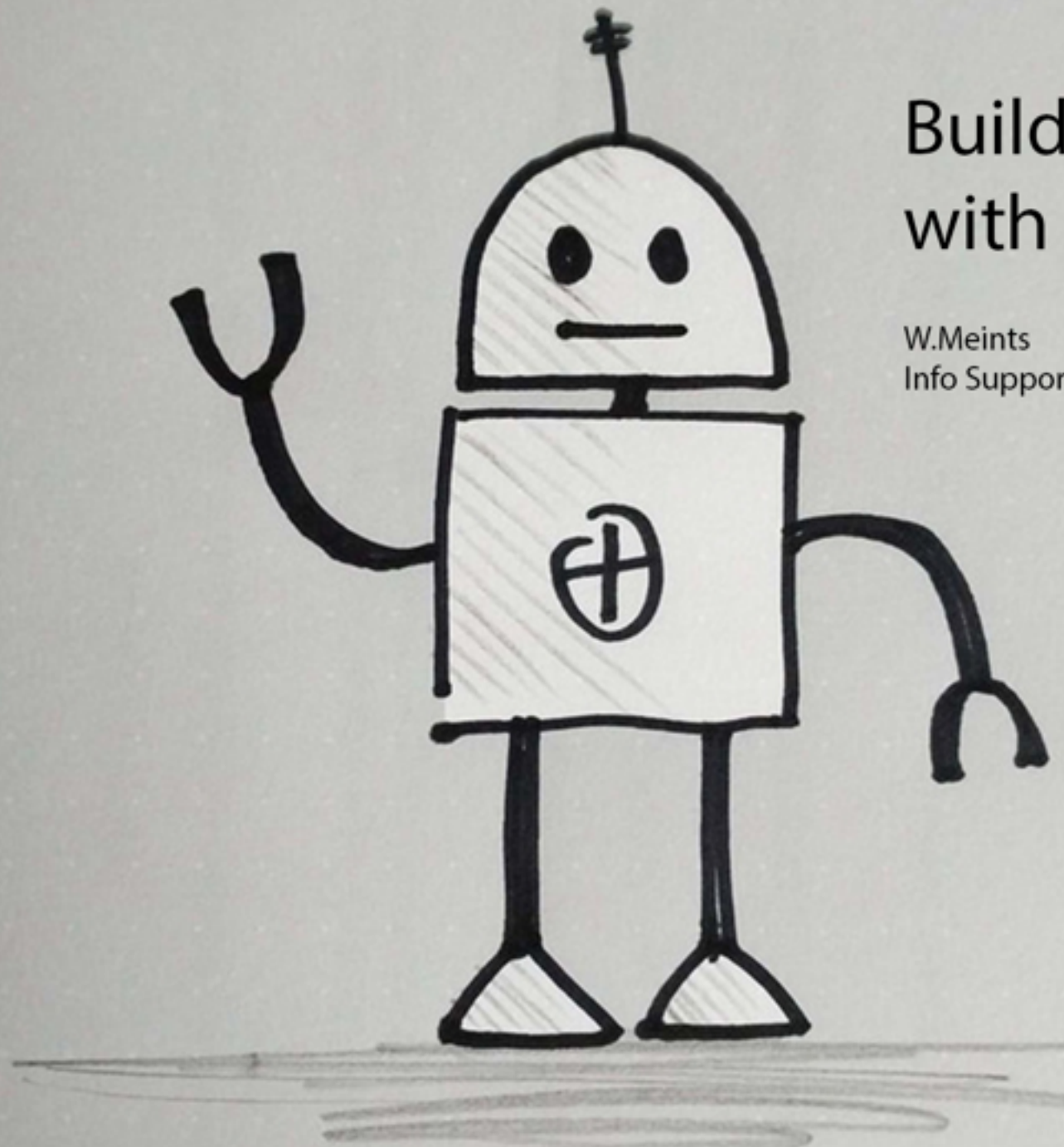


Build a Q&A bot with DeepLearning4J

W.Meints
Info Support

goto; amsterdam

Please

Ask questions through the app

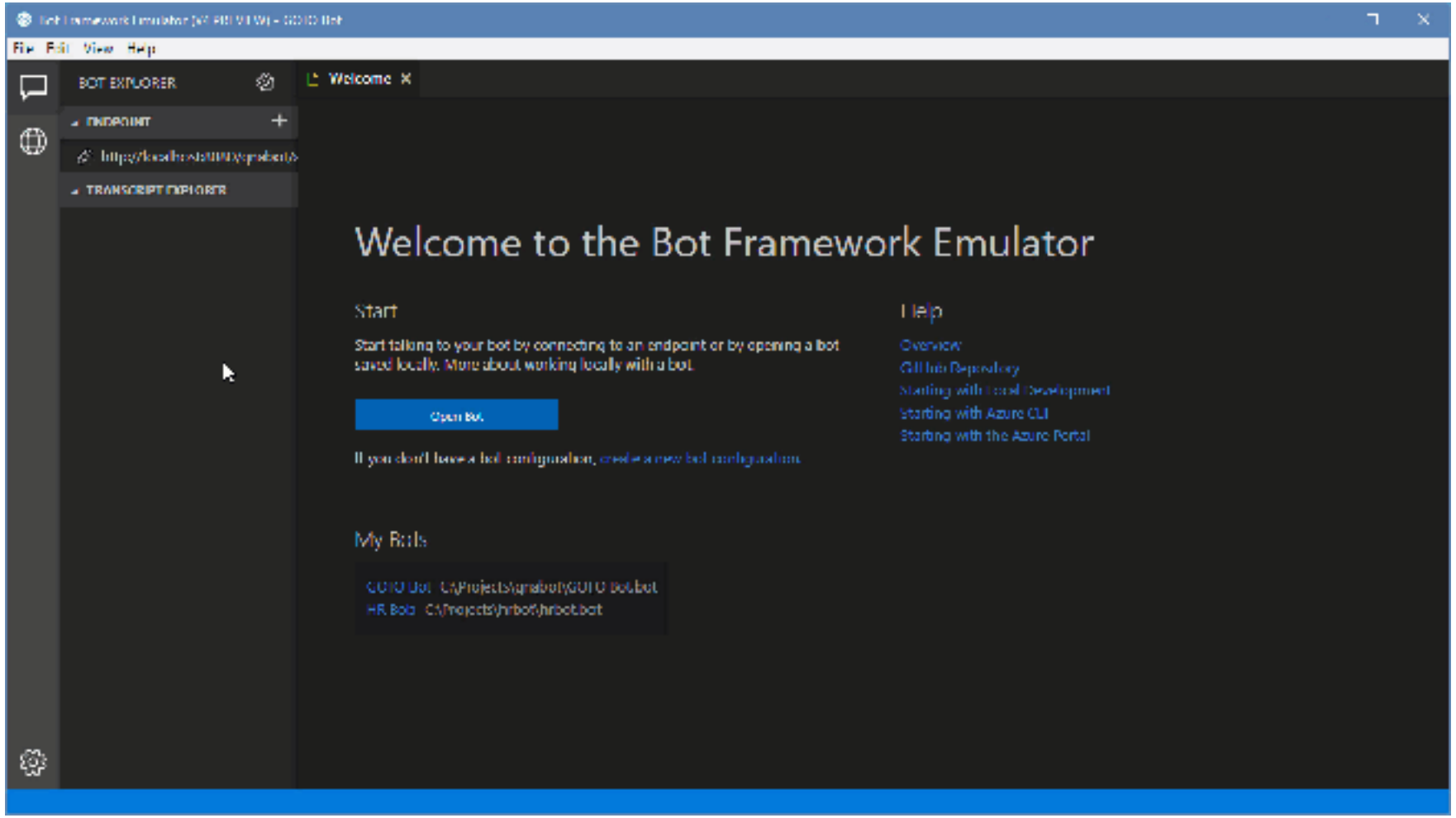
Rate Session

Thank you!

Agenda

- Setting the stage
- Building neural networks with DeepLearning4J
- How to get started yourself

Setting the stage


File Edit View Help
BOT EXPLORER
Welcome
ENDPOINT
TRANSCRIPT EXPLORER
Welcome to the Bot Framework Emulator
Start
Start talking to your bot by connecting to an endpoint or by opening a bot saved locally. More about working locally with a bot.
Open Bot
If you don't have a bot configuration, create a new bot configuration.
Help
Overview
GitHub Repository
Starting with Local Development
Starting with Azure CLI
Starting with the Azure Portal
My Bots
GDTO Bot C:\Projects\qnabot\GDTO Bot.bot
HR Bot C:\Projects\hrbot\hrbot.bot

How does the bot find an answer to your question?

A search problem

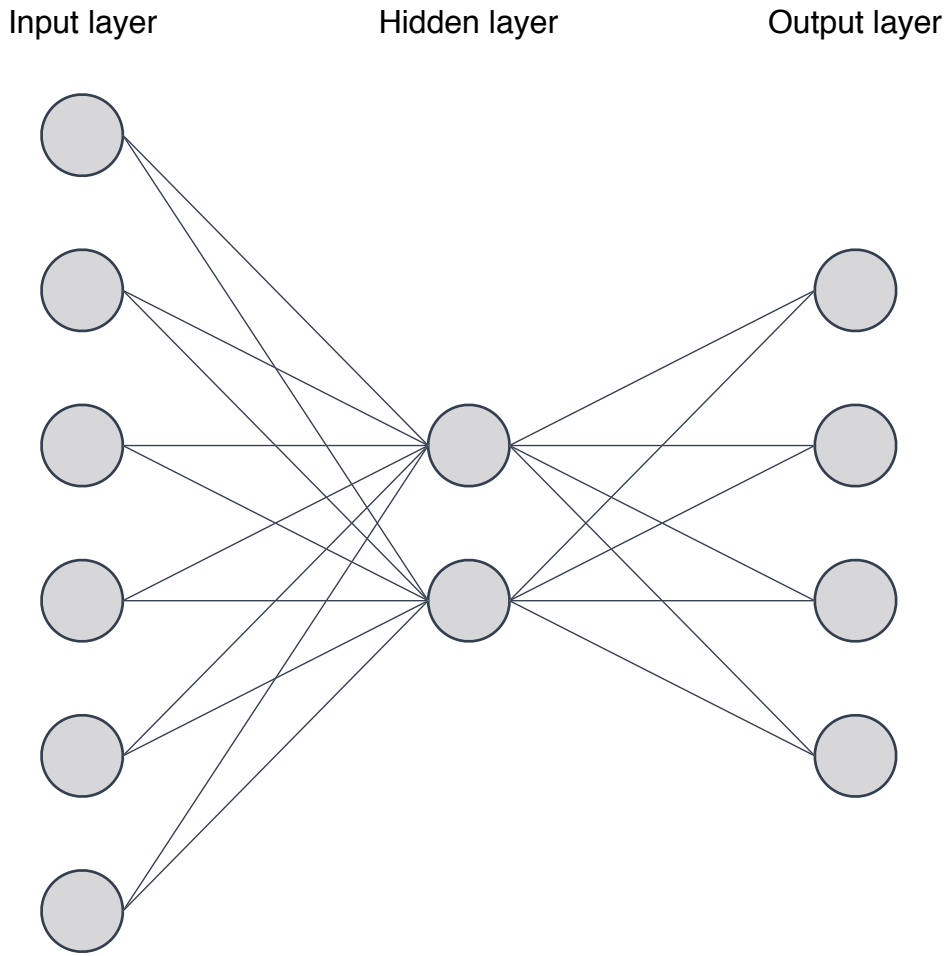
- The question somehow relates to a specific answer.

- Two parts to the answer:
  - We need to finger print the questions
  - That fingerprint then needs to be related to a specific answer

Let’s teach the computer how

By building a neural network

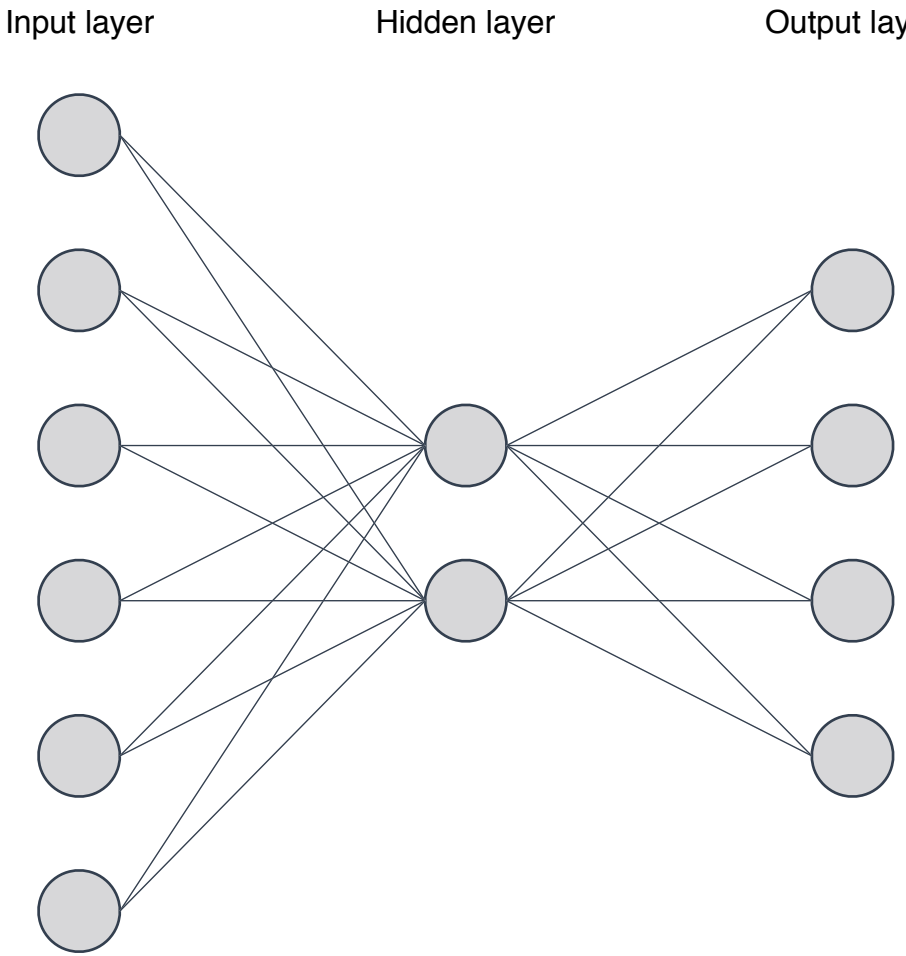
Quick introduction to neural networks



Quick introduction to neural networks

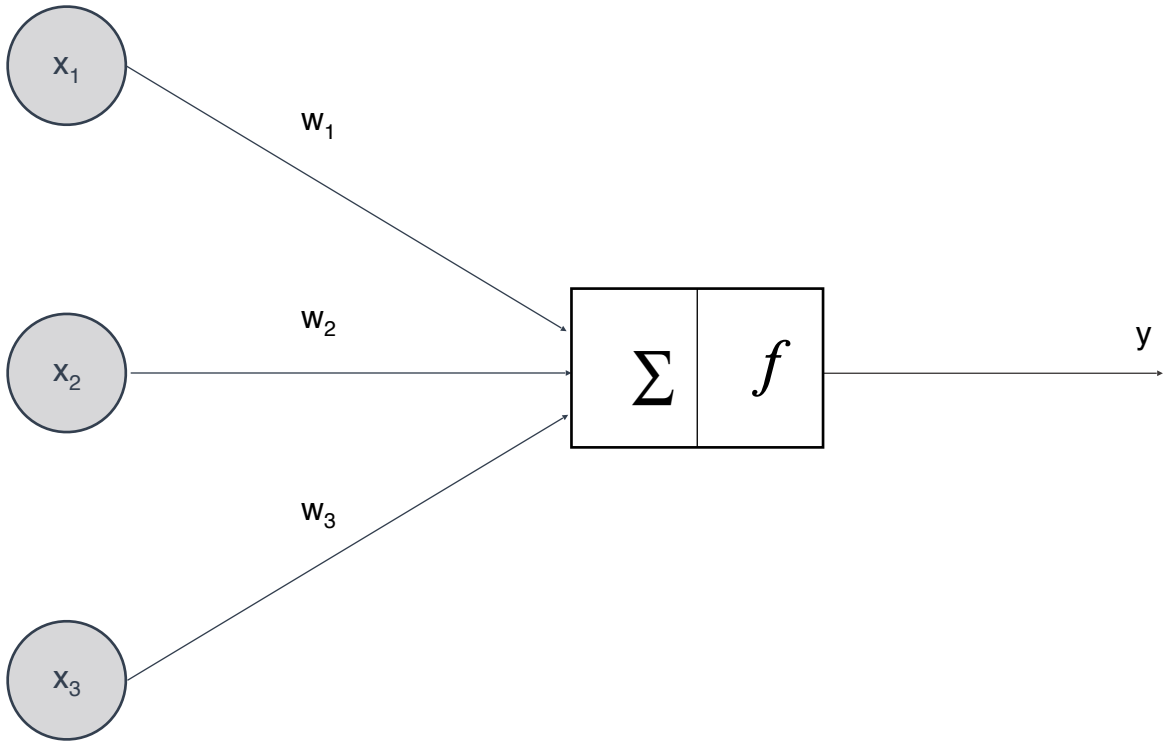
Input

$$\begin{bmatrix} 0.0 \\ \cdots \\ 1.0 \end{bmatrix}$$



Prediction

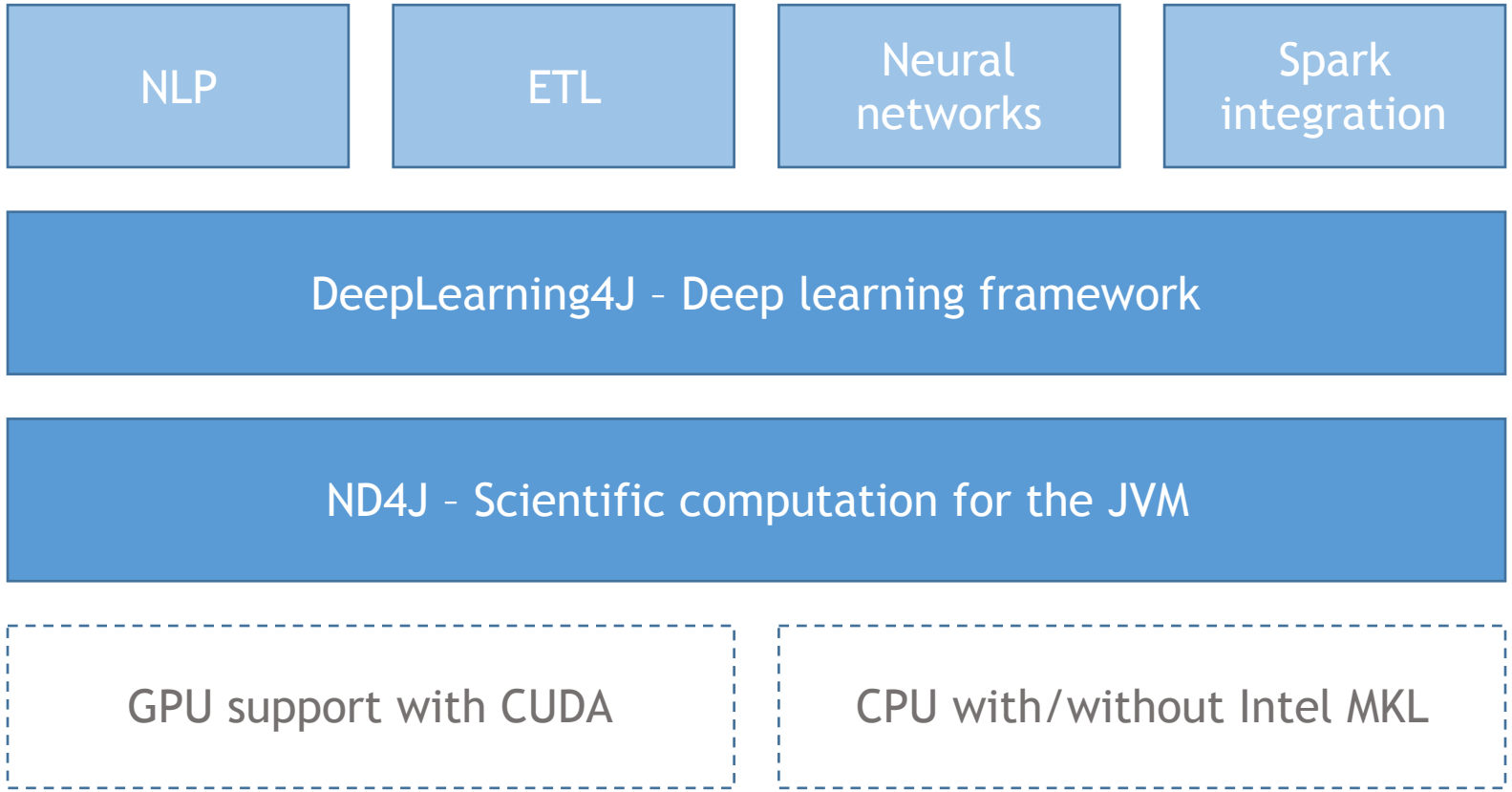
$$\begin{bmatrix} 0.73 \\ \cdots \\ 1.0 \end{bmatrix}$$



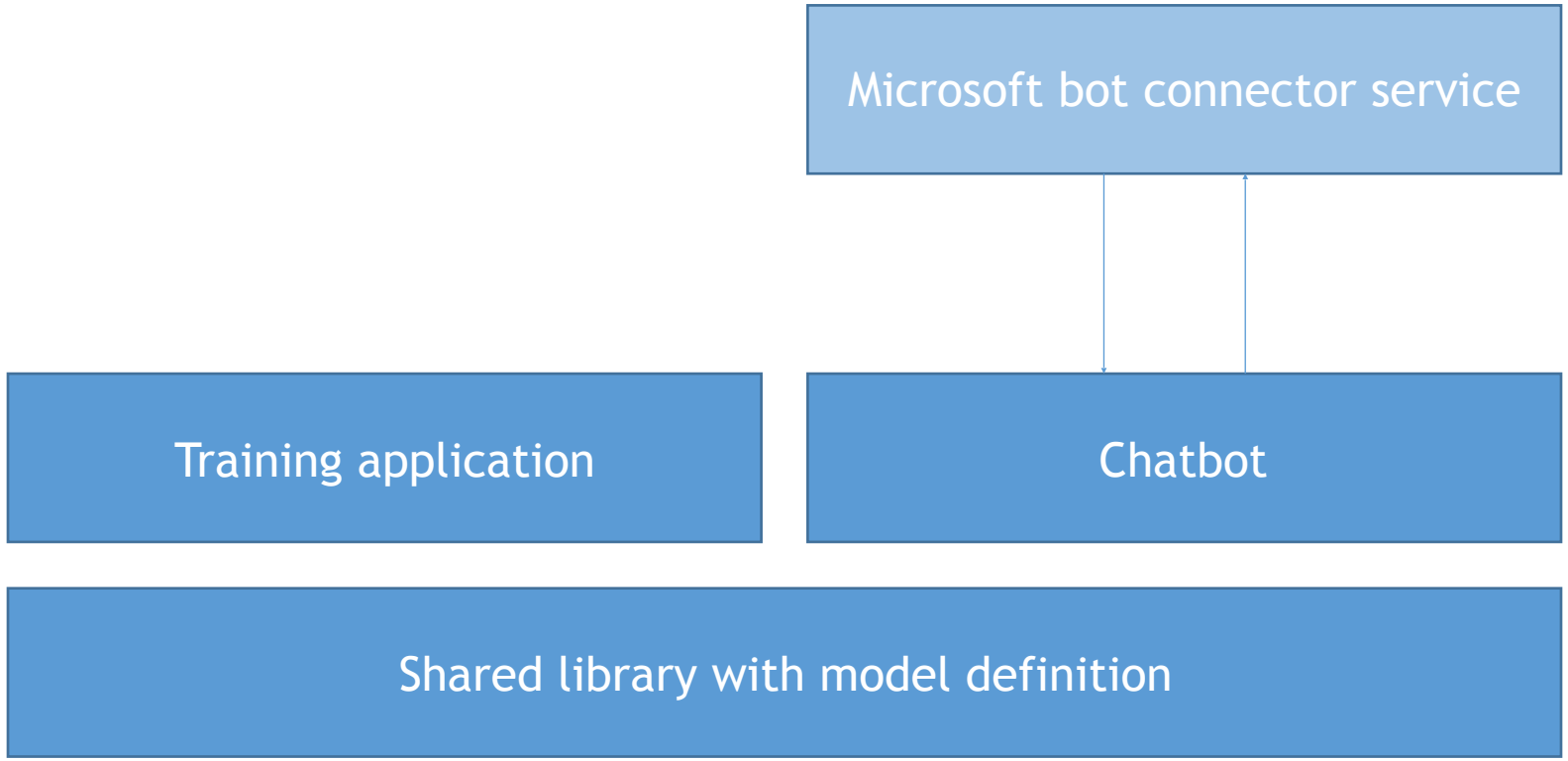
$$y = f(\sum_{i=1}^{n}(w_i * x_i))$$

Build a neural network with DeepLearning4J

DL4J


NLP
ETL
Neural networks
Spark integration
DeepLearning4J – Deep learning framework
ND4J – Scientific computation for the JVM
GPU support with CUDA
CPU with/without Intel MKL

Application architecture



We’re going to follow a 4 step recipe

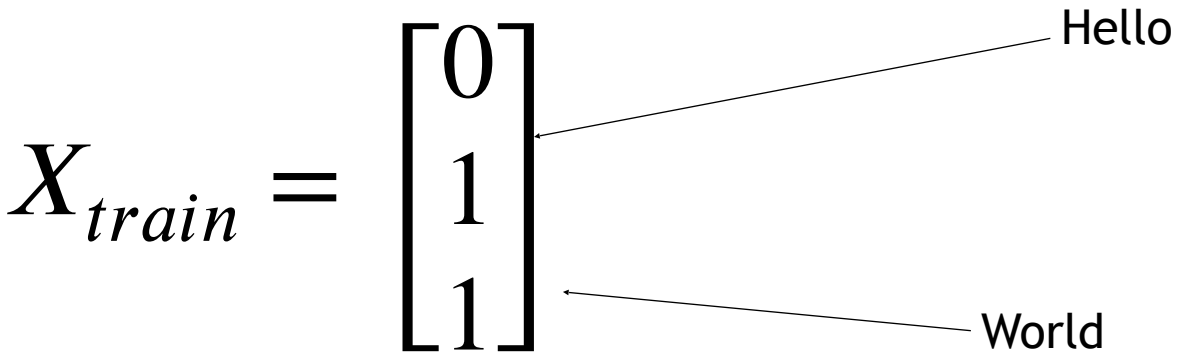
- Step 1: Encode the input
- Step 2: Map the answers
- Step 3: Build the neural network
- Step 4: Train the neural network

Step 1: Encode the input

Encoding text as a bag of words

Three steps:

1. Create a vector equal to the size of your vocabulary
2. Count word ocurrences
3. Assign the count each word a unique index in the vector

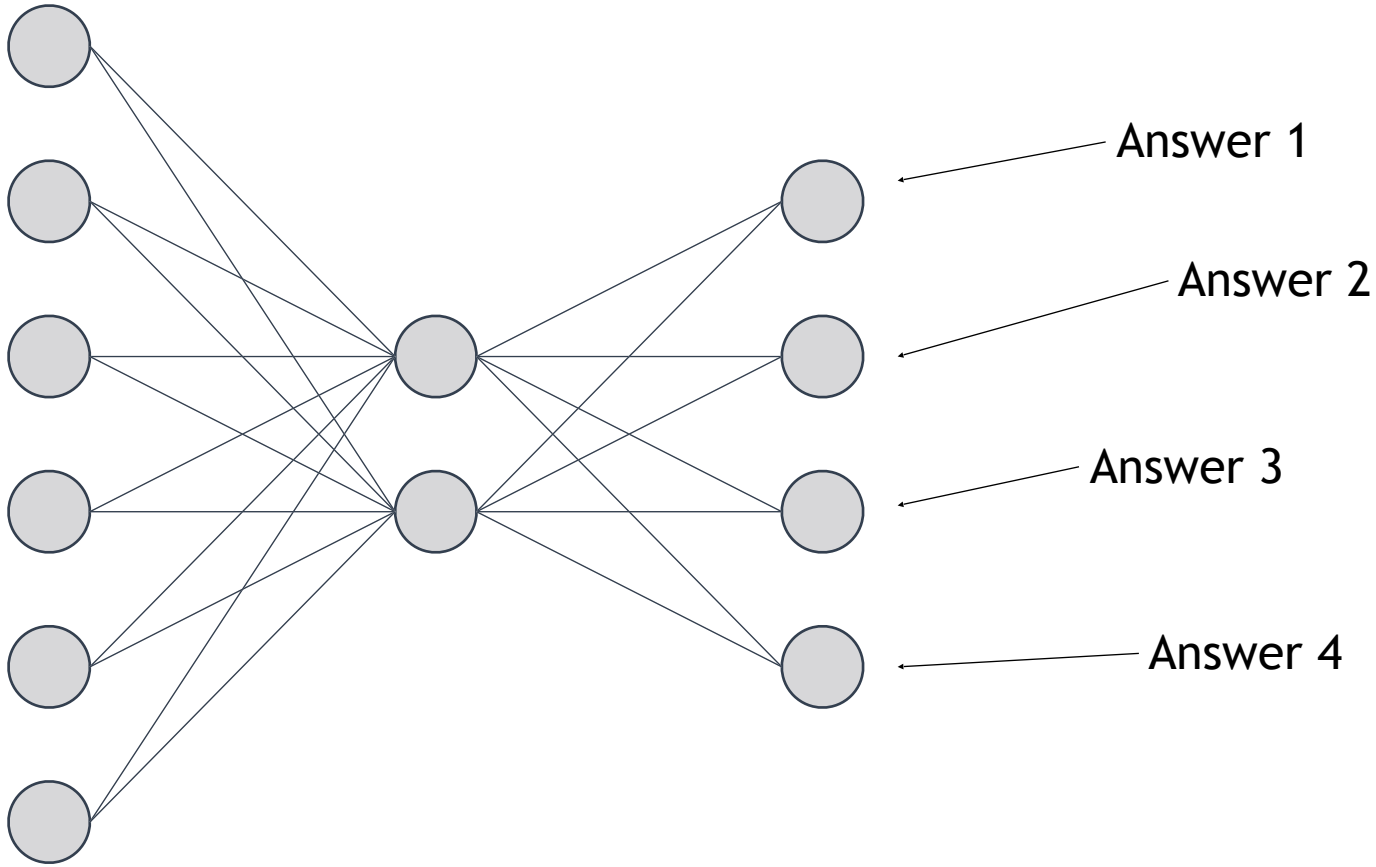

$X_{train} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$
Hello
World

Create a bag of words in DL4J

```
TokenizerFactory tokenizerFactory = new DefaultTokenizerFactory();
tokenizerFactory.setTokenPreProcessor(new CommonPreprocessor());

// This vectorizer uses the TF-IDF algorithm to produce
// a unique fingerprint for every question we feed it.
BagOfWordsVectorizer vectorizer = new BagOfWordsVectorizer.Builder()
  .setTokenizerFactory(tokenizerFactory)
  .setIterator(new CSVSentenceIterator(inputFile))
  .build();
```

Step 2: Encode answers

Encode answers

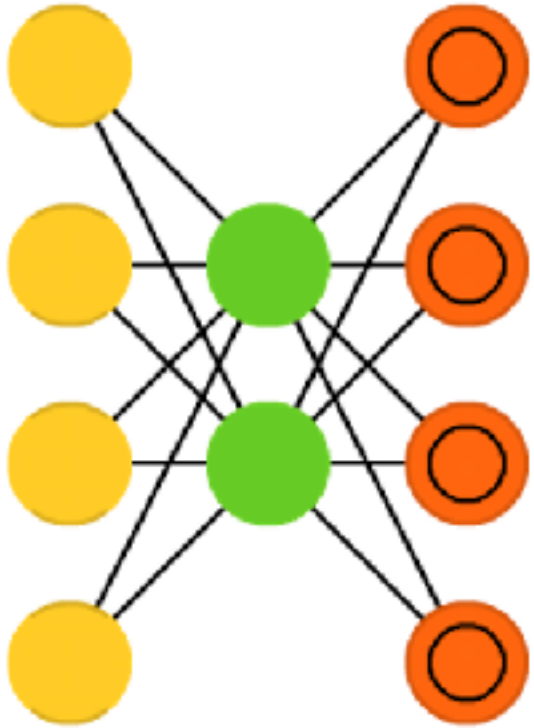


Map neurons to answers

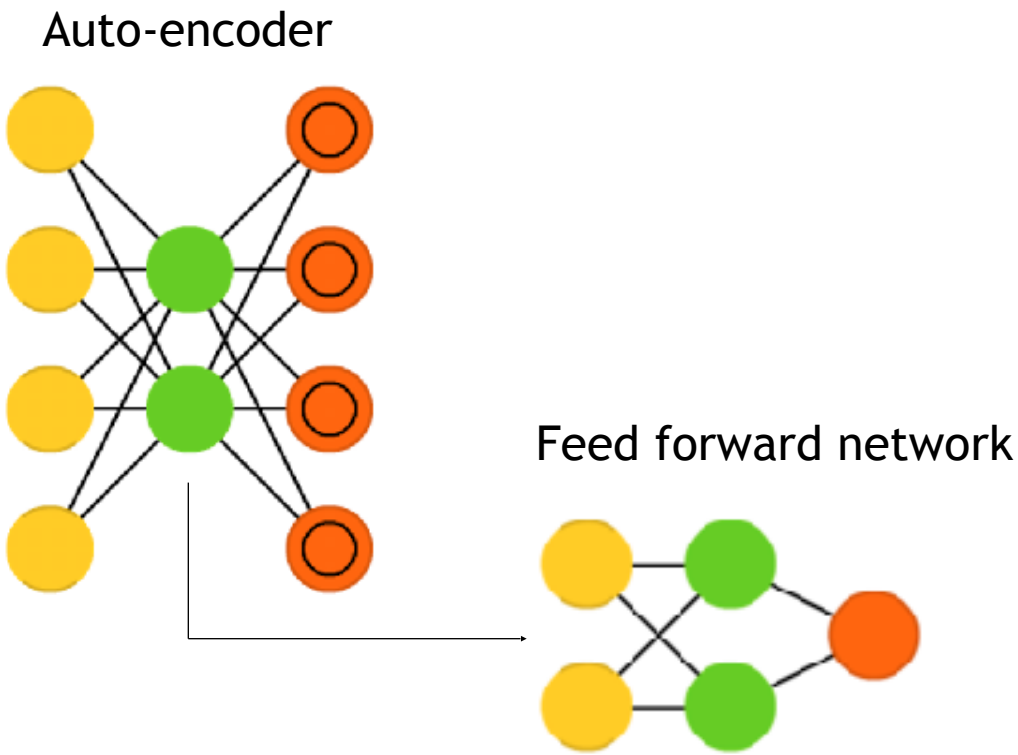
```
try (CSVRecordReader reader = new CSVRecordReader(1, ',')) {
  reader.initialize(new FileSplit(inputFile));

  Map<Integer, String> answers = new HashMap<>();

  while(reader.hasNext()) {
    List<Writable> record = reader.next();

    // Note: The answer index needs a -1, so that we get an offset mapping.
    answers.put(record.get(0).toInt() - 1, record.get(1).toString());
  }

  return answers;
}
```

Step 3: Build the neural network

Fingerprint the data with an auto-encoder



Relate the fingerprint to an answer



```
MultiLayerConfiguration networkConfiguration = new
NeuralNetConfiguration.Builder()
  .seed(1337)
  .list()
    .layer(0, new VariationalAutoencoder.Builder()
      .nIn(inputLayerSize).nOut(1024)
      .encoderLayerSizes(1024, 512, 256, 128)
      .decoderLayerSizes(128, 256, 512, 1024)
      .lossFunction(Activation.RELU, LossFunctions.LossFunction.MSE)
      .gradientNormalization(GradientNormalization.ClipElementWiseAbsoluteValu
e)
      .dropOut(0.8)
      .build())
    .layer(1, new OutputLayer.Builder()
      .nIn(1024).nOut(outputLayerSize)
      .activation(Activation.SOFTMAX)
      .lossFunction(LossFunctions.LossFunction.NEGATIVELOGLIKELIHOOD)
      .build())
  .updater(new RmsProp(0.01))
  .pretrain(true)
  .backprop(true)
  .build();
```

```
MultiLayerNetwork network = new MultiLayerNetwork(networkConfiguration);
network.setListeners(new ScoreIterationListener(1));
network.init();
```

Step 4: Train the neural network

The final training application

```
Map<Integer, String> answers = AnswersMappingFactory.create(
  new File("data/answers.csv"));

TextVectorizer vectorizer = QuestionVectorizerFactory.create(
  new File("data/"));

vectorizer.fit();
vectorizer.save(new File("model/vectorizer.bin"));

QuestionClassifier classifier = QuestionClassifierFactory.create(
  vectorizer, answers);

classifier.fit(new File("data/questions_train.csv"));
classifier.save(new File("model/classifier.bin"));
```

Making a prediction

Inside the bot framework adapter

```
String replyText =
classifier.predict(context.activity().text());
```

At neural network level

```
INDArray prediction = network.output(vectorizer.transform(text));
int answerIndex = prediction.argMax(1).getInt(0,0);

return answers.get(answerIndex);
```

How to get started yourself

You too can use deep learning

- Three tips
  - Explore the model zoo
  - Starts with small experiments and expand if it has potential
  - Choose a framework that allows you to experiment quickly

Useful resources

- The code:
  https://github.com/wmeints/qna-bot
- The model zoo:
  http://www.asimovinstitute.org/neural-network-zoo/
- DeepLearning4J website:
  http://deeplearning4j.org
- Machine learning simplified: https://www.youtube.com/watch?v=b99UVkWzYTQ&t=5s

Willem Meints

Technical Evangelist

@willem_meints
willem.meints@infosupport.com
www.linkedin.com/in/wmeints

goto; amsterdam

Please

Remember to rate this session

Thank you!

@GOTOamst

gotoams.nl