CHET: Compiler and Runtime for Homomorphic Evaluation of Tensor Programs

Roshan Dathathri*, Olli Saarikivi†, Hao Chen†, Kim Laine†, Kristin Lauter†, Saeed Maleki†, Madanlal Musuvathi†, Todd Mytkowicz†

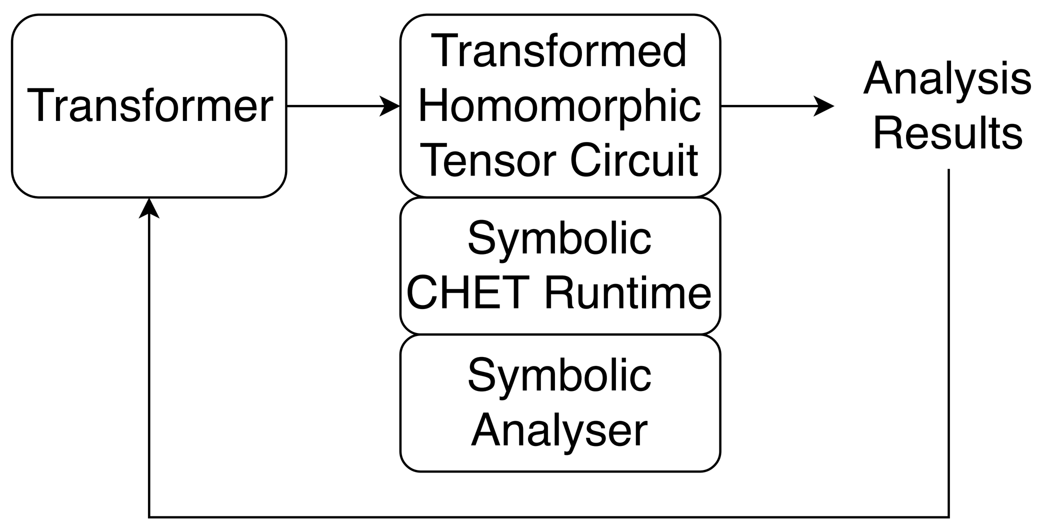
*Department of Computer Science, University of Texas at Austin, USA †Microsoft Research, USA

Introduction

Building efficient and correct applications with leveled integer FHE schemes is tedious and error-prone:

- Incorrect encryption parameters will compromise either security or performance.
- Best performance requires efficient use of batching.
- For the CKKS family of schemes correctness requires careful precision selection.

CHET is a compiler and runtime that automates many parts of this process for neural network inference tasks. The compiler applies transformations based on a framework of symbolic analysis passes.



The resulting *Optimized Homomorphic Tensor Circuit* is used by the runtime to evaluate the network on encrypted data.

Data Layout Selection

CHET selects one of four *layout policies* for the runtime:

HW and CHW use the corresponding layout throughout.
HW-conv switches to HW for convolutions, CHW otherwise.
CHW-fc uses CHW starting from the first fully connected layer.

The selection uses a cost analysis pass, which accounts for:

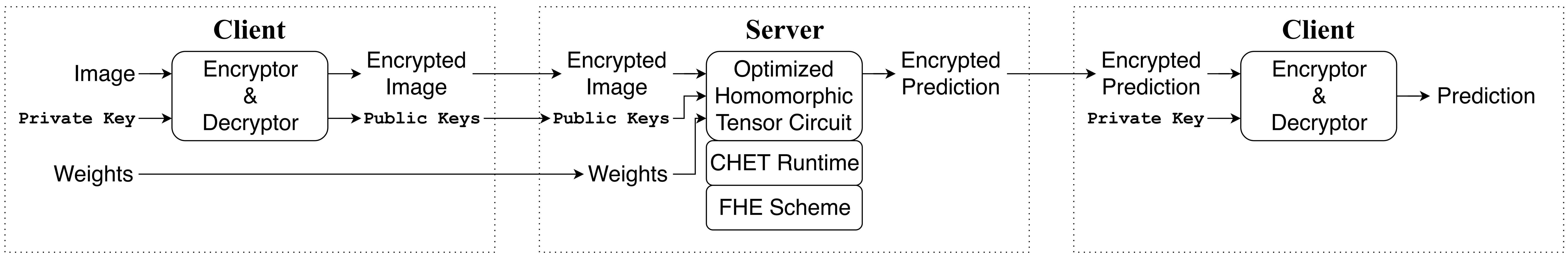
- Relative costs of operations.
- Required encryption parameters.
- Degree of parallelism in the model vs. available execution units.
- Cost of switching between layouts.

Parameter Selection

CHET supports parameter selection for both HEAAN's CKKS and SEAL's RNS-CKKS. The analysis passes simulate scaling behavior while measuring modulus consumed by *rescale* operations.

Rotation Key Selection

Using a network specific set of rotation keys can provide up to 2X performance improvement. This transformation uses a pass that records the necessary rotation keys.



Data Layouts for Vectorized Kernels

CHET includes kernels optimized for low-latency inference of CNNs, which operate on strided layouts of values into batched ciphertexts. We have considered two classes of layouts:

HW Each channel of an image is in a separate ciphertext.
CHW Each ciphertext holds multiple channels.

Consider the 2D-convolution of an image tensor of shape (IC, H, W) with a filter of shape (FH, FW, IC, OC):

$$output_{oc,h,w} = \sum_{ic=0}^{IC} \sum_{fh=0}^{FH} \sum_{fw=0}^{FW} input_{ic,h+fh-\lfloor \frac{FH}{2} \rfloor, w+fw-\lfloor \frac{FW}{2} \rfloor} \cdot filter_{fh,fw,ic,oc}$$

For the HW layout the kernel is:

```
for oc in indices(OC):
  output[oc] = zeroCipher
  for ic,fh,fw in indices(IC,FH,FW):
    weight = encode(filter[fh,fw,ic,oc], scalarScale)
    rotated = leftRotate(input[ic], fh * W + fw)
    output[oc] = multiplyPlain(rotated, weight)
  tryRescale(output[oc], cipherScale)
```

The kernel for CHW is similar, but includes extra rotations and additions to handle multiple channels in a ciphertext. Compared to HW, the kernel may perform fewer multiplications. However, HW has a lower depth, because with CKKS encoding a uniform value into all slots is exact. *These kinds of trade-offs make it challenging to choose the best layout manually.*

Evaluation

We have evaluated CHET on a set of CNNs. To our knowledge, SqueezeNet-CIFAR is the largest network evaluated on FHE to date.

Network	Layers	CHET best	Hand-written
LeNet-5-small	4	8s	14s
LeNet-5-medium	4	51s	140s
LeNet-5-large	4	265s	
Industrial	7	312s	2413s
SqueezeNet-CIFAR	10	1342s	

The following figure compares latencies for each network with different layout policies. *No single policy is best for all networks.*