机器学习：

深度学习：

迁移学习：

强化学习：

对抗学习：

机器学习对样本数据分布的假设：独立同分布

过拟合：

欠拟合：

模型退化：

梯度消失：

梯度爆炸：

数据爆炸：

梯度弥散：

损失函数

优化器

学习率

模型权重（偏置）

Dropout：将输入张量里的元素按照一定概率置为0，相当于丰富了数据的多样性，起到正则化的效果，有利于防止过拟合

Batch Normalization

图像分类（图像识别）

目标检测

图像分割（语义分割、实例分割）

模型压缩

深度可分离卷积（Depthwise Separable Convolution）：分为Depthwise Convolution与Pointwise Convolution两步，depthwise层只改变feature map的大小，不改变通道数，feature map中的每个通道只被一个卷积核卷积。而Pointwise 层则相反，只改变通道数，不改变大小，它的卷积核尺寸为1x1，卷积核的个数决定输出的feature map的通道数。主要作用是减少了参数量。

分组卷积（Group Convolution）：

双目测距

Feature map：神经网络输出结果中的一个通道叫做一个feature map，所有feature map加起来组成了一个输出层结果。

FLOPs（floating-point operations per second）：每秒浮动运算次数

网络的宽度，深度，每层的厚度的影响：根据以往的经验，增加网络的深度depth能够得到更加丰富、复杂的特征并且能够很好的应用到其他任务中，但网络的深度过深会面临梯度消失，训练困难的问题；增加网络的width能够获得更细粒度的特征并且也更容易训练，但对于width很大而且深度较浅的网络往往很难学习到更深层次的特征；增加输入网络的图像分辨率能够潜在得获得更高细粒度的特征模板，但对于非常高的输入分辨率，准确度的增益也会减少，并且大分辨率图像会增加计算量。

SOTA：

神经网络架构搜索（NAS）：

帕累托最优：

数据并行：

模型并行：

正则化：为模型损失函数添加惩罚项，是模型学出的参数值较小，让某些特征的权重变得更小，有利于防止过拟合

标准化：将数据按特定的分布（比如标准正态分布）进行缩放

归一化：将数据按比例缩放，使之落入一个小的特定空间（例如区间[0, 1]）

奇异值分解（singular value decomposition，SVD）：

网络剪枝（Network Pruning）：

余弦退火算法：

贝叶斯优化：

遗传算法：

注意力机制：观察事物时，把注意力放在最具辨识度的部分，而非从头到尾地观察。注意力机制需要三个指定的输入：Q(query)，K(key)，V(Value)，然后通过公式得到注意力的计算结果，这个结果代表query在key和value作用下的表示。Q是一段准备被概括的文本，K是给出的提示，V是大脑中对提示K的延伸，当Q=K=V时，称作自注意力机制。

多头注意力机制用多种注意力机制均衡同一种注意力机制可能产生的偏差

语言模型：nlp中的语言模型是计算句子（单词序列）的概率或序列中下一个单词的概率模型。语言模型以句子（单词序列）为输入，利用序列关系特征，输出一个在所有词汇上的概率分布

精确率（precision）：TP/(TP+FP)，预测为正标签的样本中，有多少真实标签也为正

召回率（recall）：TP/(TP+FN)，真实标签为正的样本中，有多少被正确预测出来。在检测任务中，召回率等于1-漏检率

IoU：检测结果中预测值和真实值的交集面积除以并集面积

mAP：各个检测类别对应的PR曲线下的面积的平均值

梯度：梯度是方向指向函数值增长最快的方向，大小为变化率，是有方向和大小的矢量。梯度的求解其实就是求函数偏导的问题。导数对应一元函数，偏导数对应多元函数。

反向传播：损失函数对参数（w，b）求导，过程为根据链式求导法则先对激活函数求导，然后对权重参数求导

权重更新：通过反向传播得到各个权重的梯度，用原来的权重减去学习率乘以梯度，就是更新后的权重。更新后的权重参数 = 原来的权重参数 – 学习率 x 权重的梯度。

随机梯度下降：在一个batch中随机选择一个样本来求梯度值进行权重更新，而不是求batch中所有样本权重梯度的平均值

权重衰退：weight\_decay，在损失函数中加入对权重参数的L2正则化项，为保证损失函数尽可能小，所以权重会倾向于训练出较小的值。权重值较小，说明某些多项式分支的作用很小，降低了模型的复杂程度，模型简单，泛化能力强，可以解决过拟合问题。当前迭代梯度 = 当前迭代梯度 + 权重衰退 x 上一次迭代的参数

动量：momentum，更新权重的时候在一定程度上保留之前更新的方向，同时利用当前batch的梯度微调最终的更新方向，可以在一定程度上增加稳定性，从而学习地更快，并且还有一定摆脱局部最优的能力。当前迭代梯度 = 动量 x 上一次迭代的梯度 + （1-抑制） x 当前迭代的梯度