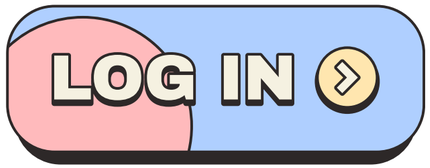
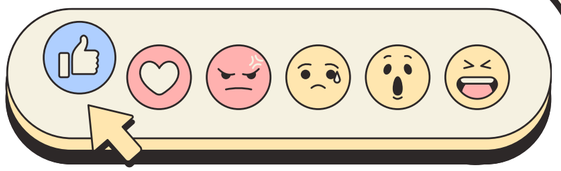
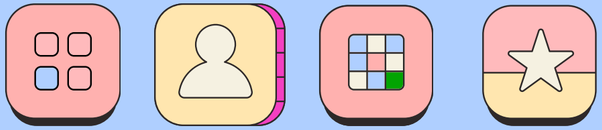


ATTENTION!

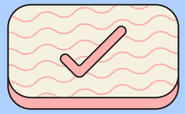


सस्ताGPT

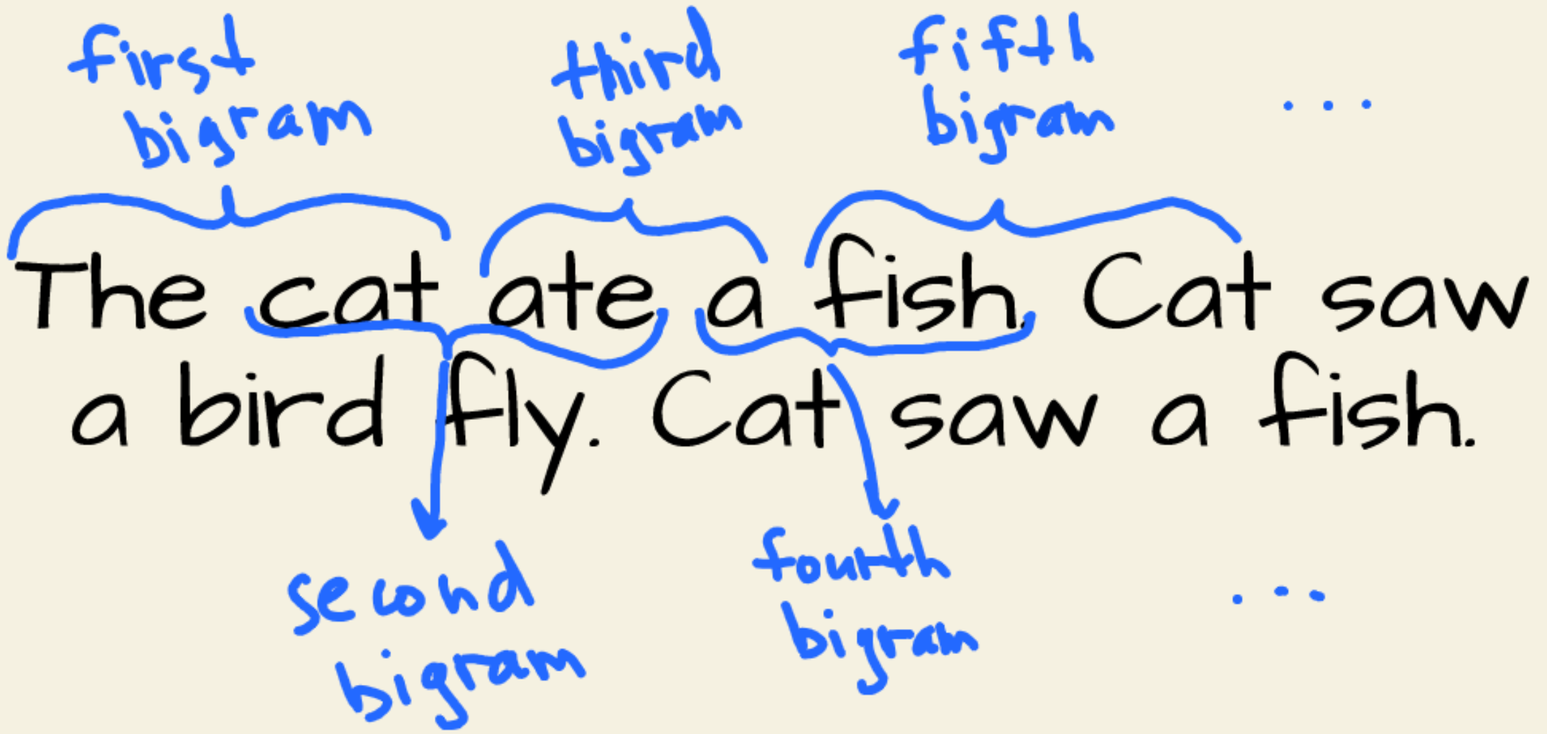
AMAN K. FOUJDAR



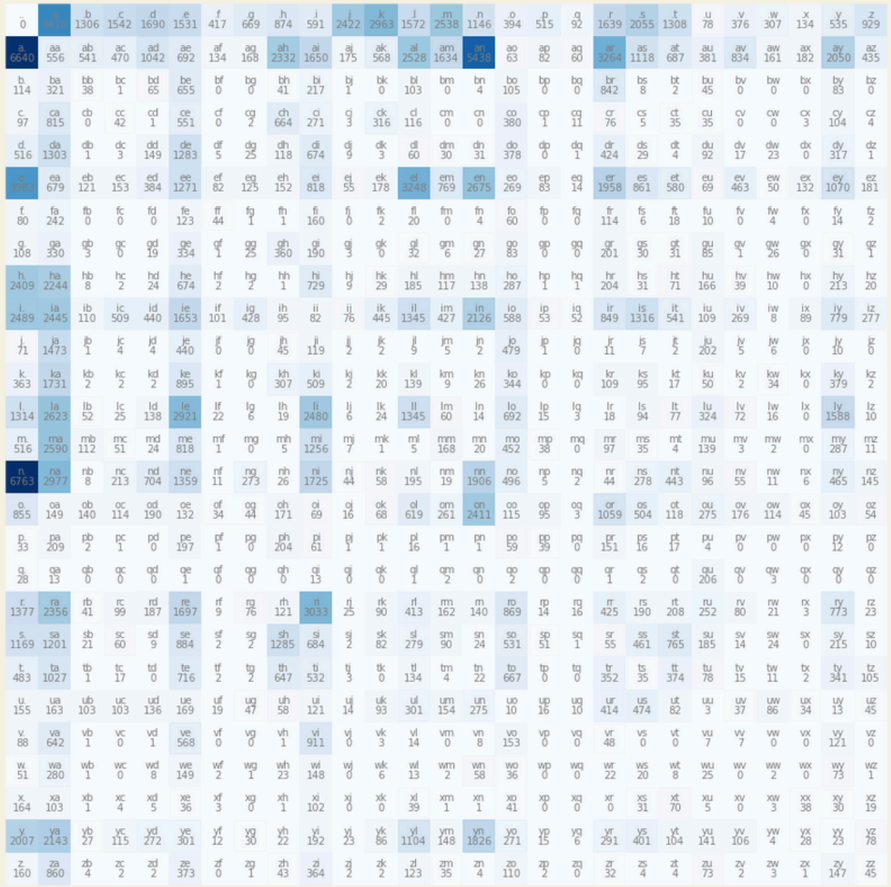
IML: PROF SANDEEP JUNEJA



BIGRAM MODELS



BIGRAM MODELS



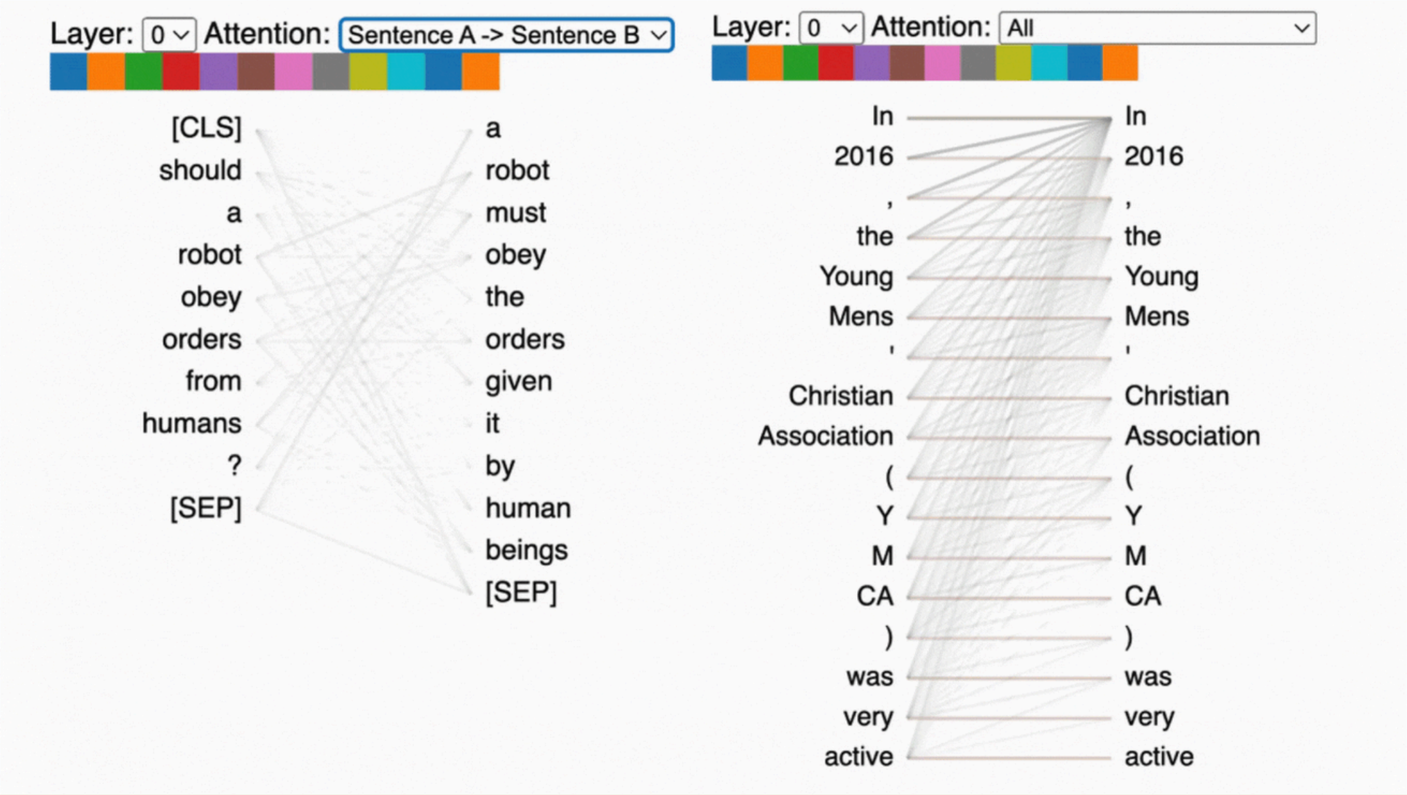
```
for w in words:
  chs = ['.'] + list(w) + ['.']
  for ch1, ch2 in zip(chs, chs[1:]):
    ix1 = stoi[ch1]
    ix2 = stoi[ch2]
    N[ix1, ix2] += 1
```

BIGRAM MODELS

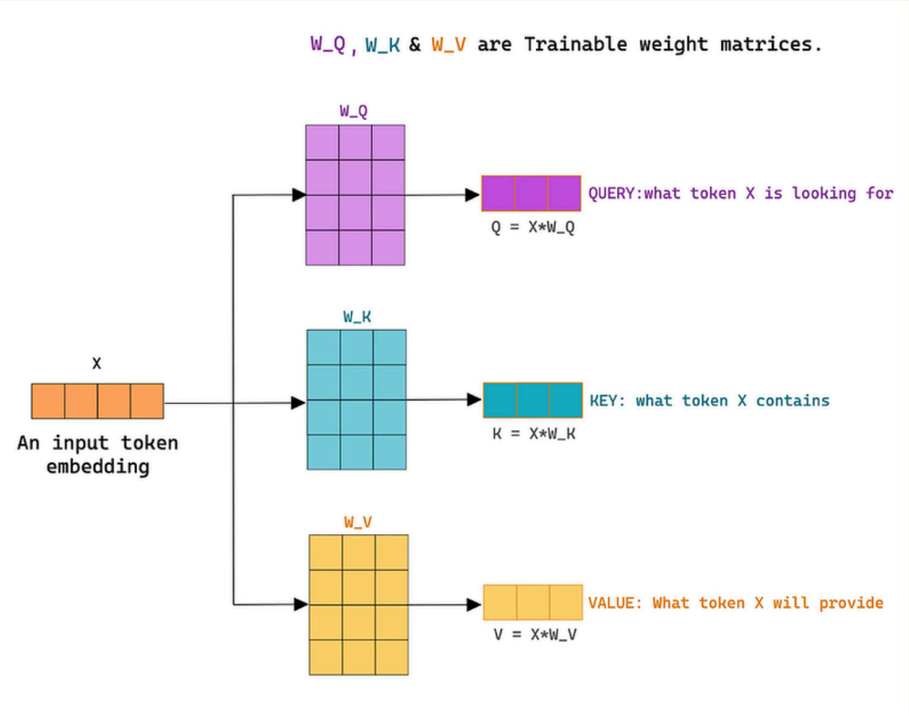
```
mor.
axx.
minaymoryles.
kondlaisah.
anchthizarie.
```

 # DRUM ROLLS 

ATTENTION

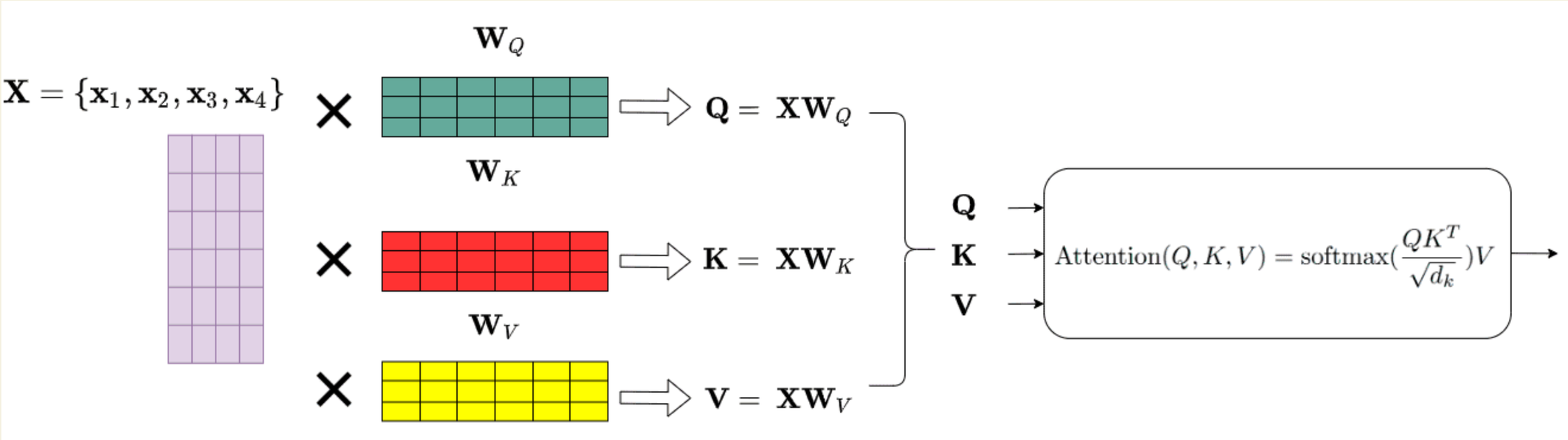

Layer: 0 Attention: Sentence A -> Sentence B
[CLS] should a robot obey orders from humans ? [SEP]
a robot must obey the orders given it by human beings [SEP]
Layer: 0 Attention: All
In 2016 , the Young Mens ' Christian Association (Y M CA) was very active
In 2016 , the Young Mens ' Christian Association (Y M CA) was very active

SELF ATTENTION

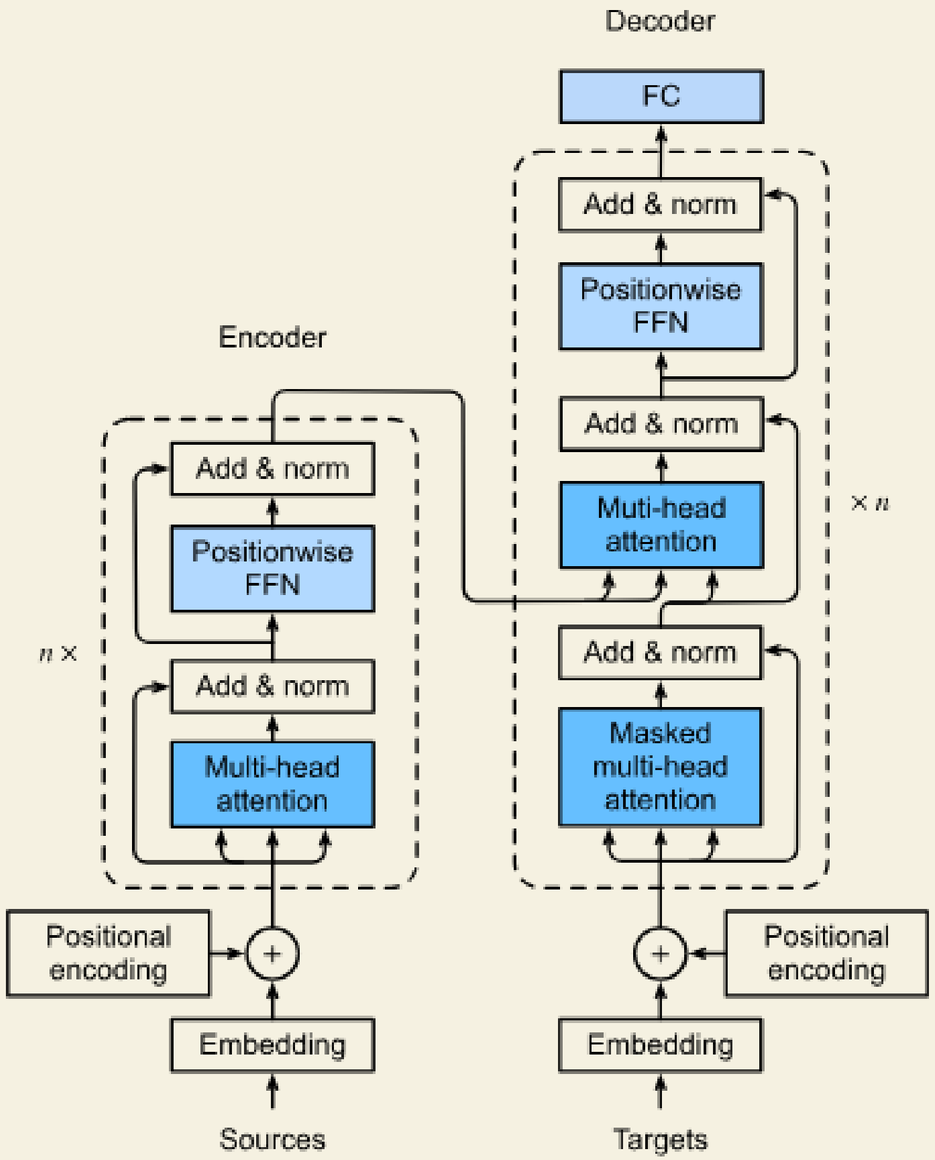



```python
#singkle head of attention
class Head(nn.Module):
    def __init__(self, head_dim):
        super().__init__()
        self.key = nn.Linear(EMBEDDING_DIM, head_dim, bias=False)
        self.query = nn.Linear(EMBEDDING_DIM, head_dim, bias=False)
        self.value = nn.Linear(EMBEDDING_DIM, head_dim, bias=False)
        self.register_buffer('tril', torch.tril(torch.ones(CONTEXT_WINDOW, CONTEXT_WINDOW)))
        self.dropout = nn.Dropout(DROPOUT_RATE)

    def forward(self, x):
        #here B-> batch, T -> time-step, C -> channels
        B, T, C = x.shape
        k, q, v = self.key(x), self.query(x), self.value(x) # dimesnsion of this is (B,T,HeadSize)

        # compute attention scores ("affinities")
        attention_scores = (q @ k.transpose(-2, -1)) * k.shape[-1]**-0.5
        attention_scores = attention_scores.masked_fill(self.tril[:T, :T] == 0, float('-inf'))
        attention_probs = F.softmax(attention_scores, dim=-1)
        attention_probs = self.dropout(attention_probs)
        return attention_probs @ v
```

SELF ATTENTION



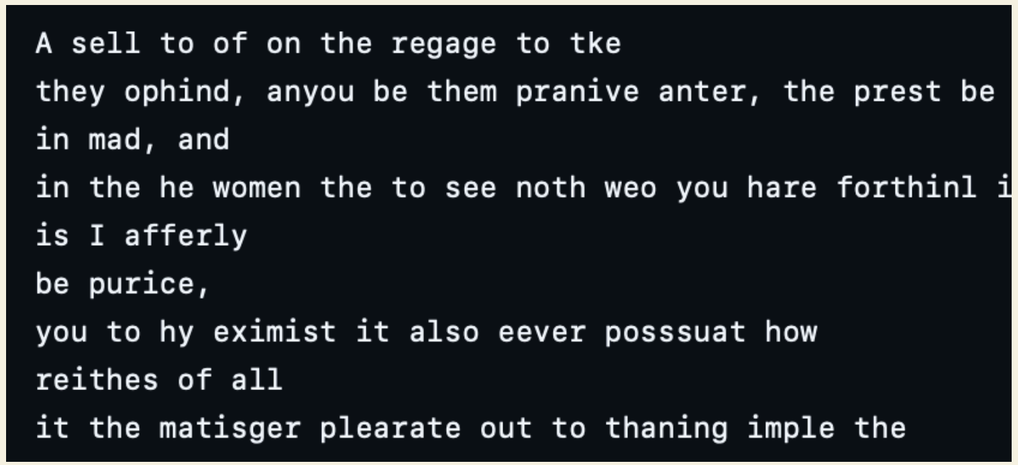
TRANSFORMER



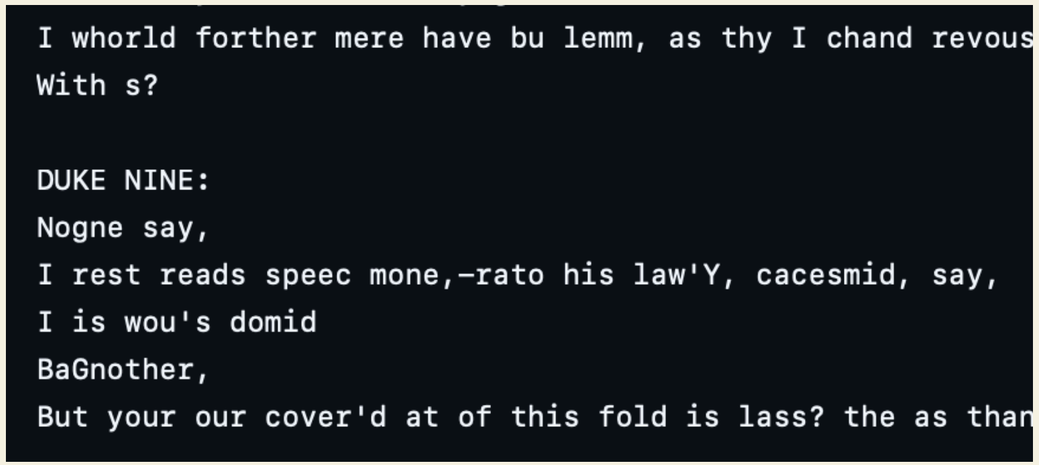

```python
class Block(nn.Module):
    def __init__(self, EMBEDDING_DIM, HEADS):
        super().__init__()
        head_size = EMBEDDING_DIM // HEADS
        self.mha = MultiHeadAttention(HEADS, head_size)
        self.ffwd = FeedForward(EMBEDDING_DIM)
        self.ln1 = nn.LayerNorm(EMBEDDING_DIM)
        self.ln2 = nn.LayerNorm(EMBEDDING_DIM)

    def forward(self, x):
        x = x + self.mha(self.ln1(x))
        x = x + self.ffwd(self.ln2(x))
        return x
```

File display

सस्ताGPT 10M



सस्ताSTOIC



सस्ताSHAKESPEARE

¬DEEPSEEK?

```
10.713691 M parameters
step 0: train loss 4.4460, val loss 4.4495
step 500: train loss 2.1563, val loss 2.2279
step 1000: train loss 2.0441, val loss 2.1356
step 1500: train loss 1.9811, val loss 2.0880
step 2000: train loss 1.9382, val loss 2.0526
step 2500: train loss 1.8820, val loss 2.0114
step 3000: train loss 1.8584, val loss 1.9786
step 3500: train loss 1.8271, val loss 1.9861
step 4000: train loss 1.8289, val loss 1.9594
step 4500: train loss 1.7941, val loss 1.9273
step 4999: train loss 1.7831, val loss 1.9317
```

File display

THANK YOU

LOG OUT