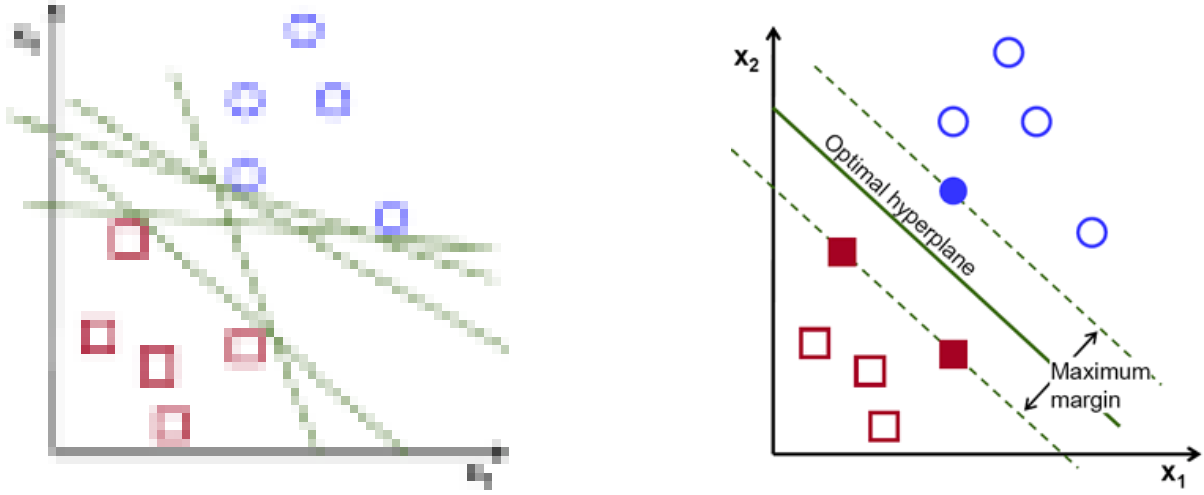
Unit: 5

Topic: Support Vector Machines, Bayesian learning, application of machine learning in computer vision, speech processing, natural language processing etc, Case Study: Image Net Competition

What is Support Vector Machines?

The objective of the support vector machine algorithm is to find a hyper plane in an N-dimensional space (N — the number of features) that distinctly classifies the data points.

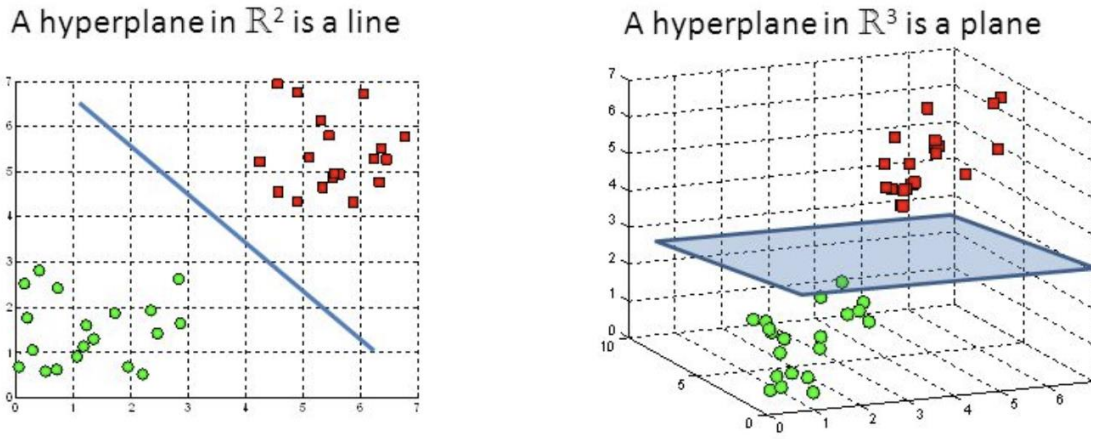
The objective of the support vector machine algorithm is to find a hyper plane in an N-dimensional space (N — the number of features) that distinctly classifies the data points.



Possible hyper planes

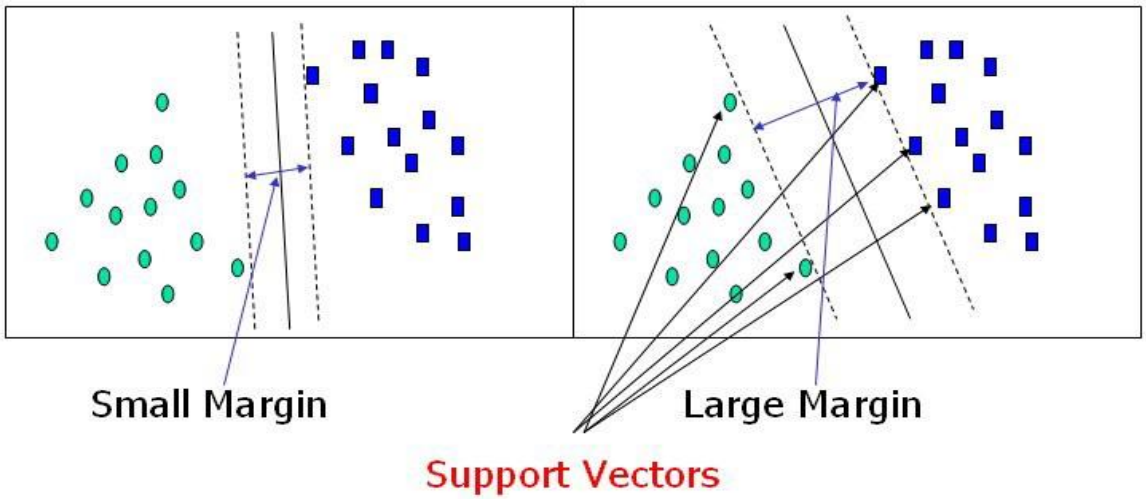
To separate the two classes of data points, there are many possible hyper planes that could be chosen. Our objective is to find a plane that has the maximum margin, i.e. the maximum distance between data points of both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence.

Hyper planes and Support Vectors



Hyper planes in 2D and 3D feature space

Hyper planes are decision boundaries that help classify the data points. Data points falling on either side of the hyper plane can be attributed to different classes. Also, the dimension of the hyper plane depends upon the number of features. If the number of input features is 2, then the hyper plane is just a line. If the number of input features is 3, then the hyper plane becomes a two-dimensional plane. It becomes difficult to imagine when the number of features exceeds 3.



Support vectors are data points that are closer to the hyper plane and influence the position and orientation of the hyper plane. Using these support vectors, we maximize the margin of the classifier. Deleting the support vectors will change the position of the hyper plane. These are the points that help us build our SVM.

Refreance:(https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms 934a444fca47)

Bayesian Learning

Bayesian machine learning is a particular set of approaches to probabilistic machine learning. (for other probabilistic models, see Supervised Learning).

Bayesian learning treats model parameters as random variables - in Bayesian learning, parameter estimation amounts to computing posterior distributions for these random variables based on the observed data.

Bayesian learning typically involves generative models - one notable exception is **Bayesian linear regression**, which is a discriminative model.

Bayesian models

Bayesian modeling treats those two problems as one.

We first have a prior distribution over our parameters (i.e. what are the likely parameters?) $P(\theta)P(\theta)$.

From this we compute a posterior distribution which combines both inference and learning:

$P(y1,\ldots,yn,\theta|x1,\ldots,xn)=P(x1,\ldots,xn,y1,\ldots,yn|\theta)P(\theta)P(x1,\ldots,xn)P(y1,\ldots,yn,\theta|x1,\ldots,xn)=P(x1,\ldots,xn,y1,\ldots,yn|\theta)P(\theta)P(x1,\ldots,xn)$

Then prediction is to compute the conditional distribution of the new data point given our observed data, which is the marginal of the latent variables and the parameters:

$P(xn+1|x1,\ldots,xn)=\int P(xn+1|\theta)P(\theta|x1,\ldots,xn)d\theta P(xn+1|x1,\ldots,xn)=\int P(xn+1|\theta)P(\theta|x1,\ldots,xn)d\theta$

Classification then is to predict the distributions of the new datapoint given data from other classes, then finding the class which maximizes it:

$P(xn+1|xc1,\ldots,xcn)=\int P(xn+1|\theta c)P(\theta c|xc1,\ldots,xcn)d\theta cP(xn+1|x1c,\ldots,xnc)=\int P(xn+1|\theta c)P(\theta c|x1c,\ldots,xnc)d\theta c$

Hidden Markov Models

HMMs can be thought of as clustering over time; that is, each state is a "cluster".

The data points and latent variables are sequences, and $\pi k \pi k$ becomes the transition probability given the state (cluster) kk. $\theta * k \theta k *$ becomes the emission distribution for xx given state kk.

Model-based clustering

- model data from heterogeneous unknown sources
- KK unknown sources (clusters)
- each cluster/source is modelled using a parametric model (e.g. a Gaussian distribution)

For a given data point ii, we have:

$zi|\pi \sim Discrete(\pi) zi|\pi \sim Discrete(\pi)$

Where zizi is the cluster label for which data point ii belongs to. This is the latent variable we want to discover.

$\pi\pi$ is the *mixing proportions* which is the vector of probabilities for each class kk, that is:

$\pi = (\pi i, \ldots, \pi K)|\alpha \sim Dirichlet(\alpha K, \ldots, \alpha K) \pi = (\pi i, \ldots, \pi K)|\alpha \sim Dirichlet(\alpha K, \ldots, \alpha K)$

That is, $\pi k = P(zi = k) \pi k = P(zi = k)$.

We also model each data point xixi as being drawn from a source (cluster) like so, where FF is however we are modeling the cluster (e.g. a Gaussian), parameterized by $\theta * zi \theta zi *$, that is some parameters for the zizi-labeled cluster:

$xi|zi, \theta * k \sim F(\theta * zi) xi|zi, \theta k * \sim F(\theta zi *)$

(Note that the star, as in $\theta * \theta *$, is used to denote the optimal solution for $\theta\theta$.)

For this approach we have two priors over parameters of the model:

- For the mixing proportions, we typically use a Dirichlet prior (above) because it has the nice property of being a conjugate prior with multinomial distributions.
- For each cluster kk we use some prior HH, that is $\theta * k|H \sim H \theta k * |H \sim H$.

Graphically, this is: Model-based clustering plate model

Naive Bayes

The main assumption of Naive Bayes is that all features are independent effects of the label. This is a really strong simplifying assumption but nevertheless in many cases Naive Bayes performs well.

Naive Bayes is also *statistically efficient* which means that it doesn't need a whole lot of data to learn what it needs to learn.

If we were to draw it out as a Bayes' net:

YYY→F1→F2…→FnY→F1Y→F2…Y→Fn

Where YY is the label and F1,F2,…,FnF1,F2,…,Fn are the features.

The model is simply:

P(Y|F1,…,Fn)∝P(Y)∏iP(Fi|Y)P(Y|F1,…,Fn)∝P(Y)∏iP(Fi|Y)

This just comes from the Bayes' net described above.

The Naive Bayes learns P(Y,f1,f2,…,fn)P(Y,f1,f2,…,fn) which we can normalize (divide by P(f1,…,fn)P(f1,…,fn)) to get the conditional probability P(Y|f1,…,fn)P(Y|f1,…,fn):

P(Y,f1,…,fn)=P(y1,f1,…,fn)P(y2,f1,…,fn)∶P(yk,f1,…,fn)=P(y1)∏iP(fi|y1)P(y2)∏iP(fi|y2)∶P(yk)∏iP(fi|yk)P(Y,f1,…,fn)=P(y1,f1,…,fn)P(y2,f1,…,fn)∶P(yk,f1,…,fn)=P(y1)∏iP(fi|y1)P(y2)∏iP(fi|y2)∶P(yk)∏iP(fi|yk)

So the parameters of Naive Bayes are P(Y)P(Y) and P(Fi|Y)P(Fi|Y) for each feature.

Inference in Bayesian models

Maximum a posteriori (MAP) estimation

A Bayesian alternative to MLE, we can estimate probabilities using *maximum a posteriori estimation*, where we instead choose a probability (a point estimate) that is most likely given the observed data:

π~MAPP(y|X)=argmaxπP(π|X)=argmaxπP(X|π)P(π)P(X)=argmaxπP(X|π)P(π)≈P(y|π~MAP)π

~MAP=argmaxπ⁡P(π|X)=argmaxπ⁡P(X|π)P(π)P(X)=argmaxπ⁡P(X|π)P(π)P(y|X)≈P(y|π~MAP)

So unlike MLE, MAP estimation uses Bayes' Rule so the estimate can use prior knowledge (P(π)P(π)) about what we expect ππ to be.

Again, this may be done with log-likelihoods:

$\theta MAP = \arg\max_\theta p(\theta|x) = \arg\max_\theta \log p(x|\theta) + \log p(\theta)$

Maximum A Posteriori (MAP)

Likelihood function $L(\theta)$ is the probability of the data D as a function of the parameters θ.

This often has very small values so typically we work with the log-likelihood function instead:

$\ell(\theta) = \log L(\theta)$

The *maximum likelihood criterion* simply involves choosing the parameter θ to

maximize $\ell(\theta)$. This can (sometimes) be done analytically by computing the derivative

and setting it to zero and yields the *maximum likelihood estimate*.

MLE's weakness is that if you have only a little training data, it can overfit. This problem is known as *data sparsity*. For example, you flip a coin twice and it happens to land on heads both times. Your maximum likelihood estimate for θ (probability that the coin lands on heads) would be 1! We can then try to generalize this estimate to another dataset and test it by measuring the log-likelihood on the test set. If a tails shows up at all in the test set, we will have a test log-likelihood of $-\infty$.

We can instead use Bayesian techniques for parameter estimation. In Bayesian parameter estimation, we treat the parameters θ as a random variable as well, so we learn a joint distribution $p(\theta, D)$.

We first require a prior distribution $p(\theta)$ and the likelihood $p(D|\theta)$ (as with maximum likelihood).

We want to compute $p(\theta|D)$, which is accomplished using Bayes' rule:

$p(\theta|D) = \frac{p(\theta)p(D|\theta)}{\int p(\theta')p(D|\theta')d\theta'} p(\theta|D) = \frac{p(\theta)p(D|\theta)}{\int p(\theta')p(D|\theta')d\theta'}$

Though we work with only the numerator for as long as possible (i.e. we delay normalization until it's necessary):

$p(\theta|D) \propto p(\theta)p(D|\theta) p(\theta|D) \propto p(\theta)p(D|\theta)$

The more data we observe, the less uncertainty there is around the parameter, and the likelihood term comes to dominate the prior - we say that the *data overwhelm the prior*.

We also have the *posterior predictive distribution* p(D′|D)p(D′|D), which is the distribution over future observables given past observations. This is computed by computing the posterior over θθ and then marginalizing out θθ:

$$p(D'|D)=\int p(\theta|D)p(D'|\theta)d\theta p(D'|D)=\int p(\theta|D)p(D'|\theta)d\theta$$

The normalization step is often the most difficult, since we must compute an integral over potentially many, many parameters.

We can instead formulate Bayesian learning as an optimization problem, allowing us to avoid this integral. In particular, we can use *maximum a-posteriori* (MAP) approximation.

Whereas with the previous Bayesian approach (the "full Bayesian" approach) we learn a distribution over θθ, with MAP approximation we simply get a point estimate (that is, a single value rather than a full distribution). In particular, we get the parameters that are most likely under the posterior:

$$\theta\text{^MAP}=\text{argmax}_\theta p(\theta|D)=\text{argmax}_\theta p(\theta,D)=\text{argmax}_\theta p(\theta)p(D|\theta)=\text{argmax}_\theta \log p(\theta)+\log p(D|\theta)\theta\text{^}$$
$$\text{MAP}=\text{argmax}_\theta p(\theta|D)=\text{argmax}_\theta p(\theta,D)=\text{argmax}_\theta p(\theta)p(D|\theta)=\text{argmax}_\theta \log p(\theta)+\log p(D|\theta)$$

Refreance:(https://frnsys.com/ai_notes/machine_learning/bayesian_learning.html)

Application of Machine learning in Computer Vision

It is not just the performance of machine learning/deep learning models on benchmark problems that is most interesting; it is the fact that a single model can learn meaning from images and performs vision tasks, obviating the need for a pipeline of specialized and hand-crafted methods.

We will look at the following computer vision problems where deep learning has been used:

1. Image Classification
2. Image Classification With Localization
3. Object Detection
4. Object Segmentation
5. Image Style Transfer
6. Image Colorization
7. Image Reconstruction
8. Image Super-Resolution
9. Image Synthesis
10. Other Problems

Image Classification

Image classification involves assigning a label to an entire image or photograph.

This problem is also referred to as "object classification" and perhaps more generally as "image recognition," although this latter task may apply to a much broader set of tasks related to classifying the content of images.

Some examples of image classification include:

- Labeling an x-ray as cancer or not (binary classification).
- Classifying a handwritten digit (multiclass classification).
- Assigning a name to a photograph of a face (multiclass classification).
- A popular example of image classification used as a benchmark problem is the MNIST dataset.

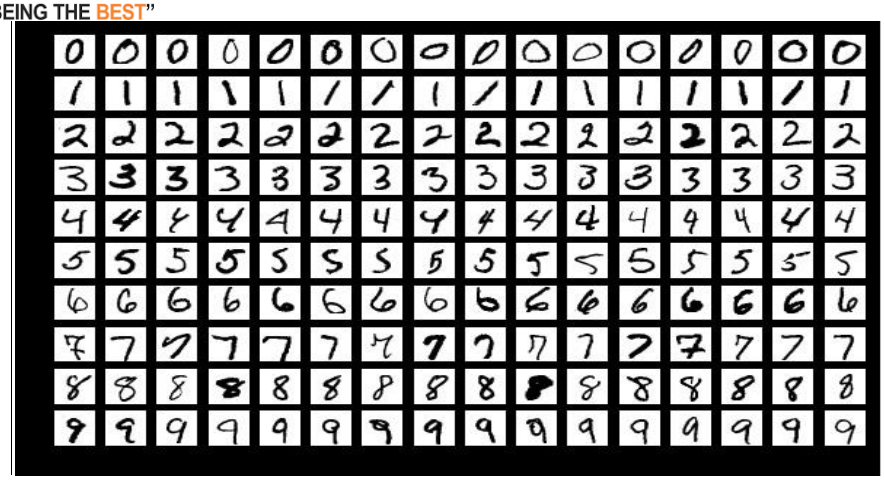


Image Classification With Localization

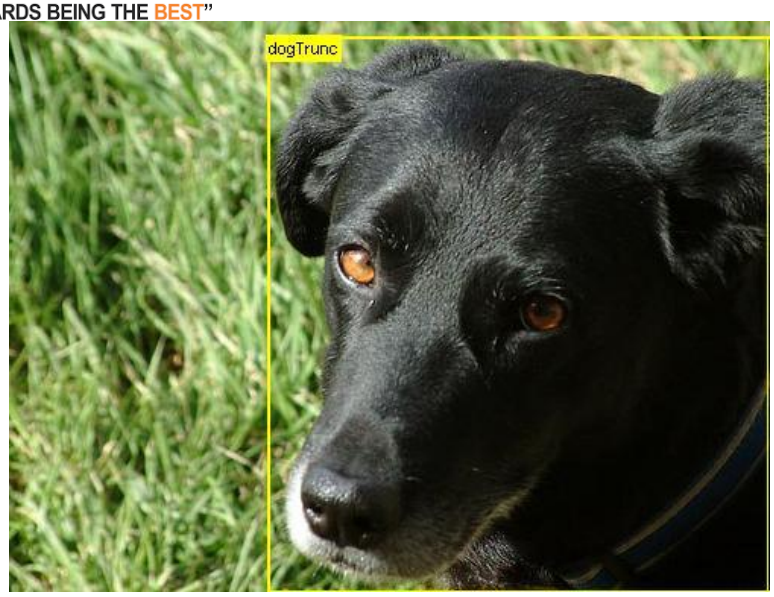
Image classification with localization involves assigning a class label to an image and showing the location of the object in the image by a bounding box (drawing a box around the object).

This is a more challenging version of image classification.

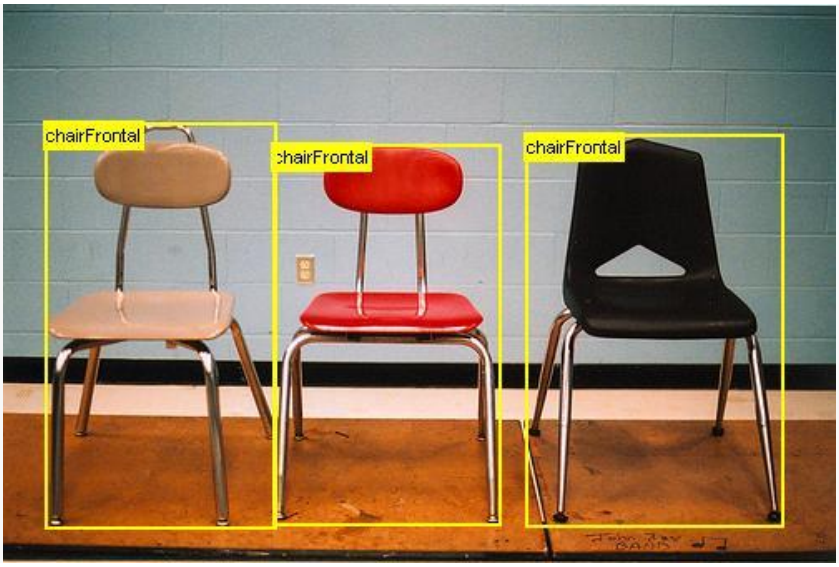
Some examples of image classification with localization include:

- Labeling an x-ray as cancer or not and drawing a box around the cancerous region.
- Classifying photographs of animals and drawing a box around the animal in each scene.

A classical dataset for image classification with localization is the PASCAL Visual Object Classes datasets, or PASCAL VOC for short (e.g. VOC 2012). These are datasets used in computer vision challenges over many years.



The task may involve adding bounding boxes around multiple examples of the same object in the image. As such, this task may sometimes be referred to as "*object detection*."



Example of Image Classification With Localization of Multiple Chairs From VOC 2012

The ILSVRC2016 Dataset for image classification with localization is a popular dataset comprised of 150,000 photographs with 1,000 categories of objects.

Some examples of papers on image classification with localization include:

- Selective Search for Object Recognition, 2013.
- Rich feature hierarchies for accurate object detection and semantic segmentation, 2014.
- Fast R-CNN, 2015.

Object Detection

Object detection is the task of image classification with localization, although an image may contain multiple objects that require localization and classification.

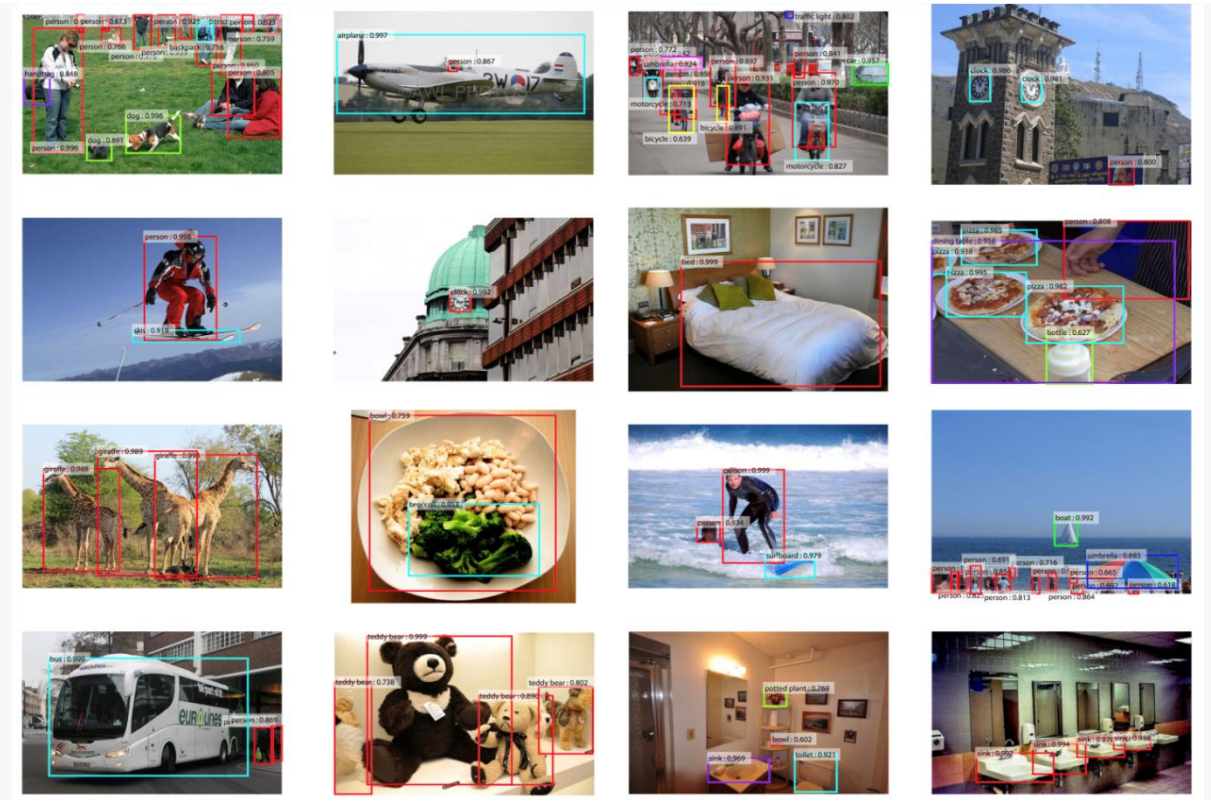
This is a more challenging task than simple image classification or image classification with localization, as often there are multiple objects in the image of different types.

Often, techniques developed for image classification with localization are used and demonstrated for object detection.

Some examples of object detection include:

- Drawing a bounding box and labeling each object in a street scene.
- Drawing a bounding box and labeling each object in an indoor photograph.
- Drawing a bounding box and labeling each object in a landscape.

The PASCAL Visual Object Classes datasets, or PASCAL VOC for short (e.g. VOC 2012), is a common dataset for object detection.
Another dataset for multiple computer vision tasks is Microsoft's Common Objects in Context Dataset, often referred to as MS COCO.



Example of Object Detection With Faster R-CNN on the MS COCO Dataset

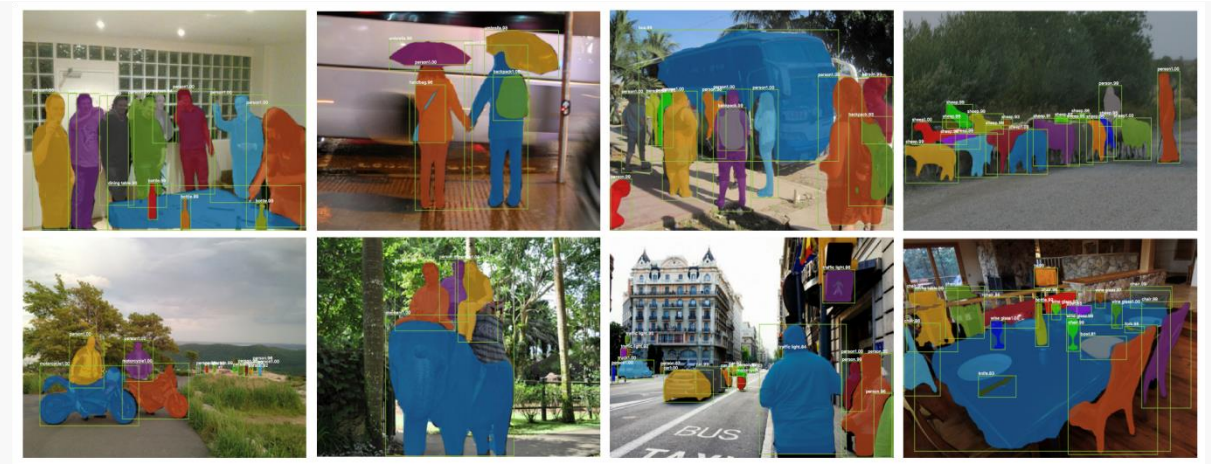
Some examples of papers on object detection include:

- Over Feat: Integrated Recognition, Localization and Detection using Convolutional Networks, 2014.
- Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, 2015.
- You Only Look Once: Unified, Real-Time Object Detection, 2015.

Object Segmentation

Object segmentation, or semantic segmentation, is the task of object detection where a line is drawn around each object detected in the image. Image segmentation is a more general problem of spitting an image into segments.

Object detection is also sometimes referred to as object segmentation. Unlike object detection that involves using a bounding box to identify objects, object segmentation identifies the specific pixels in the image that belong to the object. It is like a fine-grained localization. More generally, "*image segmentation*" might refer to segmenting all pixels in an image into different categories of object. Again, the VOC 2012 and MS COCO datasets can be used for object segmentation.



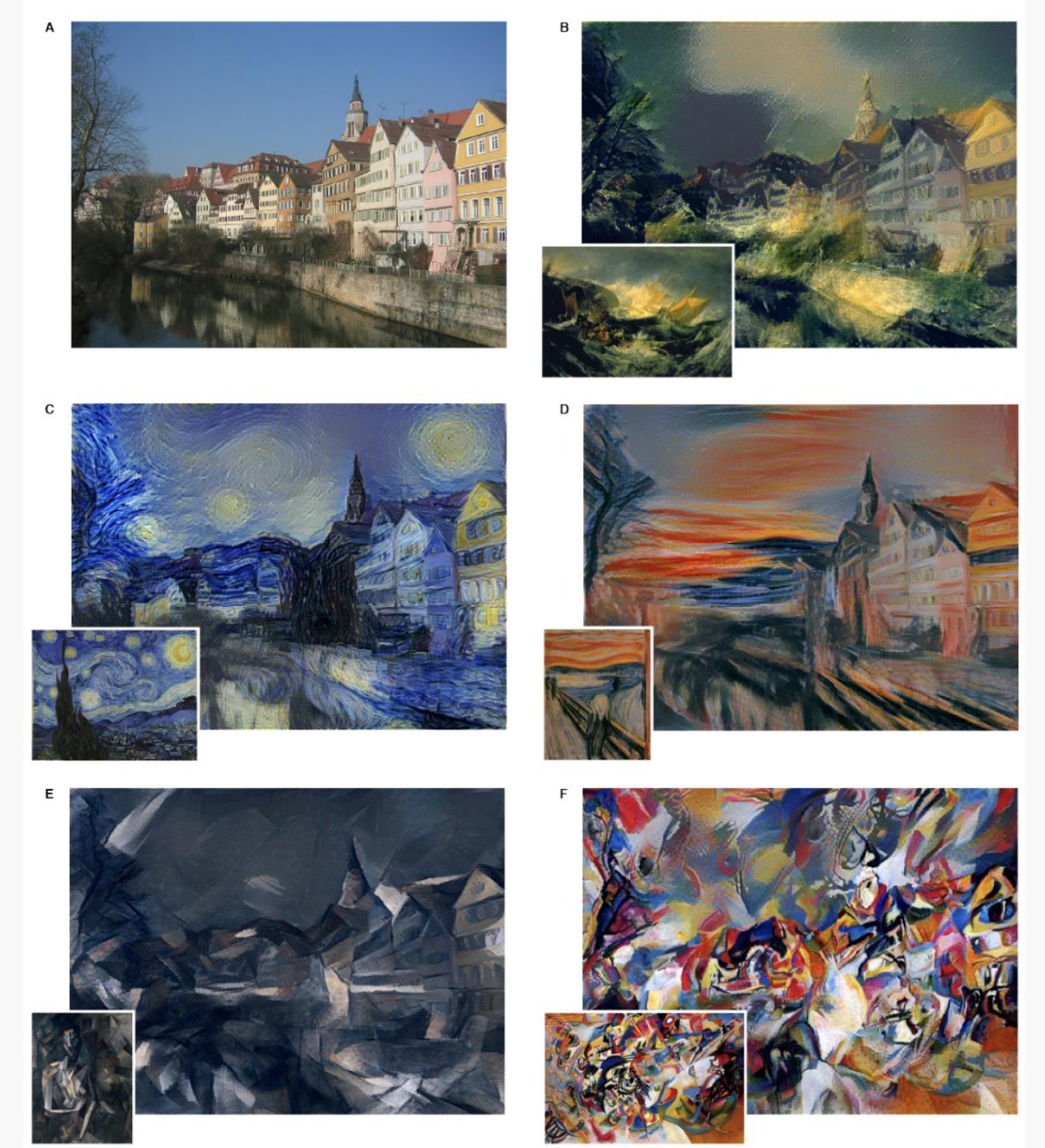
STYLE TRANSFER

Style transfer or neural style transfer is the task of learning style from one or more images and applying that style to a new image.

This task can be thought of as a type of photo filter or transform that may not have an objective evaluation.

Examples include applying the style of specific famous artworks (e.g. by Pablo Picasso or Vincent van Gogh) to new photographs.

Datasets often involve using famous artworks that are in the public domain and photographs from standard computer vision datasets.



Example of Neural Style Transfer From Famous Artworks to a Photograph Taken from "A Neural Algorithm of Artistic Style"

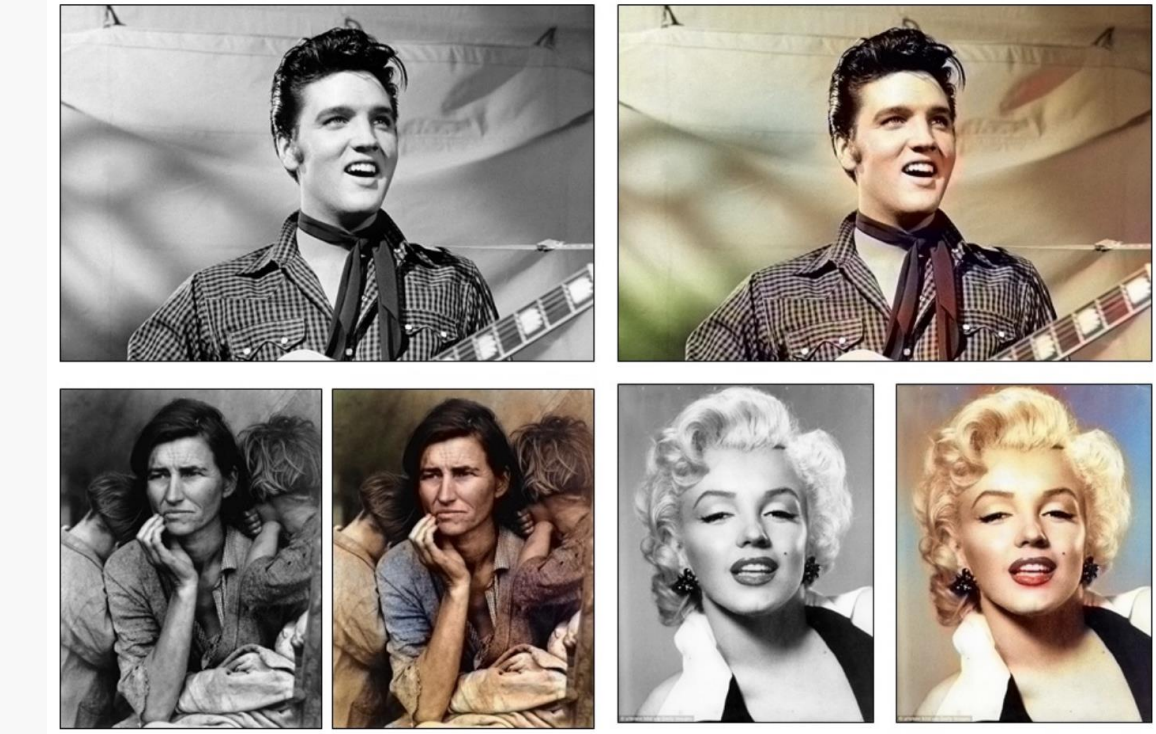
Image Colorization

Image colorization or neural colorization involves converting a grayscale image to a full color image.

This task can be thought of as a type of photo filter or transform that may not have an objective evaluation.

Examples include colorizing old black and white photographs and movies.

Datasets often involve using existing photo datasets and creating grayscale versions of photos that models must learn to colorize.



Refreance:(https://machinelearningmastery.com/applications-of-deep-learning-for-computer-vision/)

Speech processing

As a sub-field of Artificial Intelligence (AI) technology, machine learning is the method of data analysis which constructs analytical models automatically. This is a promising technology to provide the most optimal support for businesses with a variety of real-world applications, such as speech recognition and image recognition.

Machine learning uses iterative algorithms to learn from data and allows the computer to find information, hidden values that are not explicitly programmed. The repetitive aspect of Machine learning is important because when these models are exposed to new data, they can adapt independently. Machine Learning systems can quickly apply knowledge and training from large datasets to perform face recognition, speech recognition, and more.

Image Recognition

One of the most common uses of machine learning is image recognition. There are many situations where you can classify the object as a digital image. For digital images, the measurements describe the outputs of each pixel in the image.

In the case of a black and white image, the intensity of each pixel serves as one measurement. So if a black and white image has N*N pixels, the total number of pixels and hence measurement is N2.

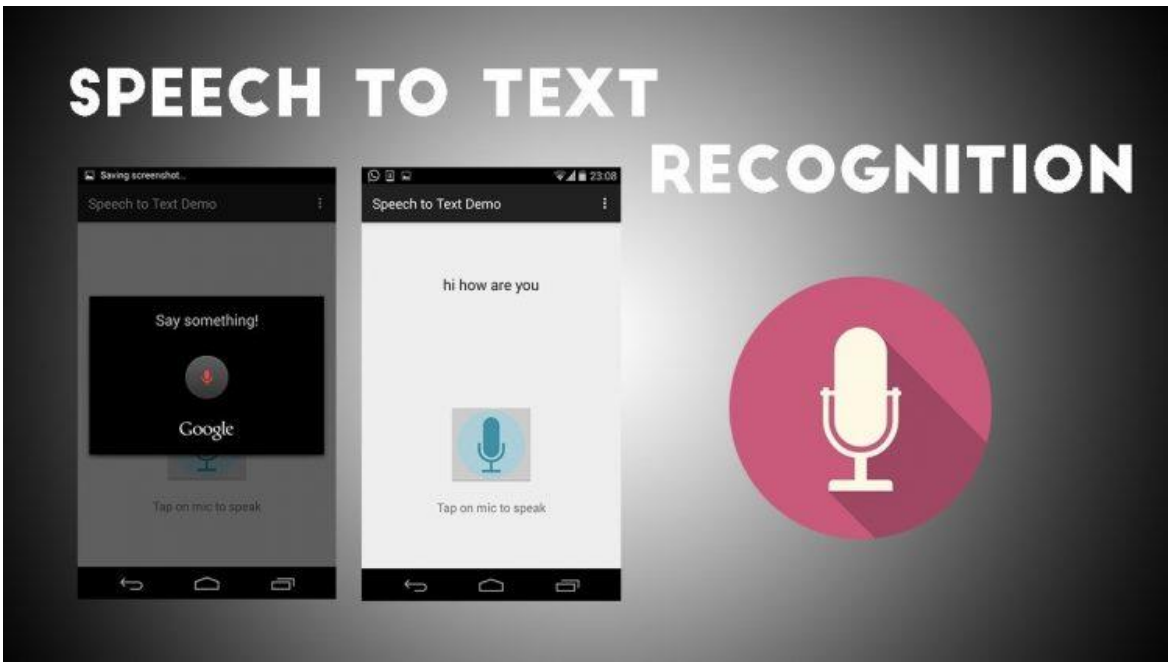
In the colored image, each pixel considered as providing 3 measurements to the intensities of 3 main color components ie RGB. So N*N colored image there are 3 N2 measurements.

- For face detection – The categories might be face versus no face present. There might be a separate category for each person in a database of several individuals.
- For character recognition – We can segment a piece of writing into smaller images, each containing a single character. The categories might consist of the 26 letters of the English alphabet, the 10 digits, and some special characters.

Image recognition system uses the machine learning technology is being used by Google in their products such as Google Photos, Google Search, Google Drive … to optimize the image detection through the keyword search of user.

Speech Recognition

Speech recognition (SR) is the translation of spoken words into text. It is also known as "*automatic speech recognition*" (ASR), "*computer speech recognition*", or "*speech to text*" (STT).



The application translating spoken words into text.

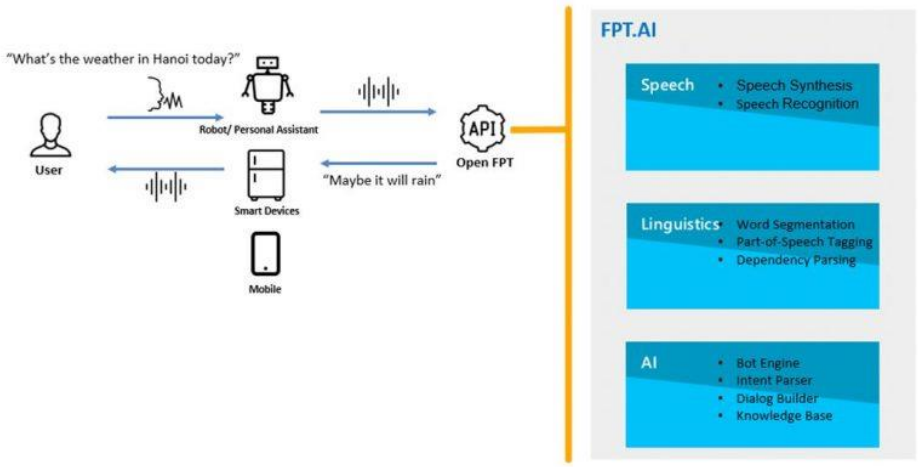
In speech recognition, a software application recognizes spoken words. The measurements in this application might be a set of numbers that represent the speech signal. We can segment the signal into portions that contain distinct words or phonemes. In each segment, we can represent the speech signal by the intensities or energy in different time-frequency bands.

Although the details of signal representation are outside the scope of this program, we can represent the signal by a set of real values.

Speech recognition applications include voice user interfaces. Voice user interfaces are such as voice dialing, call routing, domotic appliance control. It can also use as simple data entry, preparation of structured documents, speech-to-text processing, and plane.

Using Machine Learning, Baidu's research and development department have created a tool called Deep Voice – a deep neural network that is capable of producing artificial voices that are difficult to distinguish from real human voice. This network can "*learn*" features in rhythm, voice, pronunciation, and vocalization to create the voice of the speaker. In addition, Google also uses Machine Learning for other voice-related products and translations such as Google Translate, Google Text To Speech, Google Assistant.

Besides the applications in audio recognition and image recognition, Machine learning is also applied in areas such as medical analysis; arranging, classifying; data analysis and forecasting, etc, in the field such as healthcare, financial services, transportation, marketing & sale…In a near day, Devices and applications based on Machine learning technology may appear in all aspects of human life.



FPT.AI – New Generation Conversation Platform and Virtual Assistant

In order to catch up with the modern technology trend, FPT has been using Machine learning in most FPT applications and technology products such as FPT.AI – New Generation Conversation Platform and Virtual Assistant, PeoIed identification in FPT Shop, Autonomous car, Human Machine Interface – TTS, STT.

Refreance:(https://techinsight.com.vn/language/en/image-recognition-speech-recognition-machine-learning-applications-real-world/)

Natural Language Processing

NLP is a field in machine learning with the ability of a computer to understand, analyze, manipulate, and potentially generate human language.

NLP in Real Life

- Information Retrieval (Google finds relevant and similar results).
- Information Extraction (Gmail structures events from emails).
- Machine Translation (Google Translate translates language from one language to another).
- Text Simplification (Rewordify simplifies the meaning of sentences). Shashi Tharoor tweets could be used (pun intended).
- Sentiment Analysis (Hater News gives us the sentiment of the user).
- Text Summarization (Smmry or Reddit's autotldr gives a summary of sentences).
- Spam Filter (Gmail filters spam emails separately).
- Auto-Predict (Google Search predicts user search results).
- Auto-Correct (Google Keyboard and Grammarly correct words otherwise spelled wrong).
- Speech Recognition (Google Web Speech or Vocal ware).
- Question Answering (IBM Watson's answers to a query).
- Natural Language Generation (Generation of text from image or video data.)

Reference :1. Christopher M. Bishop, "Pattern Recognition and Machine Learning", Springer-Verlag New York Inc., 2nd Edition, 2011.
2. Tom M. Mitchell, "Machine Learning", McGraw Hill Education, First edition, 2017.
3. Ian Goodfellow and Yoshua Bengio and Aaron Courville, "Deep Learning", MIT Press.

(https://towardsdatascience.com/natural-language-processing-nlp-for-machine-learning-d44498845d5b)