Multimodal Generative AI 2025

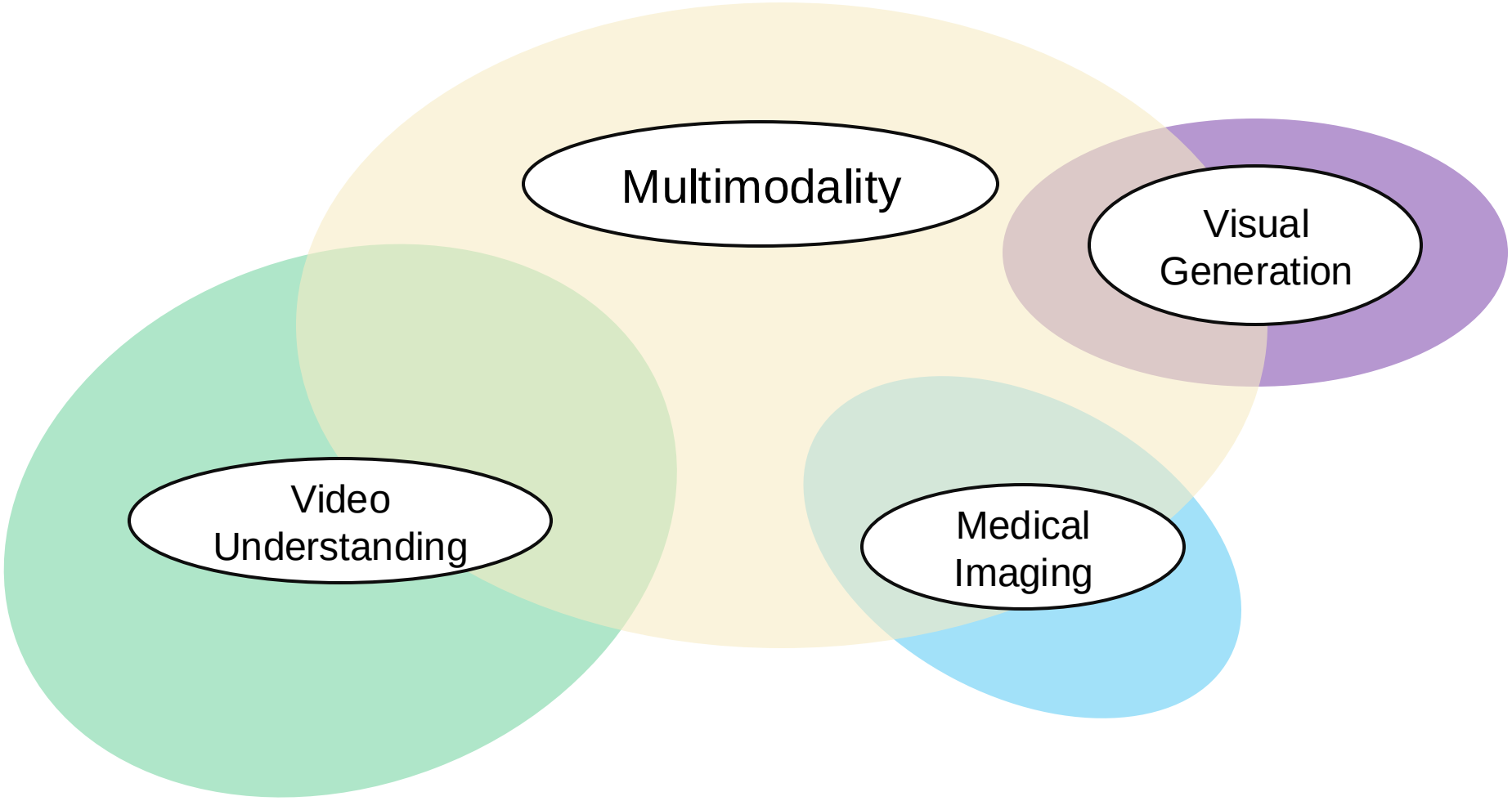
VISTA

École Polytechnique

About me

- *Assistant Professor*, 2020 –
  - VISTA Group, Ecole Polytechnique, France
  - Main CV and genAI researcher at Polytechnique
- *Research Fellow,* 2019 – 2021
- *Post-doc,* 2018 – 2020
  - Visual Geometry Group, University of Oxford, UK
  - Andrew Zisserman
- *PhD*, 2013 – 2017
  - University of Edinburgh, UK, INRIA, Grenoble, France
  - Vittorio Ferrari, Cordelia Schmid

Research agenda



Today's lecture

Part I: Practical Information

Part II: Introduction

Part III: Deep Learning architectures

Today's lecture

Part I: Practical Information

Part II: Introduction

Part III: Deep Learning architectures

Practical Information – Team

Lectures

Vicky Kalogeiton

vicky.kalogeiton@polytechnique.edu
http://vicky.kalogeiton.info/

Practical Sessions

Xi Wang
xi.wang@lix.polytechnique.fr
https://triocrossing.github.io/

Julie Mordacq
julie.mordacq@polytechnique.edu

Lucas Degeorge
lucas.degeorge@polytechnique.edu

Simo Alami
mohamed.alami-chehboune@polytechnique.edu

Lefteris Tsonis
eleftherios.tsonis@polytechnique.edu

Practical Information

Website:

https://moodle.polytechnique.fr/course/view.php?id=19324

- 9 Lectures and Practical Sessions (TD) Amphi Lagarrigue

Lectures: Mondays 14:00 – 16:00

TD's: Mondays 16:15 – 18:15

Final Exam:

- Final Presentations: Most likely March 17th 2pm-5pm

Practical Information

- **Grading:**
  - 65% by the Projects:
    - Presentation (+question answering): 40%
    - Report: 20%
    - Code & Demo: 5%
  - 5% by Participation and Answers:
    - Various questions during the class
    - Active participation in the labs
  - 30% Assignments
    - Three assignments, 10% per assignment

Practical Information

- **Final project:**
  - Main goal: design, implement and test your own method
  - Do work in a team, at most 2 people
  - We mostly evaluate performance, originality, creativity, interpretability
  - More details to come…

Practical Information

- **Collaboration policy**
  - The report needs to explicitly specify the contribution of each student
  - Both students are expected to present the project at the oral presentation and contribute equally to writing the report
  - The final projects will be checked to contain the original material. Any uncredited reuse of material (text, code, results) will be considered plagiarism and will result in zero points for the assignment / final project

- **Re-using other people's code**
  - You CAN reuse other people's code. However, you must clearly indicate in your report/presentation, what is your own code and what was provided by others (don't forget to indicate the source)
  - We expect projects balanced between implementation / experimental evaluation.
    - For example, if you implement a difficult algorithm from scratch, only a few qualitative experimental results may suffice
    - On the other hand, if you completely use someone else's implementation, we expect a strong quantitative experimental evaluation with an analysis of the obtained results and comparison with baseline methods
  - If we detect that all code is external or if we detect reusing code without reporting it, the report will get a 0

Practical Information

TD Late Policy:

- Each TD assignment is always due 1 calendar week after class (unless it's a bank holiday)
- For every extra late day **you lose 20 points** (out of 100)
- You can **not** collaborate
- Strict late policy on the final project report
- Last 4 TDs are dedicated to the project

Practical Information

Computational Resources:

For TDs: Google Colab (free, easy to use, limited resources)

For Projects:

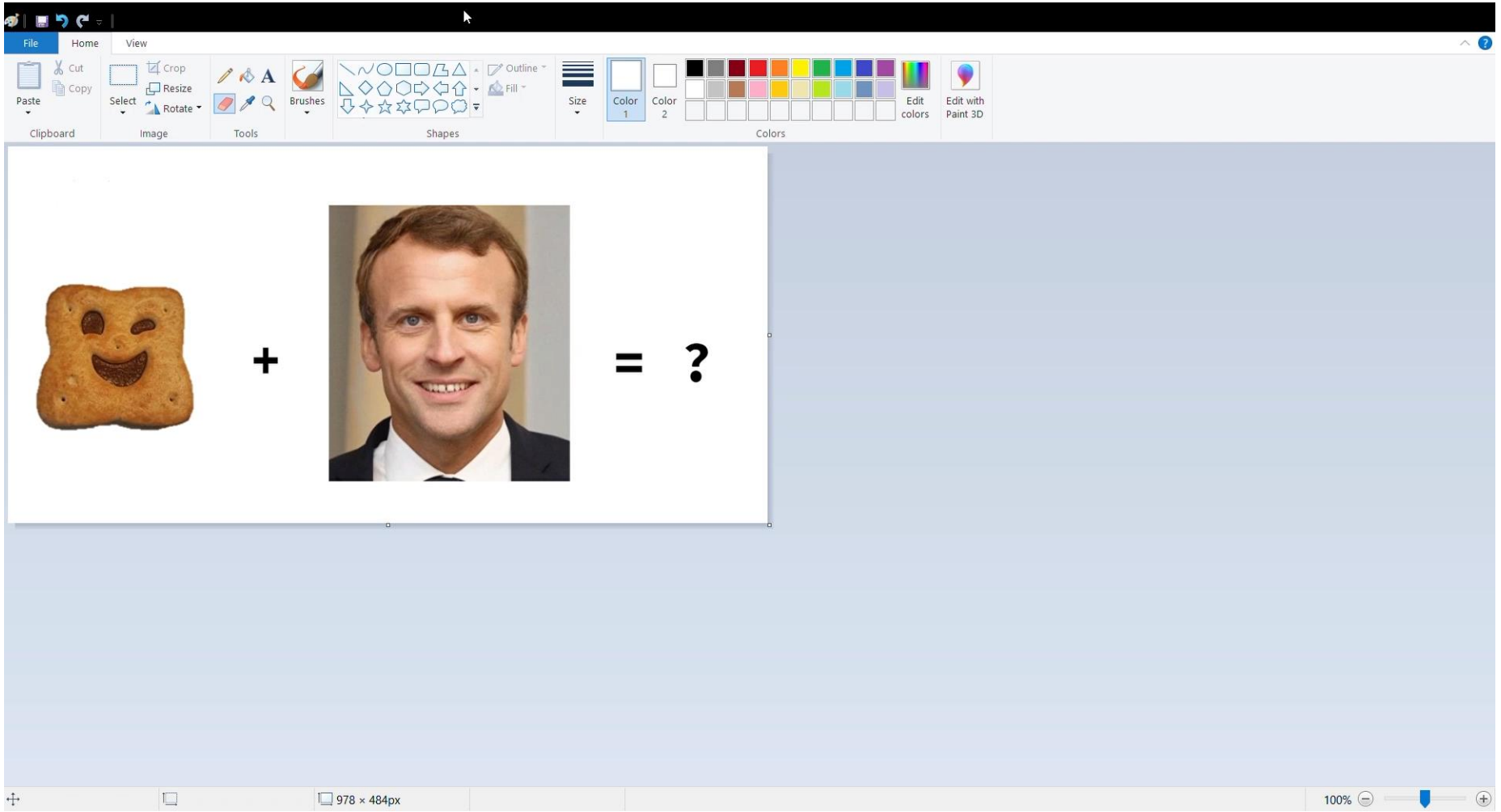
- Google Colab
- Kaggle
- Polytechnique computers
- Anything else you want!

Practical Information

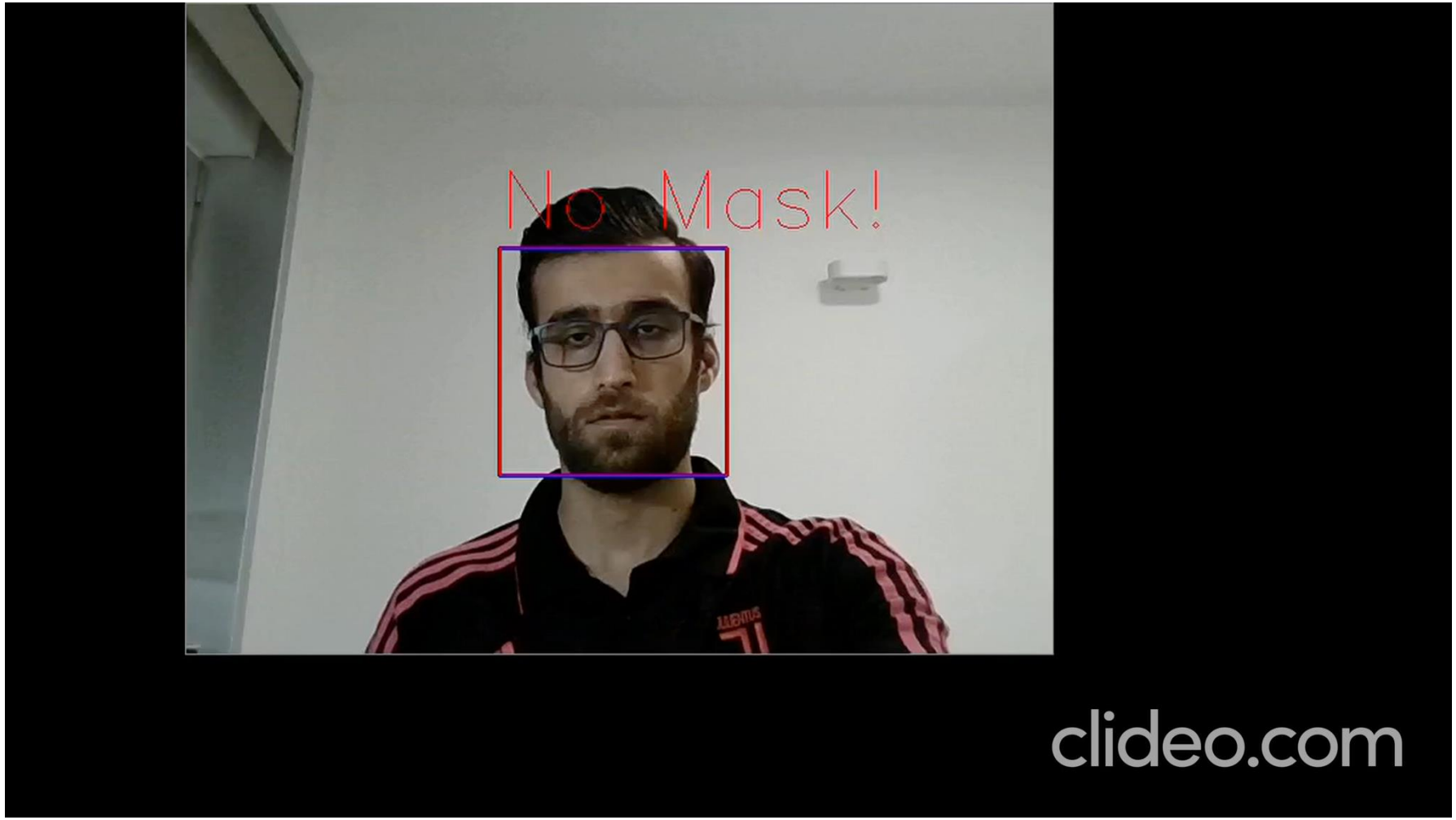
Today's lab goal:

- Familiarize yourself with
  - Python (v3 ++)
  - colab
  - PyTorch (Be able to run simple networks)

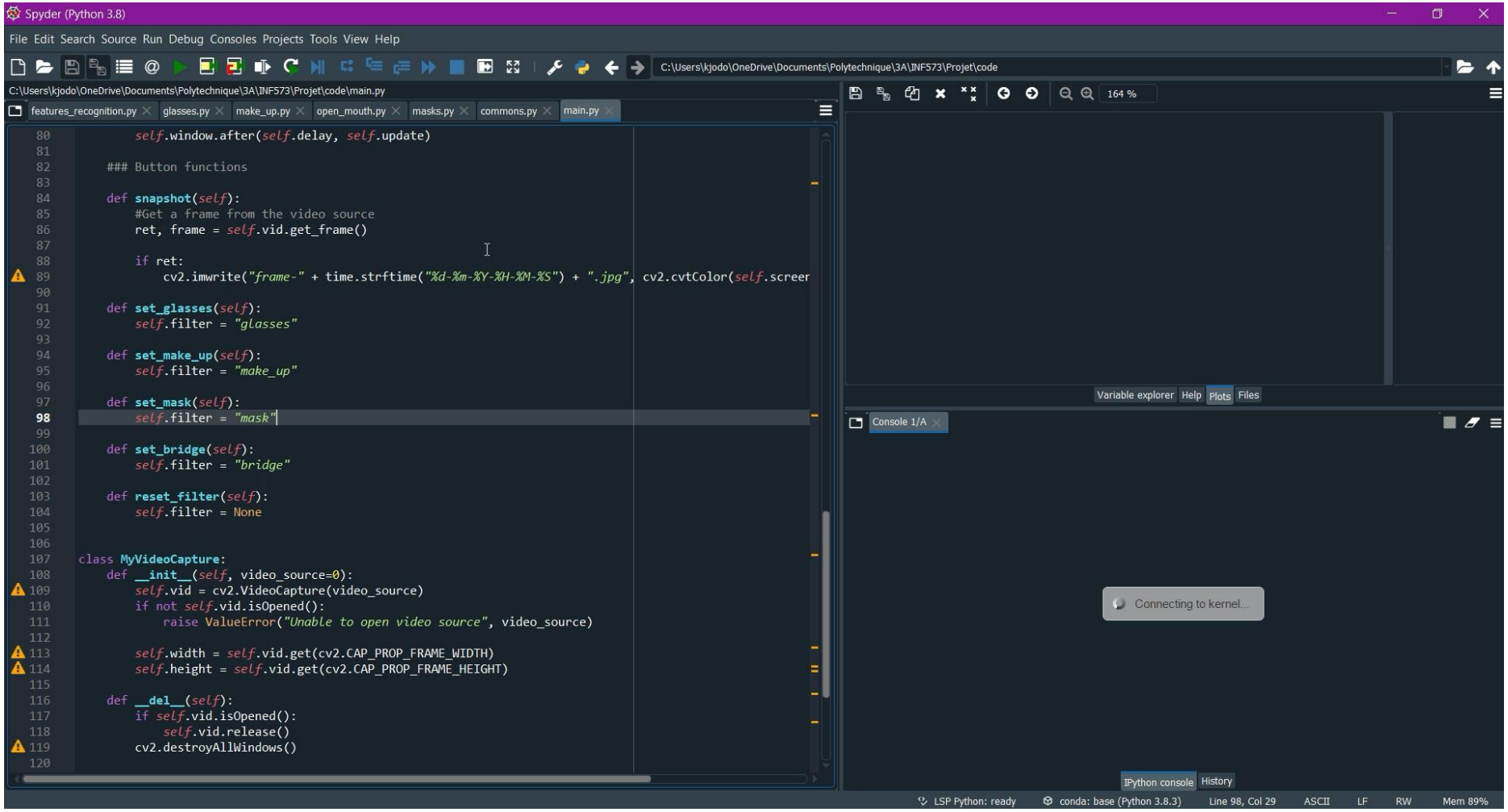
Project examples



Project examples

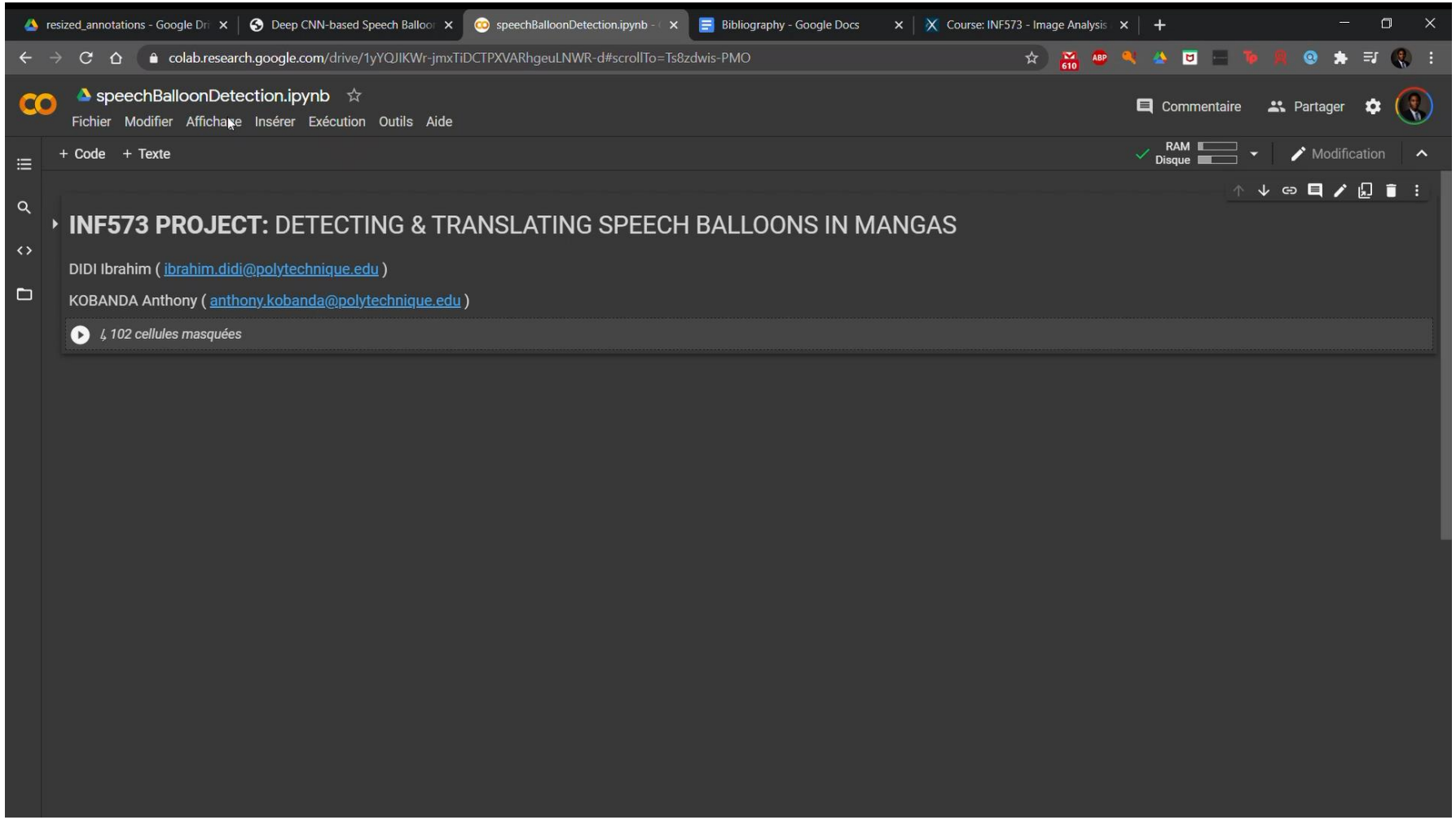


Project examples




```
80          self.window.after(self.delay, self.update)
81
82      ### Button functions
83
84      def snapshot(self):
85          #Get a frame from the video source
86          ret, frame = self.vid.get_frame()
87
88          if ret:
89              cv2.imwrite("frame-" + time.strftime("%d-%m-%Y-%H-%M-%S") + ".jpg", cv2.cvtColor(self.screer
90
91      def set_glasses(self):
92          self.filter = "glasses"
93
94      def set_make_up(self):
95          self.filter = "make_up"
96
97      def set_mask(self):
98          self.filter = "mask"
99
100     def set_bridge(self):
101         self.filter = "bridge"
102
103     def reset_filter(self):
104         self.filter = None
105
106
107 class MyVideoCapture:
108     def __init__(self, video_source=0):
109         self.vid = cv2.VideoCapture(video_source)
110         if not self.vid.isOpened():
111             raise ValueError("Unable to open video source", video_source)
112
113         self.width = self.vid.get(cv2.CAP_PROP_FRAME_WIDTH)
114         self.height = self.vid.get(cv2.CAP_PROP_FRAME_HEIGHT)
115
116     def __del__(self):
117         if self.vid.isOpened():
118             self.vid.release()
119         cv2.destroyAllWindows()
120
```

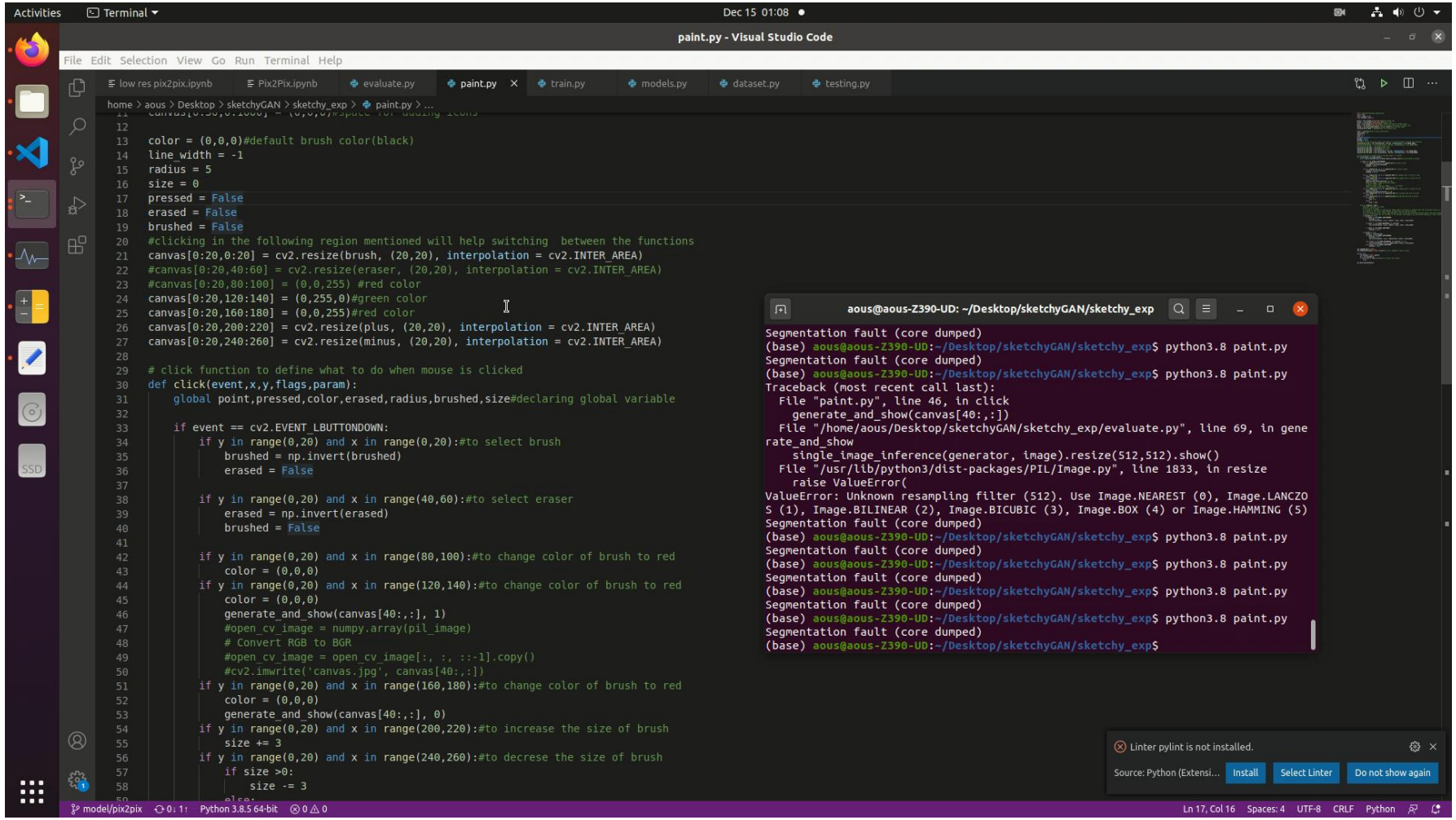
Project examples



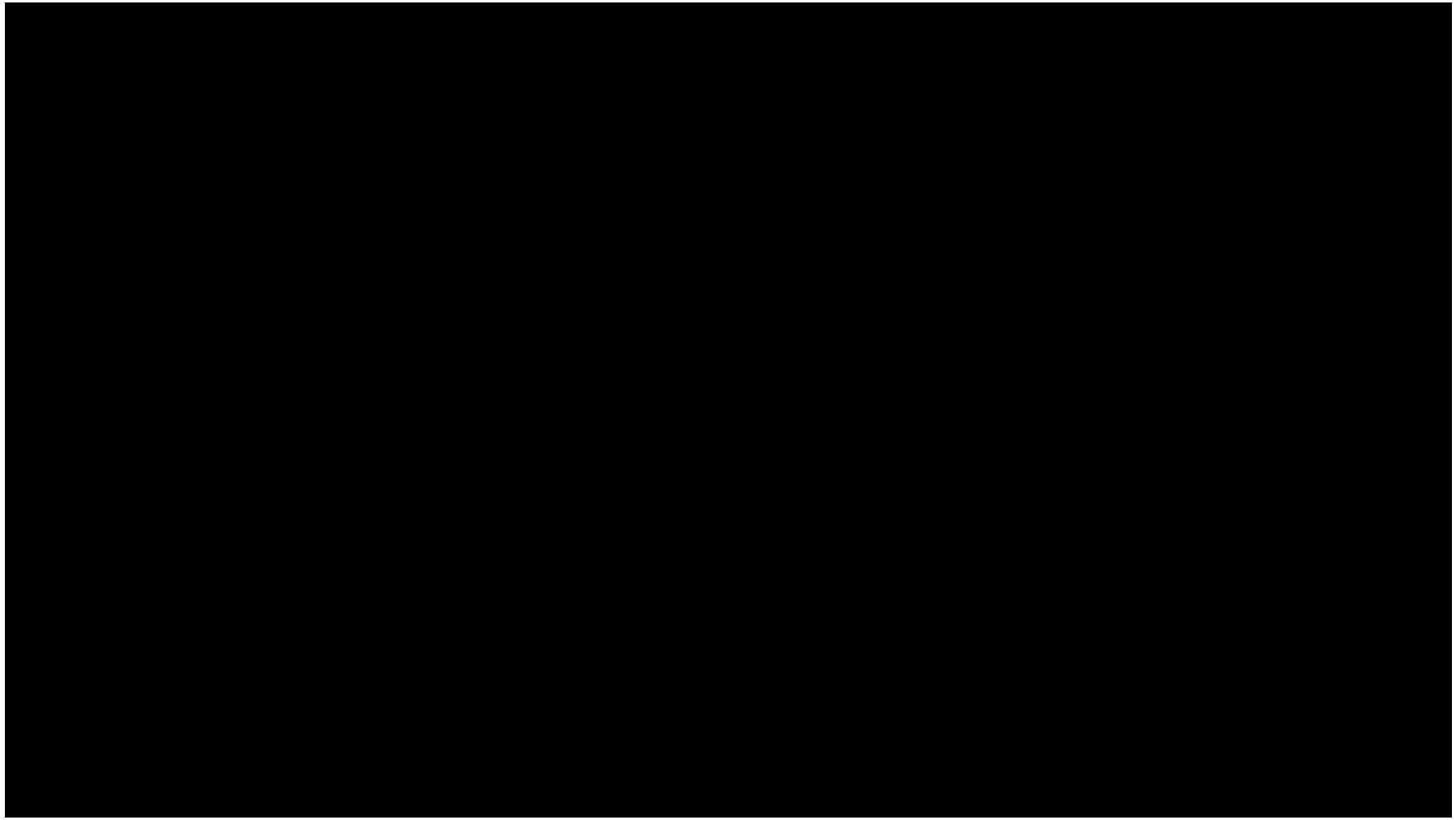
Project examples

Project examples

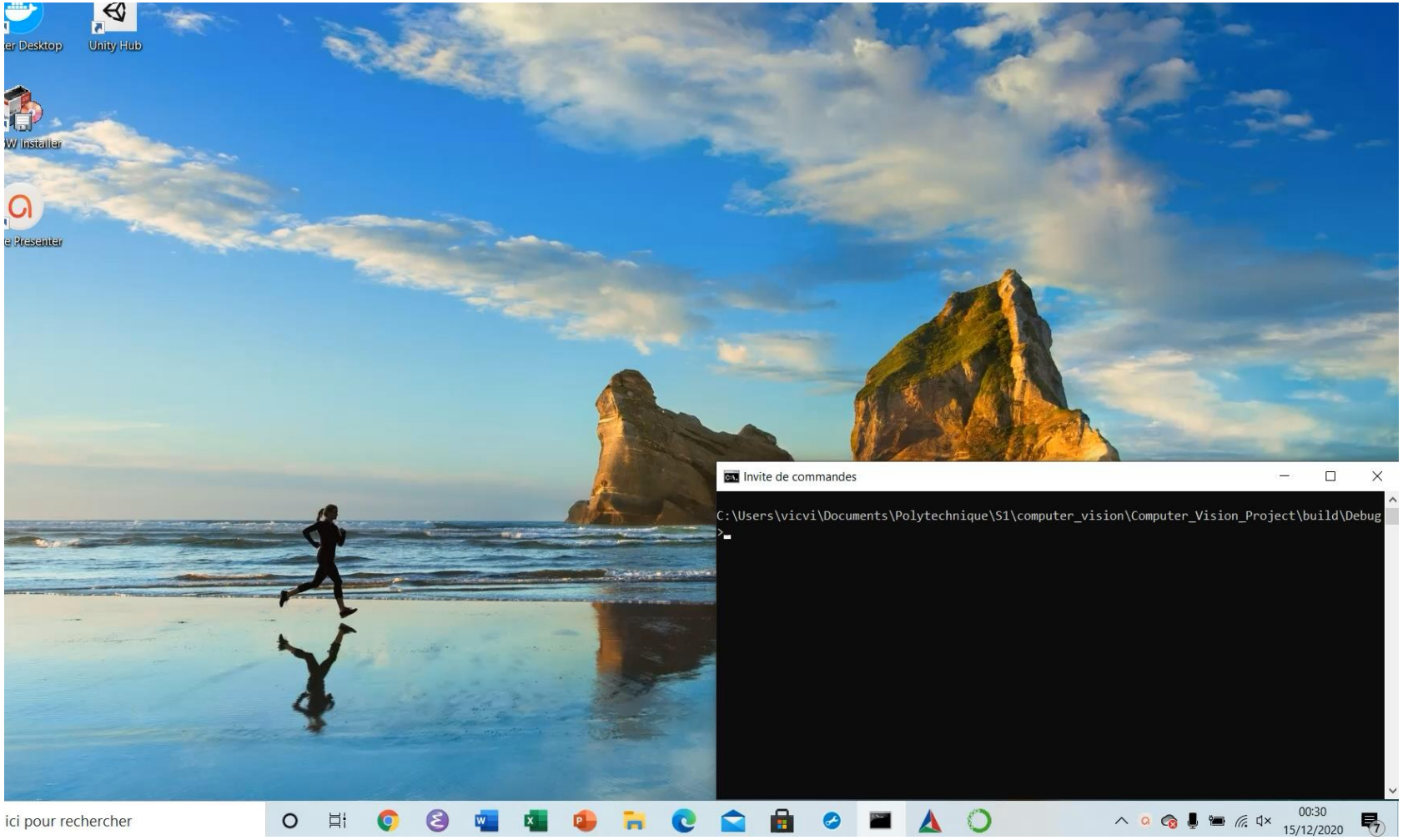
Project examples



Project examples



Project examples



Project examples

Today's lecture

Part I: Practical Information

Part II: Introduction

Part III: Deep Learning architectures

Part II: Outline

- Modern Computer Vision
- Tasks
- Multimodal learning

What is modern Computer Vision?

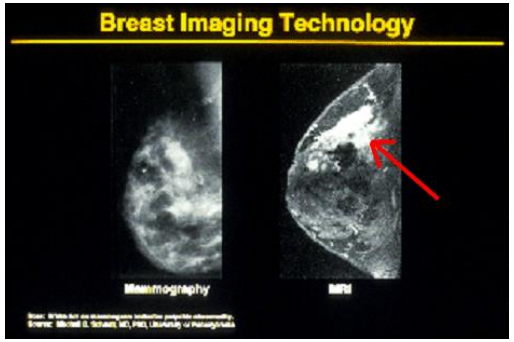
Example

- Nurse serving tea
- Old man having breakfast with a friend
- He doesn’t seem to like the tea

Why is it important?

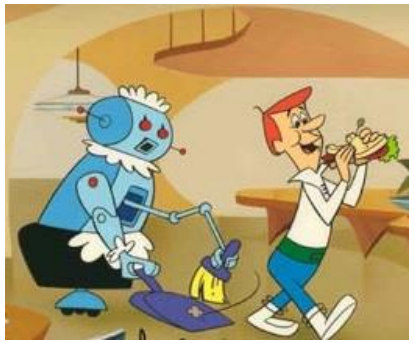
Why does it matter?

Safety



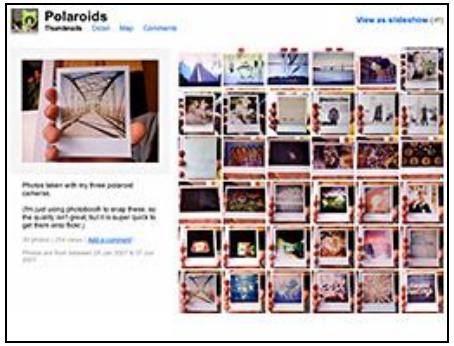
Health

Security



Comfort

Fun



Access

Modern applications
Motion capture and animation

https://www.youtube.com/watch?v=A-mYK2_XLP8

Modern applications
Automatic story telling

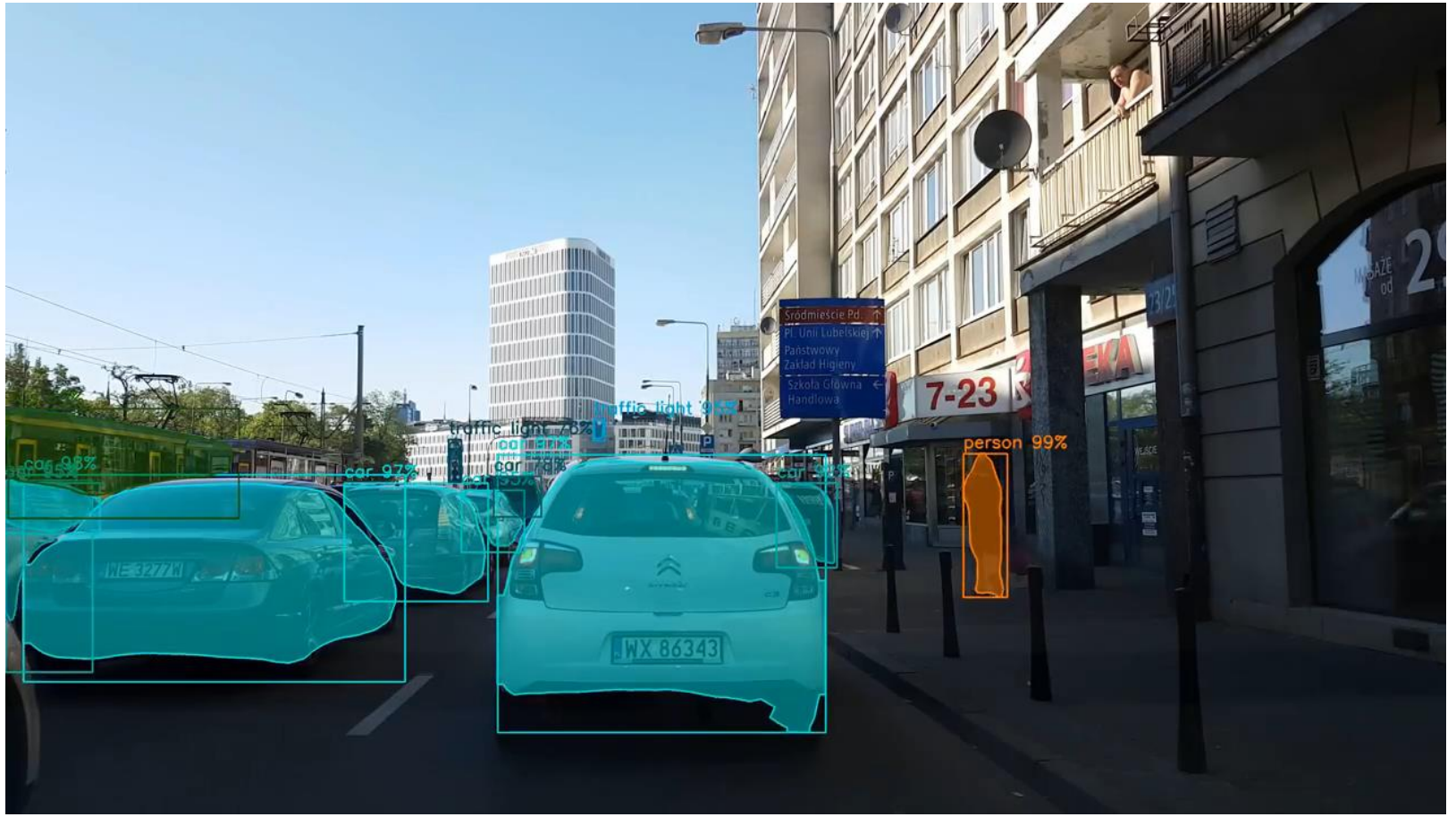
Grand entry of the king's horses and men. **ARYA**, wearing a helm and cloak, **pushes her way into a tall wagon for a better look**....

Grand entry of the king's horses and men. ARYA, wearing a helm and cloak, pushes her way into a tall wagon for a better look....
In rides JOFFREY, followed by the HOUND

Modern applications
Search images or videos

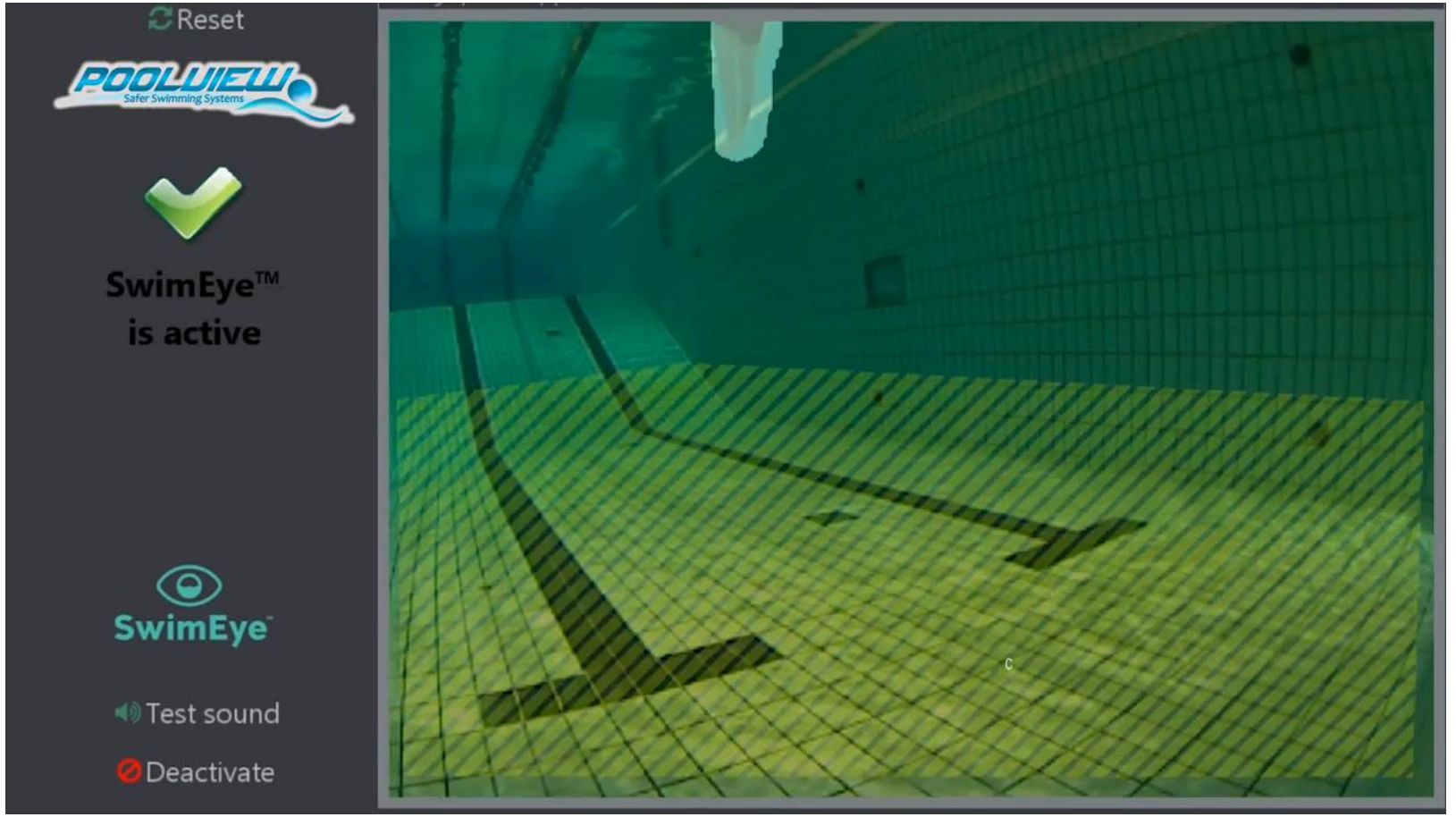
https://www.youtube.com/watch?v=dTFTSkCs-jk

Modern applications
Object detection and tracking



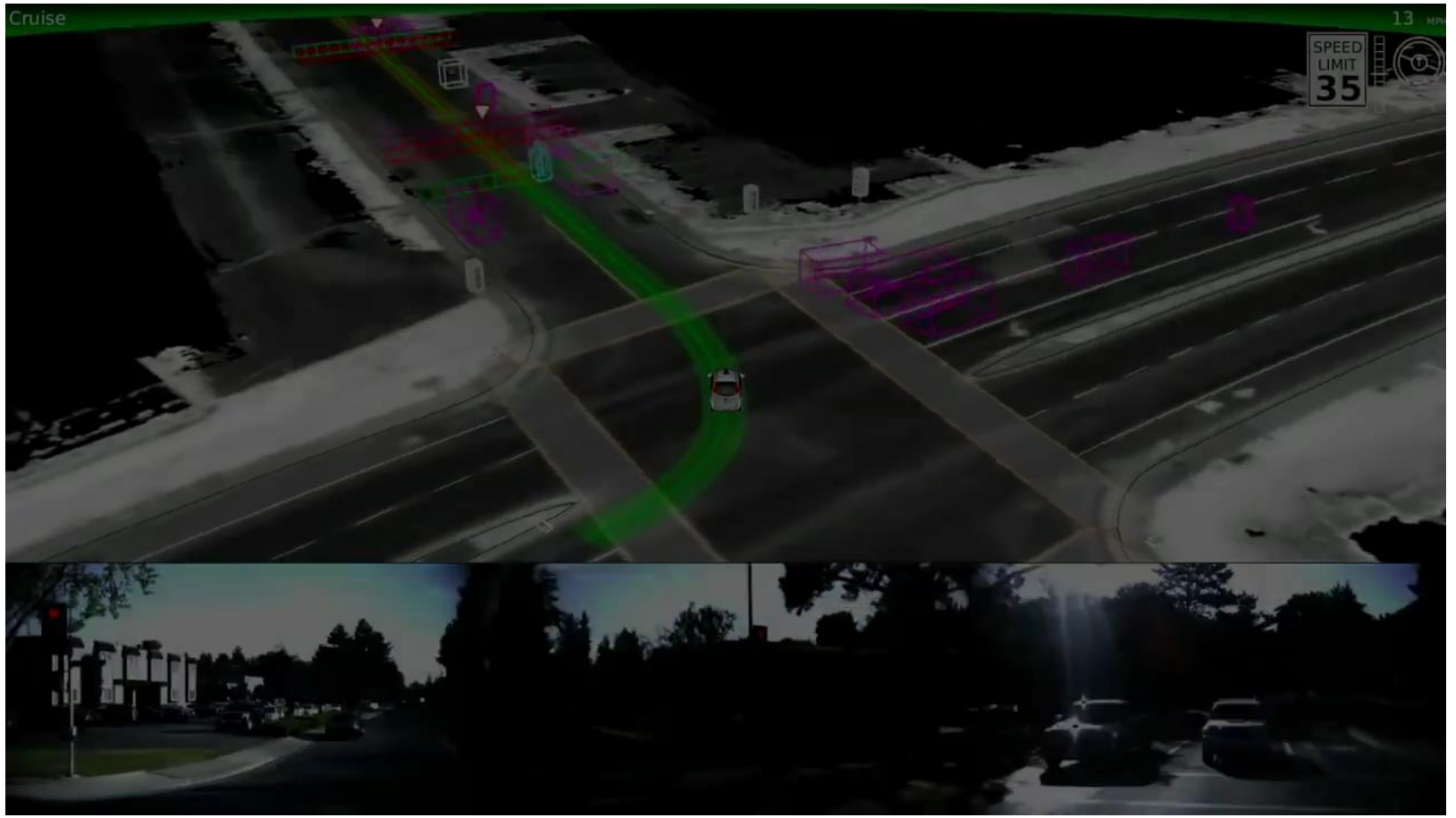
Kaiming He, et al. ICCV 2017

Modern applications
Danger detection



https://www.youtube.com/watch?v=7Pda2bsPr5o

Modern applications
Self-driving cars



https://www.ted.com/talks/chris_urmson_how_a_driverless_car_sees_the_road

Computer Vision Tasks

Image Classification

Input: Image

Output: Assign Image to one of a fixed set of categories

Cat
Dog
Deer
Bird
Car

Object detection

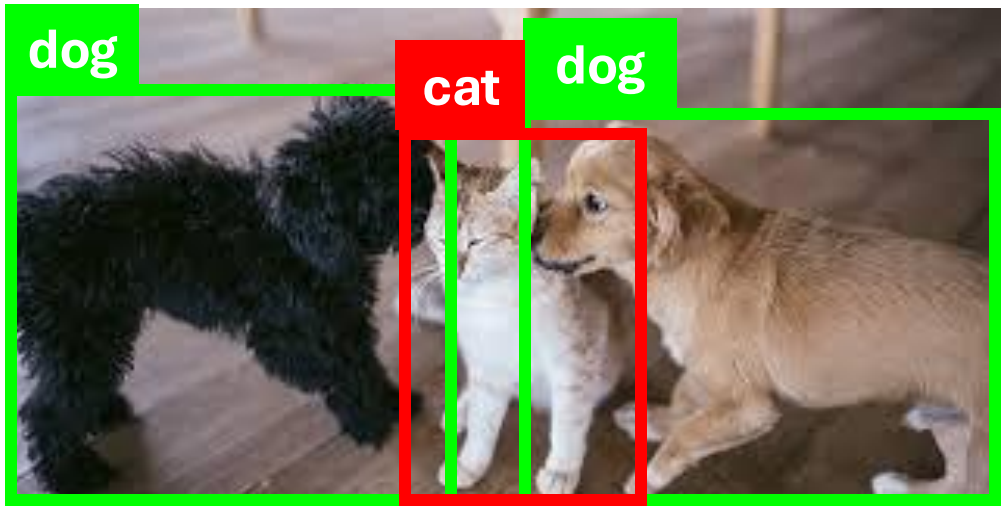
Input: Image

Output: Image category (i.e.. class) and location of the object with a rectangle, i.e. bounding box

Object detection

Input: Image

Output: Image category (i.e.. class) and location of the object with a rectangle, i.e. bounding box



Combined recognition and localization problem

Semantic Segmentation

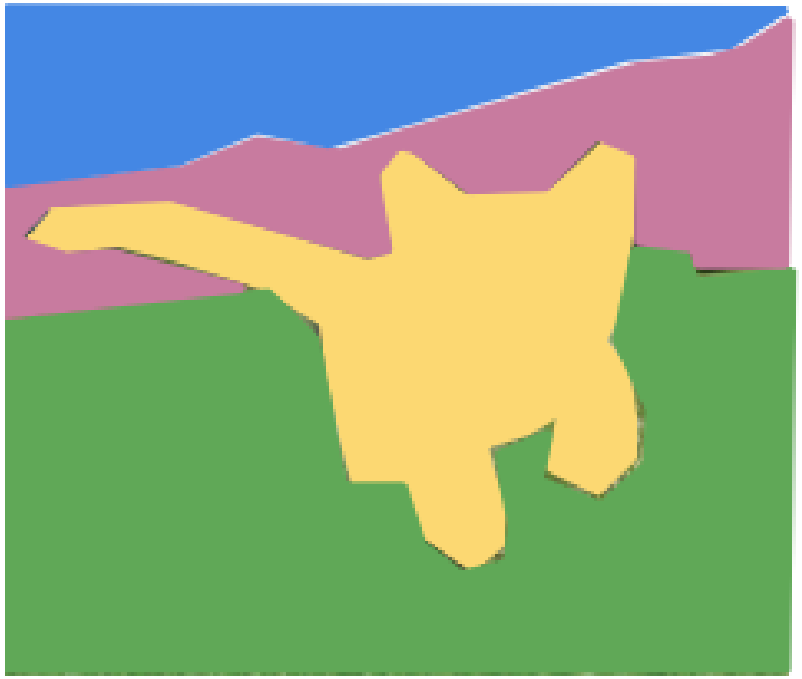
Input: Image

Output: Assign an category (i.e. class) at each pixel of the image

Semantic Segmentation

Input: Image

Output: Assign an category (i.e. class) at each pixel of the image



Grass, Cat, Tree, Sky

Instance Segmentation

Input: Image

Output: prediction of object instances and their per-pixel segmentation masks

Instance Segmentation

Input: Image

Output: prediction of object instances and their per-pixel segmentation masks

Dog, Dog, Cat

Pose estimation

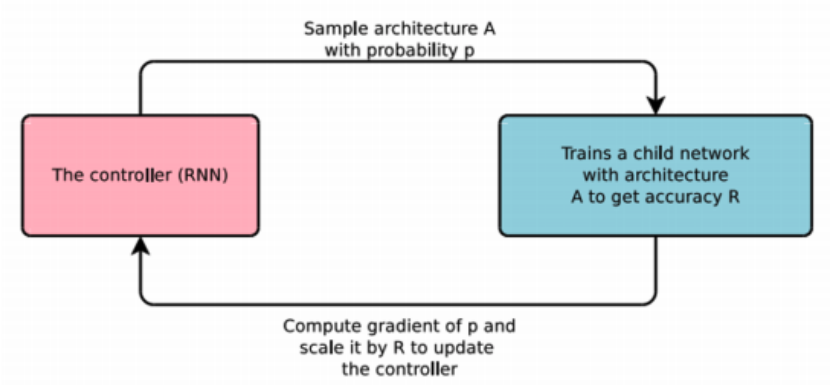
Input: Image

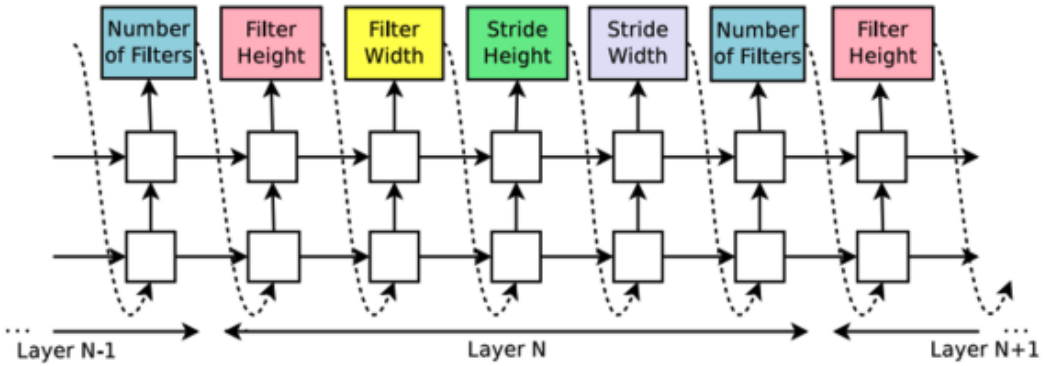
Output: localizing anatomically key-points or “parts”

Neural Architecture Search

Automatic design of NN

- One network (**controller**) outputs NN
- Sample child networks from controller and train them
- After training many child networks, gradient step on controller network (using policy gradient)
- Controller learns to output good NN





Video classification

Input: Video

Output: prediction of dominant (human) action in video

Sitting down

Video detection

Input: Video

Output: action prediction & and temporal localization of action

Sitting down

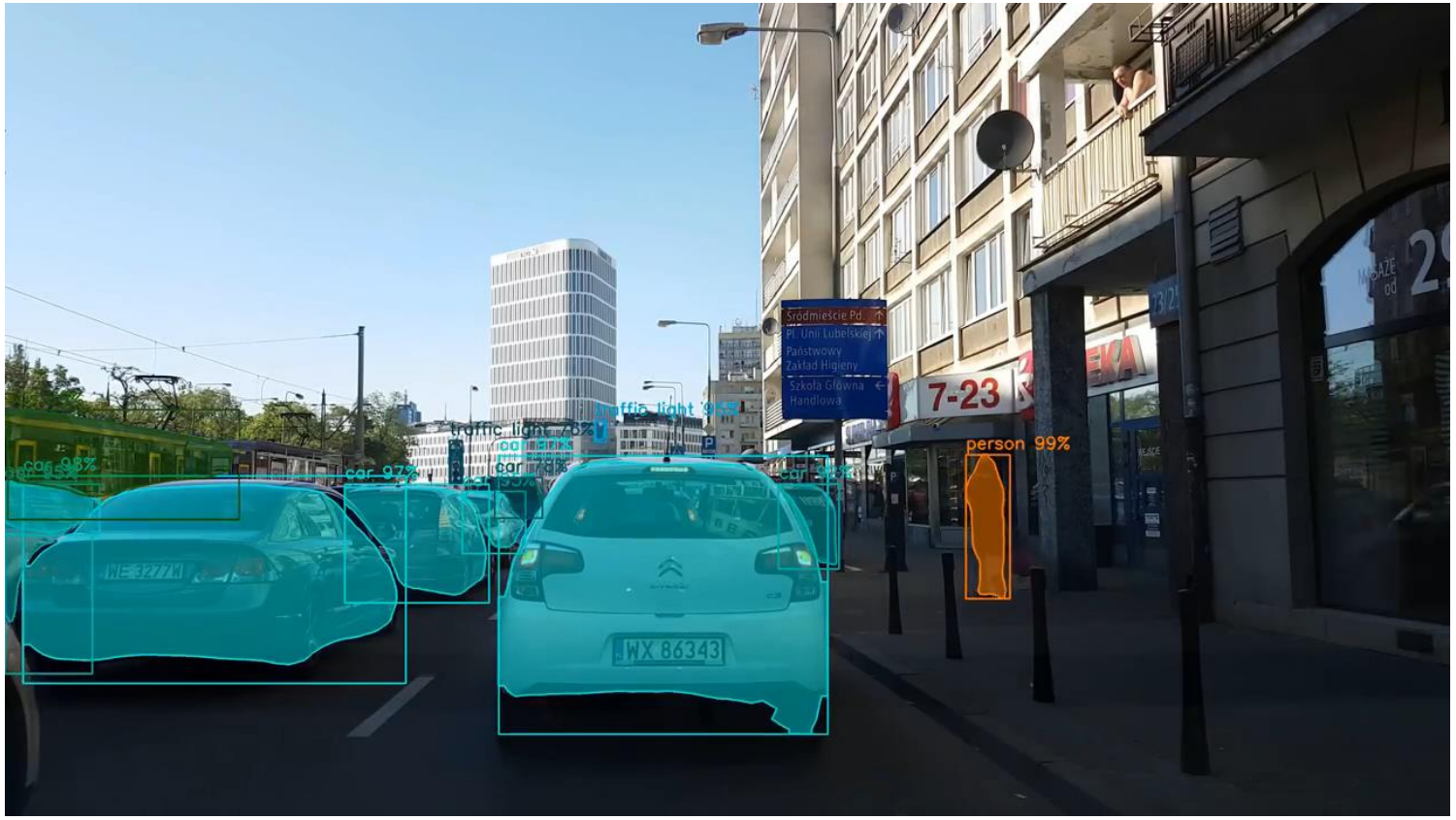
Spatio-temporal Localization

Input: Video

Output: action prediction & and *spatio-temporal* localization of action

Sitting down

Object tracking



Multi-modal visual understanding

Active Speaker Detection

Examples from *Sesame Street*

Blue = active speaker
Red = inactive speaker

Image Captioning/ Visual Question Answering

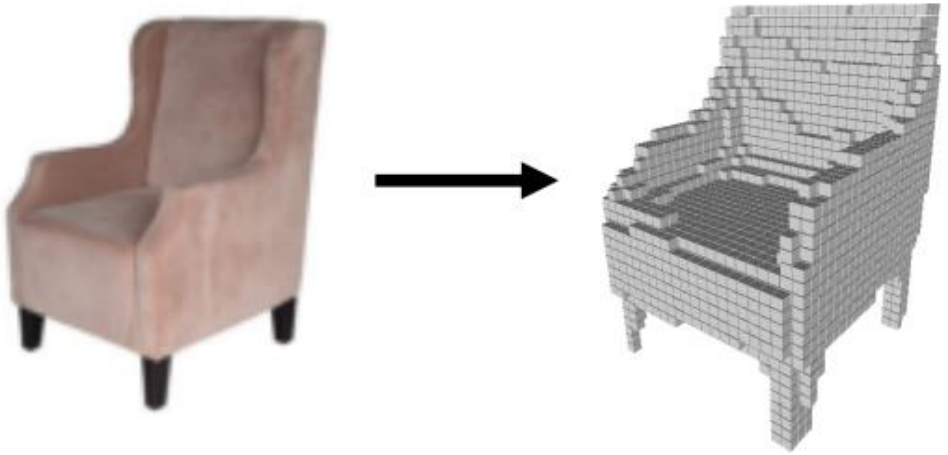
Input: Image

Output: sentence (i.e. caption) describing the image

A dog (Ketchup) is jumping in the grass with a ball

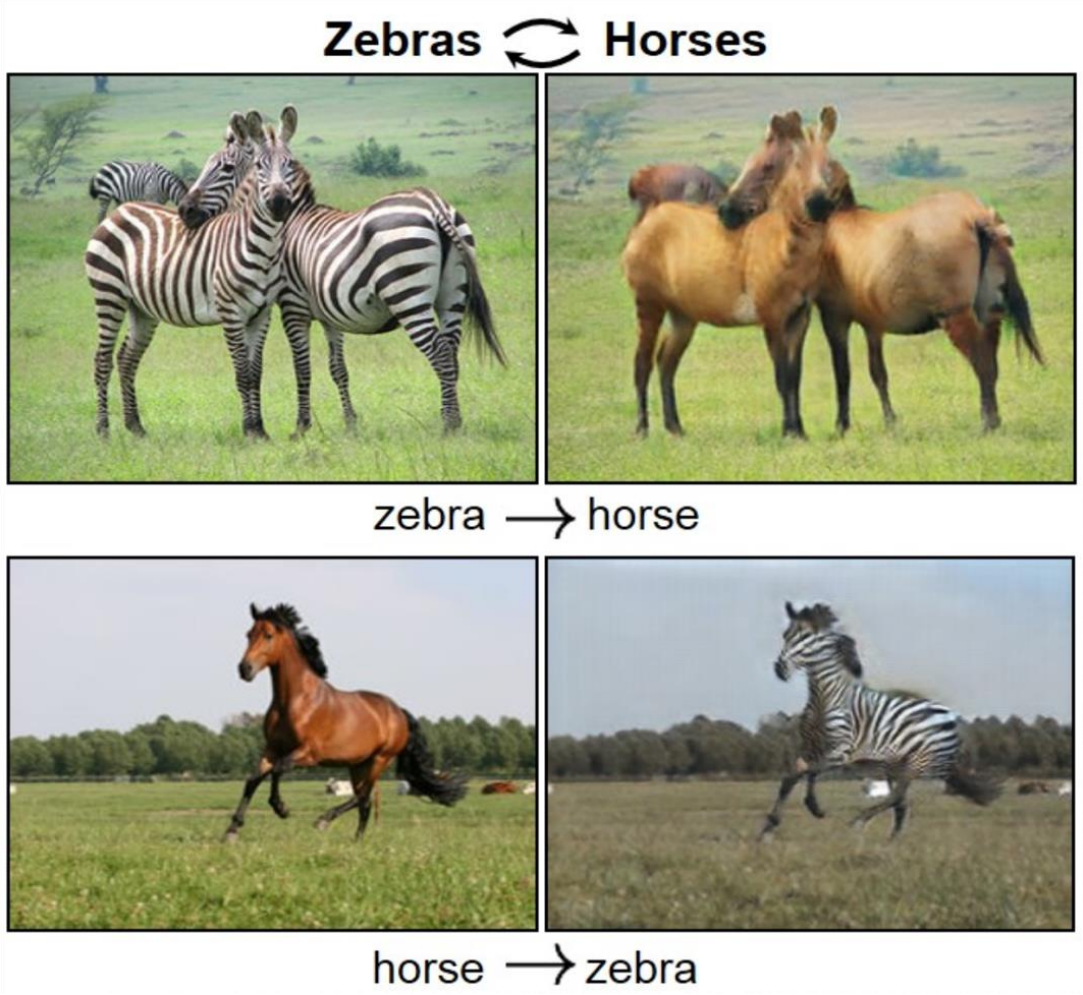
3D Vision

Input: Image **Output:** 3D shape **Processing:** category (i.e.class)



Chair

Image synthesis

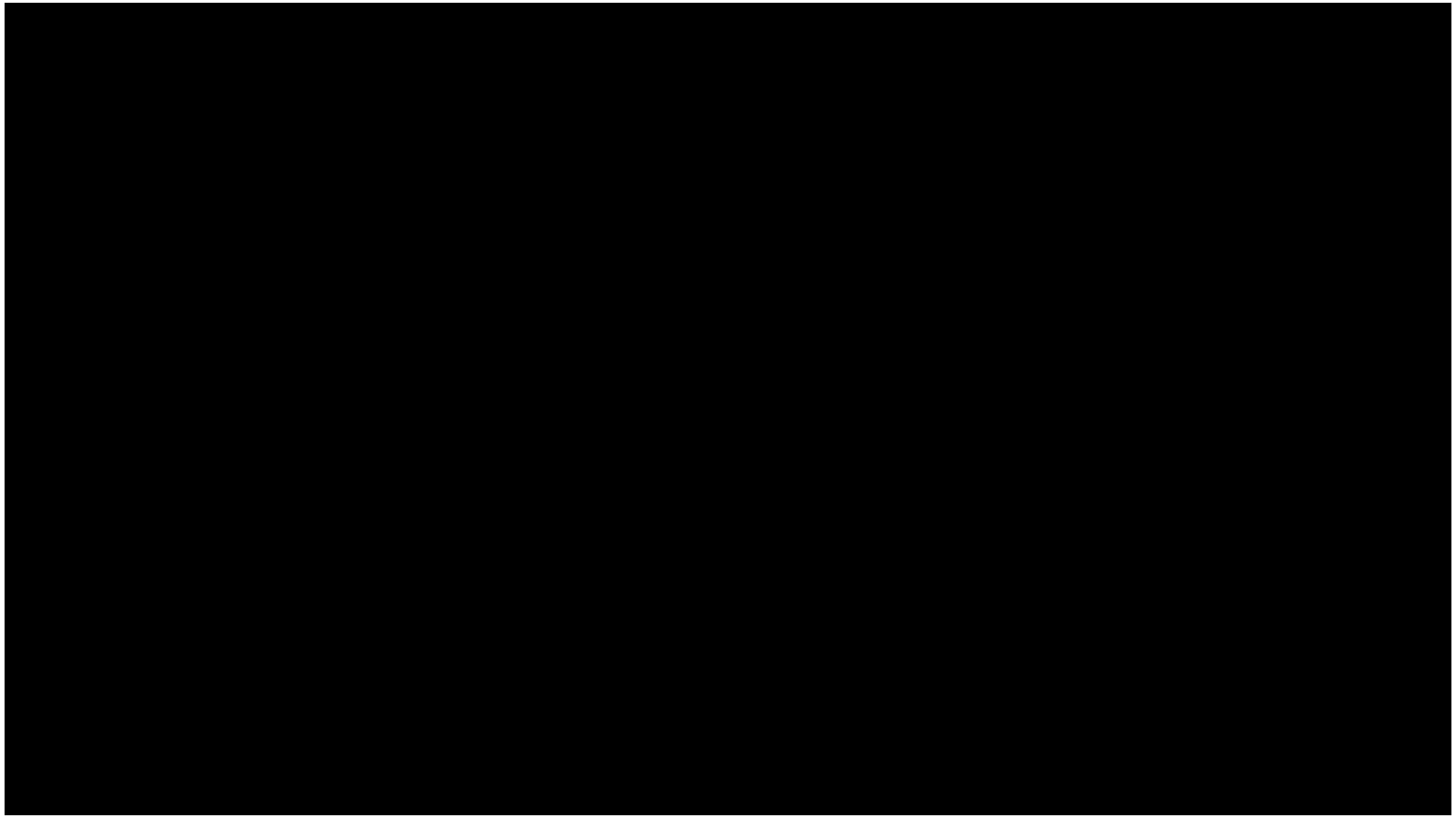


Style transfer

Jun-Yan Zhu, Taesung Park et. al. ICCV 2017

https://www.youtube.com/watch?v=9reHvktowLY

Reinforcement Learning



Multimodal learning

Multimodal learning

- Multimodal learning refers to the process of **learning representations from different types of modalities using the same model**
- Different modalities are characterized by different statistical properties
- Machine learning: input modalities include images, text, audio, etc
- **Nowadays: mostly images and language (text)**

Vision-language tasks

Classification

Retrieval

Generation

Vision-language Classification tasks

- Multimodal Affective Computing (MAC)
  - interprets visual affective activity from visual and textual input. In a way, it can be seen as multimodal sentiment analysis.
- **Natural Language for Visual Reasoning** (**NLVR**)
  - determines if a statement regarding a visual input is correct or not.

Vision-language Retrieval tasks

- **Visual Retrieval** (**VR**)
  - retrieves images based only on a textual description
- Vision-Language Navigation (VLN)
  - is the task of an agent navigating through a space based on textual instructions
- Multimodal Machine Translation (MMT)
  - involves translating a description from one language to another with additional visual information

Vision-language Generation tasks

- Visual Question Answering (VQA)
  - refers to the process of providing an answer to a question given a visual input (image or video).
- Visual Captioning (VC)
  - generates descriptions for a given visual input.
- Visual Commonsense Reasoning (VCR)
  - infers common-sense information and cognitive understanding given a visual input.
- **Visual Generation (VG)**
  - generates visual output from a textual input.

Classification

Vision-language Classification tasks
BERT-like architectures

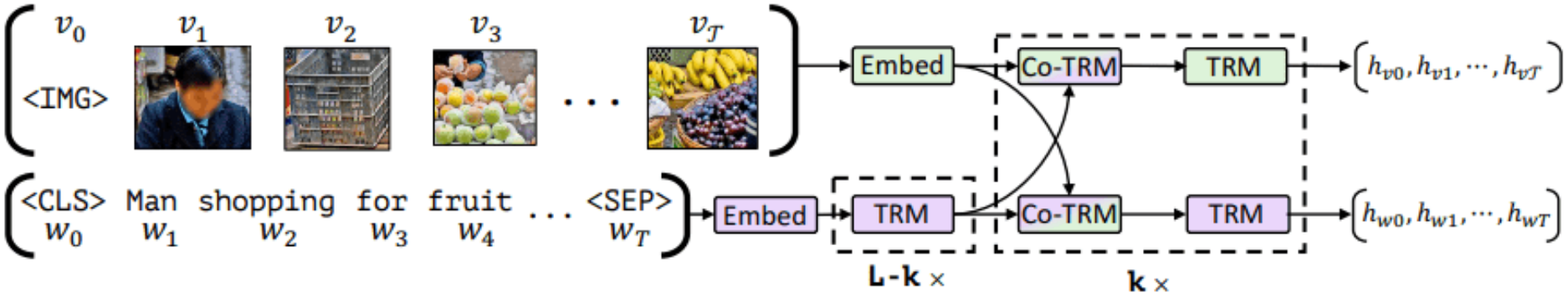
BERT-like architectures

- **Idea**: process language and images at the same time with a transformer-like architecture

- Two categories:
  - Two-stream models
    - process text and images using two separate modules
      e.g., ViLBERT, LXMERT
  - Single-stream models
    - Encode both modalities within the same module
      e.g., VisualBERT, VL-BERT, UNITER

Two-stream models
ViLBERT

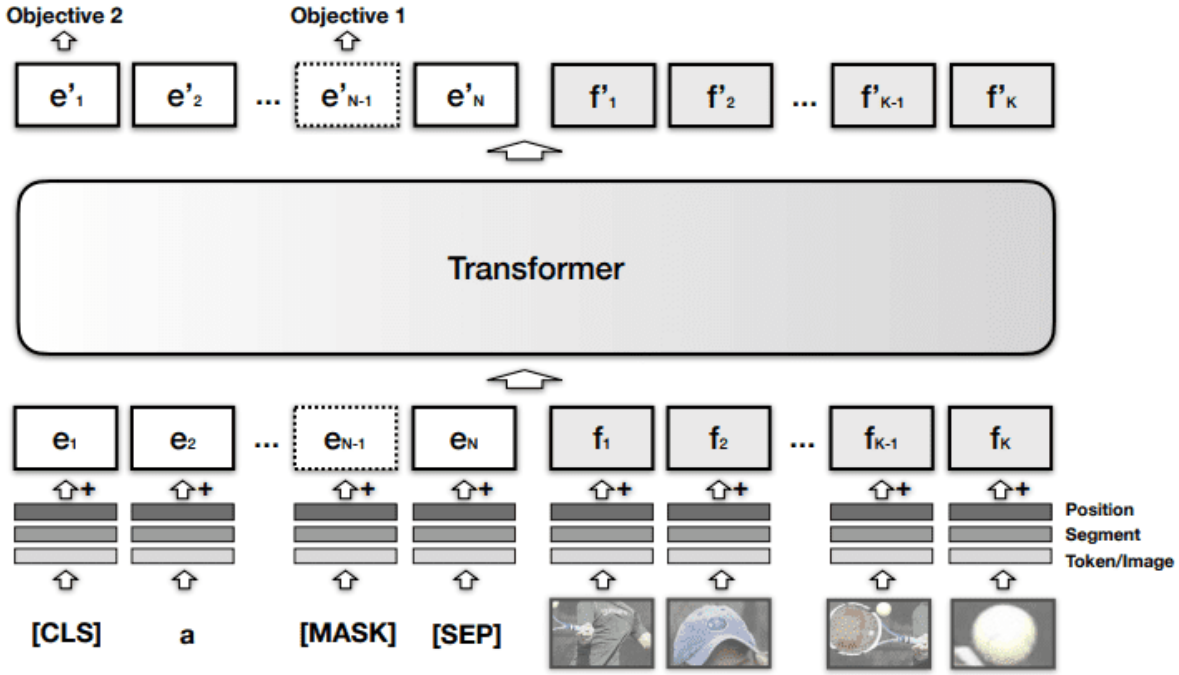
- **Text** stream (purple)
  - the weights are set by pretraining the model on standard text corpus
- **Image** stream (green): Object detection pipeline (Faster R-CNN)
- **Training**: dataset of image-text pairs
- **Objective**: understand the relationship between text and images



Single-stream models
VisualBERT

- VisualBERT: combines image + language with transformer
- Uses self-attention to discover alignments between them

- Add visual embedding to BERT:
  - A visual feature representation of the region produced by a CNN
  - A segment embedding that distinguishes image from text embeddings
  - A positional embedding to align regions with words if provided in the input



VL modeling

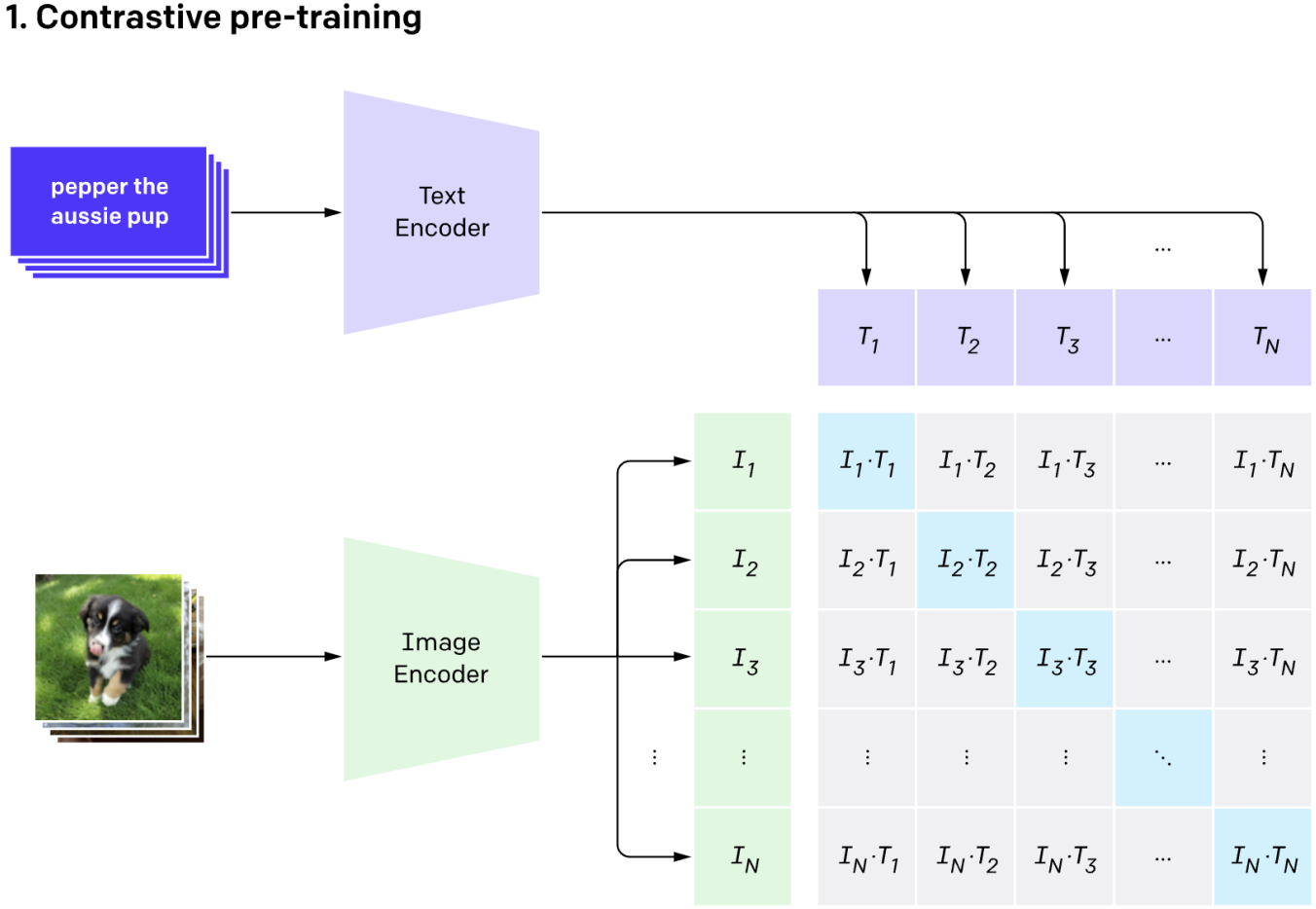
- **Unsupervised VL Pre-training**
  - Pre-training without paired image-text data but with a single modality
  - During fine-tuning, the model is fully-supervised
- **Multi-task Learnin**g
  - Joint learning across multiple tasks → transfer the learnings from one task to another
- **Contrastive Learning (e.g. CLIP)**
  - Learn visual-semantic embeddings in a self-supervised way
  - Idea: learn such an embedding space in which similar pairs stay close to each other while dissimilar ones are far apart
- **Zero-shot learning**
  - Generalize at inference time on samples from unseen classes

Classification

Vision-language Classification tasks
VL with contrastive learning

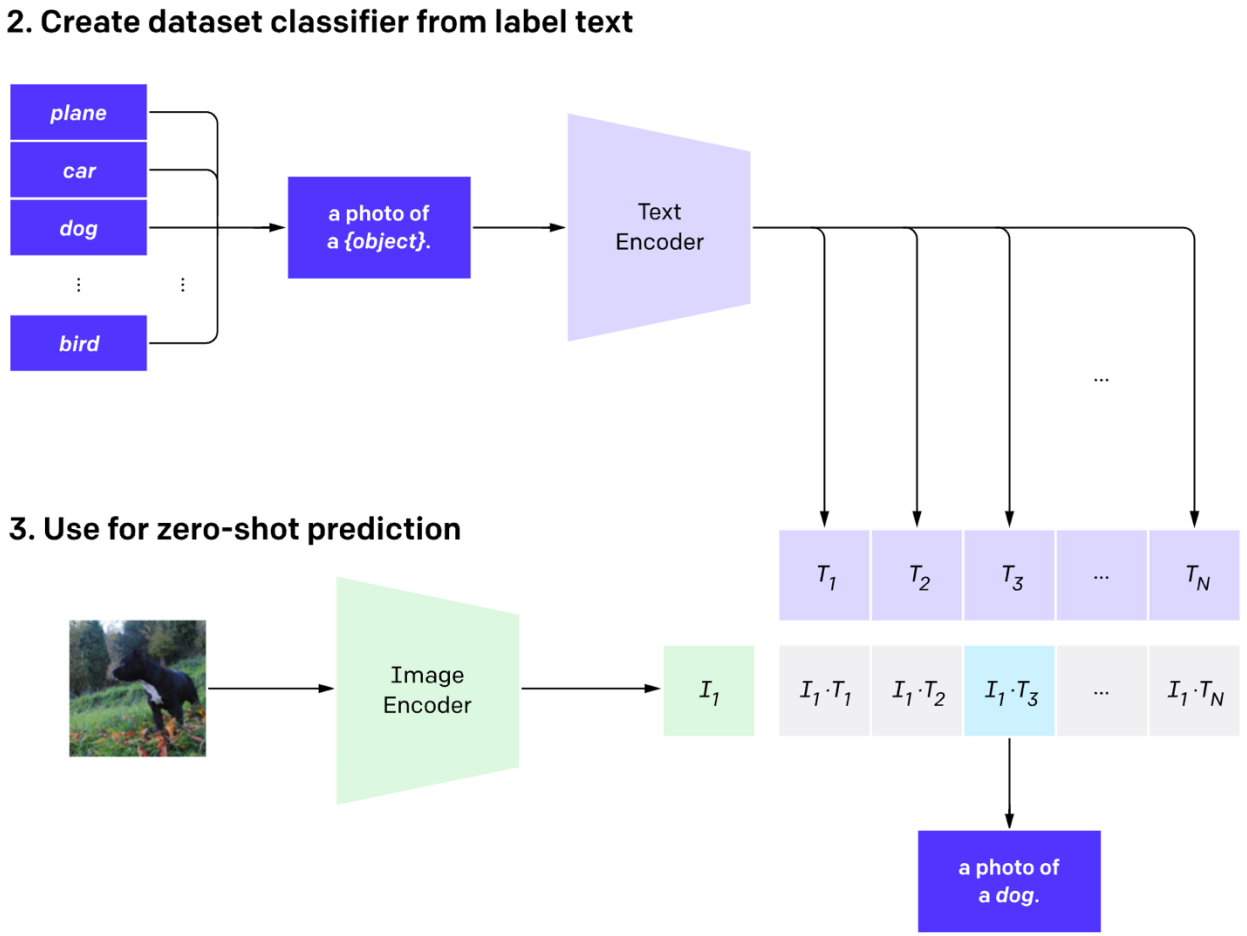
CLIP

5 Januray 2021
12+k citations in 3 years



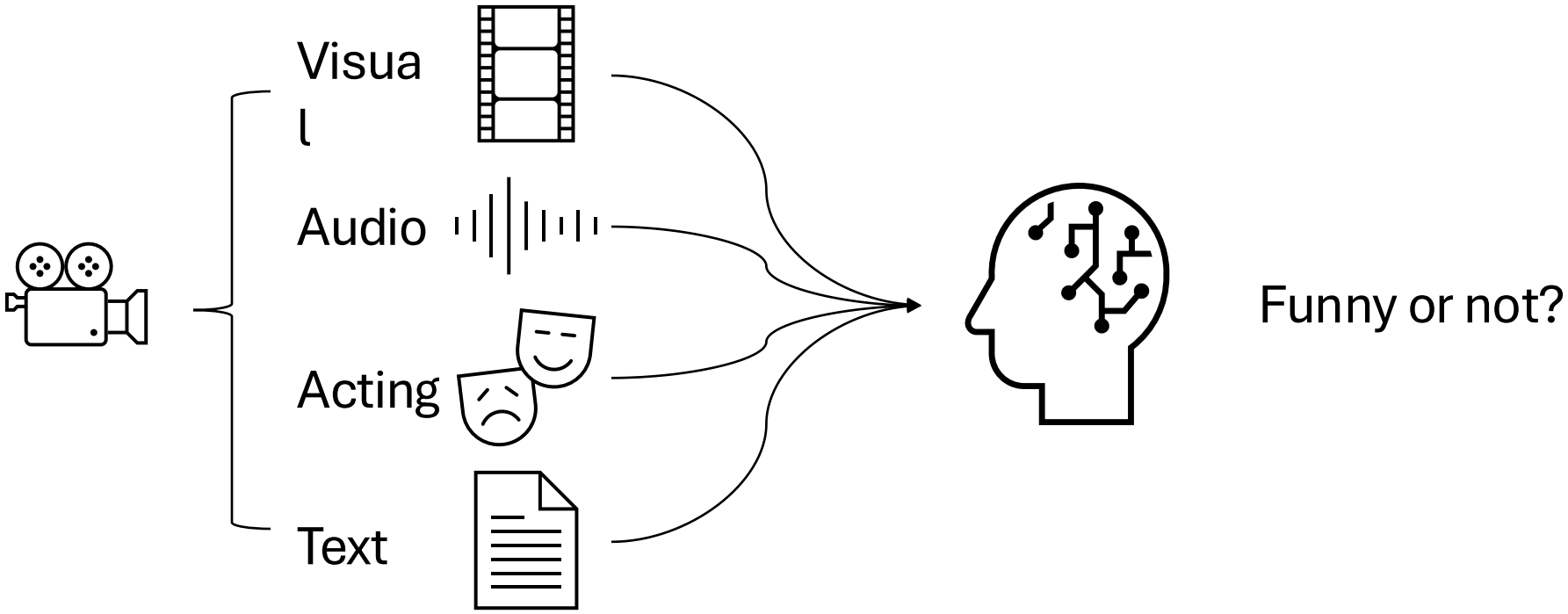
CLIP

5 Januray 2021
12+k citations in 3 years



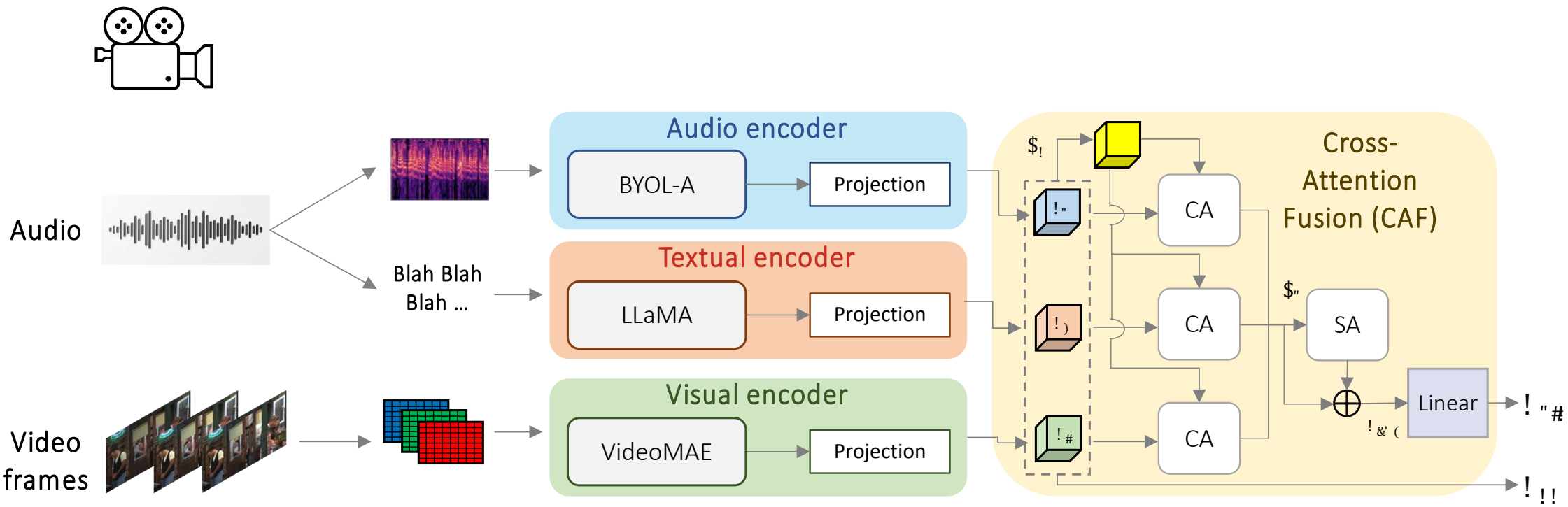
Self-supervision with multimodalities

FunnyNet-W: Multimodal learning of funny moments in videos in the wild



[Liu et al, IJCV 2024]

FunnyNet-W


Audio
Video frames
Blah Blah Blah ...
Audio encoder
BYOL-A
Projection
Textual encoder
LLaMA
Projection
Visual encoder
VideoMAE
Projection
Cross-Attention Fusion (CAF)
CA
CA
CA
SA
Linear

FunnyNet-W
Sitcom with Canned Laughter

FunnyNet-W
Sitcom without Canned Laughter

Manny: I wish I could stay home with you and fly toy airplanes.

Jay: These aren't toy airplanes, Manny.
These are *models* and they're very complicated.
You wanna fly one of these, you gotta be familiar with
air foyle, drag, lift and thrust, and these are all principles of aerodynamics.

Manny: The box says twelve and up.

Jay: What?!

Generation

VL Generative models

How do we understand the world?

- Our brains are like powerful computers, constantly predicting what will happen next based on past experiences

- Just like we use past knowledge to guess future events, generative AI attempts to forecast and create based on what it has learned

The Future of AI with Generative Models

- Generative AI could be a critical part of future AI advancements
- It combines creating new things and understanding causality—how one action leads to another
- Like us, these AI systems can simulate different scenarios and outcomes, helping to make better decisions

Precision Meets Creativity: A new era in AI

- Predict
- Classify
- Recommend

- Imagine
- Augment
- Create

AGI? Artificial General Intelligence

Transforming Business with Generative AI

- Innovating product design
  - Generative Games, word synthesis
  - Shoes, jewelry, clothing, …
  - Rapid Prototyping
  - Material Innovation

- Enhancing customer interaction
  - Personalized recommendations
  - Automated customer support
  - Interactive virtual assistant
  - Sentiment Analysis

- Streamlining content creation
  - 3D house models
  - AI-assisted Video Production, Choreography
  - Dynamic Web Content
  - Marketing Material Creation

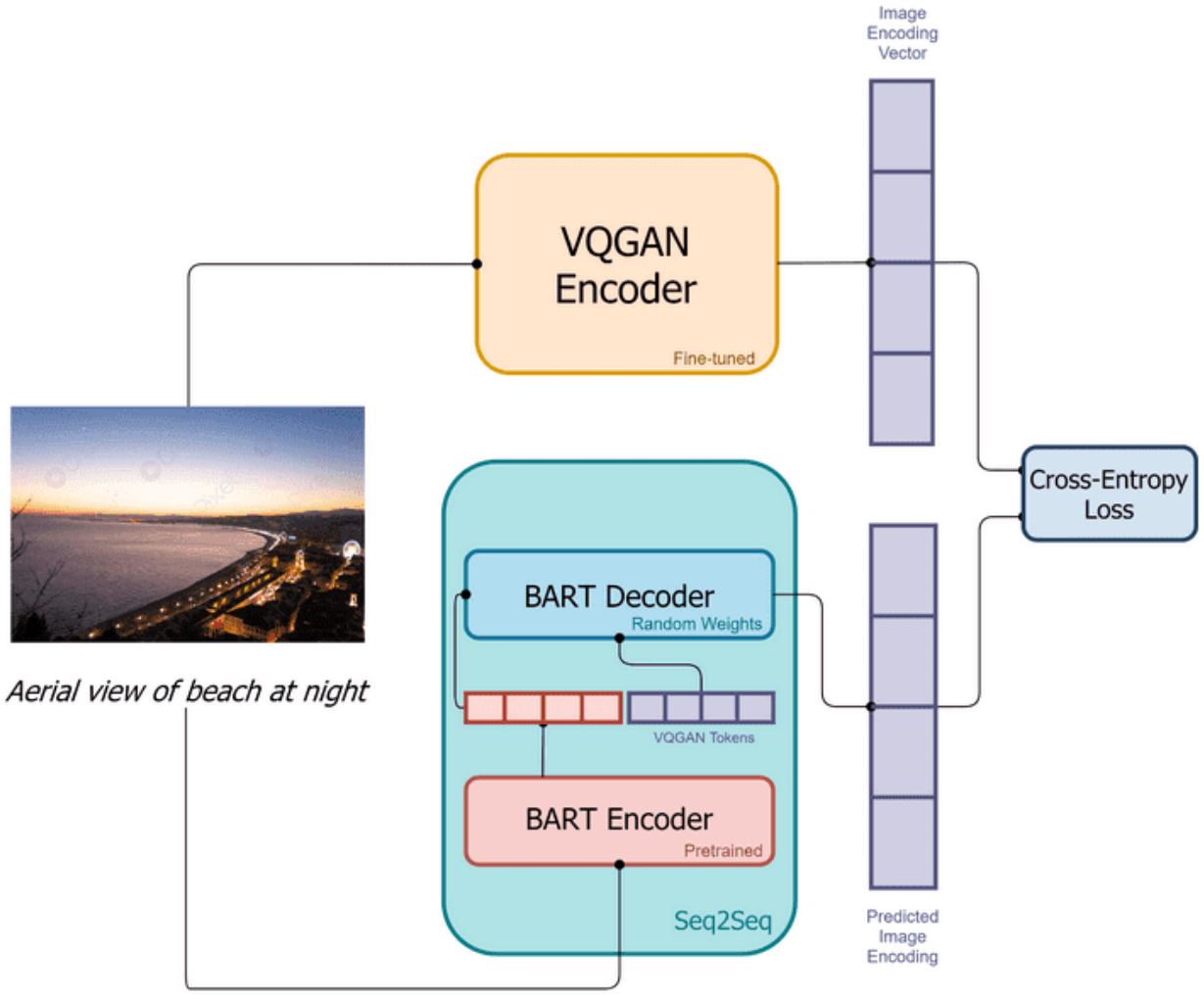
Creative palette

- Images
- Audio
- Text
- 3D models
- ….

Information

Everybody Dance Now

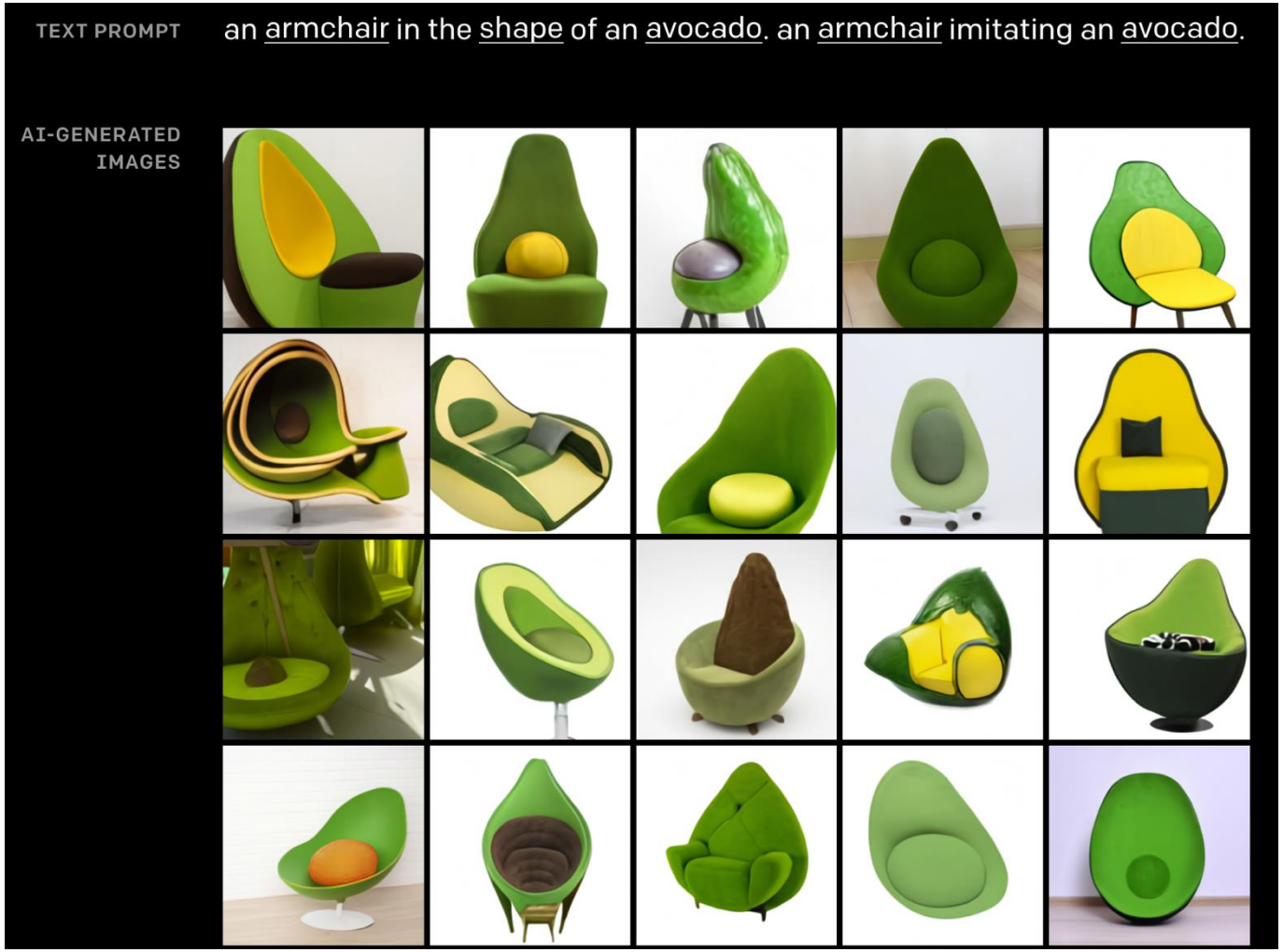
DALL-E



- Discrete variational autoencoder (VAE):
  maps images to image tokens
- dVAE uses a discrete latent space compared to a typical VAE

- Text: tokenized with byte-pair encoding
- Image and text tokens concatenated, processed as a single data stream

DALLE



https://openai.com/blog/dall-e/

Dalle-2 (Text-to-Image)

A bowl of soup as a planet in the universe

An astronaut riding a horse in a photorealistic style

Teddy bears mixing sparkling chemicals as mad scientists

Diffusion Models

OpenAI: DALL-E3

Midjourney

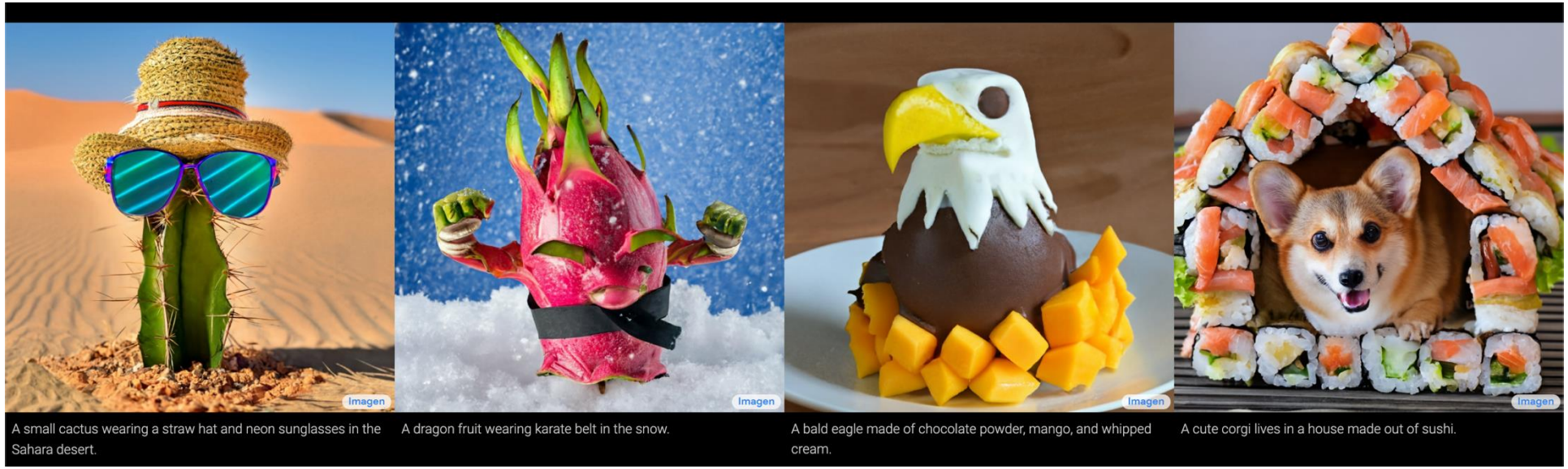
music, audio, animation, video, physical etc....

Imagen by Google AI

A photo of a Corgi dog riding a bike in Times Square. It is wearing sunglasses and a beach hat.

https://imagen.research.google/

Imagen by Google AI



https://imagen.research.google/

Make-A-Video (Text-to-Video)

A confused grizzly bear in a calculus class

A golden retriever eating ice cream on a beautiful tropical beach at sunset, high resolution

A panda playing on a swing set

Today's lecture

Part I: Practical Information

Part II: Introduction

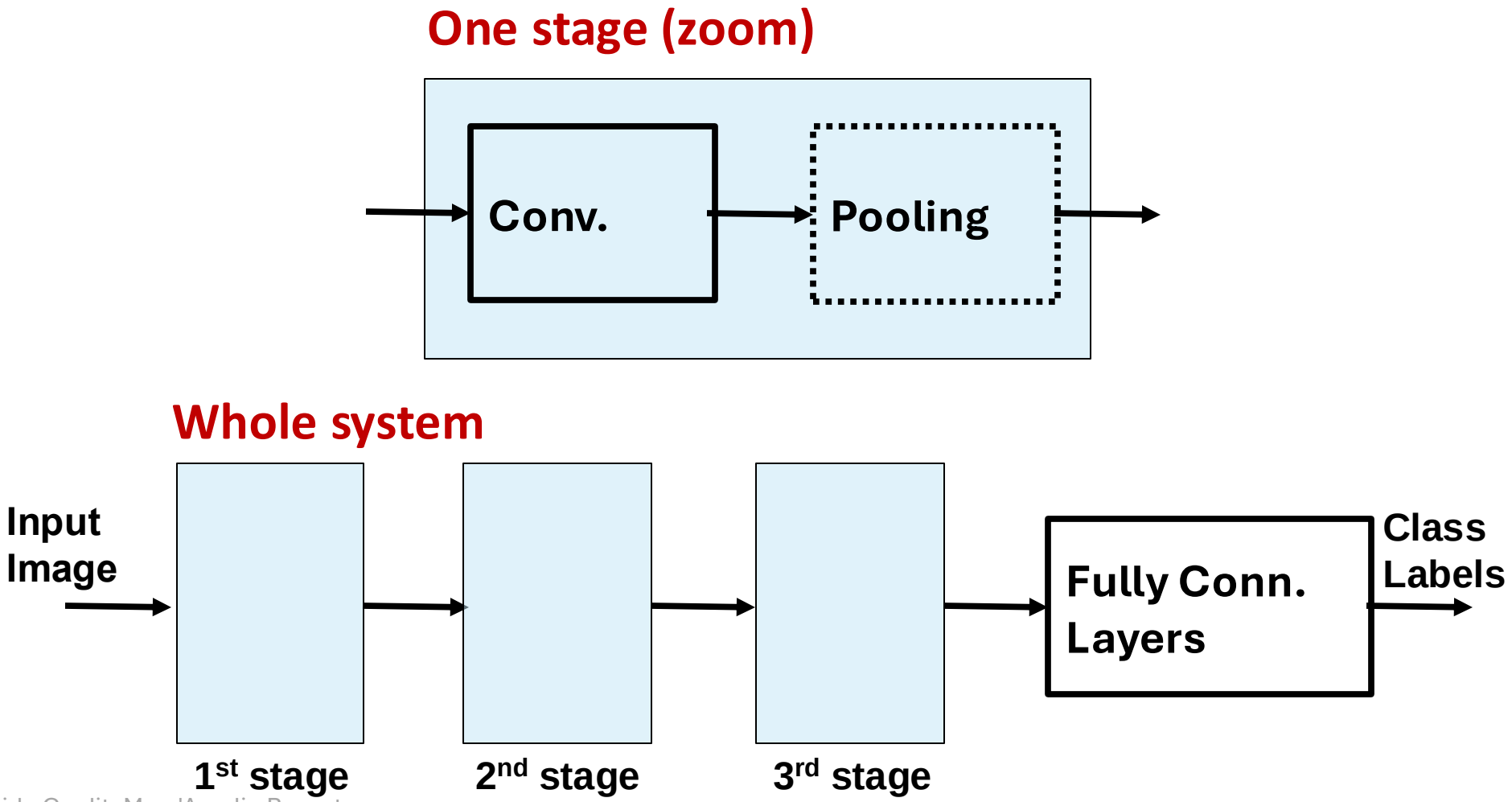
Part III: Deep Learning architectures

Part III: Outline

- CNN architectures
  - AlexNet
  - VGGNet
  - ResNet
- Transfer Learning
- Multimodal learning

CNN architectures

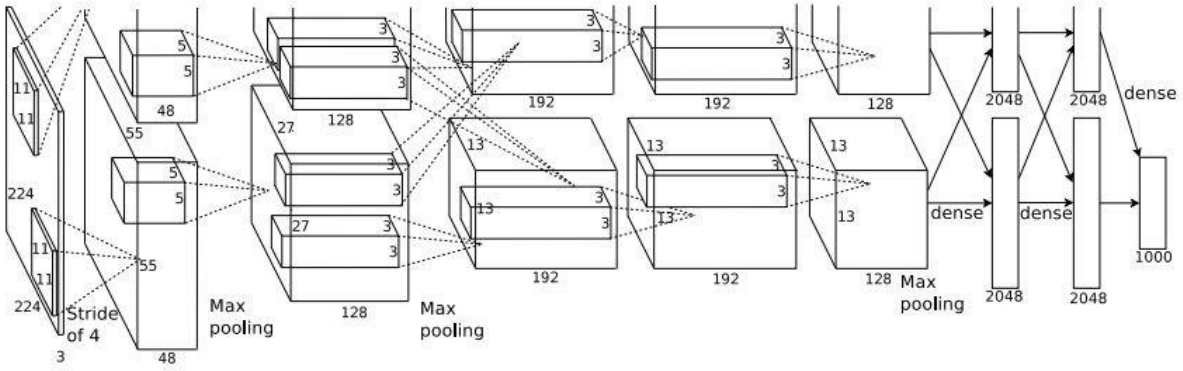
ConvNets: Typical Architecture



Slide Credit: Marc'Aurelio Ranzato

AlexNet

- 227 x 227 inputs
- 5 Convolutional layers
- Max pooling
- 3 fully-connected layers
- ReLU nonlinearities

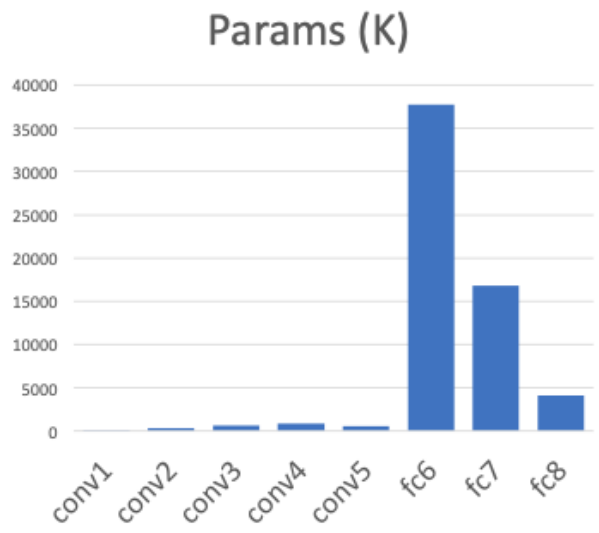


[Krizhevsky et al. 2012]

AlexNet

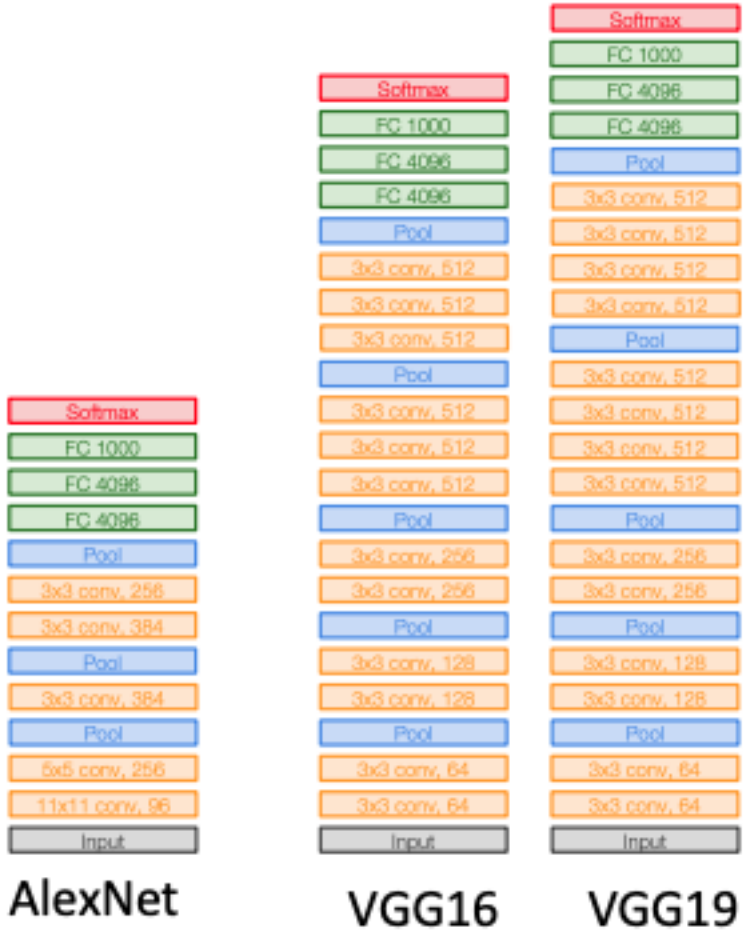
Most of the memory usage is in the early conv layers

Most of the parameters are in the fully-connected layers



VGGNet

- Regular design:
  - All conv are 3x3 stride 1, pad 1
  - All max pool are 2x2 stride 2
  - After pool, double #channels

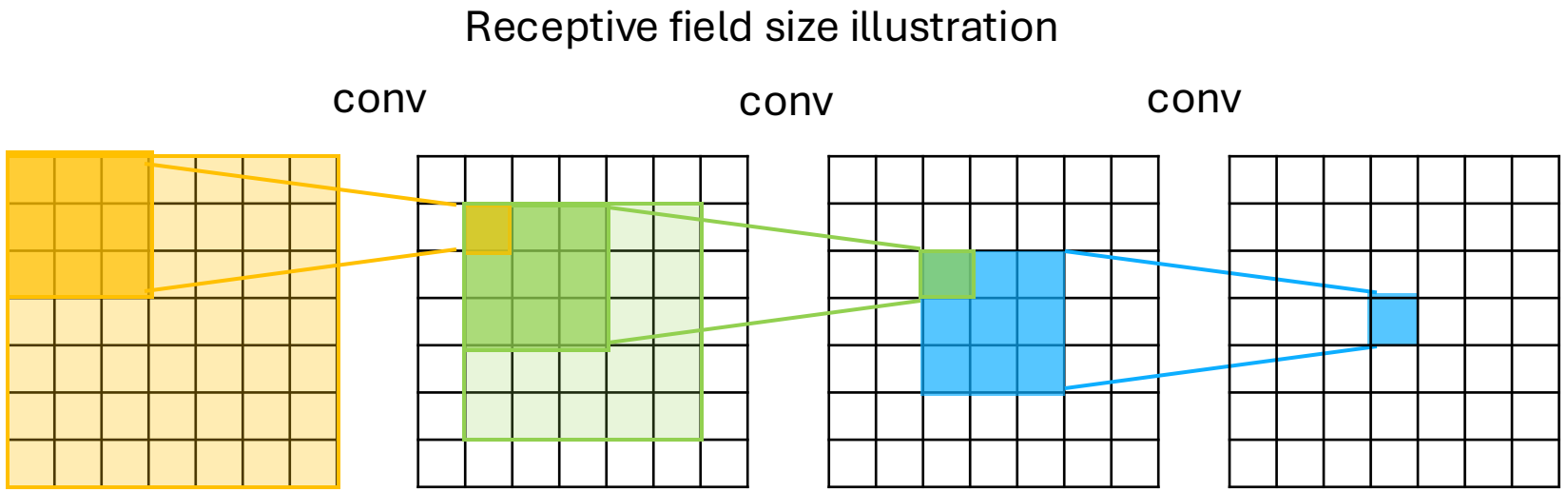


[Simonyan and Zisserman, 2015]

VGGNet

- **Why use smaller filters? (3x3 conv)**

Stack of three 3x3 conv (stride 1) layers has the same effective receptive field as one 7x7 conv layer



Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

[Simonyan and Zisserman, 2014]

VGGNet

- **Why use smaller filters? (3x3 conv)**

Stack of three 3x3 conv (stride 1) layers has the same effective receptive field as one 7x7 conv layer

- **What is the effective receptive field of three 3x3 conv (stride 1) layers?**

7x7

But deeper, more non-linearities → more complex patterns

And fewer parameters: 3 * (3^2C^2) {3 conv layers * 3x3 filter size by C^2 channels} conv vs. 7^2C^2 for C channels per layer → $27C^2$ vs $49C^2$

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

[Simonyan and Zisserman, 2014]

VGGNet

Input
3x3 conv, 64
3x3 conv, 64
Pool
3x3 conv, 128
3x3 conv, 128
Pool
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
Pool
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
Pool
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
Pool
FC 4096
FC 4096
FC 1000
Softmax

VGG16:

TOTAL memory: 24M * 4 bytes ~= 96MB / image

TOTAL params: 138M parameters

E.g. with 5 GB of memory, can only work with 50 images in a batch

[Simonyan and Zisserman, 2014]

VGGNet -- Memory and Parameters

Input	memory: 224*224*3=150K	params: 0
3x3 conv, 64	memory: 224*224*64=3.2M	params: (3*3*3)*64 = 1,728
3x3 conv, 64	memory: 224*224*64=3.2M	params: (3*3*64)*64 = 36,864
Pool	memory: 112*112*64=800K	params: 0
3x3 conv, 128	memory: 112*112*128=1.6M	params: (3*3*64)*128 = 73,728
3x3 conv, 128	memory: 112*112*128=1.6M	params: (3*3*128)*128 = 147,456
Pool	memory: 56*56*128=400K	params: 0
3x3 conv, 256	memory: 56*56*256=800K	params: (3*3*128)*256 = 294,912
3x3 conv, 256	memory: 56*56*256=800K	params: (3*3*256)*256 = 589,824
3x3 conv, 256	memory: 56*56*256=800K	params: (3*3*256)*256 = 589,824
Pool	memory: 28*28*256=200K	params: 0
3x3 conv, 512	memory: 28*28*512=400K	params: (3*3*256)*512 = 1,179,648
3x3 conv, 512	memory: 28*28*512=400K	params: (3*3*512)*512 = 2,359,296
3x3 conv, 512	memory: 28*28*512=400K	params: (3*3*512)*512 = 2,359,296
Pool	memory: 14*14*512=100K	params: 0
3x3 conv, 512	memory: 14*14*512=100K	params: (3*3*512)*512 = 2,359,296
3x3 conv, 512	memory: 14*14*512=100K	params: (3*3*512)*512 = 2,359,296
3x3 conv, 512	memory: 14*14*512=100K	params: (3*3*512)*512 = 2,359,296
Pool	memory: 7*7*512=25K	params: 0
FC 4096	memory: 4096	params: 7*7*512*4096 = 102,760,448
FC 4096	memory: 4096	params: 4096*4096 = 16,777,216
FC 1000	memory: 1000	params: 4096*1000 = 4,096,000

VGGNet

Details/Retrospectives:

- ILSVRC'14 2nd in classification, 1st in localization
- Similar training procedure as AlexNet
- No Local Response Normalisation (LRN)
- Use VGG16 or VGG19 (VGG19 only slightly better, more memory)
- Use ensembles for best results
- FC7 features generalize well to other tasks
- Trained on 4 Nvidia Titan Black GPUs for **two to three weeks**

[Simonyan and Zisserman, 2014]

VGGNet

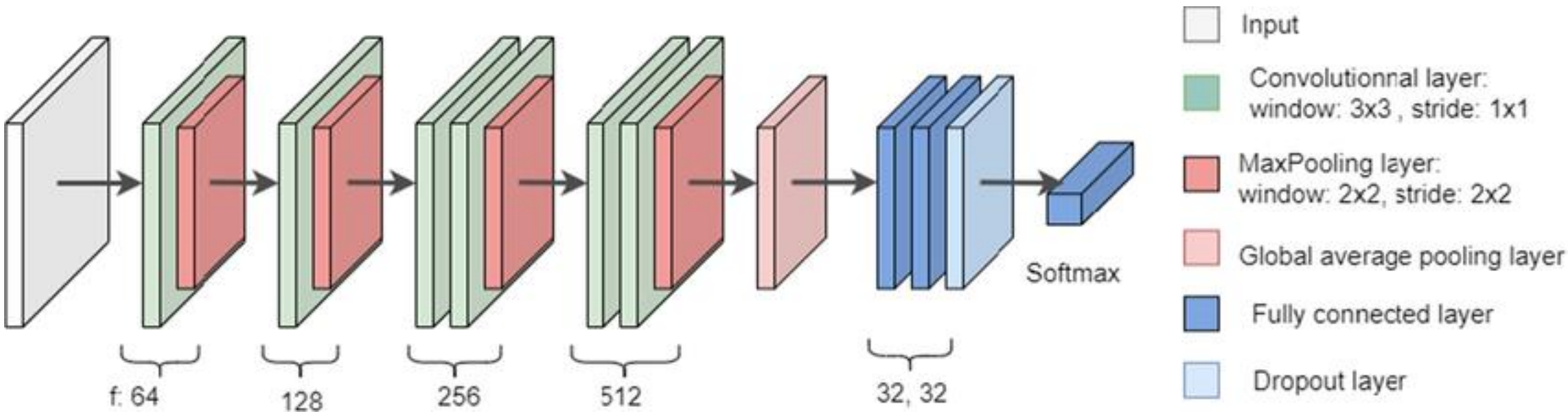
VGG Net reinforced the notion that **convolutional neural networks have to have a deep network of layers in order for this hierarchical representation of visual data to work**

Keep it deep

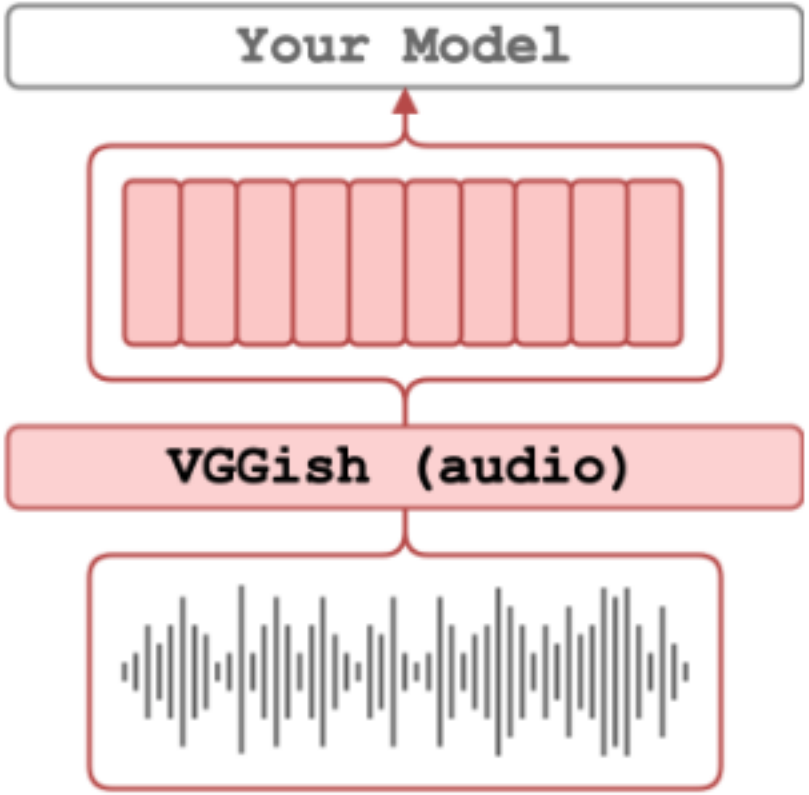
Keep it simple

[Simonyan and Zisserman, 2014]

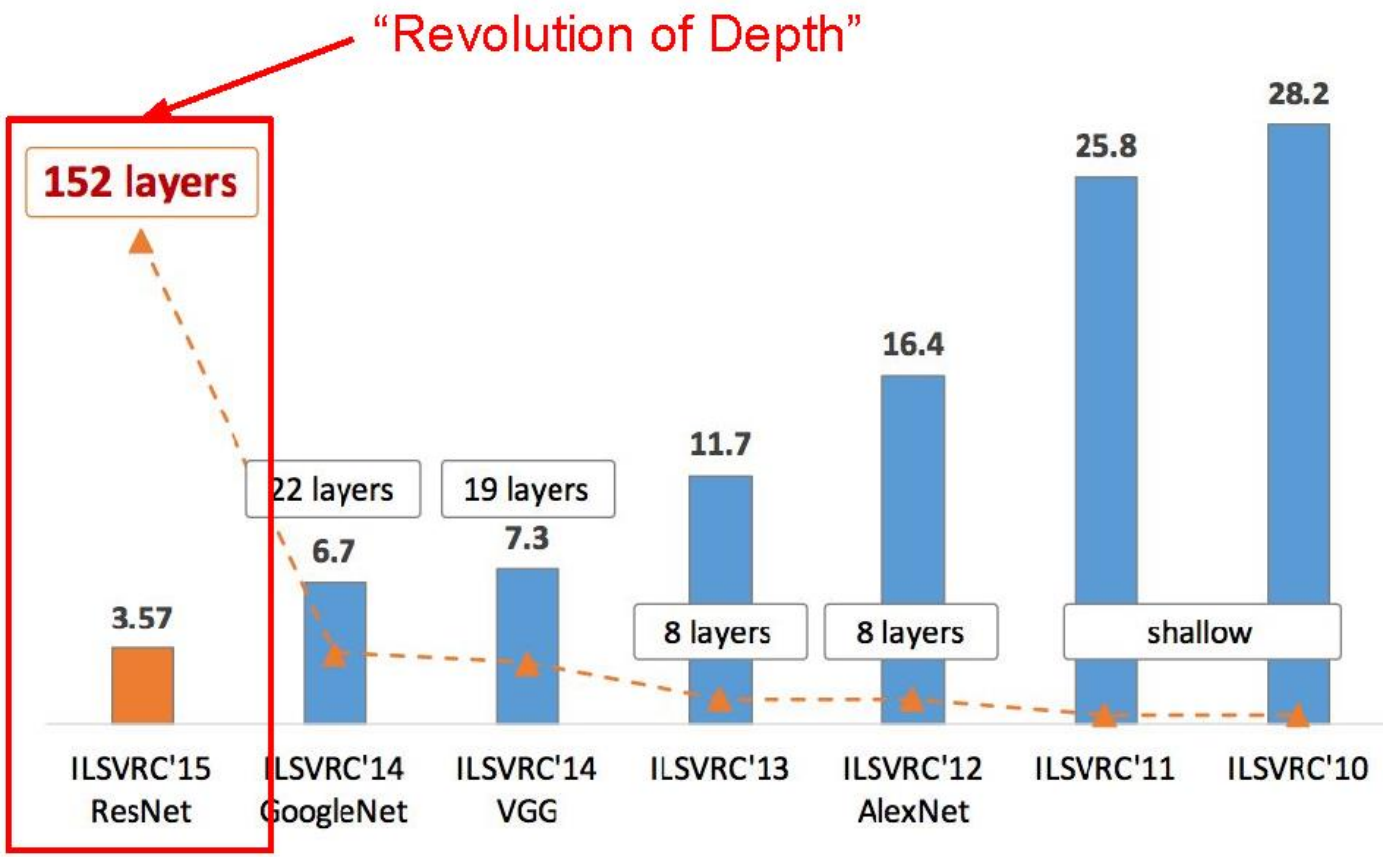
VGGish → Audio architecture



VGGish → Audio architecture



ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners



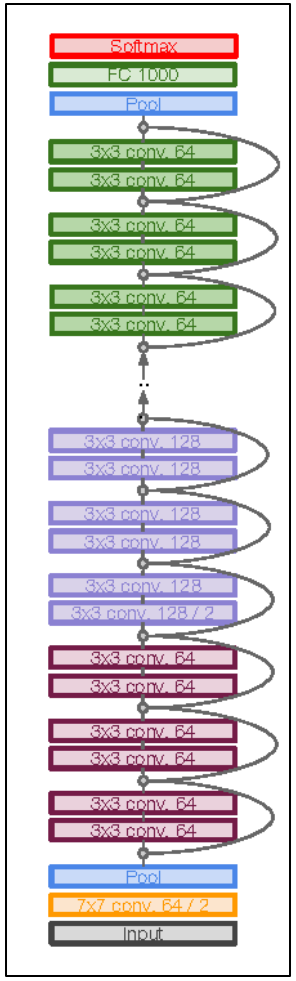
Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

ResNet

- *Deep Residual Learning for Image Recognition - Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun; 2015*
- Extremely deep network – 152 layers
- Deeper neural networks are more difficult to train
- Deep networks suffer from vanishing and exploding gradients
- Present a residual learning framework to ease the training of networks that are substantially deeper than those used previously

[He et al., 2015]

ResNet

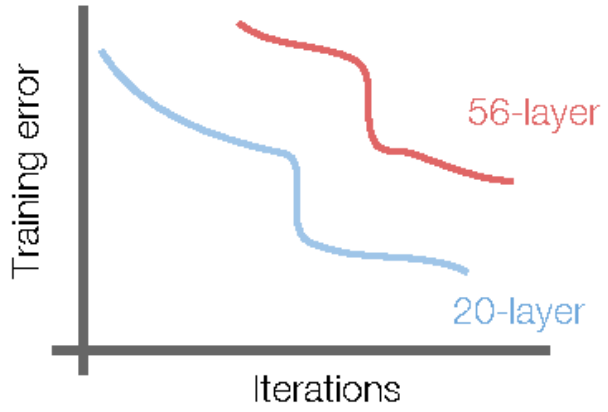


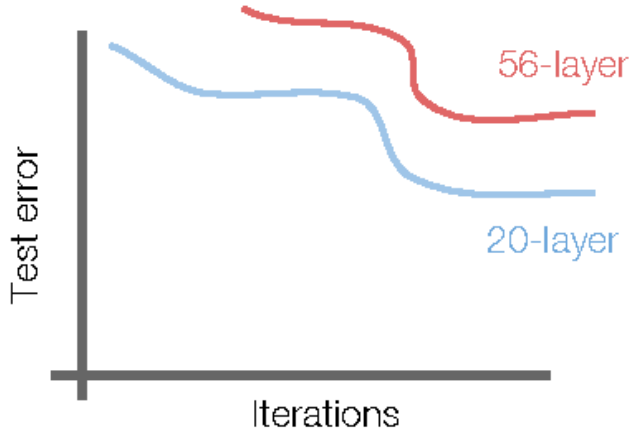
- ILSVRC'15 classification winner (3.57% top 5 error, humans generally hover around a 5-10% error rate)
  Swept all classification and detection competitions in ILSVRC'15 and COCO'15!

[He et al., 2015]

ResNet

- What happens when we continue stacking deeper layers on a convolutional neural network?





- 56-layer model performs worse on both training and test error

→ The deeper model performs worse (not caused by overfitting)!

[He et al., 2015]

Slide taken from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

ResNet

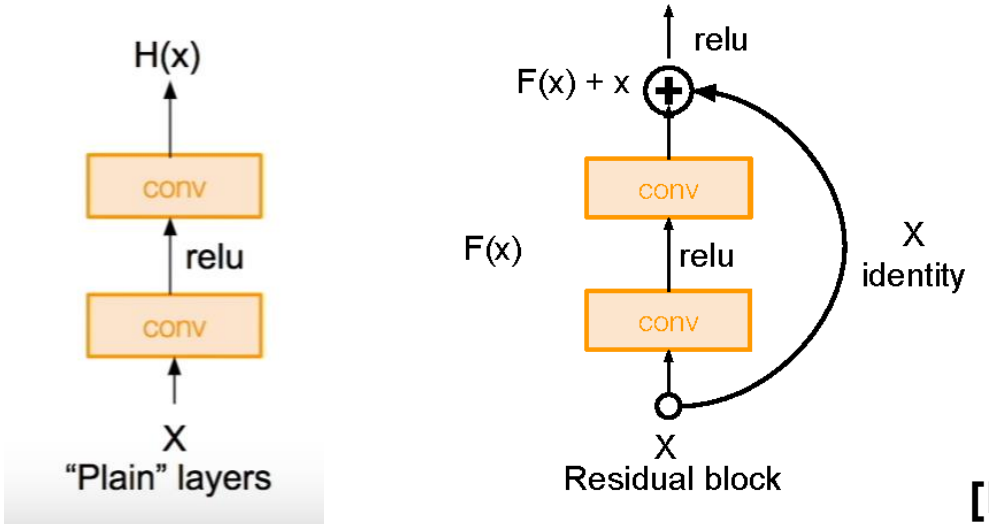
- **Hypothesis**: The problem is an optimization problem. Very deep networks are harder to optimize
- **Solution**: Use network layers to fit a ***residual mapping*** instead of directly trying to fit a desired underlying mapping
- Use **skip connections** allowing to take the activation from one layer and feed it into another layer, much deeper into the network
- Use layers to fit residual $F(x) = H(x) – x$ instead of $H(x)$ directly

[He et al., 2015]

ResNet

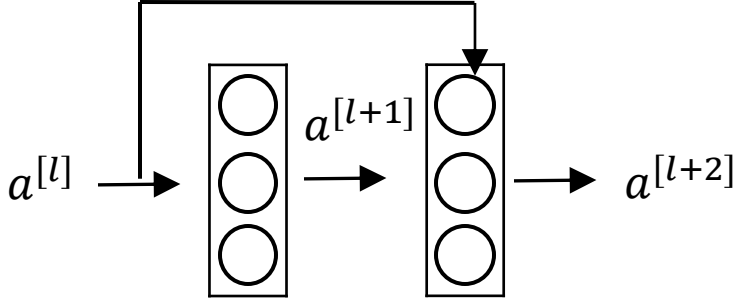
Residual Block
Input x goes through conv-relu-conv series and gives F(x). That result is then added to the original input x. Call that H(x) = F(x) + x

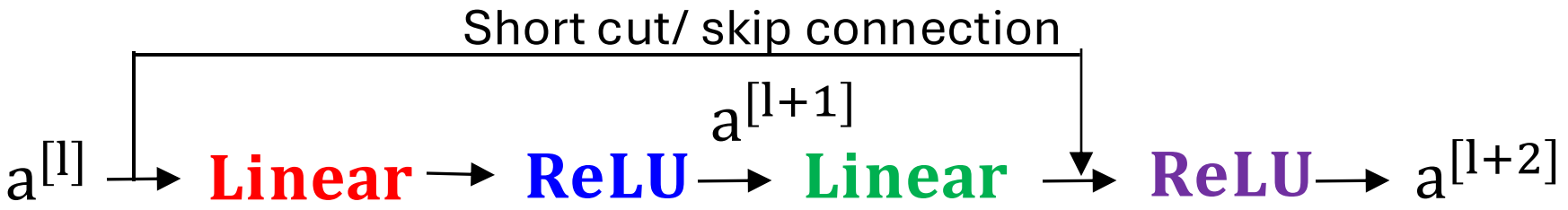
In traditional CNNs, H(x) would be equal to F(x). Instead of just computing that transformation (straight from x to F(x)), we are computing the term that we have to *add*, F(x), to the input, x.



[He et al., 2015]

ResNet





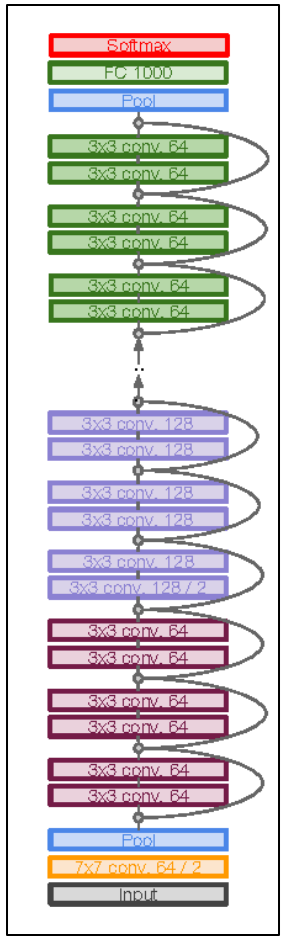
$$\mathbf{z}^{[l+1]} = \mathbf{W}^{[l+1]}\,\mathbf{a}^{[l]} + \mathbf{b}^{[l+1]} \qquad \mathbf{z}^{[l+2]} = \mathbf{W}^{[l+2]}\mathbf{a}^{[l+1]} + \mathbf{b}^{[l+2]}$$

$$\mathbf{a}^{[l+1]} = \mathbf{g}(\mathbf{z}^{[l+1]}) \qquad \mathbf{a}^{[l+2]} = \mathbf{g}(\mathbf{z}^{[l+2]})$$

$$\mathbf{a}^{[l+2]} = \mathbf{g}\left(\mathbf{z}^{[l+2]} + \mathbf{a}^{[l]}\right) = \mathbf{g}(\mathbf{W}^{[l+2]}\mathbf{a}^{[l+1]} + \mathbf{b}^{[l+2]} + \mathbf{a}^{[l]})$$

[He et al., 2015]

ResNet



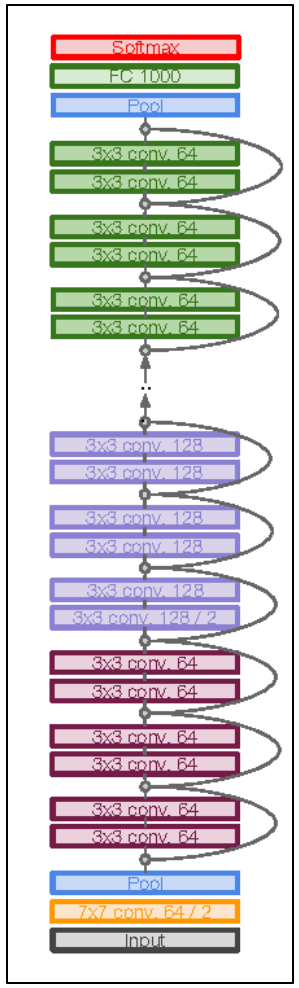
[He et al., 2015]

Full ResNet architecture:

- Stack residual blocks
- Every residual block has two 3x3 conv layers
- Periodically, double # of filters and downsample spatially using stride 2 (in each dimension)
- Additional conv layer at the beginning
- No FC layers at the end (only FC 1000 to output classes)

Slide from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

ResNet



[He et al., 2015]

- Total depths of 18, 34, 50, 101, or 152 layers for ImageNet
- For deeper networks (ResNet-50+), use "bottleneck" layer to improve efficiency (similar to GoogLeNet)

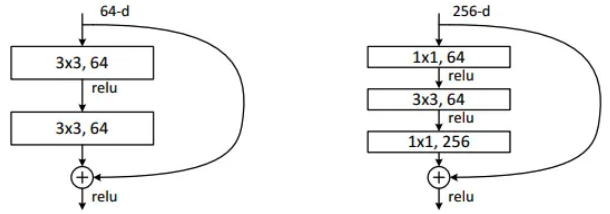


Figure 5. Common residual block vs Deeper Bottleneck residual block (Source: Original ResNet paper)

Figure 5 shows the difference between a common residual block vs a deeper bottleneck residual block. In the common residual block, there are two convolutional layers having 3x3 filters. In the bottleneck architecture, there are 3 convolutional layers instead of 2. The three layers are 1x1, 3x3, and 1x1 convolutions, where the 1x1 layers are responsible for reducing and then increasing (restoring) dimensions, leaving the 3x3 layer a bottleneck with smaller input/output dimensions.

ResNet

Experimental Results:

- Able to train very deep networks without degrading
- Deeper networks now achieve lower training errors as expected

[He et al., 2015]

Slide from Fei-Fei & Justin Johnson & Serena Yeung. Lecture 9.

ResNet

A good **baseline** CNN architecture that we currently have and is a great innovation for the idea of residual learning.

Even better than human performance!**

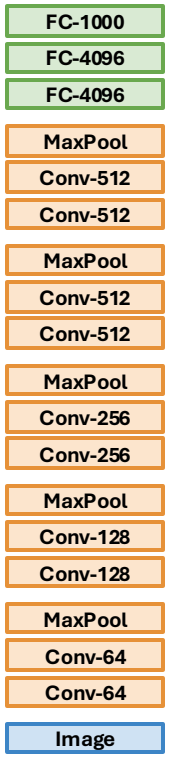
ImageNet 2017 last and final edition, the performance has saturated. 98% top5 accuracy.

**Dodge, S., and Karam L.. "*A study and comparison of human and deep learning recognition performance under visual distortions*." ICCCN 2017.

Transfer learning

Transfer Learning with CNNs

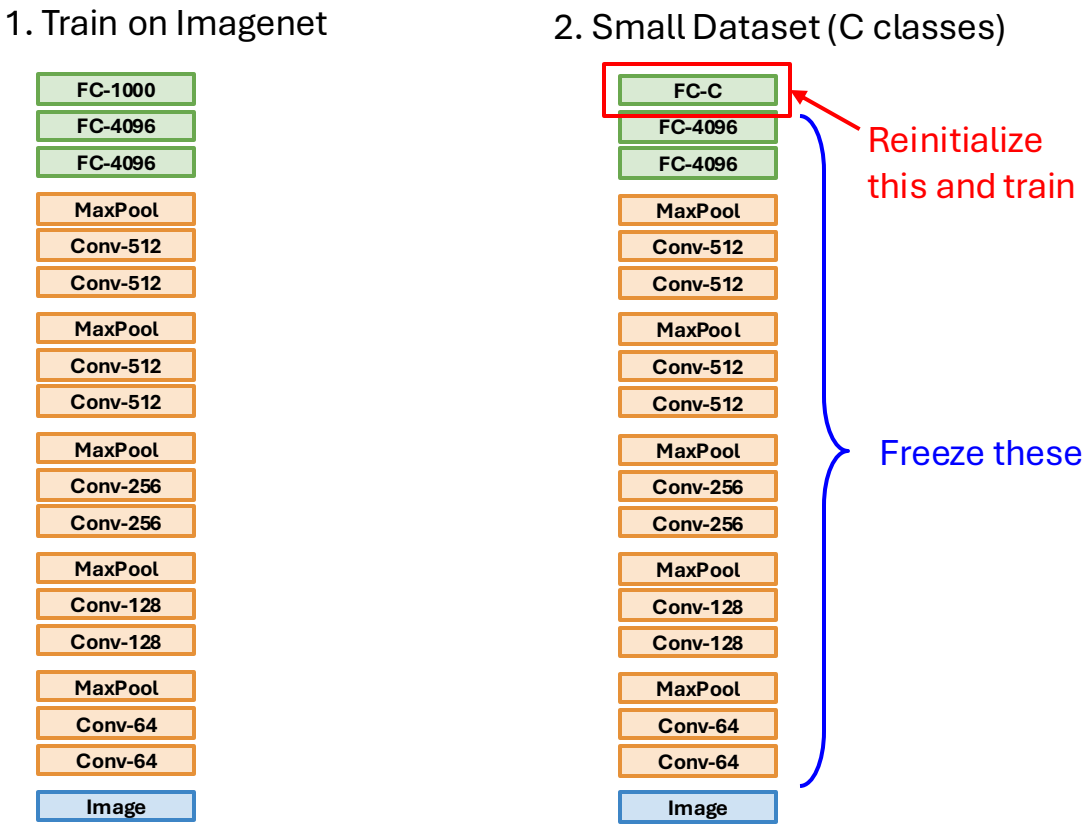
1. Train on Imagenet



Donahue et al, "DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition", ICML 2014
Razavian et al, "CNN Features Off-the-Shelf: An Astounding Baseline for Recognition", CVPR Workshops 2014

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

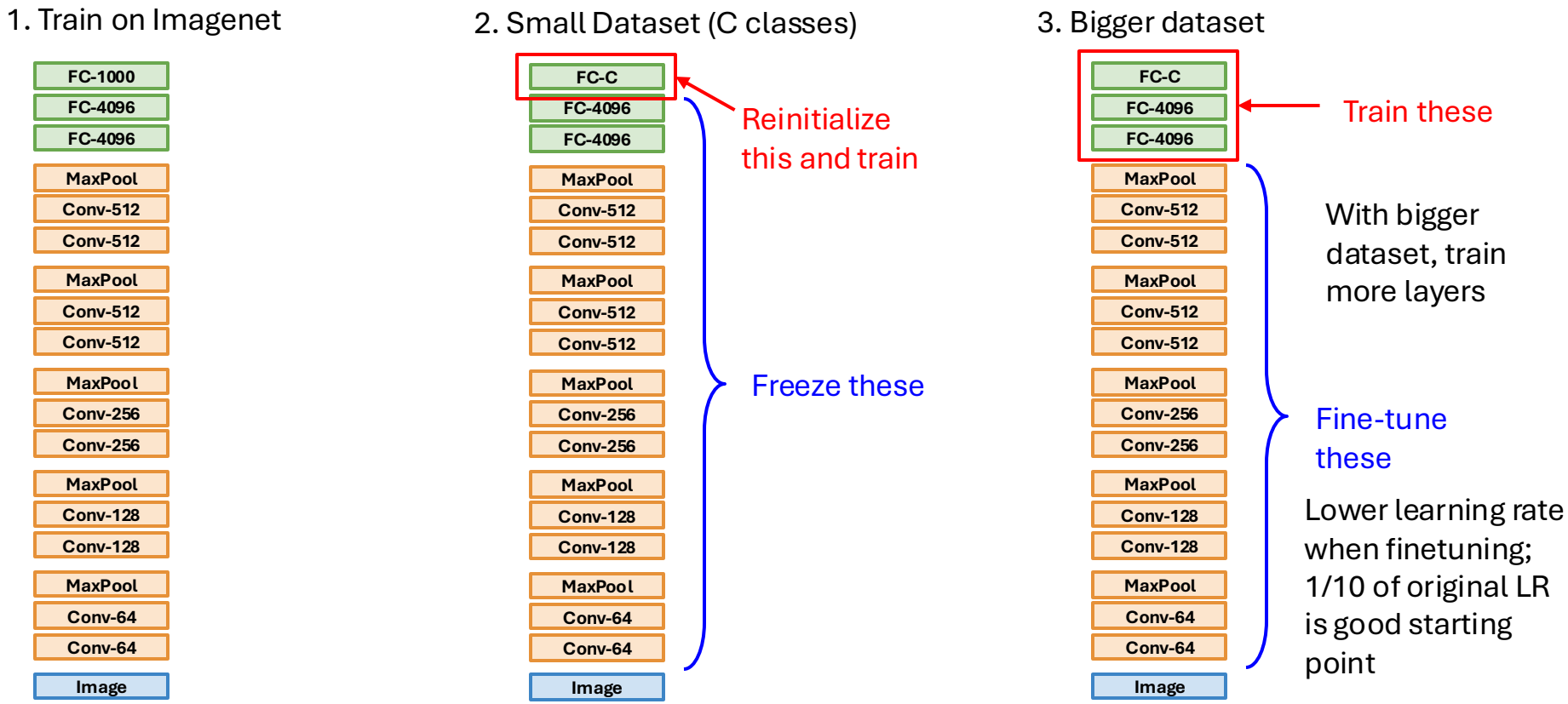
Transfer Learning with CNNs



Donahue et al, "DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition", ICML 2014
Razavian et al, "CNN Features Off-the-Shelf: An Astounding Baseline for Recognition", CVPR Workshops 2014

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

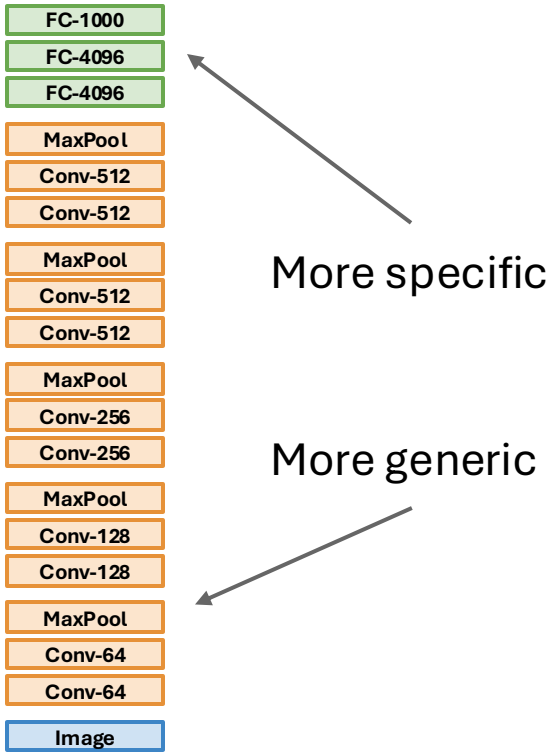
Transfer Learning with CNNs



Donahue et al, "DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition", ICML 2014
Razavian et al, "CNN Features Off-the-Shelf: An Astounding Baseline for Recognition", CVPR Workshops 2014

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

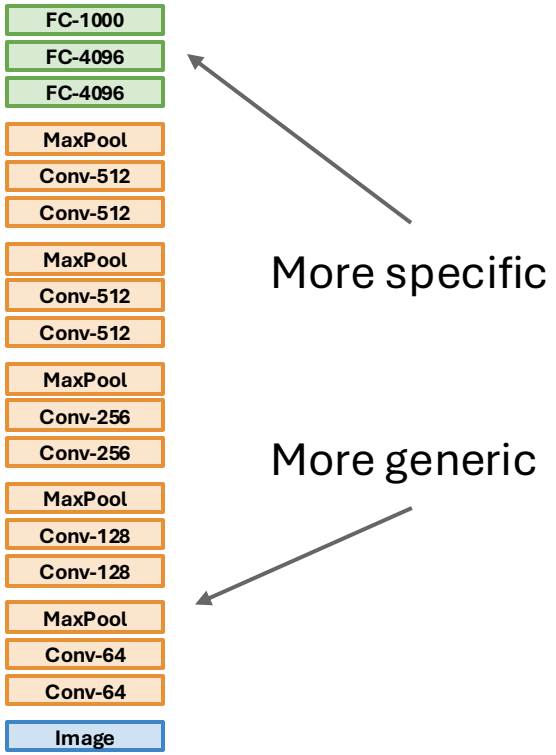
Transfer Learning with CNNs



	very similar dataset	very different dataset
very little data	?	?
quite a lot of data	?	?

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

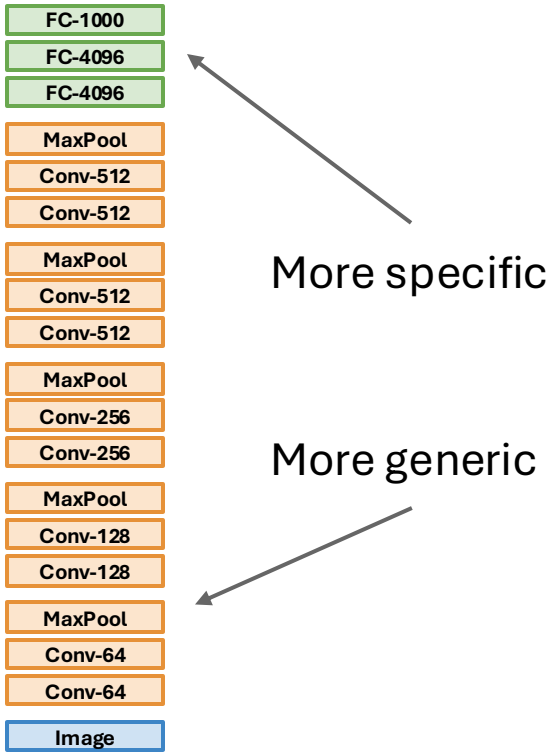
Transfer Learning with CNNs



	very similar dataset	very different dataset
very little data	Use Linear Classifier on top layer	?
quite a lot of data	Finetune a few layers	?

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Transfer Learning with CNNs

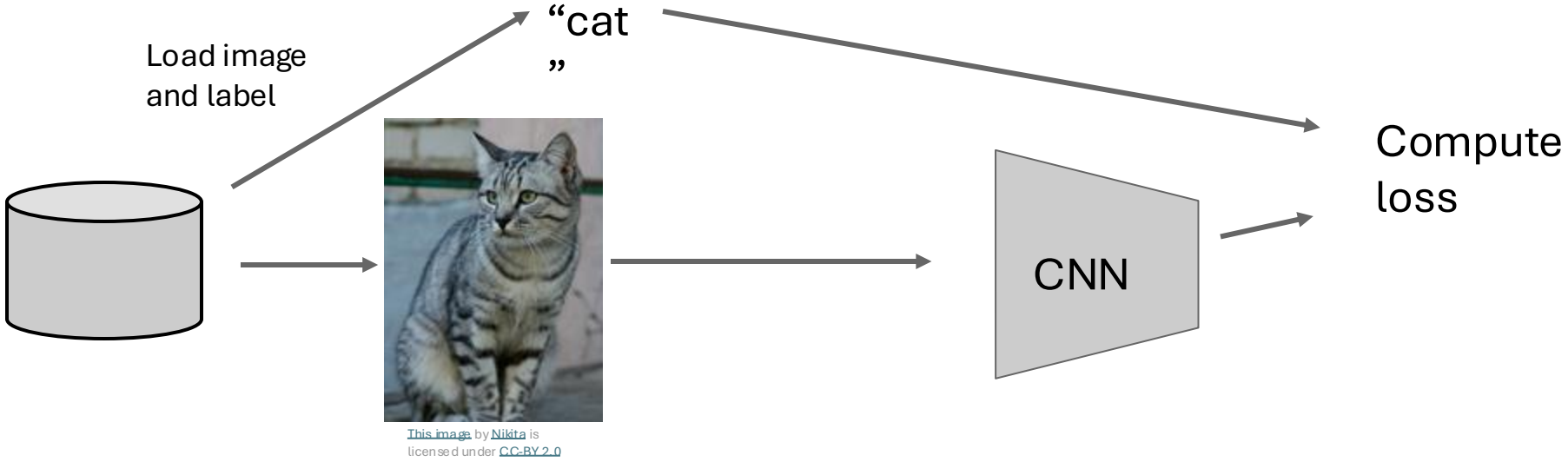


	very similar dataset	very different dataset
very little data	Use Linear Classifier on top layer	You're in trouble… Try linear classifier from different stages
quite a lot of data	Finetune a few layers	Finetune a larger number of layers

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

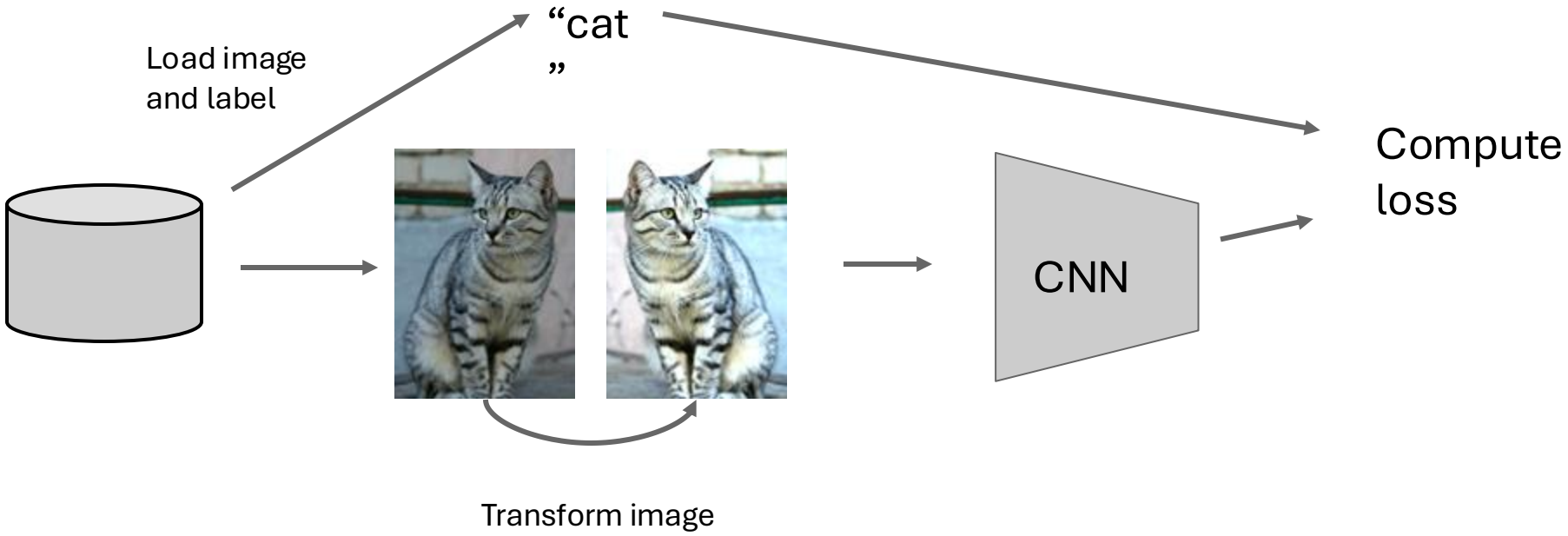
Data augmentation

Data Augmentation



Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

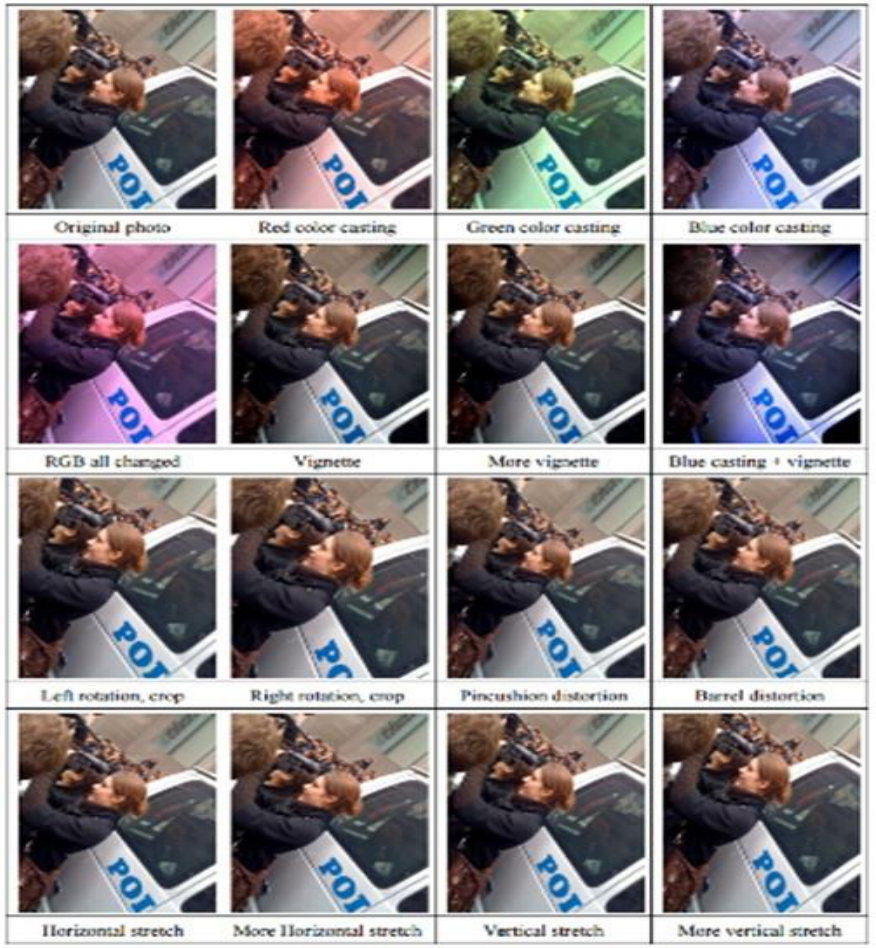
Data Augmentation



Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Data Augmentation (Jittering)

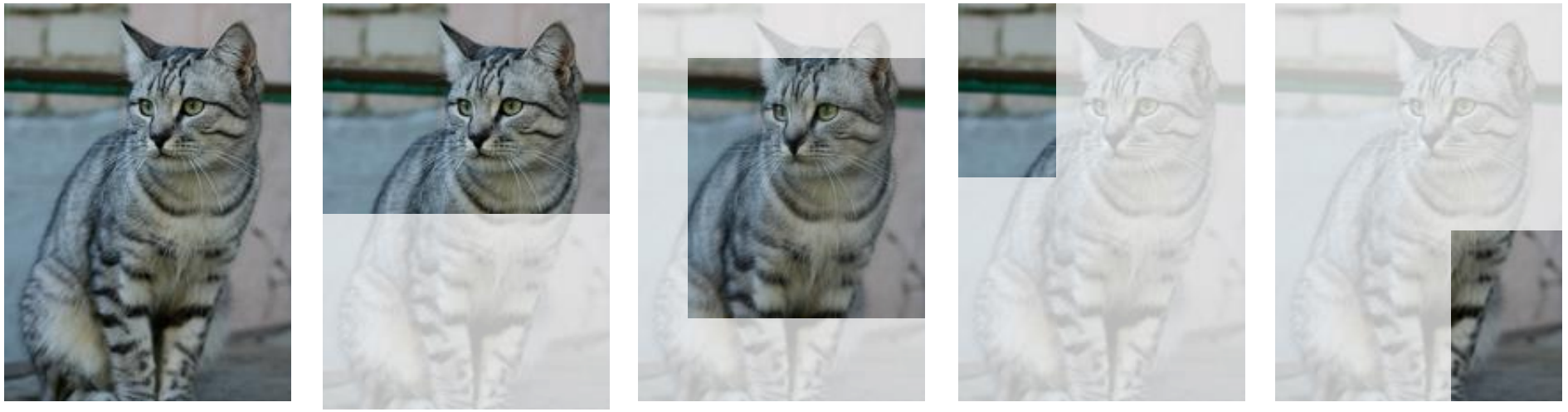
- Create *virtual* training samples
  - Horizontal flip
  - Random crop
  - Color casting
  - Geometric distortion



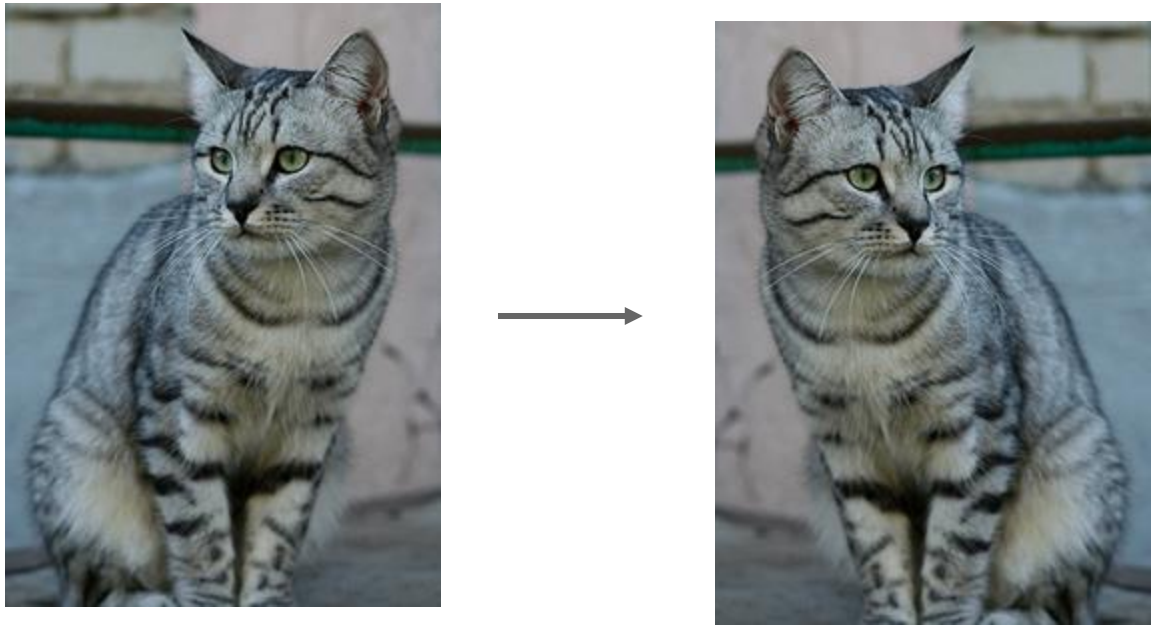
Deep Image [Wu et al. 2015]

Data Augmentation (Cropping)

Crops should contain the object of interest



Data Augmentation
Horizontal Flips



Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

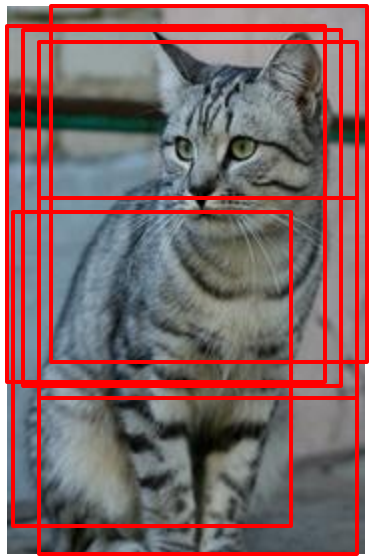
Data Augmentation

- Simple: Randomize contrast and brightness

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Data Augmentation
Random crops and scales

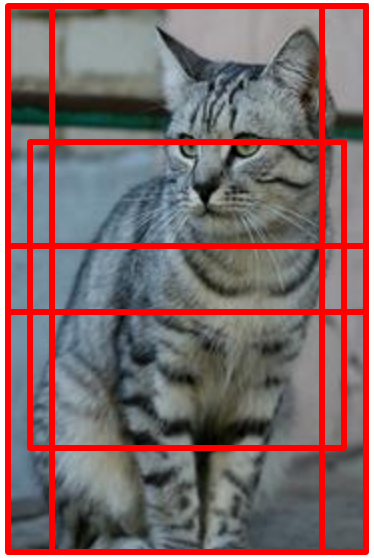
- **Training**: sample random crops / scales
  ResNet:
  1. Pick random L in range [256, 480]
  2. Resize training image, short side = L
  3. Sample random 224 x 224 patch



Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Data Augmentation
Random crops and scales

- **Training**: sample random crops / scales
  ResNet:
  1. Pick random L in range [256, 480]
  2. Resize training image, short side = L
  3. Sample random 224 x 224 patch
- **Testing**: average a fixed set of crops
- ResNet:
  1. Resize image at 5 scales: {224, 256, 384, 480, 640}
  2. For each size, use 10 224 x 224 crops: 4 corners + center, + flips



(As seen in *[Krizhevsky et al. 2012]*, ResNet, etc)

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Data Augmentation
Get creative for your problem!

- Random mix/combinations of :
  - translation
  - rotation
  - stretching
  - shearing,
  - lens distortions, … (go crazy)

Normalization

Batch Normalization

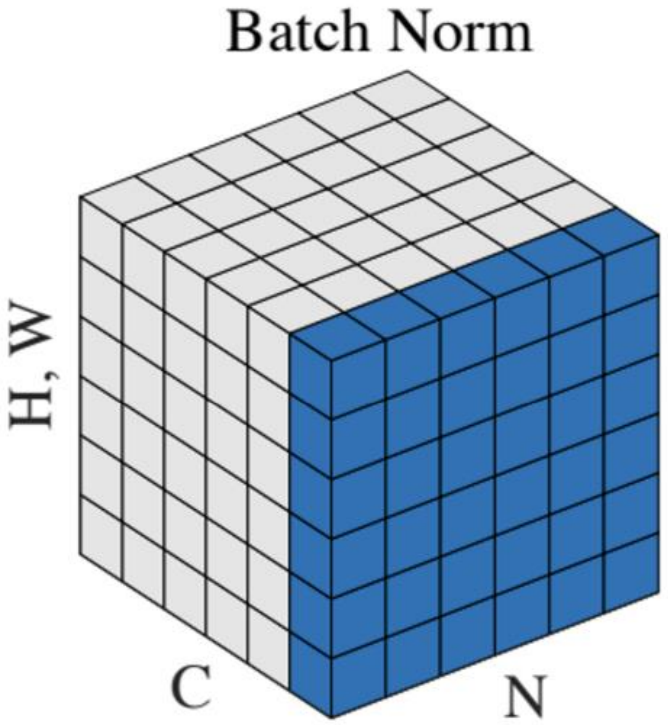
- A way to keep the weights in the gaussian range
- Idea: “Normalize” the outputs of a layer so they have zero mean and unit variance
  - Helps reduce “internal covariate shift”
  - improves optimization
- Before non-linearity – run batch normalization empirically on the specific batch

$$\hat{x} = \frac{x - E[x]}{\sqrt{Var[x]}}$$

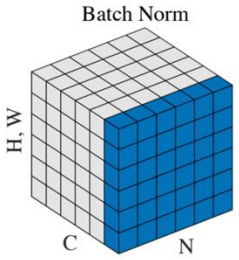
This is a differentiable function, so we can use it as an operator in our networks and backprop through it!

Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift [Ioffe and Szegedy 2015]

Batch Normalization



[Wu and He, “Group Normalization”, ECCV 2018]



Batch Normalization

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

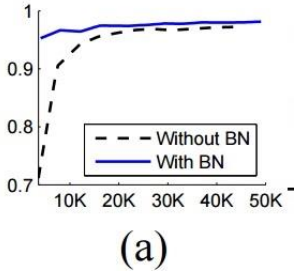
$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

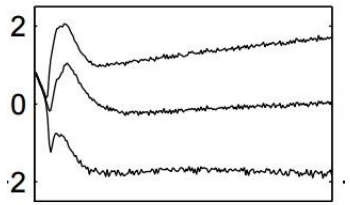
Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift [Ioffe and Szegedy 2015]

Test accuracy

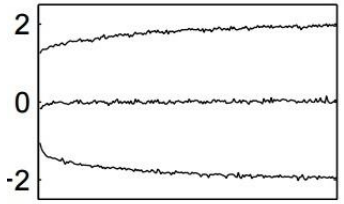


(a)

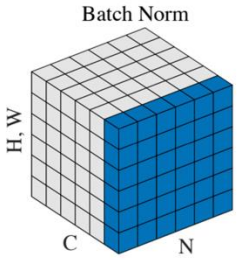
Responses of different layers over iterations



(b) Without BN



(c) With BN



Batch Normalization

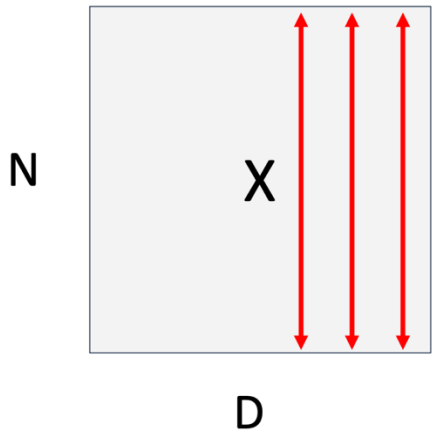
Input: $x : N \times D$

Learnable params: $\gamma, \beta : D$

Intermediates: $\mu, \sigma : D$

$\hat{x} : N \times D$

Output: $y : N \times D$



$$\mu_j = \frac{1}{N}\sum_{i=1}^{N} x_{i,j}$$

Per-channel mean, shape is D

$$\sigma_j^2 = \frac{1}{N}\sum_{i=1}^{N}(x_{i,j} - \mu_j)^2$$

Per-channel std, shape is D

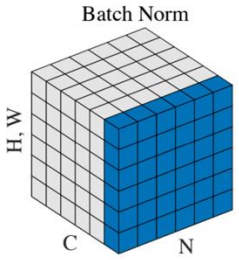
$$\hat{x}_{i,j} = \frac{x_{i,j} - \mu_j}{\sqrt{\sigma_j^2 + \varepsilon}}$$

Normalized x, shape is NxD

$$y_{i,j} = \gamma_j \hat{x}_{i,j} + \beta_j$$

Output, shape is NxD

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n



Batch Normalization

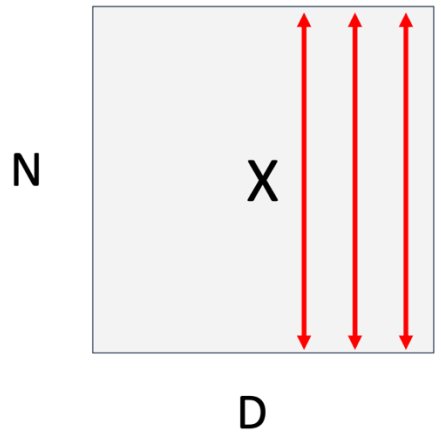
Input: $x : N \times D$

Learnable params: $\gamma, \beta : D$

Intermediates: $\mu, \sigma : D$

$\hat{x} : N \times D$

Output: $y : N \times D$



Estimate mean and variance from minibatch; Can't do this at test-time

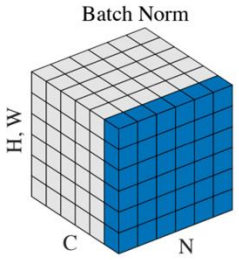
$$\mu_j = \frac{1}{N}\sum_{i=1}^{N} x_{i,j}$$ Per-channel mean, shape is D

$$\sigma_j^2 = \frac{1}{N}\sum_{i=1}^{N}(x_{i,j} - \mu_j)^2$$ Per-channel std, shape is D

$$\hat{x}_{i,j} = \frac{x_{i,j} - \mu_j}{\sqrt{\sigma_j^2 + \varepsilon}}$$ Normalized x, shape is NxD

$$y_{i,j} = \gamma_j \hat{x}_{i,j} + \beta_j$$ Output, shape is NxD

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n



Batch Normalization: Test Time

Input: $x : N \times D$

Learnable params: $\gamma, \beta : D$

Intermediates: $\mu, \sigma : D$

$\hat{x} : N \times D$

Output: $y : N \times D$

$\mu_j =$ (Running) average of values seen during training — Per-channel mean, shape is D

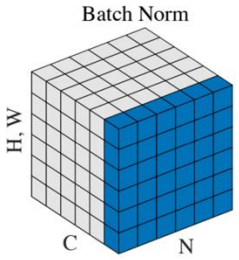
$\sigma_j^2 =$ (Running) average of values seen during training — Per-channel std, shape is D

$\hat{x}_{i,j} = \frac{x_{i,j} - \mu_j}{\sqrt{\sigma_j^2 + \varepsilon}}$ — Normalized x, shape is NxD

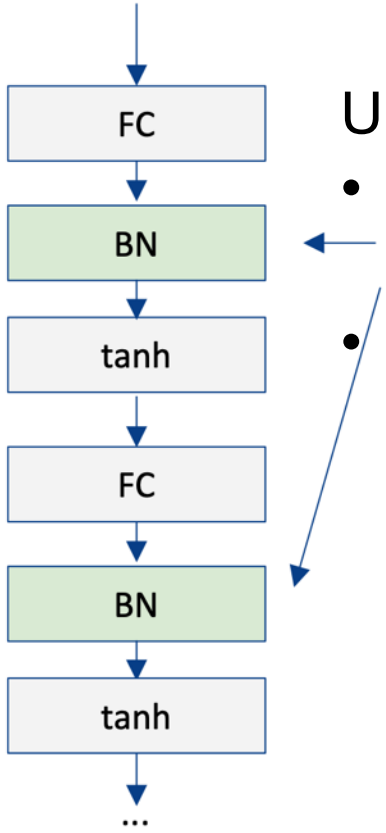
$y_{i,j} = \gamma_j \hat{x}_{i,j} + \beta_j$ — Output, shape is NxD

During testing batchnorm becomes a linear operator!
Can be fused with the previous fully-connected or conv layer

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n



Batch Normalization

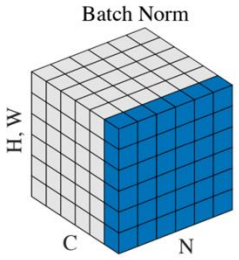


Usually added

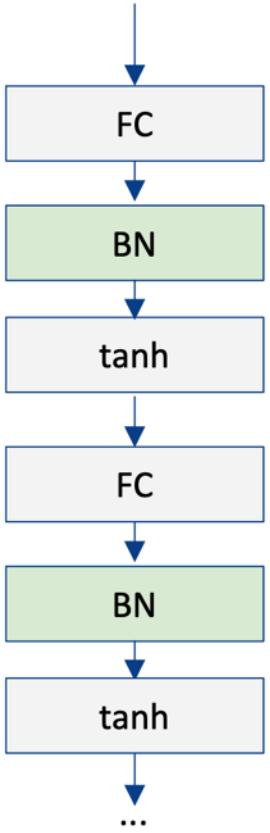
- after Fully Connected / (or Convolutional) layers, and
- before nonlinearity

$$\hat{x} = \frac{x - E[x]}{\sqrt{Var[x]}}$$

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

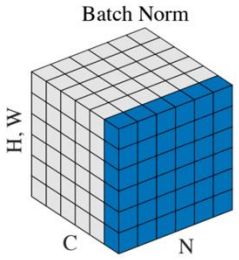


Batch Normalization

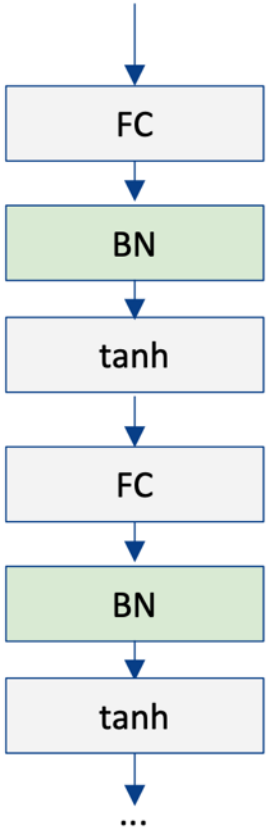


- Makes deep networks much easier to train!
- Allows higher learning rates, faster convergence
- Networks become more robust to initialization
- Acts as regularization during training
- Zero overhead at test-time: can be fused with conv!

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n



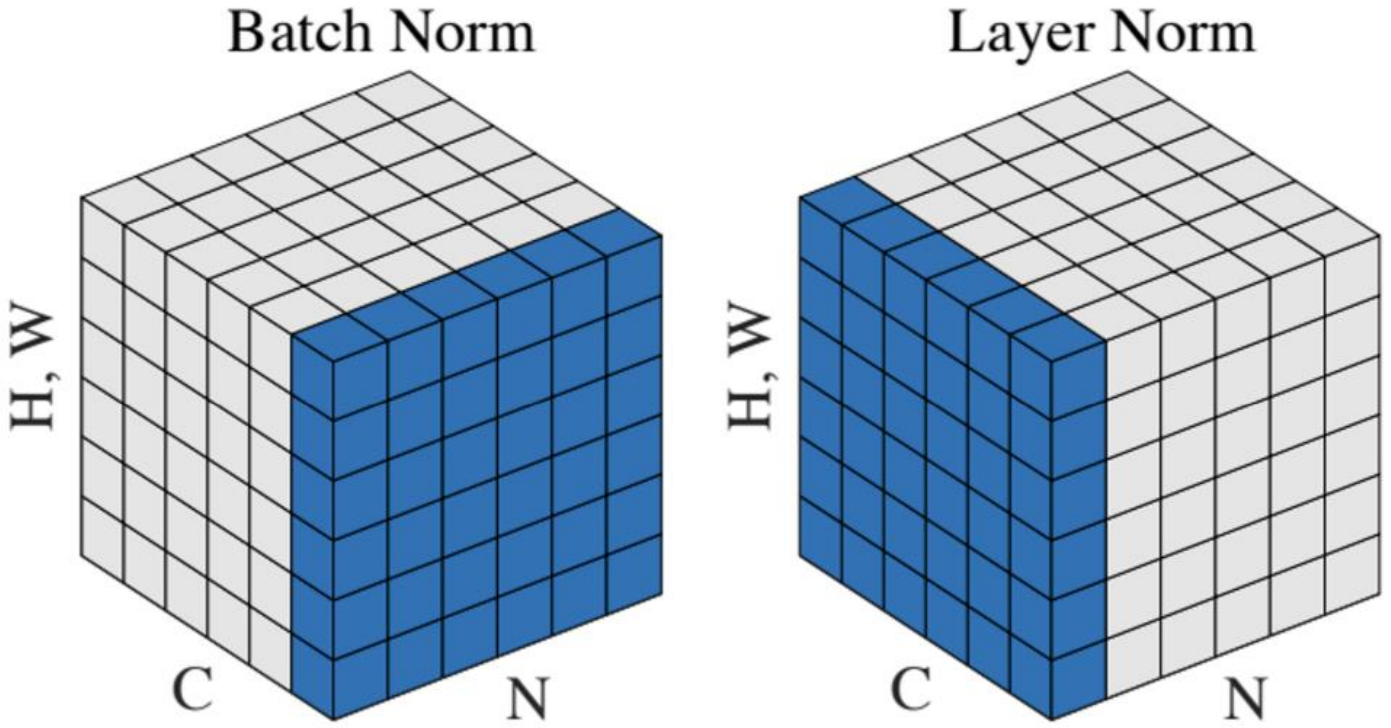
Batch Normalization



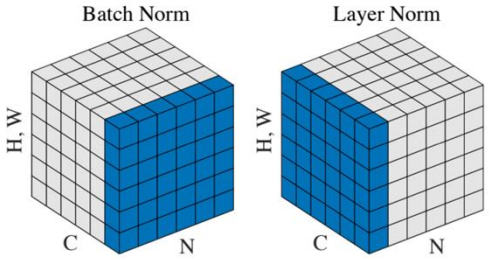
- Makes deep networks much easier to train!
- Allows higher learning rates, faster convergence
- Networks become more robust to initialization
- Acts as regularization during training
- Zero overhead at test-time: can be fused with conv!
- Not well-understood theoretically (yet)
- Behaves differently during training and testing: this is a very common source of bugs!

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Layer Normalization



[Wu and He, “Group Normalization”, ECCV 2018]



Layer Normalization

- Batch Normalization for fully-connected networks

$$x : \boldsymbol{N} \times D$$

Normalize

$$\mu, \sigma : 1 \times D$$

$$\gamma, \beta : 1 \times D$$

$$y = \frac{(x - \mu)}{\sigma}\gamma + \beta$$

- Layer Normalization for fully-connected networks
- Same behavior at train and test time!
  Used in RNNs &

$$x : N \times D$$

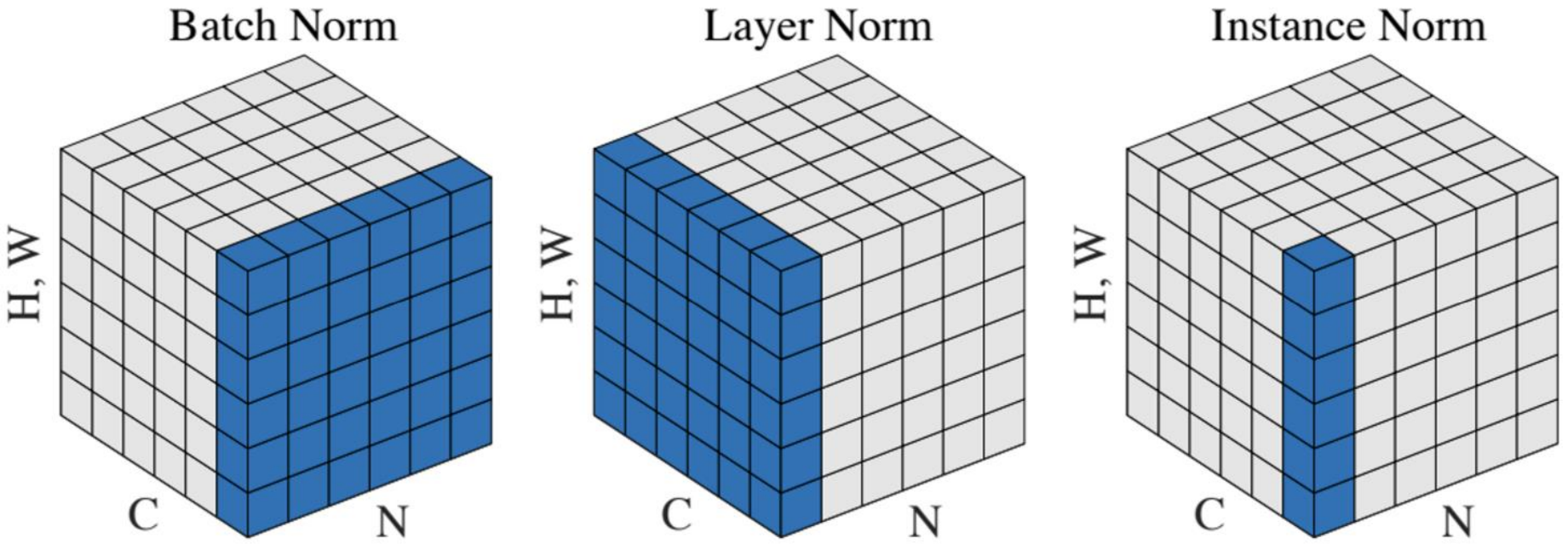
Normalize

$$\mu, \sigma : N \times 1$$

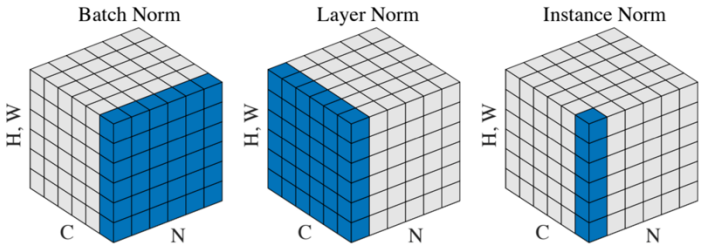
$$\gamma, \beta : 1 \times D$$

$$y = \frac{(x - \mu)}{\sigma}\gamma + \beta$$

Instance Normalization



[Wu and He, “Group Normalization”, ECCV 2018]



Instance Normalization

- Batch Normalization for convolutional networks

$$x : N \times C \times H \times W$$

Normalize

$$\mu, \sigma : 1 \times C \times 1 \times 1$$

$$\gamma, \beta : 1 \times C \times 1 \times 1$$

$$y = \frac{(x - \mu)}{\sigma}\gamma + \beta$$

- Instance normalization for convolutional networks

$$x : N \times C \times H \times W$$

Normalize

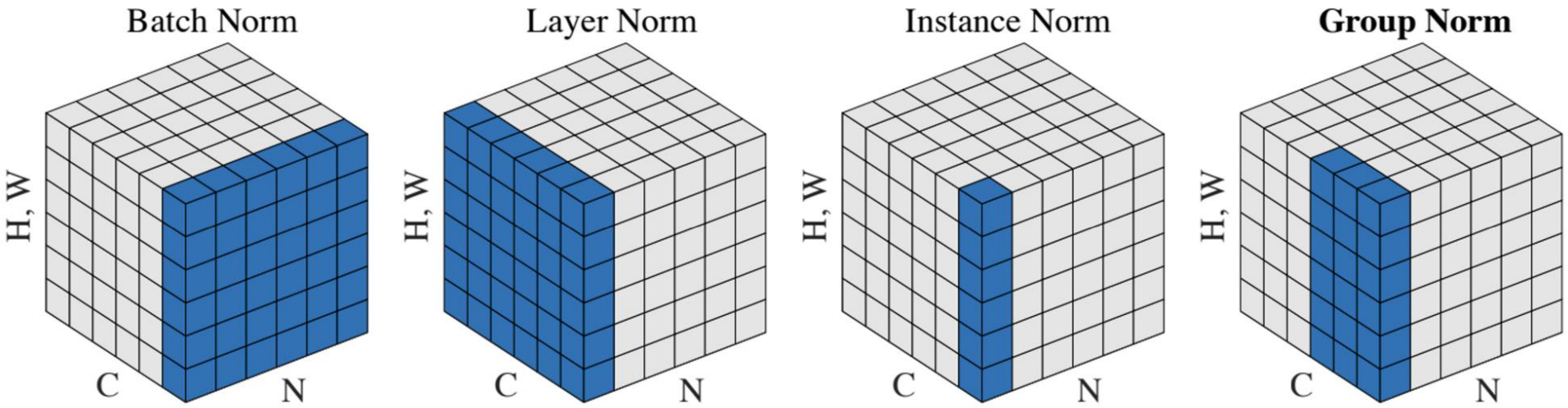
$$\mu, \sigma : N \times C \times 1 \times 1$$

$$\gamma, \beta : 1 \times C \times 1 \times 1$$

$$y = \frac{(x - \mu)}{\sigma}\gamma + \beta$$

Slide Credit: Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Group Normalization



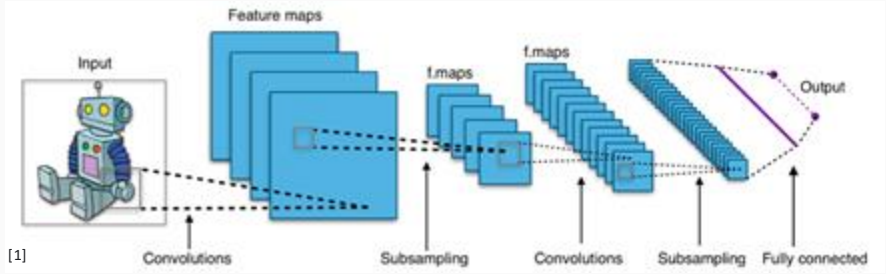
[Wu and He, “Group Normalization”, ECCV 2018]

Towards unification of fields

Towards unification of fields

Computer Vision

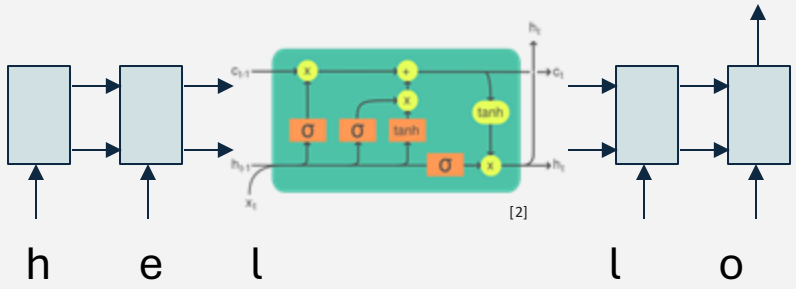
Convolutional NNs (+ResNets)



[1]

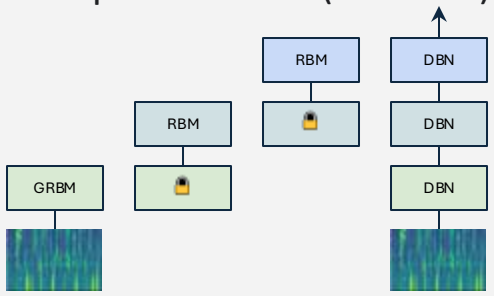
Natural Lang. Proc.

Recurrent NNs (+LSTMs)



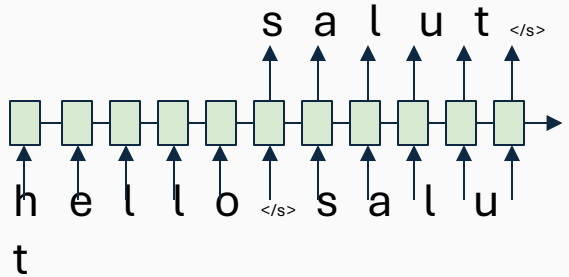
Speech

Deep Belief Nets (+non-DL)



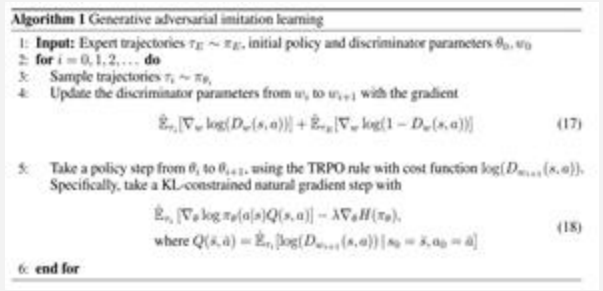
Translation

Seq2Seq



RL

BC/GAIL



Algorithm 1 Generative adversarial imitation learning

1: **Input:** Expert trajectories $\tau_E \sim \pi_E$, initial policy and discriminator parameters θ_0, w_0
2: **for** $i = 0, 1, 2, \ldots$ **do**
3: Sample trajectories $\tau_i \sim \pi_{\theta_i}$
4: Update the discriminator parameters from w_i to w_{i+1} with the gradient

$$\hat{\mathbb{E}}_{\tau_i}[\nabla_w \log(D_w(s,a))] + \hat{\mathbb{E}}_{\tau_E}[\nabla_w \log(1 - D_w(s,a))] \quad (17)$$

5: Take a policy step from θ_i to θ_{i+1}, using the TRPO rule with cost function $\log(D_{w_{i+1}}(s,a))$. Specifically, take a KL-constrained natural gradient step with

$$\hat{\mathbb{E}}_{\tau_i}[\nabla_\theta \log \pi_\theta(a|s) Q(s,a)] - \lambda \nabla_\theta H(\pi_\theta),$$
$$\text{where } Q(\hat{s},\hat{a}) = \hat{\mathbb{E}}_{\tau_i}[\log(D_{w_{i+1}}(s,a)) | s_0 = \hat{s}, a_0 = \hat{a}] \quad (18)$$

6: **end for**

[1] CNN image CC-BY-SA by Aphex34 for Wikipedia https://commons.wikimedia.org/wiki/File:Typical_cnn.png
[2] RNN image CC-BY-SA by GChe for Wikipedia https://commons.wikimedia.org/wiki/File:The_LSTM_Cell.svg

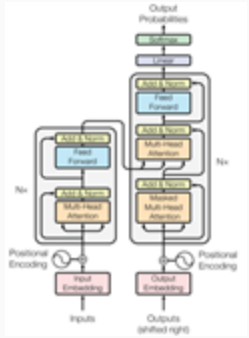
Slide credit: Lucas Beyer

The Transformer's takeover:

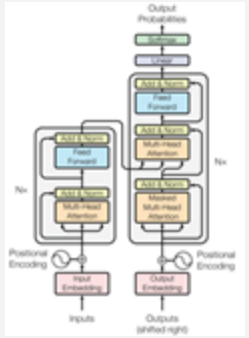
One community at a time

Next time: Transformers

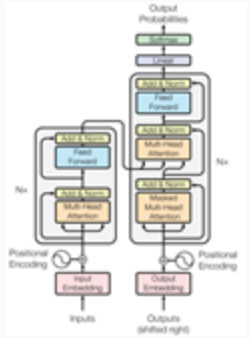
Computer Vision



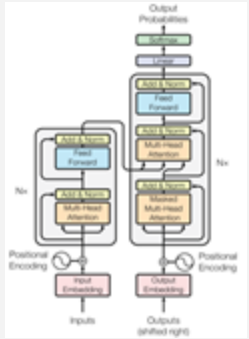
Natural Lang. Proc.



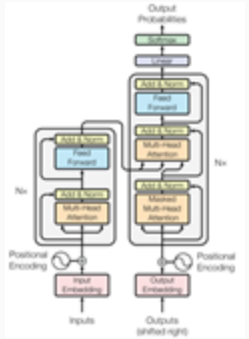
Reinf. Learning



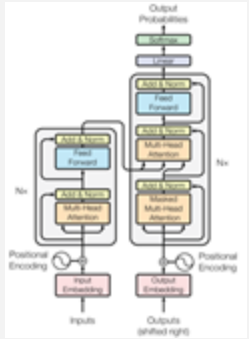
Speech



Translation



Graphs/Science



Slide credit: Lucas Beyer

Thank you