Barcelona Supercomputing Center
Centro Nacional de Supercomputación

EXCELENCIA SEVERO OCHOA

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

intel

AI and Predictive Analytics in Data-Center Environments

Distributed Computing using Spark

Distributing Neural Networks using Spark and Intel BigDL

Josep Ll. Berral @BSC

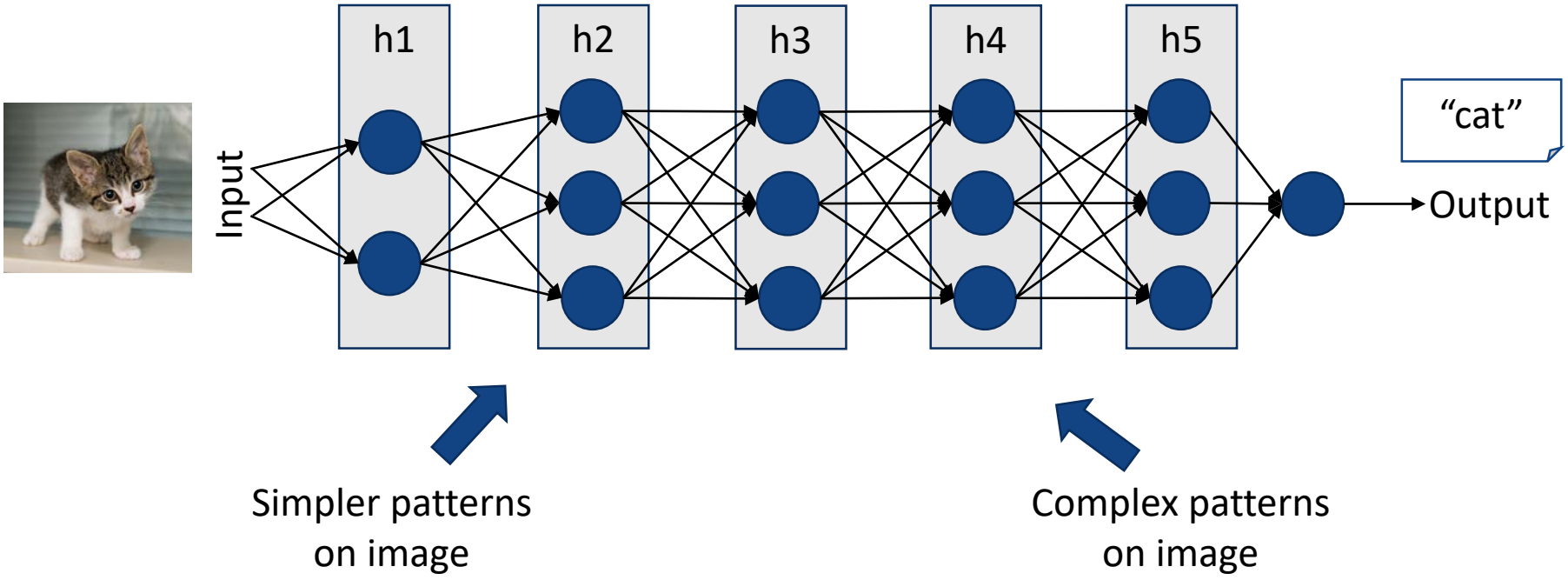
Intel Academic Education Mindshare Initiative for AI

Introduction

“There are solutions for distributing Deep Learning, and some optimized for leveraging specific computing architectures”

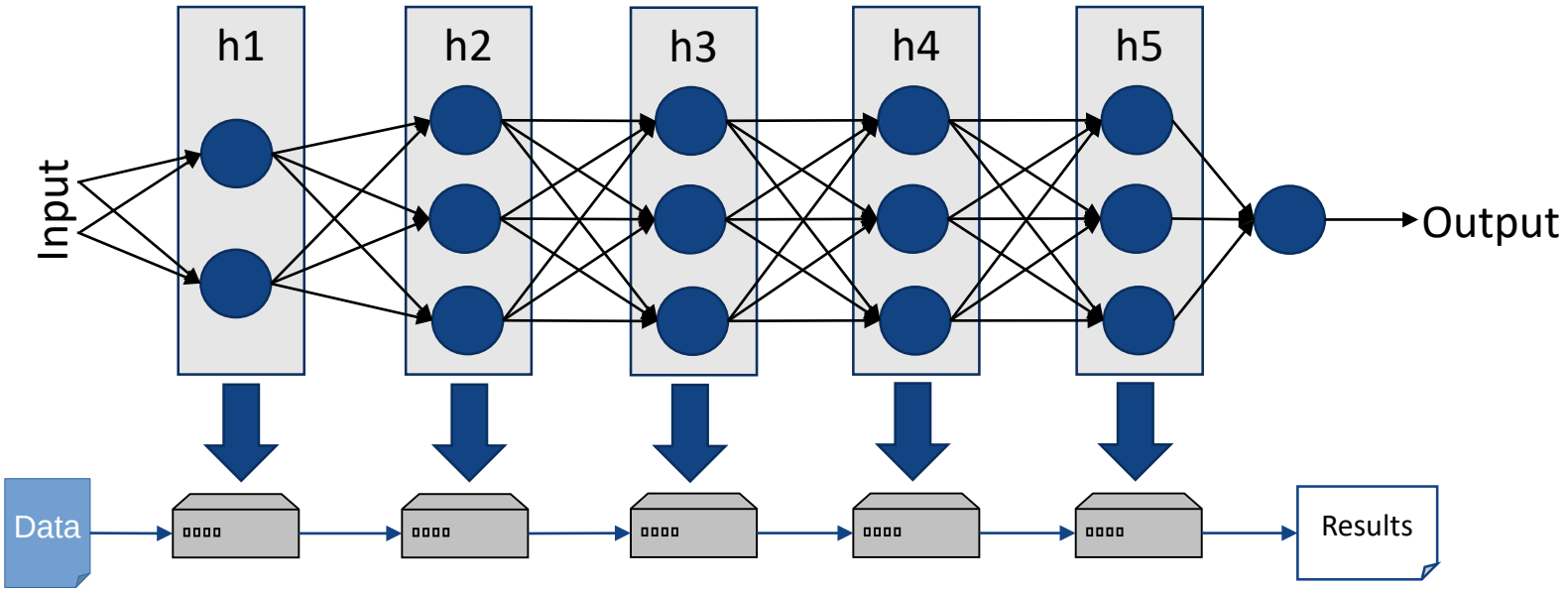
Deep Learning

- Multiple Layers on Neural Networks
  - Model abstract patterns on data
  - … on different levels
  - … with several “stages”



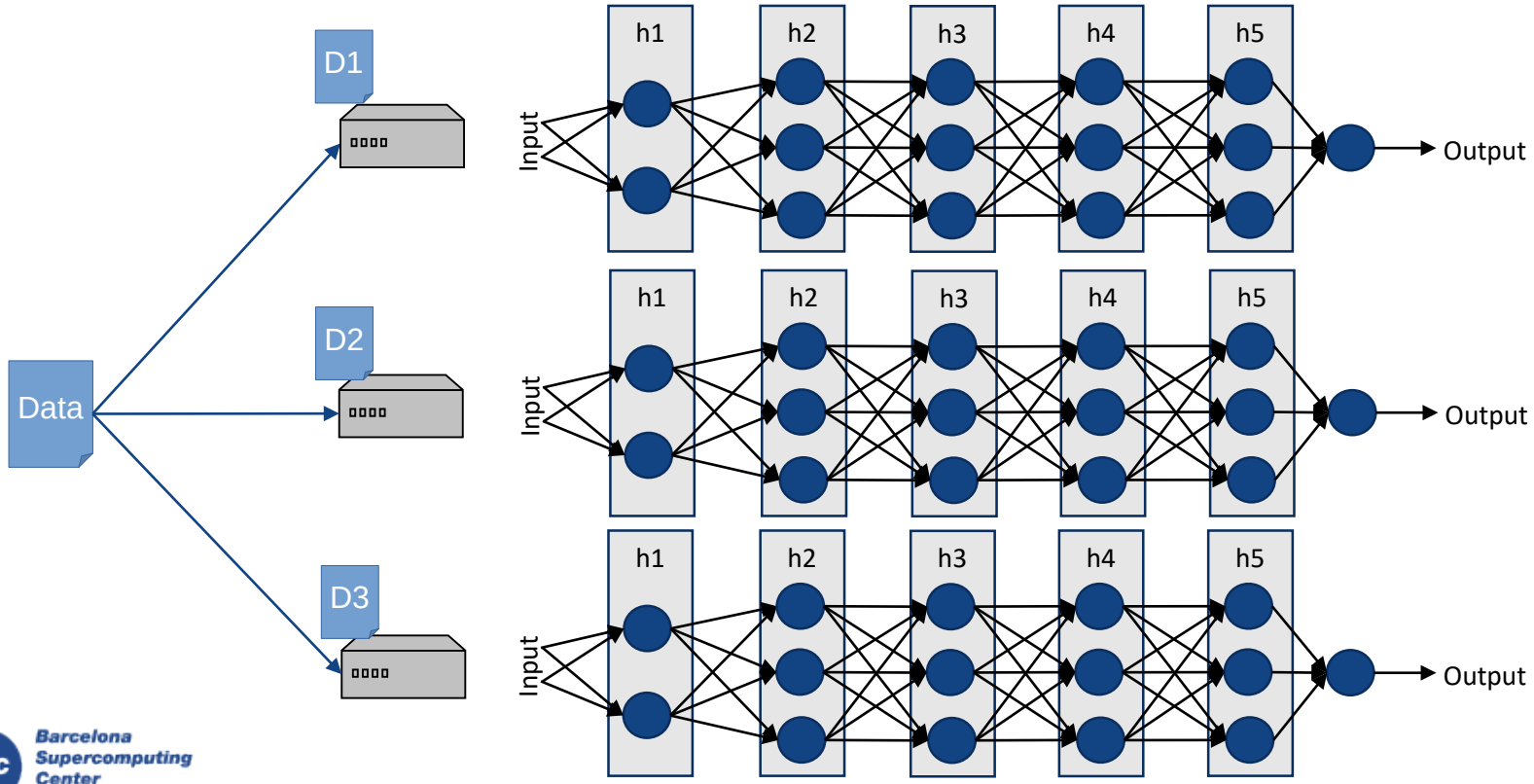
Deep Learning

- "How to distribute those stages?"
  - Do we distribute layers?
    - How do training adjustments communicate?



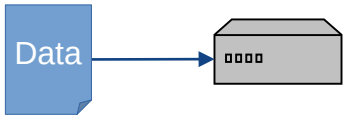
Deep Learning

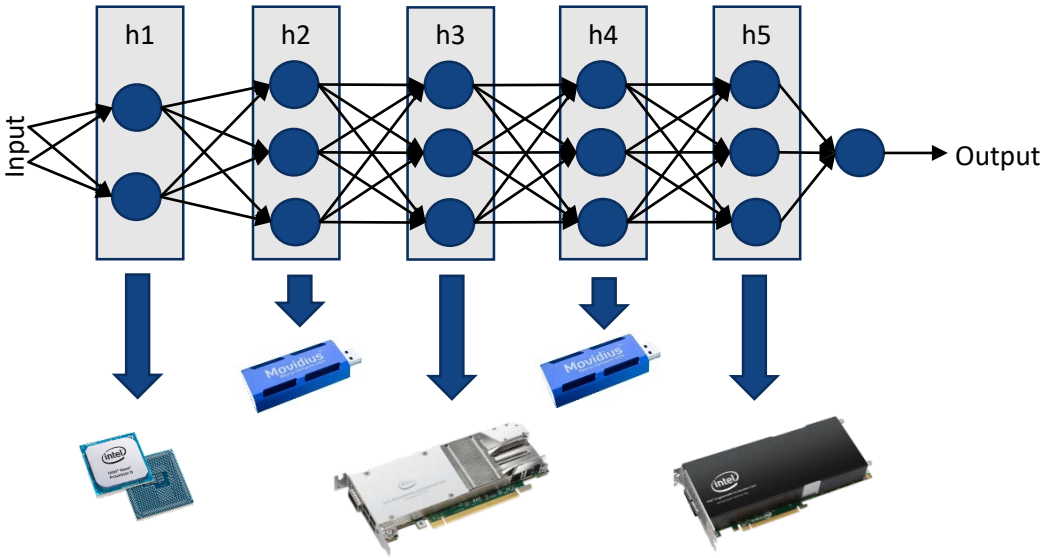
- "How to distribute those stages?"
  - Do we distribute data?
    - How do we distribute models?
    - How do we aggregate models?



Deep Learning

- "How to distribute those stages?"
  - Do we distribute …stuff?
    - Tensor units
    - FPGAs
    - GPUs
    - Accelerators





Architecture Awareness

- Platform Aware of Architecture
  - A platform requires High Performance
  - Machinery/Hardware/Components companies:
    - … create versions/libraries for such platforms
    - … optimize them for their hardware
    - … offer them as an optimized alternative
  - This is usual at Intel
    - E.g. Compilers
      - GCC (GNU multi-processor) → ICC (Optimized for Intel processors)

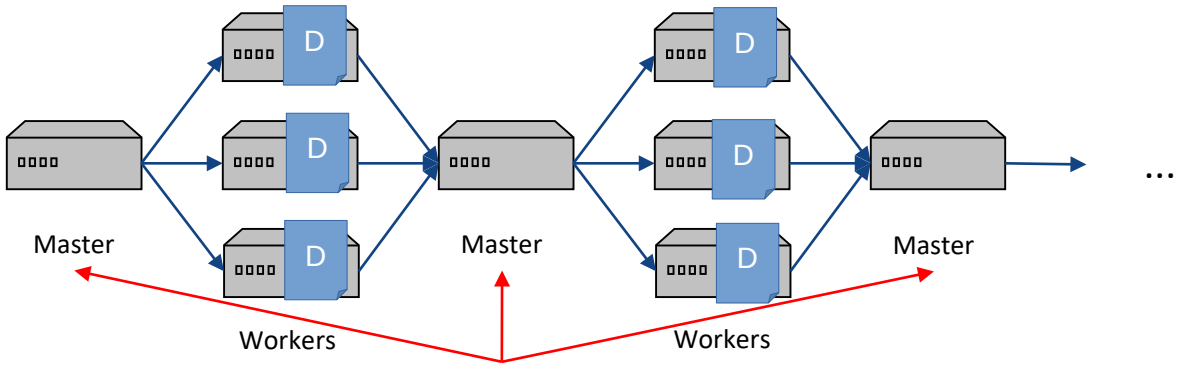
Intel BigDL

- Tensor processing platforms
  - E.g. Torch
    - … popular platform for tensor processing
    - … has a common syntax for programming (+python)
    - … oriented to Neural Networks and Deep Learning
    - … can be mounted on Spark offering DL functionalities

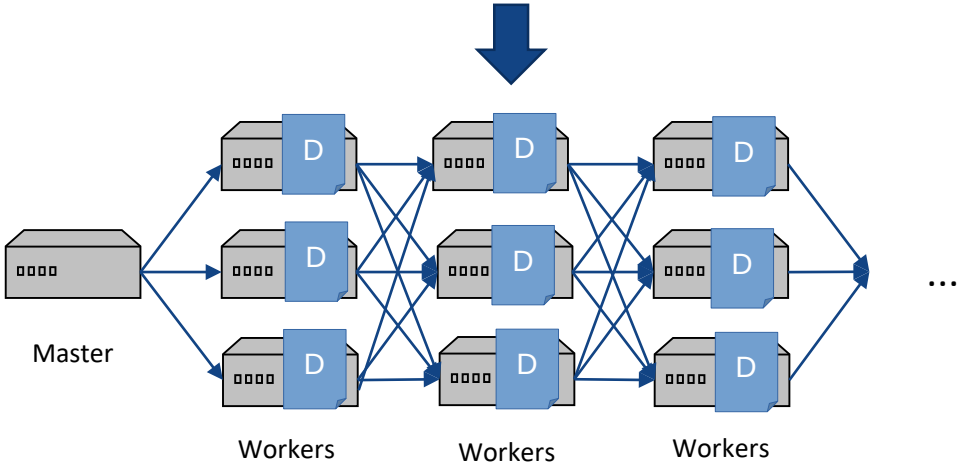
  - Intel BigDL
    - Uses a syntax identical to Torch
    - … it is optimized for Intel technologies
      - Intel MKL (Math Kernel Library)
      - Multi-Threading enabled
    - … it is provided as Spark libraries
    - … tries to integrate better into Spark distribution

Intel BigDL

- Shuffling Minimization
  - Beware of communication between workers → Minimize!



Manages the Shuffling



Direct Sharing of Data

Programming a NN

- Spark + BigDL

1. Load the BigDL libraries and components
   - If we loaded *spark.ml* elements, now we load *bigdl* elements

```
from bigdl.nn.layer import *
```

2. Define Layers
   - E.g. a NN with 1 Hidden layer, with 5 input features, 2 output classes, and a Log SoftMax

```
lr_seq = Sequential()
lr_seq.add(Linear(5, 2))
lr_seq.add(LogSoftMax());
```

3. Define the Optimizer

```
from bigdl.nn.criterion import *
from bigdl.optim.optimizer import *
optimizer = Optimizer(
          model = lr_seq,
          training_rdd = train_rdd,
          criterion = ClassNLLCriterion(),
          end_trigger = MaxEpoch(20),
          optim_method = SGD(learningrate=0.05),
          batch_size = 16)
```

Programming a NN

- Spark + BigDL

4. Set some validation

```
optimizer.set_validation(
    batch_size = 16,
    val_rdd = validation_rdd,
    trigger = EveryEpoch(),
    val_method = [Loss()])
```

5. Then fit the model

```
optimizer.optimize();
```

6. Also perform evaluation

```
test_results = lr_seq.evaluate(
       test_rdd,
       batch_size = 16,
       [Loss()])
```

Programming a NN

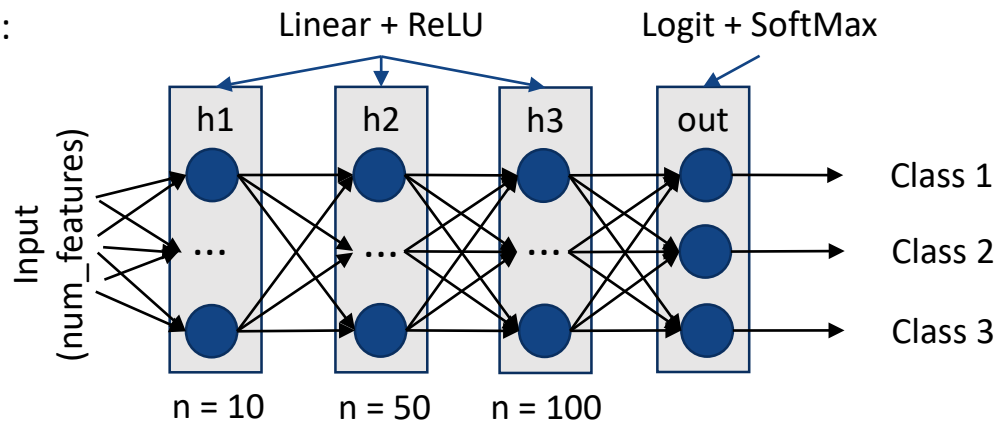
- Architecture Example:

```
from bigdl.nn.layer import *

num_hidden = [10, 50, 100]
num_classes = 3

ff_seq = Sequential()
ff_seq.add(Linear(num_features, num_hidden[0]))
ff_seq.add(ReLU())
ff_seq.add(Linear(num_hidden[0], num_hidden[1]))
ff_seq.add(ReLU())
ff_seq.add(Linear(num_hidden[1], num_hidden[2]))
ff_seq.add(ReLU())
ff_seq.add(Linear(num_hidden[2], num_classes))
ff_seq.add(LogSoftMax());
```

- Corresponding to:



Hands-On

- Next, let’s move to the Hands-On
  - Play with BigDL
  - See some NN examples

```
$ $SPARK_HOME/bin/pyspark
Python 3.6.5 (default, Mar 31 2018, 19:45:04) [GCC] on linux
Type "help", "copyright", "credits" or "license" for more information.
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use
setLogLevel(newLevel).
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\ version 2.4.1
      /_/
Using Python version 3.6.5 (default, Mar 31 2018 19:45:04)
SparkSession available as 'spark'.
>>>
```

Summary

- Distributing Neural Networks
- Tensor processing frameworks
  - Intel BigDL framework
  - Architecture Aware and Optimization
- Programming a Deep Neural Network
  - Training
  - Evaluation
  - Induction