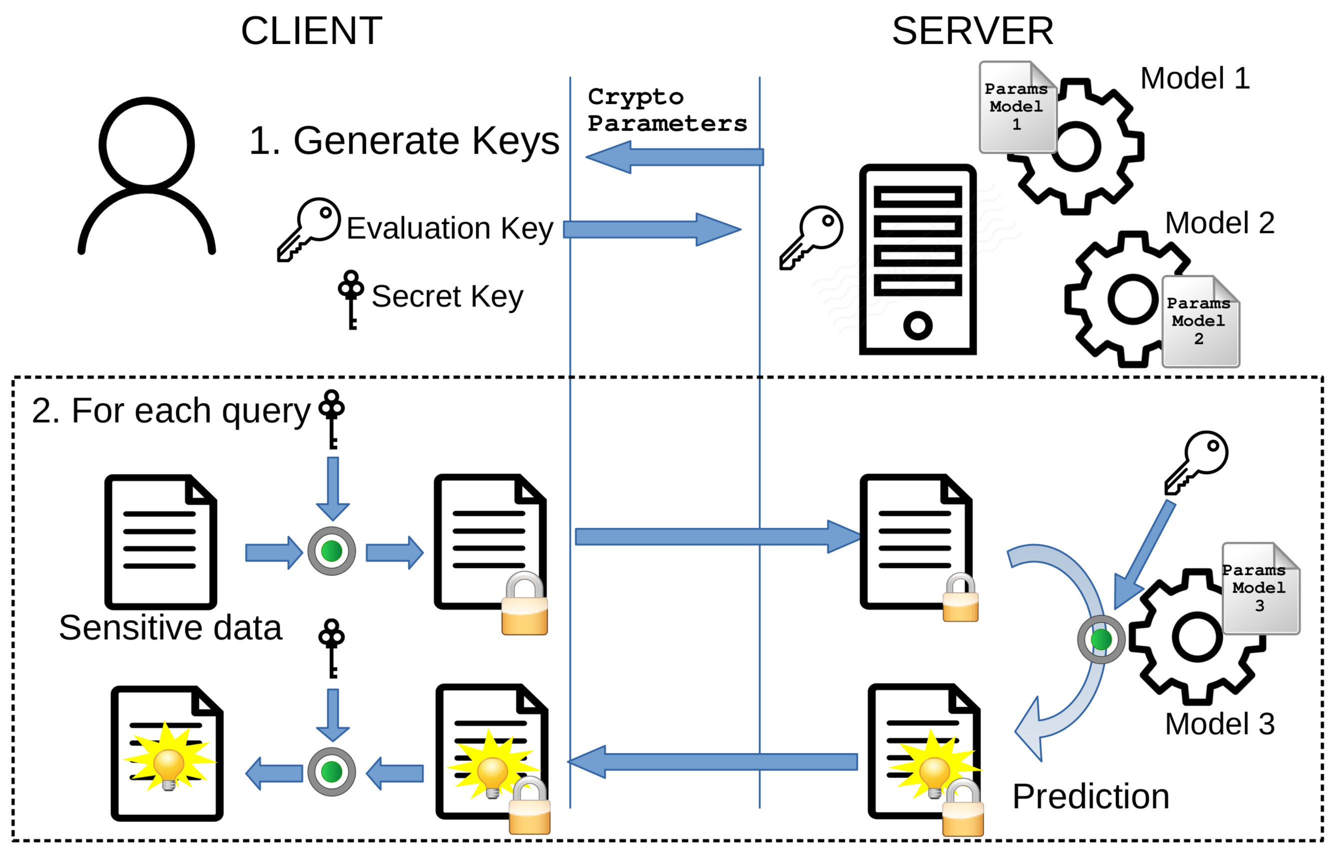
CONCRETE-ML: A DATA-SCIENTIST-FRIENDLY TOOLKIT FOR MACHINE LEARNING OVER ENCRYPTED DATA

Benoit Chevallier-Mames Jordan Frery Arthur Meyre Andrei Stoian

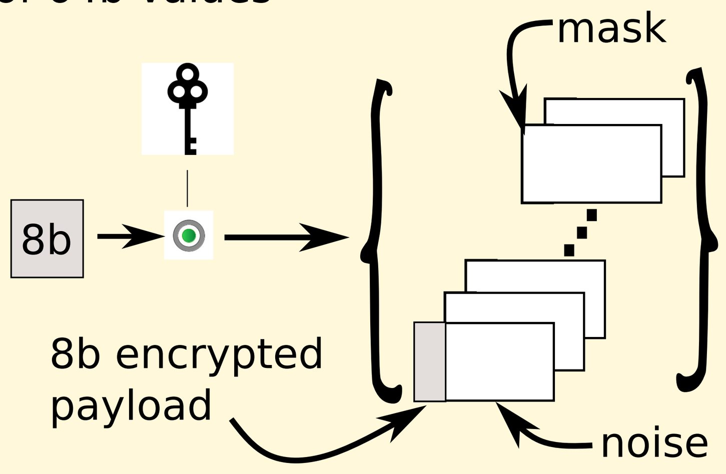
FHE.Org · 2022

Using Machine Learning services with FHE

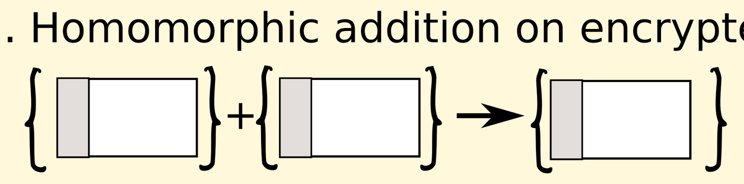


Torus FHE Operations

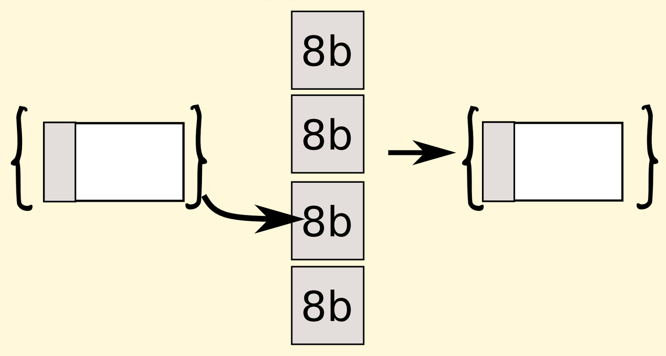
Encryption of 8b values produces high dimensional LWE vectors of 64b values



1. Homomorphic addition on encrypted values



2. Lookup encrypted values in a table, produces encrypted results



Constraints of FHE as implemented in Concrete Numpy:

1. Process only integers, integers can have up to 8 bits
2. Operations allowed:
   - addition of two encrypted values
   - multiplication of encrypted with a clear constant: convolution, GEMM
   - arbitrary lookup-tables: activations, quantization, normalization

Converting float models to integer FHE models

Model quantization

- Reduce the representation precision of model weights and activations
- Post Training Quantization: finds the best set of discrete weights and activations starting with a float model
- Quantization Aware Training: trains the best performing model under the constraint that weights and activations are discrete

Concrete-ML

A machine learning toolkit that data-scientists can use to create machine learning models that operate on encrypted data

1. *scikit-learn*, *xgboost* compatible
2. *pytorch*, *ONNX* converters available, *keras/tf* supported through ONNX import

Concrete Stack

Concrete-ML is built upon the **Concrete Stack**:

- Concrete-Framework: cryptographic primitives and compilation of linear algebra programs to FHE
- Concrete-Numpy : numpy to FHE converter through compilation of numpy programs

Supported models in Concrete-ML

- Tree-based models:
  - DecisionTree
  - RandomForest
  - XGBoost
- Neural Networks
  - Convolutional
  - Fully Connected
- Linear models
  - Linear Regression
  - Logistic Regression
  - Generalized Linear Model
  - Support Vector Classifier
  - Support Vector Regression

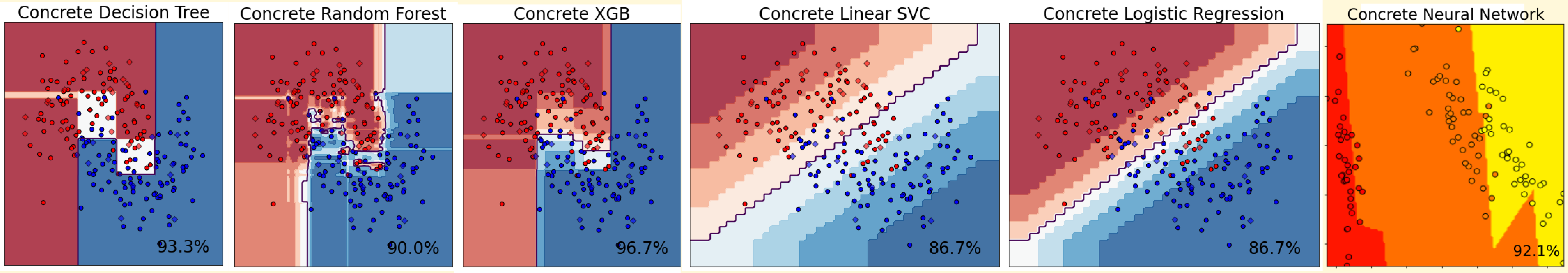
Usage

The Concrete-ML API has minimal differences with respect to scikit-learn

- Concrete-ML models are a drop-in replacement of scikit-learn models
- To compile and to execute in FHE requires just a single function call for each

```
q_linreg = ConcreteLinearRegression(n_bits=3)
q_linreg.fit(x_train, y_train)
q_linreg.compile(X)
y_pred_q = q_linreg.predict(x_test)
y_pred_fhe = q_linreg.predict(x_test,
                execute_in_fhe=True)
```

Experiments



Model	Dataset	Metric	fp32 result	Quantized result	FHE result
Linear Regression	Synthetic	r2 score	88.5%	87.3%	87.3%
Logistic Regression	Synthetic	Accuracy	90%	85%	85%
Decision Tree	spambase	f1-score	87.7%	88.7%	88.7%
Fully Connected Neural Network	IRIS	Accuracy	100%	92.1%	92.1%
Fully Connected Neural Network	MNIST	Accuracy	98%	95%	95%