MANGO SOLUTIONS

Deep Learning in R with Keras

Doug Ashton
Twitter: @mangothecat
Email: training@mango-solutions.com

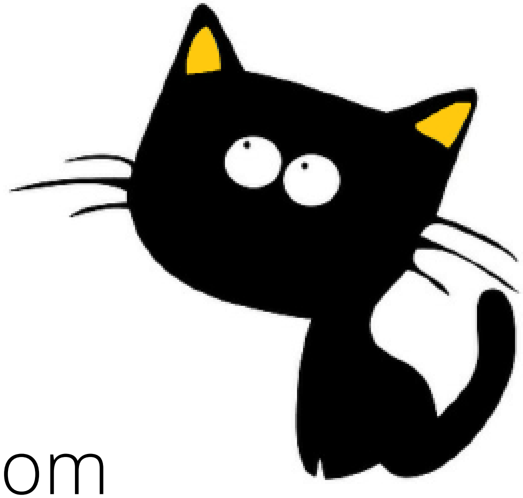
Agenda

- Introduction to Deep Learning
- First neural network with Keras
- Networks for Spatial Data (CNN)
- What next?

About Me

- Me:
  - Principal Data Scientist @ Mango
  - @dougashton

- Mango
  - @mangothecat
  - mangothecat
  - training@mango-solutions.com



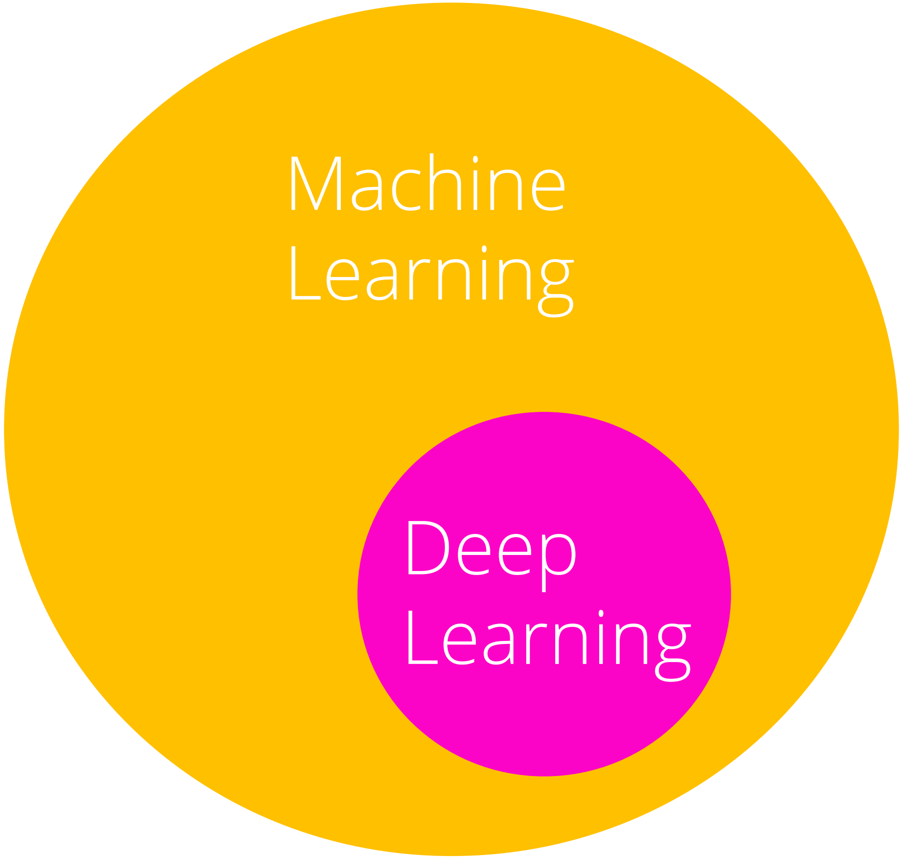
About Them

Big thank you to co-writers

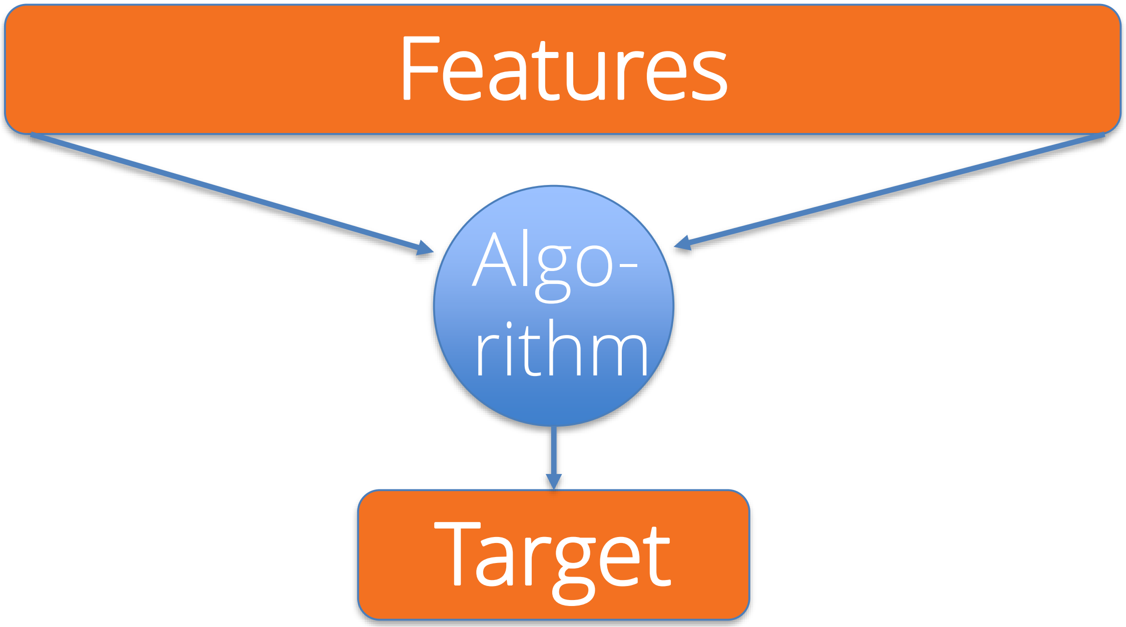
- Aimée Gott
- Owen Jones
- Alex Pavlides
- Mark Sellors*

Introduction to Deep Learning

What is Deep Learning?

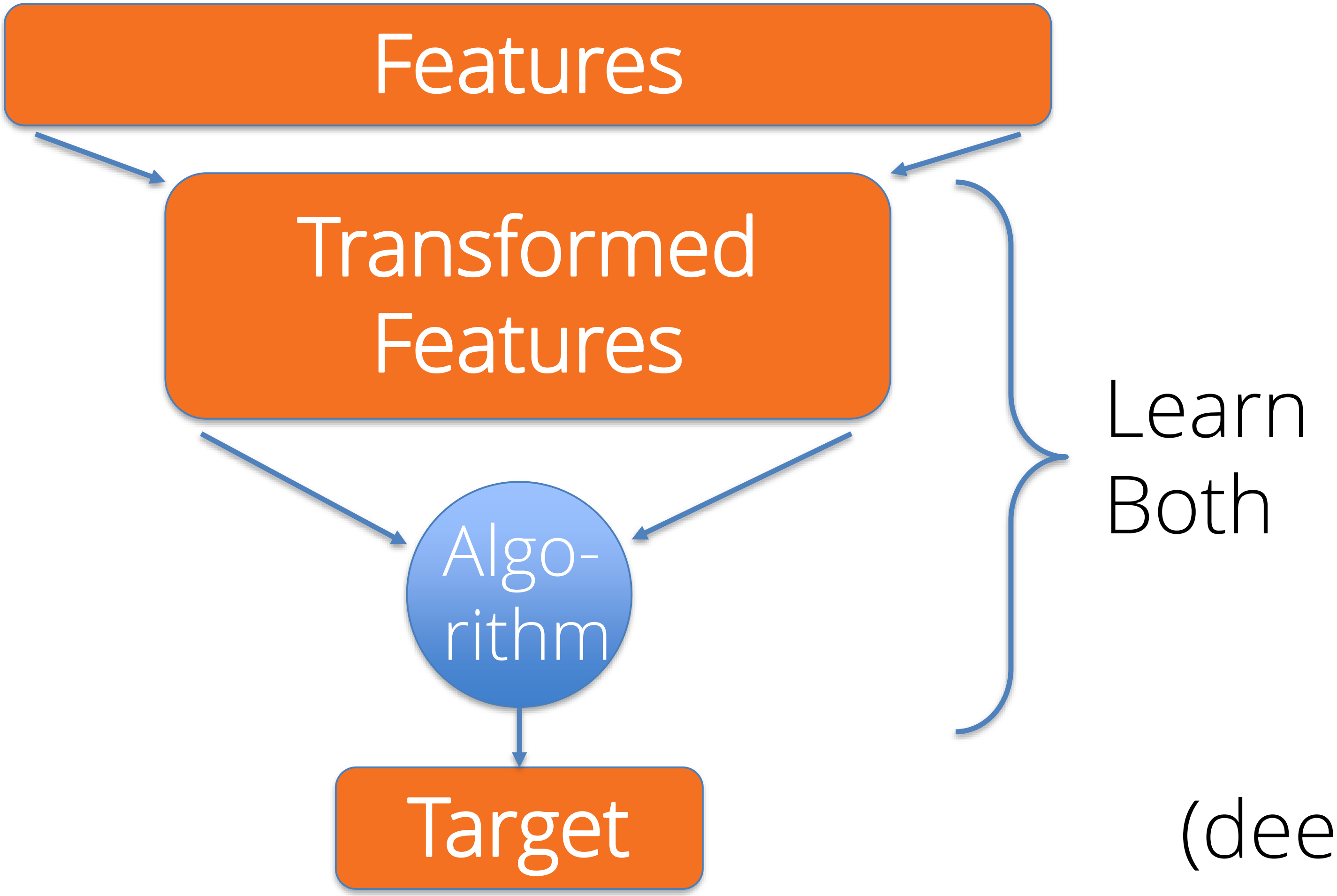


What is Deep Learning?



(shallow)

What is Deep Learning?



(deep)

What Does it Solve?

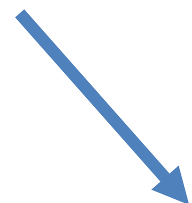
- Unstructured
  - Features are learned rather than designed
- Big
  - Generally need lots of data
- Familiar
  - Can reuse models on new problems

Why Now?

- Breakthrough in underlying algorithm
  - Back Propagation
- Massive increase in computer power
  - GPU / TPU
- Much larger datasets available
- Keras...

Neural Networks

nodes

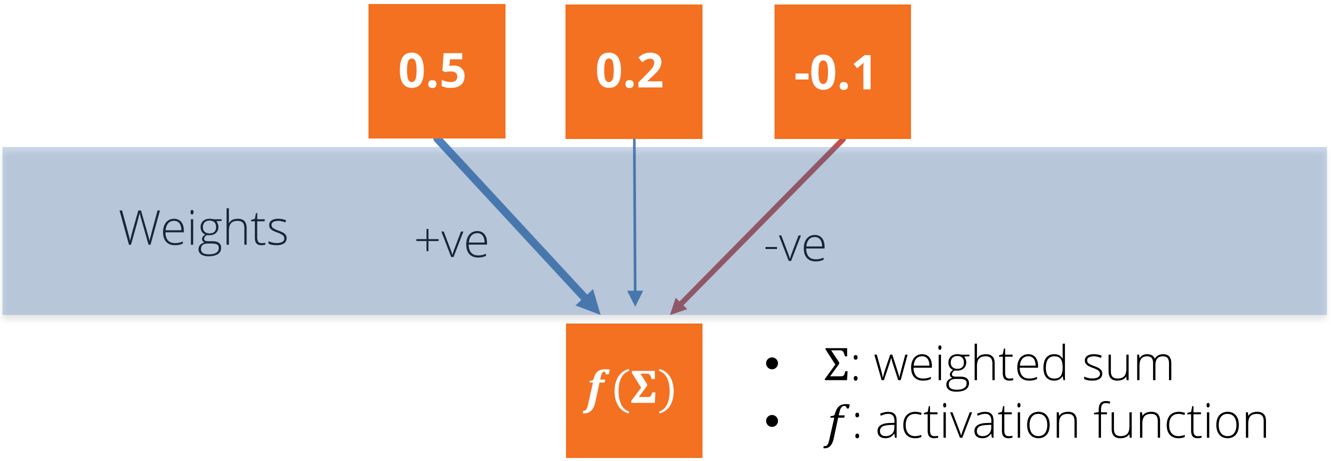


edges

A Neuron

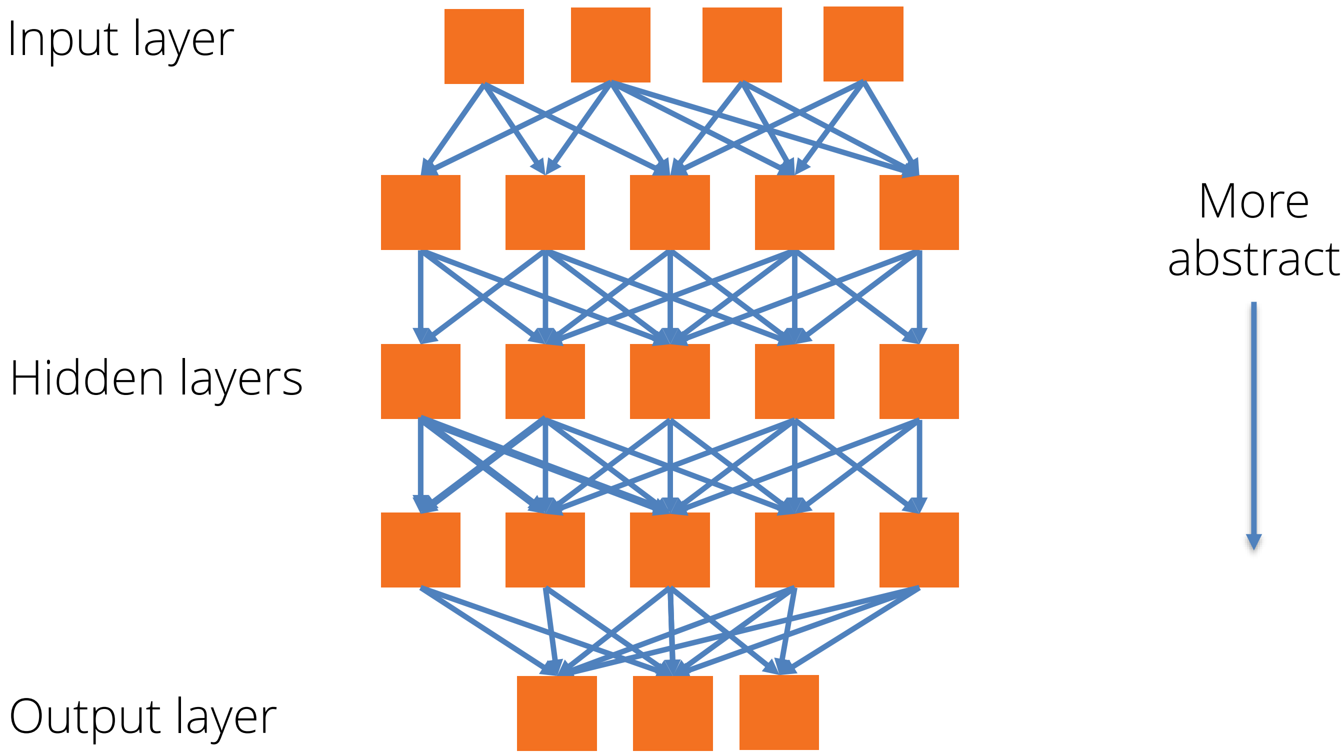
0.2

Neurons



- Σ: weighted sum
- f: activation function

Neural Network

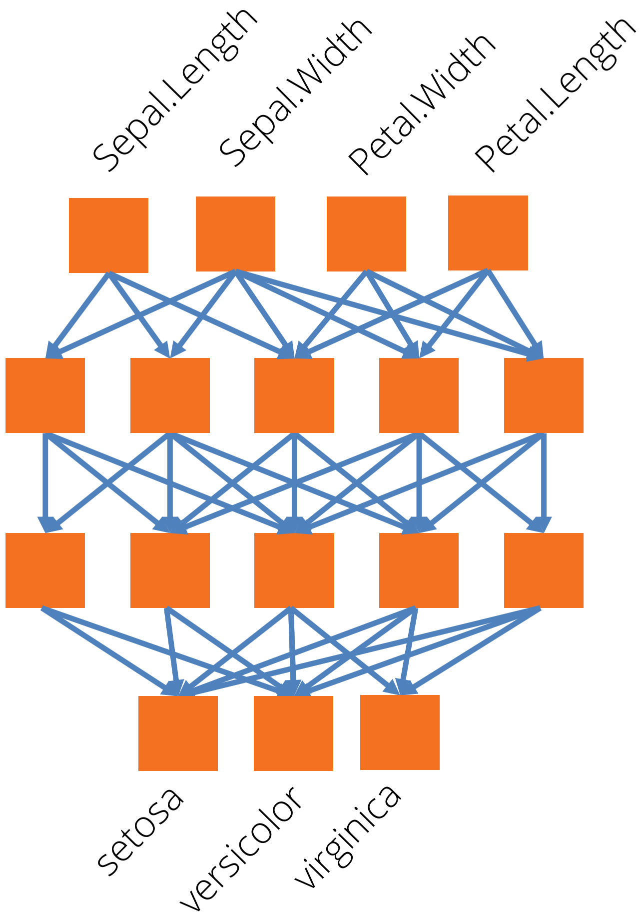


Iris Neural Network

`iris[1,1:4]`	Sepal.Length	Sepal.Width	Petal.Width	Petal.Length	Features x
	5.1	3.5	1.4	0.2	

`iris[1,5]`	setosa	versicolor	virginica	Target y
	1	0	0	

Iris Neural Network



TensorFlow

- Turns equations into dataflow graphs
  - https://www.tensorflow.org

- Efficient numerical solver
- Built for CPU, GPU, and TPU
- Not only for neural networks

TensorFlow

TensorFlow and R

- RStudio built an R interface
  - https://tensorflow.rstudio.com

- Python <-> R handled by reticulate
  - https://rstudio.github.io/reticulate

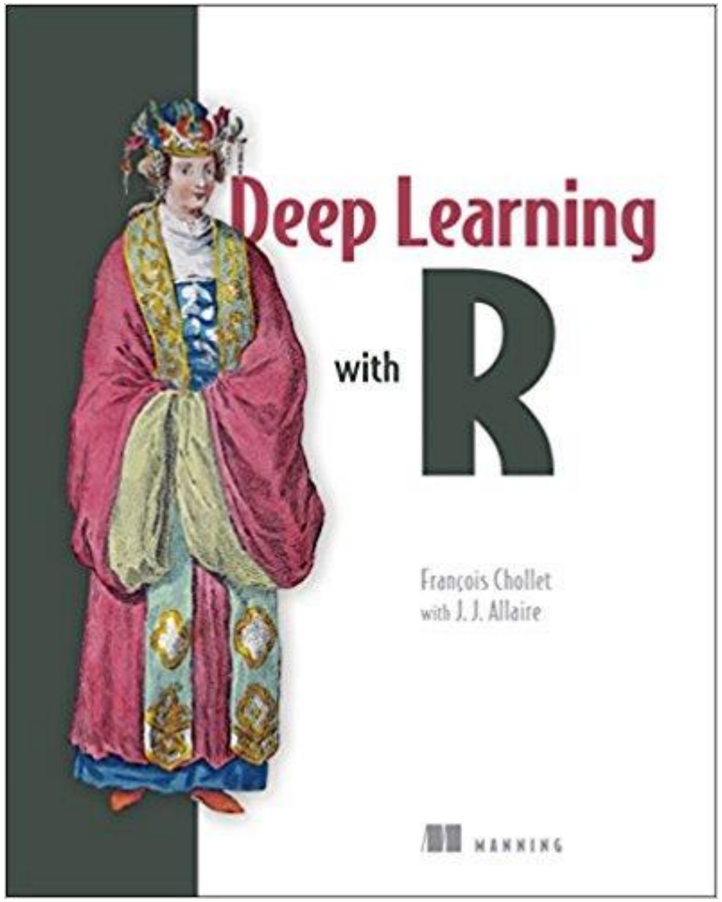
Keras

- High level interface specifically for neural networks
  - https://keras.io
  - François Chollet
- Works with multiple backends
  - TensorFlow, CNTK, Theano

Keras and R

- Rstudio built an interface to Keras
  - https://keras.rstudio.com
- Works with multiple backends
  - TensorFlow, CNTK, Theano

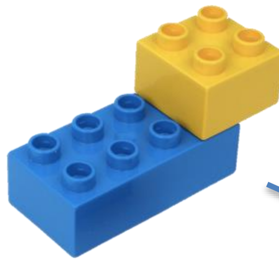
Keras and R Book



Deep Learning with R
- François Chollet
- J. J. Allaire

Manning

How it fits together



Keras

$$\sigma = \sum_i z_i$$

$$p_j = \frac{e^{x^T w_j}}{\sum_k e^{x^T w_k}}$$

TensorFlow

00100110

Alternatives for R Users

- MXNet
  - https://mxnet.incubator.apache.org/api/r/
- H2O Deep Water
  - https://www.h2o.ai/deep-water/

RStudio Server

http://odsc.mangodatalabs.com

- Username/Password from card
- All libraries pre-installed
- Copy code out at the end
  - Server won't be checkpointed

github.com/mangothecat/keras-workshop

On your own machine

```
install.packages(c("tidyverse",
                   "caret",
                   "keras"))

library(keras)
install_keras() # can take a while
```

Limit CPU Use

```
library(keras)
# Use this to limit cpu
use_session_with_seed(1234)
```

Because otherwise tensorflow might take all the cores

First Keras Model

First Keras Model

- Prepare Data
- Model
- Evaluate

Prepare Data

Prepare Data

- Split train and test
- Numeric Matrices/Arrays
  - Factors
  - Scaling
  - Missing values

Prepare - Split Data

```
library(caret)
library(tidyverse)

## Sample IDs for training set
trainID <- createDataPartition(iris$Species, p = 0.8)

trainingData <- iris %>%
  slice(trainID$Resample1)

testData <- iris %>%
  slice(-trainID$Resample1)

fullData <- list(train = trainingData,
                 test = testData)
```

Prepare - One Hot Encode

```
dummy <- dummyVars(~ Species,
                   data = iris)

irisDummy <- map(fullData, predict,
                 object = dummy)

head(irisDummy$train)
```

Prepare - Centre Scaling

```
numericIris <- map(fullData,
                   select_if,
                   is.numeric)

scaledIris <- map(numericIris,
                  scale)
```

Prepare - NAs

- Can't have NAs
- Impute 0 (mean)
  - `map(scaledIris, replace_na, replace = 0)`
- Or look at caret preprocessors
- No NAs in `iris`

Prepare - Matrices

```
## Create x and y matrix

xIris <- map(scaledIris, as.matrix)

yIris <- map(irisDummy, as.matrix)
```

Model

Model

- Networks can have complex shapes
- Sequential models are linear stack

```
model <- keras_model_sequential()
```

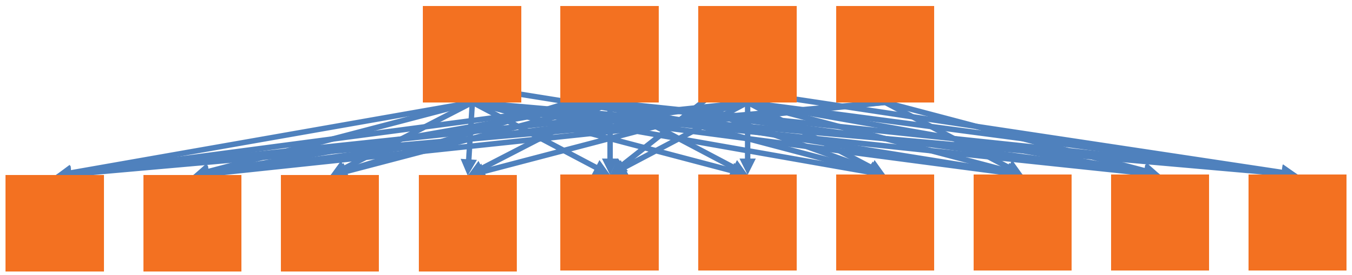
- Model objects *change in place*

Model - Layers

```
model %>%
    layer_dense(units = 10,
                input_shape = 4)
```

- Only need `input_shape` once
- Shape doesn't include observations

Model - Dense Layers



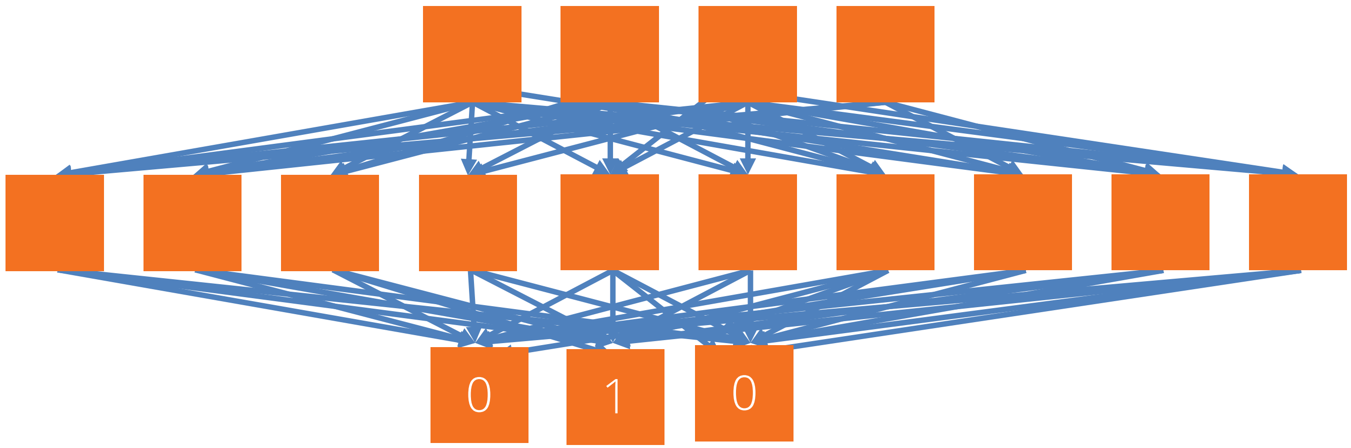
```
model %>%
    layer_dense(units = 10,
                input_shape = 4)
```

Model - Softmax Layer

```
model %>%
    layer_dense(units = 3,
                activation = 'softmax')
```

- Usually on the output
- Use for categorical output

Model - Softmax Layer



Model - Summary

```
> model
Model
________________________________________________________________
Layer (type)                 Output Shape              Param #
================================================================
dense_1 (Dense)              (None, 10)                50
________________________________________________________________
dense_2 (Dense)              (None, 3)                 33
================================================================
Total params: 83
Trainable params: 83
Non-trainable params: 0
________________________________________________________________
```

Compile

```
model %>% compile(
  optimizer = 'rmsprop',
  loss = 'categorical_crossentropy',
  metrics = 'accuracy’
)
```

- Optimizer: Mostly rmsprop
- Metrics: Mostly accuracy
- Loss: 3 main choices

Compile - Loss

Output	Loss Function
Binary Classification	binary_crossentropy
Multi-class Classification (single label)	categorical_crossentropy
Multi-class Classification (multiple labels)	binary_crossentropy
Regression	mse

Fit

```
history <-
  model %>%
    fit(xIris$train,
        yIris$train,
        epochs = 100,
        validation_data =
             list(xIris$test,
                  yIris$test))
```

Exercise

- Get the iris model working
- Try adding a layer

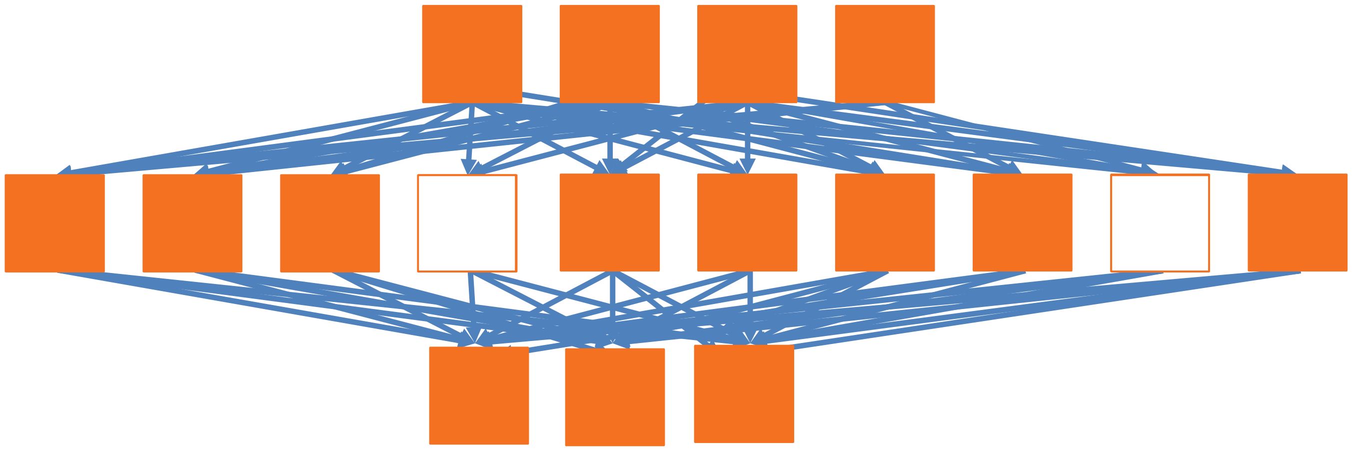
Evaluate


5.13
ROLEX
2.26
SETS
GAMES
POINTS
PREVIOUS SETS
J. HENIN-HARDENNE
V
Miss E. DANIILIDOU
1 5 0
1 6 15

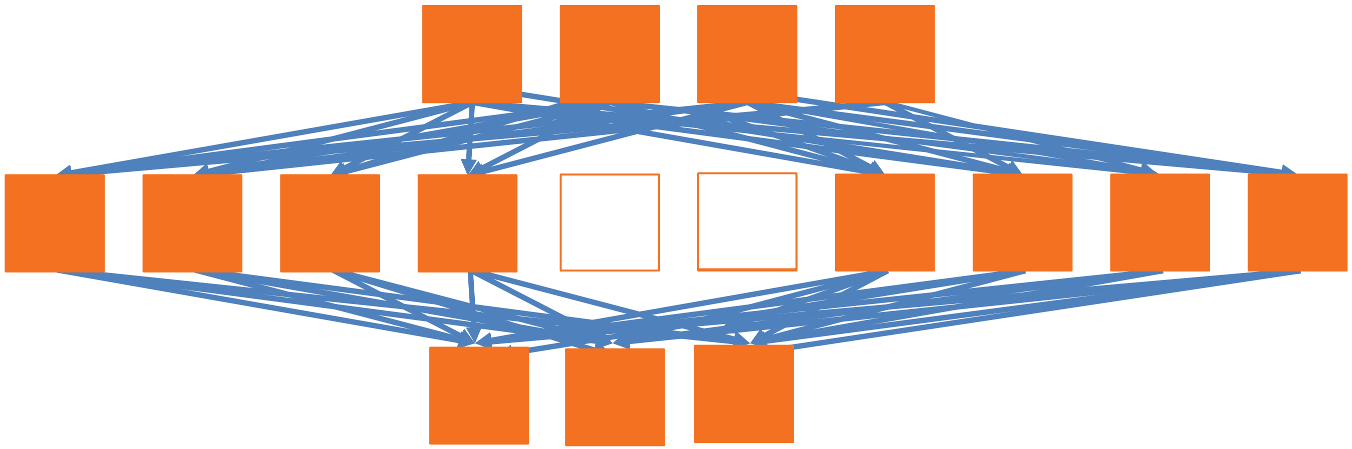
Improving the Model

- Change number of hidden units
- Add more layers
- Add dropout
  - Helps prevent overfitting
- Mostly trial and error

Dropout



Dropout

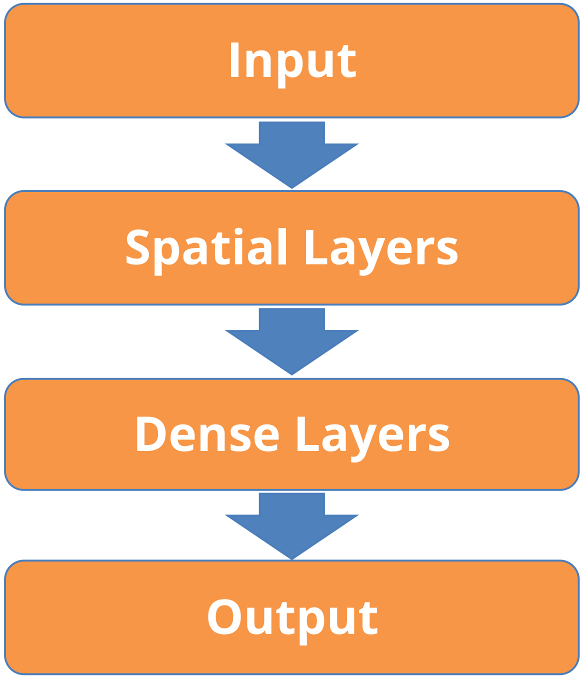


Exercise

- Build a simple model for mtcars to predict mpg
  - Scale, one-hot-encode
  - Single output unit with linear activation
- How does it do against `lm`?

Networks for Spatial Data

Convolutional Neural Networks



Walking Data

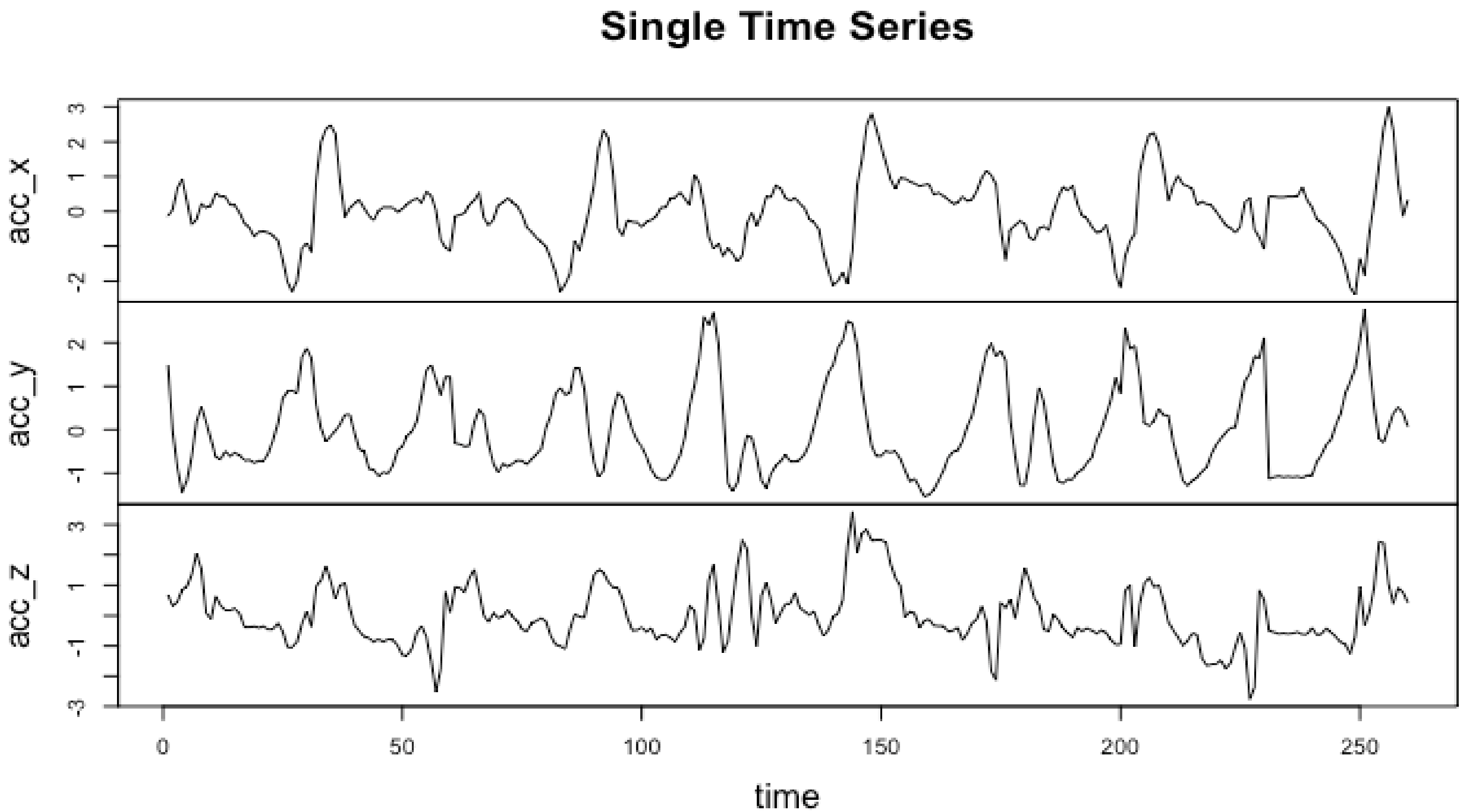
```
walking <- readRDS("/data/walking.rds")

xWalk <- readRDS("/data/xWalk.rds")

yWalk <- readRDS("/data/yWalk.rds")
```

Walking Data

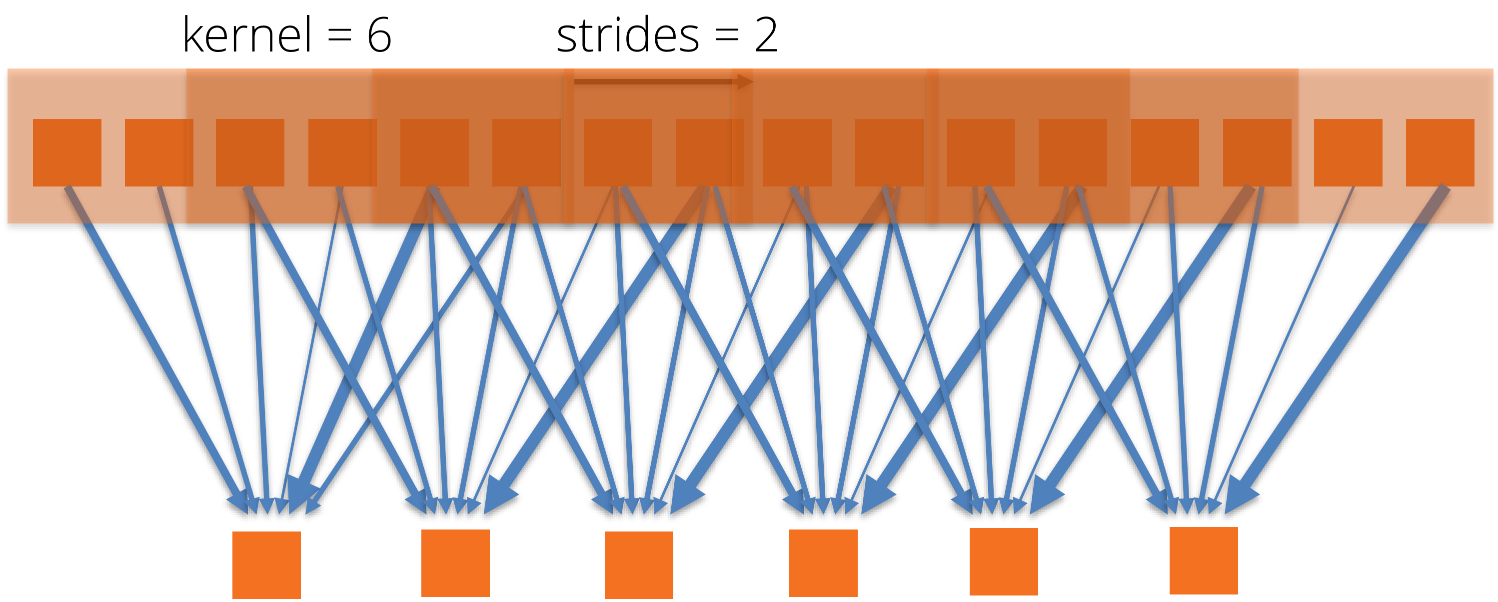
- Accelerometer data from the UCI
- Filtered to walking activity
- Can we recognise someone by their gait?
- Chopped into 5 second chunks

```
> dim(walking$x)
[1] 6792  260    3
```

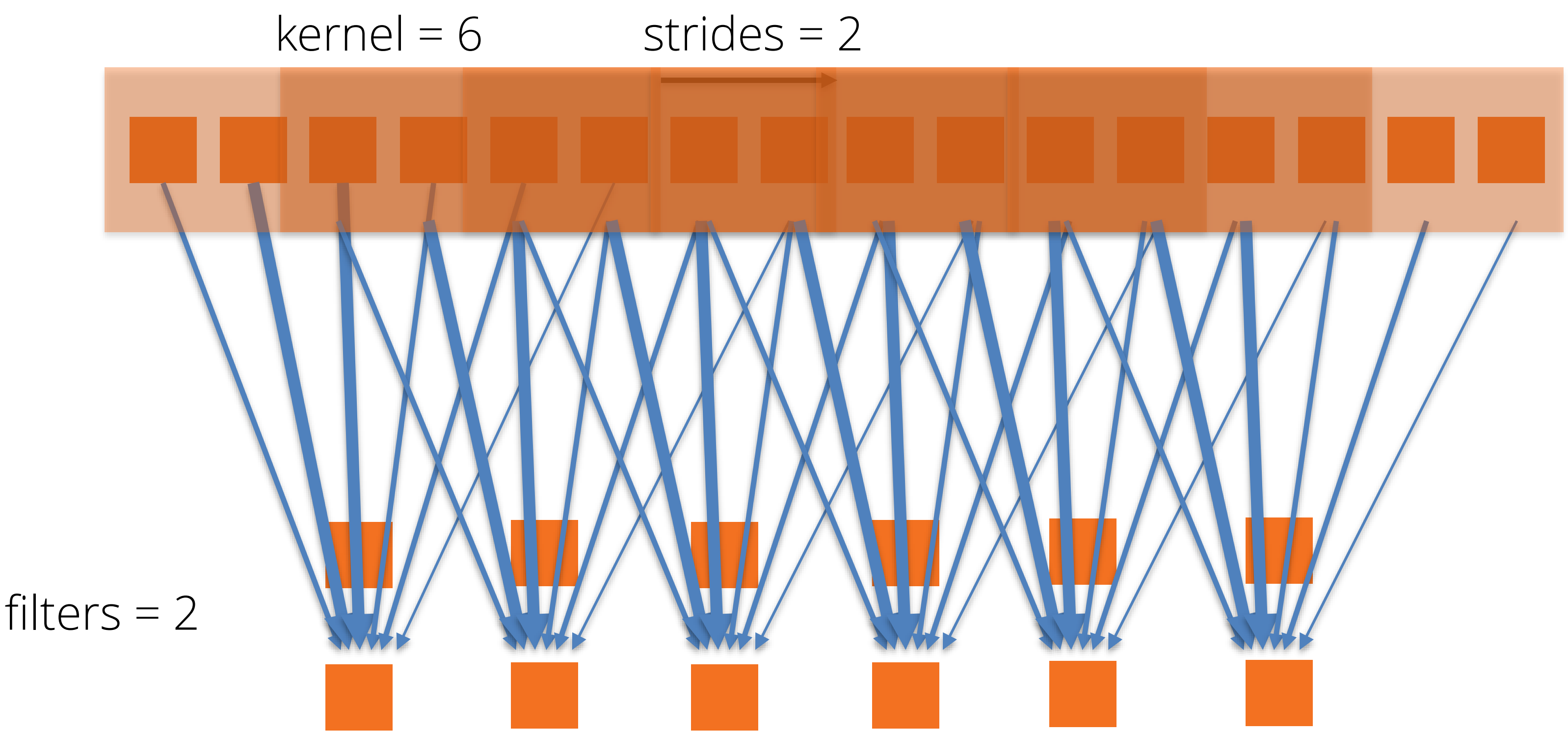
Walking Data



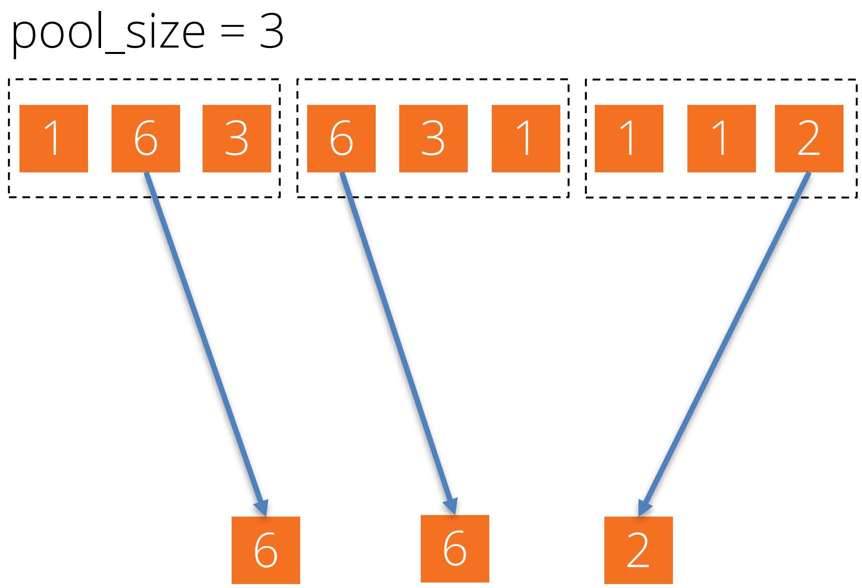
Convolution Layer



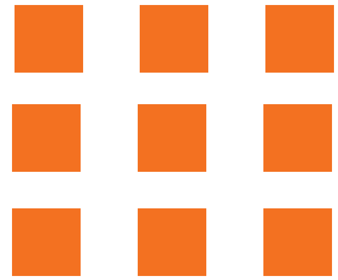
Convolution Layer - Filters



Max Pooling



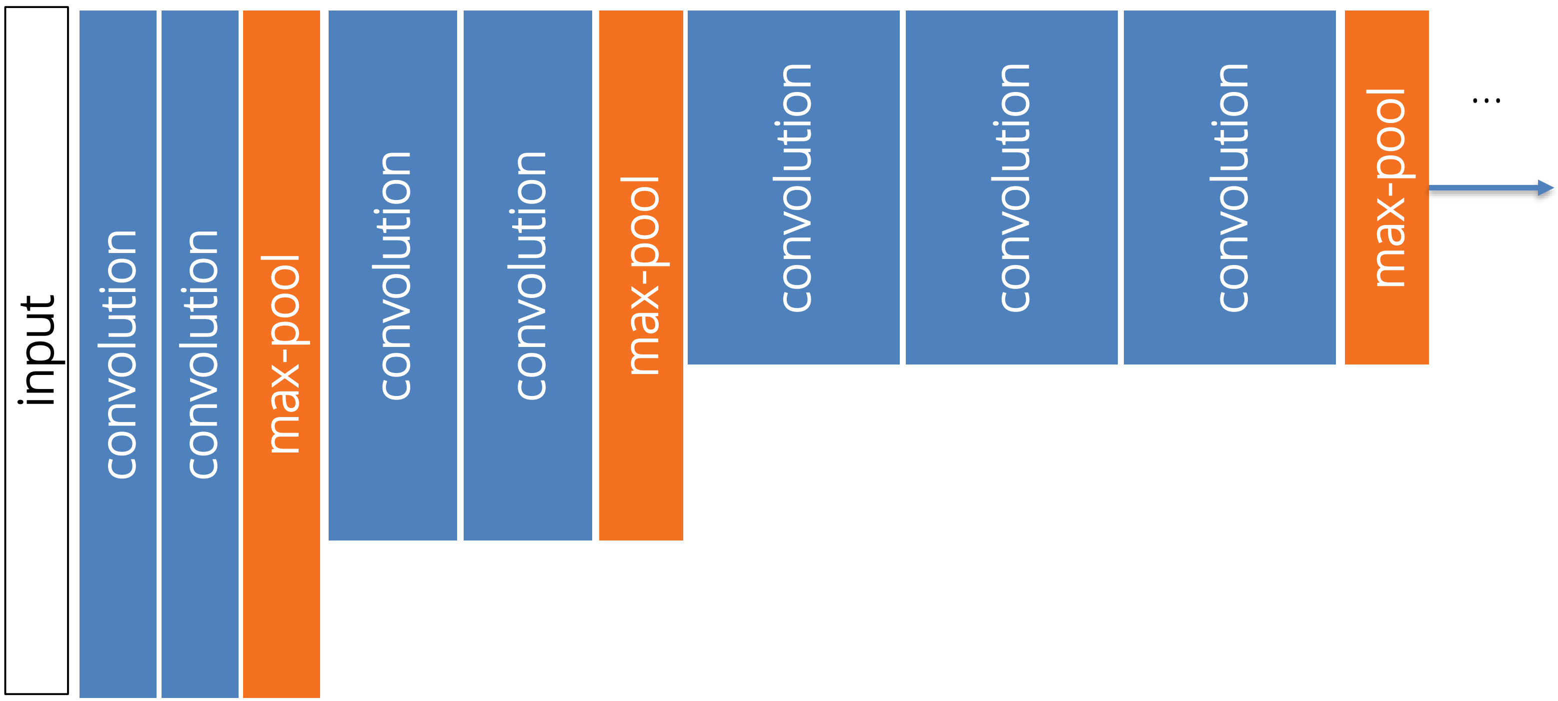
Flattening



Exercise

- Try adding more conv layers to your model
- Does dropout improve overfitting?

CNN Architectures - VGG



What Next?

- Pre-trained Networks
- CloudML