具身智能-02

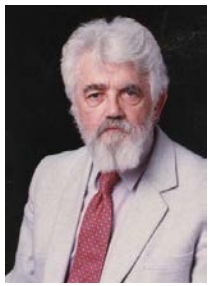
刘华平

2024年2月26日

背景

➢ AI的历史

Marvin Lee Minsky
(1927-2016)

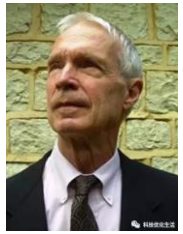


John McCarthy
(1927-2011)

达特茅斯会议
（1956）

John Haugeland
(1945-2010)

GOF AI



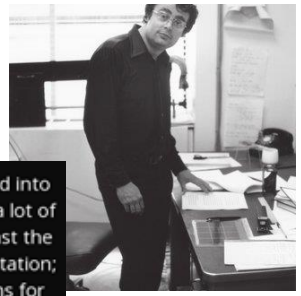
Hopfield
（1933-）

Rumelhart
（1942-2011）

Hilton
（1947-）

神经网络->反向传播->深度学习

- 模式识别
- 数据挖掘
- 机器学习



When David Marr at MIT moved into computer vision, he generated a lot of excitement, but he hit up against the problem of knowledge representation; he had no good representations for knowledge in his vision systems.

— Marvin Minsky —

David Marr
(1945-1980)

Pfeifer
(1947-)

Moravec
(1948-)

Brooks
(1954-)

Cangelosi
(1967-)

- 智能是具身化和情境化的，智能需要一个身体

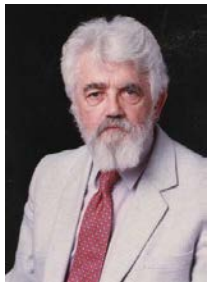
- 机器人学
- 机构学
- 形态智能

- 强化

背景

➢ 神经网络

Marvin Lee Minsky
(1927-2016)

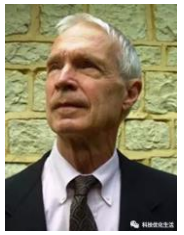


John McCarthy
(1927-2011)

达特茅斯会议
（1956）

John Haugeland
(1945-2010)

GOF AI



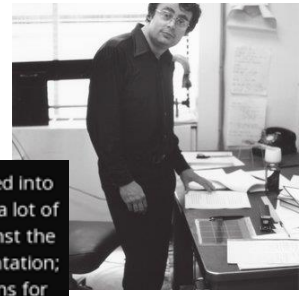
Hopfield
（1933-）

Rumelhart
（1942-2011）

Hilton
（1947-）

神经网络->反向传播->深度学习

- 数据挖掘
- 机器学习
- 模式识别



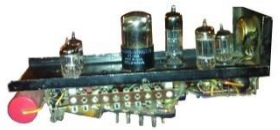
When David Marr at MIT moved into computer vision, he generated a lot of excitement, but he hit up against the problem of knowledge representation; he had no good representations for knowledge in his vision systems.

— Marvin Minsky —

David Marr
(1945-1980)

- 1950年Marvin Minsky和Dean Edmonds建造了第一台神经网络计算机SNARC，使用3000个真空管和自动指示装置模拟40个神经元构成的网络。

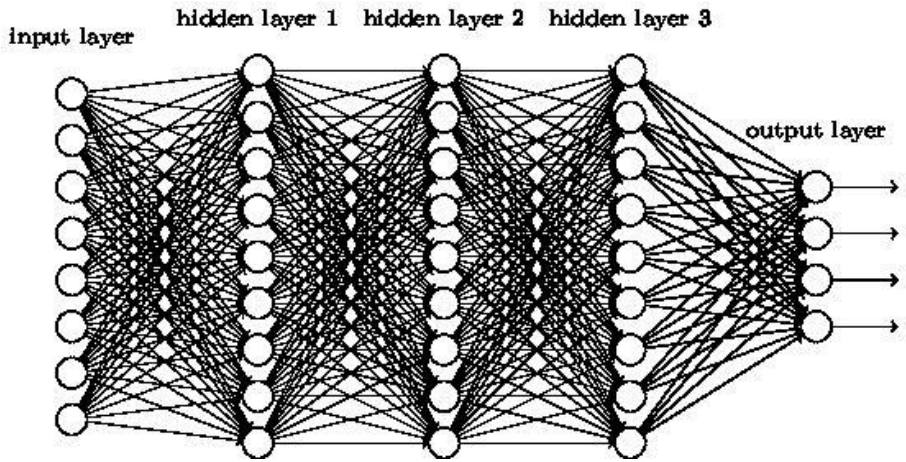
如果它现在不是，
那么总有一天会是

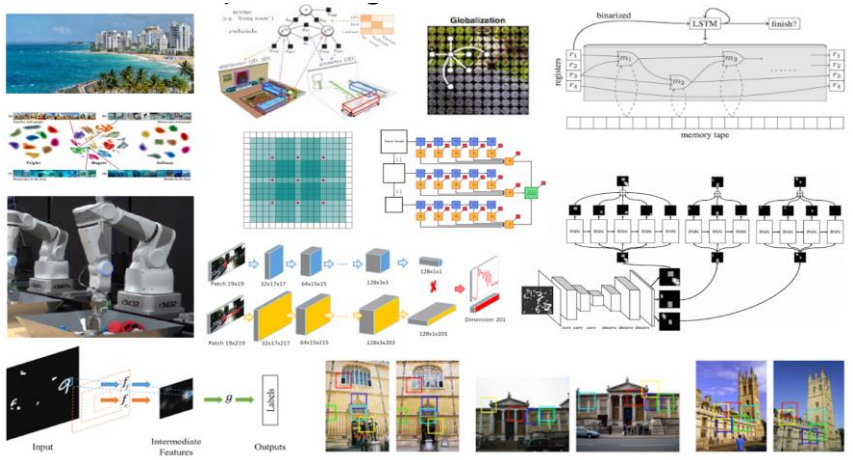


The Stochastic Neural Analog Reinforcement Calculator (SNARC) embodied a type of neural network that is designed to learn from experience and improve its performance through a process of trial and error (similar to reinforcement learning).

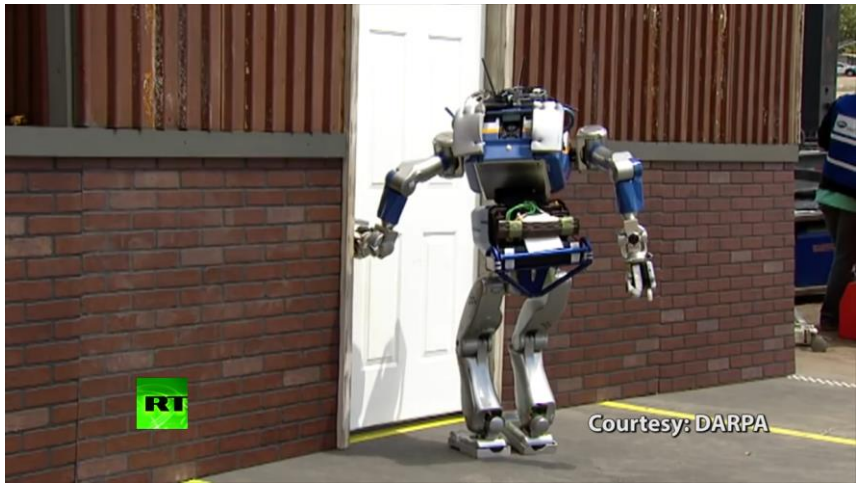
背景

➢ 神经网络

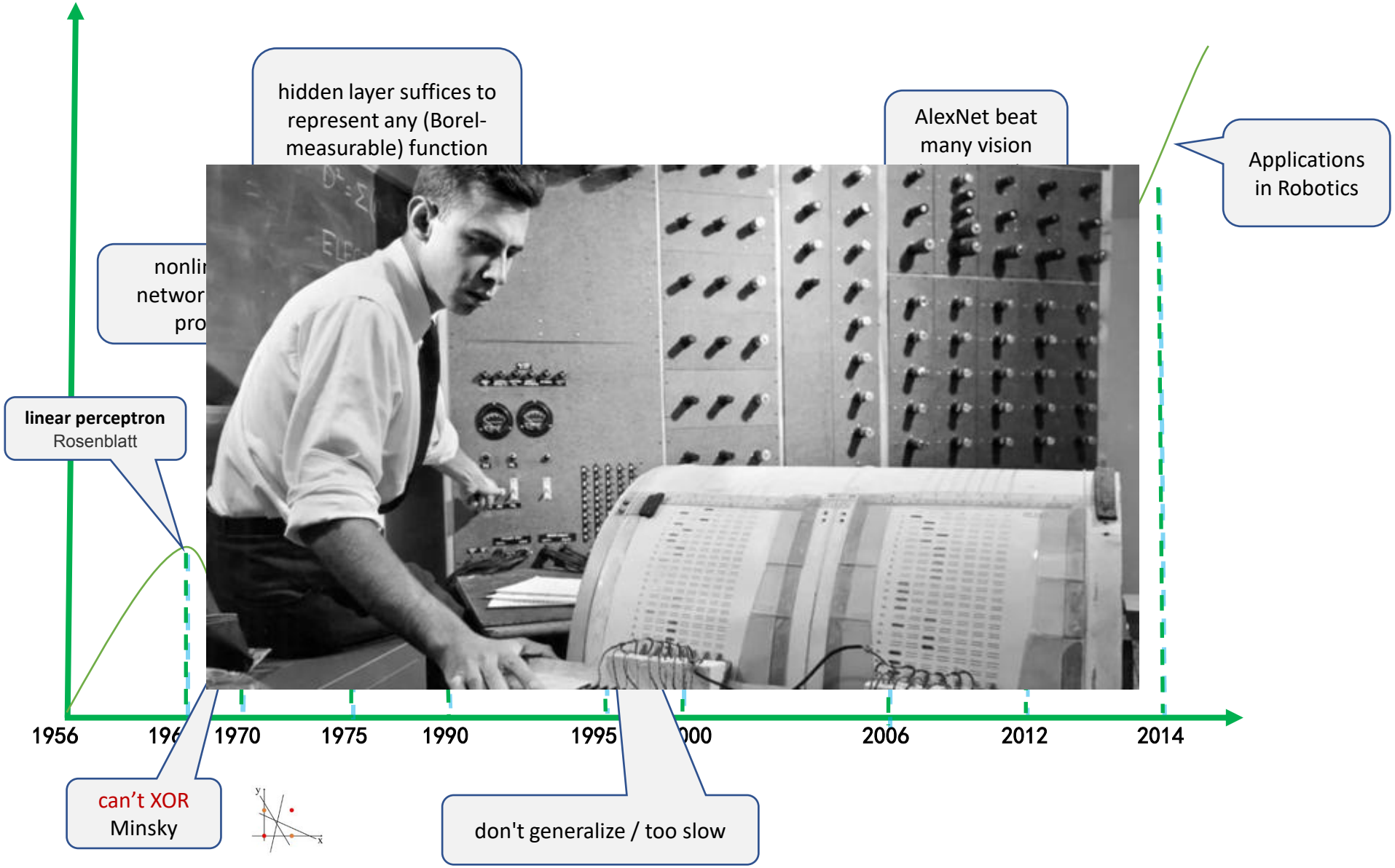

input layer
hidden layer 1
hidden layer 2
hidden layer 3
output layer


Globalization
binarized
LSTM
finish?
memory tape
Input
Intermediate Features
Outputs
Labels

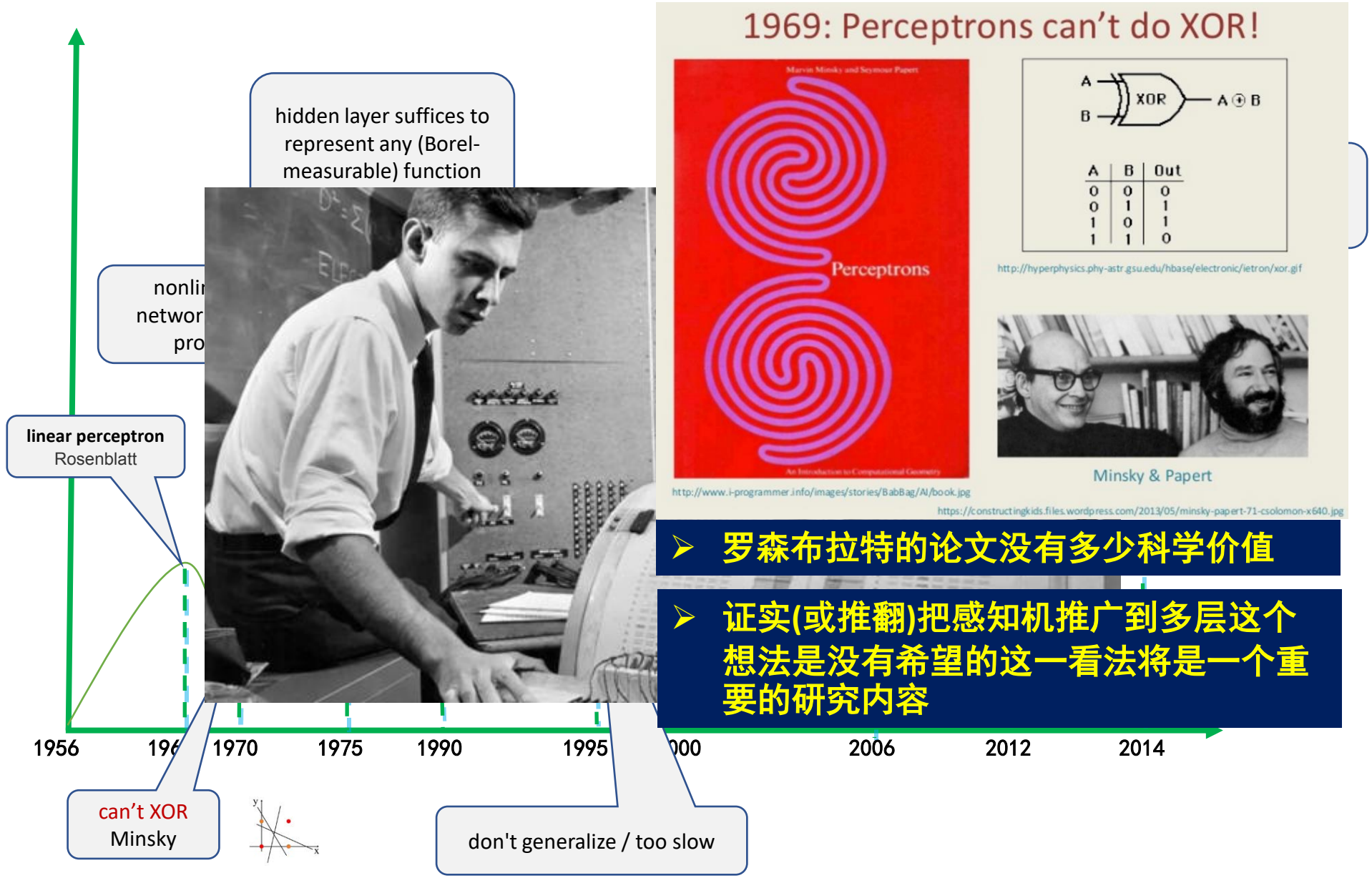

FAIRPLEX
FAIRPLEX
6:16:34 05/06/2015


RT
Courtesy: DARPA

神经网络

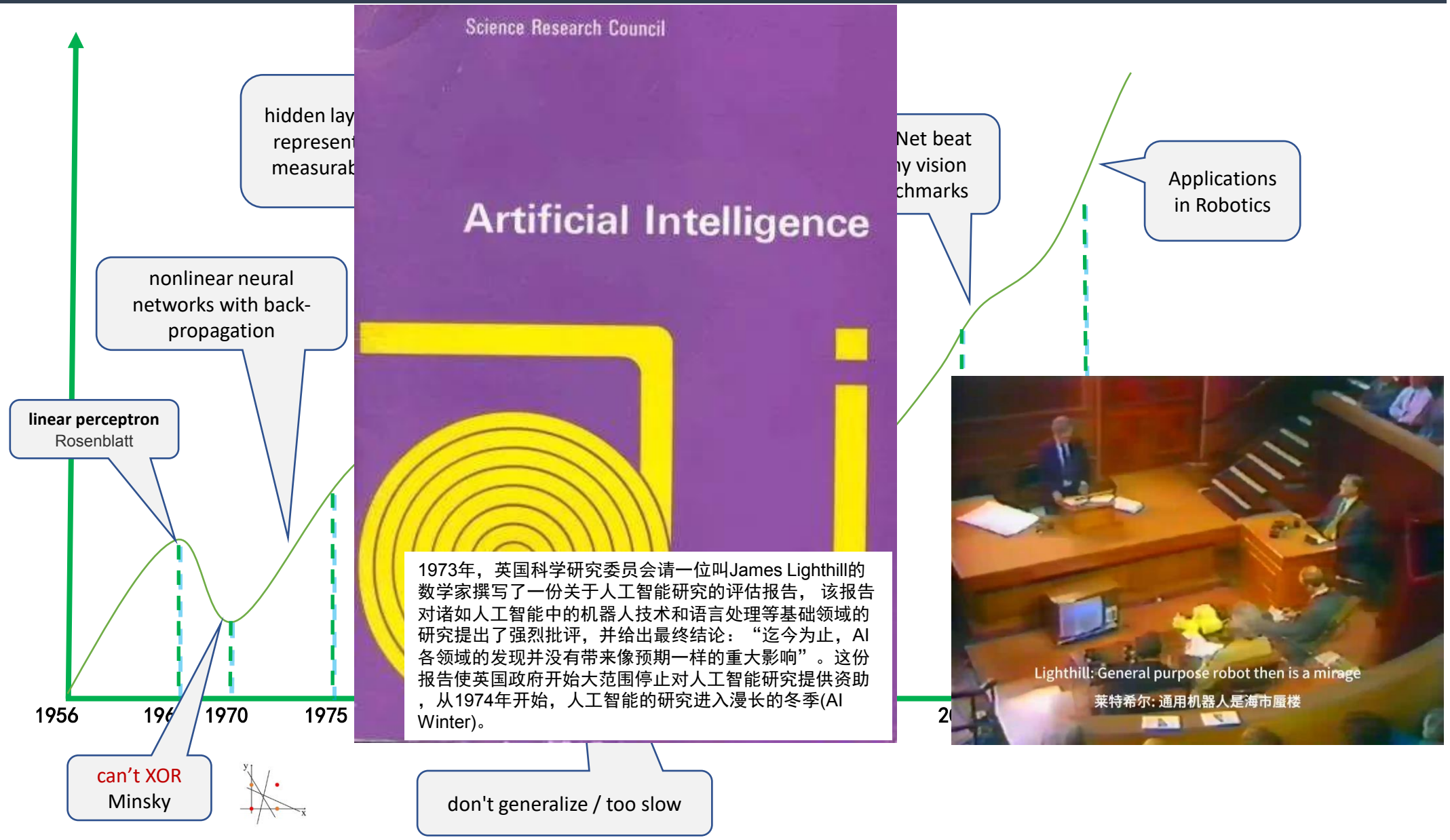

hidden layer suffices to represent any (Borel-measurable) function
AlexNet beat many vision
Applications in Robotics
nonli
networ
pro
linear perceptron
Rosenblatt
1956
196
1970
1975
1990
1995
000
2006
2012
2014
can’t XOR
Minsky
don't generalize / too slow

神经网络

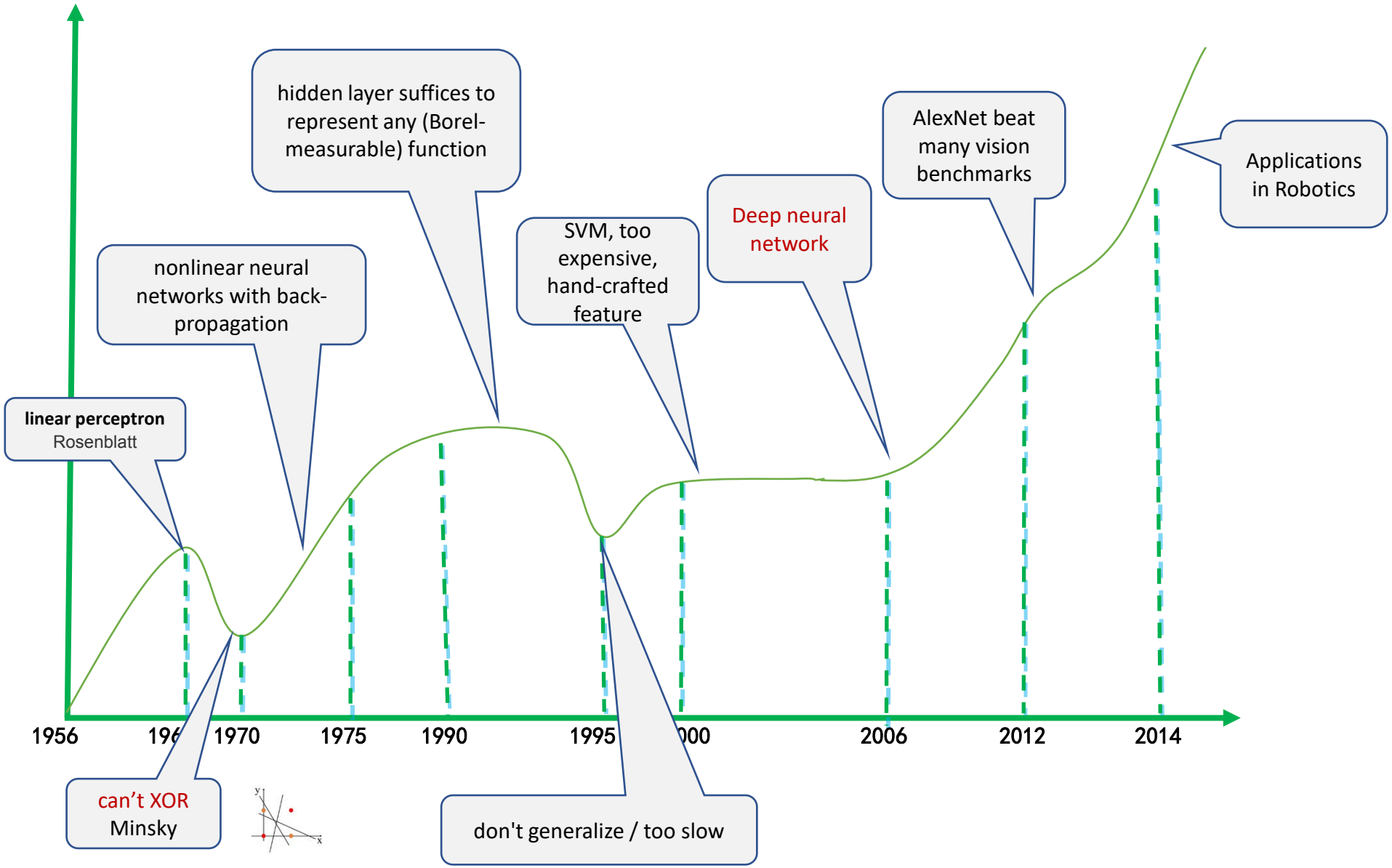


- 罗森布拉特的论文没有多少科学价值
- 证实(或推翻)把感知机推广到多层这个想法是没有希望的这一看法将是一个重要的研究内容

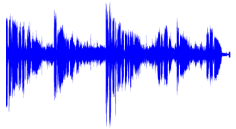
神经网络


hidden lay
represent
measurab
Net beat
y vision
chmarks
Applications
in Robotics
nonlinear neural
networks with back-
propagation
linear perceptron
Rosenblatt
1956
196
1970
1975
2
can't XOR
Minsky
don't generalize / too slow
Science Research Council
Artificial Intelligence
1973年，英国科学研究委员会请一位叫James Lighthill的数学家撰写了一份关于人工智能研究的评估报告， 该报告对诸如人工智能中的机器人技术和语言处理等基础领域的研究提出了强烈批评，并给出最终结论：“迄今为止，AI各领域的发现并没有带来像预期一样的重大影响”。这份报告使英国政府开始大范围停止对人工智能研究提供资助，从1974年开始，人工智能的研究进入漫长的冬季(AI Winter)。
Lighthill: General purpose robot then is a mirage
莱特希尔: 通用机器人是海市蜃楼

神经网络


hidden layer suffices to represent any (Borel-measurable) function
AlexNet beat many vision benchmarks
Applications in Robotics
nonlinear neural networks with back-propagation
SVM, too expensive, hand-crafted feature
Deep neural network
linear perceptron
Rosenblatt
1956
1970
1975
1990
1995
2006
2012
2014
can’t XOR
Minsky
don't generalize / too slow

2010: Speech Recognition



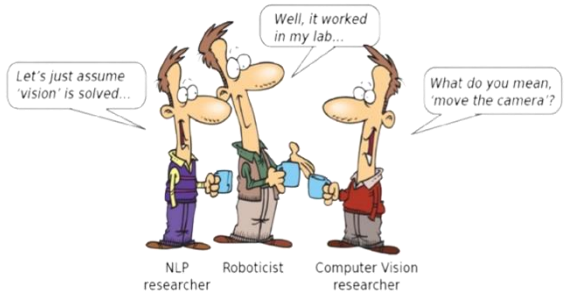
Audio ⇨ Acoustic Model ⇨ Phonetic Model ⇨ Language Model ⇨ Text

2012: Computer Vision

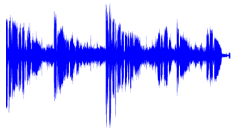
Pixels ⇨ Key Points ⇨ SIFT features ⇨ Deformable part model ⇨ Labels

2014: Machine Translation

Text ⇨ Reordering ⇨ Phrase Table/Dictionary ⇨ Language Model ⇨ Text



2010: Speech Recognition



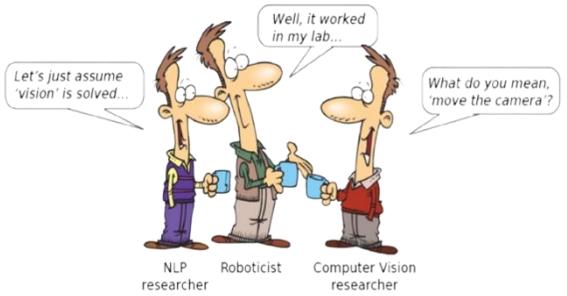
Audio → **Deep neural network** → Text

2012: Computer Vision

Pixels → **Deep neural network** → Labels

2014: Machine Translation

Text → **Deep neural network** → Text



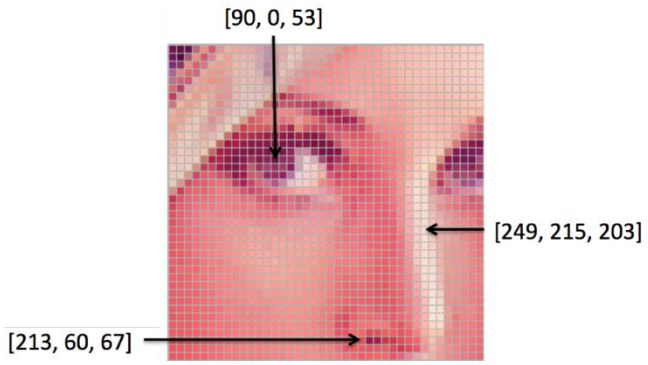
（1）一般介绍

（2）从多层感知机到卷积神经网络

（3）应用

一般介绍

➢ 数字图像处理——图像滤波器



$$g[n,m] = \begin{cases} 255, & f[n,m] > 100 \\ 0, & \text{otherwise.} \end{cases}$$

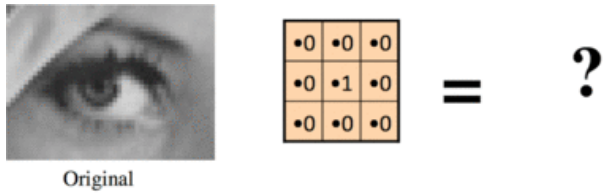
23	25	30	35	30
25	30	35	37	40
45	40	37	43	45
38	40	43	42	46
35	40	42	45	47

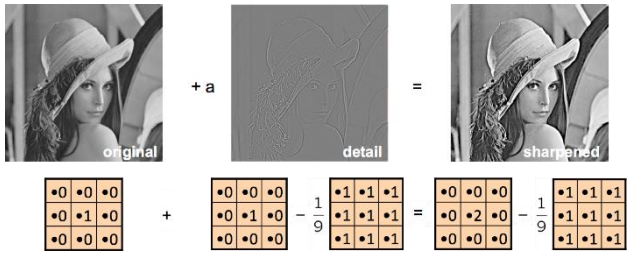
		39		

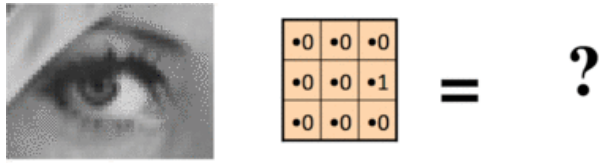
Value = (30 + 35+ 37+ 40 + 37+ 43+ 40+ 43+ 42) /9 = 38.55 = 39

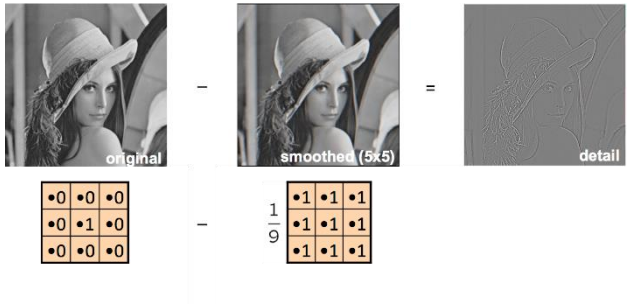
一般介绍

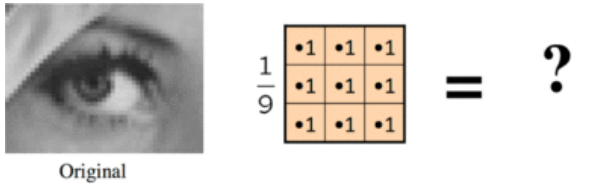
➢ 数字图像处理——图像滤波器





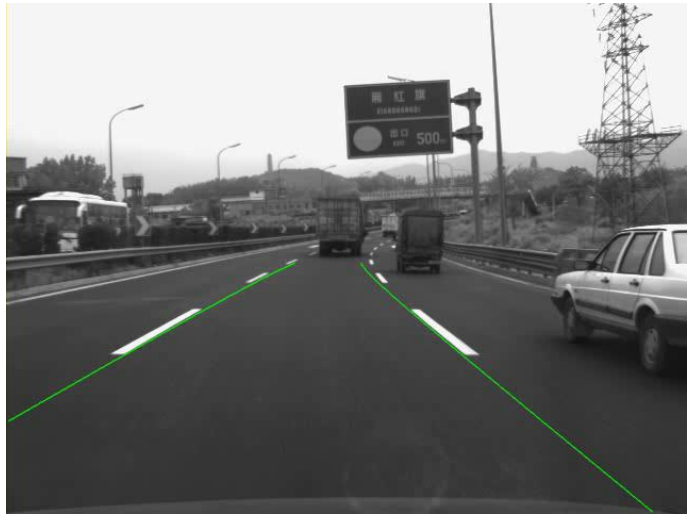


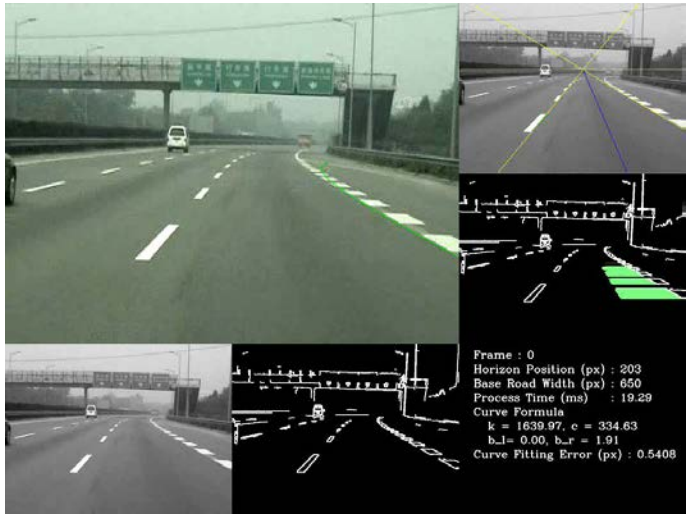


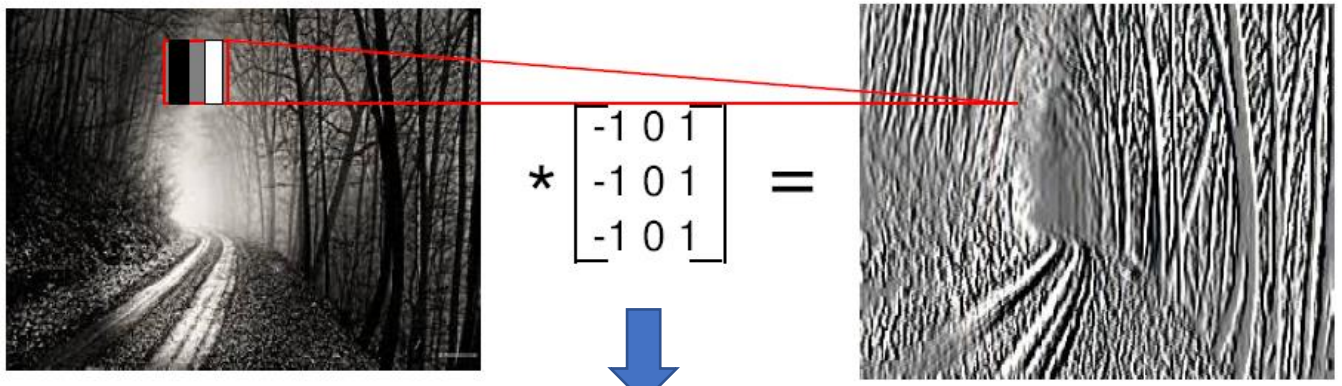


一般介绍

➢ 数字图像处理——图像滤波器



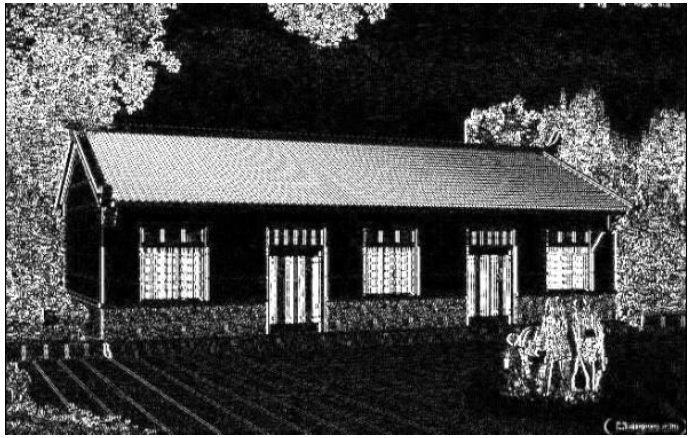


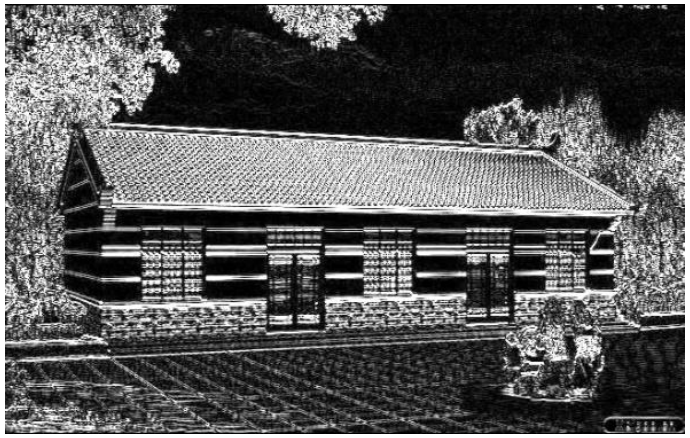


一般介绍

Sobel、Canny、Roberts、Prewitt，……

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * I \qquad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} * I$$





一般介绍

hand-crafted
Feature Extractor

"Simple" Trainable
Classifier

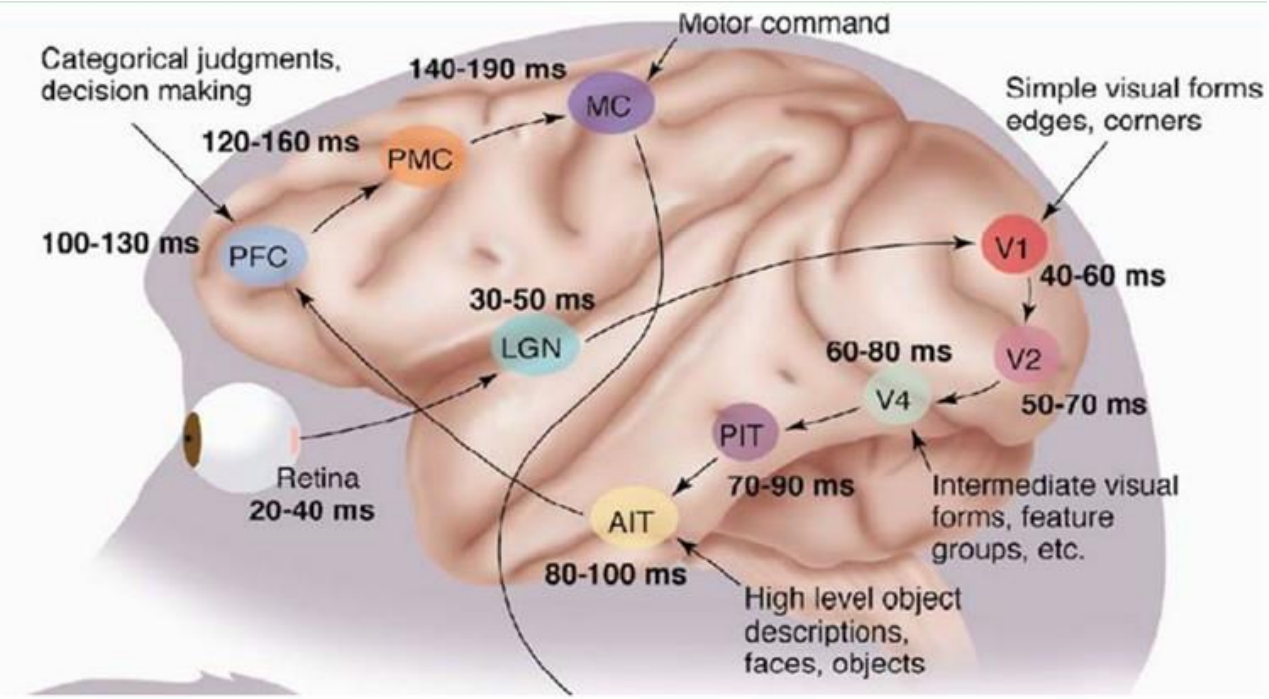
Trainable
Feature Extractor

Trainable
Classifier

Feature learning, representation learning

一般介绍

1981年，科学家们发现人的视觉系统的信息处理是分级的，人对物品的识别可能是一个不断迭代不断抽象的过程。

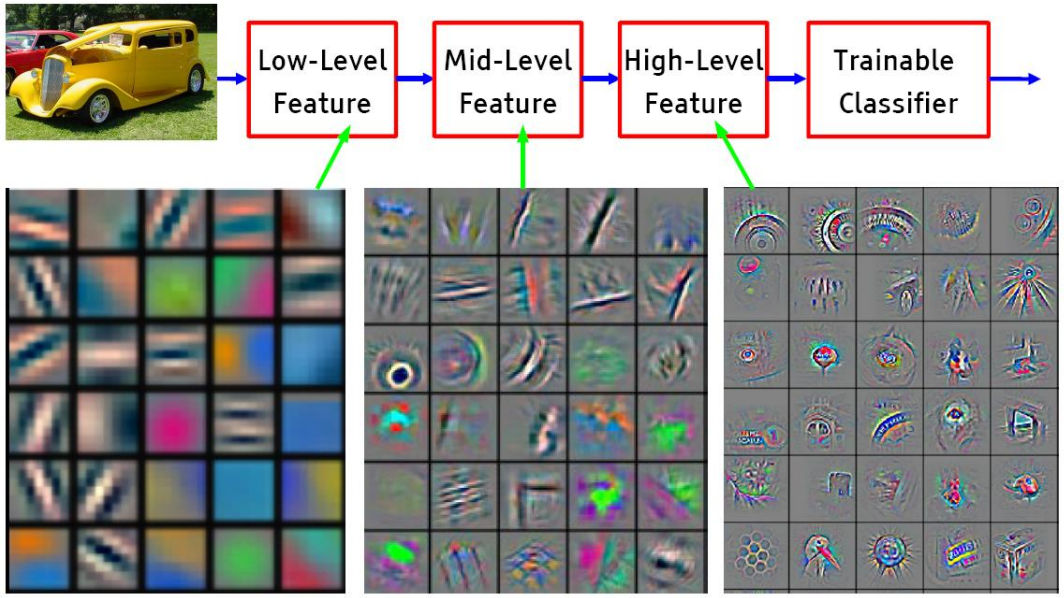


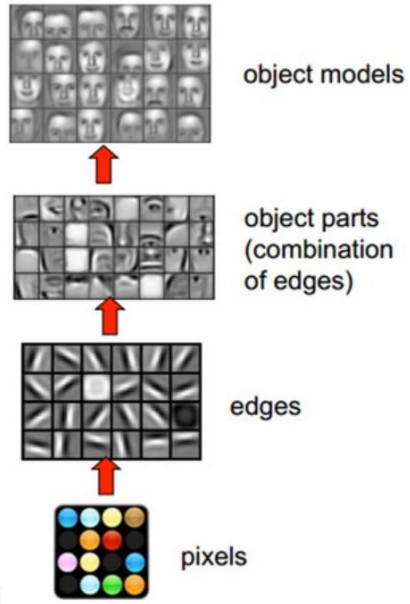
Retina - LGN - V1 - V2 - V4 - PIT - AIT ....

Lots of intermediate representations

一般介绍

Hierarchy of representations with increasing level of abstraction



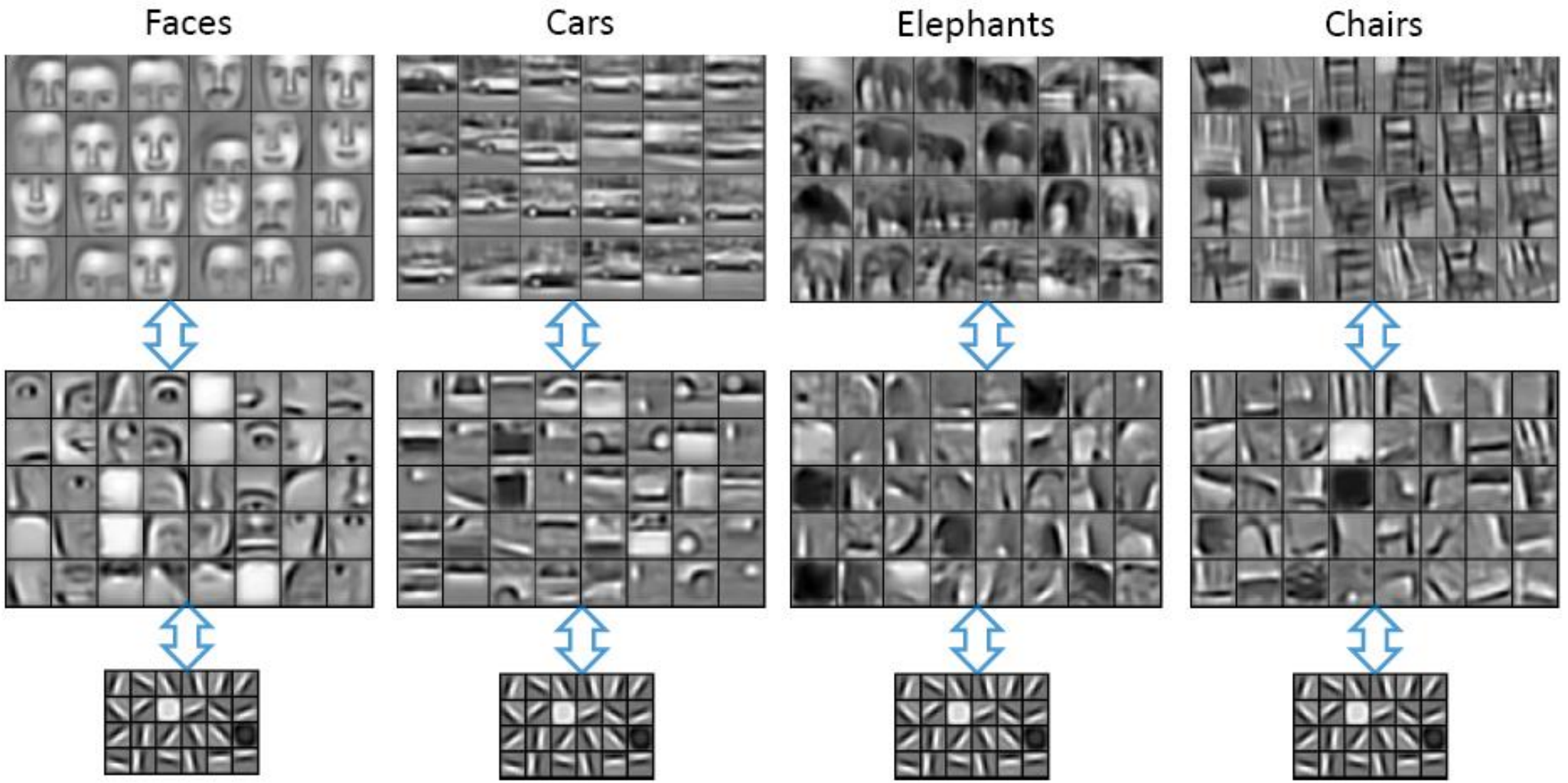


Pixel → edge → texton → motif → part → object

It's deep if it has more than one stage of non-linear feature transformation

Deep Learning addresses the problem of learning hierarchical representations with a single algorithm

一般介绍



一般介绍

“Solving a problem simply means representing it so as to make the solution transparent.”

Herbert A. Simon

ICLR 2020
8th International Conference on Learning Representations
Addis Ababa, Ethiopia
April 26-30, 2020

一般介绍

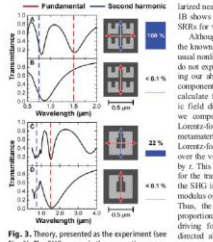


Fig. 3. Theory, presented as the experiment (see Fig. 1). The SHG source is the magnetic component of the Lorentz force on metal electrons in the SRRs.

The setup for measuring the SHG is described in the supporting online material (22). We expect that the SHG strongly depends on the resonance that is excited. Obviously, the incident polarization and the detuning of the laser wavelength from the resonance are of particular interest. One possibility for controlling the detuning is to change the laser wavelength for a given sample, which is difficult because of the extremely broad tuning range required. Thus, we follow an alternative route, lithographic tuning (in which the incident laser wavelength of 1.5 μm, as well as the detection system, remains fixed), and tune the resonance positions by changing the SRR size. In this manner, we can also guarantee that the optical properties of the SRR constituent materials are identical for all configurations. The blue bars in Fig. 1 summarize the measured SHG signals. For excitation of the *LC* resonance in Fig. 1A (horizontal incident polarization), we find an SHG signal that is 500 times above the noise level. As expected for SHG, this signal closely scales with the square of the incident power (Fig. 2A). The polarization of the SHG emission is nearly vertical (Fig. 2B). The small angle with respect to the vertical is due to deviations from perfect mirror symmetry of the SRRs (see electron micrographs in Fig. 1). Small detuning of the *LC* resonance toward smaller wavelength (i.e., to 1.3-μm wavelength) reduces the SHG signal strength from 100% to 20%. For excitation of the Mie resonance with vertical incident polarization in Fig. 1D, we find a small signal just above the noise level. For excitation of the Mie resonance with horizontal incident polarization in Fig. 1C, a small but significant SHG emission is found, which is again polarized nearly vertically. For completeness, Fig. 1B shows the off-resonant case for the smaller SRRs for vertical incident polarization.

Although these results are compatible with the known selection rules of surface SHG from usual nonlinear optics (*23*), these selection rules do not explain the mechanism of SHG. Following our above argumentation on the magnetic component of the Lorentz force, we numerically calculate first the linear electric and magnetic field distributions (*22*); from these fields, we compute the electron velocities and the Lorentz-force field (fig. S1). In the spirit of a metamaterial, the transverse component of the Lorentz-force field can be spatially averaged over the volume of the unit cell of size a by a by t. This procedure delivers the driving force for the transverse SHG polarization. As usual, the SHG intensity is proportional to the square modulus of the nonlinear electron displacement. Thus, the SHG strength is expected to be proportional to the square modulus of the driving force, and the SHG polarization is directed along the driving-force vector. Corresponding results are summarized in Fig. 3 in the same arrangement as Fig. 1 to allow for a direct comparison between experiment and theory. The agreement is generally good, both for linear optics and for SHG. In particular, we find a much larger SHG signal for excitation of those two resonances (Fig. 3, A and C), which are related to a finite magnetic-dipole moment (perpendicular to the SRR plane) as compared with the purely electric Mie resonance (Figs. 1D and 3D), despite the fact that its oscillator strength in the linear spectrum is comparable. The SHG polarization in the theory is strictly vertical for all resonances. Quantitative deviations between the SHG signal strengths of experiment and theory, respectively, are probably due to the simplified SRR shape assumed in our calculations and/or due to the simplicity of our modeling. A systematic microscopic theory of the nonlinear optical properties of metallic metamaterials would be highly desirable but is currently not available.

References and Notes

1. J. B. Pendry, A. J. Holden, D. J. Robbins, W. J. Stewart, *IEEE Trans. Microw. Theory Tech.* **47**, 2075 (1999).
2. J. B. Pendry, *Phys. Rev. Lett.* **85**, 3966 (2000).
3. R. A. Shelby, D. R. Smith, S. Schultz, *Science* **292**, 77 (2001).
4. T. J. Yen *et al.*, *Science* **303**, 1494 (2004).
5. S. Linden *et al.*, *Science* **306**, 1351 (2004).
6. C. Enkrich *et al.*, *Phys. Rev. Lett.* **95**, 203901 (2005).
7. A. N. Grigorenko *et al.*, *Nature* **438**, 335 (2005).
8. G. Dolling, M. Wegener, S. Linden, C. Hormann, *Opt. Express* **14**, 1842 (2006).
9. G. Dolling, C. Enkrich, M. Wegener, C. M. Soukoulis, S. Linden, *Science* **312**, 892 (2006).
10. J. B. Pendry, D. Schurig, D. R. Smith, *Science* **312**, 1780; published online 25 May 2006.
11. U. Leonhardt, *Science* **312**, 1777 (2006); published online 25 May 2006.
12. M. W. Klein, C. Enkrich, M. Wegener, C. M. Soukoulis, S. Linden, *Opt. Lett.* **31**, 1259 (2006).
13. W. J. Padilla, A. J. Taylor, C. Highstrete, M. Lee, R. D. Averitt, *Phys. Rev. Lett.* **96**, 107401 (2006).
14. D. R. Smith, S. Schultz, P. Markos, C. M. Soukoulis, *Phys. Rev. B* **65**, 195104 (2002).
15. S. O'Brien, D. McPeake, S. A. Ramakrishna, J. B. Pendry, *Phys. Rev. B* **69**, 241101 (2004).
16. J. Zhou *et al.*, *Phys. Rev. Lett.* **95**, 223902 (2005).
17. A. K. Popov, V. M. Shalaev, available at http://arxiv.org/abs/physics/0603255 (2006).
18. V. G. Veselago, *Sov. Phys. Usp.* **10**, 509 (1968).
19. M. Wegener, *Extreme Nonlinear Optics* (Springer, Berlin, 2005).
20. H. W. Barlow, *Nature* **173**, 41 (1954).
21. S. Y. Chen, M. Maksimchuk, D. Umstadter, *Nature* **396**, 653 (1998).
22. Materials and Methods are available as supporting material on Science Online.
23. F. Guyot-Sionnest, W. Chen, Y. R. Shen, *Phys. Rev. B* **33**, 8254 (1986).
24. We thank the groups of S. W. Koch, J. V. Moloney, and C. M. Soukoulis for discussions. The research of M.W. is supported by the Leibniz award 2000 of the Deutsche Forschungsgemeinschaft (DFG), that of S.L. through a Helmholtz-Hochschul-Nachwuchsgruppe (VH-NG-232).

Supporting Online Material
www.sciencemag.org/cgi/content/full/1127568/DC1
Materials and Methods
Figs. S1 and S2
References

26 April 2006; accepted 22 June 2006
10.1126/science.1127198

Reducing the Dimensionality of Data with Neural Networks

G. E. Hinton* and R. R. Salakhutdinov

Department of Computer Science, University of Toronto, 6 King's College Road, Toronto, Ontario M5S 3G4, Canada.
*To whom correspondence should be addressed. E-mail: hinton@cs.toronto.edu

High-dimensional data can be converted to low-dimensional codes by training a multilayer neural network with a small central layer to reconstruct high-dimensional input vectors. Gradient descent can be used for fine-tuning the weights in such "autoencoder" networks, but this works well only if the initial weights are close to a good solution. We describe an effective way of initializing the weights that allows deep autoencoder networks to learn low-dimensional codes that work much better than principal components analysis as a tool to reduce the dimensionality of data.

Dimensionality reduction facilitates the classification, visualization, communication, and storage of high-dimensional data. A simple and widely used method is principal components analysis (PCA), which finds the directions of greatest variance in the data set and represents each data point by its coordinates along each of these directions. We describe a nonlinear generalization of PCA that uses an adaptive, multilayer "encoder" network to transform the high-dimensional data into a low-dimensional code and a similar "decoder" network to recover the data from the code.

Starting with random weights in the two networks, they can be trained together by minimizing the discrepancy between the original data and its reconstruction. The required gradients are easily obtained by using the chain rule to backpropagate error derivatives first through the decoder network and then through the encoder network (*1*). The whole system is called an "autoencoder" and is depicted in Fig. 1.

It is difficult to optimize the weights in nonlinear autoencoders that have multiple hidden layers (*2*–*4*). With large initial weights, autoencoders typically find poor local minima; with small initial weights, the gradients in the early layers are tiny, making it infeasible to train autoencoders with many hidden layers. If the initial weights are close to a good solution, gradient descent works well, but finding such initial weights requires a very different type of algorithm that learns one layer of features at a time. We introduce this "pretraining" procedure for binary data, generalize it to real-valued data, and show that it works well for a variety of data sets.

An ensemble of binary vectors (e.g., images) can be modeled using a two-layer network called a "restricted Boltzmann machine" (RBM) (*5*, *6*) in which stochastic, binary pixels are connected to stochastic, binary feature detectors using symmetrically weighted connections. The pixels correspond to "visible" units of the RBM because their states are observed; the feature detectors correspond to "hidden" units. A joint configuration ($\mathbf{v}$, $\mathbf{h}$) of the visible and hidden units has an energy (*7*) given by

$$E(\mathbf{v},\mathbf{h}) = -\sum_{i \in \text{pixels}} b_i v_i - \sum_{j \in \text{features}} b_j h_j - \sum_{i,j} v_i h_j w_{ij} \quad (1)$$

where v_i and h_j are the binary states of pixel i and feature j, b_i and b_j are their biases, and w_{ij} is the weight between them. The network assigns a probability to every possible image via this energy function, as explained in (*8*). The probability of a training image can be raised by adjusting the weights and biases to lower the energy of that image and to raise the energy of similar, "confabulated" images that the network would prefer to the real data. Given a training image, the binary state h_j of each feature detector j is set to 1 with probability $\sigma(b_j + \sum_i v_i w_{ij})$, where $\sigma(x)$ is the logistic function $1/(1 + \exp(-x))$, b_j is the bias of j, v_i is the state of pixel i, and w_{ij} is the weight between i and j. Once binary states have been chosen for the hidden units, a "confabulation" is produced by setting each v_i to 1 with probability $\sigma(b_i + \sum_j h_j w_{ij})$, where b_i is the bias of i. The states of the hidden units are then updated once more so that they represent features of the confabulation. The change in a weight is given by

$$\Delta w_{ij} = \varepsilon\left(\langle v_i h_j \rangle_{\text{data}} - \langle v_i h_j \rangle_{\text{recon}}\right) \quad (2)$$

where ε is a learning rate, $\langle v_i h_j \rangle_{\text{data}}$ is the fraction of times that the pixel i and feature detector j are on together when the feature detectors are being driven by data, and $\langle v_i h_j \rangle_{\text{recon}}$ is the corresponding fraction for confabulations. A simplified version of the same learning rule is used for the biases. The learning works well even though it is not exactly following the gradient of the log probability of the training data (*6*).

A single layer of binary features is not the best way to model the structure in a set of images. After learning one layer of feature detectors, we can treat their activities—when they are being driven by the data—as data for learning a second layer of features. The first layer of feature detectors then become the visible units for learning the next RBM. This layer-by-layer learning can be repeated as many times as desired. It can be shown that adding an extra layer always improves a lower bound on the log probability that the model assigns to the training data, provided the number of feature detectors per layer does not decrease and their weights are initialized correctly (*8*). This bound does not apply when the higher layers have fewer feature detectors, but the layer-by-layer learning algorithm is nonetheless a very effective way to pretrain the weights of a deep autoencoder. Each layer of features captures strong, high-order correlations between the activities of units in the layer below. For a wide variety of data sets, this is an efficient way to progressively reveal low-dimensional, nonlinear structure.

After pretraining multiple layers of feature detectors, the model is "unfolded" (Fig. 1) to produce encoder and decoder networks that initially use the same weights. The global fine-tuning stage then replaces stochastic activities by deterministic, real-valued probabilities and uses backpropagation through the whole autoencoder to fine-tune the weights for optimal reconstruction.

For continuous data, the hidden units of the first-level RBM remain binary, but the visible units are replaced by linear units with Gaussian noise (*10*). If this noise has unit variance, the stochastic update rule for the hidden units remains the same and the update rule for visible unit i is to sample from a Gaussian with unit variance and mean $b_i + \sum_j h_j w_{ij}$.

In all our experiments, the visible units of every RBM had real-valued activities, which were in the range [0, 1] for logistic units. While training higher level RBMs, the visible units were set to the activation probabilities of the hidden units in the previous RBM, but the hidden units of every RBM except the top one had stochastic binary values. The hidden units of the top RBM had stochastic real-valued states drawn from a unit variance Gaussian whose mean was determined by the input from that RBM's logistic visible units. This allowed the low-dimensional codes to make good use of continuous variables and facilitated comparisons with PCA. Details of the pretraining and fine-tuning can be found in (*8*).

To demonstrate that our pretraining algorithm allows us to fine-tune deep networks efficiently, we trained a very deep autoencoder on a synthetic data set containing images of "curves" that were generated from three randomly chosen points in two dimensions (*8*). For this data set, the true intrinsic dimensionality is known, and the relationship between the pixel intensities and the six numbers used to generate them is highly nonlinear. The pixel intensities lie between 0 and 1 and are very non-Gaussian, so we used logistic output units in the autoencoder, and the fine-tuning stage of the learning minimized the cross-entropy error $-\sum_i p_i \log \hat{p}_i - \sum_i (1 - p_i) \log(1 - \hat{p}_i)$, where p_i is the intensity of pixel i and $\hat{p}_i$ is the intensity of its reconstruction.

The autoencoder consisted of an encoder with layers of size (28 × 28)-400-200-100-50-25-6 and a symmetric decoder. The six units in the code layer were linear and all the other units were logistic. The network was trained on 20,000 images and tested on 10,000 new images. The autoencoder discovered how to convert each 784-pixel image into six real numbers that allow almost perfect reconstruction (Fig. 2A). PCA gave much worse reconstructions. Without pretraining, the very deep autoencoder always reconstructs the average of the training data, even after prolonged fine-tuning (*8*). Shallower autoencoders with a single hidden layer between the data and the code can learn without pretraining, but pretraining greatly reduces their total training time (*8*). When the number of parameters is the same, deep autoencoders can produce lower reconstruction errors on test data than shallow ones, but this advantage disappears as the number of parameters increases (*8*).

Next, we used a 784-1000-500-250-30 autoencoder to extract codes for all the handwritten digits in the MNIST training set (*11*). The Matlab code that we used for the pretraining and fine-tuning is available in (*8*). Again, all units were logistic except for the 30 linear units in the code layer. After fine-tuning on all 60,000 training images, the autoencoder was tested on 10,000 new images and produced much better reconstructions than did PCA (Fig. 2B). A two-dimensional autoencoder produced a better visualization of the data than did the first two principal components (Fig. 2).

We also used a 625-2000-1000-500-30 autoencoder with linear input units to discover 30-dimensional codes for grayscale image patches that were derived from the Olivetti face data set (*12*). The autoencoder clearly outperformed PCA (Fig. 2C).

When trained on documents, autoencoders produce codes that allow fast retrieval. We represented each of 804,414 newswire stories (*13*) as a vector of document-specific probabilities of the 2000 commonest word stems, and we trained a 2000-500-250-125-10 autoencoder on half of the stories with the use of the multiclass cross-entropy error function $[-\sum_i p_i \log \hat{p}_i]$ for the fine-tuning. The 10 code units were linear and the remaining hidden units were logistic. When the cosine of the angle between two codes was used to measure similarity, the autoencoder clearly outperformed latent semantic analysis (LSA) (*14*), a well-known document retrieval method based on PCA (Fig. 4). Autoencoders (*8*) also outperform local linear embedding, a recent nonlinear dimensionality reduction algorithm (*15*).

Layer-by-layer pretraining can also be used for classification and regression. On a widely used version of the MNIST handwritten digit recognition task, the best reported error rates are 1.0% for randomly initialized backpropagation and 1.4% for support vector machines. After layer-by-layer pretraining in a 784-500-500-2000-10 network, backpropagation using steepest descent and a small learning rate achieves 1.2% (*8*). Pretraining helps generalization because it ensures that most of the information in the weights comes from modeling the images. The very limited information in the labels is used only to slightly adjust the weights found by pretraining.

It has been obvious since the 1980s that backpropagation through deep autoencoders would be very effective for nonlinear dimensionality reduction, provided that computers were fast enough, data sets were big enough, and the initial weights were close enough to a good solution. All three conditions are now satisfied. Unlike nonparametric methods (*15*, *16*), autoencoders give mappings in both directions between the data and code spaces, and they can be applied to very large data sets because both the pretraining and the fine-tuning scale linearly in time and space with the number of training cases.

References and Notes

1. D. C. Plaut, G. E. Hinton, *Comput. Speech Lang.* **2**, 35 (1987).
2. D. DeMers, G. Cottrell, *Advances in Neural Information Processing Systems 5* (Morgan Kaufmann, San Mateo, CA, 1993), pp. 580–587.
3. R. L. Hecht-Nielsen, *Science* **269**, 1860 (1995).
4. N. Kambhatla, T. Leen, *Neural Comput.* **9**, 1493 (1997).
5. P. Smolensky, *Parallel Distributed Processing: Volume 1: Foundations*, D. E. Rumelhart, J. L. McClelland, Eds. (MIT Press, Cambridge, 1986), pp. 194–281.
6. G. E. Hinton, *Neural Comput.* **14**, 1711 (2002).
7. J. J. Hopfield, *Proc. Natl. Acad. Sci. U.S.A.* **79**, 2554 (1982).
8. See supporting material on Science Online.
9. G. E. Hinton, S. Osindero, Y. W. Teh, *Neural Comput.* **18**, 1527 (2006).
10. M. Welling, M. Rosen-Zvi, G. Hinton, *Advances in Neural Information Processing Systems 17* (MIT Press, Cambridge, MA, 2005), pp. 1481–1488.
11. The MNIST data set is available at http://yann.lecun.com/exdb/mnist/index.html.
12. The Olivetti face data set is available at www.cs.toronto.edu/~roweis/data.html.
13. The Reuter Corpus Volume 2 is available at http://trec.nist.gov/data/reuters/reuters.html.
14. S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, R. A. Harshman, *J. Am. Soc. Inf. Sci.* **41**, 391 (1990).
15. S. T. Roweis, L. K. Saul, *Science* **290**, 2323 (2000).
16. J. A. Tenenbaum, V. J. de Silva, J. C. Langford, *Science* **290**, 2319 (2000).
17. We thank D. Rumelhart, M. Welling, S. Osindero, and S. Roweis for helpful discussions, and the Natural Sciences and Engineering Research Council of Canada for funding. G.E.H. is a fellow of the Canadian Institute for Advanced Research.

Supporting Online Material
www.sciencemag.org/cgi/content/full/1127568/DC1
Materials and Methods
Figs. S1 to S5
MatLab Code

20 March 2006; accepted 1 June 2006
10.1126/science.1127647

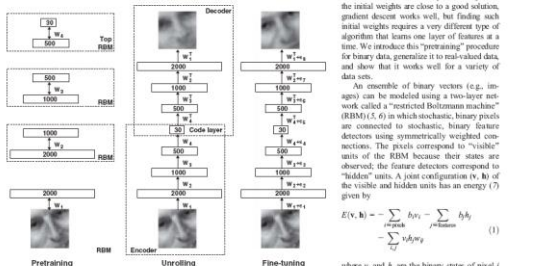


Fig. 1. Pretraining consists of learning a stack of restricted Boltzmann machines (RBMs), each having only one layer of feature detectors. The learned feature activations of one RBM are used as the "data" for training the next RBM in the stack. After the pretraining, the RBMs are "unrolled" to create a deep autoencoder, which is then fine-tuned using backpropagation of error derivatives.

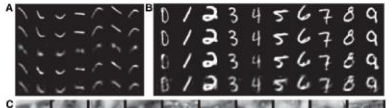


Fig. 2. (**A**) Top to bottom: Random samples of curves from the test data set; reconstructions produced by the six-dimensional deep autoencoder; reconstructions by "logistic PCA" (*8*) using six components; reconstructions by logistic PCA and standard PCA using 18 components. The average squared error per image for the last four rows is 1.44, 7.64, 2.45, 5.90. (**B**) Top to bottom: A random test image from each class; reconstructions by the 30-dimensional autoencoder; reconstructions by 30-dimensional logistic PCA and standard PCA. The average squared errors for the last three rows are 3.00, 8.01, and 13.87. (**C**) Top to bottom: Random samples from the test data set; reconstructions by the 30-dimensional autoencoder; reconstructions by 30-dimensional PCA. The average squared errors are 126 and 135.

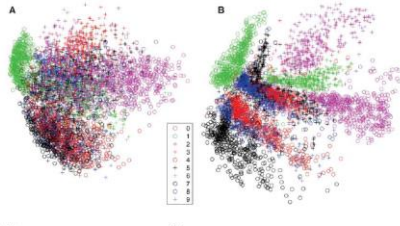


Fig. 3. (**A**) The two-dimensional codes for 500 digits of each class produced by taking the first two principal components of all 60,000 training images. (**B**) The two-dimensional codes found by a 784-1000-500-250-2 autoencoder. For an alternative visualization, see (*8*).

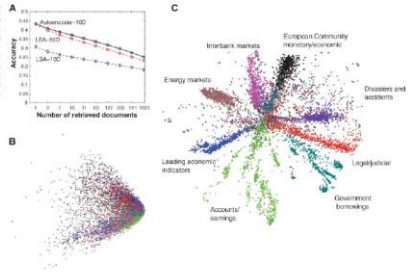


Fig. 4. (**A**) The fraction of retrieved documents in the same class as the query when a query document from the test set is used to retrieve other test set documents, averaged over all 402,207 possible queries. (**B**) The codes produced by two-dimensional LSA. (**C**) The codes produced by a 2000-500-250-125-2 autoencoder.

一般介绍

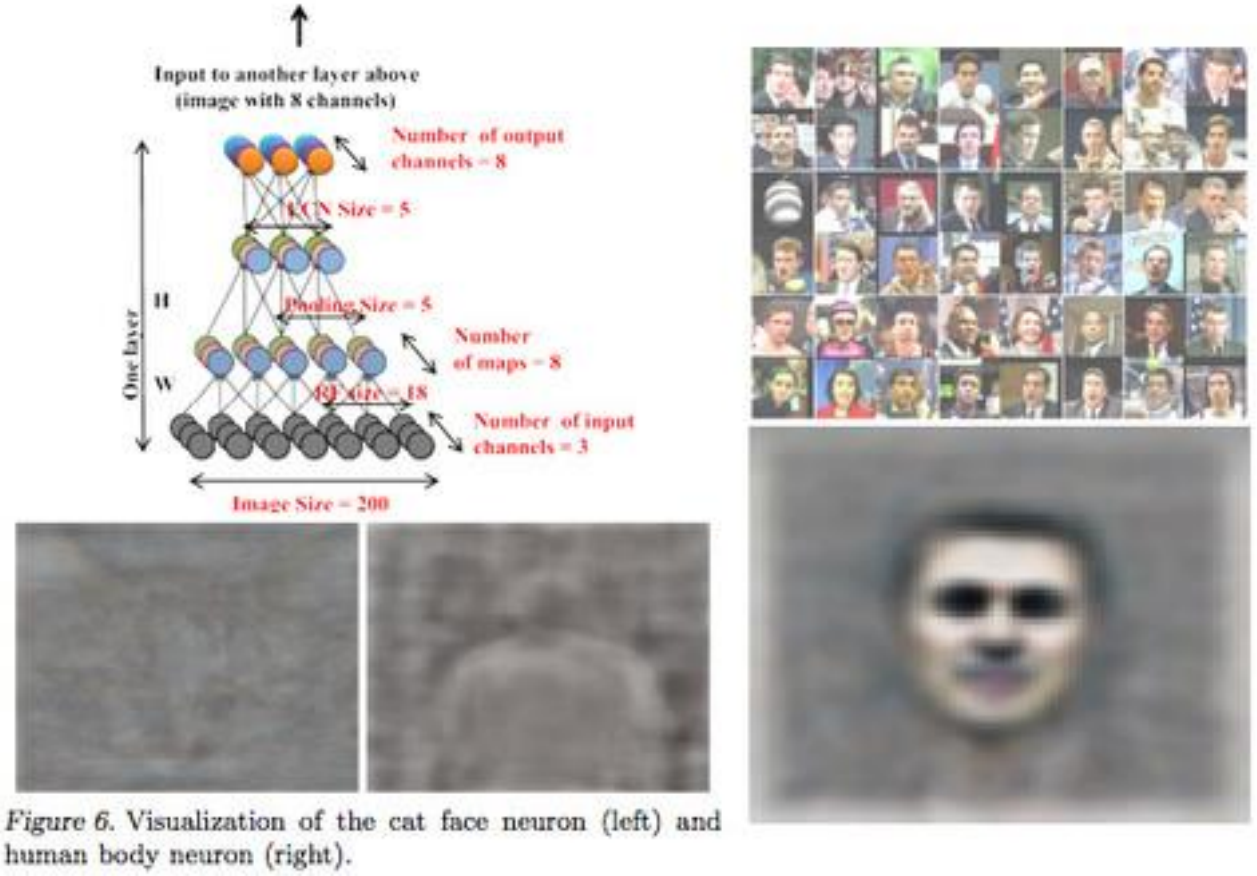


Figure 6. Visualization of the cat face neuron (left) and human body neuron (right).

1000台机器，一共16000个Core训练了三天，参数的个数是1000000000个

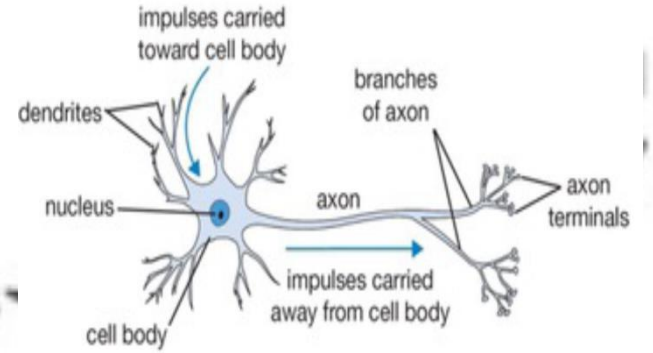
（1）一般介绍

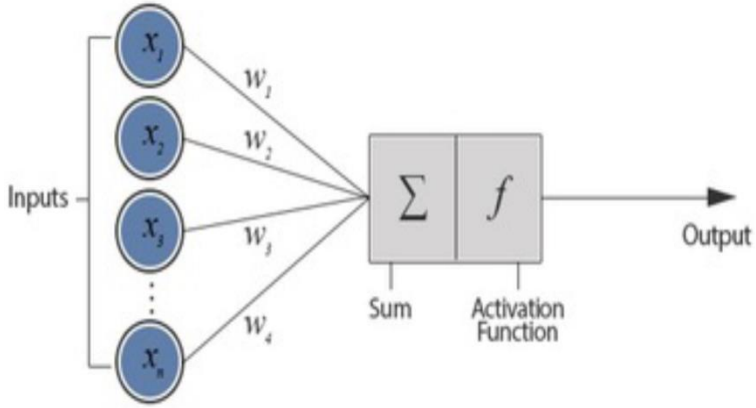
（2）从多层感知机到卷积神经网络

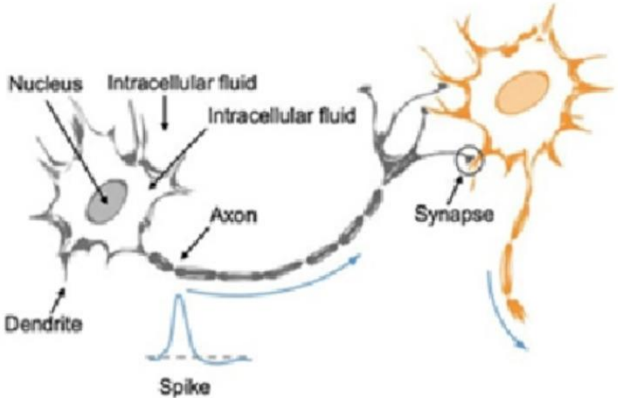
（3）应用

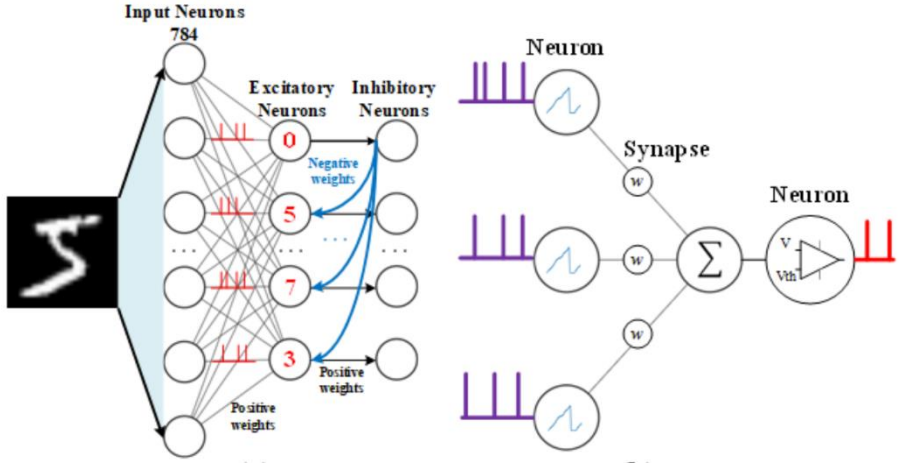
从MLP到CNN

Biological Neuron versus Artifcial Neural Network





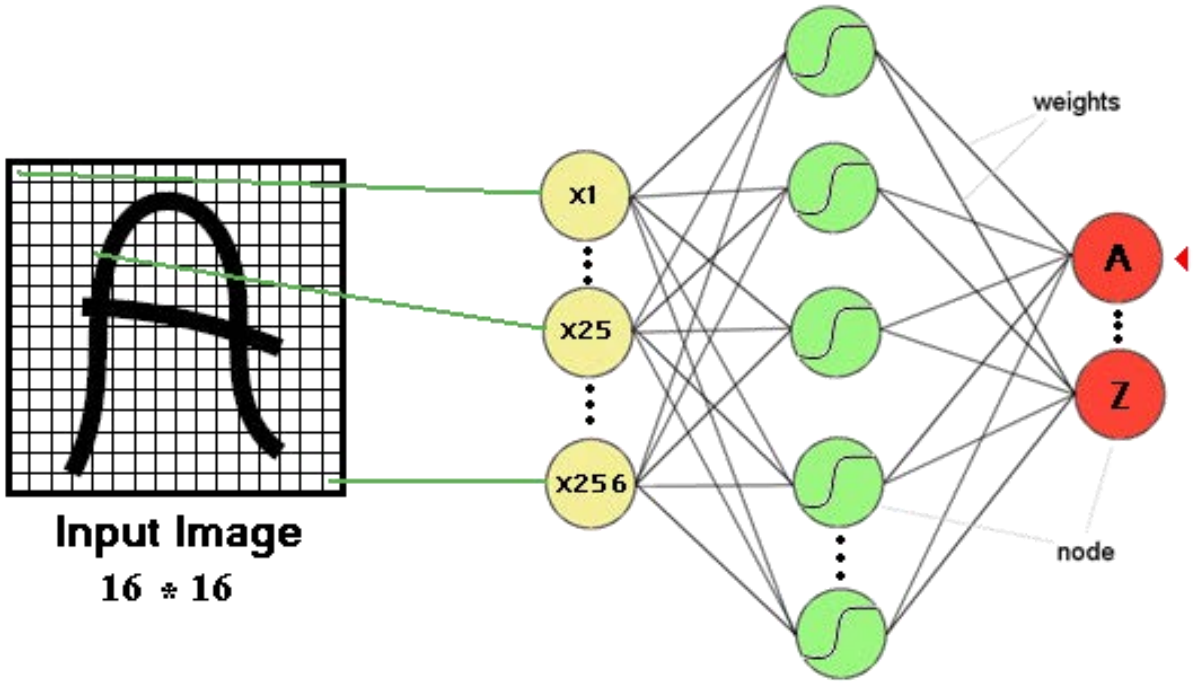




从MLP到CNN

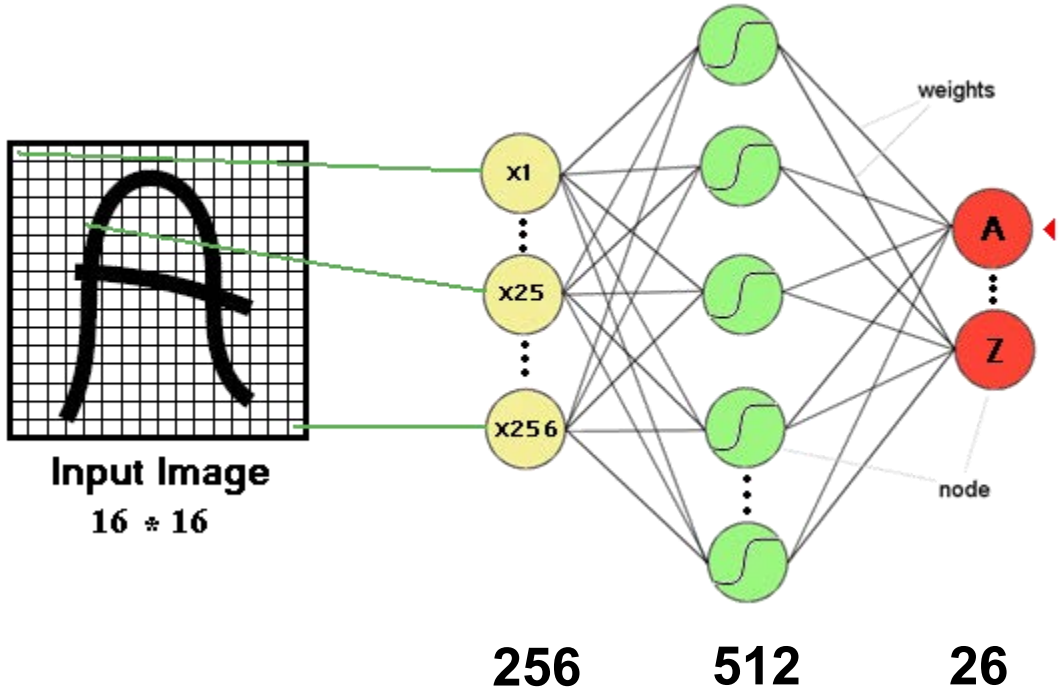
- 多个隐层
- Sigmoid激励函数

$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$



从MLP到CNN

- 256*512+512*26
- 大量的训练参数



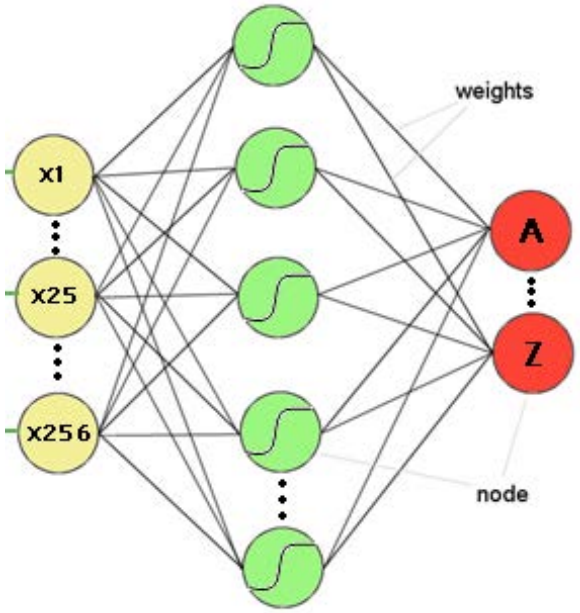
Backpropagation (BP) algorithm

- 误差 $e_k(n) = T_k(n) - y_k(n)$

  target output

- 代价函数 $E = \frac{1}{2}\sum e_k^2(n)$

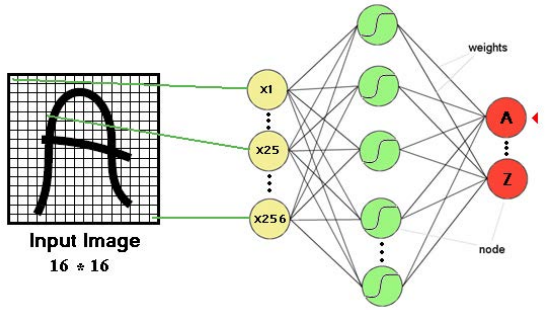
- 权值调整 $\Delta w_{kj}(l) = -\frac{\partial E}{\partial w_{kj}(l)}$



- BP algorithm was proposed by Paul Werbos in 1974 and reinvented in 1985 by Rumelhart et al.

从MLP到CNN

- 对*shifting, scaling, rotation*等非常敏感
- 过拟合!

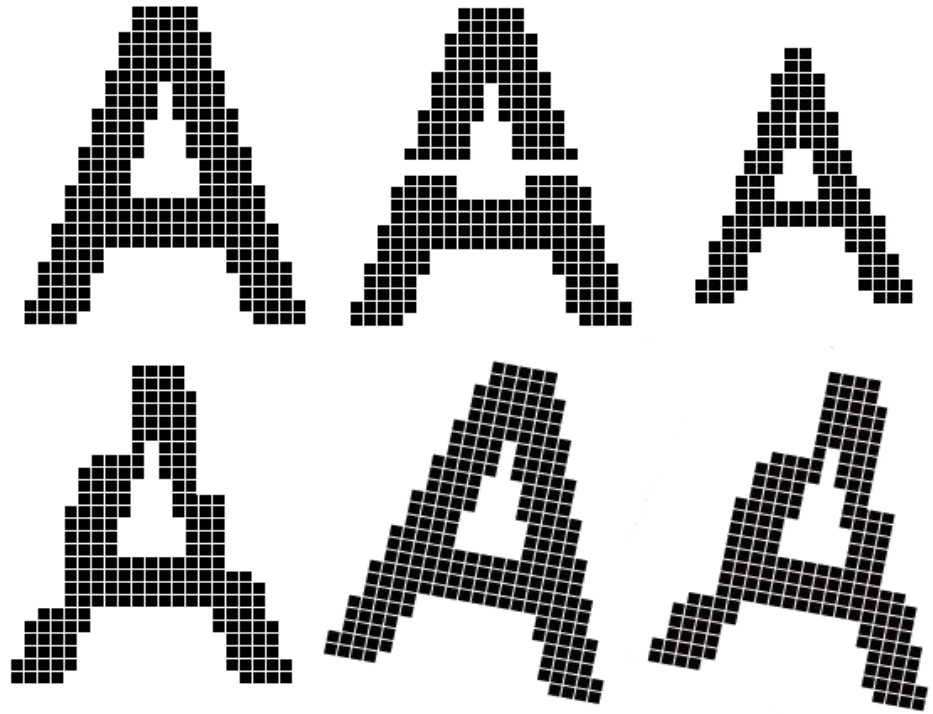


✓训练时间长
✓网络规模大
✓调整参数多

从MLP到CNN

“好” 的特征

- 充分刻画结构
- 对*shifting, scaling, rotation*等不敏感
- 训练参数少

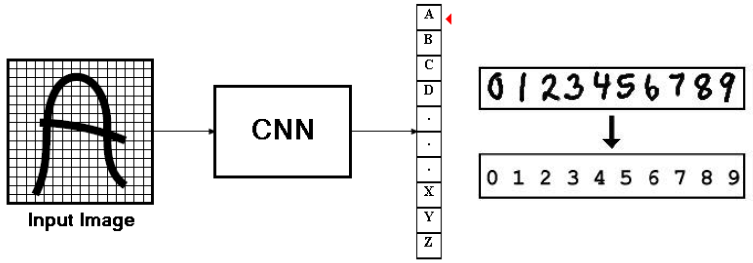


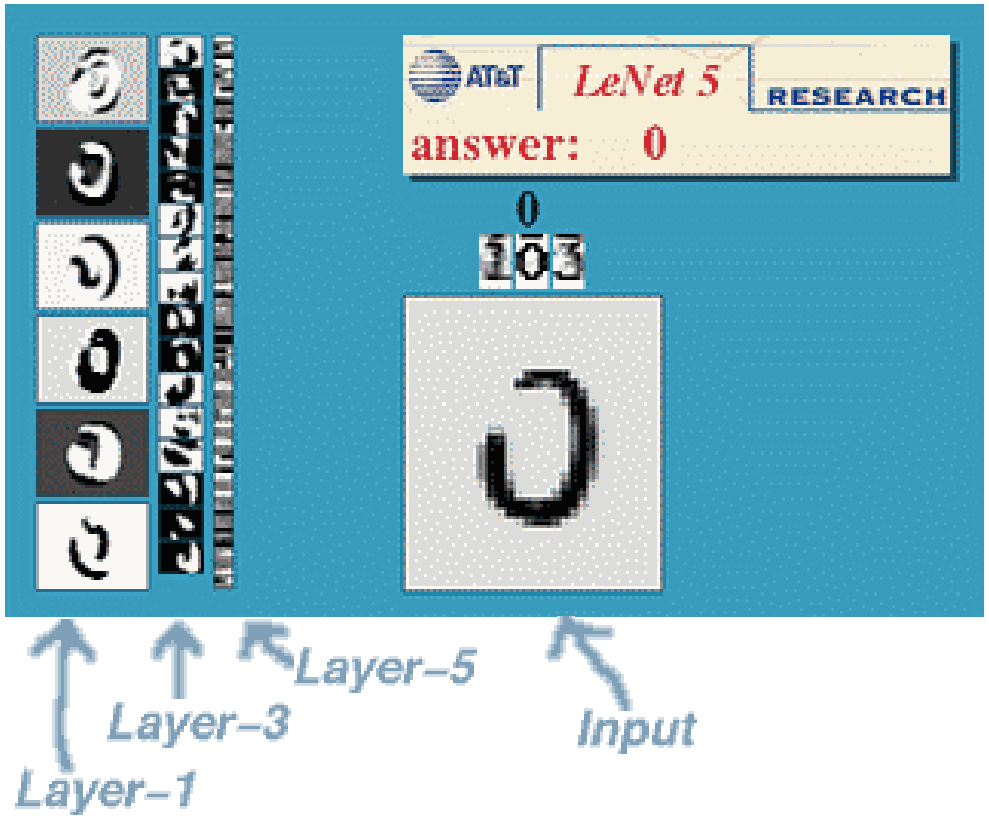
Convolutional Neural Network

Yann LeCun

Yoshua Bengio

In 1995, Yann LeCun and Yoshua Bengio introduced the concept of convolutional neural networks.

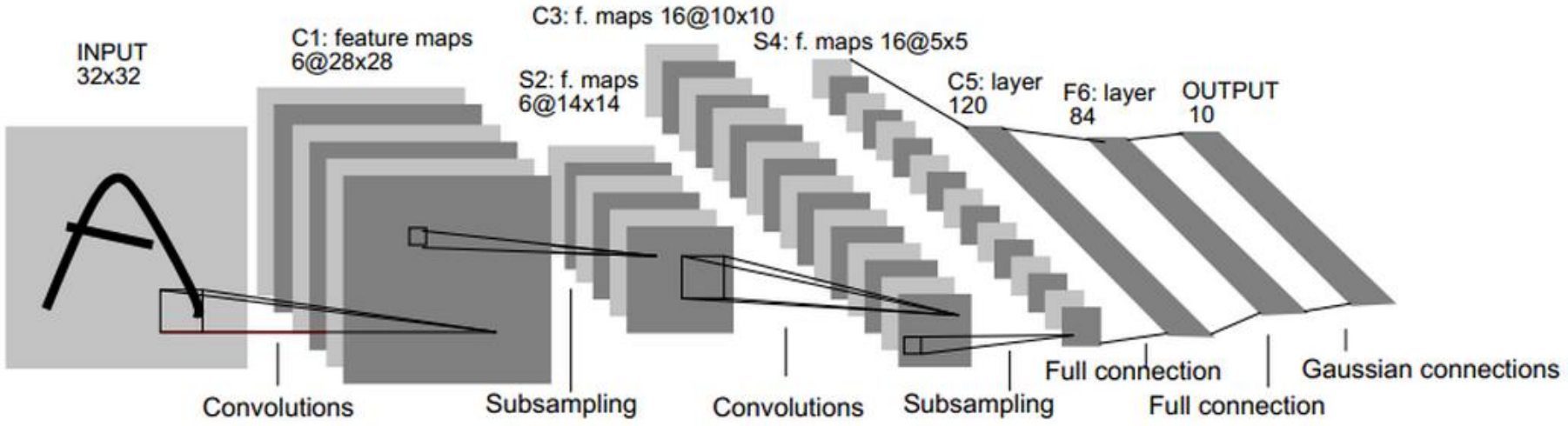




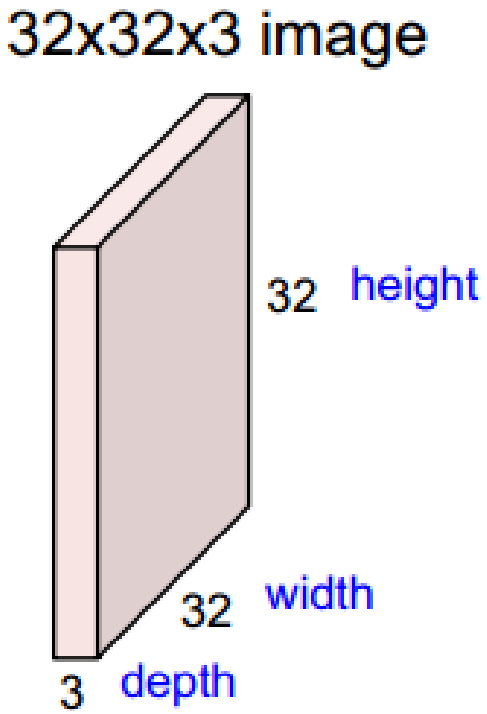
从MLP到CNN

- C1,C3: Convolutional layer.
- 5 × 5 Convolution matrix.
- S2 , S4 : Subsampling layer.
- Subsampling by factor 2.
- F6 : Fully connected layer.

LeNet-5

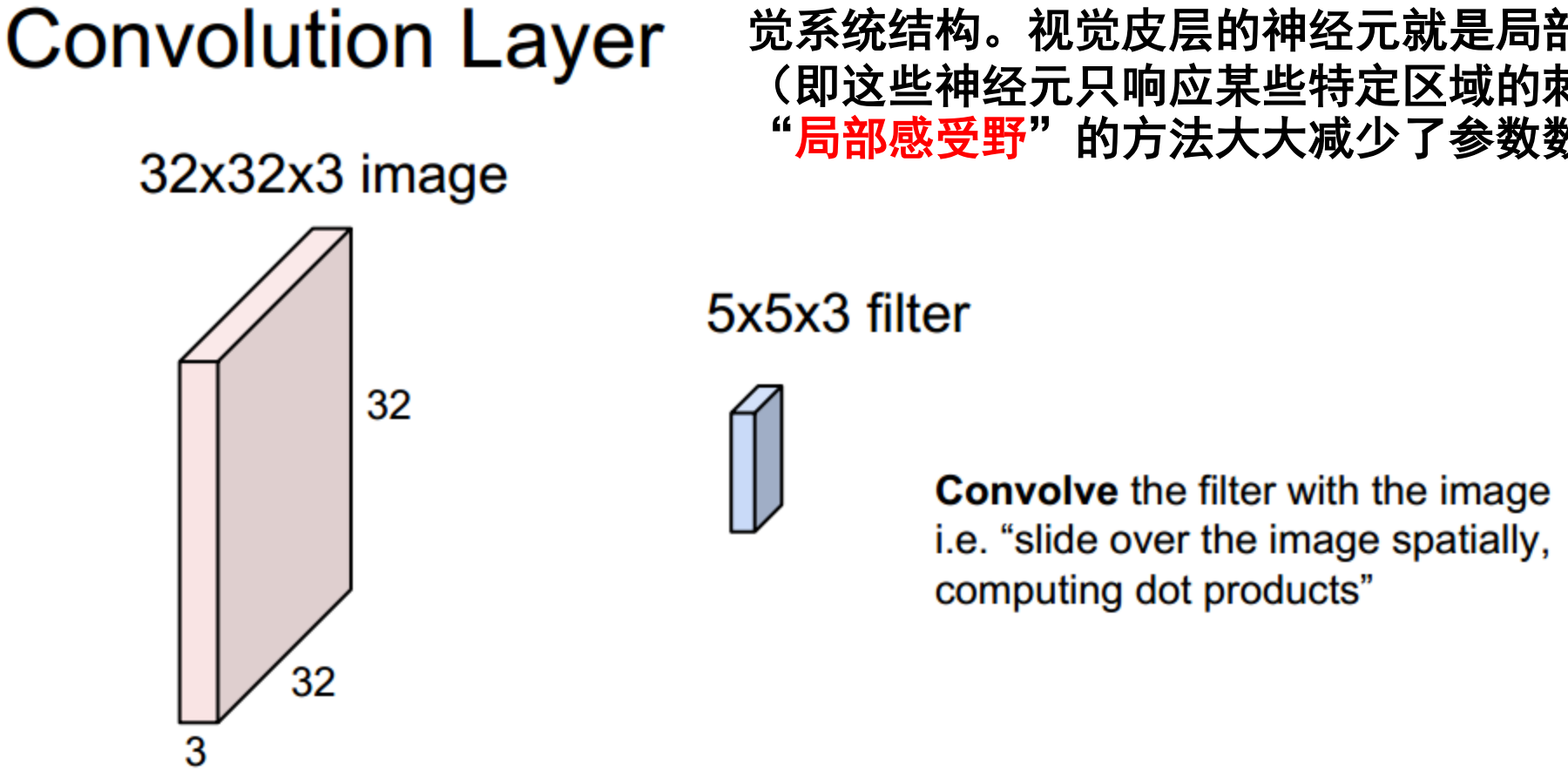


Convolution Layer

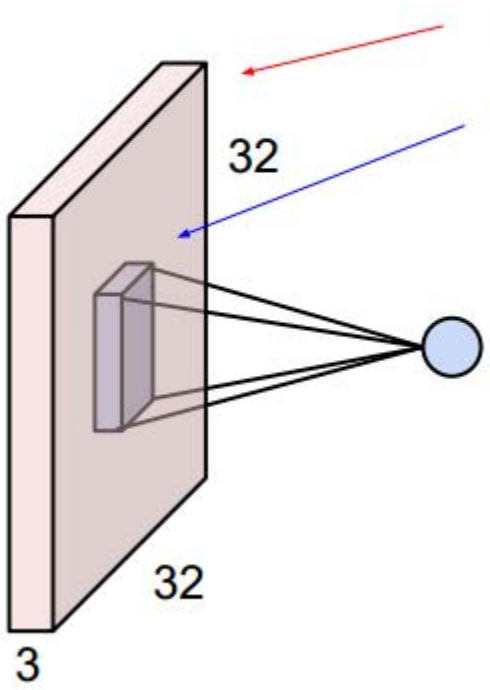


从MLP到CNN

网络部分连通的思想，也是受启发于生物学里面的视觉系统结构。视觉皮层的神经元就是局部接受信息的（即这些神经元只响应某些特定区域的刺激）。“局部感受野”的方法大大减少了参数数量。

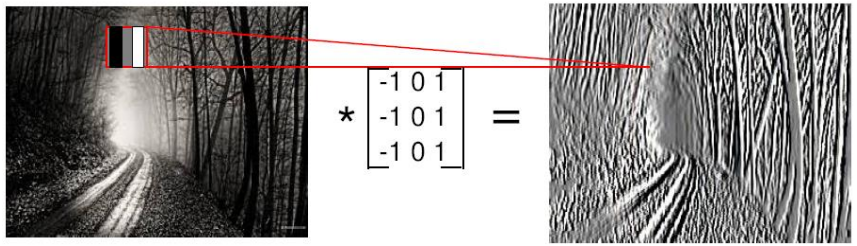


Convolution Layer

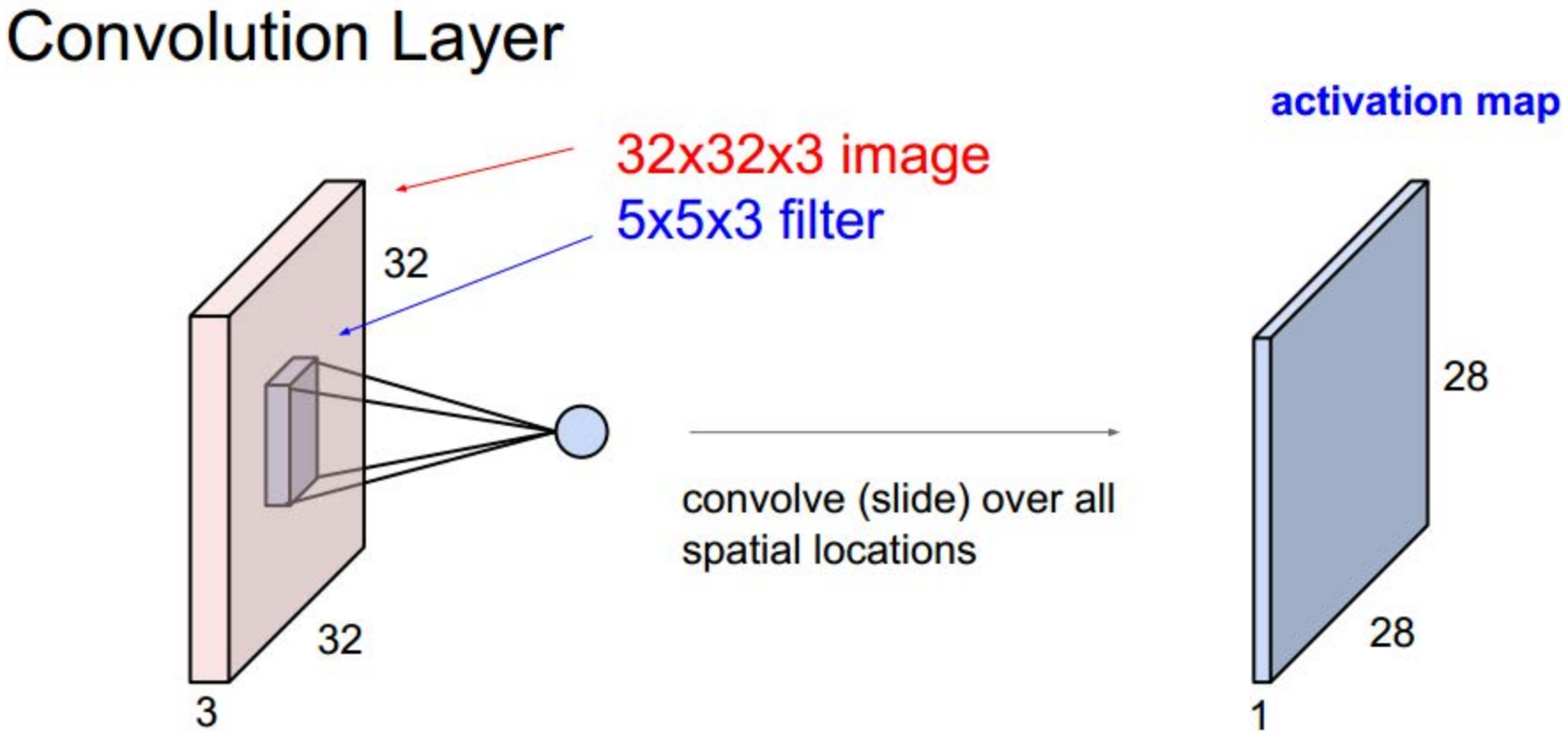


1 number:
the result of taking a dot product between the filter and a small 5x5x3 chunk of the image
(i.e. 5*5*3 = 75-dimensional dot product + bias)

$$w^T x + b$$

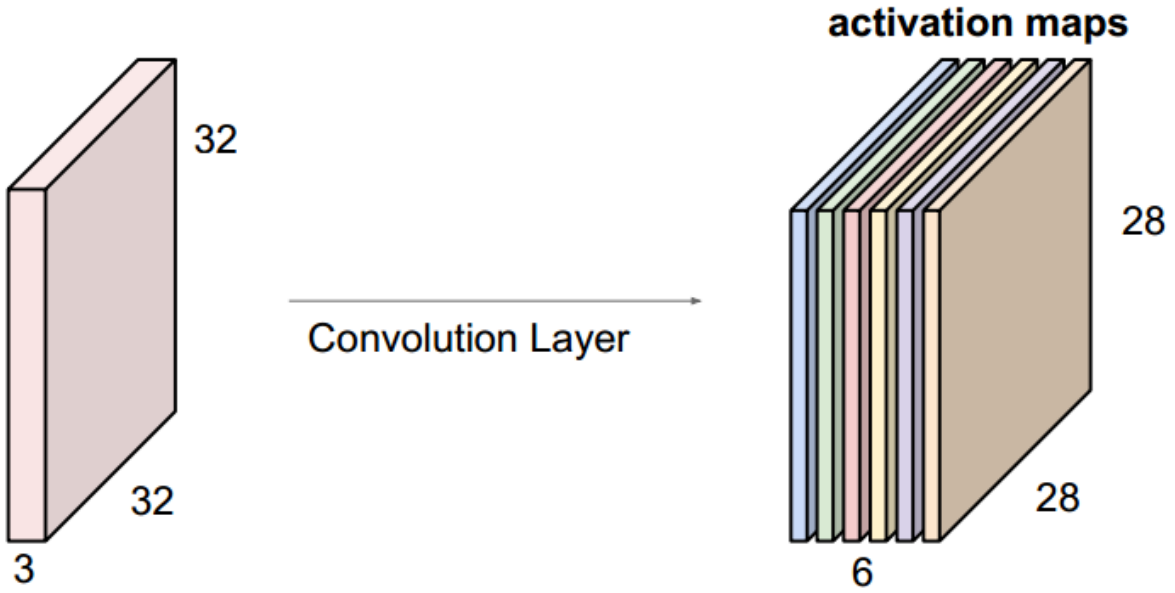


从MLP到CNN

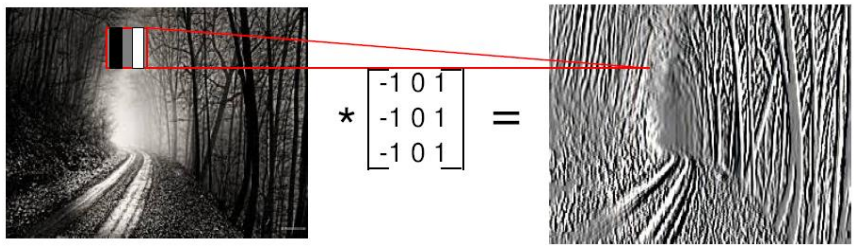

Convolution Layer
activation map
32x32x3 image
5x5x3 filter
32
32
3
convolve (slide) over all
spatial locations
28
28
1

Convolution layer

For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:

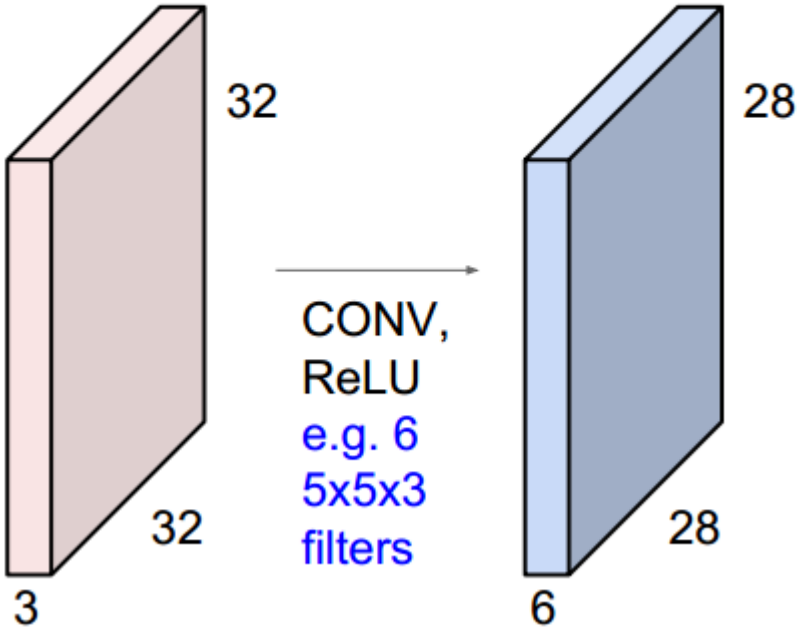


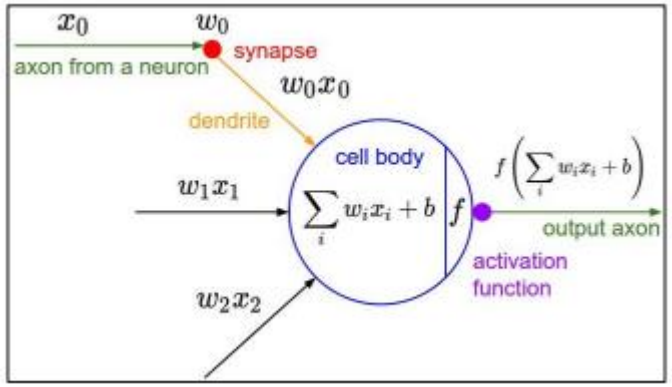
We stack these up to get a "new image" of size 28x28x6!



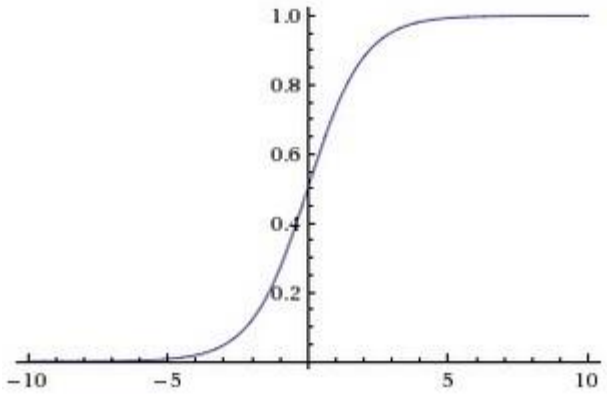
Convolution layer

Preview: ConvNet is a sequence of Convolution Layers, interspersed with activation functions

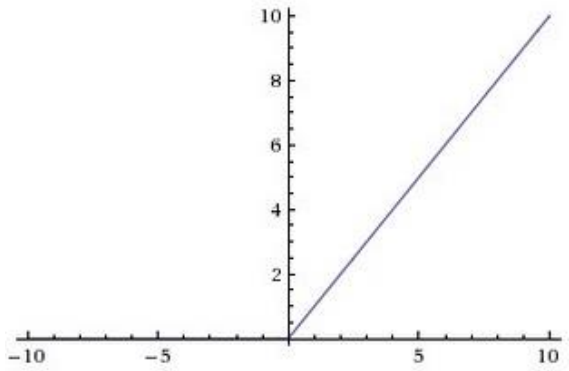




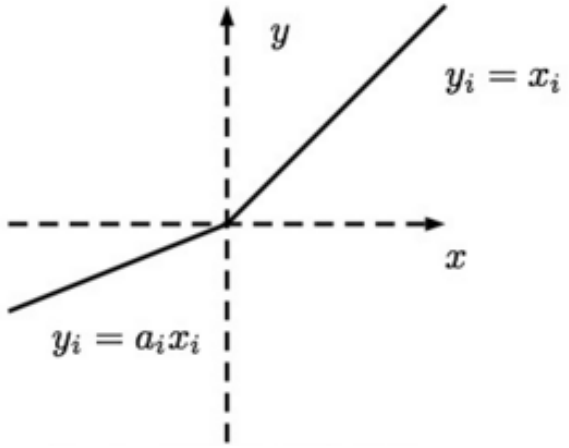
Activation function



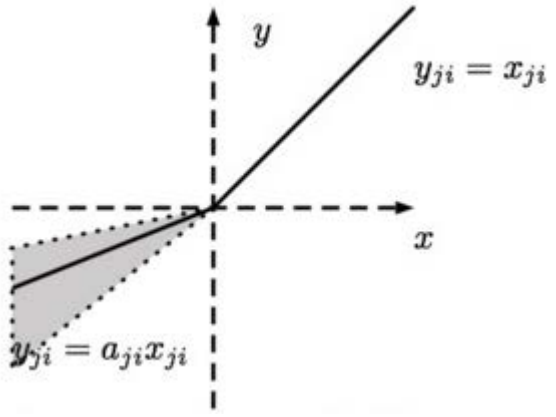
sigmoid



ReLU

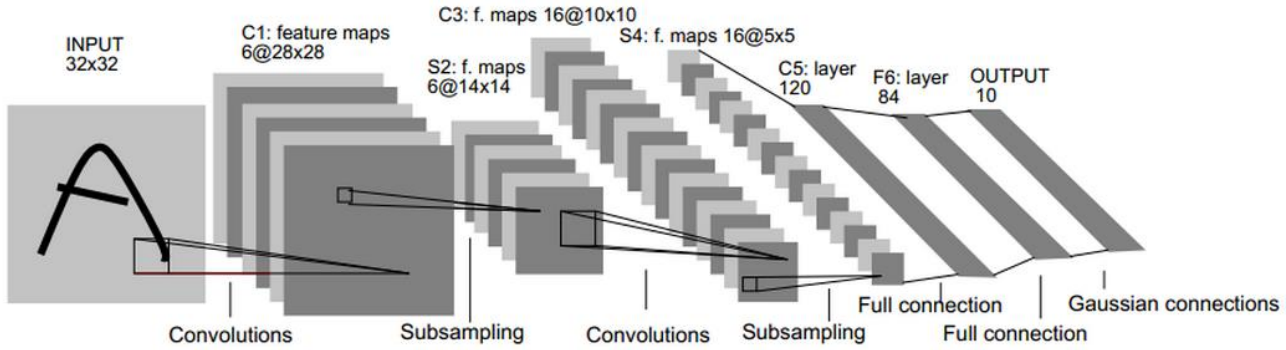


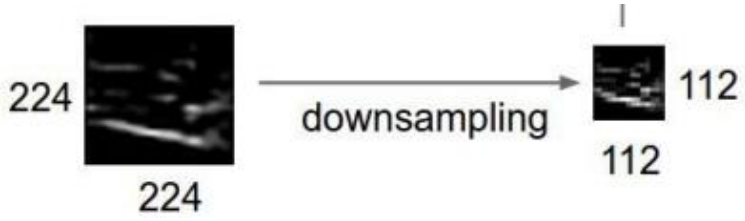
PReLU

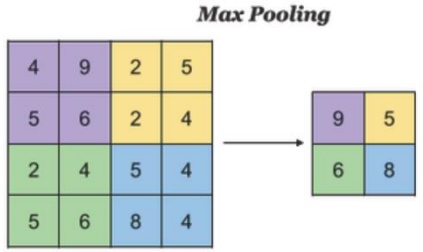


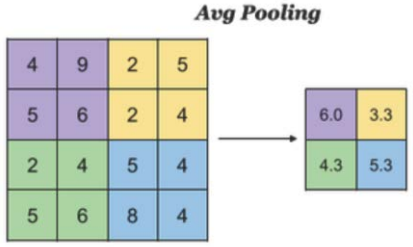
Random PReLU

Subsampling layer





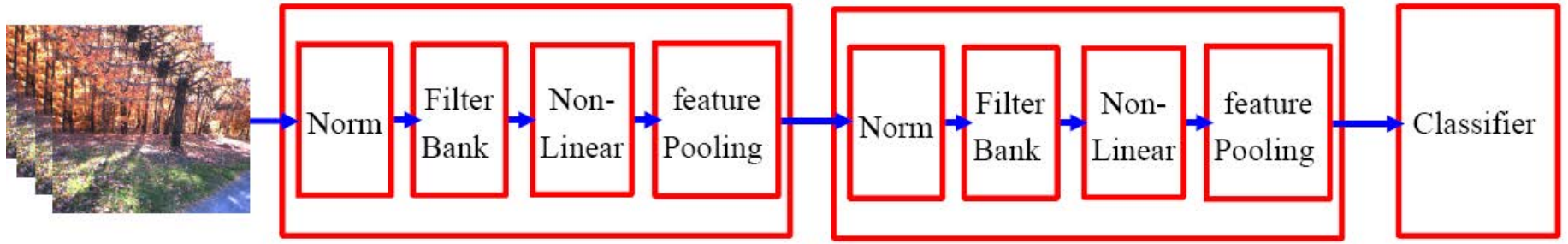




Max/Min/Average Pooling

从MLP到CNN

➢ 总结：一般结构



Normalization: variations on whitening
Subtractive: average removal, high pass filtering
Divisive: local contrast normalization, variance normalization

Filter Bank: dimension expansion, projection on overcomplete basis

Non-Linearity: sparsification, saturation, lateral inhibition....
Rectification (ReLU), Component-wise shrinkage, tanh, winner-takes-all

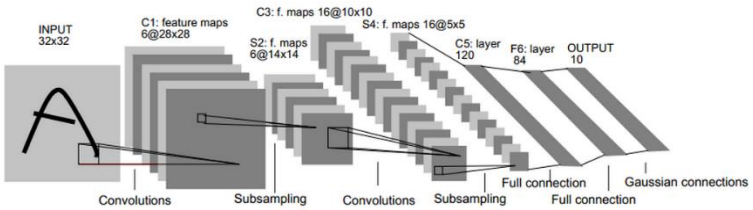
Pooling: aggregation over space or feature type

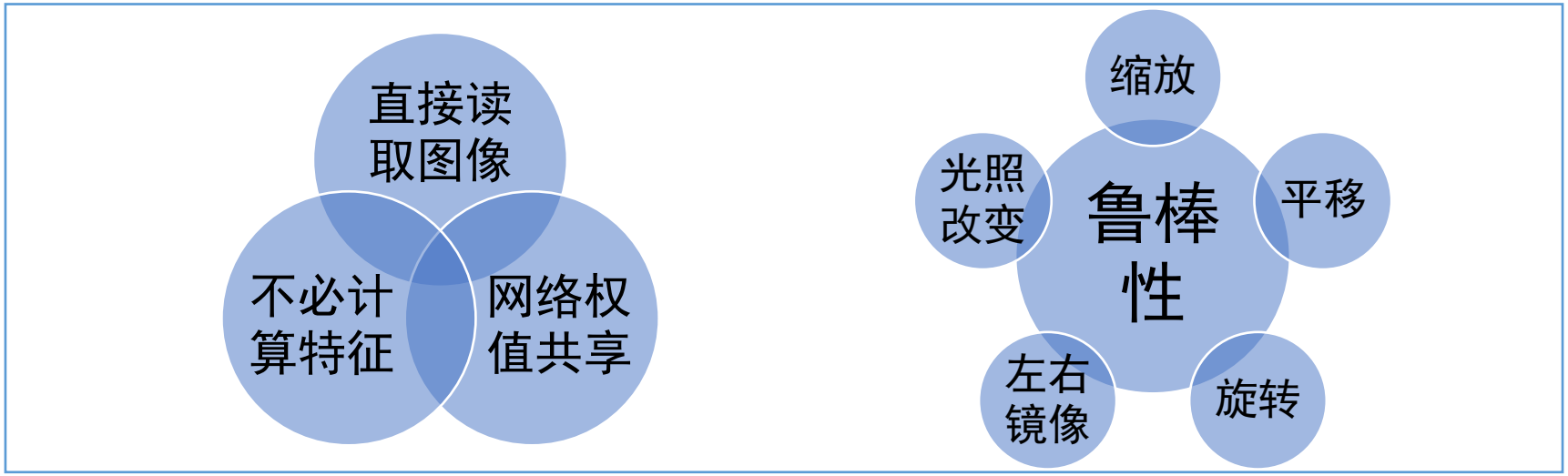
从MLP到CNN

➢ 总结：一般结构

传统的图像识别方法，需要预先使用例如SIFT/HOG等方法对图像进行特征提取，再使用SVM或稀疏编码等方法进行分类，面对复杂的问题往往难以提取合适的特征。

CNN算法能利用网络中间某一层的输出当做是数据的另一种表达，可以经过网络学习到适合于分类的特征。



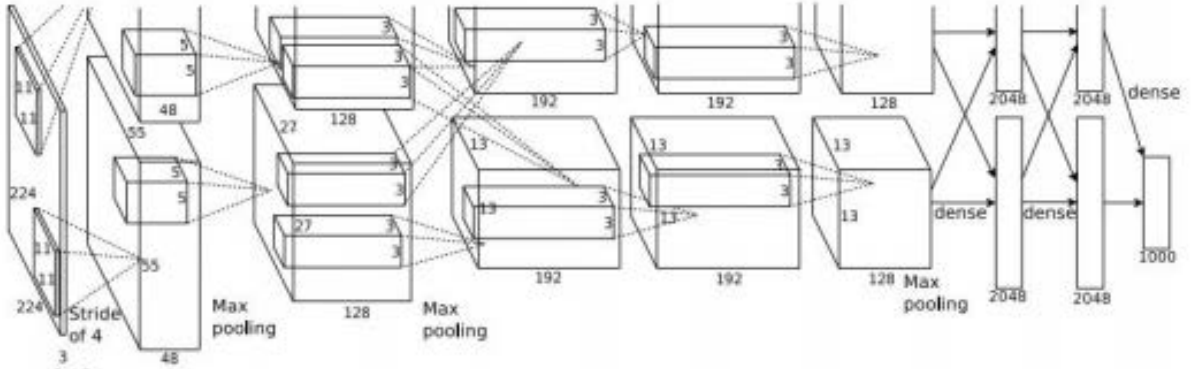


卷积神经网络的优点

从MLP到CNN

Case Study: AlexNet

[Krizhevsky et al. 2012]



Full (simplified) AlexNet architecture:
[227x227x3] INPUT
[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0
[27x27x96] MAX POOL1: 3x3 filters at stride 2
[27x27x96] NORM1: Normalization layer
[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2
[13x13x256] MAX POOL2: 3x3 filters at stride 2
[13x13x256] NORM2: Normalization layer
[13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1
[13x13x384] CONV4: 384 3x3 filters at stride 1, pad 1
[13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1
[6x6x256] MAX POOL3: 3x3 filters at stride 2
[4096] FC6: 4096 neurons
[4096] FC7: 4096 neurons
[1000] FC8: 1000 neurons (class scores)

We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry.

从MLP到CNN

Case Study: VGGNet

[Simonyan and Zisserman, 2014]

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

best model

11.2% top 5 error in ILSVRC 2013
->
7.3% top 5 error

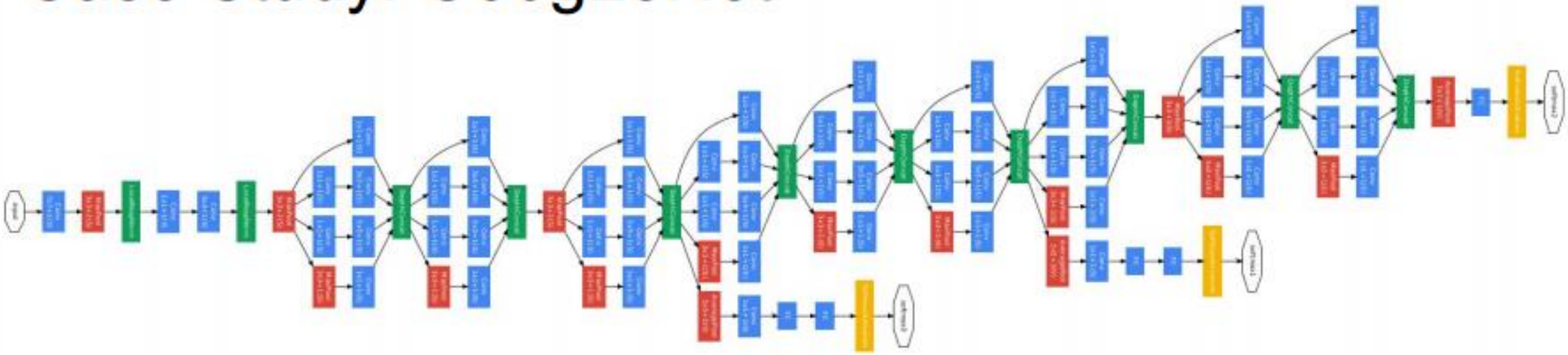
ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

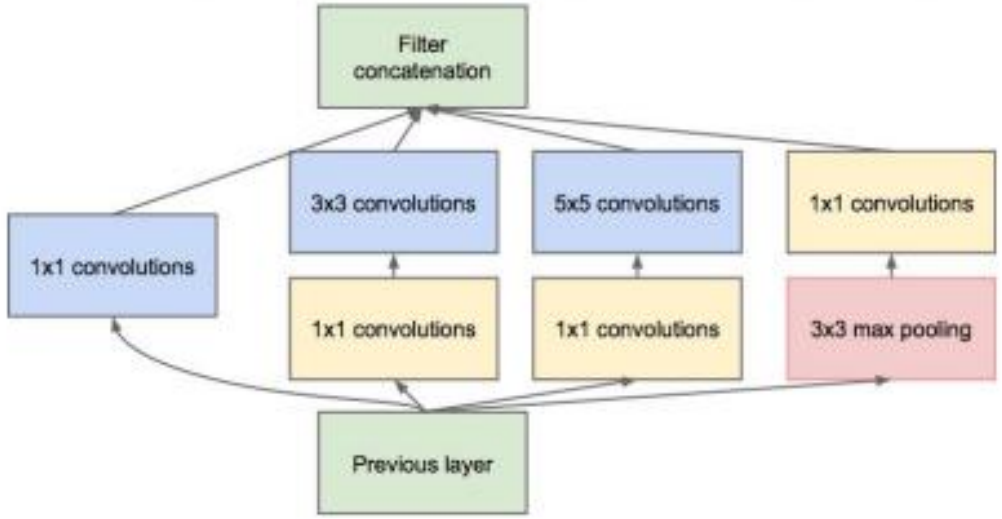
Table 2: **Number of parameters** (in millions).

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

Case Study: GoogLeNet

[Szegedy et al., 2014]





Inception module

ILSVRC 2014 winner (6.7% top 5 error)

Case Study: ResNet *[He et al., 2015]*

ILSVRC 2015 winner (3.6% top 5 error)

Microsoft Research

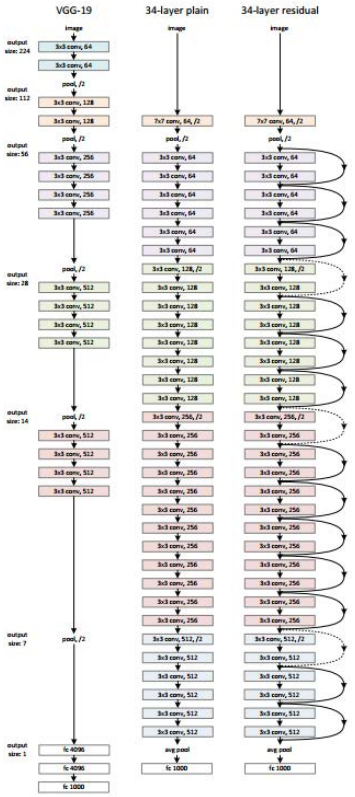
MSRA @ ILSVRC & COCO 2015 Competitions

- **1st places in all five main tracks**
  - ImageNet Classification: "*Ultra-deep*" (quote Yann) 152-layer nets
  - ImageNet Detection: 16% better than 2nd
  - ImageNet Localization: 27% better than 2nd
  - COCO Detection: 11% better than 2nd
  - COCO Segmentation: 12% better than 2nd

*improvements are relative numbers

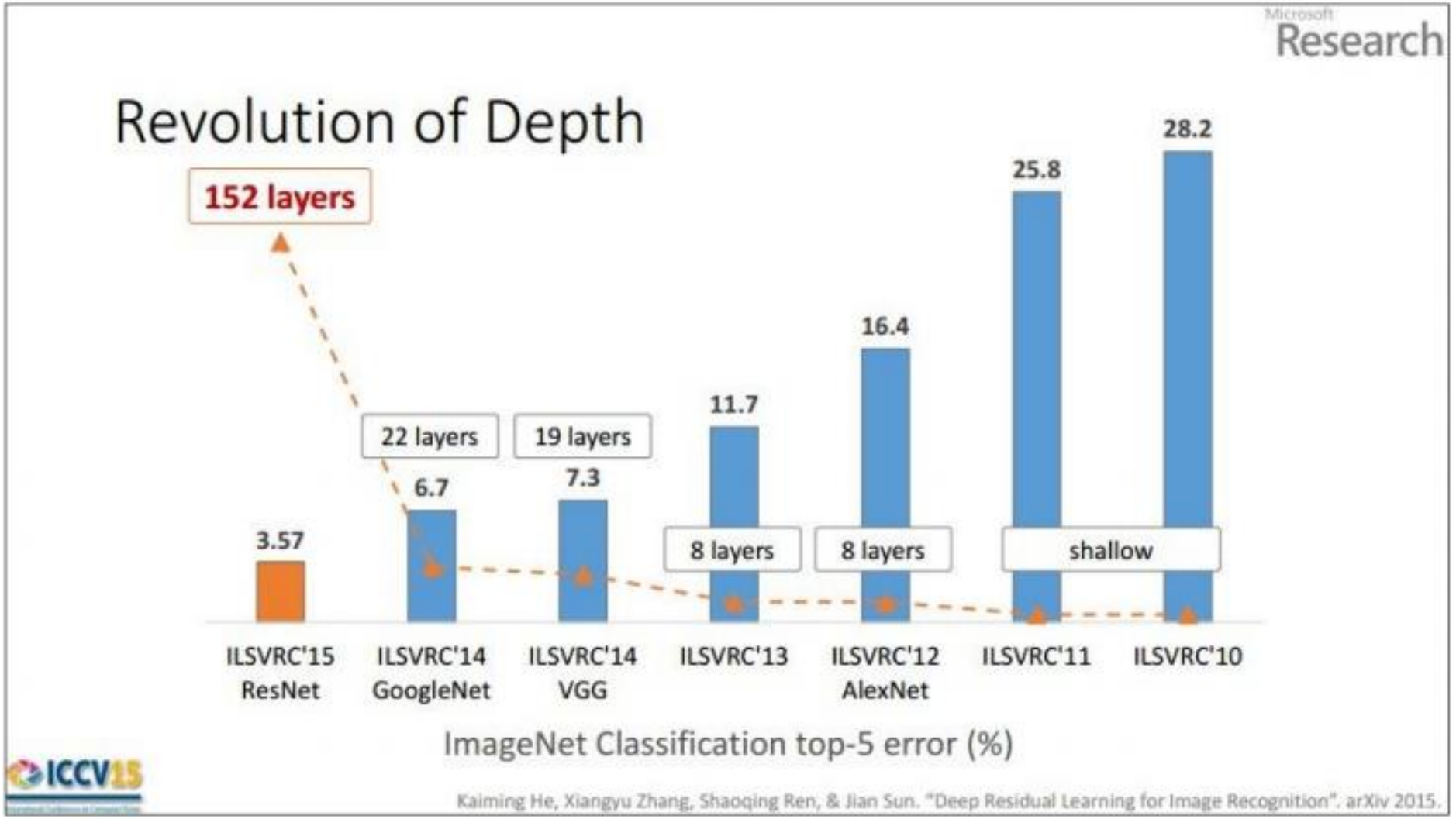
ICCV15

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



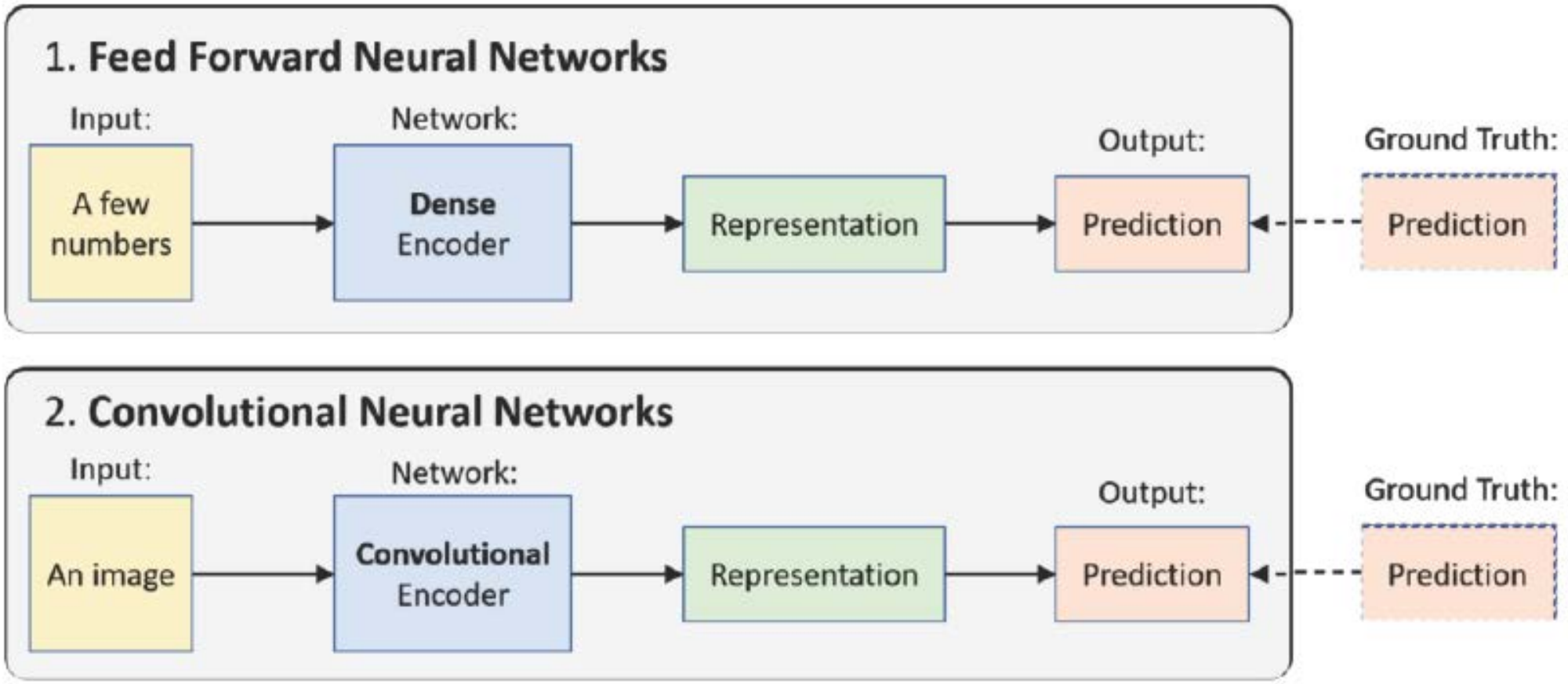
Slide from Kaiming He's recent presentation https://www.youtube.com/watch?v=1PGLj-uKT1w

从MLP到CNN



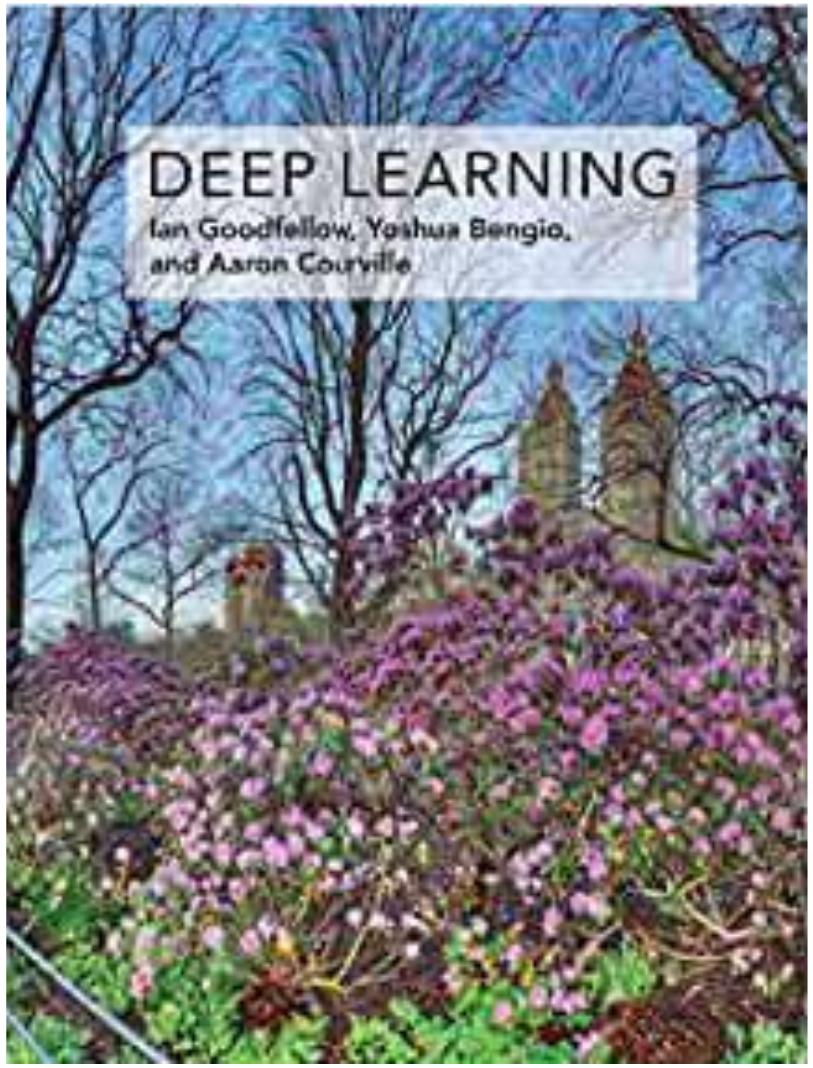
(slide from Kaiming He's recent presentation)

从MLP到CNN



$$\Delta w_{kj}(l) = -\frac{\partial E}{\partial w_{kj}(l)}$$

从MLP到CNN



The International Journal of Robotics Research

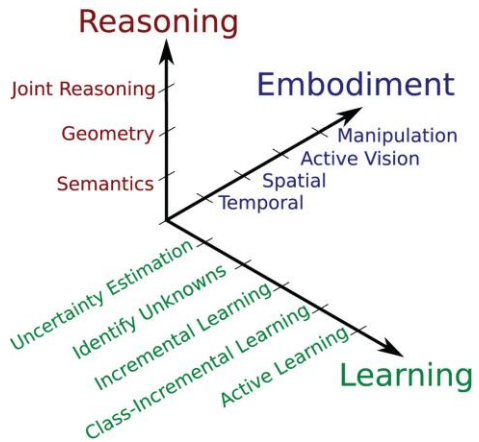
Impact Factor: 6.887

Available access | Research article | First published online April 27, 2018

The limits and potentials of deep learning for robotics

Niko Sünderhauf, Oliver Brock, [...], and Peter Corke +8 View all authors and affiliations

Volume 37, Issue 4-5 | https://doi.org/10.1177/0278364918770733



（1）一般介绍

（2）从多层感知机到卷积神经网络

（3）应用

应用

➢ 目标检测

CAT

(x, y)

h,heigth,
高度

(x, y, w, h)

w,width,宽度

Input: image

Only one object,
simpler than detection

Neural Net

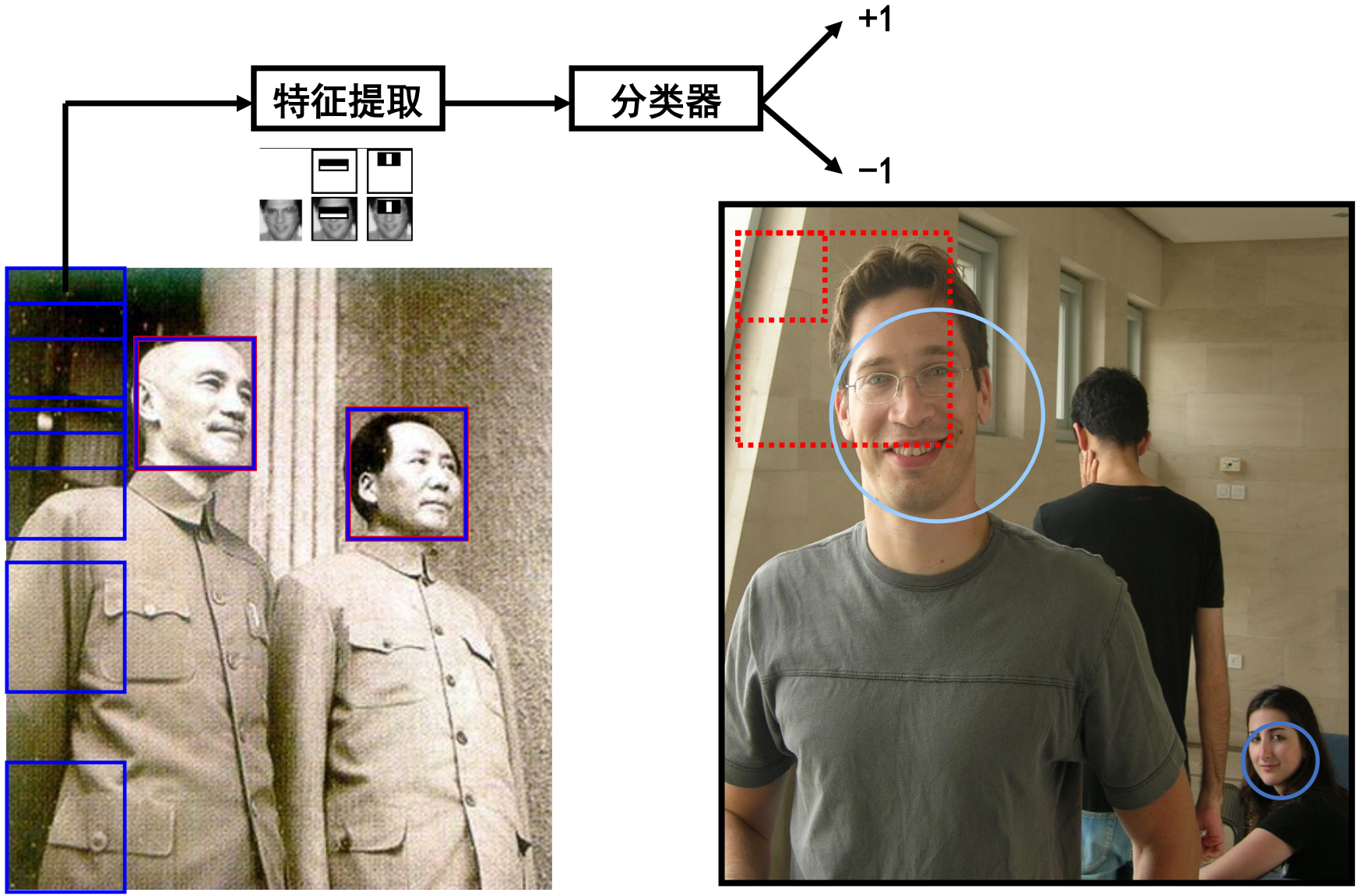
Output:
Box coordinates
(4 numbers)

Correct output:
box coordinates
(4 numbers)

Loss:
L2 distance
用欧氏距离作损失函数

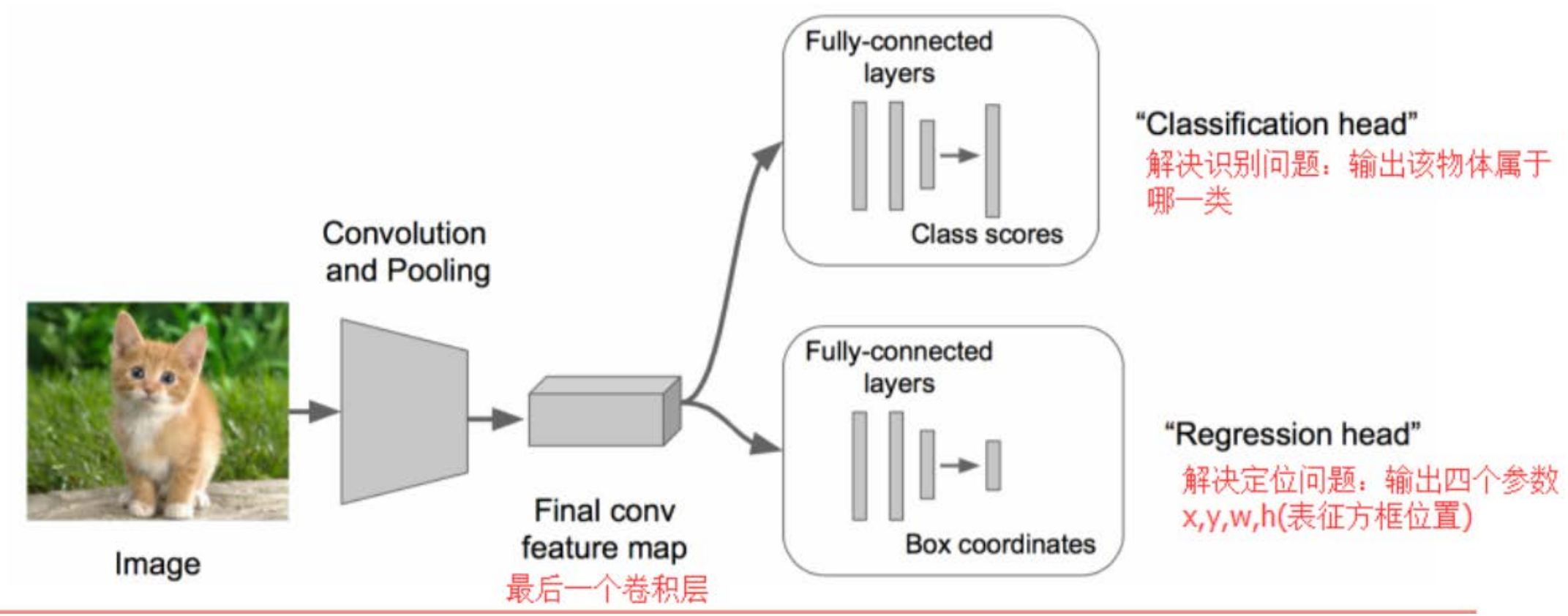
应用

- 目标检测：Adaboost



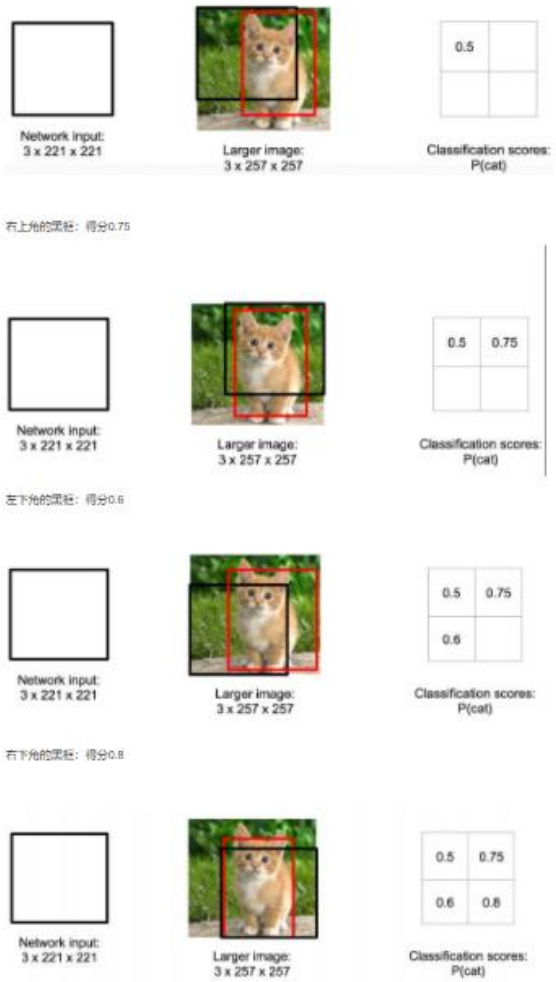
应用

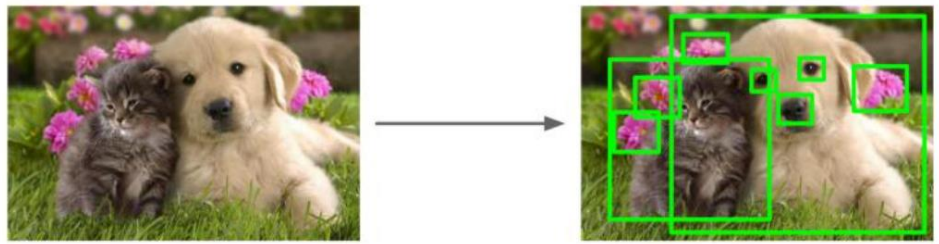
➢ 目标检测：深度学习的思路



应用

➢ 目标检测：深度学习的思路



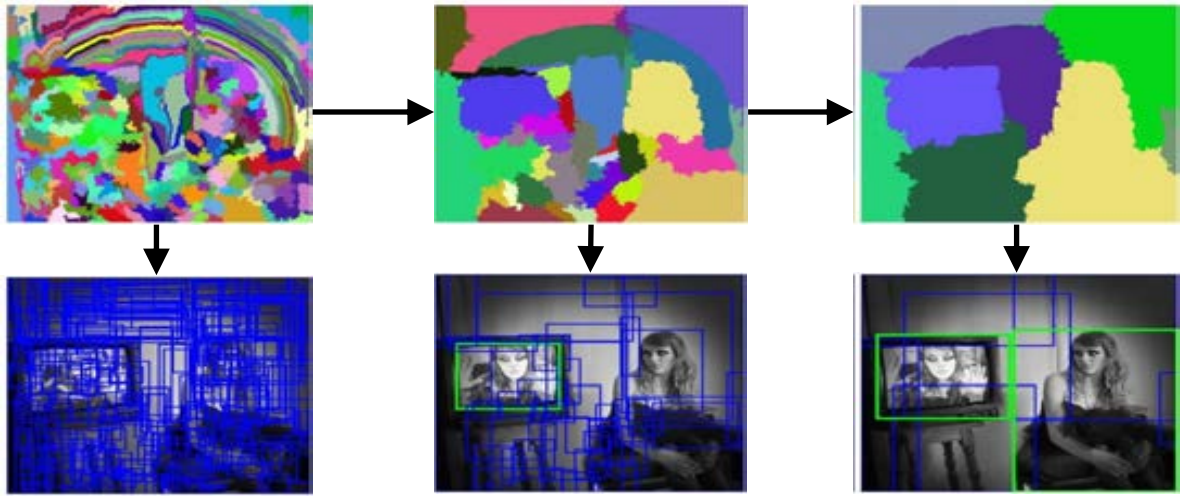


Method	Approach	Outputs Segments	Outputs Score	Control #proposals	Time (sec.)	Repea-tability	Recall Results	Detection Results
Bing [18]	Window scoring		✓	✓	0.2	⋆⋆⋆	⋆	·
CPMC [19]	Grouping	✓	✓	✓	250	-	⋆⋆	⋆
EdgeBoxes [20]	Window scoring		✓	✓	0.3	⋆⋆	⋆⋆⋆	⋆⋆⋆
Endres [21]	Grouping	✓	✓	✓	100	-	⋆⋆⋆	⋆⋆
Geodesic [22]	Grouping	✓		✓	1	⋆	⋆⋆⋆	⋆⋆
MCG [23]	Grouping	✓	✓	✓	30	⋆	⋆⋆⋆	⋆⋆⋆
Objectness [24]	Window scoring		✓	✓	3	·	⋆	·
Rahtu [25]	Window scoring		✓	✓	3	·	·	⋆
RandomizedPrim's [26]	Grouping	✓		✓	1	⋆	⋆	⋆⋆
Rantalankila [27]	Grouping	✓		✓	10	⋆⋆	·	⋆⋆
Rigor [28]	Grouping	✓		✓	10	⋆	⋆⋆	⋆⋆
SelectiveSearch [29]	Grouping	✓	✓	✓	10	⋆⋆	⋆⋆⋆	⋆⋆⋆
Gaussian				✓	0	·	·	⋆
SlidingWindow				✓	0	⋆⋆⋆	·	·
Superpixels		✓			1	⋆	·	·
Uniform				✓	0	·	·	·

应用

➢ 目标检测：深度学习的思路

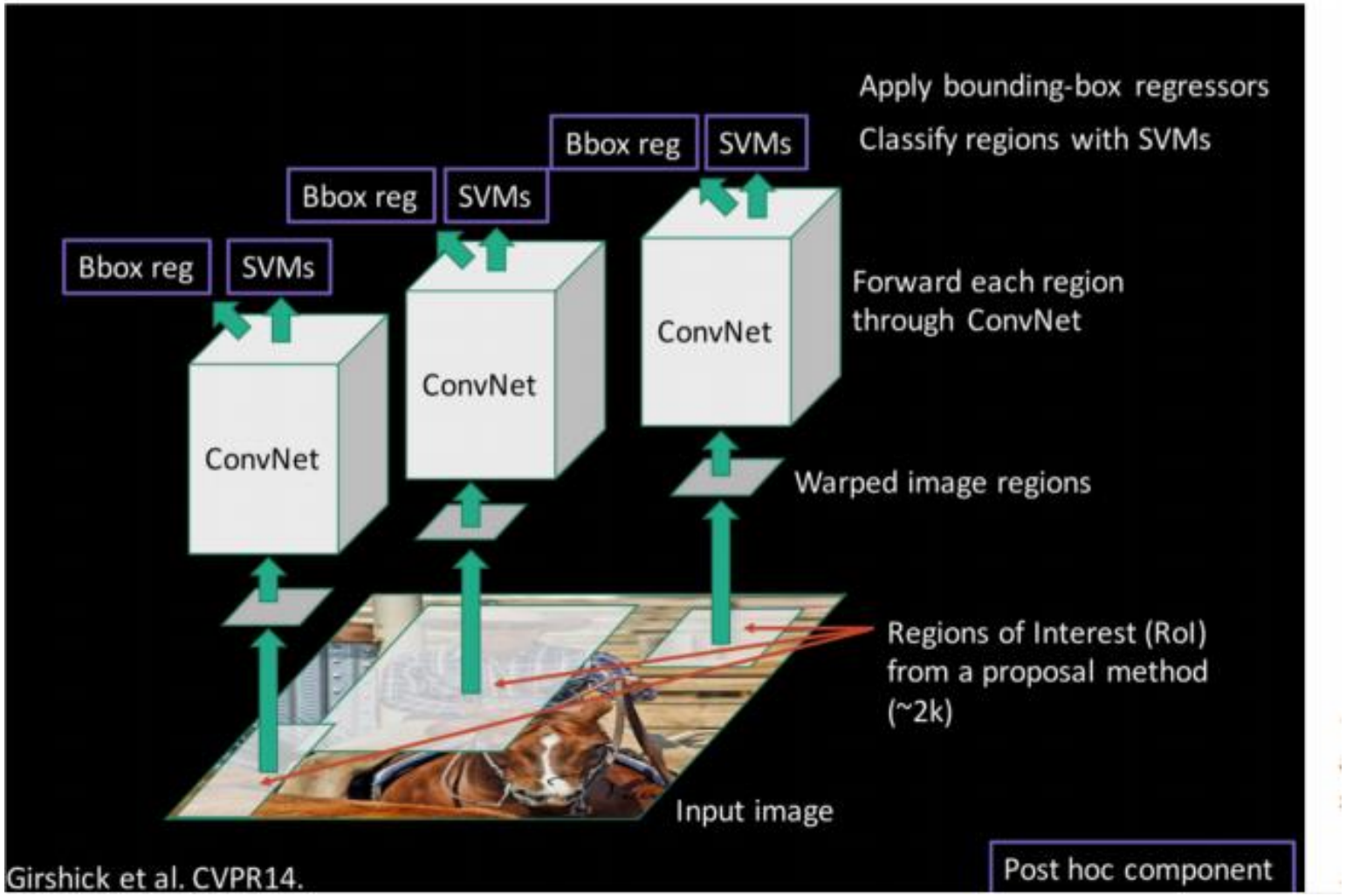
Region proposals, Selective Search



Method	Approach	Outputs Segments	Outputs Score	Control #proposals	Time (sec.)	Repea-tability	Recall Results	Detection Results
Bing [18]	Window scoring		✓	✓	0.2	⋆⋆⋆	⋆	·
CPMC [19]	Grouping	✓	✓	✓	250	-	⋆⋆	⋆
EdgeBoxes [20]	Window scoring		✓	✓	0.3	⋆⋆	⋆⋆⋆	⋆⋆⋆
Endres [21]	Grouping	✓	✓	✓	100	-	⋆⋆⋆	⋆⋆
Geodesic [22]	Grouping	✓		✓	1	⋆	⋆⋆⋆	⋆⋆
MCG [23]	Grouping	✓	✓	✓	30	⋆	⋆⋆⋆	⋆⋆⋆
Objectness [24]	Window scoring		✓	✓	3	·	⋆	·
Rahtu [25]	Window scoring		✓	✓	3	·	·	⋆
RandomizedPrim's [26]	Grouping	✓		✓	1	⋆	⋆	⋆⋆
Rantalankila [27]	Grouping	✓		✓	10	⋆⋆	·	⋆⋆
Rigor [28]	Grouping	✓		✓	10	⋆	⋆⋆	⋆⋆
SelectiveSearch [29]	Grouping	✓	✓	✓	10	⋆⋆	⋆⋆⋆	⋆⋆⋆
Gaussian				✓	0	·	·	⋆
SlidingWindow				✓	0	⋆⋆⋆	·	·
Superpixels		✓			1	⋆	·	·
Uniform				✓	0	·	·	·

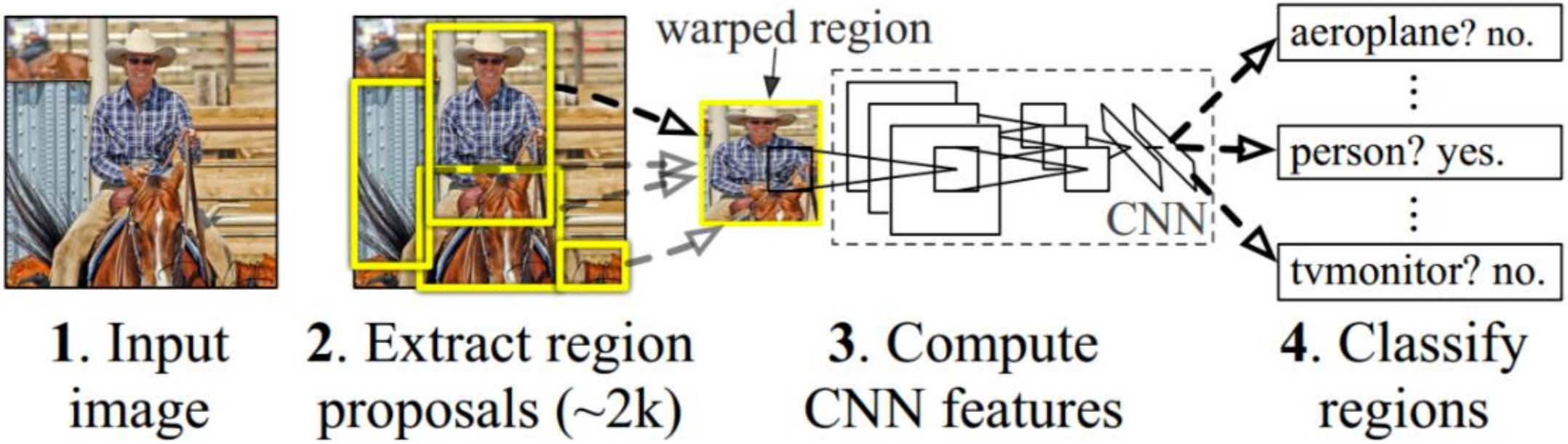
应用

➢ 目标检测：R-CNN


Apply bounding-box regressors
Classify regions with SVMs
Bbox reg
SVMs
Bbox reg
SVMs
Bbox reg
SVMs
ConvNet
ConvNet
ConvNet
Forward each region through ConvNet
Warped image regions
Regions of Interest (RoI) from a proposal method (~2k)
Input image
Girshick et al. CVPR14.
Post hoc component

应用

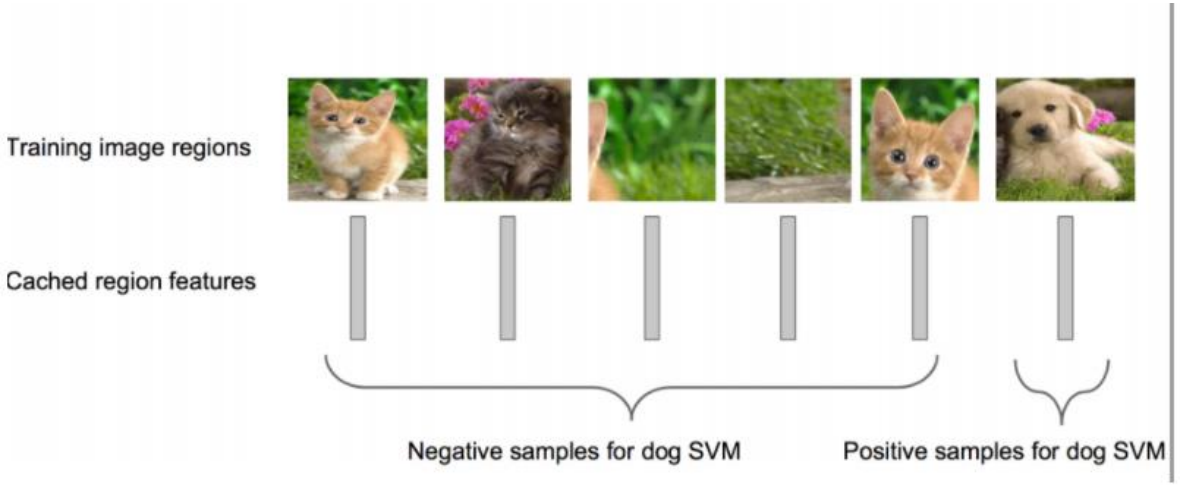
➢ 目标检测：R-CNN



- R. Girshick, J. Donahue, T. Darrell, J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR 2014*

应用

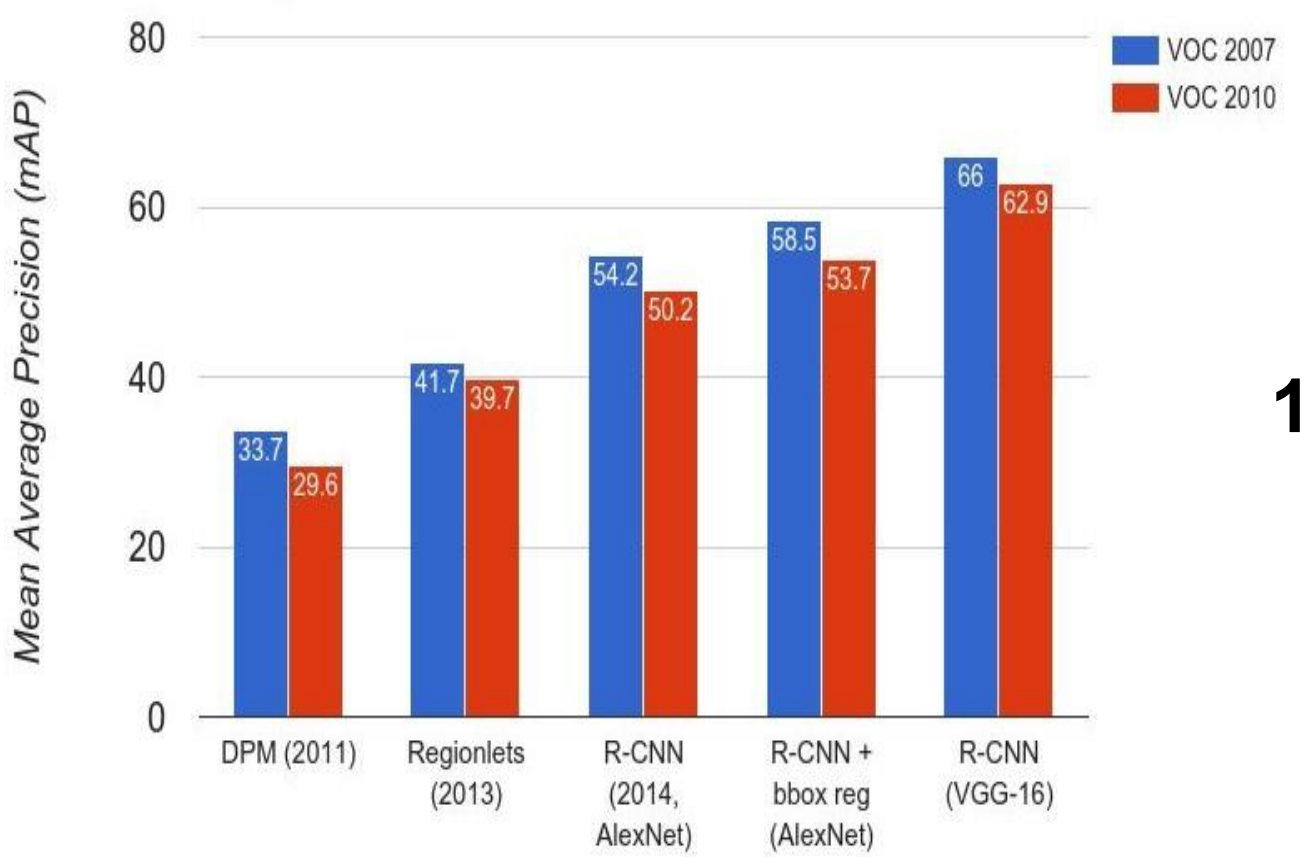
➢ 目标检测：R-CNN



步骤五：使用回归器精细修正候选框位置：对于每一个类，训练一个线性回归模型去判定这个框是否框得完美

应用

➢ 目标检测：R-CNN

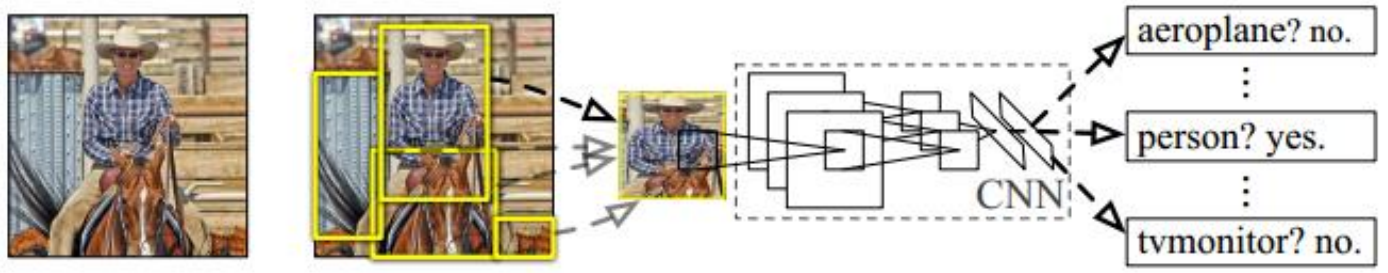


10+ AP gains

应用

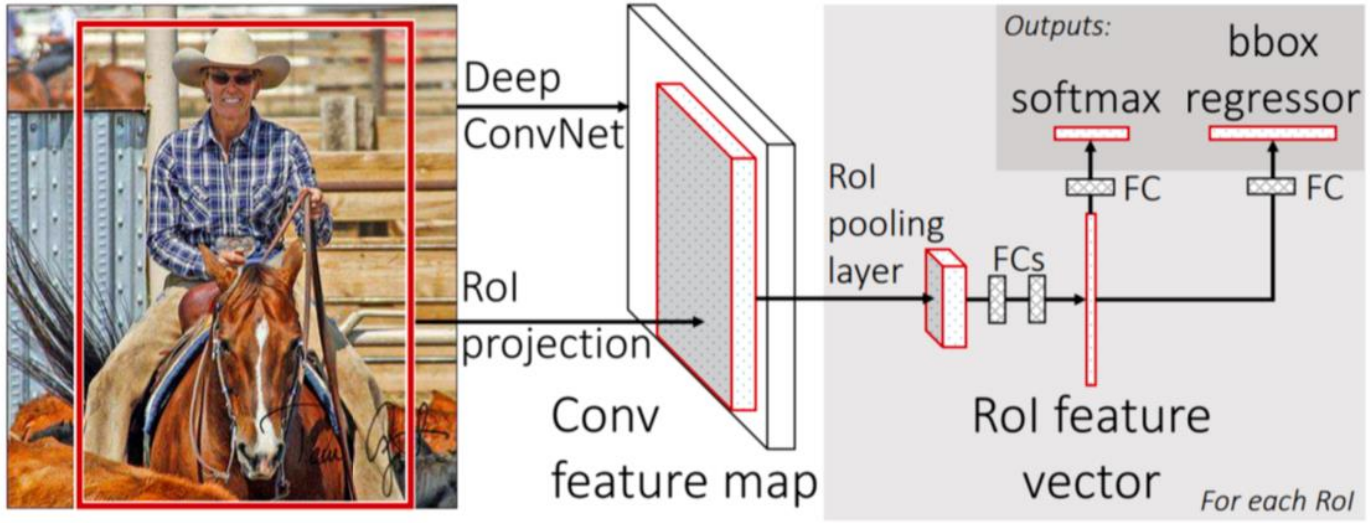
➢ 目标检测：R-CNN

- Training and testing is expensive in space and time.
  - --Why: for each image, one need to extract CNN features for about 2k object proposals
  - and make classification. But the features are shared in space.
- Relays on the region proposal method (selective search)
- Training is a multi-stage pipeline

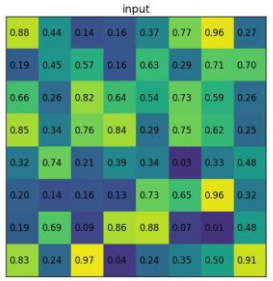


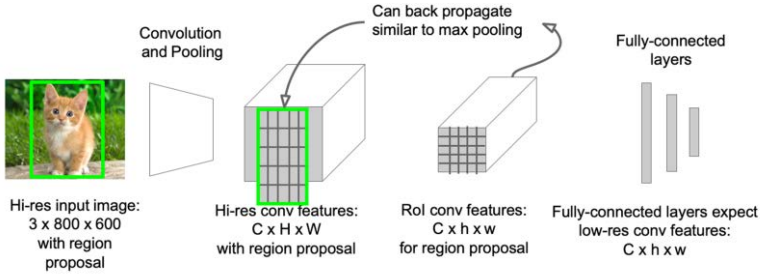
应用

➢ 目标检测：Fast RCNN



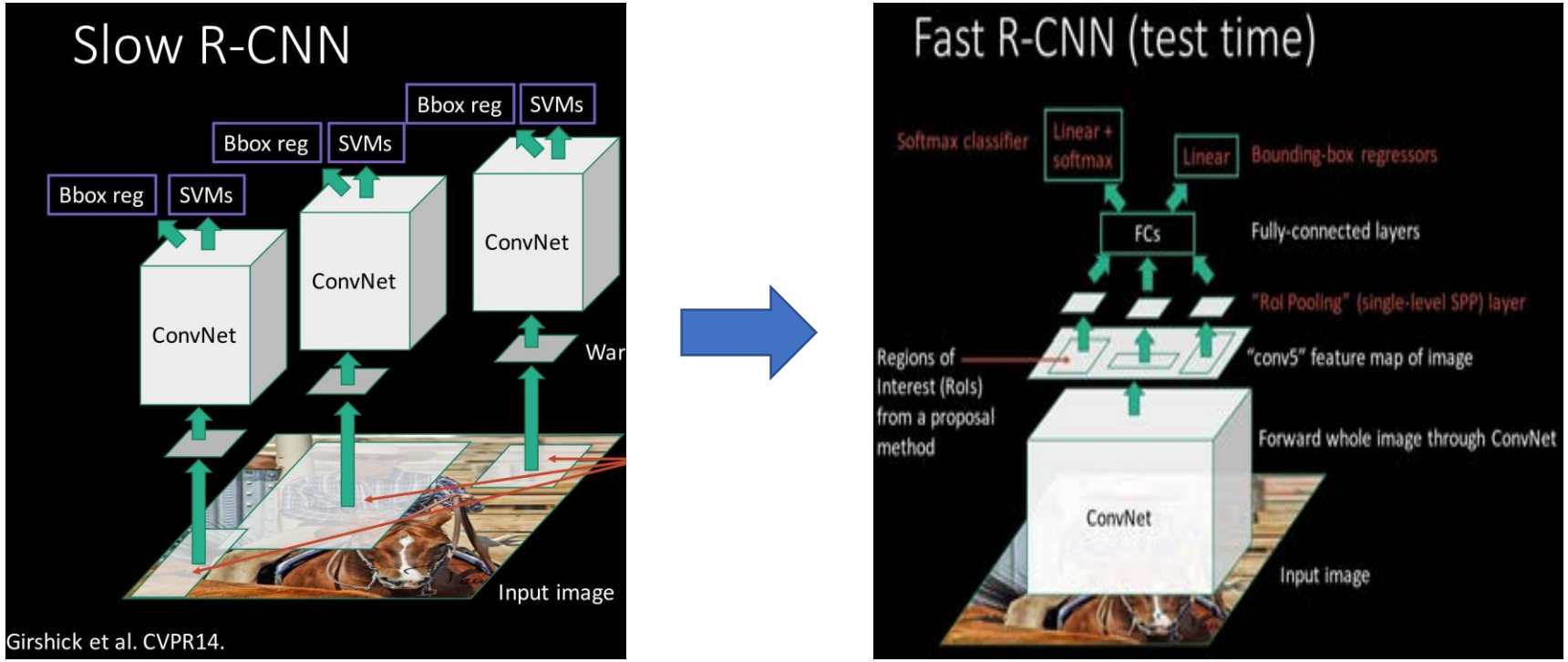
Sliding windows on the feature maps





应用

➢ 目标检测：Fast RCNN



应用

➢ 目标检测：Fast RCNN

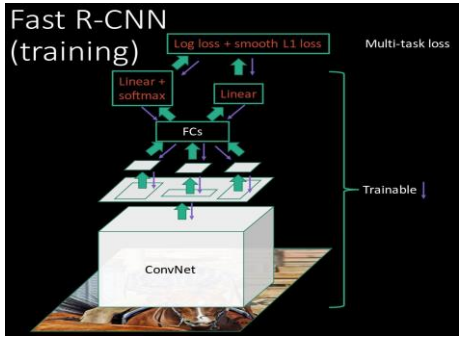
		R-CNN	Fast R-CNN
Faster!	Training Time:	84 hours	**9.5 hours**
	(Speedup)	1x	**8.8x**
FASTER!	Test time per image	47 seconds	**0.32 seconds**
	(Speedup)	1x	**146x**
Better!	mAP (VOC 2007)	66.0	**66.9**

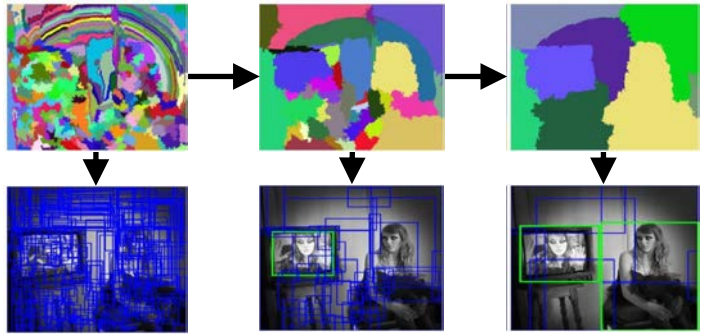
Using VGG-16 CNN on Pascal VOC 2007 dataset

应用

➢ 目标检测：Fast RCNN

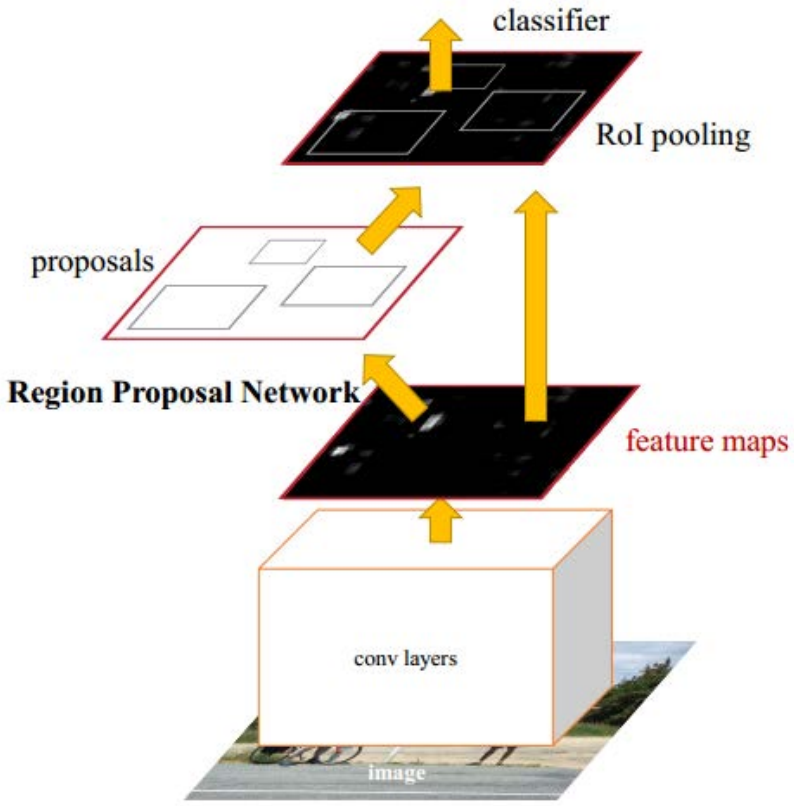
	R-CNN	Fast R-CNN
Test time per image	47 seconds	**0.32 seconds**
(Speedup)	1x	**146x**
Test time per image with Selective Search	50 seconds	**2 seconds**
(Speedup)	1x	**25x**





应用

➢ 目标检测：Faster RCNN



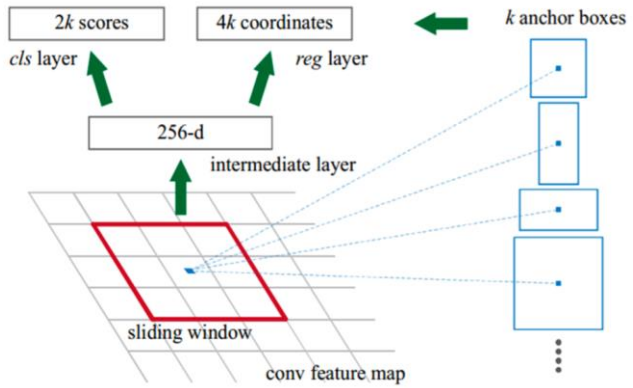
Region Proposal Net

+

Fast RCNN

➢ 目标检测：Faster RCNN

- Region proposal network

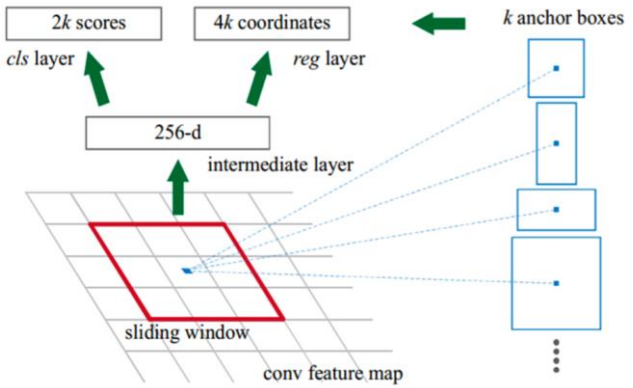


- Input an image of any size
- Generate conv feature map
- Map to a lower-dimensional feature
- Output objectness score and bounding box

The region proposal network is a FCN which outputs K*(4+2) sized vectors.

➢ 目标检测：Faster RCNN

- Region proposal network



LOSS

- Positive: Among K anchors, one with highest IOU (IOU >= 0.7)
- Negative: IOU <= 0.3
- Others: Do not contribute

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*)$$

The mini-batch size(256)

The number of anchor locations

应用

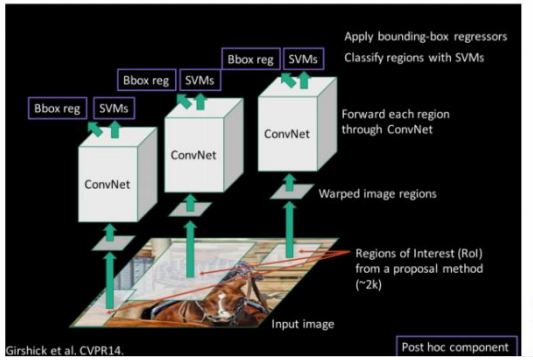
➢ 目标检测：Faster RCNN

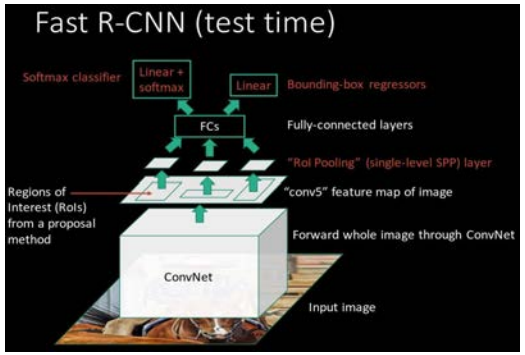
	R-CNN	Fast R-CNN	Faster R-CNN
Test time per image (with proposals)	50 seconds	2 seconds	**0.2 seconds**
(Speedup)	1x	25x	**250x**
mAP (VOC 2007)	66.0	**66.9**	**73.2**

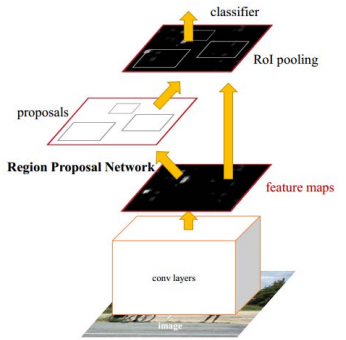
应用

➢ 目标检测：总结

- **RCNN**
  1. 在图像中确定约1000-2000个候选框 (使用选择性搜索)
  2. 每个候选框内图像块缩放至相同大小，并输入到CNN内进行特征提取
  3. 对候选框中提取出的特征，使用分类器判别是否属于一个特定类
  4. 对于属于某一特征的候选框，用回归器进一步调整其位置
- **Fast RCNN**
  1. 在图像中确定约1000-2000个候选框 (使用选择性搜索)
  2. 对整张图片输进CNN，得到feature map
  3. 找到每个候选框在feature map上的映射patch，将此patch作为每个候选框的卷积特征输入到SPP layer和之后的层
  4. 对候选框中提取出的特征，使用分类器判别是否属于一个特定类
  5. 对于属于某一特征的候选框，用回归器进一步调整其位置
- **Faster RCNN**
  1. 对整张图片输进CNN，得到feature map
  2. 卷积特征输入到RPN，得到候选框的特征信息
  3. 对候选框中提取出的特征，使用分类器判别是否属于一个特定类
  4. 对于属于某一特征的候选框，用回归器进一步调整其位置

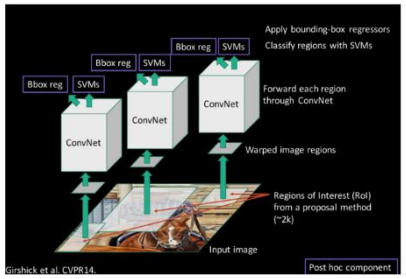


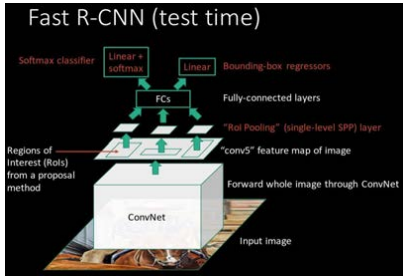


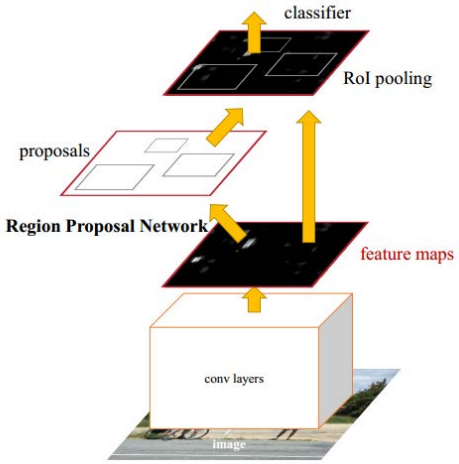


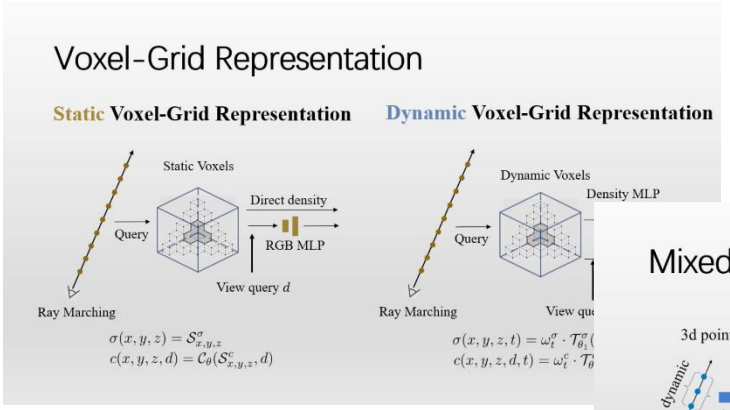
应用

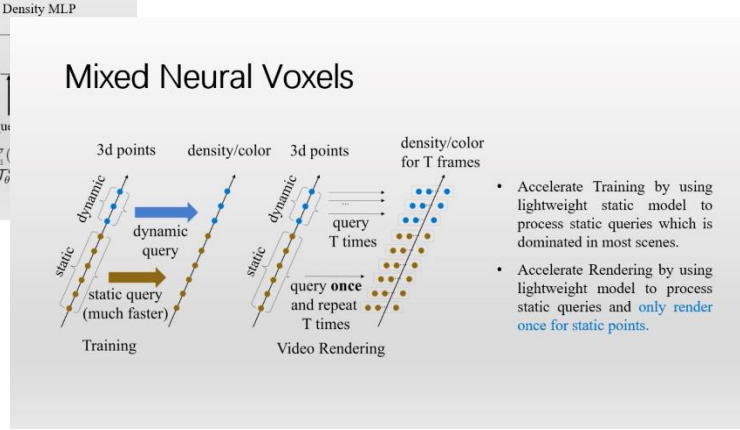
➢ 目标检测：扩展



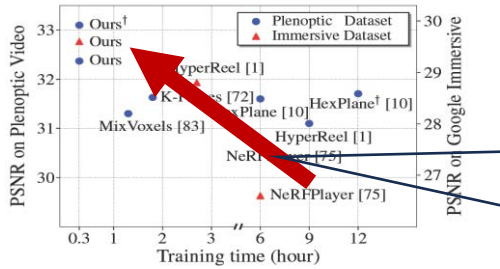








动态场景的三维重构



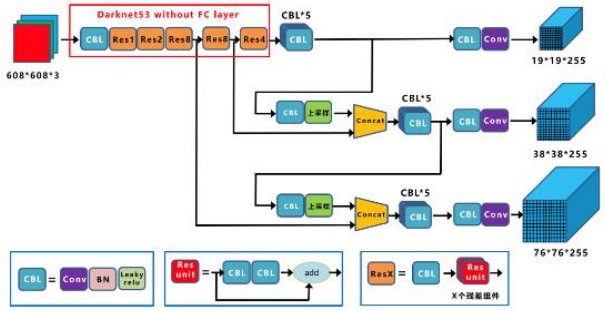
动态场景：渲染时间从12小时提升至20分钟，画质更高

- Mixed neural voxels for fast multi-view video synthesis, **ICCV** , 2023 , **Oral**
- Masked space-time Hash encoding for efficient dynamic scene reconstruction, **NeurIPS**, 2023, **Spotlight**

应用

➢ 目标检测：发展

Yolo



使用one-stage的方式，直接通过一系列CBL模块（Conv-Batch Normalization-Leaky Relu）提出特征金字塔，并在此基础上根据预先设置的anchor prior对各region进行分类和回归

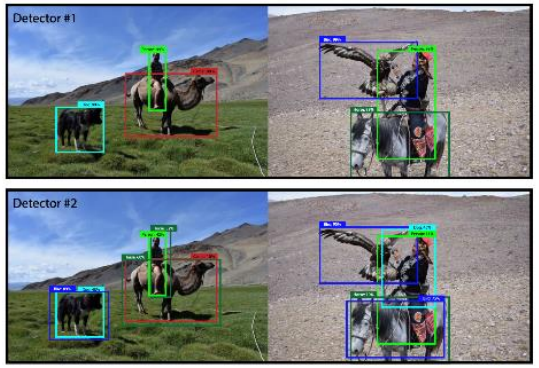
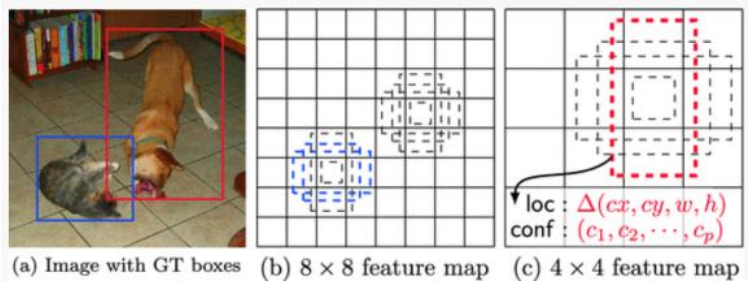


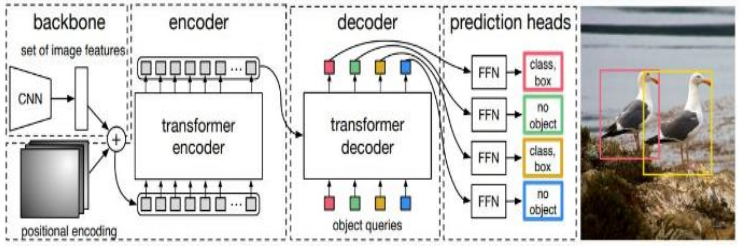
Figure 5. These two hypothetical detectors are perfect according to mAP over these two images. They are both perfect. Totally equal.

SSD

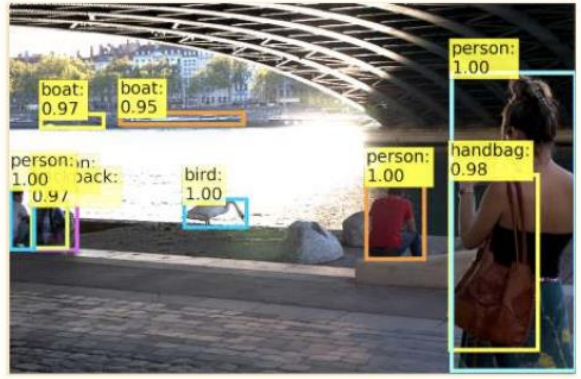


SSD是one-stage检测器，其速度要优于两阶段检测器，但是精度上与有些许损失。SSD较其他单阶段检测器相比其最大的亮点在于提出利用浅层特征检测小目标而深层特征检测大目标。

DeTR



将CNN的feature map拉平后视为一个序列，并通过transformer encoder进行编码。之后通过一些可学习的位置编码作为decoder的输入，将解码后的各vector送入到前向传播网络中，对其类别和包围盒进行预测和回归



➢ 目标检测：发展

YOLO-V5: https://github.com/ultralytics/yolov5，最新的YOLO检测器地址，其安装和推理十分简便

▼ Inference

Inference with YOLOv5 and PyTorch Hub . Models download automatically from the latest YOLOv5 release.

```
import torch

# Model
model = torch.hub.load('ultralytics/yolov5', 'yolov5s')  # or yolov5m, yolov5l, yolov5x, custom

# Images
img = 'https://ultralytics.com/images/zidane.jpg'  # or file, Path, PIL, OpenCV, numpy, list

# Inference
results = model(img)

# Results
results.print()  # or .show(), .save(), .crop(), .pandas(), etc.
```

应用

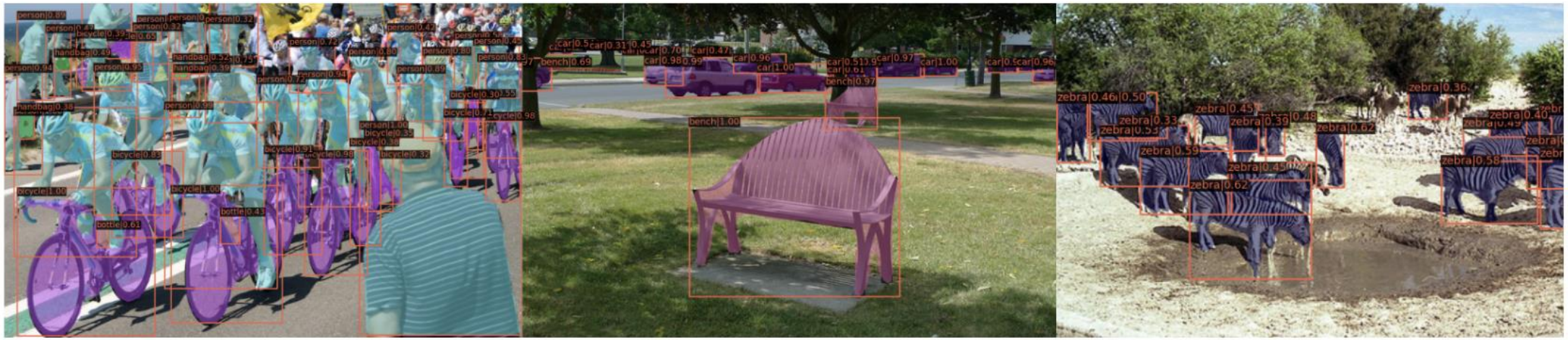
➢ 目标检测：发展

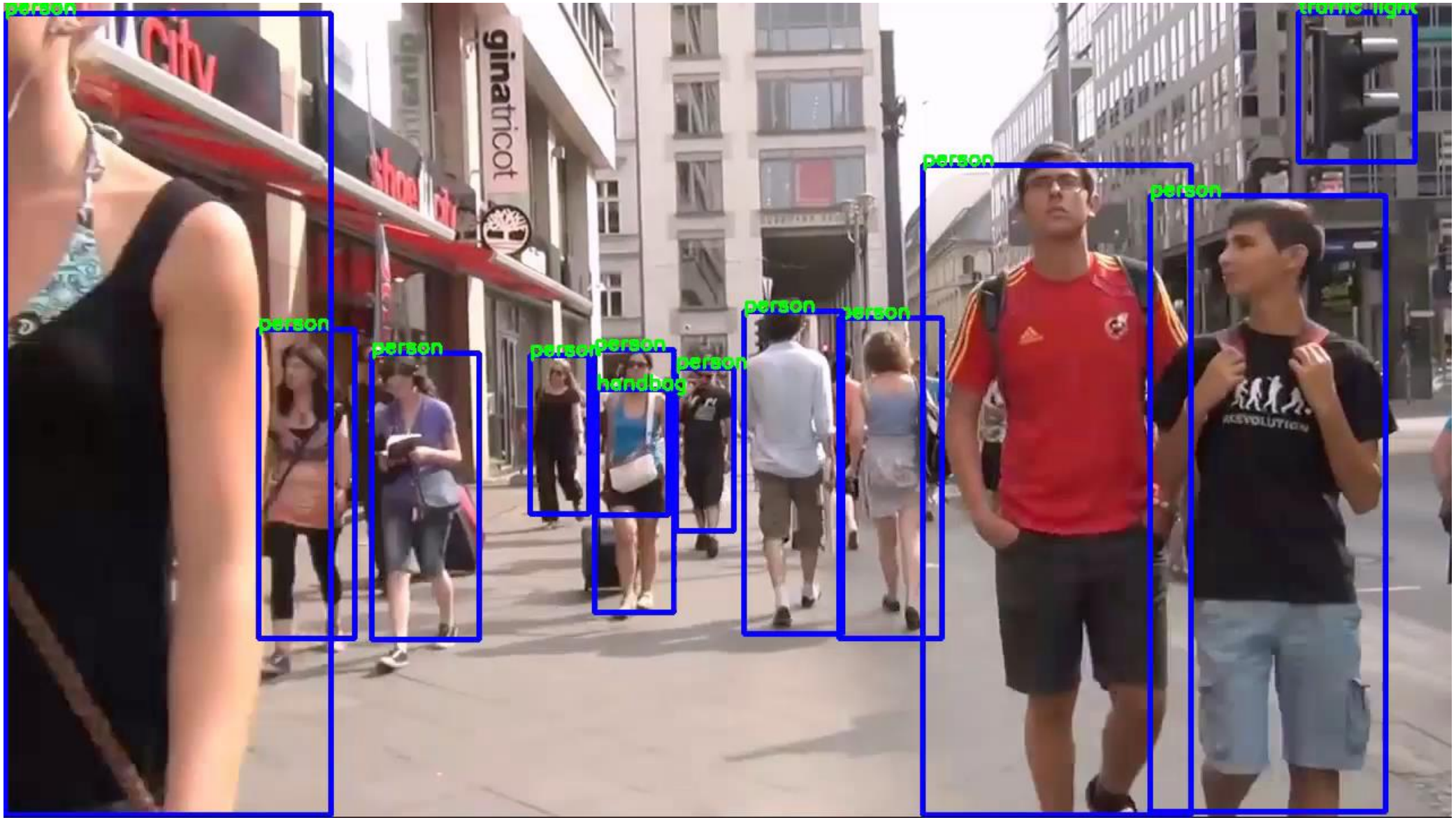
Faster RCNN: https://github.com/jwyang/faster-rcnn.pytorch，通过pytorch实现的faster rcnn，需要先通过C++编译目标检测中常用的函数（如RoIPooling、NMS等），之后进行训练。Backbone网络包括VGG-16和ResNet-101可以使用。

应用

➢ 目标检测：发展

mmdetection: https://github.com/open-mmlab/mmdetection
mmdetection是通过pytorch实现的多个object detector的开源库，可通过制定config文件和模型参数来通过init_detector方法加载不同的detector，并通过inference_detector方法来对image进行目标检测。




person
City
ginatricot
person
person
person
person
handbag
person
person
person
person
person
traffic light

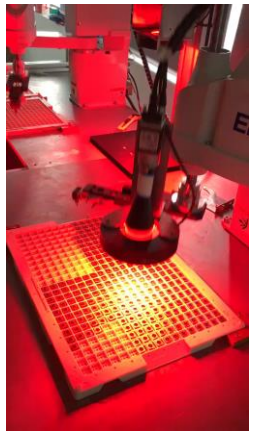
应用

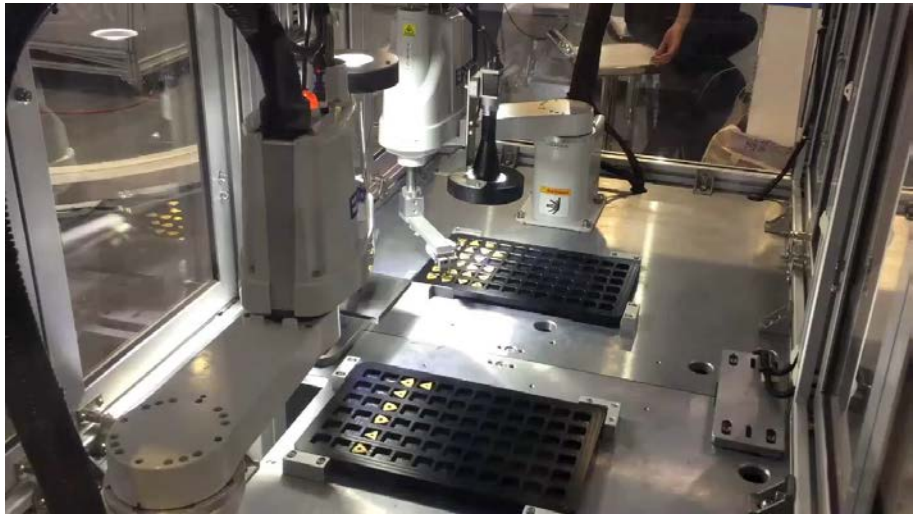
➢ 目标检测

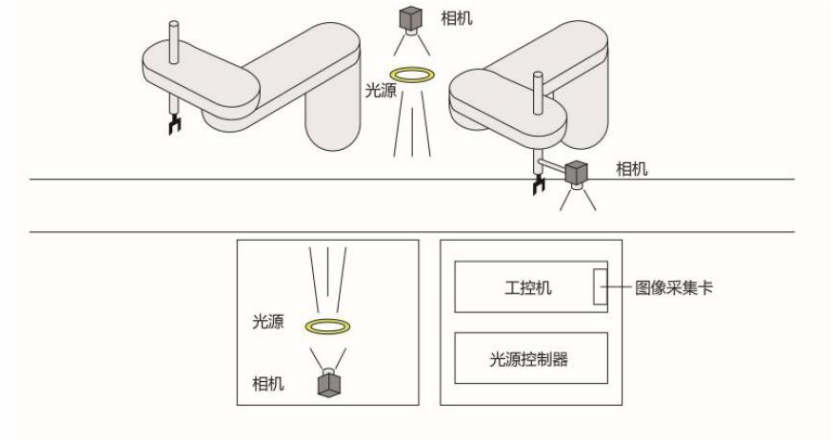
应用

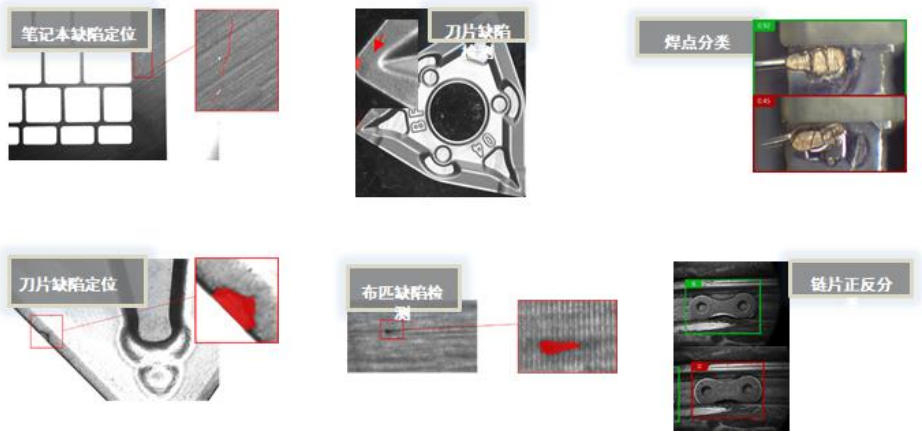
➢ 目标检测

具有持续自学习能力的智能视觉缺陷检测系统



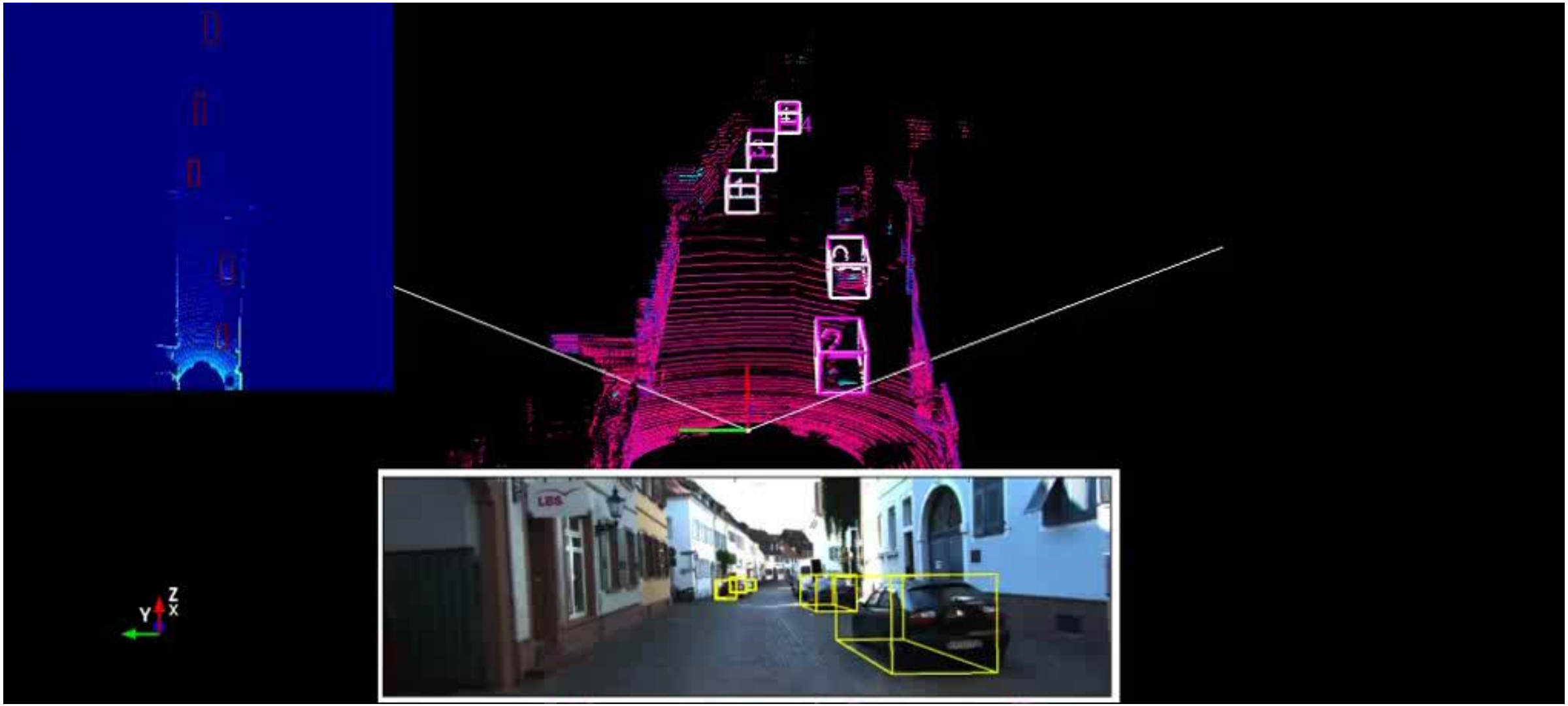






应用

➢ 目标检测：多模态



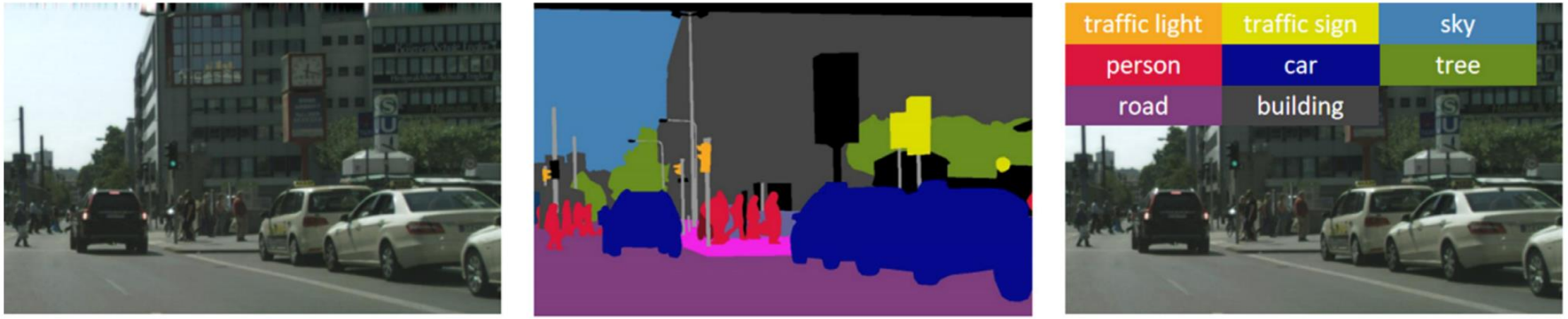
应用

➢ 目标检测：多模态

- 传统方法
- 机器学习

应用

➢ 语义分割

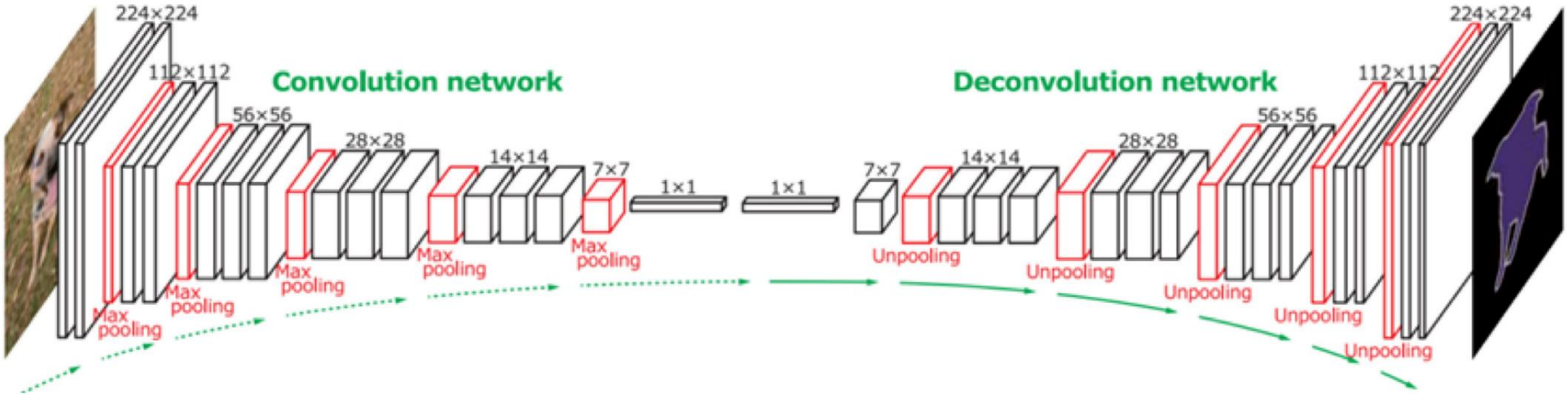


(a) Image　(b) Semantic segmentation　(c) Image classification

语义性是图像中很重要的性质，准确的语义信息可以提升许多下游任务的表现。例如机器人的操作（manipulation），导航（navigation），图像/视频的问答（Visual / Video QA），图像/视频的标注（Visual / Video captioning）等等。并且，最近用于自监督预训练的对比学习（contrastive learning）也是通过contrastive loss来帮助模型学习图像的语义性。因此，语义分割是计算机视觉中十分重要的应用，且语义分割的好坏可以一定程度上反应由网络抽取的视觉特征的好坏。

应用

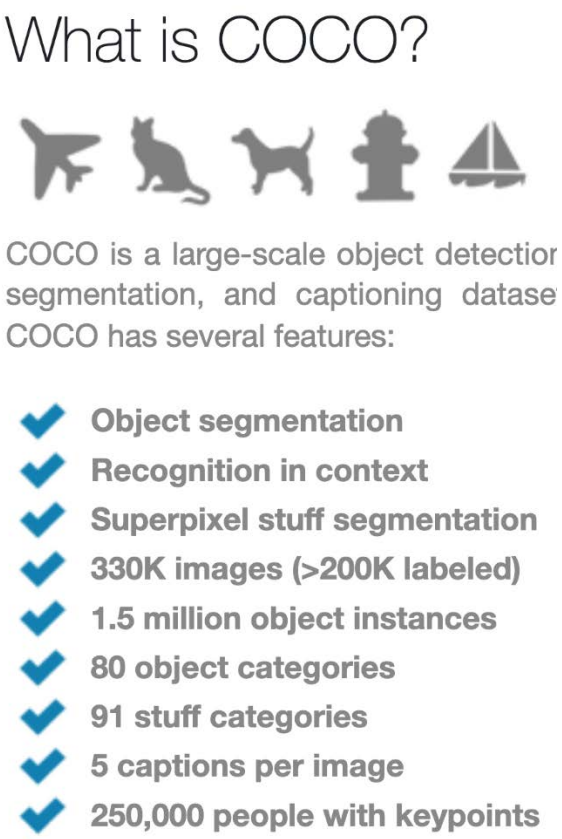
➢ 语义分割



经典的语义分割由一系列down sampling和up sampling组成。如上图所示，原始的224x224的图像先通过一系列卷积层（convolution layer）池化层（max pooling）得到了2048x1x1的向量。之后，通过一系列对称的反卷积操作将该向量上采样得到224x224的张量。并对其中每个像素点进行回归，来让网络学习每个pixel对应的semantic label。
也可以在一系列下采样后直接在feature map上进行预测，并对预测的结果进行差值还原到与原始图像相同的尺寸。

应用

➢ 语义分割



https://cocodataset.org/#home

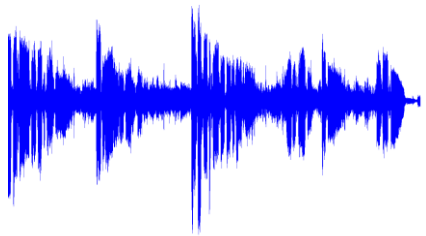
COCO dataset 是微软构建的一个数据集，其中包含了80类为物体，33万张图片，可以支持object detection、semantic segmentation等任务。

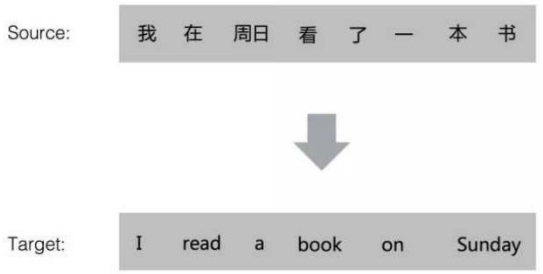
每个Image不但提供了各类物体的label和bounding box坐标，还额外给出了mask信息，即bounding box中哪些像素点是该物体。

2010: Speech Recognition

2012: Computer Vision

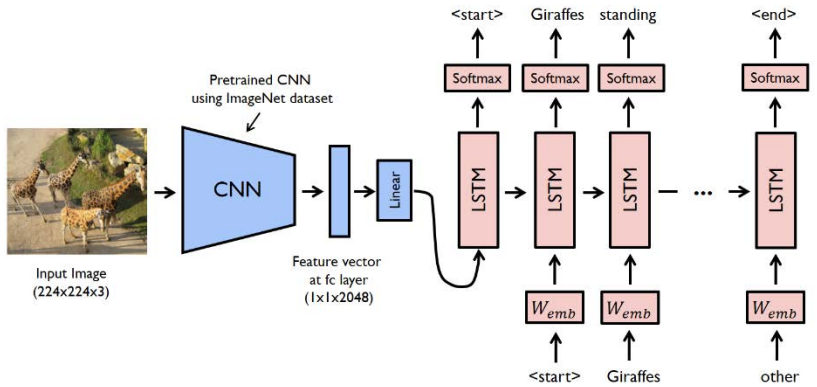
2014: Machine Translation

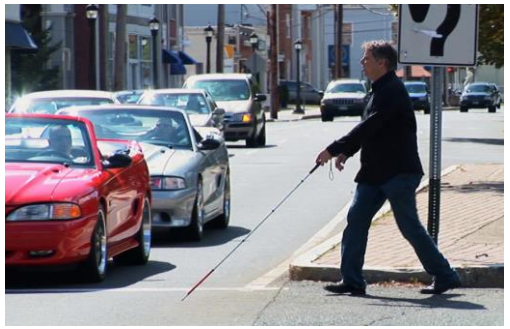




应用

➢ 语言生成





前方道路有车辆行驶，请等待

谢　谢！