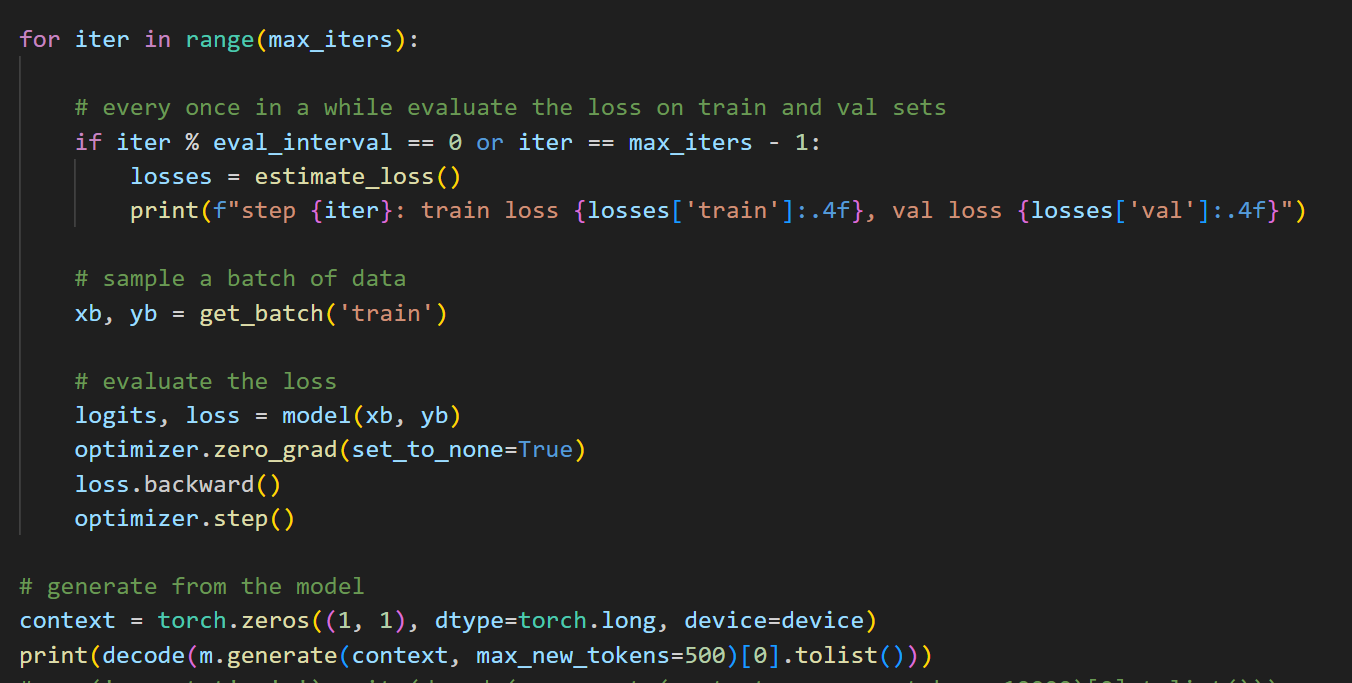
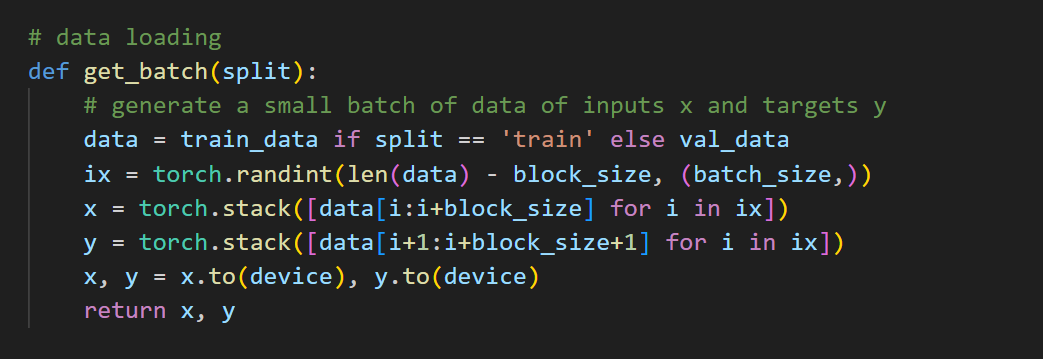


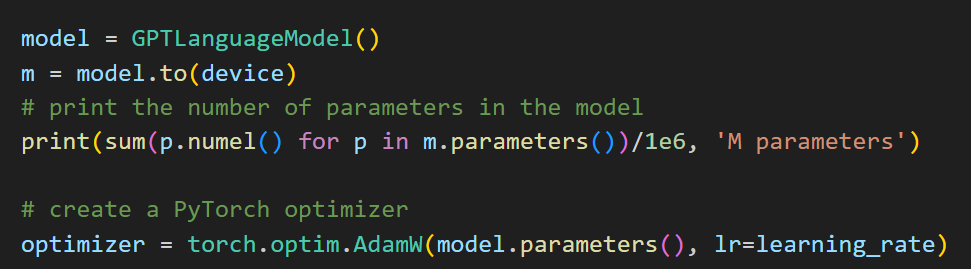
* Here are the hyperparameters (parameters which can be edited not learned)



* Intital character given to model is 0
* Max 500 characters to be generated
* get\_batch called (explained below)

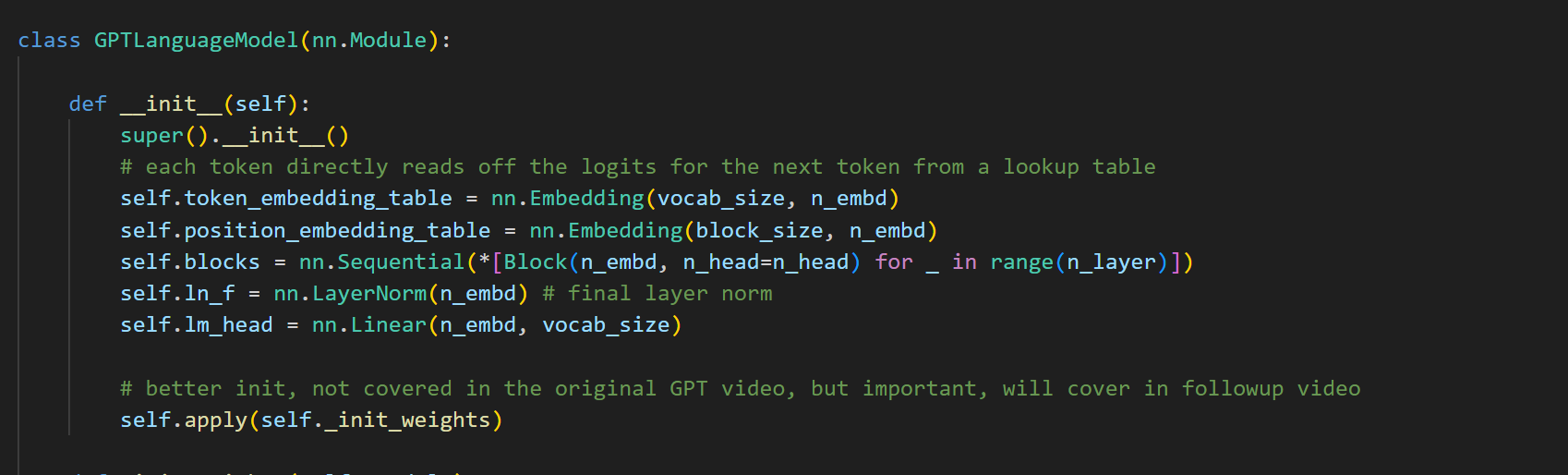


* Generates random indexes from which block size chunk of tokens can be samples(batch number times ix)



* Printing number of parameters
* Torch.numel returns the number of elements in input
* Used adam optimizer (read about various optimizer)

CLASS GPTLanguadeModel

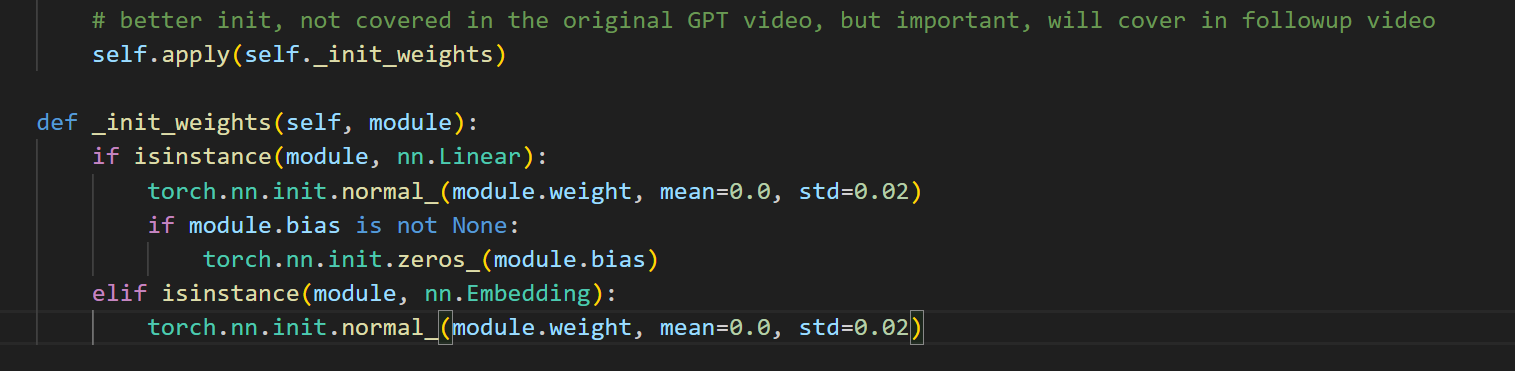


* Why use super?

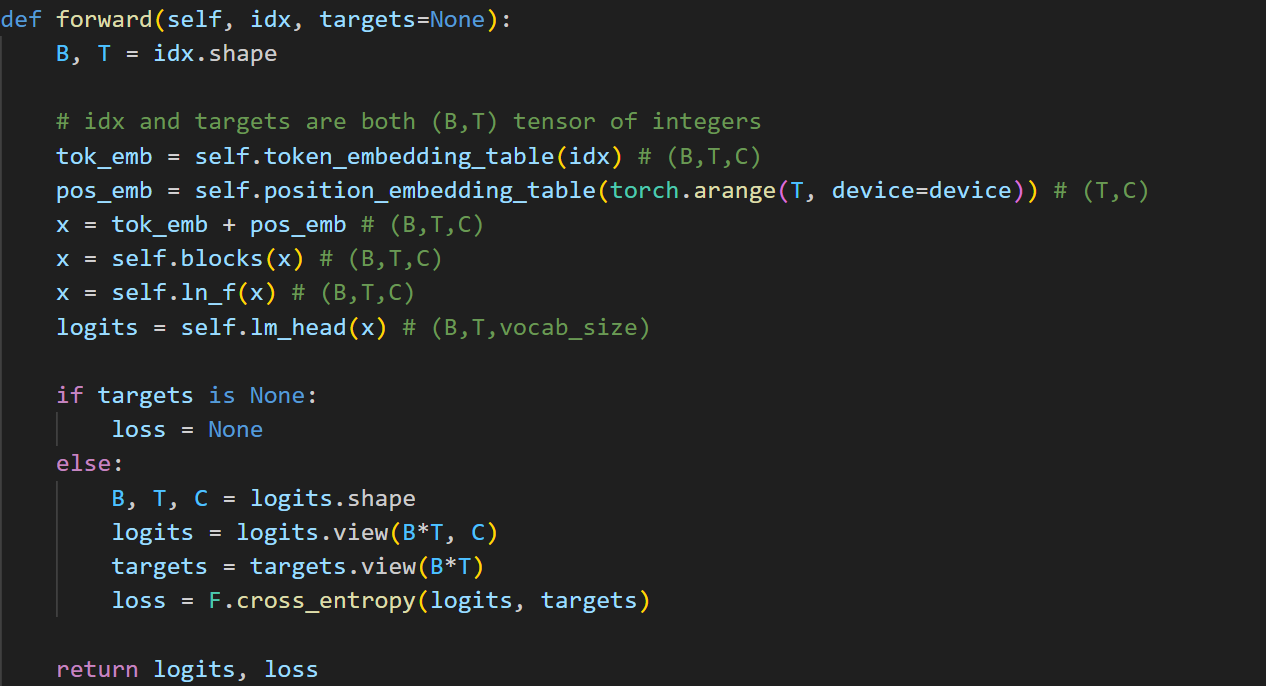
[Exploring the Power of Python super() Function (hubspot.com)](https://blog.hubspot.com/website/python-super#:~:text=The%20super()%20function%20is%20often%20used%20with%20the%20__,attributes%20of%20the%20parent%20class.&text=Copy-,In%20this%20example%2C%20we%20have%20a%20Parent%20class%20and%20a,the%20name%20and%20age%20attributes.)

[class - What does 'super().\_\_init\_\_()' mean in python 3.x? - Stack Overflow](https://stackoverflow.com/questions/19530508/what-does-super-init-mean-in-python-3-x)

* Token\_embedding gives a embedding for each token (65 in our case)
* Position encoding , encodes every position in a block (sequence length) to a position encoding
* Blocks contain attention blocks
* ln\_f just applies the layer normalization
* lm\_head generated final vocal\_size prob values



* Initialized the weights

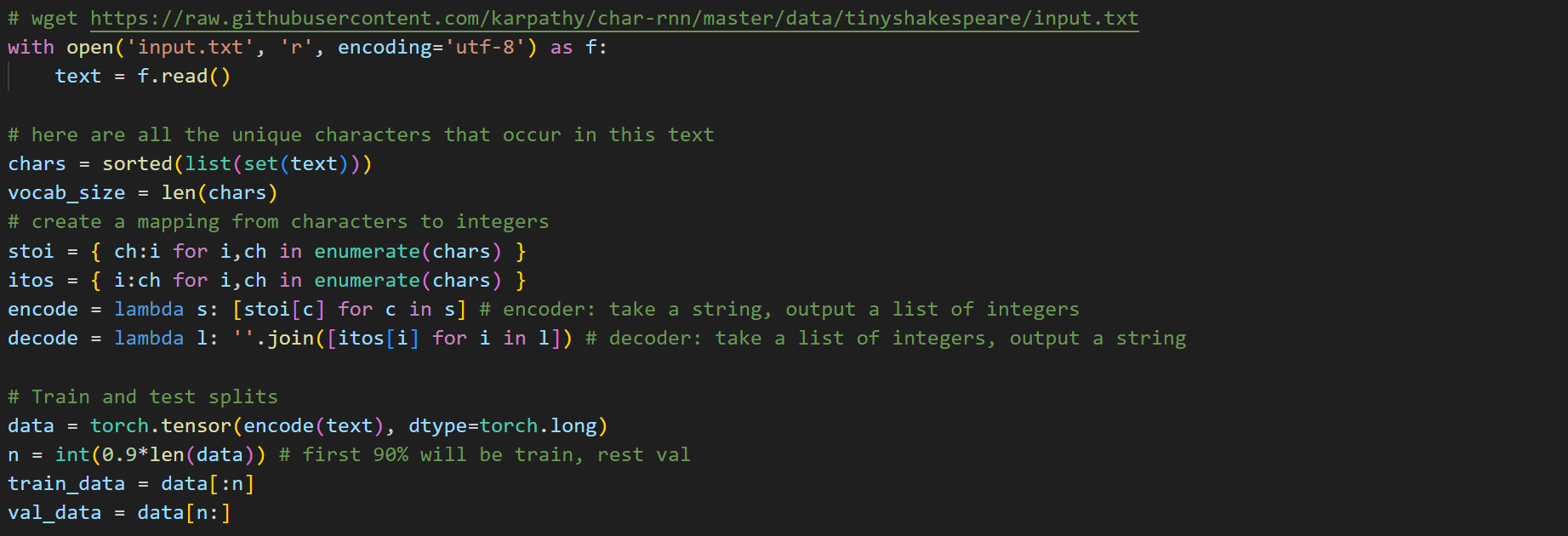


* Generates tok\_emb and pos\_emb for whole sequence (BXTXC)
* Logits are the final probabilities
* Cross entropy can take unnormalized logits

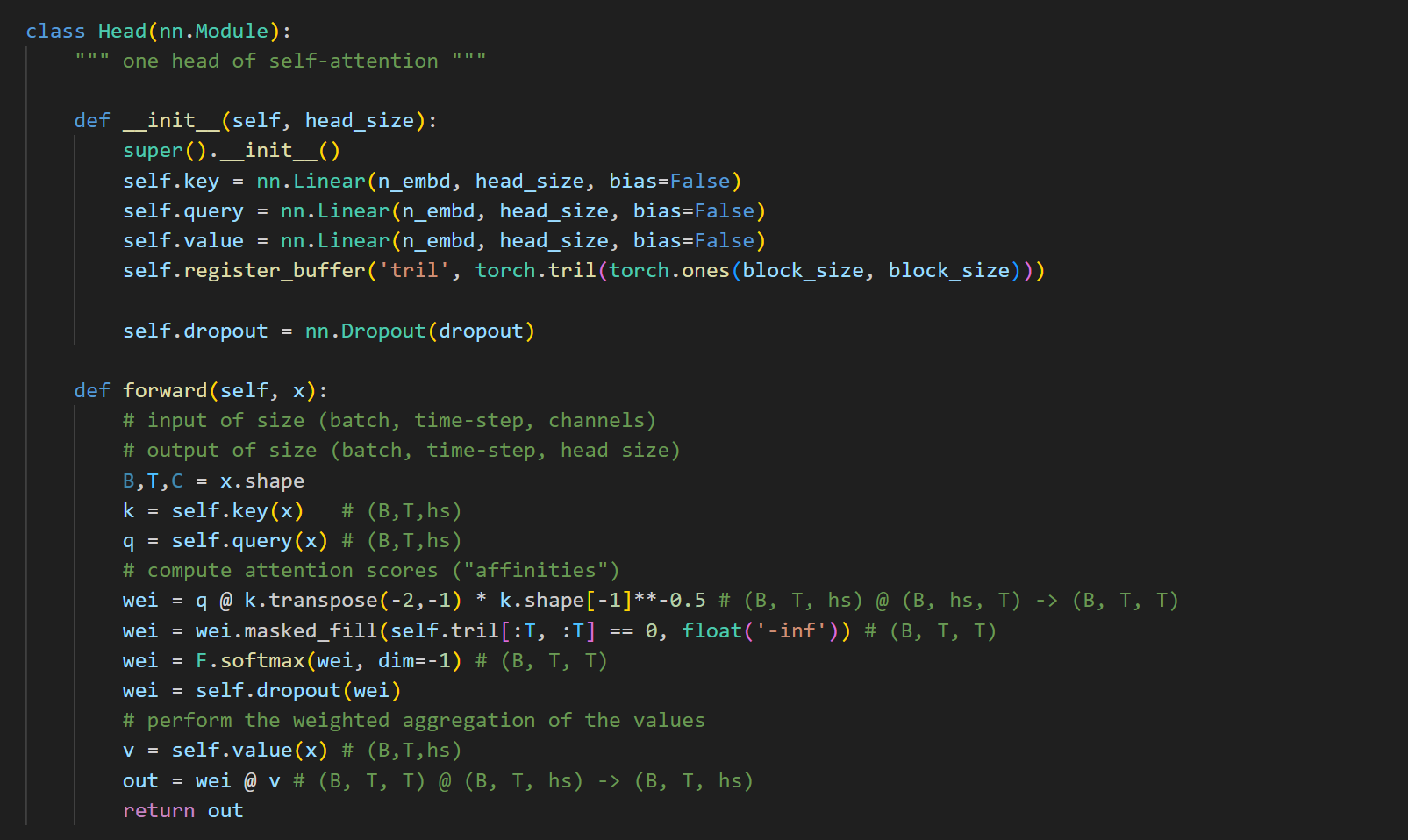


* Crop idx to block size as model trained to only produces sequences based on block size at most
* Get the logits of only T (using the whole block size as context)
* Softmax it and multimonial picking according to probabilities
* Add idx\_next to idx so that it can be used for context in next token generation

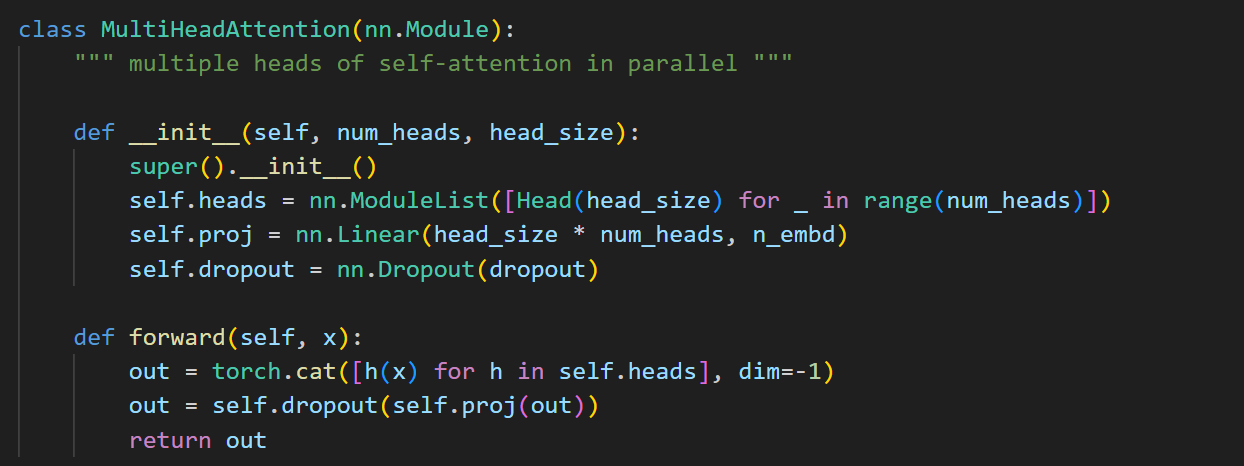
Tokenizing the characters and splitting them into test/val



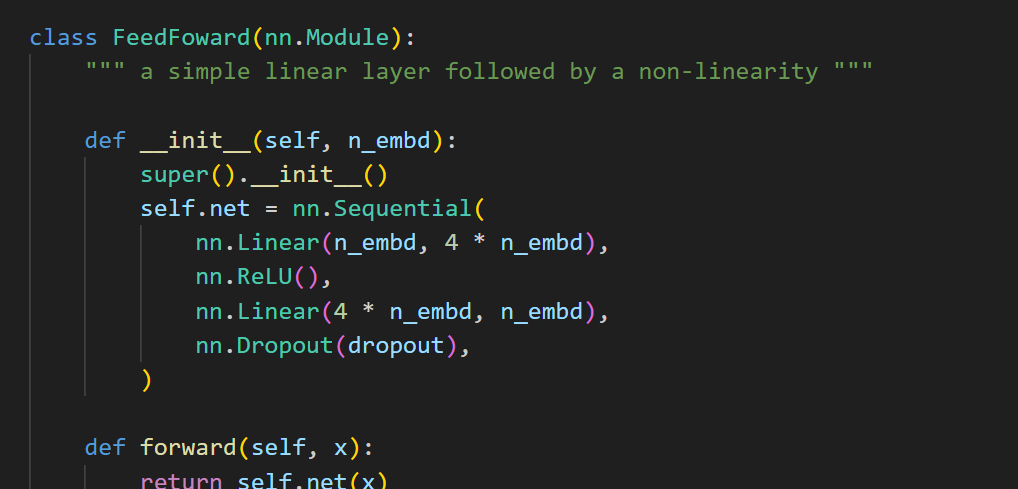
Head Node



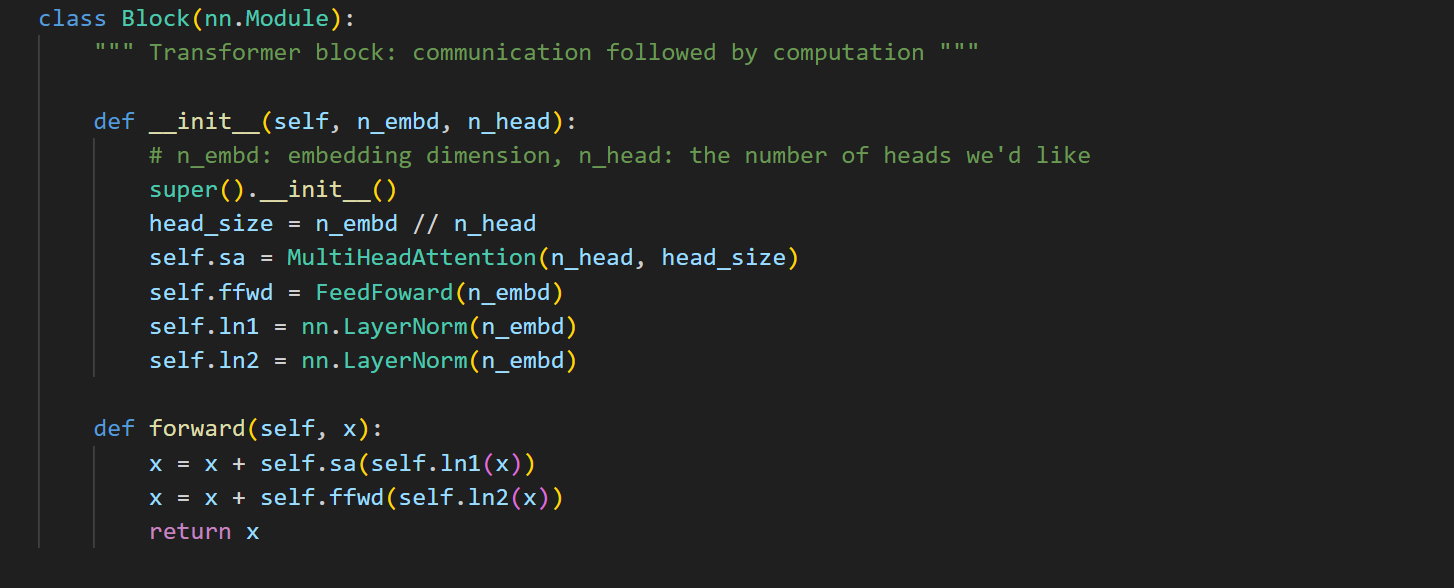
* Head generated key,query and value (thus self attention)
* Trill is a lower triangular matrix
* Q.K to find affinities of current token with previous tokens, it is scaled by sqrt head size to make variance wei similar to k and q
* We mask the upper triangular as -inf so that current token cannot talk to future token
* We then multiply with value to aggregate



* Multiple heads in parallel
* Proj converts head output back to emb\_size



* Feedforward network of the block



* Block contains Multiheads,feedfowrd, and then normalization
* X+ is the residual addition (done so that gradient can directly travel to input in deep networks)
* Layer normalization [Build Better Deep Learning Models with Batch and Layer Normalization | Pinecone](https://www.pinecone.io/learn/batch-layer-normalization/)

READ TRANSFORMER PAPER