_k^{(l+1)} \left(w_{kj}^{(l+1)} f\left(u_j^{(l)}\right) \right)$$

តាមទំនាក់ទំនងនេះបង្ហាញថាយើងអាចគណនា $\delta_j^{(l)}$បានដោយប្រើតម្លៃនៃ$\delta_k^{(l+1)}$ $(k = 1,2,\dots)$។ មានន័យថា បើយើងដឹងតម្លៃ δ របស់ថ្នាក់ខាងលើពោលគឺ$(l+1)$ នោះយើងអាចគណនាតម្លៃ δ នៅថ្នាក់ក្រោមបន្តបន្ទាប់តាមទំនាក់ទំនងនេះ។ រូបទី២(ស្តាំ) បង្ហាញពីដំណើរការនៃការគណនា

δ ពីថ្នាក់លើឆ្ពោះទៅថ្នាក់ក្រោមបែបនេះ។ ទំនាក់ទំនងខាងលើនេះពិតជានិច្ចចំពោះគ្រប់ថ្នាក់ទាំងអស់នៃបណ្តាញ។

ដូច្នេះបើ δ នៅថ្នាក់លទ្ធផលចុងក្រោយត្រូវបានគណនា នោះ យើងអាចគណនា δ ពោលគឺដេរីវេនៃប៉ារ៉ាម៉ែត្រនៅថ្នាក់ក្រោមជាបន្តបន្ទាប់បានដោយអនុវត្តតាមទំនាក់ទំនងងាយខាងលើ។ ដោយសារតែលំដាប់នៃការគណនា δ នៅទីនេះមានទិសដៅផ្ទុយពីទិសដៅបញ្ជូនសញ្ញាណក្នុងការប៉ាន់ស្មានលទ្ធផលរបស់បណ្តាញ ដូចនេះទើបគេហៅឈ្មោះវិធីសាស្ត្រនេះថាជា backpropagation។

ត្រលប់មកផ្នែកនៅសល់នៃ$\frac{\partial E_n}{\partial w_{ji}^{(l)}} = \frac{\partial E_n}{\partial u_j^{(l)}}\frac{\partial u_j^{(l)}}{\partial w_{ji}^{(l)}}$ ។ ដោយ$\frac{\partial E_n}{\partial u_j^{(l)}}$អាចគណនាបានដោយគណនា$\delta$ ដូច្នេះ យើងពិនិត្យលើ $\frac{\partial u_j^{(l)}}{\partial w_{ji}^{(l)}}$។ តាមទំនាក់ទំនង $u_j^{(l)} = \sum_{i=0}^{k} w_{ji}^{(l)} z_i^{(l-1)}$ នោះ $\frac{\partial u_j^{(l)}}{\partial w_{ji}^{(l)}} = z_i^{(l-1)}$ ហេតុនេះ យើងបាន

$$\frac{\partial E_n}{\partial w_{ji}^{(l)}} = \delta_j^{(l)} z_i^{(l-1)}$$

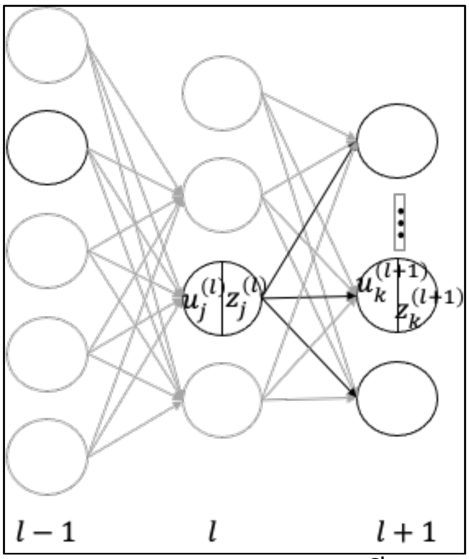
ដូចបង្ហាញក្នុងទំនាក់ទំនងដែលទាញបាននេះ ការគណនាដេរីវេដោយផ្នែកនៃអនុគមន៍កម្រិតលម្អៀងធៀបប៉ារ៉ាម៉ែត្រ$w_{ji}^{(l)}$ដែលភ្ជាប់ណឺរ៉ូនទី i នៃថ្នាក់ទី$(l-1)$និងណឺរ៉ូនទី j នៃថ្នាក់ទី(l) អាចគណនាបានយ៉ាងងាយដោយប្រើ $\delta_j^{(l)}$ នៃណឺរ៉ូនទី j នៃថ្នាក់ទី(l) និងលទ្ធផល$z_i^{(l-1)}$នៃណឺរ៉ូនទី i នៃថ្នាក់ទី$(l-1)$។ ដូចបានបញ្ជាក់ខាងលើ $\delta_j^{(l)}$ អាចគណនាបានដោយគណនាជាបន្តបន្ទាប់ពីថ្នាក់ខាងលើតាមទំនាក់ទំនងដែលបានទាញពីខាងដើម។ ក្នុងករណីនេះ $\delta_j^{(L)}$នៃថ្នាក់ខាងលើបំផុត(ថ្នាក់លទ្ធផលចុងក្រោយ)អាចកំណត់បានដោយ

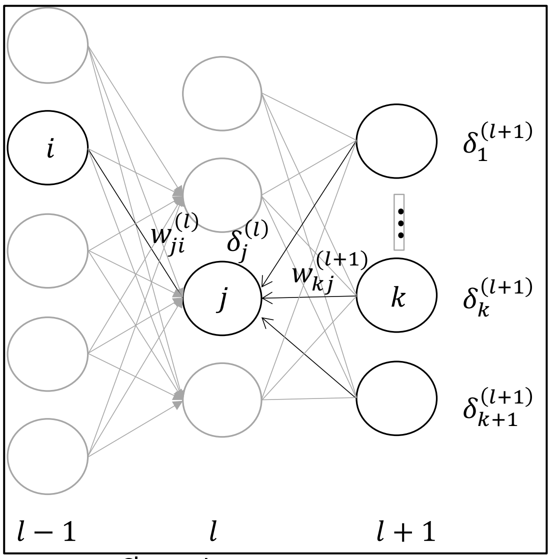
$$\delta_j^{(L)} = \frac{\partial E_n}{\partial u_j^{(L)}}$$

ដែលការគណនាជាក់ស្តែងប្រែប្រួលទៅតាមប្រភេទនៃអនុគមន៍កម្រិតលម្អៀង(ទៅតាមចំណោទ)។

ដោយបូកសរុបការគណនាខាងលើ នៅពេលដែលគម្រូទិន្នន័យសម្រាប់រៀន$(\boldsymbol{x}_n, \boldsymbol{t}_n)$ ត្រូវបានផ្តល់ gradientឬដេរីវេនៃអនុគមន៍កម្រិតលម្អៀងអាចគណនាបានតាមលំដាប់លំដោយខាងក្រោម។ ក្នុងករណីនៃការរៀនជាក្រុមតូច(minibatch) ផលបូកនៃgradient បានមកពីការគណនាចំពោះគម្រូទិន្នន័យនិមួយៗត្រូវបានយកជាgradientនៃក្រុមតូចនោះ

$$\frac{\partial E}{\partial w_{ji}^{(l)}} = \sum_n \frac{\partial E_n}{\partial w_{ji}^{(l)}}$$





រូបទី២ ករណីទូទៅនៃ FNN ច្រើនថ្នាក់

Input : គម្រូទិន្នន័យសម្រាប់រៀន $(\boldsymbol{x}_n, \boldsymbol{t}_n)$

Output : gradientឬដេរីវេនៃអនុគមន៍កម្រិតលម្អៀង $\frac{\partial E_n}{\partial w_{ji}^{(l)}}$ $(l = 2, \dots, L)$

---

(1) ដោយយក $\boldsymbol{z}^{(1)} = \boldsymbol{x}$ គណនាតម្លៃនៃ $\boldsymbol{u}^{(l)}, \boldsymbol{z}^{(l)}$ $(l = 2, \dots, L)$តាមលំដាប់

(2) គណនា $\delta_j^{(L)}$ (តាមធម្មតាវាត្រូវបានកំណត់ដោយ$\delta_j^{(L)} = z_j - t_j$)

(3) ចំពោះថ្នាក់កណ្តាល$(l = L-1, L-2, \cdots, 2)$ គណនាតម្លៃ$\delta_j^{(l)}$តាមលំដាប់ដោយ

$$\delta_j^{(l)} = \sum_k \delta_k^{(l+1)} \left(w_{kj}^{(l+1)} f\left(u_j^{(l)}\right) \right)$$

(4) ចំពោះថ្នាក់ទី l $(l = 2, \cdots, L)$ គណនាតម្លៃ$\frac{\partial E_n}{\partial w_{ji}^{(l)}}$ដោយ

$$\frac{\partial E_n}{\partial w_{ji}^{(l)}} = \delta_j^{(l)} z_i^{(l-1)}$$

---

4. ការគណនា$\delta_j^{(L)}$នៃថ្នាក់លទ្ធផលចុងក្រោយ

ដូចបានបង្ហាញខាងលើ ការគណនា$\delta_j^{(L)}$នៃថ្នាក់លទ្ធផលចុងក្រោយអាស្រ័យនឹងប្រភេទអនុគមន៍កម្រិតលម្អៀងដែលប្រើ ពោលគឺអាស្រ័យនឹងប្រភេទនៃចំណោទ។

4.1. ករណីចំណោទតម្រែតម្រង់ Regression

ក្នុងករណីចំណោទតម្រែតម្រង់អនុគមន៍កម្រិតលម្អៀងគឺជាផលបូកការេនៃលម្អៀងលើគម្រូទិន្នន័យនិមួយៗ។

$$E_n = \frac{1}{2}\|\boldsymbol{y}(\boldsymbol{x}_n) - \boldsymbol{t}_n\|^2 = \frac{1}{2}\sum_j (y_j - t_j)^2$$

ក្នុងករណីនេះ អនុគមន៍សកម្ម(activation)នៃថ្នាក់លទ្ធផលចុងក្រោយនៃFNNគឺជាអនុគមន៍ identity: $y_j = z_j^{(L)} = u_j^{(L)}$ ។ ហេតុនេះ $\delta_j^{(L)}$នៃថ្នាក់លទ្ធផលចុងក្រោយគឺ

$$\delta_j^{(L)} = u_j^{(L)} - t_{nj} = z_j^{(L)} - t_j = y_j - t_j$$

ពោលគឺគម្លាតរវាងតម្លៃលទ្ធផលនៃបណ្តាញ(ណឺរ៉ូន)និងលទ្ធផលក្នុងទិន្នន័យសម្រាប់រៀន។

4.2. ករណីចំណោទចំណាត់ថ្នាក់ក្រុម Classificaiton

ក្នុងចំណោទចំណាត់ថ្នាក់២ក្រុម អនុគមន៍កម្រិតលម្អៀងត្រូវបានកំណត់ដោយ

$$E_n = -t\log y - (1-t)\log(1-y)$$

ក្នុងករណីនេះ អនុគមន៍សកម្មនៃថ្នាក់លទ្ធផលចុងក្រោយនៃFNNគឺជាអនុគមន៍ sigmoid ហេតុនេះ

$$y = \frac{1}{1+\exp(-u)}\ ,\frac{dy}{du} = -\frac{\exp(-u)}{(1+\exp(-u))^2} = y(1-y)$$

ដូច្នេះយើងបាន

$$\begin{aligned}\delta_j^{(L)} &= -\frac{t}{y}\cdot\frac{dy}{du} - \frac{1-t}{1-y}\left(-\frac{dy}{du}\right)\\ &= -t(1-y) - (1-t)y = y - t\end{aligned}$$

ក្នុងចំណោទចំណាត់ថ្នាក់ច្រើនក្រុម អនុគមន៍កម្រិតលម្អៀងត្រូវបានកំណត់ដោយ

$$E_n = -\sum_k t_k \log y_k = -\sum_k t_k \log\left(\frac{\exp\left(u_k^{(L)}\right)}{\sum_i \exp\left(u_i^{(L)}\right)}\right)$$

ក្នុងករណីនេះ អនុគមន៍សកម្មនៃថ្នាក់លទ្ធផលចុងក្រោយនៃFNNគឺជាអនុគមន៍ softmax ហេតុនេះ

$$y = \frac{\exp\left(u_k^{(L)}\right)}{\sum_i \exp\left(u_i^{(L)}\right)}$$

ដូច្នេះយើងបាន

$$\begin{aligned}\delta_j^{(L)} &= -\sum_k t_k \frac{1}{y_k}\cdot\frac{\partial y_k}{\partial u_j^{(L)}}\\ &= -t_j(1-y_j) - \sum_{k\neq j} t_k(-y_j)\\ &= (y_j - t_j)\sum_k t_k\\ &= y_j - t_j \quad \left(\because \sum_k t_k = 1\ (\text{one} - \text{hot vector})\right)\end{aligned}$$

តាមលទ្ធផលខាងលើ ទាំងករណីចំណោទតម្រែតម្រង់ ទាំងករណីចំណោទចំណាត់ថ្នាក់ក្រុម $\delta_j^{(L)}$នៃថ្នាក់លទ្ធផលចុងក្រោយគឺ

$$\delta_j^{(L)} = y_j - t_j$$

ពោលគឺគម្លាតរវាងតម្លៃលទ្ធផលនៃបណ្តាញ(ណឺរ៉ូន)និងលទ្ធផលក្នុងទិន្នន័យសម្រាប់រៀន។

Technique in Learning Process

យើងបានសិក្សារួចមកហើយអំពីវិធីក្នុងការកំណត់តម្លៃប៉ារ៉ាម៉ែត្រនៃNeural Networkនៅក្នុងដំណើរការរៀន ដូចជាវិធីstochastic gradient descent និង វិធីក្នុងការគណនាដេរីវេដោយផ្នែកប្រកបដោយប្រសិទ្ធភាពដូចជា backpropagation។ ក្នុងអត្ថបទនេះយើងនឹងលើកយកល្បិចនិងវិធីសាស្ត្រមួយចំនួនដែលត្រូវបានប្រើដើម្បីបង្កើនប្រសិទ្ធភាពនៃការរៀនក្នុងNeural Network។ ប្រសិទ្ធភាពនៃការរៀននៅទីនេះសំដៅដល់ភាពល្អនៃអត្រាត្រឹមត្រូវក្នុងការប៉ាន់ស្មានលទ្ធផលរបស់បណ្តាញស្មុគស្មាញដូចជាFNNច្រើនថ្នាក់ដែលហៅថា Deep Learning ព្រមទាំងល្បឿននៃការរៀនរបស់វា។

1. ការកាត់បន្ថយស្ថានភាពរៀនហួសកម្រិត(Overfitting)

1.1. ការដាក់កម្រិតលើទម្ងន់ផ្ទាល់នៃណឺរ៉ូន(weight decay)

បញ្ហារៀនហួសកម្រិតគឺជាស្ថានភាពដែលបណ្តាញអាចផ្តល់លទ្ធផលល្អតែចំពោះទិន្នន័យសម្រាប់រៀនប៉ុន្តែ មិនអាចផ្តល់ចម្លើយល្អប្រសើរចំពោះទិន្នន័យផ្សេងពីទិន្នន័យសម្រាប់រៀនឡើយ។ ស្ថានភាពនេះអាចបកស្រាយបានថាជាករណីដែលការធ្វើបរមាកម្មលើអនុគមន៍កម្រិតលម្អៀងបានធ្លាក់ចូលទៅចំណុចនៃអប្បបរមាធៀបដែលមិនមែនជាចំណុចអប្បបរមានៃដែនកំណត់ទាំងមូល។ បញ្ហានេះកាន់តែងាយកើតឡើងនៅពេលដែលកម្រិតសេរីភាពនៃបណ្តាញពិសេសចំនួននៃប៉ារ៉ាម៉ែត្រមានកាន់តែច្រើន។ ដោយសារតែកម្រិតសេរីភាពនៃបណ្តាញមួយបង្ហាញពីកម្រិតនៃសមត្ថភាពរបស់បណ្តាញ ដូច្នេះការកាត់បន្ថយកម្រិតសេរីភាពដើម្បីដោះស្រាយស្ថានភាពរៀនហួសកម្រិតមិនមែនជារឿងល្អឡើយបើពិនិត្យលើសមត្ថភាពទាំងមូលរបស់បណ្តាញ។ ហេតុនេះដើម្បីកាត់បន្ថយការកើតឡើងនៃស្ថានភាពរៀនហួសកំរិតនិងរក្សានូវសមត្ថភាពខ្ពស់របស់បណ្តាញផង ការគ្រប់គ្រងទៅលើកម្រិតសេរីនៃទម្ងន់ផ្ទាល់របស់ណឺរ៉ូនត្រូវបានសិក្សាព្រមគ្នានៅពេលដំណើរការការរៀន។វិធីនេះហៅថា Regularization។ វិធីសាស្ត្រងាយមួយនៃRegularizationប្រើចំពោះNeural Networkគឺការដាក់កម្រិតលើទម្ងន់ផ្ទាល់នៃណឺរ៉ូន(weight decay)។

ការដាក់កម្រិតលើទម្ងន់ផ្ទាល់នៃណឺរ៉ូនអាចធ្វើបានដូចខាងក្រោម។

$$E_t(\boldsymbol{w}) = \frac{1}{|\mathcal{D}_t|}\sum_{n\in\mathcal{D}_t} E_n(\boldsymbol{w}) + \frac{\lambda}{2}\|\boldsymbol{w}\|^2$$

នៅទីនេះ λ ជាប៉ារ៉ាម៉ែត្រដើម្បីគ្រប់គ្រងកម្រិតនៃការដាក់កម្រិតលក្ខខណ្ឌលើទំហំនៃប៉ារ៉ាម៉ែត្រទម្ងន់ផ្ទាល់នៃណឺរ៉ូន $\boldsymbol{w}$។ ជាទូទៅតម្លៃនេះត្រូវបានកំណត់ដោយចំនួនពិតតូចខ្លាំង $\lambda = 0.1 \sim 0.00001$។ ក្នុងករណីនេះការផ្លាស់ប្តូរតម្លៃប៉ារ៉ាម៉ែត្រតាម SGD ត្រូវបានធ្វើឡើងដូចខាងក្រោម។

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta_t\left(\frac{1}{|\mathcal{D}_t|}\sum_{n\in\mathcal{D}_t}\boldsymbol{\nabla}E_n + \lambda\boldsymbol{w}^{(t)}\right)$$

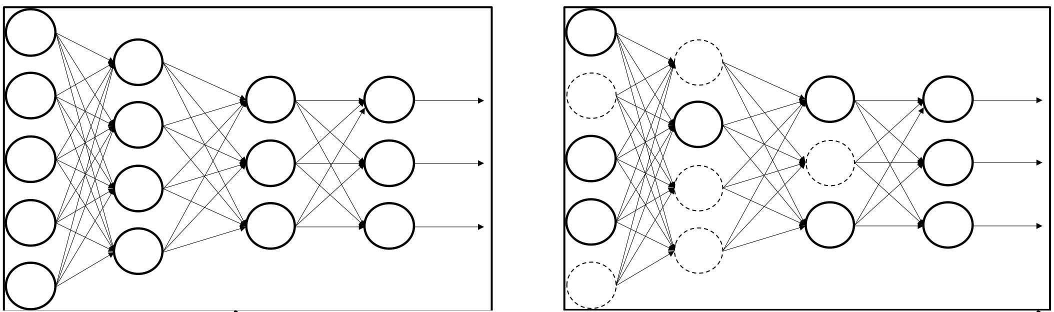
តាមកន្សោមខាងលើនេះ យើងអាចមើលឃើញថាតម្លៃនៃ ប៉ារ៉ាម៉ែត្រនឹងត្រូវបានបន្ថយជានិច្ចរាល់ពេលនៃការផ្លាស់ប្ដូរតម្លៃប៉ារ៉ាម៉ែត្រនៃដំណើរការរៀនត្រូវបានធ្វើឡើង។ ហេតុនេះហើយទើបវិធីនេះហៅថា weight decay ។ ក្រៅពីវិធីខាងលើនេះក៏មានវិធីផ្សេងដែរដូចជាការកំណត់លក្ខខណ្ឌអតិបរមានៃទំហំប៉ារ៉ាម៉ែត្រទៅក្នុងចំណោទអប្បបរមានៃអនុគមន៍កម្រិតលម្អៀងដោយទម្រង់ខាងក្រោម។

$$\|\boldsymbol{w}\|^2 = \sum_i w_{ji}^2 < c$$

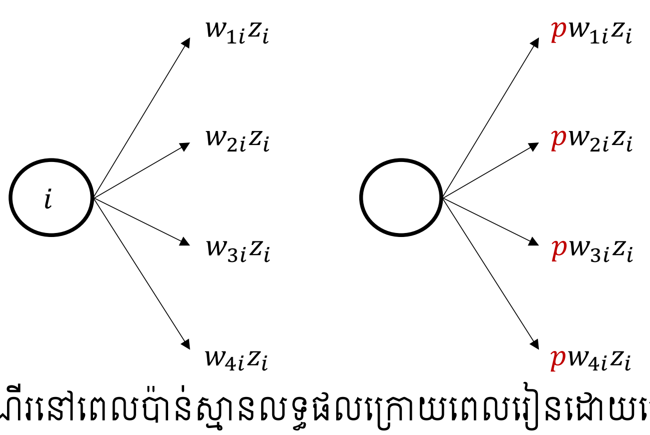
1.2. Dropout

Dropout គឺជាវិធីសាស្ត្ររៀនដែលណឺរ៉ូនត្រូវប្រើត្រូវបានជ្រើសដោយចៃដន្យជាមួយតម្លៃប្រូបាបដែលបានកំណត់។ រូបទី១បង្ហាញពីទម្រង់នៃFNNនៅពេលដែលណឺរ៉ូនមួយចំនួនត្រូវបានជ្រើសដើម្បីរៀន។ នៅពេលរៀន(learning process) ណឺរ៉ូននៃថ្នាក់ធាតុចូលនិងថ្នាក់កណ្ដាលត្រូវបានជ្រើសរើសដោយប្រូបាប p ដោយចៃដន្យ ឯណឺរ៉ូនក្រៅពីនេះត្រូវបានចាត់ទុកថាគ្មាន រួចដំណើរការរៀនដូចធម្មតា។ ប្រូបាប p អាចកំនត់ដោយតម្លៃផ្សេងគ្នានៅតាមថ្នាក់និមួយៗបាន។

ក្រោយពេលដំណើរការរៀនបានបញ្ចប់ ការប៉ាន់ស្មានលទ្ធផលត្រូវបានធ្វើជាមួយនឹងណឺរ៉ូនទាំងអស់ដោយគ្មានការជ្រើសរើសតែមួយផ្នែកដូចកាលដំណាក់កាលរៀនឡើយ។ ប៉ុន្តែនៅករណីនេះ តម្លៃនៃលទ្ធផលនៅថ្នាក់និមួយៗត្រូវបានគុណនឹងតម្លៃប្រូបាបដែលបានកំណត់កាលពីពេលរៀន។



រូបទី១ ដំណើរការនៃរៀនដោយប្រើDropout។(ឆ្វេង)បណ្ដាញដើម(ស្ដាំ)បណ្ដាញពេលរៀន



រូបទី២ ដំណើរនៅពេលប៉ាន់ស្មានលទ្ធផលក្រោយពេលរៀនដោយប្រើDropout

បំណាងនៃការអនុវត្តDropoutគឺកាត់បន្ថយកម្រិតសេរីភាពនៃបណ្តាញនៅពេលដំណើរការរៀនឱ្យតូចដើម្បីកាត់បន្ថយឱកាសនៃការកើតឡើងនូវស្ថានភាពរៀនហួសកម្រិត(overfitting)។ ការធ្វើបែបនេះសមមូលនឹងការរៀនដោយប្រើបណ្តាញច្រើនដែលឯករាជពីគ្នានិងដែលត្រូវបានកាត់បន្ថយចំនួនណឺរ៉ូន រួចធ្វើមធ្យមភាគលើលទ្ធផលនៅដំណាក់កាលប៉ាន់ស្មានលទ្ធផល។

2. វិធីសាស្ត្រជំនួយក្នុងការរៀន

2.1. ការផ្តល់ទម្រង់ស្តង់ដាលើទិន្នន័យ(standardization of data)

ក្នុងករណីដែលទិន្នន័យសម្រាប់រៀន(training data)មានស្ថានភាពលំអៀងទៅកាន់ក្រុមតម្លៃណាមួយ ការរៀននិងប៉ាន់ស្មានលទ្ធផលនឹងបានល្អប្រសើរឡើយ។ ដើម្បីដោះស្រាយបញ្ហាបែបនេះការធ្វើដំណើរការដំណាក់កាលដំបូង(preprocessing)លើសំណុំទិន្នន័យមុនពេលអនុវត្តដំណើរការរៀនត្រូវបានធ្វើឡើង។ ដំណើរការដំណាក់កាលដំបូងសាមញ្ញមួយគឺការធ្វើឱ្យទិន្នន័យទៅជាទម្រង់ស្តង់ដាដែលហៅថា standardization។ ដំណើរការនេះត្រូវធ្វើឡើងដោយការបម្លែងទិន្នន័យនិមួយៗតាមទម្រង់ខាងក្រោម។ នៅទីនេះ $\boldsymbol{x}_{ni}$ គឺជាកំប៉ូសង់ទីiនៃទិន្នន័យ$\boldsymbol{x}_n$ និង $\bar{x}_i = \frac{1}{N}\sum_{n=1}^{N} x_{ni}$ ។

$$\sigma_i = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(x_{ni}-\bar{x}_i)^2}$$

$$x_{ni} \leftarrow \frac{x_{ni}-\bar{x}_i}{\sigma_i}$$

ការអនុវត្តបែបនេះជាការបន្លែងទូទៅនៅក្នុងវិធីសាស្ត្រស្ថិតិដែលក្រោយបម្លែងរួចទិន្នន័យនឹងមានតម្លៃមធ្យមនៃកំប៉ូសង់និមួយៗស្មើ0 និងវ៉ារ្យង់ស្មើ1។

2.2. របៀបកំណត់កម្រិតរៀន(learning rate)

នៅក្នុងវិធីសាស្ត្ររៀនដូចជា gradient descend ឬ stochastic gradient descend (SGD) ការផ្លាស់ប្តូរនិងកំណត់តម្លៃនៃប៉ារ៉ាម៉ែត្រប្រែប្រួលទៅតាមតម្លៃកម្រិតរៀន(learning rate) ដែលបានកំណត់។ នៅក្នុងដំណើរការរៀនមានវិធីសាស្ត្រចំបង២ដែលត្រូវបានប្រើដើម្បីកំណត់តម្លៃកម្រិតរៀន។

វិធីសាស្ត្រទី១ គឺការកំណត់តម្លៃកម្រិតរៀនឱ្យធំនៅពេលចាប់ផ្តើមរៀន រួចបន្ថយតម្លៃនេះបន្តិចម្តងៗនៅពេលដំណើរការរៀនបន្ត។ ជាឧទាហរណ៍ការកំណត់កម្រិតរៀននៅដំណាក់កាលរៀនទី t ដោយ $\eta_t = \eta_0 - \alpha t$ ដែលη_0ជាតម្លៃកម្រិតរៀននៅពេលចាប់ផ្តើមនិងαកម្រិតនៃការបន្ថយ។

វិធីសាស្ត្រទី២ គឺការកំណត់តម្លៃកម្រិតរៀនខុសៗគ្នានៅតាមថ្នាក់ផ្សេងៗគ្នានៃបណ្តាញ ។ របៀបនៃការកំណត់អាចប្រែប្រួលទៅតាមទម្រង់នៃបណ្តាញ ។ ជាឧទាហរណ៍ការកំណត់កម្រិតរៀនឱ្យធំនៅថ្នាក់ដែលក្បែរនឹងថ្នាក់ធាតុចូល(input layer) និងកំណត់តម្លៃកាន់តែតូចនៅថ្នាក់ដែលនៅក្បែរថ្នាក់លទ្ធផលចុងក្រោយ(output layer) ។

វិធីសាស្ត្រនៃការកំណត់តម្លៃកម្រិតរៀនដោយស្វ័យប្រវត្តិដូចជា AdaGrad ក៏ត្រូវបានប្រើជាញឹកញាប់ផងដែរ ។ ឧបមាថា gradient នៃអនុគមន៍កម្រិតលម្អៀងកំណត់ដោយ $\boldsymbol{g}_t = \boldsymbol{\nabla} E_n$ និងកំប៉ូសង់ទីi កំណត់ដោយ $g_{t,i}$ ។ បើតាម SGD ប៉ារ៉ាម៉ែត្រនឹងត្រូវផ្លាស់ប្តូរតម្លៃដោយ $-\eta g_{t,i}$ ប៉ុន្តែ AdaGrad វាផ្លាស់ប្តូរដោយតម្លៃខាងក្រោម ។ ការផ្លាស់ប្តូរបែបនេះអាចបកស្រាយដោយងាយថាជាការផ្តោតការយកចិត្តទុកដាក់លើតម្លៃកំប៉ូសង់ណាដែលការផ្លាស់ប្តូរមិនសូវបានអនុវត្តជាញឹកញាប់ ។

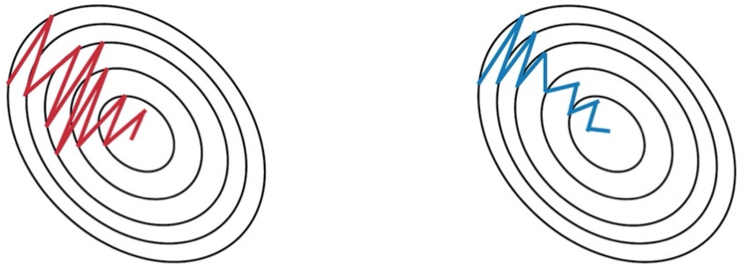
$$-\frac{\eta}{\sqrt{\sum_{k=1}^{t} g_{k,i}^2}} g_{t,i}$$

2.3. វិធីសាស្ត្រ Momentum

វិធីសាស្ត្រនៃការកំណត់តម្លៃកម្រិតរៀនដោយស្វ័យប្រវត្តិផ្សេងទៀតដែលមានប្រសិទ្ធភាពគឺវិធីសាស្ត្រ Momentum ។ វិធីនេះត្រូវបានដឹងថាជួយឱ្យដំណើរការផ្លាស់ប្តូរតម្លៃប៉ារ៉ាម៉ែត្រក្នុង gradient descend ឆាប់ចប់ ។ ក្នុងវិធីសាស្ត្រនេះប៉ារ៉ាម៉ែត្រត្រូវបានផ្លាស់ប្តូរតាមទម្រង់ខាងក្រោម ។

$$\Delta \boldsymbol{w}^{(t-1)} = \Delta \boldsymbol{w}^{(t)} - \Delta \boldsymbol{w}^{(t-1)}$$
$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \eta \boldsymbol{\nabla} E_t + \mu \Delta \boldsymbol{w}^{(t-1)}$$

នៅទីនេះ μ ច្រើនត្រូវបានកំណត់ដោយ $\mu = 0.5 \sim 0.9$ ។ វិធីនេះត្រូវបានបកស្រាយថាជាវិធីដែលជួយគ្រប់គ្រងដល់ការប្រែប្រួលនៃការផ្លាស់ប្តូរតម្លៃប៉ារ៉ាម៉ែត្រកុំឱ្យមានការប្រែប្រួលខ្លាំងដែលប៉ះពាល់ដល់ការរួចតូចនៃតម្លៃកម្រិតលម្អៀងនៅដំណាក់កាលរៀន ។



រូបទី៣ ភាពខុសគ្នានៃបម្រែបម្រួលប៉ារ៉ាម៉ែត្រដោយប្រើSGDដែលមានឬគ្មាន momentum

(Ref: Stochastic Gradient Descent on your microcontroller by SIMONE
https://eloquentarduino.github.io/2020/04/stochastic-gradient-descent-on-your-microcontroller/)

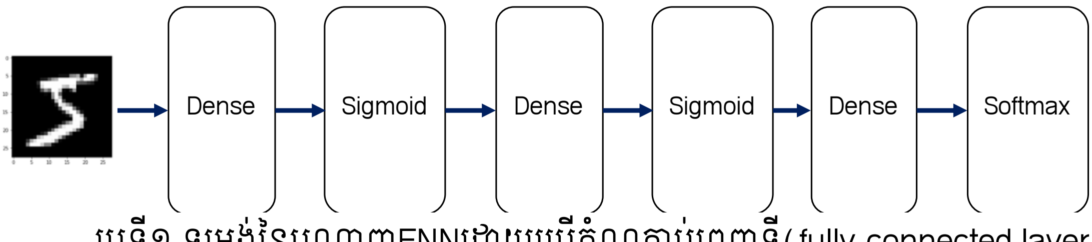
Convolutional Neural Network (CNN)

ក្នុងអត្ថបទនេះ យើងនឹងសិក្សាអំពីប្រភេទមួយផ្សេងទៀតនៃNeural Network ដែលហៅថា Convolutional Neural Network(CNN) ។ CNNត្រូវបានប្រើប្រាស់យ៉ាងទូលំទូលាយក្នុងការអនុវត្តលើបច្ចេកវិទ្យាកំណត់សម្គាល់រូបភាពឬកំណត់សម្គាល់សម្លេង និងប្រកបដោយប្រសិទ្ធភាពខ្ពស់បើធៀបនឹងបណ្តាញFNNធម្មតា ។ ក្នុងបច្ចេកវិទ្យាកំណត់សម្គាល់រូបភាពឬ បច្ចេកវិទ្យានានាទាក់ទងនឹងការប្រើប្រាស់រូបភាពក្នុងបញ្ញាសិប្បនិម្មិតត្រូវបានមើលឃើញថាមានការប្រើប្រាស់CNNជាមូលដ្ឋាន ។ ថ្មីៗនេះទៀត សូម្បីក្នុងបច្ចេកវិទ្យាភាសា(Natural Language Processing) បណ្តាញCNNក៏ត្រូវបានប្រើប្រាស់ផងដែរ ។

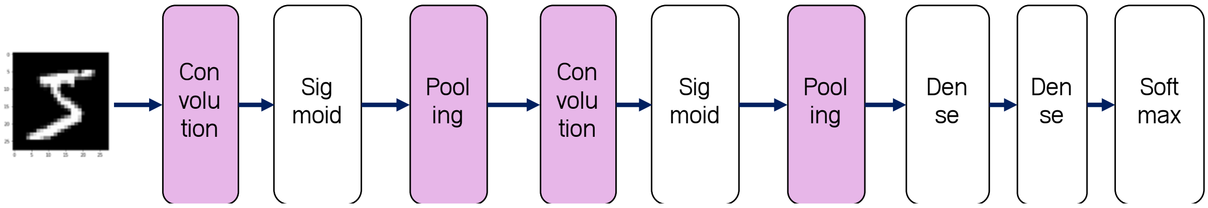
1. ទម្រង់និងដំណើរការទូទៅរបស់CNN

ដើម្បីស្វែងយល់អំពីCNN ជាដំបូងយើងនឹងពិនិត្យលើទម្រង់ជាទូទៅរបស់វាជាមុន ។ CNNក៏ដូចជាបណ្តាញNeural Network ដទៃដែរគឺកើតឡើងពីការផ្គុំបញ្ចូលគ្នានូវថ្នាក់ណឺរ៉ូនជាច្រើនថ្នាក់ ។ ប៉ុន្តែចំណុចពិសេសគឺនៅក្នុងCNN មានថ្នាក់ពិសេសពីរដែលមិនមាននៅក្នុងFeedforward Neural Network(FNN) ដែលយើងសិក្សាកន្លងមក គឺ ថ្នាក់Convolution និង ថ្នាក់ Pooling ។ យើងនឹងសិក្សាលម្អិតអំពីថ្នាក់ទាំងពីរនេះនៅចំណុចបន្ទាប់ ។

ក្នុងបណ្តាញFNN ណឺរ៉ូននៃថ្នាក់ដែលនៅជាប់គ្នាត្រូវបានភ្ជាប់គ្នាទាំងអស់ដោយទម្ងន់ផ្ទាល់របស់វា ។ ទម្រង់បែបនេះគេហៅថា តំណភ្ជាប់ពេញទី(fully-connected) ។ ក្នុងរូបនិងការបកស្រាយខាងក្រោម យើងនឹងបង្ហាញតំណភ្ជាប់ពេញទីបែបនោះដោយ Dense រីឯអនុគមន៍សកម្មនឹងបង្ហាញដោយឈ្មោះនៃអនុគមន៍នោះ ។ ក្នុងករណីនេះ ទម្រង់នៃបណ្តាញFNNអាចបង្ហាញបានដូចរូបទី១ ។ បណ្តាញFNNអាចត្រូវបាននិយាយថាជាបណ្តុំនៃតំណភ្ជាប់ពេញទីនិងអនុគមន៍សកម្មច្រើនថ្នាក់បន្តគ្នា ។ ផ្ទុយពីនេះ ទម្រង់នៃបណ្តាញCNNអាចបង្ហាញបានដូចរូបទី២ ។ ដូចដែលអ្នកបានឃើញថ្នាក់ថ្មី (Convolution និង Pooling)ត្រូវបានប្រើជាបន្តបន្ទាប់ មុនពេលដែលបណ្តុំនៃតំណភ្ជាប់ពេញទីនិងអនុគមន៍សកម្មត្រូវបានអនុវត្ត ។ ទម្រង់ដែលថ្នាក់ Convolution និង Pooling ត្រូវបានប្រើមុនត្រូវបានមើលឃើញជាទូទៅក្នុងបណ្តាញCNN ។



រូបទី១ ទម្រង់នៃបណ្តាញFNNដោយប្រើតំណភ្ជាប់ពេញទី(fully-connected layer)



រូបទី២ ទម្រង់នៃCNN

2. អំពីថ្នាក់ Convolution

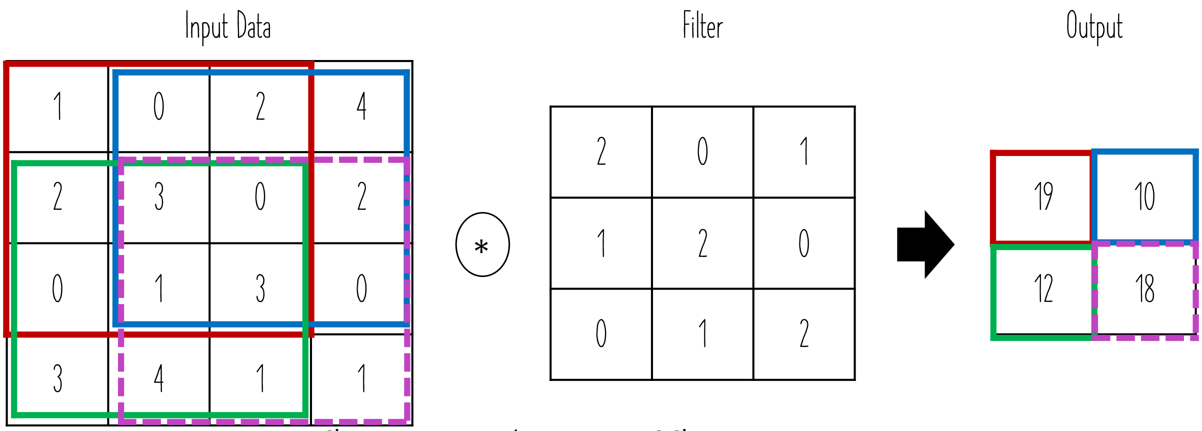
ជាដំបូងយើងនឹងពិនិត្យលើភាពខុសគ្នារវាងថ្នាក់តំណភ្ជាប់ពេញទី(fully-connected)ដែលត្រូវបានប្រើក្នុងFNNនិងថ្នាក់Convolutionនេះ។

2.1. បញ្ហានៃថ្នាក់តំណភ្ជាប់ពេញទី

នៅក្នុងបណ្ដាញFNNណឺរ៉ូននៃថ្នាក់តំណភ្ជាប់ពេញទីត្រូវបានភ្ជាប់គ្នាគ្រប់ណឺរ៉ូនទាំងអស់ដែលមាននៅថ្នាក់ជាប់គ្នា រីឯចំនួននៃណឺរ៉ូននៅថ្នាក់លទ្ធផលអាចត្រូវបានកំណត់ដោយសេរី។

បញ្ហានៃថ្នាក់តំណភ្ជាប់ពេញទីគឺ ទម្រង់នៃទិន្នន័យមិនត្រូវបានប្រើប្រាស់ឱ្យសមស្របនោះឡើយ។ ឧទាហរណ៍បើធាតុចូលជារូបថតដូចរូបទី១និងទី២ នោះធាតុចូលស្ថិតនៅក្នុងទម្រង់វិមាត្រ៣ពោល គឺ បណ្ដោយ ទទឹង និង channel(RGB)។ បើធាតុចូលនេះ ត្រូវបានបញ្ជូនទៅក្នុងFNNដែលមានថ្នាក់តំណភ្ជាប់ពេញទី នោះដូចដែលបានបង្ហាញនៅឧទាហរណ៍នៃមេរៀនអំពីFNN ទិន្នន័យនេះត្រូវបានពន្លាតជា១វិមាត្រដោយតម្រៀបគ្រប់តម្លៃនៃPixelទាំងអស់ជាកំប៉ុសង់នៃវ៉ិចទ័រ១វិមាត្រនោះ។ ក្នុងឧទាហរណ៍ដែលយើងបានសិក្សាកន្លងមក រូបថតដែលមានទម្រង់(28,28,1) ត្រូវបានប្រើជាវ៉ិចទ័រ១វិមាត្រដែលមានកំប៉ុសង់ចំនួន784(=28x28x1)។ ហេតុនេះការប្រើប្រាស់ថ្នាក់តំណភ្ជាប់ពេញទីធ្វើឱ្យខាតបង់ព័ត៌មានដែលយើងអាចទាញយកពីទម្រង់ដើមនៃទិន្នន័យដូចជាទិន្នន័យអំពីចម្ងាយនៃរបស់ក្នុងរូបដែលបង្ហាញដោយទម្រង់ RGB ច្រើនchannelជាដើម។

ផ្ទុយពីនេះ ថ្នាក់Convolution ទទួលយកធាតុចូលដោយរក្សាទម្រង់ដើមនៃទិន្នន័យ។ ក្នុងករណីរូបថតទម្រង់៣វិមាត្រដូចពណ៌នាខាងលើ ថ្នាក់Convolutionនឹងទទួលយករូបថតវិមាត្រ៣នោះជាធាតុចូលនិងបញ្ចេញរូបថតវិមាត្រ៣ជាលទ្ធផល។ ហេតុនេះ CNN អាចមានលទ្ធភាពក្នុងការសិក្សាបានស៊ីជម្រៅលើលក្ខណៈពិសេសរបស់ទិន្នន័យបាន។ នៅក្នុងCNN ធាតុចូលនៃថ្នាក់Convolution ហៅថា ផែនទីលក្ខណៈចូល(input feature map) និងធាតុចេញ(លទ្ធផល)របស់វាហៅថា ផែនទីលក្ខណៈចេញ(output feature map)។ ក្នុងករណីខ្លះយើងហៅជារួមថា feature map។



រូបទី៣ ឧទាហរណ៍នៃប្រមាណវិធីConvolution(⊛)

2.2. ប្រមាណវិធី Convolution

ក្នុងថ្នាក់Convolution ប្រមាណវិធីConvolutionត្រូវបានអនុវត្ត។ ប្រមាណវិធីនេះអាចប្រៀបបានជាដំណើរការនៃFilterនៅក្នុងបច្ចេកវិទ្យារូបភាព(Image Processing)។ ការធ្វើប្រមាណវិធី Convolution លើរូបភាពដែលមានទំហំ $W \times W$ និងFilter ដែលមានទំហំ$H \times H$ គឺកំណត់ដោយទម្រង់ខាងក្រោម។ u_{ij} ជាតម្លៃលទ្ធផលនៅទីតាំង(i,j) និង x_{st} ជាតម្លៃនៅPixel (s,t) និង h_{pq} ជាតម្លៃFilterនៃទីតាំង (p,q)។

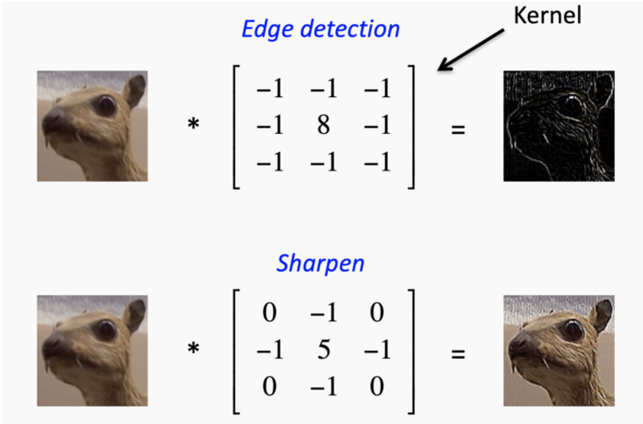
$$u_{ij} = \sum_{p=0}^{H-1} \sum_{q=0}^{H-1} x_{i+p,j+q} h_{pq}$$

រូបទី៣បង្ហាញពីឧទាហរណ៍នៃប្រមាណវិធីConvolution។ ឧបមាថាទិន្នន័យជាធាតុចូលមានទម្រង់ ២វិមាត្រ(height, width) ដែលមានទំហំ(4,4) និង Filterមានទំហំ(3,3)នោះលទ្ធផលធាតុចេញមានទំហំ(2,2)។ របៀបនៃការគណនាគឺ ចន្លោះនៃទិន្នន័យដែលមានទំហំដូចFilterត្រូវបានគណនាដោយធ្វើផលគុណគ្រប់ធាតុដែលមានទីតាំងត្រូវគ្នារួចធ្វើប្រមាណវិធីបូកបញ្ចូលគ្នា។ ឧទាហរណ៍នៅក្នុងចន្លោះពណ៌ក្រហម និងពណ៌បៃតងការគណនាត្រូវបានអនុវត្តដូចខាងក្រោម។

$$1 \times 2 + 0 \times 0 + 2 \times 1 + 2 \times 1 + 3 \times 2 + 0 \times 0 + 0 \times 0 + 1 \times 1 + 3 \times 2 = 19$$

$$2 \times 2 + 3 \times 0 + 0 \times 1 + 0 \times 1 + 1 \times 2 + 3 \times 0 + 3 \times 0 + 4 \times 1 + 1 \times 2 = 12$$

ដំណើរការConvolution នេះអាចទាញយកនូវលក្ខណៈឬទម្រង់ជាក់លាក់ដែលមាននៅទីតាំងផ្សេងៗនៃរូបភាពបានតាមការកំណត់Filterសមស្រប។ រូបទី៤បង្ហាញពីឧទាហរណ៍នៃលក្ខណៈនោះ។



រូបទី៤ ឧទាហរណ៍នៃដំណើរការទាញយកFeature MapដោយConvolution

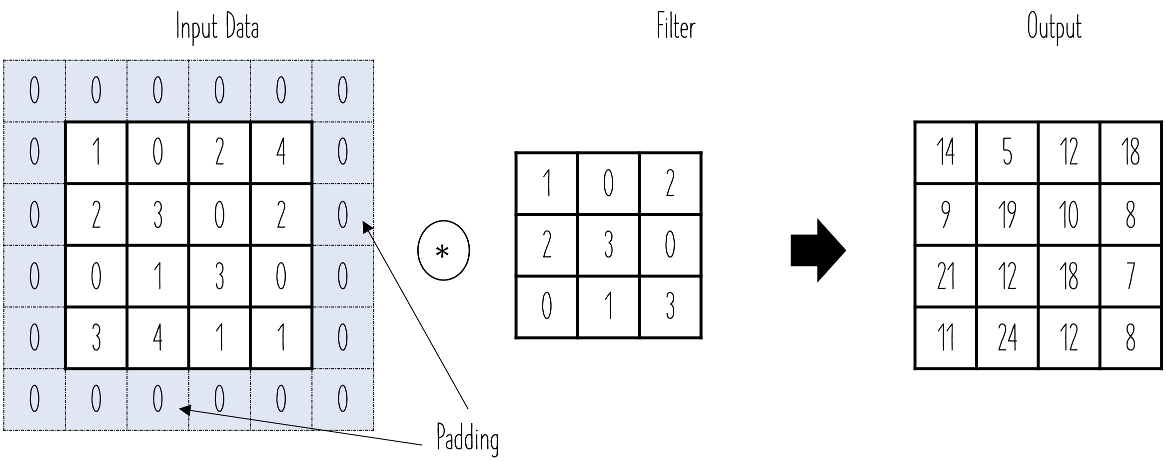
(Image from: Simple Introduction to Convolutional Neural Networks
https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac)

2.3. ការតម្រឹម Padding

ដូចដែលបានបង្ហាញខាងលើ តាមរយៈប្រមាណវិធីConvolution ទំហំនៃលទ្ធផលដែលទទួលបានមានទំហំតូចជាធាតុចូលដើមជានិច្ច។ ទំហំនៃលទ្ធផលនោះកំណត់ដោយទម្រង់ខាងក្រោម។ នៅទីនេះ $\lfloor x \rfloor$ ជាតម្លៃចំនួនគត់ធំបំផុតដែលតូចជាងx។

$$\left(W - 2\left\lfloor \frac{H}{2} \right\rfloor\right) \times \left(W - 2\left\lfloor \frac{H}{2} \right\rfloor\right)$$

ប៉ុន្តែក្នុងដំណើរការវិភាគទិន្នន័យ នៅដំណាក់កាលខ្លះការរក្សាទំហំនៃធាតុចូលនិងលទ្ធផលឱ្យដូចគ្នាជារឿងចាំបាច់។ ក្នុងករណីនេះដំណើរការនៃតម្រឹម(Padding) ត្រូវបានប្រើ។ ដំណើរការនេះគឺបន្ថែមតម្លៃនៅផ្នែកជាយជុំវិញនៃរូបភាពដោយតម្លៃសូន្យ។ រូបទី៥បង្ហាញពីគម្រូដំណើរការនេះ។



រូបទី៥ ឧទាហរណ៍នៃដំណើរការ Padding

2.4. ប្រមាណវិធីរំកិល Sride

ក្នុងការគណនាខាងលើ តំបន់ដែលប្រើដើម្បីអនុវត្តFilterត្រូវបានរំកិលម្តង១ៗពីឆ្វេងទៅស្តាំ និងពីលើចុះក្រោម។ ចន្លោះឬទំហំដែលត្រូវប្រើក្នុងដំណើរការរំកិលតំបន់ដើម្បីអនុវត្តFilter នេះហៅថា Sride។ ក្នុងករណីសាមញ្ញខាងលើ តម្លៃនៃ Sride ត្រូវបានកំណត់ដោយ 1 ។ ជាទូទៅក្នុងករណីដែលតម្លៃ Srideគឺ s លទ្ធផលនៃConvolution ត្រូវបានកំណត់ដោយទម្រង់ខាងក្រោម។

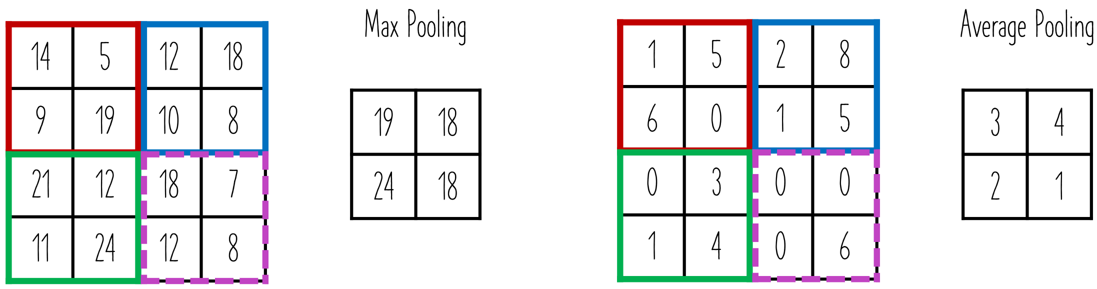
$$u_{ij} = \sum_{p=0}^{H-1}\sum_{q=0}^{H-1} x_{si+p,sj+q} h_{pq}$$

ក្នុងករណីដែលប្រើ Padding ជាមួយផង លទ្ធផលនឹងផ្តល់ឱ្យដោយទំហំខាងក្រោម។

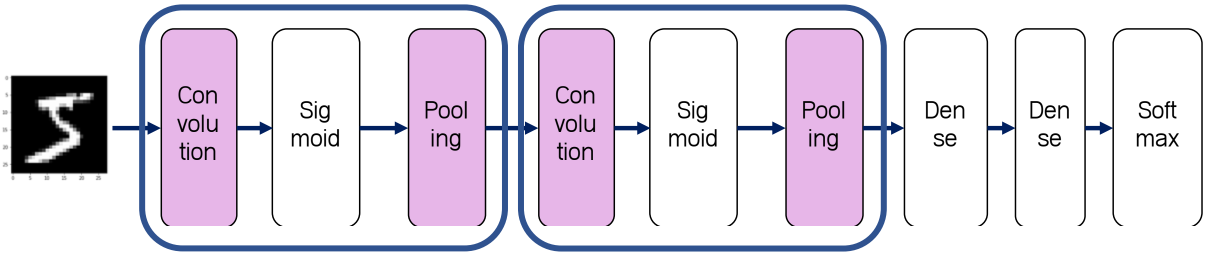
$$\left(\left\lfloor \frac{W-1}{s} \right\rfloor + 1\right) \times \left(\left\lfloor \frac{W-1}{s} \right\rfloor + 1\right)$$

3. ថ្នាក់ Pooling

ក្នុងថ្នាក់Pooling ប្រមាណវិធីដែលអនុវត្តគឺការបង្រួមទំហំនៃទិន្នន័យឱ្យរួមតូចជាងមុនដោយកំណត់យកតម្លៃតំណាងនៅក្នុងតំបន់។ ជាទូទៅថ្នាក់Pooling មានពីរផ្អែកលើប្រភេទនៃតម្លៃតំណាងដែលកំណត់យក ពោលគឺ តម្លៃអតិបរមាក្នុងតំបន់ និងតម្លៃមធ្យមក្នុងតំបន់។ ក្នុងករណីតម្លៃអតិបរមាត្រូវបានប្រើប្រាស់ត្រូវបានហៅថា Max Pooling ឯករណីតម្លៃមធ្យមត្រូវបានប្រើហៅថា Average Pooling។ លក្ខណៈពិសេសនៅដំណាក់កាលនេះគឺ គ្មានប៉ារ៉ាម៉ែត្រដែលត្រូវរៀន។



រូបទី៦ ឧទាហរណ៍នៃដំណើរការ Pooling



រូបទី៧ ទម្រង់នៃCNN

4. ទម្រង់នៃបណ្តាញCNNនិងការរៀន(Learning)

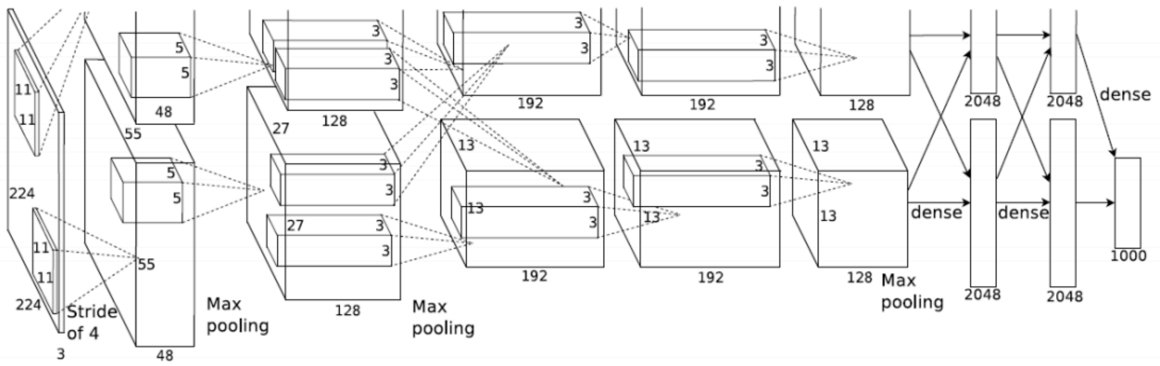
ដូចបង្ហាញខាងដើម CNN ត្រូវបានប្រើច្រើនក្នុងបច្ចេកវិទ្យារូបភាព។ ទម្រង់នៃបណ្តាញនេះច្រើនកំណត់ដោយទម្រង់ដូចរូបទី៧ ដោយប្រើថ្នាក់ដែលកើតពីConvolutionនិងPoolingច្រើនដង និងប្រើថ្នាក់ដែលកើតពីតំណភ្ជាប់ពេញទីចុងក្រោយ។ ចំពោះចំណោទធ្វើចំណាត់ថ្នាក់រូបភាពច្រើនក្រុម យើងអាចបកស្រាយបានថា ថ្នាក់ដែលប្រើConvolutionនិងPooling ជាថ្នាក់ដែលទាញយកលក្ខណៈសម្គាល់ពិសេស(feature map)ពីរូបភាព រីឯថ្នាក់កើតពីតំណភ្ជាប់(Dense)ត្រូវបានប្រើដើម្បីធ្វើចំណាត់ថ្នាក់ក្រុមដូចក្នុងFNNដែរ។

ដូចដែលអ្នកអាចធ្វើការកត់សម្គាល់បាន ថ្នាក់និមួយៗនៃCNNត្រូវបានតភ្ជាប់គ្នាជាបន្តបន្ទាប់ប្រៀបដូចជាថ្នាក់នៃFNNដែរ។ ហេតុនេះការកំណត់តម្លៃប៉ារ៉ាម៉ែត្រនៃថ្នាក់និមួយៗ(ដំណើរការរៀន) អាចធ្វើបានដោយប្រើ Backpropagation Algorithmដូចក្នុងករណីFNNផងដែរ។ អ្នកអាចប្រើប្រាស់ FrameWork ដែលផ្តល់ឱ្យប្រើជាសេរីជាច្រើនដើម្បីបង្កើតម៉ូឌែលCNNបានដូចជា Keras, PyTorch, Tensorflow ជាដើម។

5. អនុវត្តន៍នៃបណ្តាញCNN

CNNត្រូវបានអភិវឌ្ឍជាបណ្តាញធំនិងស្មុគស្មាញដើម្បីដោះស្រាយបញ្ហានានាពិសេសក្នុងបច្ចេកវិទ្យាធ្វើកំណត់សម្គាល់វត្ថុក្នុងរូបភាព។ រូបទី៨បង្ហាញពីគម្រូនៃម៉ូឌែលដែលប្រើCNNដើម្បីធ្វើចំណាត់ថ្នាក់1000ក្រុមនៃវត្ថុក្នុងរូបភាពដែលត្រូវបានស្គាល់ថាផ្តល់នូវលទ្ធផលល្អប្រសើរខ្លាំង។ សម្រាប់ចំណេះដឹងបន្ថែមទាក់ទងនឹងការប្រកួតប្រជែងបង្កើតម៉ូឌែលសម្រាប់ធ្វើចំណាត់ថ្នាក់ក្រុមវត្ថុក្នុងរូបភាព មិត្តអ្នកអានអាចចូលទៅកាន់តំណភ្ជាប់ខាងក្រោម។

http://www.image-net.org/challenges/LSVRC/



Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In Proc. of NIPS, pp. 1097-1105.

រូបទី៨ គម្រូម៉ូឌែលដែលប្រើCNN