Vision & Language

Machine Learning Recap and CNNs Primer

Second Assignment

- Is out! Due soon!

Last Class

- Sequence-to-sequence (RNNs) for Machine Translation
- Learning to Align and Translate with Soft Attention
- Image Captioning (CNNs + RNNs): Show and Tell
- Image Captioning (CNNs + RNNs + Attention): Show Attend and Tell
- Attention is All you Need!
- Encoder Transformers: BERT
- Decoder Transformers: GPT-2 – maybe next class

Today's Class

- Course Project more Details
- Review of Assignment 01 / Recap on ML
- CNNs primer -- overview

Course Project

- Due this weekend.
- You should already have a group! Maybe try a last effort to be recruited (or recruit) team members on campuswire, there's a chat group.
- Your one-page to two-page project proposal is due this weekend.
- Groups from 1 to 3 students. (You can work on your own)
- Project effort should be equivalent to at least one of the assignments – keep in mind this semester ends a bit short – so think of your project as your Assignment #4 (for grad students), Assignment #3 (for undergrad students).
- So Project should be like an Assignment #4 – but it is yours. I won't push you to do anything but it should hopefully be relevant to the class topic – vision and language. e.g. not prediction of the weather using ML – or email spam classification.
- **Project Group Formation Enabled – Project Proposal due Soon!**

Describe and motivate your work

Scoring the Sentiment of Paper Reviews in Spanish with a Multilingual BERT Transformer

Vicente Ordonez, Ziyan Yang, Paola Cascante, Anshuman Suri
Department of Computer Science
University of Virginia

In this project we aim to examine the ability for text-based classification models to score paper reviews comments in Spanish for their general assessment of the paper. We will examine how modern text-based representations based on transformers such as BERT with strong sub-word tokenizers compare to previous representations based on bag-of-word representations with word-based tokenizers. Our results indicate that for small datasets, strong state-of-the-art models such as multilingual BERT don't attain significant gains in the low data regime of the Spanish Paper Reviews Dataset of Keith et al, which was collected from reviews submitted to a scientific conference. We examine the difference across these models and present detailed experimental results.

Introduce your work – motivate and contextualize

Introduction

Automated peer-review analysis is an active area of research where the goal is to enhance and improve the process of scientific progress. Our work applies methods from automatic text analysis and natural language processing to the analysis of peer reviews for a scientific conference. Our work builds on the dataset by Keith et al. 2017 [1], which collected a dataset of 405 reviews corresponding to XX papers. These were collected between 20XX and 20XX for the conference XXXX. We additionally applied our method a second dataset consisting of….

Our work applies the transformer-based BERT (Devlin et al 2018 [2]) representation model trained on multi-lingual data which includes Spanish. We use the pretrained model and reference implementation from the huggingface transformers library and their associated subword tokenizers [3]. We additionally compare simpler models based on bag-of-words using word-based tokenizers and find that generally for this small scale problem we obtained no significant gains under a linear classifier.

We present detailed analysis of three models: (1) a text-based linear classifier using bag-of-words encoding and word-tokenization, (2) a text-based classifier using bag-of-words encoding and sub-word tokenization (BPE), and (3) a text-based linear classifier with a BERT pretrained encoding and sub-word tokenization (BPE) ….

Do a literature review even if small

Related Work

Text based classifiers: There is a lot of work in text based classifiers, traditionally people have used bag-of-words or bag-of-ngrams encoding [cite][cite], then people adopted continuous bag of words models with pretrained word embeddings [cite] [cite]. More recent work has adopted transformer-based features with robust tokenization tools in a number of applications [cite][cite][cite]. For example people have used to categorize movie reviews [cite][cite], to categorize legal documents [cite], or to score the sentiment of product reviews on e-commerce websites [cite][cite]. In our work we aim to classify scientific reviews using these methods.

Other works have attempted before us to perform scientific review categorization using text-based methods. For example the work of [cite] has explored this problem but they relied only on bag-of-word features while our work also considers the more recently proposed BERT encoding features. Other work on this general area include scoring the text of the paper itself to determine its quality [cite], in our work we instead judge a paper's quality by using the review text provided by referees. The work of [cite] has also explored this problem but this work was limited to English while our work is focused on Spanish. The only other works we found in this area using Spanish are [cite][cite][cite], although [cite] did not use paper reviews but movie reviews.

Describe your methods

Methods

First we introduce the different tokenizers we use in this work, then encoders, then our classifier and loss function:

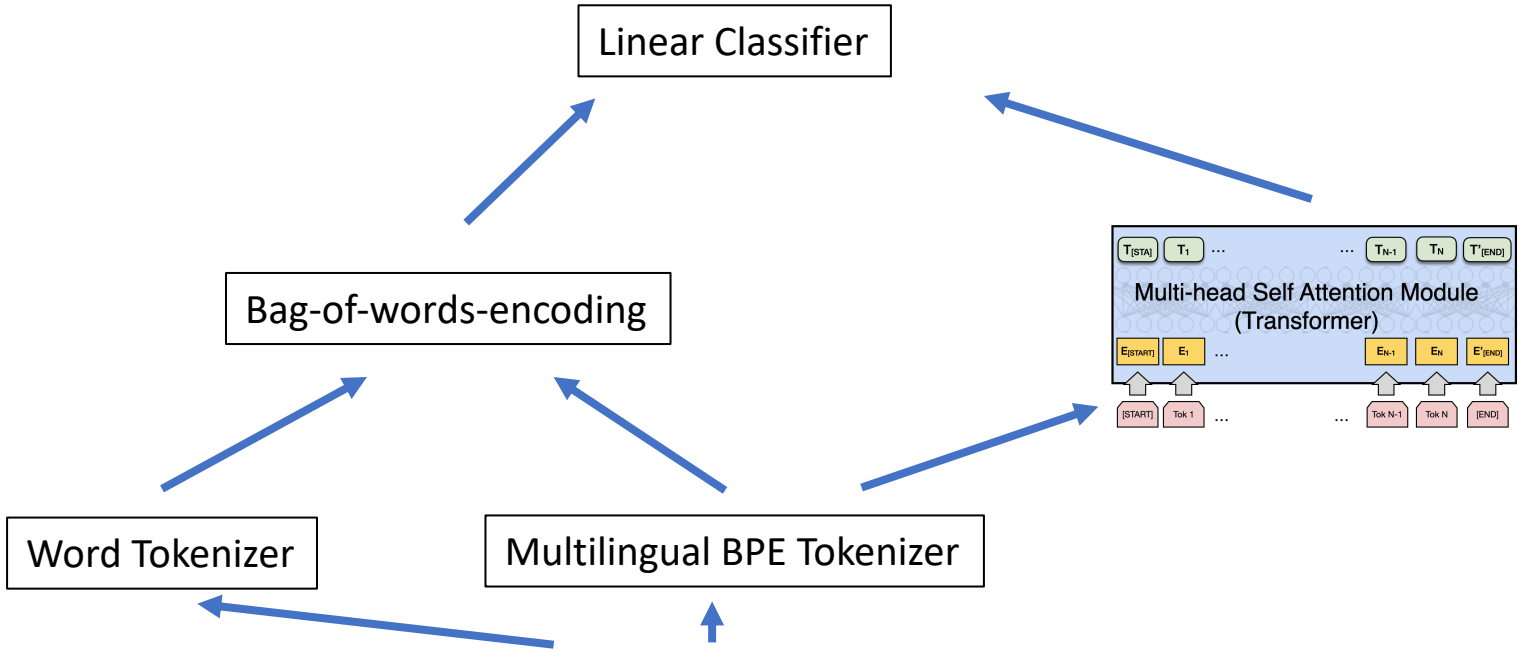
Tokenizers: We use two types of tokenizers, first a simple word tokenizers where we split the sentences into words using a spaces, which generally works well for Spanish which separates words with spaces. Then we also adopt a more general BPE tokenizers as included in the huggingface tokenizers library [cite].

Encodings: We use two different types of encoders, the bag of words encoder with single tokens and binary representation (no counts), and a BERT pretrained model as included in the huggingface transformers library [cite] which contains XXX parameters and was pretrained using a large multilingual dataset consisting of…

Classifier: We use a linear classifier with an input size matching the size of the corresponding encoding and an output sigmoid function which will be trained using a binary cross entropy loss function.

Our three possible models under evaluation are illustrated in Figure 01, where we show all the ways these components are combined to form three models (a), (b) and (c).

Use Figures and Diagrams to explain!



(a) Word-tokenizer + BoW (b) Subword-BPE tokenizer + BoW (c) Subword-BPE tokenizer + BERT

Figure 1. Overview of our work, we use three types of classifier (a) word-tokenizer with bag-of-words encoding, and (b) subword BPE tokenizer with bag-of-tokens encoding, and (c) BERT encoding with sub-word BPE tokenizer.

Describe your experiments

Experiments

We trained our classifiers until convergence for a maximum of XXX epochs using a batch size of XXX and a learning rate of XXX. The dataset was split into a training, validation and test splits of sizes XXX, YYY, and ZZZ. We experimented with various batch sizes and learning rates while developing our model and chose the parameters that led to the best accuracy on our validation set. In all of our experiments we used a Google Colab instance with one GPU and the experiments for each of our models (a), (b) and (c) took XXX minutes, YYY minutes, and ZZZ minutes respectively. For the BERT encoding models we pre-computed the encodings for accelerated execution time. However, fully tuning the BERT model is something we did not perform but that might lead to improved results.

We also performed a second experiment on a larger dataset consisting of ….

Describe your Results

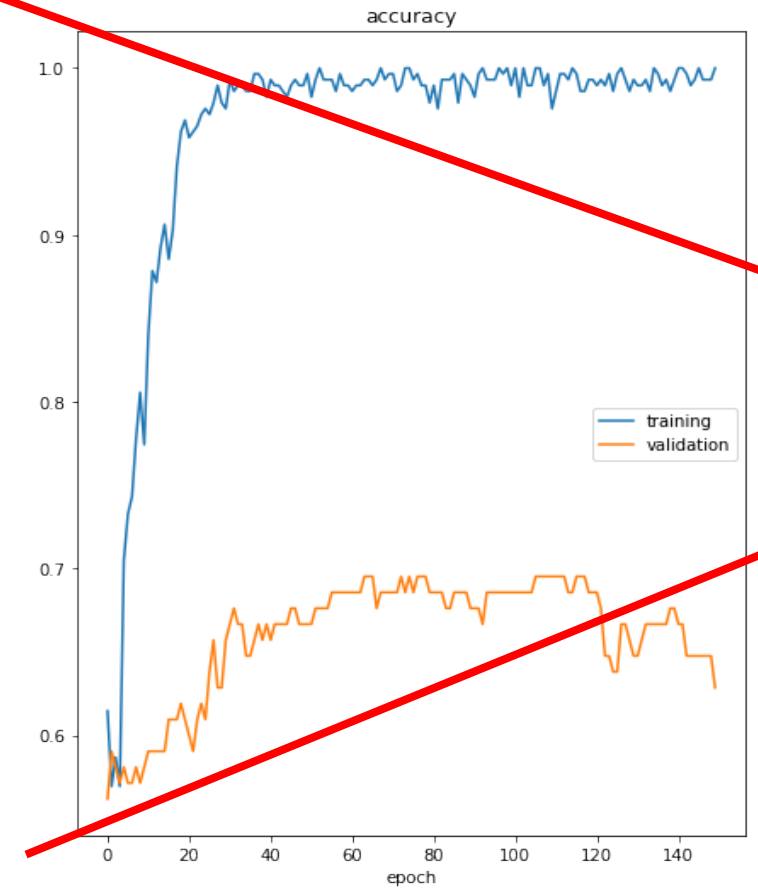
Results

We found that the classifier using word tokenizers and bag of words encoding performed the best despite its simplicity. We think this happens because the dataset is rather small and either larger or smaller models tend to overfit or underfit to the training data. However, more experiments are perhaps needed to verify this observation as well as experiments on larger datasets.

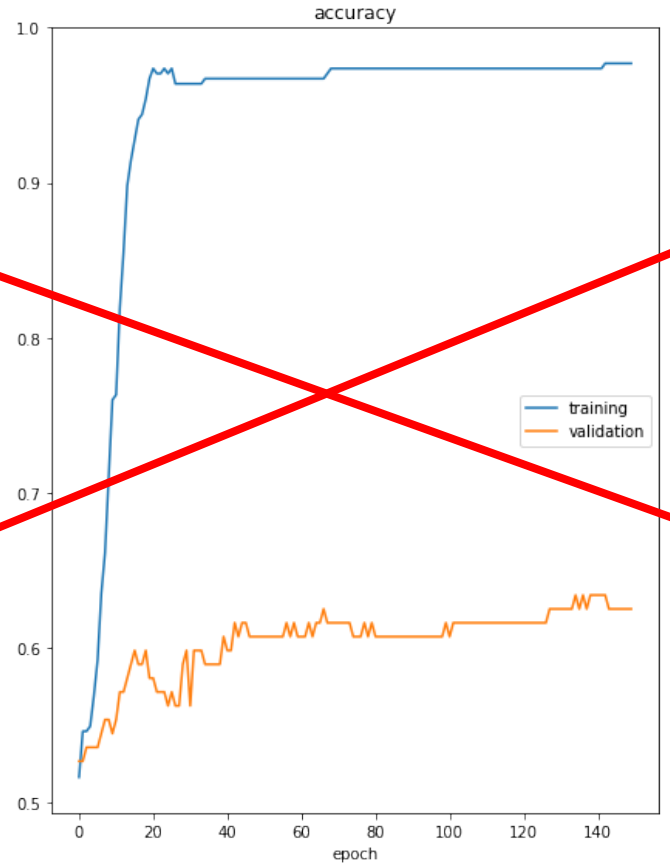
Figure 02 presents results for accuracy on our validation set while developing our model across many epochs of training. The first two models were trained for 150 epochs while the last model was trained for 300 epochs.

The dataset consists of 55% of examples in negative category and 45% of examples in the positive category so obtaining an accuracy bigger than 60% as is the case in all of our experiments is a positive result. Particularly our stronger classifier obtained an accuracy close to 70% in accuracy. We also analyzed the words that more frequently led to a positive categorization and we found words such as “bueno, bien” (good, well) correlated with positive examples, and negative words such as “no, ni”, (not, neither) as correlated with negative reviews. In addition to that we observed …

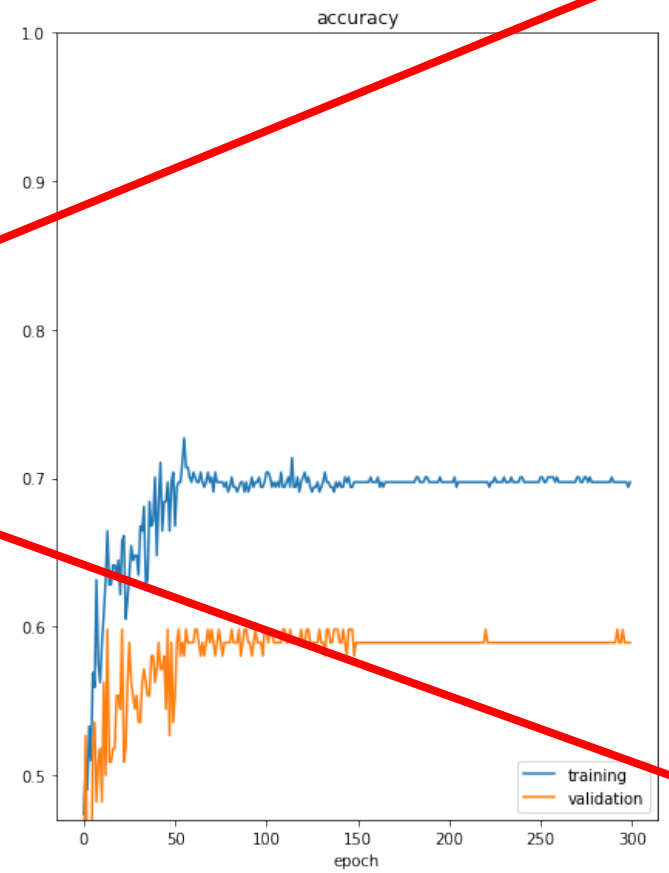
Present your results: Also, what not to do to present your results!



(a) Word-tokenizer + BoW



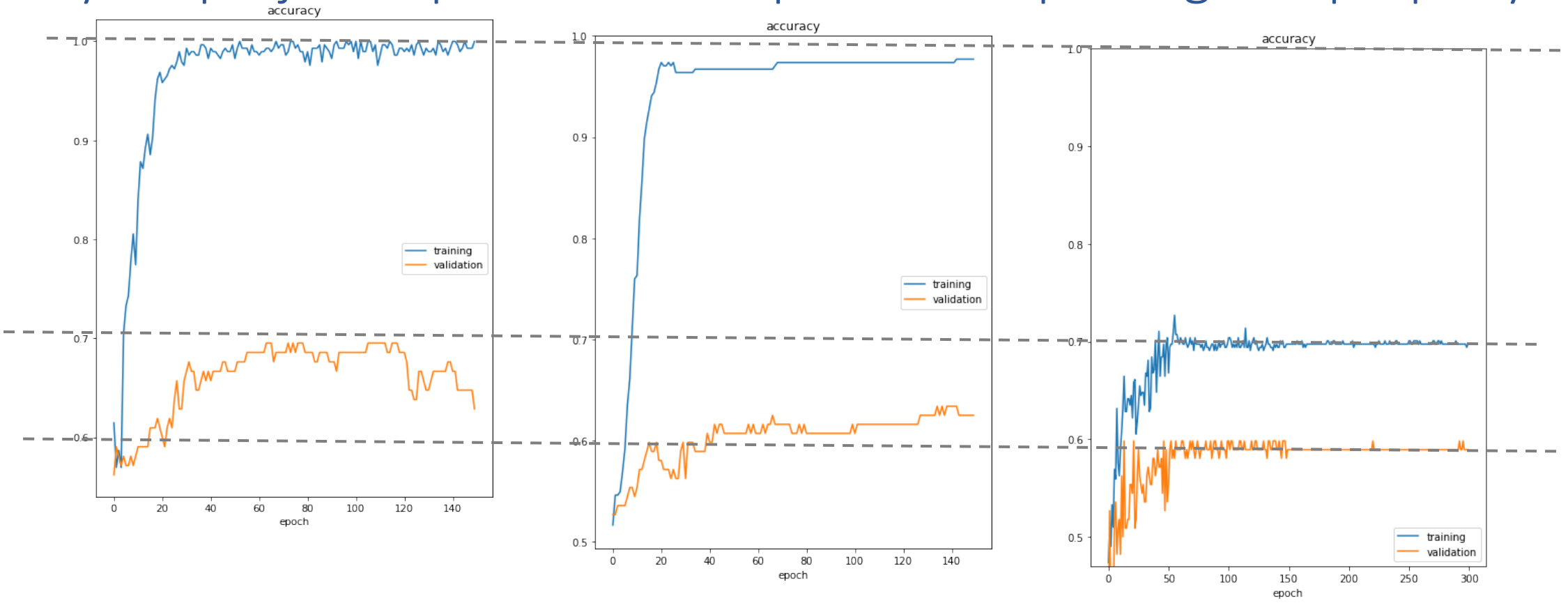
(b) Subword-BPE tokenizer + BoW



(c) Subword-BPE tokenizer + BERT

Figure 1. Results for our three classifiers (a) word-tokenizer with bag-of-words encoding, and (b) subword BPE tokenizer with bag-of-tokens encoding, and (c) BERT encoding with sub-word BPE tokenizer.

One thing is to plot things to get help for yourself and one for your project report: Learn to plot and export figures properly



(a) Word-tokenizer + BoW (b) Subword-BPE tokenizer + BoW (c) Subword-BPE tokenizer + BERT

Figure 1. Results for our three classifiers (a) word-tokenizer with bag-of-words encoding, and (b) subword BPE tokenizer with bag-of-tokens encoding, and (c) BERT encoding with sub-word BPE tokenizer.

Show Actual Results! Not just plots!

- Examples of positive reviews scored by our model.
- Examples of negative reviews scored by our model.
- Examples of mistakes our model makes. Maybe discussion on why these mistakes happened?

Write a short conclusion and future work

Conclusion

Automatic analysis of paper reviews is a challenging problem especially when not a lot of data is available for training. Perhaps it is also challenging when a language-agnostic tool such as multilingual BERT is used, as it has been show that language-specific models tend to lead to better results in other tasks. These are possible avenues of further investigation and improvement. Our work intends to make a contribution in this general direction. …..

ILSVRC:
Imagenet Large Scale Visual Recognition Challenge
[Russakovsky et al 2014]

The Problem: Classification

Classify an image into 1000 possible classes:
e.g. Abyssinian cat, Bulldog, French Terrier, Cormorant, Chickadee, red fox, banjo, barbell, hourglass, knot, maze, viaduct, etc.

cat, tabby cat (0.71)
Egyptian cat (0.22)
red fox (0.11)
…..

The Data: ILSVRC

Imagenet Large Scale Visual Recognition Challenge (ILSVRC): Annual Competition

1000 Categories

~1000 training images per Category

~1 million images in total for training

~50k images for validation

Only images released for the test set but no annotations,
evaluation is performed centrally by the organizers (max 2 per week)

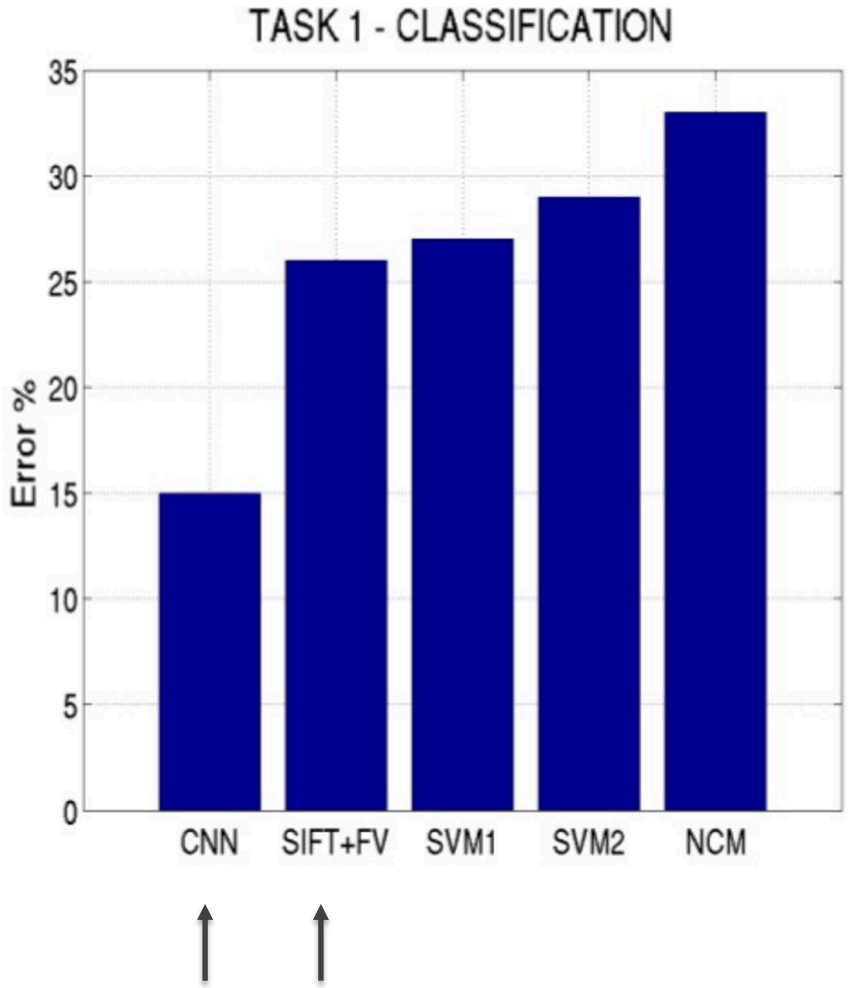
The Evaluation Metric: Top K-error

True label: Abyssinian cat

Top-1 error: 1.0	Top-1 accuracy: 0.0
Top-2 error: 1.0	Top-2 accuracy: 0.0
Top-3 error: 1.0	Top-3 accuracy: 0.0
Top-4 error: 0.0	Top-4 accuracy: 1.0
Top-5 error: 0.0	Top-5 accuracy: 1.0

cat, tabby cat (0.61)
Egyptian cat (0.22)
red fox (0.11)
Abyssinian cat (0.10)
French terrier (0.03)
…..

Top-5 error on this competition (2012)



Alexnet (Krizhevsky et al NIPS 2012)

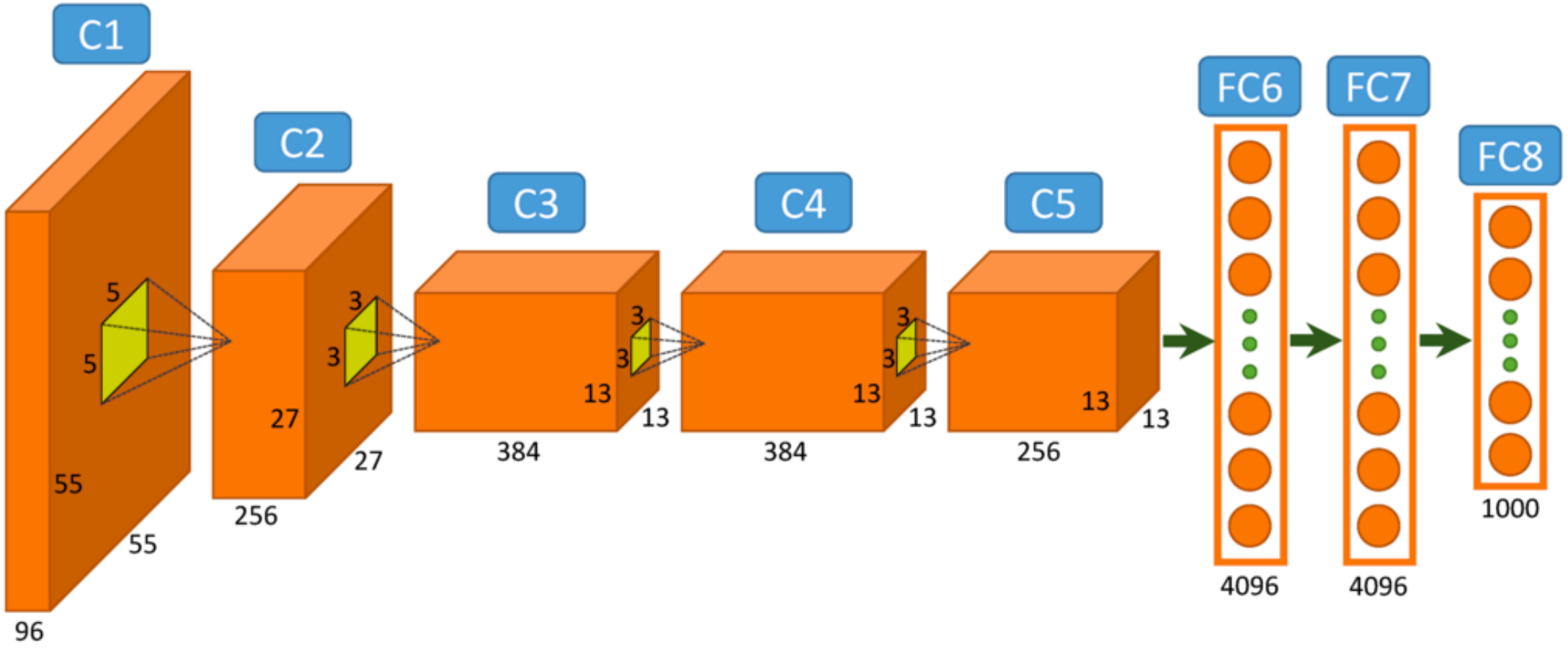
ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

Alexnet



https://www.saagie.com/fr/blog/object-detection-part1

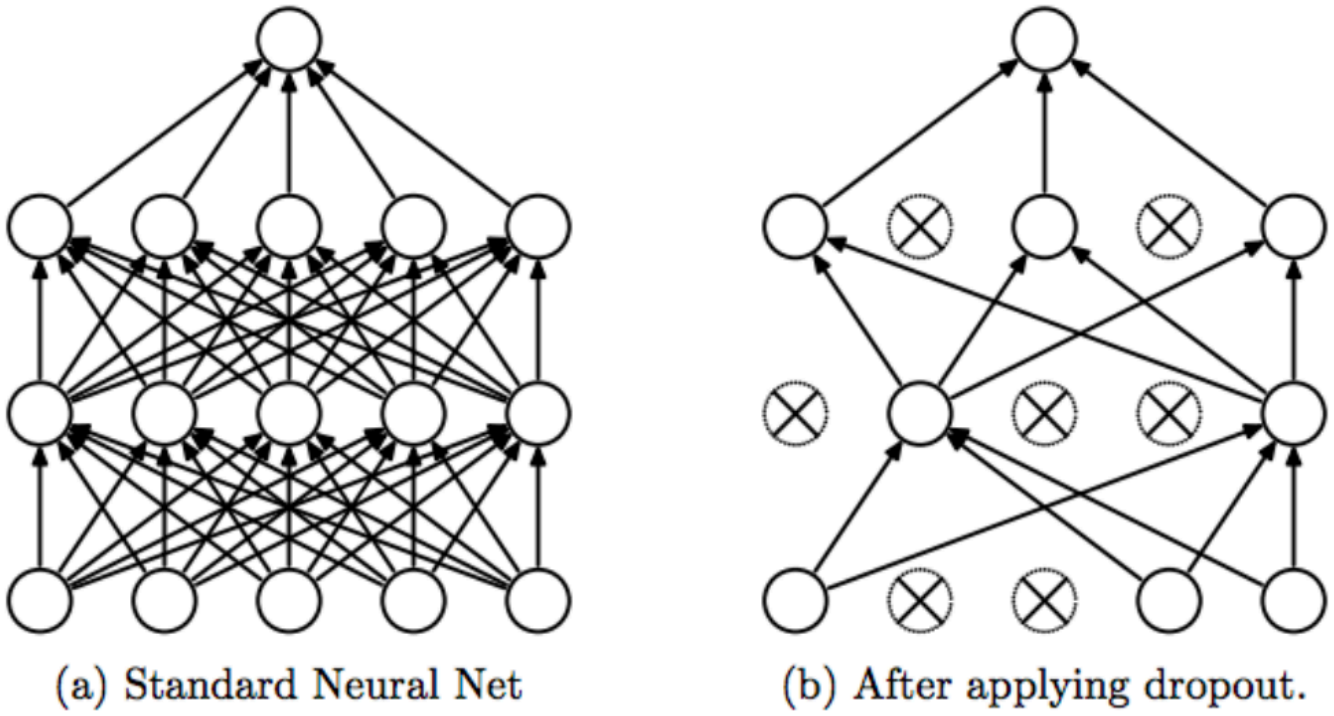
Pytorch Code for Alexnet

- In-class analysis

  https://github.com/pytorch/vision/blob/master/torchvision/models/alexnet.py

Dropout Layer

Happens for every batch for a different set of connections only during training



Srivastava et al 2014

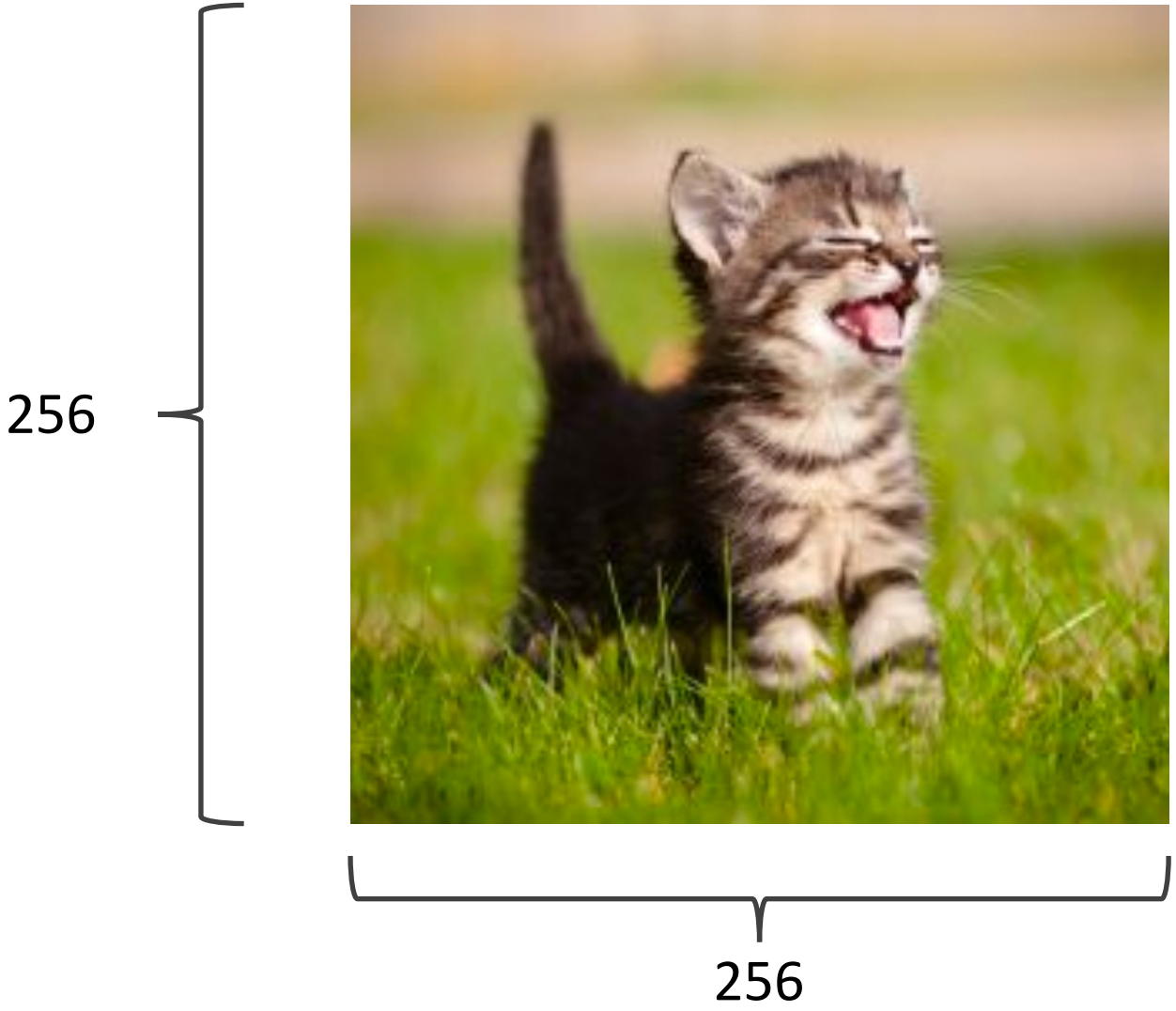
Important

model.train()

model.eval()

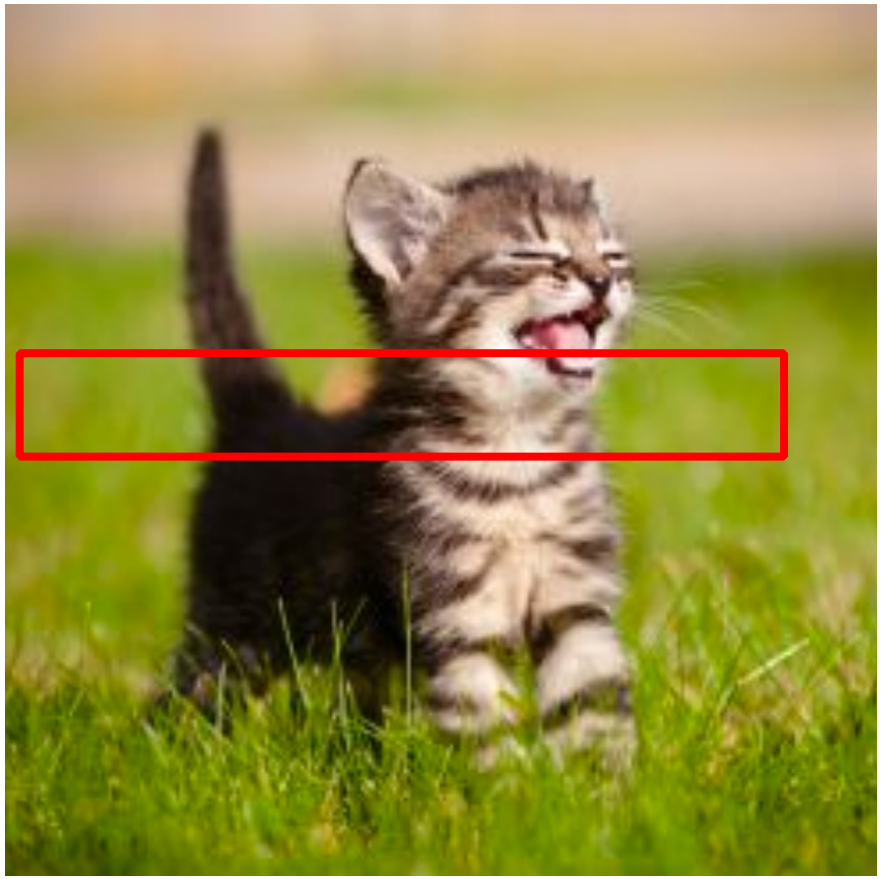
Preprocessing and Data Augmentation

Preprocessing and Data Augmentation



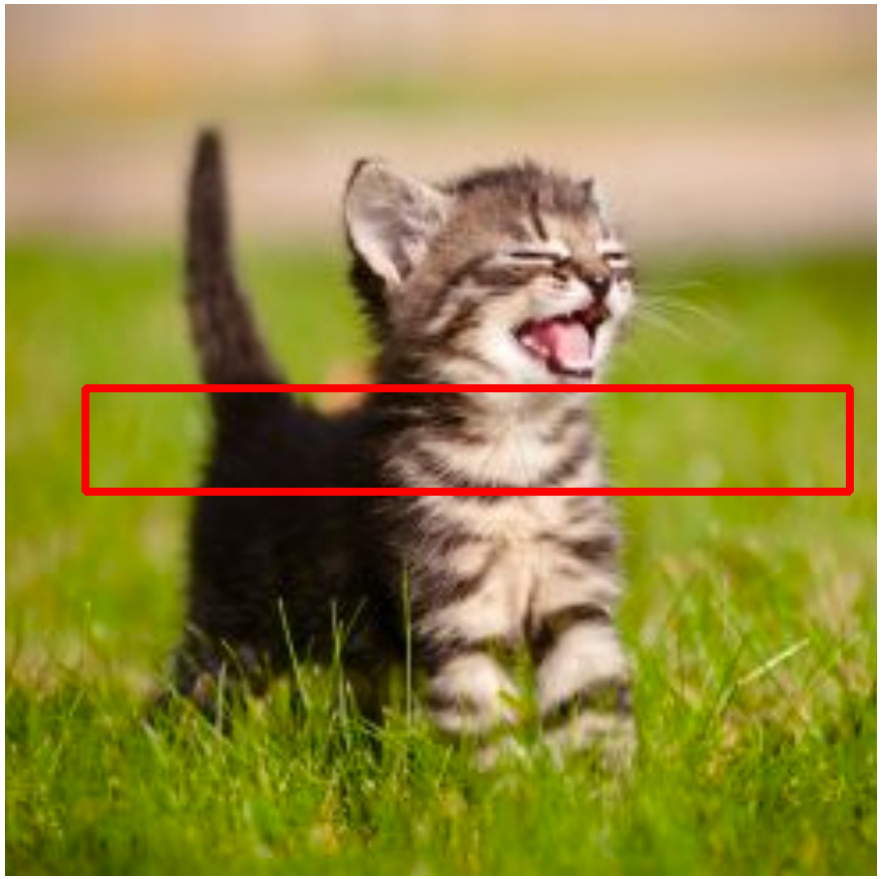
Preprocessing and Data Augmentation

224x224



Preprocessing and Data Augmentation

224x224



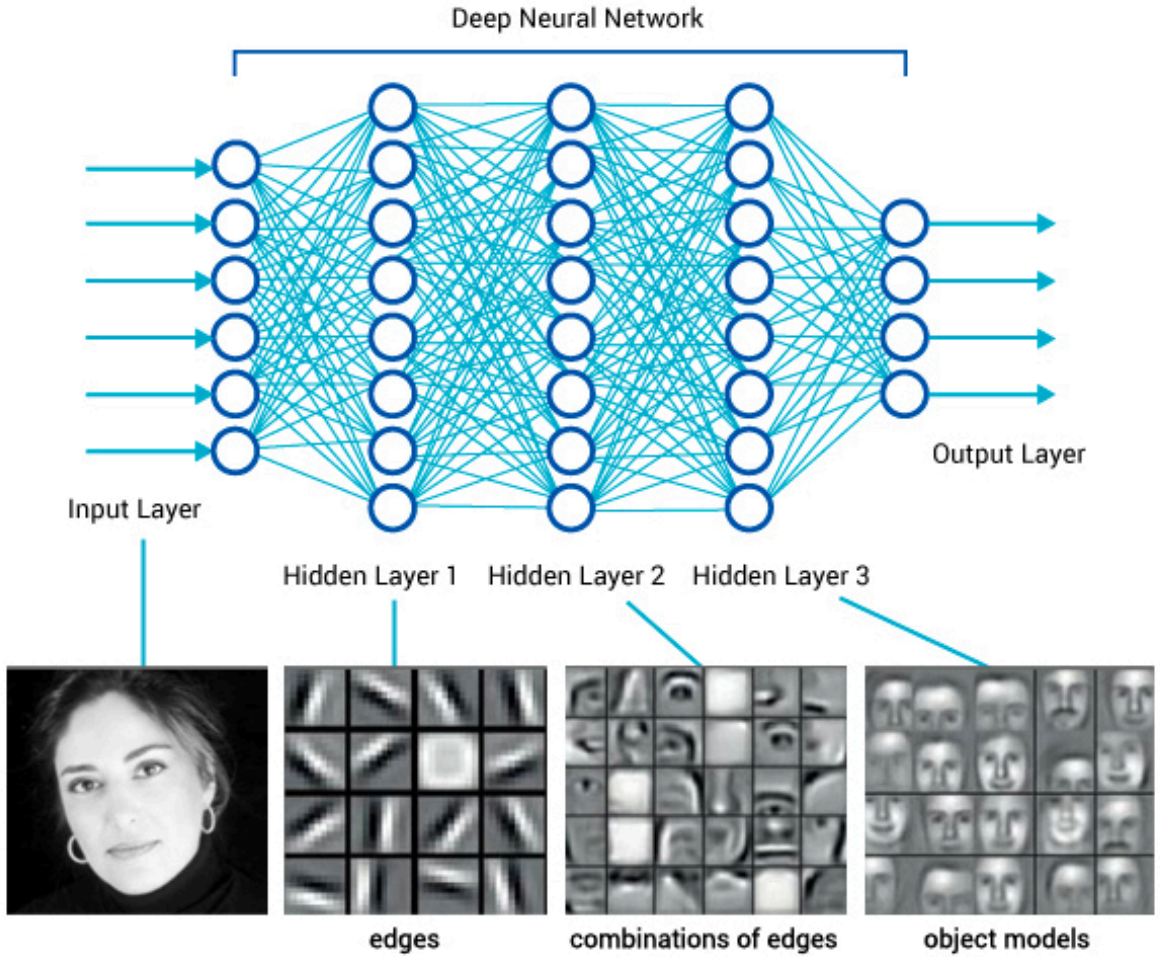
True label: Abyssinian cat

Some Important Aspects

- Using ReLUs instead of Sigmoid or Tanh
- Momentum + Weight Decay
- Dropout (Randomly sets Unit outputs to zero during training)
- GPU Computation!

Model	**Top-1**	**Top-5**
Sparse coding [2]	*47.1%*	*28.2%*
SIFT + FVs [24]	*45.7%*	*25.7%*
CNN	**37.5%**	**17.0%**

What is happening?



https://www.saagie.com/fr/blog/object-detection-part1

SIFT + FV + SVM (or softmax)

Feature extraction (SIFT)

Feature encoding (Fisher vectors)

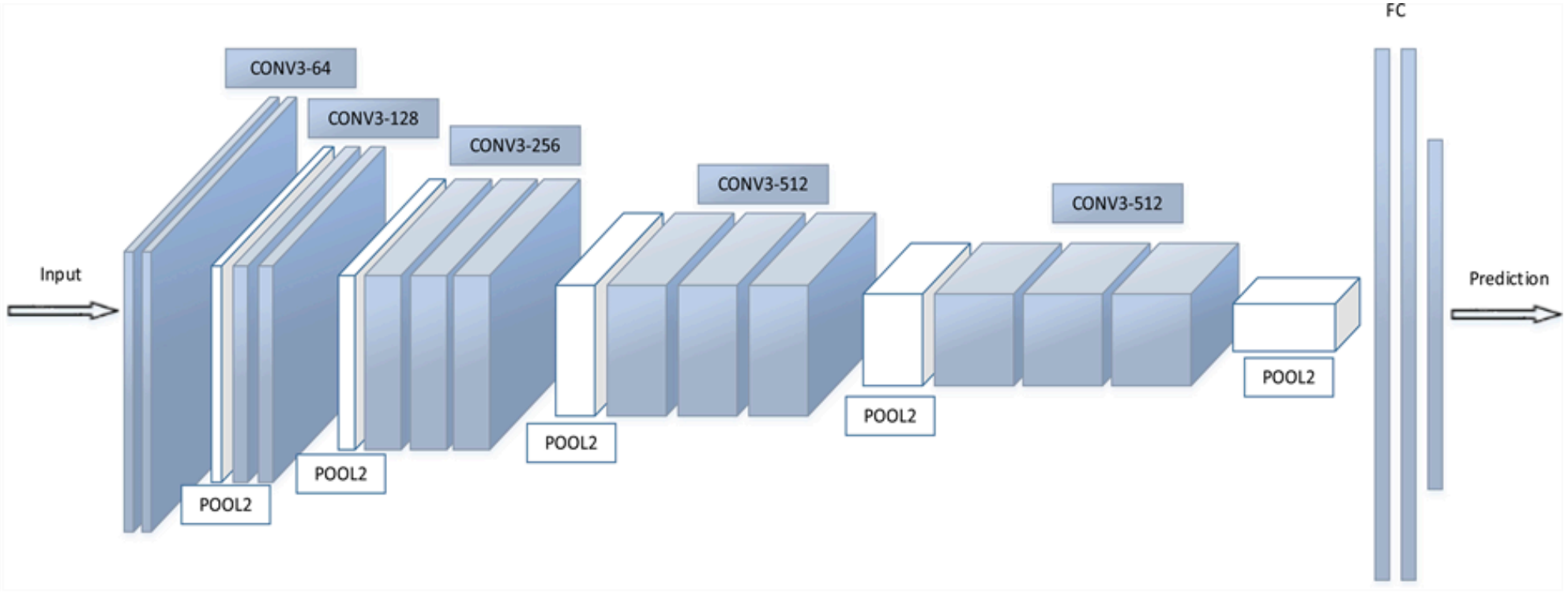
Classification (SVM or softmax)

Deep Learning

Convolutional Network
(includes both feature extraction and classifier)

VGG Network

Top-5:



https://github.com/pytorch/vision/blob/master/torchvision/models/vgg.py

Simonyan and Zisserman, 2014.

https://arxiv.org/pdf/1409.1556.pdf

BatchNormalization Layer

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

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

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

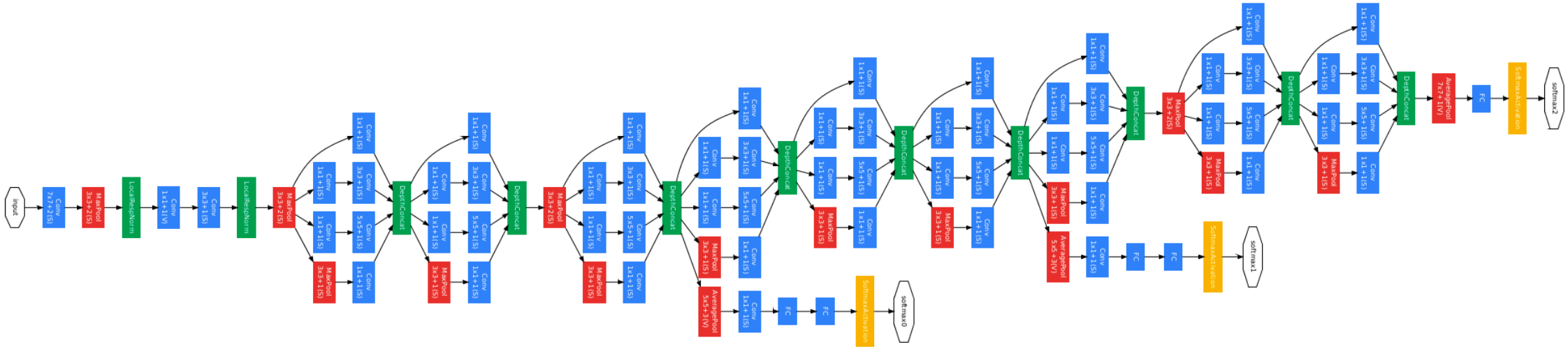
$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

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

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

https://arxiv.org/abs/1502.03167

GoogLeNet

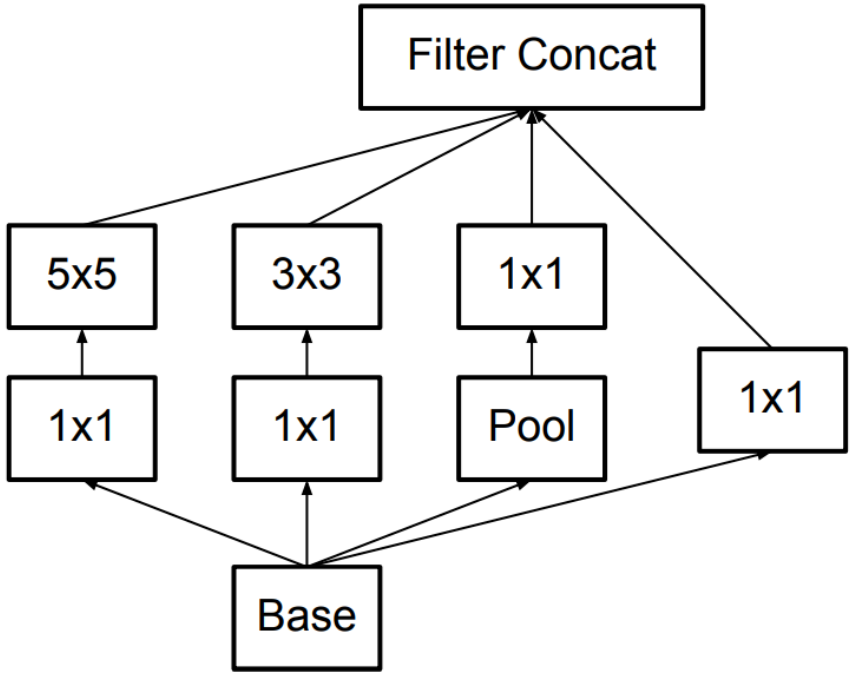


https://github.com/kuangliu/pytorch-cifar/blob/master/models/googlenet.py

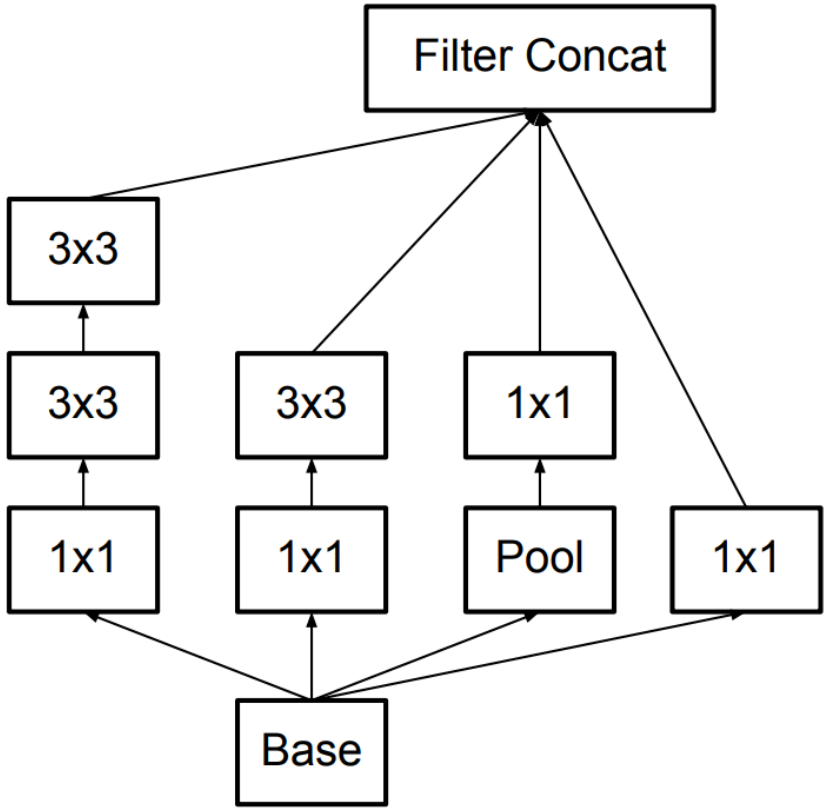
Szegedy et al. 2014

https://www.cs.unc.edu/~wliu/papers/GoogLeNet.pdf

Further Refinements – Inception v3, e.g.



GoogLeNet (Inceptionv1)



Inception v3

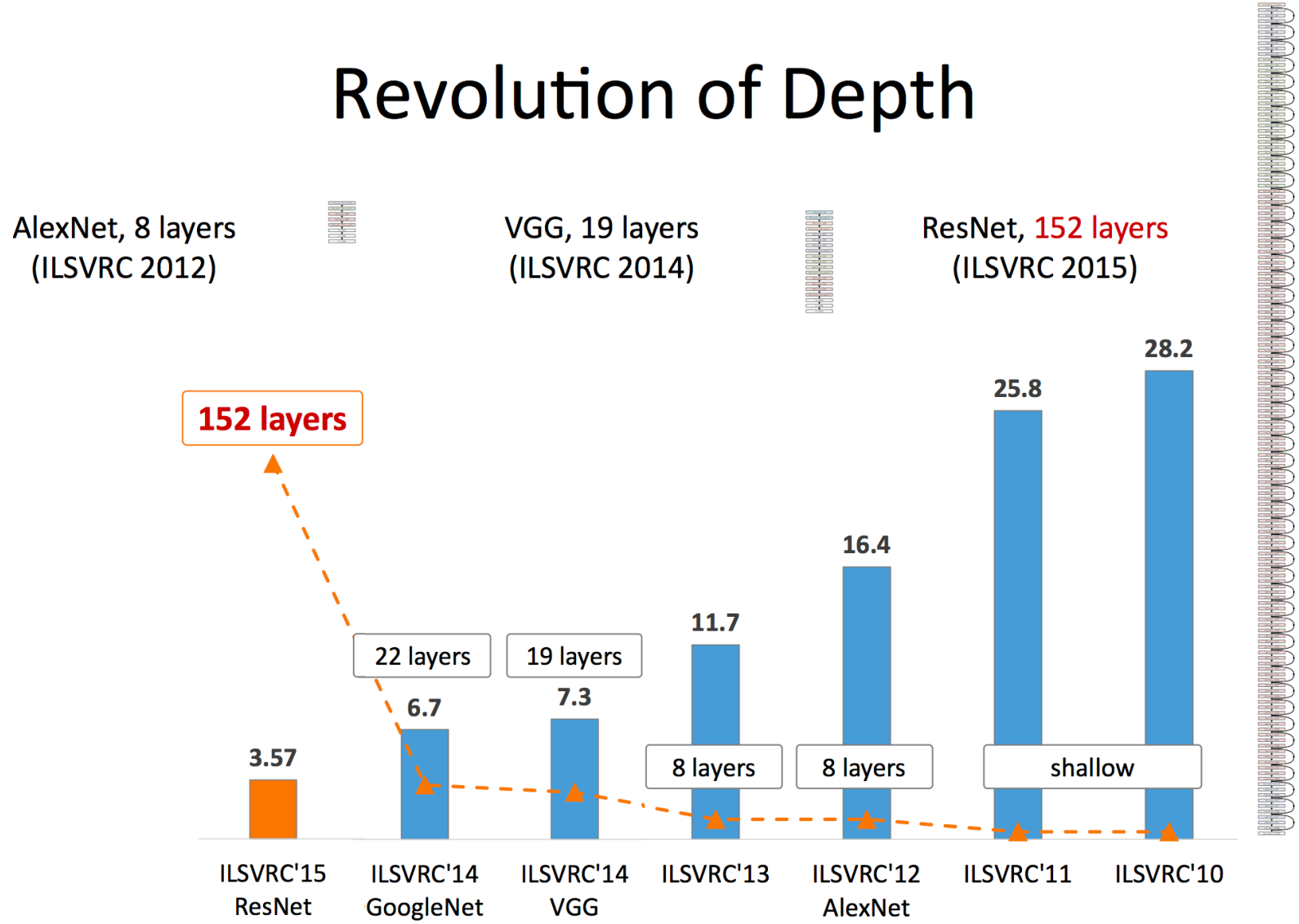
ResNet (He et al CVPR 2016)

Sorry, does not fit in slide.

http://felixlaumon.github.io/assets/kaggle-right-whale/resnet.png

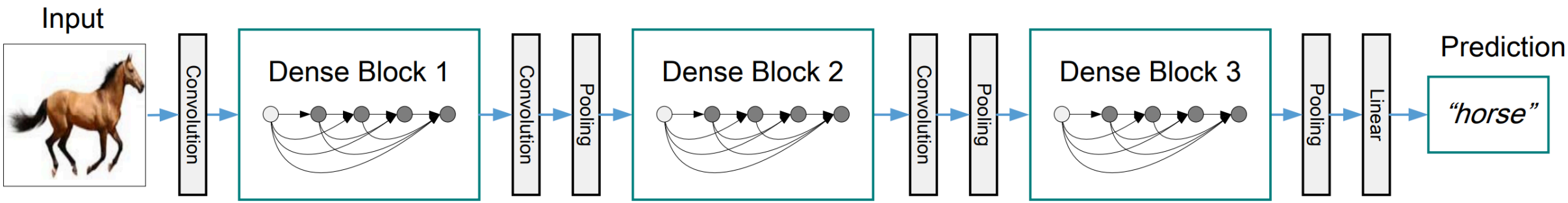
https://github.com/pytorch/vision/blob/master/torchvision/models/resnet.py

Revolution of Depth

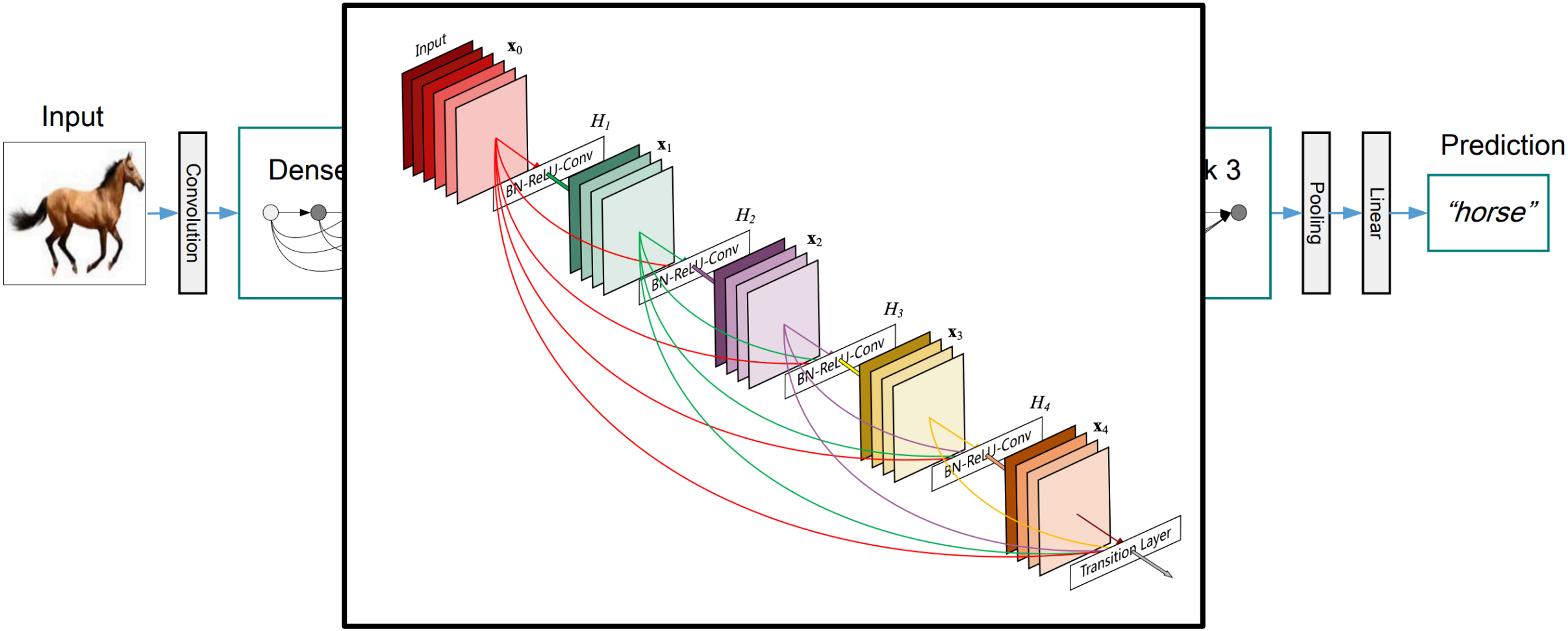


Slide by Mohammad Rastegari

Densenet

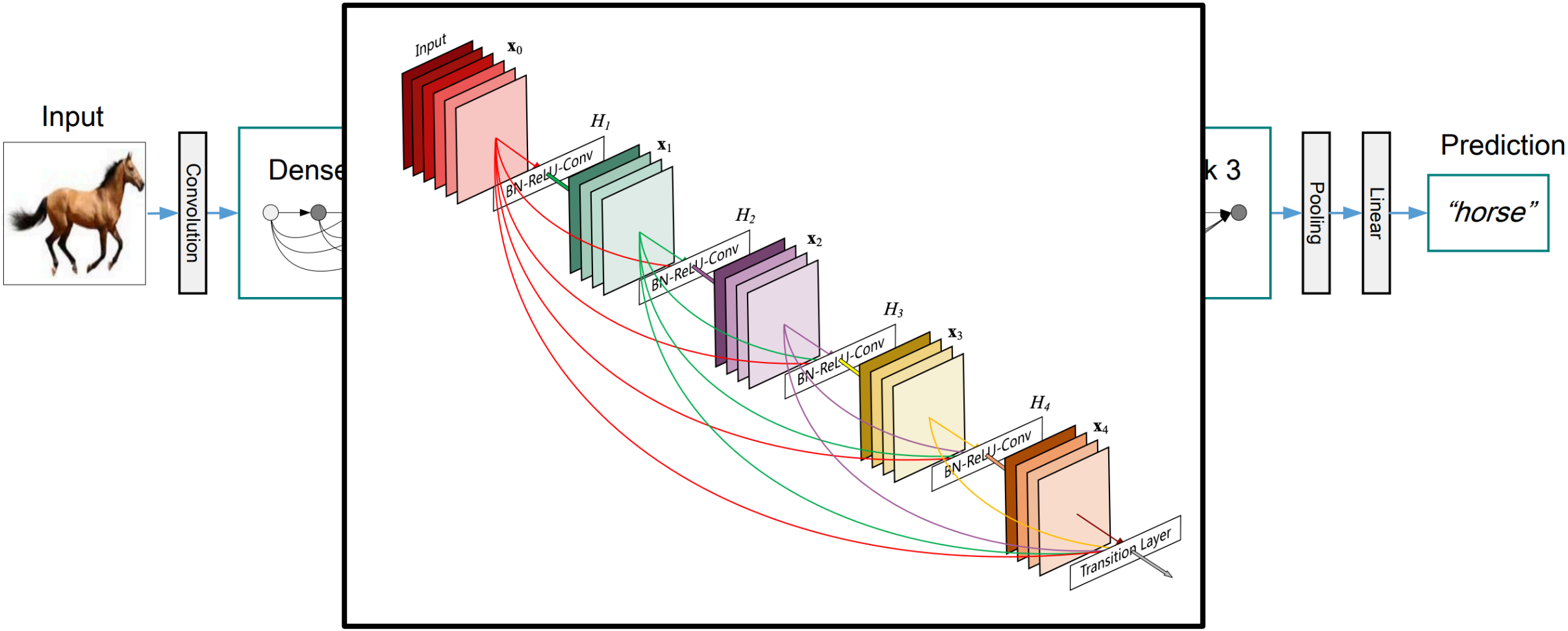

Input
Convolution
Dense Block 1
Convolution
Pooling
Dense Block 2
Convolution
Pooling
Dense Block 3
Pooling
Linear
Prediction
“horse”

Densenet



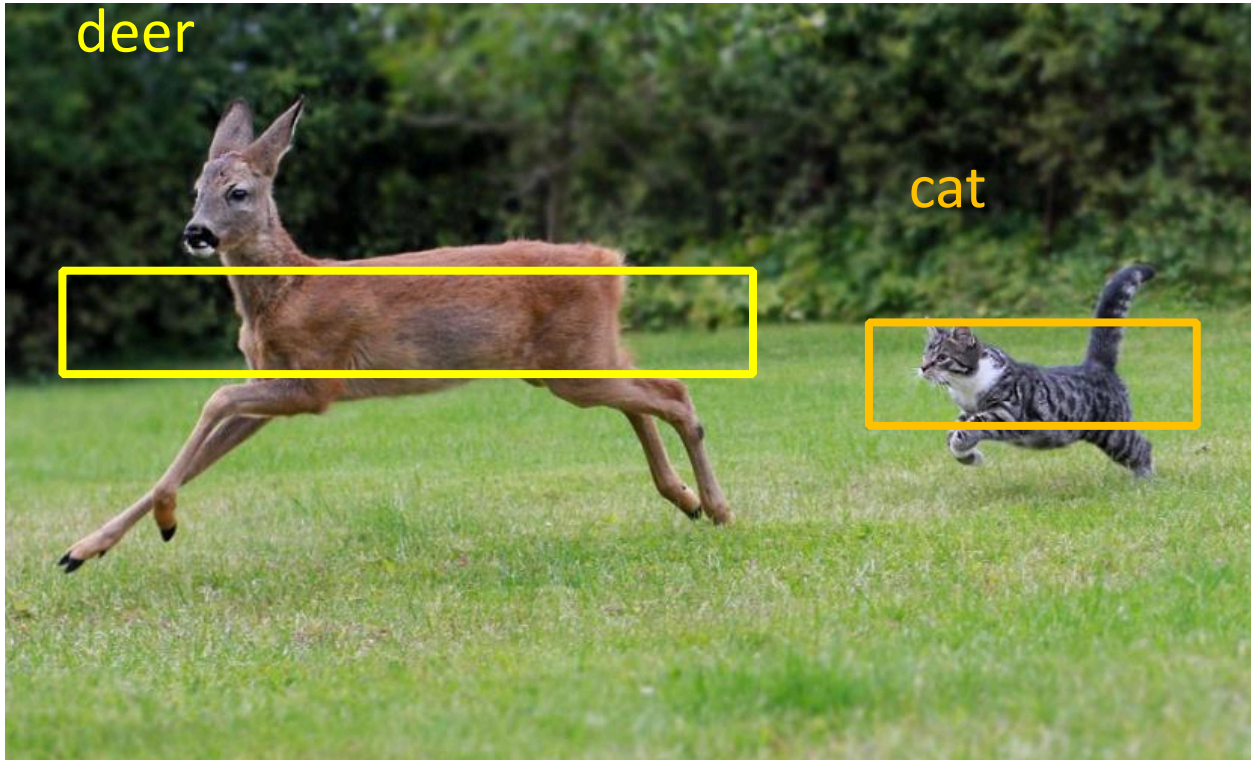
https://arxiv.org/pdf/1608.06993.pdf

Densenet

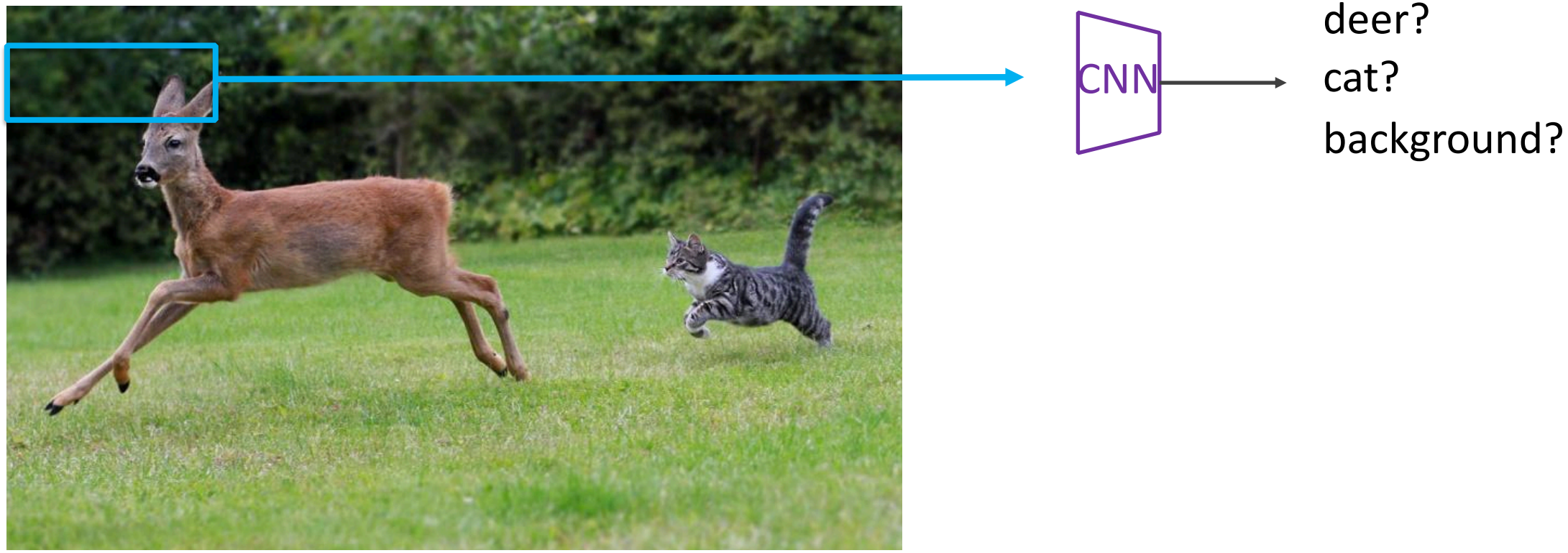


https://arxiv.org/pdf/1608.06993.pdf

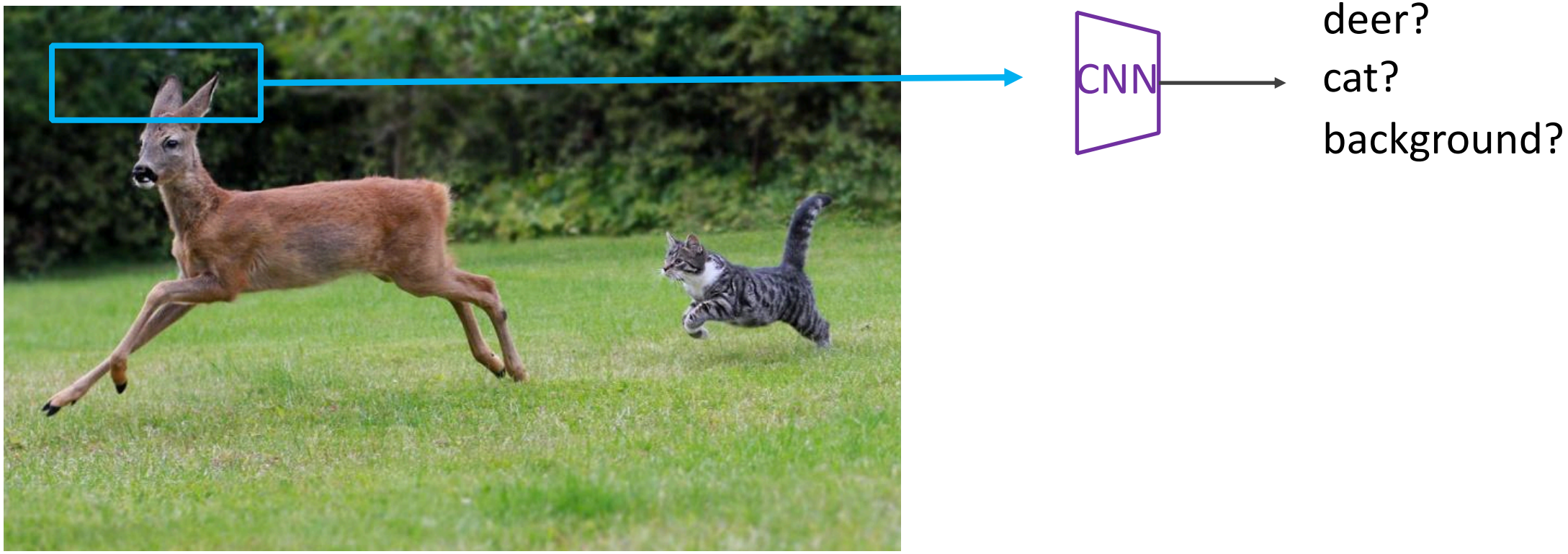
Object Detection



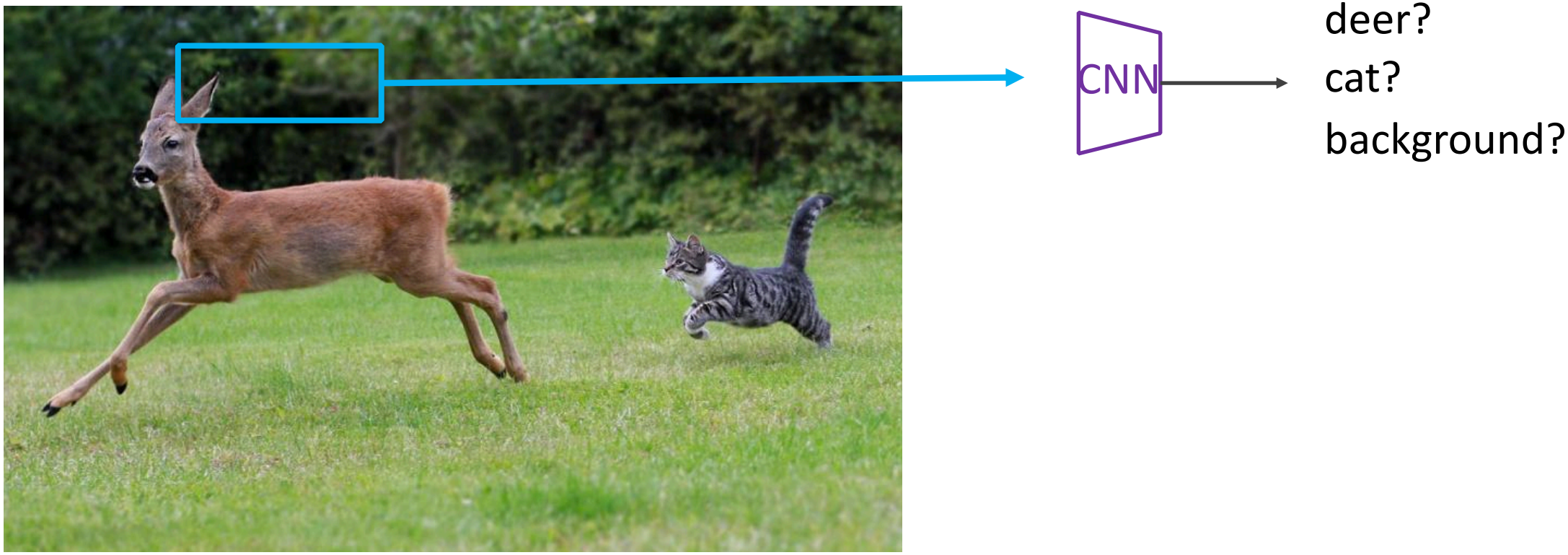
Object Detection as Classification



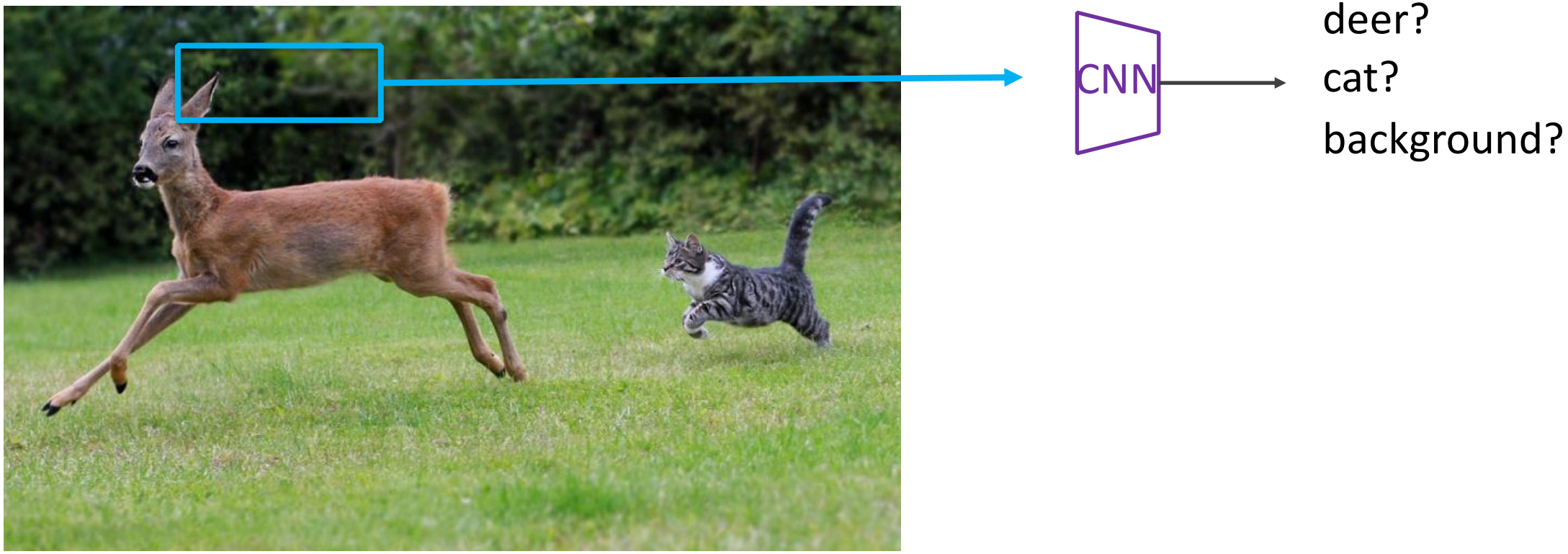
Object Detection as Classification



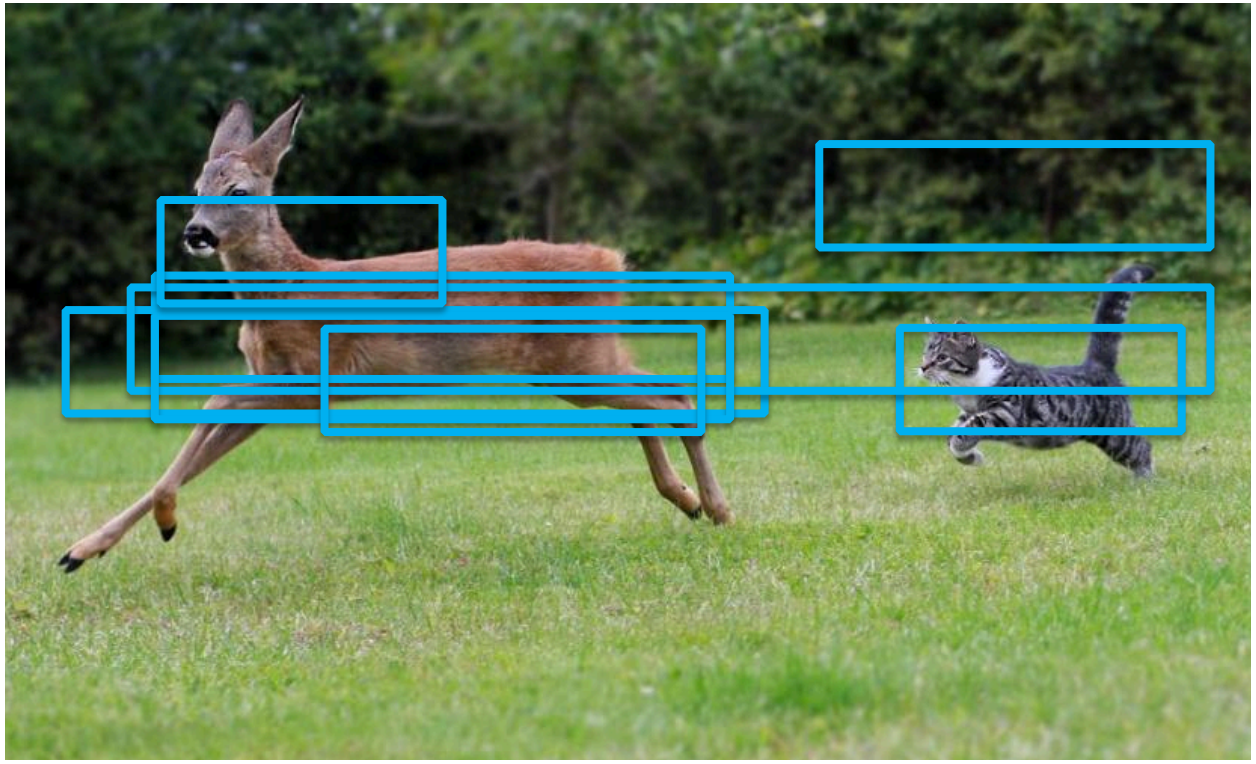
Object Detection as Classification



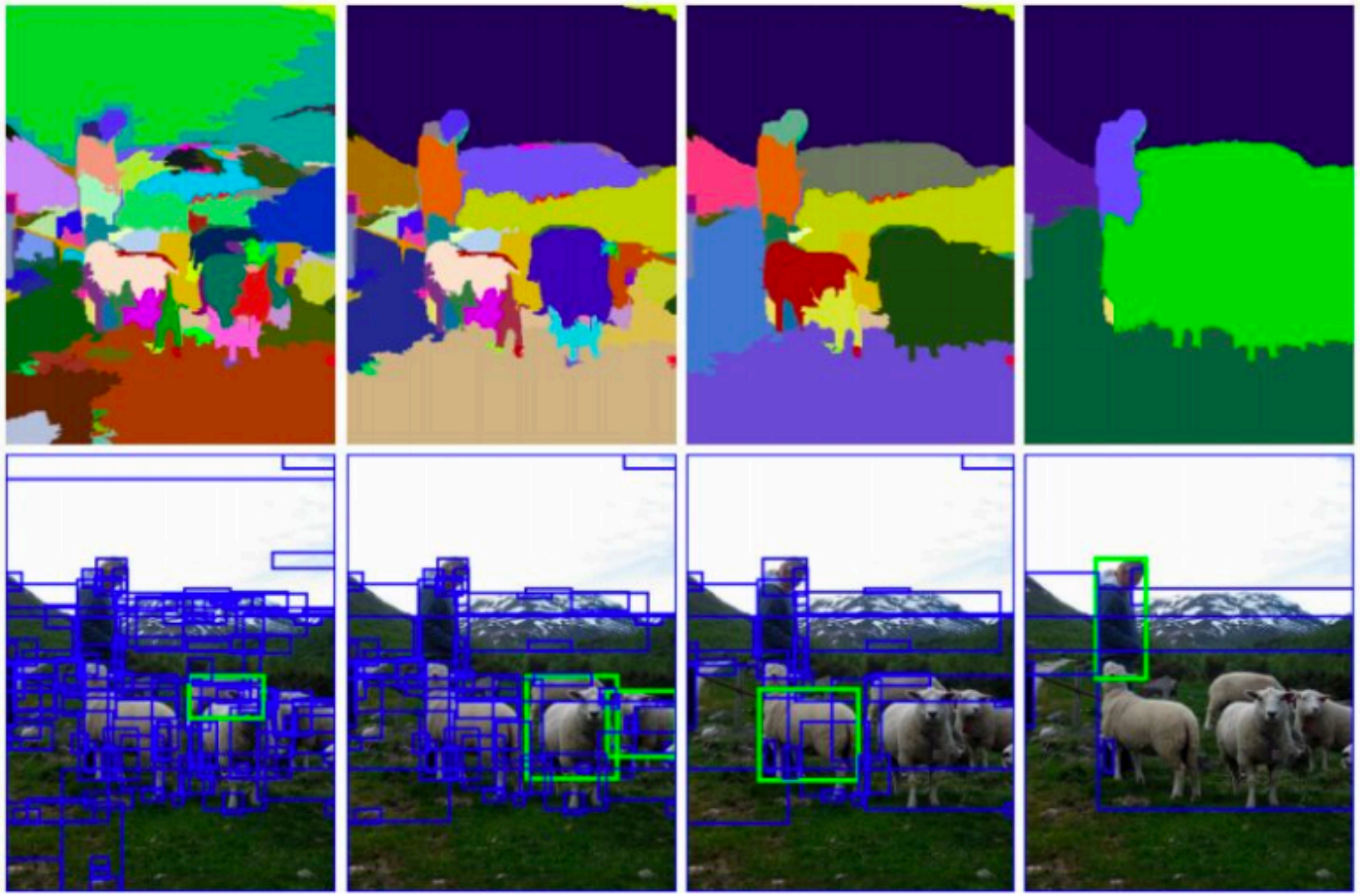
Object Detection as Classification with Sliding Window



Object Detection as Classification with Box Proposals

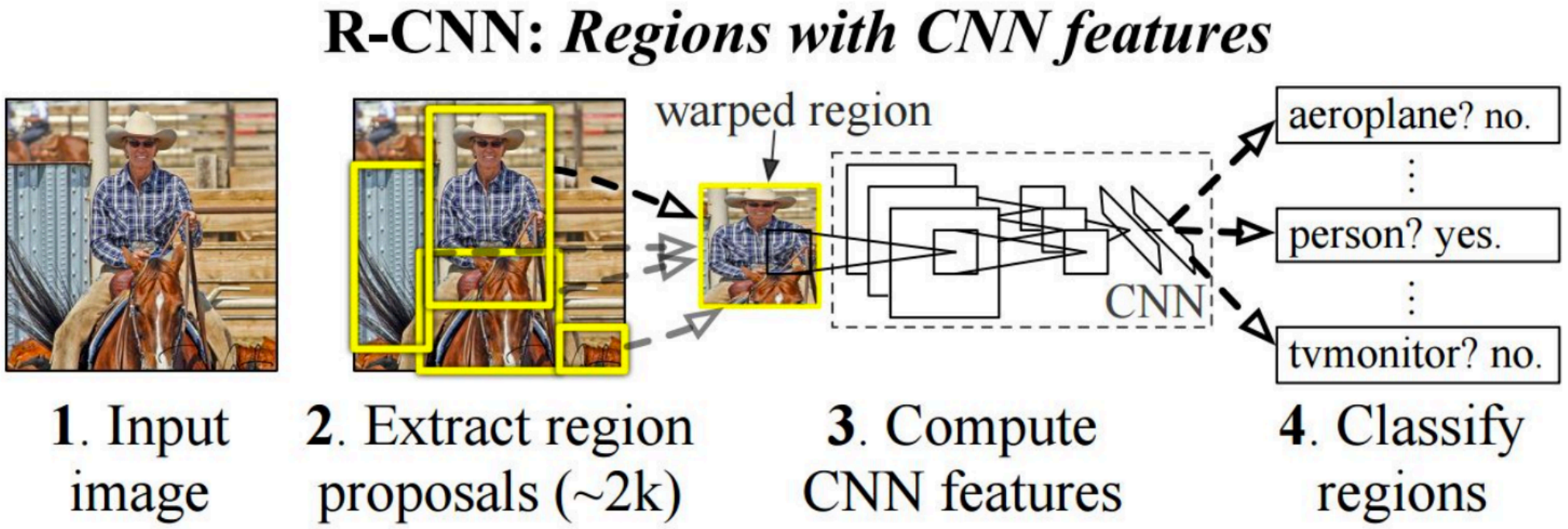


Box Proposal Method – SS: Selective Search



Segmentation As Selective Search for Object Recognition. van de Sande et al. ICCV 2011

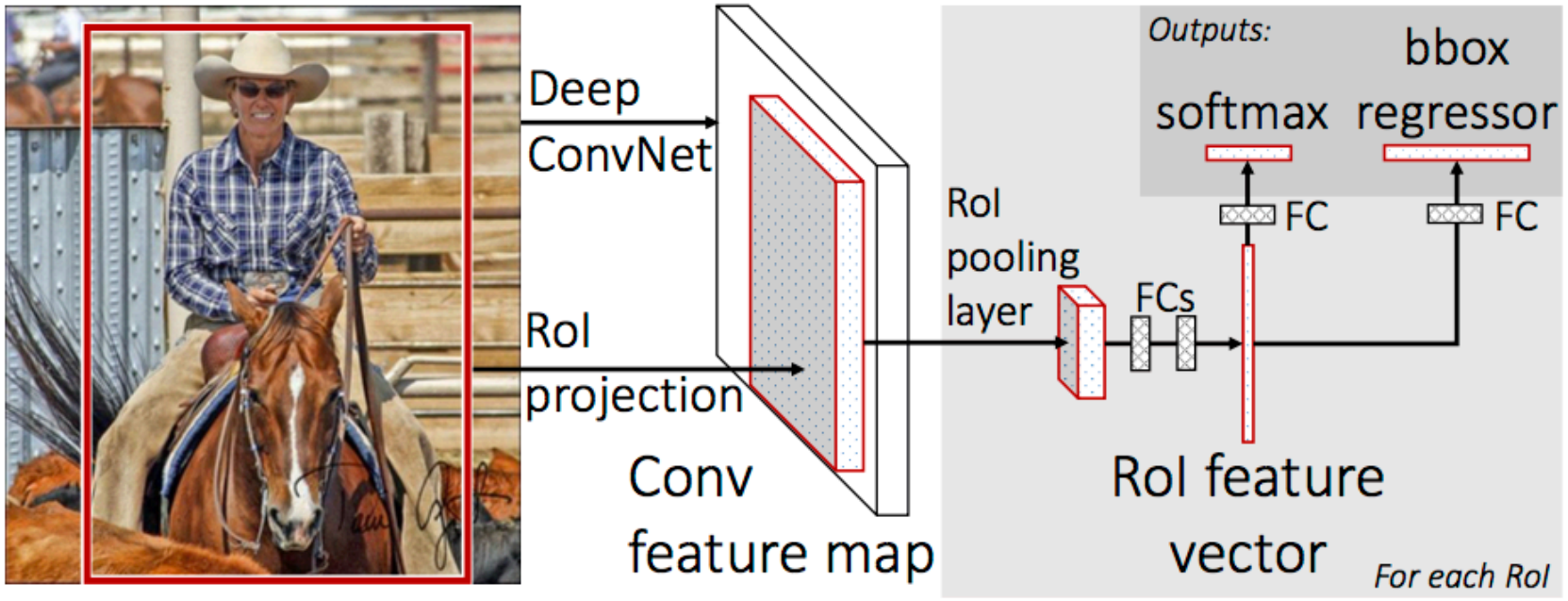
RCNN



https://people.eecs.berkeley.edu/~rbg/papers/r-cnn-cvpr.pdf

Rich feature hierarchies for accurate object detection and semantic segmentation. Girshick et al. CVPR 2014.

Fast-RCNN



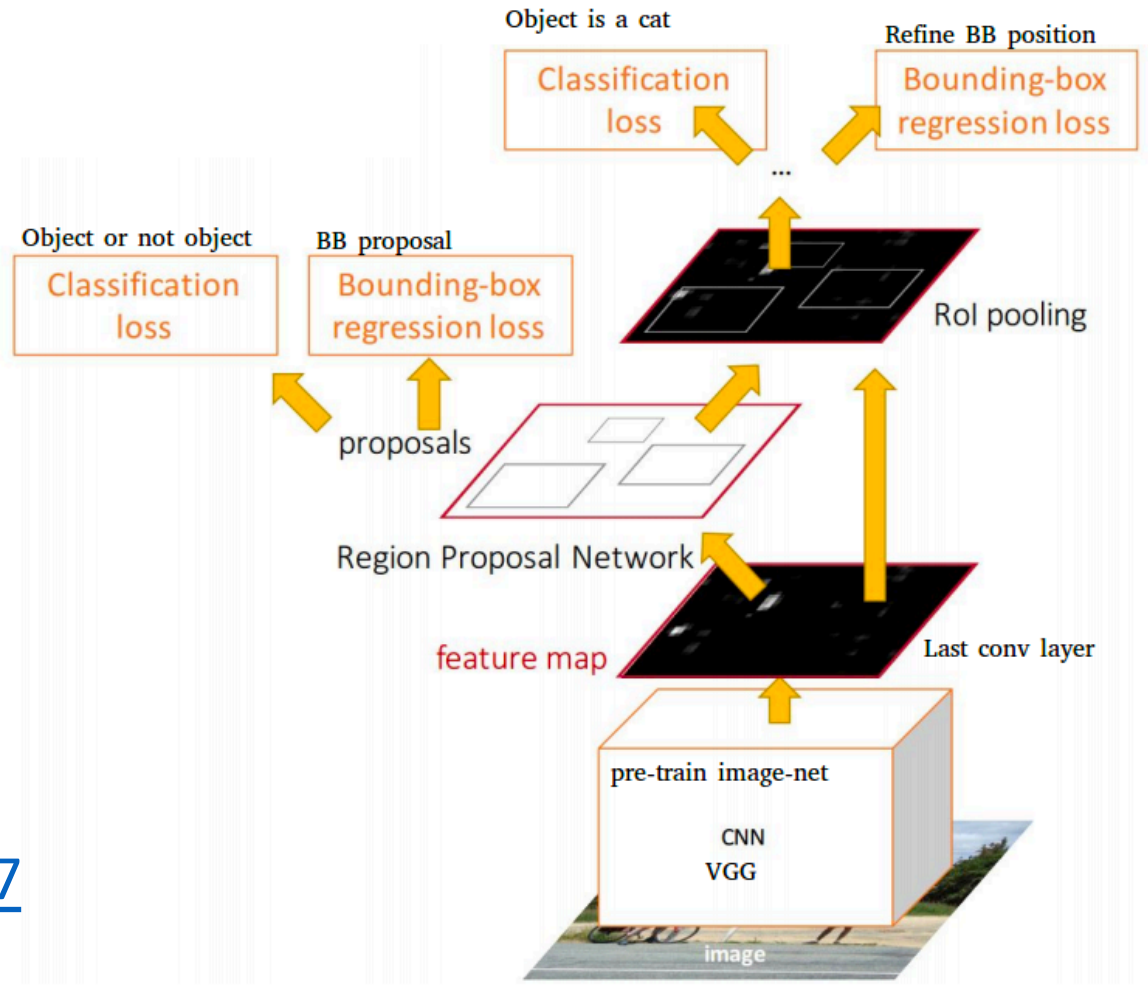
Idea: No need to recompute features for every box independently,
Regress refined bounding box coordinates.

https://arxiv.org/abs/1504.08083

Fast R-CNN. Girshick. ICCV 2015.

https://github.com/sunshineatnoon/Paper-Collection/blob/master/Fast-RCNN.md

Faster-RCNN



Idea: Integrate the Bounding Box Proposals as part of the CNN predictions

https://arxiv.org/abs/1506.01497

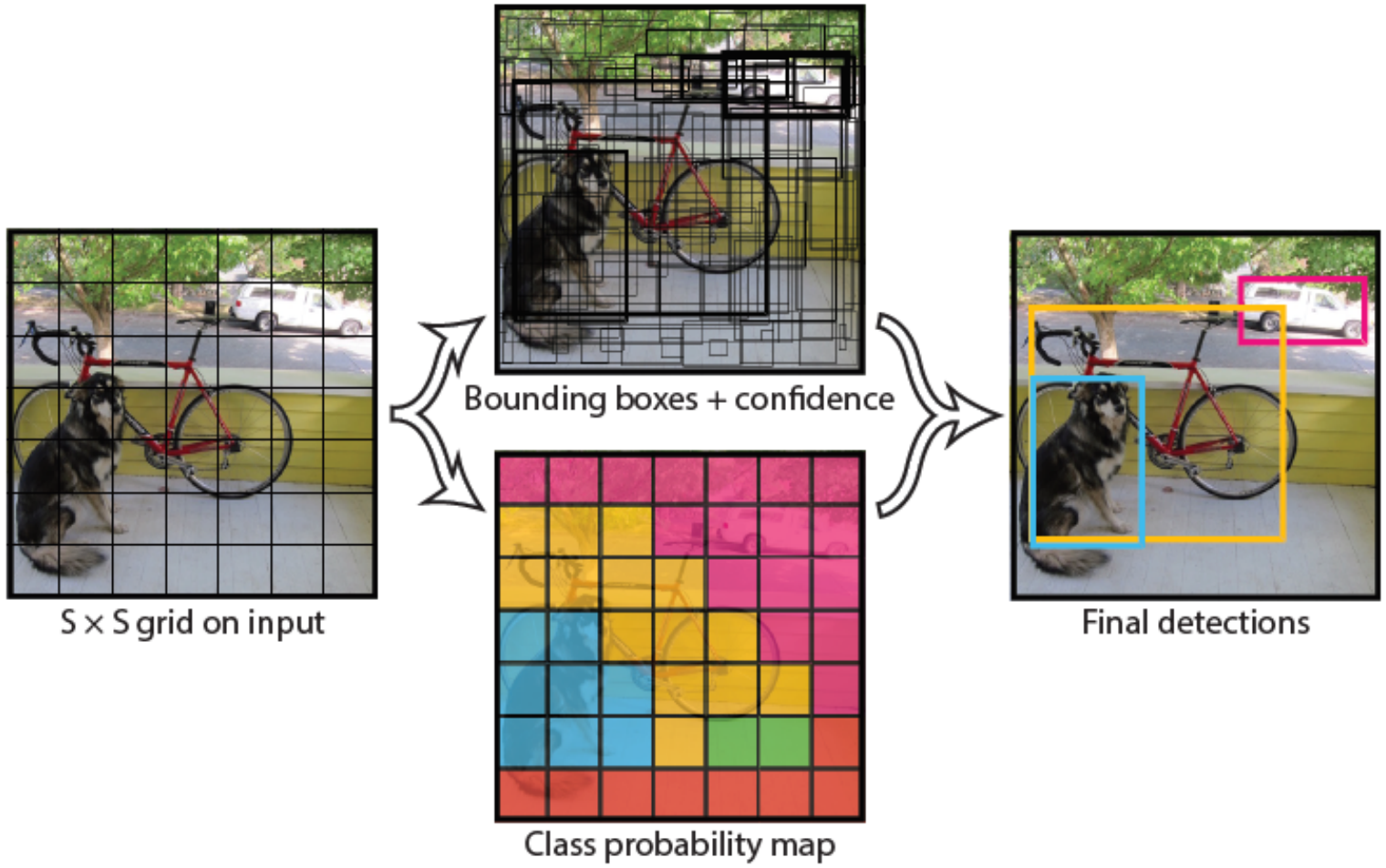
Ren et al. NIPS 2015.

Single-shot Object Detectors

- No two-steps of box proposals + Classification
- Anchor Points for predicting boxes

YOLO- You Only Look Once

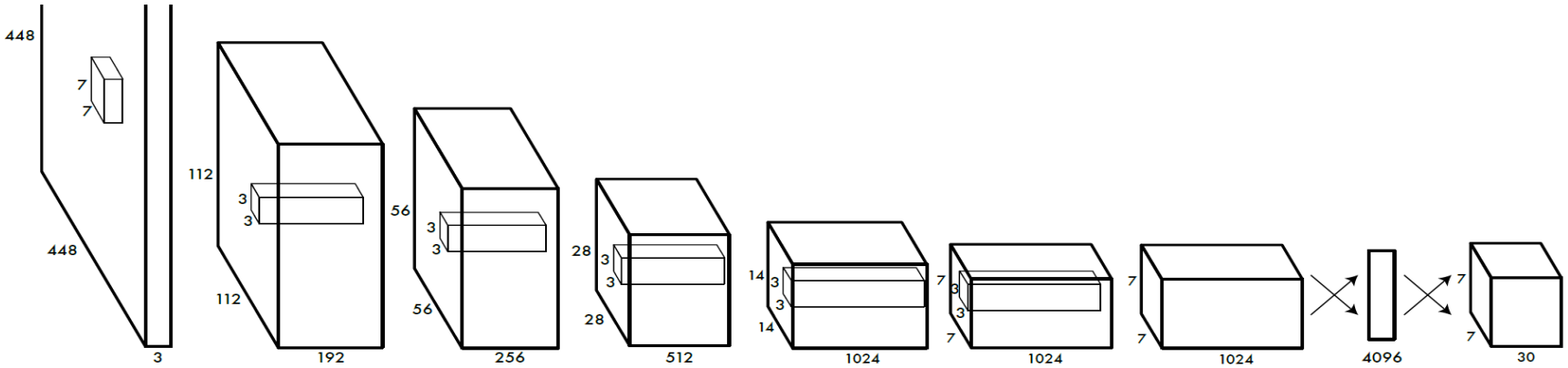
Idea: No bounding box proposals. Predict a class and a box for every location in a grid.



https://arxiv.org/abs/1506.02640

Redmon et al. CVPR 2016.

YOLO- You Only Look Once

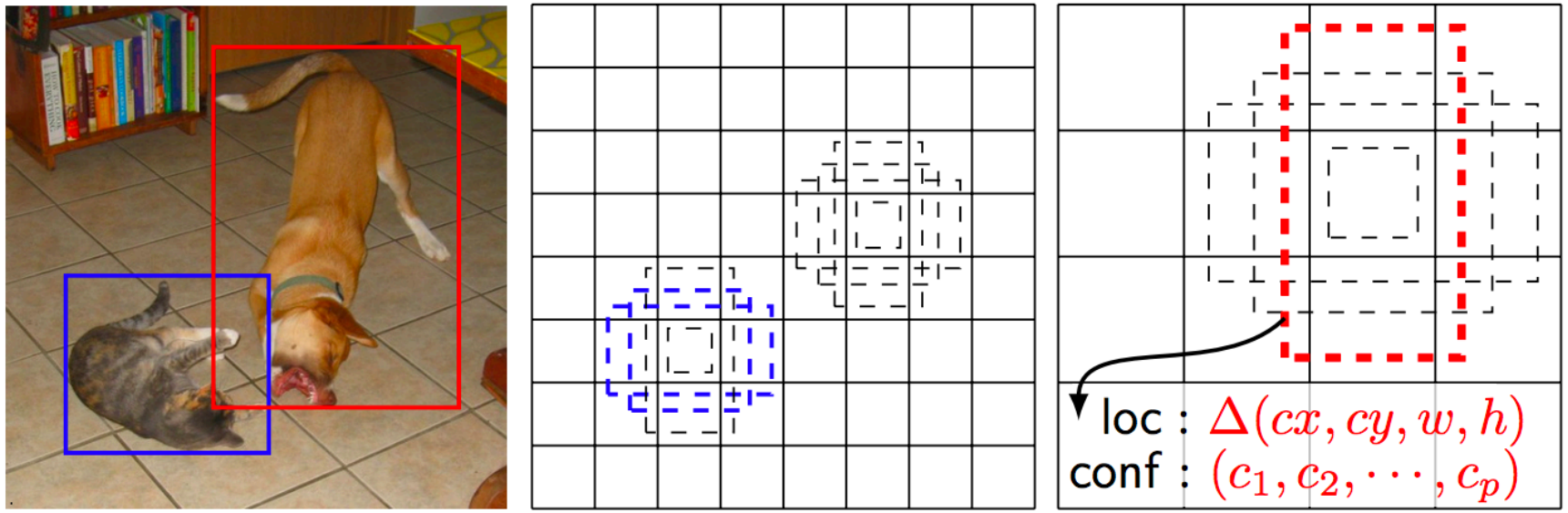


Divide the image into 7x7 cells.
Each cell trains a detector.
The detector needs to predict the object's class distributions.
The detector has 2 bounding-box predictors to predict bounding-boxes and confidence scores.

https://arxiv.org/abs/1506.02640

Redmon et al. CVPR 2016.

SSD: Single Shot Detector

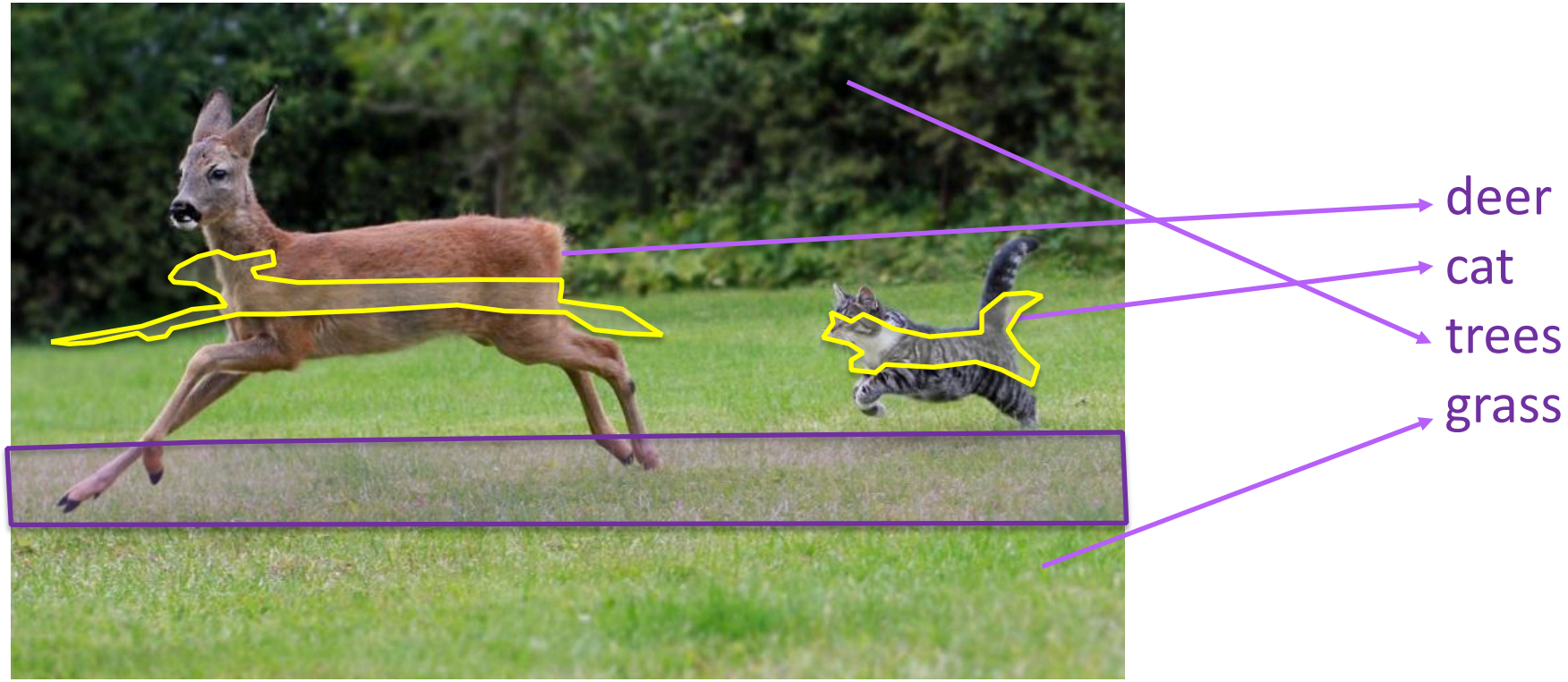


(a) Image with GT boxes (b) 8×8 feature map (c) 4×4 feature map

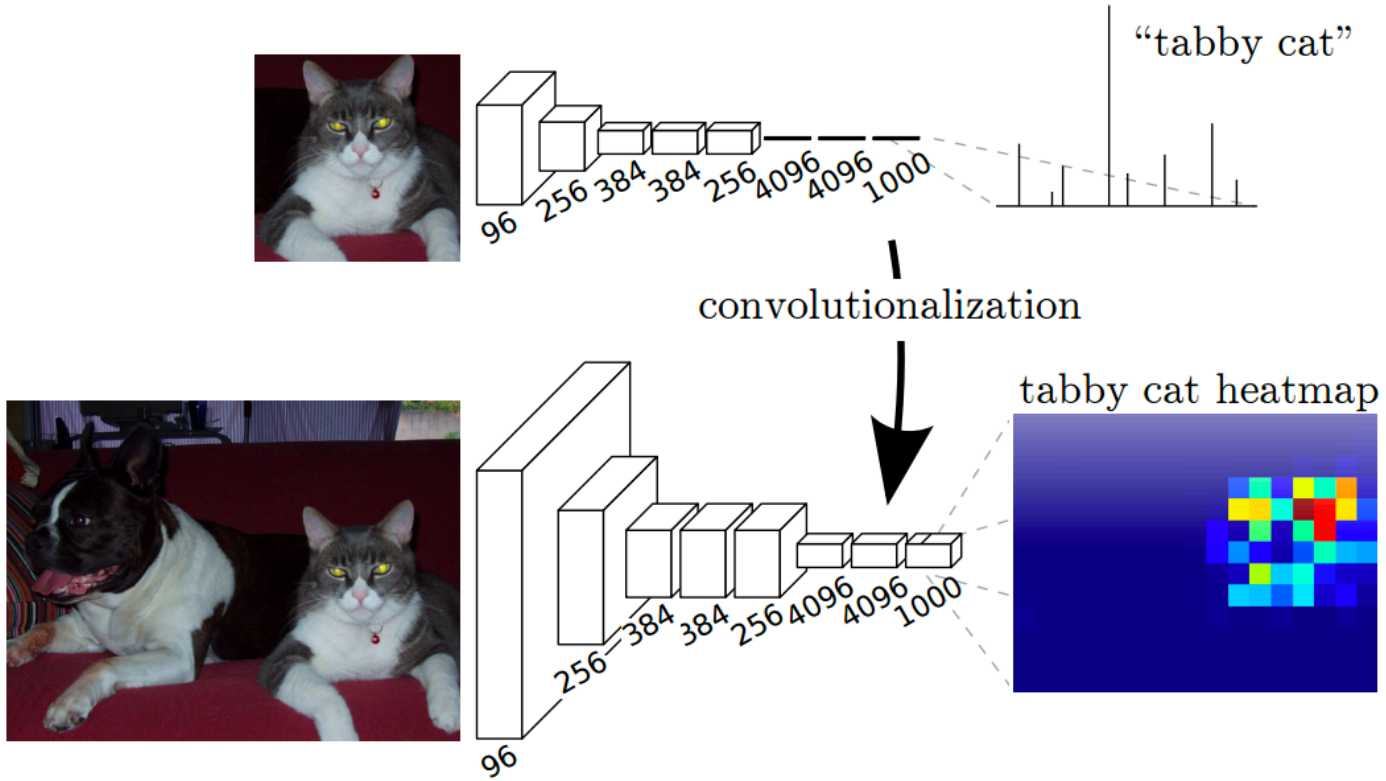
Idea: Similar to YOLO, but denser grid map, multiscale grid maps. + Data augmentation + Hard negative mining + Other design choices i n the network.

Liu et al. ECCV 2016.

Semantic Segmentation / Image Parsing



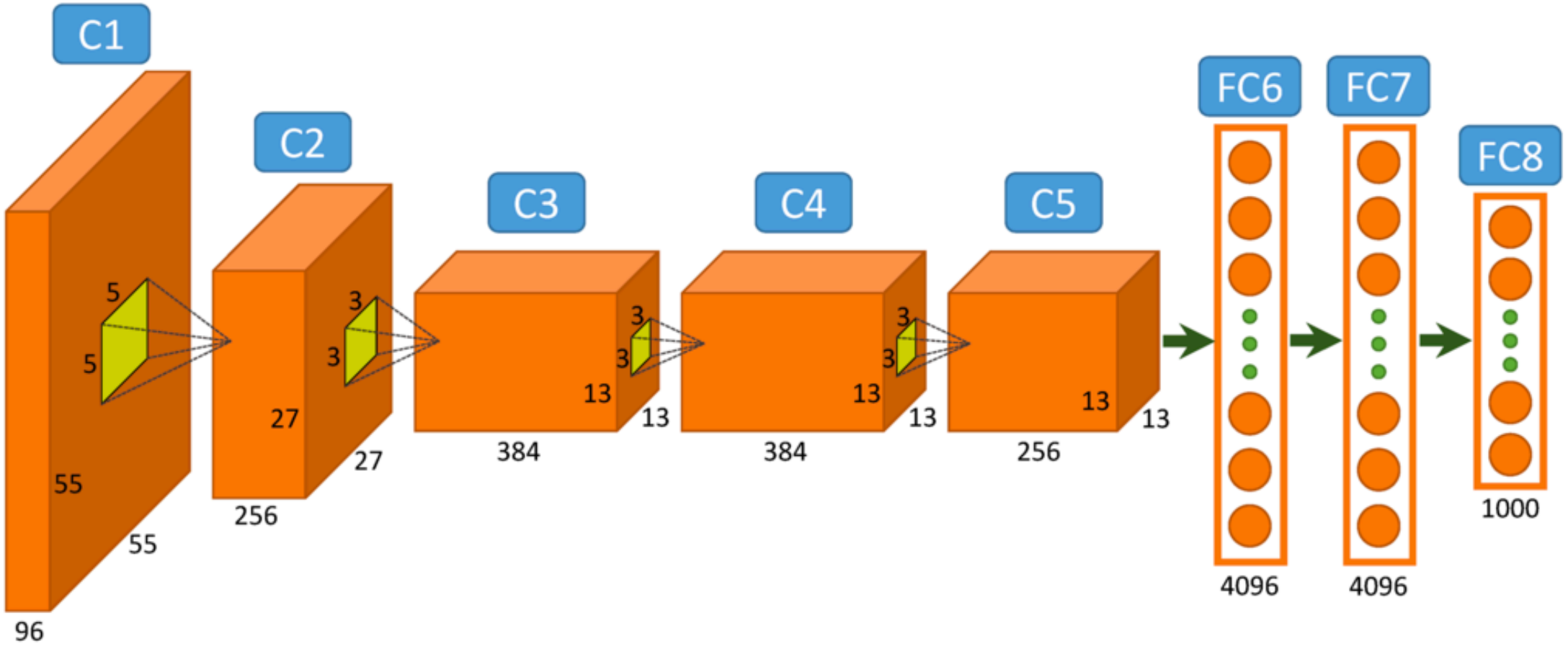
Idea 1: Convolutionalization



However resolution of the segmentation map is low.

https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf

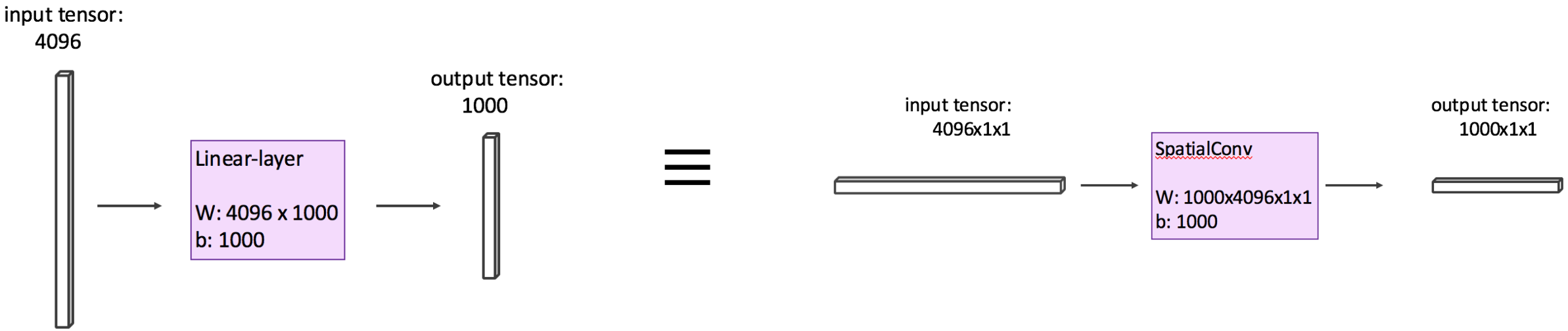
Alexnet



https://www.saagie.com/fr/blog/object-detection-part1

Idea 1: Convolutionalization

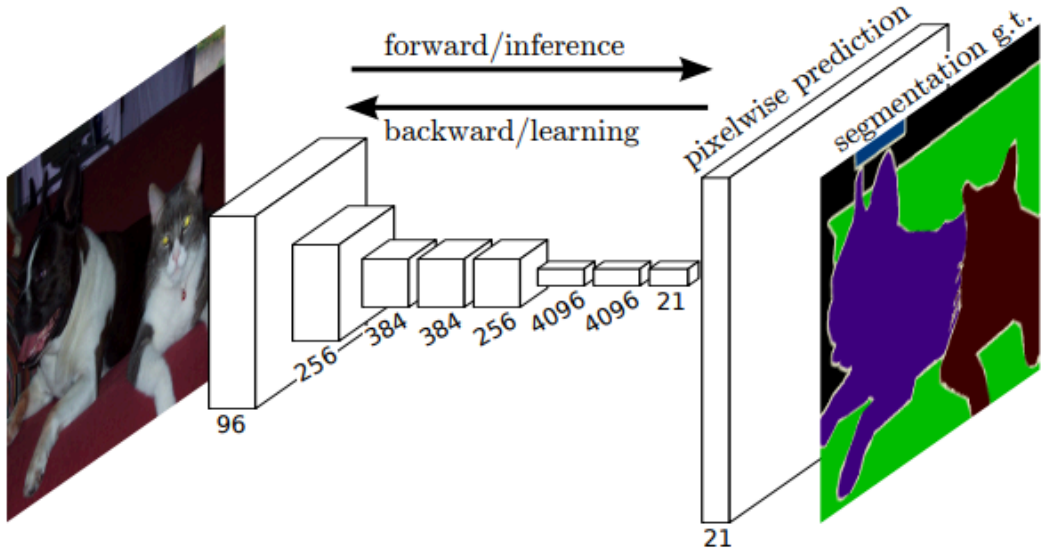
```
nn.Linear(n_inputs, n_outputs) == nn.SpatialConvolution(n_inputs, n_outputs, 1, 1, 1, 1)
```



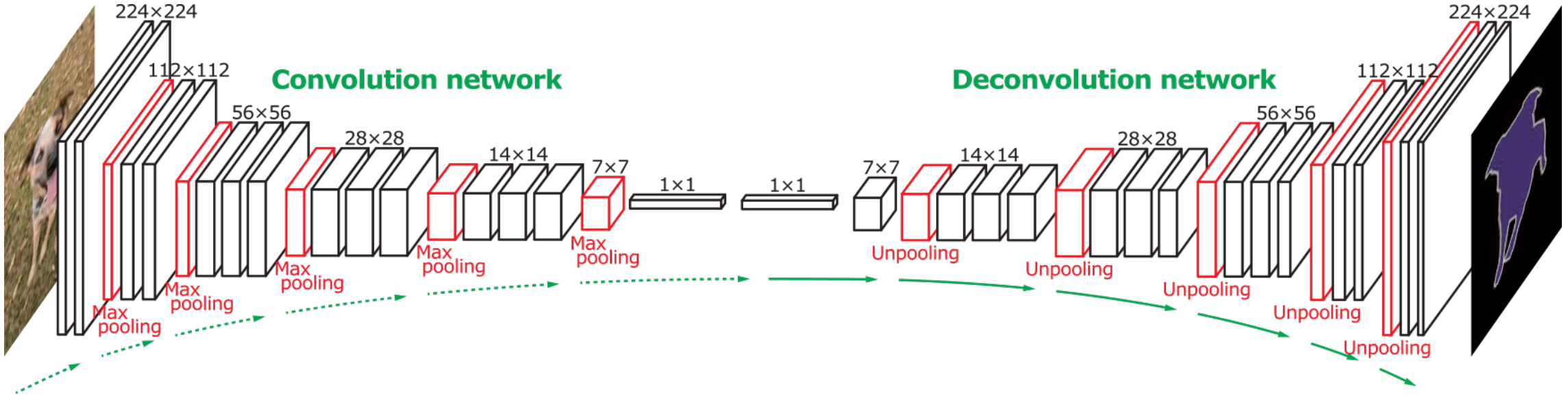
Fully Convolutional Networks (CVPR 2015)

Fully Convolutional Networks for Semantic Segmentation

Jonathan Long* Evan Shelhamer* Trevor Darrell
UC Berkeley
{jonlong,shelhamer,trevor}@cs.berkeley.edu



Idea 2: Up-sampling Convolutions or ”Deconvolutions”

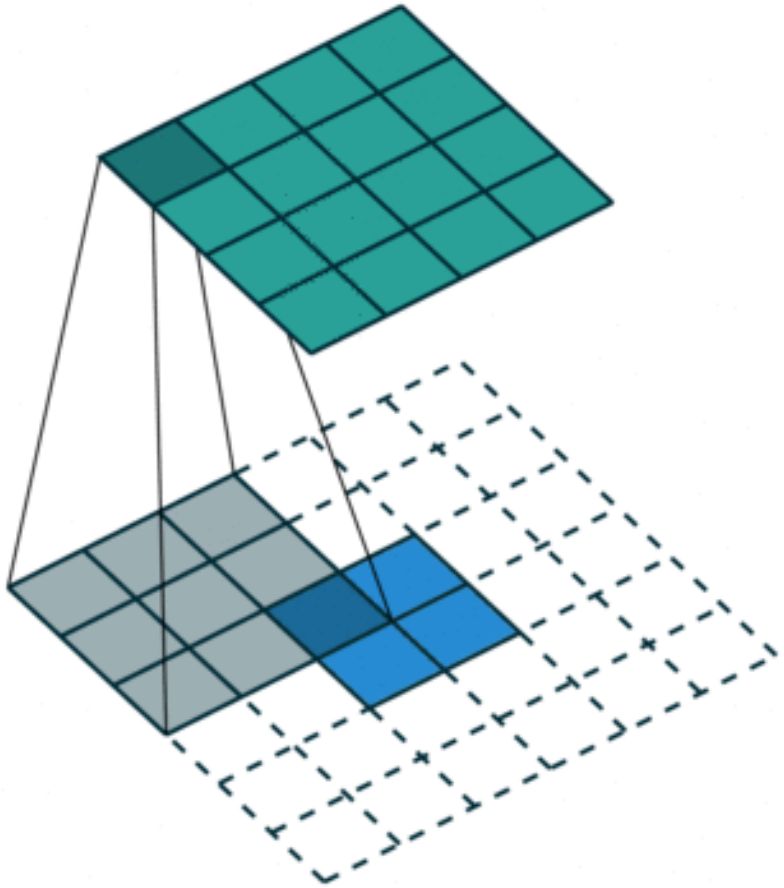


Learning Deconvolution Network for Semantic Segmentation

Hyeonwoo Noh Seunghoon Hong Bohyung Han
Department of Computer Science and Engineering, POSTECH, Korea
{hyeonwoonoh_,maga33,bhhan}@postech.ac.kr

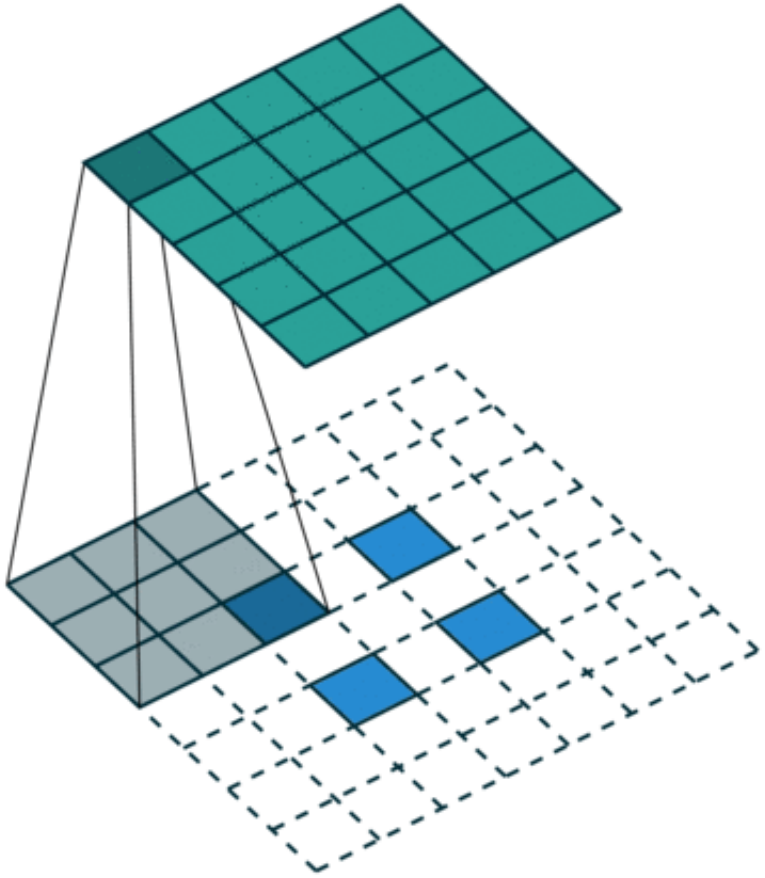
http://cvlab.postech.ac.kr/research/deconvnet/

Idea 2: Up-sampling Convolutions or ”Deconvolutions”



https://github.com/vdumoulin/conv_arithmetic

Idea 2: Up-sampling Convolutions or ”Deconvolutions”



https://github.com/vdumoulin/conv_arithmetic

Idea 2: Up-sampling Convolutions or ”Deconvolutions”

Deconvolutional Layers

Upconvolutional Layers

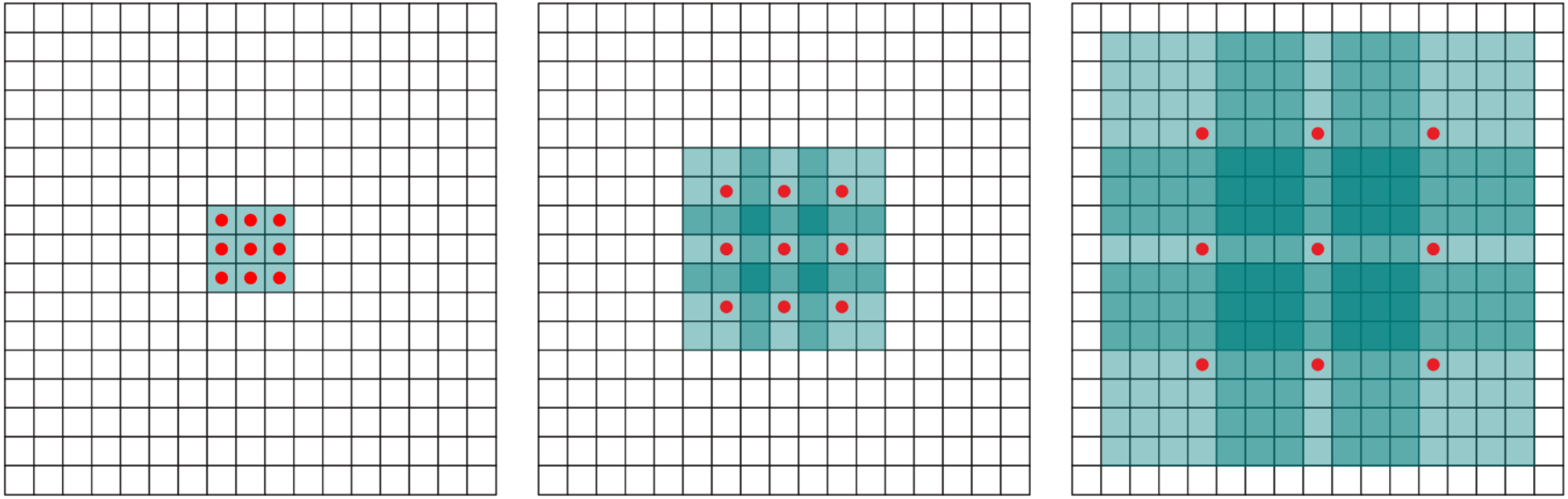
Backwards Strided
Convolutional Layers

Fractionally Strided
Convolutional Layers

Transposed
Convolutional Layers

Spatial Full
Convolutional Layers

Idea 3: Dilated Convolutions



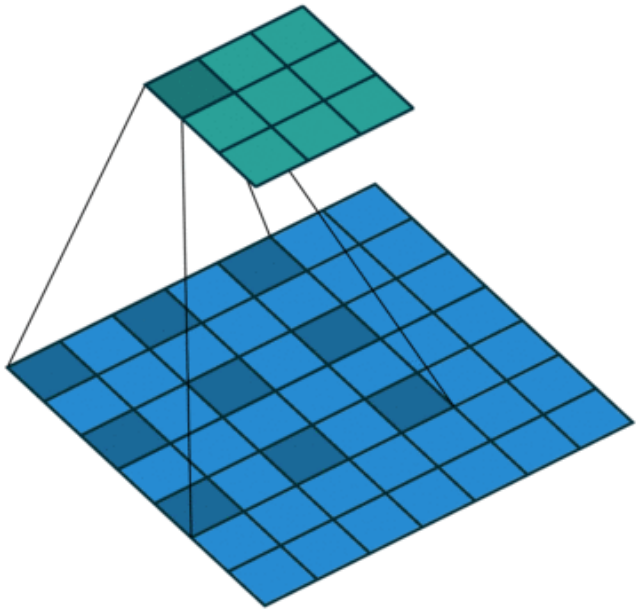
MULTI-SCALE CONTEXT AGGREGATION BY DILATED CONVOLUTIONS

Fisher Yu
Princeton University

Vladlen Koltun
Intel Labs

ICLR 2016

Idea 3: Dilated Convolutions



MULTI-SCALE CONTEXT AGGREGATION BY DILATED CONVOLUTIONS

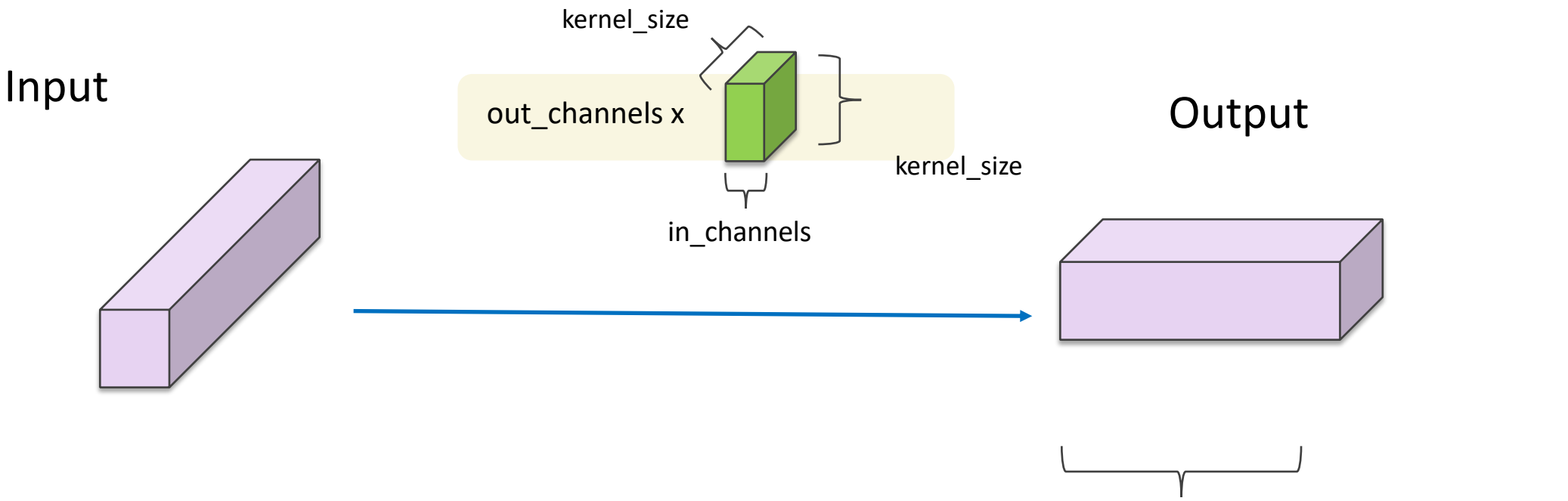
Fisher Yu
Princeton University

Vladlen Koltun
Intel Labs

ICLR 2016

Convolutional Layer in pytorch

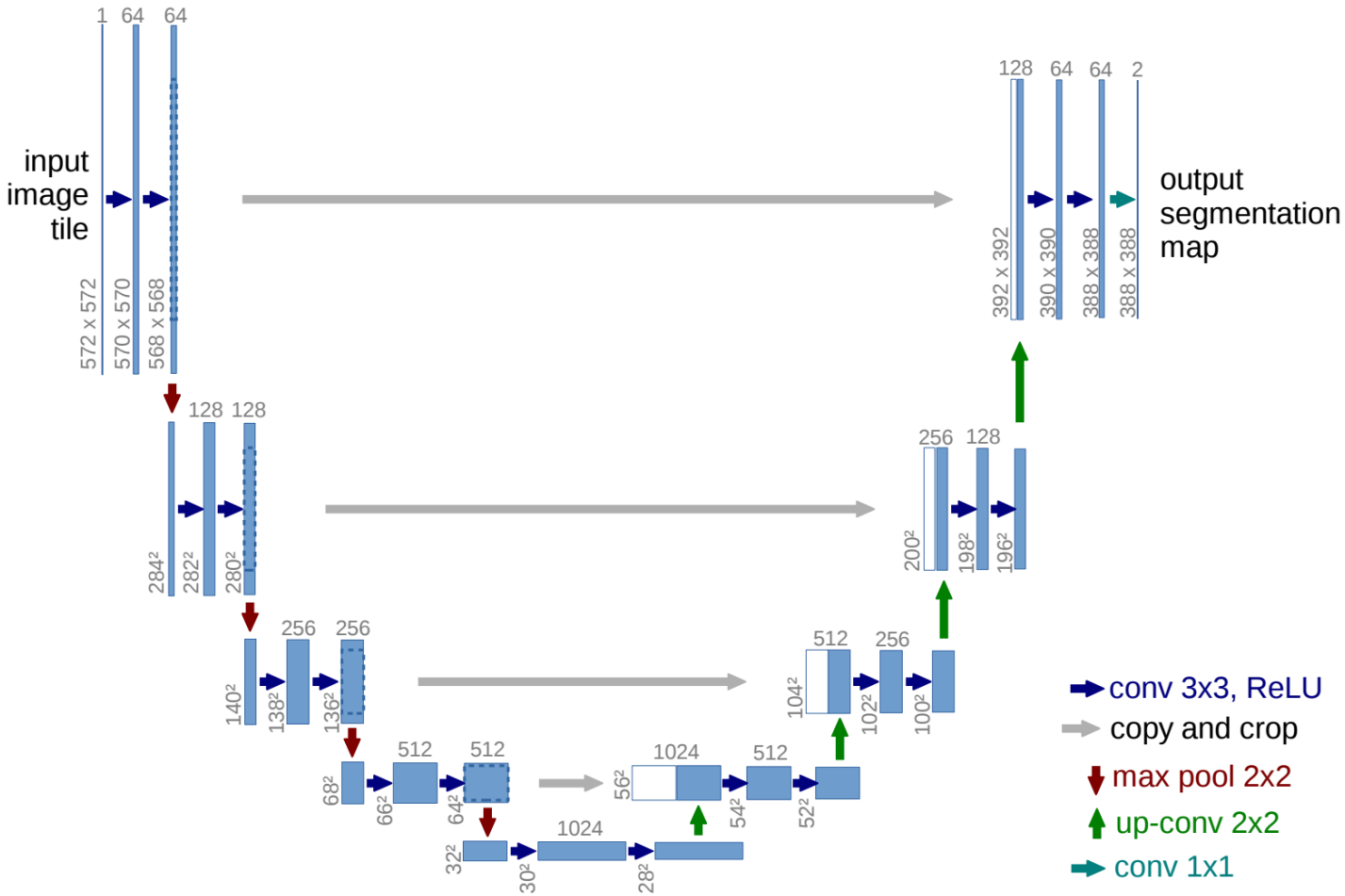
```
class torch.nn.Conv2d(in_channels, out_channels, kernel_size, stride=1, padding=0, dilation=1,
groups=1, bias=True)  [source]
```



U-Net: Convolutional Networks for Biomedical Image Segmentation

Olaf Ronneberger, Philipp Fischer, and Thomas Brox

Computer Science Department and BIOSS Centre for Biological Signalling Studies,
University of Freiburg, Germany



https://arxiv.org/abs/1505.04597

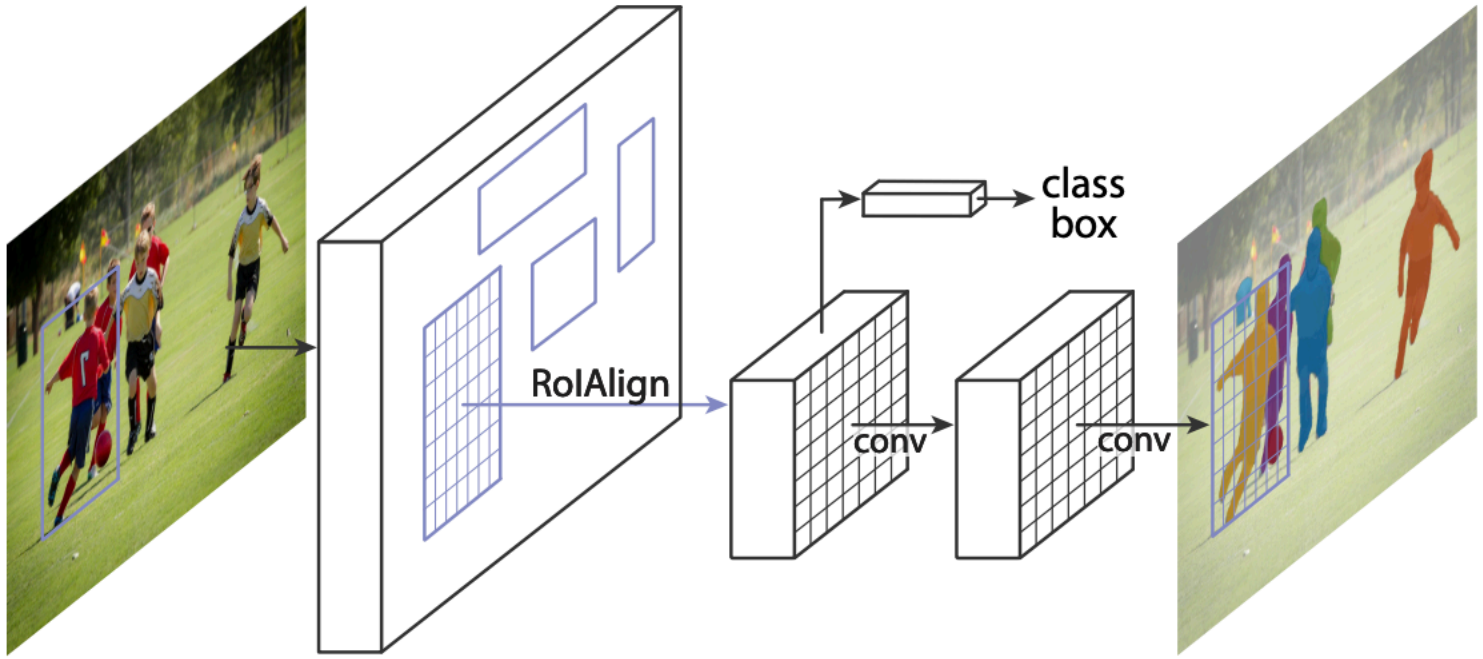
https://github.com/milesial/Pytorch-UNet

https://github.com/usuyama/pytorch-unet

Mask R-CNN

Kaiming He Georgia Gkioxari Piotr Dollár Ross Girshick

Facebook AI Research (FAIR)



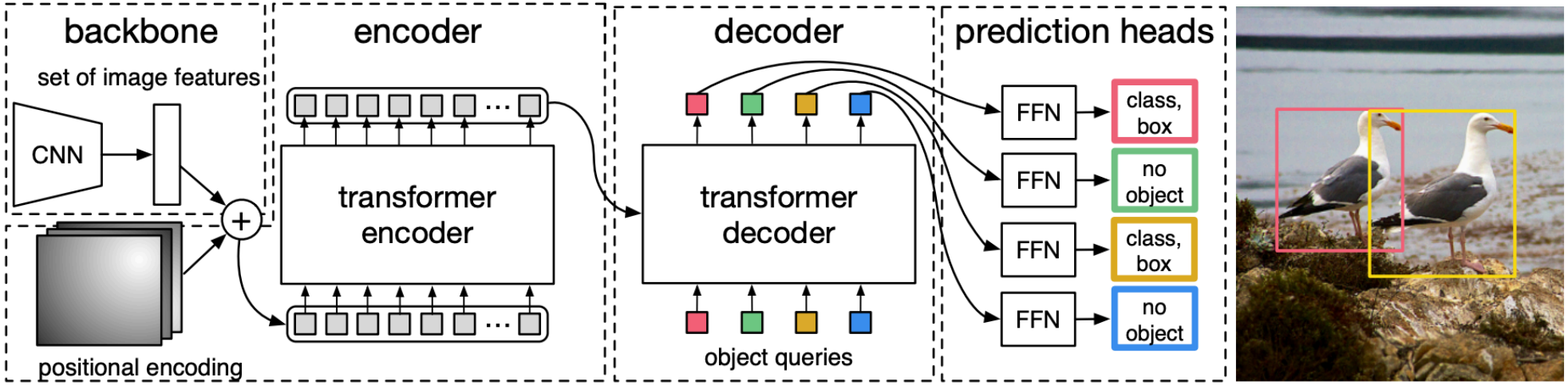
https://github.com/facebookresearch/detectron2

https://arxiv.org/abs/1703.06870

End-to-End Object Detection with Transformers

Nicolas Carion*, Francisco Massa*, Gabriel Synnaeve, Nicolas Usunier,
Alexander Kirillov, and Sergey Zagoruyko

Facebook AI



https://github.com/facebookresearch/detr

https://arxiv.org/abs/2005.12872

Questions?