CHULA ENGINEERING Foundation toward Innovation COMPUTER

Data Mining Group
Machine Intelligence and Knowledge Discovery Lab
Chulalongkorn University

Advanced Deep Learning (DL)

Prof. Peerapon Vateekul, Ph.D.
Department of Computer Engineering,
Faculty of Engineering, Chulalongkorn University
Peerapon.v@chula.ac.th
www.cp.eng.chula.ac.th/~peerapon/

Outline

- Introduction to Deep Learning (recap)
- Convolutional Neural Networks (CNN) [Imaging Task] (recap)
- Recurrent Neural Networks (RNN) [Time Series & NLP Tasks]
- Transformer models
- Hugging Face
- Note: Today's lecture introduces several DL models so that you can understand how to select the right one for your problem. You'll learn more in depth in other AI courses.

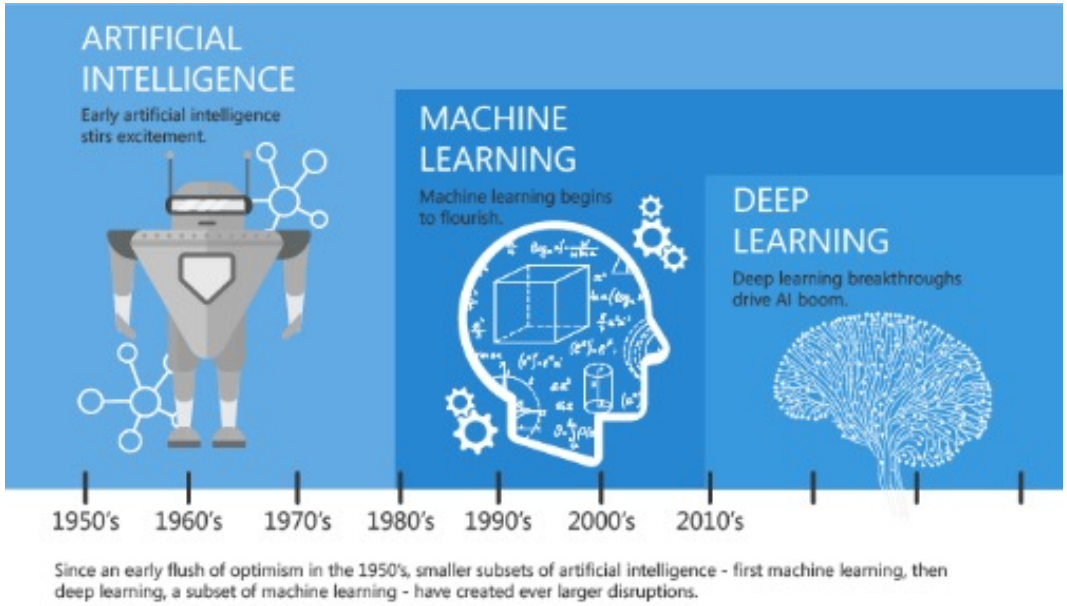
Introduction to Deep Learning (recap)

CHULA ENGINEERING COMPUTER
Foundation toward Innovation

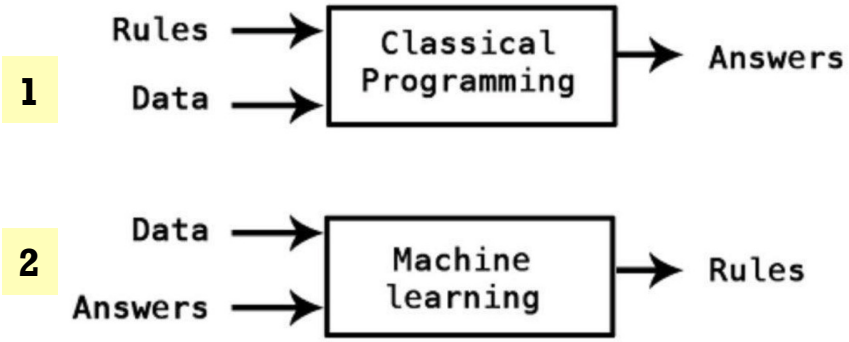
+

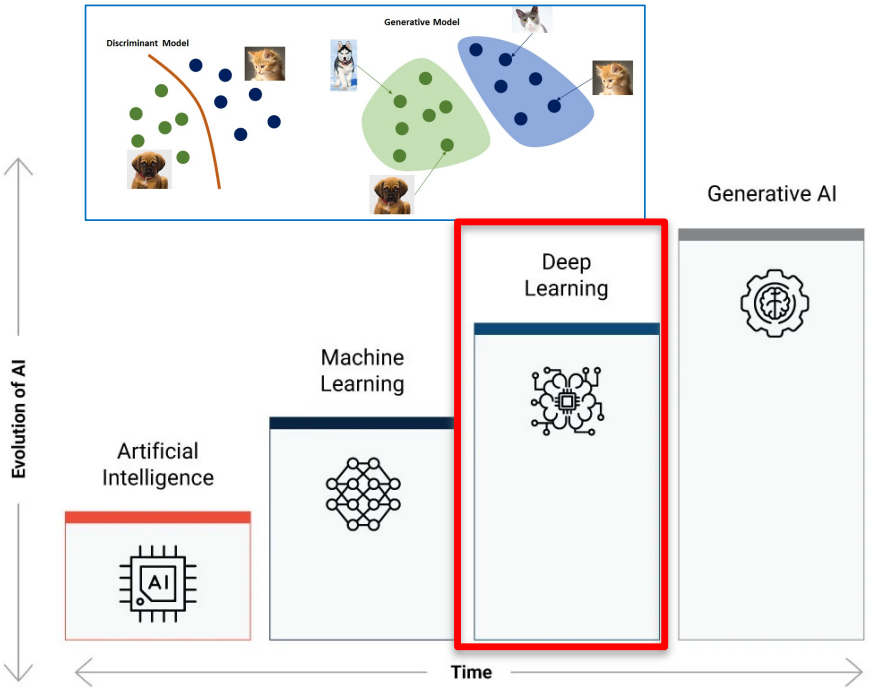
AI = Automation

- 1) Rule-based AI
- 2) Machine Learning (ML)



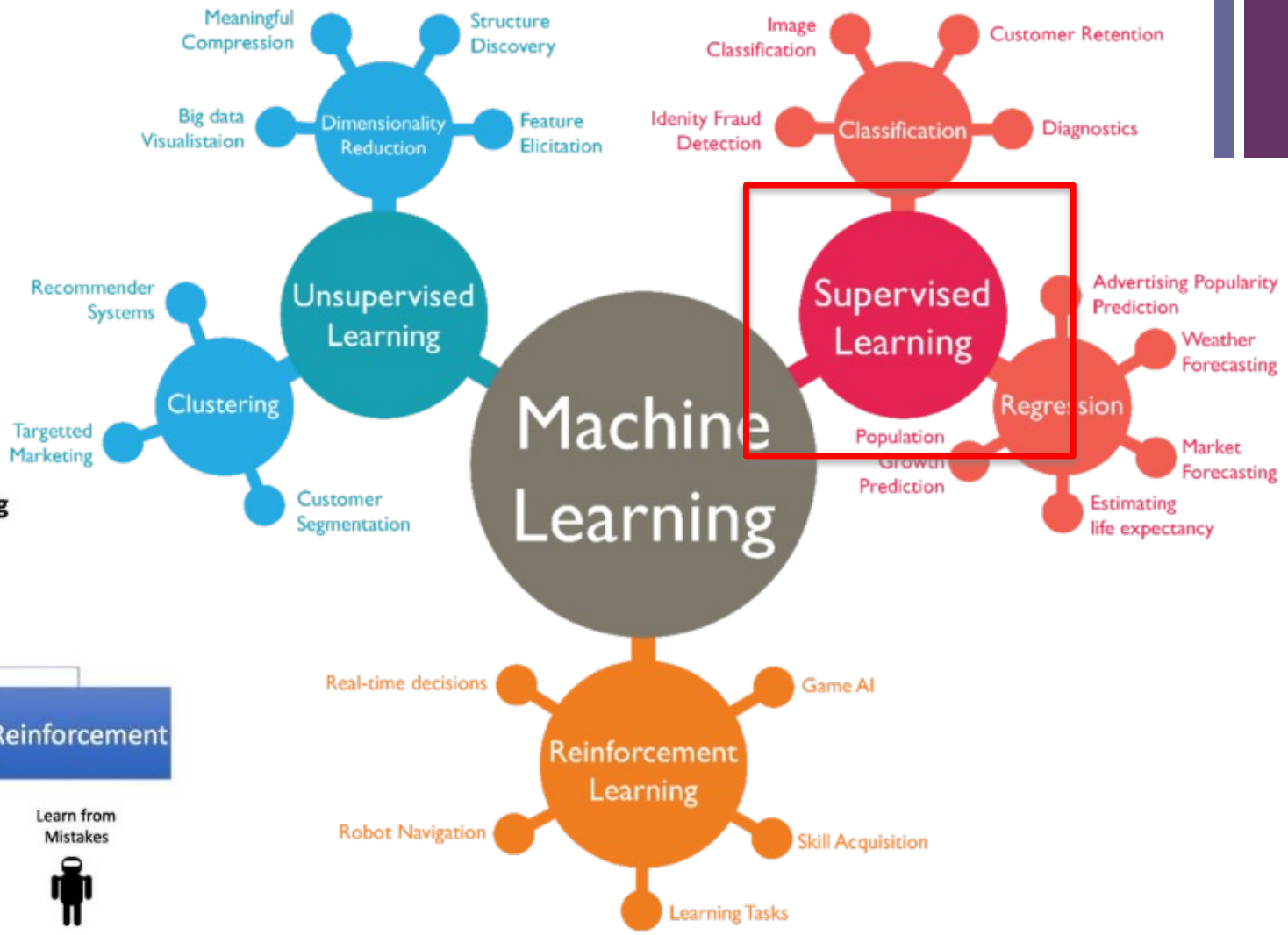
Since an early flush of optimism in the 1950's, smaller subsets of artificial intelligence - first machine learning, then deep learning, a subset of machine learning - have created ever larger disruptions.

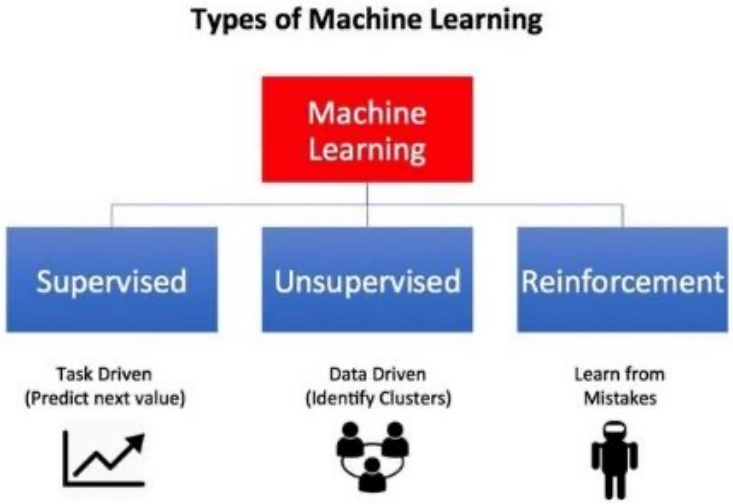




https://mc.ai/machine-learning-basics-artificial-intelligence-machine-learning-and-deep-learning/

Machine Learning (ML)





Task1: Supervised learning

Handcrafted features

Training Data

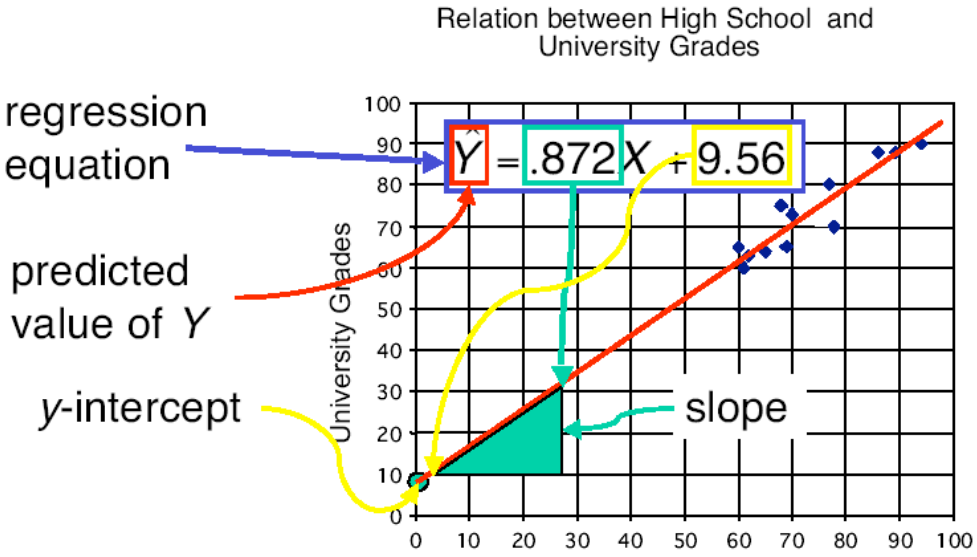
	inputs			target
Age	**Gender**	**BodyTemp**	**Cough**	**Corona**
12	Female	37	Yes	Yes
35	Female	39	No	Yes
32	Male	38	Yes	No

Testing Data

Age	**Gender**	**BodyTemp**	**Cough**	**Corona**
25	Male	40	No	?

Application: Corona Prediction

Regression – Linear Relationship



weight, coefficient

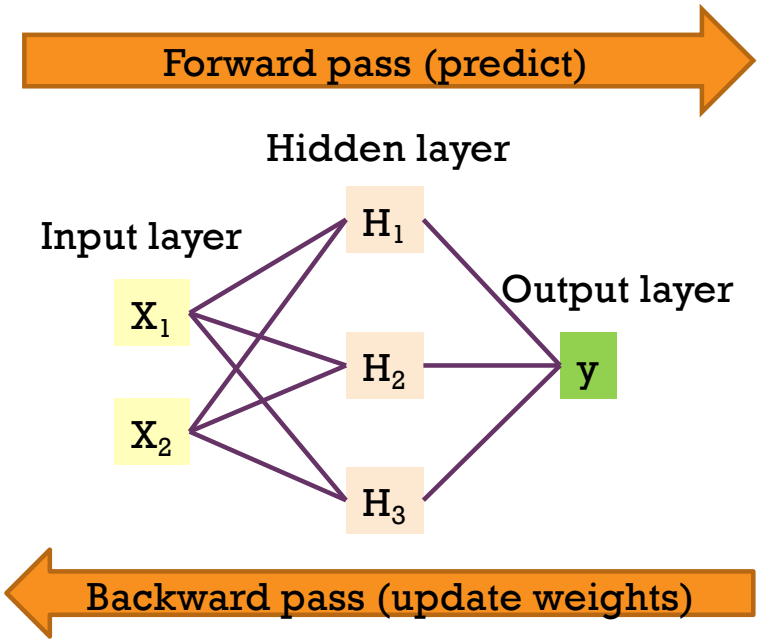
$$\underset{\text{target}}{\hat{y}} = \underset{\text{intercept}}{\hat{w}_0} + \hat{w}_1 x_1 + \hat{w}_2 \underset{\text{input}}{x_2}$$

- The least square method aims to minimize the following term

$$\sum_{\substack{training \\ data}} (y_i - \hat{y}_i)^2$$

Neural Networks (universal approximator)

Non-linear relationship



Age	Income	Gender	Province	Corona
25	25,000	Female	Bangkok	Yes
35	50,000	Female	Nontaburi	Yes
32	35,000	Male	Bangkok	No

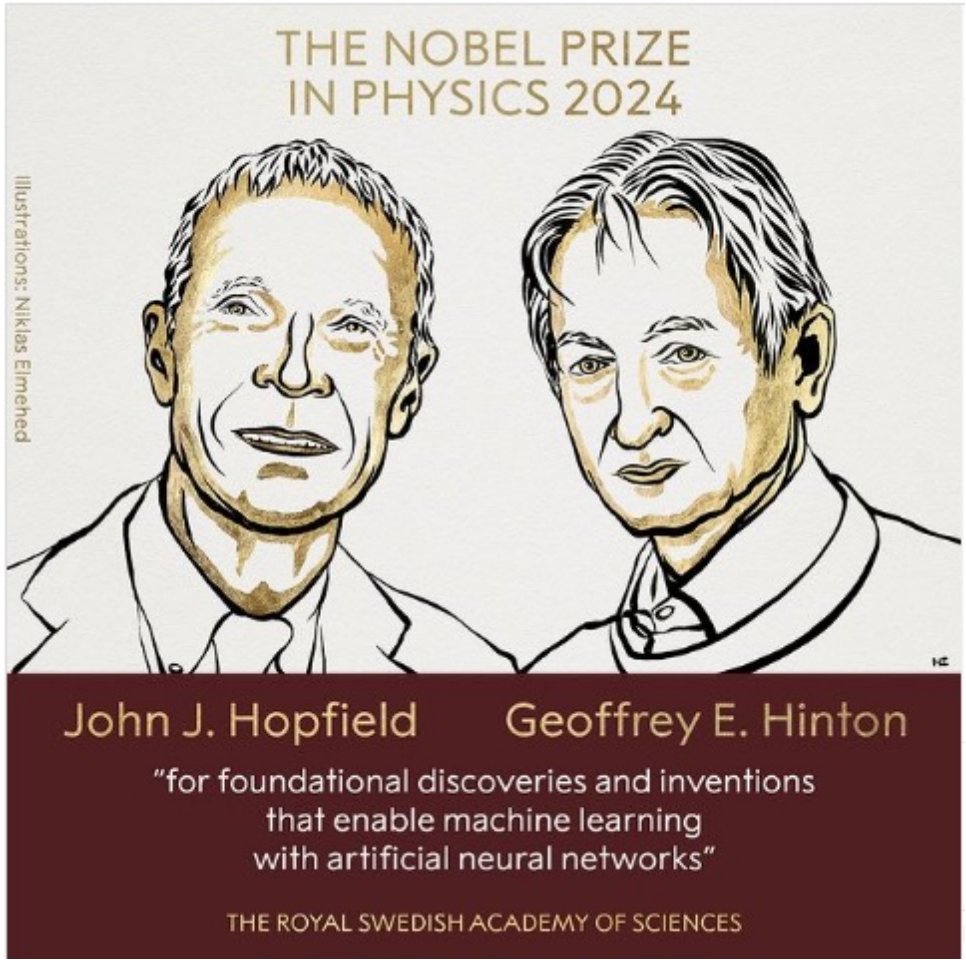
$$\hat{y} = \hat{w}_0 + \hat{w}_1 H_1 + \hat{w}_2 H_2 + \hat{w}_3 H_3$$

$$H_1 = tanh(\hat{w}_{10} + \hat{w}_{11} x_1 + \hat{w}_{12} x_2)$$

$$H_2 = tanh(\hat{w}_{20} + \hat{w}_{21} x_1 + \hat{w}_{22} x_2)$$

$$H_3 = tanh(\hat{w}_{30} + \hat{w}_{31} x_1 + \hat{w}_{32} x_2)$$

Nobel Prize to Backpropagration's Inventors

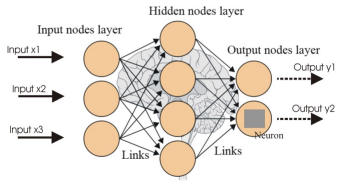


8 October 2024

The Royal Swedish Academy of Sciences has decided to award the Nobel Prize in Physics 2024 to

John J. Hopfield
Princeton University, NJ, USA

Geoffrey E. Hinton
University of Toronto, Canada



"for foundational discoveries and inventions that enable machine learning with artificial neural networks"

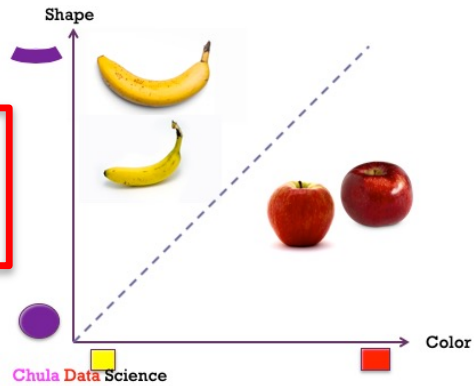
They trained artificial neural networks using physics

John Hopfield invented a network that uses a method for saving and recreating patterns. We can imagine the nodes as pixels. The *Hopfield network* utilises physics that describes a material's characteristics due to its atomic spin – a property that makes each atom a tiny magnet.

Geoffrey Hinton used the Hopfield network as the foundation for a new network that uses a different method: the *Boltzmann machine*. This can learn to recognise characteristic elements in a given type of data.

Handcrafted features

Age	Income	Gender	Province	Corona
25	25,000	Female	Bangkok	Yes
35	50,000	Female	Nontaburi	Yes
32	35,000	Male	Bangkok	No



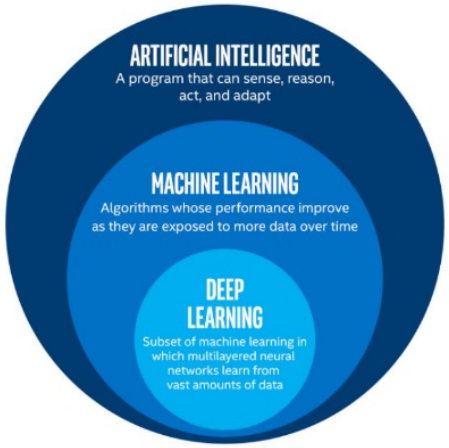
Can we still tell the features (columns)?

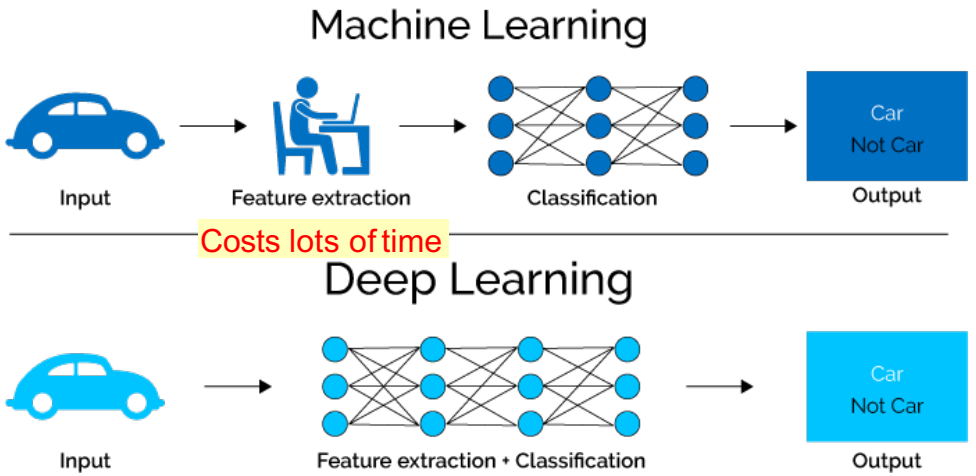
shutterstock.com · 451802557

What is Deep Learning (DL)?

Part of the machine learning field of learning representations of data. Exceptional effective at learning patterns.

Utilizes learning algorithms that derive meaning out of data by using a hierarchy of multiple layers that mimic the neural networks of our brain.



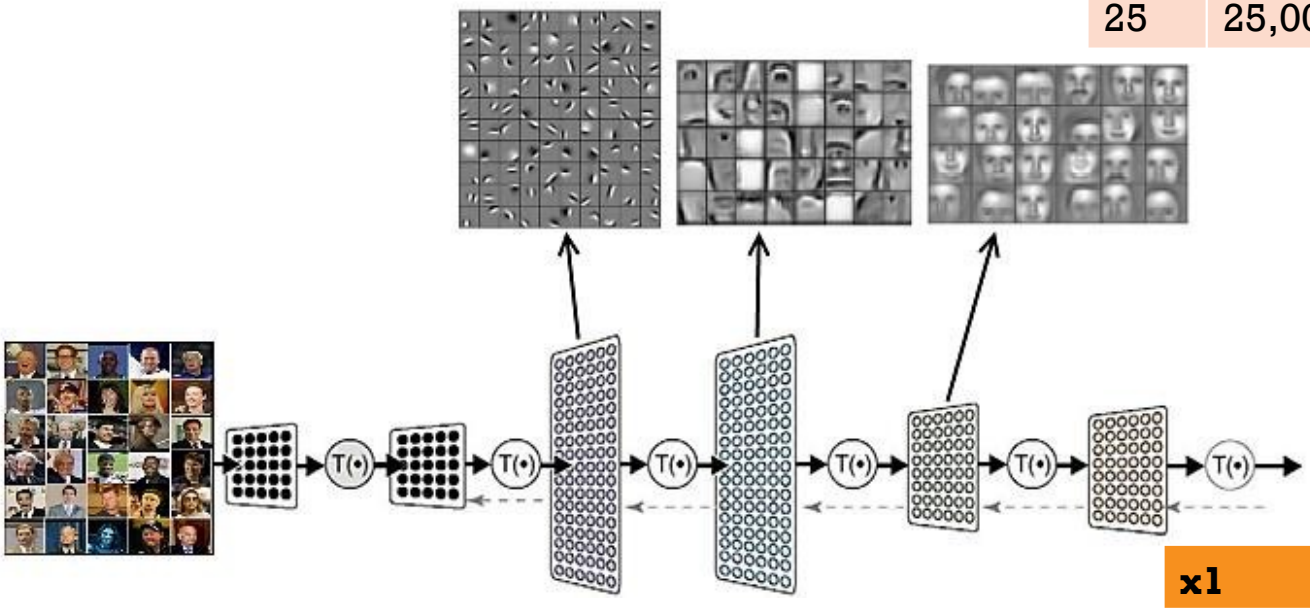


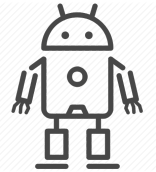
Deep Learning – Basics (cont.)

What did it learn?

A deep neural network consists of a hierarchy of layers, whereby each layer transforms the input data into more abstract representations (e.g., edge -> nose -> face). The output layer combines those features to make predictions.

Age	Income	Gender	Province	Corona
25	25,000	Female	Bangkok	Yes





x1	x2	x3	x4	Corona
0.7	0.2	-0.5	-0.1	Yes

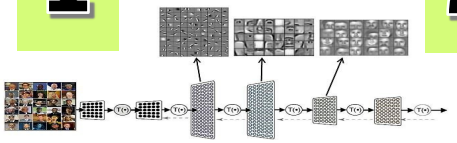
Deep Learning Application

Speech Recognition

Computer Vision

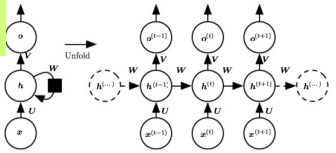
Natural Language Processing

1



CNN

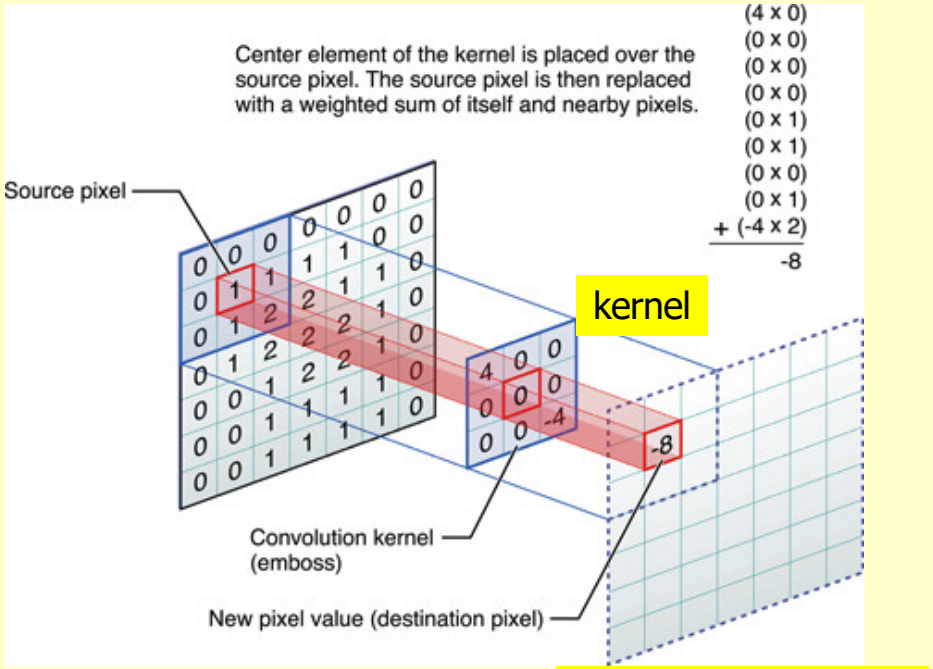
2

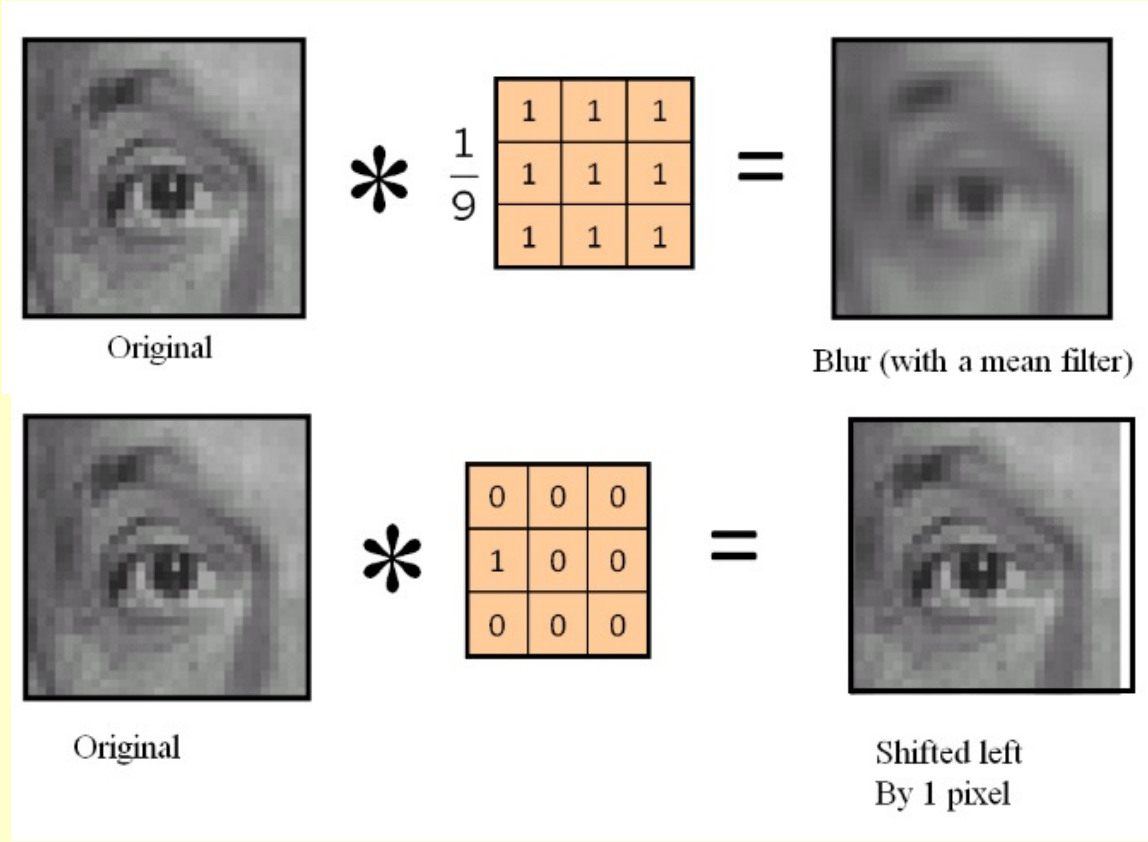


RNN

Convolutional Neural Networks (CNN) (recap)

Convolution



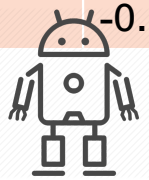


Changing the kernel (filter) results in different image processing outcomes

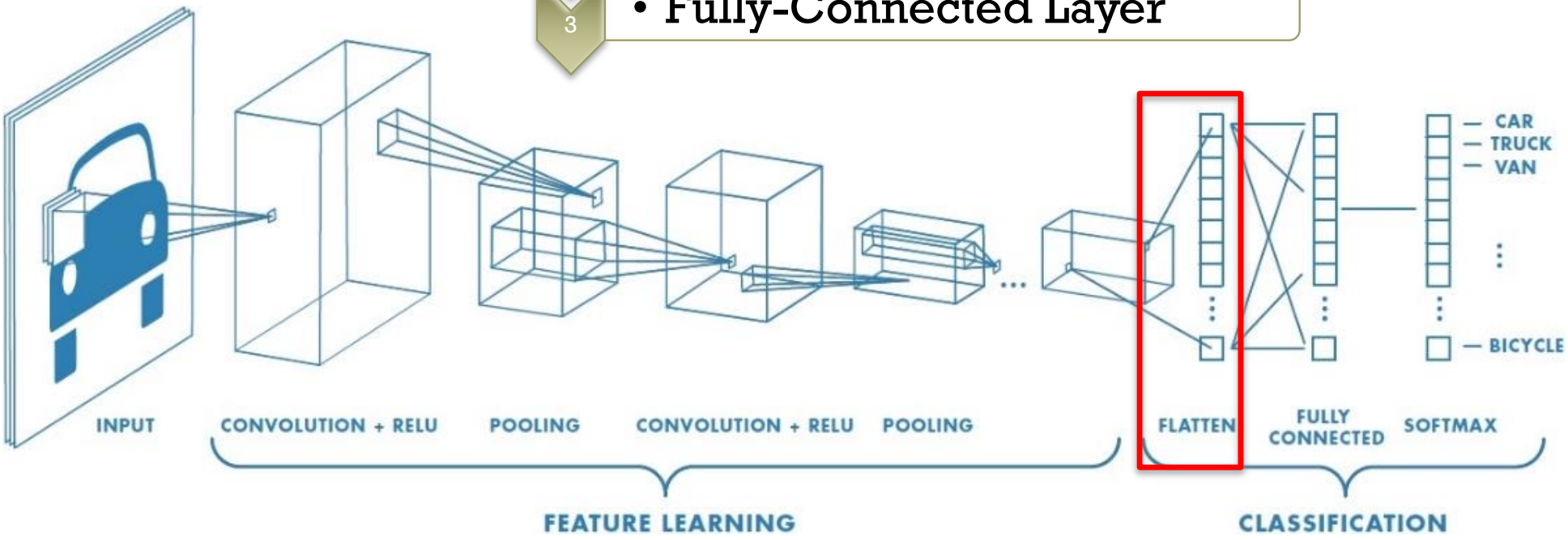
Convolutional Neural Networks

Embedding Vector

x1	x2	x3	x4	Corona
0.7	0.2	-0.5	-0.1	Yes



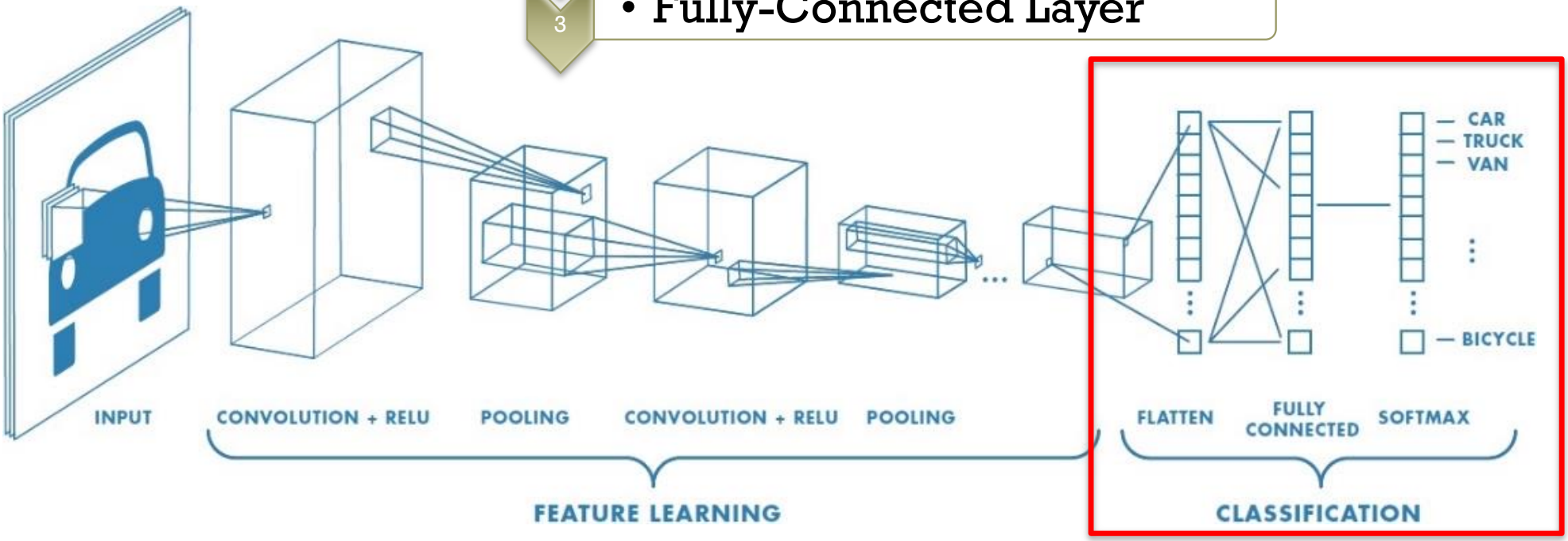
1. • Convolutional Layer
2. • Pooling Layer
3. • Fully-Connected Layer



Convolutional Neural Networks (CNN)

3) Fully-Connected Layer (FC) = Neural Networks

1. Convolutional Layer
2. Pooling Layer
3. Fully-Connected Layer



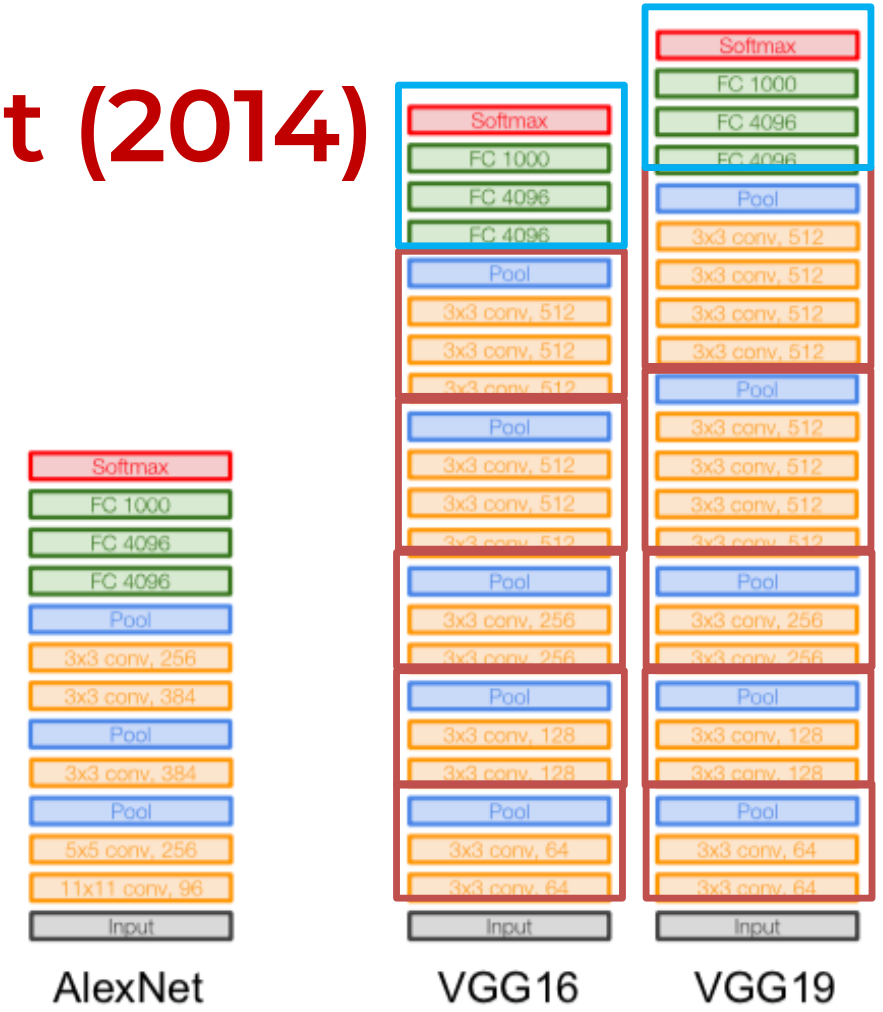
CNN Example: VGGNet (2014)

[Simonyan and Zisserman, 2014]

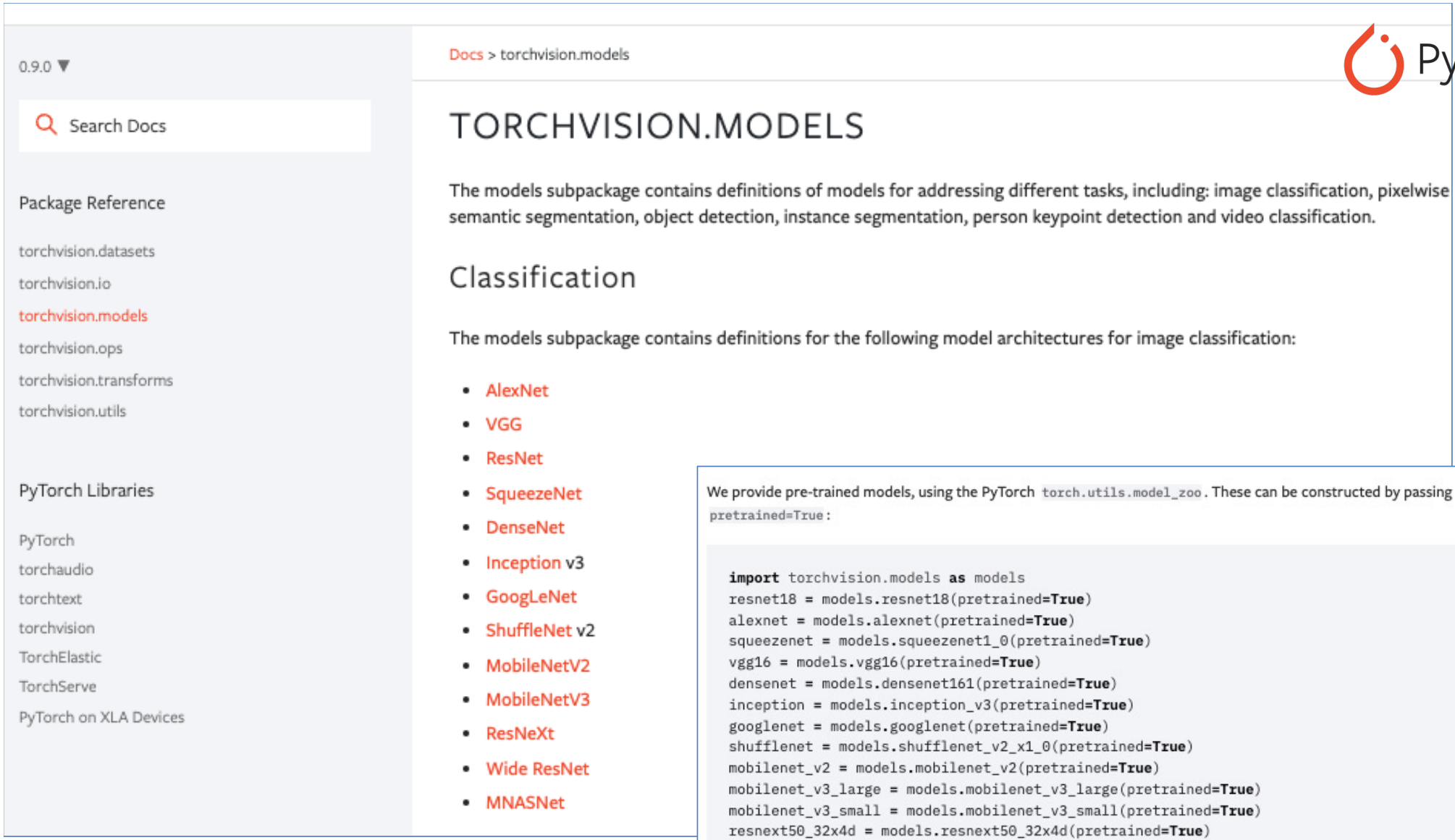
Small filters, Deeper networks

8 layers (AlexNet) → 16-19 layers (VGG16Net)

Only 3x3 CONV Stride 1, pad 1
And 2x2 MAX POOL stride 2



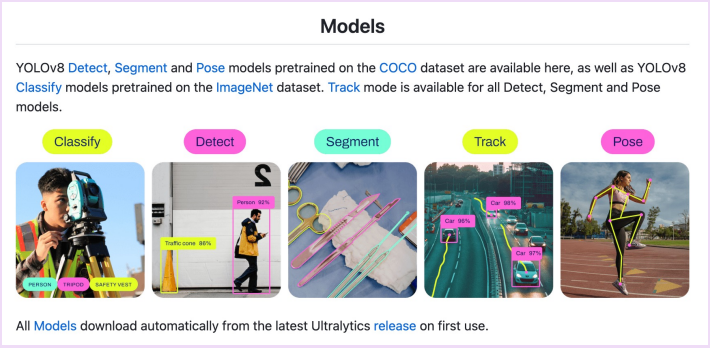
Network	A	B	C	D	E
Number of parameters (Millions)	133	133	134	138	144

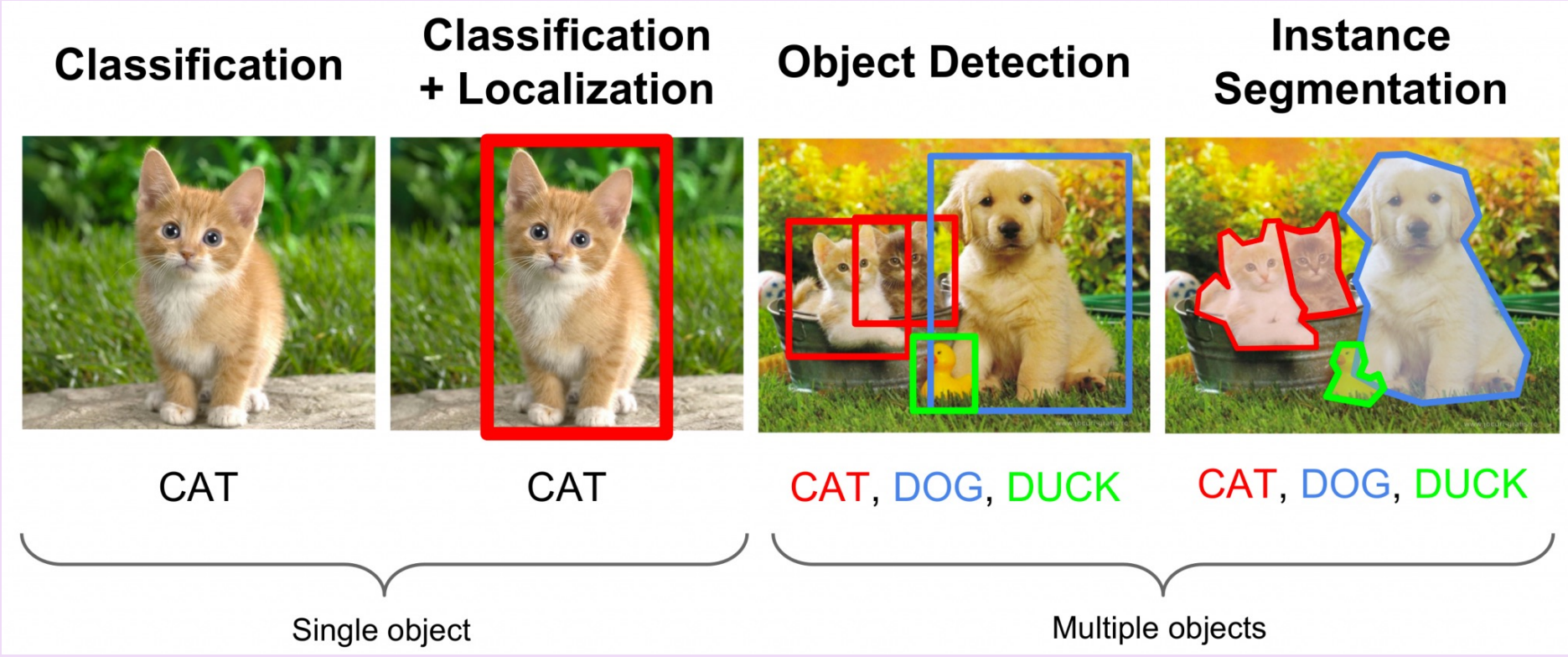


PyTorch

https://pytorch.org/vision/0.9/models.html

Image Classification Tasks





https://analyticsindiamag.com/top-5-image-classification-research-papers-every-data-scientist-should-know/

Recurrent Neural Networks (RNN)

Note: Today's lecture introduces several DL models so that you can understand how to select the right one for your problem. You'll learn more in depth in the AI course next semester.

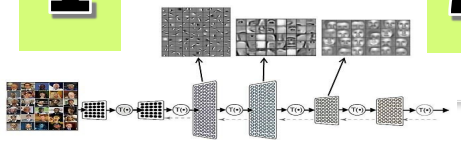
Deep Learning Application

Speech Recognition

Computer Vision

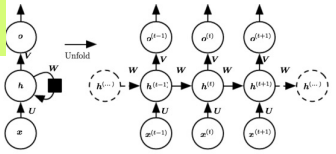
Natural Language Processing

1



CNN

2



RNN (LSTM)

Time Series Forecasting

Top 10 Strategic Technology Trends for 2020

People-centric

- Hyperautomation
- Multiexperience
- Democratization
- Human Augmentation
- Transparency and Traceability

Smart spaces

- Empowered Edge
- Distributed Cloud
- Autonomous Things
- Practical Blockchain
- AI Security

gartner.com/SmarterWithGartner

Gartner

Gartner Top Strategic Technology Trends for 2021

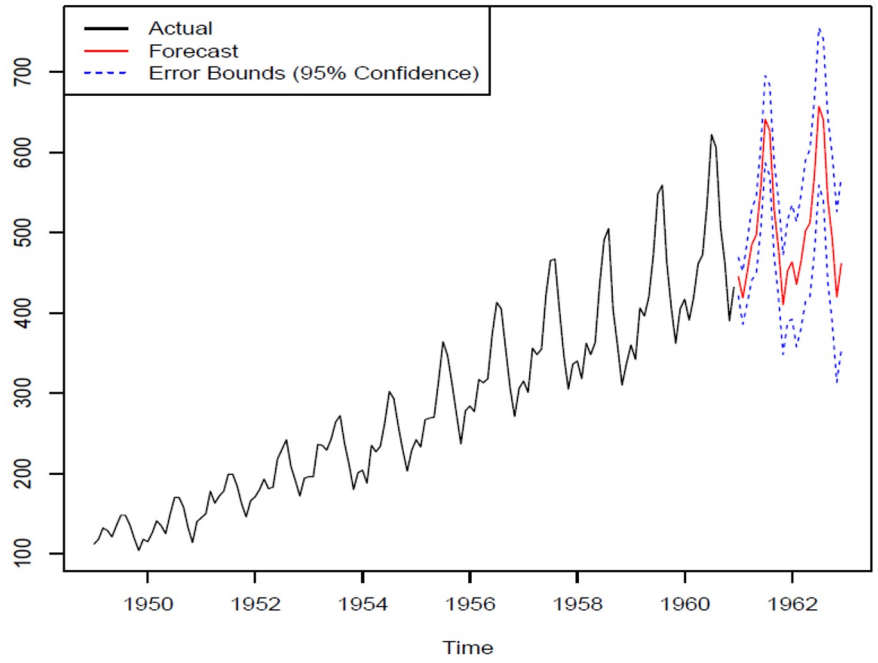
People centricity	Location independence	Resilient delivery
Internet of Behaviors	Distributed cloud	Intelligent composable business
Total experience strategy	Anywhere operations	AI engineering
Privacy-enhancing computing	Cybersecurity mesh	Hyperautomation

Combinatorial innovation

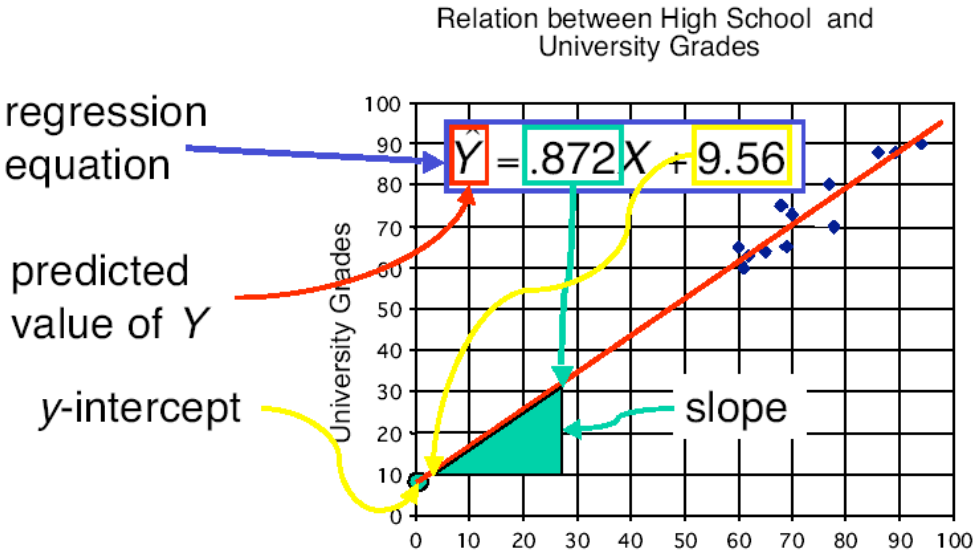
gartner.com/SmarterWithGartner

Source: Gartner

Gartner



1) Regression – Linear Relationship



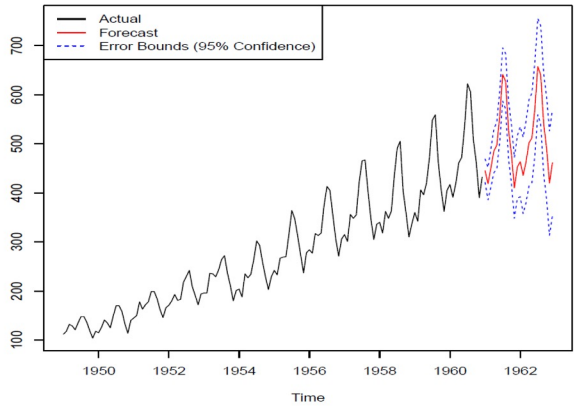
weight, coefficient

$$\underset{\text{target}}{\hat{y}} = \underset{\text{intercept}}{\hat{w}_0} + \hat{w}_1 x_1 + \hat{w}_2 x_2 \quad \text{input}$$

- The least square method aims to minimize the following term

$$\sum_{\substack{training \\ data}} (y_i - \hat{y}_i)^2$$

2) Autoregressive Model – Linear Relationship



- Based on linear regression, but using previous timestep data

$$X_t = c + \Sigma_{i=1}^{p} \varphi_i X_{t-i}$$

- Where X_t is data at timestep t, c is constant, and $\varphi_1, \varphi_2, \varphi_3, \ldots, \varphi_p$ are parameters for each timestep data

Example: When p = 2 (looking back two steps), the equation will be

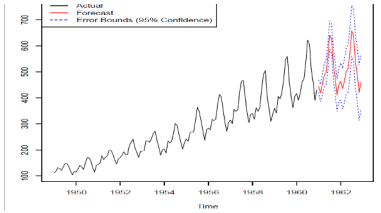
$$X_t = c + \varphi_1 X_{t-1} + \varphi_2 X_{t-2}$$

- Parameters are able to calculate using various method, such as ordinary least square procedure or Yule–Walker equations

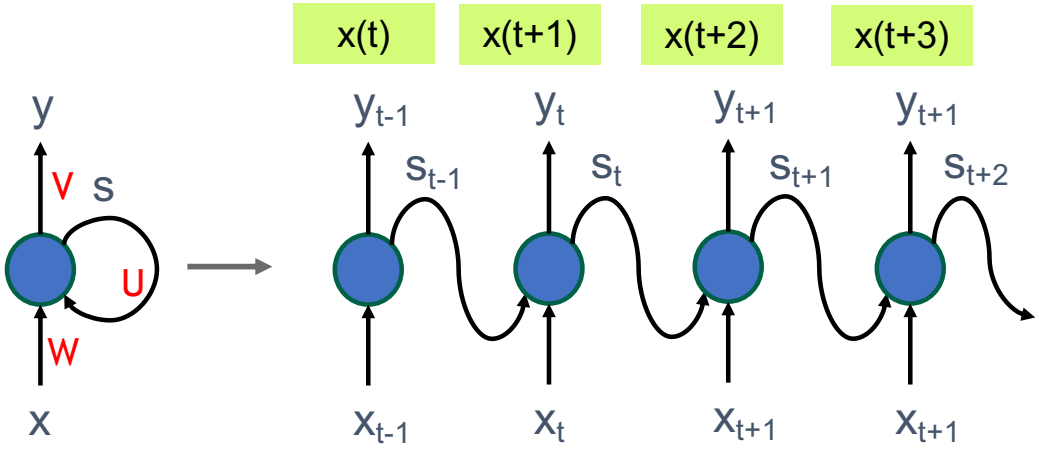
$$x(t+1) = \hat{w}_0 + \hat{w}_1 x(t) + \hat{w}_2 x(t-1)$$

3) Recurrent Neural Networks (RNN)

Non-Linear Relationship



Forward Pass



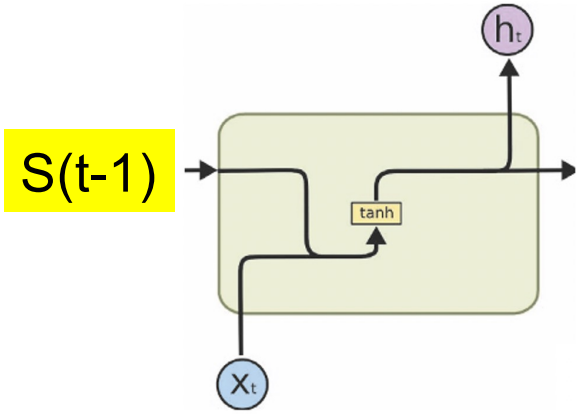
Unrolling in Time

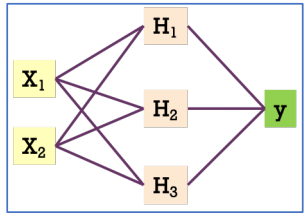
$$s_t = \tanh\left(Ux_t + Ws_{t-1}\right)$$

x(t+1) $$\hat{y}_t = softmax\left(Vs_t\right)$$

x(t+1)

y(t)



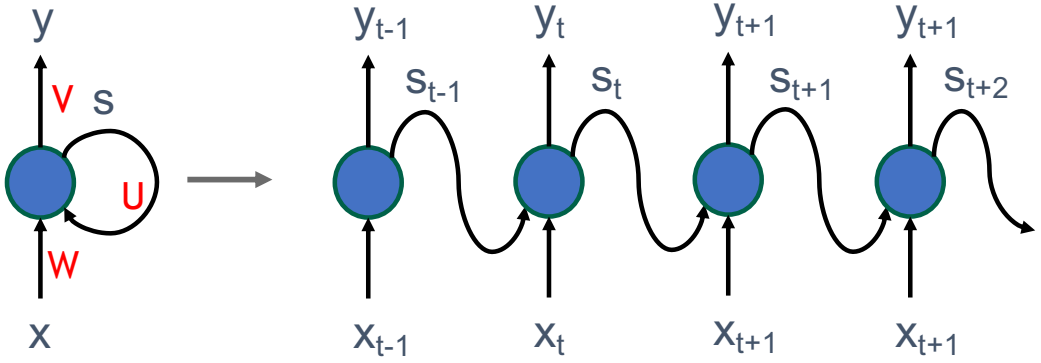


Reference: https://www.cse.iitk.ac.in/users/sigml/lec/Slides/LSTM.pdf

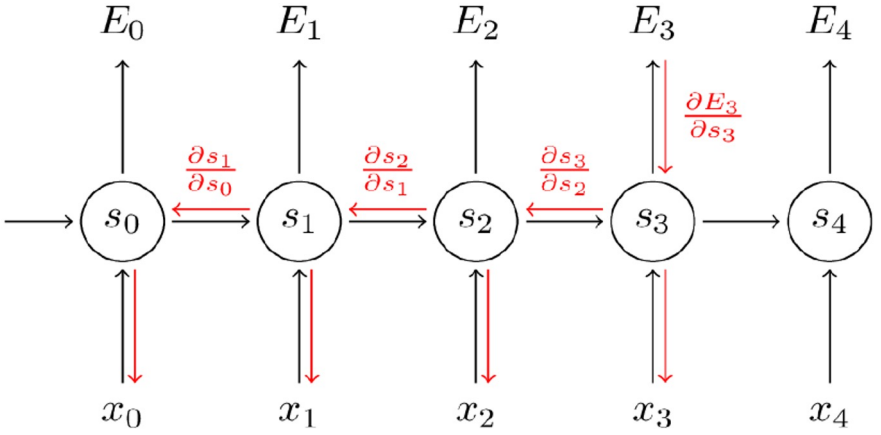
3) Recurrent Neural Networks (RNN)

Ggradient vanishing
= multiplication

Forward Pass



Backward Pass

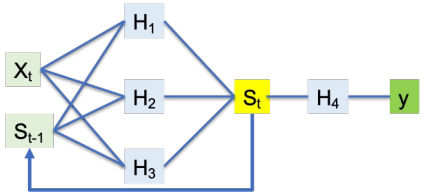


Unrolling in Time

Backpropagation-Through-Time (BPTT)

$$s_t = \tanh\left(Ux_t + Ws_{t-1}\right)$$

$$\hat{y}_t = softmax\left(Vs_t\right)$$

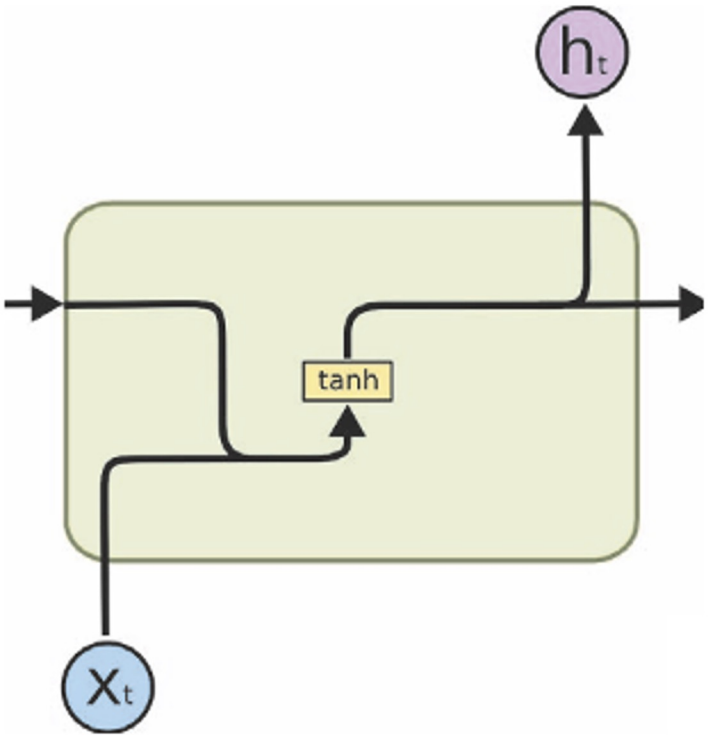


$$\frac{\partial E}{\partial W} = \sum_t \frac{\partial E_t}{\partial W}$$

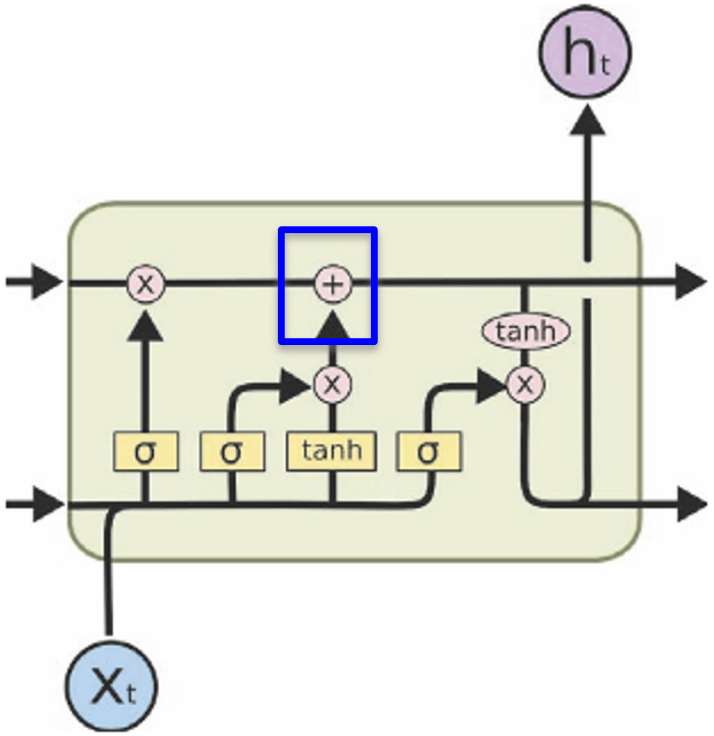
$$\frac{\partial E_3}{\partial W} = \sum_{k=0}^{3} \frac{\partial E_3}{\partial \hat{y}_3} \frac{\partial \hat{y}_3}{\partial s_3} \frac{\partial s_3}{\partial s_k} \frac{\partial s_k}{\partial W}$$

Reference: https://www.cse.iitk.ac.in/users/sigml/lec/Slides/LSTM.pdf

4) Long-Short Term Memory (LSTM)



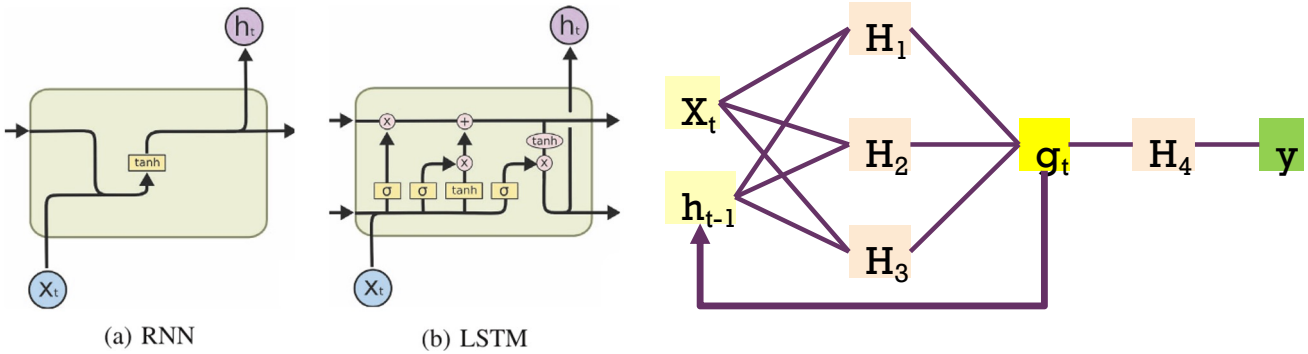
(a) RNN



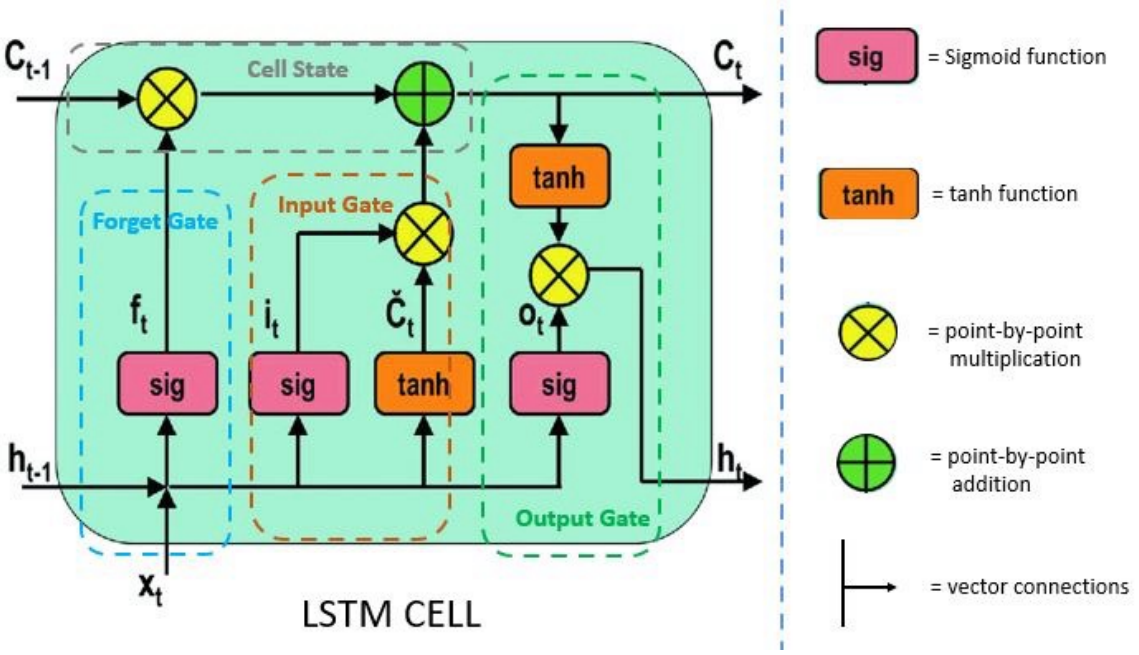
(b) LSTM

Reference: https://cs.uwaterloo.ca/~mli/Deep-Learning-2017-Lecture6RNN.ppt

Inside LSTM



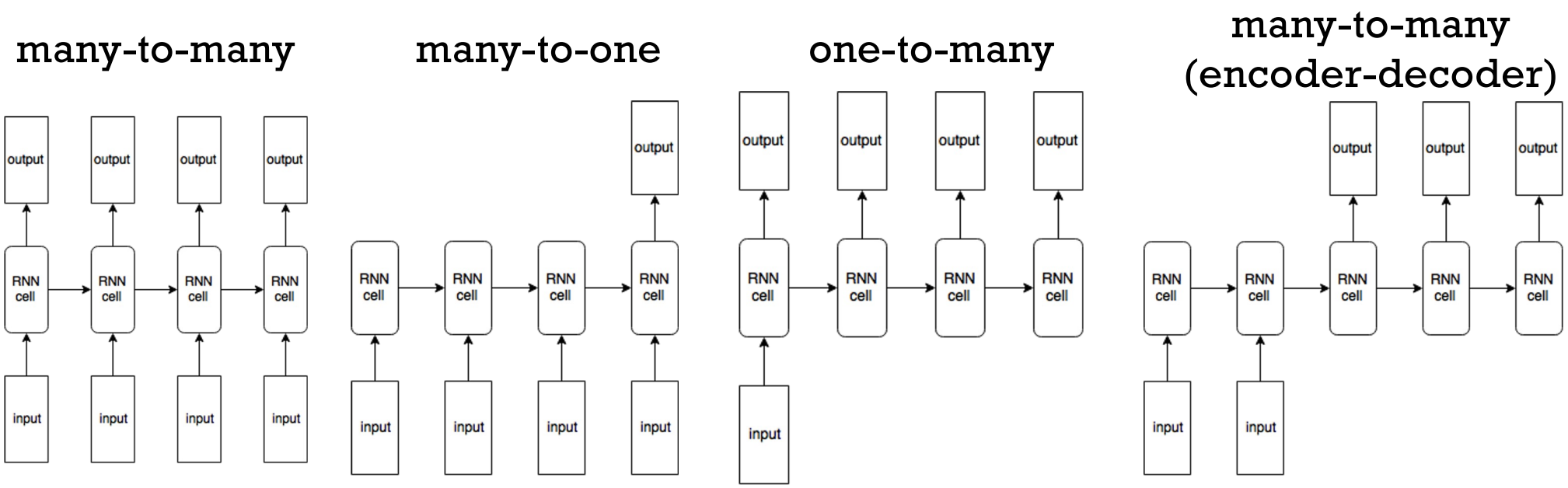
(a) RNN (b) LSTM



There are 3 gates with 2 outputs.

$$\mathbf{i}_{(t)} = \sigma(\mathbf{W}_{xi}^{T} \cdot \mathbf{x}_{(t)} + \mathbf{W}_{hi}^{T} \cdot \mathbf{h}_{(t-1)} + \mathbf{b}_i)$$
$$\mathbf{f}_{(t)} = \sigma(\mathbf{W}_{xf}^{T} \cdot \mathbf{x}_{(t)} + \mathbf{W}_{hf}^{T} \cdot \mathbf{h}_{(t-1)} + \mathbf{b}_f)$$
$$\mathbf{o}_{(t)} = \sigma(\mathbf{W}_{xo}^{T} \cdot \mathbf{x}_{(t)} + \mathbf{W}_{ho}^{T} \cdot \mathbf{h}_{(t-1)} + \mathbf{b}_o)$$
$$\mathbf{g}_{(t)} = \tanh(\mathbf{W}_{xg}^{T} \cdot \mathbf{x}_{(t)} + \mathbf{W}_{hg}^{T} \cdot \mathbf{h}_{(t-1)} + \mathbf{b}_g)$$
$$\mathbf{c}_{(t)} = \mathbf{f}_{(t)} \otimes \mathbf{c}_{(t-1)} + \mathbf{i}_{(t)} \otimes \mathbf{g}_{(t)}$$
$$\mathbf{y}_{(t)} = \mathbf{h}_{(t)} = \mathbf{o}_{(t)} \otimes \tanh(\mathbf{c}_{(t)})$$

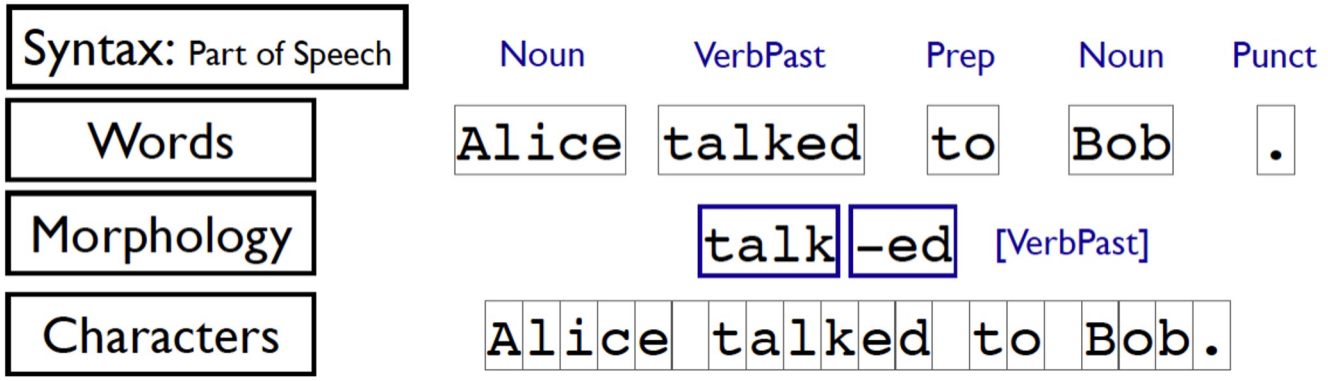
RNN in NLP Application

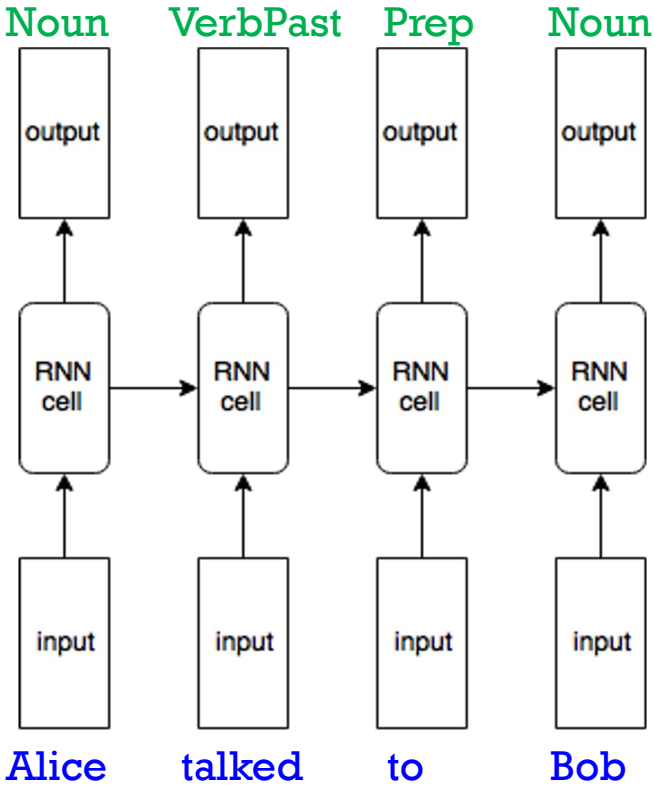


Reference: The Unreasonable Effectiveness of Recurrent Neural Networks, http://karpathy.github.io/2015/05/21/rnn-effectiveness/

1) Many-to-Many (e.g., PoS Tagging)

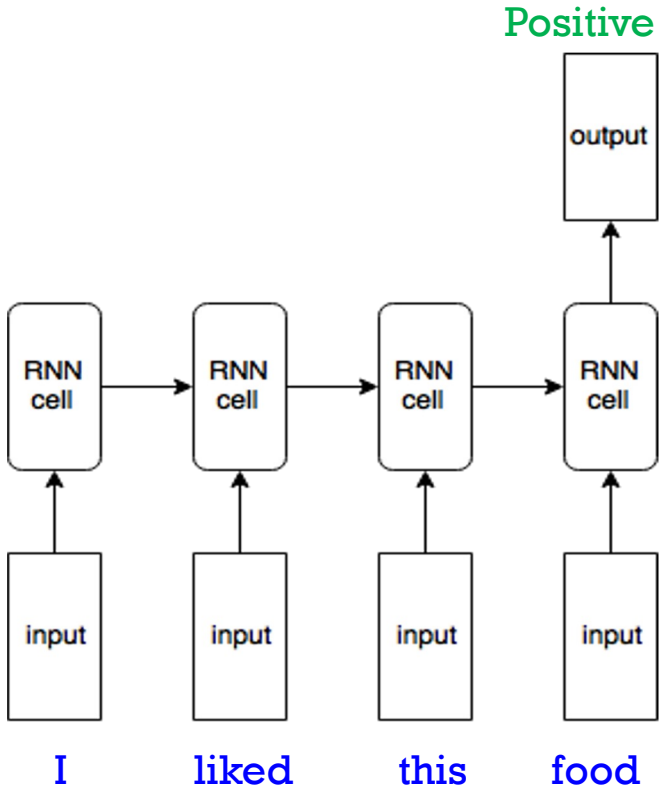
- You have seen and implemented this type of RNN architecture in your homework already.
- E.g., Tokenization, POS tagging
- Sequence Input, Sequence Output





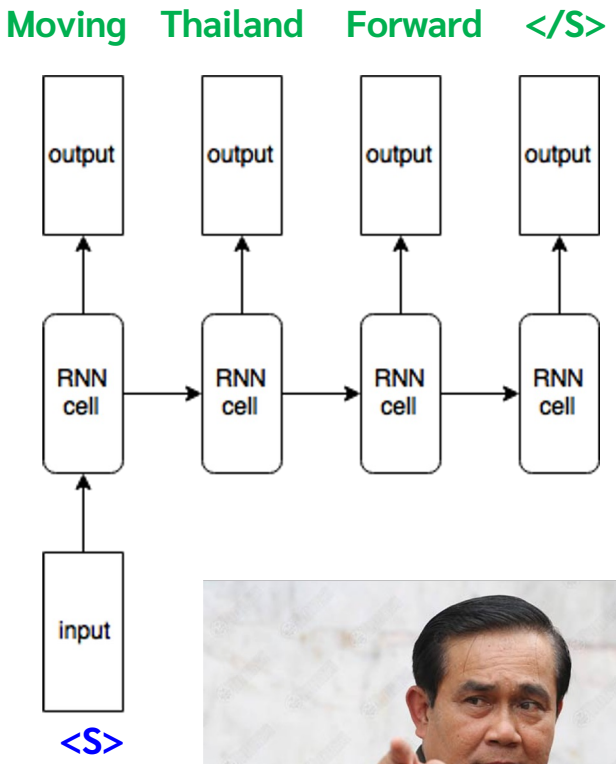
2) Many-to-One (e.g., Text Classification)

- E.g., sentiment analysis, text classification
- Sequence input, scalar output



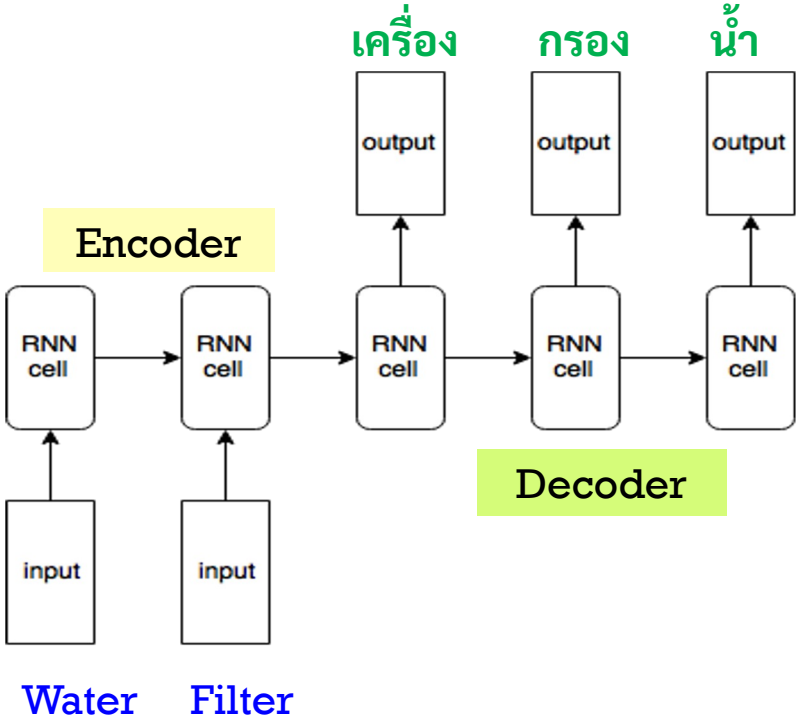
3) One-to-Many (Text Generation)

- Scalar input & Sequence output
- E.g., Music Generation, Image caption generation
- Music **generation**
  - Input: Initial seed
  - Output: Sequence of music notes
- Image caption **generation**
  - Input: Image features extracted by CNN
  - Output: Sequence of text

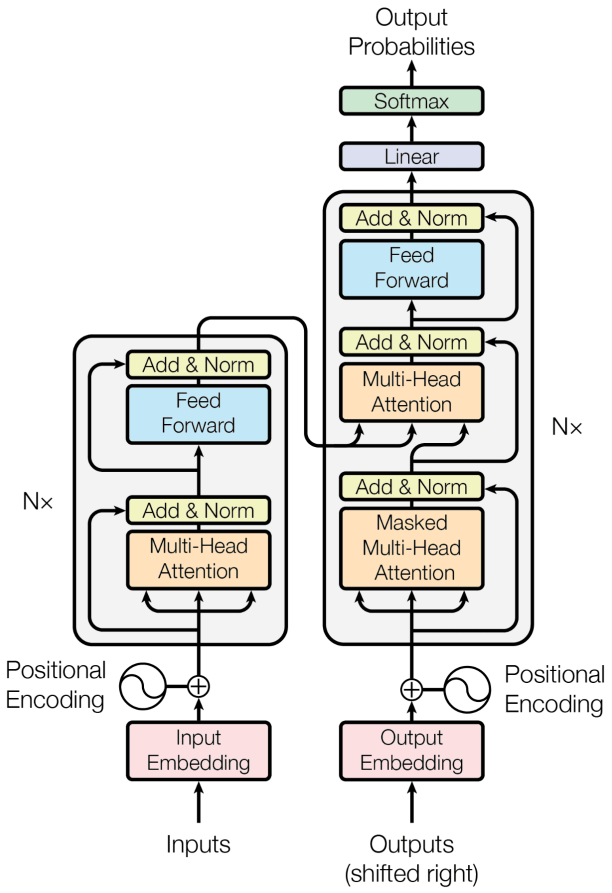


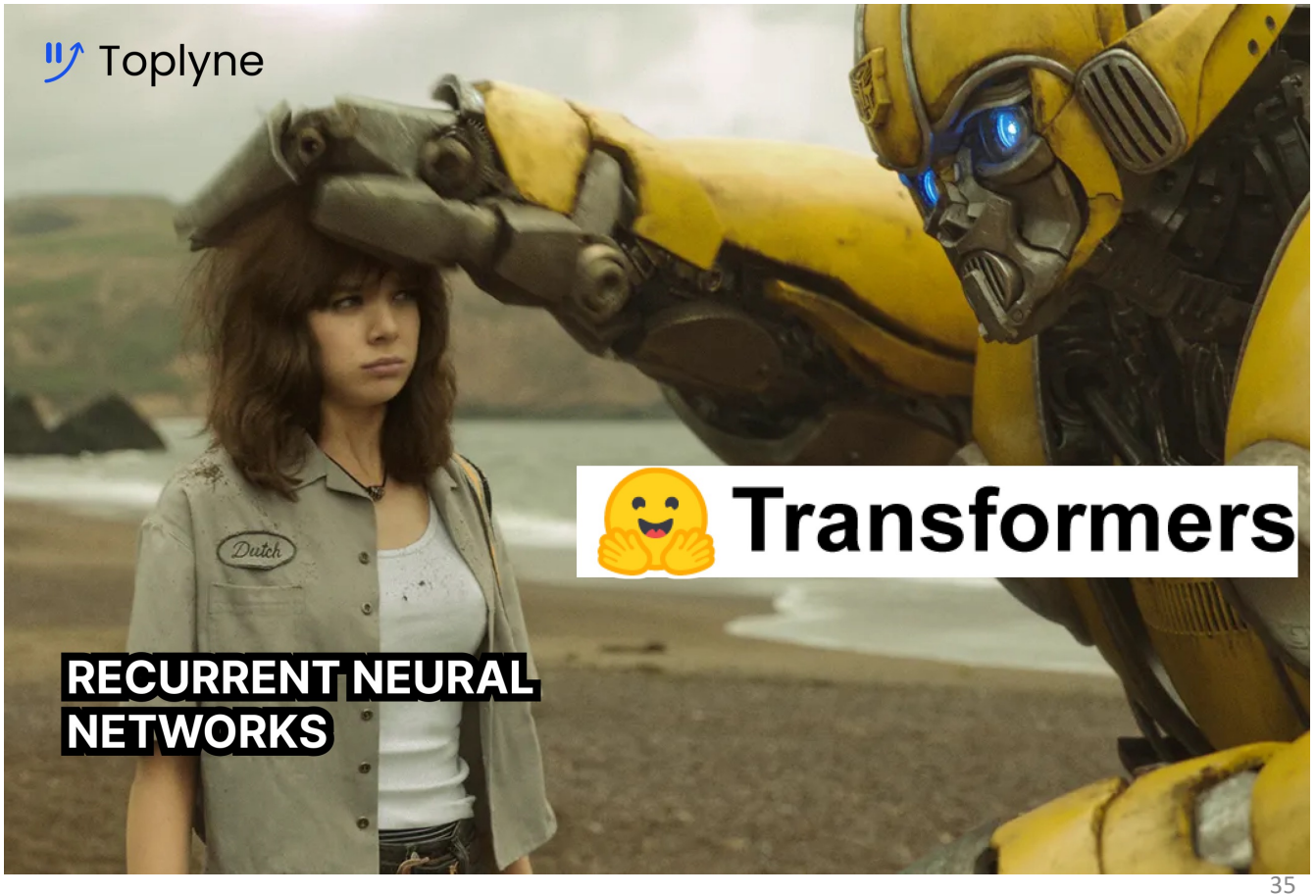
4) Many-to-Many (encoder-decoder) (e.g., Machine Translation)

- Sequence Input, Sequence output
- These two sequences can be of different length
- E.g., Machine Translation
  - Input: English Sentence
  - Output: Thai Sentence
- Machine Translation is also a **text generation task**



Rise of Transformer (2017)

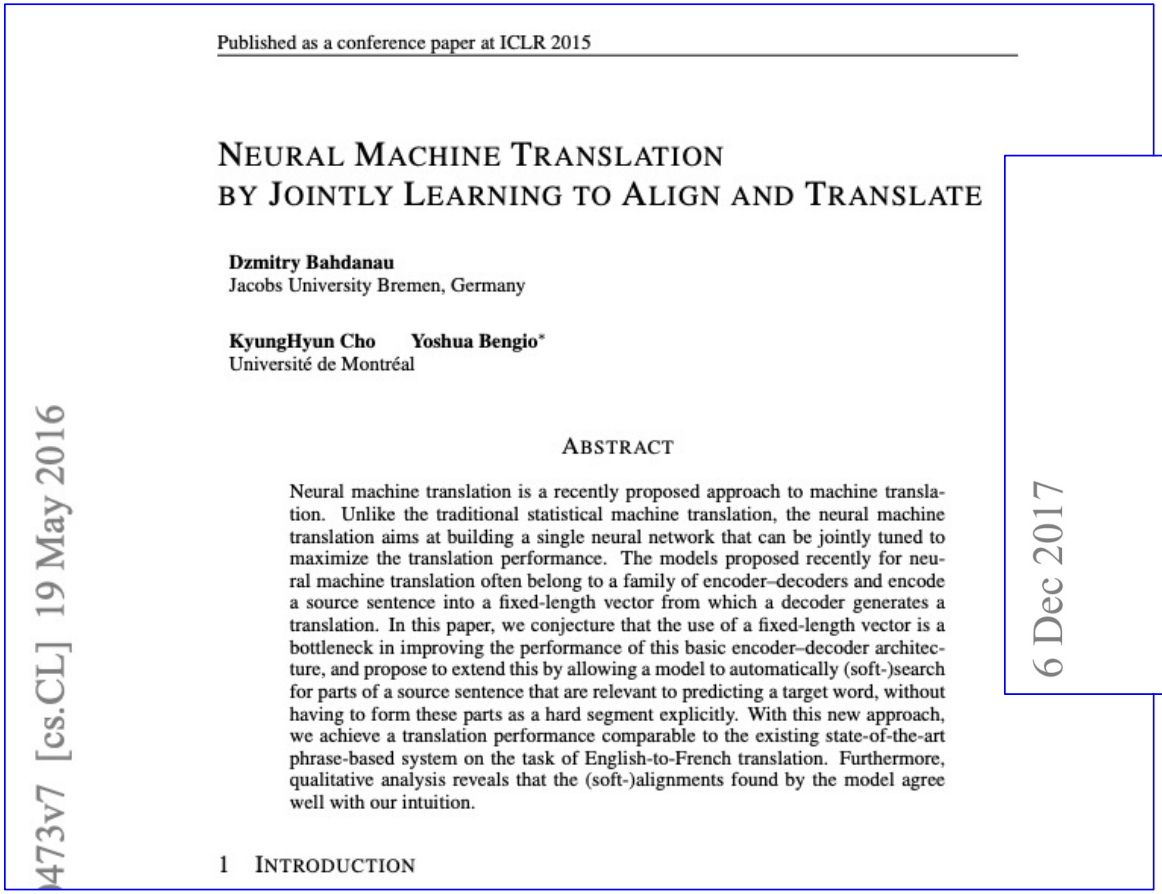




Transformer (TF)

Note: Today's lecture introduces several DL models so that you can understand how to select the right one for your problem. You'll learn more in depth in other AI courses.

Attention mechanism



Published as a conference paper at ICLR 2015

NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE

Dzmitry Bahdanau
Jacobs University Bremen, Germany

KyungHyun Cho **Yoshua Bengio***
Université de Montréal

ABSTRACT

Neural machine translation is a recently proposed approach to machine translation. Unlike the traditional statistical machine translation, the neural machine translation aims at building a single neural network that can be jointly tuned to maximize the translation performance. The models proposed recently for neural machine translation often belong to a family of encoder–decoders and encode a source sentence into a fixed-length vector from which a decoder generates a translation. In this paper, we conjecture that the use of a fixed-length vector is a bottleneck in improving the performance of this basic encoder–decoder architecture, and propose to extend this by allowing a model to automatically (soft-)search for parts of a source sentence that are relevant to predicting a target word, without having to form these parts as a hard segment explicitly. With this new approach, we achieve a translation performance comparable to the existing state-of-the-art phrase-based system on the task of English-to-French translation. Furthermore, qualitative analysis reveals that the (soft-)alignments found by the model agree well with our intuition.

1 INTRODUCTION

Reference: Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate." ICLR(2015).

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

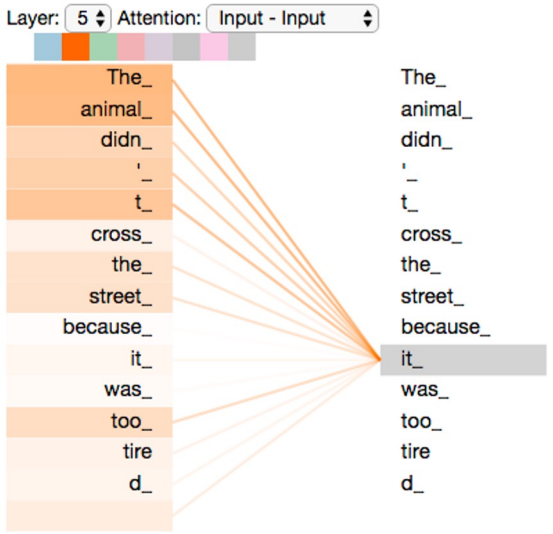
Attention Is All You Need. 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

Scaled Dot-Product Attention (1)

- Introduced in Attention is all you need (Viswani et al., 2017)
- **NO** recurrence nor convolution
- Widely used today in all Transformer-based model
- "Relating different positions of a single sequence in order to compute a representation of the sequence"

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

where the query, keys, values, and output are all vectors and d_k is a number.



Attention Is All You Need. 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

Scaled Dot-Product Attention (2)

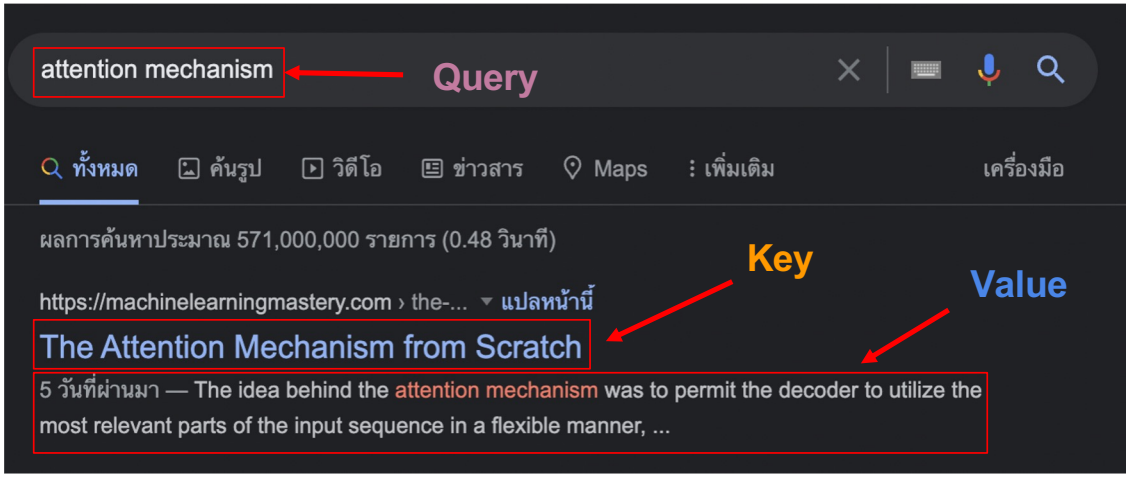
What are the “query”, “key”, and “value” vectors?

As an analogy, think of Google Search.

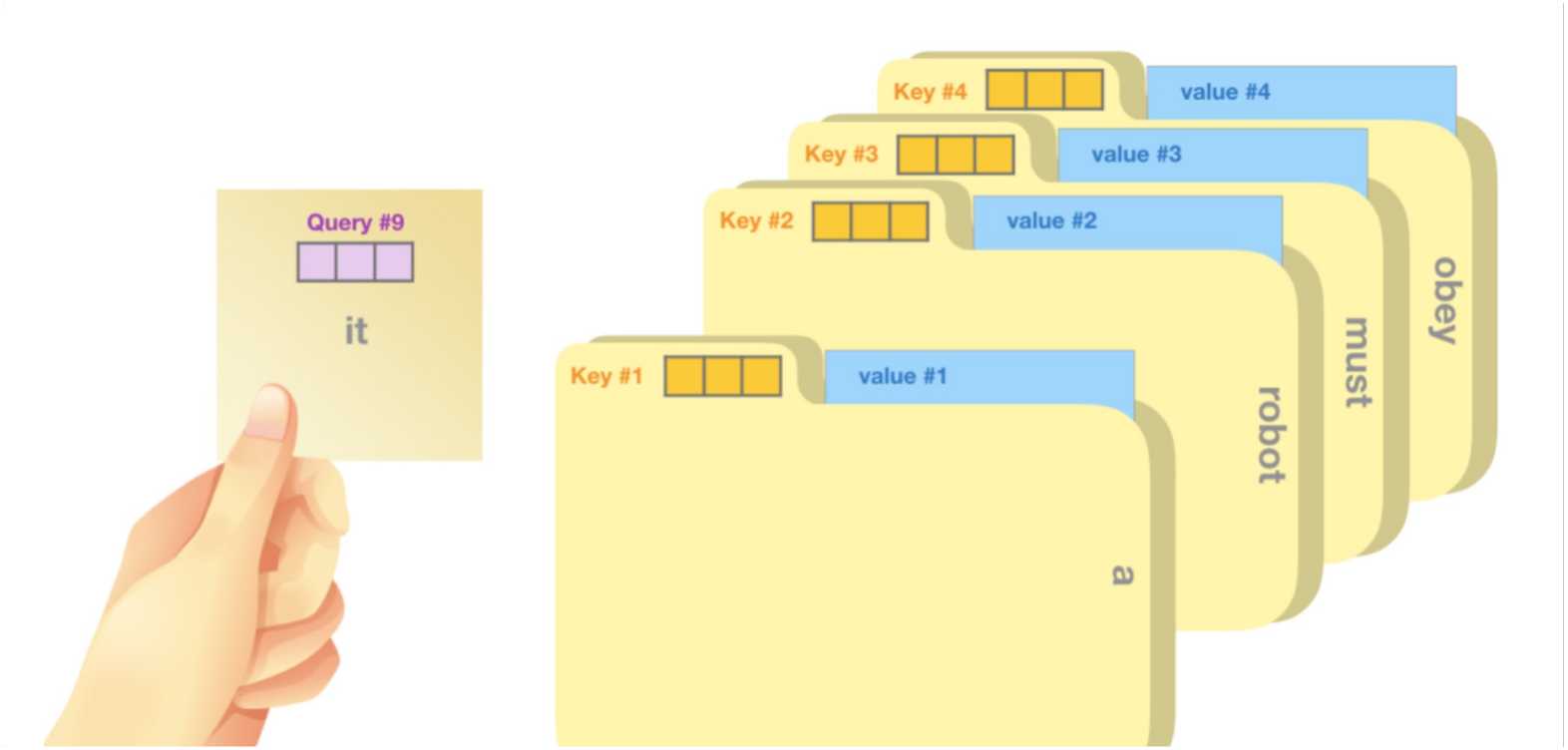
Query - what we want to know

Key - how to index information

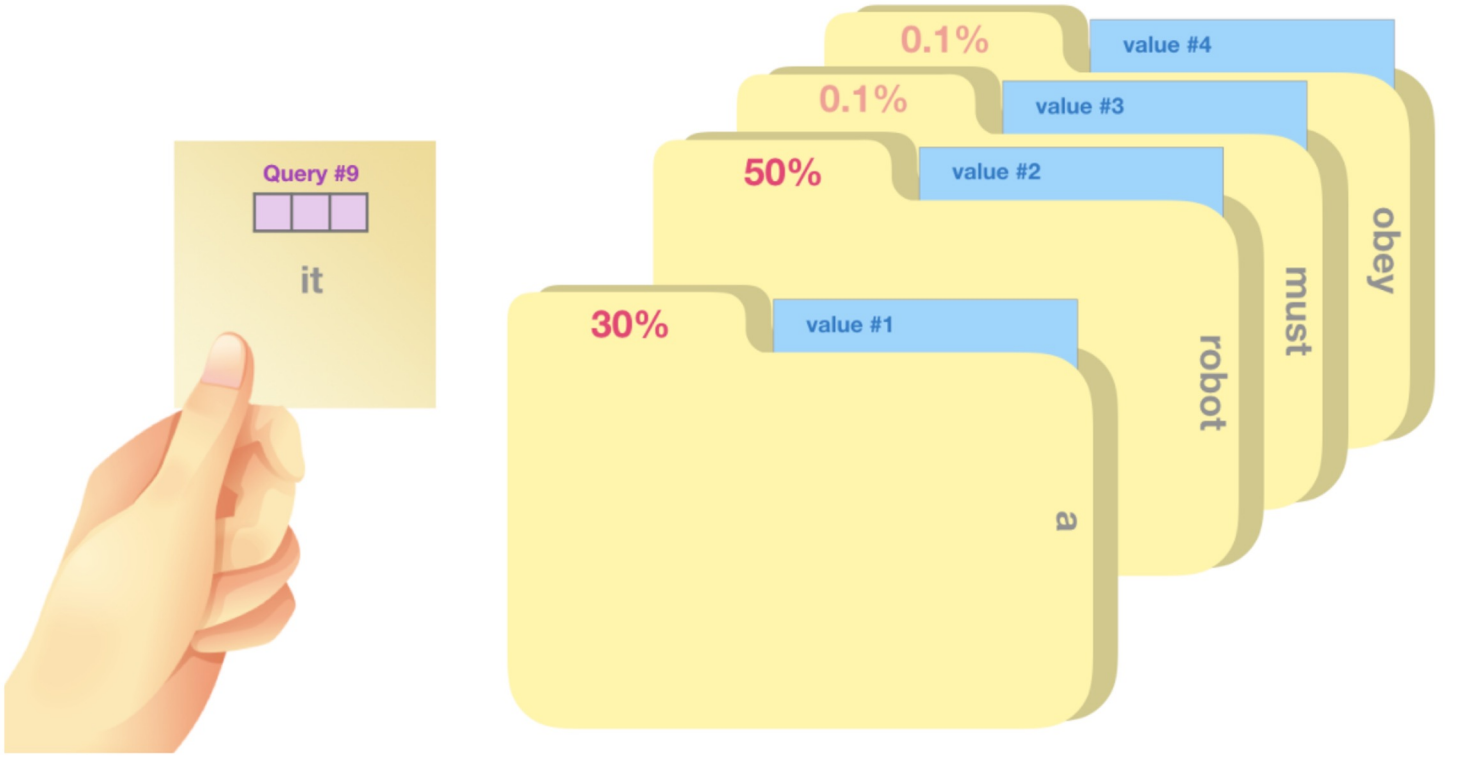
Value - what kind of info is in each website



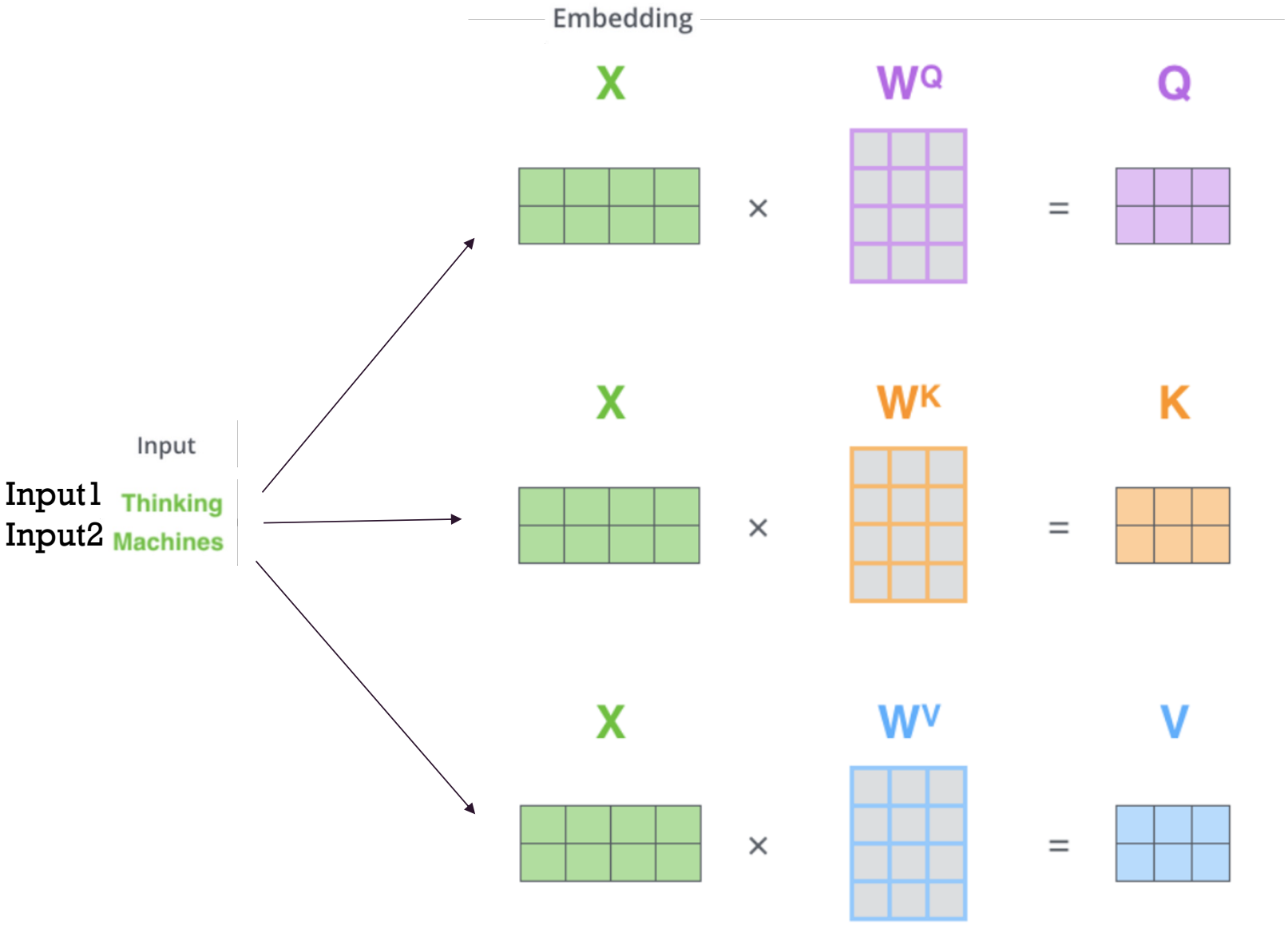
Attention Is All You Need. 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

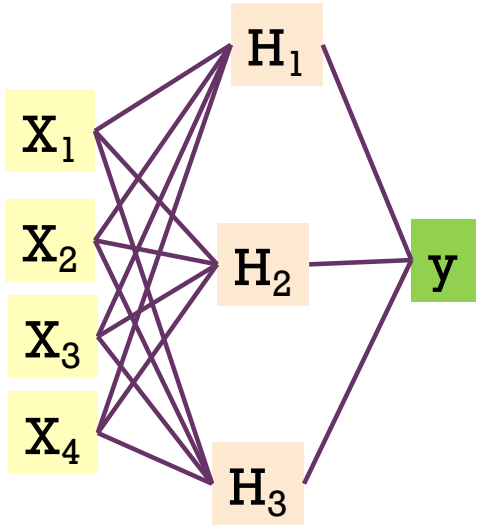


http://jalammar.github.io/illustrated-gpt2/



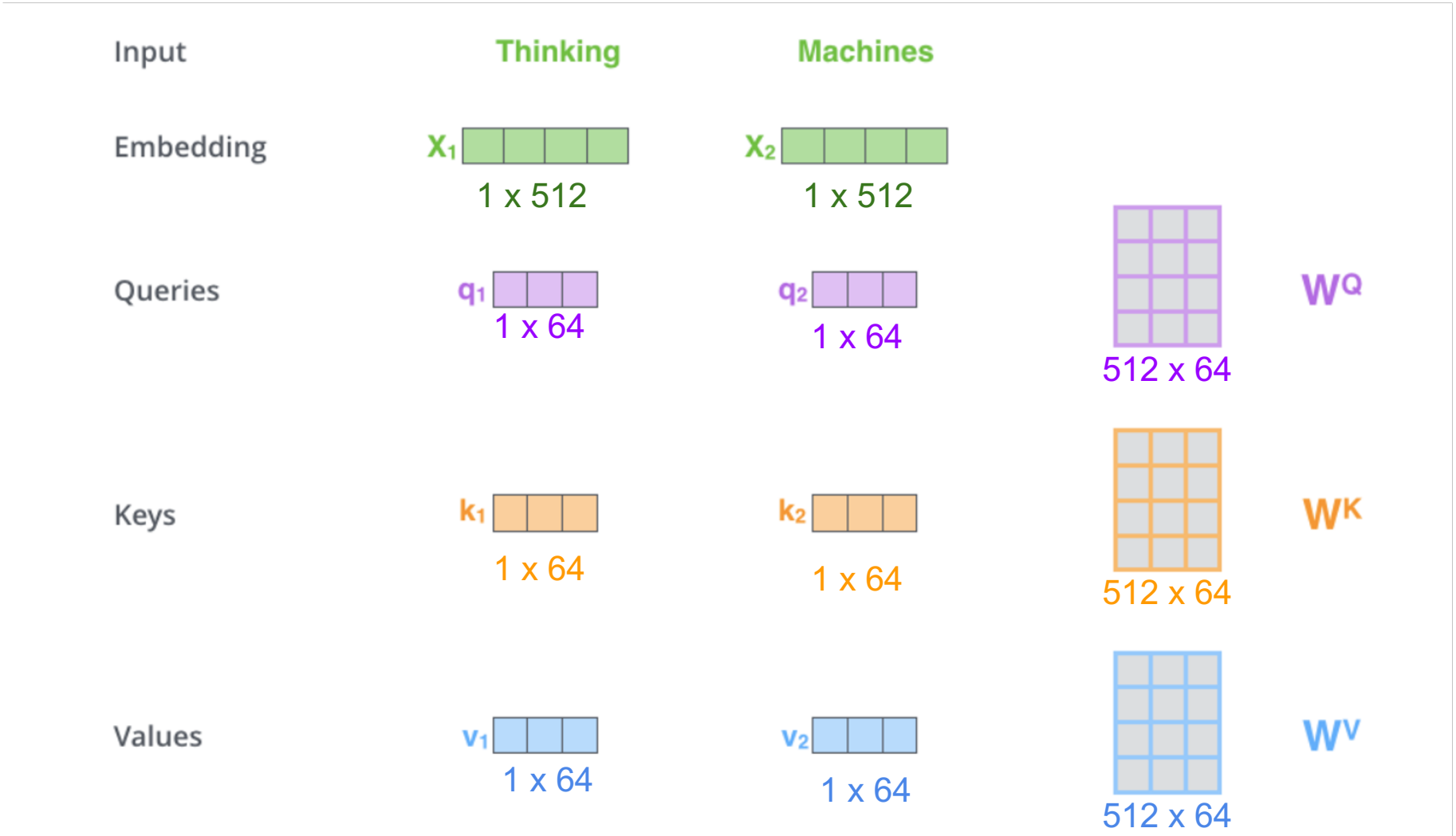
http://jalammar.github.io/illustrated-gpt2/





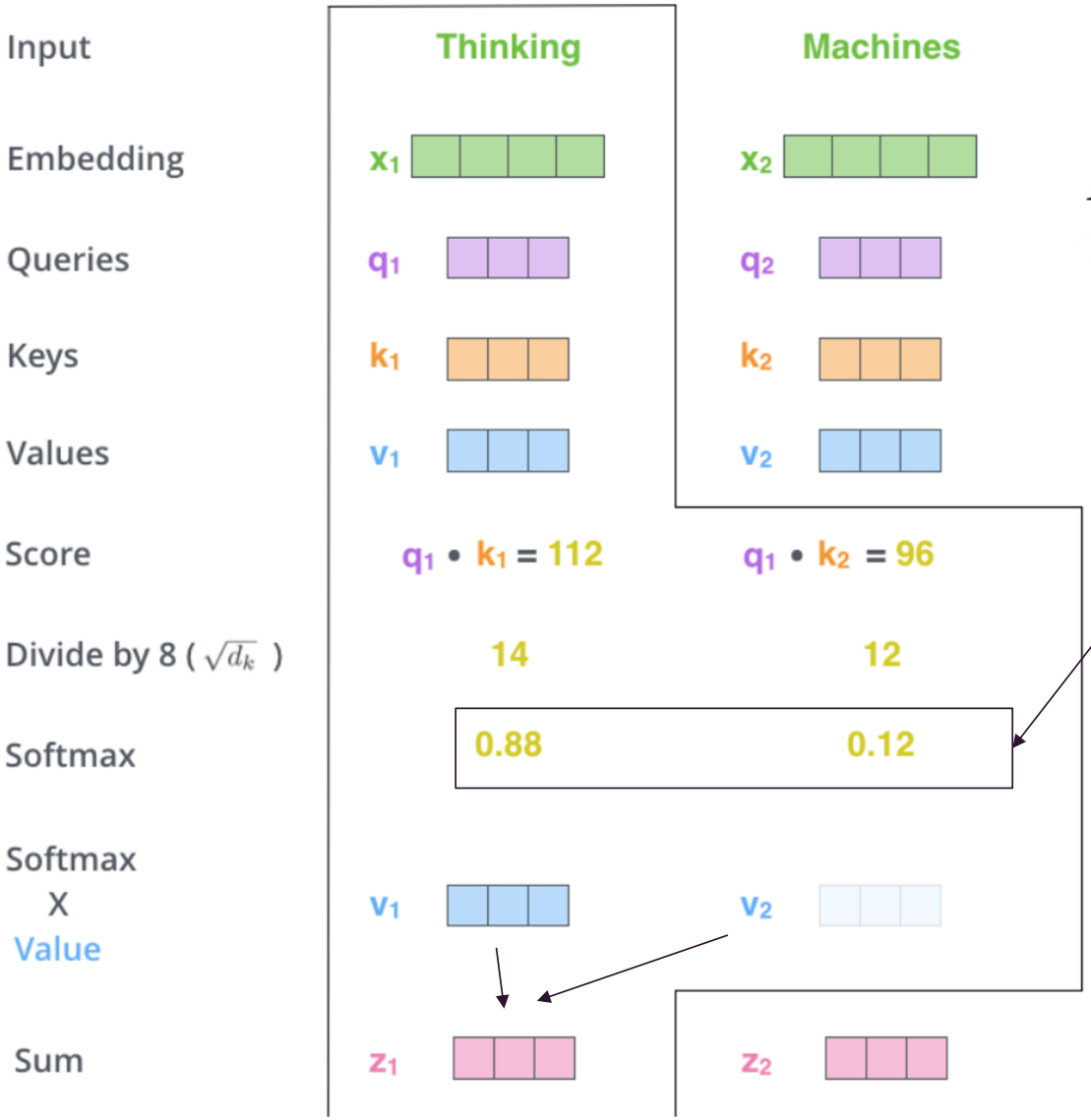
W^Q, W^K, and W^V are linear layers

http://jalammar.github.io/illustrated-transformer/



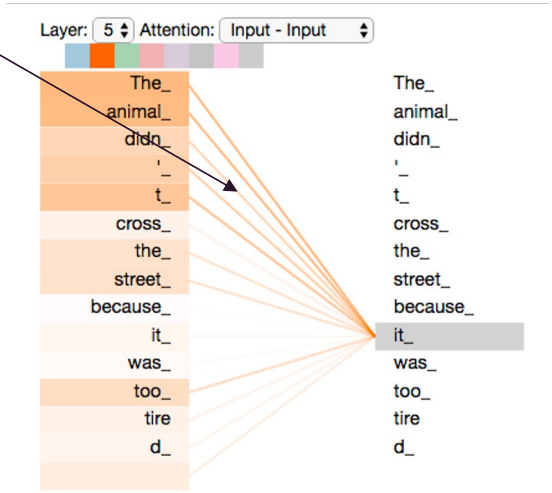
Multiplying x1 by the WQ weight matrix produces q1, the "query" vector associated with that word. We end up creating a "query", a "key", and a "value" projection of each word in the input sentence.

http://jalammar.github.io/illustrated-transformer/



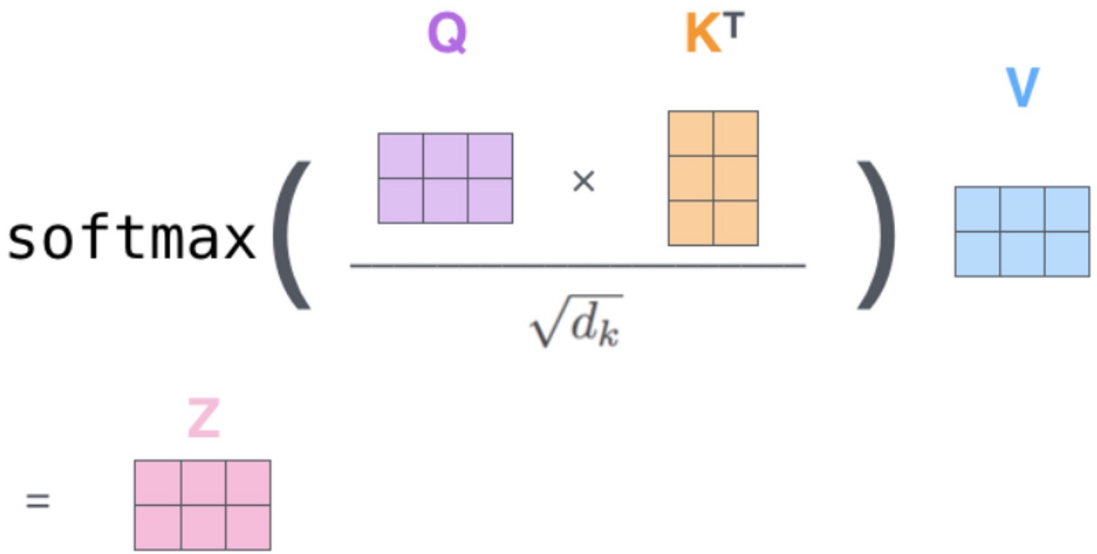
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

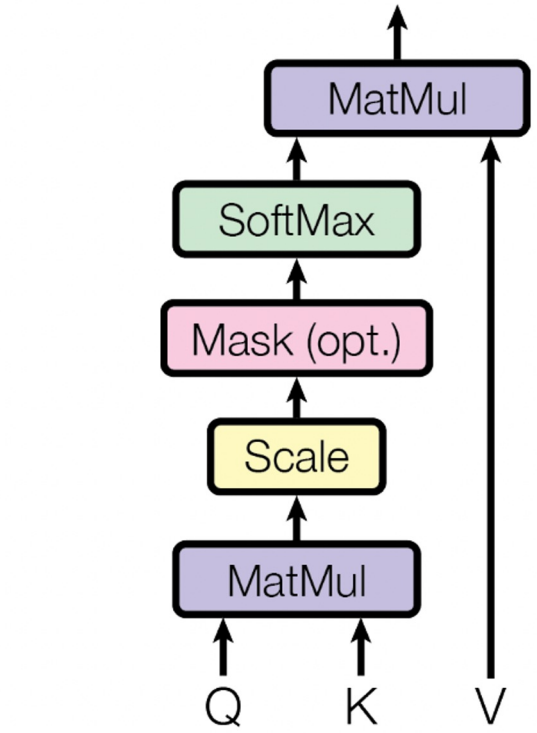
Attention scores



http://jalammar.github.io/illustrated-transformer/

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$





http://jalammar.github.io/illustrated-transformer/

Attention Is All You Need

Ashish Vaswani[*]
Google Brain
avaswani@google.com

Noam Shazeer[*]
Google Brain
noam@google.com

Niki Parmar[*]
Google Research
nikip@google.com

Jakob Uszkoreit[*]
Google Research
usz@google.com

Llion Jones[*]
Google Research
llion@google.com

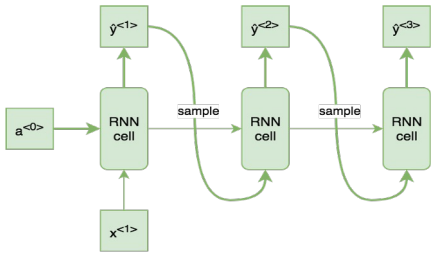
Aidan N. Gomez[*] [†]
University of Toronto
aidan@cs.toronto.edu

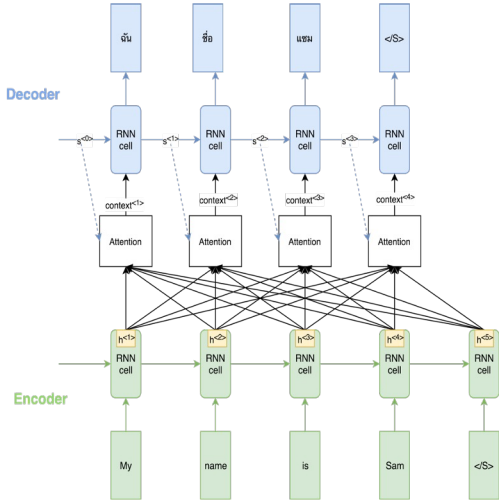
Łukasz Kaiser[*]
Google Brain
lukaszkaiser@google.com

Illia Polosukhin[*] [‡]
illia.polosukhin@gmail.com

Attention Is All You Need. 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

Transformer



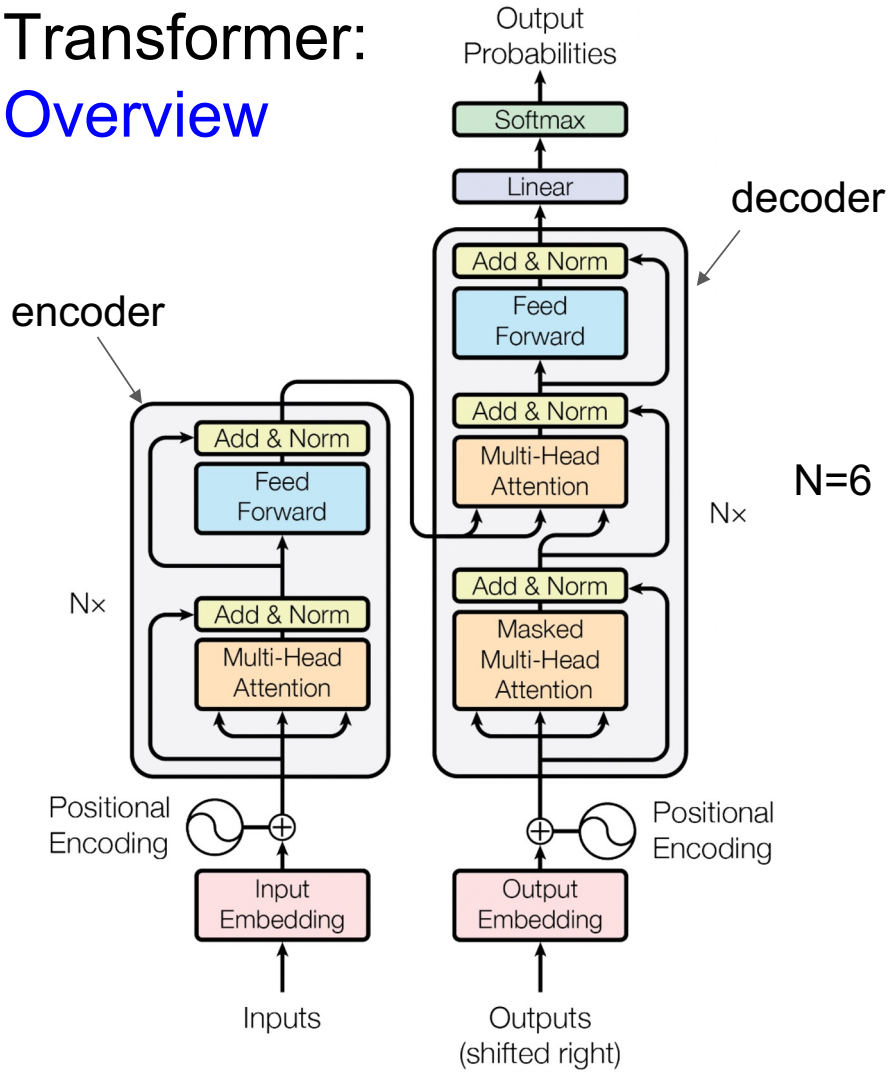


Only rely on the attention mechanism - **No RNN**!

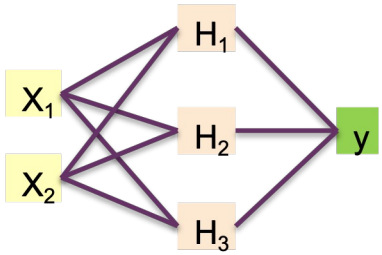
- More parallelizable -> Faster training time
- Better at longer sequence

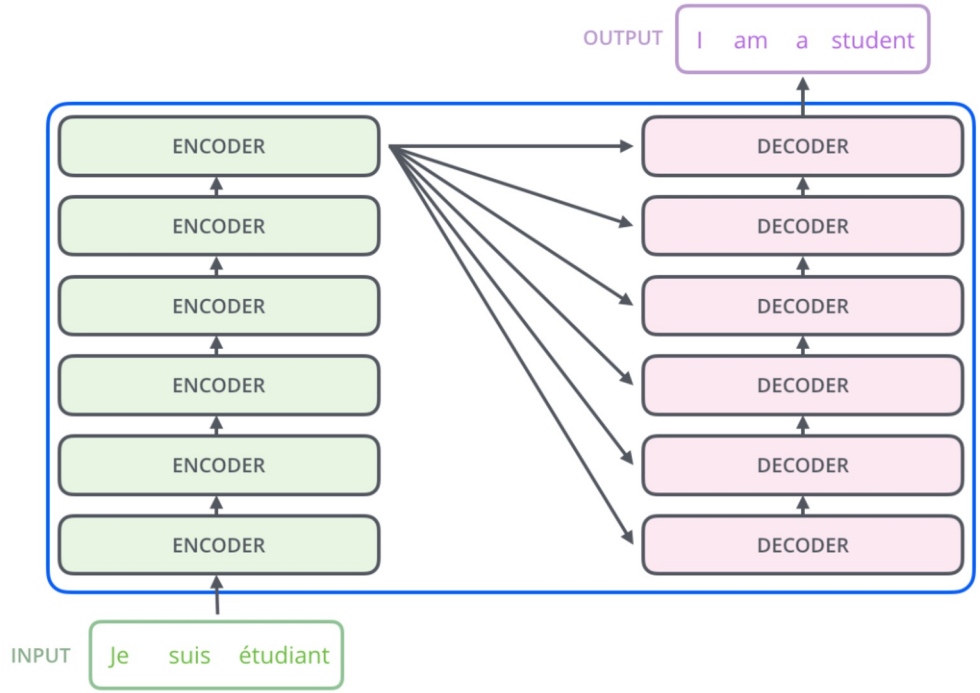
Attention Is All You Need. 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

Transformer: Overview



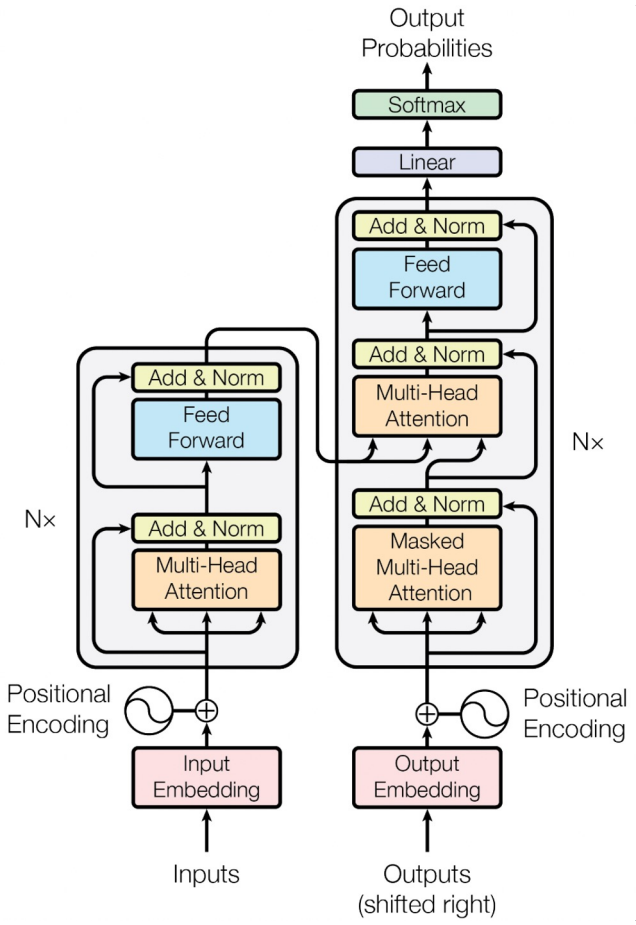
=

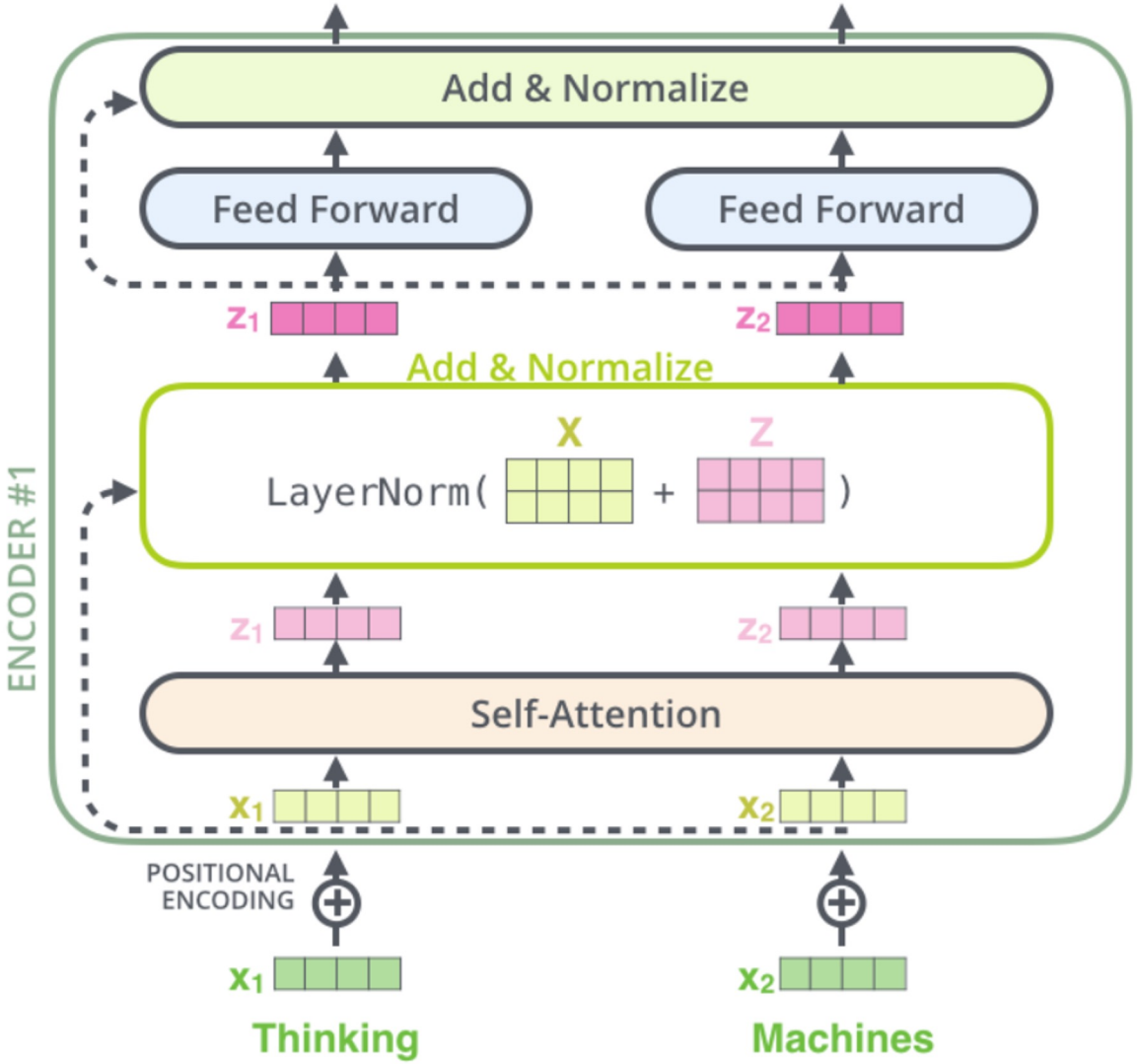




http://jalammar.github.io/illustrated-transformer/

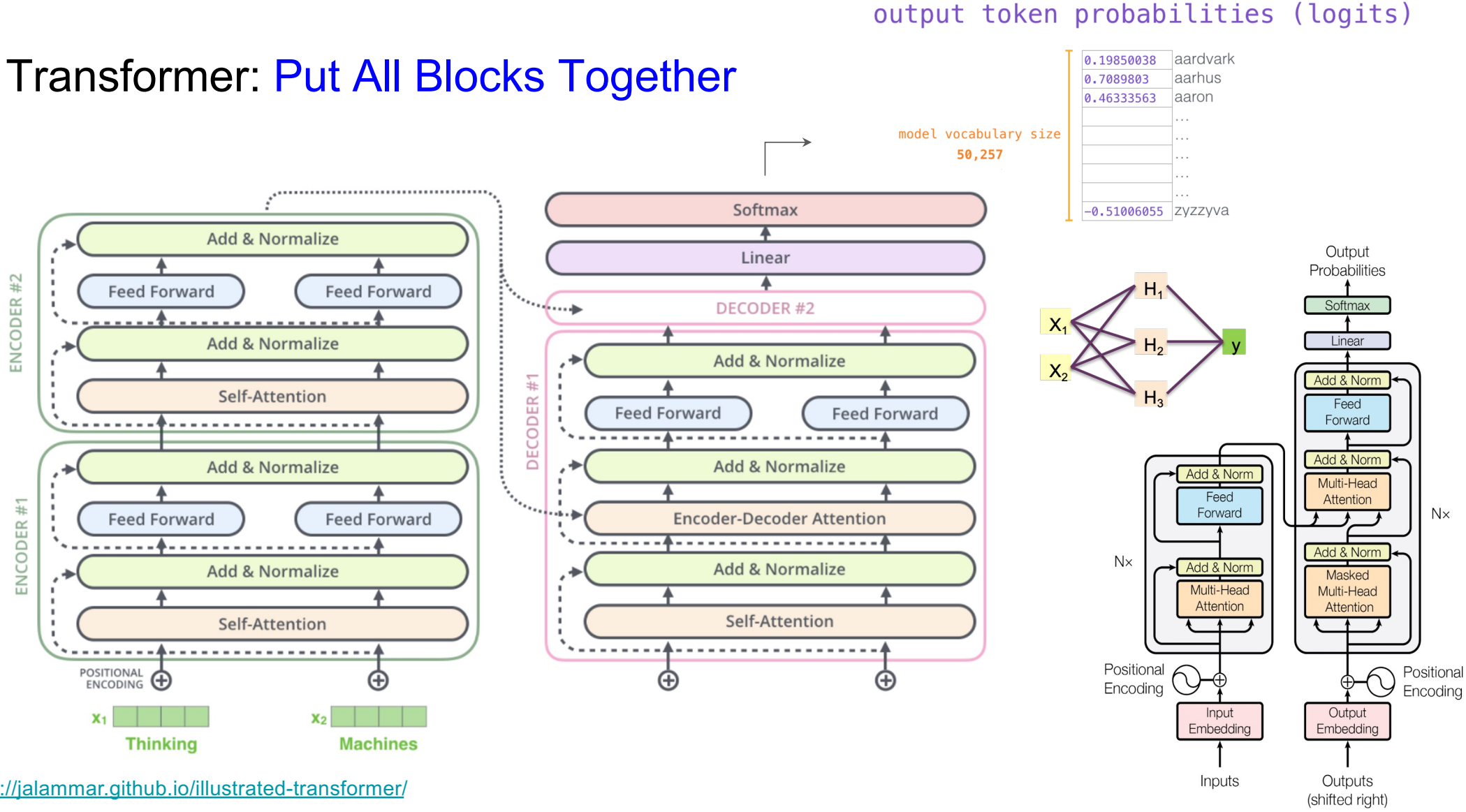
Transformer Block





http://jalammar.github.io/illustrated-transformer/

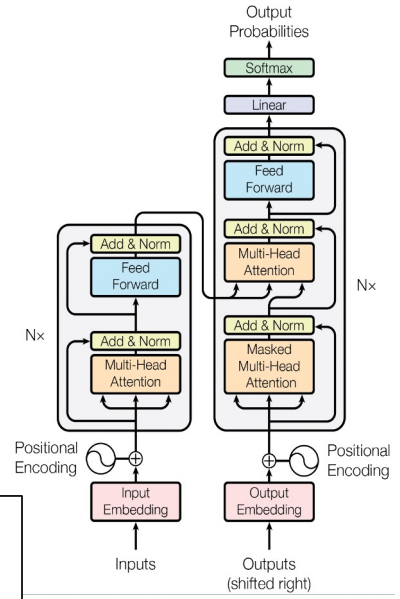
Transformer: Put All Blocks Together



http://jalammar.github.io/illustrated-transformer/

Transformer-Based Models

1. Decoder-based model: OpenAI GPT (Generative Pre-Training) [Radford, 2018]
2. Encoder-based model: BERT [Devlin, et al, 2018]:
    - Bidirectional Encoder Representation from Transformers
3. Encoder and Decoder: BART [2019]



Improving Language Understanding by Generative Pre-Training

Alec Radford, OpenAI, alec@openai.com
Karthik Narasimhan, OpenAI, karthikn@openai.com
Tim Salimans, OpenAI, tim@openai.com
Ilya Sutskever, OpenAI, ilyasu@openai.com

Abstract

Natural language understanding comprises a wide range of diverse tasks such as textual entailment, question answering, semantic similarity assessment, and document classification. Although large unlabeled text corpora are abundant, labeled data for learning these specific tasks is scarce, making it challenging for discriminatively trained models to perform adequately. We demonstrate that large gains on these tasks can be realized by *generative pre-training* of a language model on a diverse corpus of unlabeled text, followed by *discriminative fine-tuning* on each specific task. In contrast to previous approaches, we make use of task-aware input transformations during fine-tuning to achieve effective transfer while requiring minimal changes to the model architecture. We demonstrate the effectiveness of our approach on a wide range of benchmarks for natural language understanding.

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova
Google AI Language
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, Luke Zettlemoyer

We present BART, a denoising autoencoder for pretraining sequence-to-sequence models. BART is trained by (1) corrupting text with an arbitrary noising function, and (2) learning a model to reconstruct the original text. It uses a standard Tranformer-based neural machine translation architecture which, despite its simplicity, can be seen as generalizing BERT (due to the bidirectional encoder), GPT (with the left-to-right decoder), and many other more recent pretraining schemes. We evaluate a number of noising approaches, finding the best performance by both randomly shuffling the order of the original sentences and using a novel in-filling scheme, where spans of text are replaced with a single mask token. BART is particularly effective when fine tuned for text generation but also works well for comprehension tasks. It matches the performance of RoBERTa with comparable training resources on GLUE and SQuAD, achieves new state-of-the-art results on a range of abstractive dialogue, question answering, and summarization tasks, with gains of up to 6 ROUGE. BART also provides a 1.1 BLEU increase over a back-translation system for machine translation, with only target language pretraining. We also report ablation experiments that replicate other pretraining schemes within the BART framework, to better measure which factors most influence end-task performance.

Subjects: **Computation and Language (cs.CL)**; Machine Learning (cs.LG); Machine Learning (stat.ML)
Cite as: arXiv:1910.13461 [cs.CL]
(or arXiv:1910.13461v1 [cs.CL] for this version)
https://doi.org/10.48550/arXiv.1910.13461

Why Transformer models are so powerful?

- 1. Contextual Understanding, e.g., Thinking Machine
- 2. Pretraining on Massive Data (Transfer Learning)
- 3. Scalable Architecture
- 4. Powerful ecosystem, e.g., Hugging Face

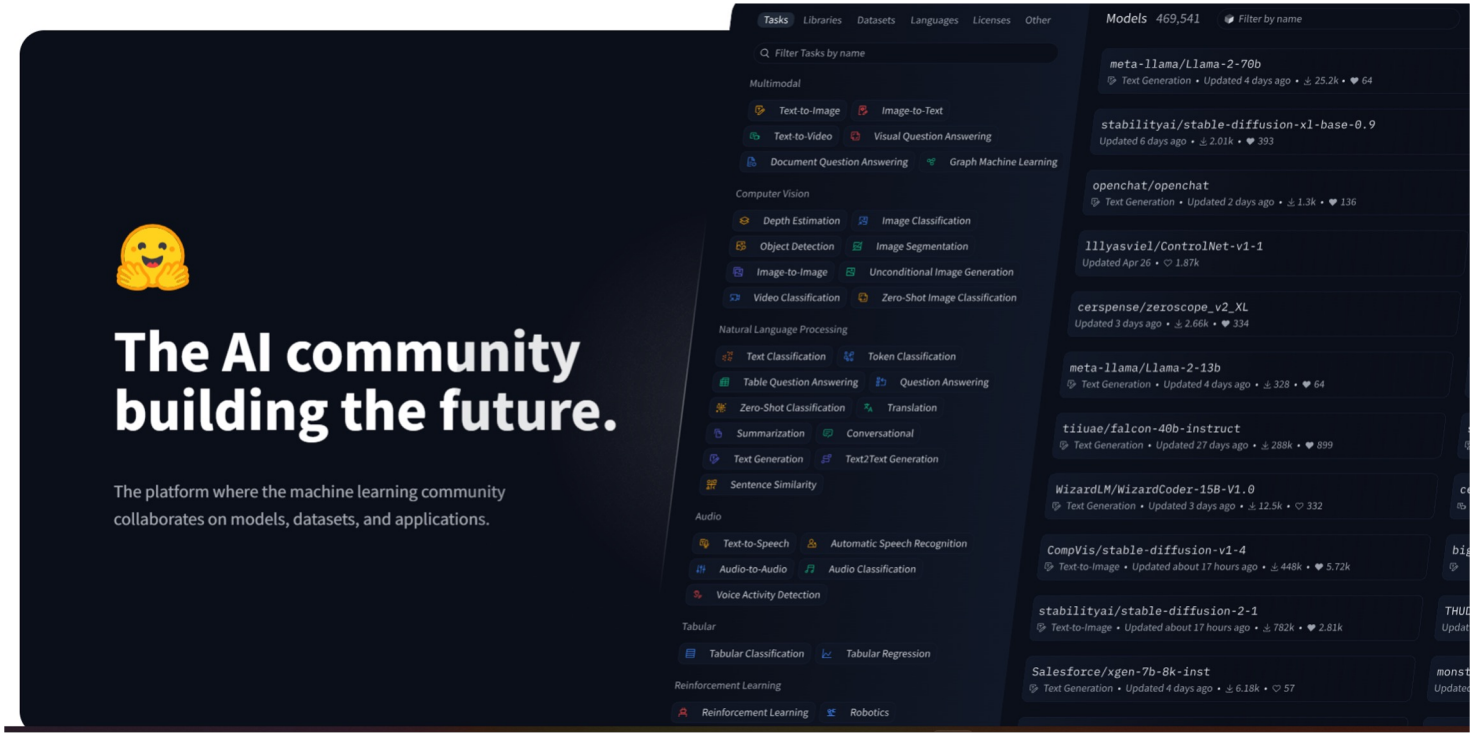
Model	Parameters	Pretraining Data
BERT-Base	110M	Wiki + Books (3.3B words)
BERT-Large	340M	Same as Base
DistilBERT	66M	Same as Base
RoBERTa	125–355M	160 GB (CommonCrawl + Books + Wiki)
ALBERT	12–235M	Same as Base

Hugging Face

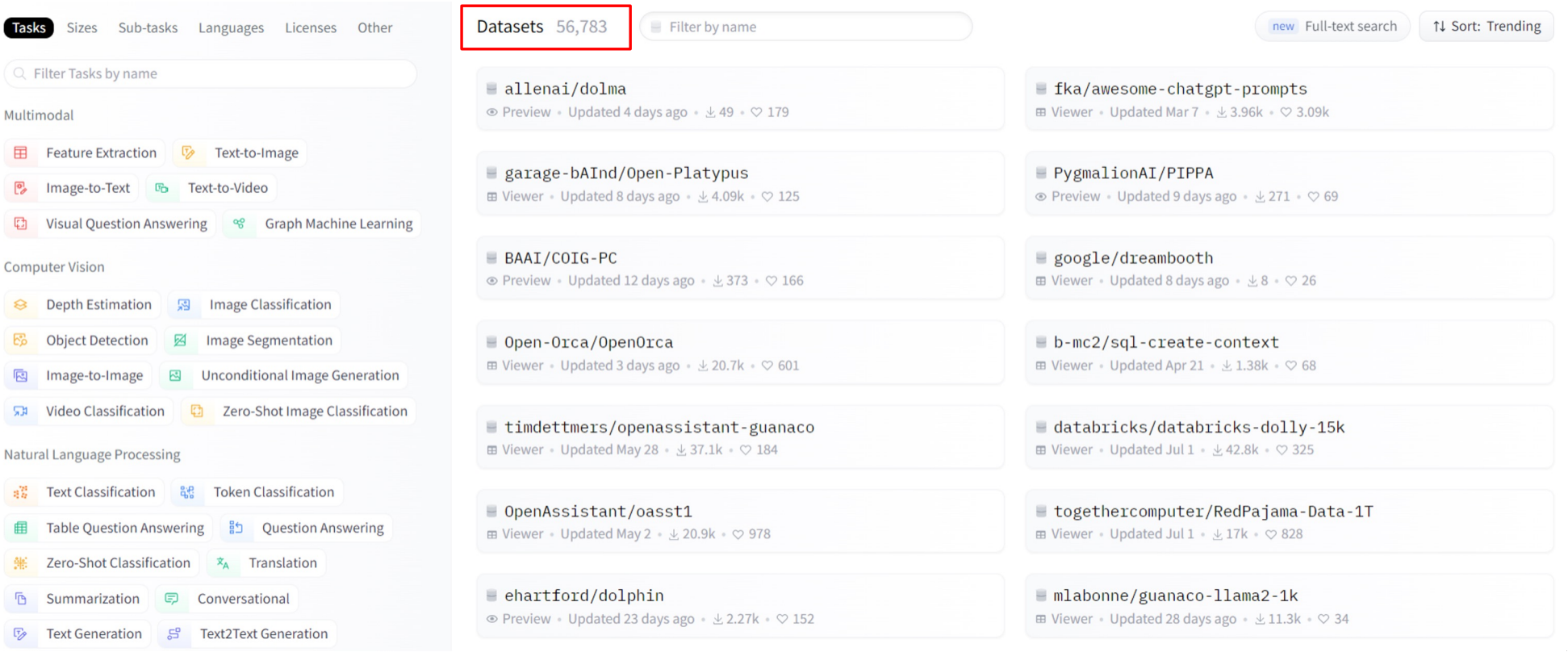
Note: Today's lecture introduces several DL models so that you can understand how to select the right one for your problem. You'll learn more in depth in the AI course next semester.

What is Huggingface?

- HuggingFace is an AI community that promotes open source contributions. It is a hub of open source models for Natural Language Processing, computer vision, and other fields where AI plays its role. Even the tech giants like Google, Facebook, AWS, Microsoft, and others use the models, datasets, and libraries.

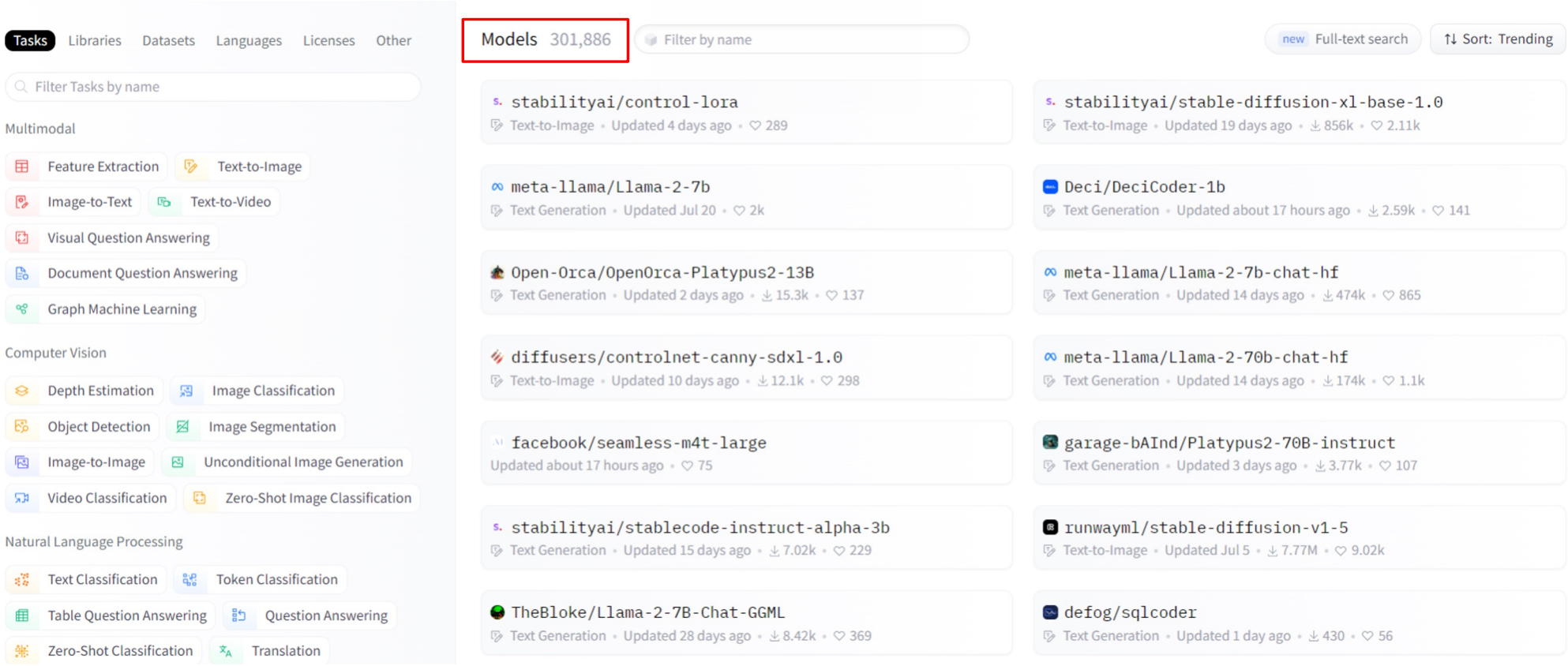


Datasets in Huggingface



Ref: https://huggingface.co/datasets?sort=trending

Models in Huggingface



Ref: https://huggingface.co/models

Examples notebook

PyTorch Examples

Natural Language Processing

Notebook	Description
Train your tokenizer	How to train and use your very own tokenizer
Train your language model	How to easily start using transformers
How to fine-tune a model on text classification	Show how to preprocess the data and fine-tune a pretrained model on any GLUE task.
How to fine-tune a model on language modeling	Show how to preprocess the data and fine-tune a pretrained model on a causal or masked LM task.
How to fine-tune a model on token classification	Show how to preprocess the data and fine-tune a pretrained model on a token classification task (NER, PoS).
How to fine-tune a model on question answering	Show how to preprocess the data and fine-tune a pretrained model on SQUAD.
How to fine-tune a model on multiple choice	Show how to preprocess the data and fine-tune a pretrained model on SWAG.

Computer Vision

Notebook	Description
How to fine-tune a model on image classification (Torchvision)	Show how to preprocess the data using Torchvision and fine-tune any pretrained Vision model on Image Classification
How to fine-tune a model on image classification (Albumentations)	Show how to preprocess the data using Albumentations and fine-tune any pretrained Vision model on Image Classification
How to fine-tune a model on image classification (Kornia)	Show how to preprocess the data using Kornia and fine-tune any pretrained Vision model on Image Classification
How to perform zero-shot object detection with OWL-ViT	Show how to perform zero-shot object detection on images with text queries
How to fine-tune an image captioning model	Show how to fine-tune BLIP for image captioning on a custom dataset
How to build an image similarity system with Transformers	Show how to build an image similarity system
How to fine-tune a SegFormer model on semantic segmentation	Show how to preprocess the data and fine-tune a pretrained SegFormer model on Semantic Segmentation
How to fine-tune a VideoMAE model on video classification	Show how to preprocess the data and fine-tune a pretrained VideoMAE model on Video Classification

Fine-tune a pretrained model: https://huggingface.co/docs/transformers/training
Ref: https://huggingface.co/docs/transformers/notebooks

+ Thank you
& any questions