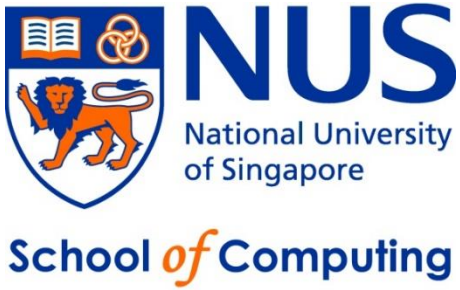
CS3237

Introduction to IoT

Introduction

colintan@nus.edu.sg

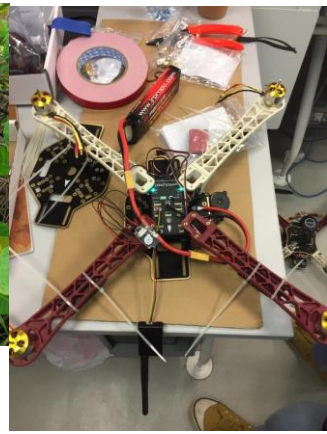


About Me

Dr. **Colin** Tan

Department of Computer Science, NUS School of Computing.

Email: colintan@nus.edu.sg



Dr. **Boyd** Anderson

I am a Kiwi (New Zealander) and did my PhD at NUS. I liked it so much here that I decided I didn't want to leave!

My research interests are in *embedded systems*, *wearable sensors*, and *gait analysis* (analysing the way you walk or run). I am also working on *wearable sensors for sports* (such as fencing and sprinting).

Contact me at: boyd@comp.nus.edu.sg My office: **COM2-03-26**

Internet of Things (IoT)

- **Physical objects or "Things" embedded with computational intelligence and connected to the network**
- **Cooperation between cyber-domain and physical world**

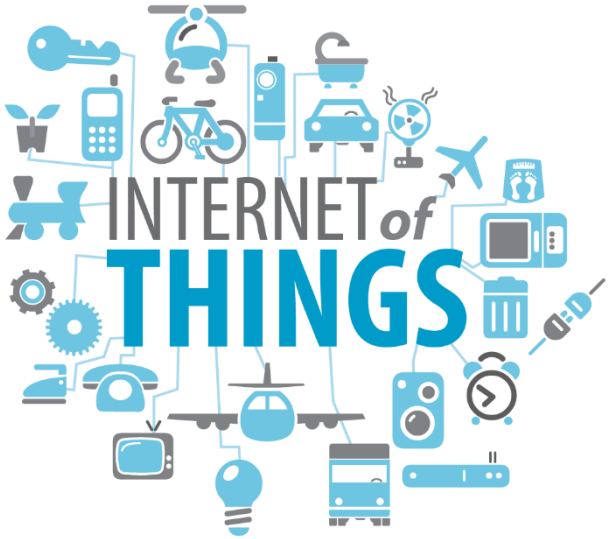


Image Credit: ilex

What is Internet of Things?

Thing

+

Computational Intelligence

+

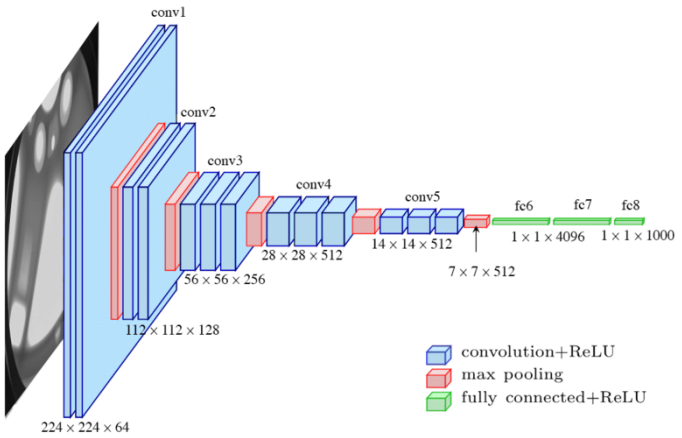
Connection to Internet

Smart Refrigerator

Image Credit: LIEBHERR
Leanna Garfield/Tech Insider

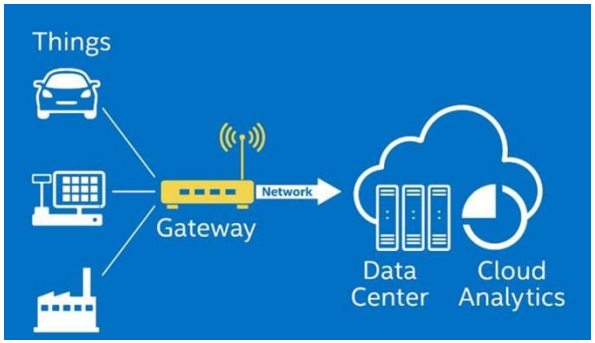
Internet of Things: Not Just Things

- **IoT is much more than just the "things":**
  - The "things" produce enormous amounts of data (more on this later).
  - Can leverage on this data to do very useful things:
    - ✓**Predict the weather.**
    - ✓**Understand when and where crowding will occur.**
    - ✓**Plot the likely trajectory of an illness.**
    - ✓**Find key vulnerabilities in a physical system.**
    - ✓**…**
- **Thus CS3237 covers both:**
  - Embedded systems – Knowledge and skills to design and build the "things".
  - Deep Learning – Knowledege and skills to leverage on data to perform intelligent actions.



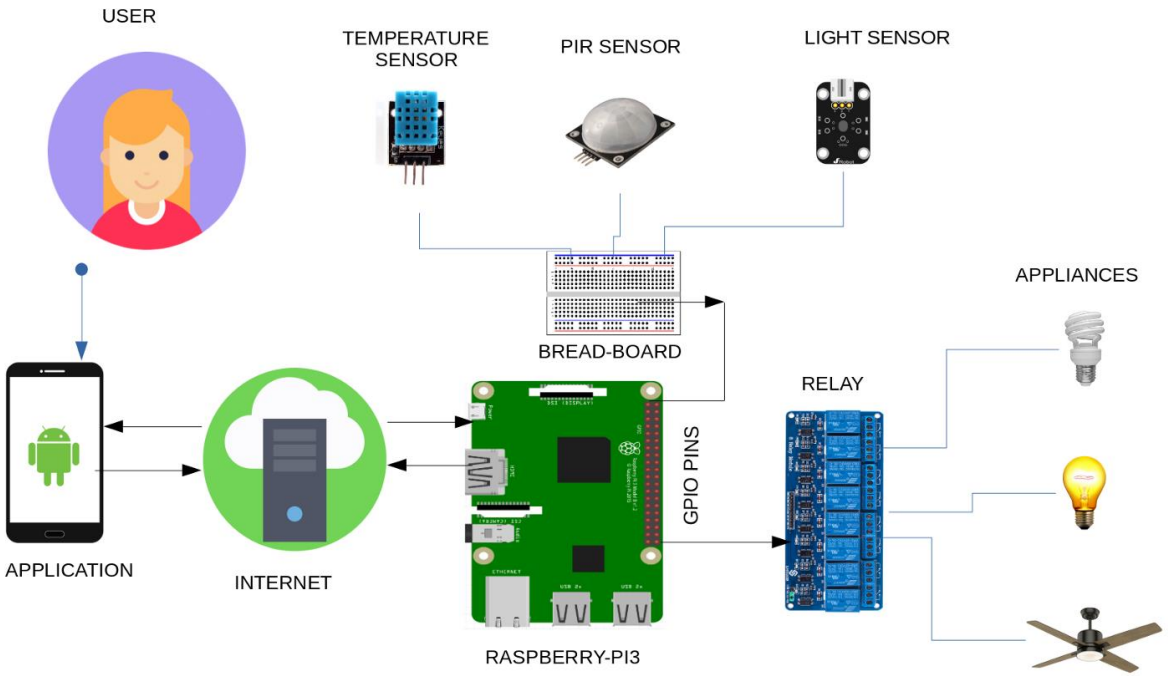
IoT Architecture: Components

- **Uniquely identifiable embedded systems (Things) connected with the internet infrastructure**
- **Node devices (Things) collect data through sensors**
- **The data is sent to the internet through the gateway**
- **Things and Gateway are generally considered as the edge (of the Internet)**
- **Data analytics takes place in the cloud or data center**
- **Following analytics, actuator commands might be sent**

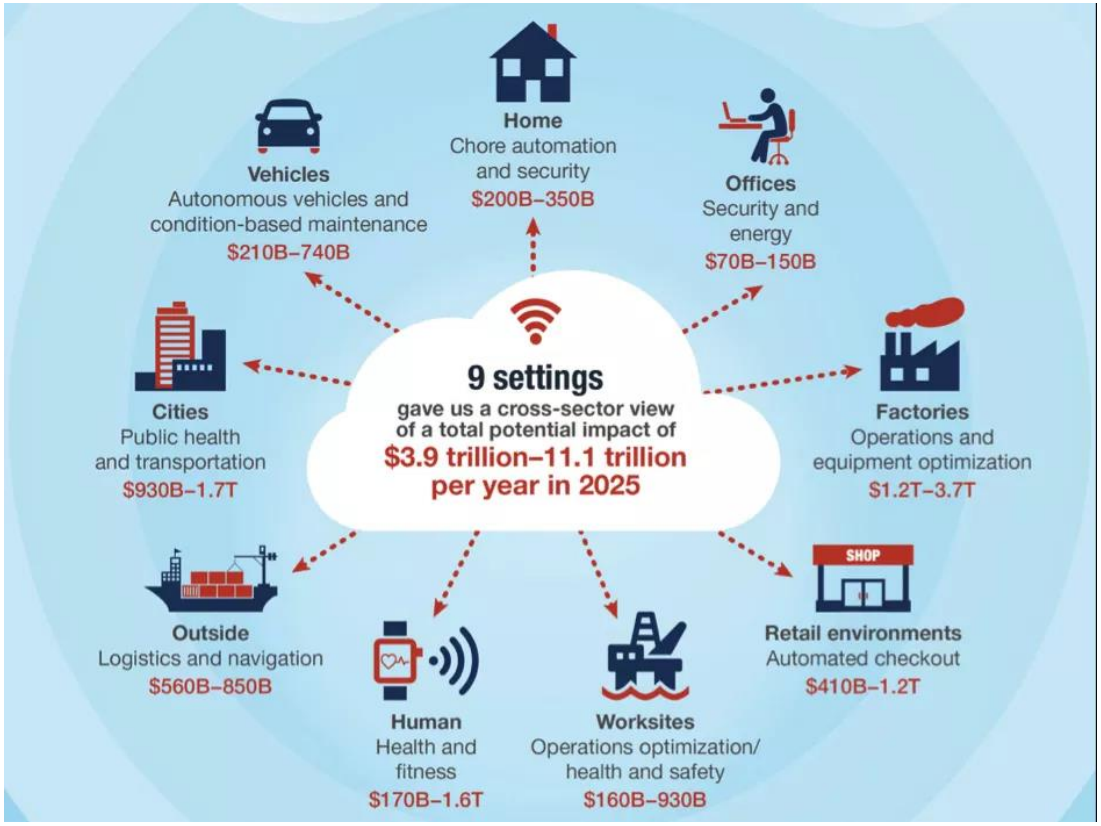


Example: Smart Home

Smart Home IoT Architecture



Endless Opportunities



Source: McKinsey & Company

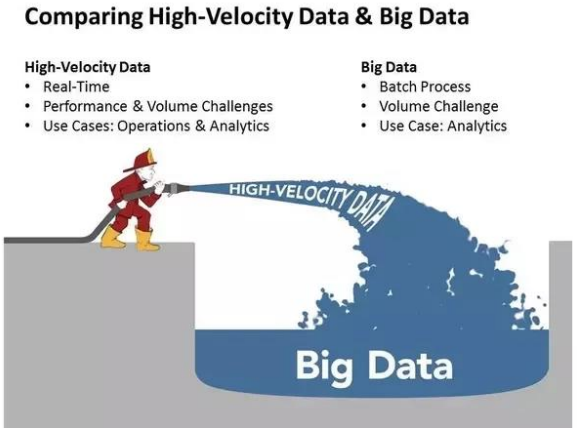
NUS National University of Singapore
School of Computing

What You Will Learn (Quick Summary):

- **My Part: The "brain" of IoT.**
  - Machine learning:
    - ✓**Statistical Methods.**
    - ✓**Neural Networks.**
    - ✓**Deep Learning Networks.**
  - Communications security.
- **Boyd's Part: The "senses and muscles" of IoT.**
  - The hardware aspects of IoT:
    - ✓**Communicating with sensors and actuators.**
    - ✓**Power management.**
    - ✓**Signal processing.**
  - Hardware security.

Understanding IoT Data

- **IoT data presents a huge challenge:**
  - Too big:
    - ✓**40 houses with smart plugs will produce 4 billion pieces of data a month.**
    - ✓**In the UK, if every house had an IoT connected meter, they are expected to produce 2.64 quadrillion pieces of data a month.**
  - Too fast:
    - ✓**Data streams from HD cameras, Lidars and other sources of information can produce over 1 GB of data per second.**
    - ✓**Often must be processed in real-time**
  - Too diverse:
    - ✓**Image data from cameras, heart rate data from ECQ, gait data from motion sensors, etc.**
    - ✓**In 2014 70% of data scientist said that diversity of data is a huge challenge that they are struggling to cope with.**



Understanding IoT Data

- **But there's lots of $$$ in IoT:**
  - In 2013 McKinsey noted a 300% growth in IoT in the previous 5 years.
  - They predicted that IoT will make an impact of US$2.7 trillion to US$6.2 trillion by 2025.
  - In 2015 these figures grew to US$4 trillion to US$11 trillion.
- **Healthcare is expected to take up to 41% of the market.**
  - This is followed by industry taking up 33%, energy market at 7%, and transportation, agriculture, urban infrastructure and retail taking up the remaining 19%.

Understanding IoT Data

- **Ways to understand IoT Data (Analytics)**
  - Descriptive:
    - ✓**Answers the question "what happened".**
    - ✓**Takes the form of describing, summarizing or presenting raw IoT data that has been gathererd.**
    - ✓**Data is decoded, interpreted in context and fused and presented in an understandable way.**
  - Diagnostic:
    - ✓**Goes one step deeper than descriptive analytics by trying to discover and present the root cause and explanation for the data.**
  - Discovery:
    - ✓**Discovers events and information that we *don't know about* and tries to explain those events.**
    - ✓**Different from previous two analytics by detecting new and novel trends and information, rather than just presenting or explaining existing data.**
- **These three techniques work on hindsight – Data that is already recorded.**

Understanding IoT Data

- **Ways to understand IoT Data (Analytics)**
  - Predictive:
    - ✓**Uses past (hindsight) data to answer the question: "What is likely to happen?"**
    - ✓**Predicts future outcomes AND the ability to assess the quality of these predictions.**
  - Prescriptive:
    - ✓**Looks at past data and predicted outcomes to suggest what to do.**
    - ✓**Enables decision makers to not only look into the future for opportunities, but also presents the best course of action to act on foresight in a timely manner.**
    - ✓**Also provides ability to evaluate "what if" scenarios.**
- **Unlike previous three techniques, these two techniques are "foresight" techniques.**

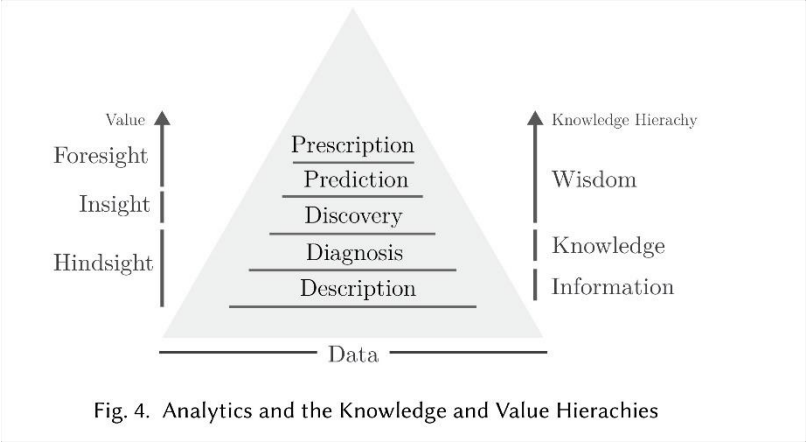


Fig. 4. Analytics and the Knowledge and Value Hierachies

Understanding IoT Data

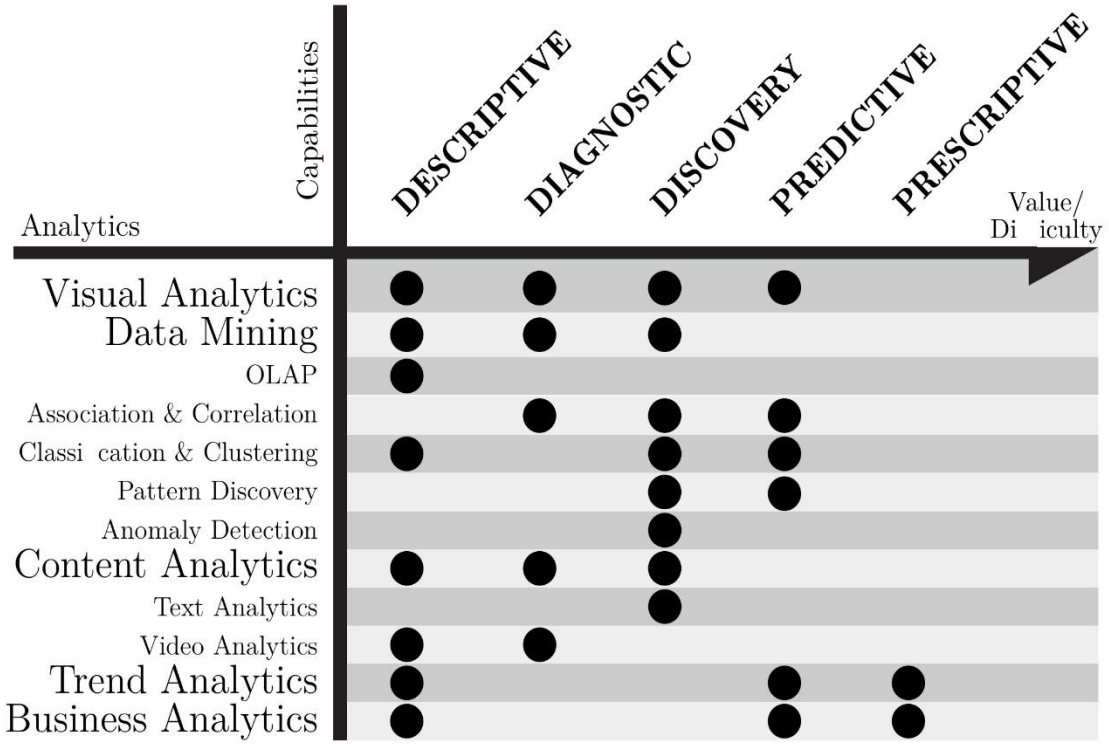


Fig. 5. Classification of Types of Analytics

Understanding IoT Data

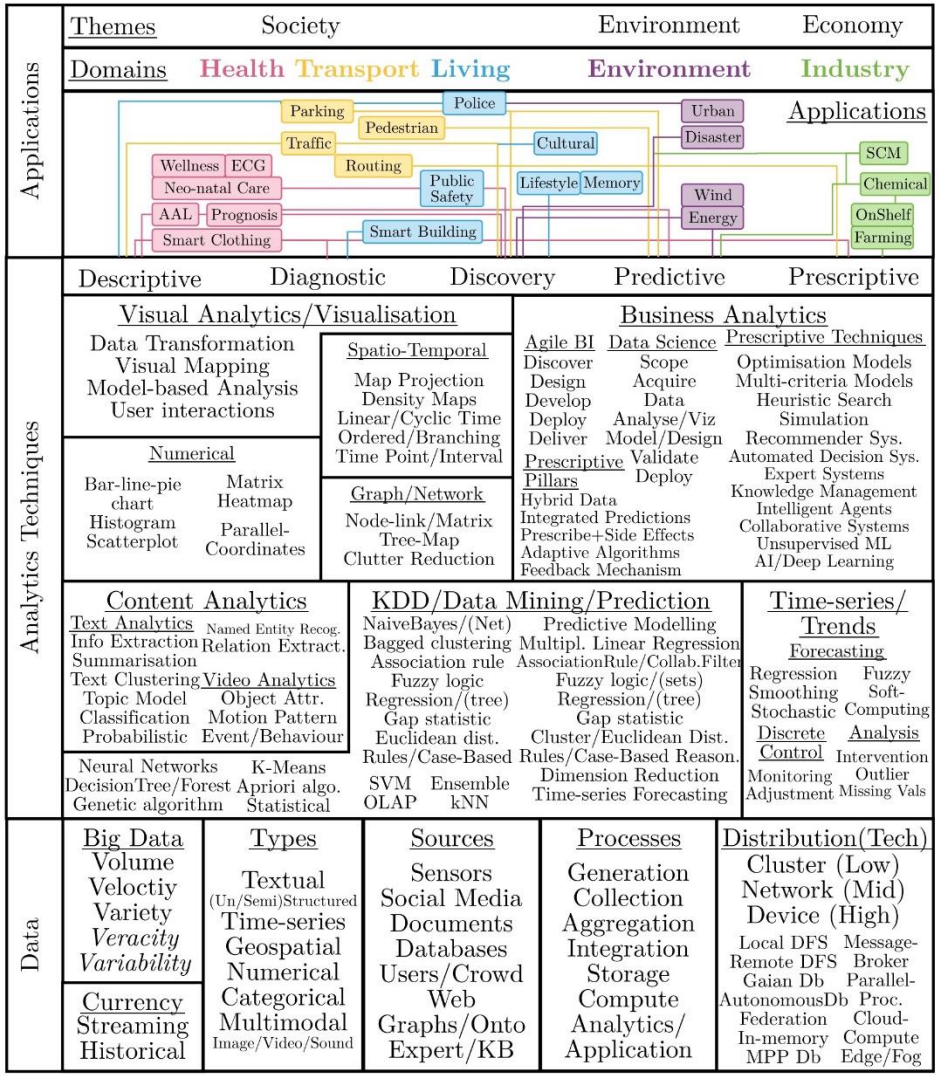
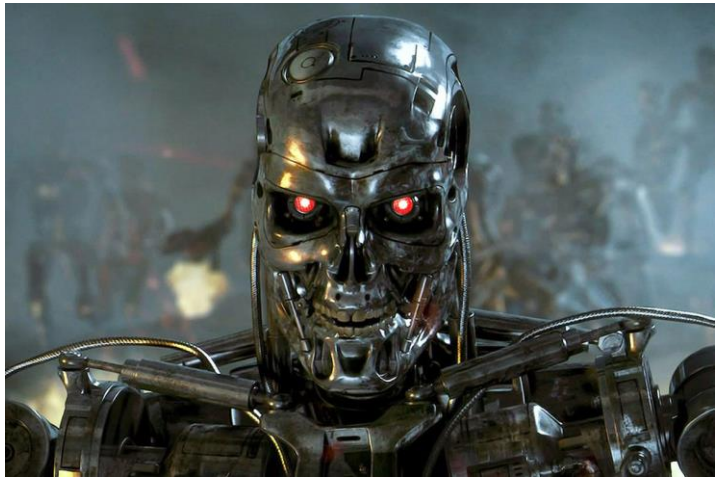


Fig. 6. Layered Taxonomy of Analytics From Data to Application

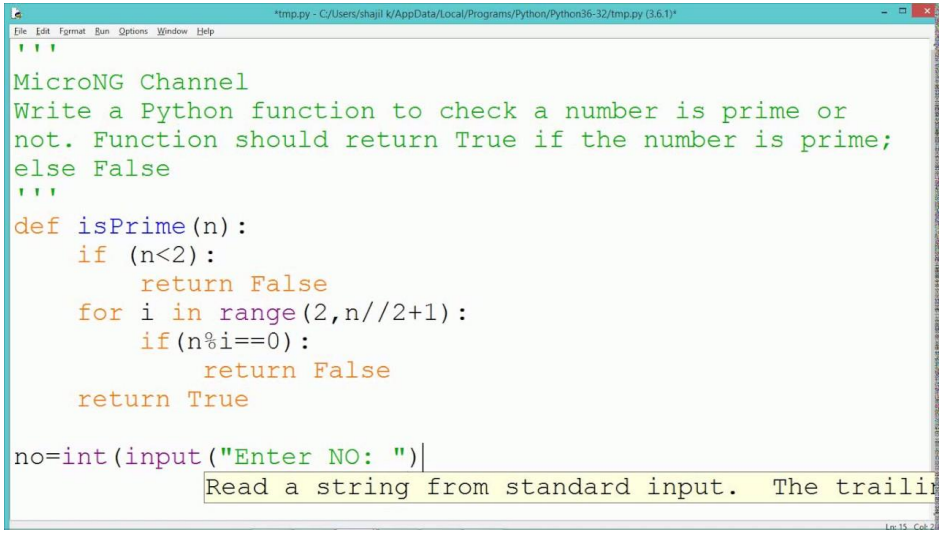
What Is Machine Learning?

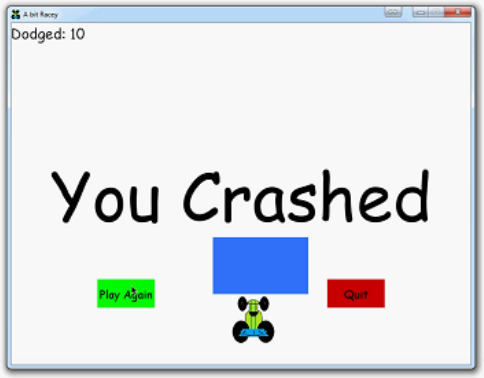
- **Terminator - Every AI Researcher's Dream.**
  - We still don't know how to build robots that behave like humans.
  - But we are getting there!



What Is Machine Learning?

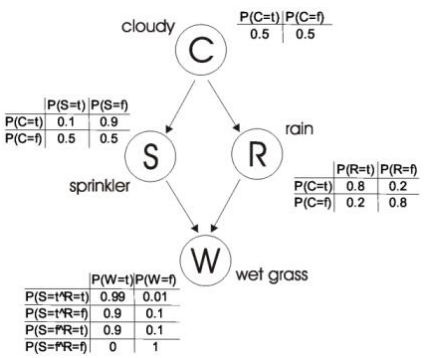
- **Seriously though, machine learning is also an alternate way of "programming":**
  - "Normal" programming in C/Python/Javascript/etc.
    - ✓**Requires you to think through step by step what you want to do.**
    - ✓**Requires you to translate the steps into a formal language that the computer can understand.**

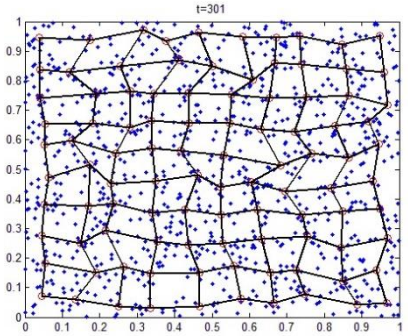


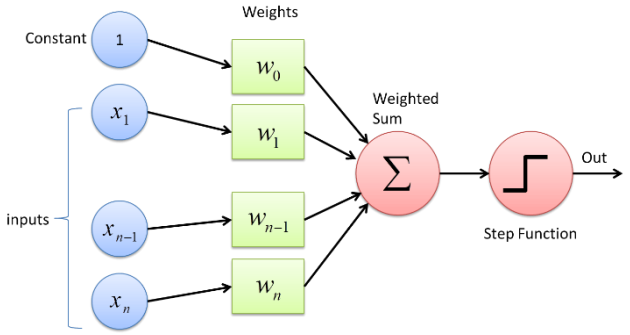


What Is Machine Learning?

- In machine learning:
  - ✓**We present the machine with A LOT of data.**
  - ✓**Machine learns on its own to recognize patterns in the data.**







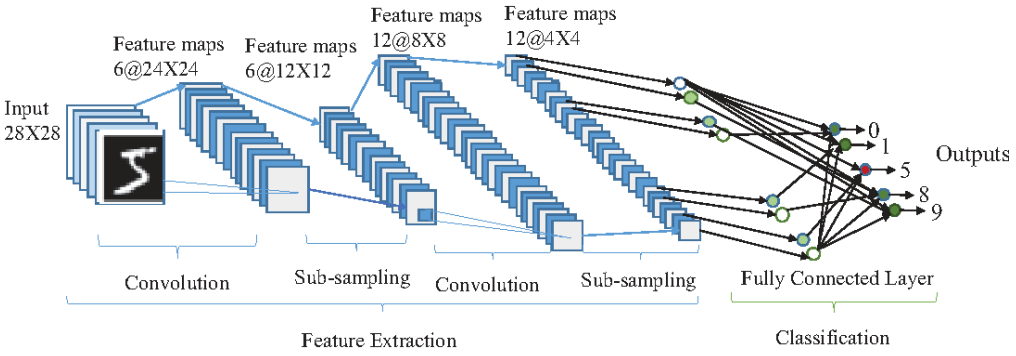
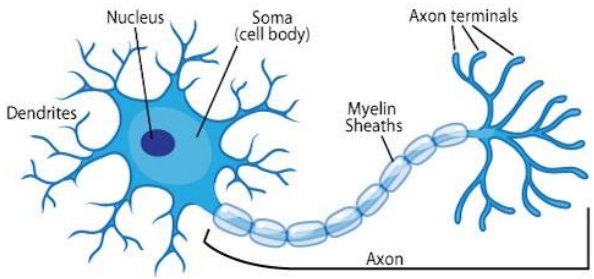
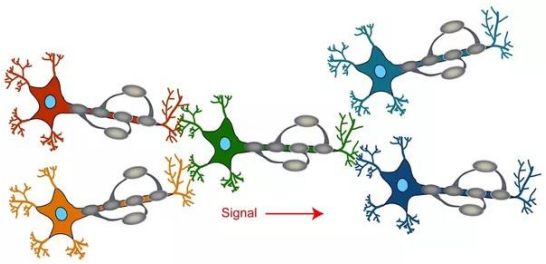


Fig. 1. CNN Block Diagram

Neural Networks: Biological Inspiration

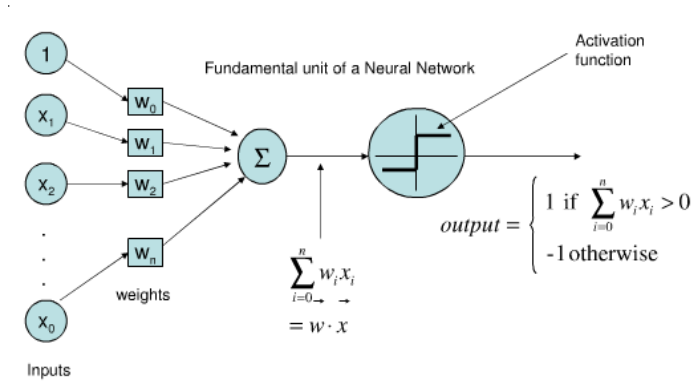
- **Neural networks are inspired by biological neurons: Units of computation that are in our brain and nervous system:**
  - Each neuron takes electrical signals from surrounding neurons.
  - Signals pass through gaps called "synapses" that pass a portion of the signal through.
    - ✓**"Strong" synapses pass more signals, "weak" synapses pass weaker signals.**
  - The neuron sums these signals and fire if they exceed a "threshold".
  - To lean, the synapses are strengthened or weakened.

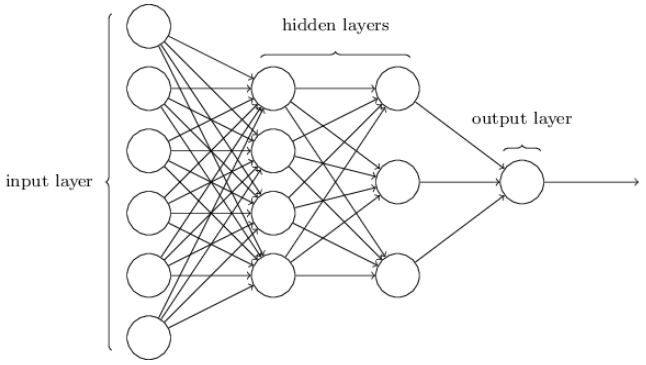




Neural Networks: Machine "Neurons"

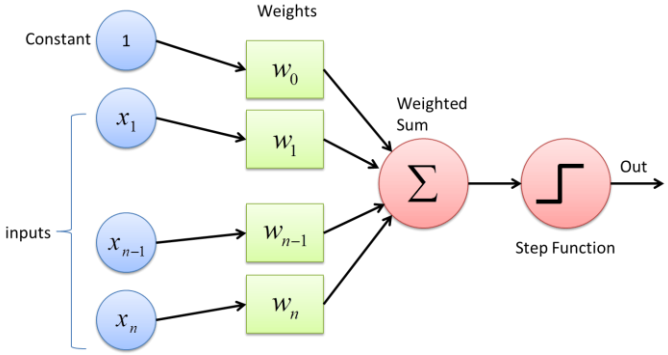
- **Many neural networks consist of computational units also called "neurons", often called "nodes":**
  - Like the biological neurons, the "neurons" here also sum inputs through weights.
  - The neuron "fires" (outputs a "1") if the sum exceeds a threshold, through an "activation function".
  - The summation is essentially a dot product $w^T x$, and the neuron can be specified by $g(w^T x)$ where $g(.)$ is the "activation function".

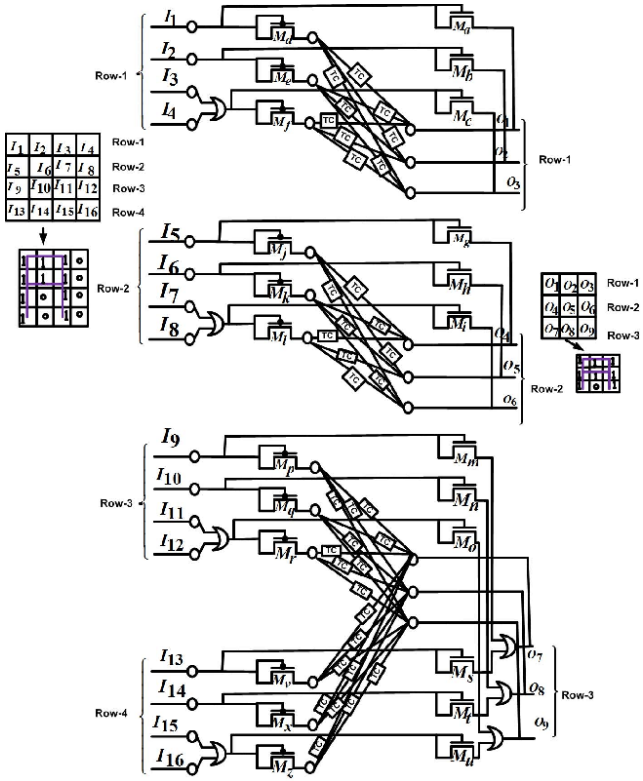




Neural Networks

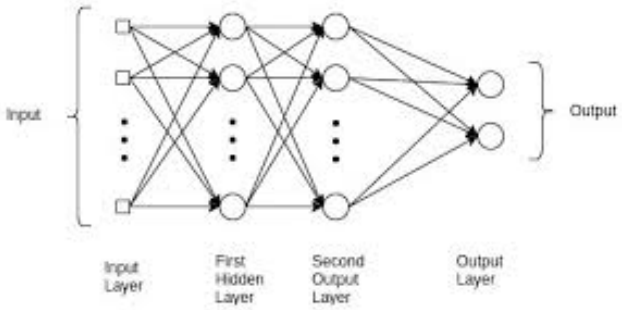
- **Neural network research has been around a long, long time.**

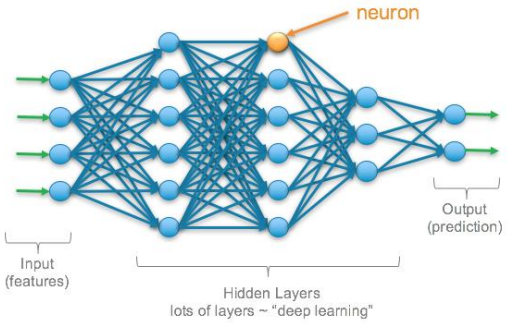


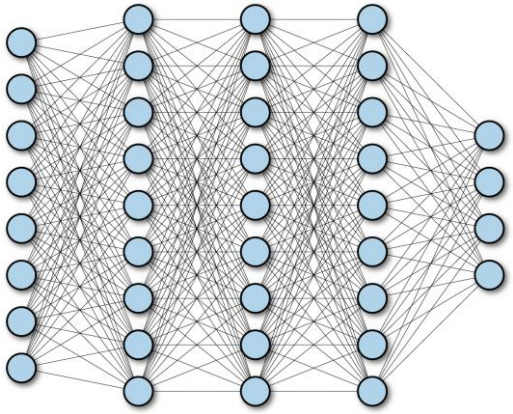


Deep Learning

- **Primary limitation of neural networks:**
  - Neural networks learn by adjusting "parameters" according to a "learning law" (We will cover learning laws in Lecture 3), based on the data presented.
  - Need large number of "parameters" to learn highly complex tasks.
    - ✓**E.g. image classification**
  - Curse of dimensionality:
    - ✓**As the number of parameters increases, the amount of training data required grows exponentially.**
    - ✓**Otherwise the neural network "overfits" – It memorizes the data it is presented instead of learning from it.**

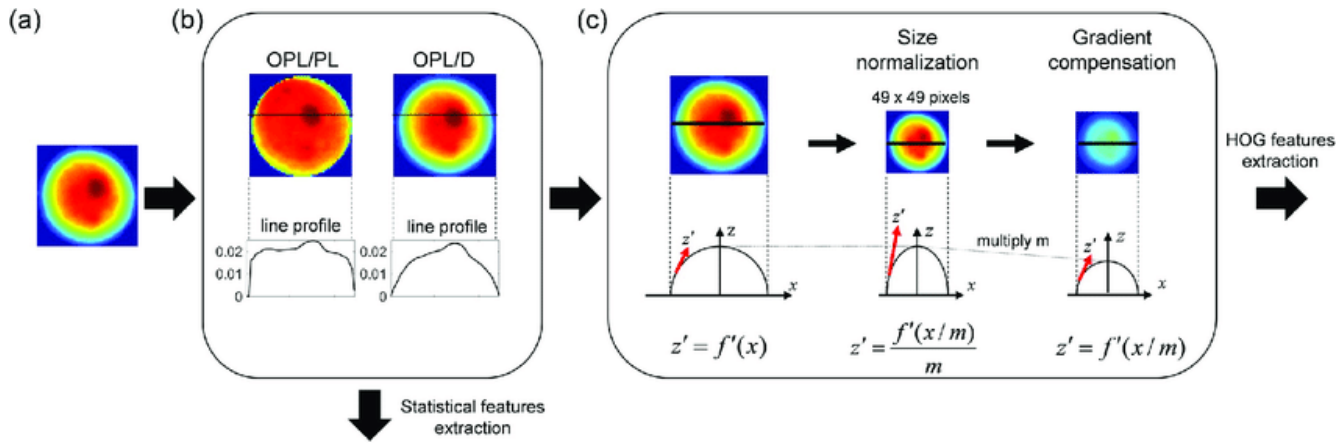






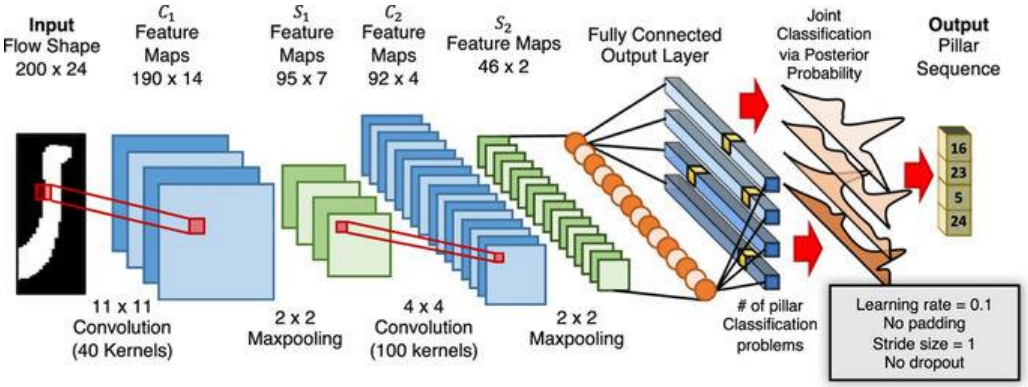
Deep Learning

- In Deep Learning we apply various techniques to “simplify” the data:
  - Extract features.
  - Make the data invariant:
    - ✓Shift invariance.
    - ✓Scale invariance.



Deep Learning Example: Convolutional Neural Networks

- **A CNN is a deep network made up of many layers:**
  - Convolution (kernel) layers consisting of small windows that scan over an image.
  - Pooling layers that summarize the previous convolution layer to ease learning.
  - Dropout layers that randomly drop information from previous layers to prevent memorizing (overfitting) of training data.
  - Dense (fully-connected) layers that learn from previous layers to classify images.



Deep Learning Example: Convolutional Neural Networks

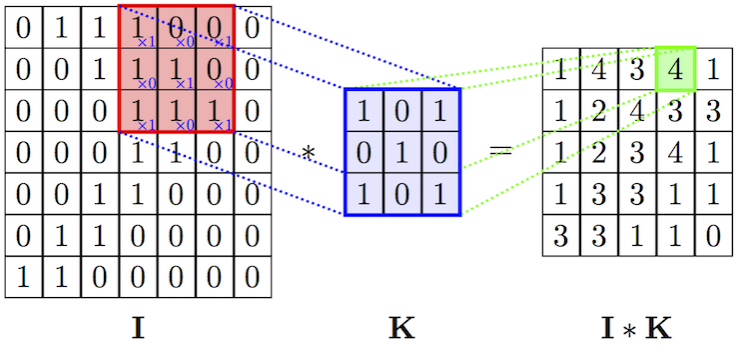
- **Let's look at CNNs in more detail:**
  - Problem statement: Computers can't "see" images the way we see them.

What We See

What Computers See

Deep Learning Example: CNN: Convolution

- **The first layer of a CNN is the convolution layer.**
  - We take a square window (e.g. 5 x 5) called "filters" of randomly initialized values that we pass over an image, forming a summary of that image called an "activation map":

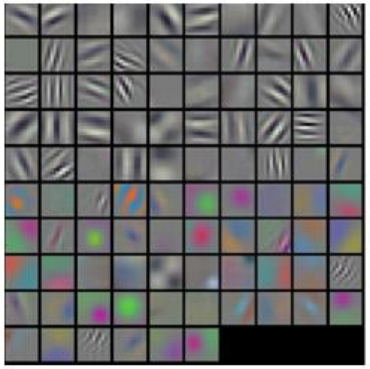


  - Formally the convolution operation is written as:

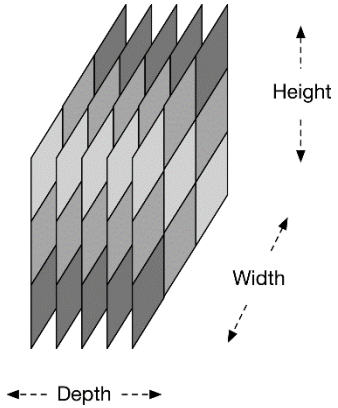
$$(I * K)_{xy} = \sum_{i=1}^{h} \sum_{j=1}^{w} K_{ij} \cdot I_{x+i-1,y+j-1}$$

Deep Learning Example: CNN: Convolution

- **We will take many (~32 to 64) of these filters and scan over the image, producing an entire volume of activation maps.**
- **Through training (usually gradient descent), the filters learn key parts of the image:**



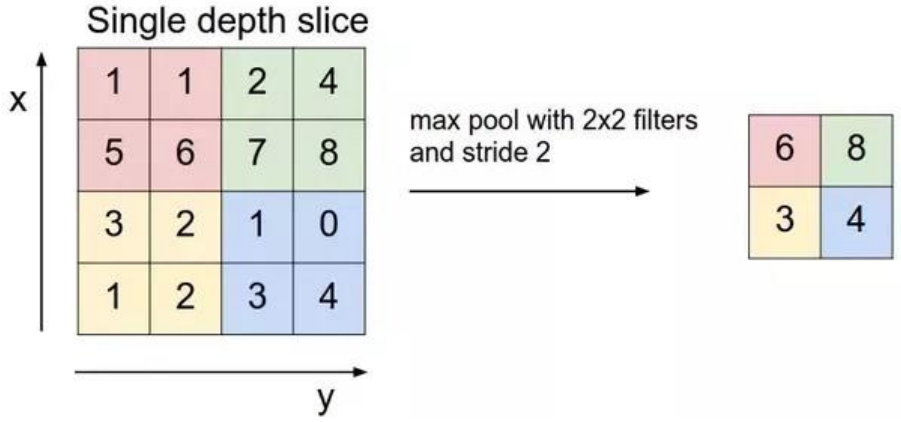
Visualizations of filters



- **We then apply ReLU activation to introduce non-linearity.**

Deep Learning Example: CNN: Convolution

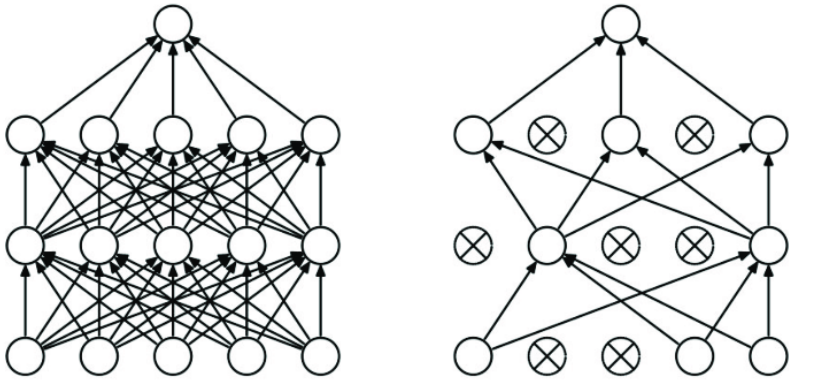
- **One issue now is the large amount of data that the filters generate.**
- **We now slide a second smaller window over the activation maps and either:**
  - Take the maximum value in the window
  - Take the average value in the window
  - ?
- **The idea is to summarize the contents of the window. This is called "pooling".**



- **Note: the distance the the pooling window moves is called its "stride".**

Deep Learning Example: CNN: Dropout

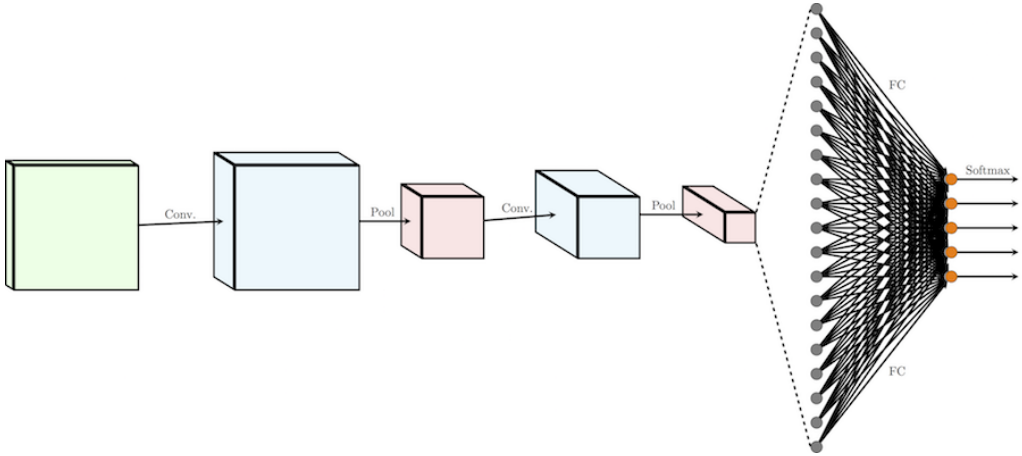
- **To prevent the CNN from memorizing the data (making it hard to recognize unseen data), information from previous layers is randomly dropped during each training iteration.**
  - In our practical work we will drop as much as 40% of previous information per iteration.



(a) Standard Neural Network (b) Neural Net with Dropout

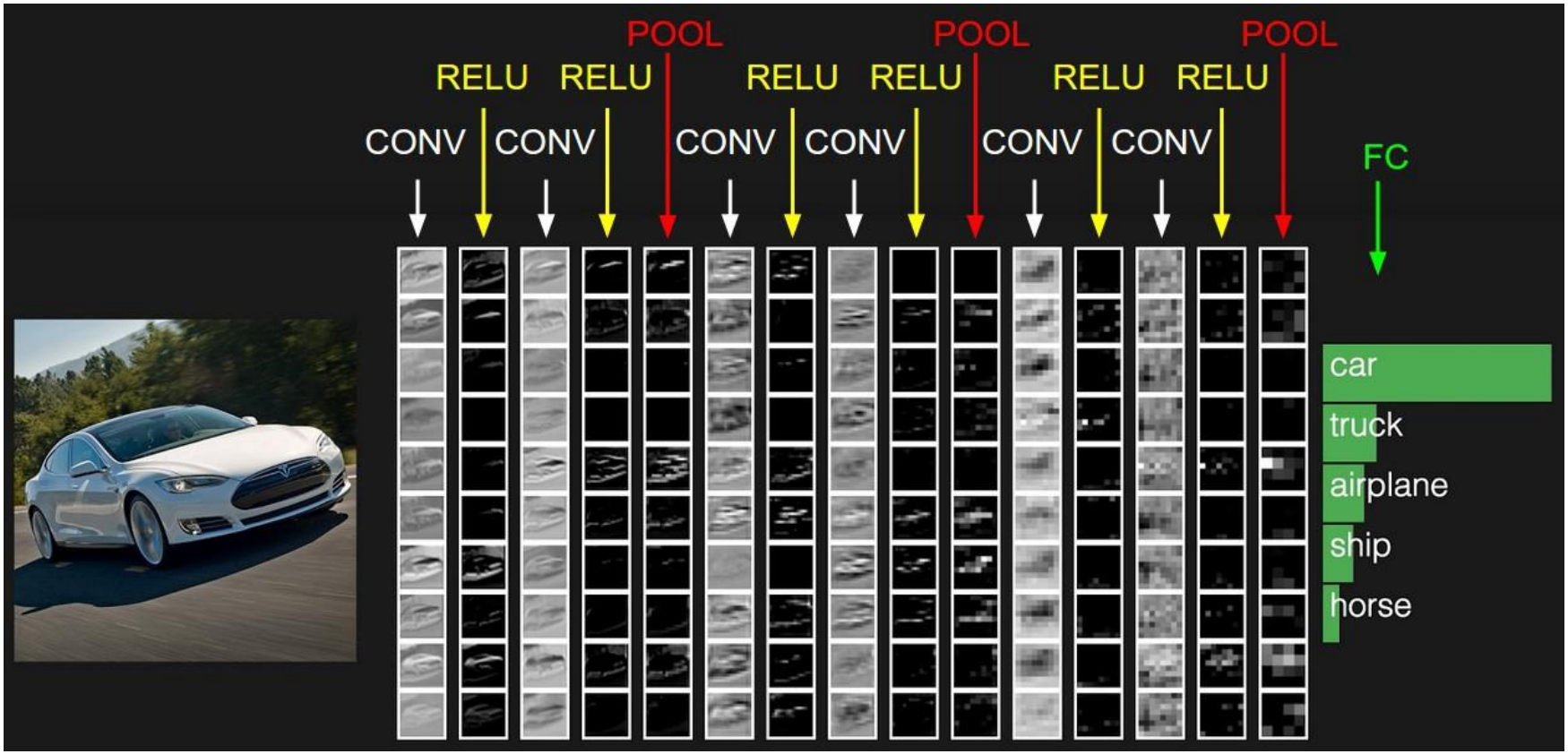
Deep Learning Example: CNN: Dense Layers

- **The last layers of a CNN are the dense layers:**
  - These are fully connected neurons, like what we've seen in SLPs and MLPs.
  - It is in these layers that we present our labels for each image, to teach the CNN.



- **Training of convolution and dense layers is usually by gradient descent with backpropagation.**

Deep Learning Example: CNN: Overall View



How Deep Learning is used in IoT

- **Example Healthcare Applications:**
  - Ambient Assisted Living
    - ✓**Sensors in the homes of the elderly gather data about their ambient environment.**
    - ✓**Healthcare sensors on the elderly gather physiological data.**
    - ✓**The ambient sensor data provides context for the physiological data, and ontological models and inferencing rules give healthcare information to the elderly, and provide updates to care-givers and to contact hospitals in emergencies.**
  - Smart Clothing
    - ✓**Smart clothing with sensors that collect ECG, body temperature and other physiological data.**
    - ✓**Coupled with visualisations on a mobile app for fitness and other monitoring.**
    - ✓**Data uploaded to a "Health Cloud" where deep learning monitors for health trends and abnormal data to alert healthcare professionals.**
    - ✓**Analytics aid in prognosis, predicting the future health of the patients based on data collected from similar patients in the past.**

How Deep Learning is used in IoT

- **Example Healthcare Applications:**
  - Human activity recognition with wearable sensors.
    - ✓**Uses wearable accelerometers, gyroscopes etc to measure movements of a person.**
    - ✓**Uses four convolutional layers with ReLUs, fllowed by two LSTM layers and a softmax (fully connected) layer to understand what the wearer is doing.**
  - Human activity recognition with mobile phone data.
    - ✓**Similar to above but using sensor data from mobile phones.**
  - Trauma Room Activity Recognition
    - ✓**Use of passive RFID tags to detect trauma room activities like blood pressure measurement, mouth examination etc using a deep CNN.**
  - Diet monitoring
    - ✓**Cameras feeding CNNs used to recognize food types and portions.**
    - ✓**Combined with nutritional data to monitor dietary intake.**

How Deep Learning is used in IoT

- **Example Healthcare Applications:**
  - Use of mammogram data – usually used for diagnosing breast cancer – to detect coronary artery diseases.
    - ✓**Using 12-layer CNN to detect signs of breast arterial calcification.**
    - ✓**Accuracy matches human experts.**
- **Other applications:**
  - CNN used for indoor localization of a person using both magnetic and visual sensor data.
  - DeepFi:
    - ✓**Use of WiFi fingerprinting and data usage patterns to locate a person accurately.**
  - Video analytics to detect traffic volumes for planning, highlighting incidents and enhancing safety.

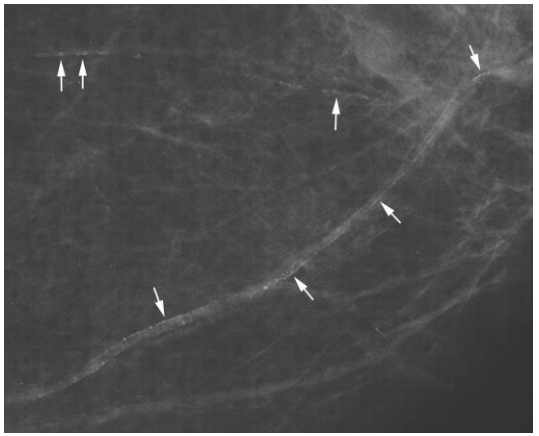


TABLE VI
DNN SIZES AND COMPLEXITIES IN DIFFERENT APPLICATIONS. (NA: NOT AVAILABLE, C: CONVOLUTIONAL LAYER, F: FULLY CONNECTED LAYER, FF: FEEDFORWARD LAYER, SG: SUMMATION (COLLAPSE) LAYER, P: POOLING LAYER, LRN: LOCAL RESPONSE NORMALIZATION LAYER, SM: SOFTMAX LAYER, L: LSTM LAYER, R: RNN LAYER, RBM: RBM HIDDEN LAYER)

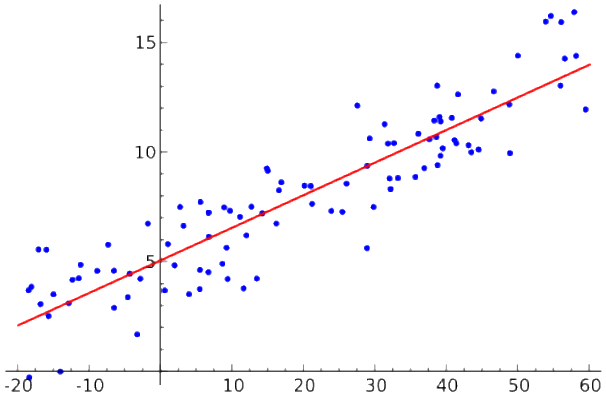
Work	Application	Type of DNN	Depth	Layers Sizes	Training Time	Test Time
[79]	Transportation analysis	RNN+RBM	2	R(100)-RBM(150)	NA	354 (s), whole test set
[92]	Localization	RBM DBN	4 4	500-300-150-50 300-150-100-50	NA	NA
[93]	Localization	SdA DBN ML-ELM SDELM	4 3 5 5	26-200-200-71 26-300-71 26-300-300-1500-71 26-300-300-1500-71	451 (s) 110 (s) 14 (s) 24 (s)	NA
[94]	Localization	SdA	5	163-200-200-200-91	NA	0.25 (s)
[98]	Pose detection	CNN	12	C(55×55×96)-LRN-P- C(27×27×256)-LRN-P- C(13×13×384)- C(13×13×384)- C(13×13×256)-P- F(4096)-F(4096)-SM	NA	0.1 (s)
[100]	Human activity detection	CNN+LSTM	7	C(384)-C(20544)-C(20544)- C(20544)-L(942592)- L(33280)-SM	340 (min)	7 (s), whole test set
[101]	Human activity detection	LSTM	5	L(4)-FF(6)-L(10)-SG-SM	NA	2.7 (ms)
[107]	FDI detection	DBN	4	50-50-50-50	NA	1.01 (ms)
[109]	Malware detection	DBN	3	150-150-150	NA	NA
[120]	Parking space	CNN	8	C(64×11×11)-C(256×5×5)- C(256×3×3)-C(256×3×3)- C(256×3×3)-F(4096)-F(2)-SM	NA	0.22 (s)
[126]	Traffic sign detection	CNN	6	C(36×36×20)-P- C(14×14×50)-P- FC(250)-SM	NA	29.6 (ms) on GPU 4264 (ms) on CPU
[128]	food recognition	CNN	22	Used GoogLeNet [75]	NA	50 (s)
[135]	Crop recognition	CNN	6	C(96×7×7)-P- C(96×4×4)-P-F(96)-F(96)	12 (h)	NA
[145]	Classroom Occupancy	CNN	5	C(6×28×28)-P- C(16×10×10)-P-F(120)	2.5 (h)	2 (s) (4 thread)
[146]	Fault diagnosis	AE	4	300-300-300-300	NA	91 (s)
[152]	Road damage detection	CNN	8	Used AlexNet [37]	NA	1.08 (s)
[155]	Classifying offensive plays	RNN	3	10-10-11	NA	10 (ms)

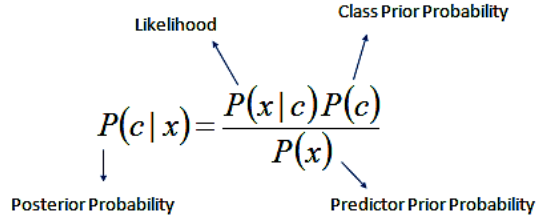
TABLE I
SUMMARY OF DEEP LEARNING MODELS.

Model	Category	Learning model	Typical input data	Characteristics	Sample IoT Applications
AE	Generative	Unsupervised	Various	• Suitable for feature extraction, dimensionality reduction • Same number of input and output units • The output reconstructs input data • Works with unlabeled data	• Machinery fault diagnosis • Emotion recognition
RNN	Discriminative	Supervised	Serial, time-series	• Processes sequences of data through internal memory • Useful in IoT applications with time-dependent data	• Identify movement pattern • Behavior detection
RBM	Generative	Unsupervised, Supervised	Various	• Suitable for feature extraction, dimensionality reduction, and classification • Expensive training procedure	• Indoor localization • Energy consumption prediction
DBN	Generative	Unsupervised, Supervised	Various	• Suitable for hierarchical features discovery • Greedy training of the network layer by layer	• Fault detection classification • Security threat identification
LSTM	Discriminative	Supervised	Serial, time-series, long time dependent data	• Good performance with data of long time lag • Access to memory cell is protected by gates	• Human activity recognition • Mobility prediction
CNN	Discriminative	Supervised	2-D (image, sound, etc.)	• Convolution layers take biggest part of computations • Less connection compared to DNNs. • Needs a large training dataset for visual tasks.	• Plant disease detection • Traffic sign detection
VAE	Generative	Semi-supervised	Various	• A class of Auto-encoders • Suitable for scarcity of labeled data	• Intrusion detection • Failure detection
GAN	Hybrid	Semi-supervised	Various	• Suitable for noisy data • Composed of two networks: a generator and a discriminator	• Localization and wayfinding • Image to text
Ladder Net	Hybrid	Semi-supervised	Various	• Suitable for noisy data • Composed of three networks: two encoders and one decoder	• Face recognition • Authentication

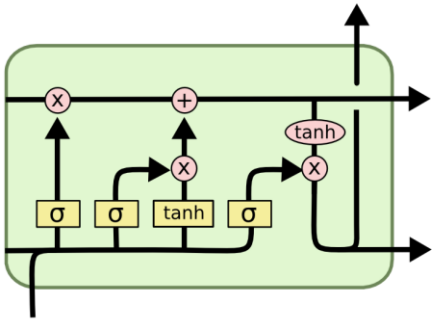
What We Will Be Doing

- **We have just looked at how deep learning is being used in IoT.**
- **We will now go through all the different algorithms we will be looking at.**
  - Think about what you proposed for your project.
  - See how you can apply these algorithms for your project.
  - Get fresh new ideas for your project!





$$P(c \mid \mathbf{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$



What We Will Be Doing

- **Statistical Models:**
  - Linear Regression
    - ✓**Given a set of data, produce a function that can predict outcomes for unseen data.**
    - ✓**E.g. Given historical room temperature by time and by season, predict the level of air-conditioning/heating needed for tomorrow.**
  - Naïve Bayes Classifier
    - ✓**Model the statistical characteristics of the data.**
    - ✓**Classify new data based on these characteristics.**
    - ✓**Good for classifying data – E.g. given a series of temperature and light readings, what season are we in? (Autumn, spring?)**
  - Decision Trees
    - ✓**Look at the data and infer how decisions were made.**
  - Support Vector Machines
    - ✓**Look at the data and create "hyperplanes" to separate different classes of data.**
    - ✓**Good for classifying data – More complex than Naïve Bayers but more robust.**

What We Will Be Doing

- **Connectionist Models:**
  - Unsupervised Learning:
    - ✓**Machine looks at data, and figures how on its own how to sort out the data into classes.**
    - ✓**Good for "clustering" – Is this colour green, yellow, or greenish-yellow?**
    - ✓**Algorithms:**
      - *–k-Means Clustering*
      - *–Kohonen Self Organizing Maps*
    - ✓**Kohonen Self Organizing Map can be automatically trained to sort out colours to do colour classification.**
    - ✓**No need for tagged data.**

What We Will Be Doing

- **Connectionist Models:**
  - Supervised Learning:
    - ✓**These are great when you have enough "tagged" data to teach the computer.**
    - ✓**Creation / acquisition of tagged data can be extremely expensive.**
    - ✓**New techniques like Generative Adverserial Networks (GANs) can produce "fake" data that looks like real data, to augment real data sets.**
    - ✓**Algorithms:**
      - *–Perceptrons (Great grandfather of deep learning)*
      - *–Multilayer Perceptrons*

What We Will Be Doing

- **Deep Learning Models:**
  - Recurrent Neural Networks
    - ✓**Great for learning patterns across time.**
  - Long-Short Term Memories
    - ✓**Like RNNs but much more robust.**
  - Convolutional Neural Networks
    - ✓**Most commonly used for images, but can be used elsewhere.**
    - ✓**Key strength are its convolutional layers (detects key features) and pooling layers (summarizes features / removes noise).**
    - ✓**We will shortly see several CNN applications in healthcare.**

What We Will Be Doing

- Generative Adversarial Networks
  - ✓**Forgers and police.**
  - ✓**Forgers fake data, police try to catch them. Forgers improve their fakery to avoid getting caught.**
  - ✓**Used to generate more data from existing samples of data.**
- Autoencoders
  - ✓**Train neural networks on this function f(x) = g(x).**
  - ✓**Why?**
    - *–Assume that g(x) is the output of some system, e.g. torque from an engine, and x is the input to the engine, e.g. throttle position.*
    - *–When f(.) is fully trained, we will get $|f(x) – g(x)| < \varepsilon$ where ε is some very small value (f(x) is never exactly g(x) because of noise in g(x), e.g. variations in oxygen levels in the air taken in by the engine, variations in temperature, etc.)*
    - *–If at some point $|f(x) – g(x)| > k\varepsilon$ where k is some constant, we know that g(x) is producing abnormal values, possibly signalling a fault.*
  - ✓**Autoencoders are thus very good at detecting faults in a system.**

What We Will Be Doing

- **Using Machine Learning is HARD!! We will look at:**
  - Network sizing.
    - ✓**Kernel dimensions?**
    - ✓**# of layers?**
    - ✓**Pooling?**
    - ✓**# of neurons in each layer?**
  - Selection of training hyper-parameters:
    - ✓**Learning algorithms**
      - *–Stochastic Gradient Descent?*
      - *–Adam?*
      - *–AdaMax?*
      - *–???*
    - ✓**Learning parameters**
      - *–Learning Rate*
      - *–Decay rate*
      - *–Momentum*

What We Will Be Doing

- Selecting Transfer Functions
  - ✓**Identity?**
  - ✓**Sine, cosine?**
  - ✓**Sigmoid?**
  - ✓**ReLU, Leaky-ReLU?**
- Dealing with Overfitting
  - ✓**Dropouts?**
  - ✓**L1 or L2 regularizers? Or none? Or both?**
  - ✓**How much L1 and how much L2 to use?**
- Getting Data
  - ✓**Hunting for data.**
  - ✓**Cleaning up the data.**
  - ✓**Finding more data.**
  - ✓**DATA DATA DATA!**

NUS National University of Singapore
School of Computing

CS3237 Introduction to IoT (Deep Learning)

- **Lectures will be a mix of theory and hands-on.**
  - Language used: Python.
  - Environment:
    - ✓**Lecture hands-on using Jupyter.**
    - ✓**Labs: Any IDE you prefer. But real programmers use vim. ;)**
- **There will be a mix of weekly labs and tutorials (but mostly labs).**
  - Covers key topics we don't do in the lecture:
    - ✓**Creating and training statistical, neural network and deep learning models.**
    - ✓**Secure communications over MQTT**
    - ✓**Generation of crypto keys**
- **This is a HEAVY but FUN course with a lot of work!**

Installing the Course Environment

- **IMPORTANT NOTE:**
  - Windows will NOT be supported in this course.
    - ✓**Windows has poor support for some of our development tools, e.g. MQTT.**
    - ✓**When something is supported, the Windows way of doing things is freaky, unintuitive and just plain slow and painful (e.g. MySQL)**
    - ✓**The Registry – basically bits of chicken poop that Windows drops on your system.**
    - ✓**It is SO MUCH easier to work on the command-line, and Windows has very poor CLI support.**
  - Thus if you use Windows:
    - ✓**Install Ubuntu on Linux Subsystem for Windows (https://docs.microsoft.com/en-us/windows/wsl/install-win10), OR**
    - ✓**Dual boot your machine to Ubuntu, OR**
    - ✓**You're on your own if something goes awry.**

Installing the Course Environment (Does not work for M1 Macs – See next page)

- **You will need to install Python 3.x (preferably 3.7 or better). Python 2.x will NOT be supported.**
- **Once installed:**
  1. Create a course directory. E.g. cs3237cool.
  2. Change to that directory.
  3. Create a virtual environment called venv: `python –m venv venv`
  4. Activate the environment:

     ```
     source venv/bin/activate
     ```
  5. Install the following: TensorFlow, keras, NumPy, Scikit-Learn, Flask, requests, Pillow:

     ```
     pip3 install tensorflow keras numpy sklearn flask
     requests pillow
     ```

  Note: To exit the virtual environment, enter the following command:

  ```
  deactivate
  ```

Installing the Course Environment (For M1 Macs)

1. **Download mini forge Miniforge3-MacOSX-arm64 from https://github.com/conda-forge/miniforge**
2. **Install mini forge with: bash ./Miniforge3-MacOSX-arm64.sh**
3. **Create a course directory. E.g. cs3237cool: mkdir cs3237cool**
4. **Change to that directory: cd cs3237cool**
5. **Copy the cs3237_env.yml file from the Lecture 1 zip file into this directory.**
6. **Create a conda environment called cs3237_env with the cs3237_env.yml file: conda env create --file=cs3237_env.yml --name=cs3237_env**

Installing the Course Environment (For M1 Macs)

7. **Activate the environment: conda activate cs3237_env**
8. **Install TensorFlow and TensorFlow Addons: pip install --upgrade --force --no-dependencies https://github.com/apple/tensorflow_macos/releases/download/v0.1alpha3/tensorflow_macos-0.1a3-cp38-cp38-macosx_11_0_arm64.whl https://github.com/apple/tensorflow_macos/releases/download/v0.1alpha3/tensorflow_addons_macos-0.1a3-cp38-cp38-macosx_11_0_arm64.whl**
9. **Note: To exit the conda environment, enter the following command: condo deactivate**

Grading

- **Take-home Quiz: 5%**
- **Labs: 25%**
- **Term Project: 70%**

Conclusion

- **We have looked at:**
  - Challenges in understanding IoT data.
  - Basic ideas of machine learning.
  - The need for deep learning.
  - Example real-world / research applications of deep learning in IoT.
  - What we are going to do.