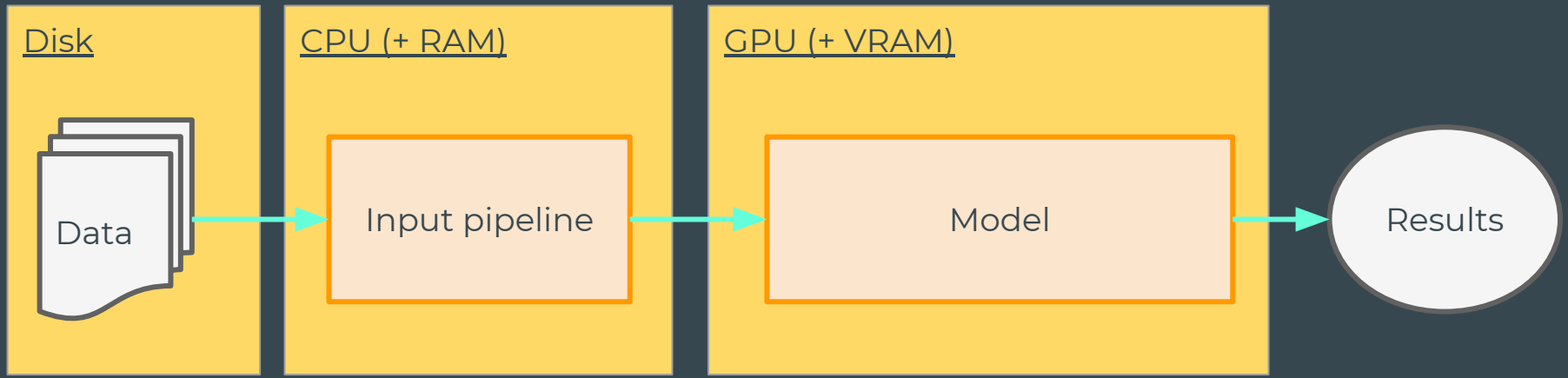
AIDL: PROJECTS

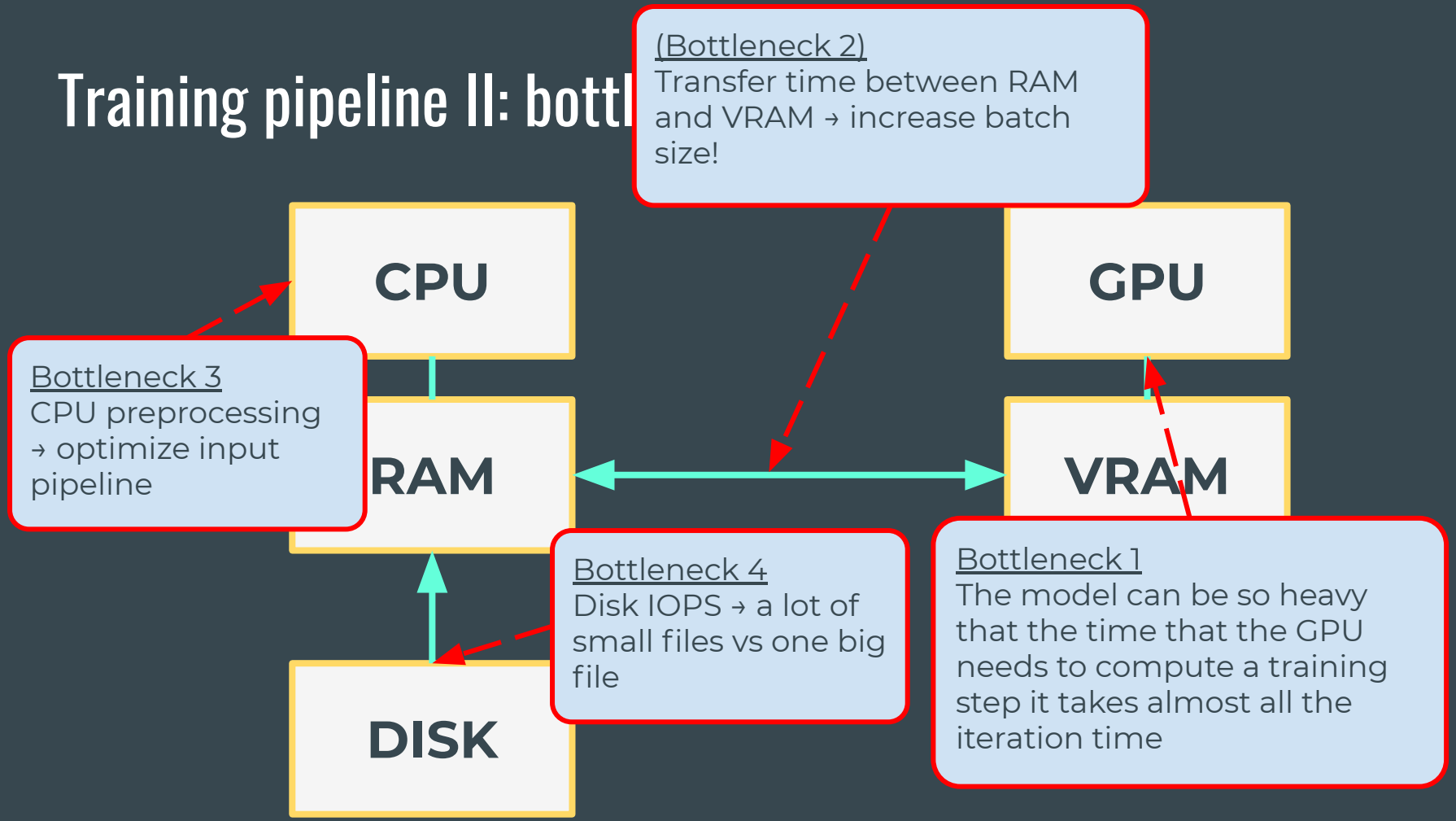
Session 7 (2019/06/11): Full model implementation II

Previously on *AIDL: Projects*...

Training pipeline I: overview



Training pipeline II: bottl



Training steps: cookbook!

1. Get to know your data and its format
2. Build your input pipeline to ingest the dataset → try to run it in an infinite loop and monitor the iteration time, trying to reduce it
3. Implement the most basic model you can and try to train it
4. Look after an overfit!! (small subset of data)
5. Make yourself a baseline
6. Improve the model analysing the error gaps between bayes error-train and train-val

Feeding data: main ops

Dataset:

- from_generator, from_tensor_slices: create dataset from source data
- shuffle: exactly that
- repeat: how many times should the dataset be repeated (epochs!)
- map: transform the samples
- batch: group samples into batches
- prefetch: create a buffer of items for improved performance
- make_one_shot_iterator: create the iterator of this dataset

Iterator:

- get_next: obtain the reference to the next samples of the dataset

Feeding data: example

```
dataset = tf.data.Dataset.range(100)
    .shuffle(10)
    .repeat(num_epochs)
    .map(parse_fn, num_parallel_calls=10)
    .batch(batch_size)
    .prefetch(prefetch_batches)

iterator = dataset.make_one_shot_iterator()
batch = iterator.get_next()
x, y = batch     # Instead of the placeholders!!!
```

Links:

https://www.tensorflow.org/guide/datasets
https://www.tensorflow.org/api_docs/python/tf/data/Dataset
https://www.tensorflow.org/guide/performance/datasets

Input pipeline II

Mapping function

```
dataset = tf.data.Dataset...
    .map(parse_fn, num_parallel_calls=10)
    ...

def parse_fn(image_path, label):
    raw_image = tf.read_file(image_path)
    image = tf.image.decode_jpeg(raw_image, channels=3)
    return image, label
```

How to: DL with Tensorflow!

Layers

Different ways to create common layers:

- tf.nn: Low level API. You need to create and provide the variables yourself
- tf.layers: High level TF API. It creates everything for you!
- tf.keras.layers: Good-old Keras style layers

Useful links:

https://www.tensorflow.org/api_docs/python/tf/nn

https://www.tensorflow.org/api_docs/python/tf/layers

https://www.tensorflow.org/api_docs/python/tf/keras/layers

Example: Conv2D

Low level API

```
x = …
kernel = tf.get_variable(‘kernel’, shape=[11, 11, 3, 96], dtype=tf.float32)
biases = tf.get_variable(‘biases’, shape=[96], dtype=tf.float32)
output = tf.nn.conv2d(x, kernel, strides=[1, 4, 4, 1], padding="SAME",
name="conv1")
output += biases
output = tf.nn.relu(output)
```

High level API

```
x = …
output = tf.layers.conv2d(x, filters=96, kernel_size=(11, 11), strides=(4,
4), padding=”SAME”, activation=tf.nn.relu, name=”conv1”)
```

Keras

```
x = …
output = tf.keras.layers.Conv2d(filters=96, kernel_size=(11, 11), strides=(4,
4), padding=”SAME”, activation=’relu’, name=”conv1”)(x)
```

Exercise V

Implement AlexNet's input pipeline

- Implement input pipeline for AlexNet

Cheatsheet

Management

- tf.Session:
  - run
- tf.Graph:
  - as_default_graph
- tf.variable_scope
- tf.name_scope
- tf.summary.FileWriter

Input pipeline

- tf.data.Dataset:
  - from_generator
  - from_tensor_slices
  - shuffle
  - repeat
  - map
  - batch
  - prefetch
  - make_one_shot_iterator
- tf.data.Iterator
  - get_next
- tf.image
  - tf.image.decode_jpeg
- tf.read_file

Models

- tf.layers
  - tf.layers.conv2d
- tf.nn
- tf.keras.layers
- tf.train:
  - tf.train.AdamOptimizer

Questions?