Handwritten Recognition using Deep Learning with R

Poo Kuan Hoong

August 17, 2016

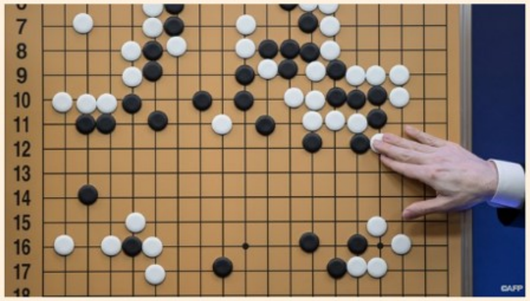
Google DeepMind Alphago

AlphaGo marks stark difference between AI and human intelligence

Daniel Susskind

Share | Author alerts | Print | Clip | Comments

Google's robot relied on processing power and data storage, writes Daniel Susskind



There are many ways of being smart that aren't smart like us." These are the words of Patrick Winston, a leading voice in the field of artificial intelligence. Although his idea is simple, its significance has been lost on most people thinking about the future of work. Yet this is the feature of AI that ought to preoccupy us the most.

From the 1950s to the 1980s, during the "first wave" of AI research, it was generally

What AlphaGo's Win Means for Your Job

COMMENTARY by Howard Yu @HowardHYu MARCH 21, 2016, 3:11 PM EDT

South Korean professional Go player Lee Se-Dol reviews the match after the fourth match against Google's artificial intelligence program, AlphaGo, during the Google DeepMind Challenge Match on March 13, 2016 in Seoul, South Korea.
Photograph by Google via Getty Images

And the future of artificial intelligence

Just last week, machines crossed a momentous milestone. Google's AlphaGo, a computer algorithm, beat Go world champion Lee Sedol 4 to 1 in the ancient Chinese board game.

RECOMMENDED FOR YOU

AlphaGo's Success Shows the Human Advantage Is Eroding Fast

Howard Yu is a professor of strategic management and innovation at IMD business school.

MARCH 9, 2016

It's impressive that computers can be programmed to conduct conversations that make them indistinguishable from a person. But Google's AlphaGo is demonstrating for the first time that machines can truly learn and think in a human way.

In 1996, IBM's Deep Blue program overwhelmed the world's greatest player, Garry Kasparov, with a brute force approach, in which the machine could account for all the possible outcomes.

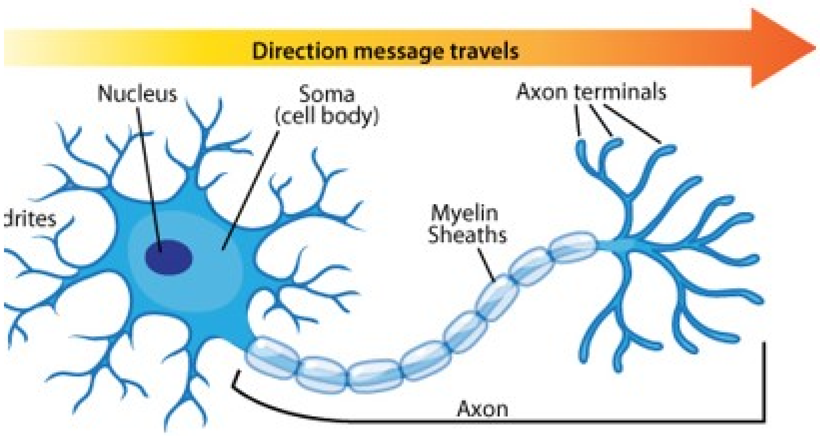
AlphaGo's approach, on the other hand, is, potentially, a game changer. It can master games by adjusting as it goes.

Unlike chess, where each move affords about 40 options, the ancient board game Go, has up to 200 choices. The permutation of outcomes quickly compounds to a bewildering range of choices — more than the total number of atoms in the entire observable universe.

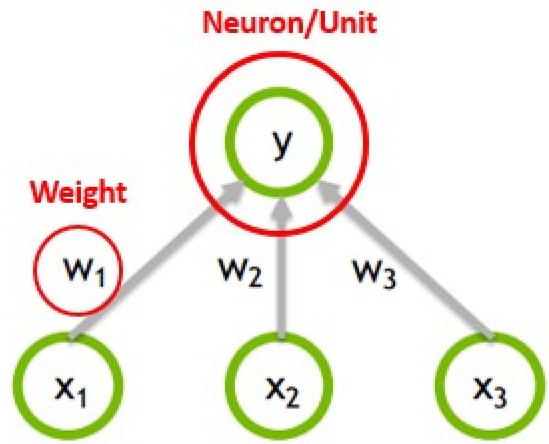
Introduction

- In the past 10 years, machine learning and Artificial Intelligence (AI) have shown tremendous progress
- The recent success can be attributed to:
  - *Explosion of data*
  - *Cheap computing cost - CPUs and GPUs*
  - *Improvement of machine learning models*
- Much of the current excitement concerns a subfield of it called “deep learning”.

Human Brain



Biological Neuron



$$y=F(w_1x_1+w_2x_2+w_3x_3)$$

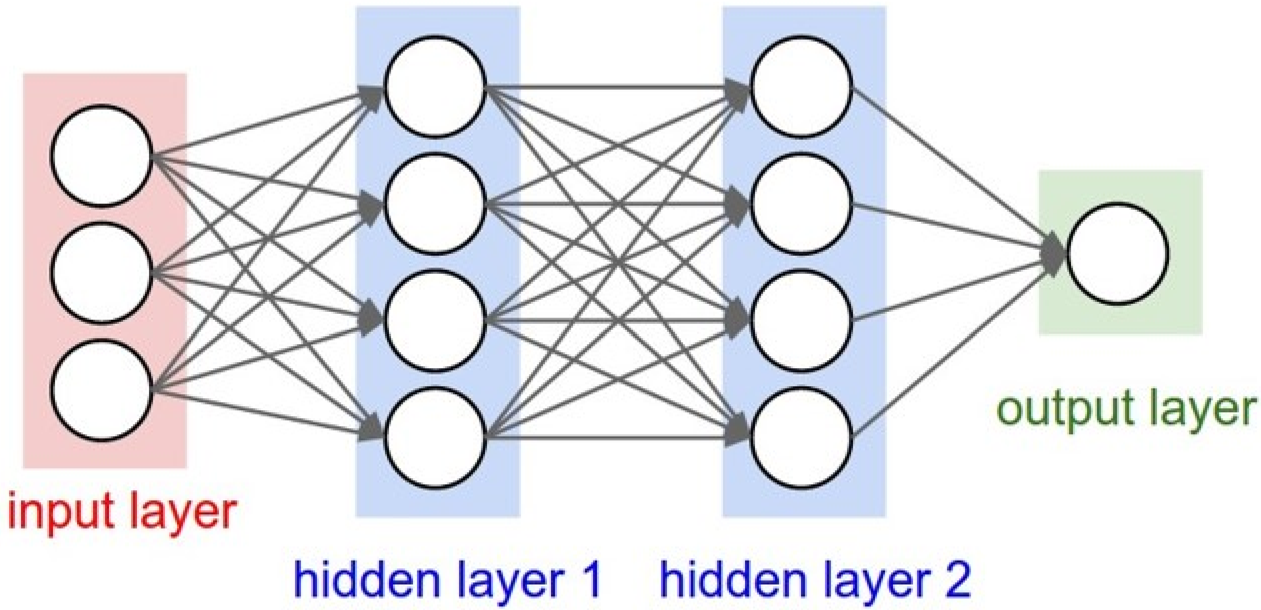
$$F(x)=\max(0,x)$$

Artificial Neuron

Neural Networks

- Deep Learning is primarily about neural networks, where a network is an interconnected web of nodes and edges.
- Neural nets were designed to perform complex tasks, such as the task of placing objects into categories based on a few attributes.
- Neural nets are highly structured networks, and have three kinds of layers - an input, an output, and so called hidden layers, which refer to any layers between the input and the output layers.
- Each node (also called a neuron) in the hidden and output layers has a classifier.

Neural Network Layers



Neural Network: Forward Propagation

- The input neurons first receive the data features of the object. After processing the data, they send their output to the first hidden layer.
- The hidden layer processes this output and sends the results to the next hidden layer.
- This continues until the data reaches the final output layer, where the output value determines the object's classification.
- This entire process is known as **Forward Propagation**, or **Forward prop**.

Neural Network: Backward Propagation

- To train a neural network over a large set of labelled data, you must continuously compute the difference between the network's predicted output and the actual output.
- This difference is called the cost, and the process for training a net is known as **backpropagation**, or **backprop**
- During backprop, *weights and biases are tweaked* slightly until the lowest possible cost is achieved.
- An important aspect of this process is the gradient, which is a measure of how much the cost changes with respect to a change in a weight or bias value.

The 1990s view of what was wrong with back-propagation

- It **required a lot of labelled** training data
  - *Almost all data is* ***unlabeled***
- The learning time did not scale well
  - *It was* ***very slow in networks*** *with multiple hidden layers.*
- It got stuck at local optima
  - *These were often surprisingly good but there was no good theory*

Deep Learning

- Deep learning refers to artificial neural networks that are composed of many layers.
- It's a growing trend in Machine Learning due to some favorable results in applications where the target function is very complex and the datasets are large.

Deep Learning: Benefits

- Robust
  - *No need to design the features ahead of time - features are automatically learned to be optimal for the task at hand*
  - *Robustness to natural variations in the data is automatically learned*
- Generalizable
  - *The same neural net approach can be used for many different applications and data types*
- Scalable
  - *Performance improves with more data, method is massively parallelizable*

Deep Learning: Weaknesses

- Deep Learning **requires a large dataset**, hence long training period.
- In term of cost, Machine Learning methods like SVMs and other tree ensembles are very easily deployed even by relative machine learning novices and can usually get you reasonably good results.
- Deep learning methods **tend to learn everything**. It's better to encode prior knowledge about structure of images (or audio or text).
- The learned features are often **difficult to understand**. Many vision features are also not really human-understandable (e.g, concatenations/combinations of different features).
- Requires **a good understanding of how to model** multiple modalities with traditional tools.

Deep Learning: Applications

H2O Library

- H2O is an open source, distributed, Java machine learning library
- Ease of Use via Web Interface
- R, Python, Scala, Spark & Hadoop Interfaces
- Distributed Algorithms Scale to Big Data
- Package can be downloaded from http://www.h2o.ai/download/h2o/r

H2O R Package on CRAN

Requirements

- The only requirement to run the "h2o" R package is R >=3.1.0 and Java 7 or later.
- Tested on many versions of Linux, OS X and Windows.

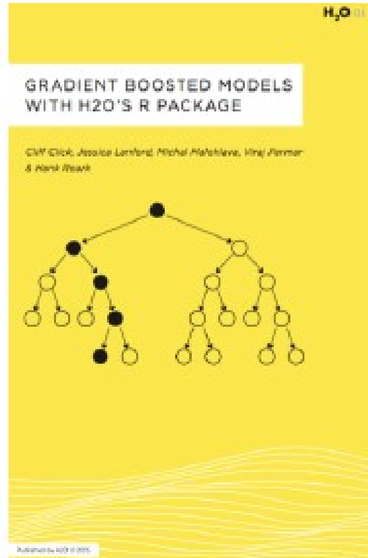
Installation

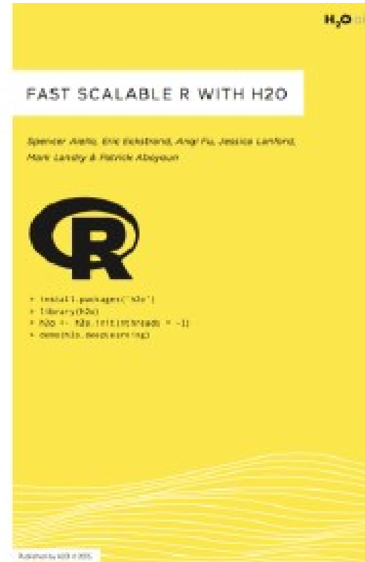
- The easiest way to install the "h2o" R package is to install directly from CRAN.
- Latest version: `http://h2o.ai/download`

Design

- No computation is ever performed in R.
- All computations are performed (in highly optimized Java code) in the H2O cluster and initiated by REST calls from R.

H2O booklets

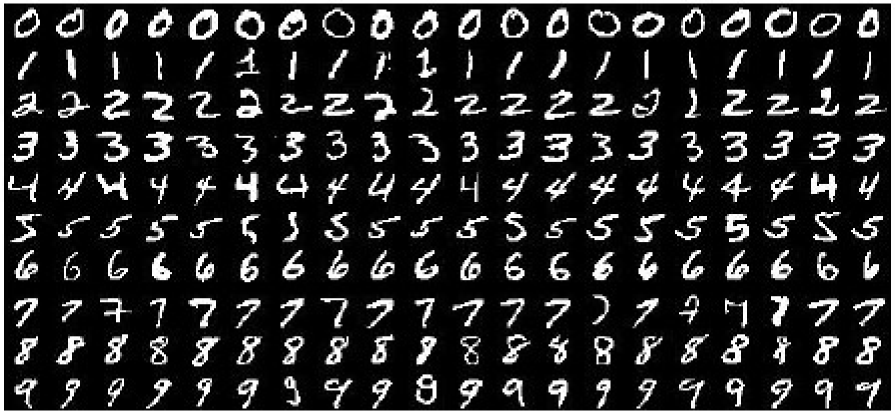




H2O reference booklets can be downloaded from https://github.com/h2oai/h2o-3/tree/master/h2o-docs/src/booklets/v2_2015/PDFs/online

MNIST Handwritten Dataset

- The MNIST database consists of handwritten digits.
- The training set has 60,000 examples, and the test set has 10,000 examples.
- The MNIST database is a subset of a larger set available from NIST. The digits have been size-normalized and centered in a fixed-size image
- For this demo, the Kaggle pre-processed training and testing dataset were used. The training dataset, (train.csv), has 42000 rows and 785 columns.



Demo

The sourcecode can be accessed from here

https://github.com/kuanhoong/myRUG_DeepLearning

Create training and testing datasets

```
#load caret library
library (caret)

#split the dataset to 80% training and 20% testing
inTrain<- createDataPartition(train$label, p=0.8, list=FALSE)
training<-train[inTrain,]
testing<-train[-inTrain,]

#store the datasets into .csv files
write.csv (training , file = "train-data.csv", row.names = FALSE)
write.csv (testing , file = "test-data.csv", row.names = FALSE)
```

Start H2O Cluster from R and load data into H2O

```
#load h2o library
library(h2o)

#start a local h2o cluster
local.h2o <- h2o.init(ip = "localhost", port = 54321, startH2O =
TRUE, nthreads=-1)

# pass dataframe from inside of the R environment to the H2O
instance
trData<-as.h2o(training)
trData[,1]<-as.factor(trData[,1])
tsData<-as.h2o(testing)
tsData[,1]<-as.factor(tsData[,1])
```

Deep Learning in R: Train & Test

```
#deep learning model
model.dl <- h2o.deeplearning(x = 2:785,
                             y = 1,
                             trData,
                             activation = "Tanh",
                             hidden=rep(160,5),
                             epochs = 20)

#use model to predict testing dataset
pred.dl<-h2o.predict(object=model.dl, newdata=tsData[,-1])
pred.dl.df<-as.data.frame(pred.dl)
```

```
> training <- read.csv ("train-data.csv")
> testing  <- read.csv ("test-data.csv")
> training[,1]<-as.factor(training[,1])
> trData<-as.h2o(training)
  |============================================================| 100%
> tsData<-as.h2o(testing)
  |============================================================| 100%
> res.dl <- h2o.deeplearning(x = 2:785, y = 1, trData, activation = "Tanh
", hidden=rep(160,5),epochs = 20)
  |===============================                             |  53%
```

Result

```
summary(pred.dl,exact_quantiles=TRUE)
test_labels<-testing[,1]

#calculate number of correct prediction
sum(diag(table(test_labels,pred.dl.df[,1])))

# shut down virtual H2O cluster
h2o.shutdown(prompt = FALSE)
```

Lastly…

Thank You

@kuanhoong

https://www.linkedin.com/in/kuanhoong

khpoo@mmu.edu.my