Introduction to Artificial Intelligence
And
Machine Learning

Suresh Manandhar
Naamii, Nepal

Talk Summary

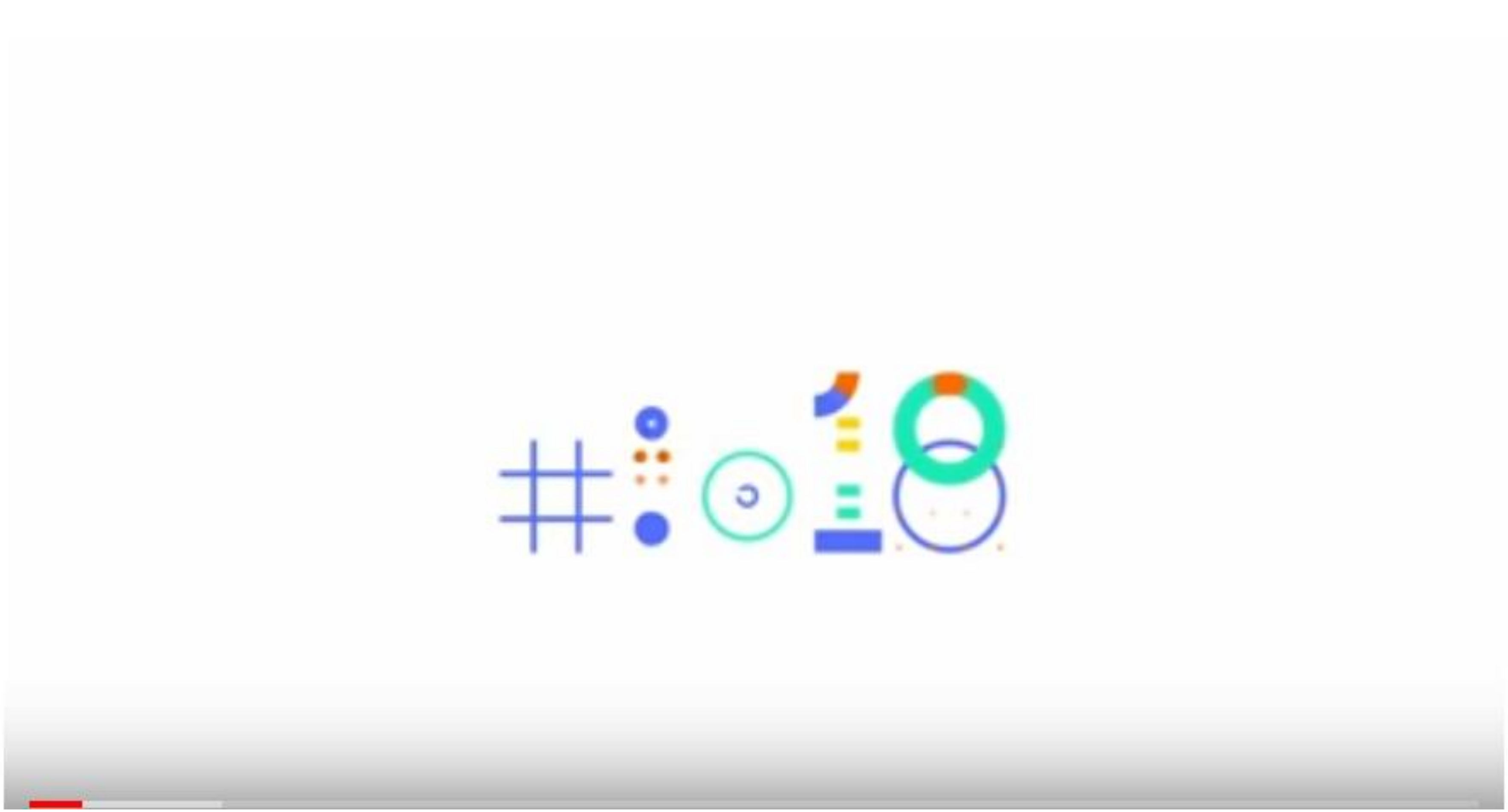
- What is Artificial Intelligence?
- Overview of ***some*** key AI areas:
  - Knowledge representation and reasoning
  - Machine learning
  - Image processing
  - Natural Language Processing

Example: Saving coral with AI

Example: Saving coral with AI (with link)

Example: Google duplex

Example: Google duplex (with link)



AI Overview

What is AI as a discipline?

AI is a multidisciplinary area that draws from:

- Philosophy
- Linguistics
- Mathematics
- Engineering
- Computer Science
- Neuroscience
- Psychology
- Biology

What are the key goals of AI?

- Develop better understanding of intelligence
  - what does being intelligent means?
- Develop computational methods for simulating intelligent behaviour
- More specifically:
  - Develop machines that can mimic human reasoning processes
  - Develop language and vision capability
  - Develop human like embodied agents (robots)
  - Ultimately perhaps machines that are human like or better

Knowledge representation and reasoning

Knowledge representation and reasoning

Goals of Knowledge representation:

- Codify human knowledge in a form that a computer can reason with
- Develop representation language for representing common sense knowledge and knowledge about the world
- Common use case : storing data extracted from the web

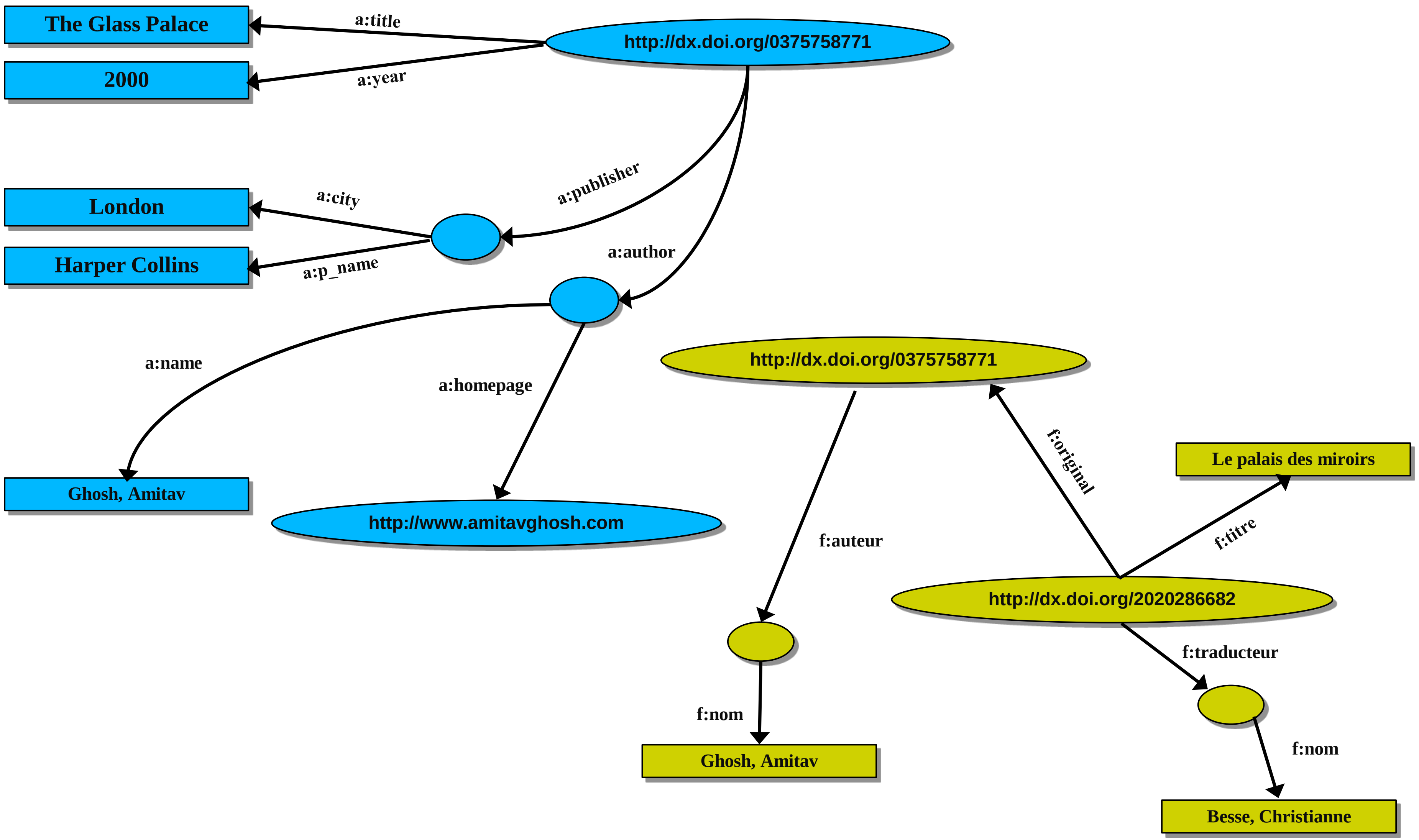
Current techniques:

- Symbolic knowledge representation focusses on representing knowledge using logic typically variants of *first order logic*.

Knowledge Graphs and Ontologies

- Logic based representations can be easily represented as graphs
- This gives rise to graph databases also known as **knowledge graphs** or **NoSQL databases**

Knowledge graph example



Source: **Ivan Herman** (**W3C**) reproduced here with kind permission.

Knowledge Graphs and Ontologies

- Logic based representations can be easily represented as graphs
- This gives rise to graph databases also known as **knowledge graphs** or **NoSQL databases**
- Ontologies are logic based rules
- Ontologies replace SQL Schemas (in relational databases)

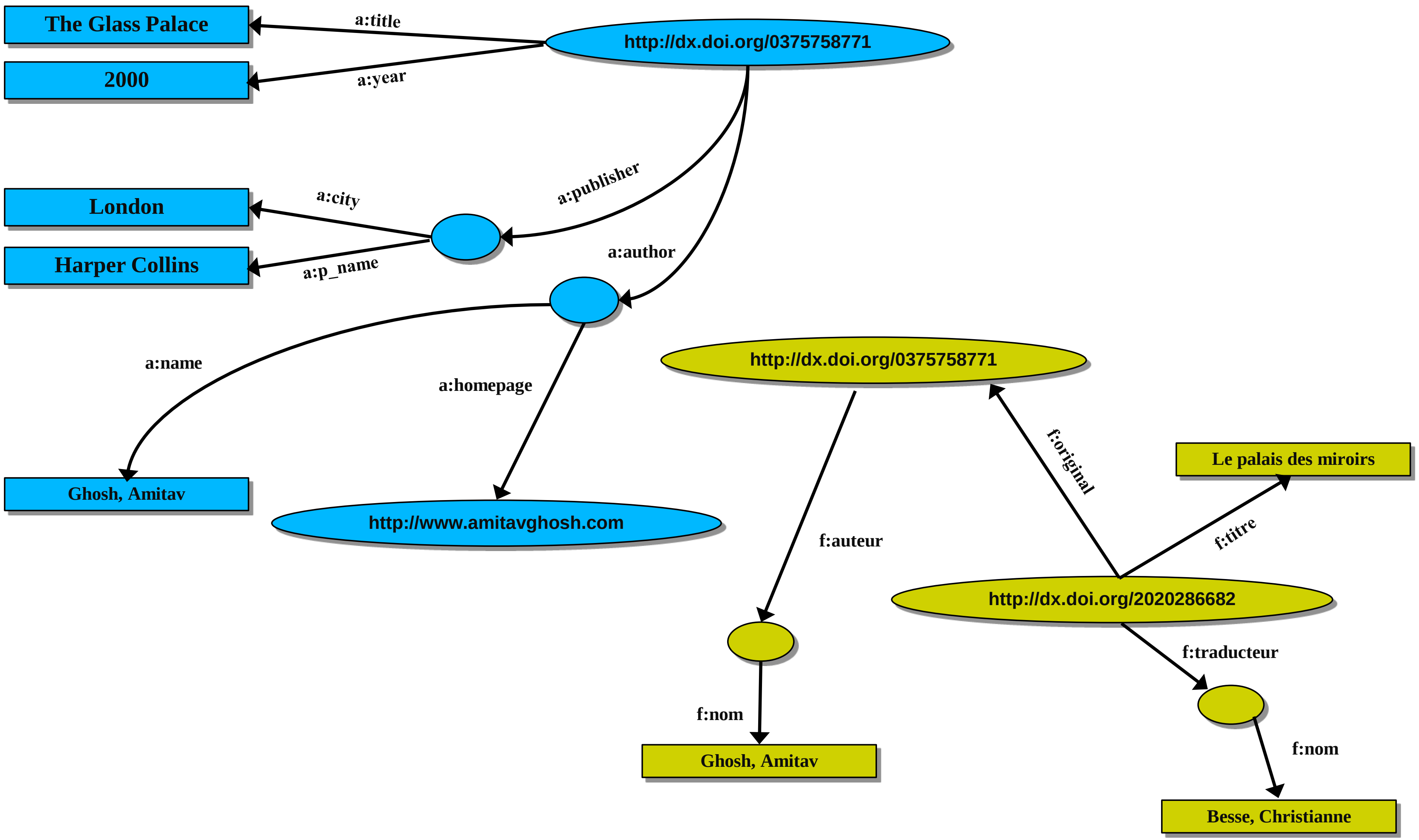
Knowledge Graphs and Ontologies

- Logic based representations can be easily represented as graphs
- This gives rise to graph databases also known as **knowledge graphs** or **NoSQL databases**
- Ontologies are logic based rules
- Ontologies replace SQL Schemas (in relational databases)
- Ontologies that follow the **Semantic Web** standard use specialised logic based language such as **OWL** to specify the Ontology/Schema
- Within such knowledge graphs each node is ***always*** a web URL (or URI)
- This ensures that the meaning of each node is guaranteed to be fixed

Knowledge Graphs and Ontologies

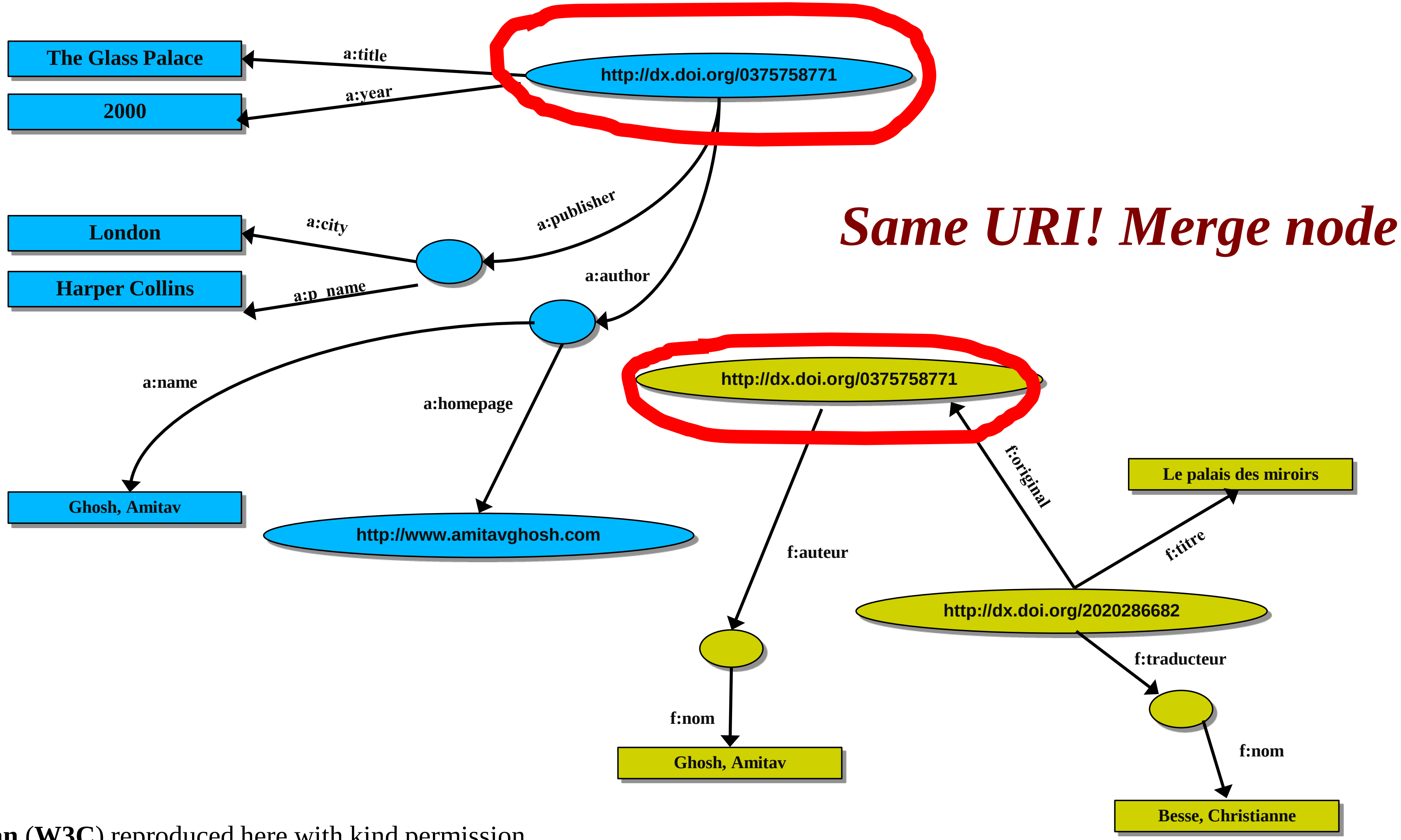
- Logic based representations can be easily represented as graphs
- This gives rise to graph databases also known as **knowledge graphs** or **NoSQL databases**
- Ontologies are logic based rules
- Ontologies replace SQL Schemas (in relational databases)
- Ontologies that follow the **Semantic Web** standard use specialised logic based language such as **OWL** to specify the Ontology/Schema
- Within such knowledge graphs each node is ***always*** a web URL (or URI)
- This ensures that the meaning of each node is guaranteed to be fixed
- Knowledge graphs are used to store massive knowledge bases
- For example, Amazon Alexa, Google assistant, Apple Siri all use knowledge graphs to answer your questions

Knowledge graph example



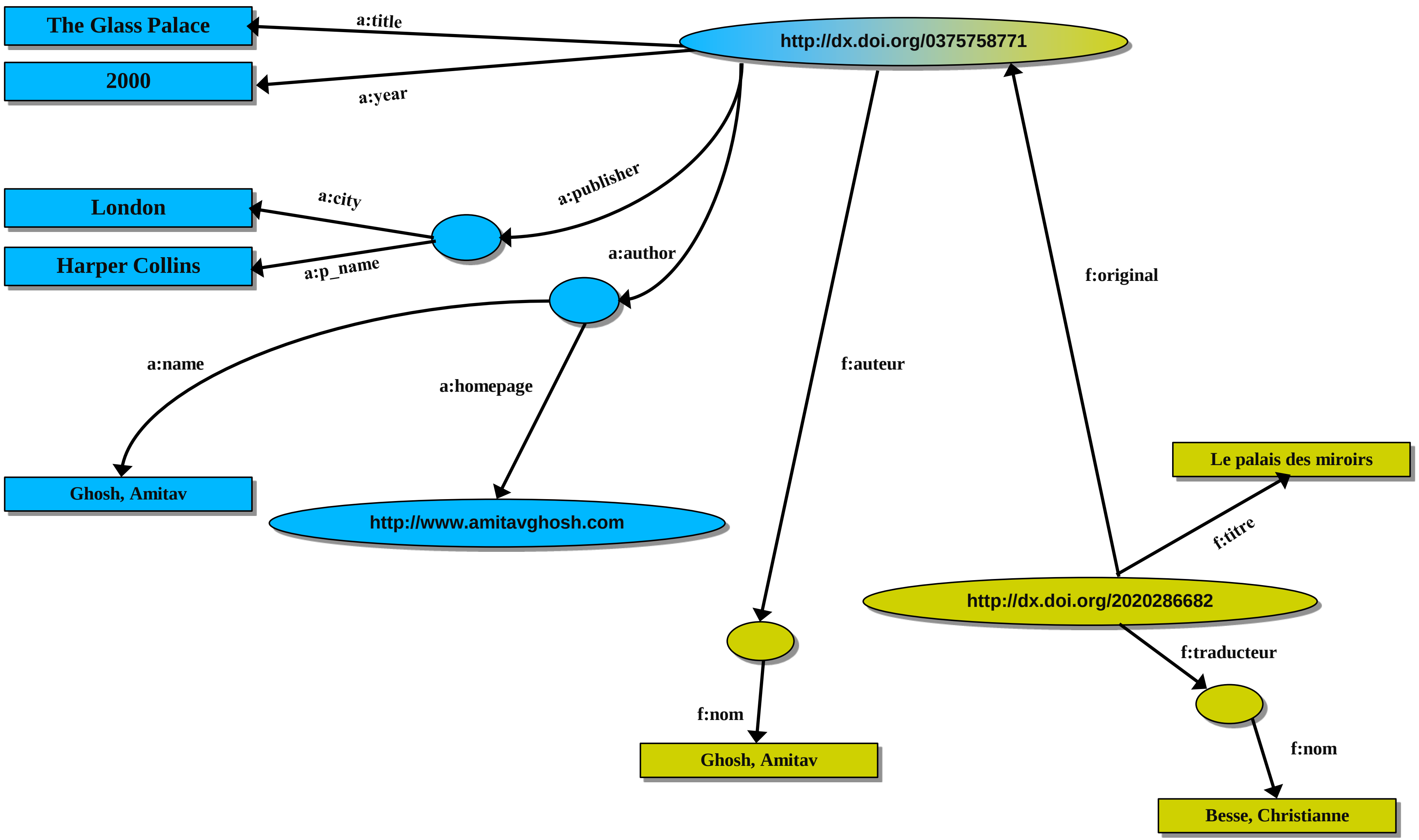
Source: **Ivan Herman** (**W3C**) reproduced here with kind permission.

Knowledge graph example



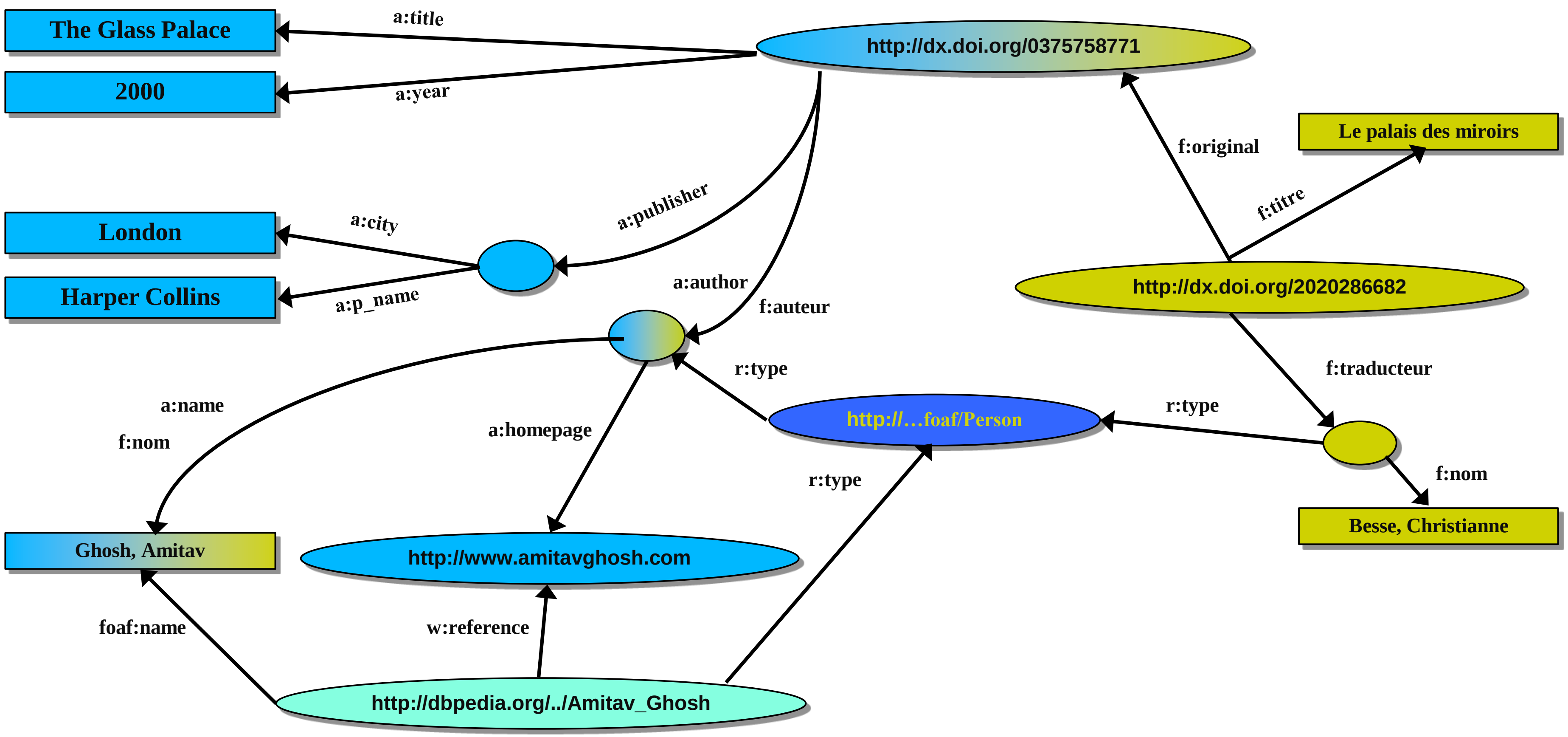
Source: **Ivan Herman** (**W3C**) reproduced here with kind permission.

Knowledge graph example



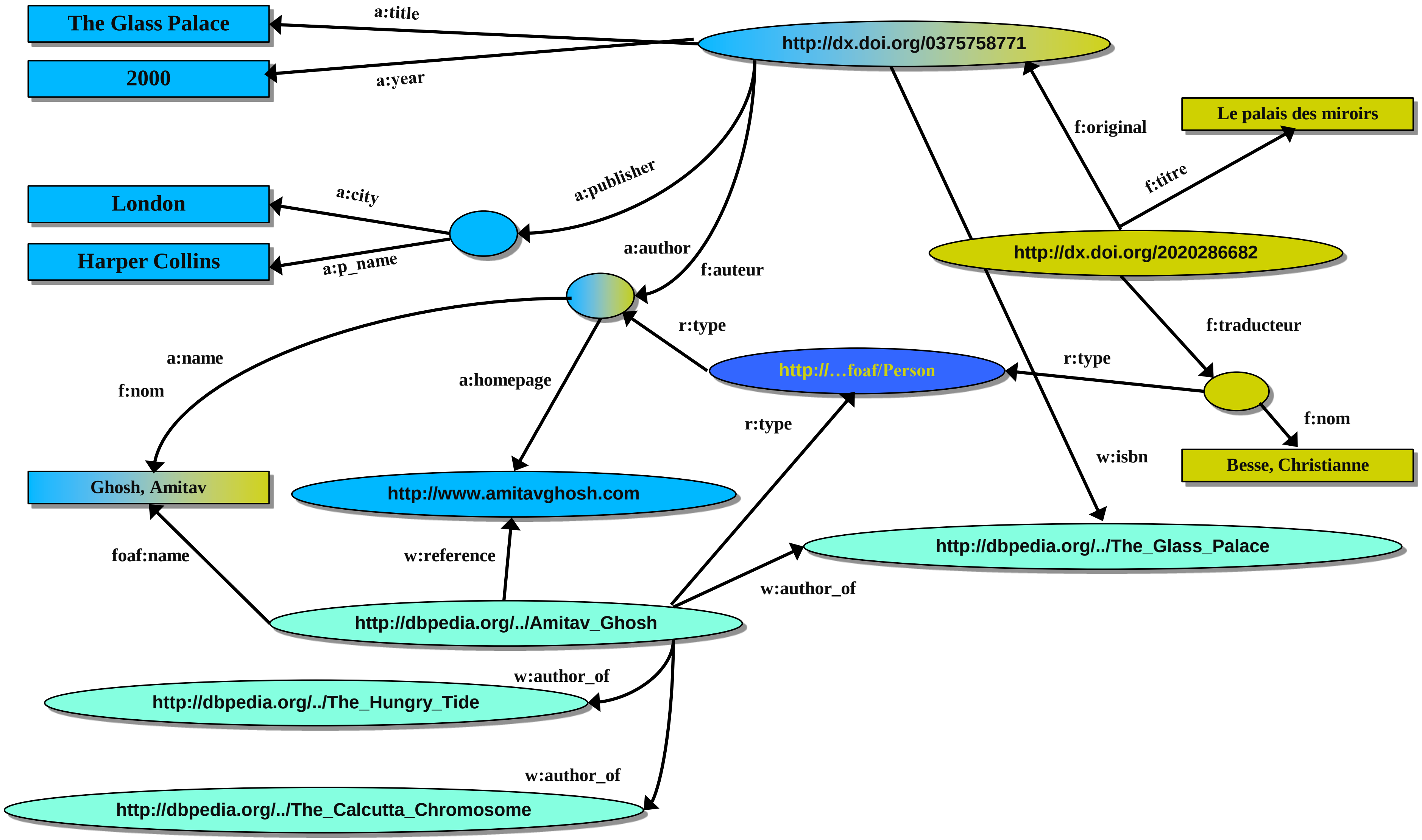
Source: **Ivan Herman** (**W3C**) reproduced here with kind permission.

Merge with Dbpedia data



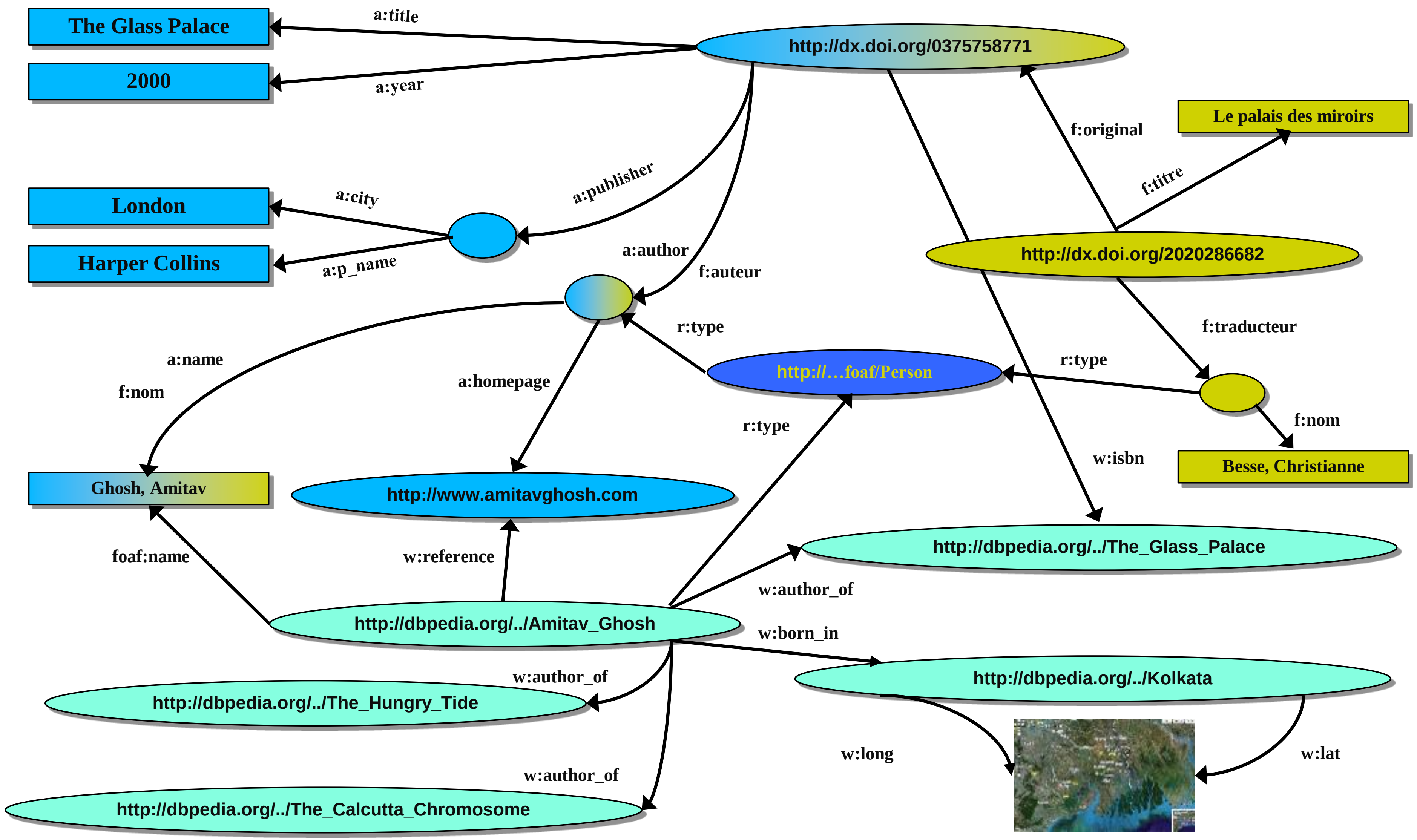
Source: **Ivan Herman** (**W3C**) reproduced here with kind permission.

Merge with Dbpedia data



Source: **Ivan Herman** (**W3C**) reproduced here with kind permission

Merge with Dbpedia data

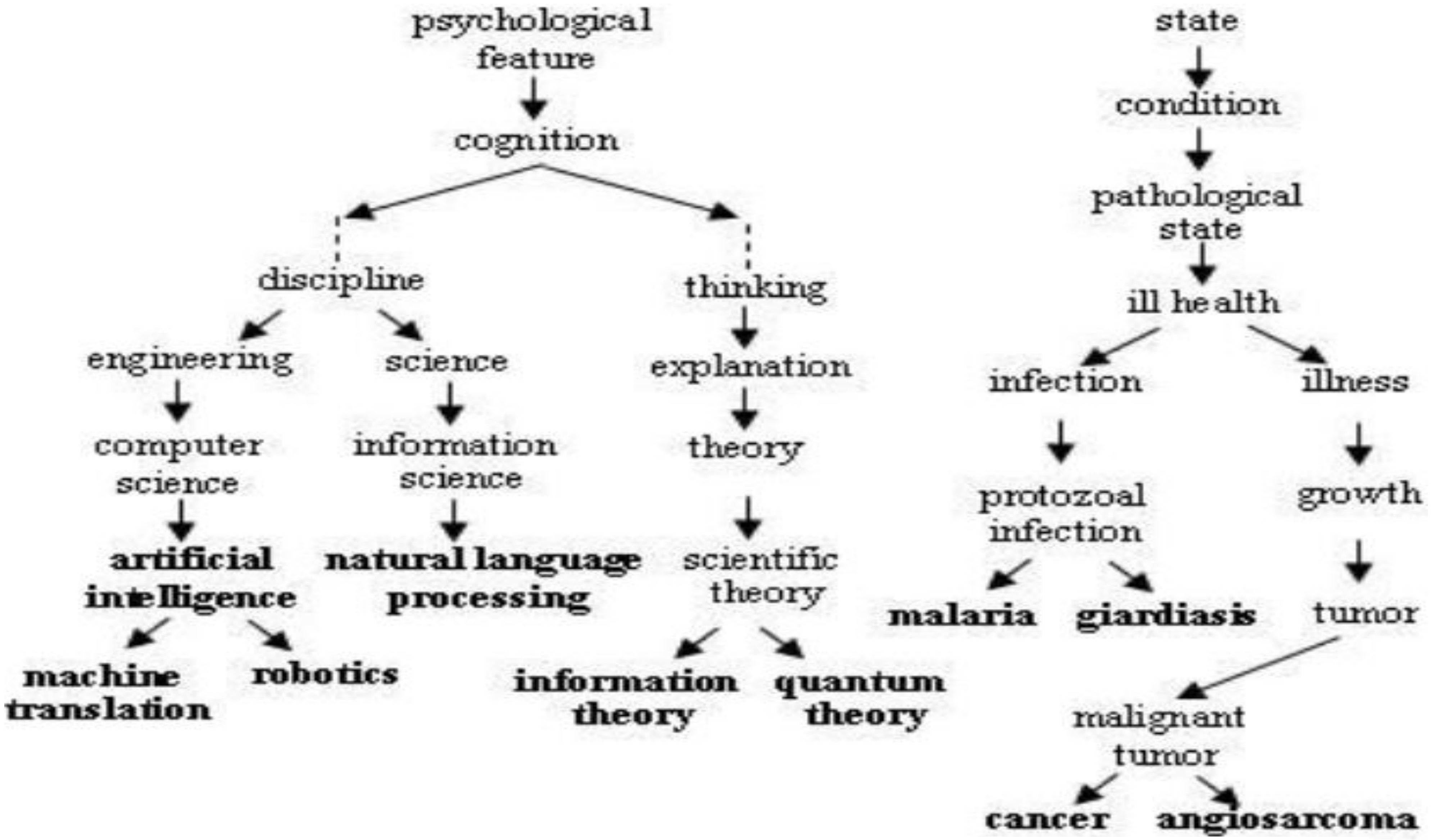


Source: **Ivan Herman** (**W3C**) reproduced here with kind permission.

Examples of knowledge bases

- Gene – Protein coding
- 3D Protein folding
- Point of information databases
- Hospital patient record systems
- WordNet, EuroWordNet, HowNet for storing human lexical knowledge

Wordnet example



Source: Goncalves et. sl. Mining Knowledge from Textual Databases: An Approach using Ontology-basedContext Vectors

Example Ontology : Schema.org

schema.org

Search

About Schemas Documentation

Welcome to Schema.org

Schema.org is a collaborative, community activity with a mission to create, maintain, and promote schemas for structured data on the Internet, on web pages, in email messages, and beyond.

Schema.org vocabulary can be used with many different encodings, including RDFa, Microdata and JSON-LD. These vocabularies cover entities, relationships between entities and actions, and can easily be extended through a well-documented extension model. Over 10 million sites use Schema.org to markup their web pages and email messages. Many applications from Google, Microsoft, Pinterest, Yandex and others already use these vocabularies to power rich, extensible experiences.

Schema.org is sponsored by Google, Microsoft, Yahoo and Yandex. The vocabularies are developed by an open community process, using the public-schemaorg@w3.org mailing list and through GitHub.

A shared vocabulary makes it easier for webmasters and developers to decide on a schema and get the maximum benefit for their efforts. It is in this spirit that the sponsors, together with the larger community have come together, to provide a shared collection of schemas.

We invite you to get started!

View our blog at blog.schema.org or see release history.

Example Ontology : Schema.org

- Properties of the **Person** type shown below

Property	Expected Type	Description
Properties from Person		
additionalName	Text	An additional name for a Person, can be used for a middle name.
address	PostalAddress or Text	Physical address of the item.
affiliation	Organization	An organization that this person is affiliated with. For example, a school/university
alumniOf	EducationalOrganization or Organization	An organization that the person is an alumni of. Inverse property: alumni.
award	Text	An award won by or for this item. Supersedes awards.
birthDate	Date	Date of birth.
birthPlace	Place	The place where the person was born.
brand	Organization or Brand	The brand(s) associated with a product or service, or the brand(s) maintained by a
children	Person	A child of the person.
colleague	Person	A colleague of the person. Supersedes colleagues.
contactPoint	ContactPoint	A contact point for a person or organization. Supersedes contactPoints.

Google Knowledge Graph Video

Google Knowledge Graph Video

Reasoning using logic

- KR languages such as OWL (a WWW standard) provide **automated reasoning** mechanisms
- The automated reasoning mechanism ensures that data stored as knowledge graphs are **consistent**

Reasoning using logic

- **Example**: Every Person has a Father and a Mother who are also Persons

$$\forall x\ (Person(x) \rightarrow \exists!\, y\ (hasParent(x,y)\ \ \&\ Male(y)\ \&\ Person(y)))$$
$$\forall x\ (Person(x) \rightarrow \exists!\, y\ (hasParent(x,y)\ \ \&\ Female(y)\ \&\ Person(y)))$$
$$\forall x\ \forall y\ \ (Male(x)\ \&\ hasChild(x,y) \rightarrow Father(x)\ )$$
$$\forall x\ \forall y\ \ (Female(x)\ \&\ hasChild(x,y) \rightarrow Mother(x)\ )$$
$$\forall x \forall y\ (hasParent(x,y) \rightarrow hasChild(y,x)\ )$$
$$\forall x \forall y\ (hasChild(x,y) \rightarrow hasParent(y,x)\ )$$

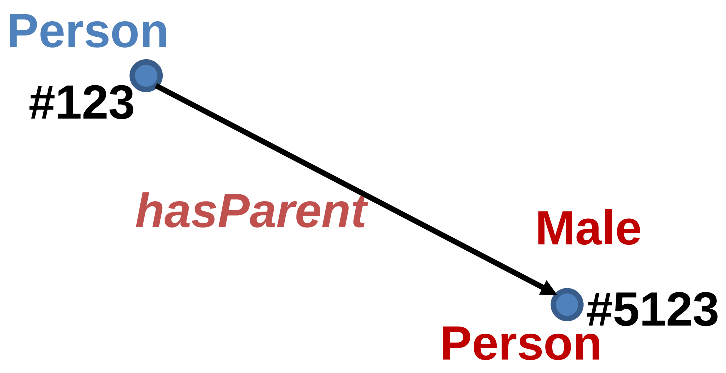
Reasoning using logic

- Example: Every Person has a Father and a Mother who are also Persons

$$\forall x\ (\boldsymbol{Person}(x) \rightarrow \exists!\, y\ (\boldsymbol{hasParent}(x, y)\ \&\ \boldsymbol{Male}(y)\ \&\ \boldsymbol{Person}(y)))$$

$\boldsymbol{Person}(\#123)$

$\boldsymbol{Person}(\#5123)$
$\boldsymbol{hasParent}(\#123, \#5123)$
$\boldsymbol{Male}(\#5123)$



Reasoning using logic

- Example: Every Person has a Father and a Mother who are also Persons

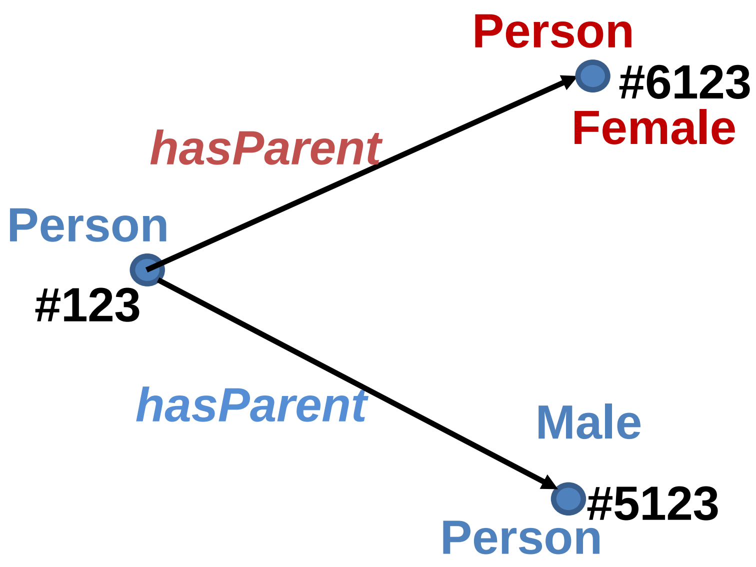
$$\forall x\ (Person(x) \rightarrow \exists!\, y\ (hasParent(x,y)\ \&\ Male(y)\ \&\ Person(y)))$$

$$\forall x\ (Person(x) \rightarrow \exists!\, y\ (hasParent(x,y)\ \&\ Female(y)\ \&\ Person(y)))$$

$Person(\#123)$

$Person(\#5123)$
$hasParent(\#123, \#5123)$
$Male(\#5123)$

$Person(\#6123)$
$hasParent(\#123, \#6123)$
$Female(\#6123)$



Reasoning using logic

- Example: Every Person has a Father and a Mother who are also Persons

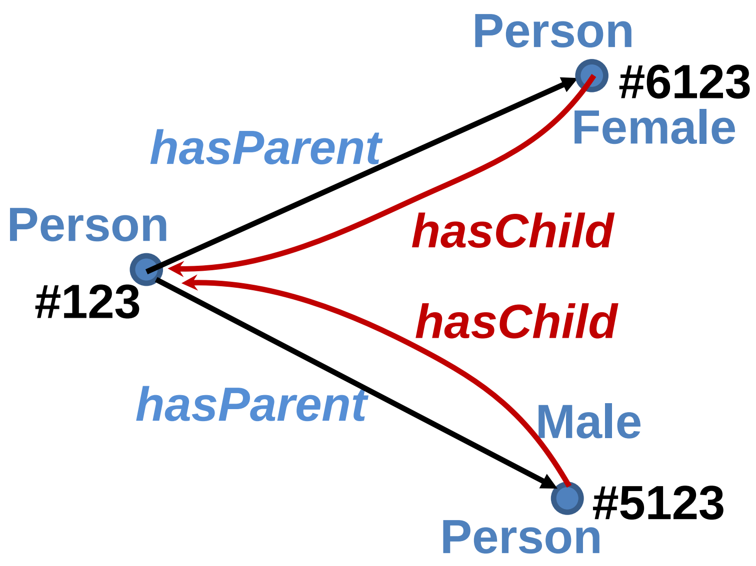
$$\forall x\ (Person(x) \rightarrow \exists!\, y\ (hasParent(x,y)\ \&\ Male(y)\ \&\ Person(y)))$$
$$\forall x\ (Person(x) \rightarrow \exists!\, y\ (hasParent(x,y)\ \&\ Female(y)\ \&\ Person(y)))$$
$$\forall x \forall y\ (hasParent(x,y) \rightarrow hasChild(y,x)\ )$$
$$\forall x \forall y\ (hasChild(x,y) \rightarrow hasParent(y,x)\ )$$

$Person(\#123)$

$hasParent(\#123, \#5123)$
$Male(\#5123)$
$Person(\#5123)$

$hasParent(\#123, \#6123)$
$Female(\#6123)$
$Person(\#6123)$

$hasChild(\#5123, \#123)$
$hasChild(\#6123, \#123)$



Reasoning using logic

- Example: Every Person has a Father and a Mother who are also Persons

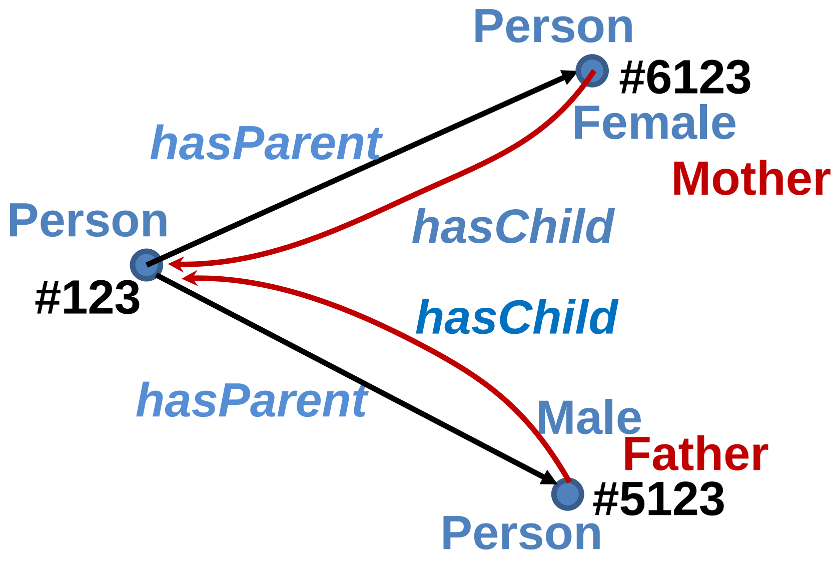
$$\forall x\,(Person(x) \rightarrow \exists!\,y\,(hasParent(x,y)\ \&\ Male(y)\ \&\ Person(y)))$$
$$\forall x\,(Person(x) \rightarrow \exists!\,y\,(hasParent(x,y)\ \&\ Female(y)\ \&\ Person(y)))$$
$$\forall x \forall y\,(hasParent(x,y) \rightarrow hasChild(y,x)\,)$$
$$\forall x \forall y\,(hasChild(x,y) \rightarrow hasParent(y,x)\,)$$
$$\forall x\,\forall y\ \ (Male(x)\ \&\ hasChild(x,y) \rightarrow Father(x)\,)$$
$$\forall x\,\forall y\ \ (Female(x)\ \&\ hasChild(x,y) \rightarrow Mother(x)\,)$$

$Person(\#123)$

$hasParent(\#123, \#5123)$
$Male(\#5123)$
$Person(\#5123)$

$hasParent(\#123, \#6123)$
$Female(\#6123)$
$Person(\#6123)$

$hasChild(\#5123, \#123)$
$hasChild(\#6123, \#123)$
$Father(\#5123)$
$Mother(\#6123)$



Machine Learning

Machine Learning

Supervised vs Unsupervised learning (Clustering)

- ***Supervised Learning***

In ***supervised learning***, we are given data along with the prediction we want to learn. For example, we want to predict **height** from **weight** and **age**:

weight	age	height
65	50	170
30	40	140

Data is typically collected from a real survey or experiment.

weight	age	height
65	50	170
30	40	140
65	50	190

Machine Learning

Supervised vs Unsupervised learning (Clustering)

- ***Unsupervised Learning***

In ***unsupervised learning***, we are given data without any predicted variable. The task is to group or cluster data points that are very similar to each other.

weight	age
85	50
30	40
32	42

The output of clustering would be set of inferred classes or labels. In this example, two people with similar weight and age have been grouped into class 1.

weight	age	class
85	50	2
30	40	1
32	42	1

Machine Learning

Regression vs Classification

- ***Regression***

In ***regression***, we are predicting a ***continuous variable*** given other variables.

weight	age	height
65	50	170
30	40	140

- ***Classification***

In ***classification***, we are predicting a ***discrete variable*** given other variables.

weight	age	height
65	50	2
30	40	1
65	50	0

e.g. 2 = tall
1 = medium
0 = short

Learning with a single neuron

- ***Example:*** Suppose we want to be able to predict ***height*** given ***weight*** and ***age***
- One row of our input data might look like:

weight	age	height
65	50	170

- With some trial and error, we can work out a weighting scheme to weight each ***feature*** to predict the *variable of interest* (i.e. height)
- So, we have a formula like: height **=** ***w1*** ***** weight **+** ***w2*** ***** age **+** ***b***
- There are many solutions, can you choose suitable values of ***w1, w2, b*** so that we get the right answer of 170?

Learning with a single neuron

- ***Cheating**: w1, w2 = 0, b = 170* so we get
-
- height **=** ***0*** ***** 65 **+** ***0*** ***** 50 **+** ***170*** = 170

weight	age	height
65	50	170

- So what is wrong with it?

Learning with a single neuron

- ***Lets add more data:***

weight	age	height
65	50	170
30	40	140

- We are in trouble now.
- Our formula will give: height **=** ***0*** ***** 30 **+** ***0*** ***** 40 **+** ***170*** = 170
- Lets have another go:

Learning with a single neuron

- ***Lets add more data:***

weight	age	height
65	50	170
70	40	140

- We are in trouble now.
- Our formula will give: height **=** ***0*** ***** 30 **+** ***0*** ***** 40 **+** ***170*** = 170
- Lets have another go: height **= -** ***0.3*** ***** weight **+** ***3*** ***** age **+** ***40***
- This gives:
- height **= -** ***0.3*** ***** 65 **+** ***3*** ***** 50 **+** ***40*** = 170.5
- height **= -** ***0.3*** ***** 70 **+** ***3*** ***** 40 **+** ***40*** = 139

Learning with a single neuron

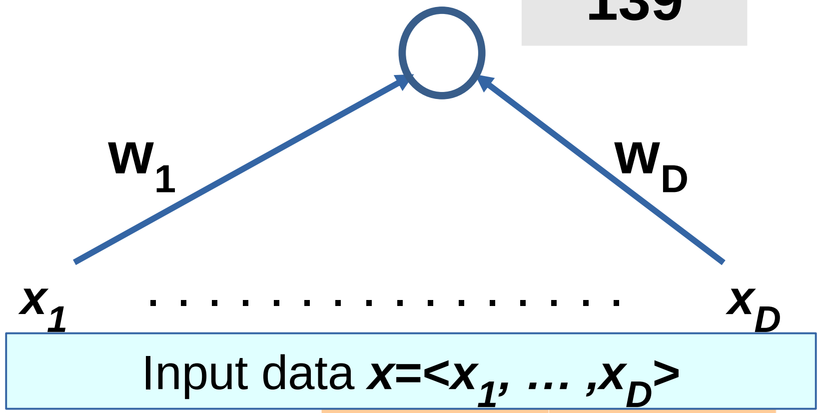
- The purpose of a ML algorithm is to learn the parameters $\mathbf{w_1, \ldots x_D, b}$ ***automatically*** from data
- For this ***linear regression*** problem we can achieve this with a single neuron

predicted

y'
170.5
139

actual

y
170
140



weight	age
65	50
70	40

Learning with a single neuron

- How do we judge which solution is better?

Solution 1

y'
170.5
139

S1 Error

E
-0.5
1

Solution 2

y'
171
139

S2 Error

E
-1
1

actual

y
170
140

Learning with a single neuron

- How do we judge which solution is better?

Solution 1	S1 Error	Solution 2	S2 Error	actual
y’	E	y’	E	y
170.5	-0.5	171	-1	170
139	1	139	1	140

- Will adding up the errors work?

Learning with a single neuron

- How do we judge which solution is better?

Solution 1	S1 Error	Solution 2	S2 Error	actual
y'	E	y'	E	y
170.5	-0.5	171	-1	170
139	1	139	1	140

- Will adding up the errors work?
- Unfortunately not. Since the positive and negative errors will cancel out

Learning with a single neuron

- How do we judge which solution is better?

Solution 1 y'	S1 Error E	Solution 2 y'	S2 Error E	actual y
170.5	-0.5	171	-1	170
139	1	139	1	140

- Will adding up the errors work?
- Unfortunately not. Since the positive and negative errors will cancel out
- So we want something that is ***always positive***

Learning with a single neuron

- **Loss function** : is a measure of error that is always positive (including zero)

Solution 1	S1 Error	Sq. Loss	Solution 2	S2 Error	Sq. Loss	actual
y'	E	$(y-y')^2$	y'	E	$(y-y')^2$	y
170.5	-0.5	0.25	171	-1	1	170
139	1	1	139	1	1	140

- ***Mean Squared Loss*** is a popular choice
- There are many choices for loss functions (e.g. *Absolute error, combinations*)
- You can design your own.
- ML engineers occasionally need to design their own. But the community now has a vast array at disposal

Learning with a single neuron

- The purpose of a ML algorithm is to learn the parameters
- **w_1, …x_D, b** ***automatically*** from data
- For this ***linear regression*** problem we can achieve this with a single neuron

predicted

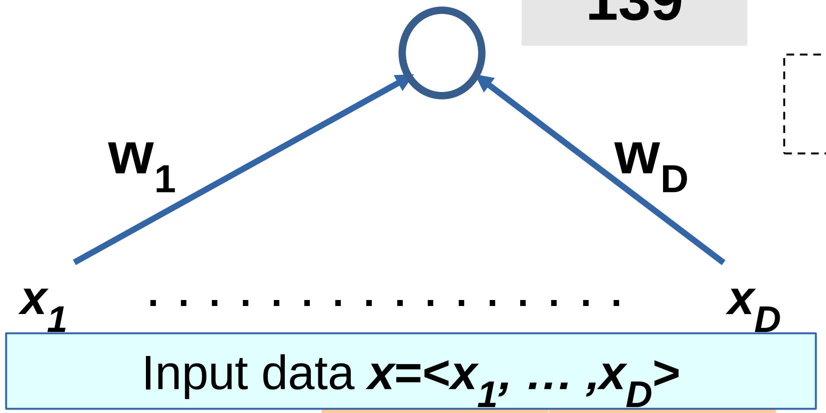
y’
170.5
139

$(y-y')^2$
0.25
1

Loss

y
170
140

actual

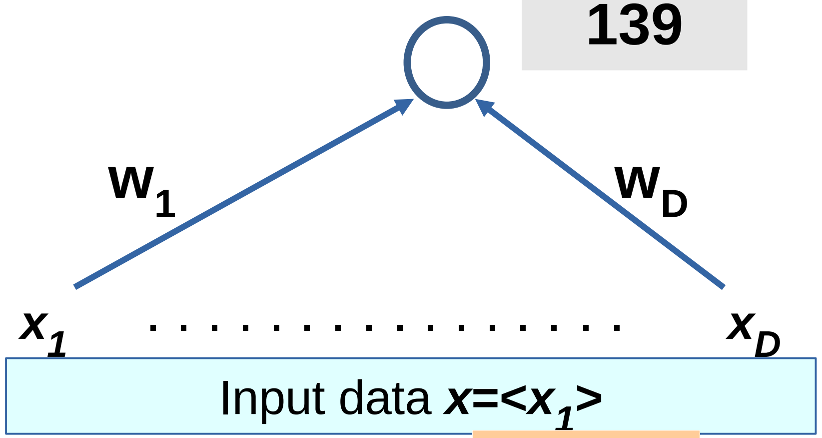


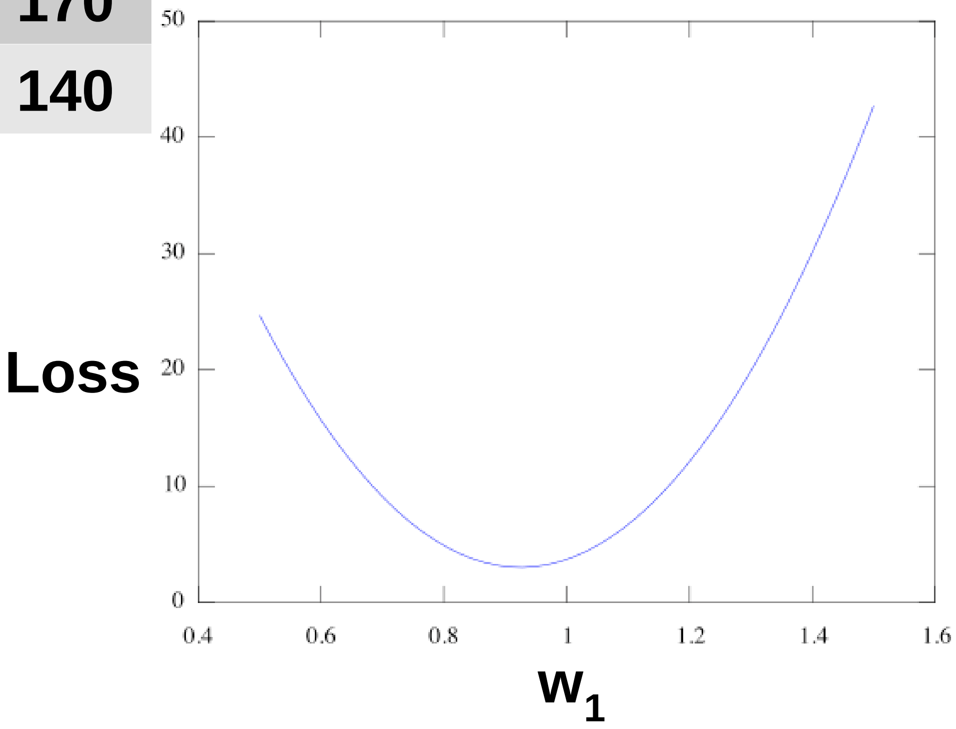
weight	age
65	50
70	40

Gradient based learning

- To be able to visualise our loss function we will assume that we only have **age** and furthermore we assume **b=0**

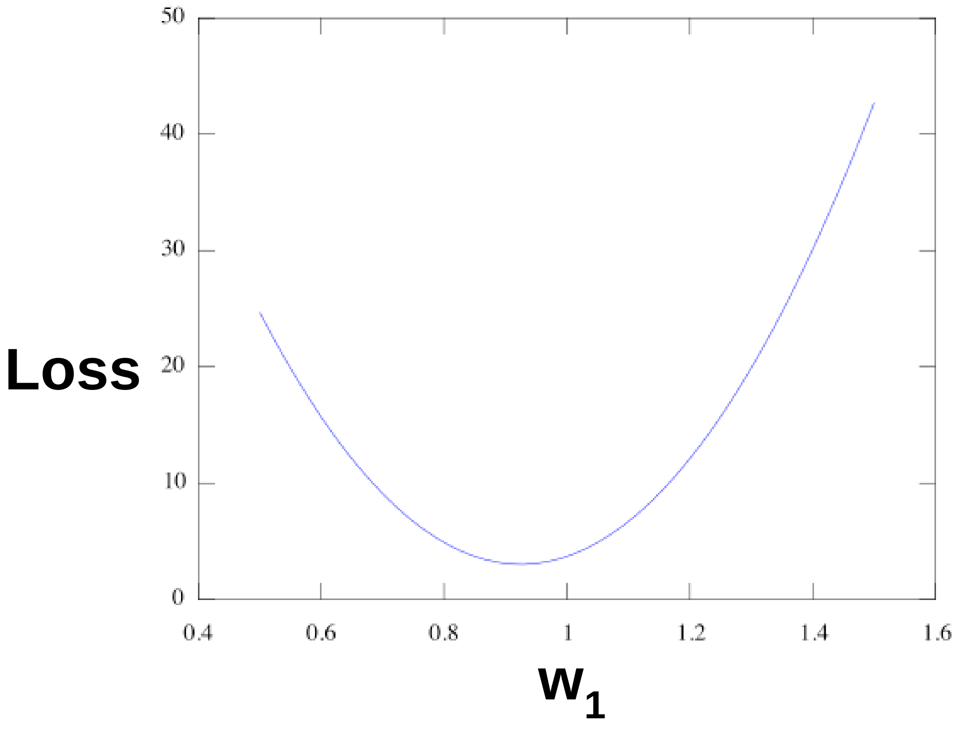
age	predicted y'	Loss $(y-y')^2$	actual y
50	170.5	0.25	170
40	139	1	140





Gradient based learning

- To be able to visualise our loss function we will assume that we only have **age** and furthermore we assume **b=0**
- This is essentially our familiar quadratic polynomial
- Guaranteed to have a single global minima



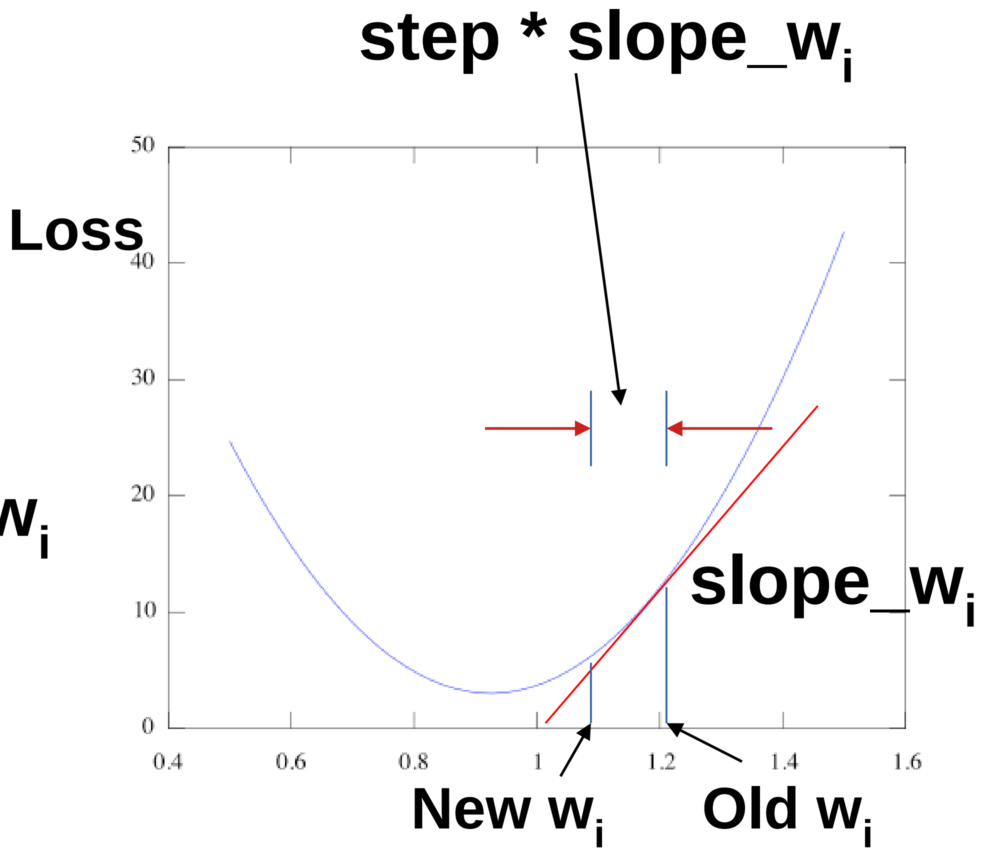
Gradient based learning

Simple gradient algorithm:

- Choose small random values for $\mathbf{w_1,\ldots,w_D}$ **,** $\mathbf{w_0}$ **(= b previously)**
- Choose a small step size ***step*** (e.g. step = 0.0001)
- **Repeat:**
  - Compute loss for current values of

    $\mathbf{w_1,\ldots,w_D}$ **,** $\mathbf{w_0}$
  - Compute gradient/slope with respect to

    each $\mathbf{w_1,\ldots,w_D}$ **,** $\mathbf{w_0}$
  - Update for each: $\mathbf{w_i := w_i - step * slope\_w_i}$
- **Until loss is small**



Calculating gradient

We can employ differential calculus to compute the gradient of **Loss(W)** in closed form

$$L(y, y') = (y - y')^2 = (y - (w_0 + w_1 x_1 + \dots + w_D x_D))^2$$

$$\frac{\delta L(y, y')}{\delta w_i} = \frac{\delta}{\delta w_i} (y - (w_0 + w_1 x_1 + \dots + w_D x_D))^2$$

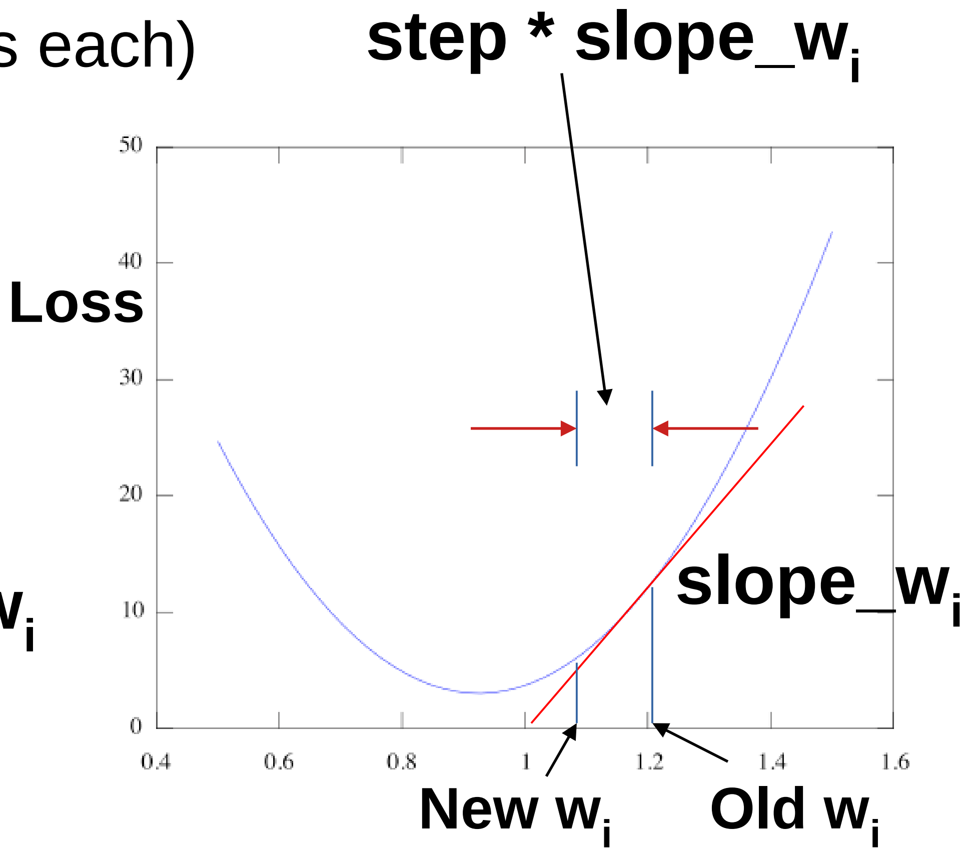
$$= -2(y - (w_0 + w_1 x_1 + \dots + w_D x_D)) \frac{\delta}{\delta w_i} (w_0 + w_1 x_1 + \dots + w_D x_D)$$

$$= -2(y - (w_0 + w_1 x_1 + \dots + w_D x_D))\, x_i$$

Gradient based learning (for linear regression)

Simple gradient algorithm:

- Choose small random values for $\mathbf{w_1,\ldots,w_D}$, $\mathbf{w_0}$ **(= b previously)**
- Choose a small step size ***step*** (e.g. step = 0.0001)
- Divide data into small batches (e.g. 200 examples each)
- **Repeat for each batch:**
  - Compute batch loss for current values of $\mathbf{w_1,\ldots,w_D}$, $\mathbf{w_0}$
  - Compute for each batch gradient/slope with respect to each $\mathbf{w_1,\ldots,w_D}$, $\mathbf{w_0}$
  - Update for each: $\mathbf{w_i := w_i - step * slope\_w_i}$
- **Until loss is small**



Representing Loss

$L(y, y') = (y - y')^2 = (y - (w_0 + w_1 x_1 + ... + w_D x_D))^2 = (y - w_0 - \mathbf{W}.X)^2$

Here we let $\mathbf{W} = \langle w_1, .., w_D \rangle$ and $\mathbf{X} = \langle x_1, .., x_D \rangle$ ***then***:

$\mathbf{W}.X = \langle w_1, .., w_D \rangle . \langle x_1, .., x_D \rangle = (w_1 x_1 + ... + w_D x_D)$

Representing Loss

$L(y, y') = (y - y')^2 = (y - (w_0 + w_1 x_1 + ... + w_D x_D))^2 = (y - w_0 - W.X)^2$

Here we let $W = \langle w_1, .., w_D \rangle$ and $X = \langle x_1, .., x_D \rangle$ ***then***:

$W.X = \langle w_1, .., w_D \rangle . \langle x_1, .., x_D \rangle = (w_1 x_1 + ... + w_D x_D)$

Alternatively, we let $W = \langle w_1, w_1, .., w_D \rangle$ and $X = \langle 1, x_1, .., x_D \rangle$ ***then*** :

$W.X = \langle w_1, w_1, .., w_D \rangle . \langle 1, x_1, .., x_D \rangle = (w_0 + w_1 x_1 + ... + w_D x_D)$

$L(y, y') = (y - y')^2 = (y - (w_0 + w_1 x_1 + ... + w_D x_D))^2 = (y - W.X)^2$

Question

What is:

$\mathbf{Y}.\mathbf{Y} = \langle y_1, .., y_N \rangle \,.\, \langle y_1, .., y_N \rangle = ?$

Answer

$$Y.Y = \langle y_1, .., y_N \rangle \cdot \langle y_1, .., y_N \rangle = y_1^2 + ... + y_N^2$$

Length of a vector

$$\|Y\| = \sqrt{(y_1^2 + \dots. + y_N^2)} = \sqrt{Y.Y} = \sqrt{<y_1, .., y_N> . <y_1, .., y_N>}$$

Length of a vector

$$\|Y\| = \sqrt{(y_1^2 + \dots. + y_N^2)} = \sqrt{Y.Y} = \sqrt{\langle y_1, .., y_N\rangle . \langle y_1, .., y_N\rangle}$$

Thus:

$$\|Y\|^2 = (y_1^2 + \dots. + y_N^2) = Y.Y = \langle y_1, .., y_N\rangle . \langle y_1, .., y_N\rangle$$

Matrix Representation of Loss

$L(y_i, y_i') = (y_i - y_i')^2 = (y_i - (w_0 + w_1 x_{i1} + ... + w_D x_{iD}))^2 = (y_i - \mathbf{W}.X_i)^2$

Matrix Representation of Loss

$$L(y_i, y_i') = (y_i - y_i')^2 = (y_i - (w_0 + w_1\, x_{i1} + \ldots. + w_D x_{iD}))^2 = (y_i - W.X_i)^2$$

$$\text{Total Loss}(W) = \sum_{I} \text{loss}(y_i, y_i') = \sum_{i} \text{loss}(y_i, W.X_i) = (y_1 - y_1')^2 + \ldots + (y_N - y_N')^2$$

$$= \langle (y_1 - y_1'), \ldots, (y_N - y_N') \rangle \,.\, \langle (y_1 - y_1'), \ldots, (y_N - y_N') \rangle$$

Matrix Representation of Loss

$L(y_i, y_i') = (y_i - y_i')^2 = (y_i - (w_0 + w_1 x_{i1} + \dots + w_D x_{iD}))^2 = (y_i - W.X_i)^2$

$$\text{Total Loss}(W) = \sum_i \text{loss}(y_i, W.X_i) = (y_1 - y_1')^2 + \dots + (y_N - y_N')^2$$

$$= \langle (y_1 - y_1'), \dots, (y_N - y_N') \rangle . \langle (y_1 - y_1'), \dots, (y_N - y_N') \rangle$$

Since $Y = \langle y_1, .., y_N \rangle$ and $Y' = \langle y_1', .., y_N' \rangle$

$Y - Y' = \langle (y_1 - y_1'), \dots, (y_N - y_N') \rangle$

Matrix Representation of Loss (contd..)

$L(y_i, y_i') = (y_i - y_i')^2 = (y_i - (w_0 + w_1 x_{i1} + \ldots + w_D x_{iD}))^2 = (y_i - W.X_i)^2$

$\text{Total Loss}(W) = \sum_i \text{loss}(y_i, W.X_i) = (y_1 - y_1')^2 + \ldots + (y_N - y_N')^2$

$= \langle (y_1 - y_1'), \ldots, (y_N - y_N') \rangle . \langle (y_1 - y_1'), \ldots, (y_N - y_N') \rangle$

Since $Y = \langle y_1, .., y_N \rangle$ and $Y' = \langle y_1', .., y_N' \rangle$

$Y - Y' = \langle (y_1 - y_1'), \ldots, (y_N - y_N') \rangle$

$\text{Total Loss}(W) = (y_1 - y_1')^2 + \ldots + (y_N - y_N')^2 = (Y - Y') . (Y - Y') = \|Y - Y'\|^2$

$= \|Y - WX\|^2$ **(gives the loss in matrix form)**

Matrix Closed form solution for Linear Regression

Total Loss(W) = $(y_1 - y_1')^2 + \dots + (y_N - y_N')^2 = (Y - Y') \,.\, (Y - Y') = ||Y - Y'||^2$

$= ||Y - WX||^2$

Let Δ Total Loss(W) = derivative with respect to W

Matrix Closed form solution for Linear Regression

Total Loss(**W**) = $(y_1 - y_1')^2 + \ldots + (y_N - y_N')^2 = (Y - Y') \cdot (Y - Y') = \|Y - Y'\|^2$

$= \|Y - WX\|^2$

Let Δ Total Loss(**W**) = derivative with respect to W

But **W**=$\langle w_1, .., w_D \rangle$ so what is derivative with respect to a vector?

Matrix Closed form solution for Linear Regression

Total Loss(W) = $(y_1 - y_1')^2 + \dots + (y_N - y_N')^2 = (Y - Y') \cdot (Y - Y') = ||Y - Y'||^2$

$= ||Y - WX||^2$

Let Δ Total Loss(W) = derivative with respect to W

Question: But $W = \langle w_0, .., w_D \rangle$ so what is derivative with respect to a vector?

Answer: Δ Total Loss(W) = $\left\langle \frac{\delta \text{ Total Loss}(W)}{\delta w_0}, .., \frac{\delta \text{ Total Loss}(W)}{\delta w_D} \right\rangle$

$= -2 X^T (Y - WX)$

Closed form solution in matrix form

We have, Δ **Total Loss(W) = -2 $(X^T Y - X^T X W)$**

Setting $\Delta L(W) = 0$ (since, derivative is 0 at extremum points)

$$-2 (X^T Y - X^T X W) = 0$$

$$X^T Y = X^T X W$$

$$(X^T X)^{-1} X^T Y = (X^T X)^{-1} X^T X \theta$$

$$(X^T X)^{-1} X^T Y = W$$

Hence: $W = (X^T X)^{-1} X^T Y$

gives *closed form* solution

But requires $X^T X$ to be ***invertible***

Machine Learning : Learning to classify

Classification problems : Is it a cat ?

YES

NO

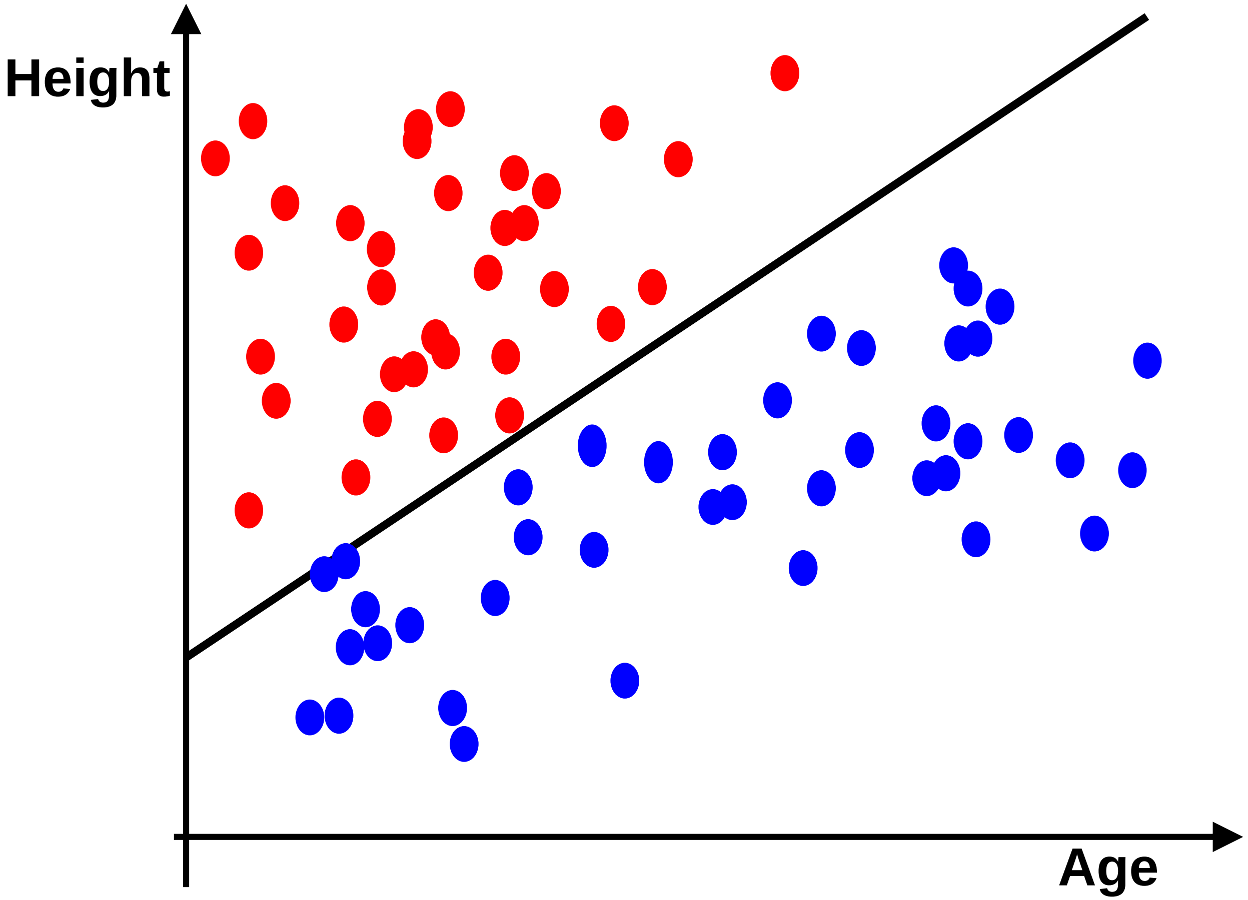
NO

YES

NO

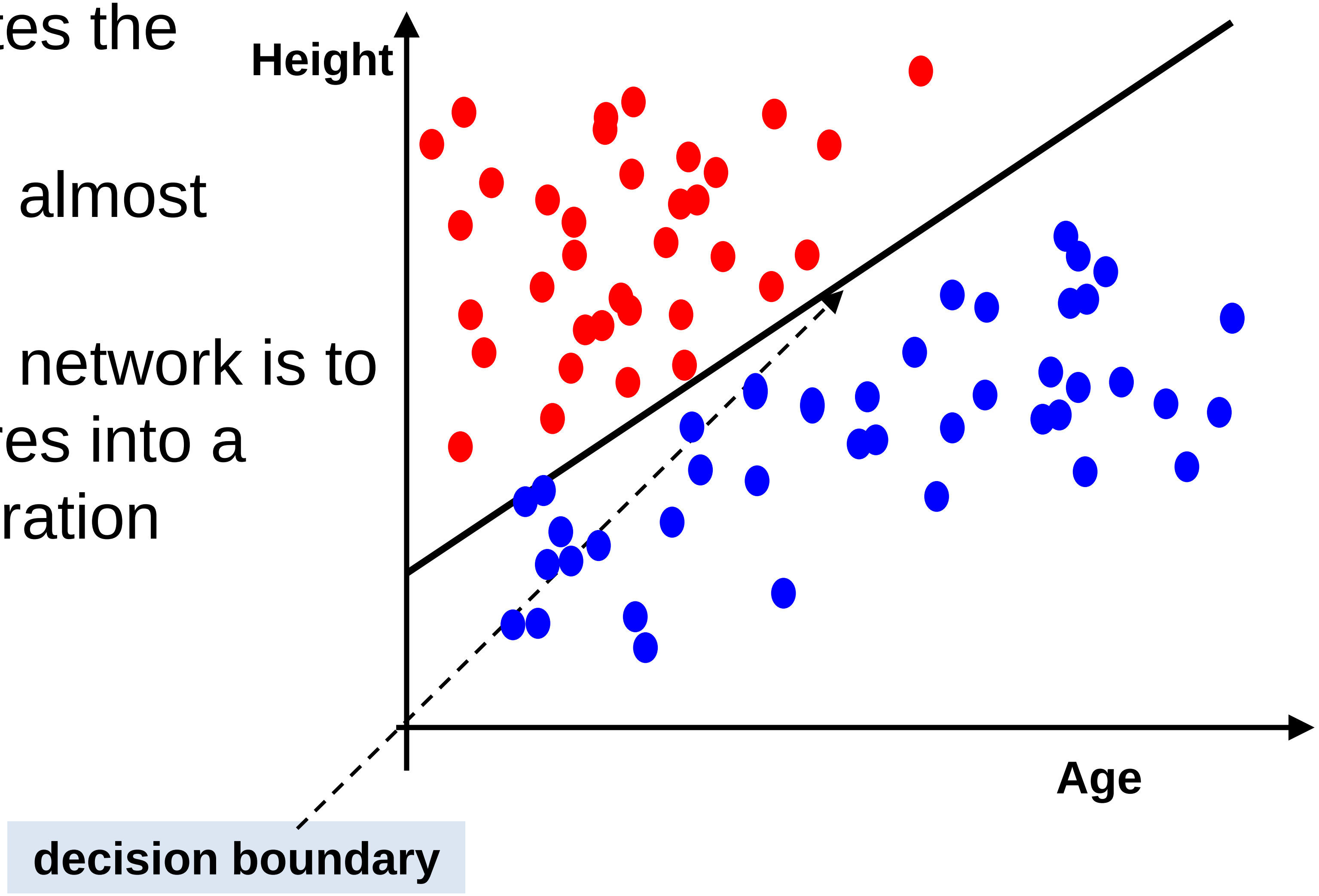
YES

Tall or Short ?



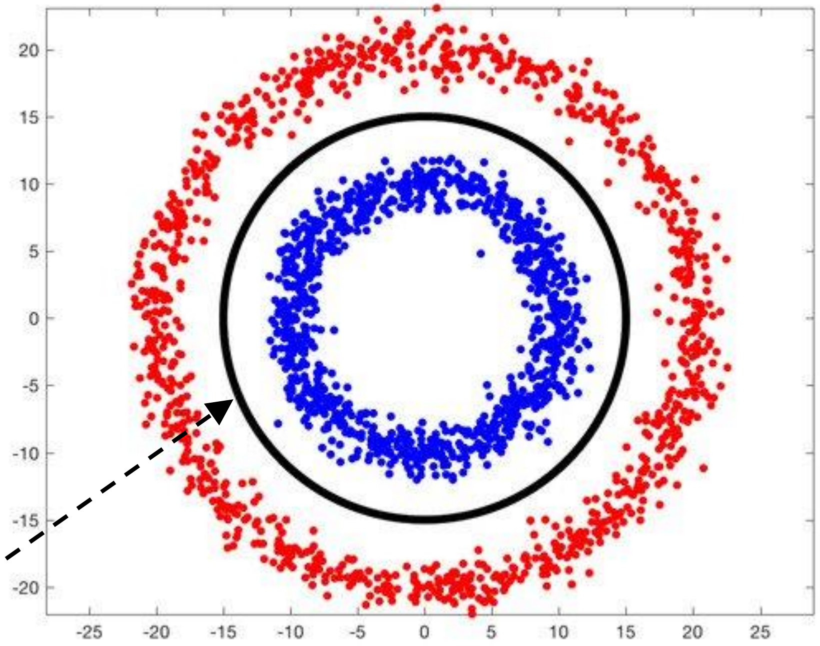
Linear separability for classification problems

- The **decision boundary** is the line (hyperplane) that separates the different classes
- The decision boundary is almost always linear
- The aim of a deep neural network is to transform the input features into a space so that linear separation becomes possible



Linear separability for classification problems

- But clearly not all real life data will be linearly separable
- **Example**: Data within the red and blue circles denote the two classes

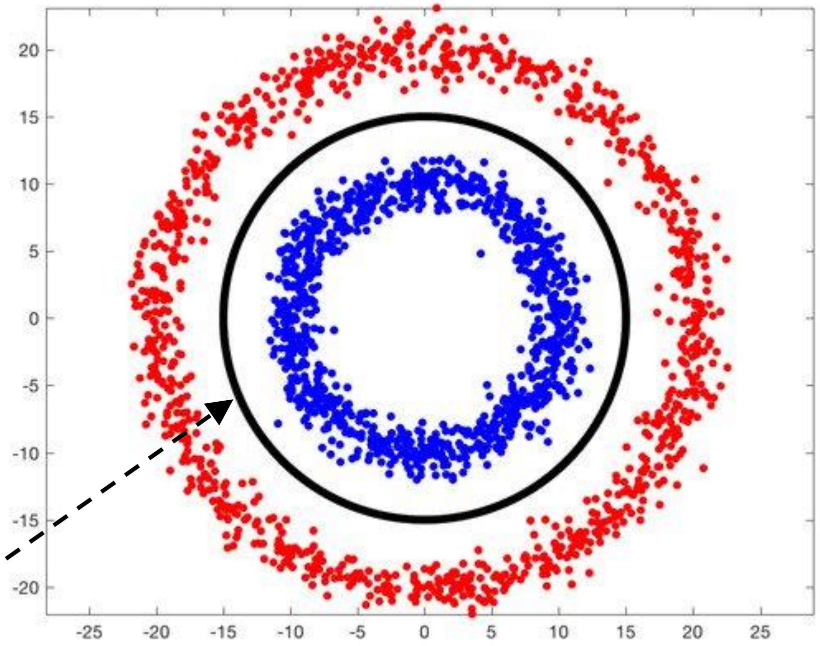


decision boundary

Source: https://www.learnopencv.com/understanding-activation-functions-in-deep-learning/

Linear separability for classification problems

- But clearly not all real life data will be linearly separable
- **Example**: Data within the red and blue circles denote the two classes
- **Question:** What function can be used to allow data such as a circle to be linearly separated ?

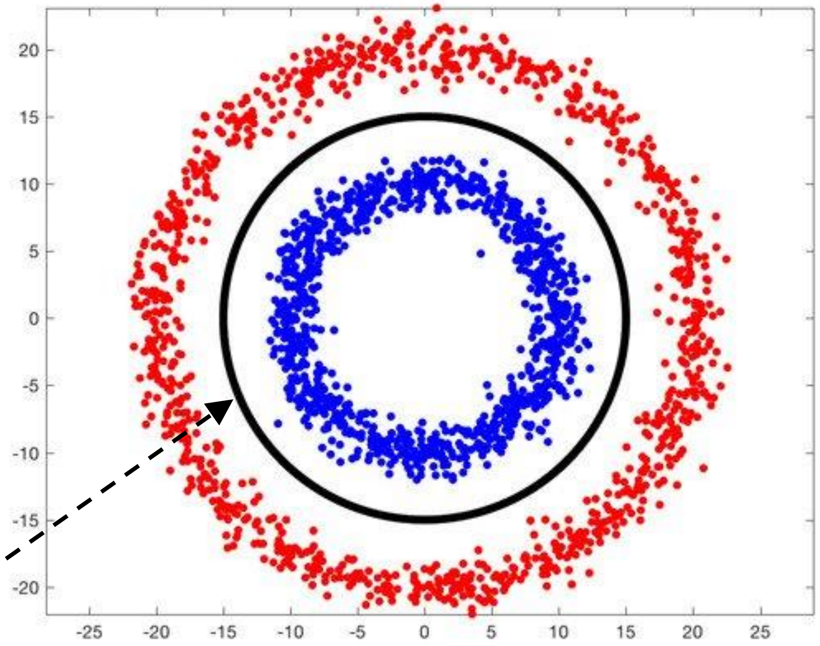


decision boundary

Source: https://www.learnopencv.com/understanding-activation-functions-in-deep-learning/

Linear separability for classification problems

- **Example**: What function can be used to allow data such as a circle to be linearly separated ?



decision boundary

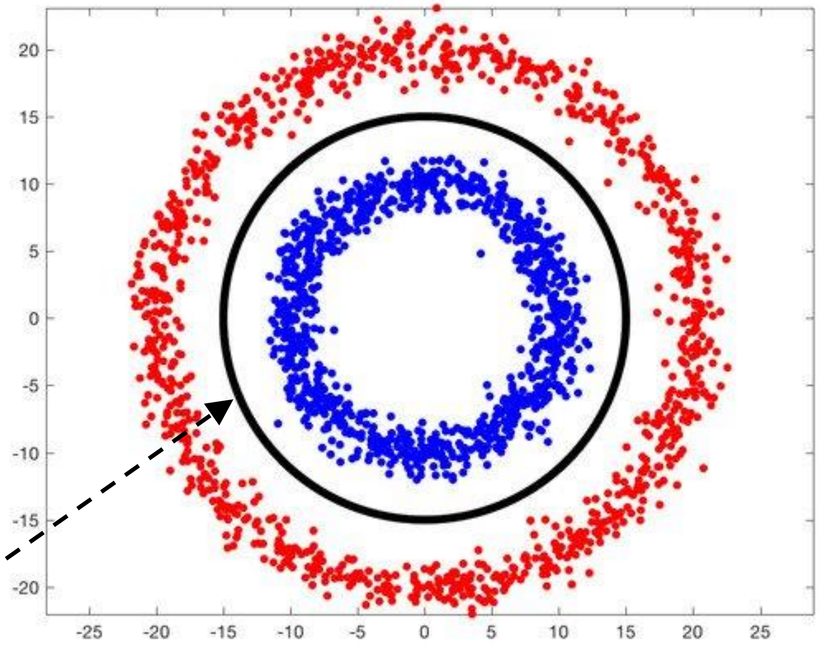
$$x^2 + y^2 = 15^2$$

Source: https://www.learnopencv.com/understanding-activation-functions-in-deep-learning/

Linear separability for classification problems

- **Example**: What function can be used to allow data such as a circle to be linearly separated ?
- **Answer**:

$$(x,y) \rightarrow (x^2, y^2)$$



decision boundary

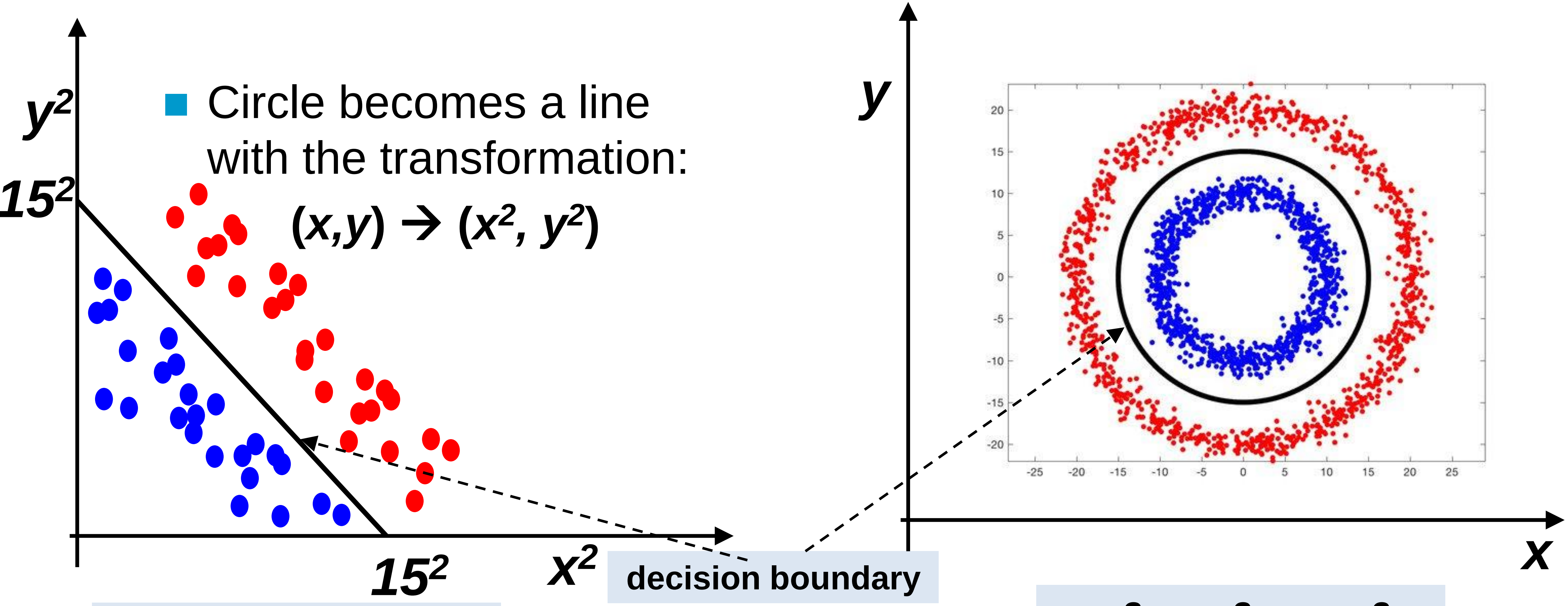
$$x^2 + y^2 = 15^2$$

Source: https://www.learnopencv.com/understanding-activation-functions-in-deep-learning/

Linear separability for classification problems

- Circle becomes a line with the transformation:

$$(x,y) \rightarrow (x^2, y^2)$$



$$a + b = 15^2$$

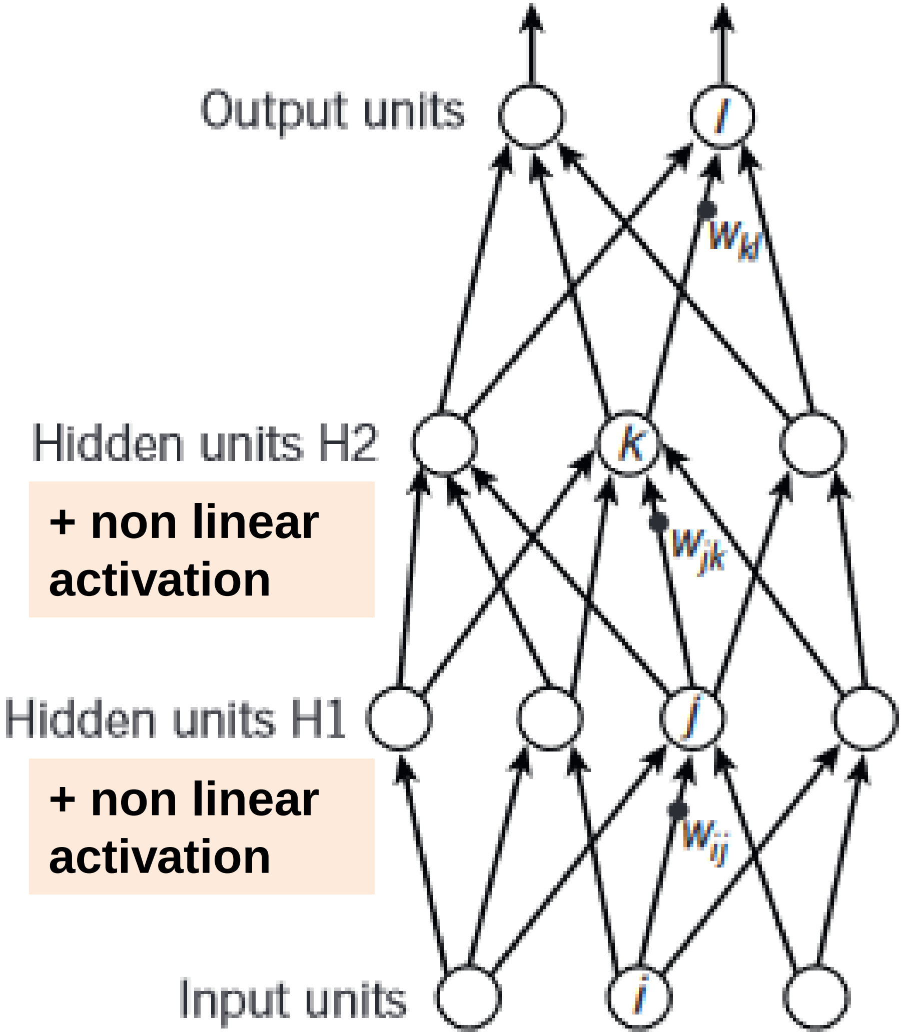
Line equation

$$x^2 + y^2 = 15^2$$

Circle equation

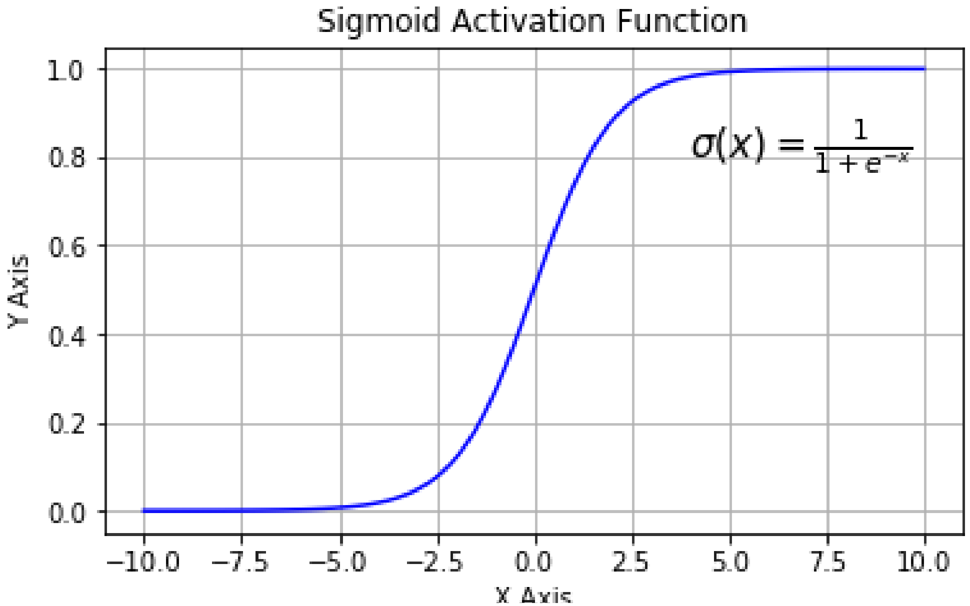
Non linear Activation functions

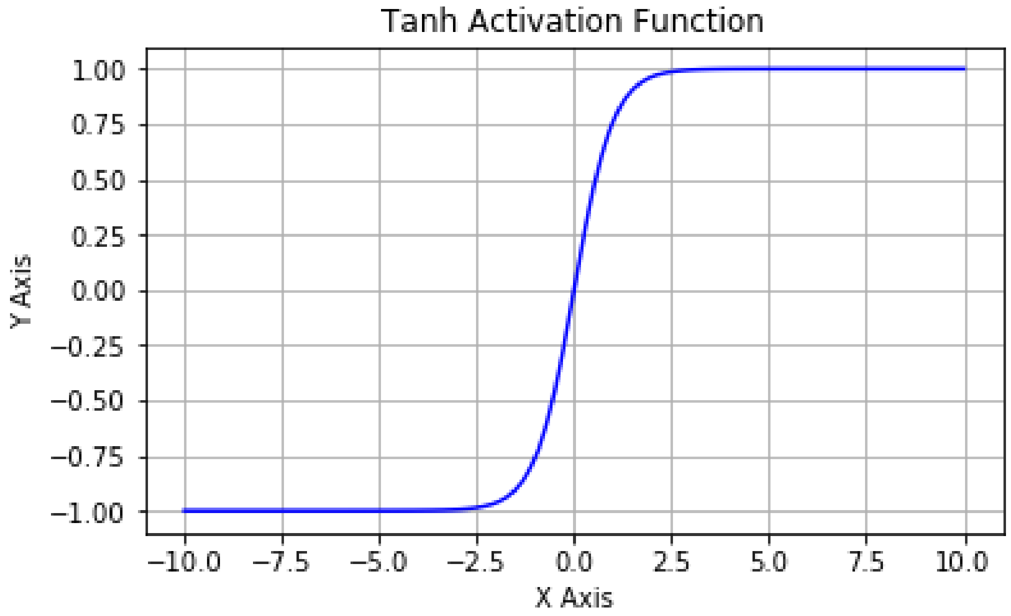
- The squaring operation is an example of a non linear function
- Also known as **non-linear activation functions**
- There are many popular non-linear activation functions:
  - Sigmoid
  - Relu
  - Tanh
  - …..

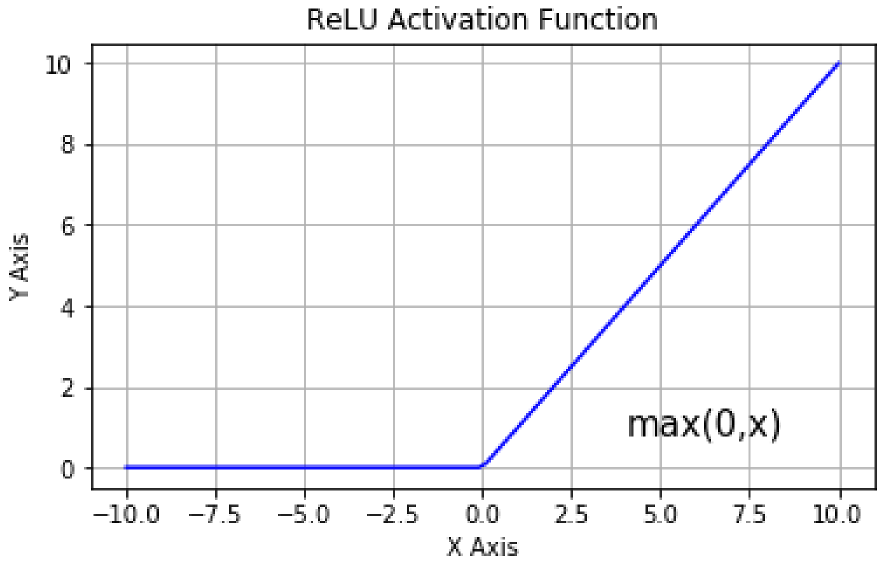


Source: LeCun et. al. Deep Learning

Activation functions



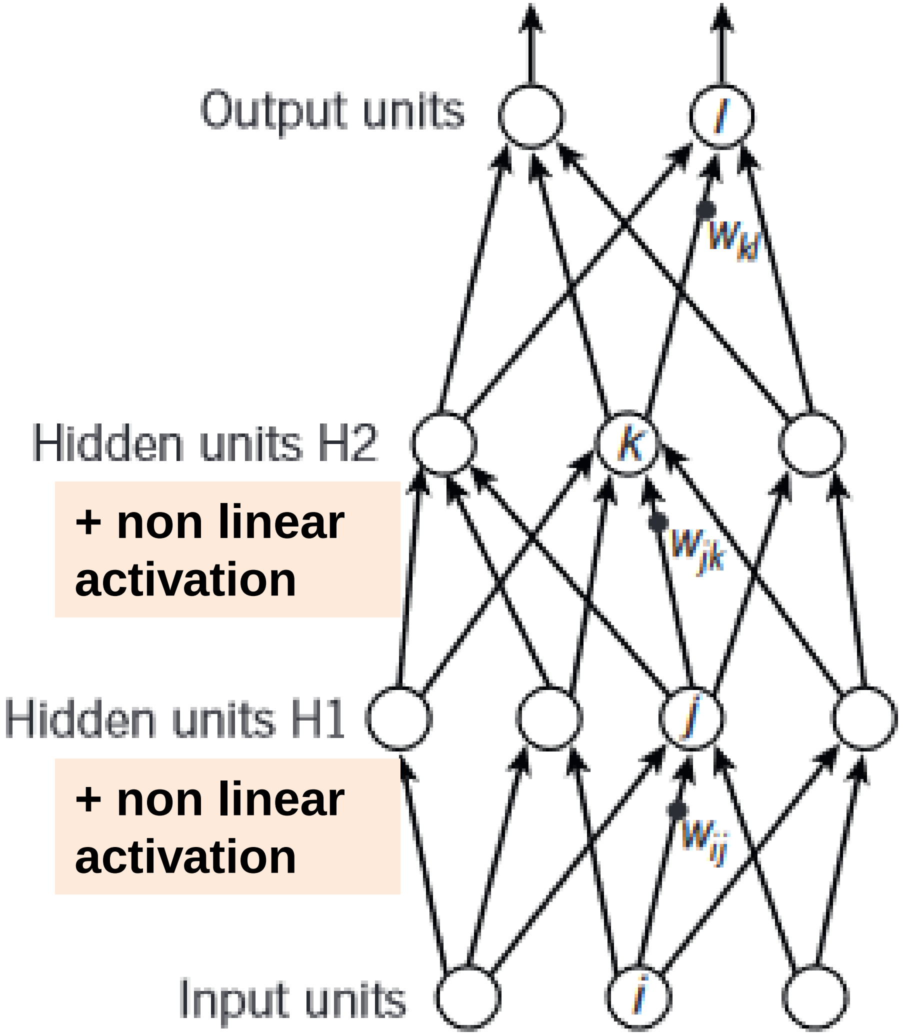




Source: https://www.learnopencv.com/understanding-activation-functions-in-deep-learning/

Multilayer neural networks – Key ideas

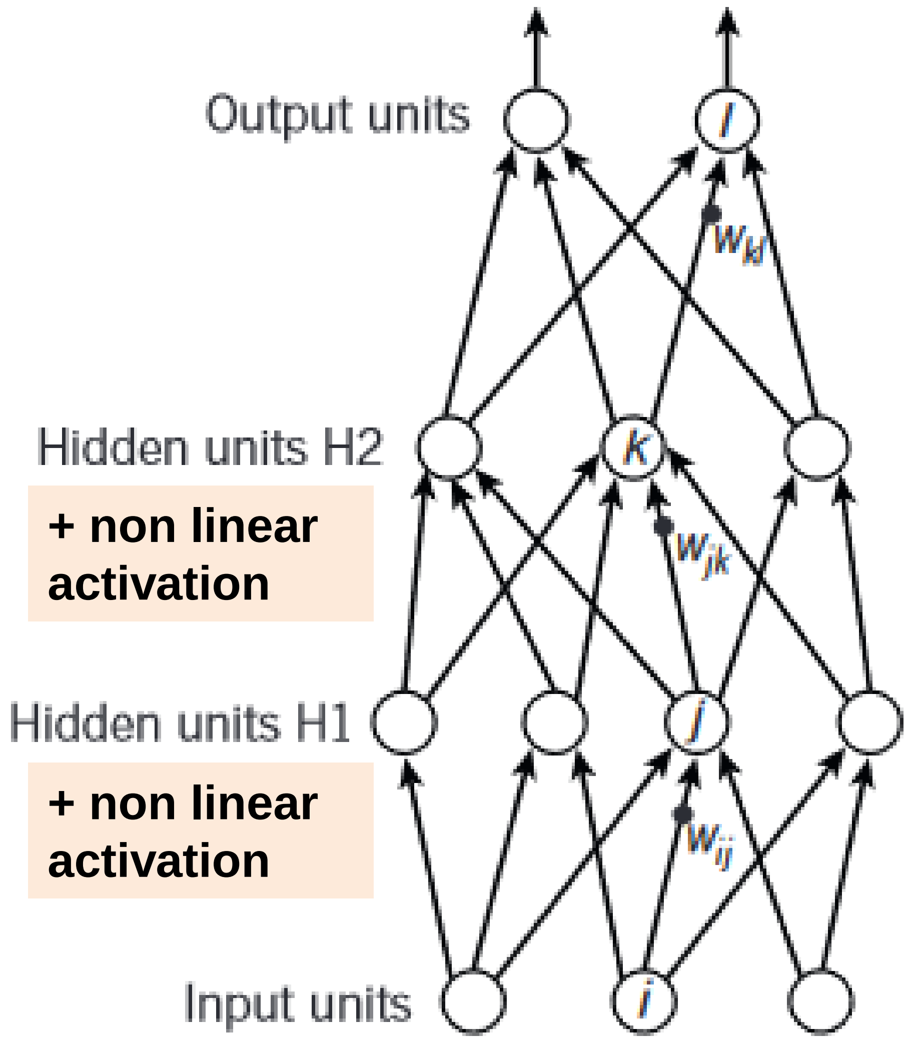
- Combine neurons with non linear activation functions to build complex neural networks



Source: LeCun et. al. Deep Learning

Multilayer neural networks – Key ideas

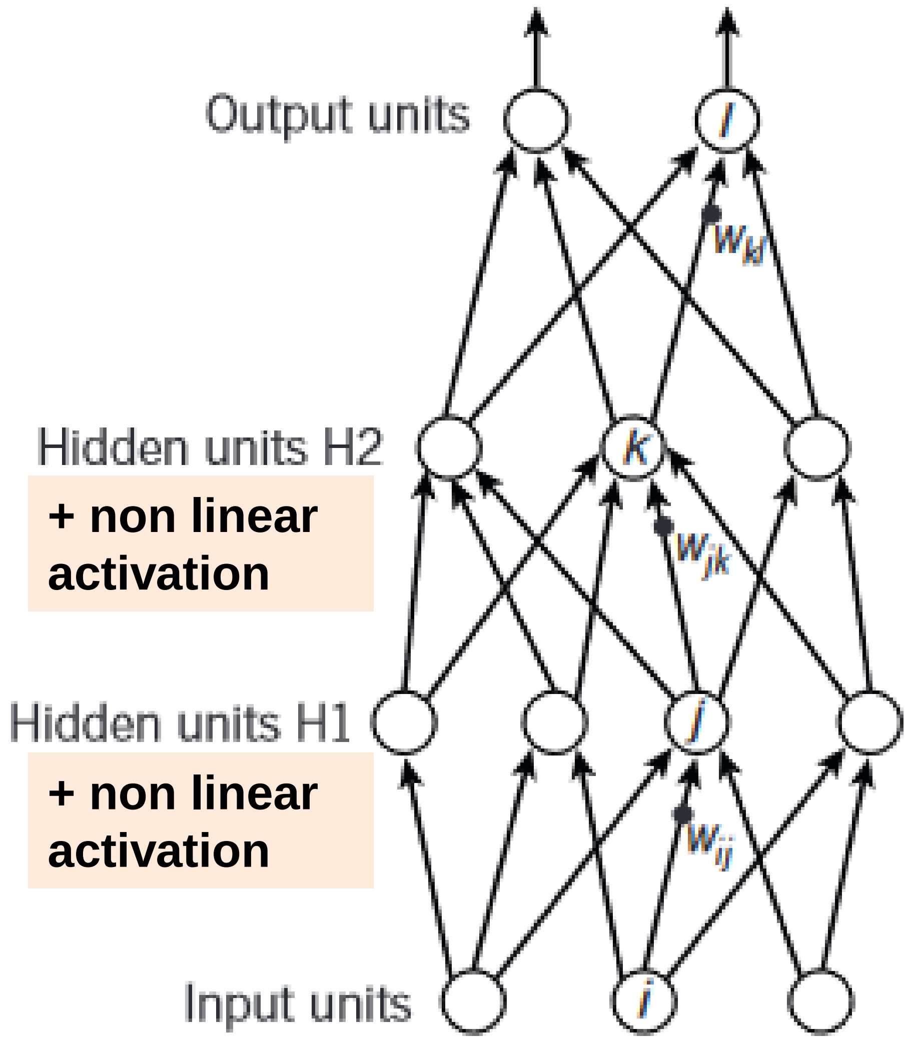
- Combine neurons with non linear activation functions to build complex neural networks
- **Combinations of different activation functions in such networks can simulate any feature transformation**
- Also known as universal function approximators



Source: LeCun et. al. Deep Learning

Multilayer neural networks – Key ideas

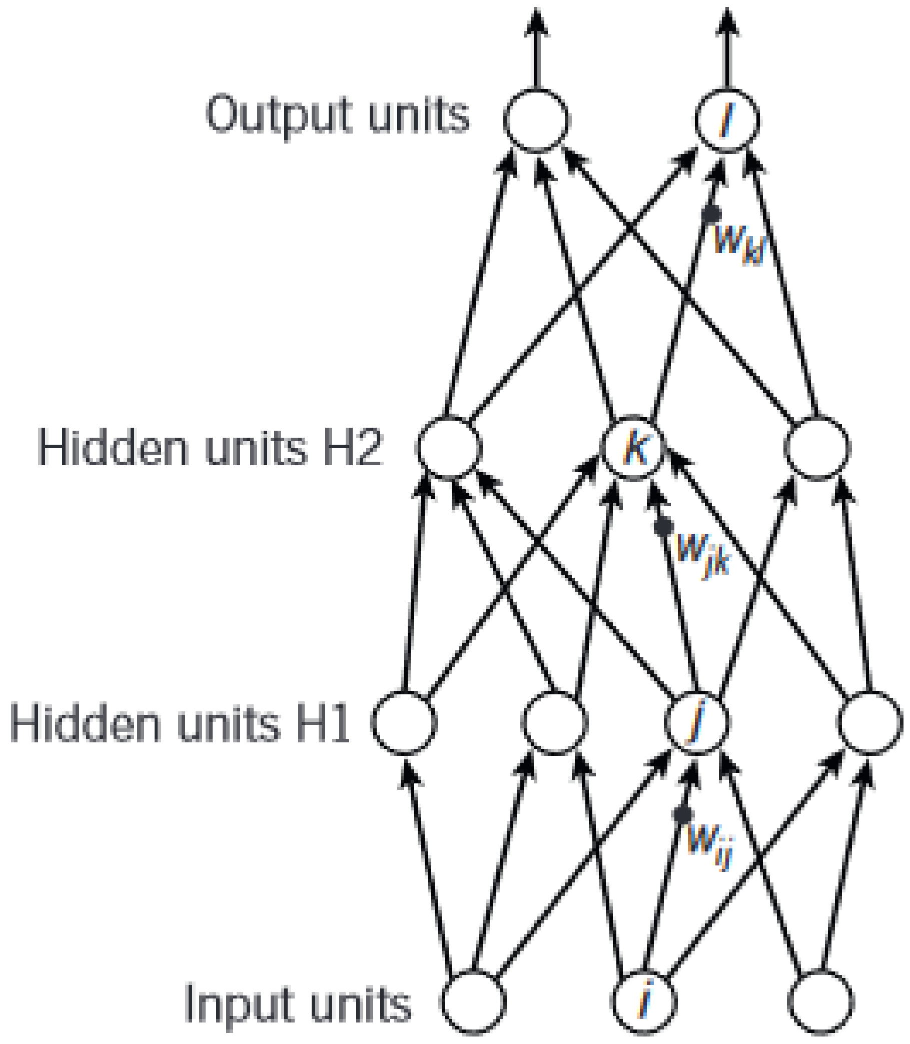
- Combine neurons with non linear activation functions to build complex neural networks
- **Combinations of different activation functions in such networks can simulate any feature transformation**
- This is a key essence of deep learning.
- Manual feature transformation i.e. human engineering is no longer needed
- Get the machines to do the hard work



Source: LeCun et. al. Deep Learning

Conversely – note that

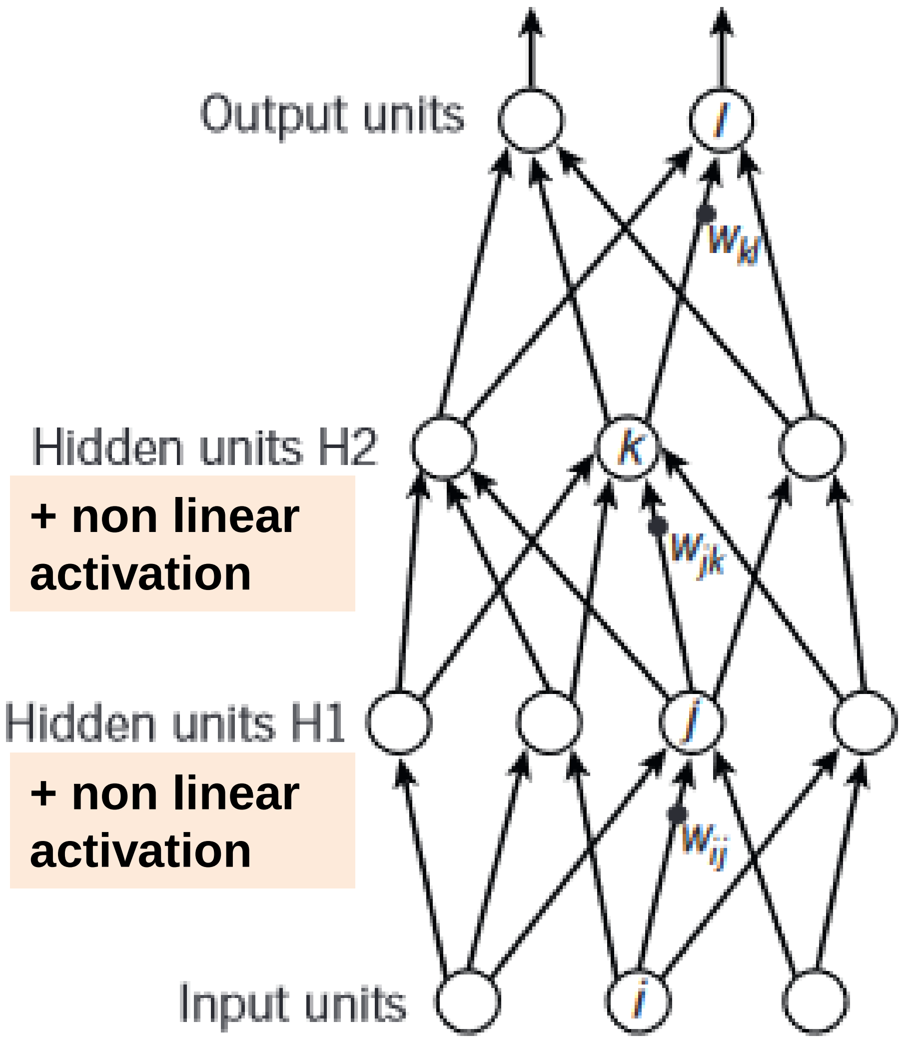
- If each unit is linear then the combination of linear units is still linear
- Thus a multilayer but linear neural network is equivalent to a single layer
- Hence stacking linear layers does not give any advantage



Source: LeCun et. al. Deep Learning

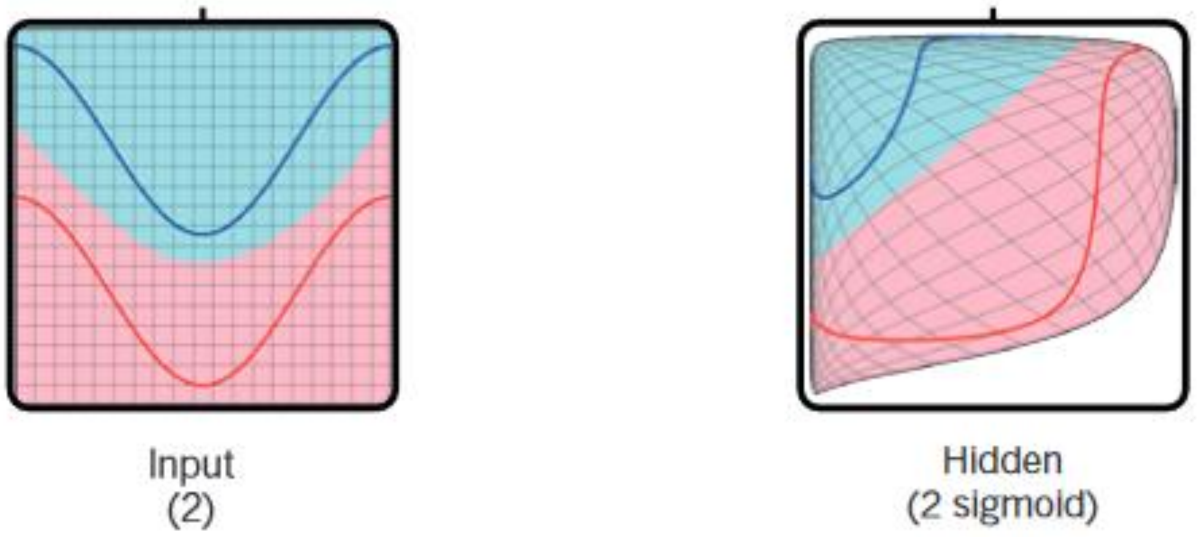
Activation functions

- Using **non-linear function** activation functions in the hidden layers will result in a complex neural network
- Such a network will not be equivalent to a simpler neural network (in general)
- Such complex networks gives deep learning algorithms the capacity to automatically learn the relevant features

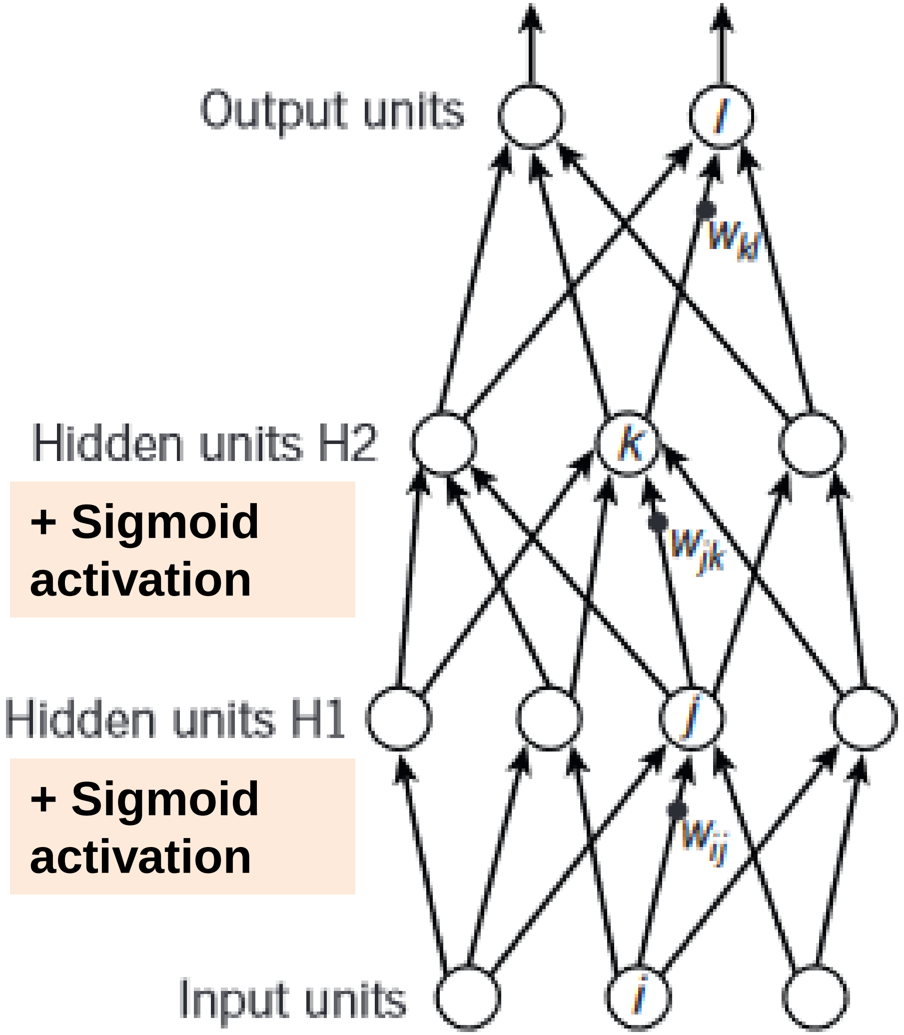


Source: LeCun et. al. Deep Learning

Activation functions



- **Non linear transformation** of the input features is achieved by the hidden units
- This permits the final layer to achieve a linear separation of the two classes



Source: LeCun et. al. Deep Learning

Computer vision

Computer Vision

High level AI goals within computer vision:

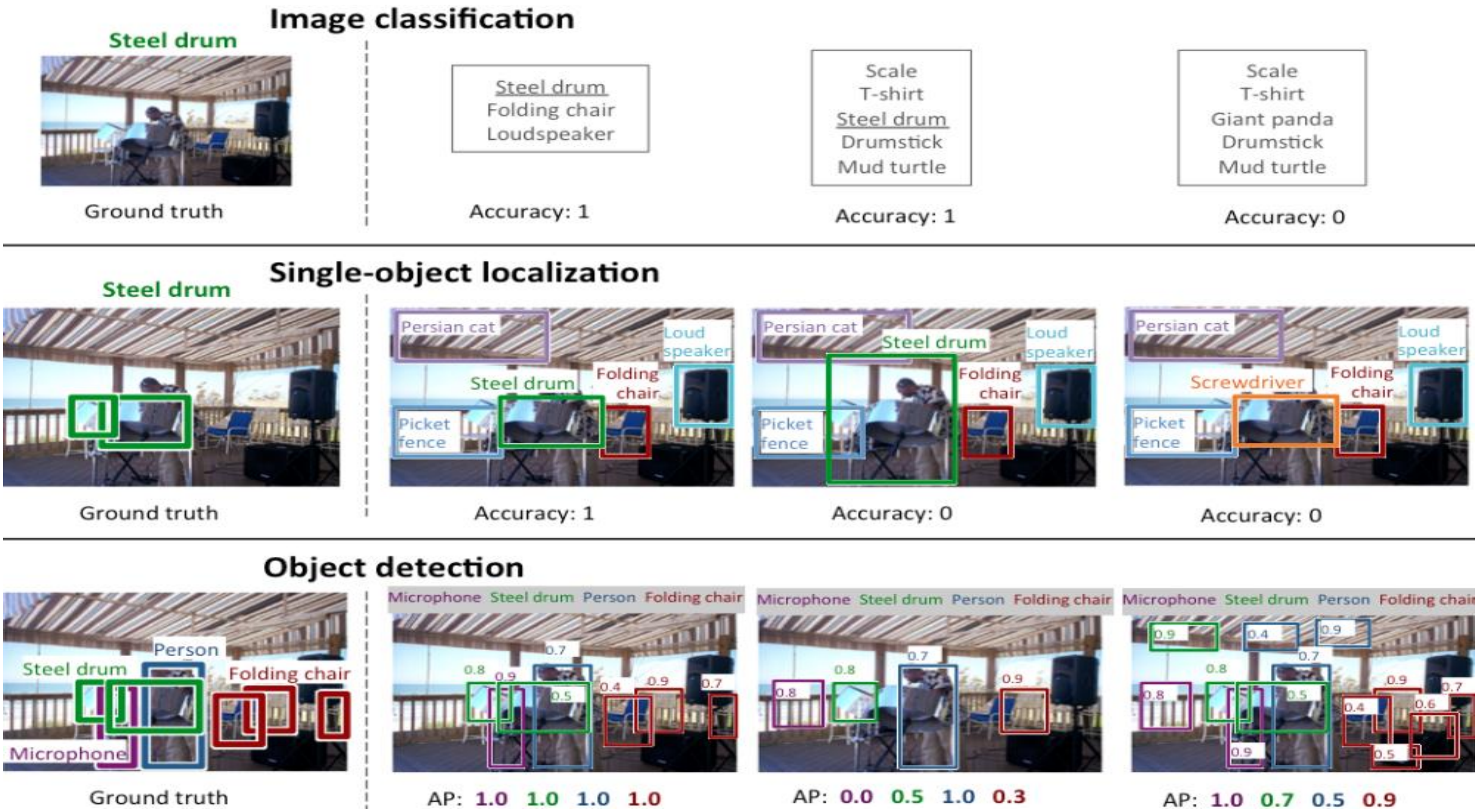
- Develop human level and beyond human level vision capabilities
- Handle multiple visual modalities – visual, infra-red, x-ray, ultrasound, UV, satellite (multispectral), video, 3D, 2D etc.
- Characteristic problems in CV : image classification, object recognition and localisation, image restoration, person/face identification, 2D to 3D reconstruction, multiview 3D reconstruction, medical image analysis, style transfer, OCR, handwriting recognition, gait analysis, …..

Imagenet large scale CV dataset

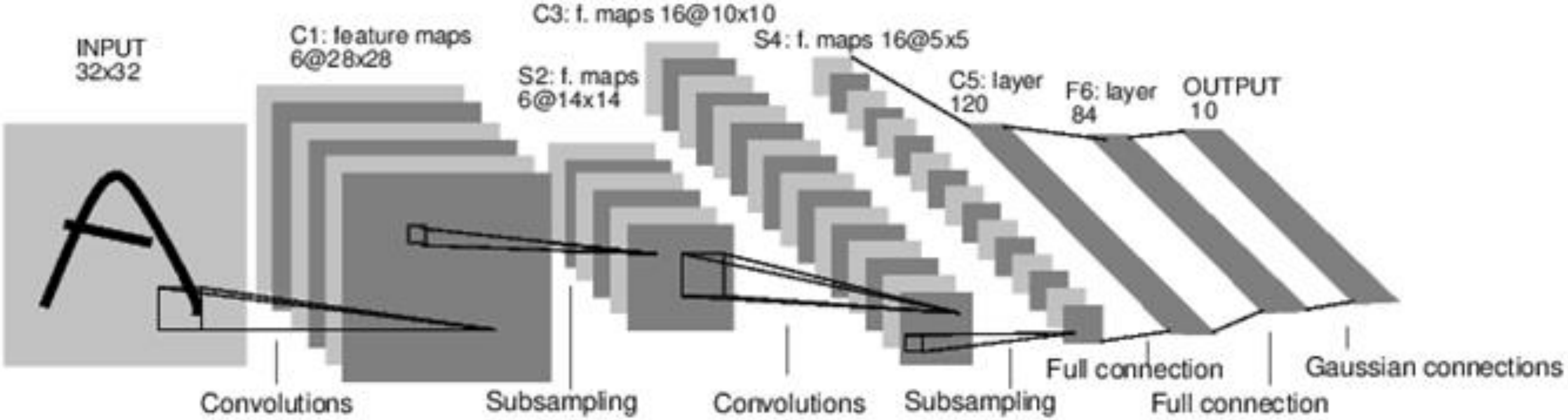
- ImageNet contains 14M+ annotated images
- Annotation done using large scale **Amazon Mechanical Turk**
- ImageNet populates 21,841 concepts of WordNet with an average of 650 manually verified and full resolution images
- ImageNet allows mapping visual concepts to words
- It permits building general purpose visual recognition systems

Imagenet classes

Imagenet Tasks



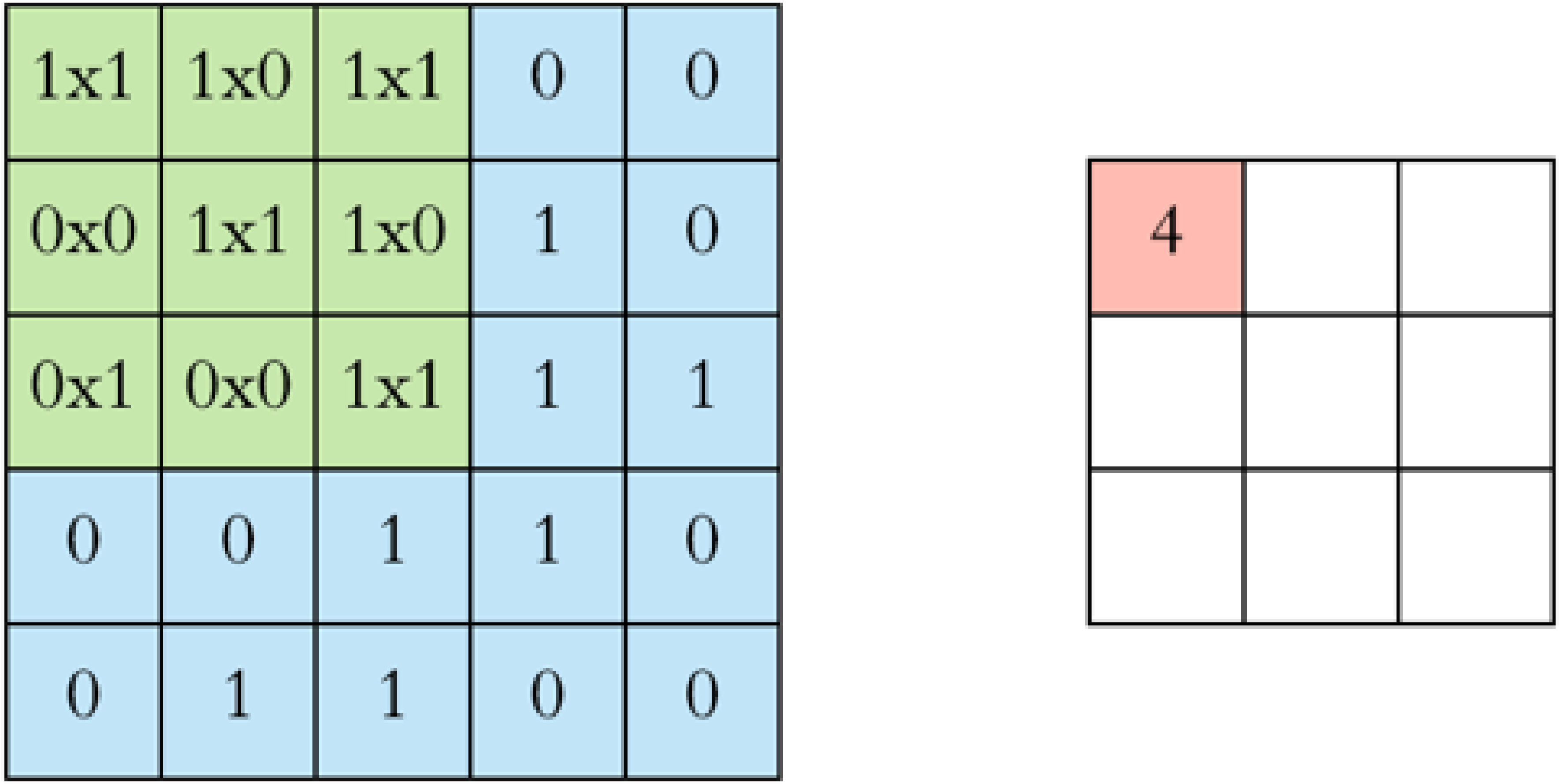
Convolutional Neural Networks



A Full Convolutional Neural Network (LeNet)

LeNet CNN

CNN filters



CNN filter in action

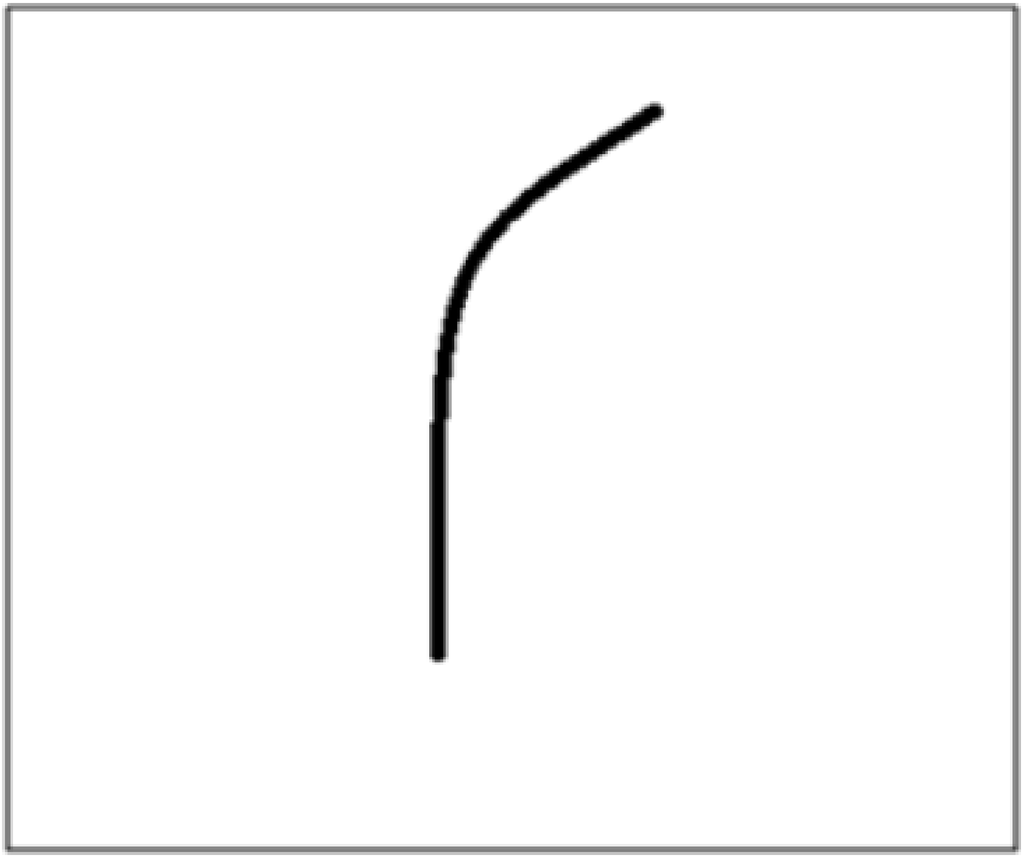
1x1	1x0	1x1	0	0
0x0	1x1	1x0	1	0
0x1	0x0	1x1	1	1
0	0	1	1	0
0	1	1	0	0

4		

CNN filter learning a shape

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

Pixel representation of filter

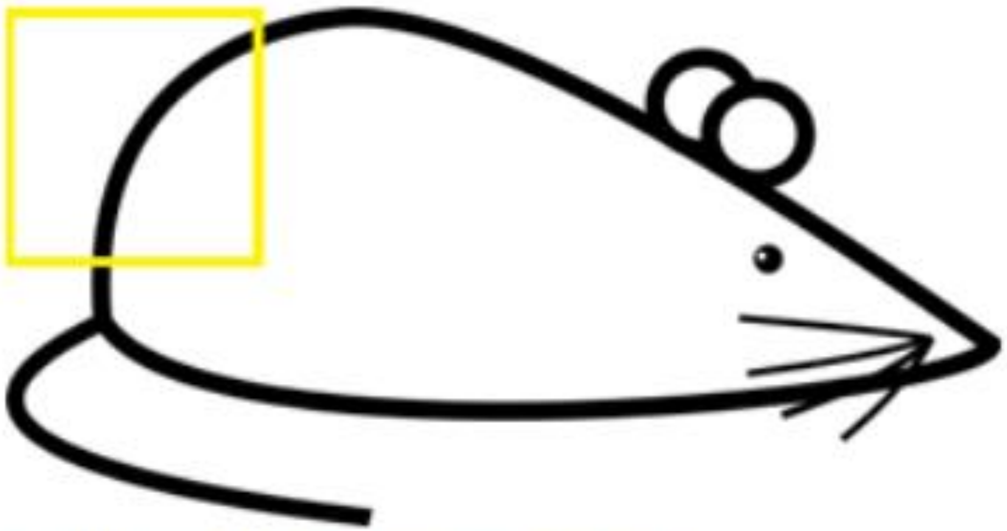


Visualization of a curve detector filter

Source: https://adeshpande3.github.io/adeshpande3.github.io

CNN filter learning a shape

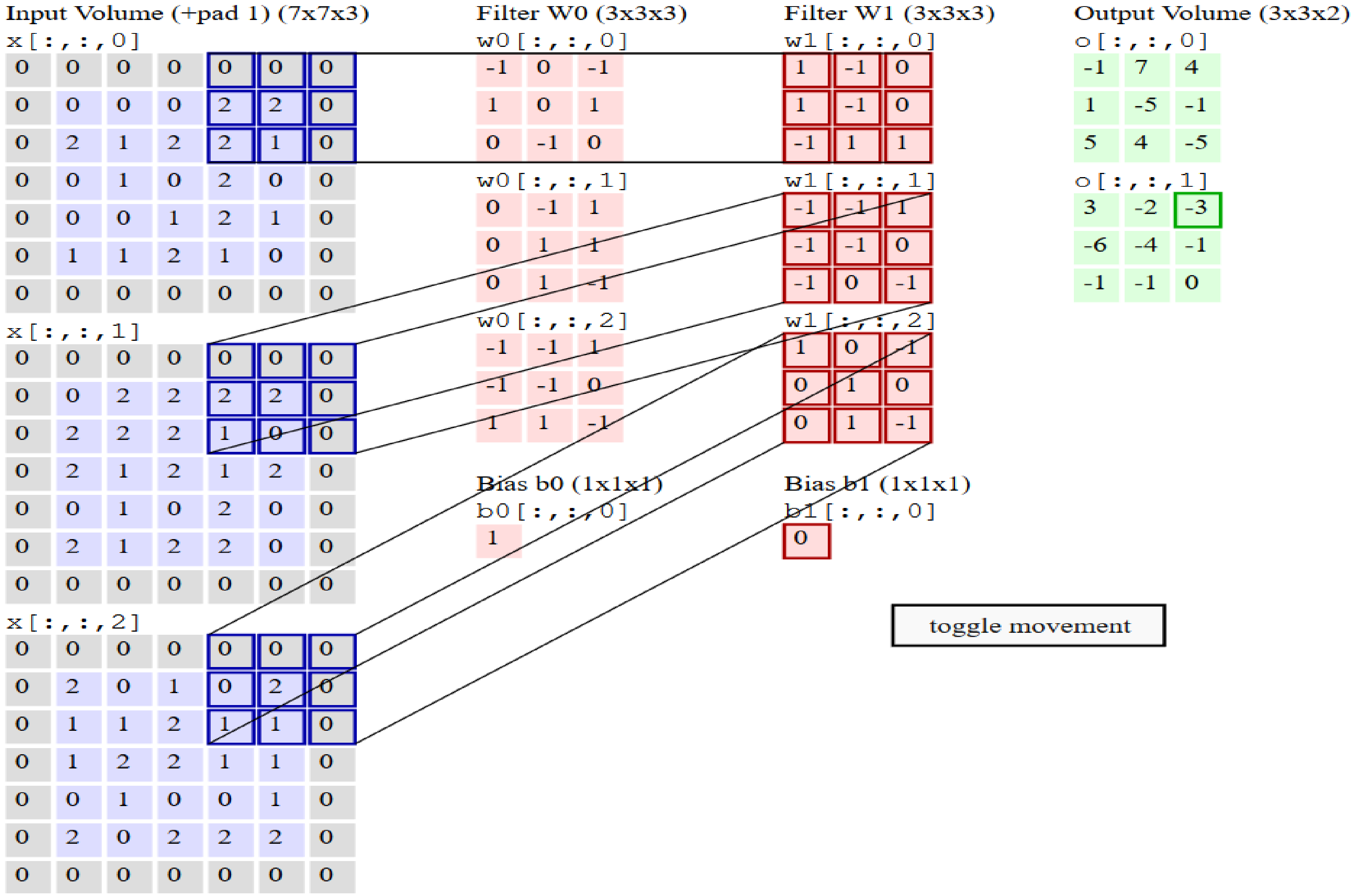
Original image



Visualization of the filter on the image

Source: https://adeshpande3.github.io/adeshpande3.github.io

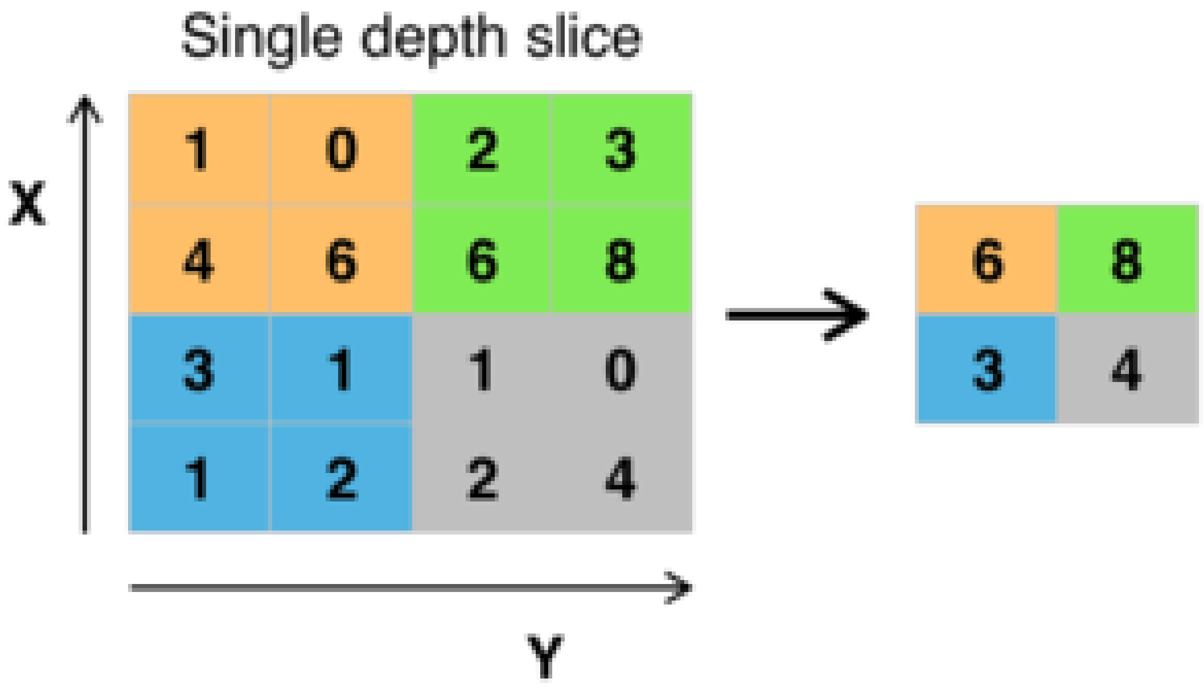
CNN filter in action



Source: https://adeshpande3.github.io/adeshpande3.github.io

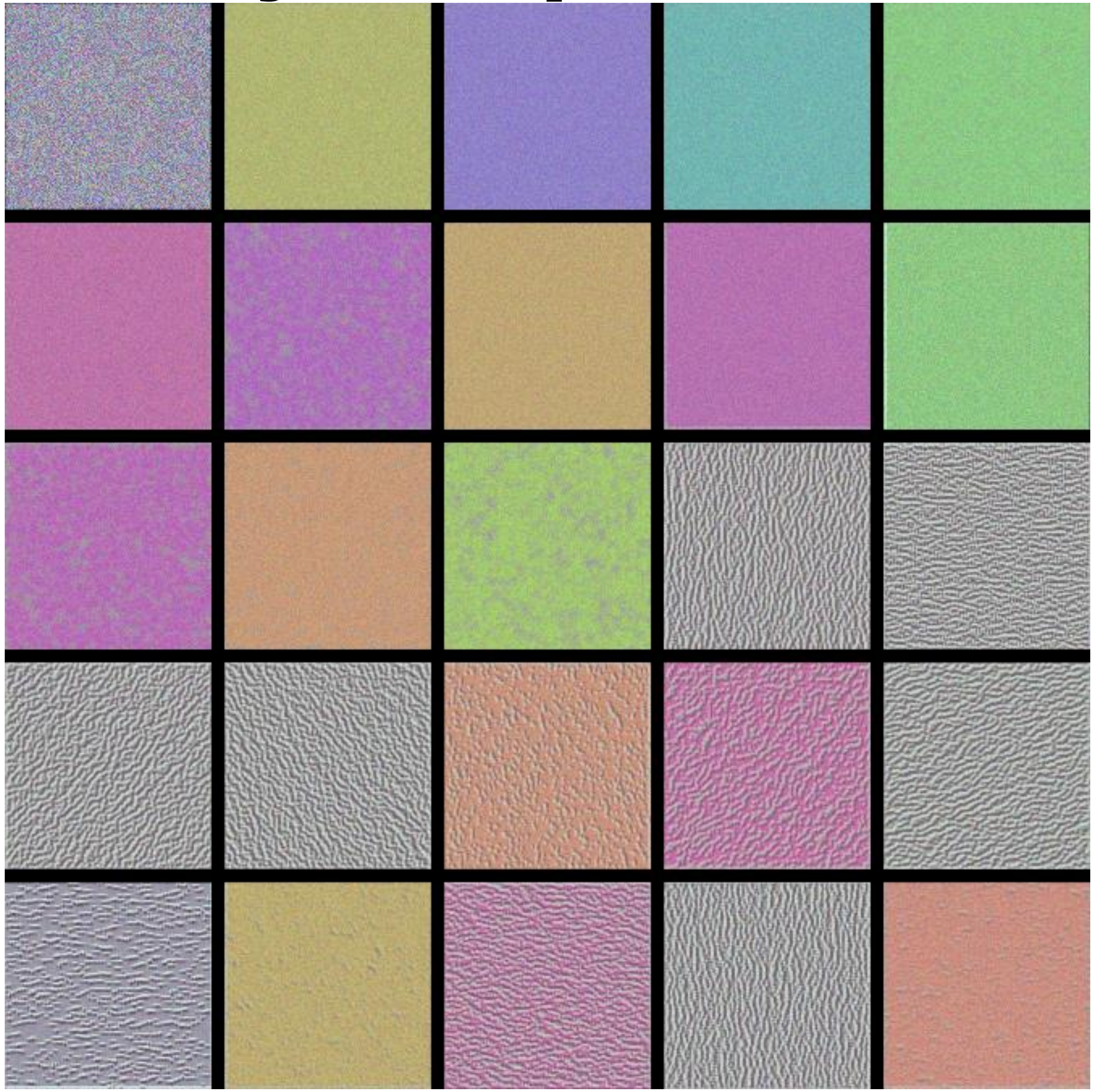
Sub-sampling – Max pooling

- Max pooling is an instance of a sub-sampling method
- Sub-sampling typically means choosing a subset of the features
- So, the idea is to identify the important features and ignore the rest



Filters learnt by deep CNNs

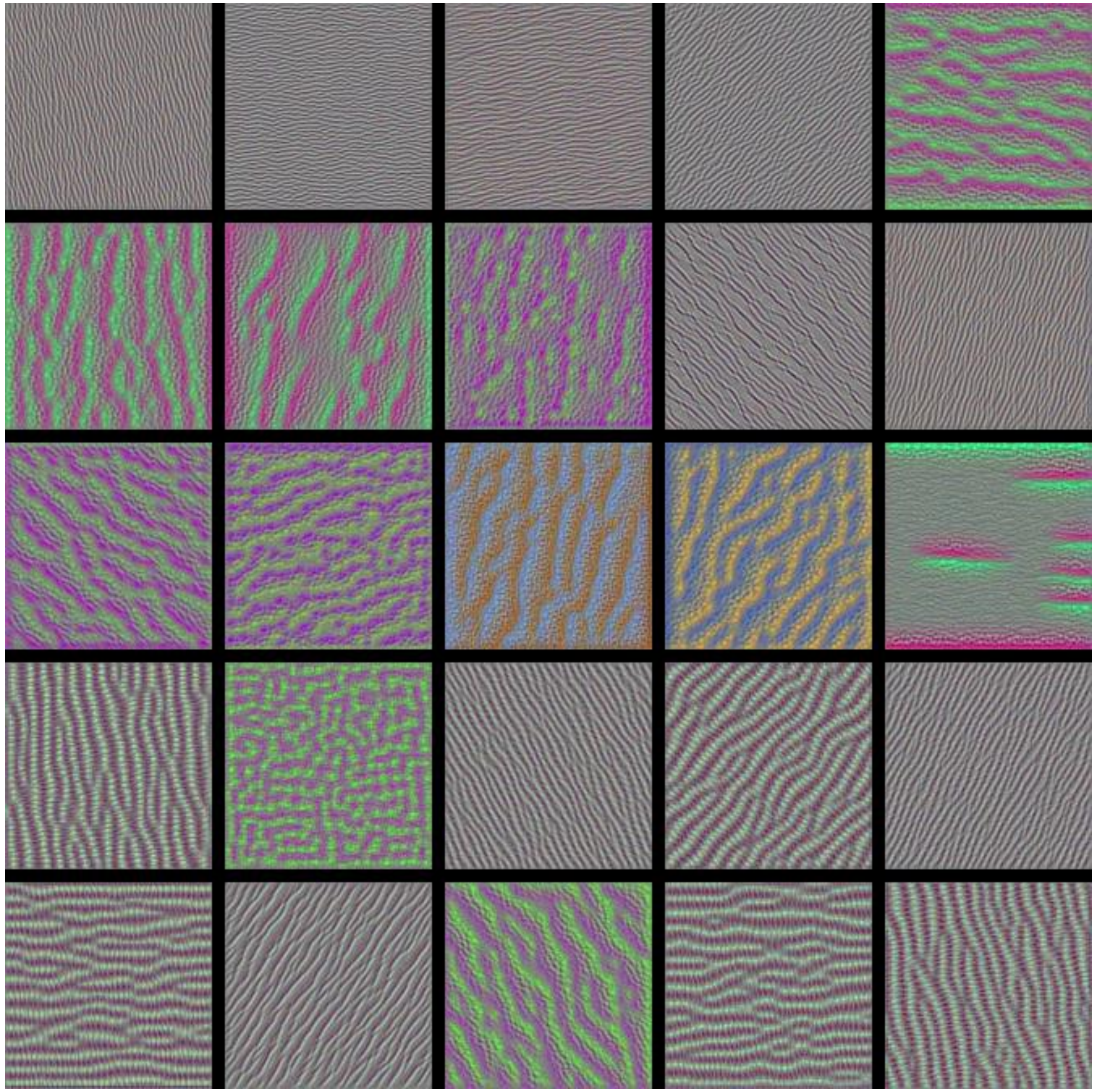
First layer



Source: https://deeplizard.com/learn/video/cNBBNAxC8l4

Filters learnt by deep CNNs

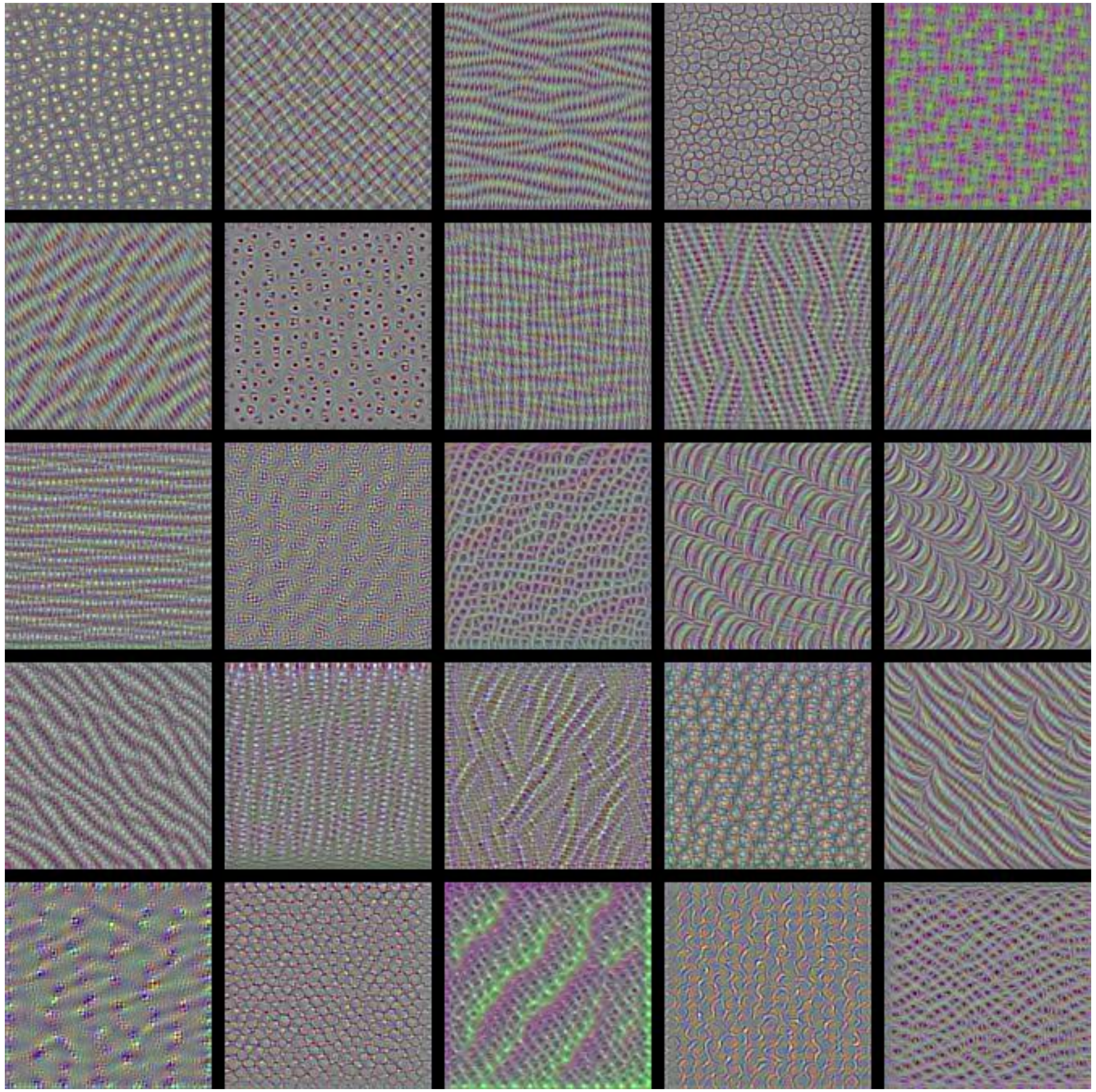
Second layer



Source: https://deeplizard.com/learn/video/cNBBNAxC8l4

Filters learnt by deep CNNs

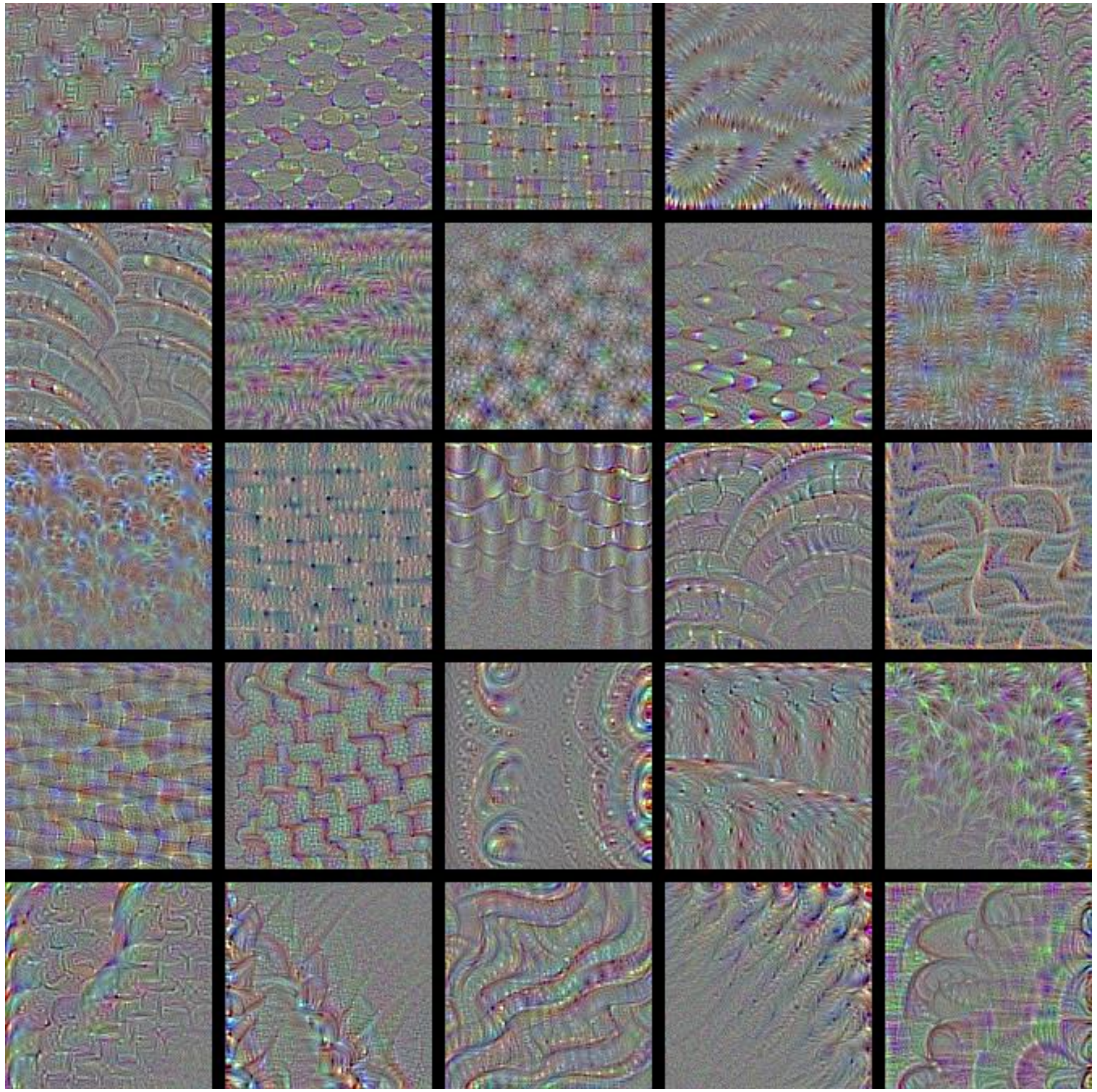
Third layer



Source: https://deeplizard.com/learn/video/cNBBNAxC8l4

Filters learnt by deep CNNs

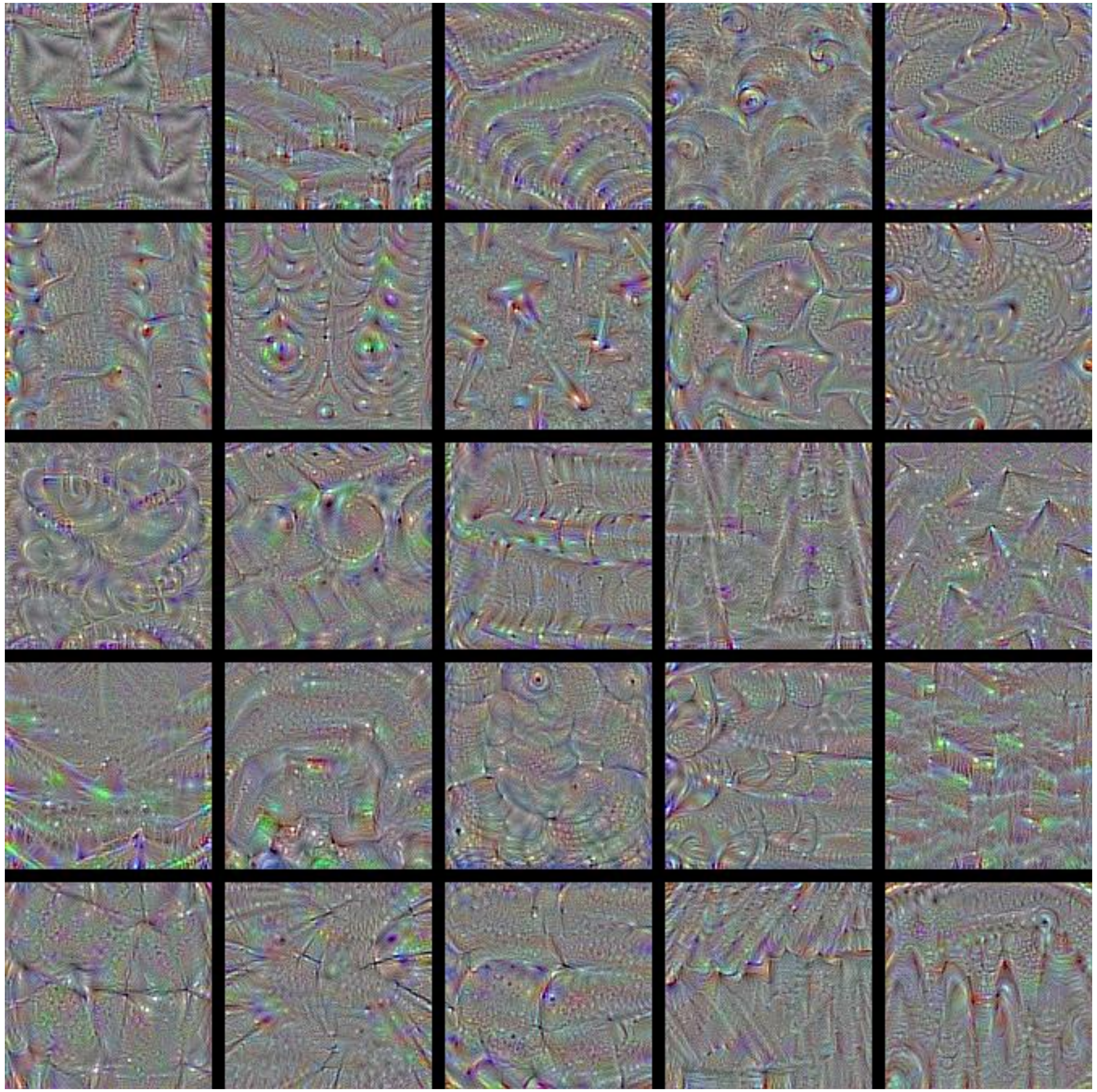
Fourth layer



Source: https://deeplizard.com/learn/video/cNBBNAxC8l4

Filters learnt by deep CNNs

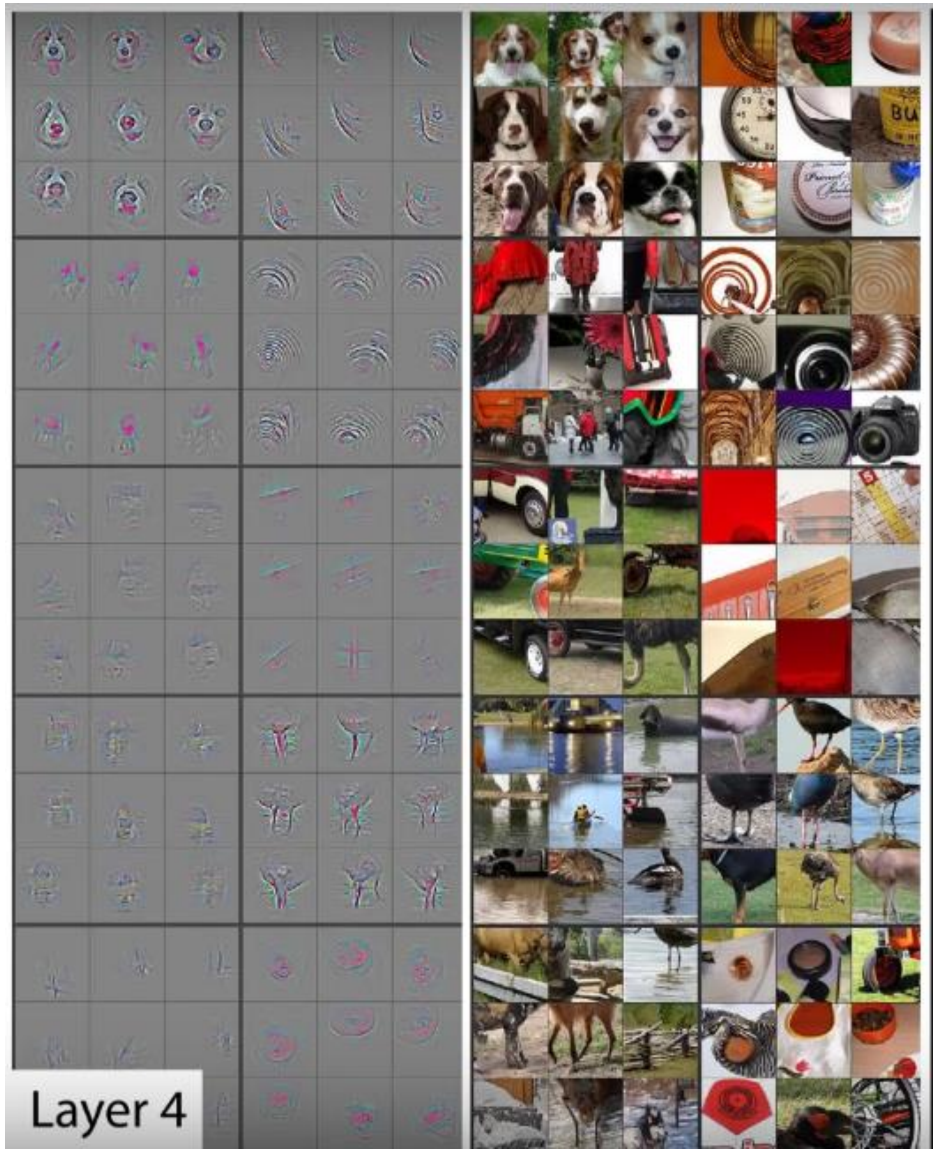
Fifth layer



Source: https://deeplizard.com/learn/video/cNBBNAxC8l4

Filters learnt by deep CNNs

Fourth layer

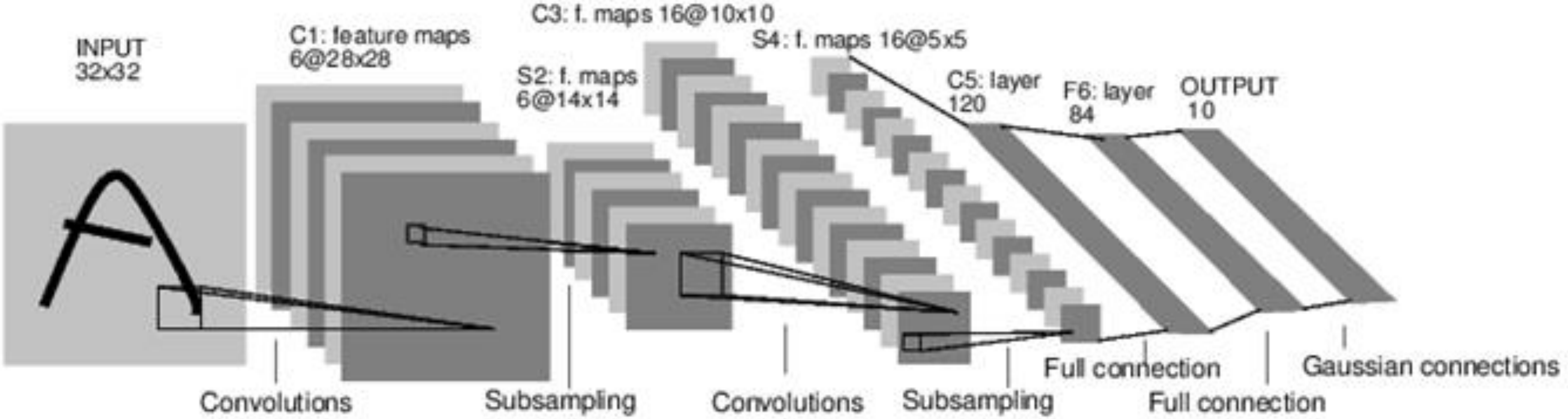


Source: https://deeplizard.com/learn/video/cNBBNAxC8l4

Filters learnt by deep CNNs

Source: https://www.auduno.com/images/vgg_filter_10_crop.jpg

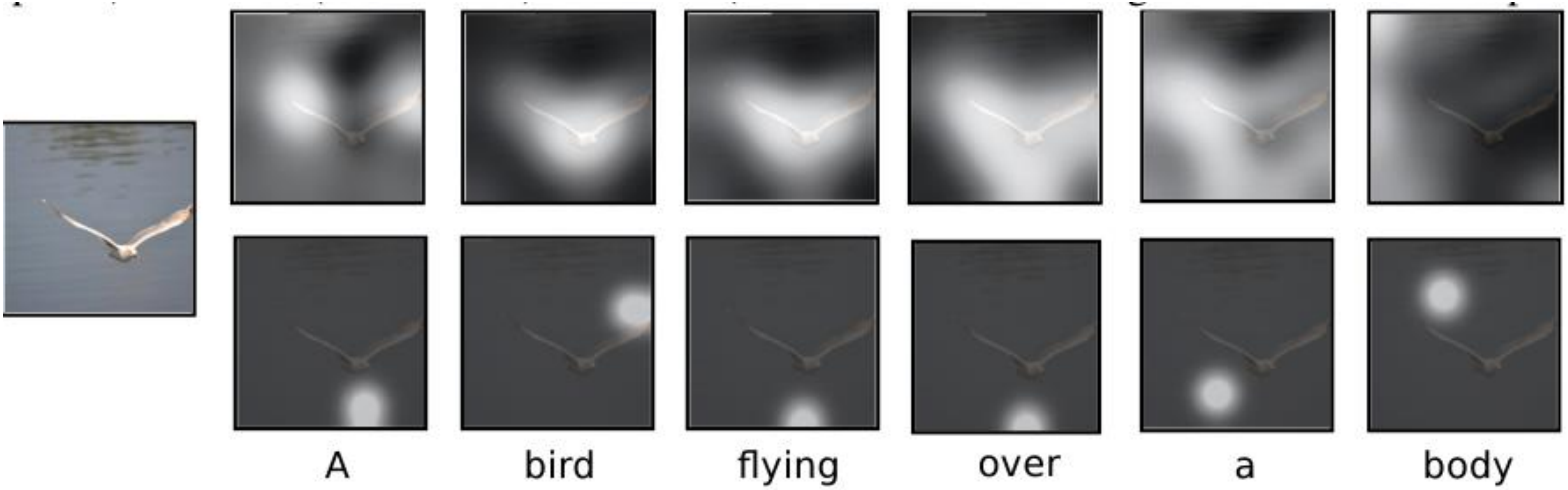
Convolutional Neural Networks



A Full Convolutional Neural Network (LeNet)

LeNet CNN

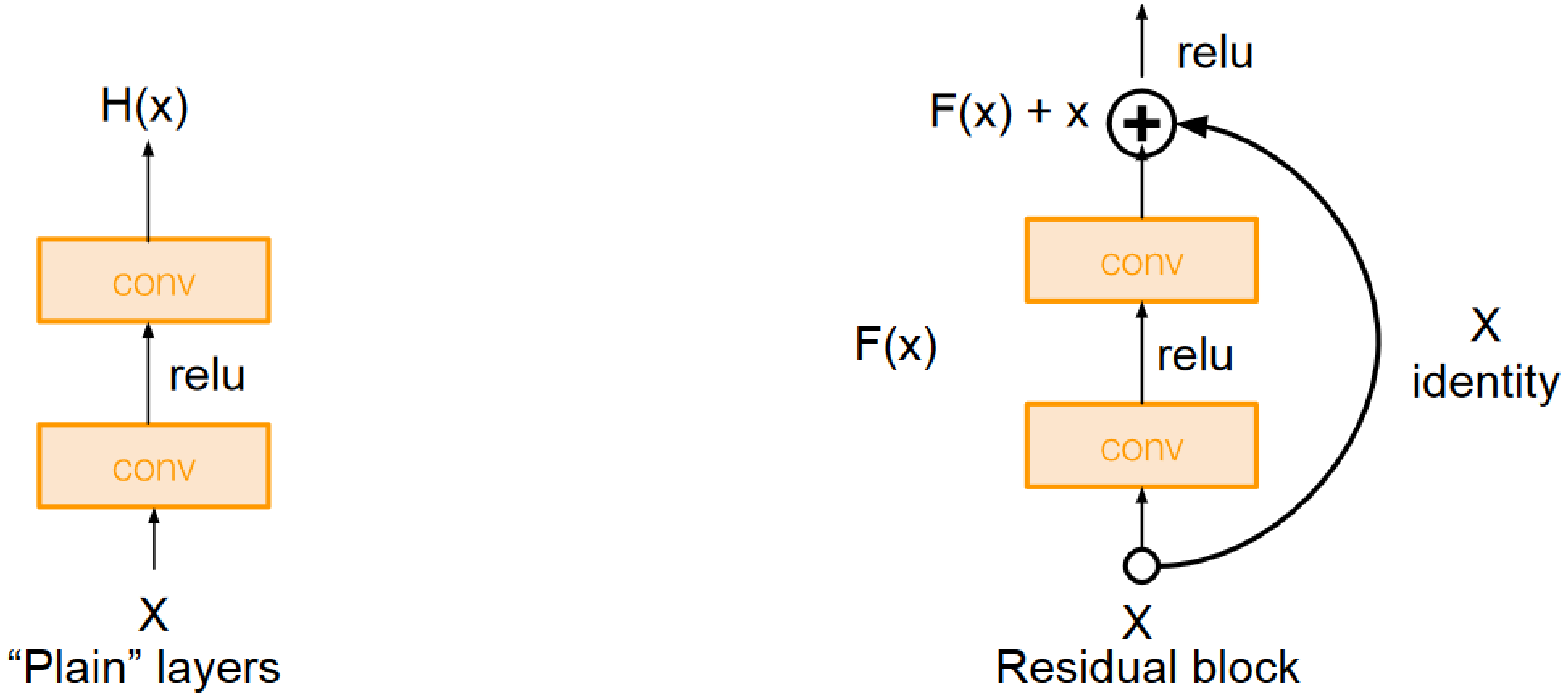
CNNs with Attention



- Attention is a masking mechanism to blank out irrelevant areas of the image
- The attention mask is multiplied with the CNN feature output (aka gating)

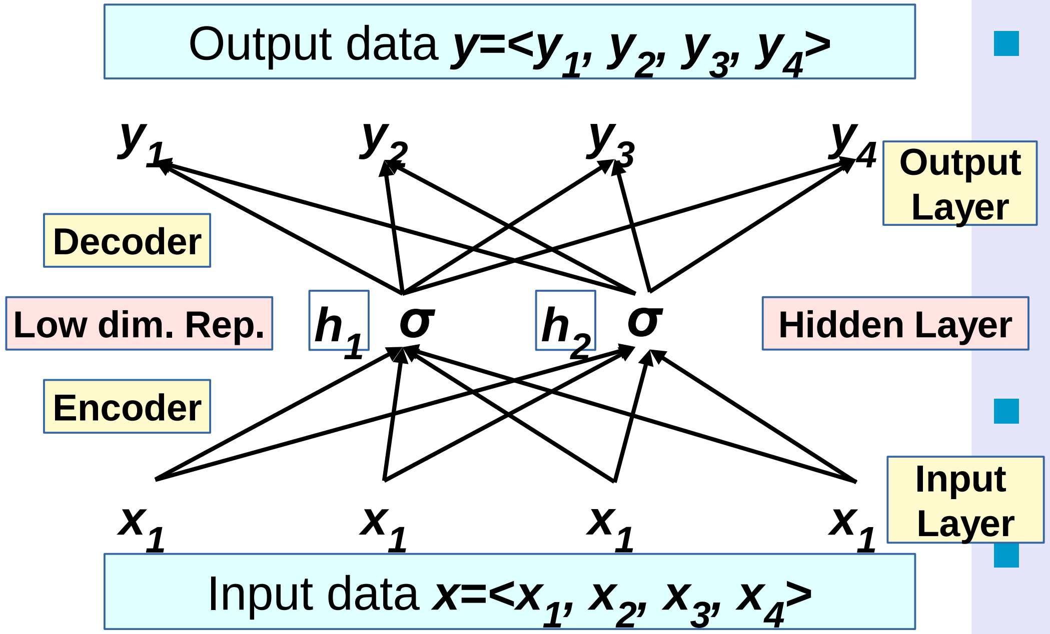
Source : Xu et. al. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention

RESIDUAL LEARNING



Autoencoder

- *Autoencoder* is a generic term used to describe a class of methods for generating low dimensional representations using neural networks.
- There are a large variety of autoencoders (including those based on more complex neural networks such as CNNs, LSTMs, GANs, VAEs etc.)
- The basic structure of an autoencoder is depicted here

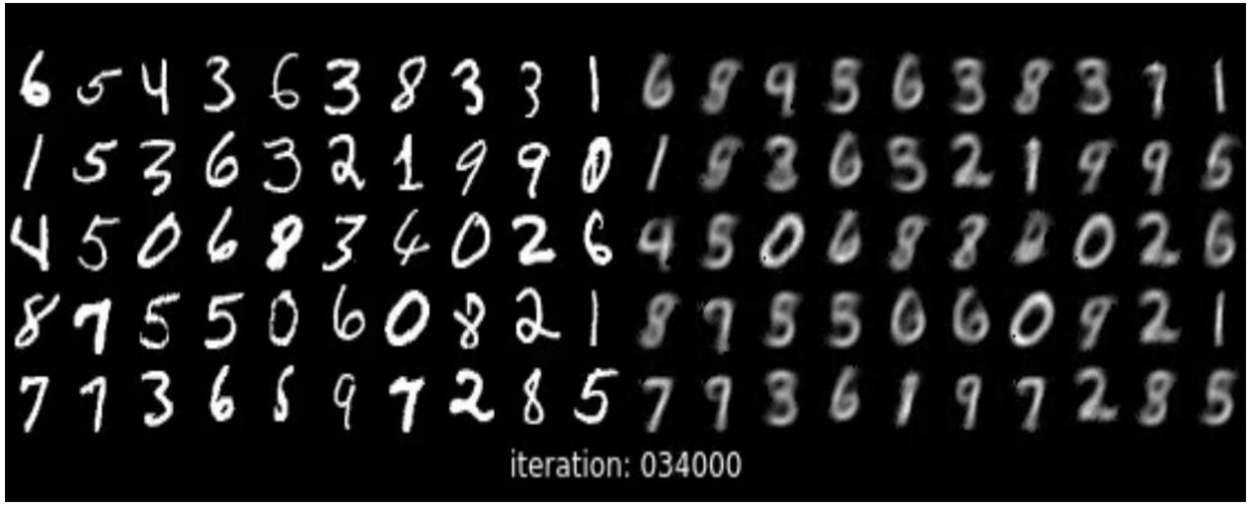


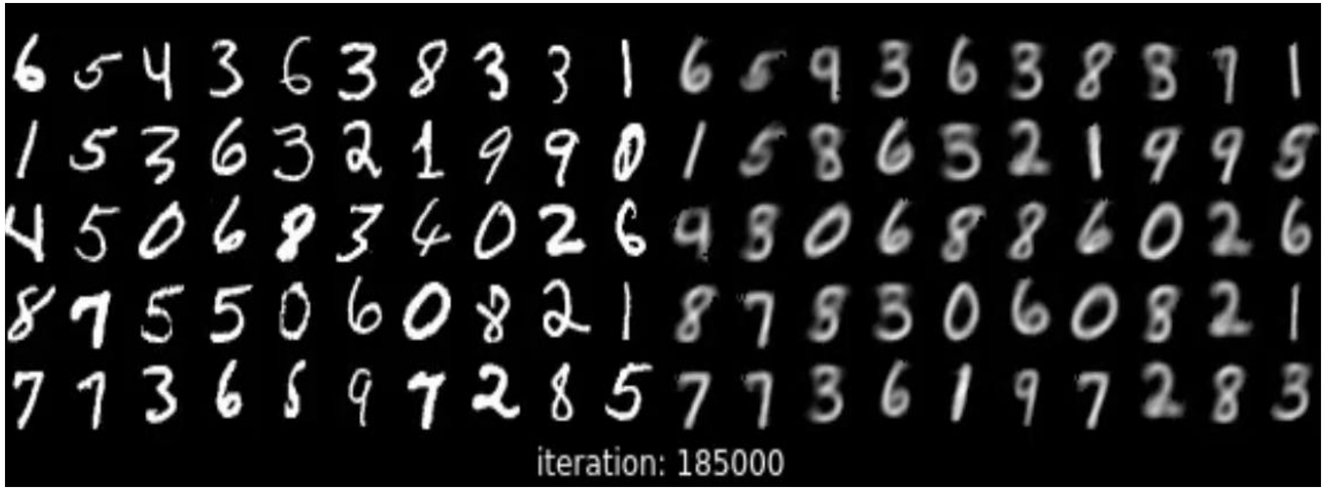
- Common to all autoencoders is that the loss is measured in terms of a **reconstruction error.** In this example, one can use (per example) *squared loss*:

$$L(x,\theta) = \|y - x\|^2 = \|f(x,\theta) - x\|^2$$

- The hidden layer, in this example, given by ***h* = <*h*$_1$, *h*$_2$>** **outputs** the ***low dimensional representation***.
- If no sigmoid units are used, then the **linear** low dim. representation will be similar to that given by PCA (without the orthogonality requirement)

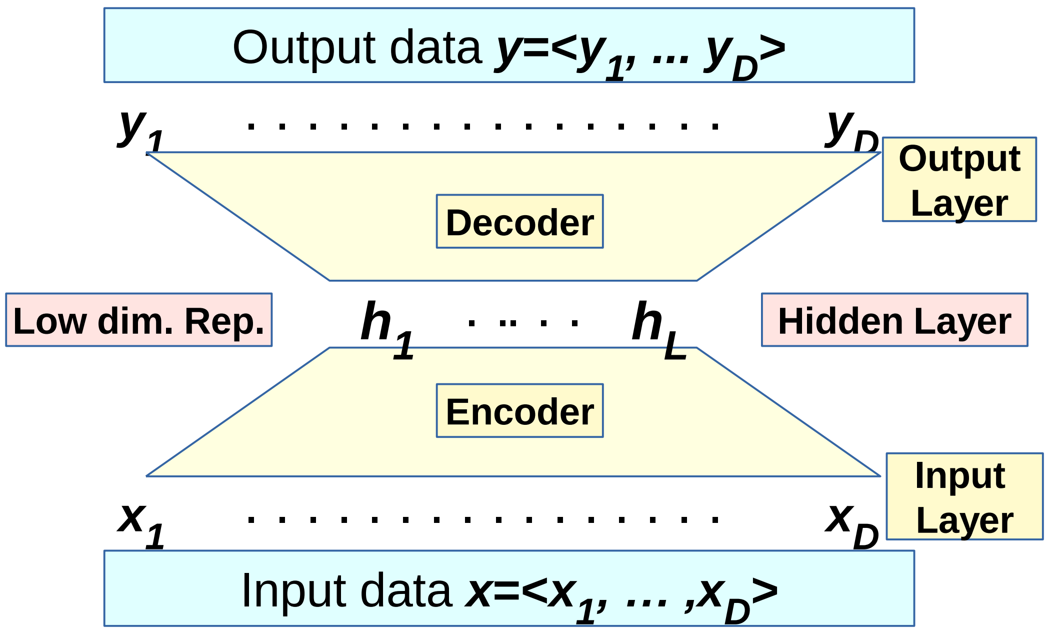
Example





- Source https://gertjanvandenburg.com/blog/autoencoder/

Denoising Autoencoder

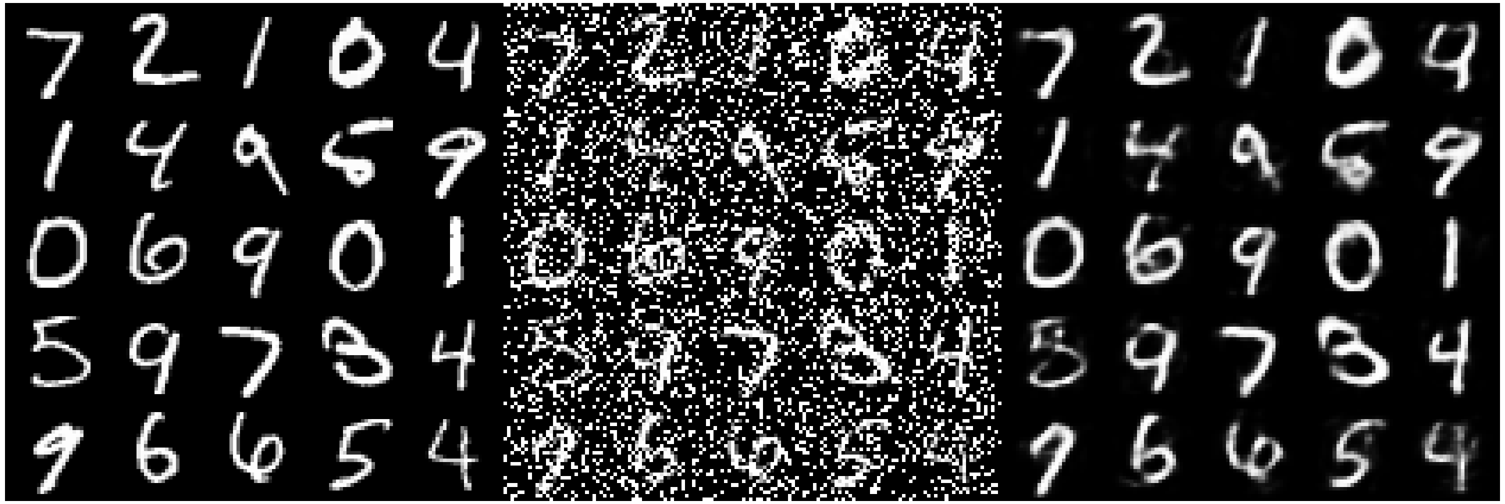


- In a denoising autoencoder the input is ***corrupted*** by adding Gaussian noise.
- The decoder is trained to reconstruct the original uncorrupted input.

$$L(x+\epsilon,\theta) = \|y-x\|^2 = \|f(x,\theta)-x\|^2$$

- The denoising autoencoder finds applications in image/video restoration (e.g WW1 movies)
- It can also be used to make neural networks more robust to noisy input data

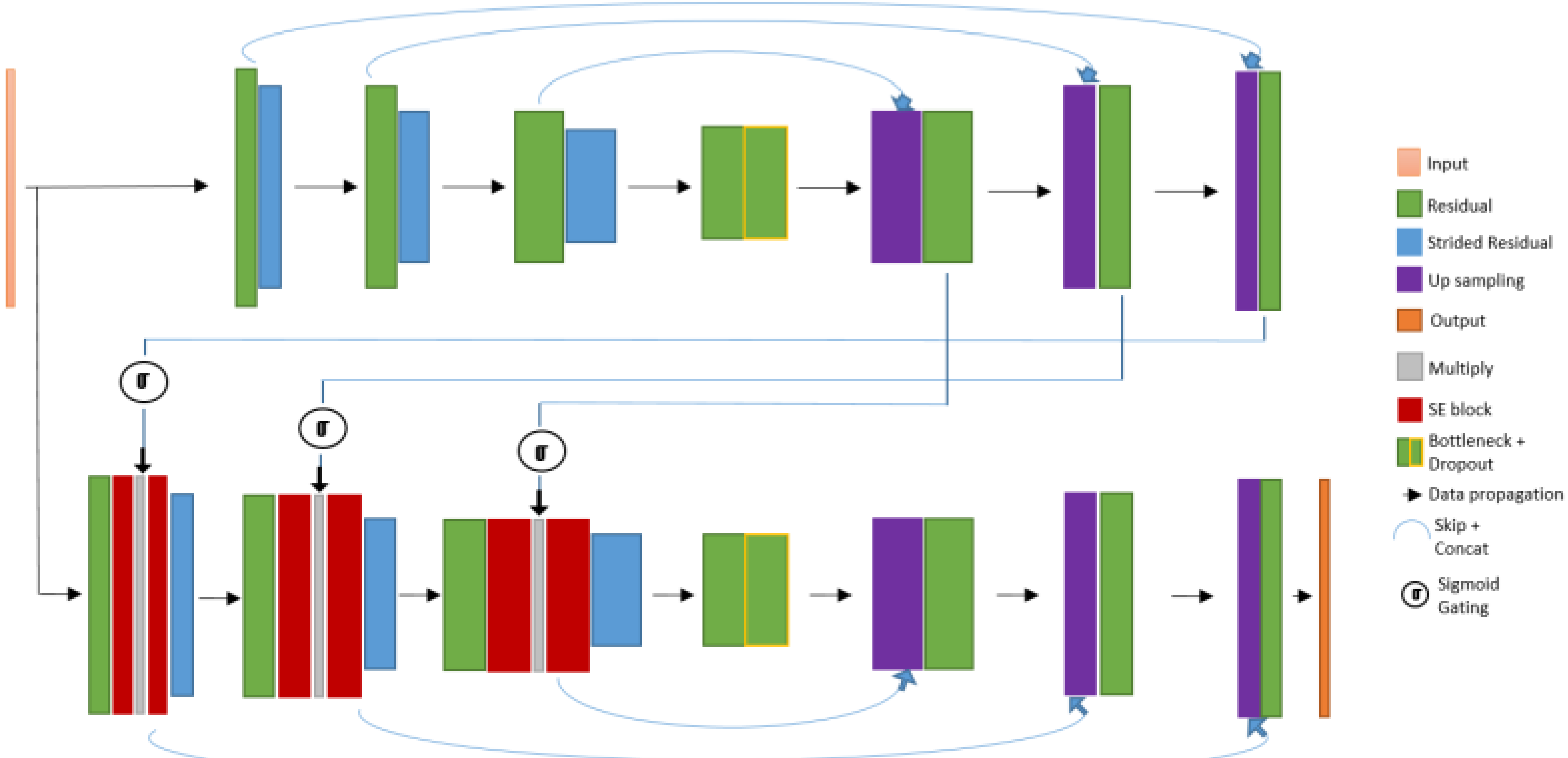
Denoising Autoencoder Example



- *Original*
- *Corrupted Input*
- *Reconstructed*

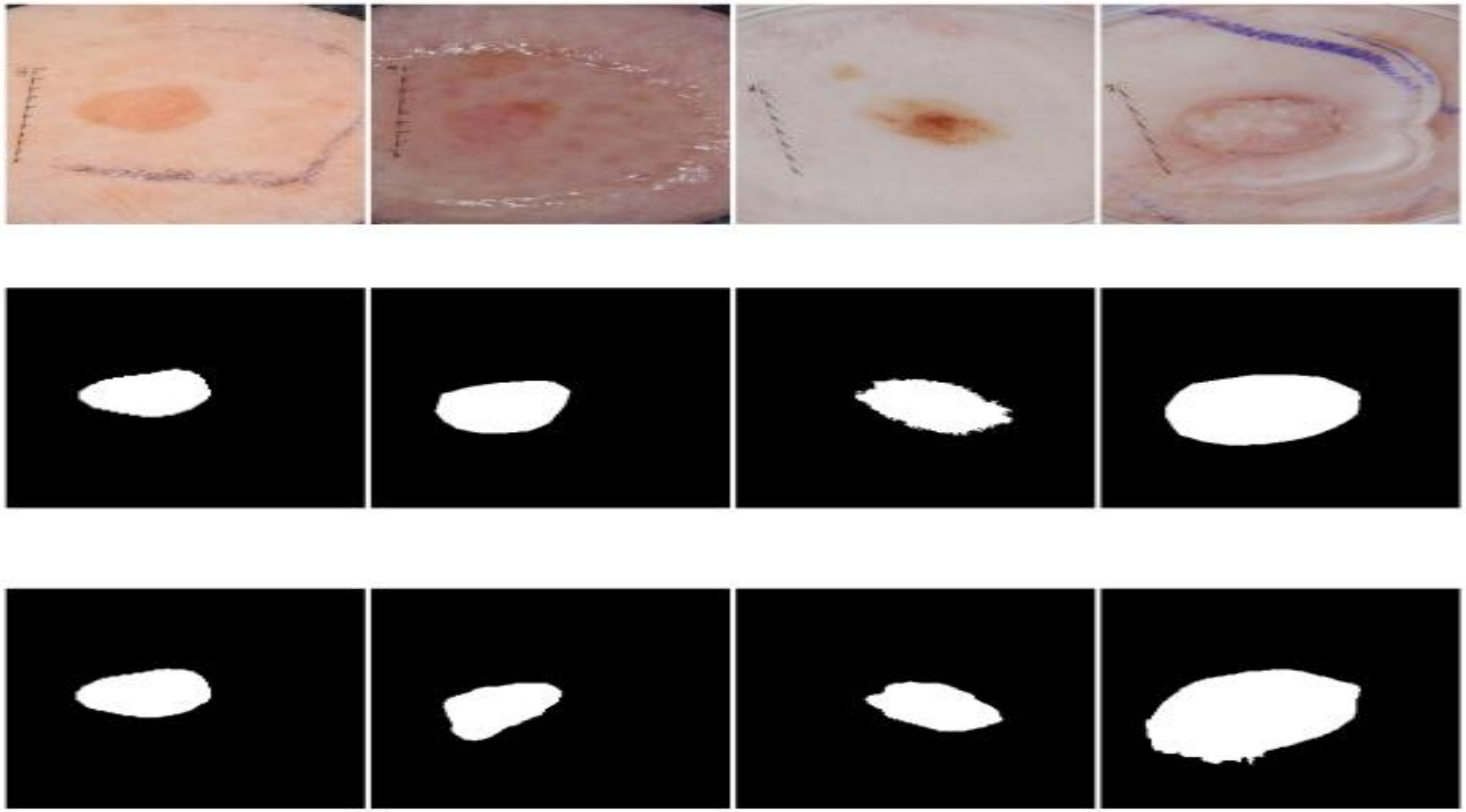
- Source: *opendeep.org*

FocusNet Architecture

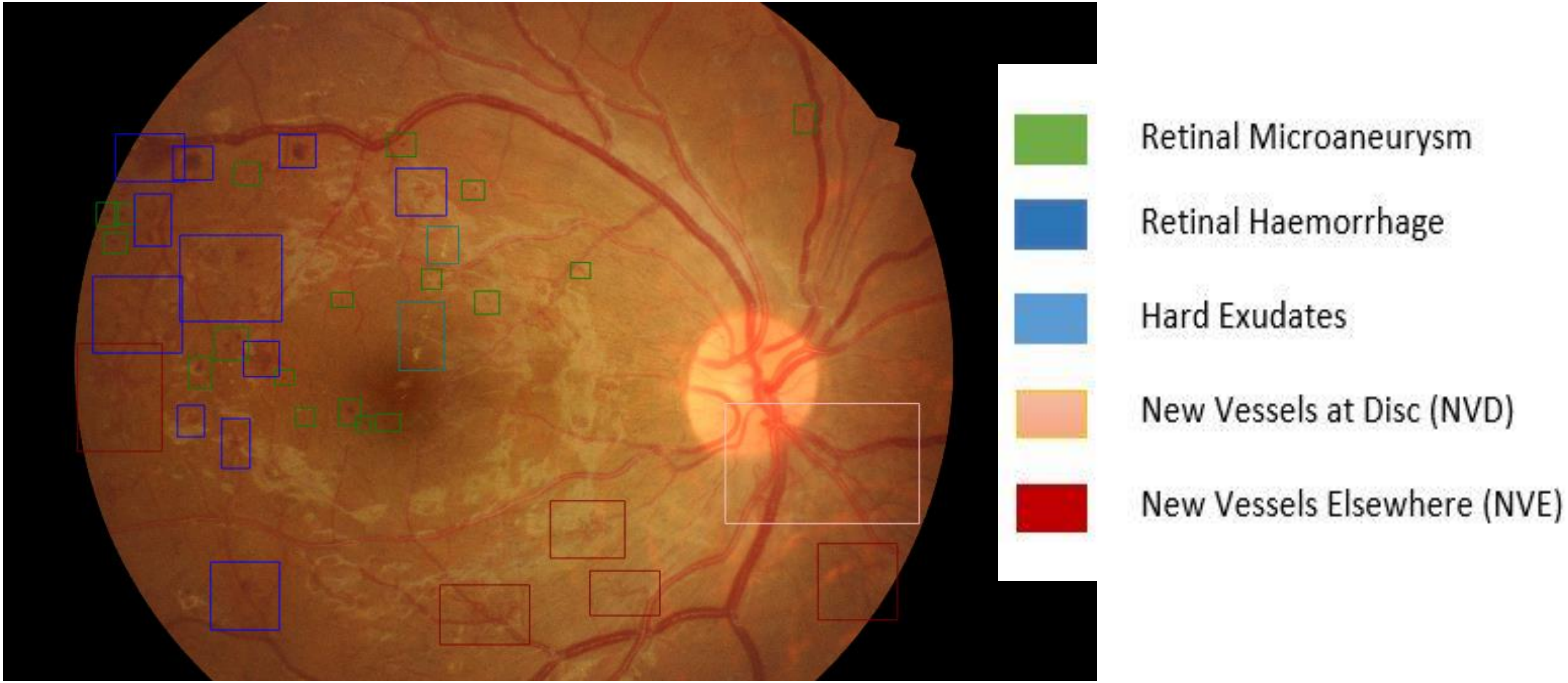


Chaitanya Kaul, Suresh Manandhar, Nick Pears. FocusNet: An attention based fully convolutional network for medical image segmentation. ISBI 2019

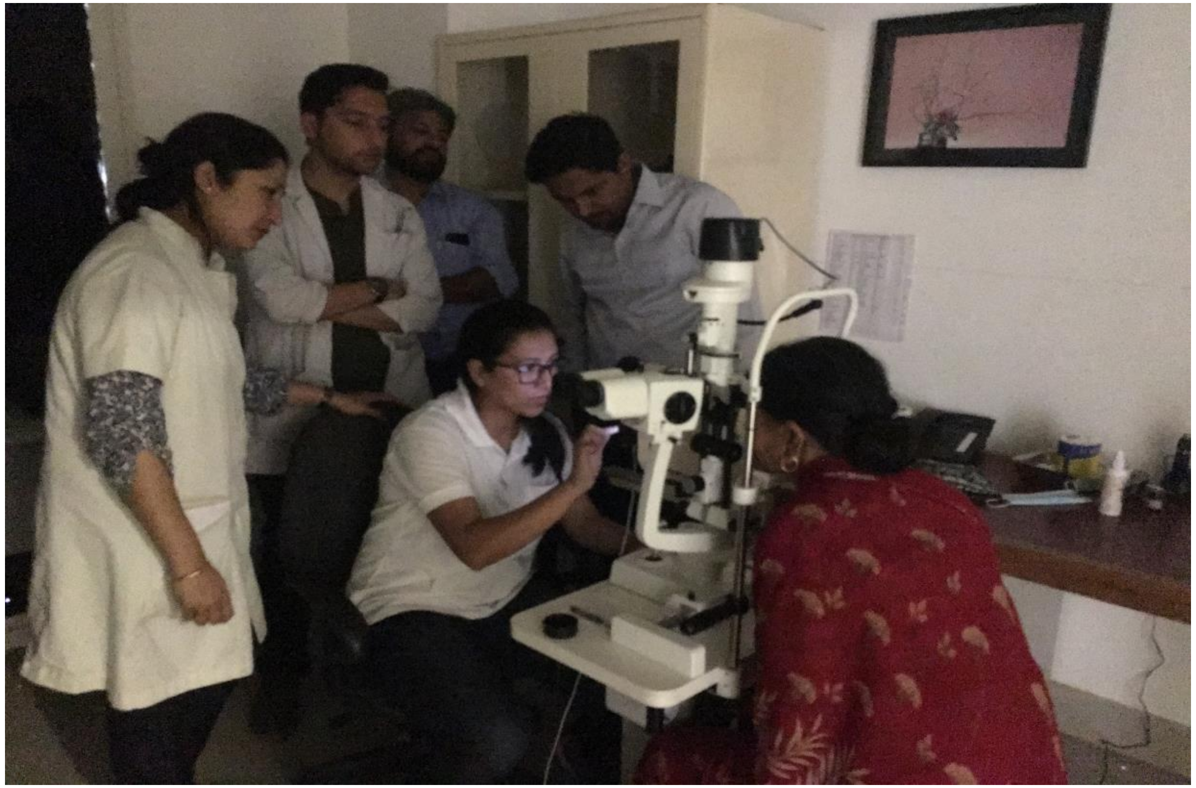
Melanoma segmentation



Diabetic retinopathy screening



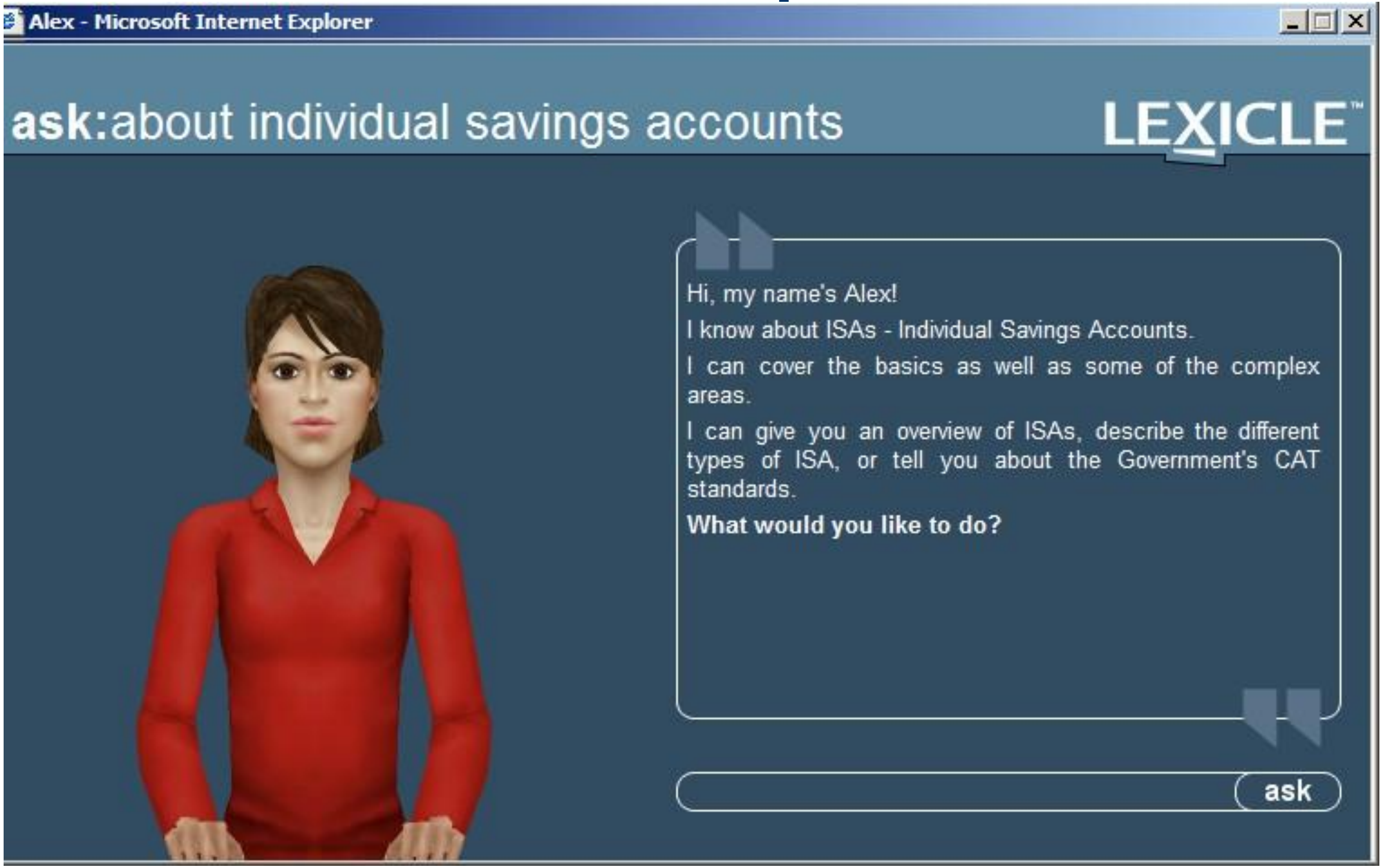
Diabetic retinopathy screening



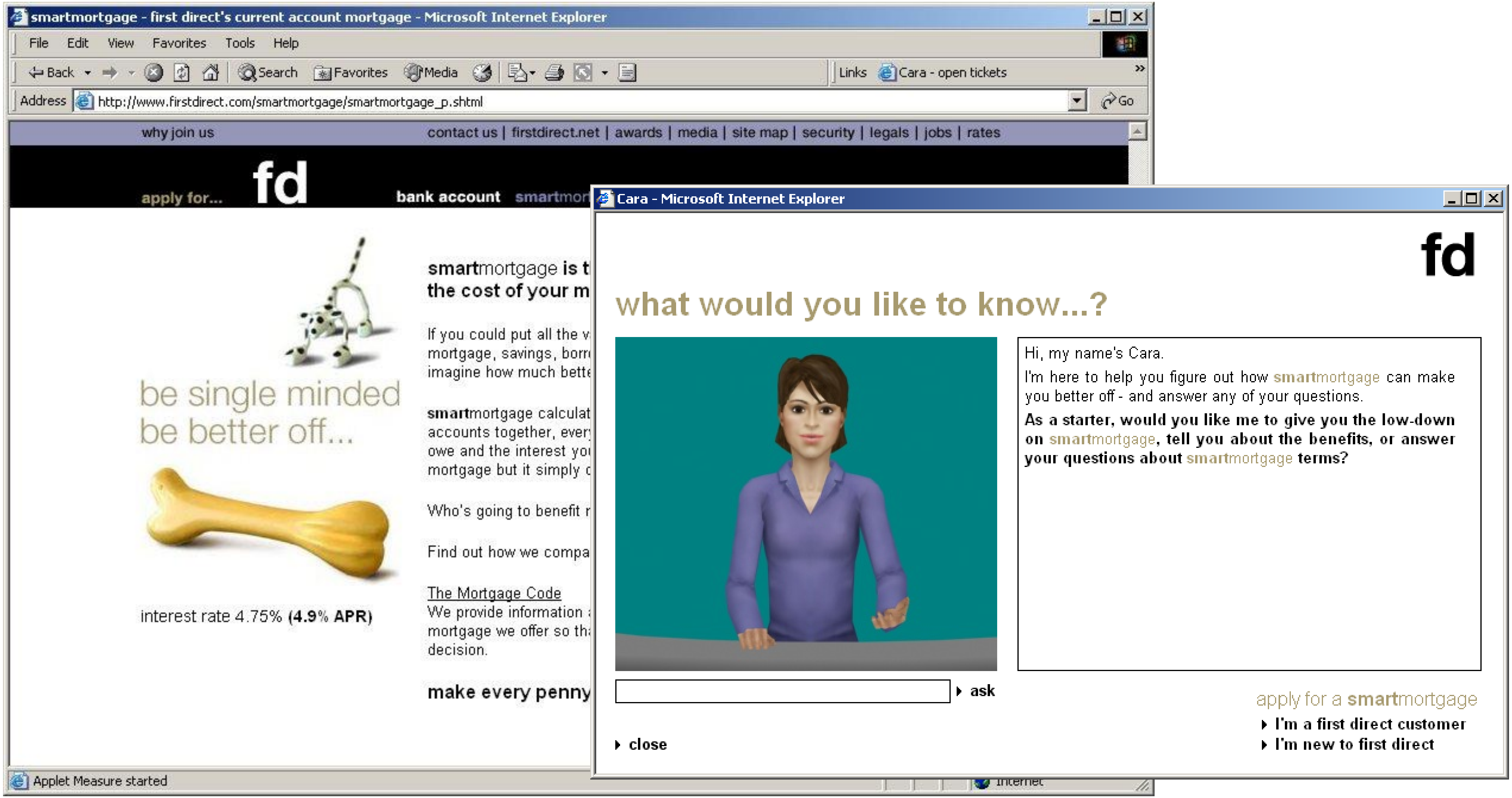
- Mobile phone based non mydriatic camera
- Phone based deployment of deep learning solution

Natural Language Processing

Virtual assistant example



Source: Lexicle Ltd UK



Source: Lexicle Ltd UK

smartmortgage - first direct's current account mortgage - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Search Favorites Media Links Cara - open tickets

Address http://www.firstdirect.com/smartmortgage/smartmortgage_p.shtml Go

be single minded
be better off...

interest rate 4.75% **(4.9% APR)**

smartmortgage **is the simple way to reduce the cost of your mortgage**

If you could put all the value of your money together - mortgage, savings, borrowing and current account - imagine how much better off you'd be.

smartmortgage calculates all the interest on your accounts together, everyday, to reduce the amount you owe and the interest you pay. It's a straightforward mortgage but it simply costs you less.

Who's going to benefit most from paying less? You are.

Find out how we compare against our competitors.

The Mortgage Code
We provide information about the different types of mortgage we offer so that you can make an informed decision.

make every penny count. everyday.

smartmortgage
- the benefits
- how it works
- moving mortgages
- faqs
- ask Cara
- in detail

ask Cara

use our calculators
- **how much could I save**
- **how much can I borrow**

get smartmortgage
- **apply**

Applet Measure started Internet

Virtual assistant example

- **Ultimate goal** -- build machines that can "***understand***" human language (without human supervision)
- Current progress:
  - Large scale robust but **shallow understanding of text**
  - Large scale unsupervised learning using automatically labelled data (self supervision)
  - Mapping between knowledge graphs and natural language
  - Question answering from text and knowledge graphs
  - Dialogue systems querying knowledge graphs and text databases (e.g. Wikipedia)
  - Large scale robust (minimally supervised) machine translation

Distributional Semantics

- **Distributional Semantics: (Harris, 1954)**

  Words that occur in similar contexts tend to have similar meanings i.e. the meaning of a word can be defined in terms of its context.

- **Word Space Model (aka Vector Space Model):**

  Meaning of a word can be represented as a co-occurrence vector built from a corpus

Distributional Semantics

- **Example:**

  Freddy is planning to buy a **house** near the city centre

  All the students are having a party in Freddy’s **house** in the city centre

	planning	buy	near	city	centre	Freddy	party	having
house	1	1	1	2	2	1	1	1

Distributional Semantics

Distributional Hypothesis (Harris, 1954)

Words that occur in similar contexts tend to have similar meanings i.e. meaning of a word can be defined in terms of its context.

Word Space Model (WSM)

Meaning of a word is represented as a co-occurrence vector built from a corpus

[I went to buy an] apartment [but the price was high] **(5 word context)**

	vector dimensions					
	animal	buy	apartment	price	rent	kill
House	⟨ 30	60	90	55	45	10 ⟩
Hunting	⟨ 90	15	12	20	33	90 ⟩

Instead of using counts we can use other measures

- **Conditional probability**

$$p(y|x) = \frac{p(y,x)}{p(x)} = \frac{\#(y,x)}{N}\frac{N}{\#(x)} = \frac{\#(y,x)}{\#(x)}$$

- Conditional probability gives a measure of directional/asymmetric association
- For window based VSMs, frequent words will have a detrimental effect i.e. if y is frequent
- **Pointwise mutual information** (**PMI**) is a symmetric measure

$$pmi(x,y) = \log\left(\frac{p(x,y)}{p(x)p(y)}\right) = log\left(\frac{\#(x,y)}{N}\frac{N}{\#(x)}\frac{N}{\#(y)}\right) = log\left(\frac{\#(x,y)}{\#(x)\#(y)}N\right)$$

- Insensitive to frequent words but can give negative values
- **Positively shifted PMI** (**PPMI**) gives smoothed positive values:

$$ppmi(x,y) = \log\left(1 + \frac{p(x,y)}{p(x)p(y)}\right)$$

Vector Space Model (VSM) for words

So, we could represent the meaning of a word as a very long vector:

Vocabulary = < animal, buy, apartment, price, rent, kill, …. >

(all words in dict)

House = < 0.1, 0.2, 0.3, 0.16, ….. >

Hunting = < 0.3, 0.07, 0.05, 0.02, ….. >

Apartment = ??

Vector Space Model (VSM) for words

So, we could represent the meaning of a word as a very long vector:

Vocabulary = < animal, buy, apartment, price, rent, kill, …. >

(all words in dict)

House = < 0.1, 0.2, 0.3, 0.16, ….. >

Hunting = < 0.3, 0.07, 0.05, 0.02, ….. >

Which one is more likely?

Apartment = < 0.1, 0.18, 0.32, 0.10, ….. > **---- 1**

Apartment = < 0.31, 0.1, 0.07, 0.05, ….. > **---- 2**

Vector Space Model (VSM) for words

So, we could represent the meaning of a word as a very long vector:

Vocabulary = < animal, buy, apartment, price, rent, kill, …. >

(all words in dict)

House = < 0.1, 0.2, 0.3, 0.16, ….. >

Hunting = < 0.3, 0.07, 0.05, 0.02, ….. >

Given the distributional hypothesis we expect that it is more likely:

Apartment = < 0.1, 0.18, 0.32, 0.10, ….. > **---- 1**

VSM as a meaning representation in vector space

- The VSM is an explicit representation that is high dimensional
  (~ vocabulary size > 30,000)
- It is also very sparse (with most entries 0). **Why**?

Computing Similarity in meaning between two words

- VSMs can recover the similarity in meaning between words e.g. using cosine similarity or KL/JS divergence

- Thus, we expect $\boldsymbol{cos(book, novel)}$ to be high

$$\boldsymbol{cos(A, B) = \frac{A.B}{|A||B|}}$$

Issues with VSMs

- However, VSMs suffer from sparsity issues and generalise poorly
- **Why**?

Issues with VSMs

- However, VSMs suffer from sparsity issues and generalise poorly
- **Why**?
- To fill all the elements of a high dimensional vector, you will need a huge amount of data.
- So, using VSMs is not an ideal solution.
- **What would be a better solution?**

Issues with VSMs

- However, VSMs suffer from sparsity issues and generalise poorly
- **Why**?
- To fill all the elements of a high dimensional vector, you will need a huge amount of data.
- So, using VSMs is not an ideal solution.
- **What would be a better solution?**
- Ideally would want a lower dimensional representation
- that generalises better (i.e. can work with smaller datasets)

Word/Sentence Embeddings – General ideas

Creating training data using distributional semantics

We can set the problem of learning word/sentence meanings as a machine learning task that **requires some semantic interpretation**:

- **The dog ____ the cat (fill in the blank)**

Word/Sentence Embeddings – General ideas

Creating training data using distributional semantics

We can set the problem of learning word/sentence meanings as a machine learning task that **requires some semantic interpretation**:

- **The dog ____ the cat** **(fill in the blank)**
- **I went to the party wearing a nice _____** **(predict the next word)**

Word/Sentence Embeddings – General ideas

Creating training data using distributional semantics

We can set the problem of learning word/sentence meanings as a machine learning task that **requires some semantic interpretation**:

- **The dog ___ the cat** **(fill in the blank)**
- **I went to the party wearing a nice _____** **(predict the next word)**
- **My neighbours have a dog that is quite scary** **(predict left/right context word)**

Word/Sentence Embeddings – General ideas

Creating training data using distributional semantics

We can set the problem of learning word/sentence meanings as a machine learning task that **requires some semantic interpretation**:

- **The dog ___ the cat** **(fill in the blank)**
- **I went to the party wearing a nice ____** **(predict the next word)**
- **My neighbours have a dog that is quite scary** **(predict left/right context word)**
- **I heated the food → The food got hot** **(entails/contradicts/unrelated)**

Word/Sentence Embeddings – General ideas

Creating training data using distributional semantics

- We can set the problem of learning word/sentence meanings as a machine learning task that requires some semantic interpretation:
  - The dog ___ the cat **(fill in the blank)**
- For each of the tasks we can generate a training dataset containing the correct and incorrect predictions.
- For example, for the **fill in the blank** task we can create training data:
  - [the, dog] [the, cat] → chases **(+ example)** should give class ***1***
  - [the, dog] [the, cat] → bites **(+ example)** should give class ***1***
  - [the, dog] [the, cat] → buy **(- example)** should give class ***0***
- Think of the → as a machine learning model that we train using this data

Classwork – create training data

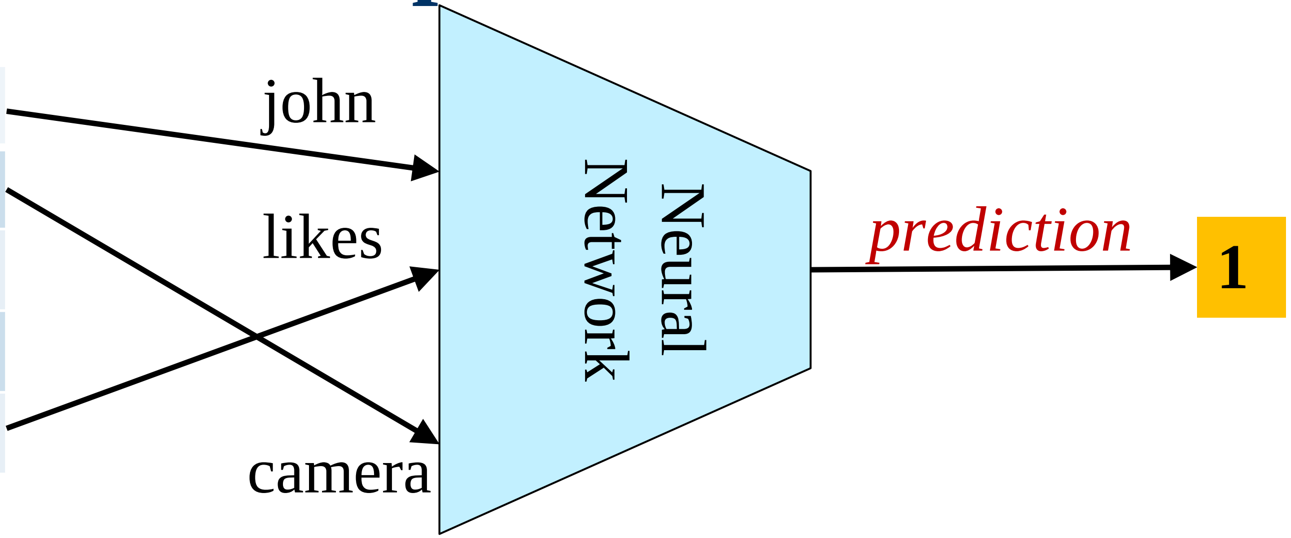
- For the left/right context prediction task:

  My neighbours have a dog that is quite scary (predict left/right context word)

- Create the training data:

Word Embeddings – The setup

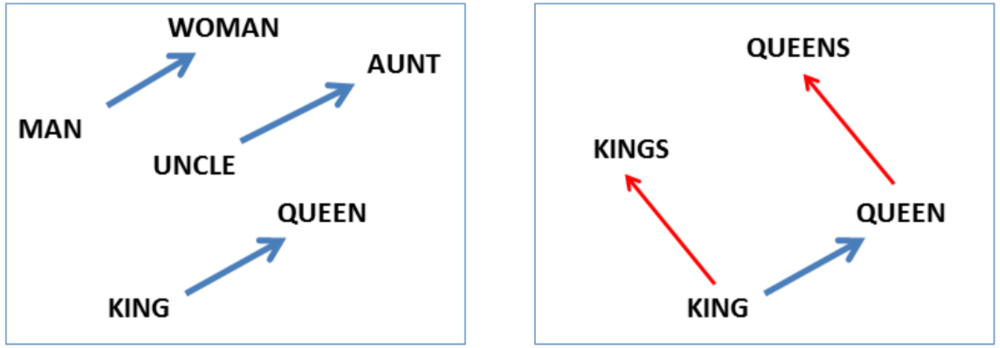
john	0.0012	0.0025	….
camera	0.20	…	
……	…		
……			
likes	0.01	0.001	



- Transfer learning using pre-trained embeddings (e.g. word2vec, GloVe)
- Domain specific learning
- Combination

Analogy tasks

- Analogy between words:
  - woman − man ≈ queen − king
  - king − man + woman ≈ queen

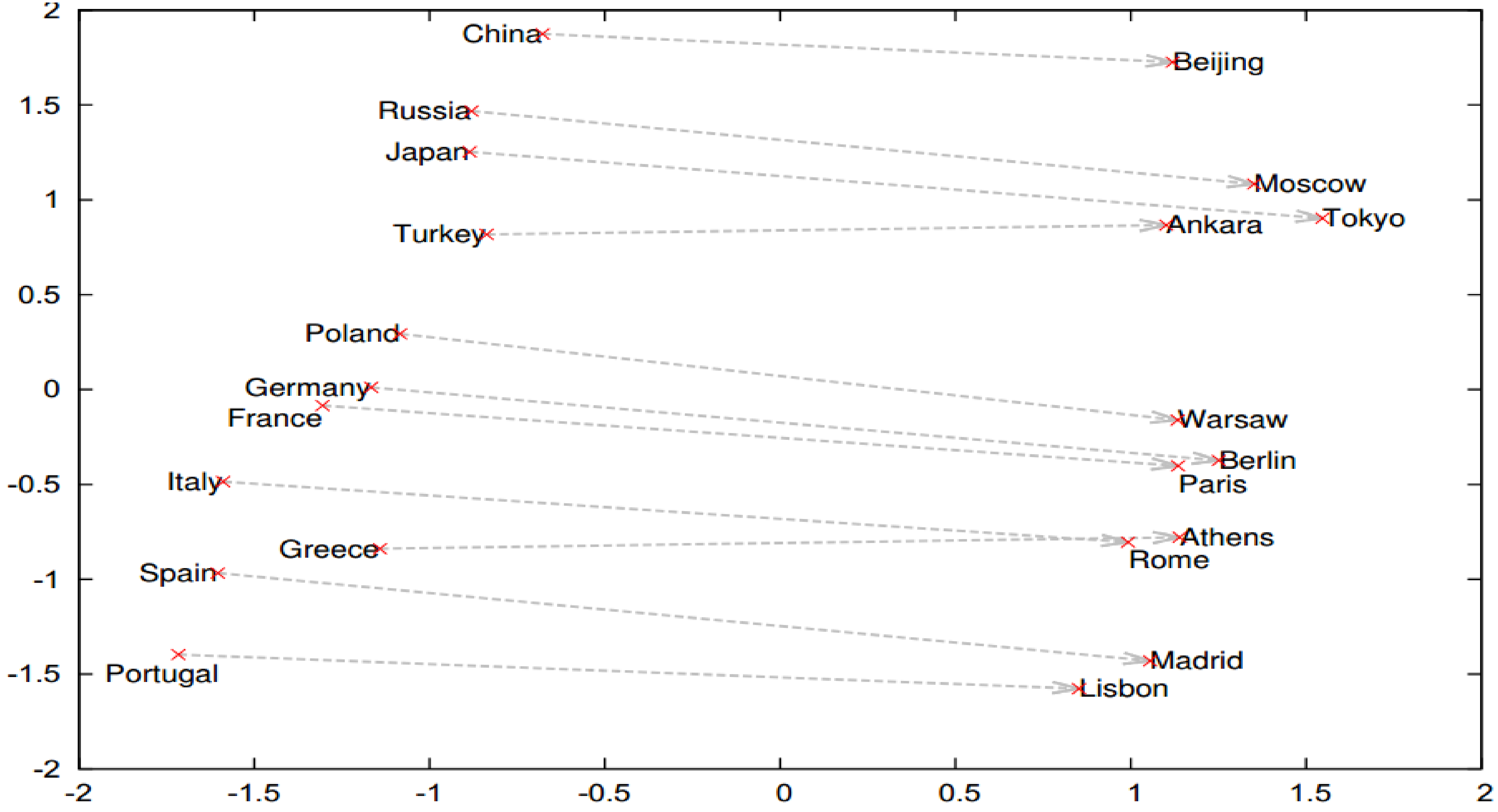


- **England** − London + Baghdad = ? Iraq
- Equivalently:

$$\underset{B'}{\arg\max}\, \cos(B', \text{England} - \text{London} + \text{Baghdad})$$

Slide from Omer Levy

Directional Similarity



Sentence level classification tasks in NLP

- **Sentiment analysis** (positive, negative, neutral etc.)

Example: The food was fine but the décor was unimpressive

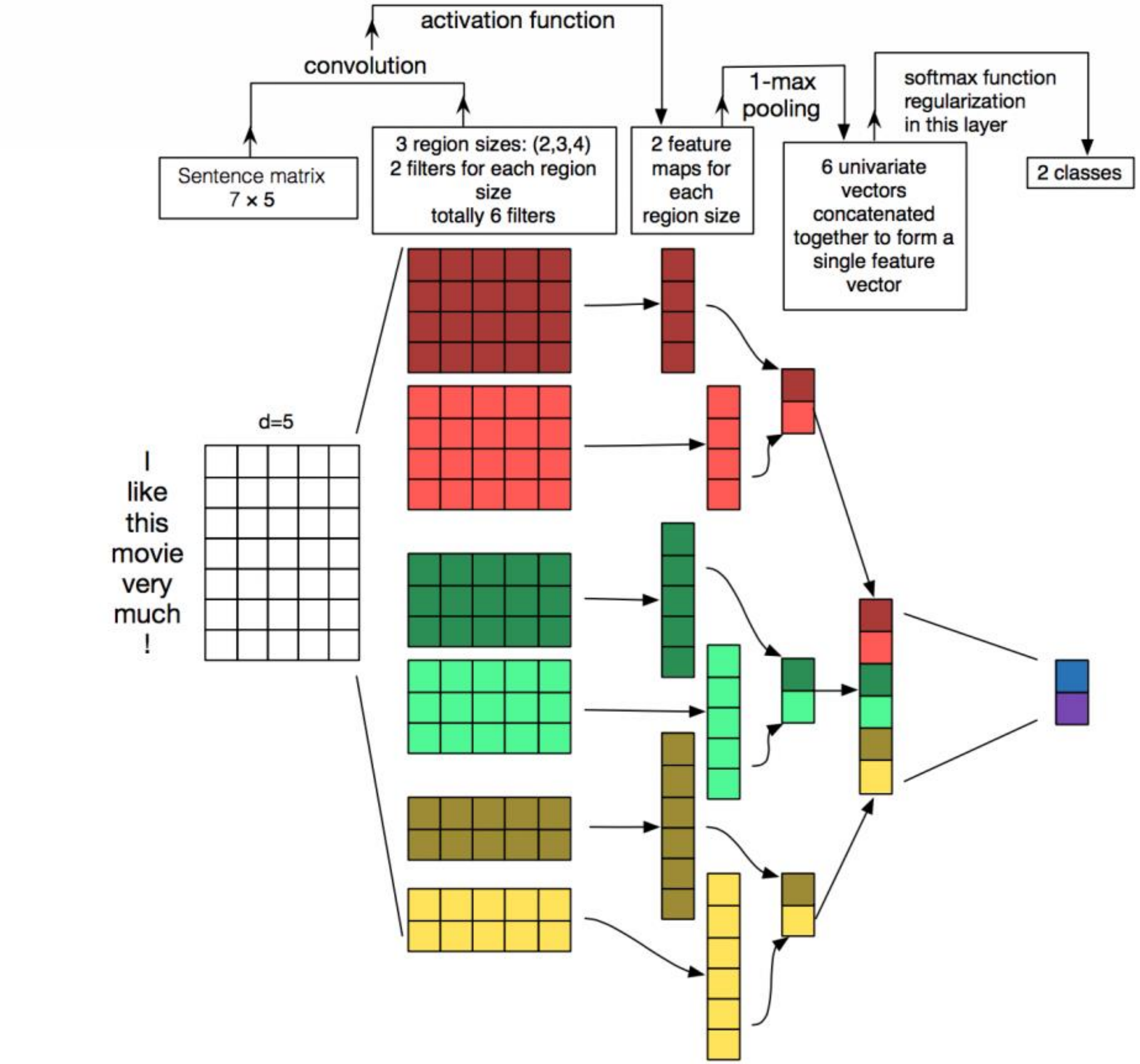
- **Subjectivity classification** (subjective vs objective)

Example: The match today was bit boring

- **Question type classification** (who i.e. person, where i.e. location, which restaurant → restaurant)

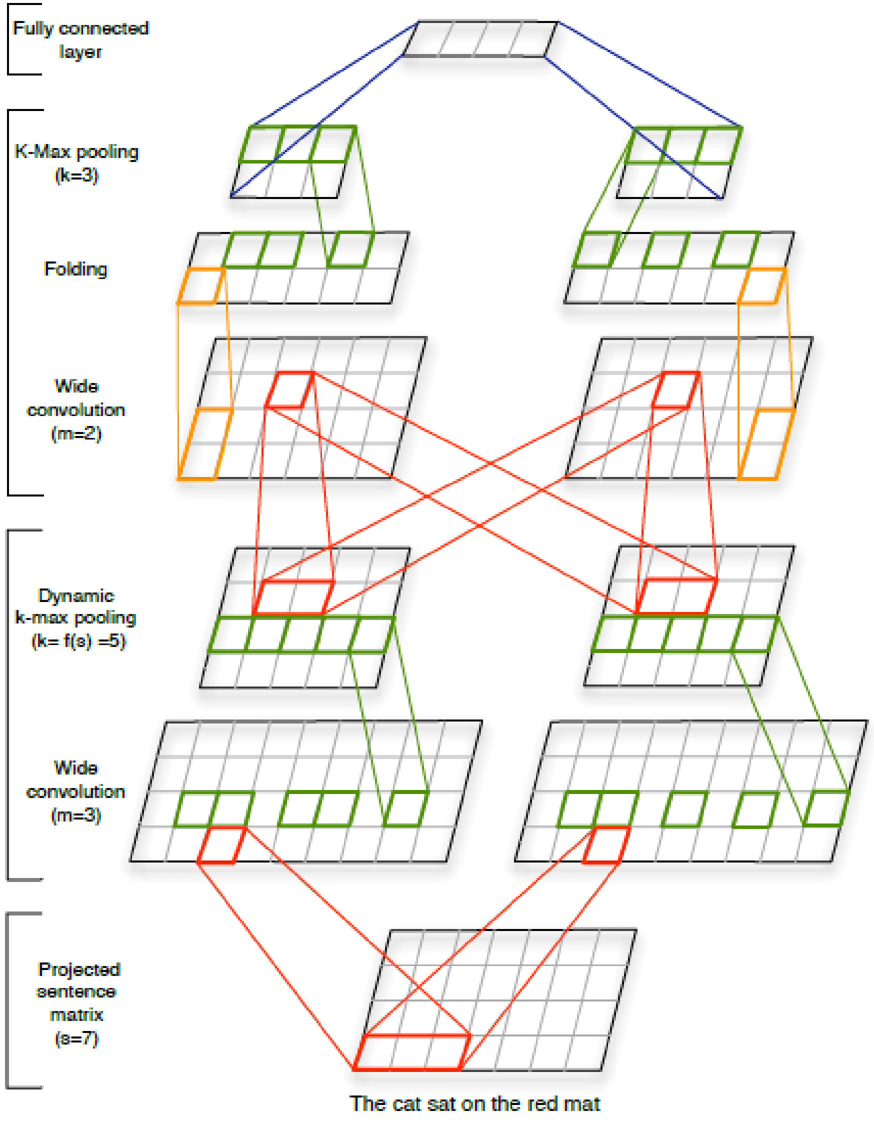
Example: Which hotel is near to the city centre?

CNN models for sentence classification



Source: Zhang, Y., & Wallace, B. (2015). A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification.

CNN models for sentence classification



Source: Kalchbrenner N., Grefenstette E., Blunsom P. A Convolutional Neural Network for Modelling Sentences.

Thank you!