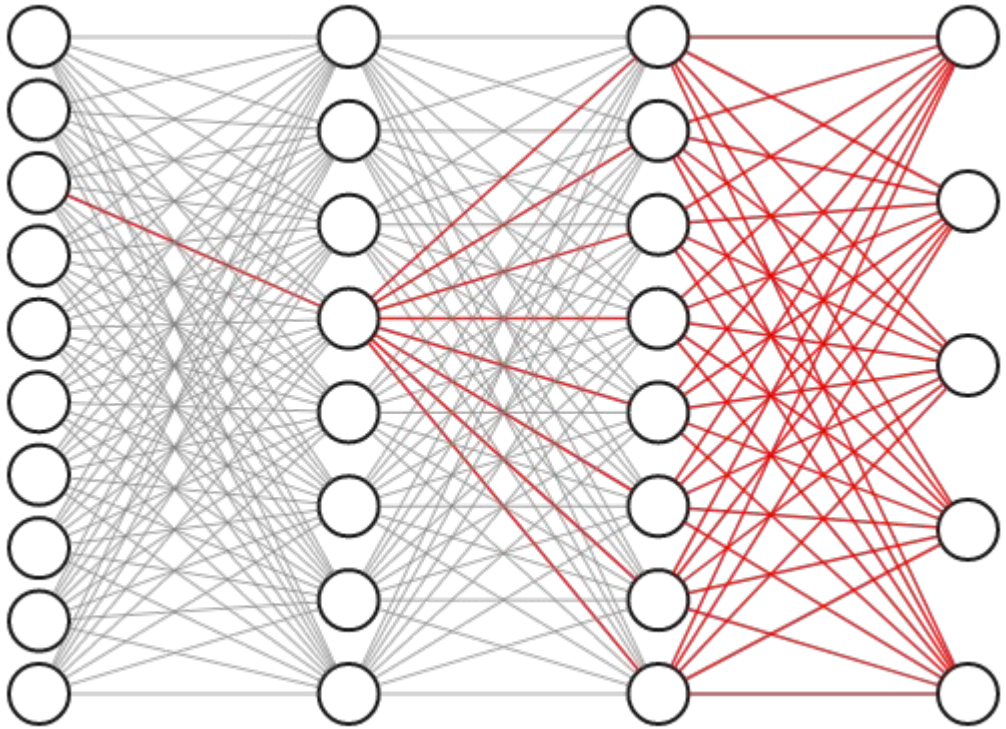
RIT

GNP: Fast and Scalable Tensor Engine

Viet Nguyen

Overview



- GNP (GPU Numpy) is a **fast** and **scalable** tensor computing library that utilize the **parallel computation** of GPUs.
- Built upon numpy library.
- GNP boost the performance by **optimizing computation** of numpy ndarray on GPU.

Requirement

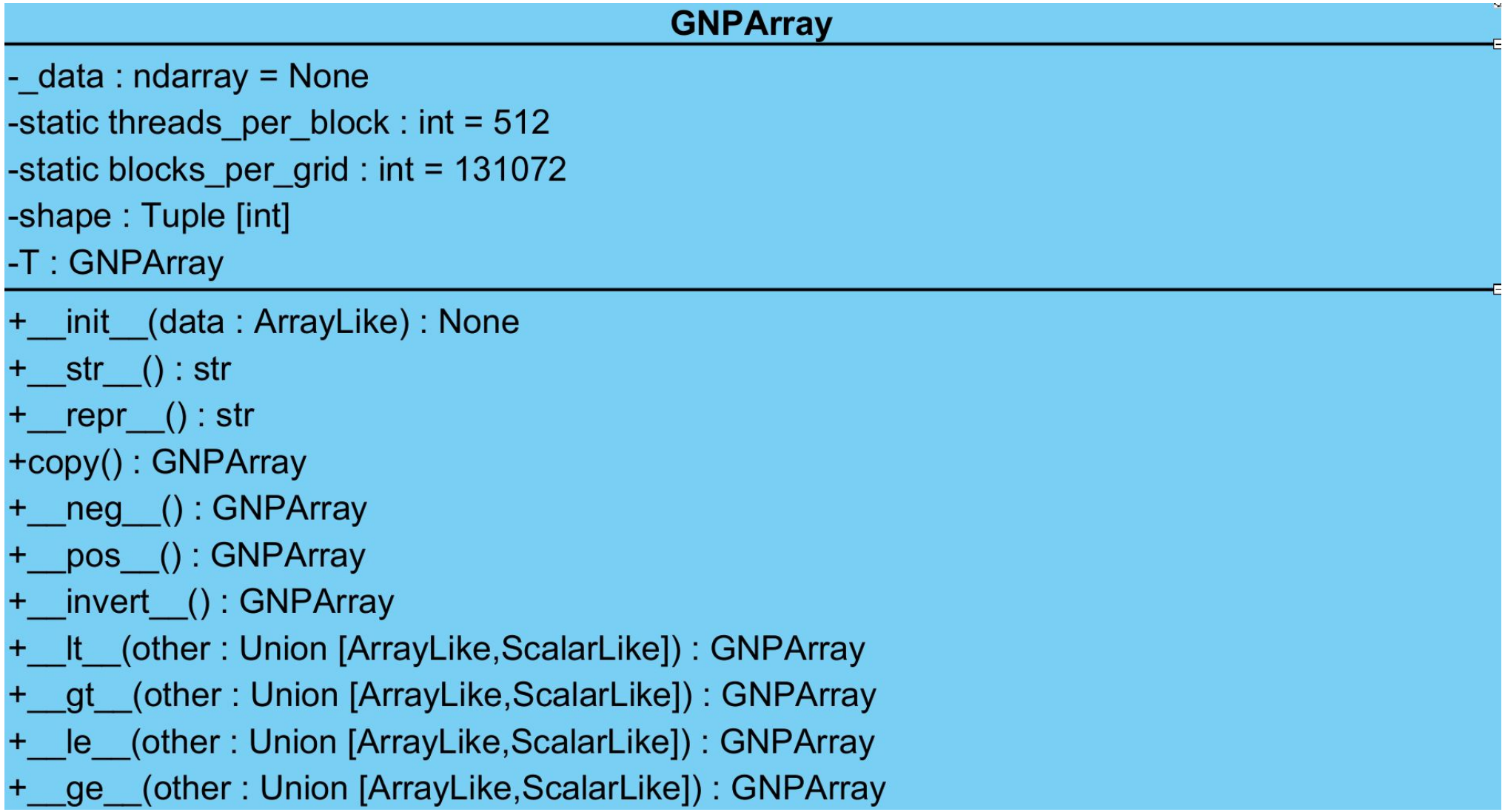
- Numba: just-in-time compiler
- Cuda kernel can be implemented in python using numba's cuda decorator

```python
import numpy as np
import numba
from numba import cuda

@cuda.jit
def increment_by_one(an_array):
    # Thread id in a 1D block
    tx = cuda.threadIdx.x
    # Block id in a 1D grid
    ty = cuda.blockIdx.x
    # Block width, i.e. number of threads per block
    bw = cuda.blockDim.x
    # Compute flattened index inside the array
    pos = tx + ty * bw
    if pos < an_array.size:  # Check array boundaries
        an_array[pos] += 1

an_array = np.asarray([1,2,3])

threadsperblock = 32
blockspergrid = (an_array.size + (threadsperblock - 1)) // threadsperblock
a = increment_by_one[blockspergrid, threadsperblock](an_array)
an_array

# (2, 3, 4)
```

GNP: supported functions

- Unary operators:
  - Negation
  - Positive Assignment
  - Invert
- Binary operators:
  - Add
  - Subtract
  - Multiply
  - True divide
  - Floor divide
  - Mod
  - Pow
  - And
- Binary operators:
  - Or
  - Xor
  - Right shift
  - Left shift
- Comparison operators:
  - Greater than
  - Less than
  - Greater than or equal to
  - Less than or equal to
  - Equal
  - Not equal
- Linear Algebra:
  - Matrix multiplication
  - Batch matrix multiplication
- (Experimental) High level APIs:
  - Neural network sequential API
  - Fully connected neural network forward and backward pass
  - SGD Optimizer, MSE Loss
  - Non-linear activations: Relu, Tanh, Sigmoid

The GNP Array class: structure



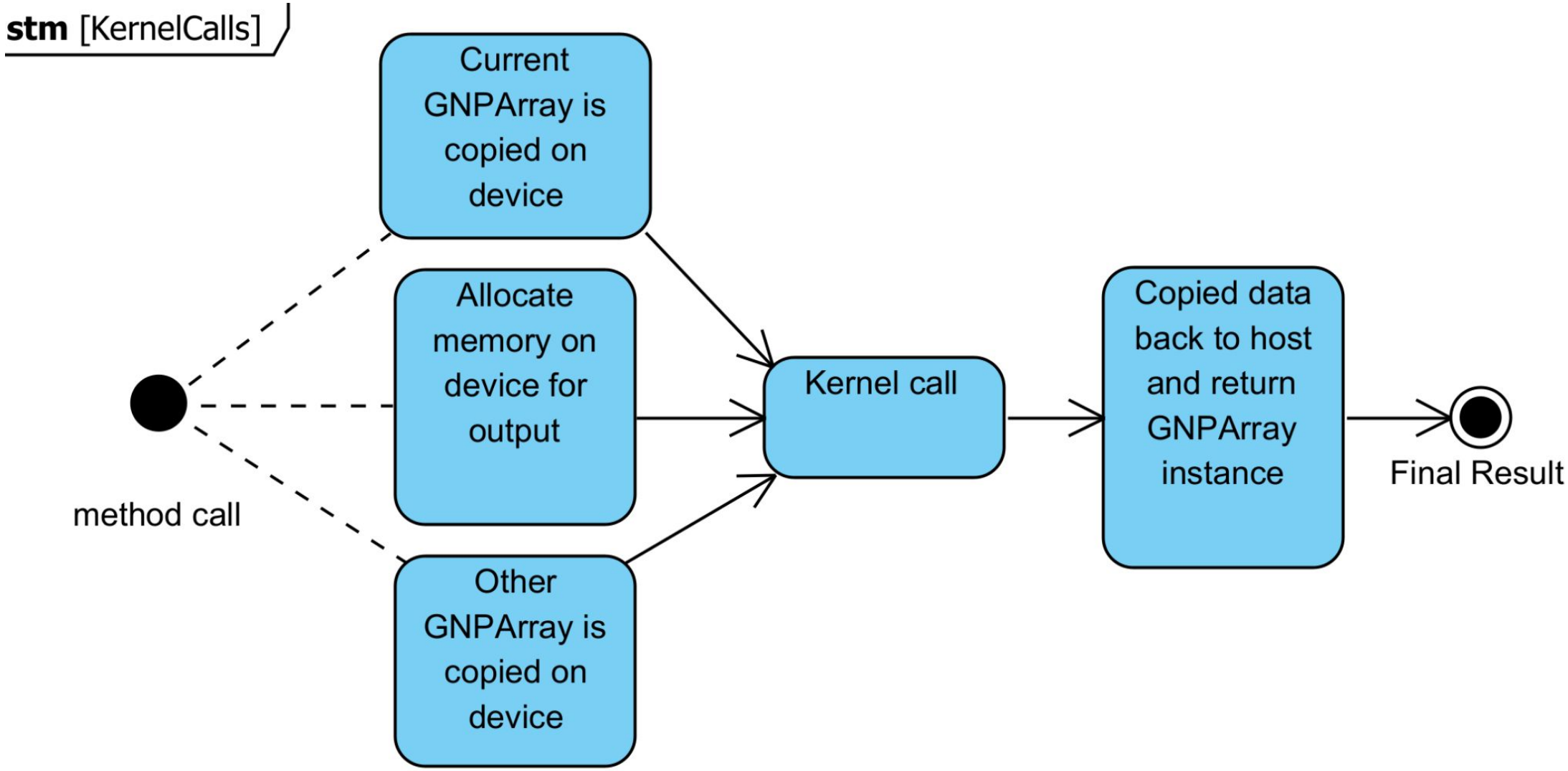
The GNP Array class: structure

+__eq__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__ne__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__add__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__sub__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__mul__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__truediv__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__floordiv__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__mod__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__pow__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__rshift__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__lshift__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__and__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__or__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__xor__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__isub__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__iadd__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__imul__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__idiv__(other : Union [ArrayLike,ScalarLike]) : GNPArray

The GNP Array class: structure

```
+__lshift__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__and__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__or__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__xor__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__isub__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__iadd__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__imul__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__idiv__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__ifloordiv__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__imod__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__ipow__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__ilshift__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__irshift__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__iand__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__ior__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__ixor__(other : Union [ArrayLike,ScalarLike]) : GNPArray
+__matmul__(other : Union [ArrayLike,ScalarLike]) : GNPArray
```

The GNP Array class: kernel calls

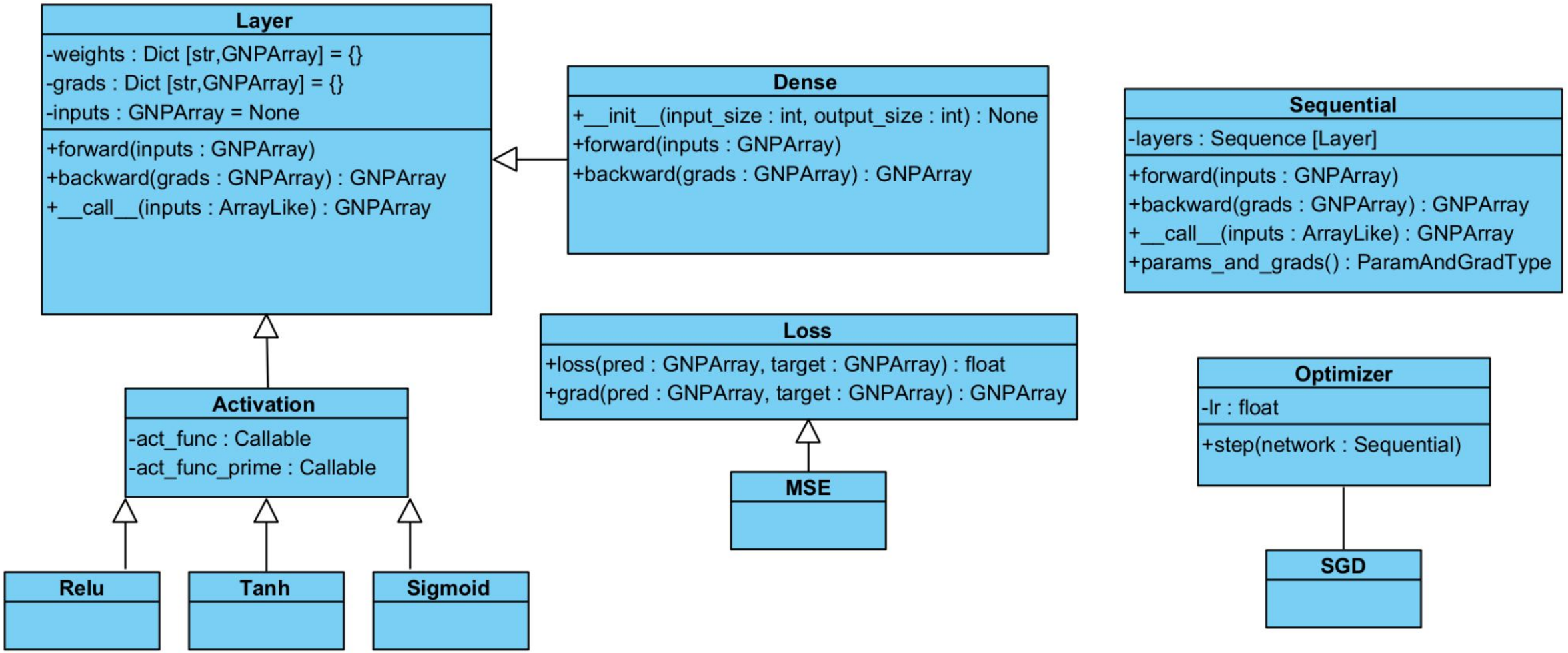


The GNP Array class: Example

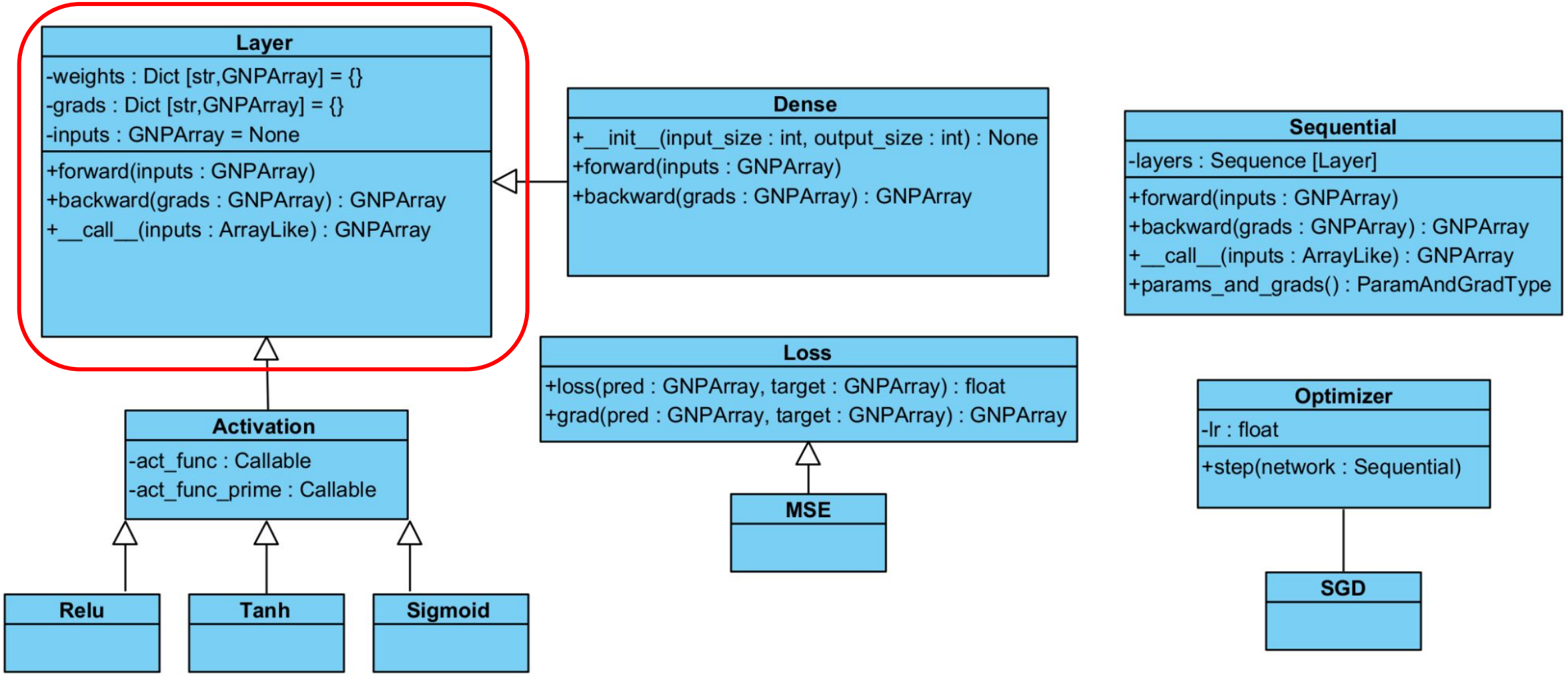
Let's test GNPArray

- `gnparray_test.py`
- `time_testing.py`: For batch matrix multiplication of two tensor shape (600, 600, 600)
  - Time for GNP computation: ~2.482605218887329 seconds
  - Time for Numpy computation: ~4.490365982055664 seconds
  - Time for numpy computation is often unstable

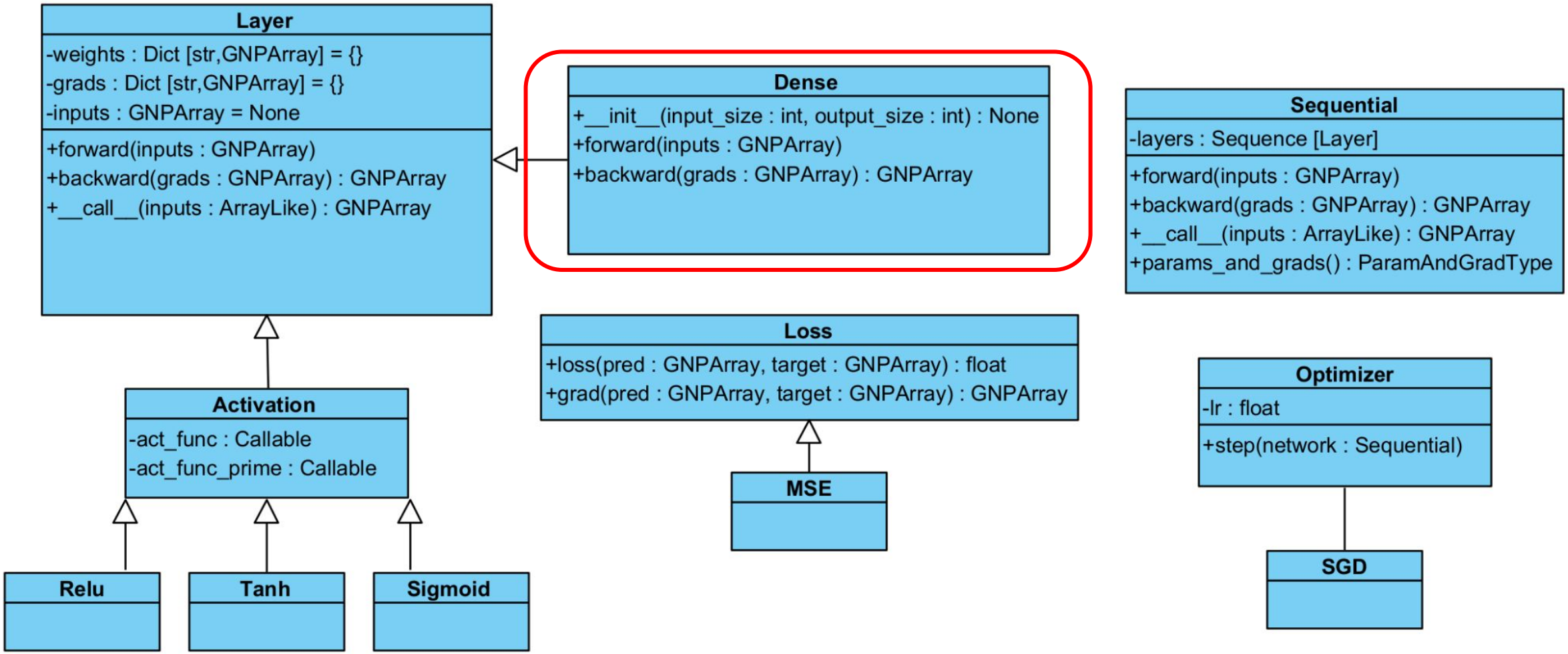
The GNP neural network APIs (experimental)



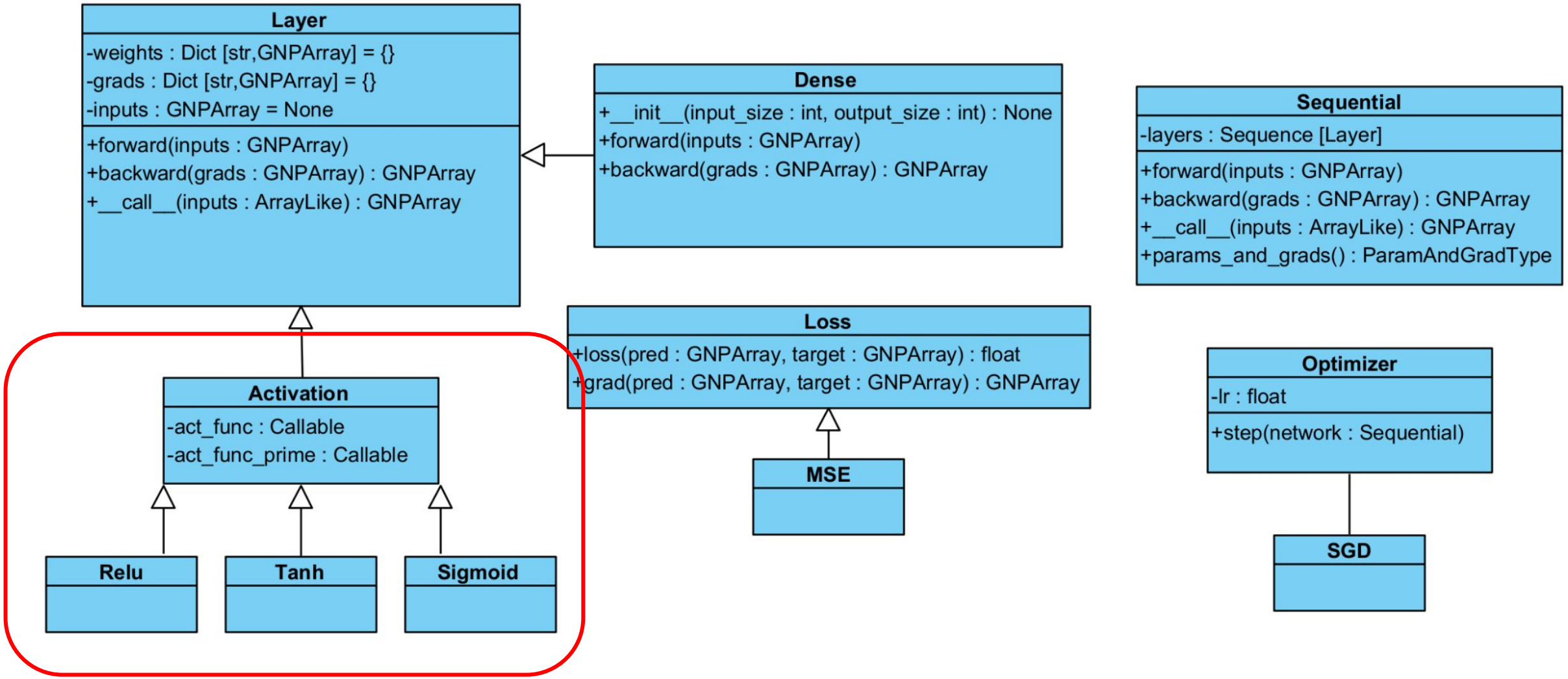
The GNP neural network APIs (experimental)



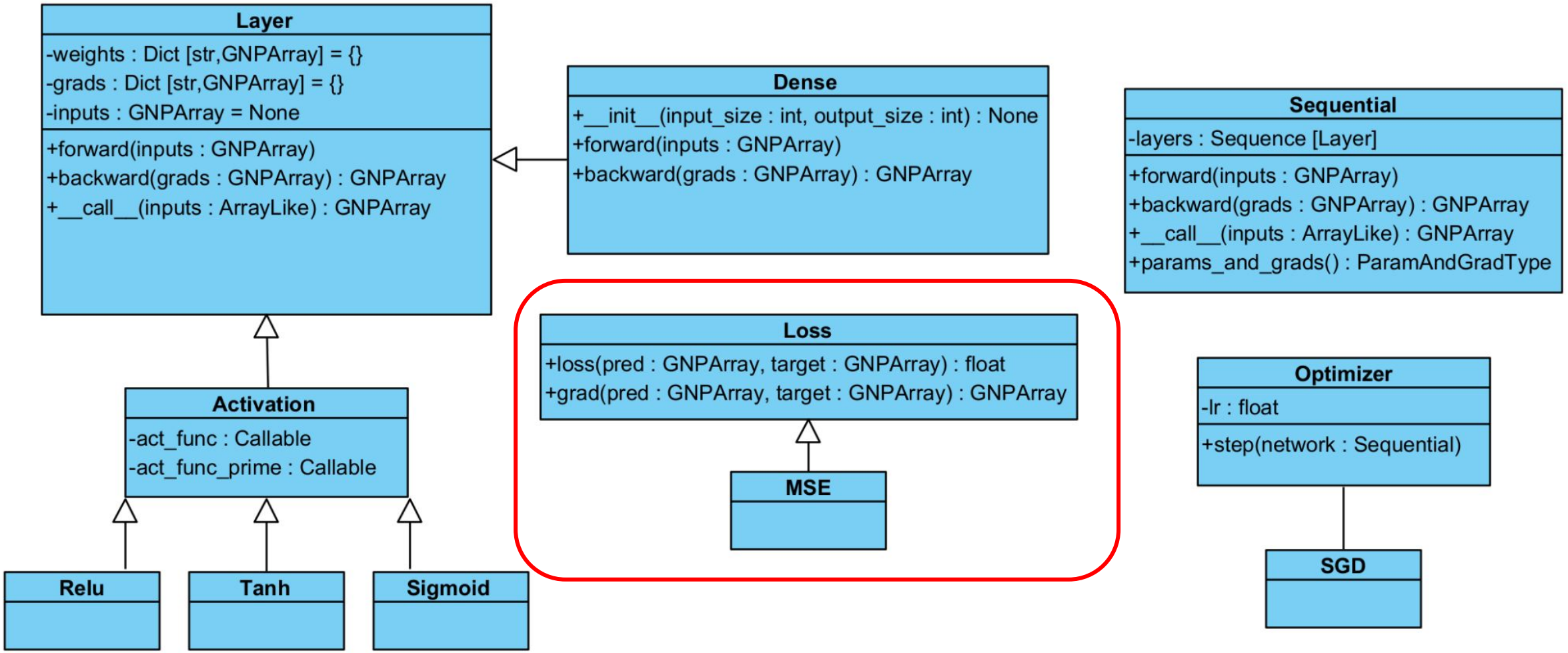
The GNP neural network APIs (experimental)



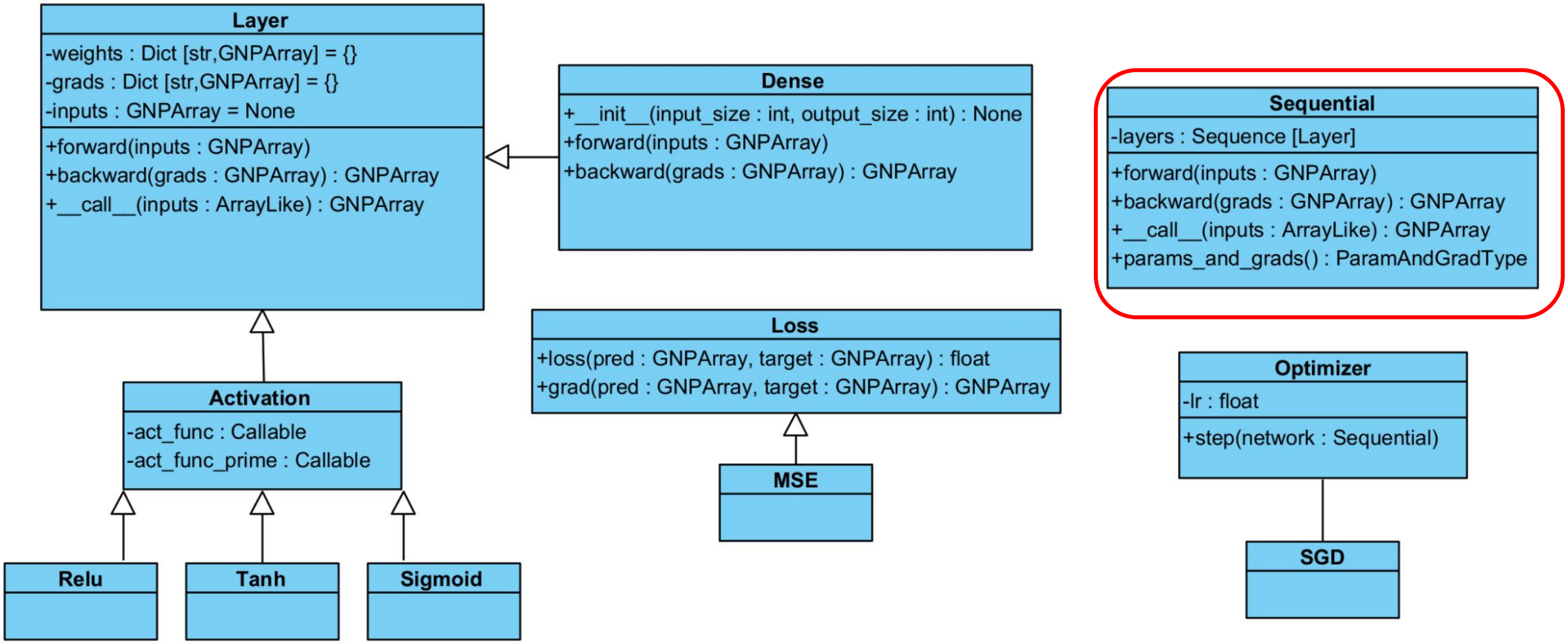
The GNP neural network APIs (experimental)



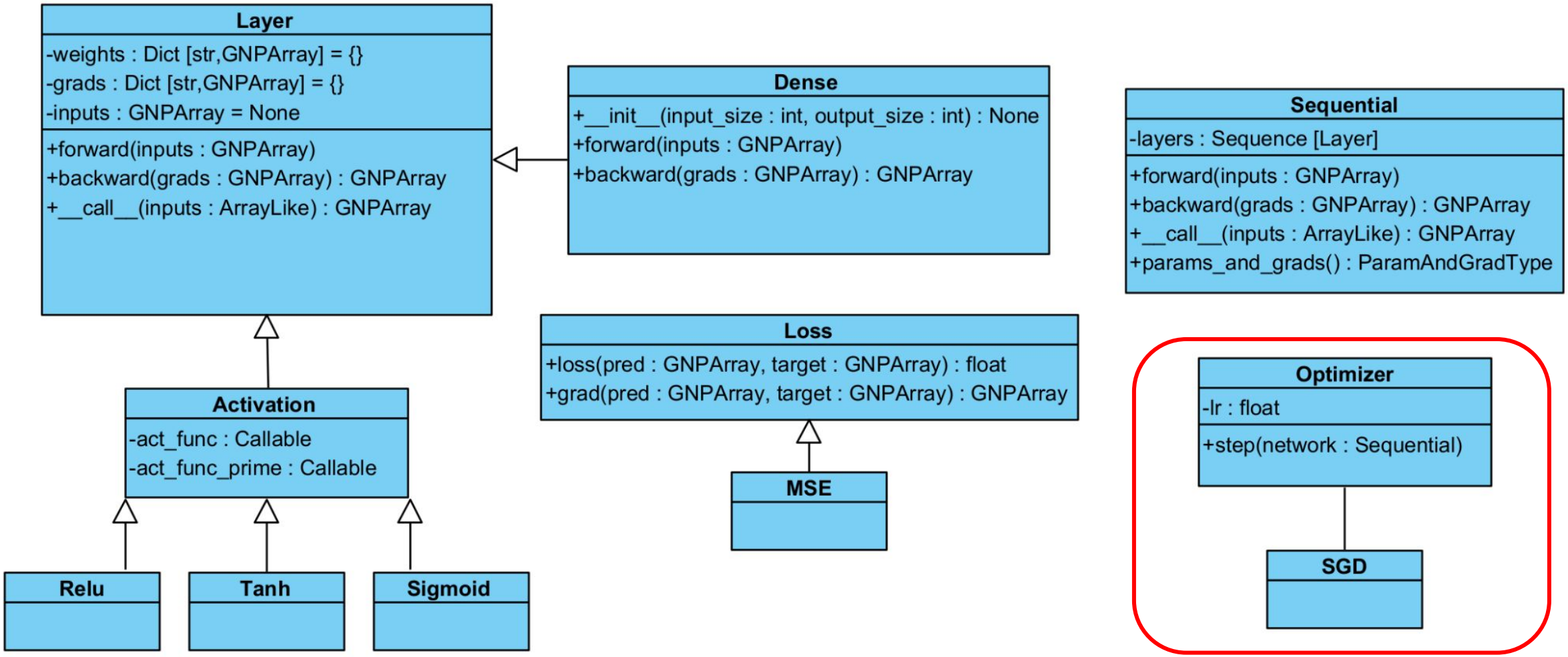
The GNP neural network APIs (experimental)



The GNP neural network APIs (experimental)



The GNP neural network APIs (experimental)



Future Work

- Efficient Array Operators and more functions
- More function in the neural network APIs:
  - Loss functions: BCE, NLL, etc.
  - Non-linear activation functions: Swish, Softmax, etc.
  - Layer types: CNN, LSTM, etc.
- More testing should be done
- Probability/Distribution APIs

RIT

Thank you for listening