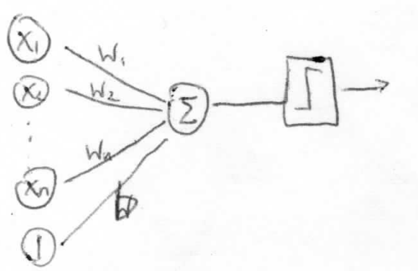
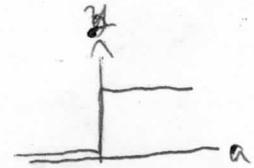
CODEN 296A HW3 Yuan-lin Hsu W1279028

1.



$$a = \sum_i w_i x_i + b$$

$$y = \begin{cases} 1 & a \geq 0 \\ 0 & 0 \end{cases}$$



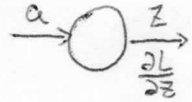
$$\frac{\partial L}{\partial w_i} = \frac{\partial L}{\partial y} \cdot \frac{\partial y}{\partial w_i}$$

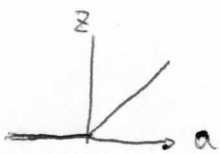
$$\frac{\partial y}{\partial w_i} = \underbrace{\frac{\partial y}{\partial a}}_{0} \cdot \frac{\partial a}{\partial w_i}$$

It turns out the gradient is always 0.

2.

$$z = \begin{cases} a & a \geq 0 \\ 0 & a < 0 \end{cases}$$

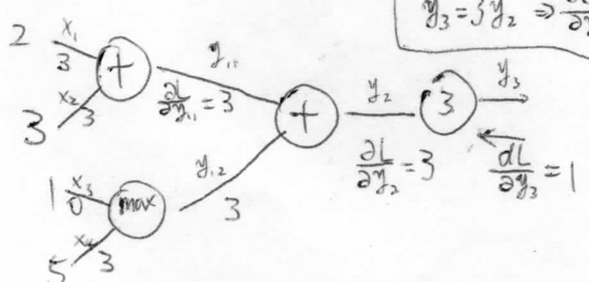




$$\frac{\partial L}{\partial a} = \frac{\partial L}{\partial z} \cdot \frac{\partial z}{\partial a} = \begin{cases} \frac{\partial L}{\partial z} & a \geq 0 \\ 0 & a < 0 \end{cases}$$

$$\frac{dz}{\partial a} = \begin{cases} 1 & a \geq 0 \\ 0 & a < 0 \end{cases}$$

3.



$$y_3 = 3y_2 \Rightarrow \frac{\partial y_3}{\partial y_2} = 3$$

$$\Rightarrow \frac{\partial L}{\partial y_2} = \frac{\partial L}{\partial y_3} \cdot \frac{\partial y_3}{\partial y_2} = 3$$

$$y_2 = y_{11} + y_{12}$$

$$\frac{\partial y_2}{\partial y_{11}} = 1 \quad \Rightarrow \frac{\partial L}{\partial y_{11}} = \frac{\partial L}{\partial y_2} \cdot \frac{\partial y_2}{\partial y_{11}} = 3 \cdot 1 = 3$$

$$\frac{\partial y_2}{\partial y_{12}} = 1 \quad \Rightarrow \frac{\partial L}{\partial y_{12}} = \frac{\partial L}{\partial y_2} \cdot \frac{\partial y_2}{\partial y_{12}} = 3$$

$$y_{11} = x_1 + x_2$$

$$\vdots$$

$$\frac{\partial L}{\partial x_1} = 3$$

$$\frac{\partial L}{\partial x_2} = 3$$

$$y_{12} = \max(x_3, x_4) = \begin{cases} x_3 & x_3 \geq x_4 \\ x_4 & x_3 < x_4 \quad \checkmark \end{cases}$$

$$5 = x_4 > x_3 = 1$$

$$y_{12} = x_4$$

$$\frac{\partial y_{12}}{\partial x_3} = 0 \qquad \frac{\partial y_{12}}{\partial x_4} = 1 \quad \Rightarrow \quad \frac{\partial L}{\partial x_3} = 0 \qquad \frac{\partial L}{\partial x_4} = 3$$

Problem 4

1. Number of Parameter (Using bias)
Dense layer(input = 10, output =50) : (10+1) * 50 = 550
Relu layer: None
Dense layer(input = 50, output = 3): (50+1)*3 = 153
Softmax layer: None
Total : 703 parameters

2.
The code is attached, basically it has two files:
layer.py defines computing layers that can compute 1-D output from 1-D input.
Main.py initilize the input and target and model, perform backprop training.

layer.py
```
import numpy as np
# A layer handles one dimensional input x, apply its computation and return 1-D result y
class Layer:
	# apply layer computation
	def forward(self, x):
		pass
	#  returns dL/dx
	#  saves gradient
	def backward(self, dy):
		pass
	# adjust weight by gradient descend
	def adjust_w(self, lr):
		pass

class dense_layer(Layer):
	def __init__(self, inputNum, unitNum):
		self.w = np.random.uniform(0., 0.1, (inputNum, unitNum))
		self.b = np.random.uniform(0., 0.1, (1, unitNum))
		self.input = None
		self.gradient = None
		self.b_grad =None
	def forward(self, x):
		self.input = x
		return np.dot (x ,self.w)+self.b
	def backward(self, dy):
		self.gradient = (dy*self.input).T
		self.b_grad = dy.T
		return np.dot(self.w , dy)
	def adjust_w(self, lr):
		self.w -= lr*self.gradient
		self.b -= lr*self.b_grad

class relu_layer(Layer):
	def __init__(self):
		self.input = None
	def forward(self, x):
		self.input = x
```

```
        y = x.copy()
        y[y<0]=0
        return y
    def backward(self, dy):
        y = self.input.copy()
        y[y>=0] = 1
        y[y<0] = 0
        return dy*y.T

class softmax_layer(Layer):
    def __init__(self):
        self.input = None
        self.output= None
    def forward(self, x):
        self.input = x
        y = np.exp(x)
        self.output = y/np.sum(y)
        return self.output
    def backward(self, dy):
        res = np.dot((np.eye(dy.shape[0]) - self.output * self.output.T), dy)
        return np.dot((np.eye(dy.shape[0]) - self.output*self.output.T),dy)
```

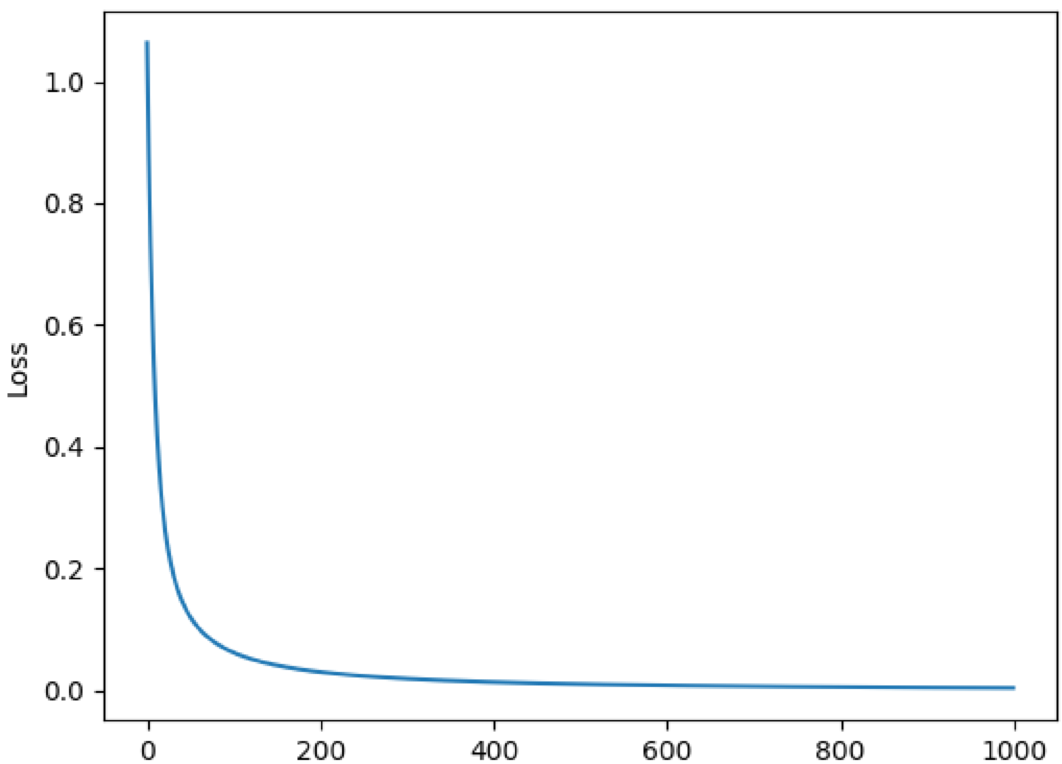
main.py

```
from layer import *
import numpy as np
import matplotlib.pyplot as plt
# Model is a list of layers
# compute the result of input_x put into model
def forward_pass(input_x, model):
    for layer in model:
        input_x = layer.forward(input_x)
    return input_x
# compute the result of dy back propagated through the model
# the result is somewhat meaningless, what's important is it calls the backward method in each layer
# that method will compute and save the gradient change of weight (if applicable)
def backward_pass(dy, model):
    for layer in model[::-1]:
        dy = layer.backward(dy)
    return dy
# Calles the adjust weight method in each layer, gradient descend.
def adjust_w(model,lr):
    for layer in model:
        layer.adjust_w(lr)
learning_rate = 0.02
num_iter =1000
test_run_times = 1
# Test run for test_run_times times
for test_time in range(test_run_times):
    # create input
    input_x = np.asarray([[0.5, 0.6, 0.1, 0.25, 0.33, 0.9 , 0.88, 0.76, 0.69, 0.95]], dtype= np.float32)
    # create target
    Target = np.asarray([[1,0,0]], dtype= np.float32)
    # create model
    model = [dense_layer(10, 50), relu_layer(), dense_layer(50, 3), softmax_layer()]
    loss_rec = []
```

```
        # training for num_iter iterations
        for i in range(num_iter):
                # forward pass
                y = forward_pass(input_x,model)
                # Loss function = -sum(t log y)
                loss_rec.append( -float(np.dot(np.log(y), Target.T)))
                dy = (y-Target).T
                backward_pass(dy,model)
                adjust_w(model, learning_rate)
        # print the output
        # print (y)
# # print out final parameter
# print ('W1--')
# print (model[0].w)
# print ('b1--')
# print (model[0].b)
# print ('W2--')
# print (model[2].w)
# print ('b2--')
# print (model[2].b)
# # plot loss function
# plt.plot(loss_rec)
# plt.ylabel('Loss')
# plt.show()
```

RESULTS: (Trained after 1000 iterations)

W1
--
[[
0.0
580
895
6
0.0
911
314
5
0.0
989
536
0.0
160
188
5
0.0
797
988
3
0.0



1143851
 0.03128024 0.10106539 0.0320729 0.02923494 0.02221748 0.05524153
 0.05445491 0.04519545 0.03270548 0.04032882 0.09527076 0.04163014
 0.07700663 0.11063762 0.07115647 0.06781337 0.03249227 0.06746283
 0.02521299 0.02945518 0.09608611 0.10260118 0.08880348 0.07072252
 0.07144613 0.09413784 0.06170571 0.06059294 0.0471615 0.03154458
 0.08465086 0.03151966 0.04707593 0.06847036 0.10529846 0.10631239
 0.08770343 0.07103896 0.0398511 0.07142172 0.08265725 0.03772662
 0.05028481 0.06518065]
[0.064567 0.02106997 0.02579032 0.11838416 0.04063822 0.05002368
 0.06607066 0.06488126 0.01813963 0.05762437 0.11453203 0.05141145
 0.09351833 0.10499778 0.07187853 0.05452165 0.11858633 0.06661754
 0.08518465 0.04031766 0.03854334 0.09365475 0.07343654 0.03497012
 0.05498424 0.01484873 0.04744681 0.07463678 0.11691485 0.04092984
 0.0831933 0.06182988 0.10335742 0.03004602 0.05786266 0.07896586
 0.05131834 0.03406516 0.03723967 0.03361553 0.0546066 0.11554644
 0.06604961 0.02163947 0.0676459 0.09536983 0.04880066 0.0605911
 0.02755653 0.00464075]
[0.02518013 0.05584189 0.05758035 0.01133409 0.02331809 0.02387723
 0.05516844 0.0124741 0.07881343 0.02689955 0.08984099 0.04029195
 0.01721395 0.07337634 0.02652364 0.02625484 0.00771451 0.03613118
 0.04665585 0.02145958 0.04813455 0.00648882 0.08897534 0.0385972
 0.10095235 0.03269028 0.00893302 0.0510623 0.09261308 0.01514977
 0.07888019 0.05789586 0.06783026 0.00988538 0.03244385 0.03856679
 0.04097895 0.02442144 0.03866803 0.07852467 0.04914144 0.01900921
 0.01284569 0.00302955 0.1028465 0.07370497 0.03347643 0.01473491
 0.05339992 0.08506344]
[0.09394175 0.10027573 0.06229422 0.01225484 0.08373915 0.07372691
 0.07821413 0.02534987 0.07108417 0.10152304 0.08827903 0.09888214
 0.0802618 0.01114953 0.05115023 0.06788173 0.09500577 0.02890238
 0.03546888 0.0682672 0.01829403 0.00849605 0.07364792 0.06952922
 0.02399921 0.03211698 0.093427 0.05862371 0.10926898 0.01273621
 0.01137748 0.00593908 0.02955383 0.01881257 0.04557077 0.07986155
 0.09863209 0.05508813 0.10382107 0.07006033 0.04677805 0.05875323
 0.0313171 0.02116702 0.06461559 0.08383384 0.10070428 0.06122441
 0.09473434 0.01082961]
[0.01000632 0.0564162 0.06362393 0.08504401 0.08577246 0.0351951
 0.09813808 0.10629549 0.10121194 0.097221 0.0313929 0.02144861
 0.0755163 0.10838985 0.0504014 0.01567844 0.0837876 0.02875432
 0.02566007 0.05817347 0.08243324 0.0135372 0.0394488 0.02461949
 0.10364146 0.04424107 0.00581366 0.06722652 0.02704596 0.02431993
 0.06390033 0.03940289 0.05768311 0.10915349 0.03900329 0.10104013
 0.02264686 0.02450194 0.06689168 0.10876105 0.10263686 0.03093682
 0.07654701 0.01531561 0.05288771 0.01420175 0.01257525 0.01379188
 0.07914409 0.08277996]
[0.07934558 0.09153747 0.03271549 0.06868235 0.07525509 0.09405787
 0.04825234 0.11056256 0.10305007 0.07695368 0.09509798 0.01523673
 0.08782484 0.10952662 0.02739284 0.07929685 0.12551231 0.11208467
 0.12812481 0.1218433 0.09617899 0.07134238 0.01483019 0.03396974

```
  0.13363912  0.03594574  0.06556115  0.13375185  0.15918998  0.08476387
  0.00807384  0.07743546  0.07287365  0.11983212  0.1508513   0.01389883
  0.01605478  0.08697082  0.09118785  0.0597366   0.08193094  0.10175946
  0.06679154  0.07217024  0.05649456  0.03553708  0.04371036  0.09727786
  0.08593235  0.02662613]
 [ 0.04100345  0.10281227  0.06220345  0.1211128   0.12795267  0.02126691
  0.0177961   0.07877285  0.08642891  0.0951072   0.07153676  0.04461239
  0.07731786  0.08578988 -0.00174622  0.0598929   0.09543352  0.05373856
  0.11606308  0.11595442  0.06349118  0.06945169  0.079751    0.04571902
  0.0857313   0.09208108  0.03436275  0.14953446  0.0964021   0.06796319
  0.01888825  0.06977422  0.11495107  0.06584041  0.05662315  0.02936783
  0.05216511  0.07013196  0.07090709  0.13581575  0.07197694  0.12667495
  0.07185898  0.09197667  0.12824357  0.08498593  0.06777458  0.09188248
  0.02238969  0.03664783]
 [ 0.07667853  0.03803191  0.1052409   0.08829164  0.13109303  0.06632565
  0.0151252   0.05683594  0.01131438  0.06486098  0.04495039  0.0317415
  0.05599756  0.10108681  0.02864504  0.02856899  0.07841248  0.05627999
  0.06186089  0.09105988  0.1259936   0.02668051  0.04557237  0.02364276
  0.03493198  0.04662474  0.01995776  0.08915096  0.0871132   0.00718985
  0.00641974  0.10596005  0.1016106   0.07693109  0.11487444  0.04686452
  0.03839376  0.01864267  0.12936948  0.05818062  0.11018869  0.06596105
  0.11129553  0.03437721  0.09940616  0.02546598  0.00526394  0.04258121
  0.06802825  0.01881304]
 [ 0.07857729  0.07704514  0.02515695  0.05026397  0.1063815   0.08854564
  0.09054952  0.0683148   0.08918206  0.10038804  0.09965872  0.09085201
  0.08627726  0.122835    0.03673736  0.03910876  0.12559007  0.03958985
  0.09906525  0.06905816  0.07465182  0.10670671  0.04514845  0.07046186
  0.06631797  0.05964784  0.06889122  0.13916355  0.14767483  0.02591129
  0.02163172  0.05482648  0.07933494  0.0437091   0.10834851  0.05792852
  0.09812927  0.02179026  0.13023513  0.0707054   0.10654697  0.08033446
  0.07305051  0.05974568  0.1181528   0.02944974  0.07985787  0.05702111
  0.10703407  0.08042671]
 [ 0.00541826  0.06322825  0.0337482   0.0872923   0.07139013  0.10686484
  0.02557412  0.06884342  0.10251861  0.04656666  0.1255019   0.0594514
  0.0227835   0.10585919  0.0063931   0.05188517  0.12381291  0.06177958
  0.08672268  0.03199471  0.1221477   0.0392057   0.05613599  0.06540413
  0.13802853  0.01491393  0.01492597  0.15142244  0.17071298  0.04656709
  0.06589383  0.03886609  0.13356947  0.04535928  0.14732928  0.0609863
  0.10020022  0.08964012  0.04942207  0.10859107  0.03054303  0.1212724
  0.12806858  0.03253636  0.0897993   0.06125982  0.02315696  0.11047774
  0.06075791  0.04228247]]
b1--
[[ 0.07648277  0.08390097  0.0458936   0.12206165  0.07426317  0.03286757
  0.00353583  0.09766271  0.01059815  0.07377983  0.13701017  0.09979437
  0.09827094  0.06910949  0.04560853  0.03987865  0.13356342  0.04388905
  0.12093293  0.10314727  0.13988762  0.08859597  0.0017254   0.03961387
  0.08829636  0.0746456   0.00797326  0.07577204  0.17523126  0.07986447
  0.03526397  0.04212546  0.07053328  0.09564979  0.10905019  0.0496339
  0.03478971  0.06125904  0.14121568  0.14312749  0.11579269  0.1191895
```

```
  0.13849766  0.10775523  0.1448925   0.11240132  0.02647883  0.09016888
  0.06898489  0.07395386]]
W2--
[[ 0.09539748 -0.04719412 -0.0273635 ]
 [ 0.09638559 -0.09621507 -0.09141525]
 [ 0.13381522 -0.02435404 -0.05079714]
 [ 0.08836264 -0.09300564 -0.13620406]
 [ 0.16118257 -0.12084766 -0.1164355 ]
 [ 0.09297639 -0.08455855 -0.06020497]
 [ 0.11533331  0.00946229 -0.03462952]
 [ 0.11474802 -0.10954376 -0.11260812]
 [ 0.08978397 -0.07561703 -0.04397272]
 [ 0.14306111 -0.09078792 -0.05538734]
 [ 0.16712402 -0.07875322 -0.14484961]
 [ 0.06686465 -0.0693865  -0.05732172]
 [ 0.08142895 -0.11263193 -0.07238829]
 [ 0.16833826 -0.13075834 -0.08455325]
 [ 0.11054697  0.00947574 -0.01147339]
 [ 0.09530449 -0.09097251  0.00147248]
 [ 0.16087249 -0.14898541 -0.15951849]
 [ 0.07489582 -0.06572471 -0.09410005]
 [ 0.17330194 -0.12693474 -0.13565562]
 [ 0.15429878 -0.1198464  -0.07067835]
 [ 0.18521263 -0.10642638 -0.11538944]
 [ 0.10865978 -0.07257466 -0.06274382]
 [ 0.06243682 -0.00122461 -0.06670557]
 [ 0.09155326 -0.04928229 -0.00627804]
 [ 0.10593718 -0.13002516 -0.11183183]
 [ 0.08490456 -0.01056927 -0.05309754]
 [ 0.11304571 -0.01238844 -0.01559002]
 [ 0.17188937 -0.18414128 -0.13106186]
 [ 0.21274284 -0.1685476  -0.20938388]
 [ 0.12286327 -0.04354345 -0.01987844]
 [ 0.13476411 -0.0238297   0.00444901]
 [ 0.07530982 -0.07312064 -0.07375482]
 [ 0.14922007 -0.11726258 -0.11679191]
 [ 0.12186799 -0.07035391 -0.11786117]
 [ 0.14844553 -0.14595067 -0.15294965]
 [ 0.07087154 -0.00333686 -0.08609652]
 [ 0.07596119 -0.05154915 -0.07163488]
 [ 0.06797005 -0.09387584 -0.01266832]
 [ 0.14271627 -0.13880781 -0.111151  ]
 [ 0.18130465 -0.09290625 -0.14740137]
 [ 0.12082977 -0.12026676 -0.06732825]
 [ 0.19463354 -0.12584711 -0.1231164 ]
 [ 0.159968   -0.10664843 -0.14981373]
 [ 0.16422355 -0.01843011 -0.0790092 ]
 [ 0.19068177 -0.09945914 -0.13863229]
 [ 0.12548933 -0.08314604 -0.04689887]
```

```
 [ 0.06066171 -0.04494361 -0.04348533]
 [ 0.12823521 -0.10133458 -0.07719611]
 [ 0.14782328 -0.0560994  -0.06595343]
 [ 0.09237416 -0.00575724 -0.04513005]]
b2--
[[ 0.20492658 -0.27426067 -0.25118906]]
```