NAS-Net:
Neural Architecture Search（NAS），即神经网络结构搜索技术，通过某种结构及算法，实现神经网络结构的自动生成，它主要包含搜索空间，以及搜索策略、性能评估策略等几个维度的知识。搜索空间可以简单理解为基于一定的前提和假设，并根据已有的经验，预置一些网络结构单元，就像堆积木一样，预先提供了各种各样的积木，最终的网络结构就是通过搜索空间中的这些原始积木组合成的。
也正是因为如此，通过这种模式生成的最终网络结构其实只是在给定的搜索空间中查找效果最优的模型结构而已，机器只是沿着人类设计好的算法，依据某些评估指标，通过不断的测试从而生成一个比较完美的网络结构。
绝大部分机器学习都不是人工智能，计算机并不是真的具有智能了，也不会无缘无故获得既定目标以外的能力。
NAS 算法所生成的网络结构是基于这样的一个前提：它认为一个大的神经网络结构是由很多小的、重复的单元所组成的。我们在构建整个神经网络时，只需要针对这些小的单元进行搜索，而不是每次针对整个网络结构进行搜索。

搜索策略（Search strategy）：搜索策略定义的则怎样去搜索。一方面，我们希望能快速找到性能良好的架构，另一方面，也应避免过早收敛到次优架构（suboptimal architeture）区域。已经有许多不同的搜索策略用于 NAS，主要有如下这些： 随机搜索（random search），贝叶斯优化（Bayesian optimazation），进化方法（evolutionaray methods），强化学习（Reinforcement Learning, RL），梯度方法（gradient-based methods）
性能评估策略（Performace estimation strategy）：NAS 的目标是希望能够自动的在给定的数据集上找到一个高性能的架构。性能评估则是指评估此性能的过程：最简单的方式是按照通常的方式对一个标准架构训练和验证来获得结果，但遗憾的是这样的计算成本太高了，并且同时限制了可以搜索的网络架构的数量。因此，最近的许多研究都集中在探索新的方法来降低这些性能评估的成本（比如 ENAS、DARTS）。

进化算法：

一般的进化算法其实大同小异，差别在如何选择变异，有比较细的变异，比如在 Large-Scale Evolution of Image Classifiers 这篇文章中，就定义了非常具体的变异，比如有改变通道数量，改变 filter 大小，改变 stride 等等；而在 Simple And Efficient Architecture Search for Convolutional Neural Networks 这篇论文中，它的变异，就借鉴了现有公认的比较好的结构，加深网络就用 conv-bn-relu3 件套，加宽网络加大通道数量，增加 skip connection。

这些进化算法在做自动模型选择时，每次迭代都不可避免的需要在整个数据集上跑若干个 epoch，而每次迭代都有许多个变异，又需要很多次迭代，导致最后的训练时间太久。

Simple And Efficient Architecture Search for Convolutional Neural Networks 这篇论文提出，我们先用一个成熟的模型去训练(也可以 fine-tune 训练)，然后在这个模型的基础上去变异，变异之后用 fine-tune 训练几个 epoch 即可。这带来两个好的结果：fine tune 减少了大量的训练时间

EAT-NAS: architectures are first searched on a small dataset, the best one is chosen as the basic architecture. Then the whole architecture is transferred with elasticity