Deep Learning in Action

With DL4J!

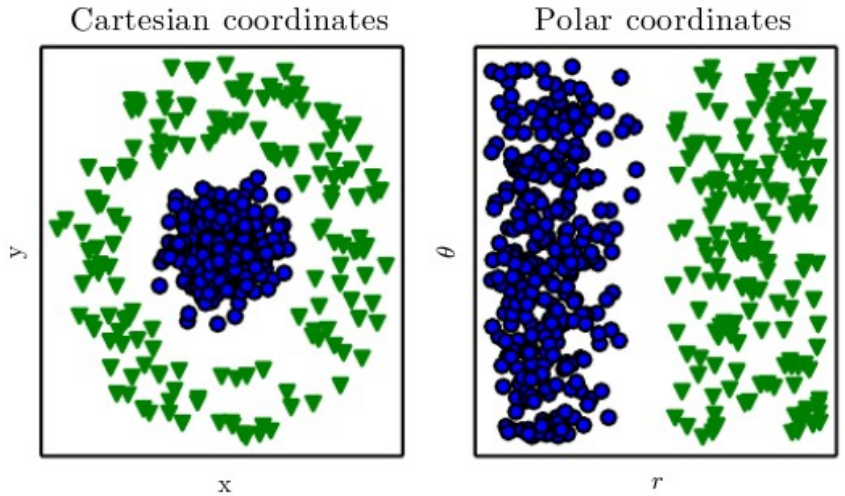
Sigrid Keydana

Deep Learning in 3 W’s

What?

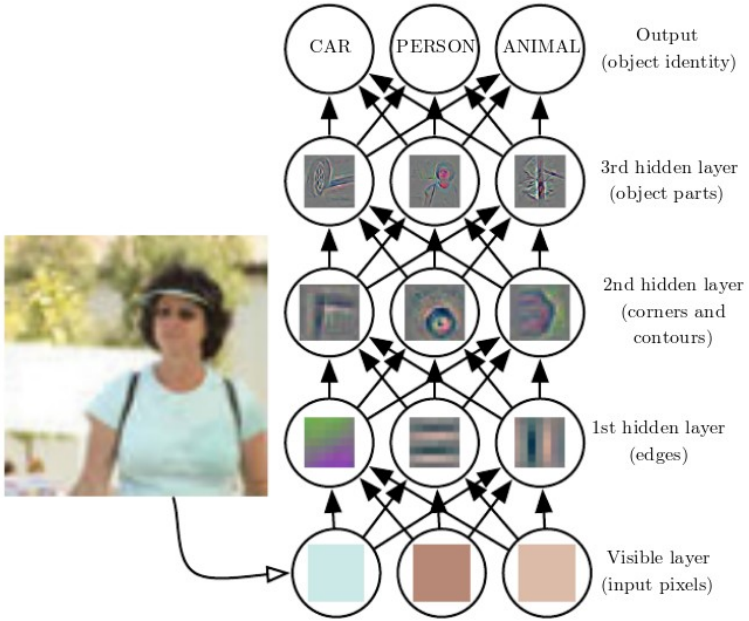
Pre-ML programs	Machine Learning (ML)	Deep Learning (DL)
in: data	in: data	in: data
in: rules	learn: mappings/functions	learn: features
out: conclusion	out: conclusion	learn: mappings/functions
		out: conclusion

Why features matter



Source: Goodfellow et al., Deep Learning, 2016

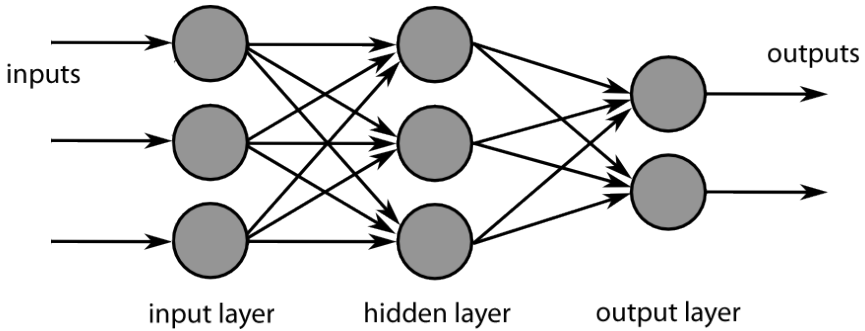
Example: Features for object classification



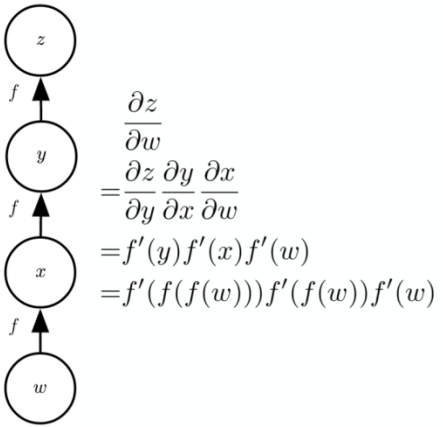
Source: Goodfellow et al., Deep Learning, 2016

hoW?

Learning from errors in a deep network



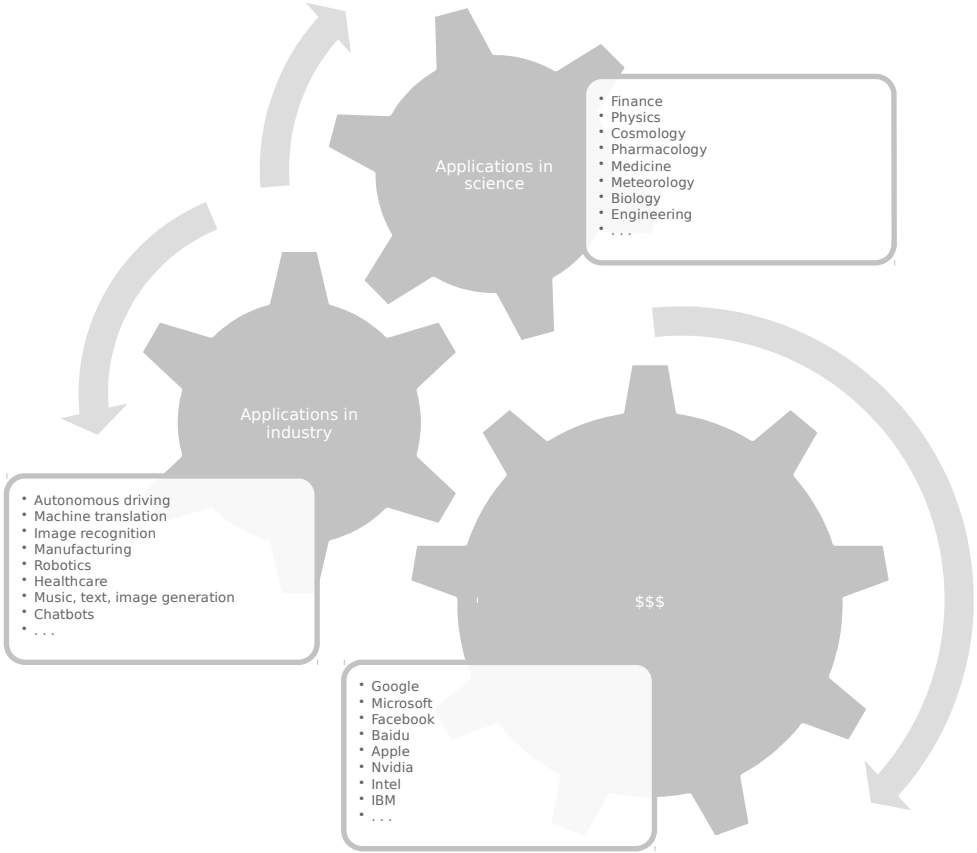
Source: Wikipedia



$$\frac{\partial z}{\partial w} = \frac{\partial z}{\partial y}\frac{\partial y}{\partial x}\frac{\partial x}{\partial w} = f'(y)f'(x)f'(w) = f'(f(f(w)))f'(f(w))f'(w)$$

Source: Goodfellow et al., Deep Learning, 2016

Why?



So is this just for the big guys?

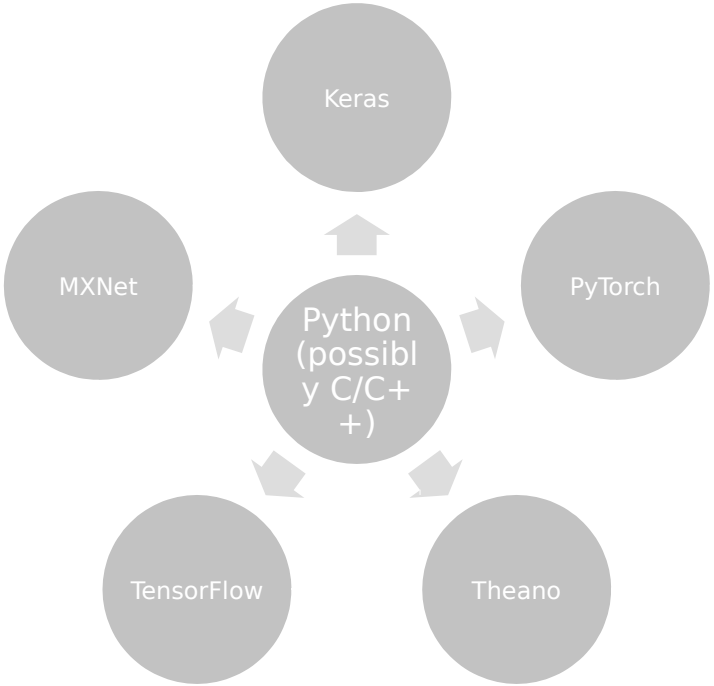
No.

Just need

- ✓ Reasonable amount of data
- ✓ Adequate hardware – or run in the cloud!
- ✓ Deep learning software – and people who know to code and run it

Let’s zoom in on the software.

Deep Learning frameworks



What if my whole environment is Java/JVM-based?

And/or I need to run on a (Spark/Hadoop) cluster?

And/or I need to optimize performance?

Enter…

DL4J

DL4J in a nutshell

- Open-source DL framework for the JVM ecosystem (maintained by Skymind)
- Distributed – runs on Spark and Hadoop
- Fast (with adequate JVM settings)
- Lots of nicely commented example code
- Helpful developers! (Github / Gitter chat)

So if the focus is on production …

… what does the developer's experience look like?

- Developer 1 (normally uses Keras for quick experiments): "I miss playing around in the notebook… What's actually going on in that code?"
- Developer 2 (too impatient for fiddling around with parameters): "Is that endless training time really necessary? Can't I just use something out-of-the-box?"
- Developer 3 (could be me): "I want to apply this method from that paper and don't have the time… perhaps it's already implemented in …?"

But first… let's just see some code!

How does linear regression look like in DL4J?

```
MultiLayerNetwork net = new MultiLayerNetwork(new NeuralNetConfiguration.Builder()
                .iterations(numEpochs)
                .optimizationAlgo(OptimizationAlgorithm.CONJUGATE_GRADIENT)
                .updater(Updater.SGD)
                .list()
                .layer(0, new DenseLayer.Builder().nIn(numFeatures).nOut(hiddenDim)
                        .activation(Activation.RELU)
                        .build())
                .layer(1, new OutputLayer.Builder(LossFunctions.LossFunction.MSE)
                        .activation(Activation.IDENTITY)
                        .nIn(hiddenDim).nOut(outputDim).build())
                .build());
net.init();
net.fit(X, Y);
```

Spot on … agile development

Keras model import

- Can experiment in Keras and then import models

```
MultiLayerNetwork network =
KerasModelImport.importKerasSequentialModelAndWeights(modelPath);
```

- Let’s see this in action!

Keras model import demo: Crack? No crack?

Spot on … pretrained models

Model zoo

- Can use e.g. VGG16, VGG19, ResNet, GoogleNet… with weights mostly trained on ImageNet
- Examples available for different forms of usage (adaptation to own classes, fine tuning)

```
ZooModel vgg16 = new VGG16();
ComputationGraph pretrainedNet = (ComputationGraph)
vgg16.initPretrained(PretrainedType.IMAGENET);
```

- Let’s try this out!

VGG 16 demo: What's this?

Source: https://www.juggle.org/claude-shannon-mathematician-engineer-genius-juggler/

Source: http://krisholm.com/en/blog/Introducing-the-KH-27.5

Spot on … available algorithms

Example: Anomaly detection

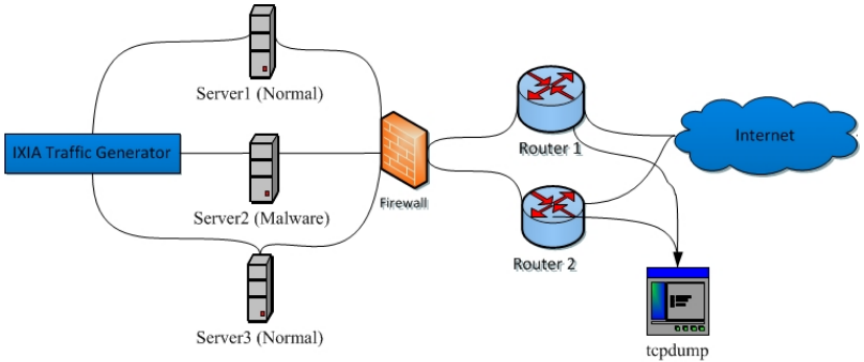
- One state-of-the-art approach: Variational Autoencoder with reconstruction probability
- Implemented in DL4J as a specific variational layer

```
MultiLayerConfiguration conf = new NeuralNetConfiguration.Builder()
        .//…
        .layer(0, new VariationalAutoencoder.Builder()
                .activation(Activation.LEAKYRELU)
                .encoderLayerSizes(encoderSizes)
                .decoderLayerSizes(decoderSizes)
                .pzxActivationFunction(latentActivation)
                .reconstructionDistribution(new
BernoulliReconstructionDistribution(Activation.SIGMOID))
                .nIn(inputSize)
                .nOut(latentSize)
                .build())
        .pretrain(true).backprop(false).build();
```

Variational autoencoder demos

MNIST handwritten digits

UNSW-NB15 intrusion detection



Conclusion: DL4J

- Nice, nicely documented, actively developed
- Many architectures and pretrained models available
- Instructive example code

$$DL\ !=\ Python$$ 

Questions? Thank you!

trivadis
makes IT easier.