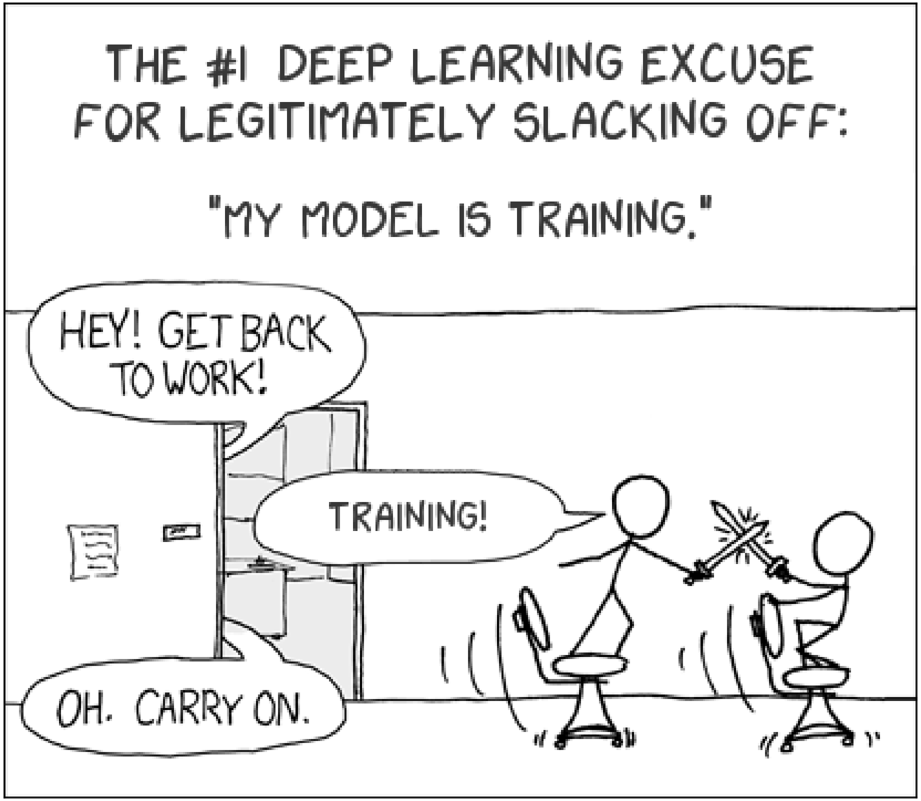
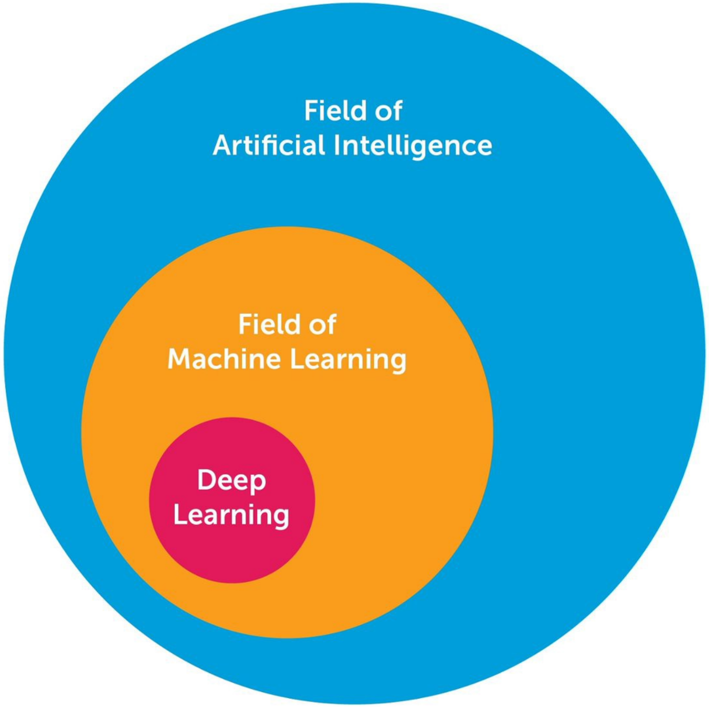
Deep Learning

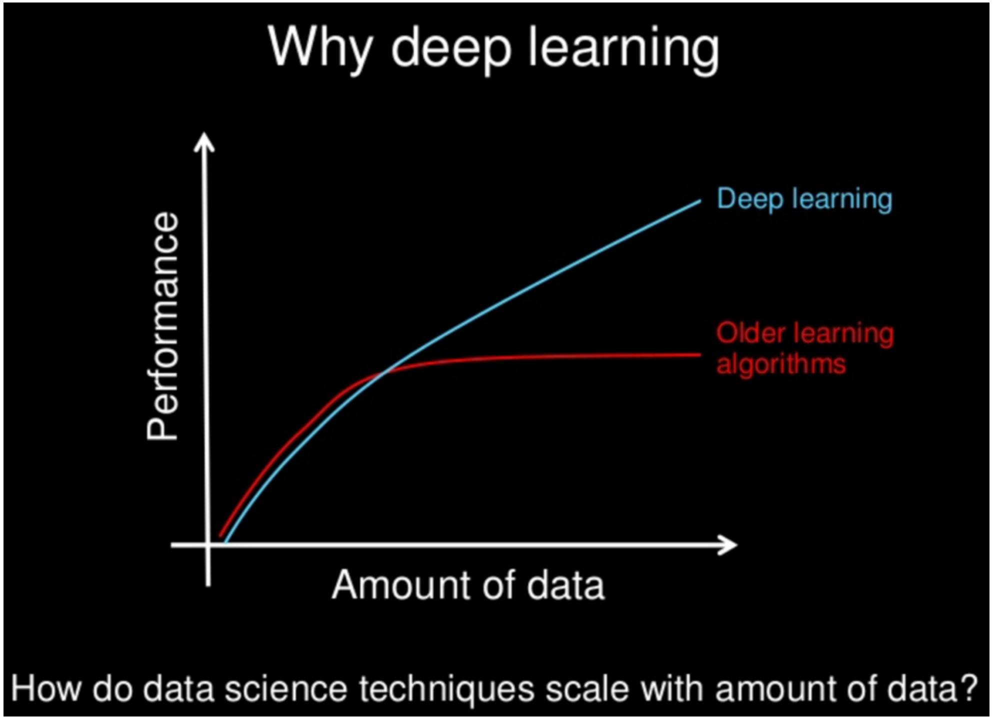
A Deep Learning Joke



What is Deep Learning?

Deep Learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural networks.





Why deep learning

How do data science techniques scale with amount of data?

Slide by Andrew Ng

Deep Learning Applications

Speech Recognition

All major commercial speech recognition systems (Microsoft Cortana, Amazon Alexa, Google Now, Apple Siri, etc) are based on deep learning.

Image recognition

Deep learning-based image recognition has become "superhuman", producing more accurate results than human contestants.

Visual art processing

See https://deepart.io/.

Natural language processing

See GPT-3 examples

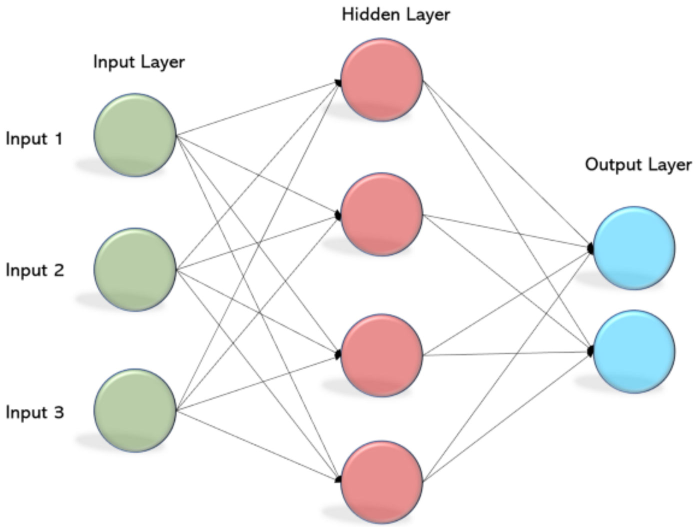
...

Multi Layer Perceptron (MLP)

MLP consists of at least three layers of nodes:

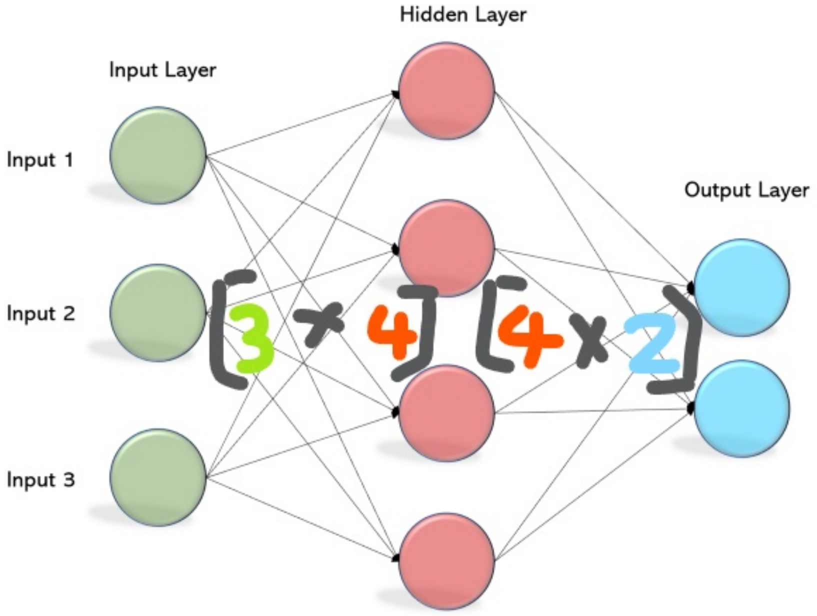
- an input layer
- a hidden layer
- an output layer

Except for the input nodes, each node is a neuron that uses a nonlinear activation function.

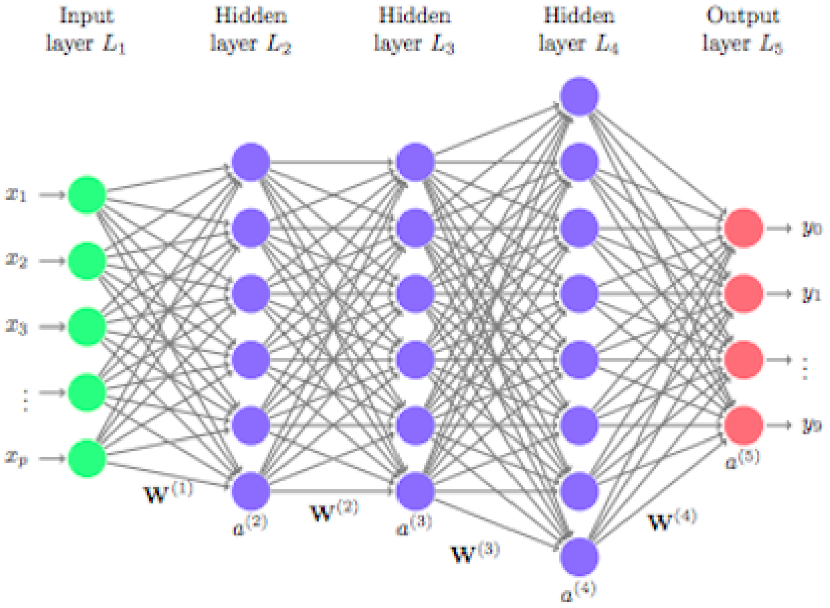


Weight Matrices

If A has dimensions [m×n] and B has dimensions [n×p] , then the product of A and B is defined, and has dimensions [m×p].



Meaning of "Deep"



Note: There are more complex network types.

Hyperparameters

- number of hidden layers: Many hidden units within a layer with regularization techniques can increase accuracy. Smaller number of units may cause underfitting.
- network weight initialization: Ideally, it may be better to use different weight initialization schemes according to the activation function used on each layer. Mostly uniform distribution is used.
- activation function: Activation functions are used to introduce nonlinearity to models, which allows deep learning models to learn nonlinear prediction boundaries.

Hyperparameters cont'd

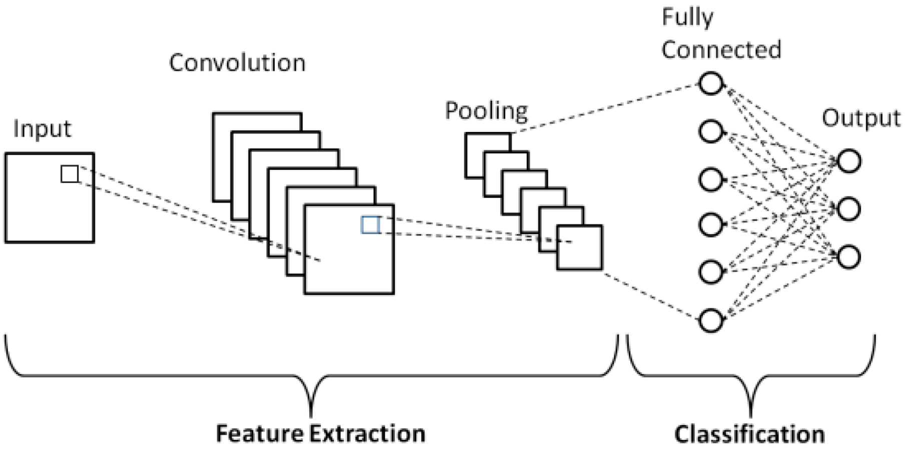
- learning rate: The learning rate defines how quickly a network updates its parameters. Low learning rate slows down the learning process but converges smoothly. Larger learning rate speeds up the learning but may not converge.

- momentum: Momentum helps to know the direction of the next step with the knowledge of the previous steps. It helps to prevent oscillations. A typical choice of momentum is between 0.5 to 0.9.

- number of epochs: Number of epochs is the number of times the whole training data is shown to the network while training. Increase the number of epochs until the validation accuracy starts decreasing even when training accuracy is increasing(overfitting).

- batch size: Mini batch size is the number of sub-samples given to the network after which parameter update happens. A good default for batch size might be 32. Also try 32, 64, 128, 256, and so on.

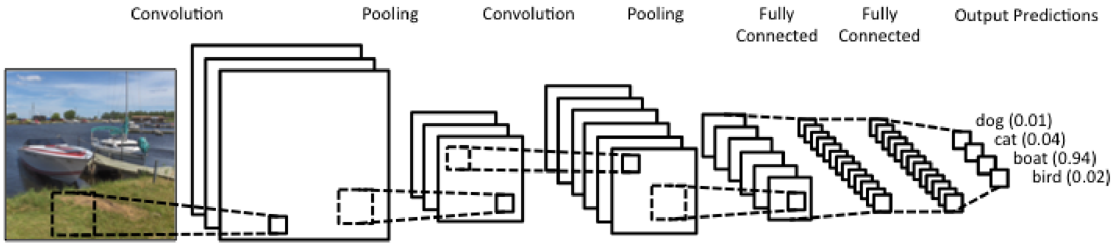
Convolutional Neural Network (CNN)



CNN explained

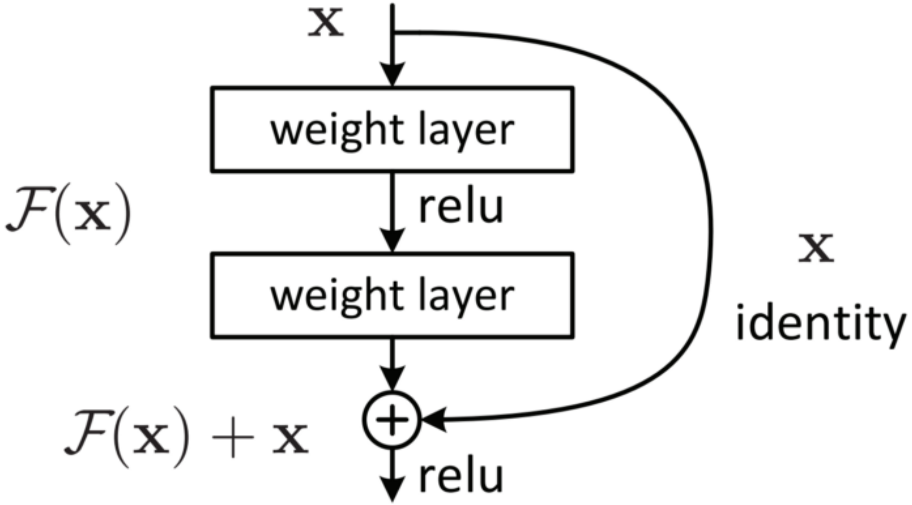
- A convolution layer defines a window by which we examine a subset of the input, and subsequently scans the entire input looking through this window.
- A pooling layer compresses the spatial dimensions, reducing the number of parameters needed to extract features in following layers

Go "Deep"



Residual Network (ResNet)

Residual Network (ResNet) is a Convolutional Neural Network (CNN) architecture which was designed to enable hundreds or thousands of convolutional layers.



ResNet cont'd

ResNet stacks up identity mappings, layers that initially don’t do anything, and skips over them, reusing the activations from previous layers.

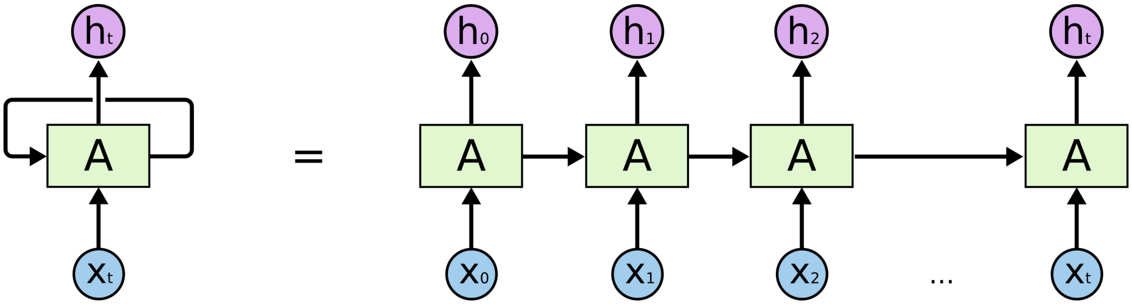
- Skipping initially compresses the network into only a few layers, which enables faster learning.
- When the network trains again, all layers are expanded and the “residual” parts of the network explore more and more of the feature space

Famous ResNet

ResNest is one of the most popular architectures in various computer vision tasks

ResNet-34, ResNet-50, ResNet-101

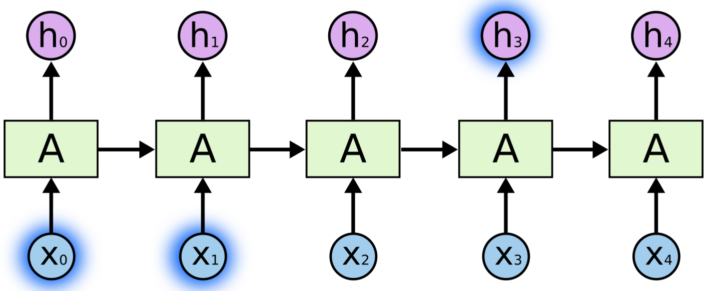
Recurrent Neural Network (RNN)

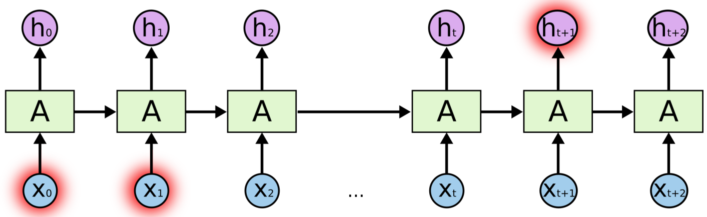


Recurrent neural networks, also known as RNNs, are a class of neural networks that allow previous outputs to be used as inputs while having hidden states.

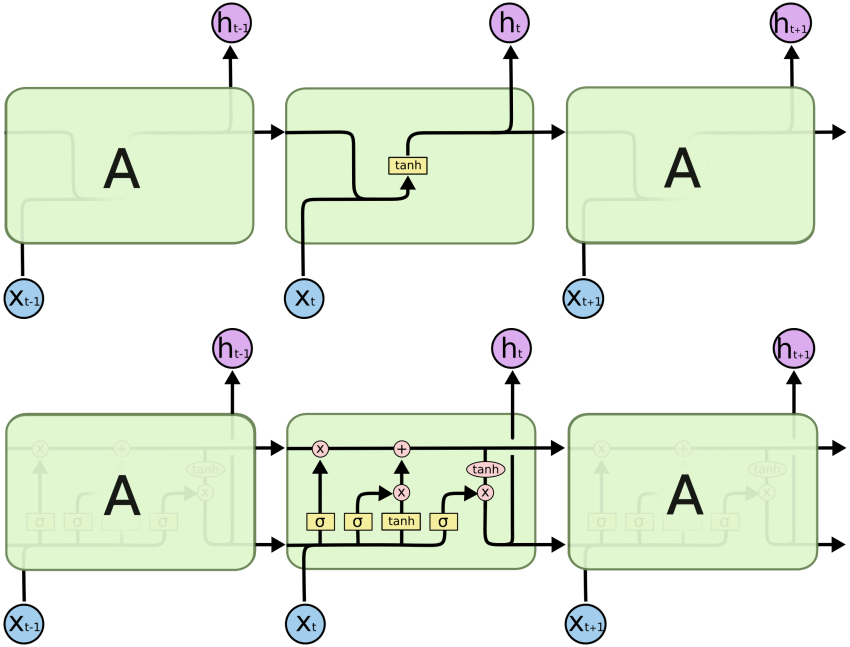
Long Short Term Memory networks (LSTM)

LSTM is a special kind of RNN, capable of learning long-term dependencies.





LSTM explained

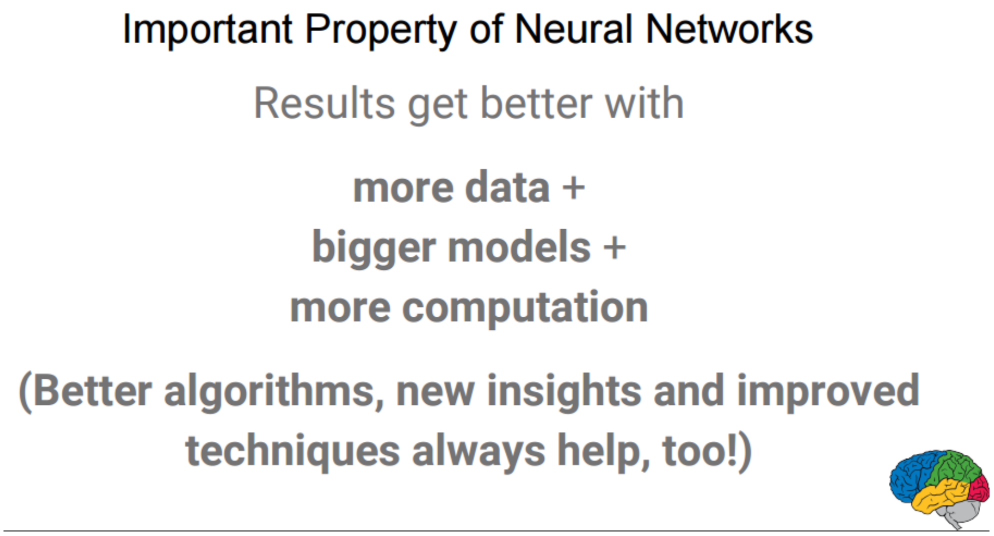


- Decide what information it's going to throw away
- Decide what new information it's going to store
- Update the weights
- Output to next layer

LSTM works very well at

- Robot control
- Music composition
- Time series prediction
- Speech recognition
- Machine Translation
- Language modeling
- ...

Recap of Deep Learning



Important Property of Neural Networks

Results get better with

more data +
bigger models +
more computation

(Better algorithms, new insights and improved techniques always help, too!)

Slide by Jeff Dean

Python Ecosystem for Deep Learning

- TensorFlow
- PyTorch
- MXNet
- fastai
- CNTK
- Keras
- Caffe
- ...

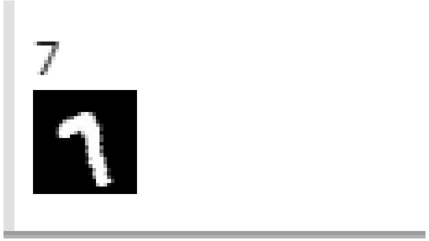
fastai

- The fastai library simplifies training fast and accurate neural nets using modern best practices.
- It's based on research in to deep learning best practices undertaken at fast.ai, including "out of the box" support for vision, text, tabular, and collab (collaborative filtering) models.

Computer Vision

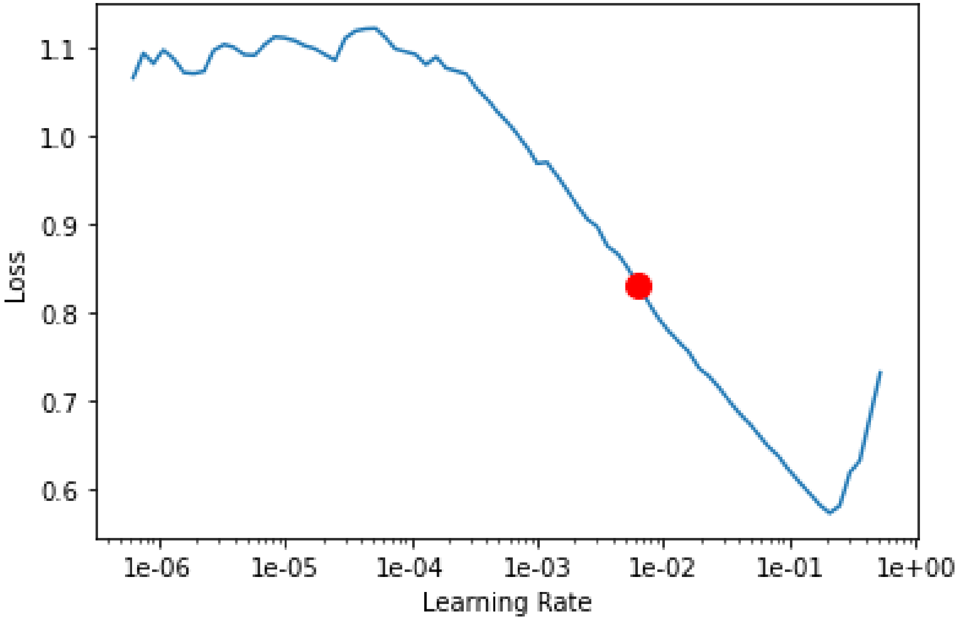
The `fastai.vision` module of the fastai library contains all the necessary functions to define a Dataset and train a model for computer vision tasks. It contains four different submodules to reach that goal:

```
from fastai.vision import *
path = untar_data(URLs.MNIST_SAMPLE)
data = ImageDataBunch.from_folder(path)

img, label = data.train_ds[0]

print(label)
img
```



```
7
```

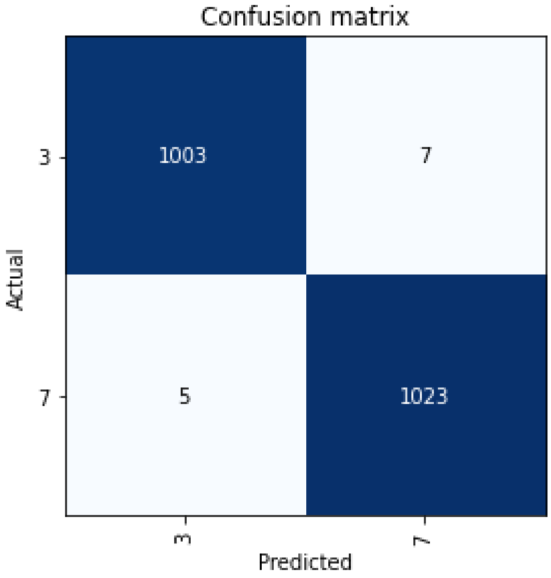
```
learn = cnn_learner(data, models.resnet18, metrics=accuracy)
learn.lr_find()
learn.recorder.plot(suggestion=True)
```



```
lr = learn.recorder.min_grad_lr
learn.fit_one_cycle(1, lr)
```

```
epoch    train_loss    valid_loss    accuracy    time
    0    0.055897    0.018004    0.994112    00:09
```

```
interp = ClassificationInterpretation.from_learner(learn)
interp.plot_confusion_matrix()
```



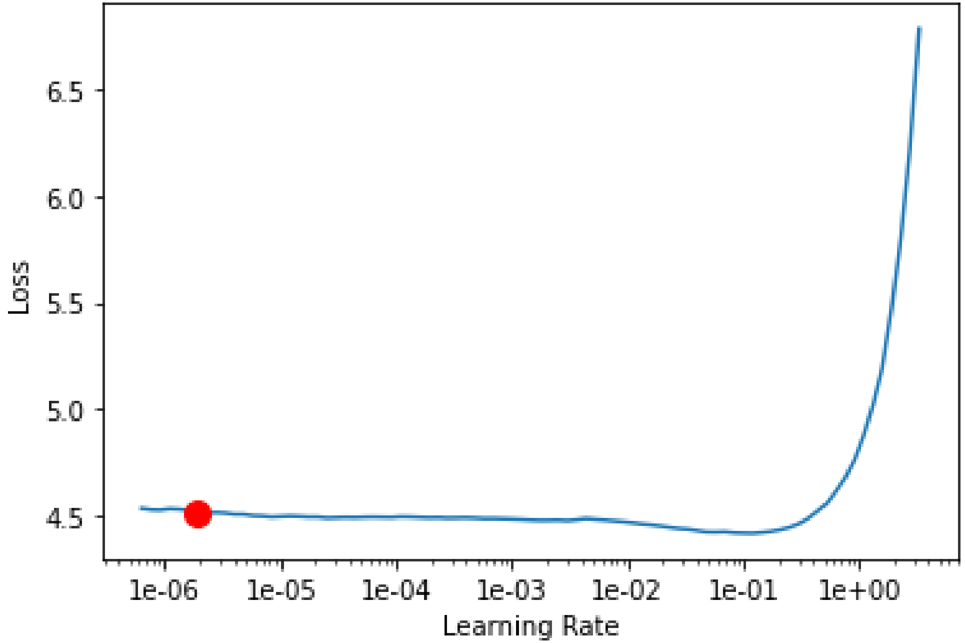
Natural Language Processing

The `fastai.text` module of the fastai library contains all the necessary functions to define a Dataset suitable for the various NLP tasks and quickly generate models you can use for them.

```python
from fastai.text import *
path = untar_data(URLs.IMDB_SAMPLE)
df = pd.read_csv(path/'texts.csv')
df.head()
```

```
        label                                               text
0    negative    Un-bleeping-believable! Meg Ryan doesn't even ...
1    positive    This is a extremely well-made film. The acting...
2    negative    Every once in a long while a movie will come a...
3    positive    Name just says it all. I watched this movie wi...
4    negative    This movie succeeds at being one of the most u...
```

```
# Language model data
data_lm = TextLMDataBunch.from_csv(path, 'texts.csv')
# Classifier model data
data_clas = TextClasDataBunch.from_csv(
    path, 'texts.csv', vocab=data_lm.train_ds.vocab, bs=32
)
learn = language_model_learner(data_lm, AWD_LSTM, drop_mult=0.5)
learn.lr_find()
learn.recorder.plot(suggestion=True)
```



```
lr = learn.recorder.min_grad_lr
learn.fit_one_cycle(1, lr)
```

```
epoch     train_loss     valid_loss     accuracy     time
    0     4.478594     4.225167     0.261193     00:06
```

```
learn.predict("This is a review about", n_words=20)
```

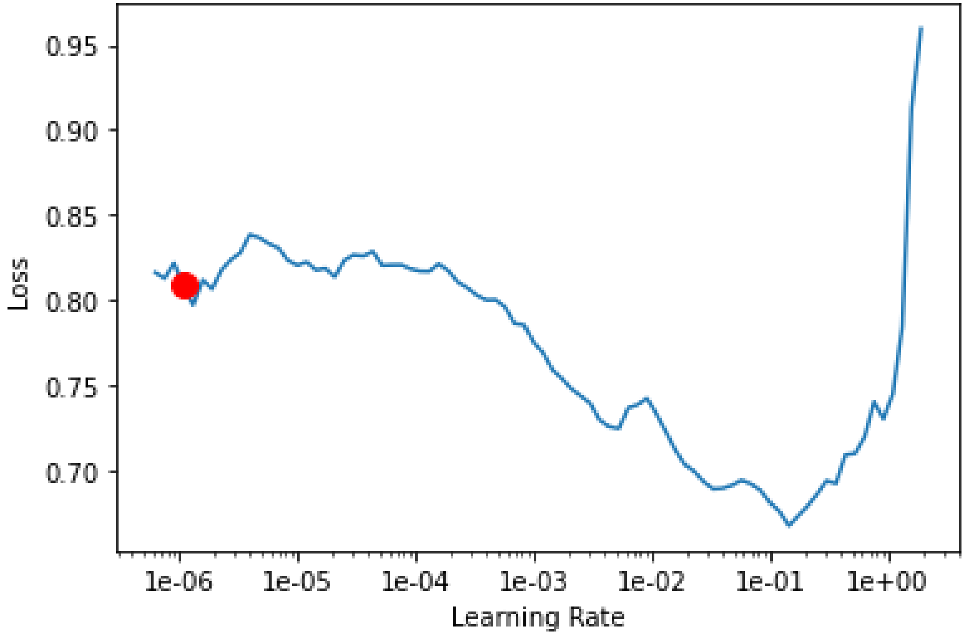
```
This is a review about European history , even Hamilton Award laws
```

```
learn.save_encoder('ft_enc')
learn = text_classifier_learner(data_clas, AWD_LSTM, drop_mult=0.5)
learn.load_encoder('ft_enc')
```

Note: Save the encoder to be able to use it for classification.

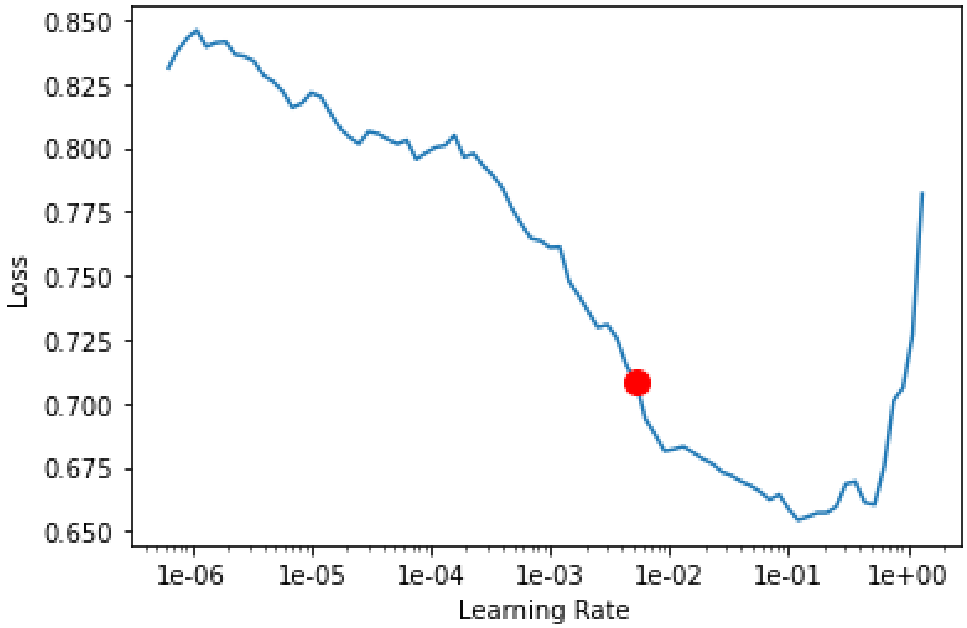
```
learn.lr_find()
learn.recorder.plot(suggestion=True)
lr = learn.recorder.min_grad_lr
learn.fit_one_cycle(1, lr)
```

```
Min numerical gradient: 1.10E-06
Min loss divided by 10: 1.45E-02
epoch     train_loss     valid_loss     accuracy     time
    0     0.825851     0.807620     0.412935     00:04
```



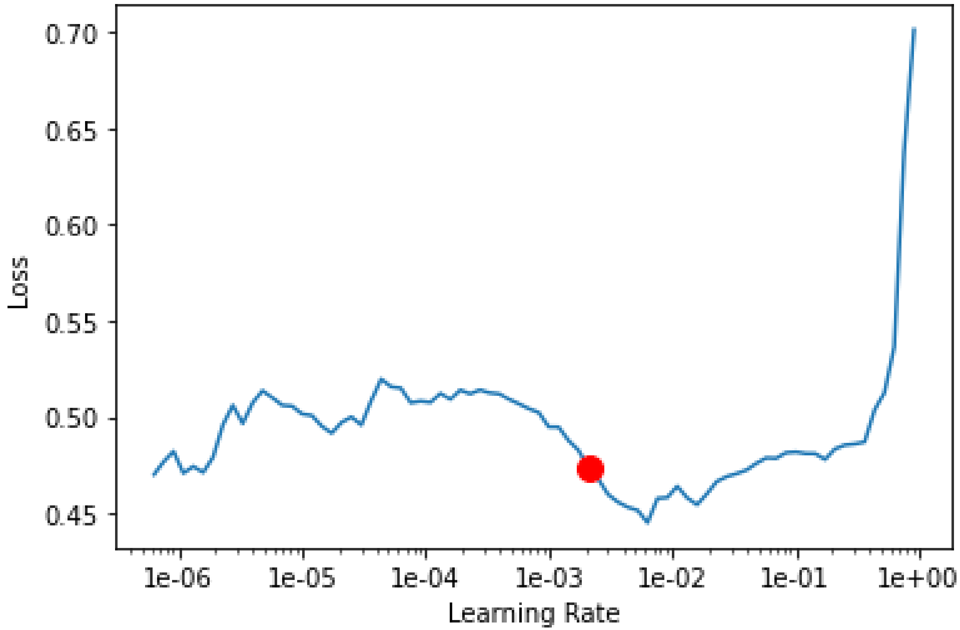
```
learn.freeze_to(-2)
learn.lr_find()
learn.recorder.plot(suggestion=True)
lr = learn.recorder.min_grad_lr
learn.fit_one_cycle(1, lr)
```

```
Min numerical gradient: 5.25E-03
Min loss divided by 10: 1.20E-02
epoch     train_loss     valid_loss     accuracy     time
    0     0.683561     0.712243     0.676617     00:06
```



```
learn.unfreeze()
learn.lr_find()
learn.recorder.plot(suggestion=True)
lr = learn.recorder.min_grad_lr
learn.fit_one_cycle(1, lr)
```

```
Min numerical gradient: 2.09E-03
Min loss divided by 10: 6.31E-04
epoch     train_loss     valid_loss     accuracy     time
    0     0.426766     0.421171     0.810945     00:12
```



```
learn.predict("This was a great movie!")
```

```
(Category tensor(1), tensor(1), tensor([0.0020, 0.9980]))
```

```
data_clas.train_ds.y.c2i
```

```
{'negative': 0, 'positive': 1}
```

Questions?