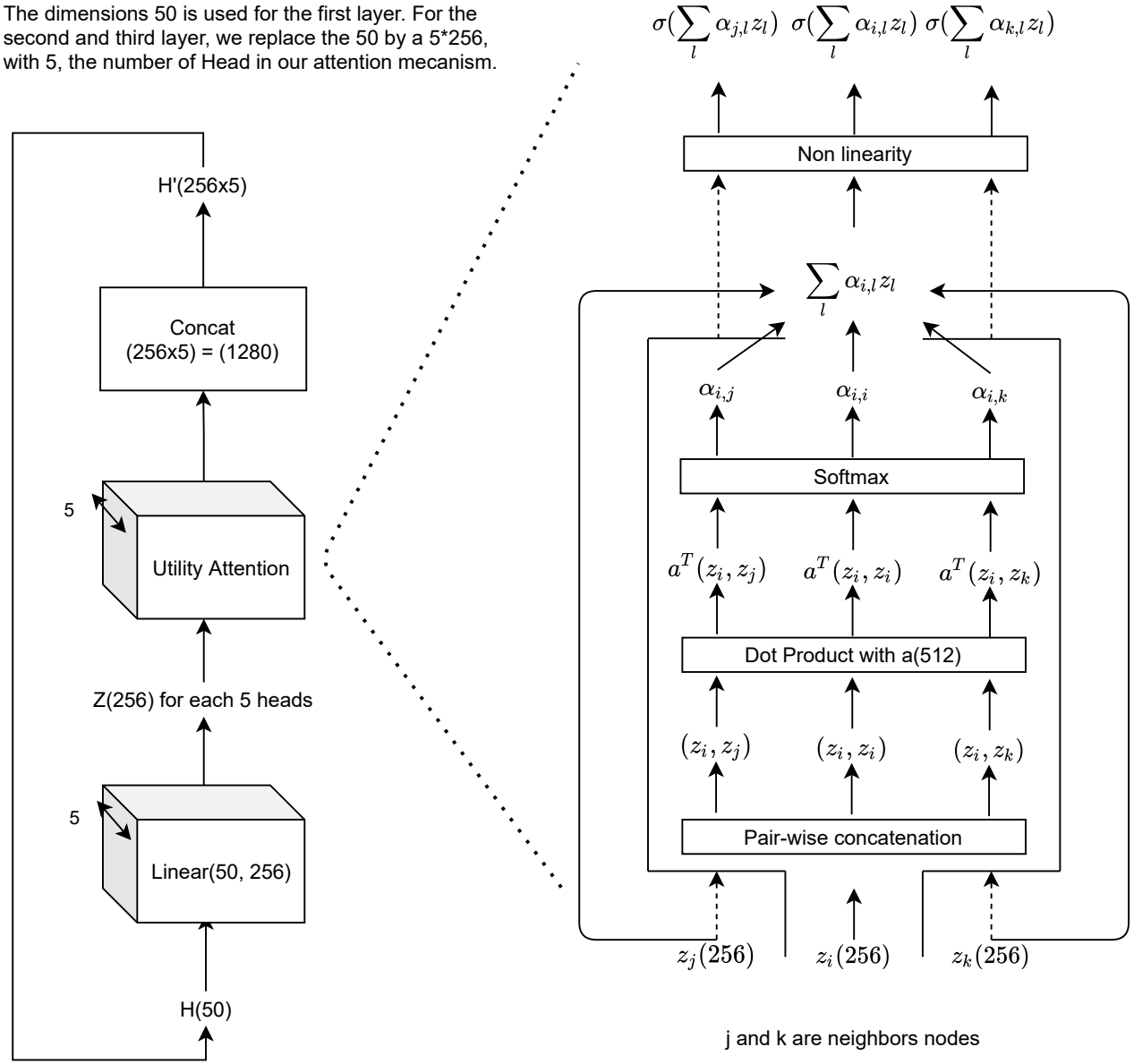
Utility Attention

The utility attention is used in the Attention graph Network.

The dimensions 50 is used for the first layer. For the second and third layer, we replace the 50 by a 5*256, with 5, the number of Head in our attention mecanism.



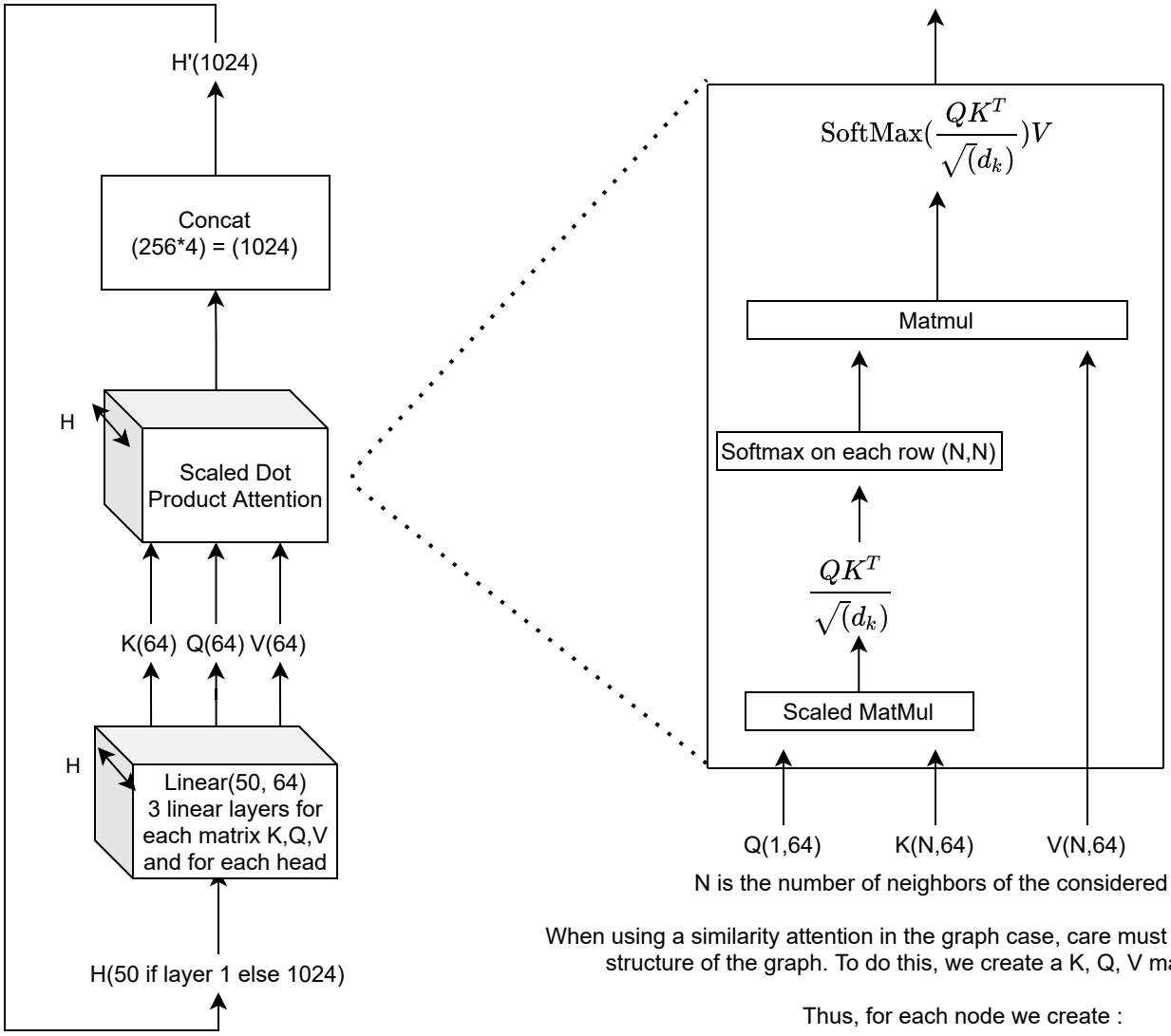
j and k are neighbors nodes

The management of the parallelisation of calculations between the different neighbors is optimised by the DGL library. The embedding of each node is computed before moving to the next layer.

Similarity Attention

The Similarity Attention was introduced in the paper Attention is all you need.

The dimensions 50 is used for the first layer. For the second and third layer, we replace the 50 by (256*4) = (1024). If we use 64 as the embedding of the Key, Query and value matrix, the number of heads H must verify : H*64=256. So H = 4.



N is the number of neighbors of the considered node.

When using a similarity attention in the graph case, care must be taken to keep the structure of the graph. To do this, we create a K, Q, V matrix per node.

Thus, for each node we create :

- The matrix K which contains the embedding of the keys of all the neighboring nodes.
- the Q matrix which contains the embedding of the query of the considered node
- the V matrix which contains the embedding of the values of the neighboring nodes

Architecture

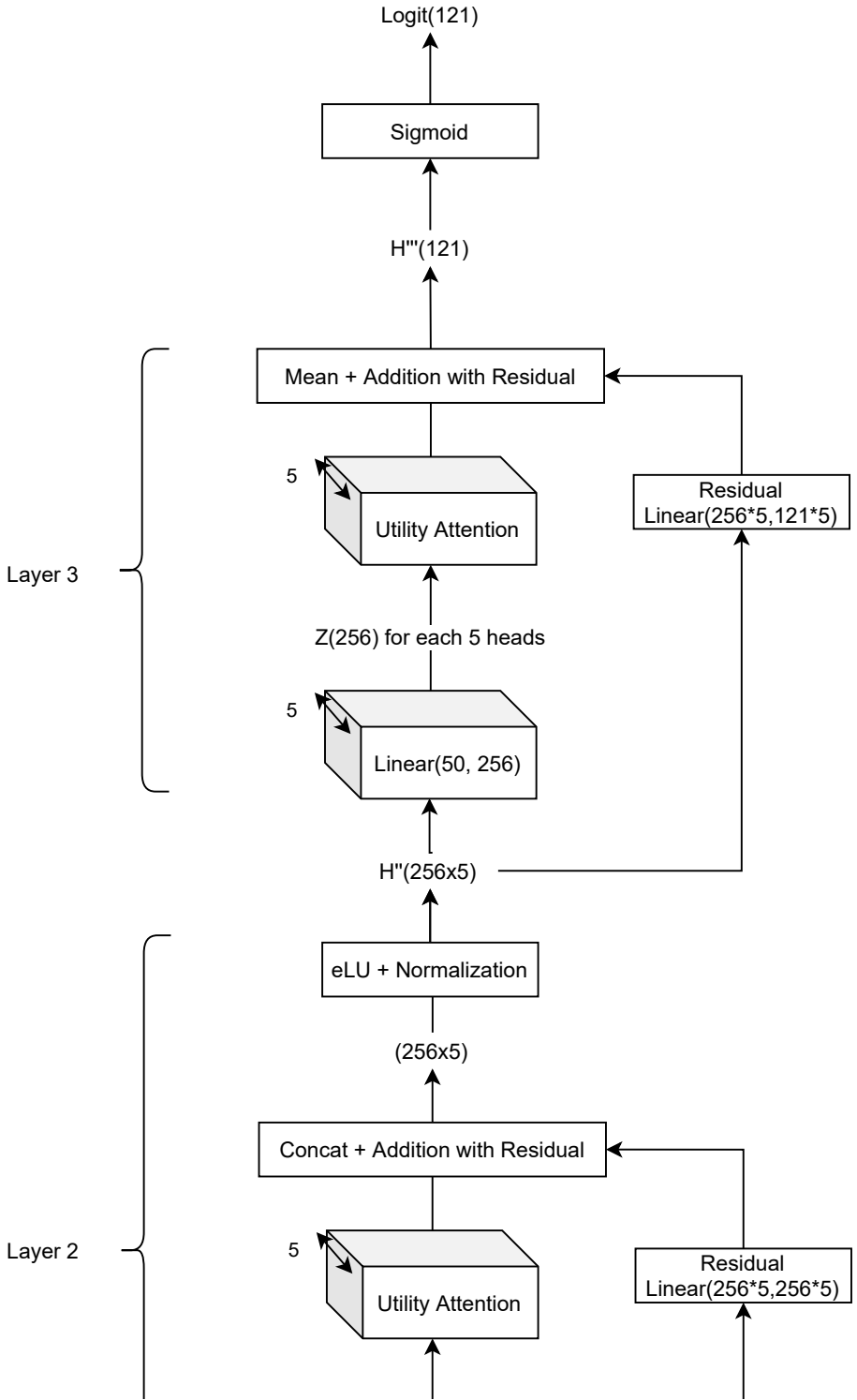
Here is a schema of the architecture used in our implementation of the Attention Graph Network.

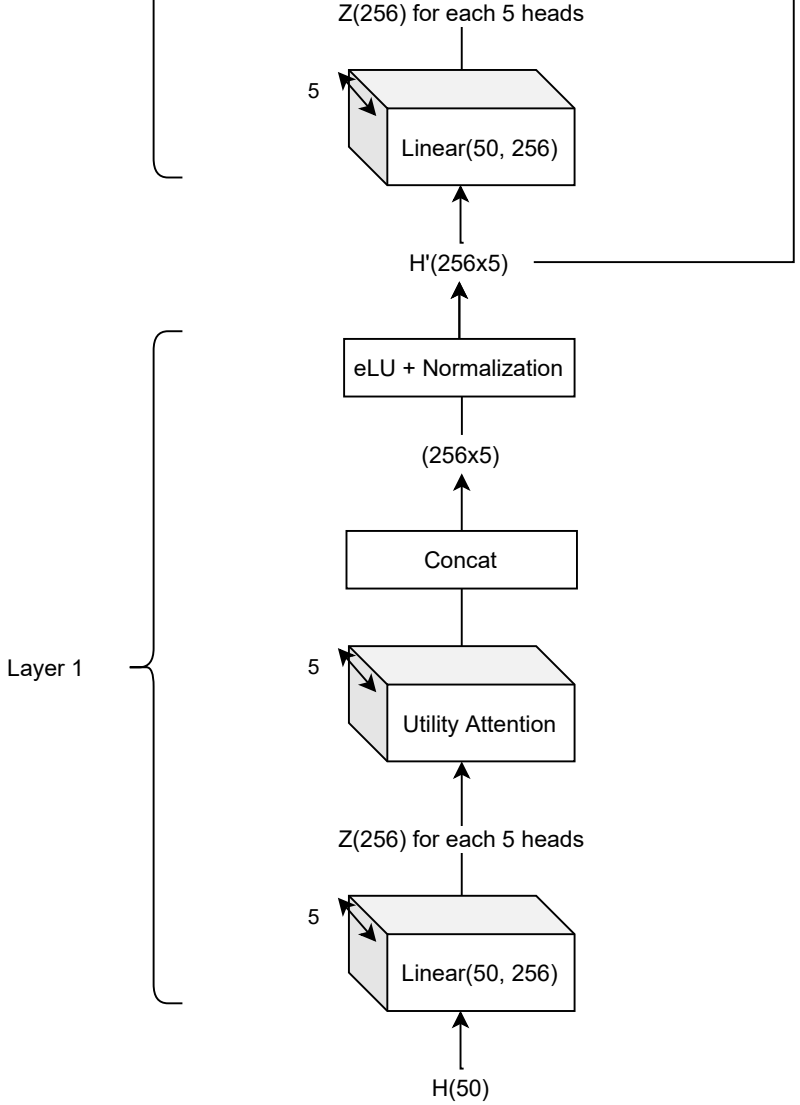
Our architecture is a slight modification of the one proposed in the paper for the Inductive Learning on the PPI dataset. The paper proposes a number of heads of 4, 4, 6 for layers 1, 2 and 3. But we take 5 heads for each layer in order to simplify a bit. As in the paper, we do not use dropout nor do we use regularization or data augmentation. But we found out that adding a normalization to mean 0 and standard deviation 1 at the end of each layer helps tremendously with the task.

The notation Linear(Entry dim, Output Dim) gives the size of the embeddings passing through it as in PyTorch.

The Layer 3 is identical to the layer 2 with just three differences:

- We do not concatenate, but we mean on the head dimension.
- There is no non-linearity such as the eLU in layer 2.
- The embedding used in the attention is not of size 256 but of size 121 which is the number of classes.





We begin with a graph with an embedding on each node containing 50 features.
We do not write the number of node here because it varies from graph to graph. But we make the calculation in parallel on each node before going to the next layer.