第 42 卷 第 5 期 自 动 化 学 报 Vol. 42, No. 5
2016 年 5 月 ACTA AUTOMATICA SINICA May, 2016

阿法狗围棋系统的简要分析

田渊栋 [1]

摘 要 谷歌的围棋系统阿法狗 (AlphaGo) 在三月的比赛中以 4:1 的成绩击败了围棋世界冠军李世石, 大大超过了许多人对计算机围棋程序何时能赶上人类职业高手的预期 (约 10～30 年). 本文在技术层面分析了阿法狗系统的组成部分, 并基于它过去的公开对局预测了它可能的弱点.

关键词 深度学习, 深度卷积神经网络, 计算机围棋, 强化学习, 阿法狗

引用格式 田渊栋. 阿法狗围棋系统的简要分析. 自动化学报, 2016, **42**(5): 671−675

DOI 10.16383/j.aas.2016.y000001

A Simple Analysis of AlphaGo

TIAN Yuan-Dong[1]

Abstract In March 2016, the AlphaGo system from Google DeepMind beat the World Go Champion Lee Sedol 4:1 in a historic five-game match. This is a giant leap filling the gap between Go AI and top human professional players, which was once regarded to be filled in at least 10～30 years. In this paper, based on published results [Silver et al., 2016], i analyze the components of AlphaGo and predict its potential technical weakness based on the public games of AlphaGo.

Key words Deep learning, deep convolutional neural network, computer Go, reinforcement learning, AlphaGo

Citation Tian Yuan-Dong. A simple analysis of AlphaGo. *Acta Automatica Sinica*, 2016, **42**(5): 671−675

AlphaGo 这个系统[1] 主要由几个部分组成:

1) 走棋网络 (Policy network), 给定当前局面, 预测/采样下一步的走棋.

2) 快速走子 (Fast rollout), 目标和走棋网络一样, 但在适当牺牲走棋质量的条件下, 速度要比走棋网络快 1 000 倍.

3) 估值网络 (Value network), 给定当前局面, 估计是白胜还是黑胜.

4) 蒙特卡罗树搜索 (Monte Carlo tree search, MCTS), 把以上这三个部分连起来, 形成一个完整的系统.

我们的 DarkForest[2] 和 AlphaGo 同样是用蒙特卡罗树搜索搭建的系统. DarkForest 较 AlphaGo 而言, 在训练时加强了走棋网络, 而少了快速走子和估值网络, 12 月时以开源软件 Pachi 的缺省策略 (Default policy) 部分替代了快速走子的功能, 2 个月后部分实现了 AlphaGo 快速走子的能力.

以下详细介绍各部分.

1 走棋网络

走棋网络把当前局面作为输入, 预测/采样下一步的走棋. 它的预测不只给出最强的一手, 而是对棋盘上所有可能的下一着给一个分数. 棋盘上有 361 个点, 它就给出 361 个数, 好招的分数比坏招要高. DarkForest 在这部分有创新, 通过在训练时预测三步而非一步, 提高了策略输出的质量, 和他们在使用增强学习进行自我对局后得到的走棋网络 (Reinforced network, RL network) 的效果相当. 当然, 他们并没有在最后的系统中使用增强学习后的网络, 而是用了直接通过训练学习到的网络 (Supervised network, SL network), 理由是 RL network 输出的走棋缺乏变化, 对搜索不利.

有意思的是在 AlphaGo 为了速度上的考虑, 只用了宽度为 192 的网络, 而并没有使用最好的宽度为 384 的网络 (见图 1, 即文献 [1] 中 Figure 2 的左图), 所以要是图形处理器 (Graphics processing unit, GPU) 更快一点 (或者更多一点), AlphaGo 肯定是会变得更强的.

所谓的 0.1 秒走一步, 就是纯粹用这样的网络, 下出有最高置信度的合法着法. 这种做法完全不搜索, 大局观非常强, 不会陷入局部战斗中, 说它建模了 "棋感" 一点也没有错. 从去年八月开始我们秉持开放的目的, 第一个把基于深度学习的走棋网络直接放上 KGS Go Server 给大家试下并且达到了 3d 的水平, 当时引起了挺大的轰动, 并且在今年 1 月的 KGS 锦标赛上差点拿了冠军. 受此影响, 今年 3 月份在日本举行的 UEC 杯 (日本电气通信大学杯) 进

收稿日期 2016-04-14 录用日期 2016-05-10
Manuscript received April 14, 2016; accepted May 10, 2016
本文责任编委 周志华
Recommended by Associate Editor ZHOU Zhi-Hua
1. 脸书人工智能研究所 加利福尼亚州 94025 美国
1. Facebook AI Research (FAIR) Facebook Inc., CA 94025, USA

决赛的前 8 支队里已经有 6 支用上了深度卷积神经网络 (Deep convolutional neural network, DCNN), 特别是传统强队 Zen 和 CrazyStone 都已经用上了. 可以说, 这一波围棋人工智能 (Artificial intelligence, AI) 的突破, 主要得益于走棋网络的突破. 这个在以前是不可想像的, 以前用的是基于规则, 或者基于局部形状再加上简单线性分类器训练的走子生成法, 需要慢慢调参数数年, 才有进步.

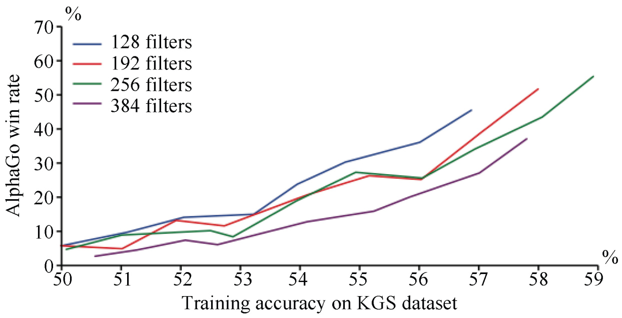


图 1 AlphaGo 的分析 1

Fig. 1 Analysis 1 of AlphaGo

当然, 只用走棋网络问题也很多, 就我们在 DarkForest 上看到的来说, 会不顾大小无谓争劫、会无谓脱先、不顾局部死活、对杀出错等. 特别走棋网络对提劫特别喜爱, 有提劫时提劫往往会排第一, 这样就常给对方通过小提劫救活一块大龙的机会. 总得来说, 它有点像高手不经任何思考的随手棋, 原因自然是走棋网络没有价值判断, 只是凭 "直觉" 在下棋, 只有在加入搜索之后, 电脑才有价值判断的能力.

2 快速走子

那有了走棋网络, 为什么还要做快速走子呢? 有两个原因, 首先走棋网络的运行速度是比较慢的, AlphaGo 说是 3 毫秒, 我们这里也差不多, 而快速走子能做到几微秒级别, 差了 1 000 倍. 所以在走棋网络没有返回的时候让 CPU 不闲着先搜索起来是很重要的, 等到网络返回更好的着法后, 再更新对应的着法信息. 这是第一种用法.

其次, 快速走子可以用来评估盘面. 由于天文数字般的可能局面数, 围棋的搜索是毫无希望走到底的, 搜索到一定程度就要对现有局面做个估分. 在没有估值网络的时候, 不像国际象棋可以通过算棋子价值之和来对盘面做简单但是相对准确的估值, 围棋盘面的估计要通过模拟走子来进行, 从当前盘面一路走到底, 不考虑岔路地算出胜负, 然后把胜负值作为当前盘面价值的一个估计. 这里有个需要权衡的地方: 在同等时间下, 模拟走子的质量高, 单次估值精度高但走子速度慢; 模拟走子速度快乃至使用随机走子, 虽然单次估值精度低, 但可以多模拟几次算平均值, 效果未必不好. 所以说, 如果有一个质量高又速度快的走子策略, 那对于棋力的提高是非常有帮助的.

为了达到这个目标, 神经网络的模型就显得太慢, 还是要用传统的局部特征匹配 (Local pattern matching) 加线性回归 (Logistic regression) 的方法, 这办法虽然不新但非常好使, 几乎所有的广告推荐、竞价排名、新闻排序、都是用这种方法. 与更为传统的基于规则的方案相比, 它在吸纳了众多高手对局之后就具备了用梯度下降法自动调参的能力, 所以性能提高起来会更快更省心. AlphaGo 用这个办法达到了 2 微秒的走子速度和 24.2 % 的走子准确率. 24.2 % 的意思是说它的最好预测和围棋高手的下子有 0.242 的概率是重合的, 相比之下, 走棋网络在 GPU 上用 2 毫秒能达到 57 % 的准确率. 这里,

表 1 阿法狗在快速走子中使用的盘面特征

Table 1 Input features for rollout and tree policy

Feature	# of patterns	Description
Response	1	Whether move matches one or more response pattern features
Save atari	1	Move saves stone(s) from capture
Neighbour	8	Move is 8-connected to previous move
Nakade	8 192	Move matches a nakade pattern at captured stone
Response pattern	32 207	Move matches 12-point diamond pattern near previous move
Non-response pattern	69 338	Move matches 3×3 pattern around move
Self-atari	1	Move allows stones to be captured
Last move distance	34	Manhattan distance to previous two moves
Non-response pattern	32 207	Move matches 12-point diamond pattern centred around move

(Features used by the rollout pollcy (the first set) and tree policy (the first and second sets). Patterns are based on stone colour (black/white/empty) and liberties (1, 2, $\geq$ 3) at each intersection of the pattern.)

我们就看到了走子速度和精度的权衡.

和训练深度学习模型不同, 快速走子用到了局部特征匹配, 自然需要一些围棋的领域知识来选择局部特征. 对此 AlphaGo 只提供了局部特征的数目 (见表 1, 即文献 [1] 中 Extended Table 4), 而没有说明特征的具体细节. 今年二三月份我也实验了他们的办法 (感谢复旦大学王琳同学提供 Tygem 数据集), 达到了 30% 左右的准确率和 6～7 微秒的走子速度. 虽然 Top-1 高了几个百分点, 但全系统整合下来并没有复现他们的水平. 我认为 24.2% 并不能完全概括他们快速走子的水准, 因为只要走错关键的一步, 局面判断就完全错误了; 而图 2 (即文献 [1] 中 Figure 2 的右图) 直接衡量在对局的不同时期, 快速走子对局面判断的精确性, 更能体现 AlphaGo 的形势估计能力.

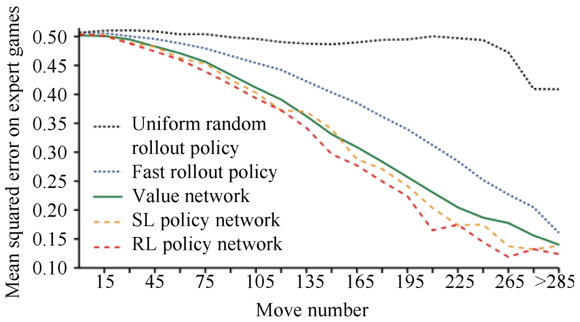


图 2　AlphaGo 的分析 2

Fig. 2　Analysis 2 of AlphaGo

同样是预测下一步, 走子网络和快速走子的要求是截然不同的. 快速走子的首要目的是稳定判分, 保证每块棋的死活大体正确, 不要把死的棋下成活的或者反之, 而对大局观的要求并不高. 理论上双方完全可以配合着把每块棋下清楚, 然后转战另一块, 而不是说抢着去别处占先手. 因此, 快速走子的走棋质量变好未必对应于更精准的盘面估计. 但走子网络对于脱先和抢先手的感觉需要非常灵敏, 不然下出来的棋只会在意于局部的纠缠, 而失去了对全局的把控力. 近年来计算机围棋的大进展, 正是因为卷积神经网络能从大量数据中抽取全局感.

在图 2 中, DarkForest 目前处于 280 步后约 26% 的错误率的水平, 与 AlphaGo 快速走子的曲线相比还有一定差距. 据我们自己的分析, 及三月份在日本参加 UEC 杯时与 Zen 及 CrazyStone 作者的讨论, 在处理一些复杂对杀及局部死活的情况时, 仍然需要在快速走子时添加各种规则, 以保证在关键时下出百分百正确的招法, 而只用回归模型的话, 给出的预测结果往往是依从概率的. 两大软件的作者都承认用了数年时间去积累规则, 并且也没有太好的办法去从大量对局样本中自动习得规则. 对此 AlphaGo 并没有公开细节, 所以也无从知晓他们的具体做法.

在 AlphaGo 有了快速走子之后, 不需要走棋网络和估值网络, 不借助任何深度学习和 GPU 的帮助, 不使用增强学习, 在单机上就已经达到了 3d 的水平 (见表 2 (即文献 [1] 中 Extended Table 7) 倒数第二行), 这是相当厉害的了. 任何使用传统方法在单机上达到这个水平的围棋程序, 都需要花费数年的时间. 在 AlphaGo 之前, Huang 是 CrazyStone 作者 Remi 的学生, 又写过两个非常不错的围棋程序, 在这方面经验丰富.

3　估值网络

AlphaGo 的估值网络可以说是锦上添花的部分, 有了它, 去年十月的单机版 AlphaGo 才达到了职业水平, 不然则是 7d～8d 的顶级业余水平. 考虑到估值网络是整个系统中最难训练的部分 (需要三千万局自我对局), 它应当是最晚做出来并且最有可能进一步提高的. 时隔半年后, 在对阵李世石时 AlphaGo 取得的三子半的长足进步, 我相信主要是估值网络的进一步增强, 对估值网络判断的进一步信赖, 及自我对局的某种创新性用法.

说实话, 估值网络如此成功是比较让人吃惊的. 传统的围棋程序往往要做适当的局部死活及对杀分析, 而他们则纯粹用基于大量数据的暴力训练法得到了一个相当不错的估值网络. 这在一定程度上说明深度卷积网络 (Deep convolutional neural networks, DCNN) 有自动将问题分解成子问题, 并分别解决的能力. 围棋虽说总变化多过宇宙中的原子数, 但很多局部仍然一直重复出现, 人类在学棋时正是不自觉地利用了这一点以达到举一反三的效果. 现在深度卷积网络的出现让计算机也获得了相似能力. 到目前为止, 对于如何解释这个能力, 学术界还没有一致的结论.

从图 2 和表 2 来看, 少了估值网络, 等级分少了 480 分, 但是少了走棋网络, 等级分就会少掉 800 至 1 000 分. 特别有意思的是, 如果只用估值网络来评估局面 (2 177), 其效果还不及只用快速走子 (2 416), 只有将两者合起来才有更大的提高. 我认为, 估值网络和快速走子对盘面估计是互补的, 在棋局一开始时, 大家下得比较和气, 估值网络会比较重要; 但在有复杂的死活或是对杀时, 通过快速走子来估计盘面就变得更重要了.

估值网络的训练是比较麻烦的, 因为这个网络的输出只有一个标量, 即当前盘面下某一方的胜率, 所以训练时回传的梯度信号非常微弱, 并且极易过拟合 (也就是把看过的局面背出来). 以我们的经验, 直接将输入和输出送进网络里面训练几乎不能收

敛, 需要用走棋网络去初始化前几层的权值才可以. DarkForest 之所以采用预测三步而非一步的方案, 正是因为这样做的梯度信号更丰富, 训练更容易一些.

关于估值网络训练数据的生成, 值得注意的是文献 [1] 中 Value network: regression 小节中的内容. 与走棋网络不同, 每一盘棋只取一个样本来训练以避免过拟合, 不然对同一对局而言输入稍有不同 (盘面差一两步) 而输出都相同 (都是黑胜), 对训练是非常不利的. 这就是为什么需要三千万局, 而非三千万个盘面的原因. 对于每局自我对局, 取样本是很有讲究的, 先用 SL network 保证走棋的多样性, 然后随机走子, 取盘面, 然后用更精确的 RL network 走到底以得到最正确的胜负估计.

另外, 我猜测他们在取训练样本时, 判定最终胜负用的是中国规则. 所以说 3 月与李世石对局的时候也要求用中国规则, 不然如果换成别的规则, 就需要重新训练估值网络. 至于为什么一开始就用的中国规则, 据我在 DarkForest 上的经验, 编程方便是很重要的因素. 相比之下, 日本规则有一些例外情况需要特殊处理, 而数目方便的特征在能用计算机点目的条件下则不是很有意义.

4 蒙特卡罗树搜索

这部分基本用的是传统方法, 没有太多可以评论的, 他们用的是带先验的 Upper confidence bound 1 applied to trees (UCT) (Prior UCT, PUCT), 即先考虑 DCNN 认为比较好的着法, 然后等到每个着法探索次数多了, 选择更相信探索得来的胜率高的下法. 而 DarkForest 是直接用了 DCNN 推荐的前三或是前五的着法进行搜索. 与之相比, AlphaGo 的办法更灵活, 特别对一些 DCNN 认为不好但却对局面至关重要的着法, 在允许使用大量搜索次数的情况下, 用 PUCT 可以探索到.

一个有趣的地方是在每次搜索到叶子节点时, 没有立即展开叶子节点, 而是等到访问次数到达一定数目 (40) 才展开, 这样避免产生太多的分支, 分散搜索的注意力, 也能节省 GPU 的宝贵资源, 同时在展开时, 对叶节点的盘面估值会更准确些. CrazyStone 的作者也确认他用了这个办法.

除此之外, 他们也用了一些技巧, 以在搜索一开始时, 避免多个线程同时搜索一路变化, 这部分我们在 DarkForest 做了有所不同的改进.

总的来说, 与 AlphaGo 相比, DarkForest 还有挺大的差距. 今年三月份在日本举办的 UEC 杯上 DarkForest 输给了 Zen 拿了亚军, 与小林光一还有三子的距离, 而 AlphaGo 展现出来的水平已明显超过了所有的围棋职业选手. 当然从两方投入上看, AlphaGo 是 DeepMind 大团队倾力打造的明星项目, 而 DarkForest 只是研究员个人发起的探索性研究项目, 两者的目的是不一样的. 研究员本人之前也没有相关经验, 而 AlphaGo 的二十人作者列表里至少有四至五人在计算机围棋上有数年经验. 最后, DarkForest 只进行了八至九个月, 而集中力量攻关的时间则更少至三到四个月, 相比之下, AlphaGo 团队从组建到现在已经历时 18 个月.

4.1 可能的弱点

赛后 AlphaGo 团队自己说与李世石对战的版本其等级分已经达到了 4 500, 远高于李世石的 350 水准. 如果 4 500 的等级分是准确的话, 按照等级分与胜率的换算关系, 胜率会在是 99 % 至 100 % 这个区间, 这样 AlphaGo 与李世石的对战就应是 5 比 0. 然而在实际比赛中李胜了一局, 可能的原因有以下几个.

表 2 不同版本阿法狗的等级分比较 (等级分由一场内部锦标赛决出)

Table 2 Results of a tournament between different variants of AlphaGo

Short name	Policy network	Value network	Rollouts	Mixing constant	Policy GPUs	Value GPUs	Elo rating
α_{rvp}	p_σ	v_θ	p_π	$\lambda = 0.5$	2	6	2 890
α_{vp}	p_σ	v_θ	–	$\lambda = 0$	2	6	2 177
α_{rp}	p_σ	–	p_π	$\lambda = 1$	8	0	2 416
α_{rv}	$[p_\tau]$	v_θ	p_π	$\lambda = 0.5$	0	8	2 077
α_v	$[p_\tau]$	v_θ	–	$\lambda = 0$	0	8	1 655
α_r	$[p_\tau]$	–	p_π	$\lambda = 1$	0	0	1 457
α_p	p_σ	–	–	–	0	0	1 517

Evaluating positions using rollouts only (α_{rp}, α_r), value nets only (α_{vp}, α_v), or mixing both ($\alpha_{rvp}, \alpha_{rv}$); either using the policy network $p_\sigma(\alpha_{rvp}, \alpha_{vp}, \alpha_{rp})$ or no policy network ($\alpha_{rvp}, \alpha_{vp}, \alpha_{rp}$), that is, instead using the placeholder probabilities from the tree policy p_τ throughout. Each program used 5 s per move on a single machine with 48 CPUs and 8 GPUs. Elo ratings were computed by BayesElo.

首先这个等级分完全是由不同版本的 AlphaGo 内战决定的, 因此有可能每代 AlphaGo 都有共同的缺点而未能发现, 在这个层次上的对局, 人已经看不出弱手, 去研究大量的内部对局也需要棋手的大量精力. 其次是 AlphaGo 的估值网络可能存在问题, 从它的训练样本的生成上来看, 所有训练样本都是从自我对局中得到的, 而 DCNN 虽然大局观非常好, 但局部死活和对杀经常犯低级错误, 所以两个 DCNN 对下会有死活问题, 最后可能会错进错出. 以这样的样本代入估值网络进行训练, 训练出的网络同样会死活不分. 另外, 从文献 [1] 得知, 不管快速走子那边模拟了多少步, AlphaGo 始终给估值网络以 0.5 的权重 (据未经证实的说法, 在三月新版的 AlphaGo 里面给了估值网络更高的权重), 这样如果估值网络出了问题, 快速走子是无法弥补的. 在第四局中, AlphaGo 在李世石的 78 手挖后的十余步里仍然保持着己方大胜的估计, 可能就是估值网络对接下来的局面判断一直有错, 直到黑右边的大龙被屠为止. 另外, 李世石下的 78 手挖, 在 DarkForest 的走子网络中排第 31 位, 是非常少见的下法. 这个下法并非错招, 计算机需要正确应对才能保持优势. 可能 AlphaGo 在之前的搜索时, 因为用的是 PUCT, 会按走子网络的置信度来优先展开下一步, 因此对这一步关注较少, 在同样一分钟的思考时间里搜索的深度不够导致出错.

5 总结

总的来说, 这整篇文章是一个系统性的工作, 而不是一两个小点有了突破就能达到的胜利. 在成功背后, 是作者们, 特别是两位第一作者 Silver 和 Huang, 在博士阶段及毕业以后五年以上的积累, 非一朝一夕所能完成的. 他们能做出 AlphaGo 并享有现在的荣誉, 是实至名归的.

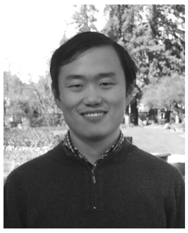
从以上分析也可以看出, 与之前的围棋系统相比, AlphaGo 较少依赖围棋的领域知识, 但还远未达到通用系统的程度. 职业棋手可以在看过了寥寥几局之后明白对手的风格并采取相应策略, 一位资深游戏玩家也可以在玩一个新游戏几次后很快上手, 但到目前为止, 人工智能系统要达到人类水平, 还是需要大量样本的训练的. 可以说, 没有千年来众多棋手在围棋上的总结和对局积累, 就没有围棋人工智能的今天. 基于 DCNN 的围棋系统, 从一开始就需要大量的高水平对局以建立走子网络, 在此基础上才能训练出估值网络来. 若要将之应用于更大的棋盘 (如 21×21), 则同样需要大量样本重新训练; 而职业棋手可能在大棋盘上下个几十盘之后, 就会对边角中腹等有大致的概念, 而快速达到高水平. 从这一点上可以看到人类和人工智能的巨大差别. DeepMind 说他们下一步会试验完全摆脱大量已有的高水平样本, 从零开始训练围棋程序, 鉴于围棋的难度, 这将是个非常艰巨的任务.

在 AlphaGo 中, 增强学习 (Reinforcement learning) 所扮演的角色并没有想像中那么大. 在理想情况下, 我们希望人工智能系统能在对局中动态地适应环境和对手的招式并且找到办法反制之, 但是在 AlphaGo 中增强学习更多得是用于提供更多质量更好的样本, 给有监督学习 (Supervised learning) 以训练出更好的模型. 在这方面增强学习还有很长的路要走.

References

1 Silver D, Huang A, Maddison C J, Guez A, Sifre L, van den Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, Dieleman S, Grewe D, Nham J, Kalchbrenner N, Sutskever I, Lillicrap T, Leach M, Kavukcuoglu K, Graepel T, Hassabis D. Mastering the game of go with deep neural networks and tree search. *Nature*, 2016, **529**(7587): 484−489

2 Tian Y D, Zhu Y. Better computer go player with neural network and long-term prediction. In: International Conference on Learning Representation (ICLR). San Juan, Puerto Rico, 2016.



田渊栋 脸书人工智能研究所研究员. 主要研究方向为深度学习及计算机视觉. 2013 至 2014 年曾任谷歌无人车组研究员/软件工程师. 2008 年毕业于上海交通大学获硕士学位, 2013 年于美国卡耐基梅隆大学机器人系获博士学位, 曾获 2013 年国际计算机视觉会议 (ICCV) 马尔奖提名.
E-mail: yuandong@fb.com
(**TIAN Yuan-Dong** Research scientist in Facebook AI Research, working on deep learning and computer vision. Prior to that, he was a researcher/software engineer in Google Self-driving Car Team in 2013～2014. He received Ph. D. in Robotics Institute, Carnegie Mellon University in 2013, Bachelor and Master degrees in computer science in Shanghai Jiao Tong University. He is the recipient of 2013 ICCV Marr Prize Honorable Mentions.)