**Boosting VS. Bagging**相同点：基预测器的种类单 不同点： Bagging：基预测器可并行训练 主要在优化variance（即模型的鲁棒性）bagging 方法可以减小过拟合，所以通常在强分类器和复杂模型上使用时表现的很好（例如，完全生长的决策树，fully developed decision trees） boosting 方法 基预测器串行训练 boosting主要在优化bias（即模型的精确性）在弱模型上表现更好（例如，浅层决策树）**XGBoost改进点**： 损失函数的改变：（导数和正则项）用泰勒展开式对模型损失残差的近似；同时XGBoost对模型的损失函数进行的改进，并加入了模型复杂度的正则项 工具的优化：（趋势值和并行）shrinkage（缩减），相当于学习速率（XGBoost中学习速率为eta） 列抽样 对缺失值的处理 XGBoost工具支持并行

与XGBoost对比：1. XGBoost采用的是level-wise的分裂策略， 而LightGBM采用了leaf-wise的策略2. LightGBM使用了基于histogram的决策树算法，这一点不同与XGBoost中的

exact 算法（tree\_method 可以使用 hist参数）， histogram算法在内存和计算代价上都有不小优势。（1）内存上优势。（2）计算上的优势。3. 直方图做差加速一个子节点的直方图可以通过父节点的直方图减去兄弟节点的直方图得到，从而加速计算。4. LightGBM支持直接输入categorical 的feature在对离散特征分裂时，每个取值都当作一个桶，分裂时的增益算的是”是否属于某个category“的gain。类似于one-hot编码。5. 多线程优化 **神经网络**1. 定义问题，收集数据集2. 选择衡量成功的指标 要在验证数据上监控哪些指标？3. 确定评估方法 留出验证？ K折验证？你应该将哪一部分数据用于验证？4. 准备数据5. 开发比基准更好的模型6. 扩大模型规模：开发过拟合的模型7. 模型正则化与调节超参数：基于模型在验证数据上的性能 （目前被过度关注） **微调**，是指将其卷机基的顶部（靠近输出层）的几层“解冻”，并将这解冻的几层和新增加的部分（本例中是全连接分类器）联合训练。• 微调时，要求解冻层的权值更新不要变化太大，因而学习率设置得偏小。**VGG成功的原因**• 更深的卷积神经网络，更多的卷积层和非线性激活函数，提升分类准确率• 使用规律的多层小卷积替代大卷积，减少参数数量，提高训练收敛速度• 部分网络层参数的预初始化，提高训练收敛速度**AlexNet**• Local Response Normalization (LRN)• 局部响应归一化：• 对图像的每个“位置”，提升高响应特征，抑制低响应特征；• 减少高激活神经元数量，提高训练速度，抑制过拟合；• 但后来被研究者发现无明显效果，故现在很少使用**1×1卷积层的作用**：保持输入的宽和高不变，可自主控制输出的通道数。• 给神经网络添加了一个非线性函数，从而减少或保持输入层中的通道数量不变；也可以增加通道数量。• 使用1×1卷积，形成“瓶颈层”，可有效减少计算量和参数量**R-CNN**有很多缺点：重复计算： R-CNN虽然不再是穷举，但依然有两千个左右的候选框，这些候选框都需要进行CNN操作，计算量依然很大，其中有不少其实是重复计算；SVM模型：而且还是线性模型，在标注数据不缺的时候显然不是最好的选择；训练测试分为多步：区域提名、特征提取、分类、回归都是断开的训练的过程，中间数据还需要单独保存； 训练的空间和时间代价很高GPU上处理一张图片需要13秒， CPU上则需要53秒。**主流的目标检测算法**（1） One-Stage目标检测算法，这类检测算法不需要Region Proposal阶段，可以通过一个Stage直接产生物体的类别概率和位置坐标值，比较典型的算法有YOLO、SSD和CornerNet；（2） Two-Stage目标检测算法，这类检测算法将检测问题划分为两个阶段：• 第一个阶段首先产生候选区域（Region Proposals），包含目标大概的位置信息；• 第二个阶段对候选区域进行分类和位置精修，这类算法的典型代表有R-CNN (Region-CNN)， Fast R-CNN， Faster RCNN等

**Deep learning**训练过程中的“三部曲”，与机器学习中的“三部曲”进行对照比较异同Model定义：定义一个network structure反向传播：用于更新模型参数 会计算梯度（即偏导）

前向传播：用于预测输出，计算Loss**Fast R-CNN**• 改进之处：（与R-CNN相比）• 直接对输入的整张图像做卷积，不再对每个候选区域分别做卷积，从而减少大量重复计算；• 用ROI pooling对不同候选框的特征进行尺寸归一化；• 多任务损失函数：将边界框回归器放进网络一起训练，每个类别对应一个回归器；• 用softmax代替SVM分类器• 缺点：• 候选区域提取仍使用selective search，目标检测时间大多消耗在selective search。**YOLO**的核心思想:利用整张图作为网络的输入，直接在输出层回归bounding box的位置和bounding box所属的类别。 每个bounding box要预测(x, y, w, h)和confidence共5个值 每个网格还要预测一个类别信息，记为C类 (1) 给个一个输入图像，首先将图像划分成SxS的网格(2) 对于每个网格，我们都预测B个边框（每个边框的预测值包括：每个边框是目标的置信度以及每个边框区域在多个类别上的概率）(3) 根据上一步可以预测出SxSxB个目标窗口，然后根据阈值去除可能性比较低的目标窗口，最后NMS去除冗余窗口即可。 **总结**： YOLO对相互靠的很近的物体，还有很小的群体 检测效果不好，这是因为一个网格中只预测了两个框，并且只属于一类。 对测试图像中，同一类物体出现的新的不常见的长宽比和其他情况是。泛化能力偏弱。 由于损失函数的问题，定位误差是影响检测效果的主要原因。尤其是大小物体的处理上，还有待加强**SSD（单发多框检测器）**• SSD， 全称Single Shot MultiBox Detector， 是Wei Liu在ECCV 2016上提出的一种多尺度的目标检测算法。• 相比Faster-RCNN有明显的速度优势， 相比YOLO又有明显的mAP优势。• SSD具有如下主要特点：1. 从YOLO中继承了将detection转化为regression的思路，同时一次即可完成网络训练2. 基于Faster-RCNN中的anchor，提出了相似的priorbox；3. 加入基于特征金字塔（Pyramidal FeatureHierarchy）的检测方式，相当于半个FPN思路总结： • **SSD算法的优点**很明显：运行速度可以和YOLO媲美，检测精度可以和Faster RCNN媲美。• **缺点**：• 需要人工设置prior box的min\_size， max\_size和aspect\_ratio值。网络中prior box的基础大小和形状不能直接通过学习获得，而是需要手工设置。而网络中每一层feature使用的prior box大小和形状恰好都不一样，导致调试过程非常依赖经验。• 虽然采用了pyramdial feature hierarchy的思路，但是对小目标的recall依然一般，并没有达到碾压Faster

RCNN的级别。**损失函数**（目标函数） ——在训练过程中需要将其最小化。它能够衡量当前任务是否已成功完成。– 如何选择正确的损失函数？• 对于二分类问题，你可以使用二元交叉熵（binary crossentropy）损失函数；• 对于多分类问题，可以用分类交叉熵（categorical crossentropy）损失函数；• 对于回归问题，可以用均方误差（mean-squared error）损失函数；• 对于序列学习问题，可以用联结主义时序分类（CTC，connectionist temporal classification） 损失函数。



