System for Artificial Intelligence

Machine Learning Compilation

Siyuan Feng
Shanghai Innovation Institute

OUTLINE

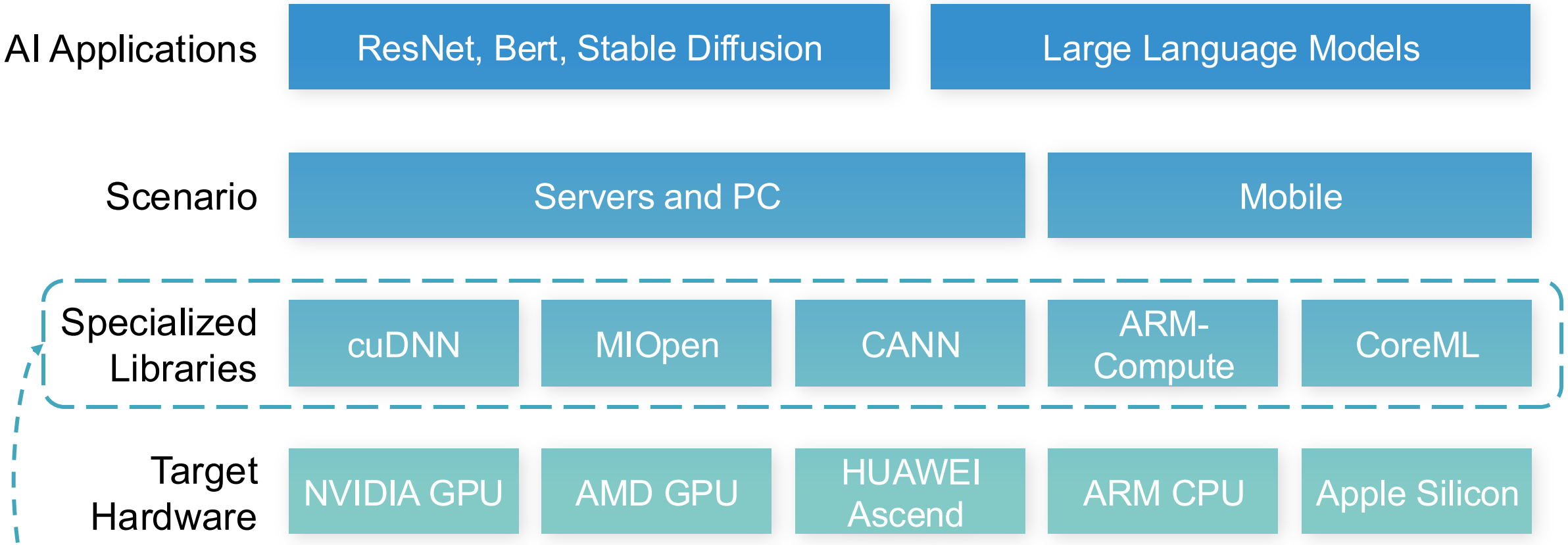
01 Overview of ML Compilation

02 Example ML Compilation Flow

01

Overview of ML Compilation

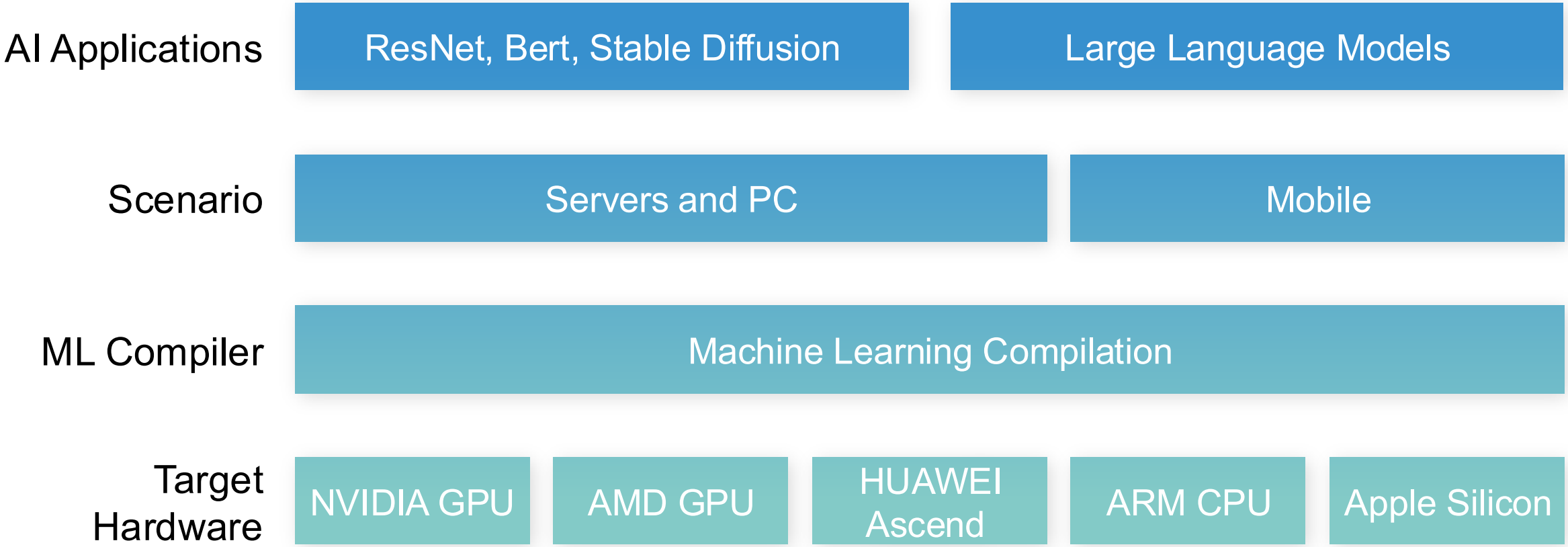
ML System Optimization Problem



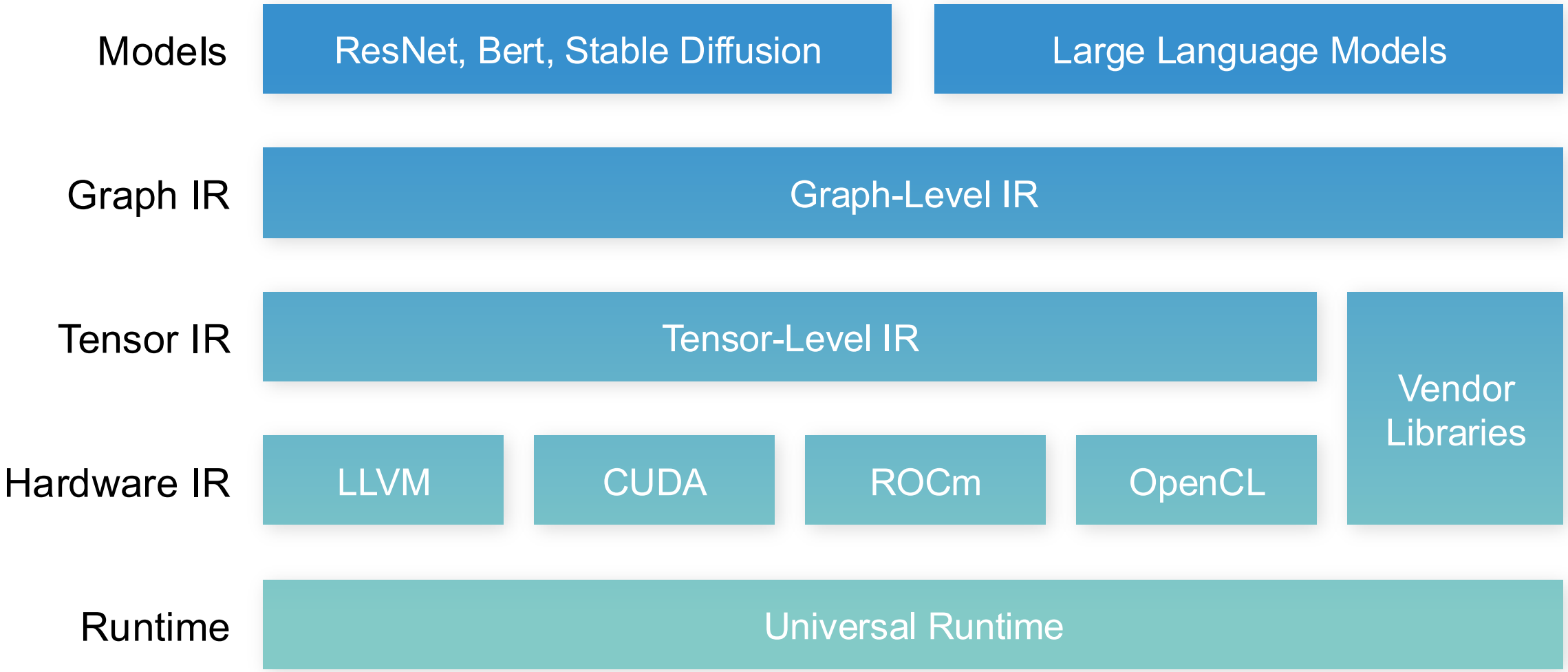
Specialized libraries for each backend (labor intensive)
Hard to be optimal for various models

ML Model Deployment Optimization Problem

上海创智学院
Shanghai Innovation Institute



Machine Learning Compilation Abstractions

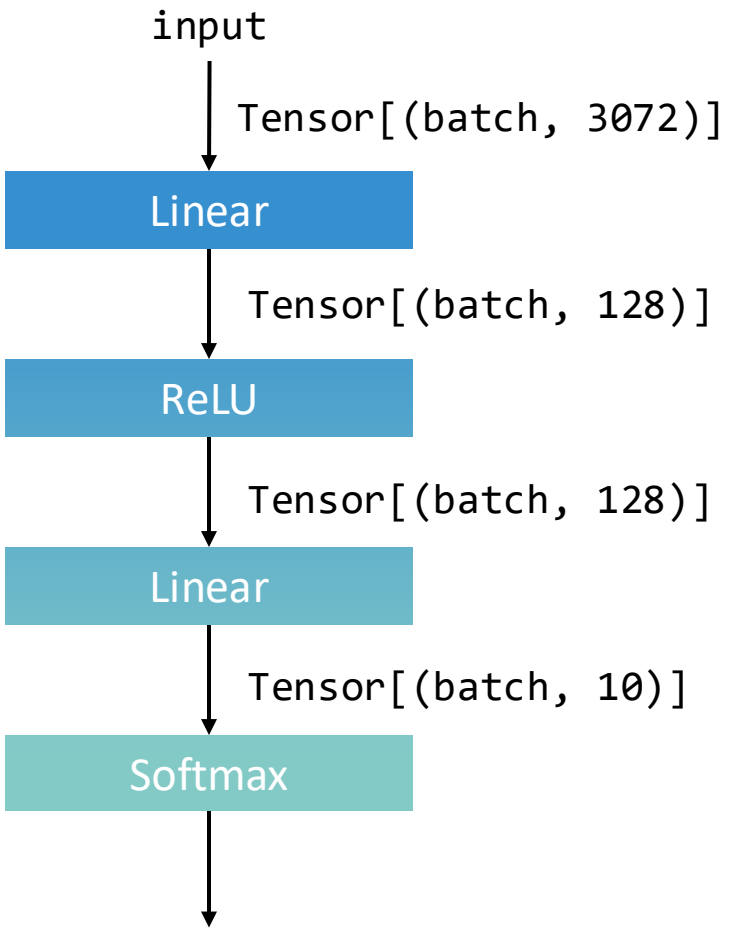


ML Compilation Goals

上海创智学院
Shanghai Innovation Institute

- There are many equivalent ways to run the same model execution. The common theme of MLC is optimization in different forms:
  - Minimize memory usage.
  - Minimize dependencies.
  - Improve execution efficiency.
  - Scaling to multiple heterogeneous nodes.
  -

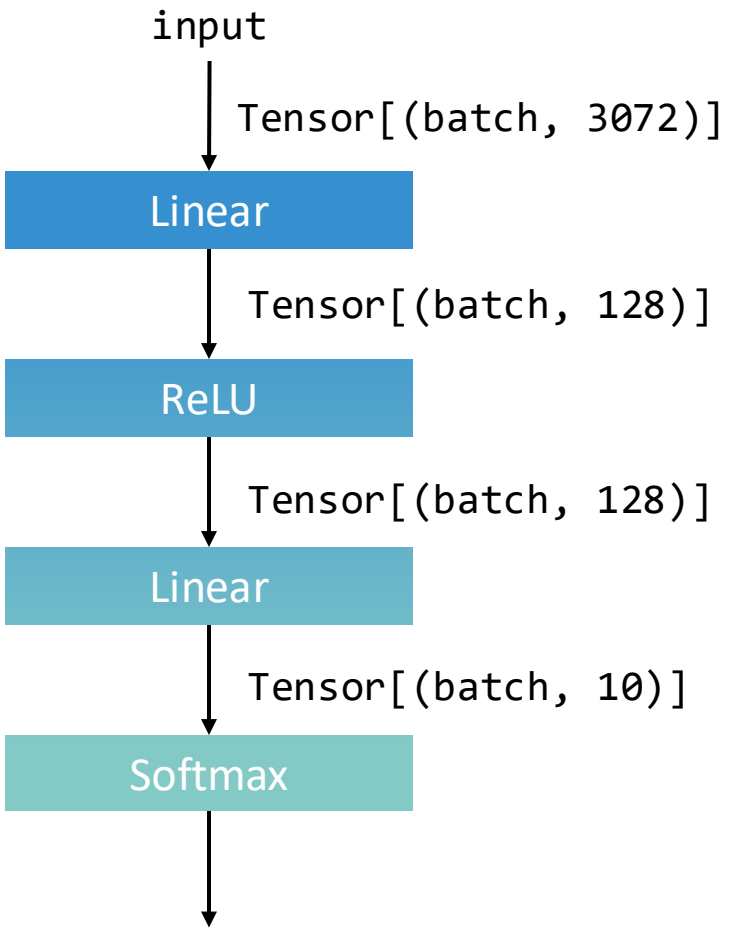
Key Elements in Machine Learning Compilation



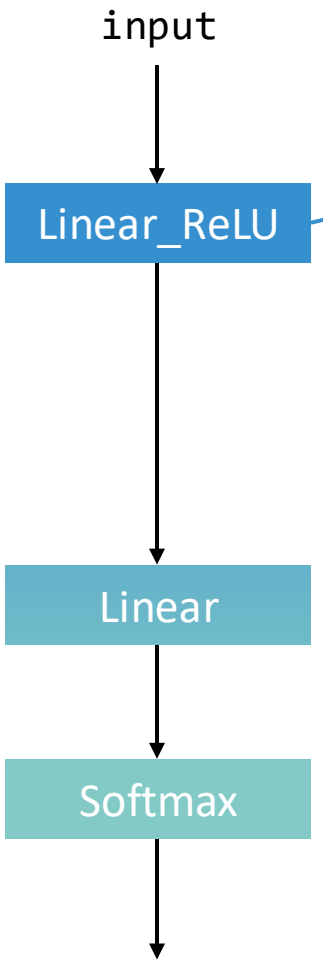
Tensor: multi-dimensional array that stores the input, output and intermediate results of model executions.

Tensor Functions that encodes computations among the input/output. Note that a tensor function can contain multiple operations

Example Compilation Process



Development



Deployment

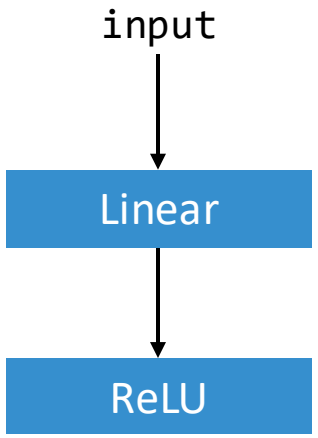
```
def linear_relu(x, w, out):
  for i in range(1):
    for j in range(200):
      out[i, j] = 0
      for k in range(3072):
        out[i, j] += x[i, k] * w[j, k]
      out[i, j] = max(out[i, j], 0)
```

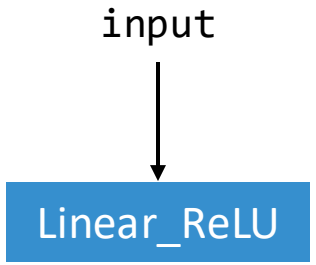
In this particular example, two tensor functions are folded into one (linear-relu). With a specialized implementation (in reality, they will be implemented using low-level primitives).

Abstraction and Implementation

上海创智学院 Shanghai Innovation Institute

- **Abstraction** refers to different ways to represent the same system interface (tensor function)

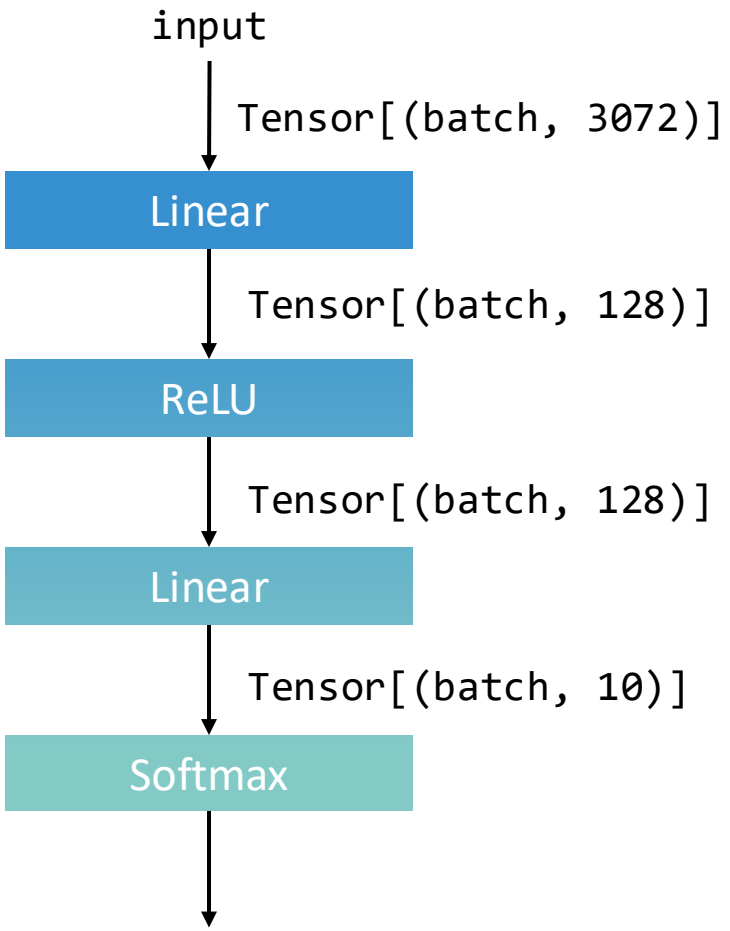




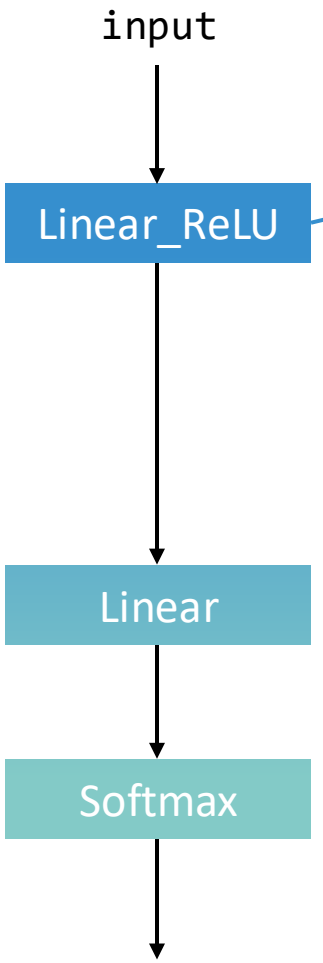

```
def linear_relu(x, w, out):
  for i in range(1):
    for j in range(200):
      out[i, j] = 0
      for k in range(3072):
        out[i, j] += x[i, k] * w[j, k]
      out[i, j] = max(out[i, j], 0)
```

- Three abstraction ways to represent the same tensor function (`linear_relu`), each providing a different level of details. In practice, we usually say that the more specialized version is an **implementation** of higher-level abstraction.

MLC as Tensor Function Transformation (with different abstractions)



Development




```
def linear_relu(x, w, out):
  for i in range(1):
    for j in range(200):
      out[i, j] = 0
      for k in range(3072):
        out[i, j] += x[i, k] * w[j, k]
      out[i, j] = max(out[i, j], 0)
```

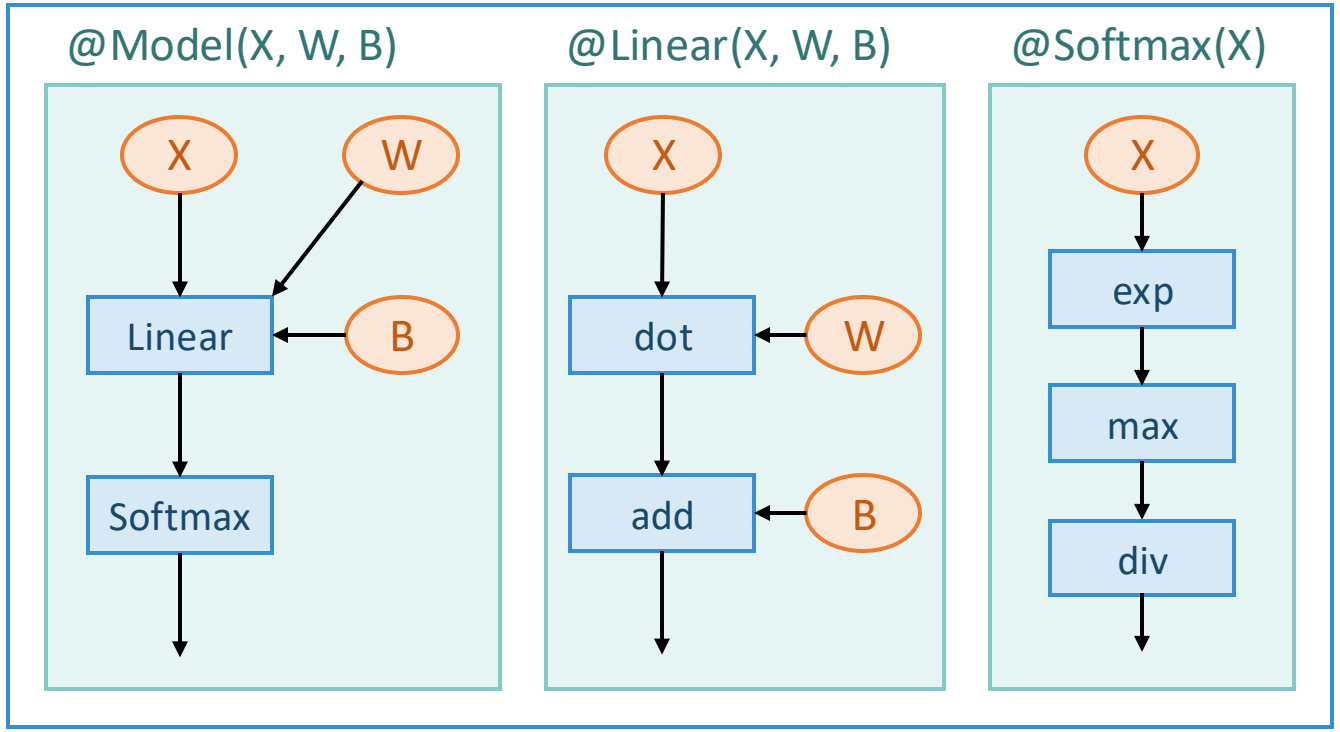
Deployment

Most MLC process can be viewed as transformation among tensor functions (that can be represented with different abstractions).

02

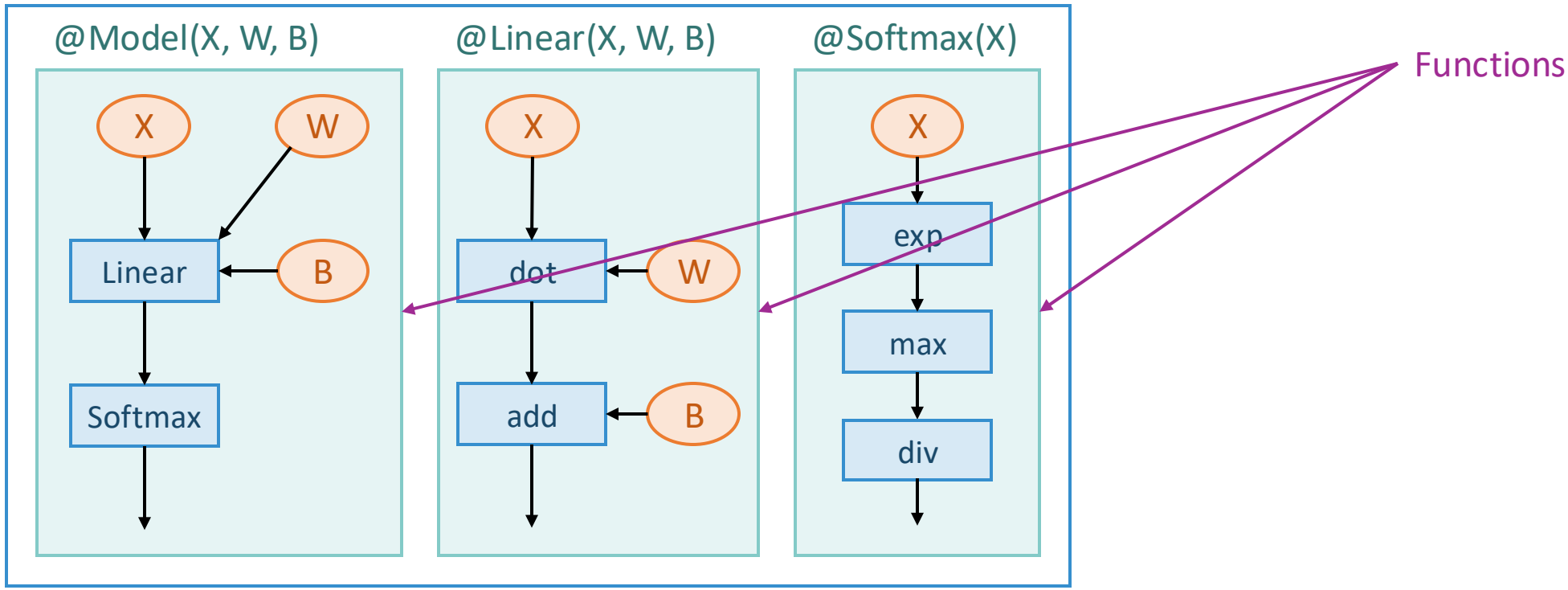
Example ML Compilation Flow

Compiler Representation of a ML Model

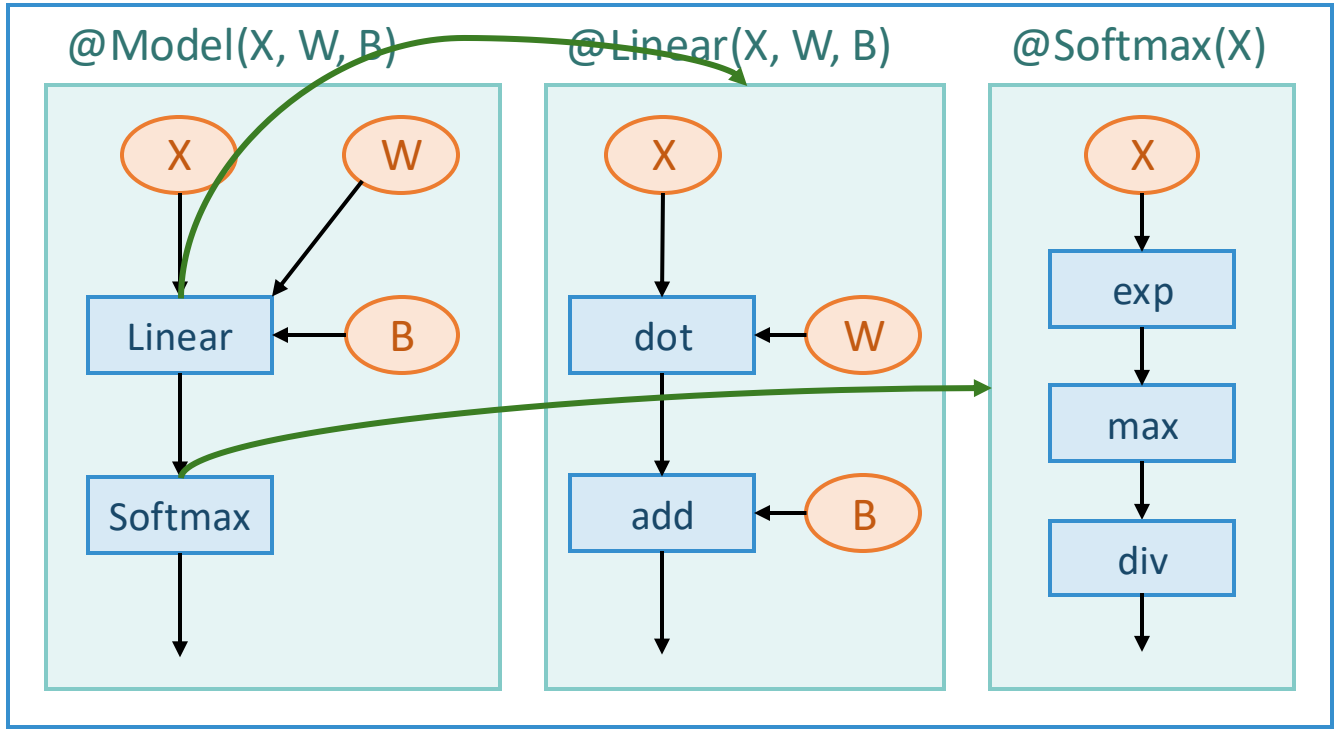


IRModule:
a collection of interdependent functions

Compiler Representation of a ML Model



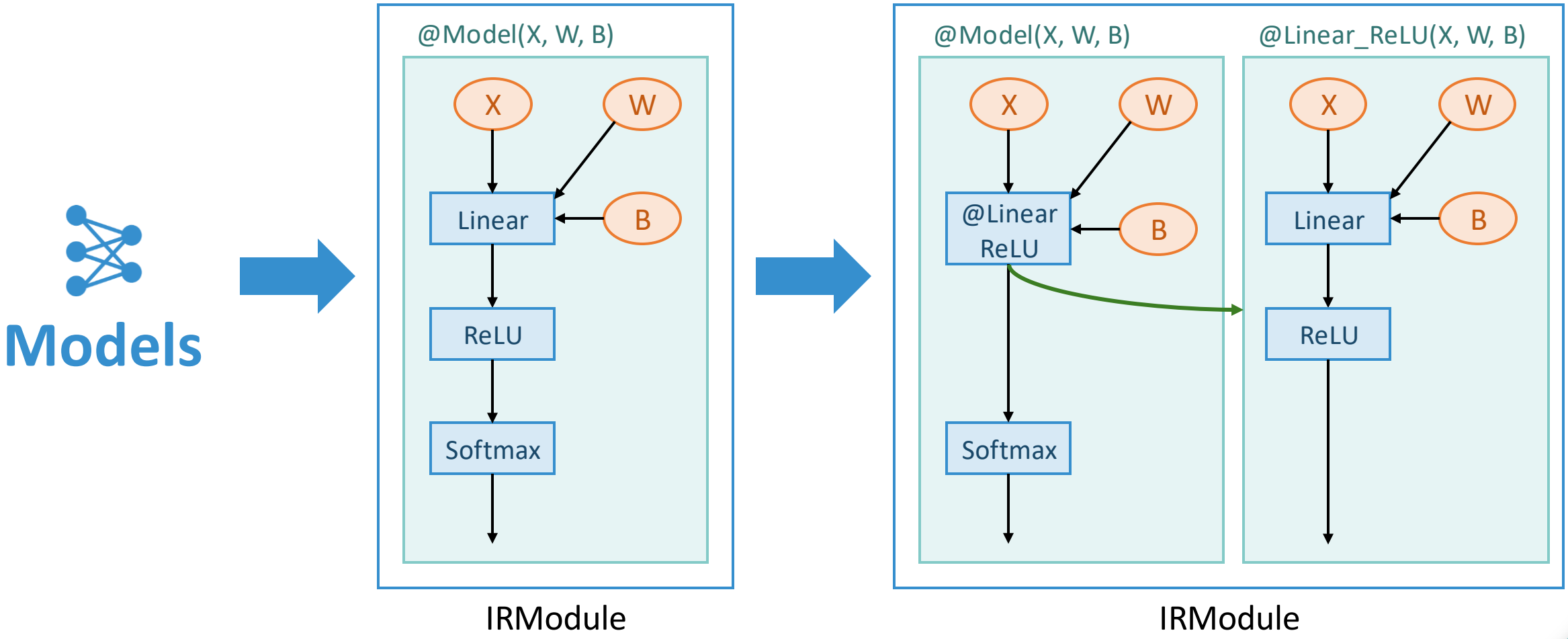
Compiler Representation of a ML Model



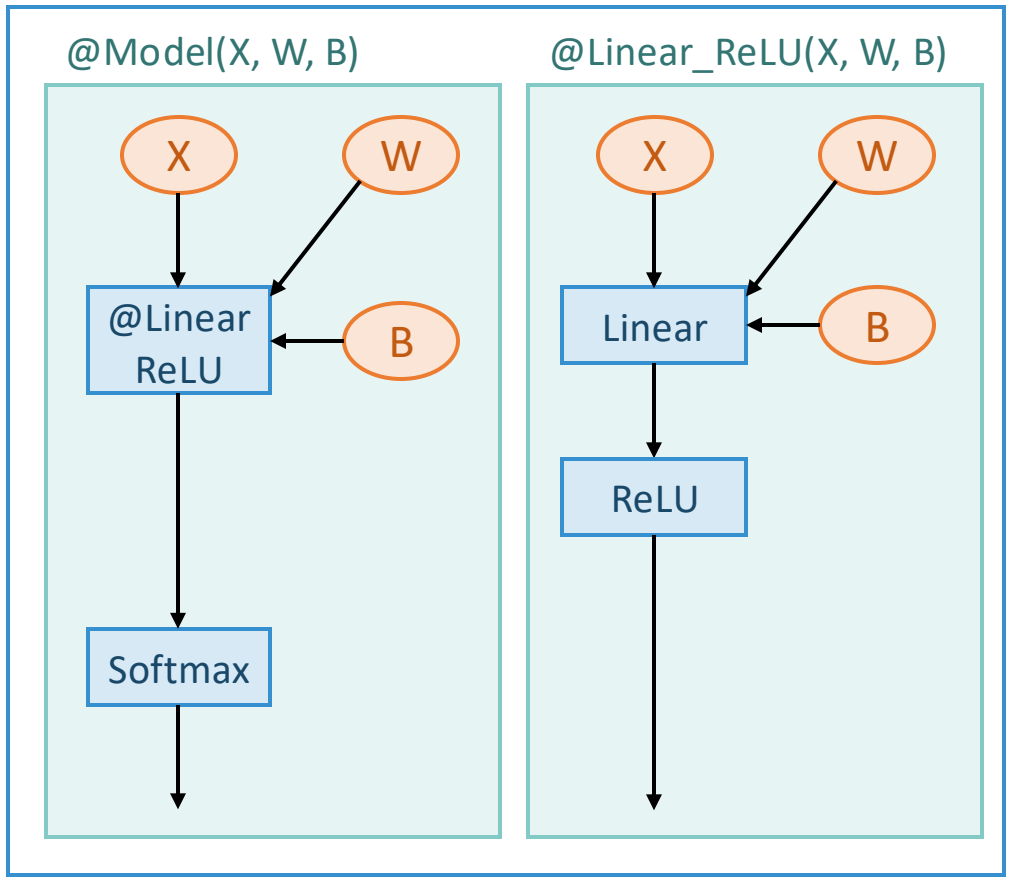
Function calls between models and functions

Example Flow: High-Level Transformations

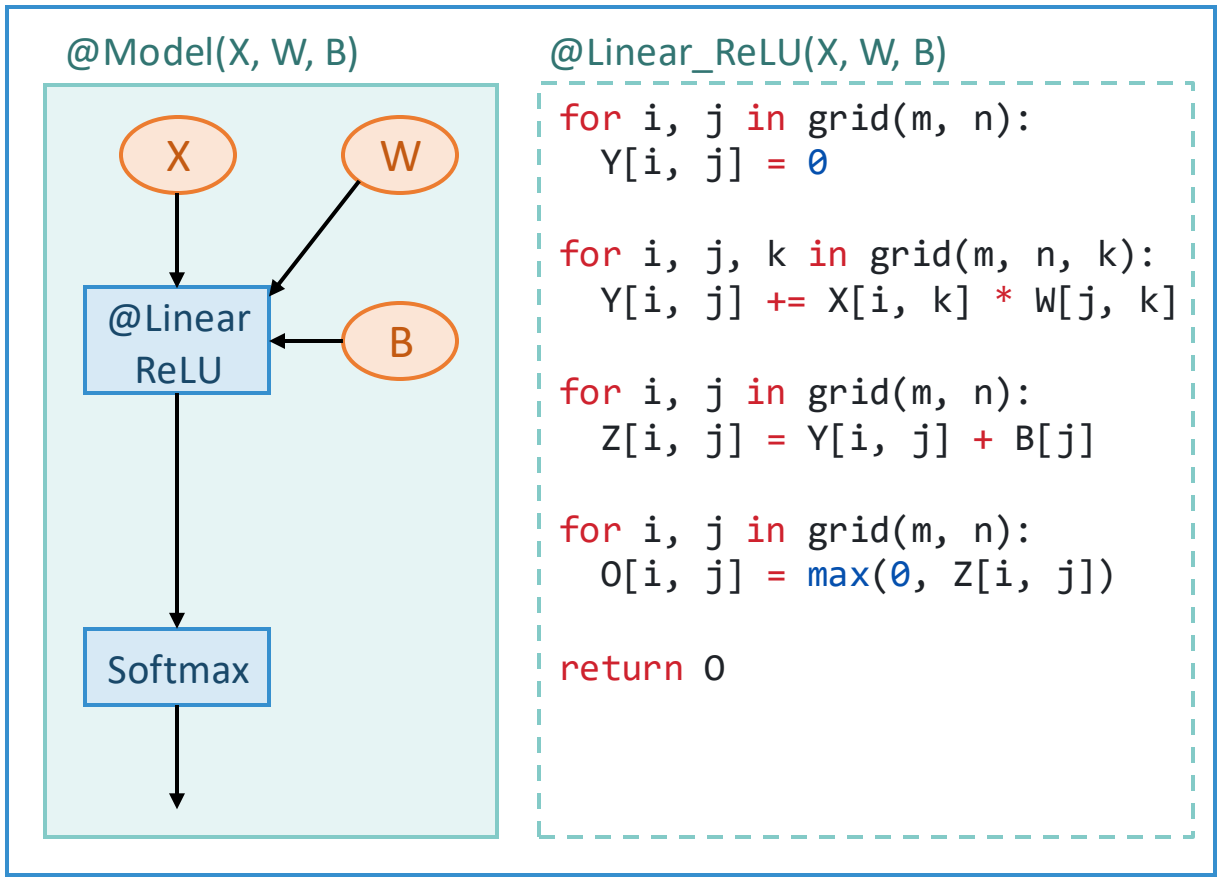
上海创智学院
Shanghai Innovation Institute



Example Flow: Lowering to Loop IR

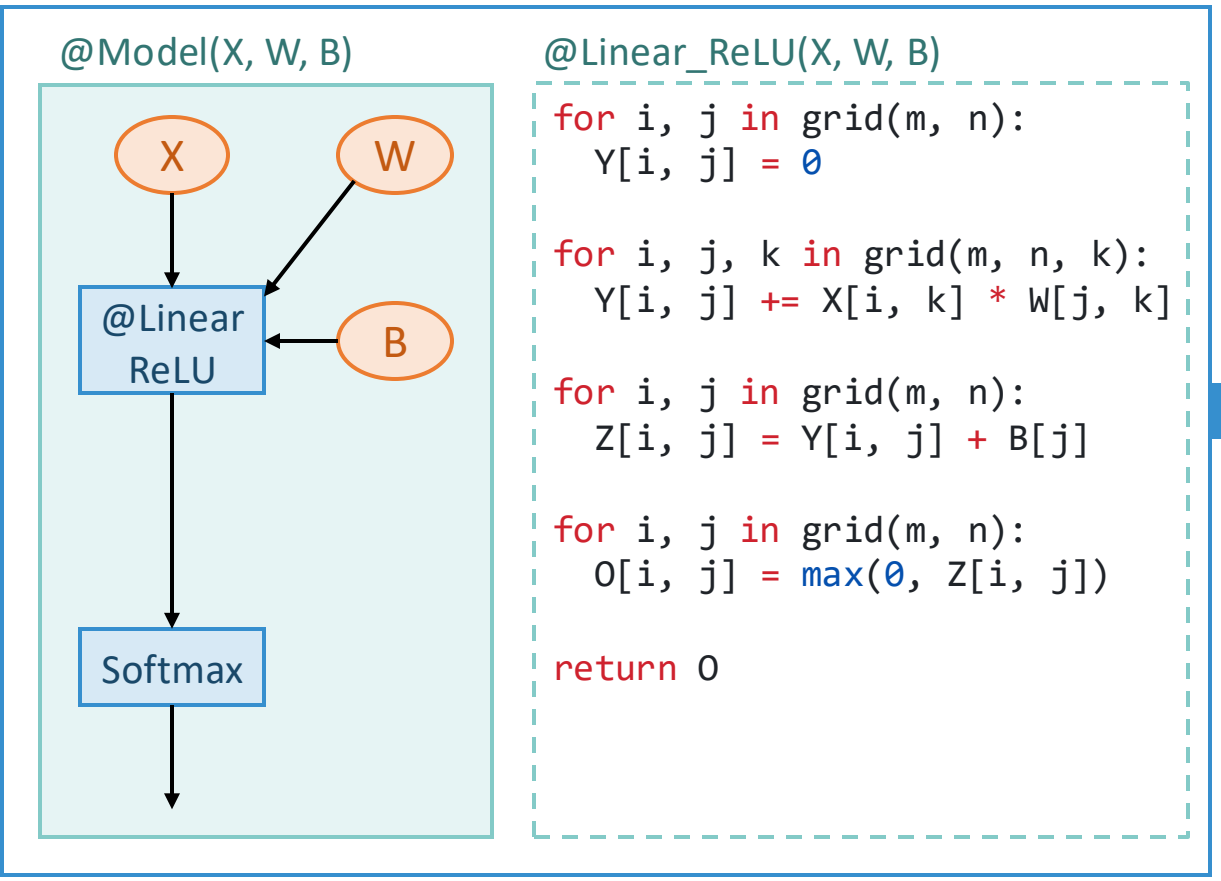


IRModule

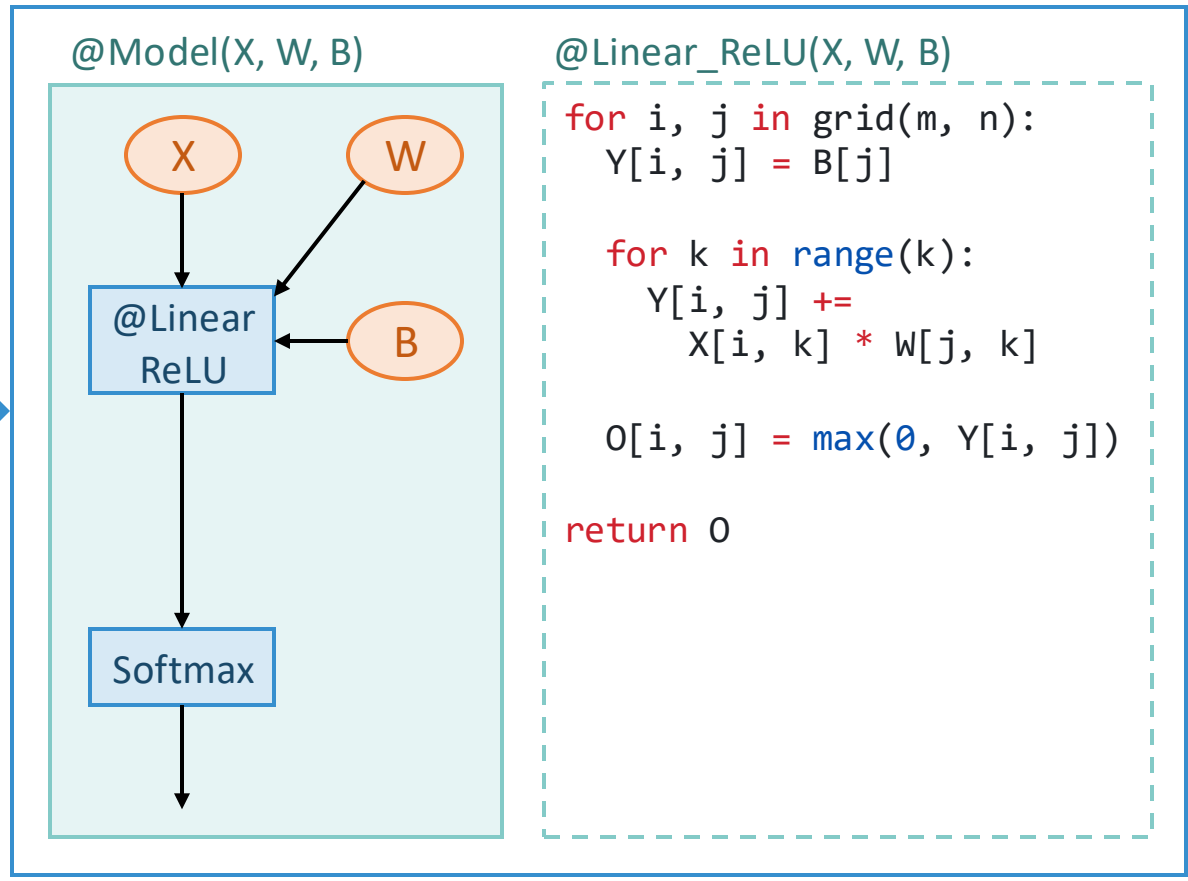


IRModule

Example Flow: Low Level Transformations

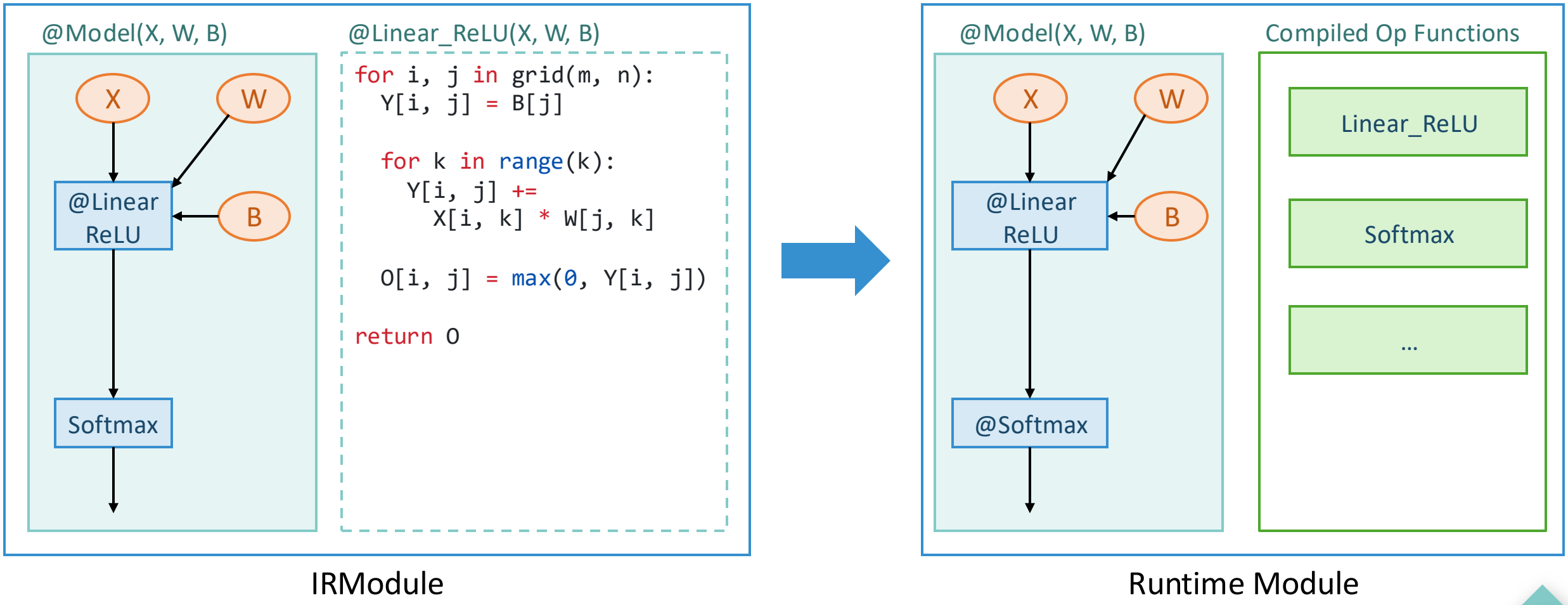


IRModule



IRModule

Example Flow: CodeGen and Execution



IRModule

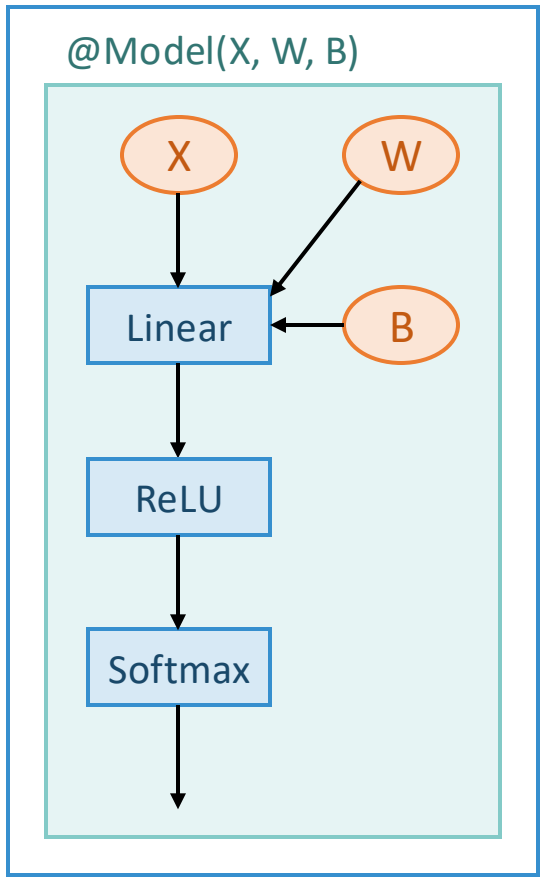
Runtime Module

Discussion

上海创智学院
Shanghai Innovation Institute

- What are possible ways to represent a function in ML Compiler
- The possible set of optimizations we can perform in each type of representations.

High-level IR and Optimizations



- Computation graph(or graph-like) representation
- Each node is a tensor operator(e.g. convolution)
- Can be transformed (e.g. fusion) and annotated (e.g. device placement)
- Most ML frameworks have this layer

Low-level Code Optimization

上海创智学院 Shanghai Innovation Institute

```
for y, x, k in grid(64, 64, 64):
    C[y, x] += A[y, k] * B[k, x]
```

```
for yo, xo, ko in grid(16, 16, 16):
  block
  for y, x, k in grid(4, 4, 4):
    C[by*4+y, bx*4+x] +=
      A[by*4+y, bk*4+k] * B[bk*4+k, bx*4+x]
```

Tensorized body (matmul4x4) isolated from the outer loop nests

Transform loops with tensorized operations

Tensorized Programs

```
for yo, xo, k in grid(4, 4, 16):
  for yi, xi in grid(4, 4):
    block
    Tensorized body (matmul4x4)
```

Key Ideas

- Divide problem into sub-tensor computation blocks
- Generalize loop optimization for tensorized computation
- Combination of the above approaches in any order

Option 0: Tensorized body (matmul4x4)

```
for y, x in grid(4, 4):
  C[by*16+y, bx*16+x] +=
    accel.dot(A[by*4+y, bk*4:bk*4+4],
              B[bk*4:bk*4+4, bx*4+x])
```

Option 1: Tensorized body (matmul4x4)

```
for y, x, k in grid(4, 4, 4):
  C[by*4+y, bx*4+x] +=
    A[by*4+y, bk*4+k] *
    B[bk*4+k, bx*4+x]
```

Acknowledgement

The development of this course, including its structure, content, and accompanying presentation slides, has been significantly influenced and inspired by the excellent work of instructors and institutions who have shared their materials openly. We wish to extend our sincere acknowledgement and gratitude to the following courses, which served as invaluable references and a source of pedagogical inspiration:

- Machine Learning Systems[15-442/15-642], by **Tianqi Chen** and **Zhihao Jia** at **CMU**.
- Advanced Topics in Machine Learning (Systems)[CS6216], by **Yao Lu** at **NUS**

While these materials provided a foundational blueprint and a wealth of insightful examples, all content herein has been adapted, modified, and curated to meet the specific learning objectives of our curriculum. Any errors, omissions, or shortcomings found in these course materials are entirely our own responsibility. We are profoundly grateful for the contributions of the educators listed above, whose dedication to teaching and knowledge-sharing has made the creation of this course possible.

System for Artificial Intelligence

Thanks

Siyuan Feng
Shanghai Innovation Institute