Convolutional Neural Network

Data Workshop #7

dataworkshop.eu

Alekseichenko Vladimir

Vladimir Alekseichenko

biznesmysli.pl

Architect
General Electric

dataworkshop.eu

DATA
WORKSHOP

vova.me
slon1024
hello@vova.me

Disclaimer

Data Workshop *[all time]* focuses on the **intuition** and **practical** tips.

For a formal treatment, see something else.*

* papers or classical machine learning books

Environment

github.com/dataworkshop/**prerequisite**
github.com/dataworkshop/**environment**

github.com/dataworkshop/**cnn**

Packages

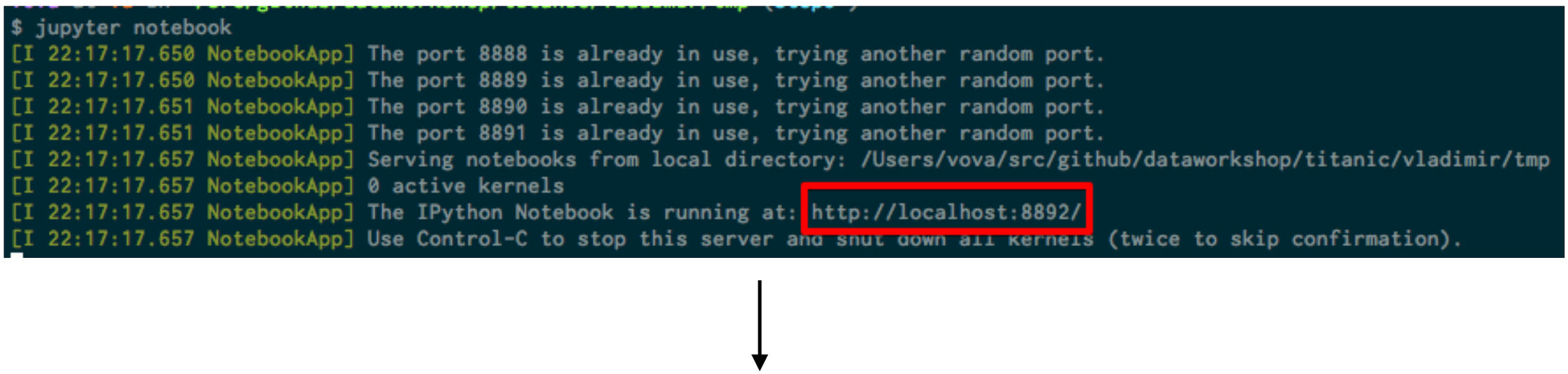
github.com/**dataworkshop/prerequisite**

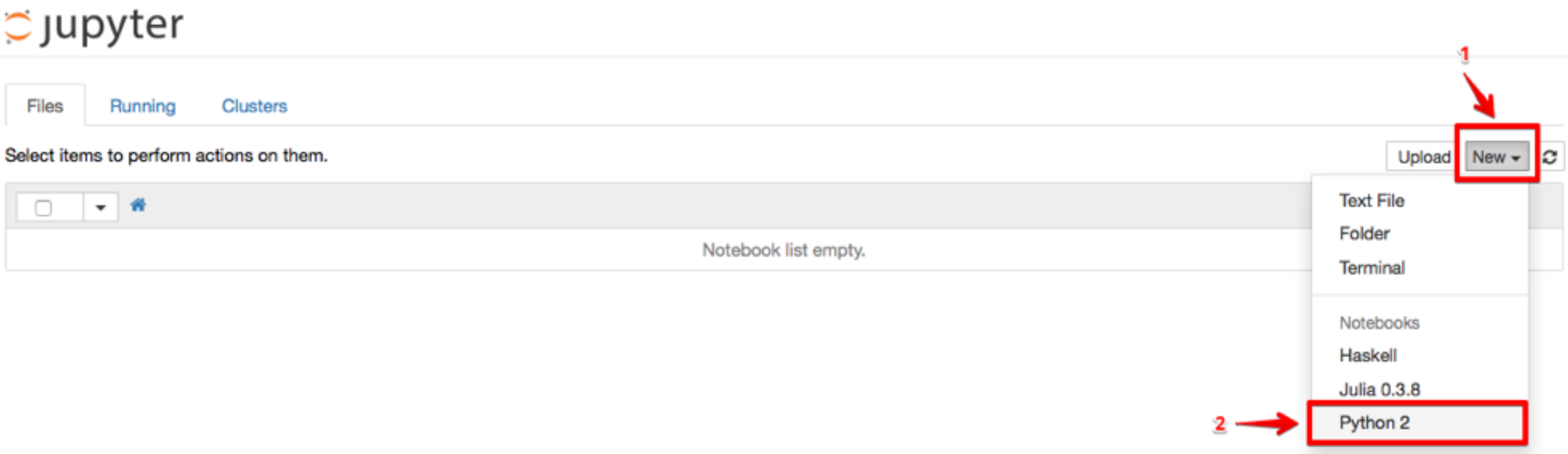
```
$ python run.py
seaborn-0.7.0 - OK
xgboost-0.4 - OK
matplotlib-1.5.1 - OK
IPython-4.1.2 - OK
numpy-1.11.0 - OK
pandas-0.18.0 - OK
sklearn-0.17.1 - OK

==================================================
All right, you are ready to go on Data Workshop!
```

```
$ python run.py
seaborn-0.6 should be upgraded to seaborn-0.7
xgboost-0.4 - OK
matplotlib-1.5.1 - OK
IPython-4.1.2 - OK
numpy-1.11.0 - OK
pandas-0.18.0 - OK
sklearn-0.17.1 - OK

==================================================
RECOMENDATION (without upgrade some needed features could be missing)
pip install --upgrade seaborn
```

```
$ python run.py
seaborn-0.7.0 - OK
xgboost - missing
matplotlib-1.5.1 - OK
IPython-4.1.2 - OK
numpy-1.11.0 - OK
pandas-0.18.0 - OK
sklearn-0.17.1 - OK

==================================================
REQUIRED
Please install those packages before Data Workshop: xgboost
pip install xgboost
More info how to install xgboost: http://xgboost.readthedocs.org/en/latest/build.html
```

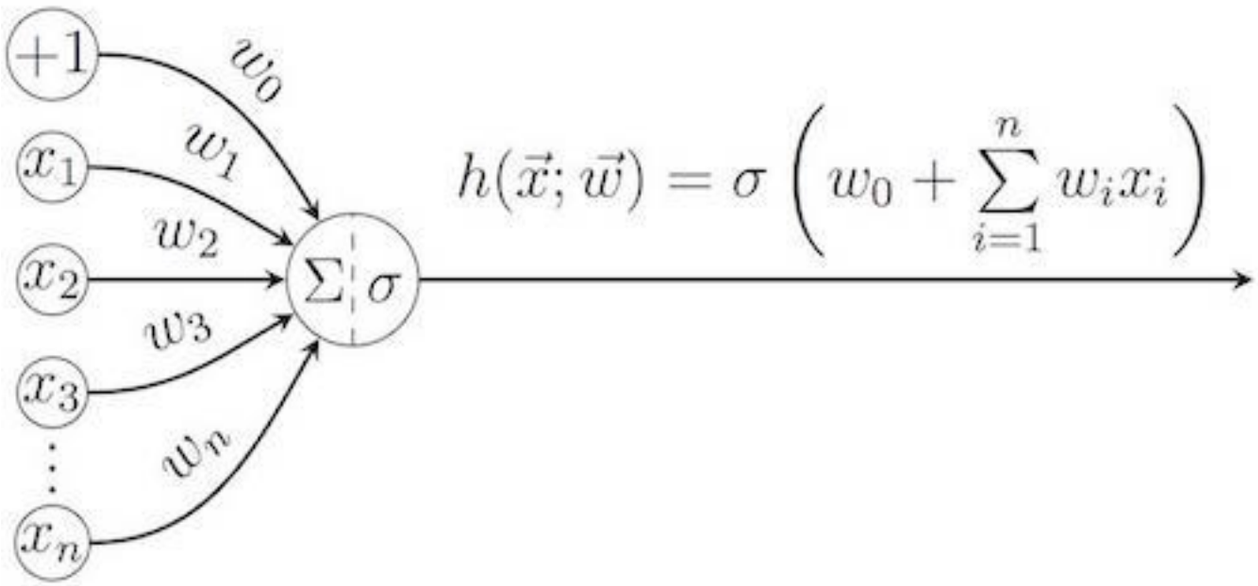
jupyter notebook





Motivation

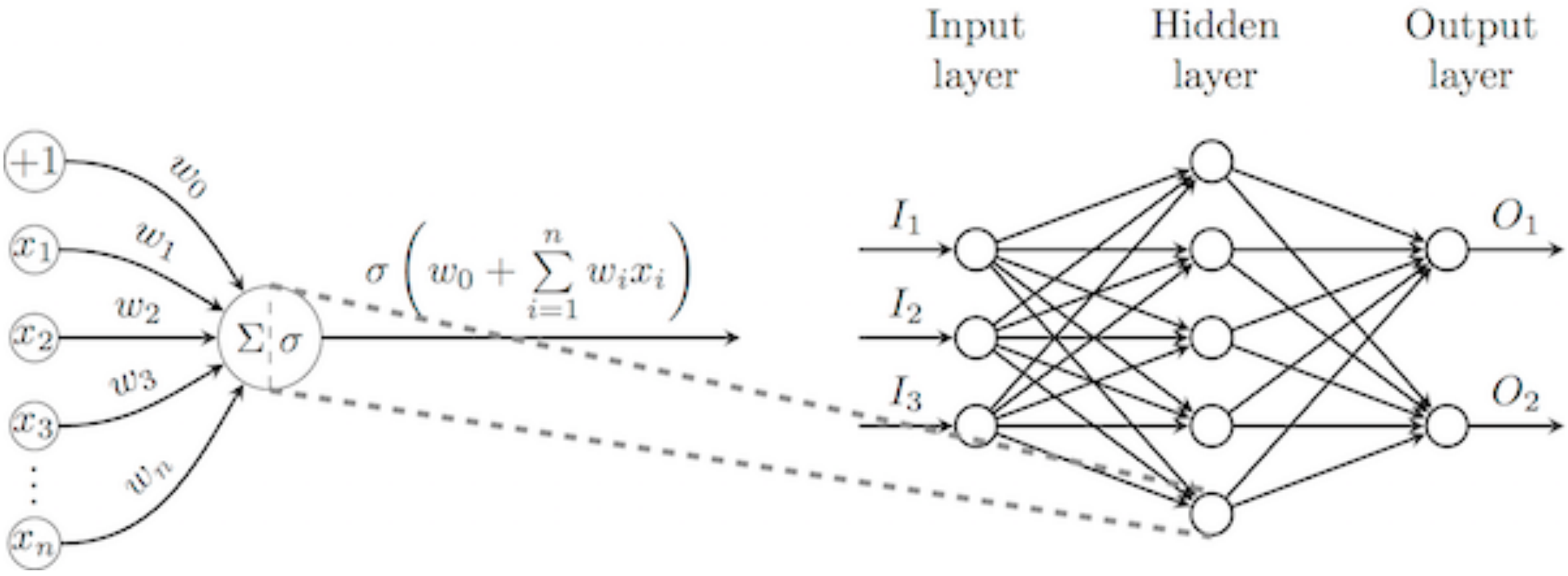
Neuron


+1
x_1
x_2
x_3
x_n
w_0
w_1
w_2
w_3
w_n
Σ
σ
$h(\vec{x};\vec{w}) = \sigma\left(w_0 + \sum_{i=1}^{n} w_i x_i\right)$

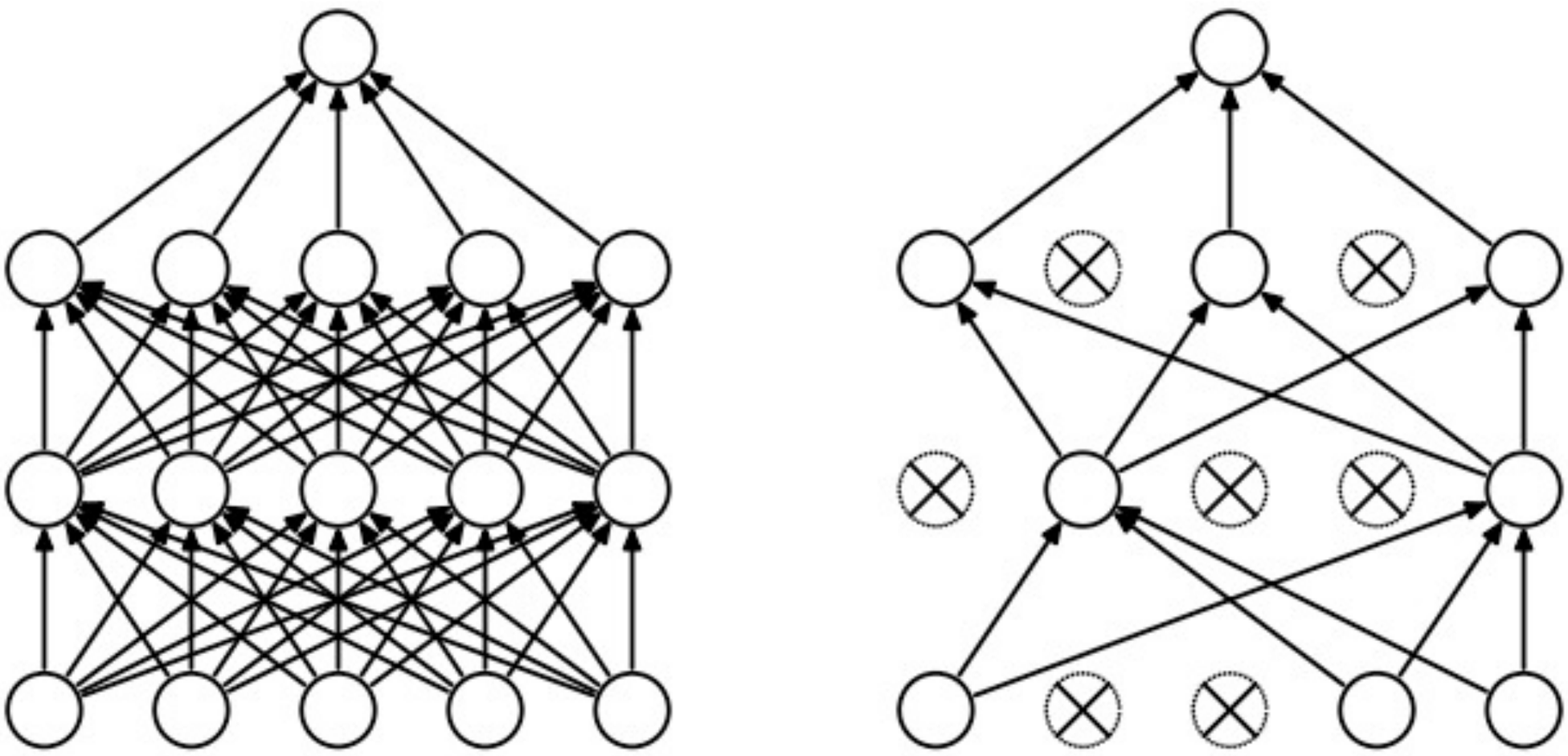
Activation function

- Tanh
- ReLU
- PRELU
- Sigmoid
- SoftMax (normalized exponential function)

(Artificial) Neural Network



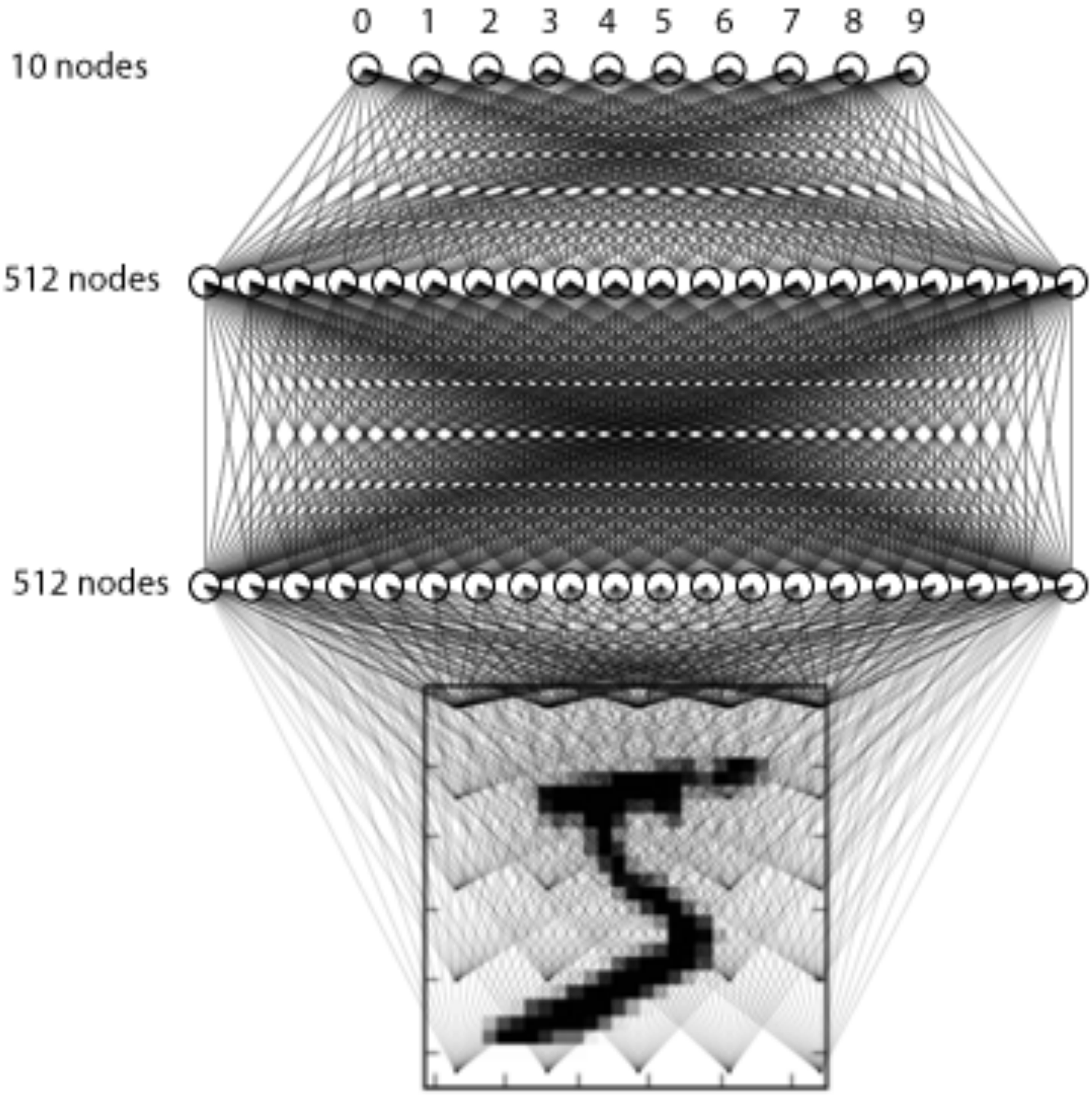
Dropout



(a) Standard Neural Net

(b) After applying dropout.

http://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf

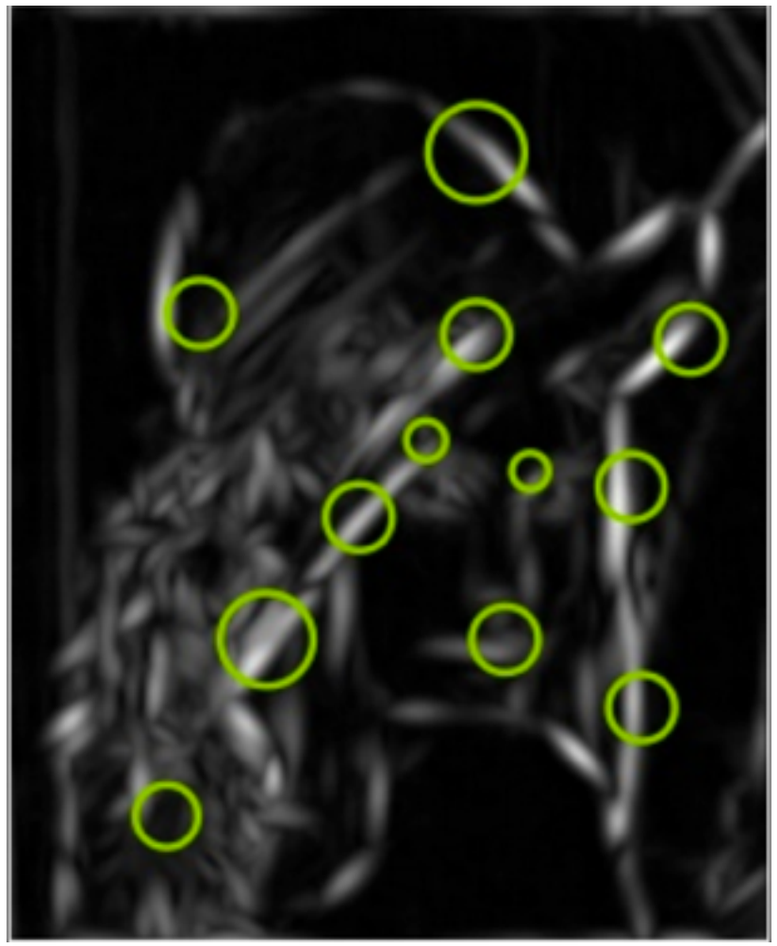

0 1 2 3 4 5 6 7 8 9
10 nodes
512 nodes
512 nodes

MNIST

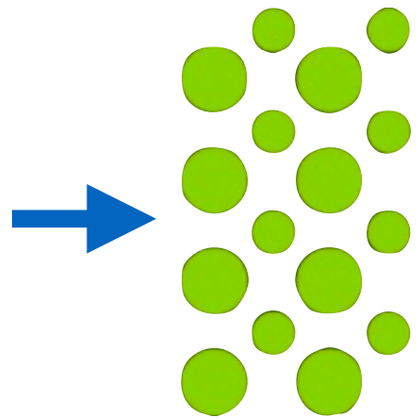
Computer Vision

Original

Simplified



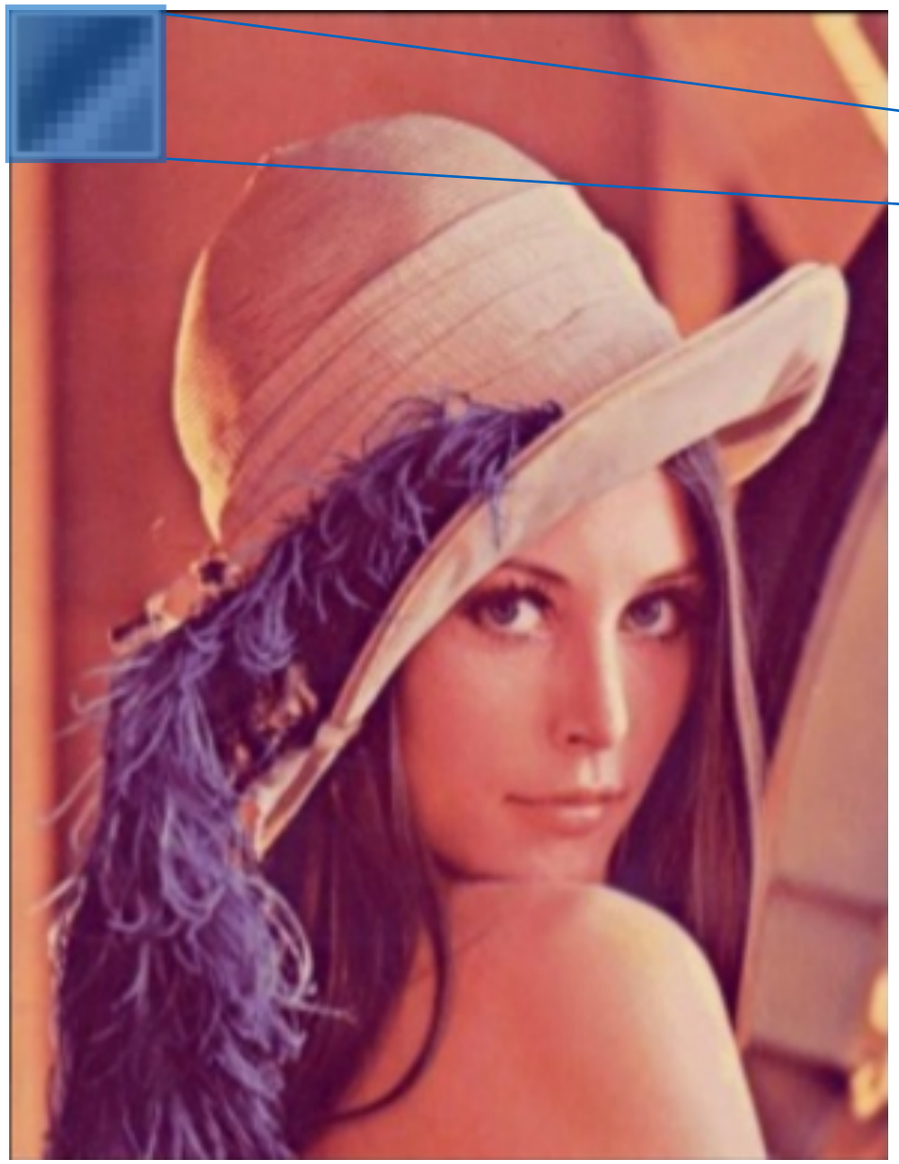
Features



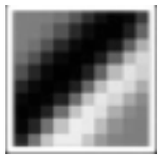
“Lenna"

Feature Map

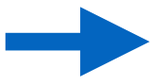
X



W



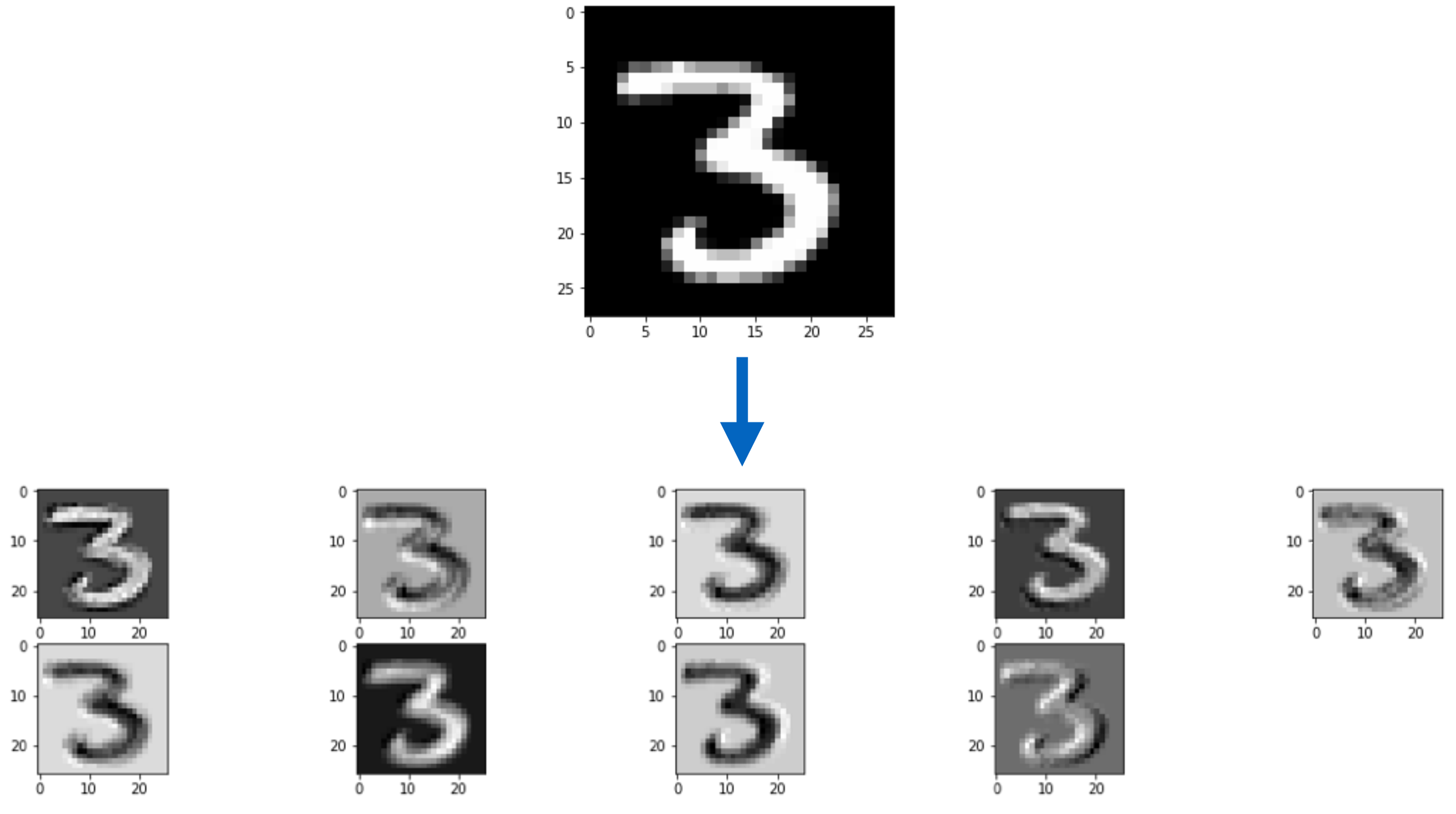
*



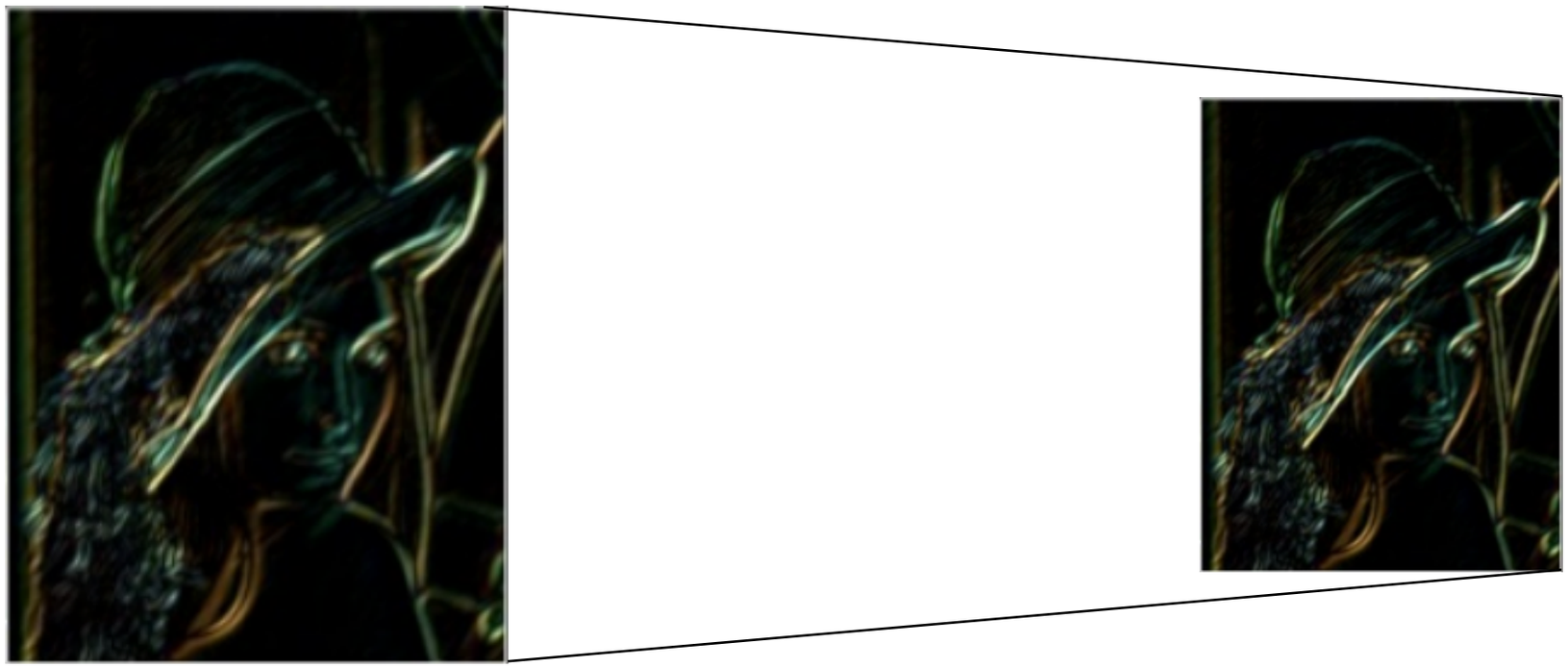
Feature Maps

*

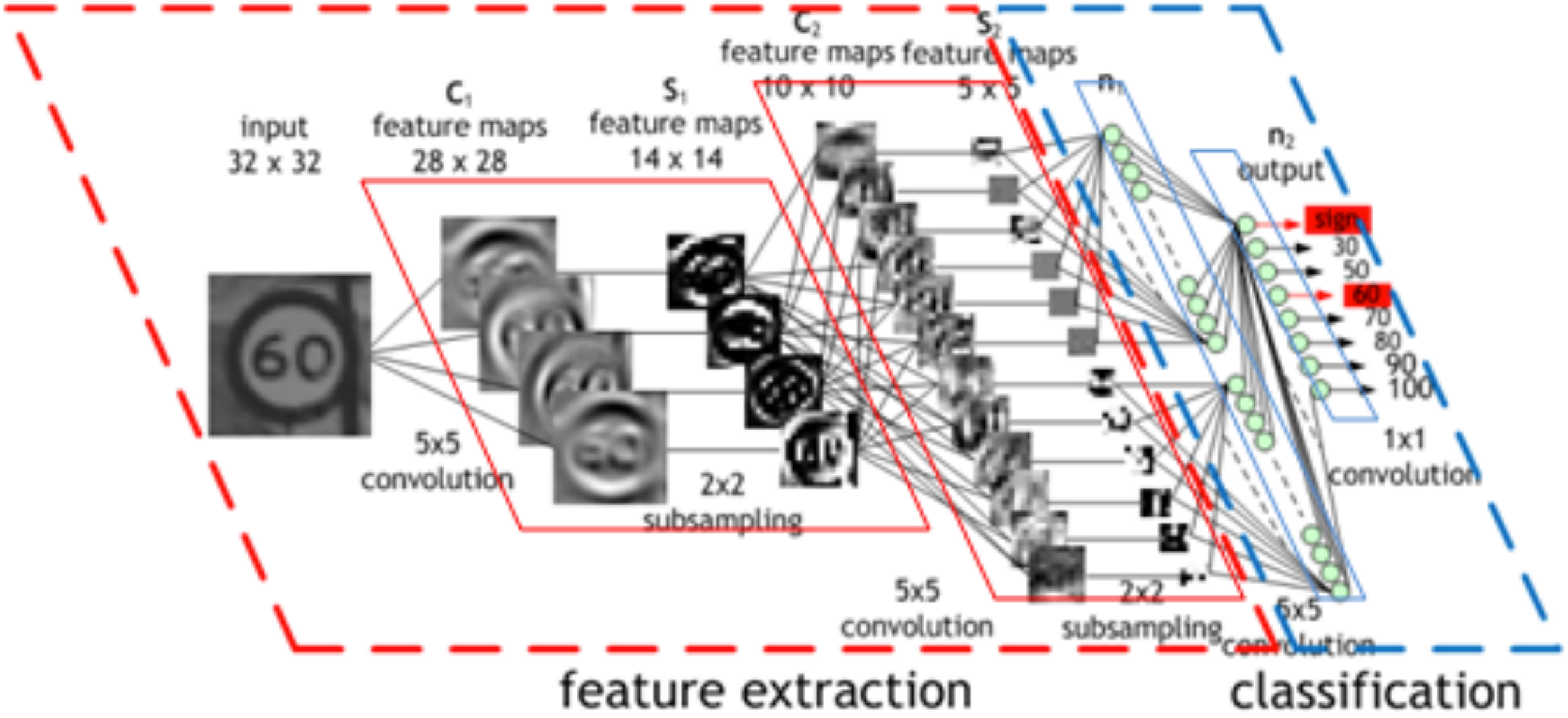
Feature Maps

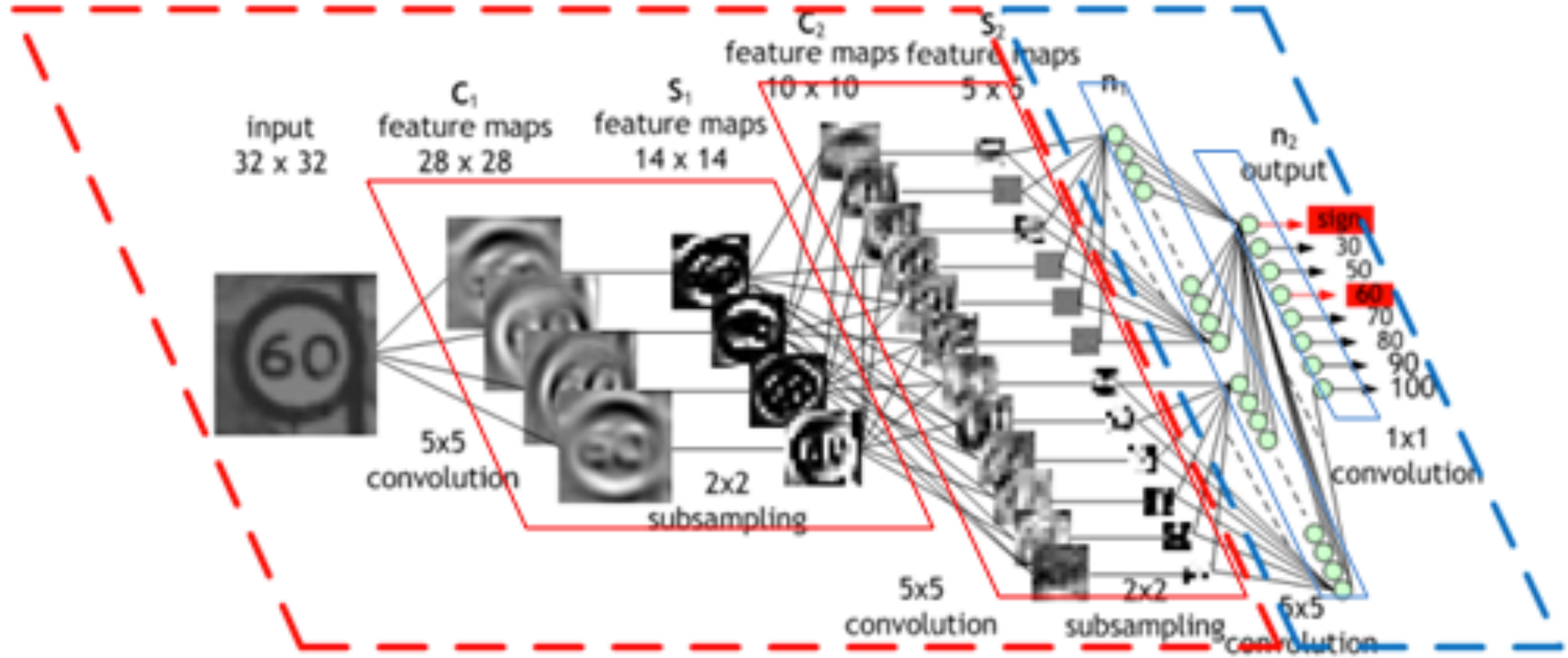


Pooling == resize



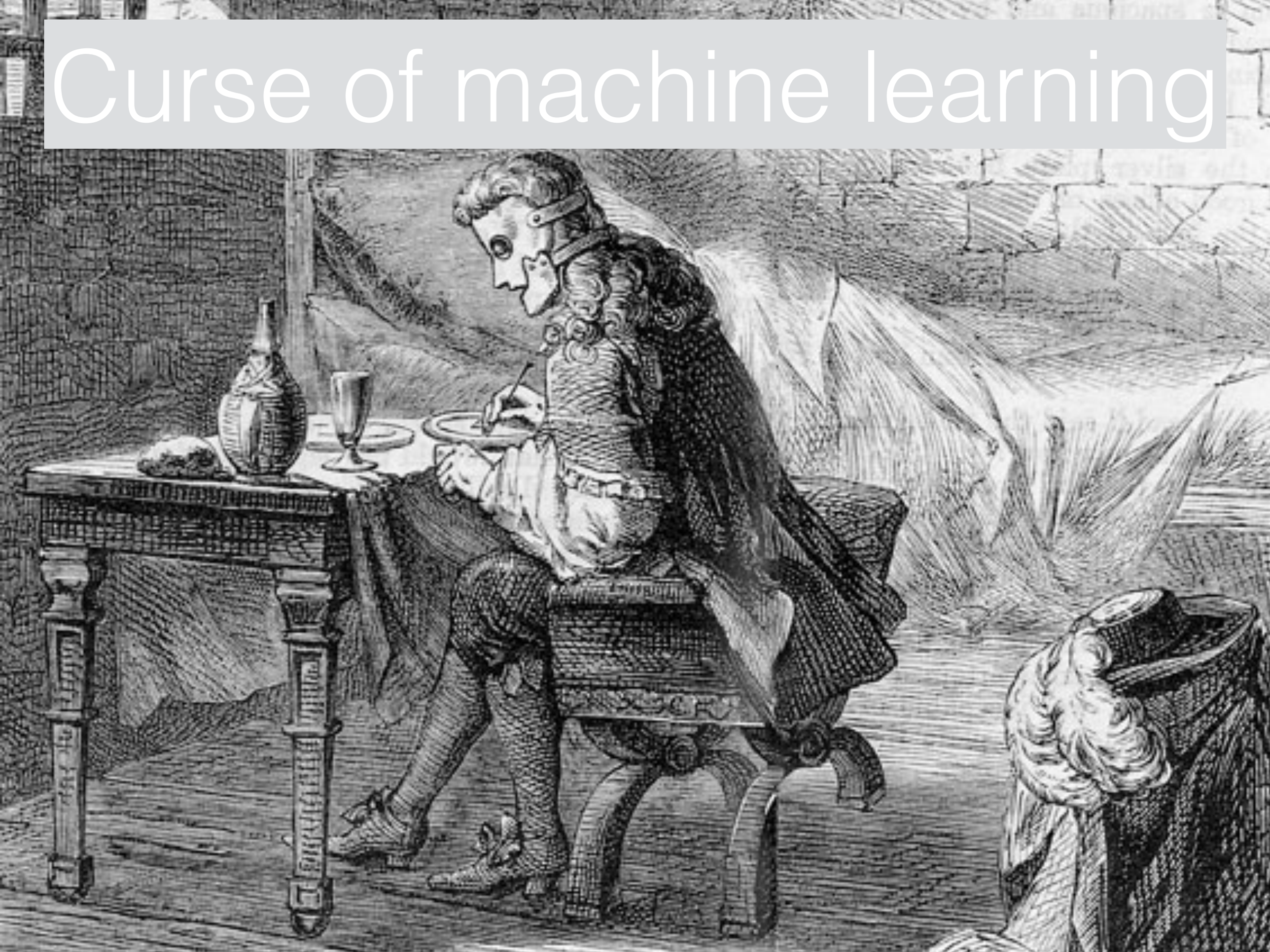
Convolutional Neural Networks






```
model = Sequential()

model.add(Conv2D(4, kernel_size=(5, 5), activation='relu', input_shape=(1, 32, 32)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(12, (5, 5), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dense(100, activation='softmax'))

model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adadelta(),
              metrics=['accuracy'])
```

Curse of machine learning



Reuse knowledge

Transfer Learning

Lifelong Learning

Gradual Learning

Curriculum Learning

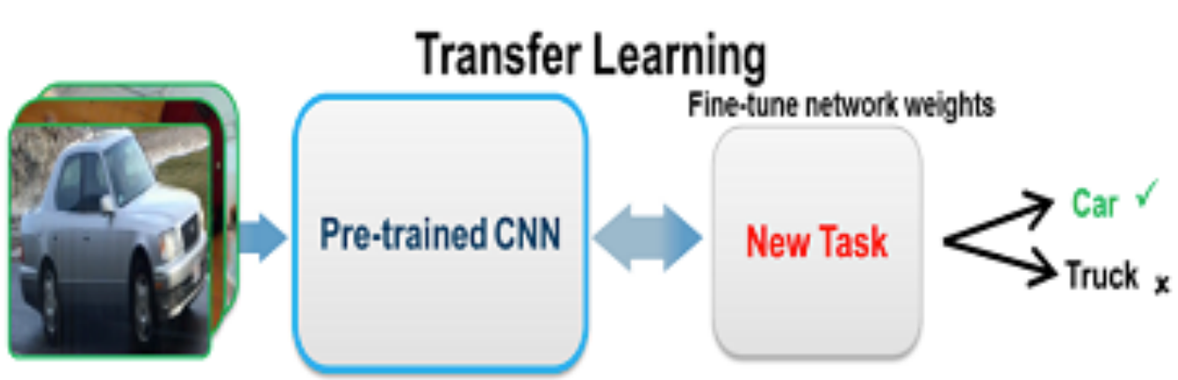
Compositional Learning

Continual Learning

Sequential Learning

Never-Ending Learning

Transfer Learning & Fine-tuning

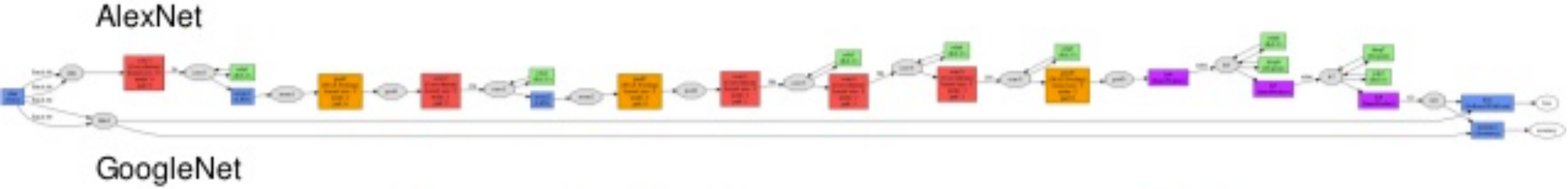


Training data	1000s to millions of labeled images
Computation	Compute intensive
Training Time	Days to Weeks for real problems
Model accuracy	High (can over fit to small datasets)

Training data	100s to 1000s of labeled images (small)
Computation	Moderate computation
Training Time	Seconds to minutes
Model accuracy	Good, depends on the pre-trained CNN model

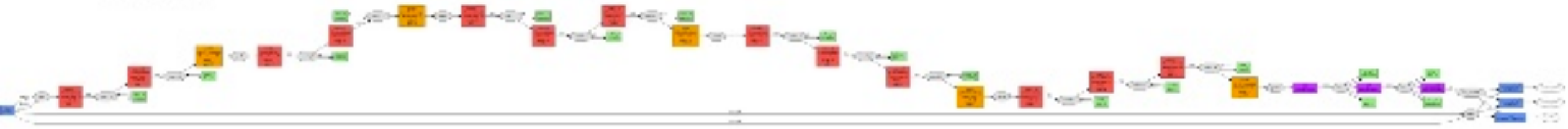
Known Topology - ImageNet

AlexNet



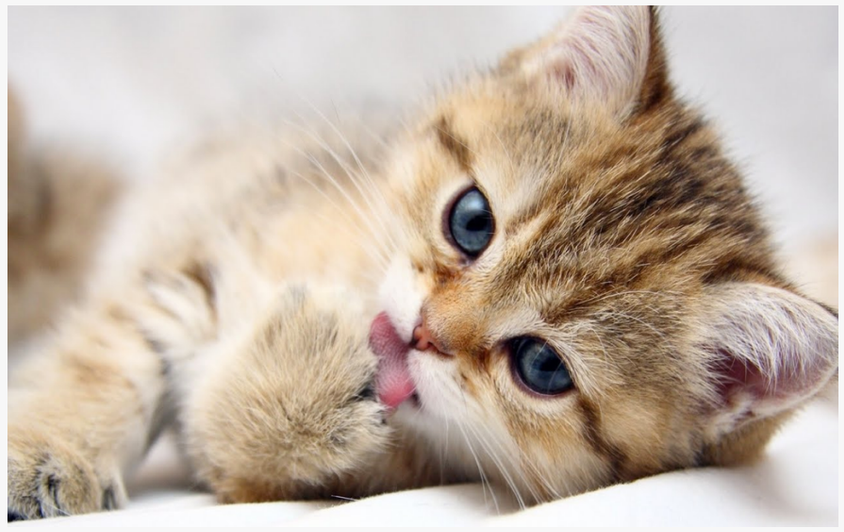
GoogleNet

VGG-16



ResNet-50

```python
from keras.applications.resnet50 import ResNet50
from keras.preprocessing import image
from keras.applications.resnet50 import preprocess_input, decode_predictions
import numpy as np

model = ResNet50(weights='imagenet')

img = image.load_img('cat.jpg', target_size=(224, 224))
x = np.expand_dims(image.img_to_array(img), axis=0)
x = preprocess_input(x)

preds = model.predict(x)
print('Predicted:', decode_predictions(preds, top=3)[0])
```



```
('Predicted:', [(u'n02123045', u'tabby', 0.47442555), (u'n02127052', u'lynx', 0.39875993),
 (u'n02124075', u'Egyptian_cat', 0.048101209)])
```

Tabby 47.4%

Lynx 39.9%

Egyptian cat 4.8%

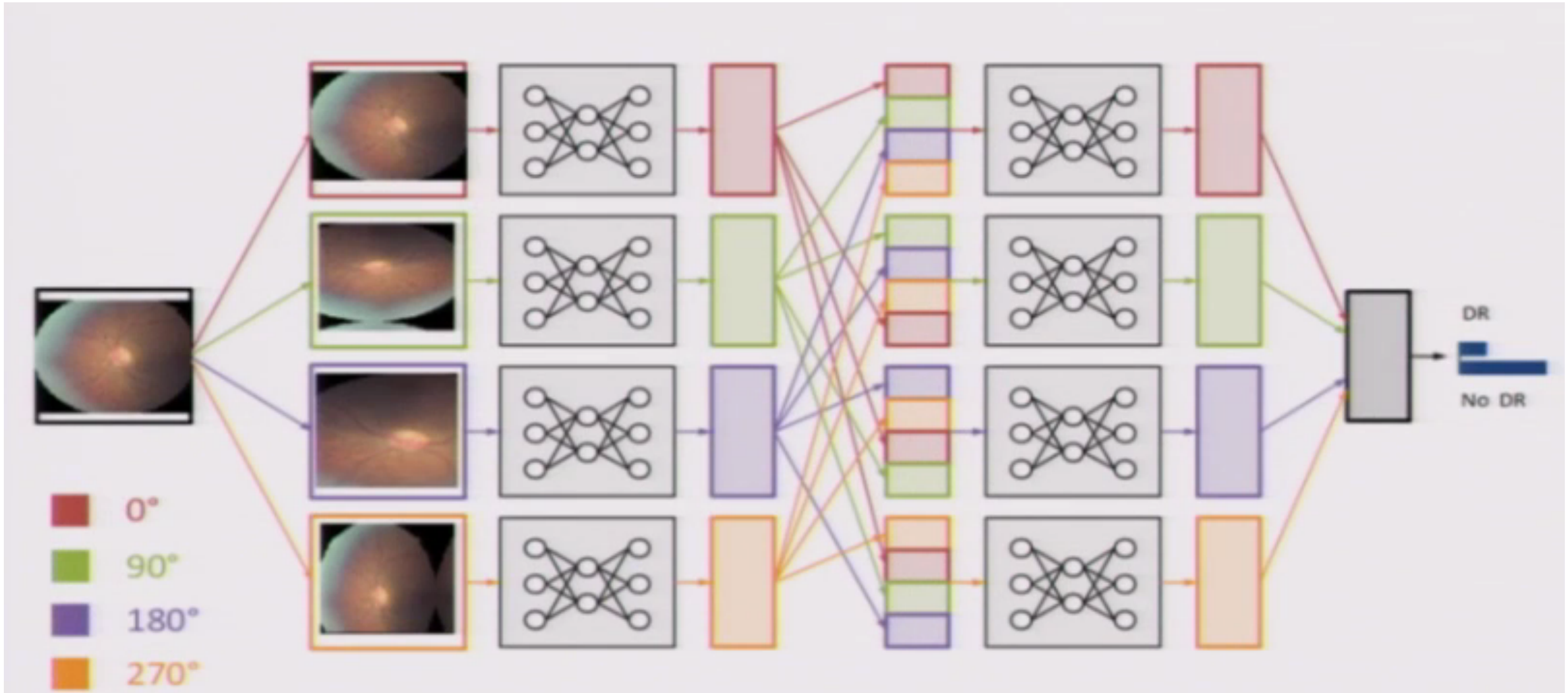


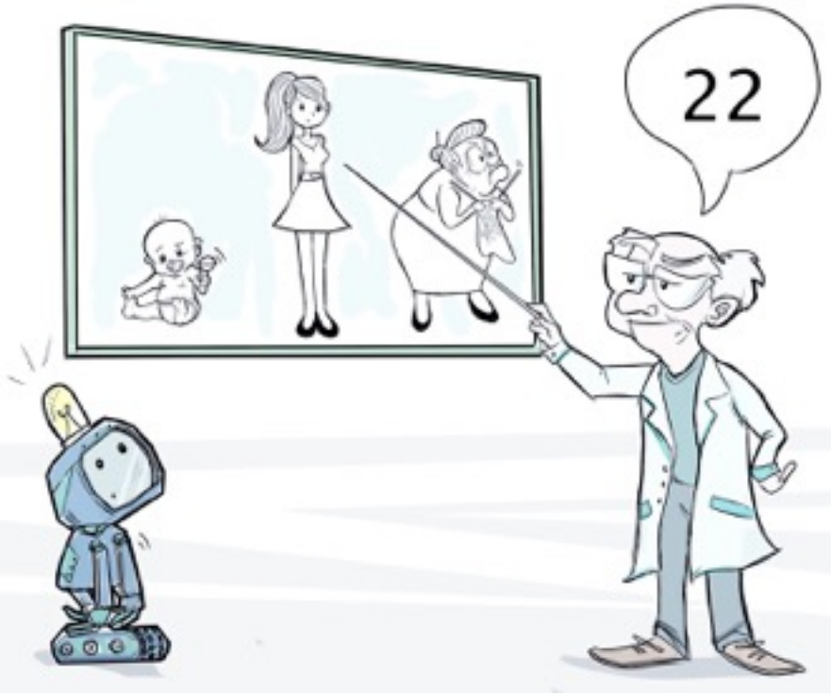
IMAGE CLASSIFICATION	MEAN ACCURACY
Our fine-tuned DCNN	**0.96**
Feature Based SVM	0.82

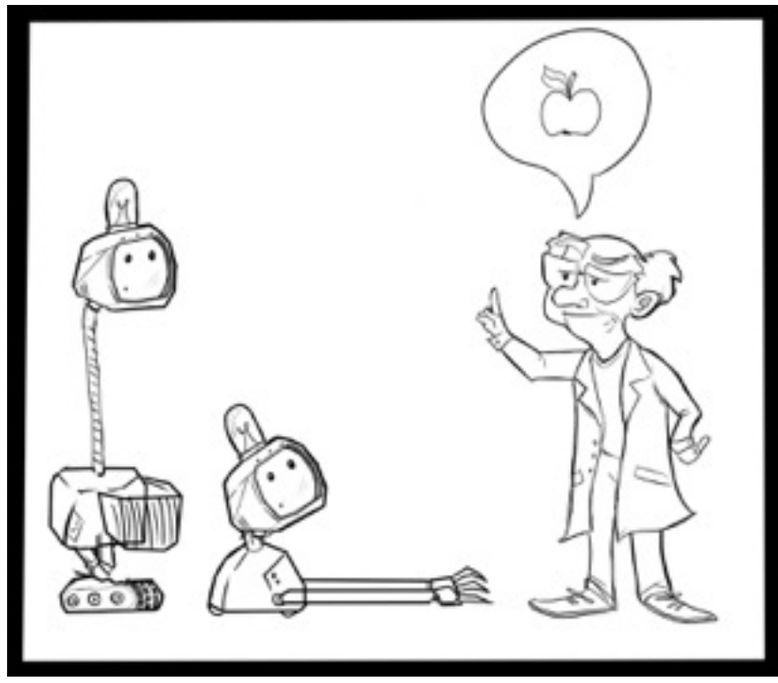
https://www.youtube.com/watch?v=nB15sW-mkHA

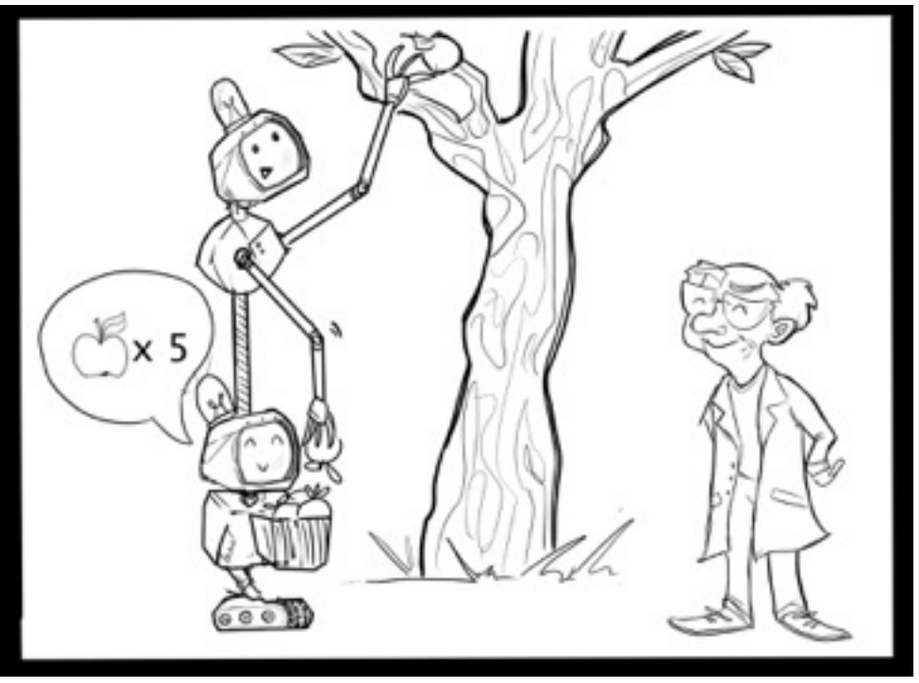
BOOK

Humanly about machine learning

goo.gl/tbqina


22




x 5

Three things

if you can remember only three…

- CNN boost computer vision
- CNN this is also about NLP, language translation…
- CNN required a lot of resources (GPU)

Thank you

@slon1024
hello@vova.me
dataworkshop.eu