Submission to SuperAGI

Amit Prasad (2020SIY7564)

November 2023

1 Question 1

$w_{new_0} = w_{old_0}$
$w_{new_1} = w_{old_1}$
.
.
.
$w_{new_n} = w_{old_n}$
$w_{new_{n+1}} = 0$
since the last attribute is redundant, the weight associated with it will be 0.

2 Question 2

a. We have too little data as variance needs to be provided to know the spread of the distributions for a t-test.

3 Question 3

The cost of the mentioned computation is the cost of calculating the following update

$$w^{t+1} \leftarrow w^t + \sum_{i}^{m} (y_i - h(x_i))x_i \tag{1}$$

where m is the number of examples and $h(x_i)$ is the value of the sigmoid function.

3.1 Cost of $(y_i - h(x_i))$

$O(n)$ as there are n subtractions. For simplicity, we do not factor in the cost encountered to find the value of exponentials.

3.2 Cost of $(y_i - h(x_i)) * x_i$

this comprises of $O(n)$ multiplications.

3.3 Cost of $\sum_{i=1}^{m}(y_i - h(x_i)) * x_i$

this comprises of $O(m)$ additions, each of which has a cost $O(n)$ as there are n elements. So $O(mn)$.

3.4 Cost of update

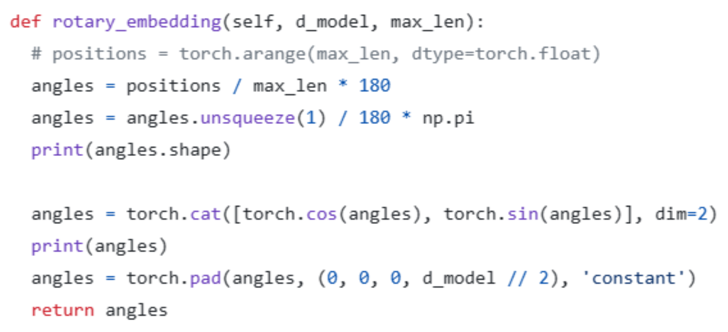
it requires one addition and one assignment of cost $O(n)$.

3.5 Total cost

$O(mn + n) = O(mn)$. Considering sparsity, this cost is $O(km + k)$.

3.6 Task 2 Rotary embedding

We added the following change for replacing positional embedding with rotary positional embedding. Output can be seen on github. Changes made are in the model.py file uploaded on github.



```
def rotary_embedding(self, d_model, max_len):
  # positions = torch.arange(max_len, dtype=torch.float)
  angles = positions / max_len * 180
  angles = angles.unsqueeze(1) / 180 * np.pi
  print(angles.shape)

  angles = torch.cat([torch.cos(angles), torch.sin(angles)], dim=2)
  print(angles)
  angles = torch.pad(angles, (0, 0, 0, d_model // 2), 'constant')
  return angles
```