

Convolutional Neural Networks (CNNs)

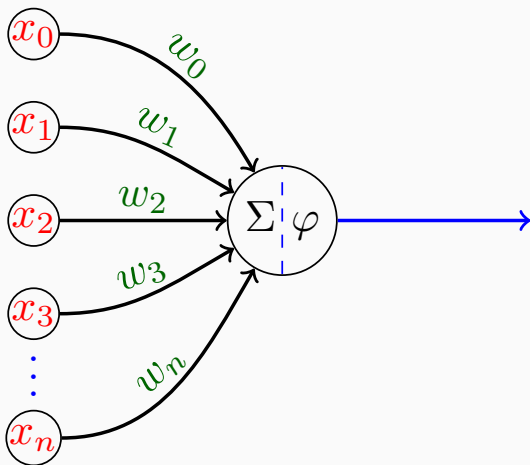
Theory and Applications

Martin Thoma – tinyurl.com/CNN-Intro

22. February 2019

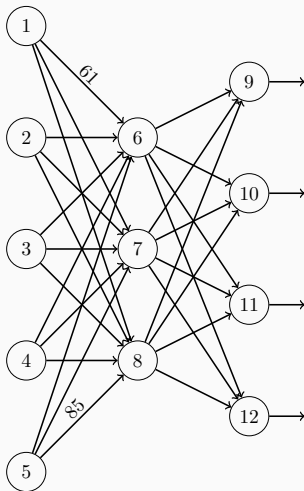
Artificial Neuron (Perceptron)

$$f : \mathbb{R}^n \rightarrow \mathbb{R}$$



Multi-Layer Perceptron (MLP)

$$f : \mathbb{R}^n \rightarrow \mathbb{R}^m$$



- Predict housing prices: (bed rooms, size, age) $\rightarrow$ Price

- Predict housing prices: (bed rooms, size, age) $\rightarrow$ Price
- Product categorization: (weight, volume, price) $\rightarrow$ {shoe, handbag, shirt}

- Predict housing prices: (bed rooms, size, age) $\rightarrow$ Price
- Product categorization: (weight, volume, price) $\rightarrow$ {shoe, handbag, shirt}
- Image classification: List of pixel colors $\rightarrow$ {cat, dog}

Data

Necessary Data

- $f(x) = w_0$
- $f(x) = w_1 \cdot x + w_0$
- $f(x) = w_2^2 \cdot x^2 + w_1^2 \cdot x + w_0$
- $\sin, \cos, \tan, \dots$

Convolution

$$I \in \mathbb{R}^{7 \times 7}$$

104	116	116	112	58	47	47
109	97	114	116	105	110	45
116	104	111	109	97	46	100
101	47	109	97	115	116	101
114	47	99	97	116	99	97
116	99	97	116	46	112	104
112	63	118	61	49	46	48

9	-3	-1
-6	5	3
2	-8	0

Filter kernel
 $F \in \mathbb{R}^{3 \times 3}$

936	-333	-109
-282	545	291
94	-792	0

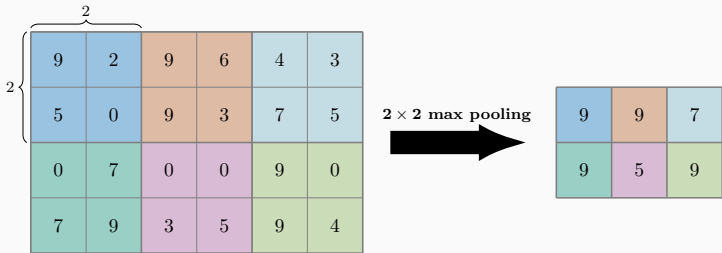
Result of point-
wise multiplication

$$I' \in \mathbb{R}^{7 \times 7}$$

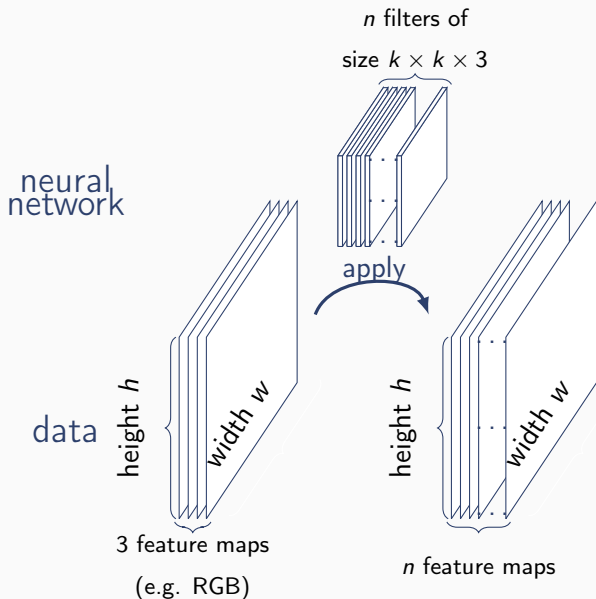
-4	-254	-498	-662	-849	-642	187
-520	45	240	211	388	215	-861
-340	559	-105	185	-138	-180	503
-718	429	350	173	251	268	-655
-567	-53	-75	80	571	-128	24
-408	596	-550	368	26	976	156
302	647	879	223	811	54	660

martin-thoma.com/graphic-filters

Max Pooling



Convolutional Layer

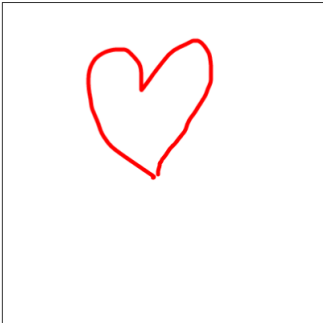


Applications

Symbol recognizer

[Write Math](#) [Classify](#) [Train](#) [Gallery](#) [Ranking](#) [About](#) Martin Thoma

Classify



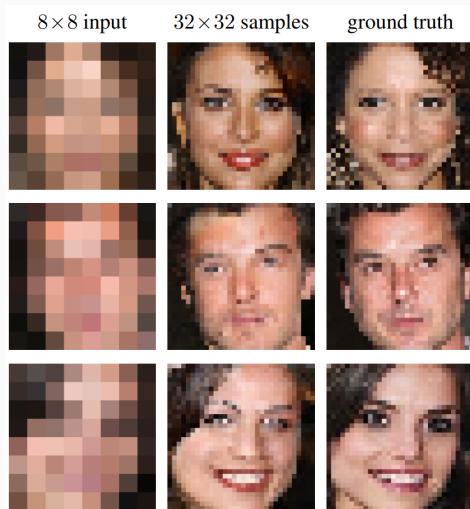
[Get PNG](#) [Drawing](#) [Clear](#)

α	LaTeX	%	vote
♥	\heartsuit	99.98	☒
♣	\clubsuit	0.01	☒
◯	\fullmoon	0.00	☒
♂	\male	0.00	☒
▼	\triangledown	0.00	☒
∇	\nabla	0.00	☒
◇	\lozenge	0.00	☒
☉	\astrosun	0.00	☒

write-math.com

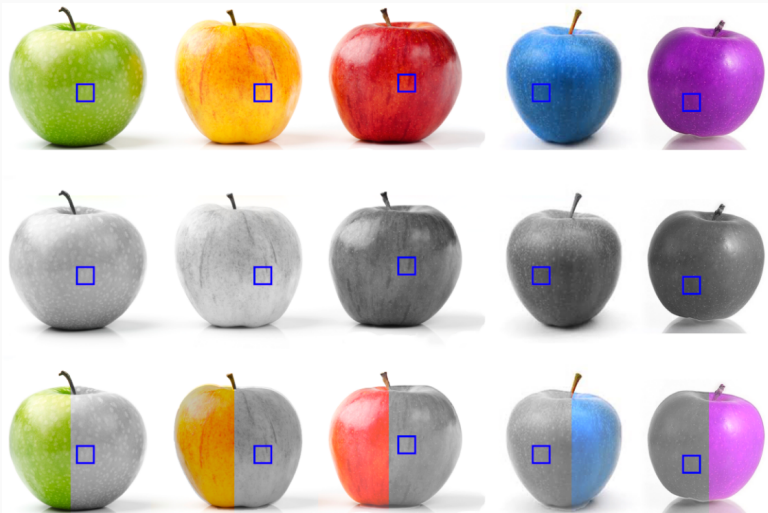
```
1 import data
2
3 from keras.layers import Dense, Flatten, Conv2D, MaxPooling2D
4 from keras.models import Sequential, load_model
5
6 model = Sequential()
7 model.add(Conv2D(16, (3, 3)))
8 model.add(MaxPooling2D(pool_size=(2, 2)))
9 model.add(Conv2D(16, (3, 3)))
10 model.add(Flatten())
11 model.add(Dense(128, activation='relu'))
12 model.add(Dense(data.n_classes, activation='softmax'))
13
14 model.compile(loss='categorical_crossentropy', optimizer='adam')
15 model.fit(data.x_train, data.y_train)
16
17 model.save('model.h5')
18 model = load_model('model.h5')
19 y_predicted = model.predict(data.x_test)
```

Super Resolution



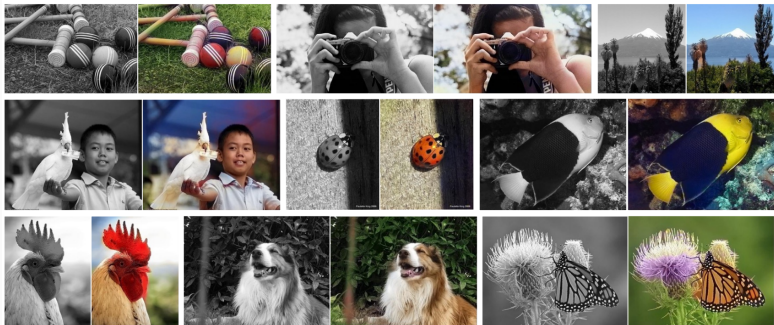
Dahl, Norouzi, Shlens: Pixel recursive super resolution (2017)

Colorization: The Problem



Cinarel: Automatic Colorization of Webtoons Using Deep Convolutional Neural Networks (2018)

Colorization - Photographs



Zhang, Isola, Efros: Colorful Image Colorization (2016)

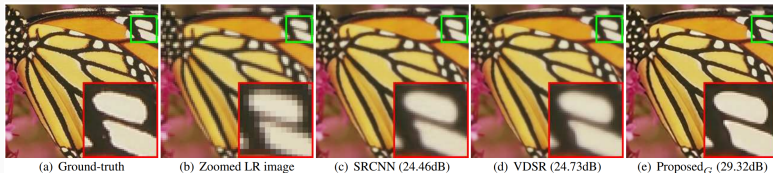
Interactive Demo: richzhang.github.io/colorization

Model Lab: github.com/MartinThoma/model-lab

Colorization - Comic

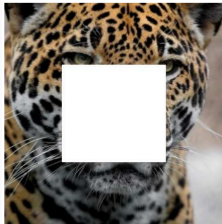


Ci, Ma, Wang, Li, Luo: User-Guided Deep Anime Line Art Colorization with Conditional Adversarial Networks (2018)

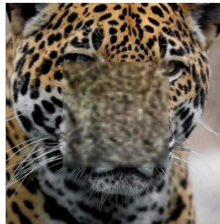


Zhang, Zuo, Gu, Zhang: Learning Deep CNN Denoiser Prior for Image Restoration (2017)

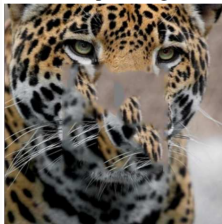
Image Inpainting (Watermark removal)



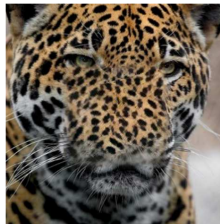
(a) Input Image



(b) Context Encoder



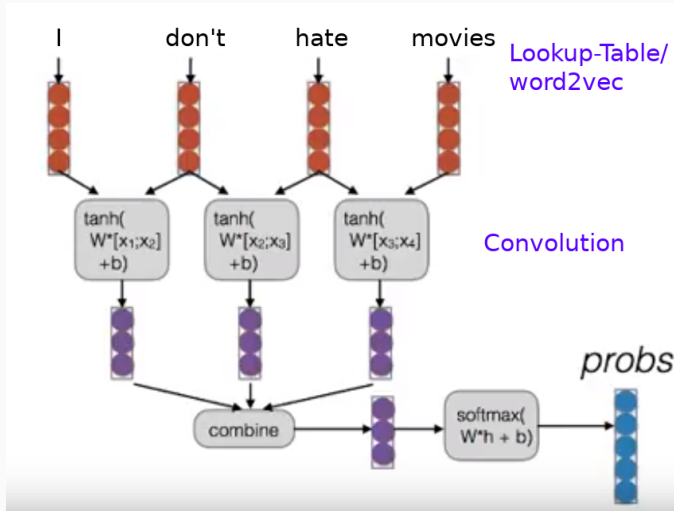
(c) PatchMatch



(d) Our Result

Yang, Lu, Lin, Shechtman, Wang, Li: High-Resolution Image Inpainting using Multi-Scale Neural Patch Synthesis (2017)

CNNs in NLP



Collobert, Weston, Bottou, Karlen, Kavukcuoglu, Kuksa: Natural Language Processing (almost) from Scratch (2011)