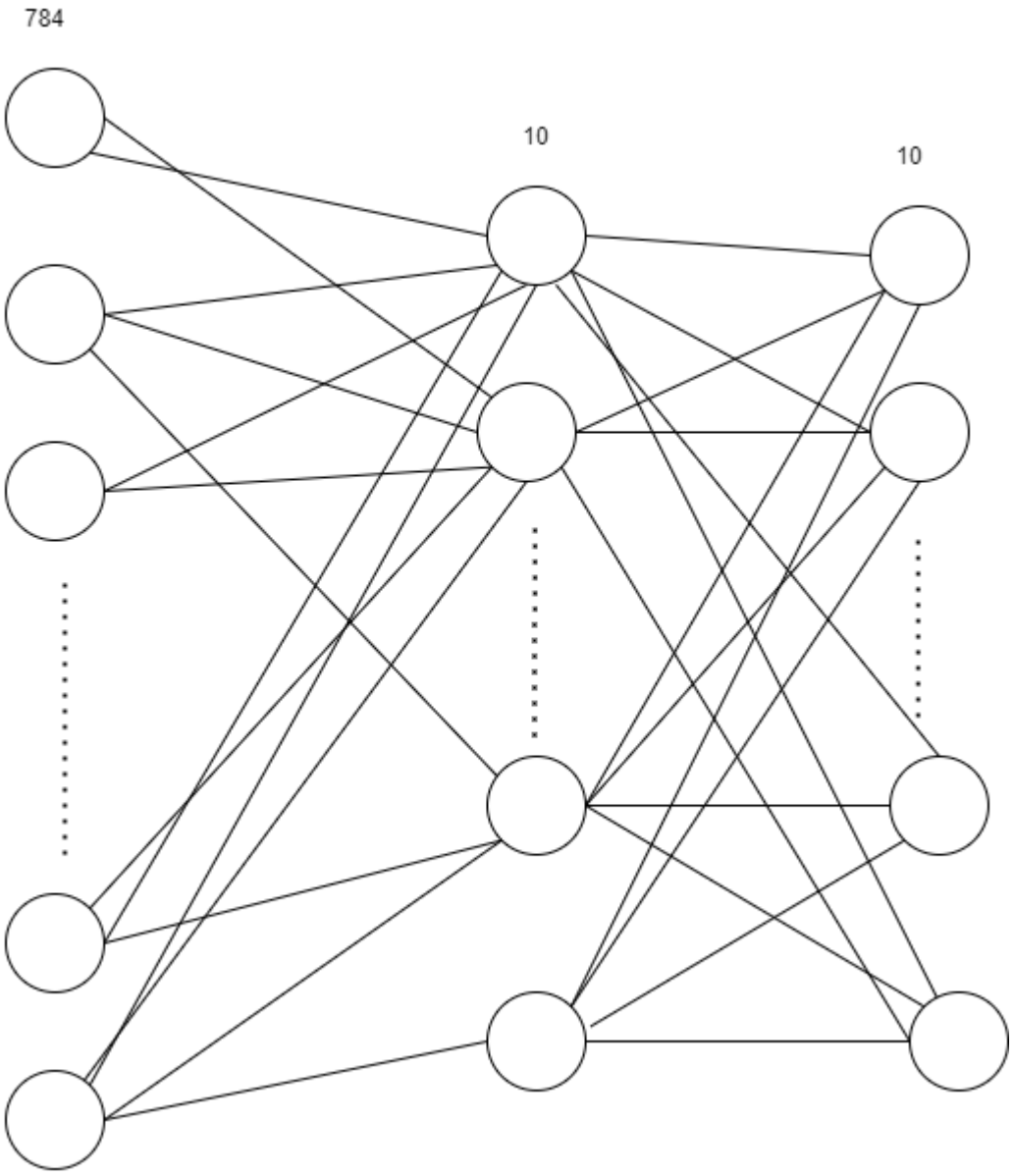
NPM : 6181801040

Nama : David Dimas Patty

Tugas Besar Deep Learning

Model Neural Network



Kelompok kami memilih model neural network dengan 3 layer (1 input) dengan layer pertama memiliki 784 node (dimensi gambar 28 x 28), layer kedua memiliki 10 node, dan layer output memiliki 10 node dikarenakan hasil dari beberapa percobaan yang gagal (terutama saat percobaan back prop), selain itu kami juga mengiginkan layer akhir berisi

value yang paling besar dari 10 node yang nantinya value yang paling besar tersebut adalah merupakan output dari model kami. Selain itu model juga terpengaruh dari refrensi (https://www.youtube.com/watch?v=w8yWXqWQYmU&t=915s).

Activation Function

Activation yang digunakan pada model yang kelompok kami buat yaitu ReLu dan Softmax. Digunakan Relu karena untuk memangkas weight*input+bias yang negative pada saat masuk ke layer pertama dan dijadikan 0. Lalu, dari hasil ReLu pada layer 1 akan diteruskan ke layer 2. Setelah perhitungan weight*input+bias pada setiap node di layer 2, maka perhitungan tersebut masuk ke fungsi aktifasi softmax. Dimana pada softmax hasil dari perhitungan pada layer 2, masing-masing node akan dibagi dengan jumlah semua node sehingga meminimalisir value dari array >1.

- Layer pertama semua node menggunakan menggunakan ReLu
    - Menggunakan ReLu dikarenakan agar menghilangkan Z yang < 0, karena akan berpengaruh saat perhitungan softmax pada layer selanjutnya dan memungkinkan divider pada softmax untuk 0 dan menghasilkan infinity value pada array.
- Layer kedua semua node menggunakan menggunakan Softmax
    - Menggunakan Softmax dikarenakan

Forward Prop

- Layer 1 Formula:

$$Z^{[1]} = W^{[1]} * X + b^{[1]}$$

- Relu Activation Function Formula :

$$A^{[1]} = gReLU(Z^{[1]})$$

- Layer 2 Formula:

$$Z^{[2]} = W^{[2]} * A^{[1]} + b^{[2]}$$

- Softmax Activation Function Fornula:

$$A^{[2]} = gsoftmax(Z^{[2]})$$

Backward Prop

M merupakan banyaknya data pada training

- Layer 2 Deverative:

$$dZ^{[2\ ]} = A^{[2]} - Y$$

Y merupakan array kosong yang berisi 1 pada indeks=label

- Weight 2 derivative:

$$dW^{[2]} = 1 / m * dZ^{[2]} * A^{[1]T}$$

- Bias 2 derivative:

$$dB^{[2]} = 1 / m * \Sigma (dZ^{[2]})$$

- Layer 1 derivative:

$$dZ^{[1]} = W^{[2].T} * dZ[2] .* g^{[1]'} (z^{[1]})$$

g merupakan derivative

- Weight 1 derivative:

$$dW^{[1]} = 1/ m * dZ^{[1]} * A^{[0]T}$$

- Bias 1 derivative:

$$dB^{[1]} = 1 / m * \Sigma (dZ^{[1]})$$

Optimasi:

- Setiap weight dan bias dikurangi dengan deverative dari masing-masing (variable-variable dW,db didapatkan dari backprop) dan dikali learning rate(α):

$$W^{[2]} = W^{[2]} – \alpha * dW^{[2]}$$

$$b^{[2]} = b^{[2]} – \alpha * db^{[2]}$$

$$W^{[1]} = W[1] - \alpha * dW^{[1]}$$

$$b^{[1]} = b^{[1]} – \alpha * db^{[1]}$$

Hasil Training:

Setelah 10000 kali iterasi model yang dibuat mengalami gradient explode dimana weight menjadi nan sehingga error mengalami pembengkakan. Oleh sebab itu, diimplementasikan early stop. Dimana method early stop disini memberhentikan jika variable loop ke- variable stop iterasi selanjutnya akurasi mengalami penurunan.

Tanpa early stop:

```
Predict:  tensor([7, 2, 2,  ..., 2, 7, 0]) Y: tensor([7, 2, 1,  ..., 5, 7, 8])
torch.Size([8000])
0.053
Iteration:  9996
Predict:  tensor([7, 2, 2,  ..., 2, 7, 0]) Y: tensor([7, 2, 1,  ..., 5, 7, 8])
torch.Size([8000])
0.053
Iteration:  9997
Predict:  tensor([7, 2, 2,  ..., 2, 7, 0]) Y: tensor([7, 2, 1,  ..., 5, 7, 8])
torch.Size([8000])
0.053
Iteration:  9998
Predict:  tensor([7, 2, 2,  ..., 2, 7, 0]) Y: tensor([7, 2, 1,  ..., 5, 7, 8])
torch.Size([8000])
0.053
Iteration:  9999
Predict:  tensor([7, 2, 2,  ..., 2, 7, 0]) Y: tensor([7, 2, 1,  ..., 5, 7, 8])
torch.Size([8000])
0.053
```

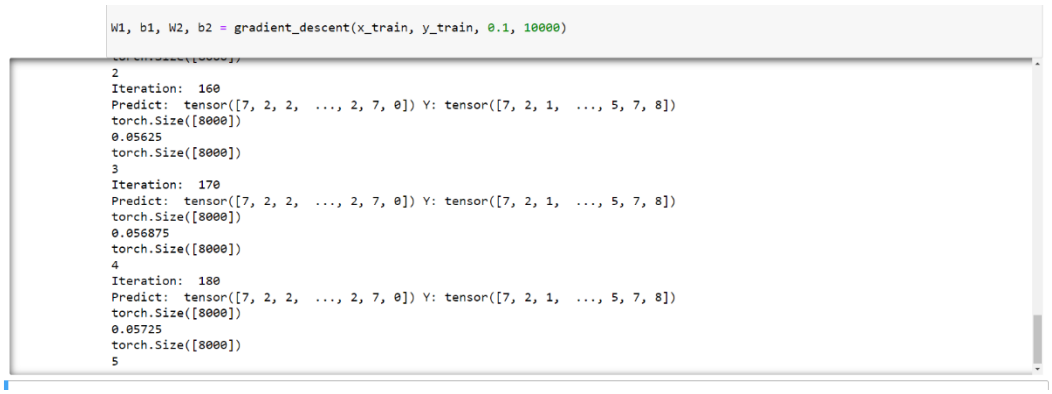
Saat memakai early stop dengan variable fatal=8 :

```
W1, b1, W2, b2 = gradient_descent(x_train, y_train, 0.01, 10000,8)
torch.Size([8000])
5
Iteration:  1000
Predict:  tensor([7, 2, 2,  ..., 0, 7, 0]) Y: tensor([7, 2, 1,  ..., 5, 7, 8])
torch.Size([8000])
0.05825
torch.Size([8000])
6
Iteration:  1010
Predict:  tensor([7, 2, 2,  ..., 0, 7, 0]) Y: tensor([7, 2, 1,  ..., 5, 7, 8])
torch.Size([8000])
0.0585
torch.Size([8000])
7
Iteration:  1020
Predict:  tensor([7, 2, 2,  ..., 0, 7, 0]) Y: tensor([7, 2, 1,  ..., 5, 7, 8])
torch.Size([8000])
0.0585
torch.Size([8000])
8
```

Diketahui jika memakai metode early stop, akurasi yang didapatkan dari model hanya mencapai 5,8%. Akurasi tersebut masih sangat rendah.

Variabel Hyper Parameter:

- Learning Rate= 0.01
    - Pada model kami, jika LR >= 0.1 akan mengalami explode pada gradien.



```
W1, b1, W2, b2 = gradient_descent(x_train, y_train, 0.1, 10000)
```

```
2
Iteration:  160
Predict:  tensor([7, 2, 2,  ..., 2, 7, 0]) Y: tensor([7, 2, 1,  ..., 5, 7, 8])
torch.Size([8000])
0.05625
torch.Size([8000])
3
Iteration:  170
Predict:  tensor([7, 2, 2,  ..., 2, 7, 0]) Y: tensor([7, 2, 1,  ..., 5, 7, 8])
torch.Size([8000])
0.056875
torch.Size([8000])
4
Iteration:  180
Predict:  tensor([7, 2, 2,  ..., 2, 7, 0]) Y: tensor([7, 2, 1,  ..., 5, 7, 8])
torch.Size([8000])
0.05725
torch.Size([8000])
5
```

Dapat dilhat dari gambar diatas hanya dalam 190 iterasi, sudah mengalami early stop dikarenakan akurasinya yang selalu berkurang.

- epoch = 10000
    - Jika epoch > 10000, karena model yang kita buat memiliki gradient explode hasil A2 (error pada layer 2) akan sama saja.

- stop = 5
    - Fatal merupakan variable yang dibuat untuk melihat berapa banyak akurasi turun, jika turun selama 5 (value) iterasi maka program akan berhenti

Kesimpulan:

- Model neural network yang kami buat masih mengalami exploding gradient dimana terdapat kesalahan pada permodelannya sehingga ada saatnya saat iterasi, A2 hasilnya sama seperti iterasi sebelumnya. Beberapa kemungkinan kesalahan tersebut mungkin terjadi pada saat:
    - Backward Prop dikarenakan jika divider dW = 0, maka semua index pada array W terupdate menjadi inf.
    - Pada saat penggunaan activation function Soft Max dari pytorch, hasil yang diinginkan oleh kelompok kami dari activation function tersebut adalah 0<x<1 dimana x merupakan index dari setiap element di A2 . Namun seringkali Soft Max dari pytorch berisi x>1 sehingga A2 menjadi invalid
    - Pada saat menghitung Z, mendapatkan value yang tidak valid dikarenakan point 1 dan point 2

REFRENSI TUBES:

- https://www.kaggle.com/code/wwsalmon/simple-mnist-nn-from-scratch-numpy-no-tf-keras/notebook
- https://www.youtube.com/watch?v=w8yWXqWQYmU&t=915s
- https://www.youtube.com/watch?v=c36lUUr864M&t=2s