SCIENCE
PASSION
TECHNOLOGY

Designing approximated Machine Learning models in Python for Homomorphic evaluation.

Sarah Hartinger Aikata

Bachelor Thesis

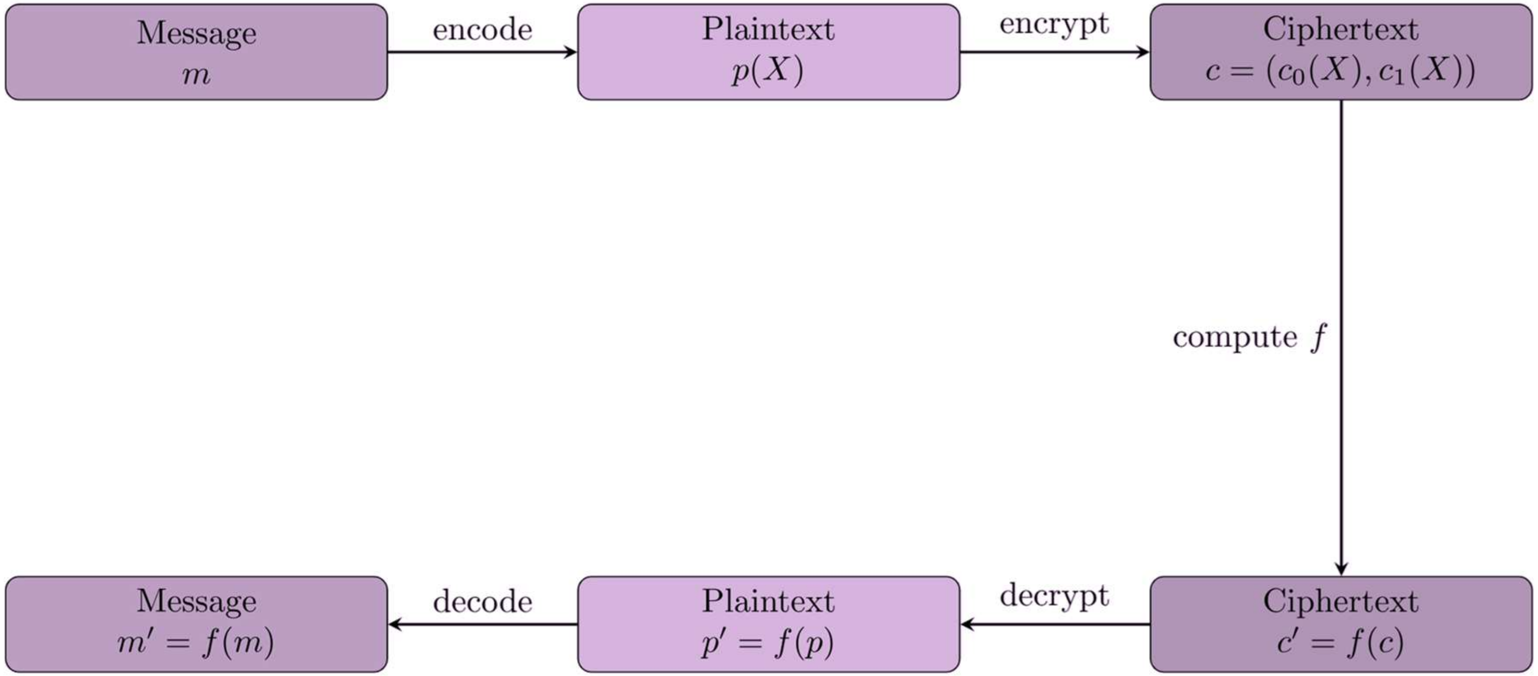
Motivation

- Privacy-Preserving Machine Learning
- Homomorphic Encryption
- Addition, Multiplication, Rotation
- Softmax function

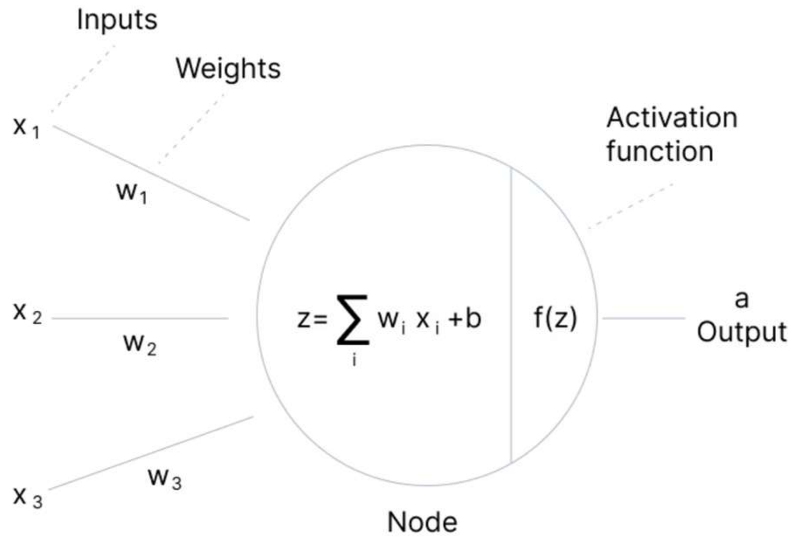
Implement an approximation of a vision transformer by Tin Nguyen, trained on the CIFAR 10 dataset.

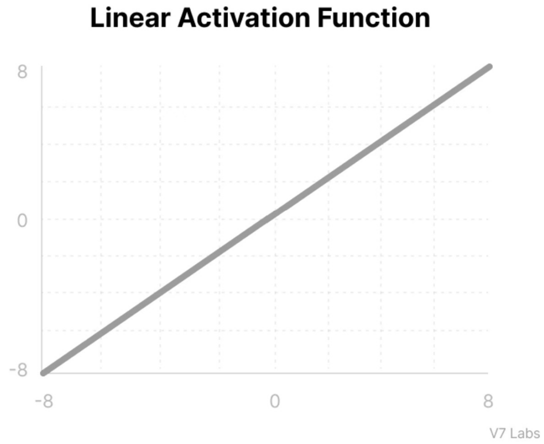
Main Goal

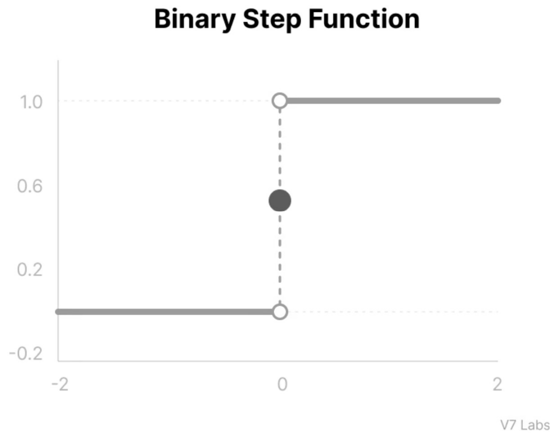
Cheon-Kim-Kim-Song Scheme



Artificial Neurons/Nodes

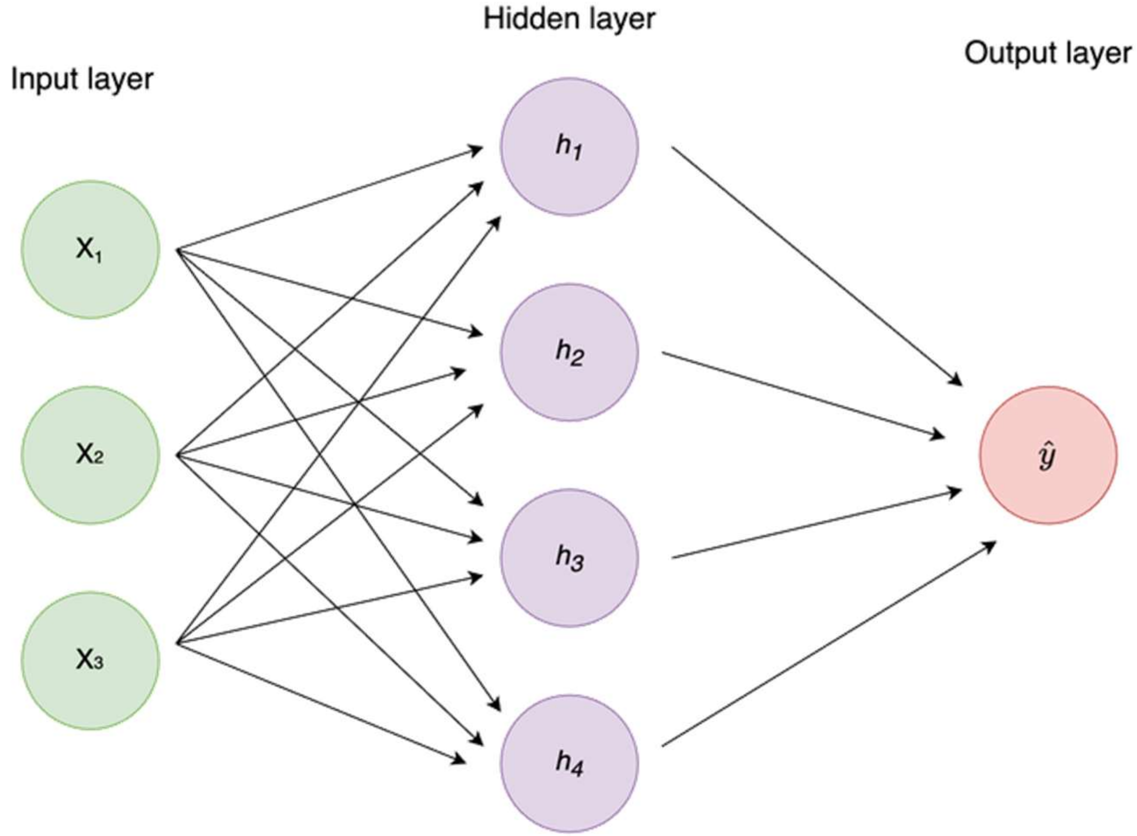


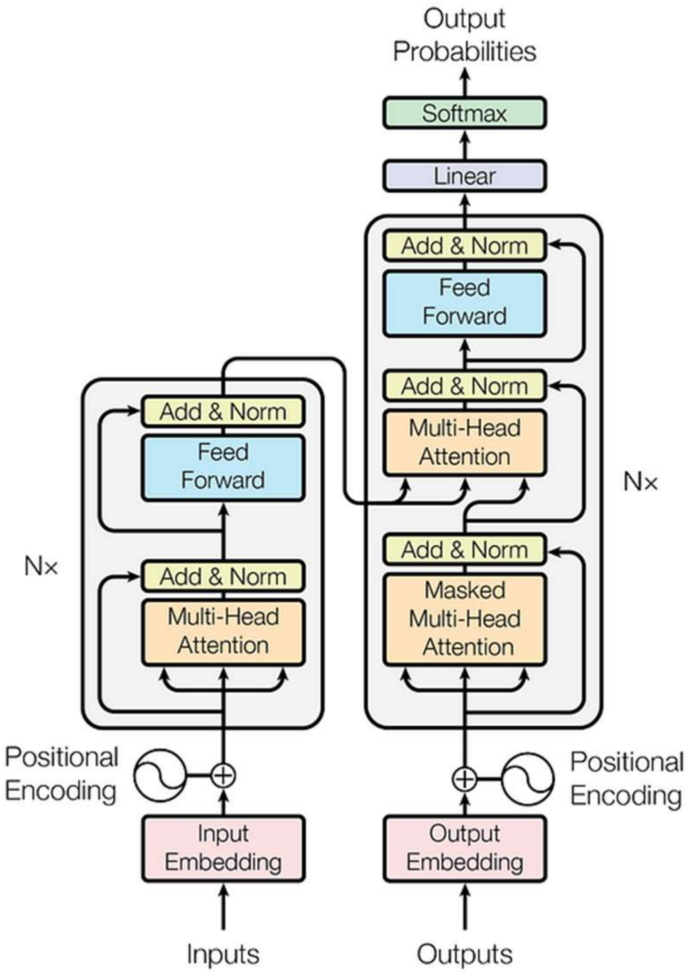




Artificial Neuronal Network

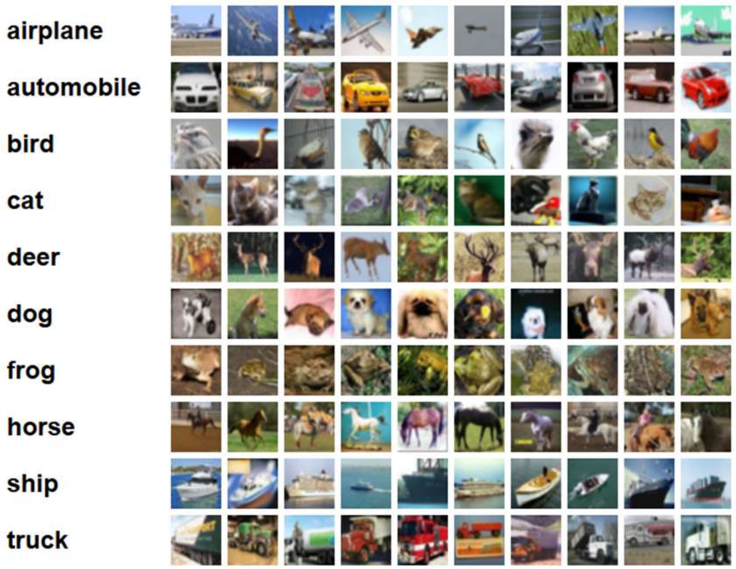
- Feed forward
- Recurrent





Transformer Model

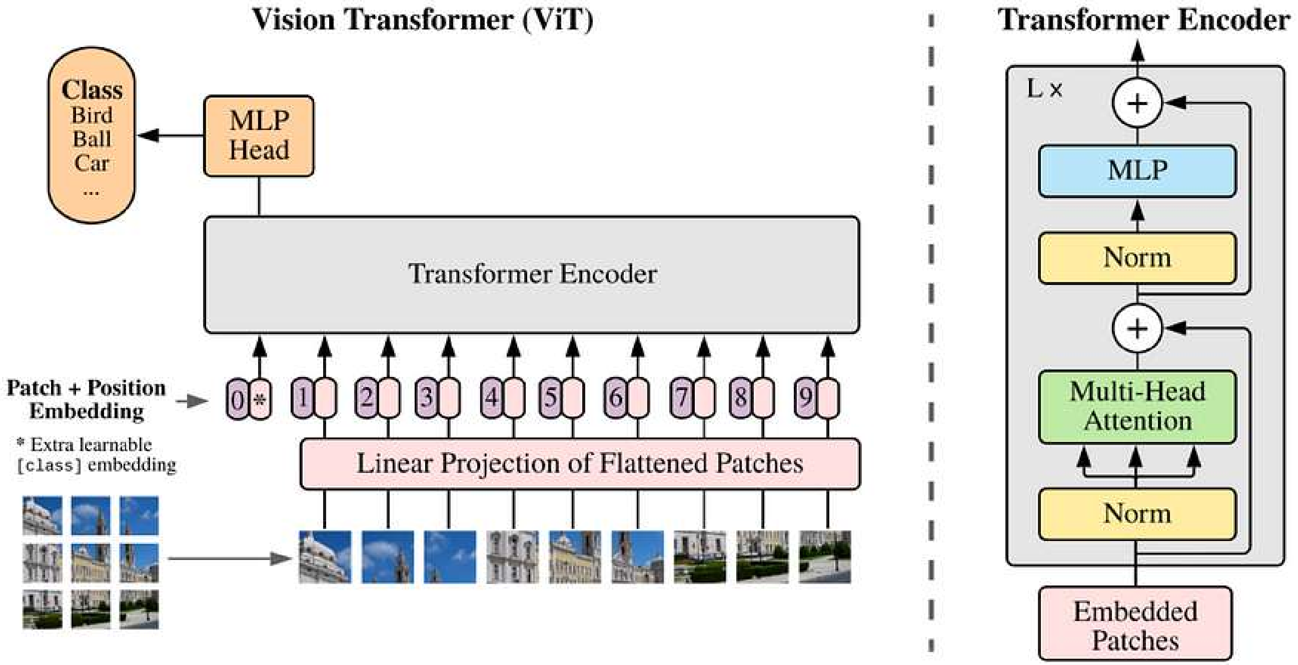
- Attention is all you need – Google 2017
- Feedforward
- Attention



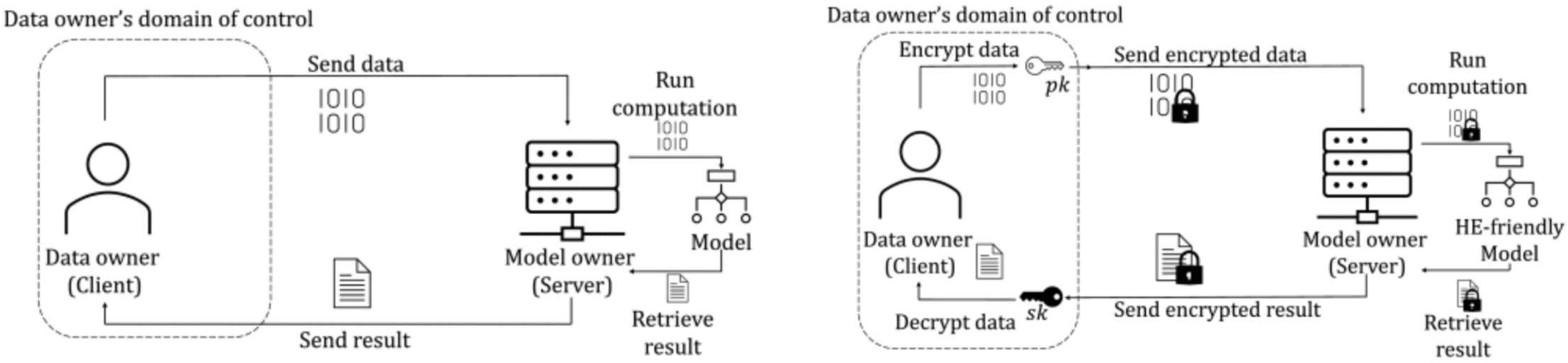
Vision Transformer

- Tin Nguyen – „Vision Transformer form Scratch"
- Inspired by "An Image is worth 16x16 Words: Transformers for Image Recognition at Scale"
- CIFAR-10 dataset

Vision Transformer Model



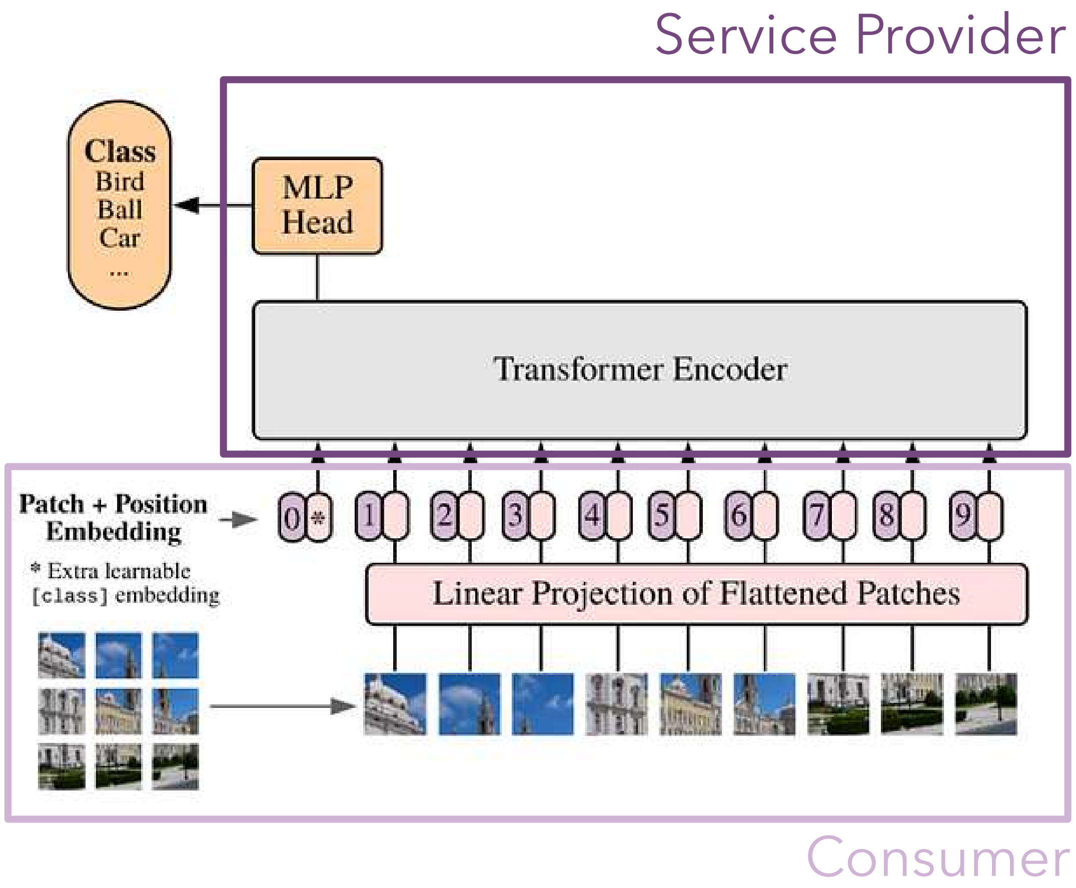
Machine Learning



(a) Without privacy preservation (b) With privacy preservation

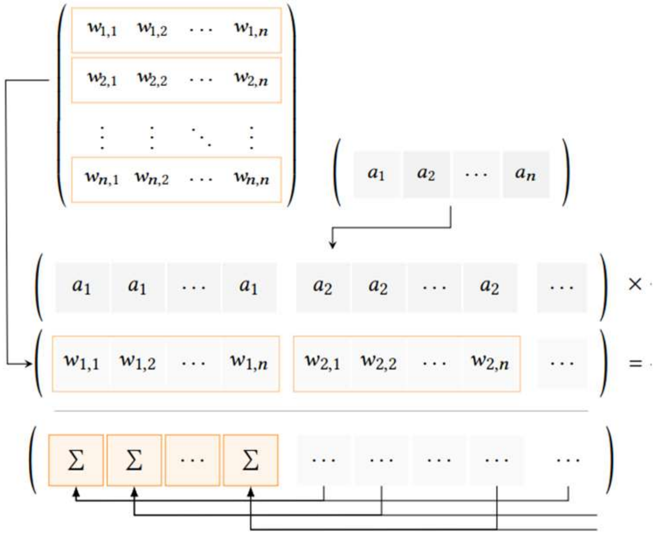
HE Vit Setting

- **Input:** Encrypted Input Embeddings
- **Output:** Encrypted CLS Token

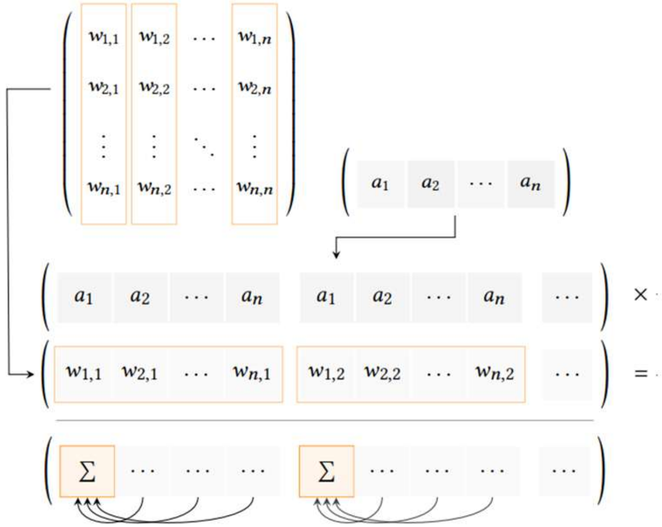


Multiplication

Expanded Vector x Row-major Matrix

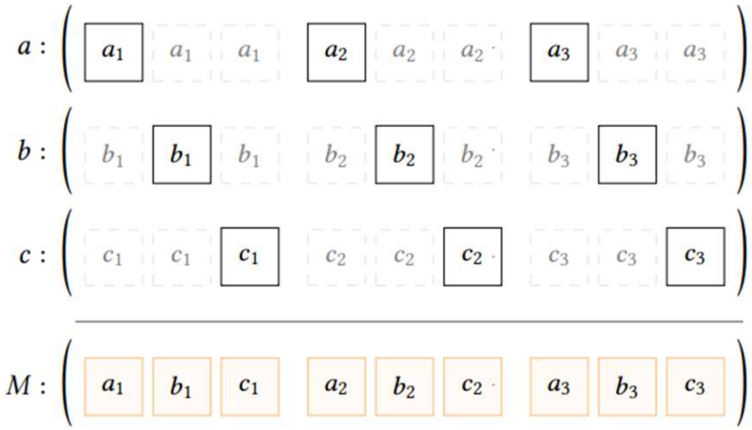


Repeated Vector x Column-major Matrix

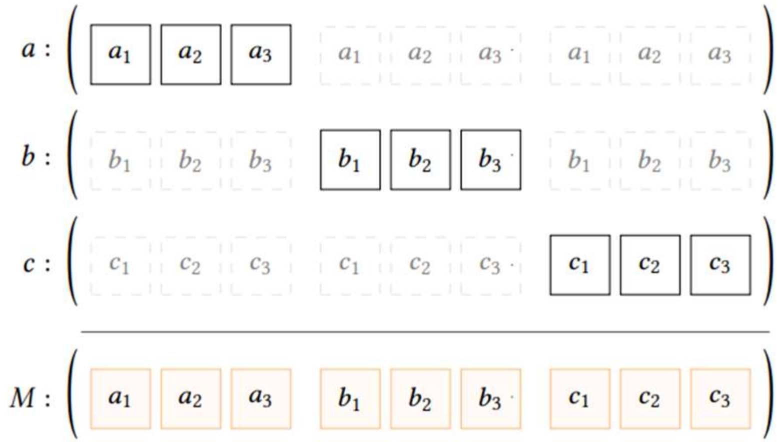


Wrap Up

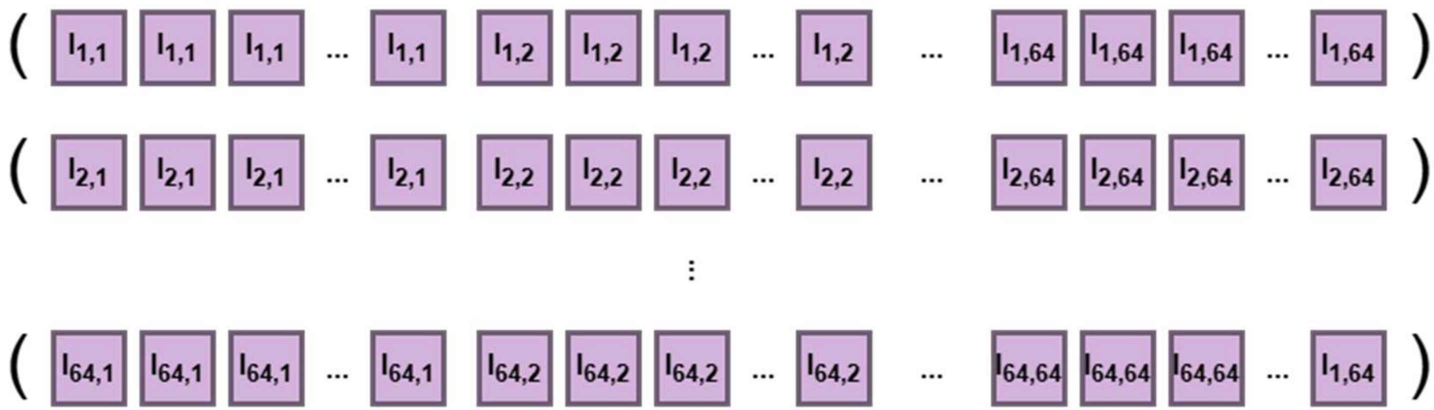
Wrap Up Expanded



Wrap Up Repeated



Input


(I1,1 I1,1 I1,1 ... I1,1 I1,2 I1,2 I1,2 ... I1,2 ... I1,64 I1,64 I1,64 ... I1,64)
(I2,1 I2,1 I2,1 ... I2,1 I2,2 I2,2 I2,2 ... I2,2 ... I2,64 I2,64 I2,64 ... I2,64)
⋮
(I64,1 I64,1 I64,1 ... I64,1 I64,2 I64,2 I64,2 ... I64,2 ... I64,64 I64,64 I64,64 ... I1,64)

Norm

$$y = \alpha * \left(\frac{x - mean}{\sqrt{variance + \in}} \right) + bias$$

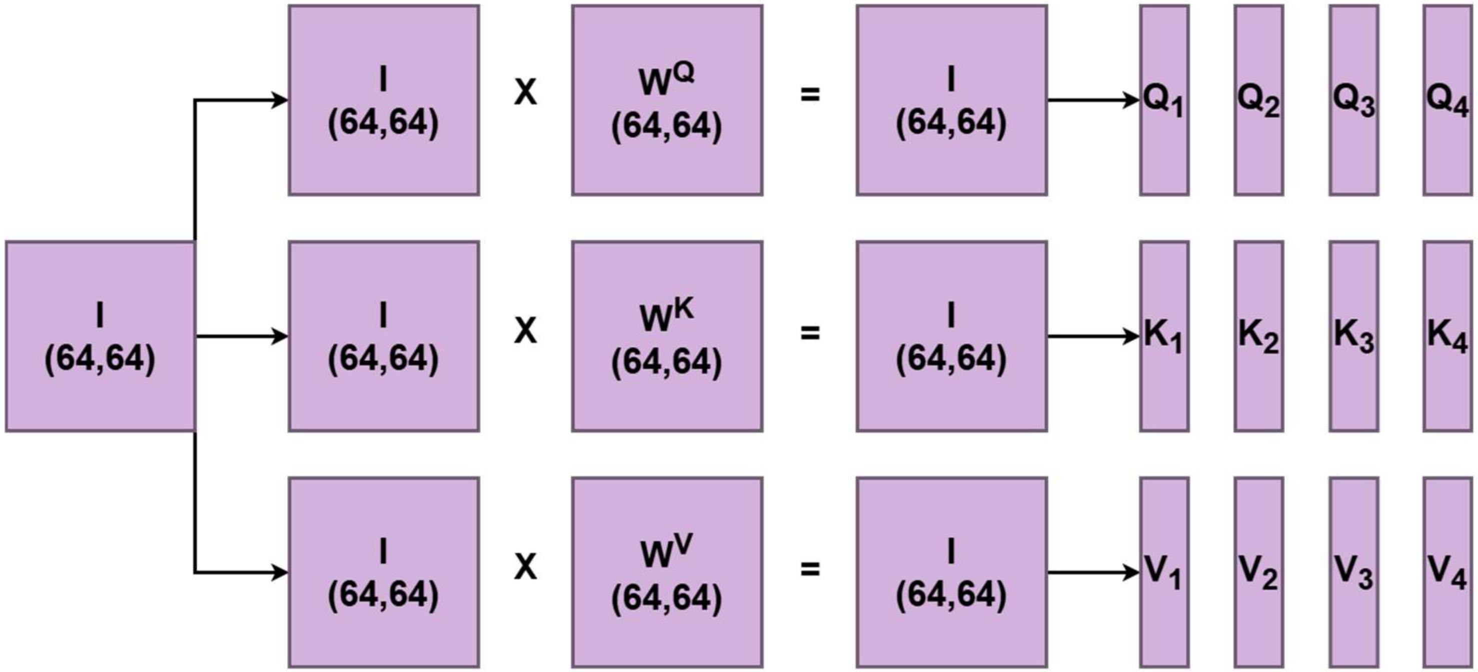
$$y' = (x_i - pre.mean) * (pre.std * \alpha) + b$$

Attention

$$Attention(Q, K, V) = SoftMax(\frac{QK^T}{\sqrt{features}}) * V$$

$$SoftMax(x) = \frac{e^{x\_i}}{\sum_{j=1}^{n} e^{x\_j}}$$

Multi Head Attention



Multi Head Attention

$$Head(Q_i, K_i, V_i) = SoftMax(\frac{Q_i K_i^T}{\sqrt{values\ per\ head}}) * V_i$$

$$MultiHead(Q, K, V)$$
$$= Concat(head_1, head_2, \ldots, head_n) * W^O + b^O$$

Multi Head Attention - Softmax

$$\frac{1}{64} + \frac{63}{64^2} * x_j + \frac{63 \; * 62}{64^3} * x_j^2 - \frac{1}{64^2} * \sum_{i\,=\,1, i\,!=\,j}^{n} x_i - \frac{62}{64^3} * \sum_{i\,=\,1, i\,!=\,j}^{n} x_i^2$$

$$- \; \frac{2 * 62}{64^3} * x_j * \sum_{i\,=\,1, i\,!=j}^{n} x_i + \frac{2}{64^3} * \sum_{i\,=\,1, i\,!=\,k,\; i\,!=\,j}^{n} x_i * \sum_{k\,=\,1, k\,!=i,\; k\,!=\,j}^{n} x_k$$

Multi Head Attention

$$Head(Q_i, K_i, V_i) = SoftMax(\frac{Q_i K_i^T}{\sqrt{values\ per\ head}}) * V_i$$

$$MultiHead(Q, K, V) = Concat(head_1, head_2, \ldots, head_n) * W^O + b^O$$

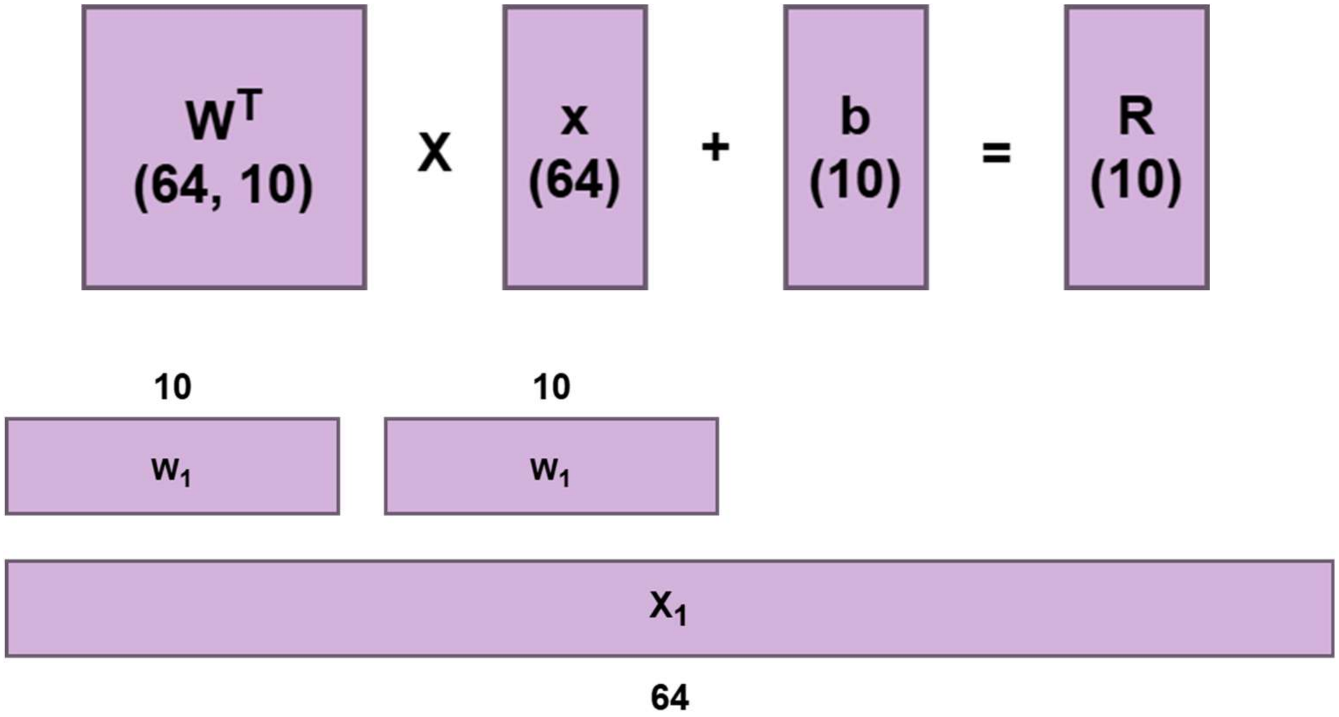
MLP - GELU

$$MLP(x) = ApproxGELU(x * W_1 + b_1) * W_2 + b_2$$

$$ApproxGELU(x)$$
$$= 0.5 * x (1 + \tanh(\sqrt{}(2/pi) * (x + 0.044715 * x\text{^}3))$$

$$ApproxTanh(x)$$
$$= \tanh(a) + \left(1 - \tanh^2(a)\right) * (x - a) - \left(\tanh(a) - \tanh^3(a)\right)$$
$$* (x - a)^2$$

Classifier



Result

- OpenFHE library
- Ring Dimension: 2^16
- Mulipliative Depth: 25
- Accuracy compared to classifications: 43%
- Accuracy of original model: 46%
- Runtime: approx. 2 hours

Conclusion

- Privacy Preserving Vision Transformer using Homomorphic Evaluation
- Enable the use of Machine Learning in sensitive sectors
- Currently focused on inference but might be expanded to training in the future