Advanced NN Architectures: CNNs

Roberto Basili, Danilo Croce

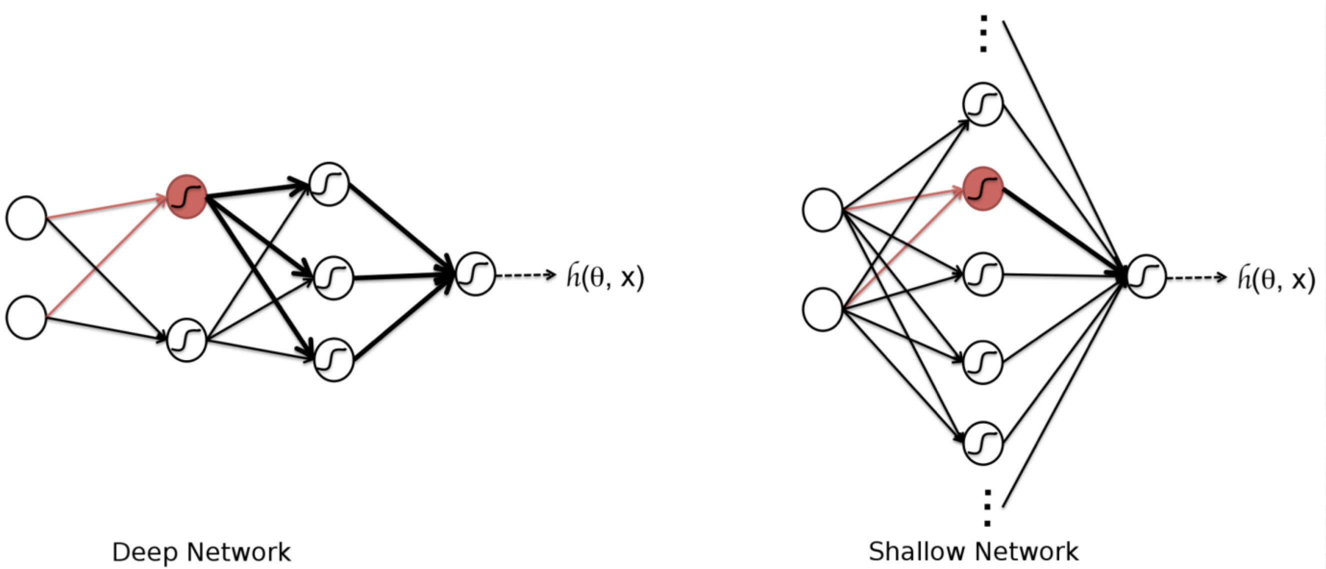
Machine Learning, Web Mining & Retrieval 2021/2022

Outline

- Architectures and tasks
- Convolutional Neural Networks
  - Filters and Convolutions
  - Pooling
- Imagenet
- Applications of NNs:
  - Image processing: classification, Object Recognition
  - Text Classification:
    - Convolutional NNs over texts
    - Sentiment analysis
    - The Movie Review Dataset

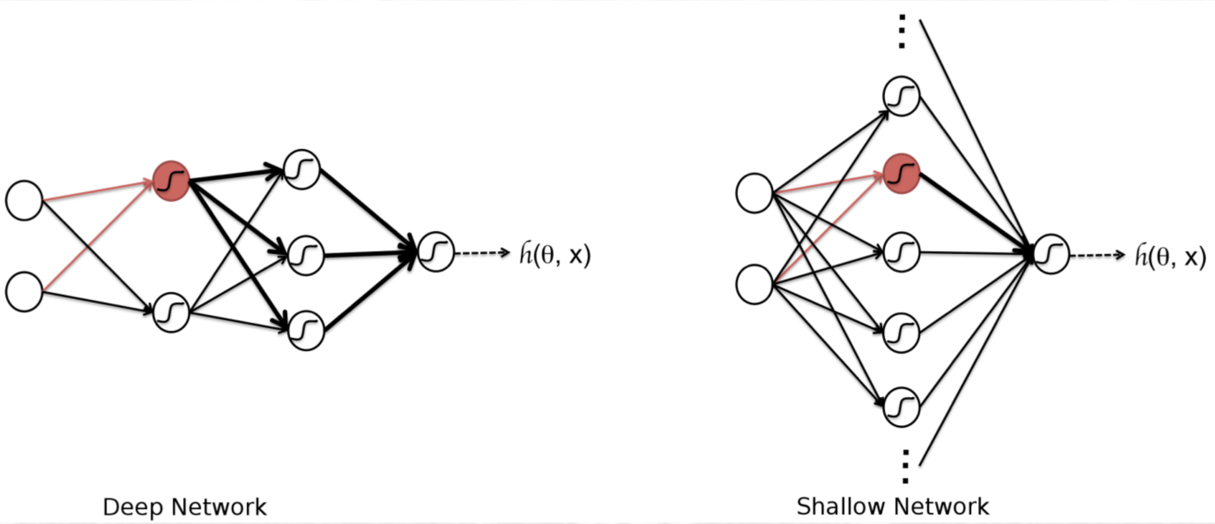
Deep vs Shallow Networks

- Deep networks should be preferred to Shallow ones
  - when problems are non-linear;
  - is has been observed that a shallow network needs about 10x number of neurons for reaching the expressivity of a deep one



Deep vs Shallow Networks: Intuition

- Think of a neuron as a program routine
  - in Deep Networks a neuron computation is re-used many times in the computation
  - in a Shallow Network it is used only once
- Using a shallow network is similar to writing a program without the ability of calling subroutines



Deep Networks vs. Kernel

- A kernel machine can be thought of as a shallow network having a huge hidden layer
  - this hidden layer is never computed thanks to the kernel trick
- Kernel methods however are expensive
  - they rely on a set of examples, support vectors
  - for large dataset and complex problems this set can be large as well
- Neural networks computation
  - is independent on the dataset,
  - but only on the number of connections that have been chosen

NN architectures

- Multilayer perceptron (Rumelhart MCClelland,1980)
- Self-Organizing maps (Kohonen, 1990)
- Boltzman Machines (Hinton, 1998)
- Convolutional Neural Networks (Neocogitron, Fukushima (1980))
- Recurrent Neural Networks (Jordan, 1986), (Elman, 1990)
  - Bidirectional RNNs (Schuster and Paliwal, 1997)
  - BP Through-Time (Robinson & Fallside, 1987)
  - Long Short Time Memories LSTMS, (Hochreiter & Schmidhuber, 1997)
  - Attention mechanisms (firstly discussed by (Larochelle & Hinton, 2010; Denil et al., 2012)).
- Autoencoders (Bengio et al., 2007), Encoder-Decoders (Cho et al., 2015)

Recent successes in Deep Learning

- Convolution Neural Networks
  - Images related tasks
- Recurrent Neural Networks
  - Language models
  - Speech to Text
  - Machine Translation, Conversation Models
- Attentional Networks
  - Attention mechanisms toamplify dependencies across network components
- Trasformers:
  - Encoding-decoding networks for powerful pretraining
  - Avoid the forgetting problems typical of recurrent acrhitectures
- Advanced architectures
  - Image to Captions

Architectures

Architetctures & tasks:

multitask learning

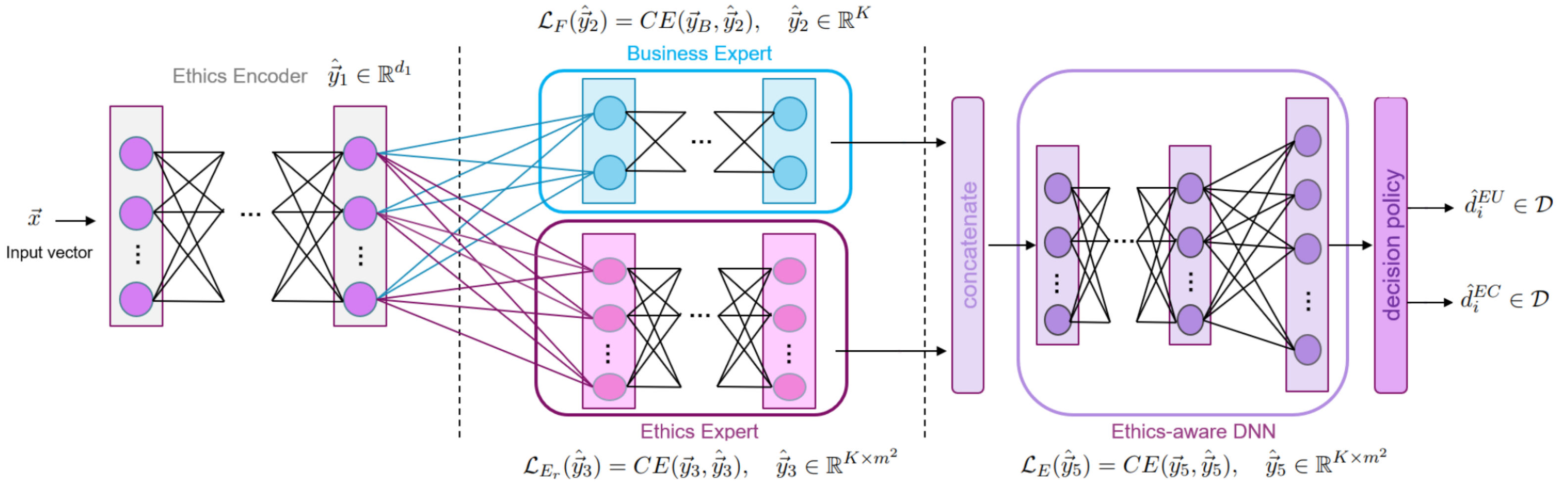

$\mathcal{L}_F(\hat{\vec{y}}_2) = CE(\vec{y}_B, \hat{\vec{y}}_2), \quad \hat{\vec{y}}_2 \in \mathbb{R}^K$
Business Expert
Ethics Encoder $\hat{\vec{y}}_1 \in \mathbb{R}^{d_1}$
$\vec{x}$
Input vector
concatenate
decision policy
$\hat{d}_i^{EU} \in \mathcal{D}$
$\hat{d}_i^{EC} \in \mathcal{D}$
Ethics Expert
Ethics-aware DNN
$\mathcal{L}_{E_r}(\hat{\vec{y}}_3) = CE(\vec{y}_3, \hat{\vec{y}}_3), \quad \hat{\vec{y}}_3 \in \mathbb{R}^{K \times m^2}$
$\mathcal{L}_E(\hat{\vec{y}}_5) = CE(\vec{y}_5, \hat{\vec{y}}_5), \quad \hat{\vec{y}}_5 \in \mathbb{R}^{K \times m^2}$

Figure 1: The architecture of the Ethical by Design Neural Network.

AAAI-20
Thirty-Fourth AAAI Conference on Artificial Intelligence
February 7-12 2020, Hilton New York Midtown, New York, New York USA

Types of RNNs

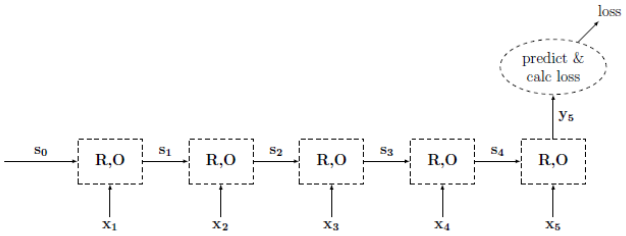


Figure 7: Acceptor RNN Training Graph.

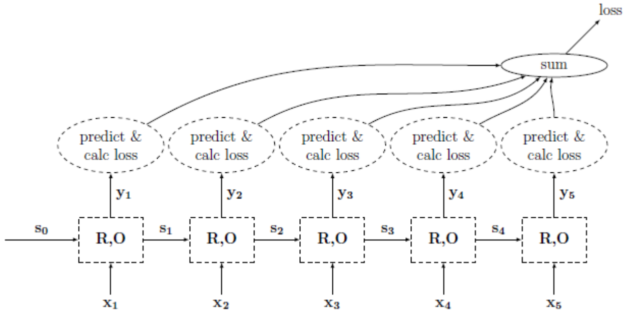


Figure 8: Transducer RNN Training Graph.

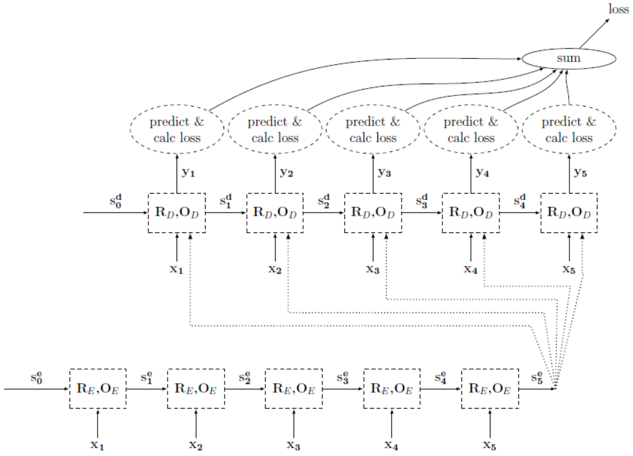


Figure 9: Encoder-Decoder RNN Training Graph.

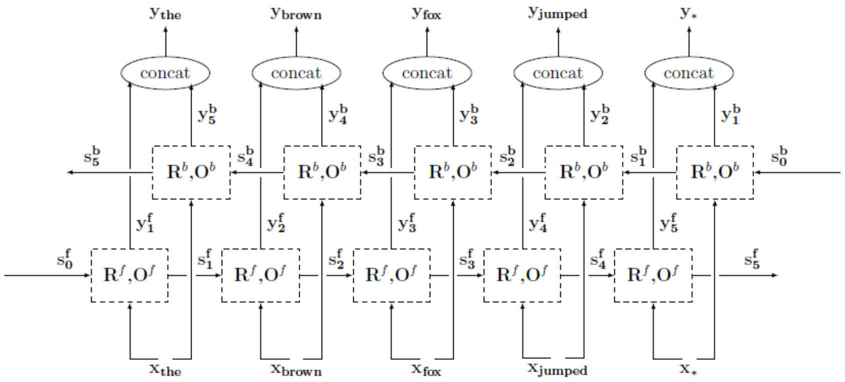


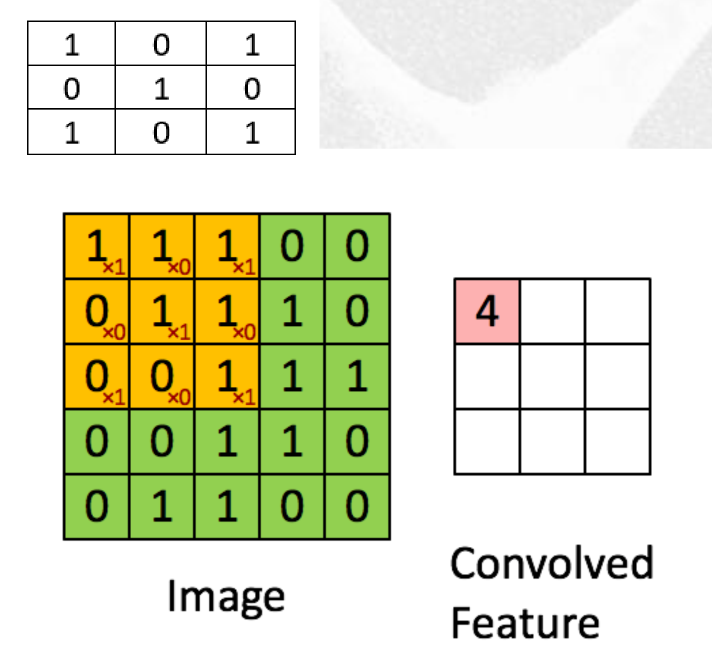
Figure 11: biRNN over the sentence "the brown fox jumped .".

Outline

- Architectures and tasks
- Convolutional Neural Networks
  - Filters and Convolutions
  - Pooling
- Imagenet
- Applications of NNs:
  - Image processing: classification, Object Recognition
  - Text Classification:
    - Convolutional NNs over texts
    - Sentiment analysis
    - The Movie Review Dataset

Convolutional Neural Networks
(Le Cun, 1998)

- Mainly used for images related tasks
  - image classification
  - face detection
  - etc…
- **Learn feature representations**
  - by ***convolving*** over the input
  - with a ***filter***, that slides over the input image
- **Compositionality** (local)
  - Each filter composes a local patch of lower-level features into a higher-level representation
- **Location Invariance**
  - the detection of specific patterns is independent of where it occurs



1	0	1
0	1	0
1	0	1

$1_{\times 1}$	$1_{\times 0}$	$1_{\times 1}$	0	0
$0_{\times 0}$	$1_{\times 1}$	$1_{\times 0}$	1	0
$0_{\times 1}$	$0_{\times 0}$	$1_{\times 1}$	1	1
0	0	1	1	0
0	1	1	0	0

Image

4		

Convolved Feature

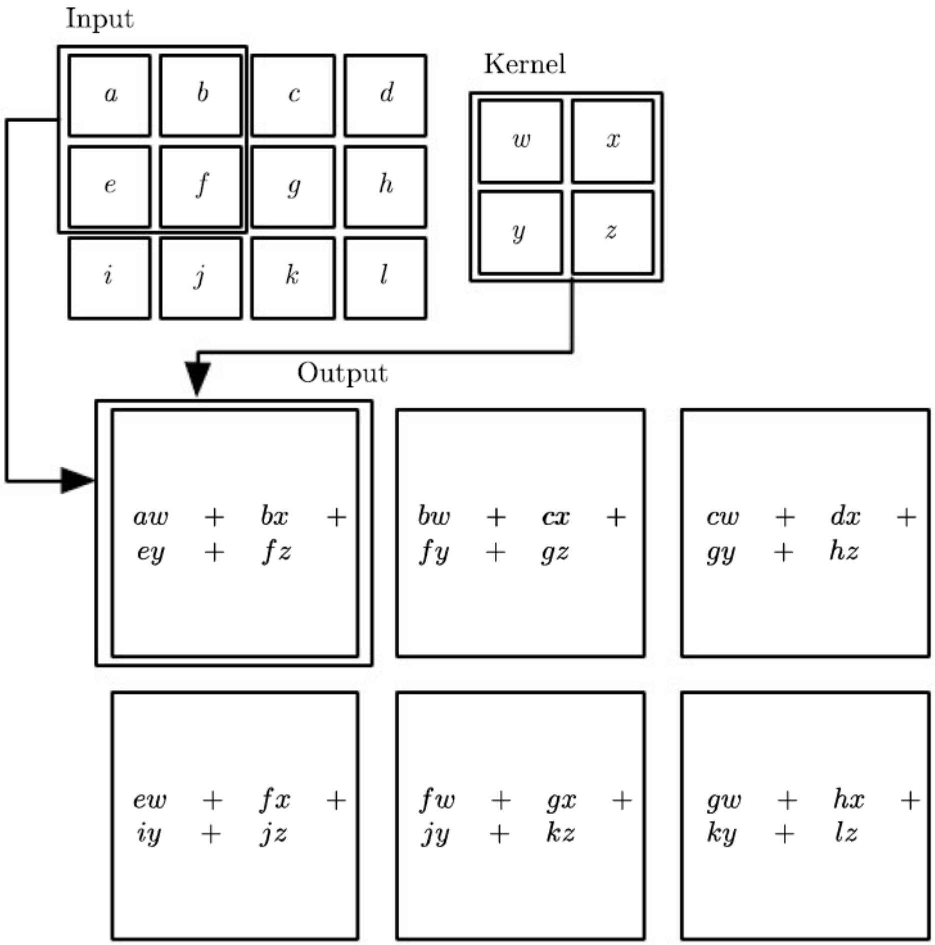
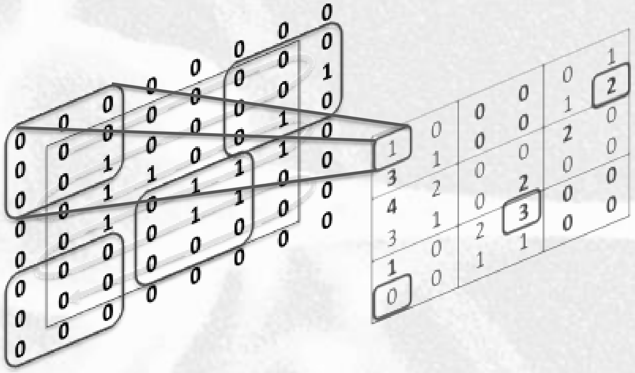


Figure 9.1: An example of 2-D convolution without kernel flipping. We restrict the output to only positions where the kernel lies entirely within the image, called "valid" convolution in some contexts. We draw boxes with arrows to indicate how the upper-left element of the output tensor is formed by applying the kernel to the corresponding upper-left region of the input tensor.

A futher example of: convolution with pooling, and decimation operations

Filter

-1	0	1
-2	0	2
-1	0	1



Max Pooling

3	3	0	0	2	2
3	3	0	0	2	2
4	4	2	2	2	2
4	4	2	2	2	2
1	1	3	3	0	0
1	1	3	3	0	0

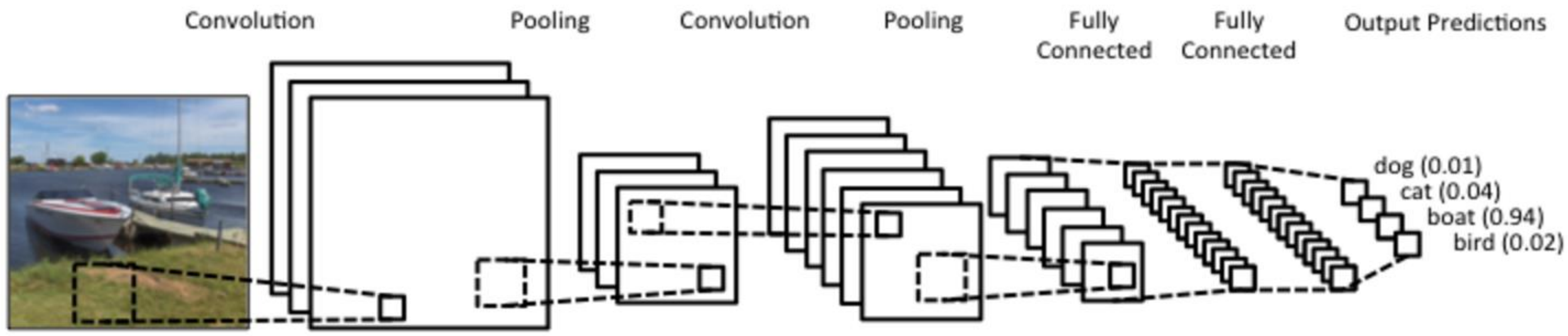
Decimation

3	0	2
4	2	2
1	3	0

- An image is convolved with a filter; curved rectangular regions in the first large matrix depict a random set of image locations
- Maximum values within small 2×2 regions are indicated in bold in the central matrix
- The results are pooled, using max-pooling then decimated by a factor of two, to yield the final matrix

Convolutional Neural Networks

- CNNs automatically learn the parameters of the filters
  - a filter is a matrix of parameters
  - the key aspect is that a filter is adopted for the whole image
- Convolution can be applied in **multiple** layers
  - a layer l+1 is computed by convolving over output produced in layer l
  - Pooling is an operation often adopted for taking the most informative features that are learned after a convolution step



Pooling and subsampling layers

- What are the consequences of backpropagating gradients through max or average pooling layers?

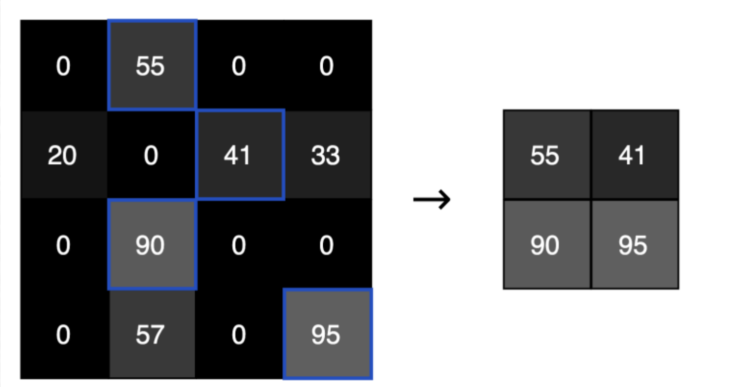
- **Max pooling**: the units that are responsible for the maximum within each zone *j, k* —the "winning units"— are the only to get the *backpropagated gradient*

- **Average pooling**: the averaging is simply a special type of convolution with a fixed kernel that computes the (possibly weighted) average of pixels in a zone
  - the required gradients are therefore like std conv. layers

- The subsampling step either samples every n^{th} output, or avoids needless computation by only evaluating every n^{th} pooling computation

Training in CNN:
Backpropagation and Max Pooling

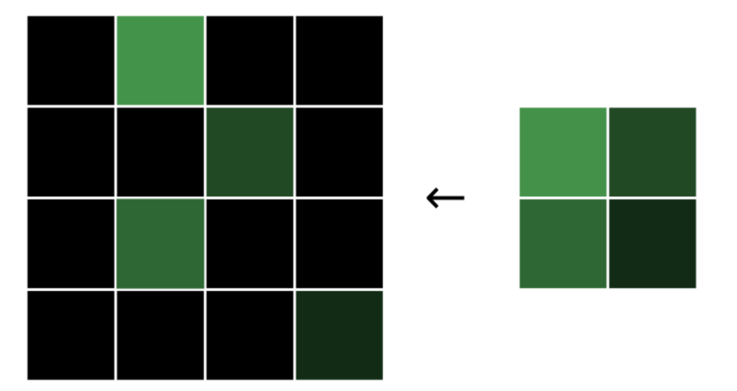
- A Max Pooling layer can’t be trained because it doesn’t actually have any weights
  - It still supports a method for it to calculate gradients



0	55	0	0
20	0	41	33
0	90	0	0
0	57	0	95

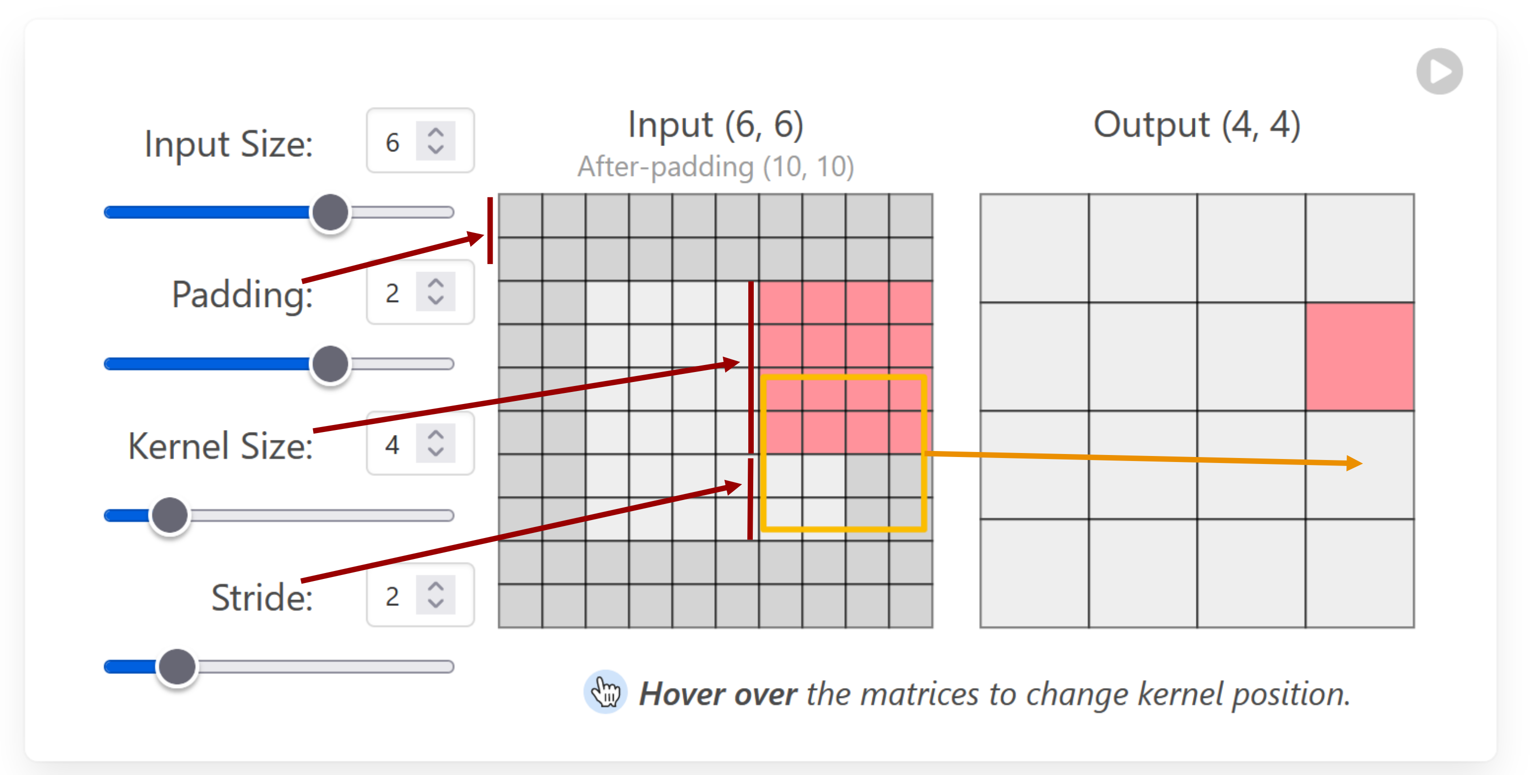
→

55	41
90	95



- How is ∂L / ∂inputs ?
  - An input pixel that isn’t the max value in its 2x2 block have *zero* marginal effect on the loss, as any slightly change of its value wouldn’t change the output at all!
    - ∂L / ∂inputs = 0 for any non-max pixels.
  - On the other hand, an input pixel that *is* the max value would have its value passed through to the output, so ∂output / ∂input = 1, meaning ∂L / ∂input = ∂L / ∂output.

Training a CNN: terminology



Input Size: 6

Padding: 2

Kernel Size: 4

Stride: 2

Input (6, 6)

After-padding (10, 10)

Output (4, 4)

***Hover over** the matrices to change kernel position.*

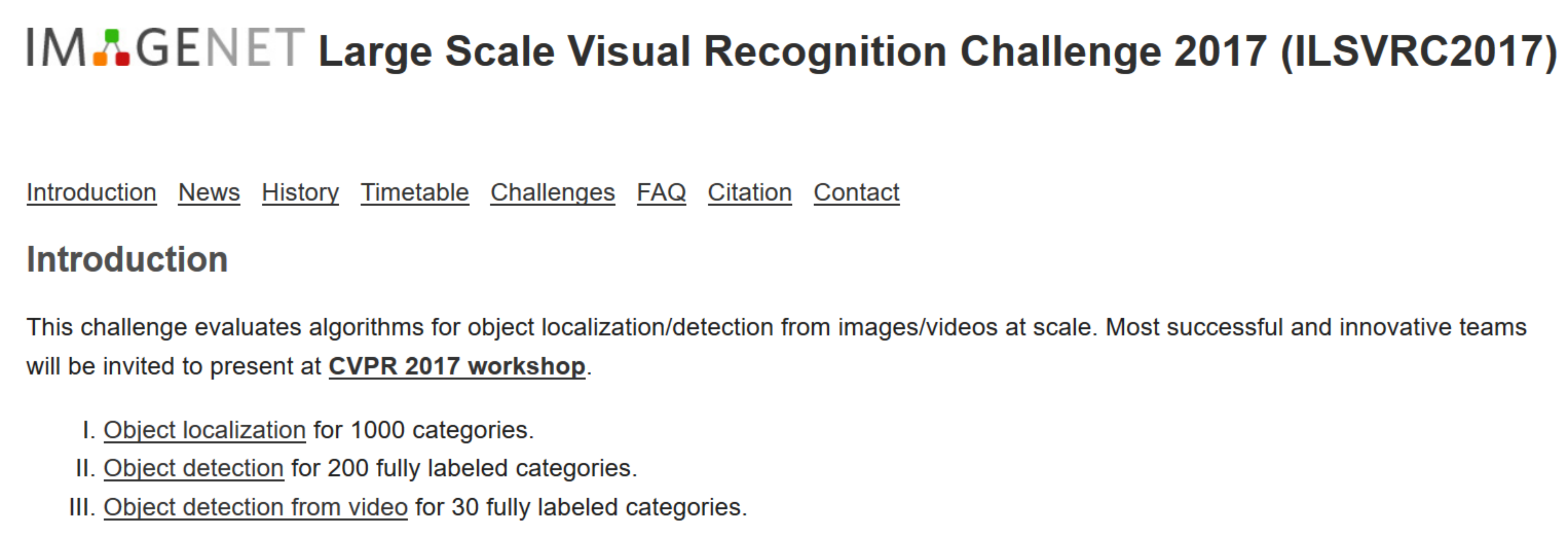
Outline

- Architectures and tasks
- Convolutional Neural Networks
  - Filters and Convolutions
  - Pooling
- Imagenet
- Applications of NNs:
  - Image processing: classification, Object Recognition
  - Text Classification:
    - Convolutional NNs over texts
    - Sentiment analysis
    - The Movie Review Dataset

The ImageNet challenge

- Crucial in demonstrating the effectiveness of deep CNNs
- Problem: recognize object categories in Internet imagery
- The 2012 ImageNet Large Scale Visual Recognition Challenge (ILSVRC) classification task - classify image from Flickr and other search engines into 1 of 1000 possible object categories
- Serves as a standard benchmark for deep learning
- The imagery was hand-labeled based on the presence or absence of an object belonging to these categories
- There are 1.2 million images in the training set with 732-1300 training images available per class
- A random subset of 50,000 images was used as the validation set, and 100,000 images were used for the test set where there are 50 and 100 images per class respectively

ImageNet Home Page



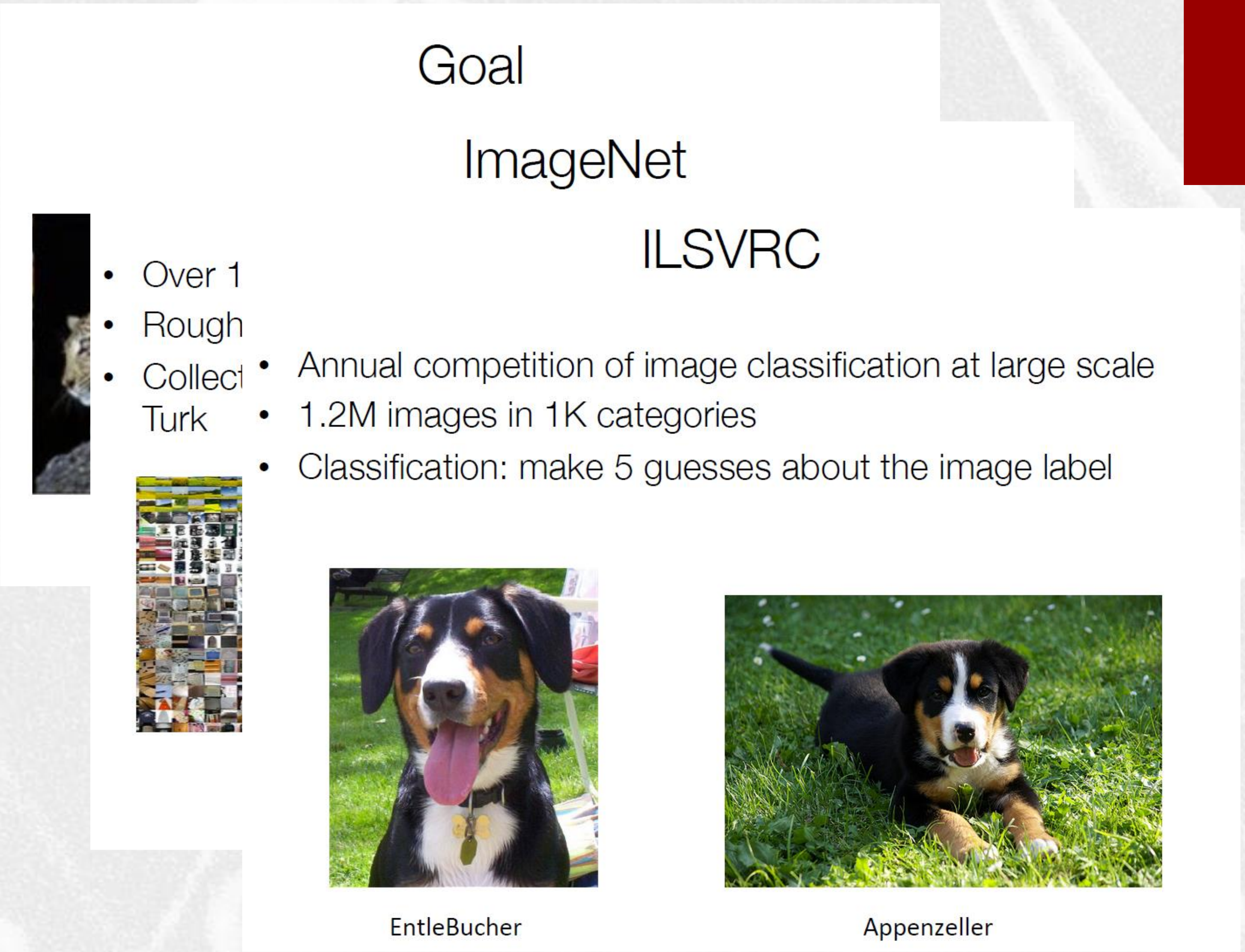
IMAGENET Large Scale Visual Recognition Challenge 2017 (ILSVRC2017)

Introduction News History Timetable Challenges FAQ Citation Contact

Introduction

This challenge evaluates algorithms for object localization/detection from images/videos at scale. Most successful and innovative teams will be invited to present at **CVPR 2017 workshop**.

I. Object localization for 1000 categories.
II. Object detection for 200 fully labeled categories.
III. Object detection from video for 30 fully labeled categories.



Goal

ImageNet

- Over 1
- Rough
- Collect
  Turk

ILSVRC

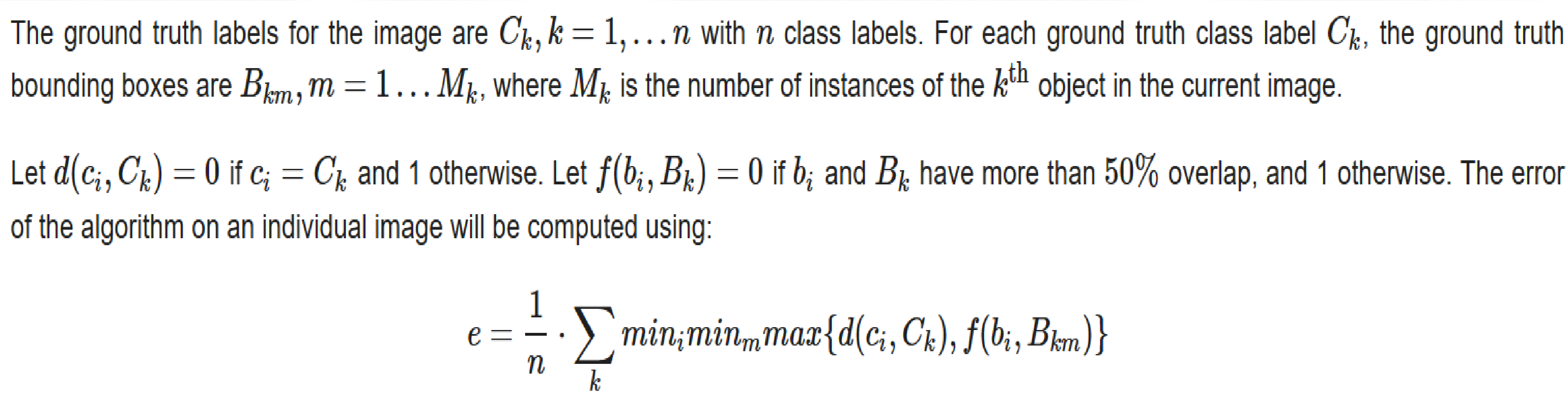
- Annual competition of image classification at large scale
- 1.2M images in 1K categories
- Classification: make 5 guesses about the image label

EntleBucher

Appenzeller

Object Location task

- Images, Class labels and Bounding boxes

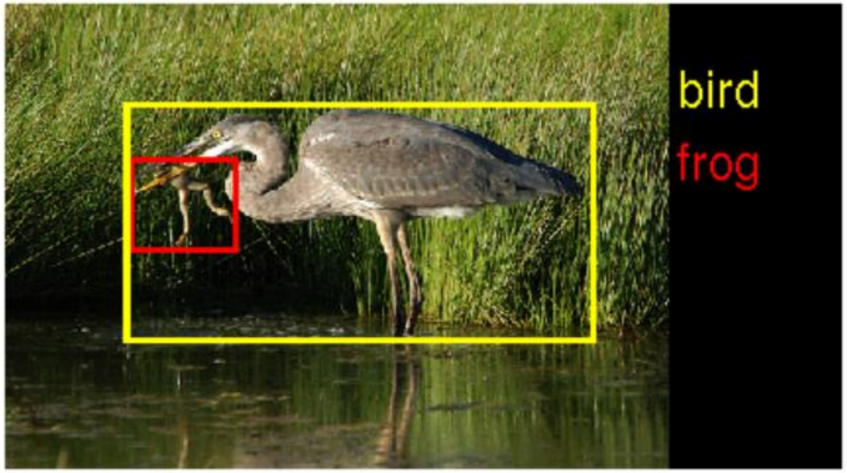


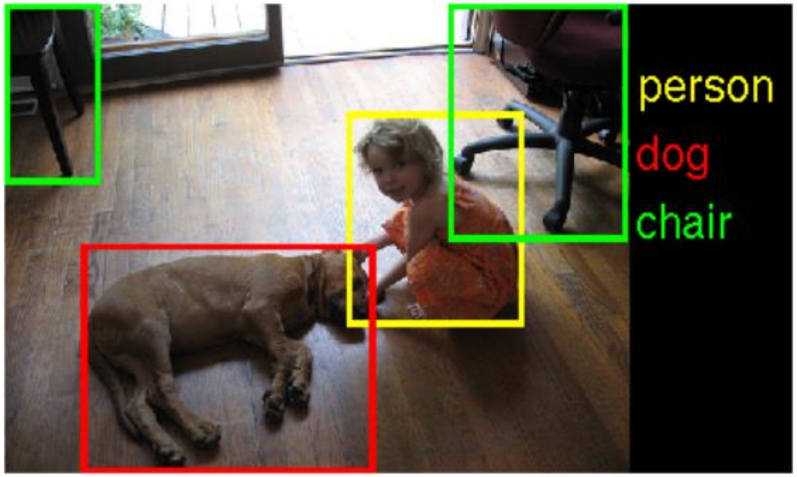
The ground truth labels for the image are $C_k, k = 1, \dots n$ with n class labels. For each ground truth class label C_k, the ground truth bounding boxes are $B_{km}, m = 1 \dots M_k$, where M_k is the number of instances of the k^{th} object in the current image.

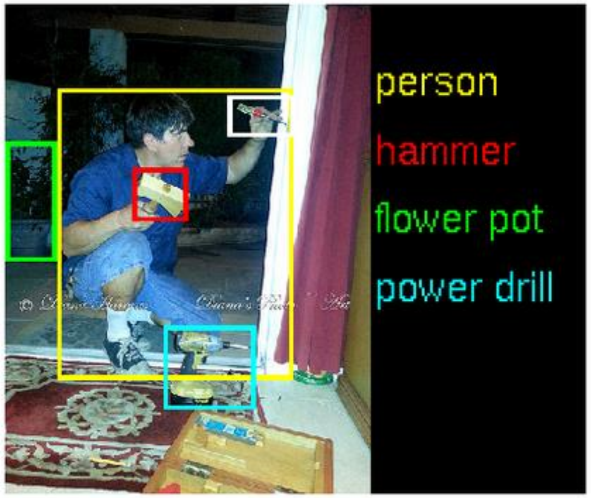
Let $d(c_i, C_k) = 0$ if $c_i = C_k$ and 1 otherwise. Let $f(b_i, B_k) = 0$ if b_i and B_k have more than 50% overlap, and 1 otherwise. The error of the algorithm on an individual image will be computed using:

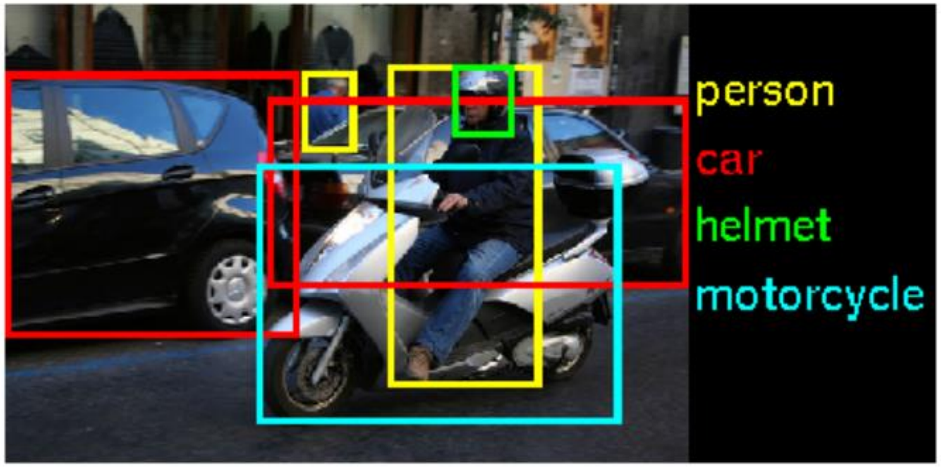
$$e = \frac{1}{n} \cdot \sum_k min_i min_m max\{d(c_i, C_k), f(b_i, B_{km})\}$$

ILSVRC2014 Examples









A plateau, then rapid advances

- "Top-5 error" is the % of times that the target label does not appear among the 5 highest-probability predictions
- Visual recognition methods not based on deep CNNs hit a plateau in performance at 25%

Name	Layers	Top-5 Error (%)	References
AlexNet	8	15.3	Krizhevsky et al. (2012)
VGG Net	19	7.3	Simonyan and Zisserman (2014)
ResNet	152	3.6	He et al. (2016)

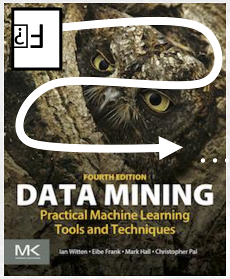
- Note: the performance for human agreement has been measured at 5.1% top-5 error
- Smaller filters have been found to lead to superior results in deep networks: the methods with 19 and 152 layers use filters of size 3×3

Simple filtering example

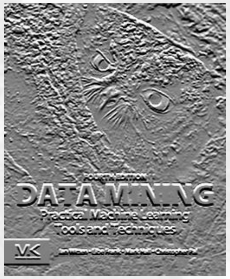
- Ex. consider the task of detecting edges in an image
- A well known technique is to filter an image with so-called "Sobel" filters, which involves convolving it with

$$\mathbf{W}_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \qquad \mathbf{W}_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

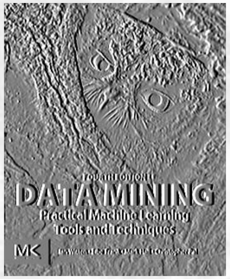
- Applied to the image X below, we have:


…
FOURTH EDITION
DATA MINING
Practical Machine Learning
Tools and Techniques
MK
Ian Witten • Eibe Frank • Mark Hall • Christopher Pal

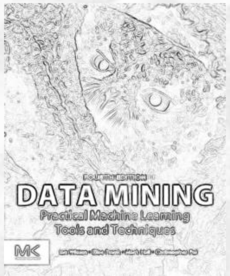
$\mathbf{X}$



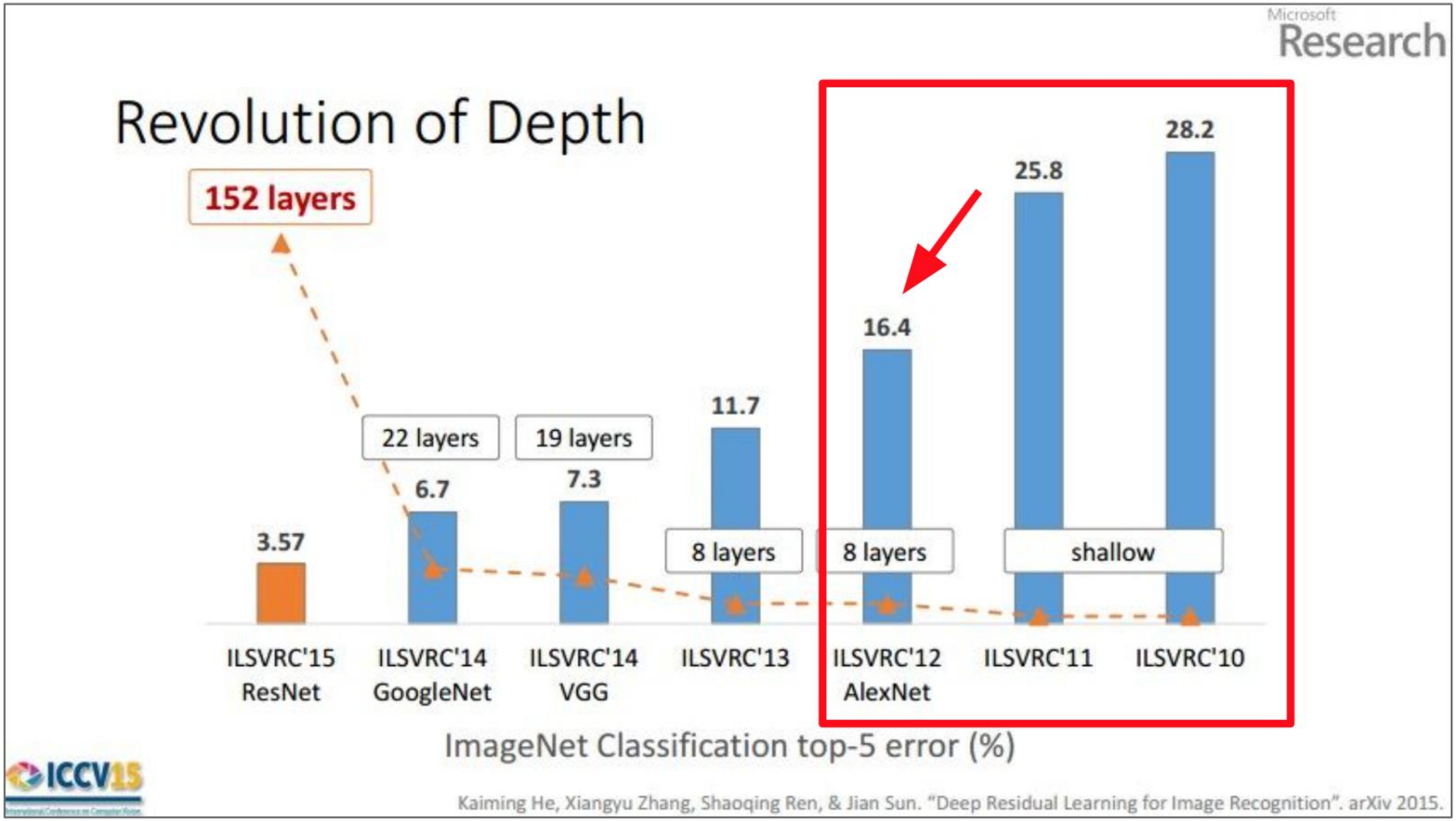
$\mathbf{G}_x = \mathbf{W}_x * \mathbf{X}$



$\mathbf{G}_y = \mathbf{W}_y * \mathbf{X}$



$\mathbf{G} = \sqrt{\mathbf{G}_x^2 + \mathbf{G}_y^2}$



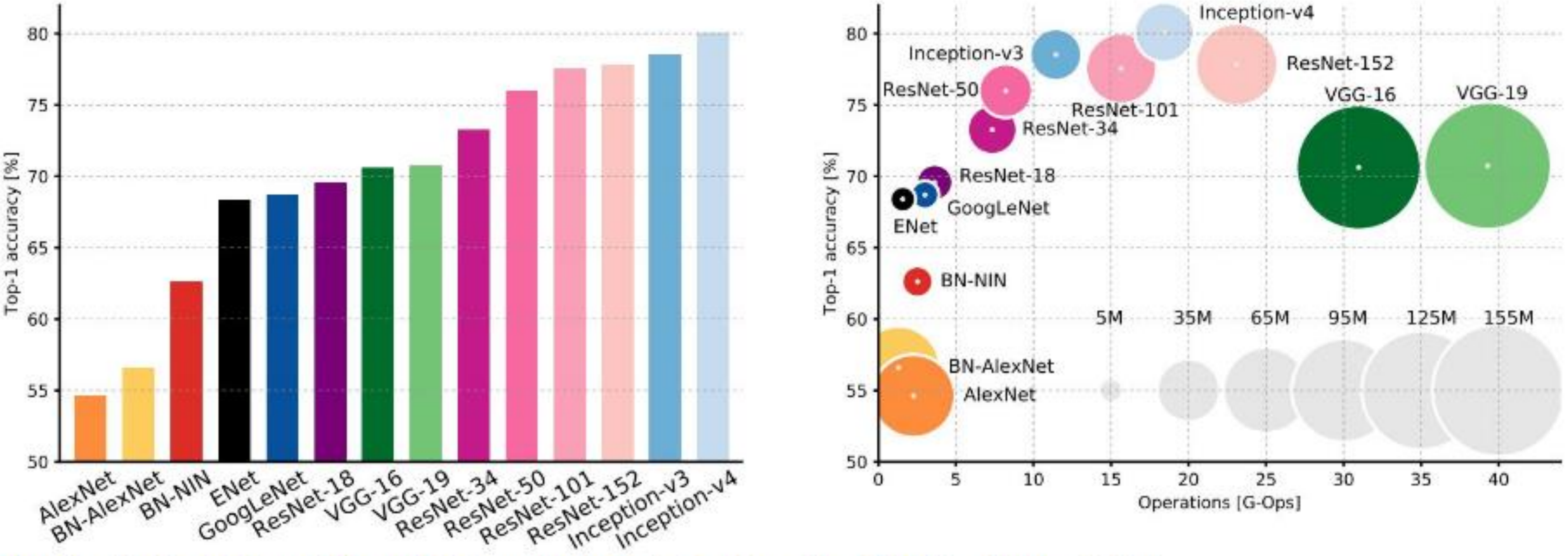
Microsoft Research

Revolution of Depth

ImageNet Classification top-5 error (%)

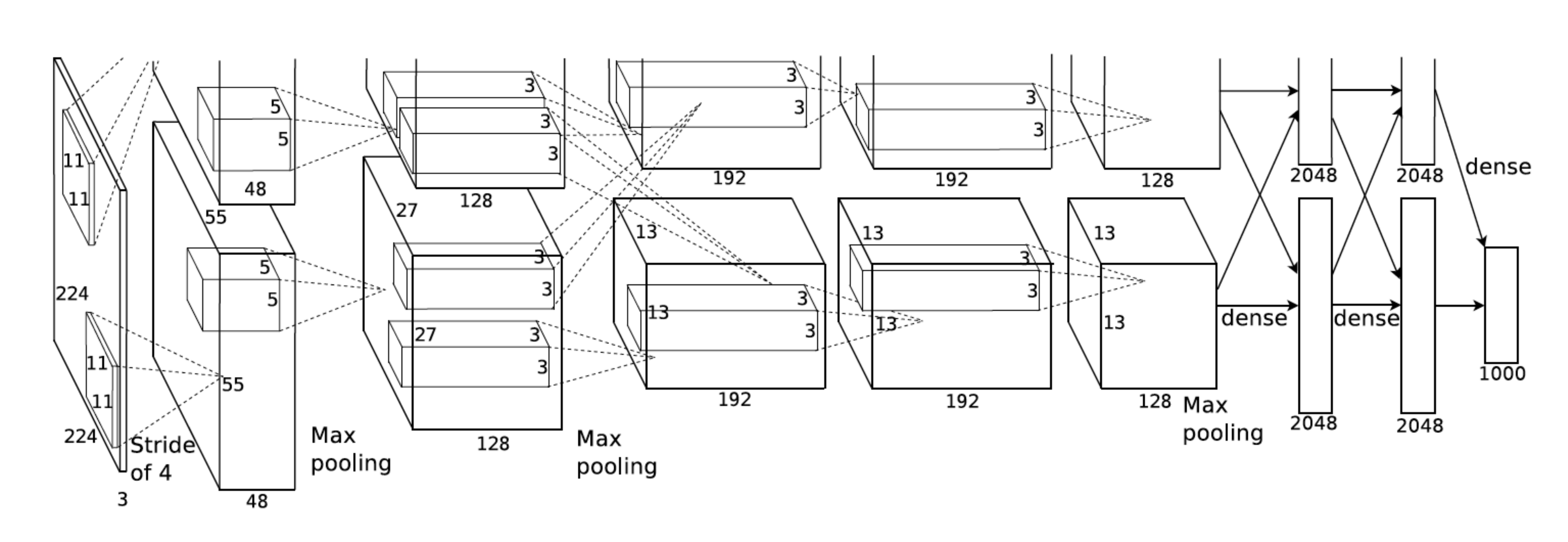
Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". arXiv 2015.

(slide from Kaiming He’s recent presentation)

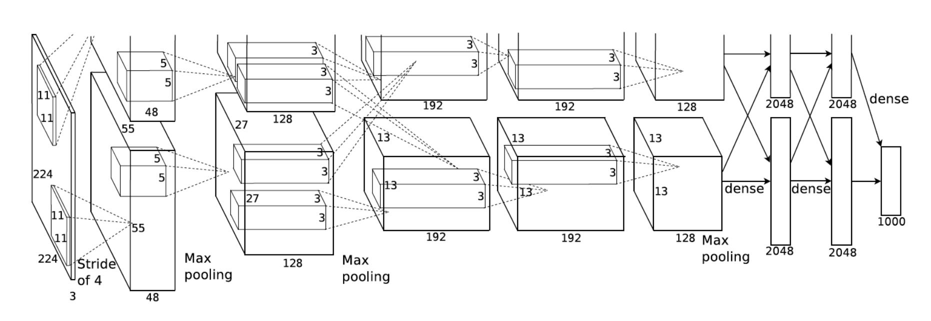


An Analysis of Deep Neural Network Models for Practical Applications, 2017.

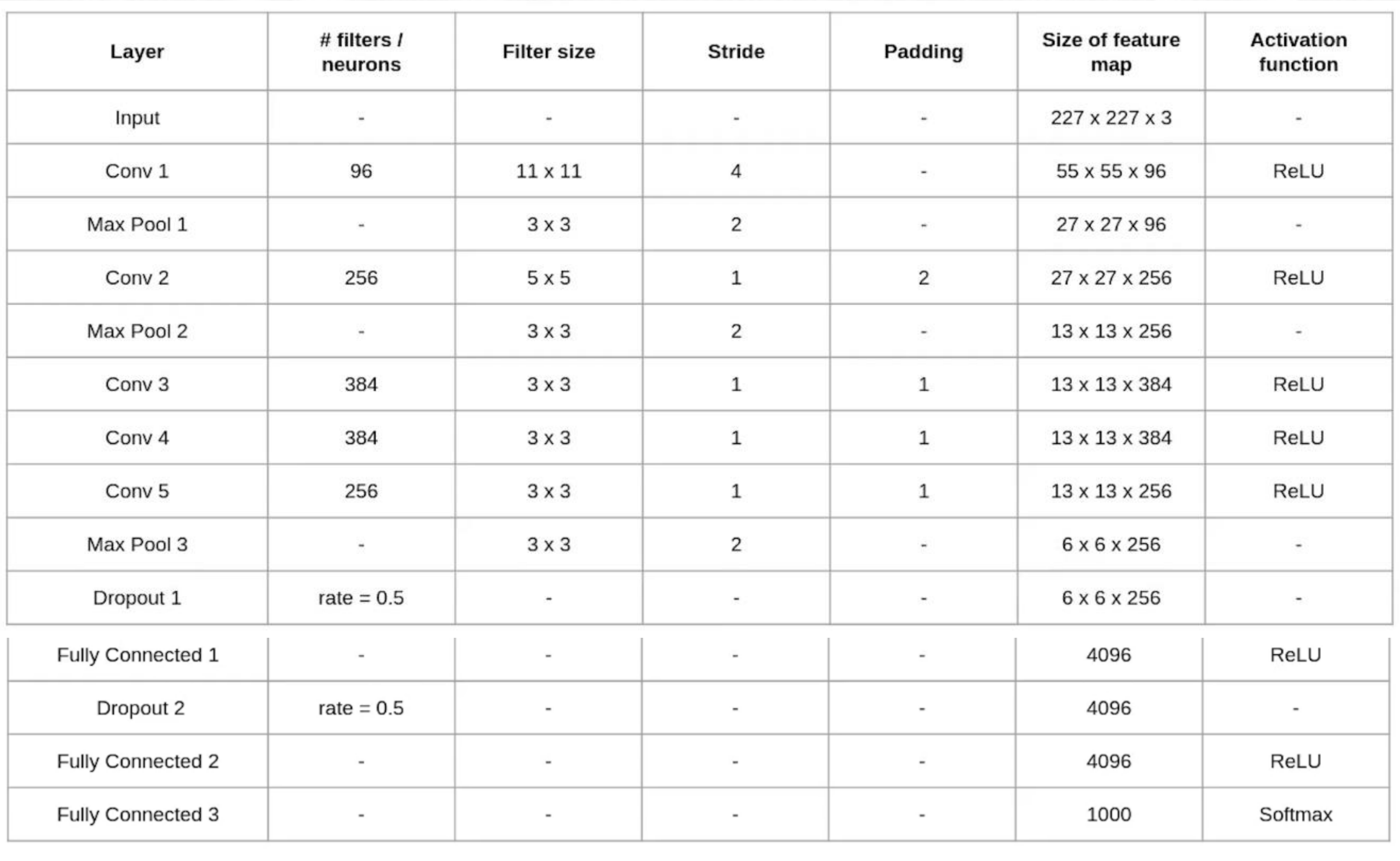
An example: ALexNet (8 Layers)



AlexNet won the 2012 ImageNet competition with a top-5 error rate of 15.3%, compared to the second place top-5 error rate of 26.2%



AlexNet: Overview



Layer	# filters / neurons	Filter size	Stride	Padding	Size of feature map	Activation function
Input	-	-	-	-	227 x 227 x 3	-
Conv 1	96	11 x 11	4	-	55 x 55 x 96	ReLU
Max Pool 1	-	3 x 3	2	-	27 x 27 x 96	-
Conv 2	256	5 x 5	1	2	27 x 27 x 256	ReLU
Max Pool 2	-	3 x 3	2	-	13 x 13 x 256	-
Conv 3	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 4	384	3 x 3	1	1	13 x 13 x 384	ReLU
Conv 5	256	3 x 3	1	1	13 x 13 x 256	ReLU
Max Pool 3	-	3 x 3	2	-	6 x 6 x 256	-
Dropout 1	rate = 0.5	-	-	-	6 x 6 x 256	-
Fully Connected 1	-	-	-	-	4096	ReLU
Dropout 2	rate = 0.5	-	-	-	4096	-
Fully Connected 2	-	-	-	-	4096	ReLU
Fully Connected 3	-	-	-	-	1000	Softmax

AlexNet: the architecture

- It has 8 layers with learnable parameters.
- The input to the Model is RGB images.
- It has 5 convolution layers with a combination of max-pooling layers.
- Then it has 3 fully connected layers.
- The activation function used in all layers is **Relu**, whereas Softmax is used in the output layer is
- It used **two Dropout layers**.
- The total number of parameters in this architecture is **62.3 million**.

What has been learnt?

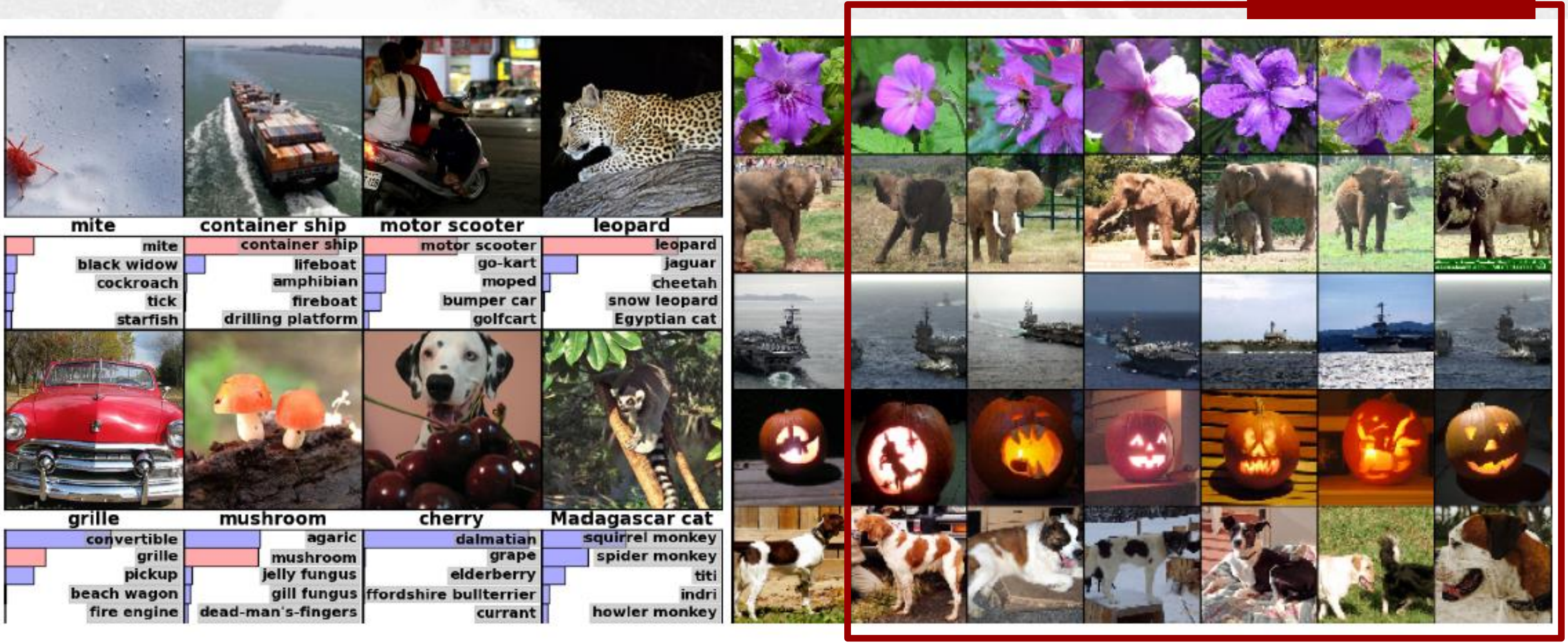
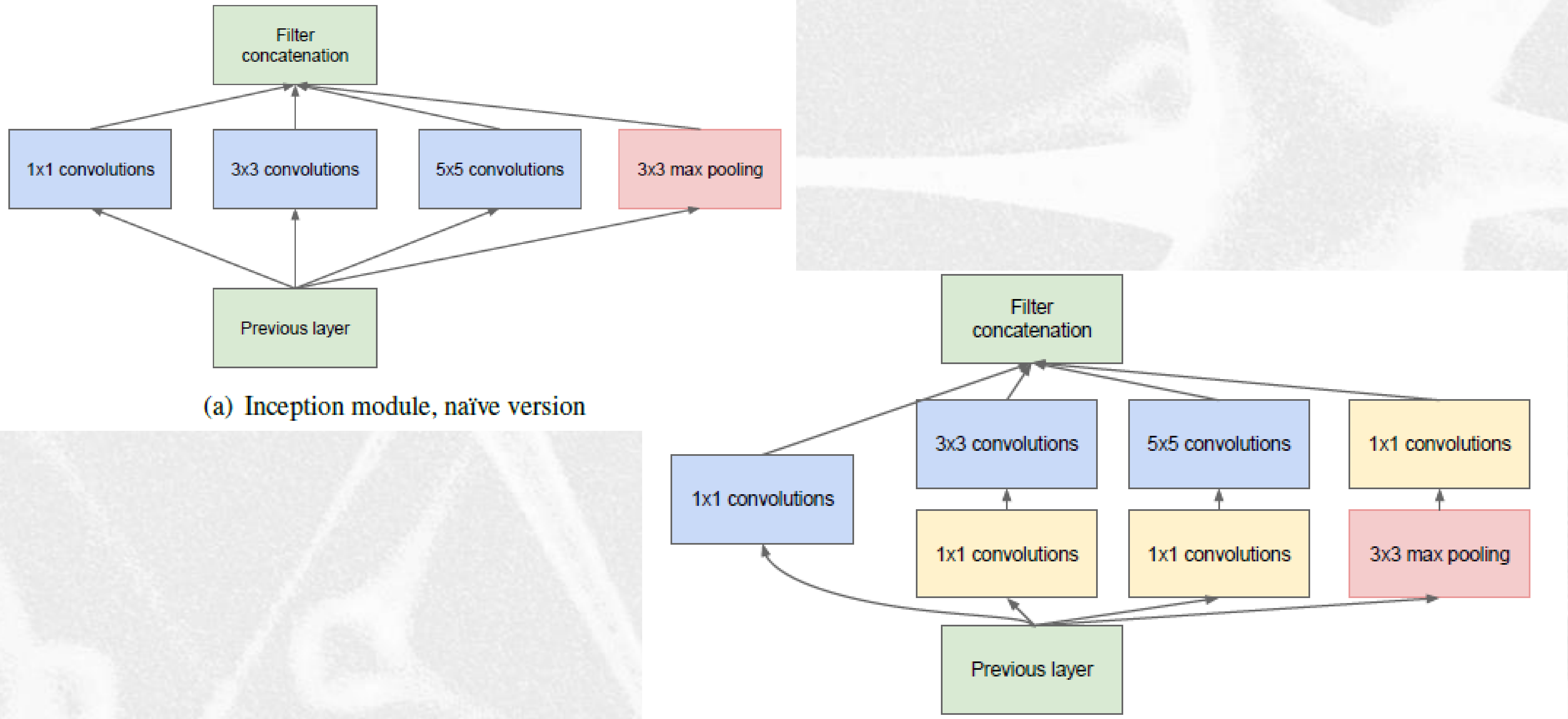


Figure 4: **(Left)** Eight ILSVRC-2010 test images and the five labels considered most probable by our model. The correct label is written under each image, and the probability assigned to the correct label is also shown with a red bar (if it happens to be in the top 5). **(Right)** Five ILSVRC-2010 test images in the first column. The remaining columns show the six training images that produce feature vectors in the last hidden layer with the smallest Euclidean distance from the feature vector for the test image.

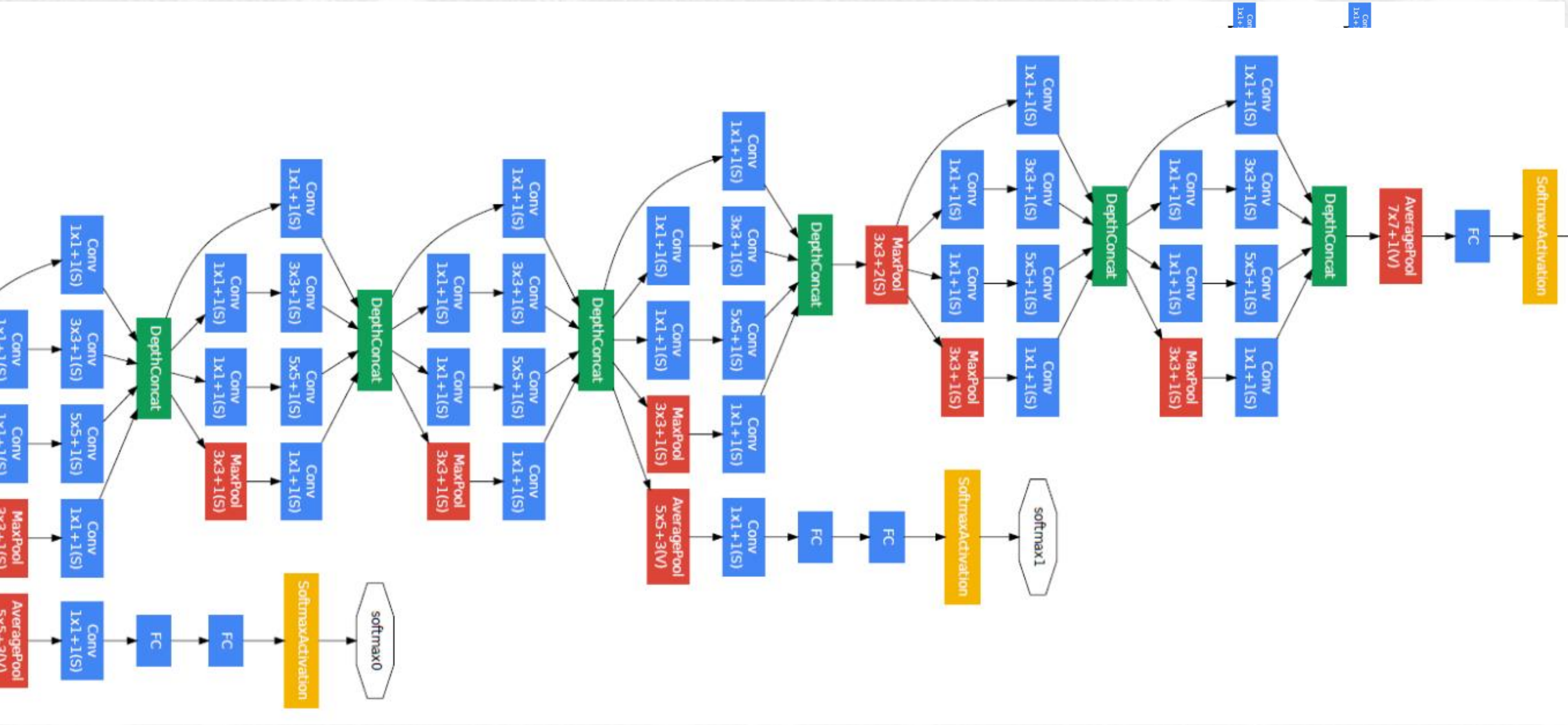
GoogleLeNet (Inception V1)



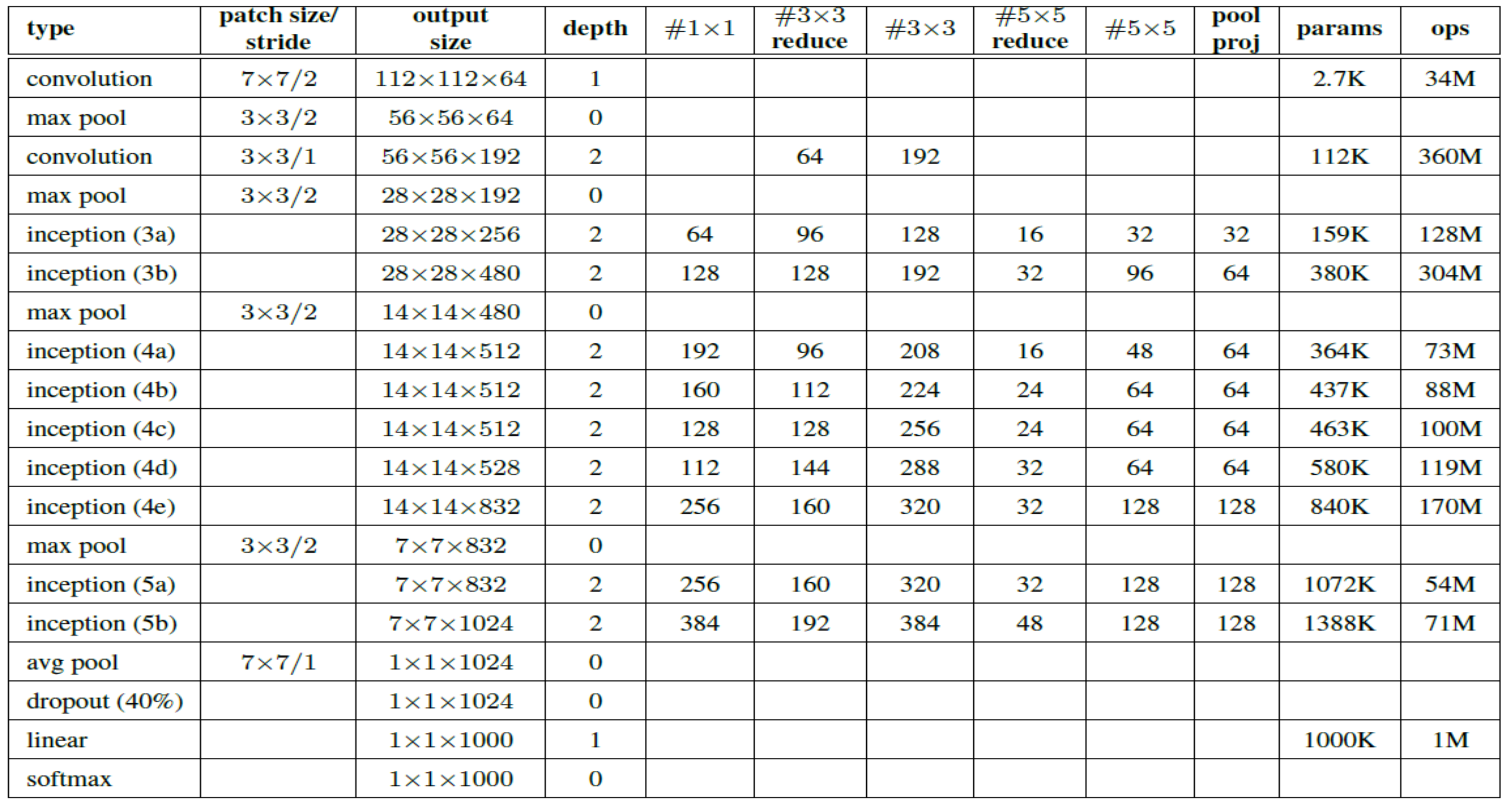
(a) Inception module, naïve version

(b) Inception module with dimensionality reduction

The full architecture



Parameters in GoogleLeNet



type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

Visualization

Visualizing the filters learned by a CNN

- Learned edge-like filters and texture-like filters are frequently observed in the early layers of CNNs trained using natural images
- Since each layer in a CNN involves filtering the feature map below, so as one moves up the receptive fields become larger
- Higher- level layers learn to detect larger features, which often correspond to textures, then small pieces of objects

How to visualize hidden layers

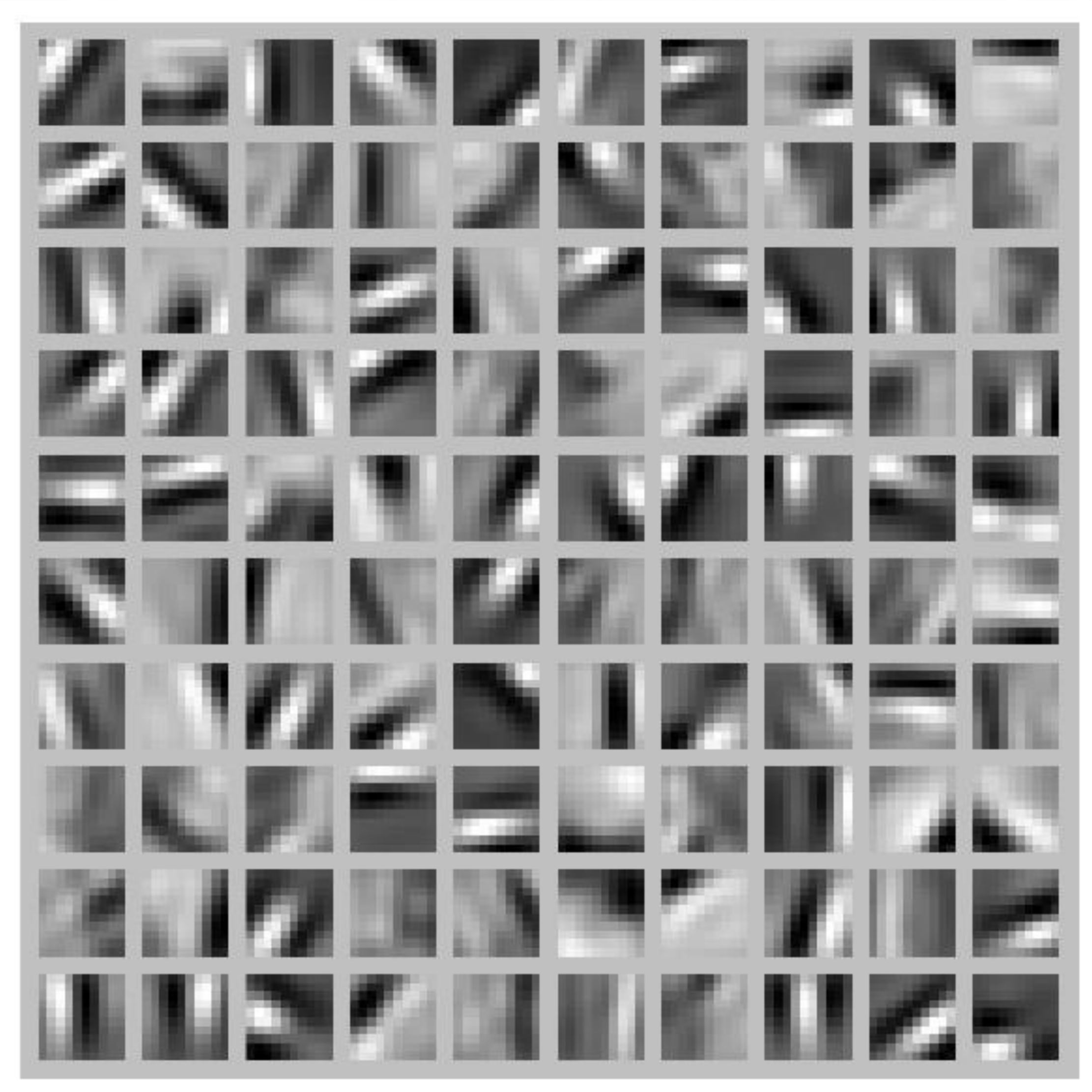
- Imagine to train a neural classifier on 10 × 10 images, so that n = 100. Each hidden unit *i* computes a function of the input:

$$a_i^{(2)} = f\left(\sum_{j=1}^{100} W_{ij}^{(1)} x_j + b_i^{(1)}\right)$$

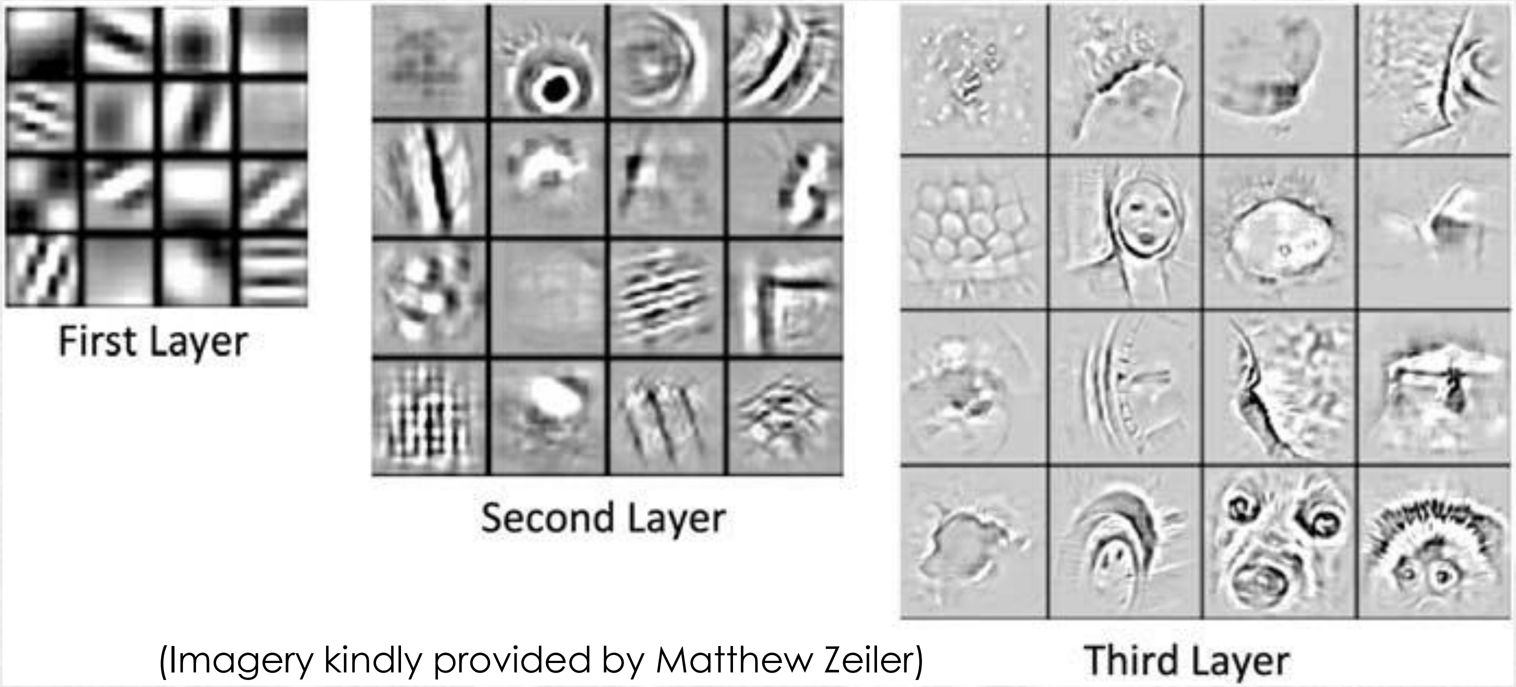
- What input image x would cause $a^{(1)}_i$ to be maximally activated?
- (When $||x||^2 = \sum_{i=1}^{100} x_i^2 \leq 1$) the input which maximally activates hidden unit *i* is given by setting pixel x_j to:

$$x_j = \frac{W_{ij}^{(1)}}{\sqrt{\sum_{j=1}^{100} (W_{ij}^{(1)})^2}}$$

Example: 100 hidden units



Visualizing the filters learned by a CNN

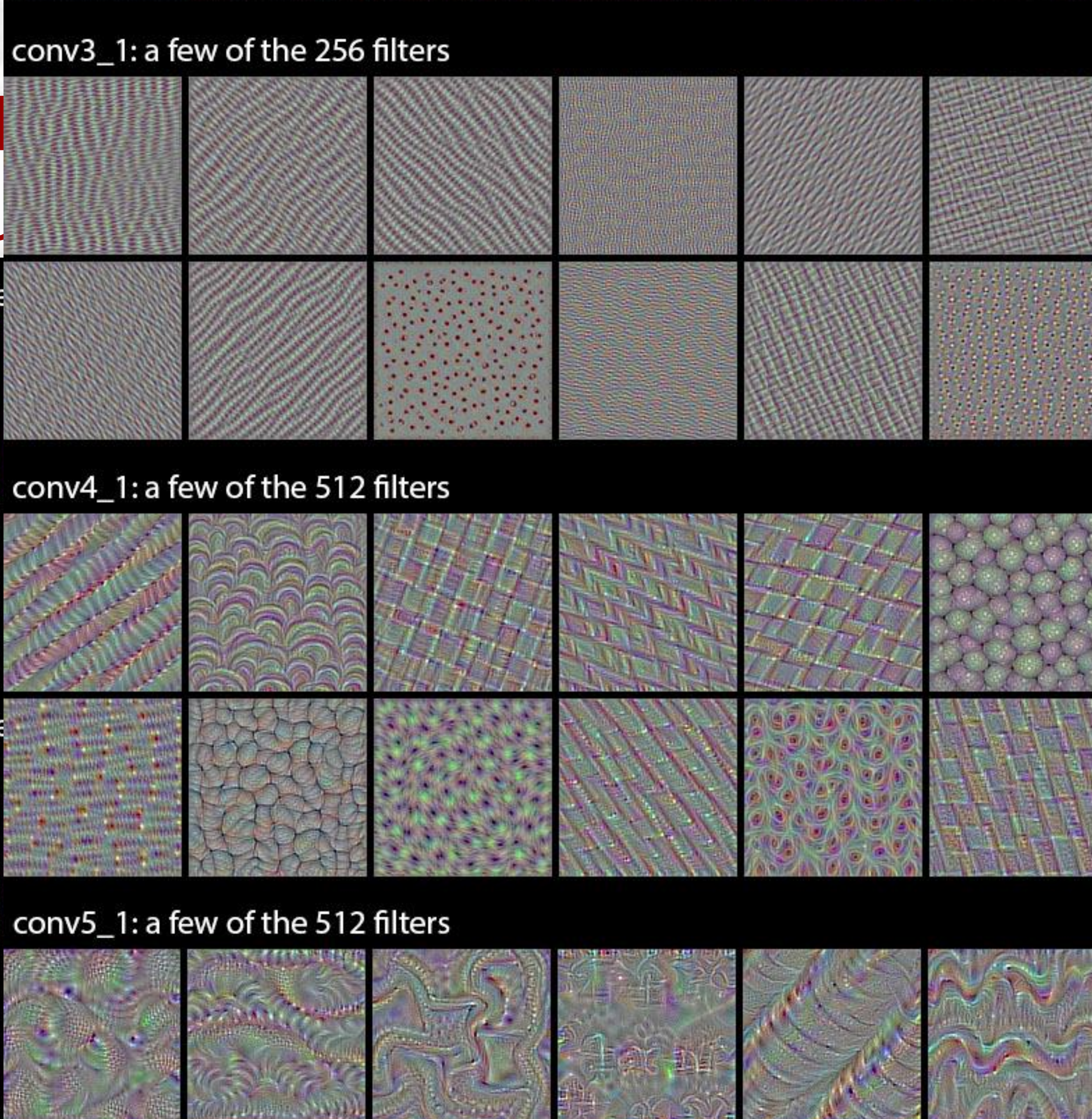


(Imagery kindly provided by Matthew Zeiler)

- Above are the strongest activations of random neurons projecting the activation back into image space using the deconvolution approach of Zeiler and Fergus (2013).

ConvNet
filters visu

conv1_1: a fe
conv2_1: a fe


conv3_1: a few of the 256 filters
conv4_1: a few of the 512 filters
conv5_1: a few of the 512 filters

Current CNNs: Yolo

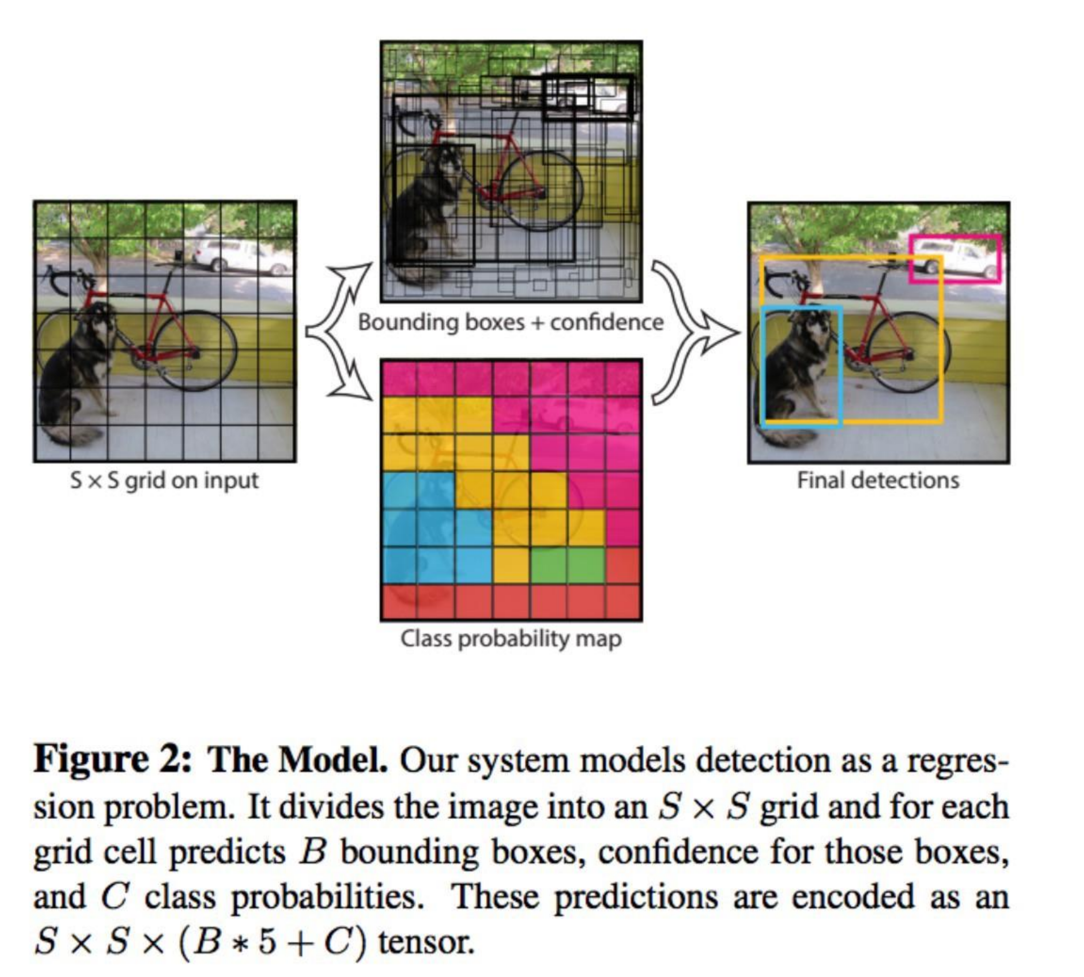


Figure 2: The Model. Our system models detection as a regression problem. It divides the image into an $S \times S$ grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.

Current CNNs: Yolo

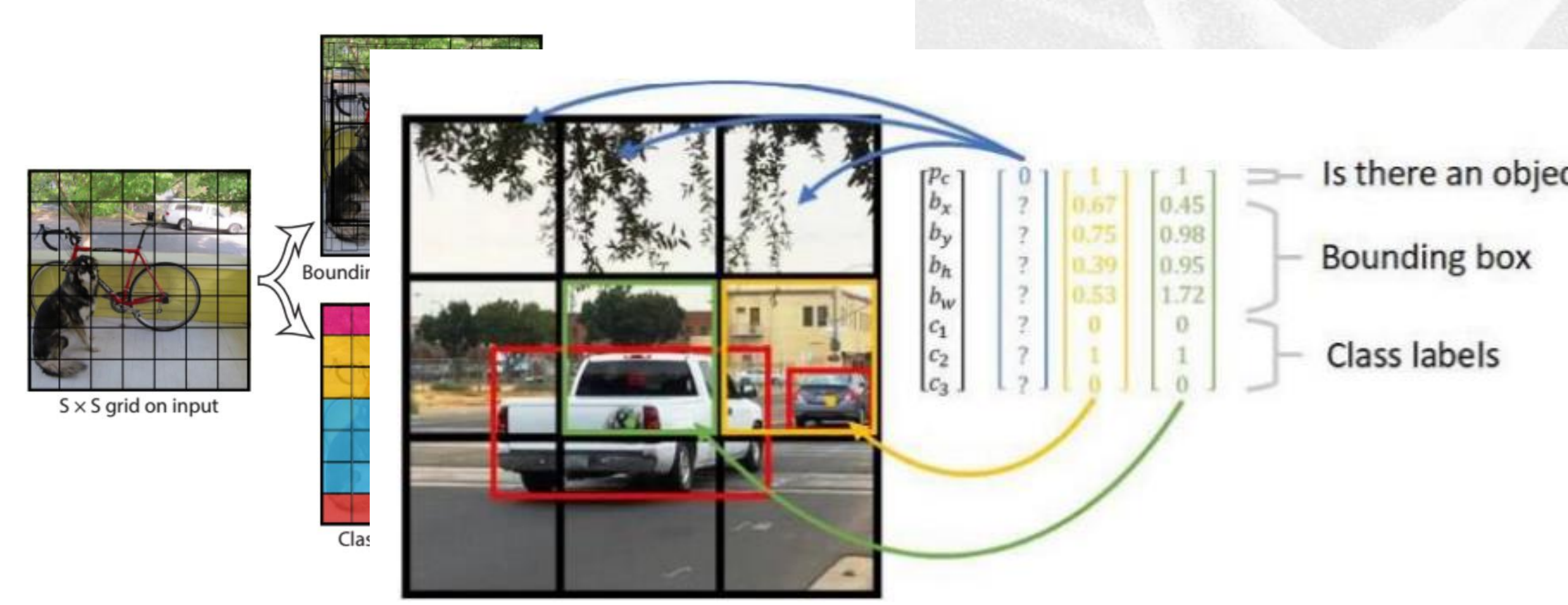
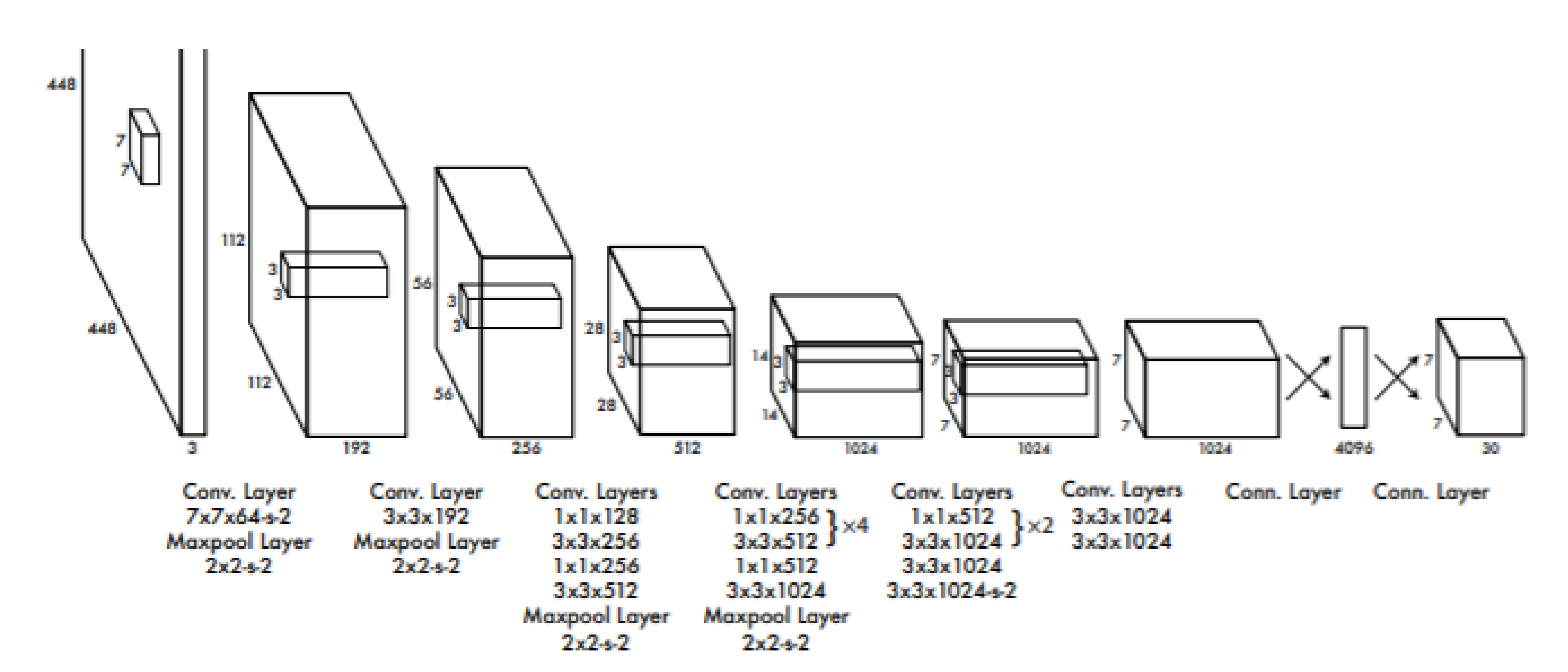


Figure 2: The Model. Our system models detection as a regression problem. It divides the image into an $S \times S$ grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.

Yolo: the architecture



Yolo: Results

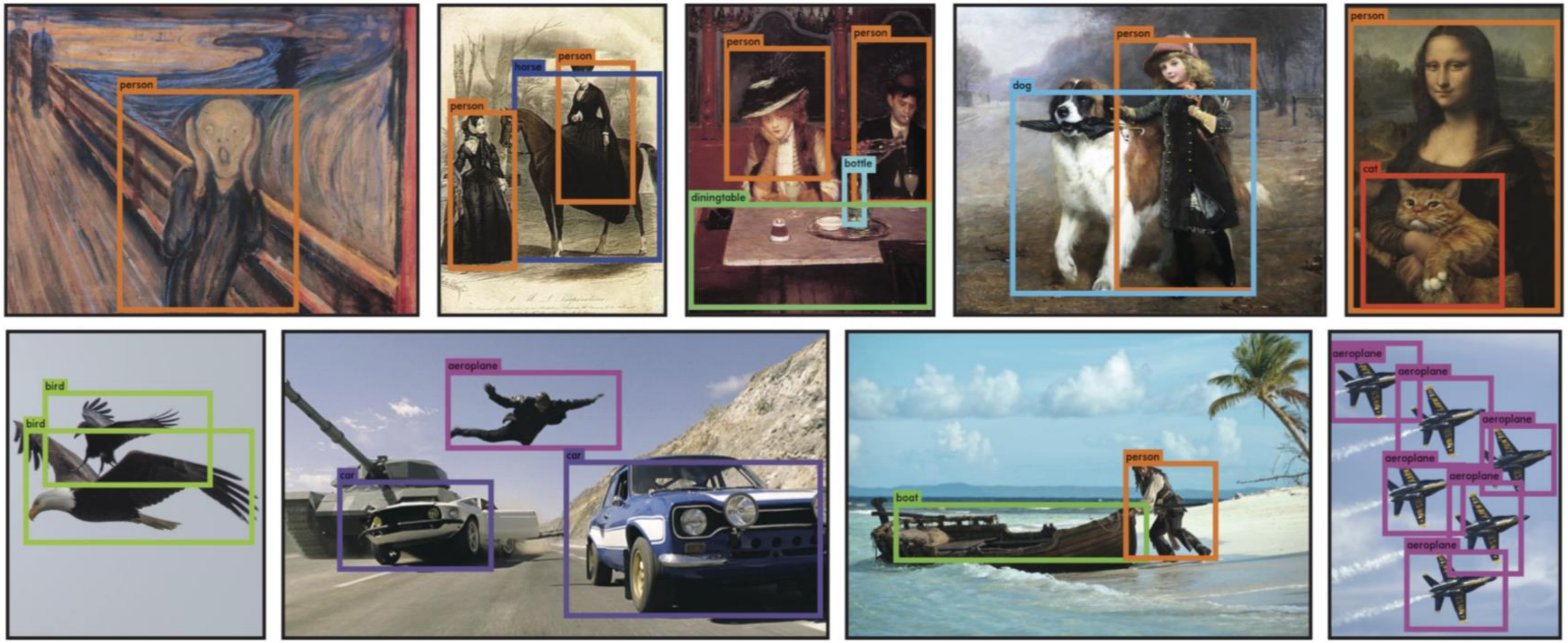


Figure 6: Qualitative Results. YOLO running on sample artwork and natural images from the internet. It is mostly accurate although it does think one person is an airplane.

Outline

- Architectures and tasks
- Convolutional Neural Networks
  - Filters and Convolutions
  - Pooling
- Imagenet
- Applications of NNs:
  - Image processing: classification, Object Recognition
  - Text Classification:
    - Convolutional NNs over texts
    - Sentiment analysis
    - The Movie Review Dataset

Sentence encoding & convolution

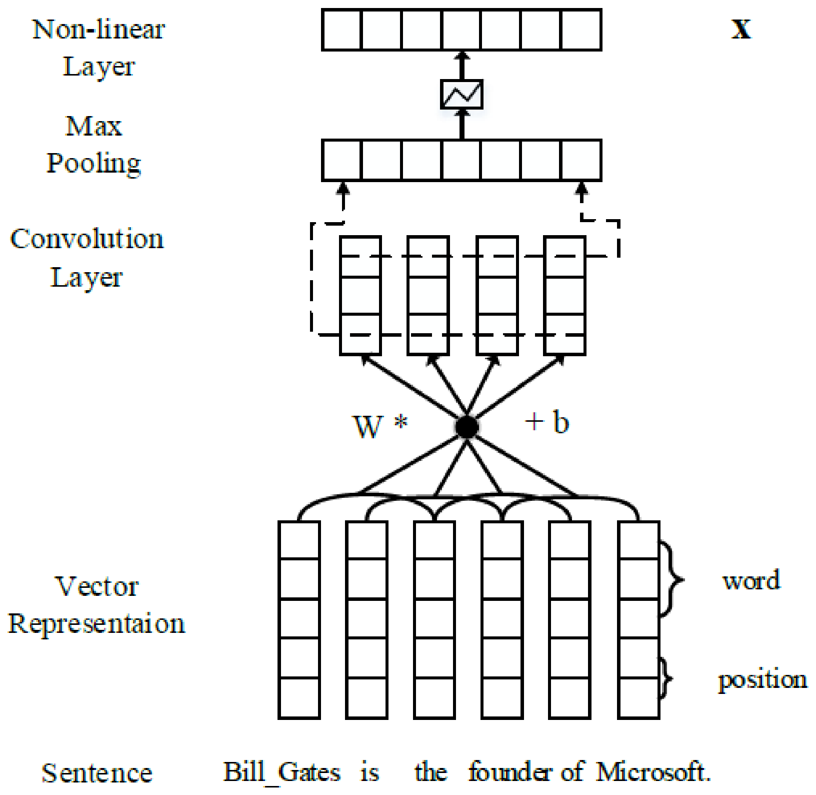
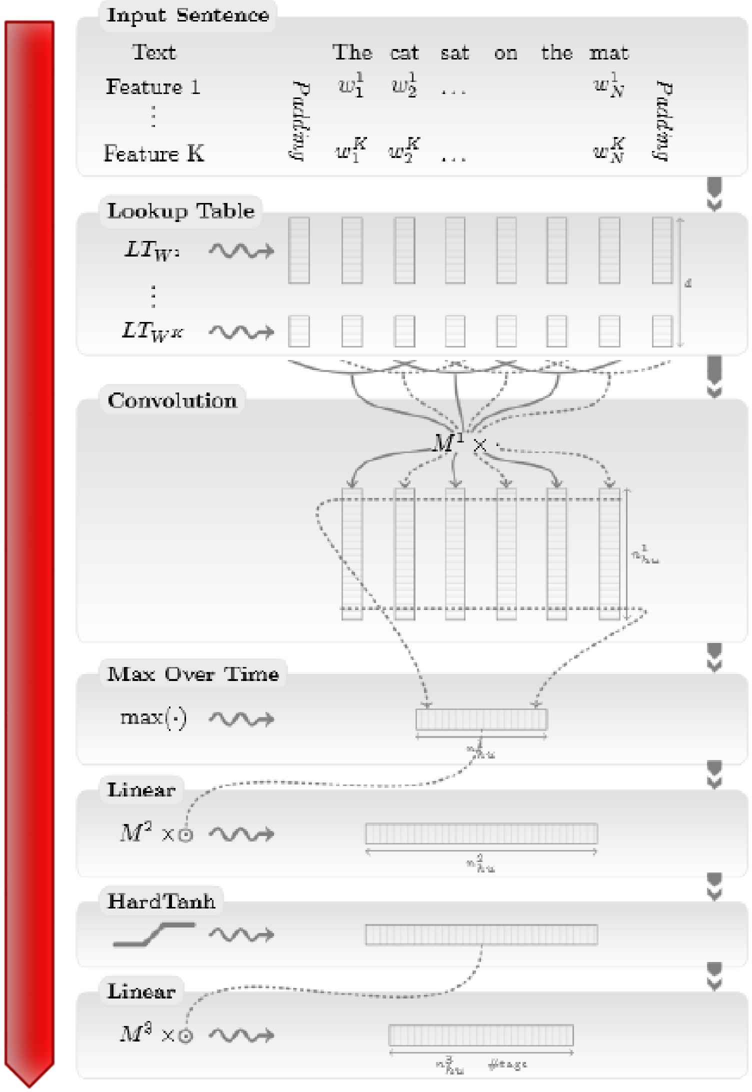


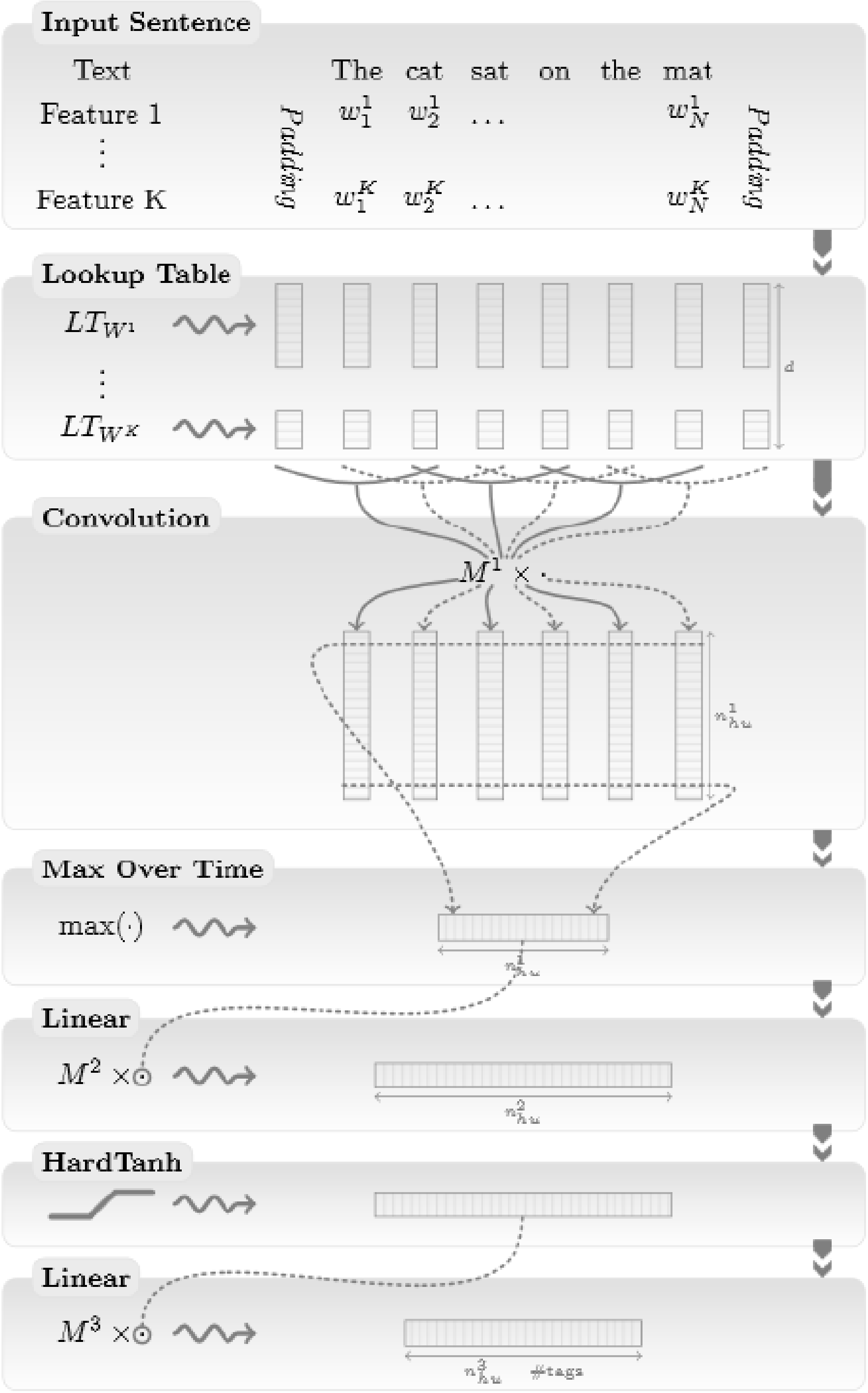
Figure 2: The architecture of CNN/PCNN used for sentence encoder.

From Collobert et al., 2011

- In this contribution, we try to excel on **multiple benchmarks** while *avoiding task-specific engineering*. Instead *we use a single learning system* able to **discover adequate internal representations**. In fact we view the *benchmarks as indirect measurements of the relevance of the internal representations* discovered by the learning procedure, and we posit that these *intermediate representations are more general than any of the benchmarks*.
- Our desire to avoid task-specific engineered features prevented us from using a large body of linguistic knowledge
- The architecture takes the input sentence and learns several layers of feature extraction that process the inputs. The features computed by the deep layers of the network are automatically trained by backpropagation to be relevant to the task.

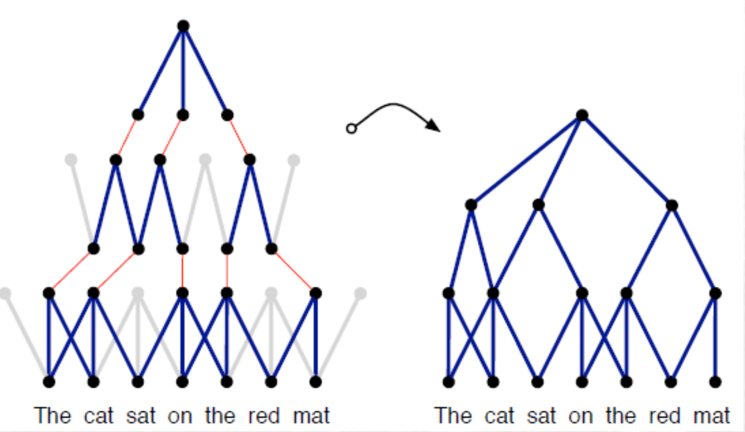
- Collobert and Weston used CNNs to achieve (near) state-of-the-art results on many traditional NLP tasks, such as POS tagging, SRL, etc.
- CNN at the bottom + CRF on top.
- Collobert et al., "Natural Language Processing (almost) from scratch", JLMR 2011.

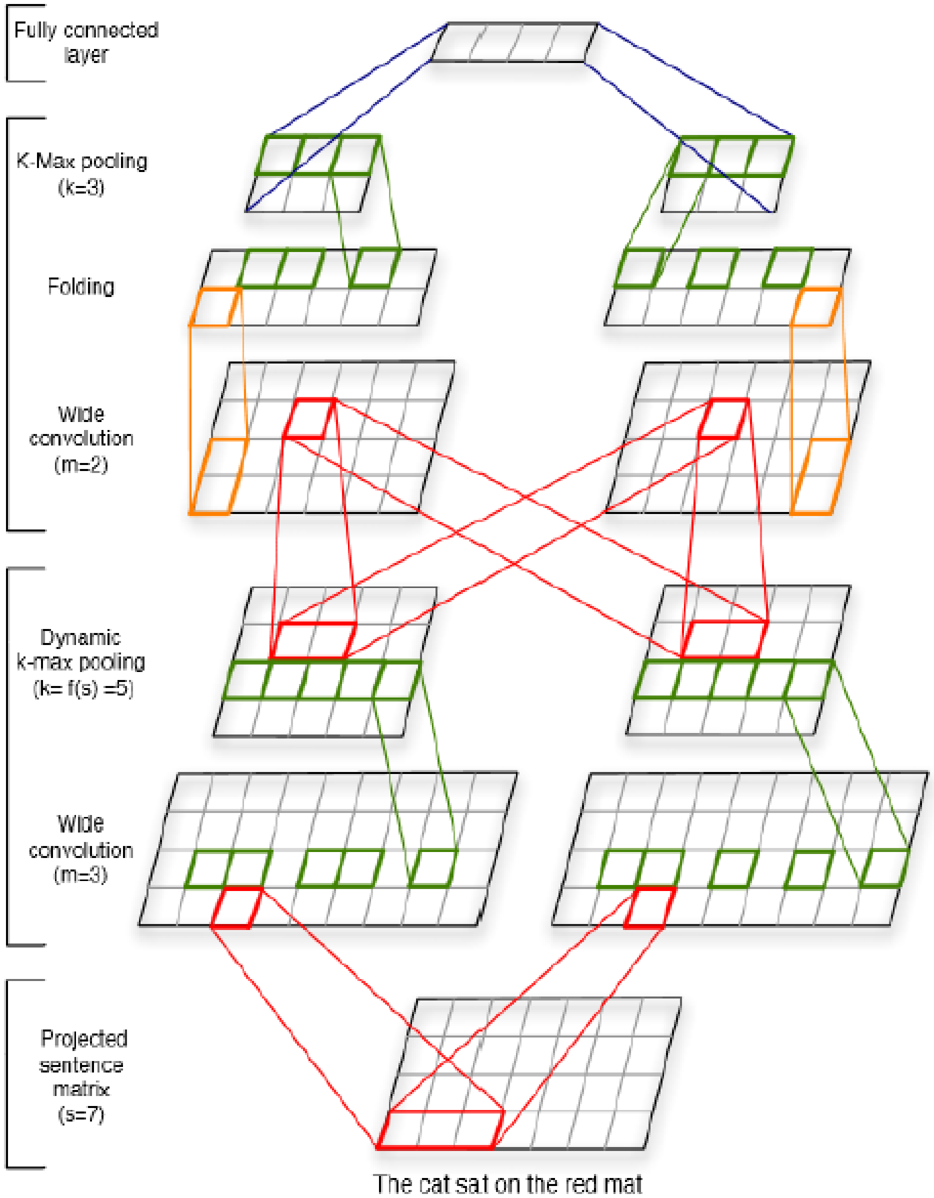



Input Sentence
Text
The cat sat on the mat
Feature 1
w_1^1 w_2^1 … w_N^1
Feature K
w_1^K w_2^K … w_N^K
Padding
Padding
Lookup Table
LT_{W^1}
LT_{W^K}
d
Convolution
$M^1 \times \cdot$
n_{hu}^1
Max Over Time
$\max(\cdot)$
n_{hu}^1
Linear
$M^2 \times \odot$
n_{hu}^2
HardTanh
Linear
$M^3 \times \odot$
n_{hu}^3 #tags

Dynamic CNNs

- Kalchbrenner et al., "*Convolutional Neural Network for Modelling Sentences*", ACL 2014
- Take the dot-product of a filter with every n-gram of the sentence

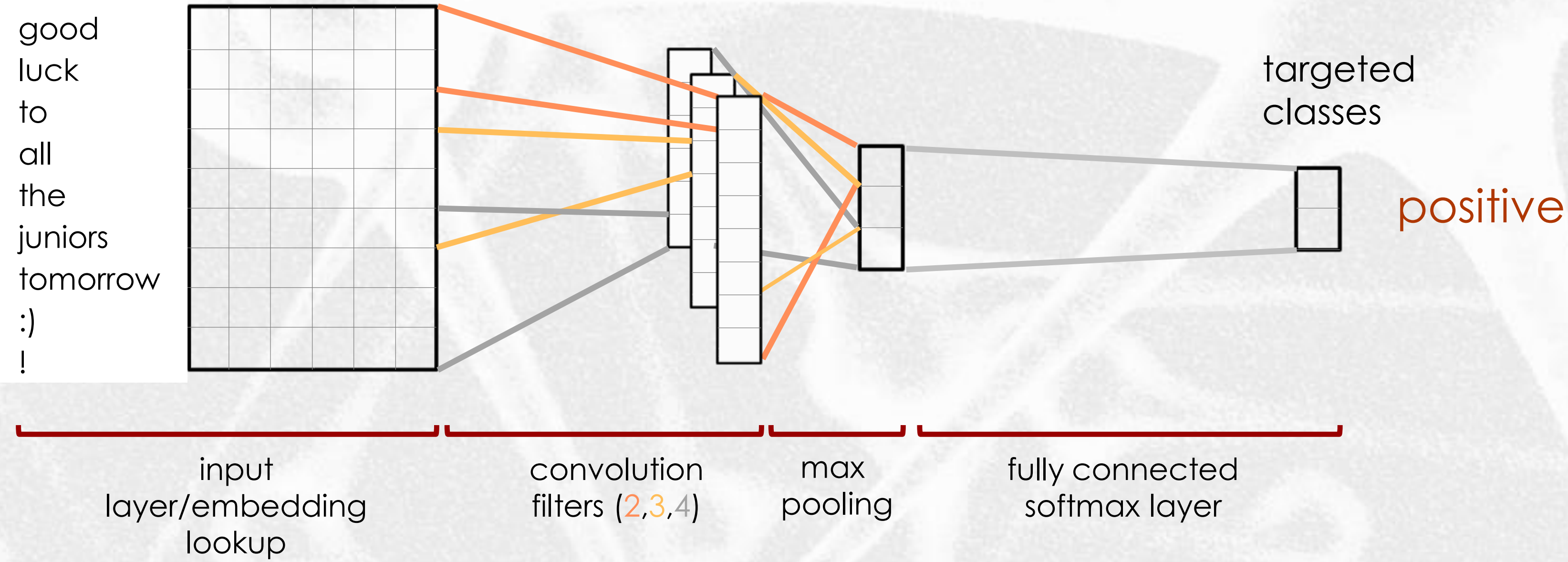



Fully connected layer
K-Max pooling (k=3)
Folding
Wide convolution (m=2)
Dynamic k-max pooling (k= f(s) =5)
Wide convolution (m=3)
Projected sentence matrix (s=7)
The cat sat on the red mat

A DCNN for the seven word input sentence. Word embeddings have sized= 4. The network has two convolutional layers with two feature maps each. The widths of the filters at the two layers are respectively 3 and 2. The (dynamic) *k*-max pooling layers have values *k* of 5 and 3

A CNN architecture for sentence classification (Kim,2014)

good luck to all the juniors tomorrow :) !


good
luck
to
all
the
juniors
tomorrow
:)
!
targeted classes
positive
input layer/embedding lookup
convolution filters (2,3,4)
max pooling
fully connected softmax layer

Multi-channel CNNs

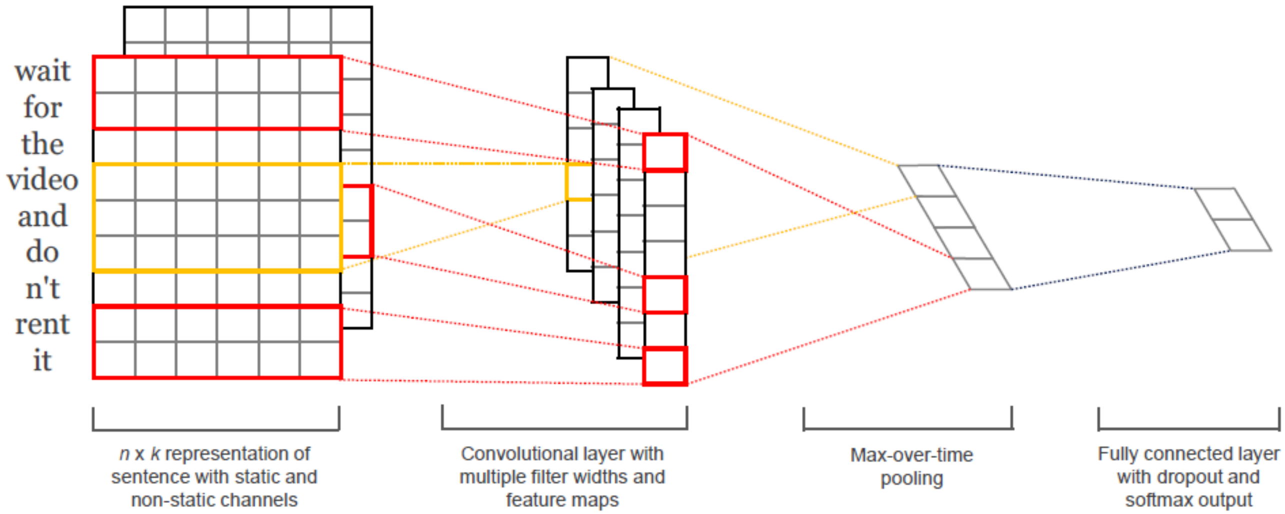

wait
for
the
video
and
do
n't
rent
it
n x *k* representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output

Figure 1: Model architecture with two channels for an example sentence.

- Two “channels” of embeddings (i.e. look-up tables).
- One is allowed to change, while one is kept fixed.
- Both initialized with `word2vec`.

Datasets

Sentence/phrase-level classification tasks

Data	c	l	N	$\|V\|$	$\|V_{pre}\|$	Prev SotA
MR	2	20	10662	18765	16448	79.5
SST-1	5	18	11855	17836	16262	48.7
SST-2	2	19	9613	16185	14838	87.8
Subj	2	23	10000	21323	17913	93.6
TREC	6	10	5952	9592	9125	95.0
CR	2	19	3775	5340	5046	82.7
MPQA	2	3	10606	6246	6083	87.2

- c: number of labels
- l: average sentence length
- N: number of sentences
- $|V|$: vocab size ($|V_{pre}|$ is words already in `word2vec`)

MR dataset (Pang & Lee, 2005)

- **Negative**:
  - “it's so laddish and juvenile , only teenage boys could possibly find it funny . “
  - while the performances are often engaging , this loose collection of largely improvised numbers would probably have worked better as a one-hour *tv* documentary .

- **Positive**
  - if you sometimes like to go to the movies to have fun , wasabi is a good place to start .
  - gosling provides an amazing performance that dwarfs everything else in the film .

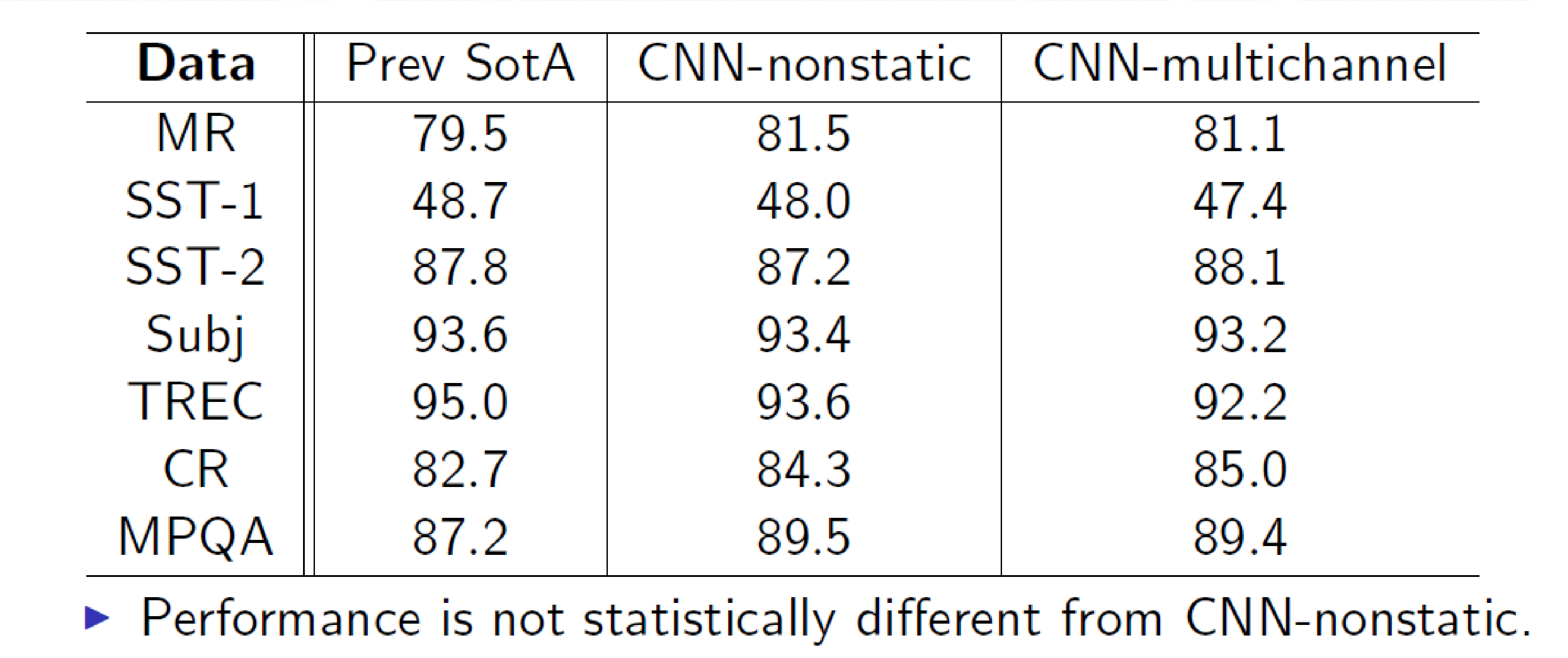
SST (Stanford Sentiment Treebank, 2013)

- *This was the worst restaurant I have ever had the misfortune of eating at.*
- *The restaurant was a bit slow in delivering their food, and they didn’t seem to be using the best ingredients.*
- *This restaurant is pretty decent— its food is acceptable considering the low prices.*
- *This is the best restaurant in the Western Hemisphere, and I will definitely be returning for another meal!*

Complex cases:
I do not hate this restaurant. (Negation)
I just love being served cold food! (Sarcasm)
The food is unnervingly unique. (Negative words being positive)

Data	Prev SotA	CNN-rand	CNN-static	CNN-nonstatic
MR	79.5	76.1	81.0	81.5
SST-1	48.7	45.0	45.5	48.0
SST-2	87.8	82.7	86.8	87.2
Subj	93.6	89.6	93.0	93.4
TREC	95.0	91.2	92.8	93.6
CR	82.7	79.8	84.7	84.3
MPQA	87.2	83.4	89.6	89.5

- Fine-tuning vectors helps, though not that much.
- Perhaps our embeddings are overfitting (given the relatively small training sample)?



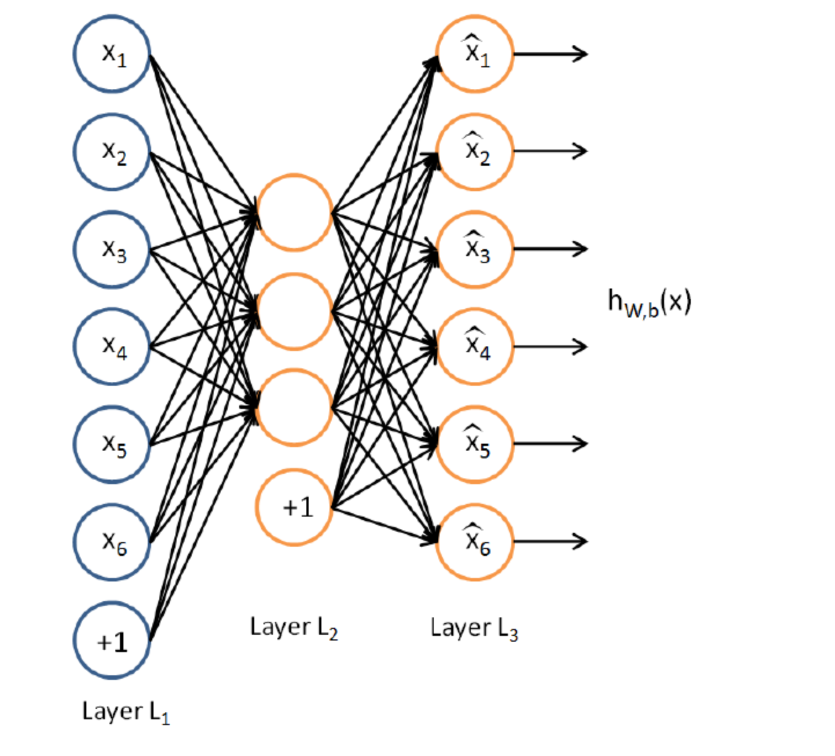
Data	Prev SotA	CNN-nonstatic	CNN-multichannel
MR	79.5	81.5	81.1
SST-1	48.7	48.0	47.4
SST-2	87.8	87.2	88.1
Subj	93.6	93.4	93.2
TREC	95.0	93.6	92.2
CR	82.7	84.3	85.0
MPQA	87.2	89.5	89.4

- Performance is not statistically different from CNN-nonstatic.

What’s next: autoencoders

- An autoencoder neural network is an unsupervised learning algorithm that applies backpropagation, setting the target values to be equal to the input itself , i.e., it uses

$$y(i) = x(i)$$



Autoencoders

- Suppose the inputs x are the pixel intensity values from a *10×10 image* (100 pixels) so *n* = 100, and
- there are *s2 = 50* hidden units in layer L2.
- Note that we also have $y \in R^{100}$.
- Since there are only 50 hidden units, the network is forced to learn a *compressed representation* of the input. I.e., given only the vector of hidden unit activations $a^{(2)} \in R^{50}$, it must try to reconstruct the 100-pixel input x.
- Compressed representation may be seen as *lower dimensional embeddings*
- *Images? Sentences? Longer Texts?*

Bibliography

- Y. LeCun, L. D. Jackel, B. Boser, J. S. Denker, H. P. Graf, I. Guyon, D. Henderson, R. E. Howard and W. Hubbard: Handwritten Digit Recognition: applications of Neural Net Chips and Automatic Learning, IEEE Communication, 41-46, invited paper, November 1989
- Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278{2324, 1998a.
- Bengio Yoshua. Learning Deep Architectures for AI. Foundations and Trends in Machine Learning 2 (1): 1–127.
- Deep Visual-Semantic Alignments for Generating Image Descriptions. Andrej Karpathy, Li Fei-Fei, CVPR 2015
- Y. Kim, Convolutional Neural Networks for Sentence Classification, Proc. of EMNLP, Doha, Qatar, 2014.
- Convolutional Neural Networks tutorial:
  - http://cs231n.github.io/ : stanford course on CNN for visual recognition with online (free) materials
  - http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/
  - Yann LeCun and Yoshua Bengio. 1998. Convolutional networks for images, speech, and time series. In The handbook of brain theory and neural networks, Michael A. Arbib (Ed.). MIT Press, Cambridge, MA, USA 255-258.

Resources

- Most of this slides are based on
  - https://cs.stanford.edu/~quocle/tutorial1.pdf
  - http://web.stanford.edu/class/cs294a/sparseAutoencoder_2011new.pdf
- Software packages
  - Tensorflow
  - Keras
  - Pytorch
- Other useful resources can be found on the course website