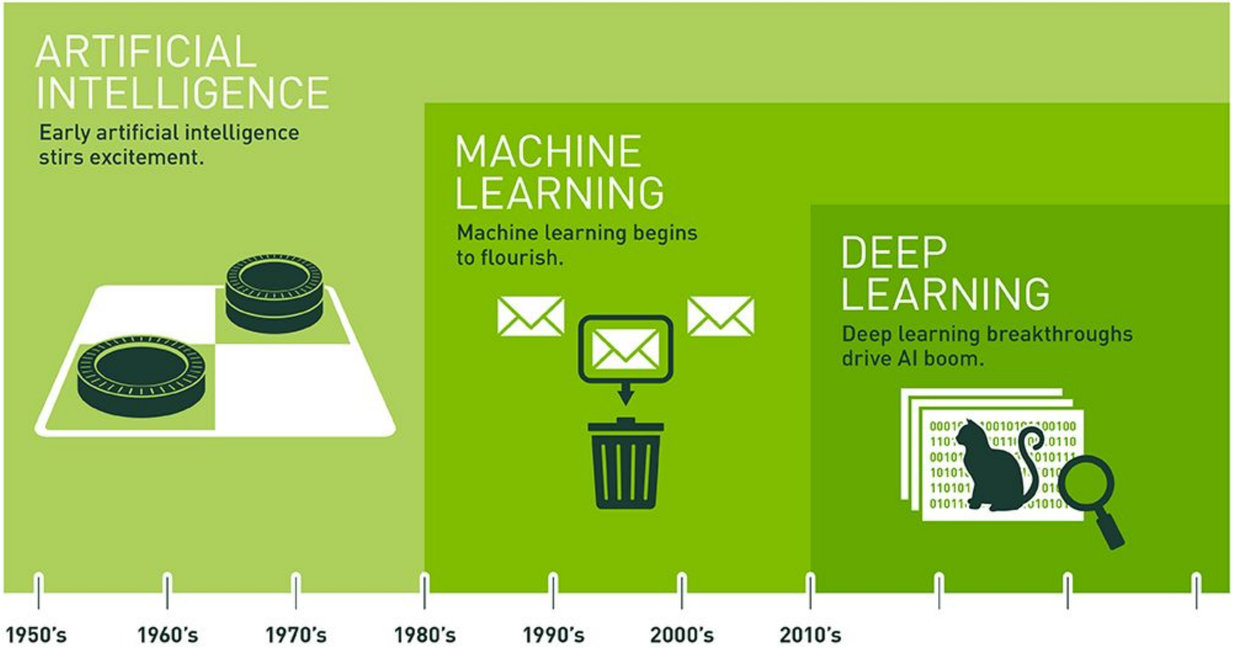
Machine Learning in Astronomy

Szymon J. Nakoneczny - 17.03.2021

Plan

1. Introduction to machine learning
2. Software
3. Examples in astronomy

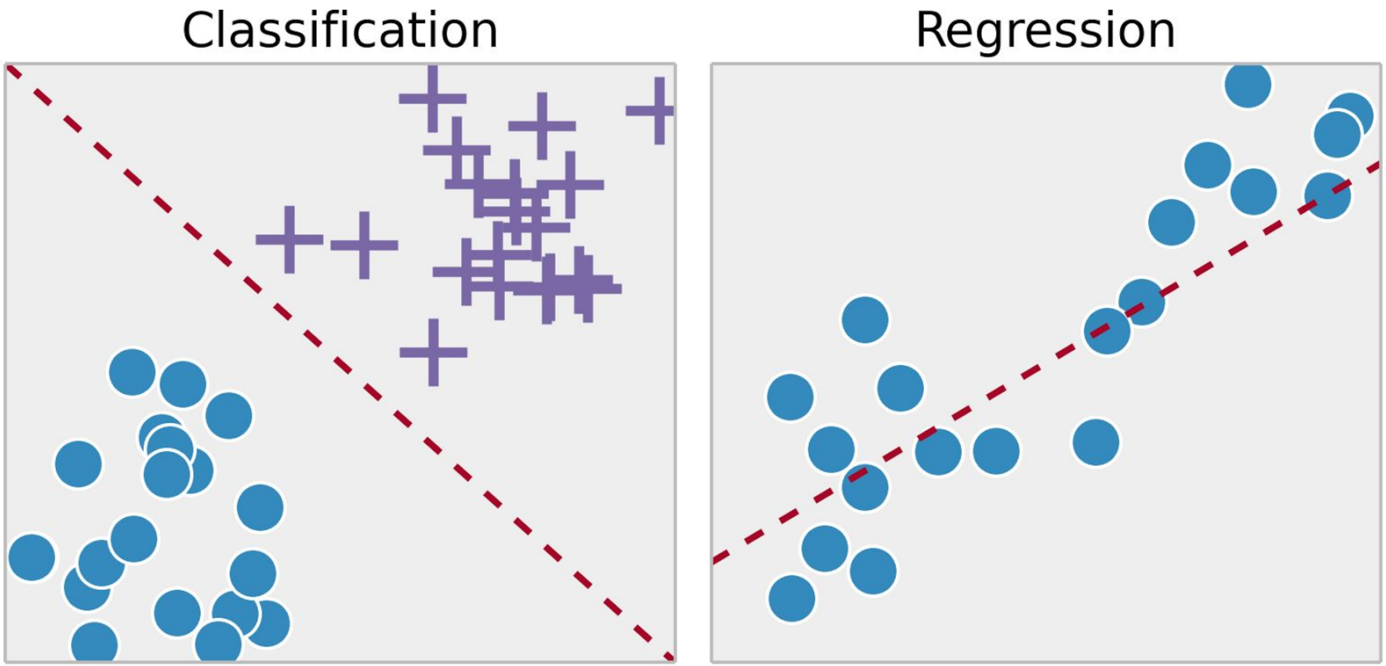
Where are we?



source: https://www.datasciencecentral.com/profiles/blogs/artificial-intelligence-vs-machine-learning-vs-deep-learning

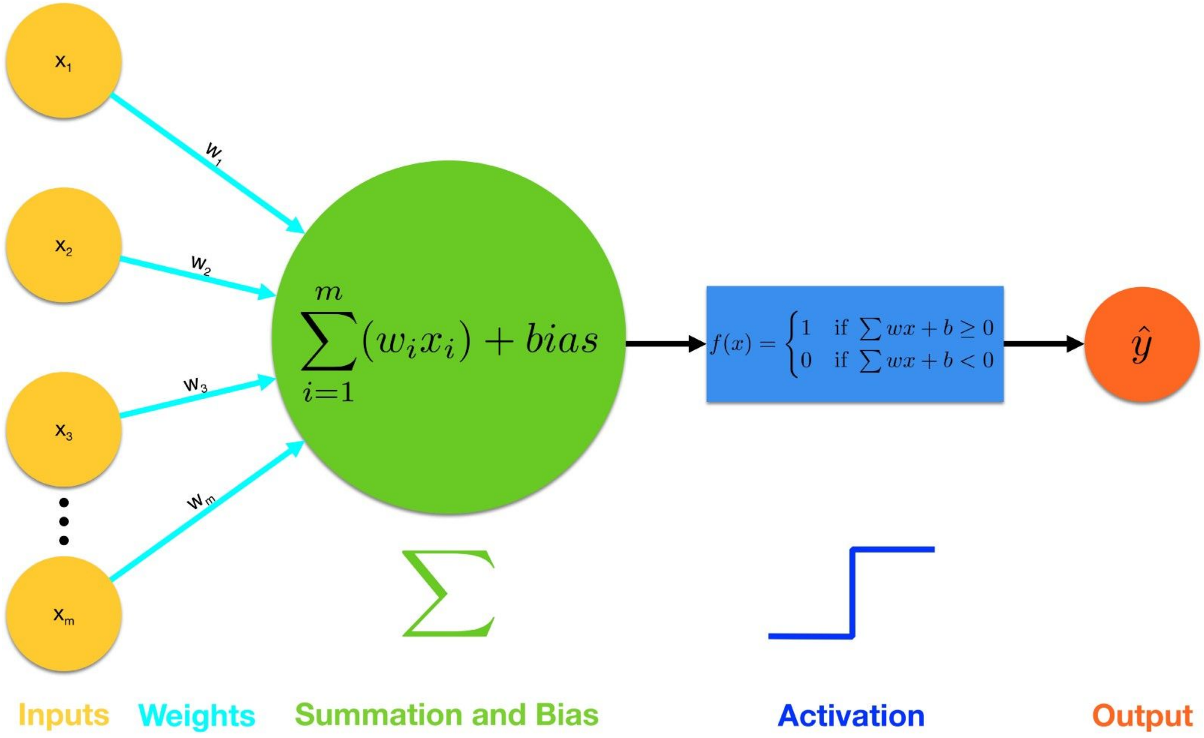
(un)supervised

Supervised learning



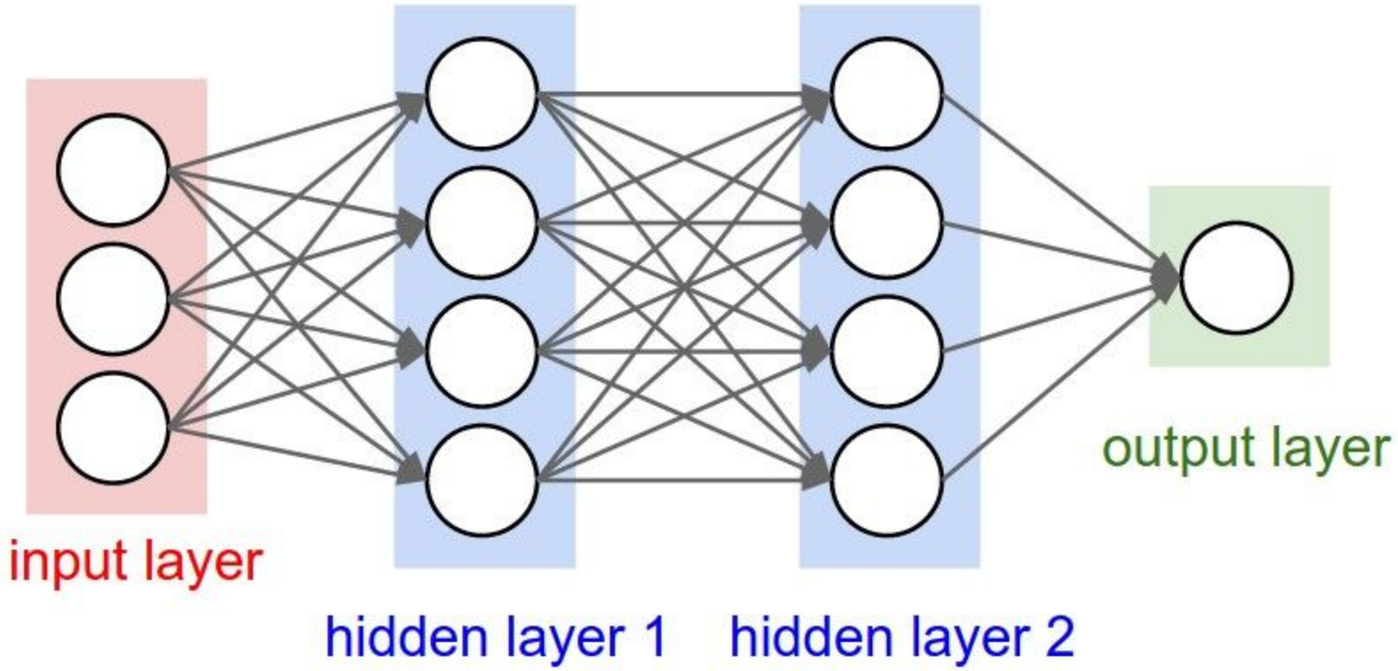
source: https://srconstantin.wordpress.com/2017/01/28/performance-trends-in-ai

Perceptron



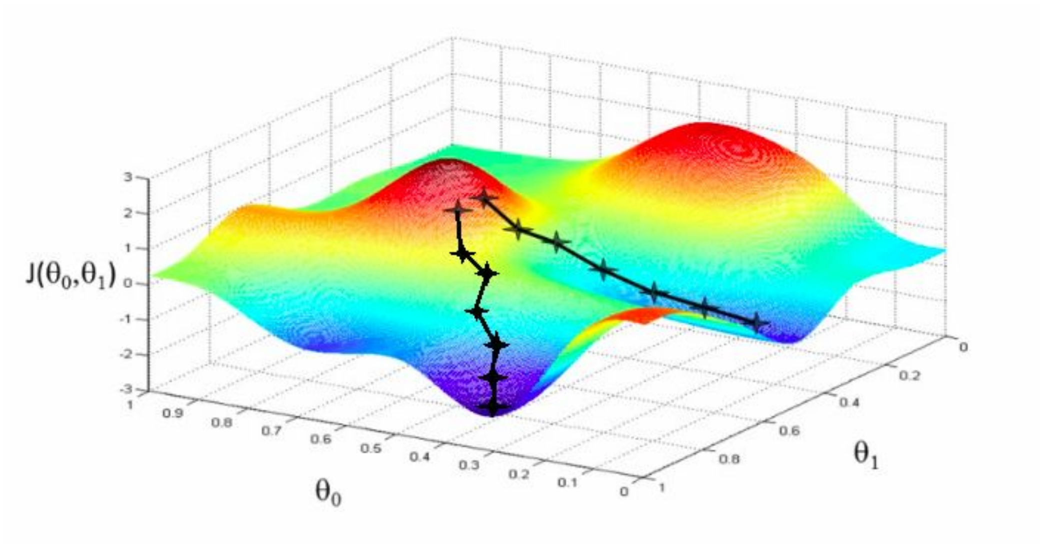
source: https://towardsdatascience.com/introducing-deep-learning-and-neural-networks-deep-learning-for-rookies-1-bd68f9cf5883

Fully connected layer



source: http://cs231n.github.io/convolutional-networks/

Gradient descent



Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x$

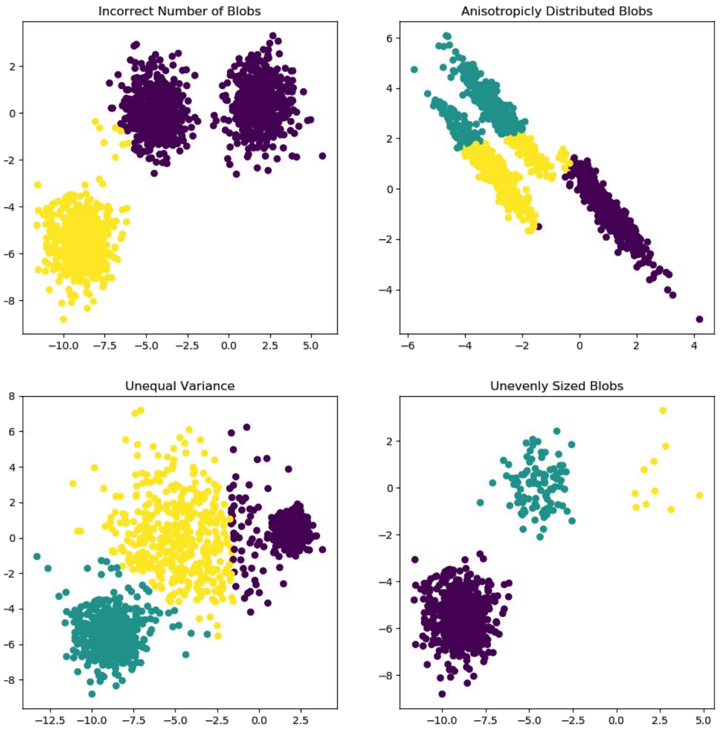
Parameters: θ_0, θ_1

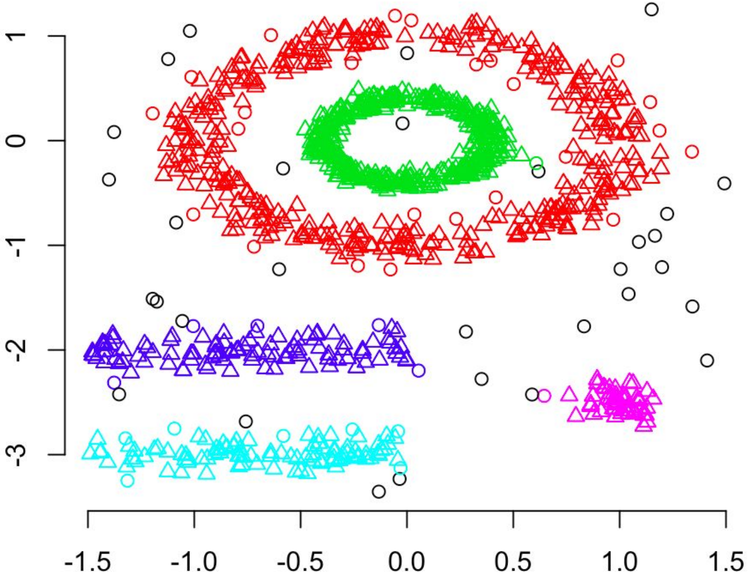
Cost Function: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$

Goal: $\underset{\theta_0, \theta_1}{\text{minimize}}\ J(\theta_0, \theta_1)$

source: https://www.youtube.com/watch?v=5u4G23_OohI

Clustering

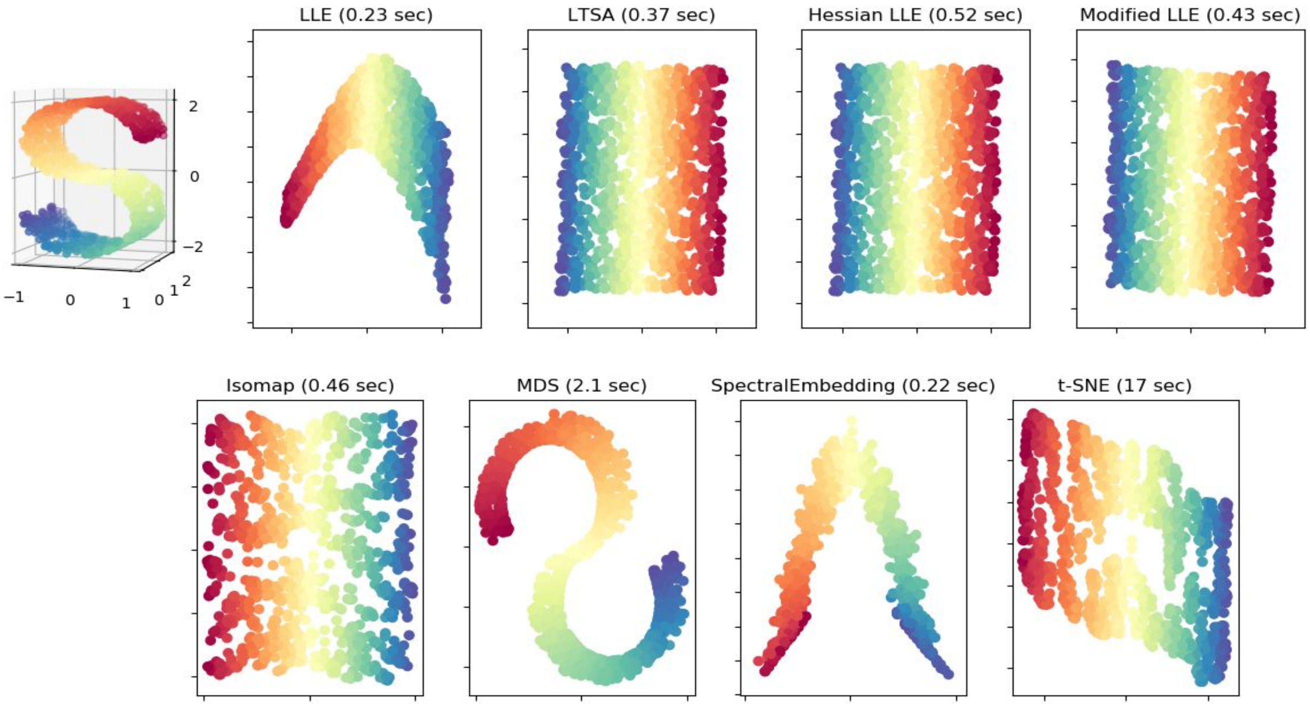




source: http://scikit-learn.org/stable/modules/clustering.html

source: http://www.sthda.com/english/wiki/print.php?id=246

Manifold learning

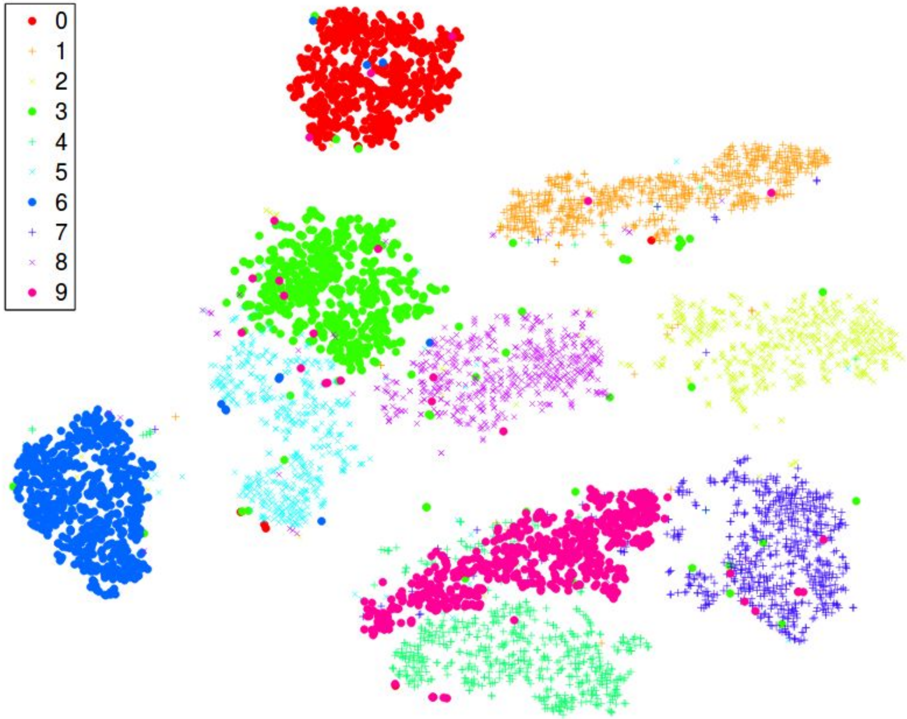


source: http://scikit-learn.org/stable/modules/manifold.html

T-SNE (2008)

t-distributed Stochastic Neighbour Embedding

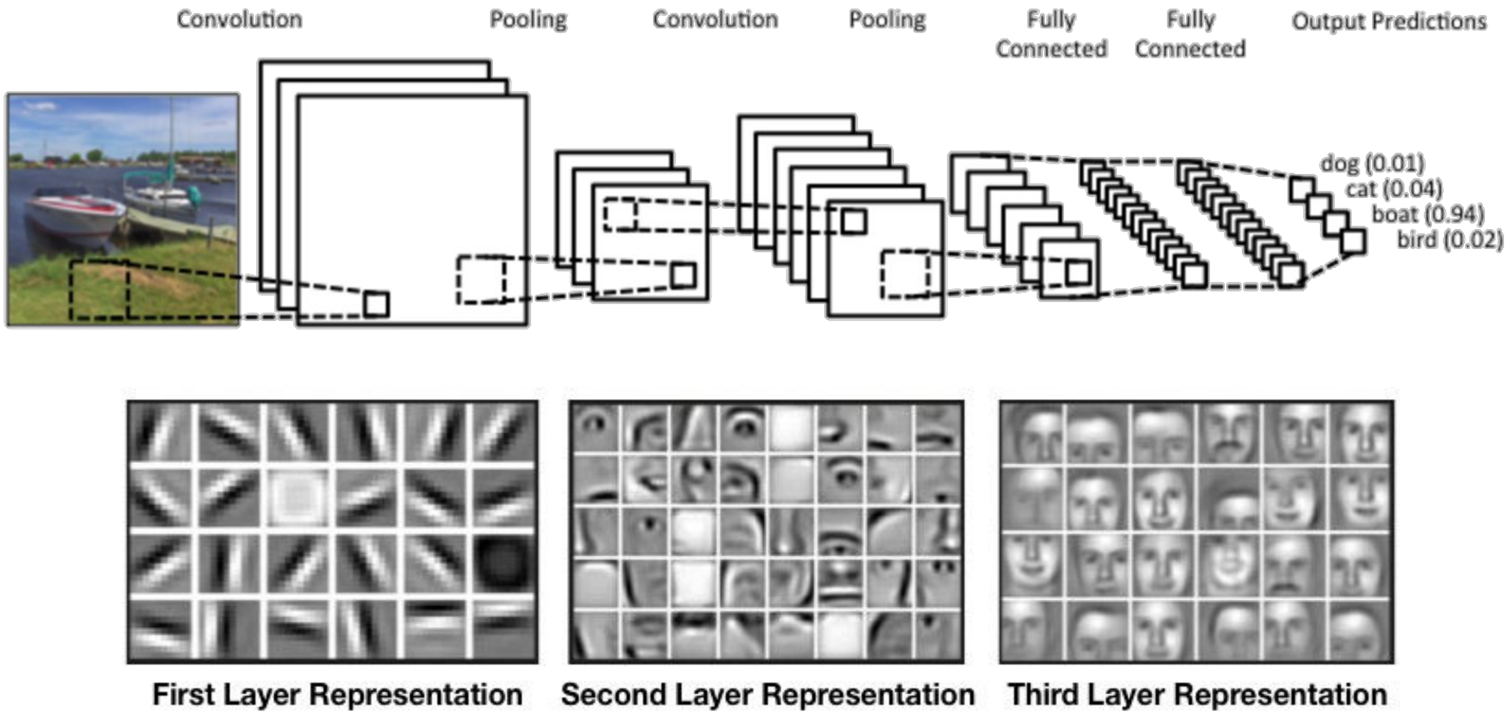
Visualization of high-dimensional data



source: http://scikit-learn.org/stable/modules/manifold.html

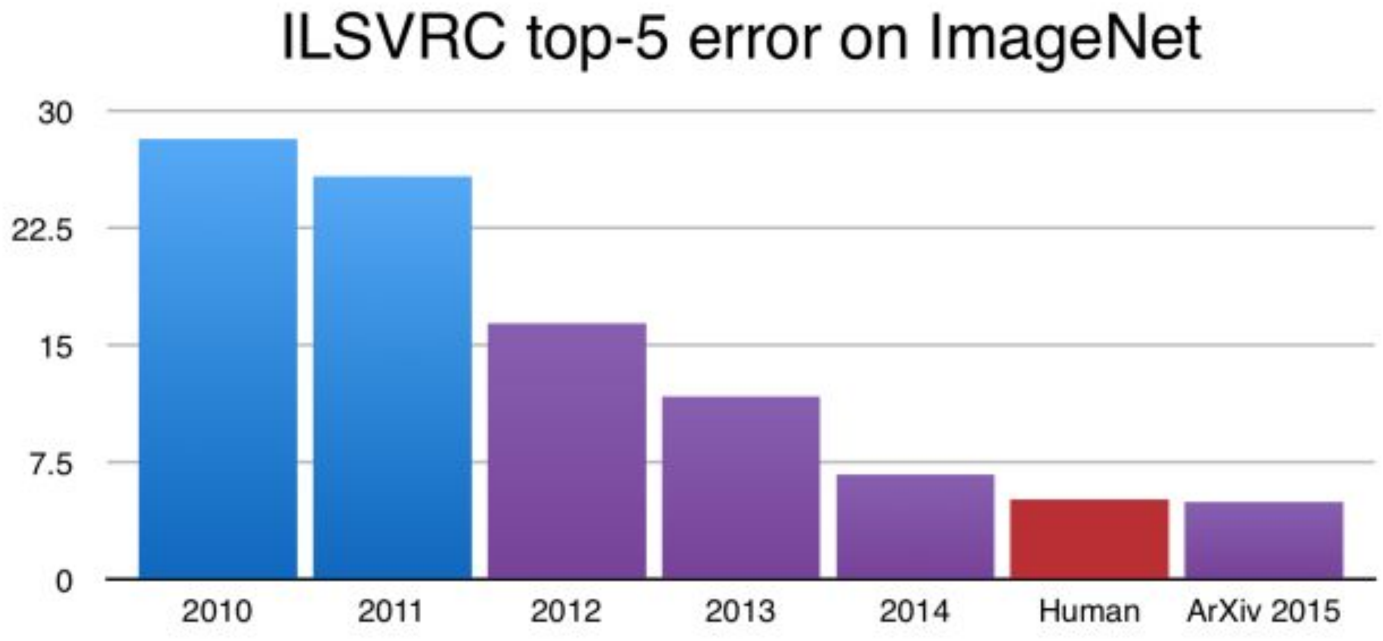
Deep learning

Each layer learns something



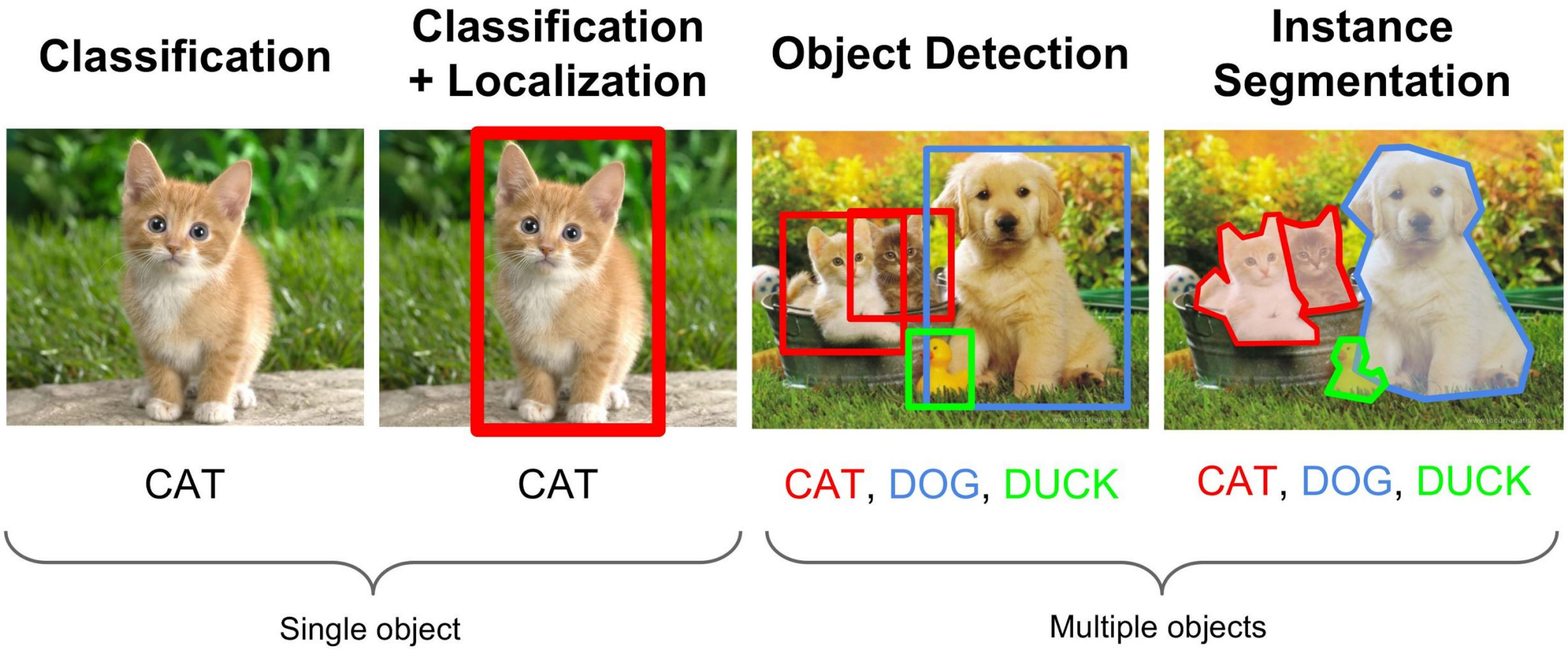
sources: https://srconstantin.wordpress.com/2017/01/28/performance-trends-in-ai/, https://www.clarifai.com/technology

Recent rise of deep learning



source: https://srconstantin.wordpress.com/2017/01/28/performance-trends-in-ai

Different problems



https://arxiv.org/pdf/1511.04164.pdf

Natural language object retrieval

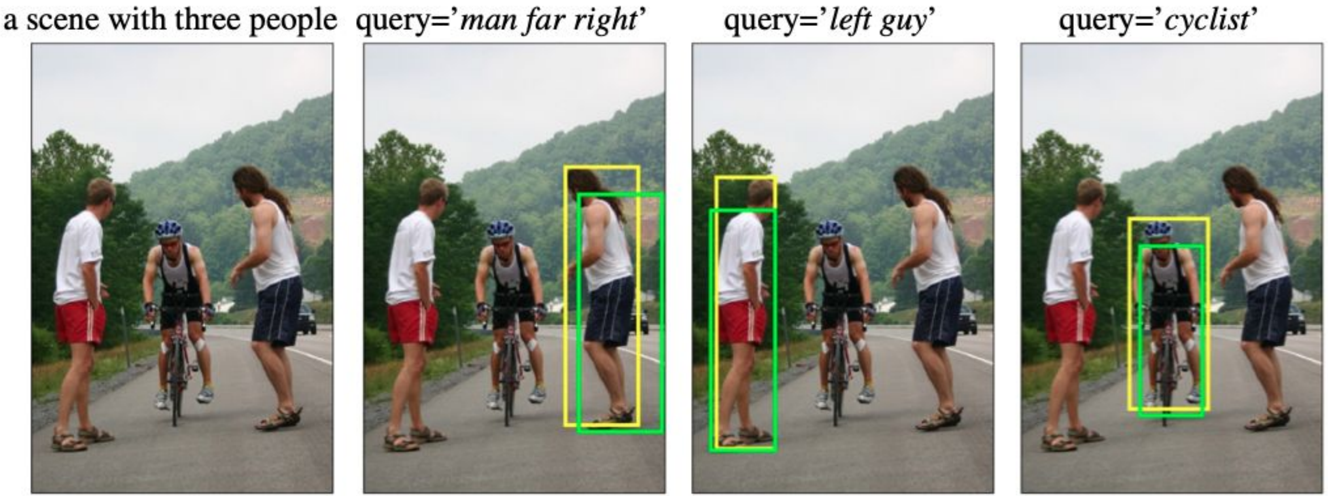
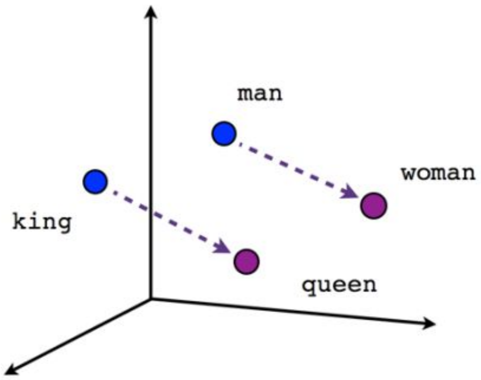
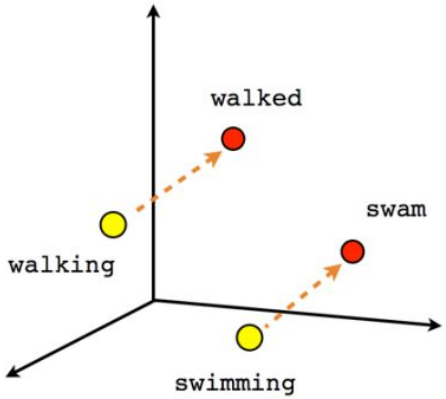


Figure 3. An example image in ReferIt dataset where objects are described based on other objects in the scene. When referring to one of the three "people" in the image, expressions based on both the object and the context are used to make the description discriminative. Our model can handle such object descriptions in context by incorporating these information into the recurrent neural network. In the images above, yellow boxes are ground truth and green boxes are correctly retrieved results by our model using highest scoring candidate from 100 EdgeBox proposals.

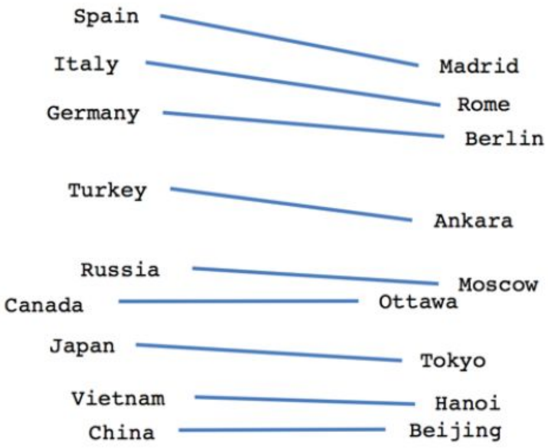
Semi-supervised representation learning - Word2Vec



Male-Female



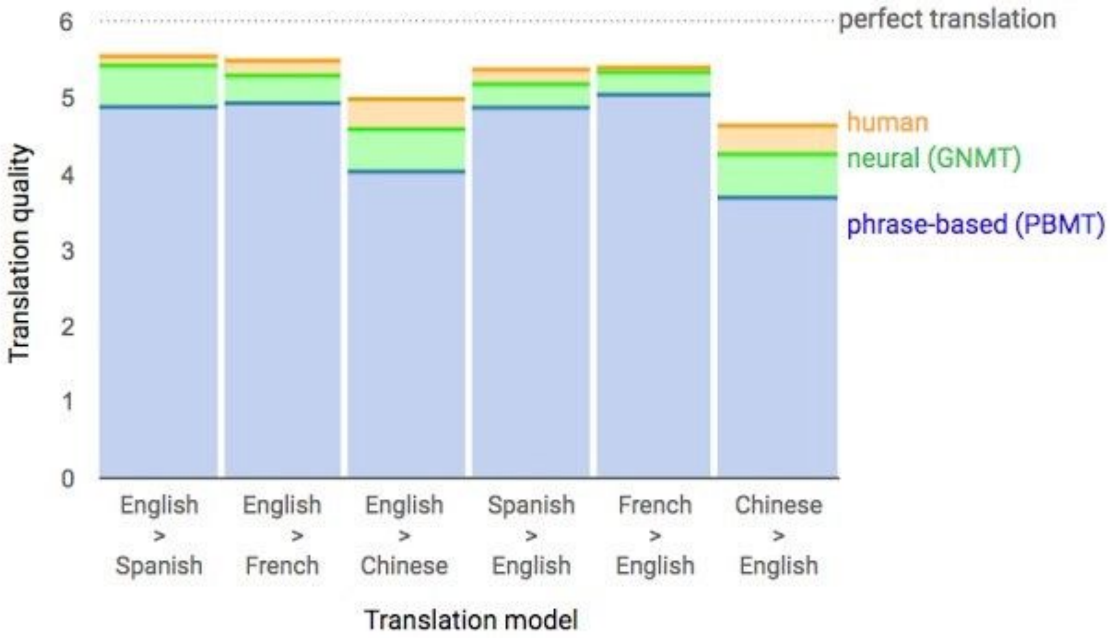
Verb tense



Country-Capital

source: https://www.tensorflow.org/tutorials/word2vec

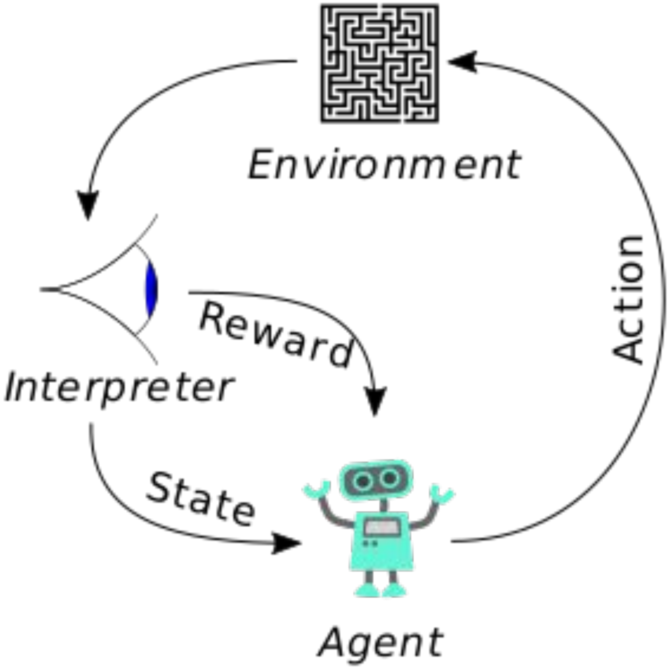
Google Translate model (2017)



source: https://blog.statsbot.co/deep-learning-achievements-4c563e034257

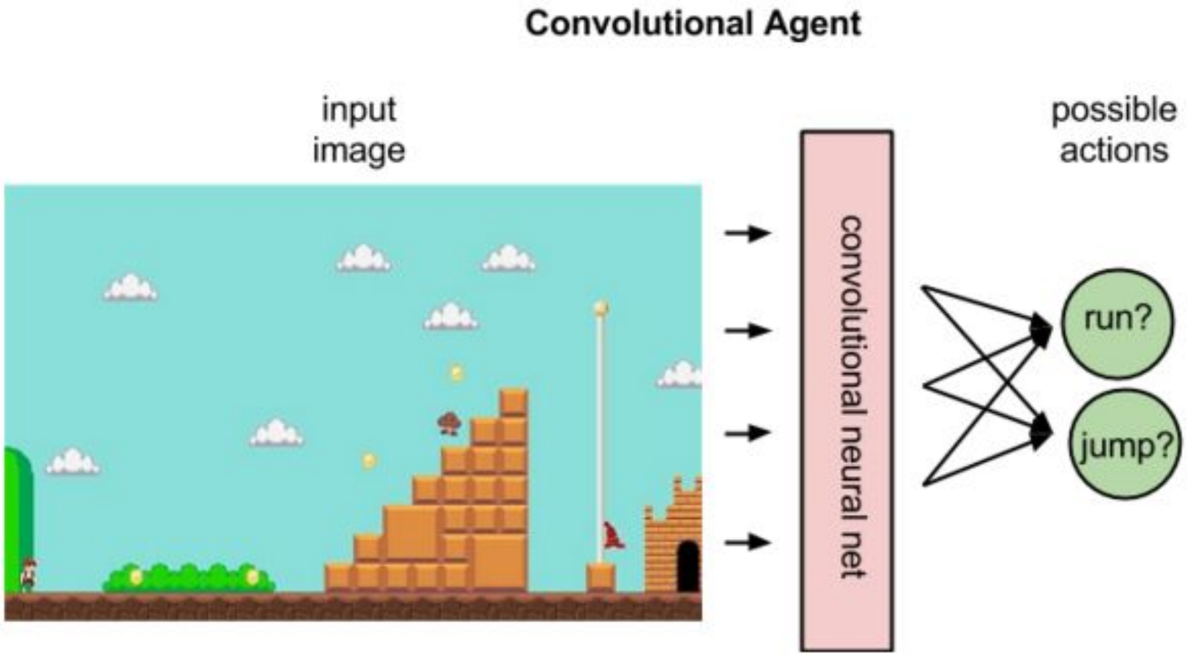
Reinforcement learning

Idea behind Reinforcement Learning



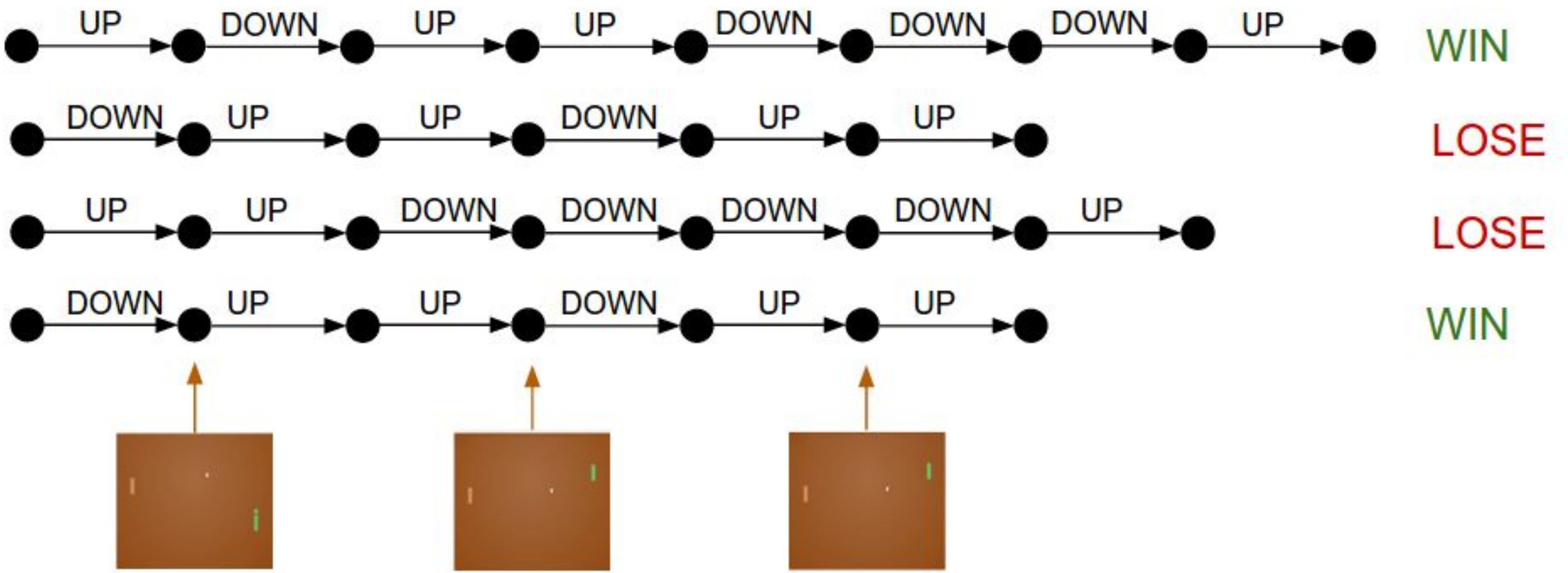
source: https://en.wikipedia.org/wiki/Reinforcement_learning

Reinforcement learning in games



source: https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/deep_reinforcement_learning.html

Training RL models



source: https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/deep_reinforcement_learning.html

Quake 3 Arena (2018)

Outdoor map overview

source: https://www.theverge.com/2018/7/4/17533898/deepmind-ai-agent-video-game-quake-iii-capture-the-flag

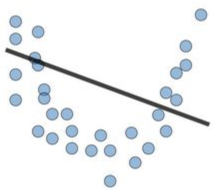
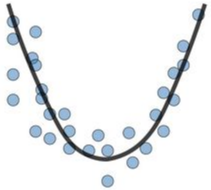
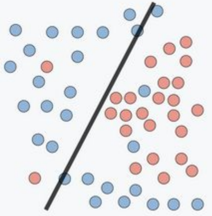
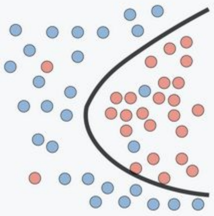
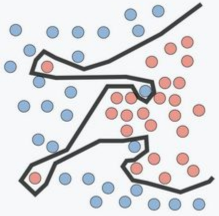
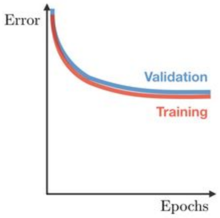
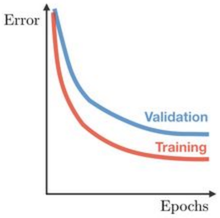
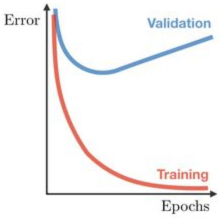
Deep or dumb?

Data leaks

Bias vs variance tradeoff

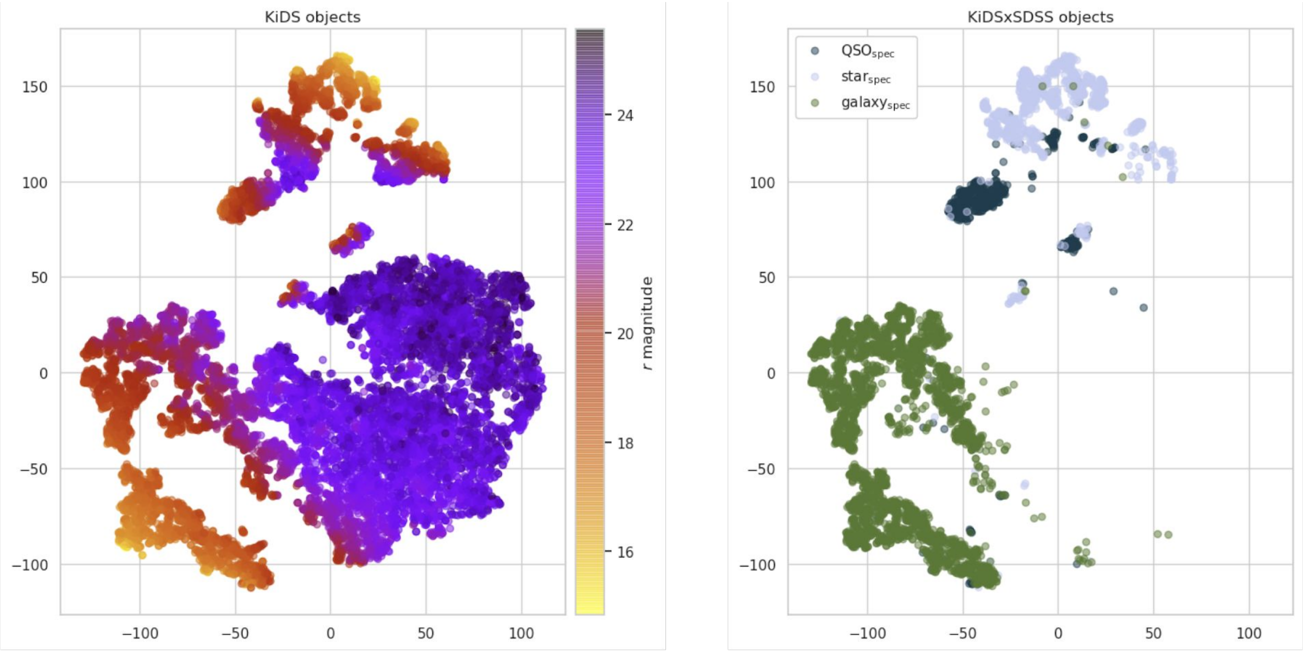
in machine learning

- Simpler model - higher bias
- More complex model - higher variance

	Underfitting	Just right	Overfitting
Symptoms	• High training error • Training error close to test error • High bias	• Training error slightly lower than test error	• Very low training error • Training error much lower than test error • High variance
Regression illustration			
Classification illustration			
Deep learning illustration			
Possible remedies	• Complexify model • Add more features • Train longer		• Perform regularization • Get more data

Train vs inference / model testing

https://ui.adsabs.harvard.edu/abs/2021arXiv210106010B/abstract



t-SNE projection of **KiDSxSDSS data**. *Left*: *r* magnitude, *right*: SDSS spectroscopic classification.

Why is it worth knowing?

- Multidisciplinary

- Perhaps leads to general AI in the next decades

- Useful in science

- Lack of ML oriented referees, bad ML papers in astronomy

Software

Languages

Python data toolkit

Scipy
scientific computing

Numpy
vectors and matrices

Pandas
working with data

Matplotlib
visualizations

Seaborn
powered-up visualizations

Jupyter notebooks
science oriented environment

Anaconda
“One to rule them all”

Machine learning tools in Python

scikit-learn
general purpose ML

Scikit image
image processing

OpenCV
image processing

NLTK
natural language processing

Librosa
working with audio

Deep learning libraries in Python

TensorFlow

Keras

vs

PyTorch

Deep learning example

```
def make_model(input_shape, num_classes):
    inputs = keras.Input(shape=input_shape)
    # Image augmentation block
    x = data_augmentation(inputs)

    # Entry block
    x = layers.experimental.preprocessing.Rescaling(1.0 / 255)(x)
    x = layers.Conv2D(32, 3, strides=2, padding="same")(x)
    x = layers.BatchNormalization()(x)
    x = layers.Activation("relu")(x)

    x = layers.Conv2D(64, 3, padding="same")(x)
    x = layers.BatchNormalization()(x)
    x = layers.Activation("relu")(x)

    previous_block_activation = x  # Set aside residual

    for size in [128, 256, 512, 728]:
        x = layers.Activation("relu")(x)
        x = layers.SeparableConv2D(size, 3, padding="same")(x)
        x = layers.BatchNormalization()(x)

        x = layers.Activation("relu")(x)
        x = layers.SeparableConv2D(size, 3, padding="same")(x)
        x = layers.BatchNormalization()(x)

        x = layers.MaxPooling2D(3, strides=2, padding="same")(x)

        # Project residual
        residual = layers.Conv2D(size, 1, strides=2, padding="same")(
            previous_block_activation
        )
        x = layers.add([x, residual])  # Add back residual
        previous_block_activation = x  # Set aside next residual

    x = layers.SeparableConv2D(1024, 3, padding="same")(x)
    x = layers.BatchNormalization()(x)
    x = layers.Activation("relu")(x)

    x = layers.GlobalAveragePooling2D()(x)
    if num_classes == 2:
        activation = "sigmoid"
        units = 1
    else:
        activation = "softmax"
        units = num_classes

    x = layers.Dropout(0.5)(x)
    outputs = layers.Dense(units, activation=activation)(x)
    return keras.Model(inputs, outputs)


model = make_model(input_shape=image_size + (3,), num_classes=2)
keras.utils.plot_model(model, show_shapes=True)
```

Transfer learning

Classify ImageNet classes with ResNet50

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

model = ResNet50(weights='imagenet')

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

preds = model.predict(x)
# decode the results into a list of tuples (class, description, probability)
# (one such list for each sample in the batch)
print('Predicted:', decode_predictions(preds, top=3)[0])
# Predicted: [(u'n02504013', u'Indian_elephant', 0.82658225), (u'n01871265', u'tusker',
```

Model	Size	Top-1 Accuracy	Top-5 Accuracy	Parameters	Depth
Xception	88 MB	0.790	0.945	22,910,480	126
VGG16	528 MB	0.713	0.901	138,357,544	23
VGG19	549 MB	0.713	0.900	143,667,240	26
ResNet50	98 MB	0.749	0.921	25,636,712	-
ResNet101	171 MB	0.764	0.928	44,707,176	-
ResNet152	232 MB	0.766	0.931	60,419,944	-
ResNet50V2	98 MB	0.760	0.930	25,613,800	-
ResNet101V2	171 MB	0.772	0.938	44,675,560	-
ResNet152V2	232 MB	0.780	0.942	60,380,648	-
InceptionV3	92 MB	0.779	0.937	23,851,784	159
InceptionResNetV2	215 MB	0.803	0.953	55,873,736	572
MobileNet	16 MB	0.704	0.895	4,253,864	88
MobileNetV2	14 MB	0.713	0.901	3,538,984	88
DenseNet121	33 MB	0.750	0.923	8,062,504	121
DenseNet169	57 MB	0.762	0.932	14,307,880	169
DenseNet201	80 MB	0.773	0.936	20,242,984	201
NASNetMobile	23 MB	0.744	0.919	5,326,716	-
NASNetLarge	343 MB	0.825	0.960	88,949,818	-
EfficientNetB0	29 MB	-	-	5,330,571	-
EfficientNetB1	31 MB	-	-	7,856,239	-
EfficientNetB2	36 MB	-	-	9,177,569	-
EfficientNetB3	48 MB	-	-	12,320,535	-
EfficientNetB4	75 MB	-	-	19,466,823	-
EfficientNetB5	118 MB	-	-	30,562,527	-
EfficientNetB6	166 MB	-	-	43,265,143	-
EfficientNetB7	256 MB	-	-	66,658,687	-

Where to next?

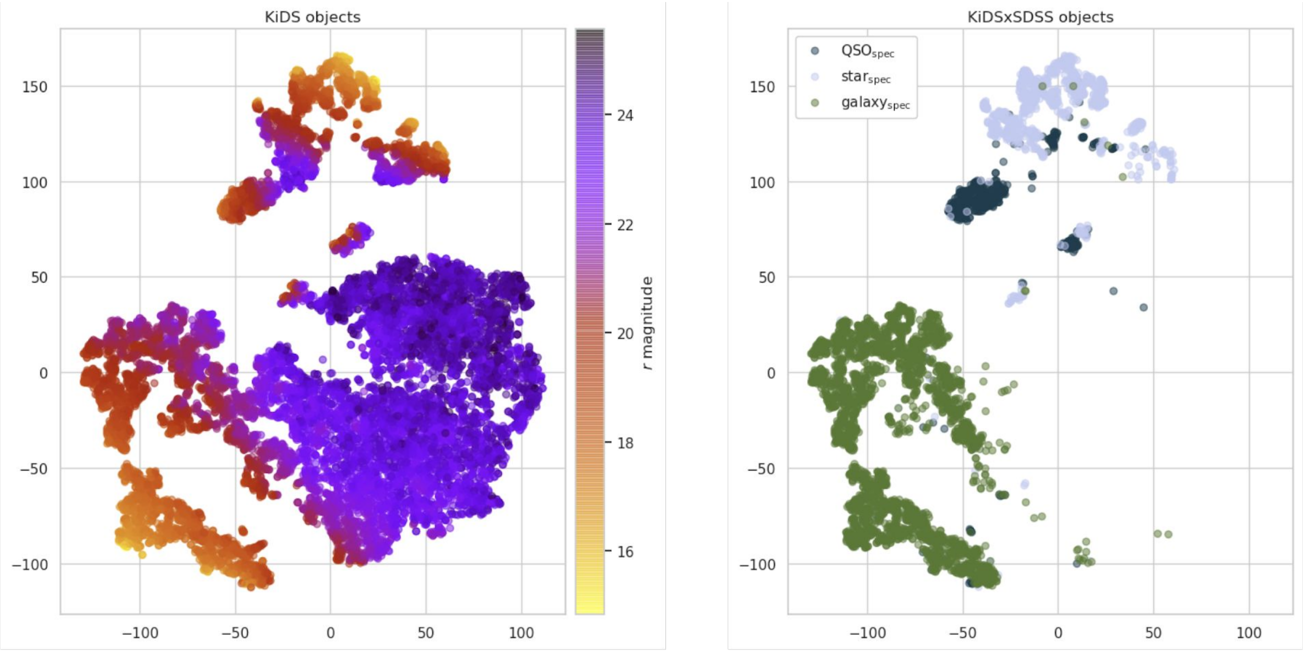
1. Kaggle courses and playground competitions
   https://www.kaggle.com/learn/overview

2. Coursera
   https://www.coursera.org/learn/machine-learning

Examples

Classification

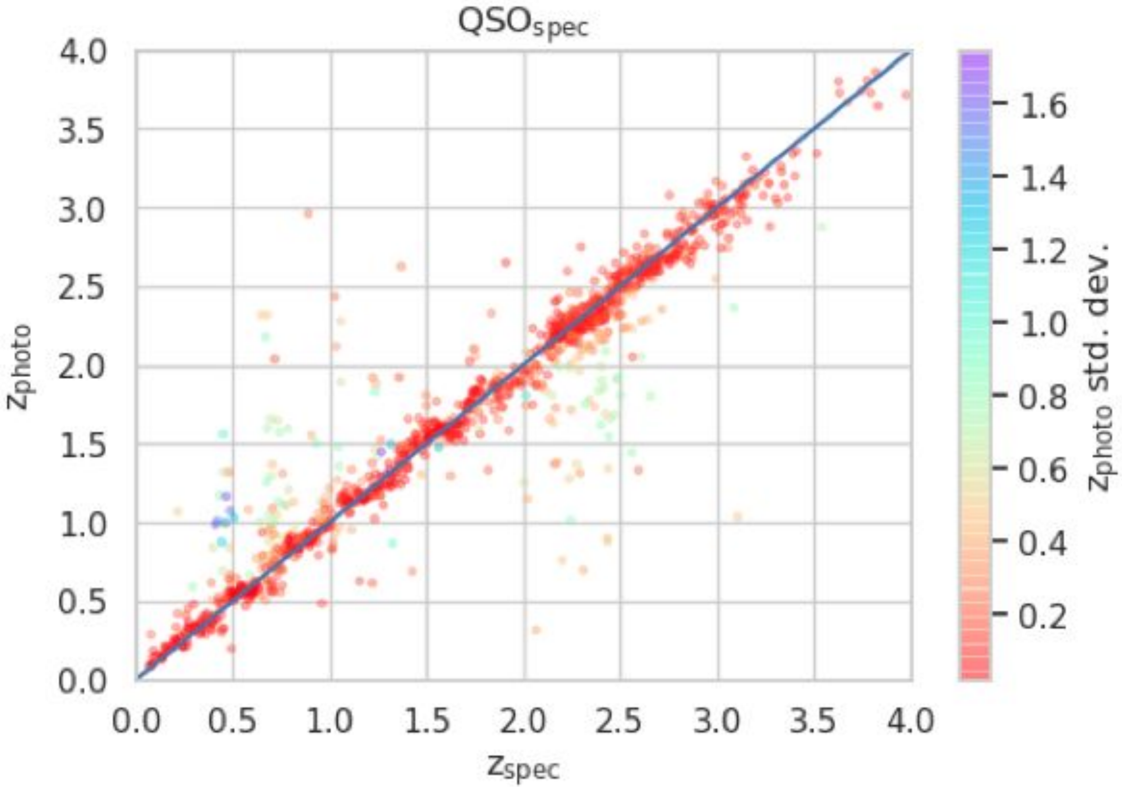
https://ui.adsabs.harvard.edu/abs/2021arXiv210106010B/abstract



t-SNE projection of **KiDSxSDSS data**. *Left*: *r* magnitude, *right*: SDSS spectroscopic classification.

Photometric redshifts

https://ui.adsabs.harvard.edu/abs/2021arXiv210106010B/abstract



https://arxiv.org/pdf/1603.00882.pdf

Supernovae variables

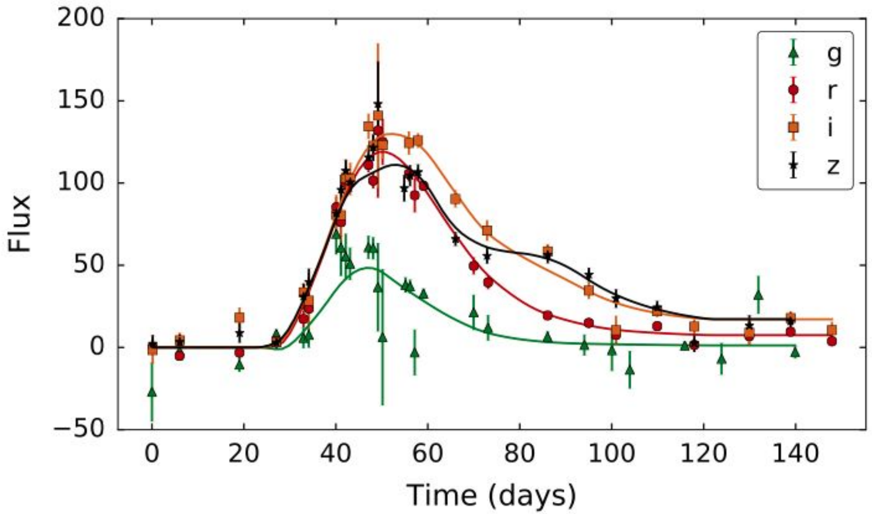


FIG. 1.— An example simulated DES type Ia supernova light curve (object ID 009571), at redshift 0.42, from the Supernova Photometric Classification Challenge (Kessler et al. 2010a,b). The photometric light curve is measured in the four DES *griz* filter bands, showing the rise in brightness after the explosion of the star and subsequent decay. The points and error bars are the data points, while the curves are from the best fitting SALT2 model (see Sec. 3).

https://www.cfa.harvard.edu/~avanderb/kepler90i.pdf

Exoplanets

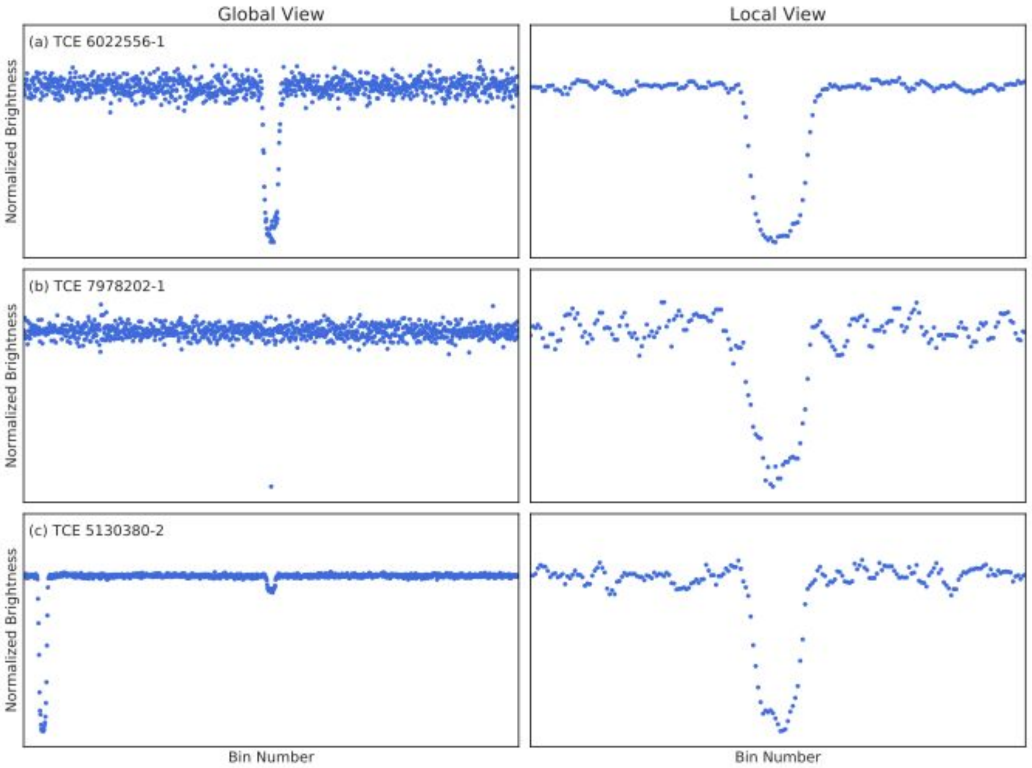


FIG. 3.— Light curve representations that we use as inputs to our neural network models. The "global view" is a fixed-length representation of the entire light curve, and the "local view" is a fixed-length representation of a window around the detected transit. (a) Strong planet candidate. (b) Long-period planet where the transit falls into just one bin in the global view. (c) Secondary eclipse that looks like a planet in the local view.

https://inspirehep.net/literature/1487487

Glitches in LIGO

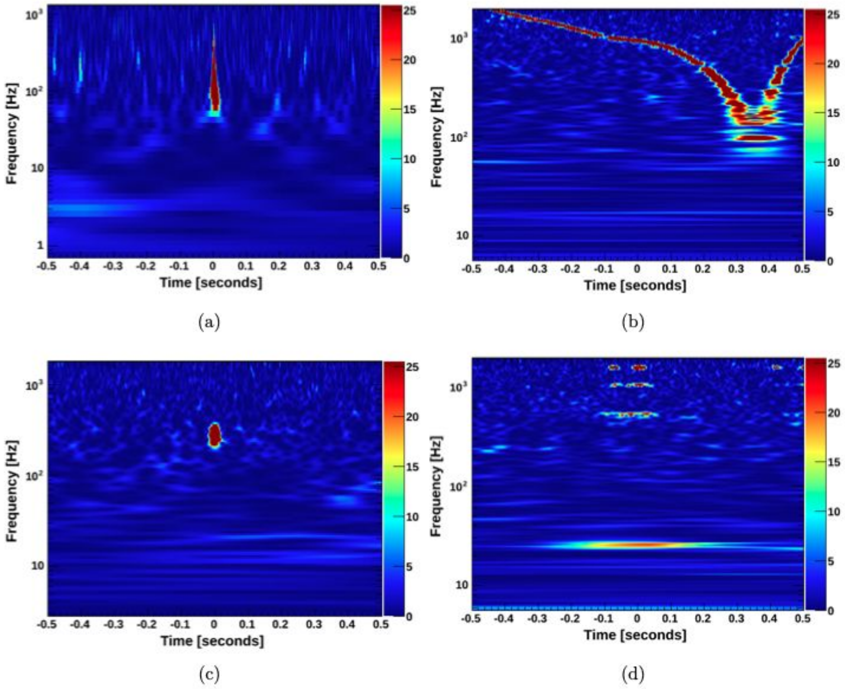


Figure 2. Spectrograms of typical transient types found in the aLIGO Livingston ER7 data. They are generated using the Omega scan tool in LigoDV-Web [14], which matches the data to sine Gaussians. (a) A transient characterized by a tear drop shape in the spectrogram. (b) A "whistle" glitch that often has a long duration and occurs at high frequencies. (c) A hardware injection. (d) A transient type characterized by high frequency lines and lower frequency features.

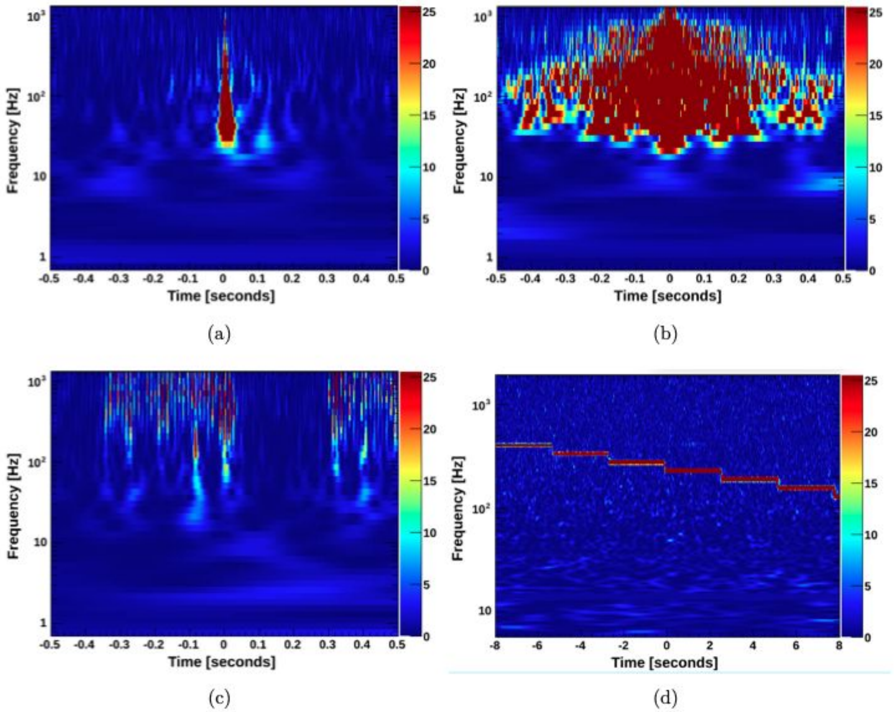
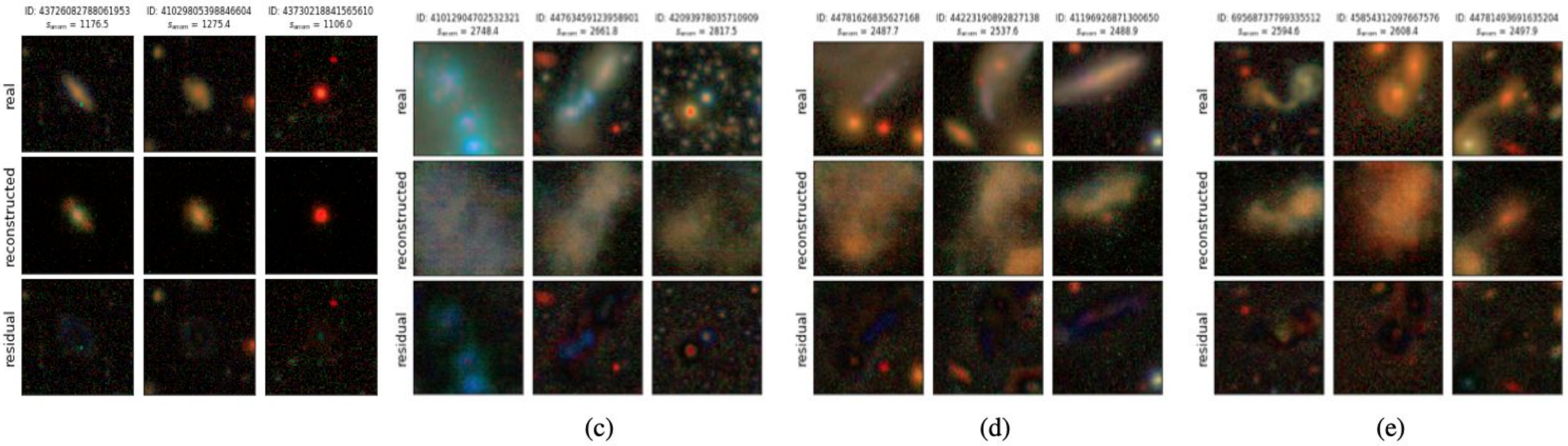


Figure 3. Examples of some of the most common transient types found in the Hanford ER7 data (a) A tear drop glitch. (b) Transients of this type have a large SNR and duration. They created significant drops in the detectors range. (c) A high frequency transient type. (d) A longer duration line occurring at the beginning of a number of data segments.

Anomalies

https://ui.adsabs.harvard.edu/abs/2020arXiv201208082S/abstract



and shown in (c)-(e). We find (c) blue star-forming regions (dark blue stars), (d) extended galaxies with active regions (light blue diamonds), and (e) galaxy mergers (green triangles).

Thank you! Questions?