SPRi Issue Report
2016. 2. 제2016-001호 ver. 0.9

AlphaGo의 인공지능

- 구글의 바둑인공지능 AlphaGo, 인간 챔피온을 꺾다.

김석원 책임연구원†
안성원 선임연구원
추형석 선임연구원

SPRi 소프트웨어정책연구소

○ 본 보고서는 「미래창조과학부 정보통신진흥기금」을 지원받아 제작한 것으로 미래창조과학부의 공식의견과 다를 수 있습니다.

○ 본 보고서의 내용은 연구진의 개인 견해이며, 본 보고서와 관련한 의문사항 또는 수정・보완할 필요가 있는 경우에는 아래 연락처로 연락해 주시기 바랍니다.

- 소프트웨어정책연구소 SW융합연구실 김석원 책임연구원(skimaza@spri.kr)

《 Executive Summary 》

바둑은 고전 게임 중 탐색 범위가 가장 넓고 보드의 상황을 평가하는 것이 어려워서 인공지능의 가장 큰 숙제 중 하나였다. 체스와 퀴즈게임에서 세계챔피언을 이기는 현재도 당분간 바둑만큼은 사람이 인공지능보다 나을 것으로 예상되어 왔다. 그런데 이 예상도 의외로 빨리 깨지게 될지 모르겠다. 구글이 개발한 AlphaGo라는 프로그램이 유럽챔피언을 5대0으로 완파한 것이다. 이 유럽챔피언은 중국에서 자라면서 바둑을 배운 판후이 프로2단으로 실질적으로는 세계 최강 중국의 바둑 2단을 이긴 것으로 볼 수 있다. 이 프로그램이 이번엔 진짜 세계 최고 수준의 바둑에 도전하기 위해 우리나라의 이세돌 9단과 오는 3월에 서울에서 대결을 펼치기로 했다. 과연 인공지능이 바둑에서도 세계챔피언을 꺾을 수 있을까 궁금해 하기 전에 구글의 AlphaGo가 어떤 것인지 살펴보자.

인공지능으로 바둑 게임을 구현하려면 어떤 경기 상황에서 다음에 둘 수에 대한 선택 확률과 바둑의 수읽기와 마찬가지로 향후 여러 수가 진행되었을 때 형세가 어떨지 예상해 볼 수 있는 지능적 게임 시뮬레이션 기능이 필요하다. 이번에 발표된 구글의 AlphaGo에서는 딥러닝과 강화학습(Reinforcement Learning), 몬테카를로 트리 탐색(Monte Carlo Tree Search) 등 인공지능과 게임이론의 최신 기술을 적극 활용하고 구글의 거대한 계산 자원을 활용하여 프로기사를 이길 수 있는 수준으로 지능을 끌어올렸다.

바둑의 상태를 보고 모든 빈 칸에 대해 성공 가능성을 계산하는 다음 수 선택 확률은 다시 두 가지로 나눠서 구현했다. 첫째는 딥러닝 기법 중 컨벌루션신경망을 적용하여 과거 기보를 지도학습하는 단계이다. 컨벌루션신경망은 페이스북에서 얼굴인식에 사용한 것으로 유명해진 딥러닝 기술이며, 입력 이미지를 작은 구역으로 나누어 부분적인 특징을 인식하고 신경망 단계가 깊어지면서 이것이 결합하여 전체를 인식하는 특성을 가진다. 바둑에서도 사활문제같은 국지적 패턴이 중요하고 이런 부분적 패턴이 전반적인 형세와 점진적으로 연관되기 때문에 컨벌루션신경망을 이용하는 것은 적절한 선택이다. AlphaGo에서는 입력으로 19x19 크기의 바둑판 상황이 들어가고 출력도 19x19 각 바둑판 위치의 선택 확률 분포가 나오는 13단계의 신경망을 구성하고 KGS Go Server에 있는 3천만가지 바둑판 상태를 학습했다. 페이스북이 9단계 신경망으로 얼굴인식을 구현한 것에 비교하면 이것은 막대한 계산량이 필요한 일이며 생각은 있어도 실제 시도할 수 있는 곳이 세계적으로 몇 안 될 것이다.
두번째 단계로 지도학습한 신경망끼리 게임을 하고 이긴 쪽으로 가중치를 조정하는 강

화학습을 적용했다. 첫 단계가 기보를 배워서 기보를 둔 사람 수준의 바둑을 목표로 한다면 이 단계에서는 기보를 넘어서는 성능을 쌓기 위한 개인 훈련으로 볼 수 있다. 실제로 이 단계를 통해 성능이 많이 개선되었고 첫 단계만 학습한 신경망과의 대결에서 80%의 승률을 보인다고 한다. 두번째 단계도 입력과 출력은 19x19 바둑판 상황과 선택 확률 분포이다. 이 두 가지 신경망을 정책망(policy network)라고 부르며 첫 단계 결과는 SL정책망(Supervised Learning Policy Network), 둘째 단계의 결과는 RL정책망(Reinforcement Learning Policy Network)이라고 부른다.

정책망과 더불어 바둑 상황에 대해 승리할 기대값을 예측하는 신경망(v: value network)도 학습지도와 유사한 컨벌루션신경망으로 학습한다. 앞서 설명한 1단계 정책망 학습과 다른 점은 출력이 확률분포가 아닌 특정 값이 나온다는 점이다. 이 값이 바둑용어로 형세판단의 결과라고 볼 수 있을 것이다.

강화학습에서 신경망끼리 게임을 하려면 형세판단을 하고 다음 수를 결정하는 지능적 게임 실행 기능이 필요하며 이것은 실제 바둑 경기를 할 때도 필요하다. AlphaGo에서는 2000년대 중반에 발표되어 인기를 끌고 있는 몬테카를로 트리 탐색 방법을 정책망과 결합하여 활용한다. AlphaGo의 몬테카를로 트리 탐색은 다음 수를 찾기 위해 현 상태에서 나와 상대가 모두 동일한 정책망을 가진 것으로 가정하고 여러 번 시뮬레이션을 돌려서 가장 높은 빈도로 선택한 수를 택하는 방식이다. 이 방법은 모든 트리를 탐색하지 않아도 좋은 성능의 장비에 의해 충분한 시뮬레이션을 돌릴 수 있으면 최적에 가까운 결과를 내는 것으로 알려져 있다. 구글은 충분한 시뮬레이션을 위해 병렬처리를 지원하고 많은 CPU와 GPU를 할당하여 계산한다. 또한 시뮬레이션을 게임 종료까지 실행하는 대신 적당한 깊이까지 탐색하고 그 이후는 앞서 계산한 value network과 보다 단순화시킨 시뮬레이션을 종합한 결과로 대치하여 신속한 계산을 하도록 구현했다. 구글은 실험에 의해 이 단순화가 승리 확률에 영향을 미치지 않는다는 것을 확인했다고 한다.

이 연구의 가장 큰 기여는 인공지능 딥러닝 기술을 활용해서 기존 게임 탐색 방법의 품질을 획기적으로 개선할 수 있다는 점을 보인 것이다. 그리고 그 결과를 제시하는 방법으로 현실의 강력한 목표를 찾아서 실증하여 일반인에게 강한 인상을 주게 된 것이다. 국내 연구 현실을 생각한다면 연구자가 자신의 아이디어를 실험해 볼 수 있는 대용량 컴퓨팅 환경을 제공하는 것이 필요하고, 연구자도 현실적 문제에 도전하여 해결하고 실증하는 사례가 늘어나야 할 것으로 생각한다.

이세돌 9단은 종종 파격적인 전술로 상대방을 공략하는 것을 좋아하는 바둑천재다. 일반적인 전략으로 AlphaGo를 상대한다면 인공지능 기술이 얼마나 발전했는지 확인하는 것이 흥미로울 것이다. 그러나 상대가 컴퓨터 프로그램임을 고려한다면 그동안 다른 사람이 잘 사용하지 않았던 전략과 전술로 상대하는 것이 승산을 높이는 것이 아닐까 예상된다. 판후이 선수를 완파한 것으로 보아 AlphaGo의 학습범위나 게임 성능은 상당

한 수준일 것이며 기존 기보의 학습이나 심지어 이세돌 9단의 과거 기보도 학습이 되어 있을 것 같다. 이세돌 9단의 특기인 파격적 발상을 최대한 활용하여 부담없이 경기에 임하면 게임도 재미있고 과연 AlphaGo가 새로운 패턴에 어떻게 대응하는지 배울 수 있는 좋은 기회가 될 것 같다.

《 목 차 》

1. AlphaGo 인공지능 바둑 프로그램의 등장

□ 바둑은 예전부터 인공지능 분야의 도전과제

○ **(복잡성)** 탐색 범위가 가장 넓고 판의 상황을 평가하는 것이 어려웠음 (인공지능 < 인간)

○ **(구글 AlphaGo)** 구글 DeepMind팀이 개발한 AlphaGo의 등장

- 유럽 바둑챔피온 판후이(Fan Hui) 프로2단을 5:0으로 완파

  * 공식경기 5회, 비공식 경기 5회 대결. 비공식 경기에서는 3:2로 AlphaGo 승리

  * 중국식 규칙, 덤 7.5집, 제한시간 1시간, 초읽기 30초 (공식), 비공식 경기에서는 초읽기만 30초의 초속기 방식

- 우리나라의 이세돌 9단과 3월에 서울에서 대결 예정

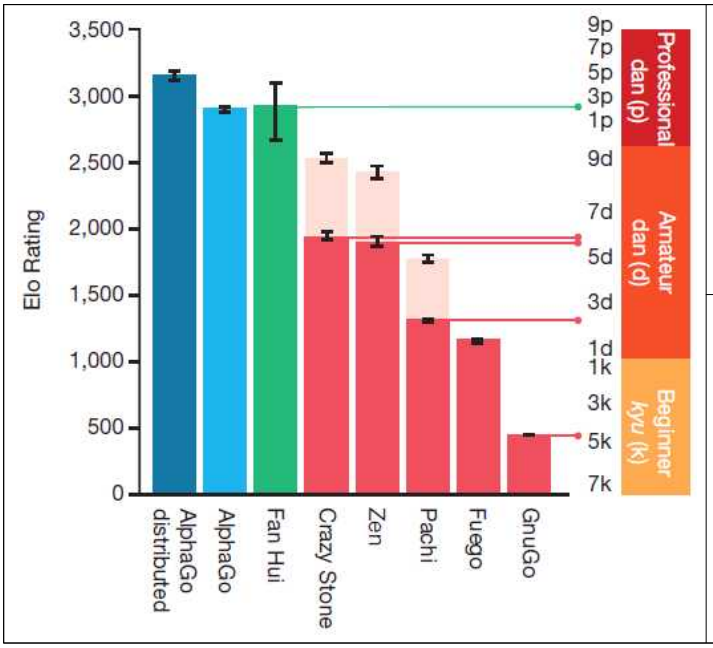
  * 우승 상금 100만달러는 구글이 제공

2. AlphaGo의 성능과 바둑게임 프로그램 비교

□ AlphaGo의 성능

○ AlphaGo는 KGS 기준 프로 2~5단 수준

- 기존의 바둑게임 프로그램과의 성능 비교



<AlphaGo의 HW spec.>
* 최종버전(싱글)은 40개의 탐색 쓰레드, 48개 CPU, 8개 GPU를 사용
* 분산 구현 버전은 40개의 탐색 쓰레드, 1202개 CPU, 176개 GPU를 사용.

- 그림의 Elo Rating 은 체스등과 같은 보드게임에서 플레이어들의 실력을 나타내는 수치
- 상용 바둑 프로그램 성능은 GnuGo 5급, Fuego 아마1단, Panchi 아마2단, Zen 아마 6단, Crazy Stone 아마 6단 수준

[그림 1] AlphaGo 와 상용 바둑 프로그램 성능 비교

○ 기존의 바둑 게임 프로그램

명칭	개발자	출시 년도	최신 버전	사용알고리즘	전적	수준
Crazy Stone	Coulom R(프랑스)	2005	2015	MCTS + Pattern Learning (Bradley-Terry 모델 적용)	2007, 2008 UEC컵 우승, 2013년 제1회 전성전에서 이시다 九단에게 3집 승리	6d
Zen	요지 오지마 (일본)	2009	5	MCTS	2009년, 2011년 컴퓨터 올림피아드 우승, 2012년에 다케미야 마사키 九단 에게 5점, 4점 접바둑으로 각각 11집, 20집 승	6d
Pachi	Baudiš, P(체코)	2012	10.99	수정 MCTS + UCT (오픈소스)		2d
Fuego	Muller	2010	svn 1989	MCTS + UCT (오픈소스)		1d
GnuGo		2009	3.8	(오픈소스)		5k

<표 1> 상용 바둑 게임 프로그램 비교

○ AlphaGo vs. 기존의 바둑 게임 프로그램의 토너먼트 결과

- 돌을 놓는데 걸리는 계산시간은 최대 5초
- AlphaGo는 기존의 바둑 게임 프로그램들과의 토너먼트에서 총 495게임 중 494번 승리 (승률 99.8%)
- 4점 접바둑 게임의 경우에는 CrazyStone, Zen, Pachi 와의 대국에서 각각 77%, 86%, 99%의 승률을 보임
- (분산 AlphaGo vs. AlphaGo) 분산 AlphaGo의 승률이 77%
- 토너먼트 결과 AlphaGo 는 Elo Rating 2890 의 높은 점수를 얻음. (분산의 경우 3140)

Computer Player	Version	Time settings	CPUs	GPUs	KGS Rank	Elo
Distributed AlphaGo	See Methods	5 seconds	1202	176	–	3140
AlphaGo	See Methods	5 seconds	48	8	–	2890
CrazyStone	2015	5 seconds	32	–	6d	1929
Zen	5	5 seconds	8	–	6d	1888
Pachi	10.99	400,000 sims	16	–	2d	1298
Fuego	svn1989	100,000 sims	16	–	–	1148
GnuGo	3.8	level 10	1	–	5k	431

[그림 2] 토너먼트 결과

3. 게임 트리 탐색 알고리즘

□ 인공지능으로 게임을 구현하면 주로 게임 트리를 구성하고 최적의 경로를 예측하는 게임 트리 탐색 알고리즘을 사용

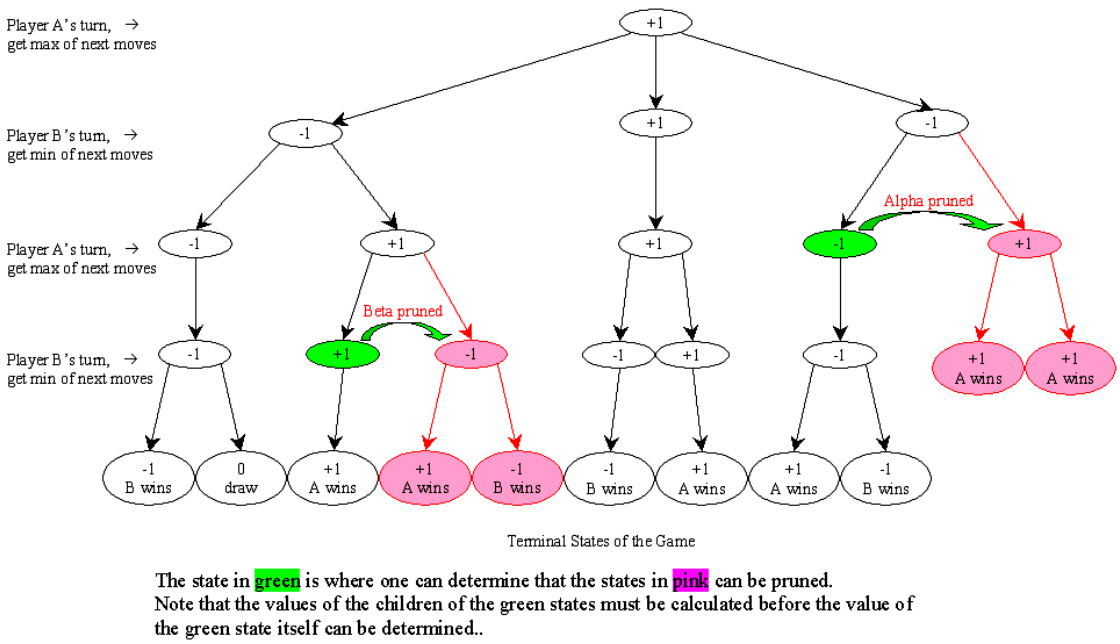
○ 체스나 장기, tic-tac-toe 와 같이 두 플레이어가 번갈아가며 수를 두는 게임에 주로 사용되는 알고리즘

- 트리의 각 노드는 게임의 상태 (게임의 특정 상태에서 말의 위치 혹은 배치)를 의미

- 각 노드의 자식노드(Child Node)는 한 수(게임상 말의 움직임)이후 도달할 수 있는 다음 상태를 의미

- 게임 트리를 이용하여 다음 수를 미리 예측하고 가장 유리한 수를 찾음

* 게임에서 한 플레이어에게 유리한 수는 다음 플레이어에게는 불리한 수 (그림 3)[1)]



[그림 3] 게임 트리의 탐색

1) https://www.clear.rice.edu/comp202/08-fall/lectures/games2/

□ 게임 트리 탐색 알고리즘의 인공지능은 막대한 계산량을 지능적인 방법으로 줄이는 것이 목표

○ 게임에서 시작부터 끝까지 모든 상태에 대한 완전한 탐색은 단순한 게임을 제외하고 현실적으로 불가능

- 바둑(19x19)보다 탐색 정도가 낮은 체스(8x8)의 경우만 해도 완전한 게임 트리에는 약 1040개의 노드가 존재 (약 35^{80} 가지의 경우의 수)
- 효율적인 탐색을 위해 휴리스틱(Heuristic) 기법, 깊이 또는 너비 우선 탐색 기법이 사용되지만, 이것조차도 복잡한 게임에서는 충분한 도움이 되지 않음
- 바둑은 게임중에서도 극단적으로 계산량이 많아서 가장 어려운 문제로 알려져 있음 (약 250^{150} 가지의 경우의 수)

4. MCTS 알고리즘

□ 고전적 게임 탐색 방법과 몬테카를로 시뮬레이션을 결합하여 게임 트리 탐색

○ MCTS(Monte Carlo Tree Search)는 모든 트리 노드를 대상으로 하는 대신 게임 시뮬레이션을 통해 가장 가능성이 높아 보이는 방향으로 행동을 결정하는 탐색 방법

- 게임 트리를 탐색할 때 가능성이 높은 방향으로 게임 시뮬레이션을 실행하여 결과를 확인하고 가능성을 조정하는 트리 탐색 방식
- 다음 수를 선택할 때 시뮬레이션을 충분히 많이 할 수 있으면 최적에 가까운 선택을 할 수 있으며 특히 복잡도가 높은 바둑게임에서도 좋은 성과를 보임
- 가능성이 높은 방향을 짐작하기 위해 **정책(policy)**이 필요하고 트리의 각 노드에서 승리할 가능성을 계산 또는 추정하기 위해 **가치(value)** 함수가 필요함
- 트리의 깊이가 깊은 바둑의 경우에는 시뮬레이션도 최종 노드(계가 단계)까지 실행하는 대신 적절한 단계에서 추정값으로 대치할 수 있음. 그래도 시뮬레이션으로 여러 단계를 더 진행했으므로 보다 정확한 추정값을 계산할 수 있음. 바둑게임의 수읽기와 유사.
- 정책이 필요한 이유는 가치 함수가 추정치이기 때문. 일반적으로 게임 상태를 보고 계산하는 가치 함수에 의존하는 것보다 게임의 특성과 경험이 반영된 정책으로 방향을 짐작하는 것이 더 효과적

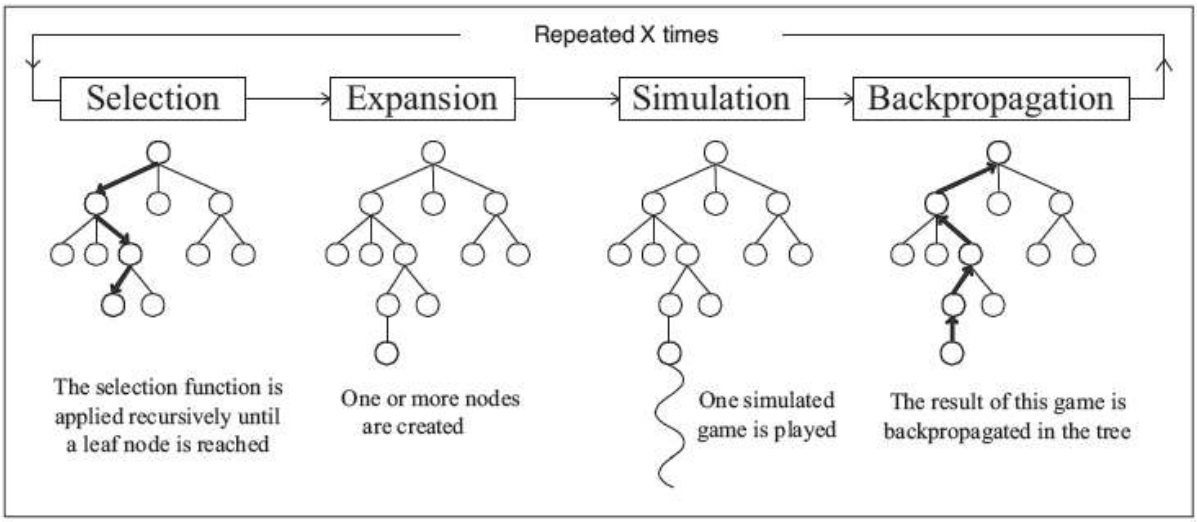
□ 바둑에서의 MCTS 알고리즘

- 바둑에서 **정책**은 “2선을 기지마라”, “빈삼각은 두지마라”와 같은 전문가의 전략이 될 수도 있고 과거 기보 데이터에서 많은 사람들이 선호한 패턴이 될 수도 있는 등 지식의 집결체임 (선호되는 방향을 정의)

- 가치함수는 바둑 국면의 형세판단을 수치화한 것으로 볼 수 있으며 시뮬레이션으로 끝까지 가보기 전에는 계산이 어려움

①선택(Selection) : 뿌리노드에서 시작하여 현재까지 펼쳐진 트리를 선택
②확장(Expansion) : ① 에서 선택한 트리에서 게임의 종료가 되지 않는 경우 하나 이상의 자식노드를 생성하여 선택
③시뮬레이션(Simulation) : ②에서 선택한 자식노드에서 게임의 시뮬레이션을 돌려 게임이 종료될 때 까지 수행
④역전파(Backpropagation) : 선택된 경로의 노드에 시뮬레이션 결과를 반영

<표 2> MTSC의 4단계 과정



[그림 4] MCTS 알고리즘

- 기존의 MCTS를 사용한 바둑 프로그램 경우 아마추어급 성능 실현

□ 딥러닝 기술의 등장과 계산능력의 발전으로 정책과 가치함수의 성능을 획기적으로 개선할 수 있는 가능성이 생김

○ 구글이 이런 아이디어를 구현한 결과가 AlphaGo임

5. AlphaGo의 특징

□ AlphaGo의 특징

○ AlphaGo가 기존연구와 차별되는 기여는 정책(Policy) 선정과 가치(Value) 함수

- 몬테카를로 트리 탐색(Monte Carlo Tree Search, MCTS) 알고리즘을 사용하지만 정책(탐색할 수 결정)과 가치함수 계산에 딥러닝 기법을 적용하여 성능을 획기적으로 개선

- AlphaGo에서 사용한 딥러닝 기법은 지도학습과 강화학습

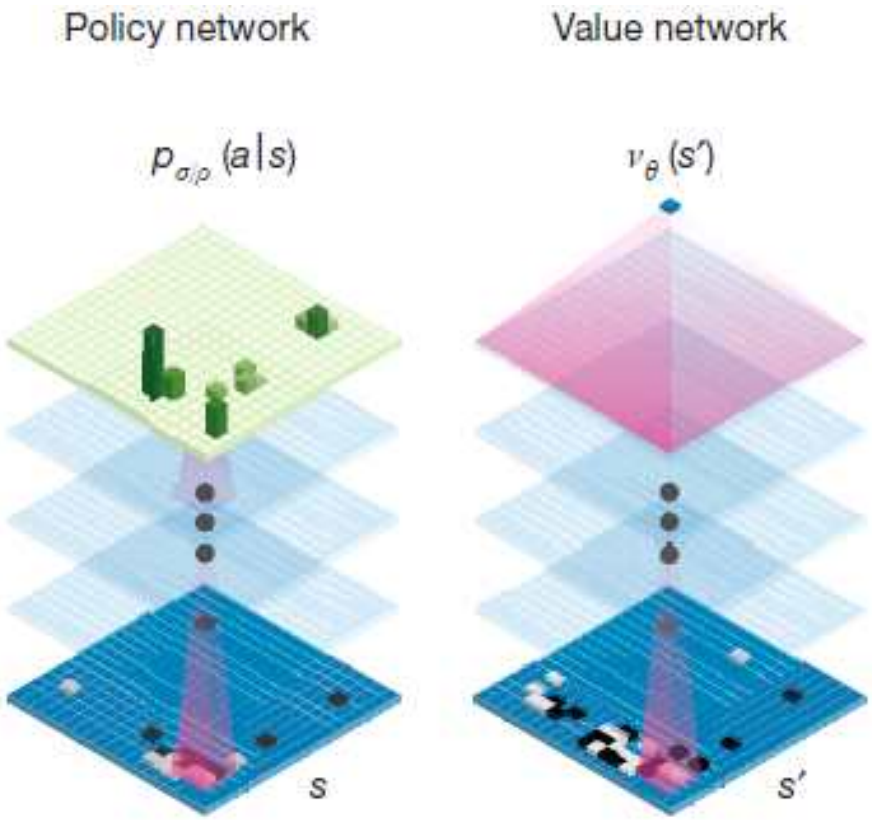
○ **정책 네트워크(Policy Network) : 정책 계산을 위한 딥러닝 신경망**

- 정책 네트워크에서 사용된 딥러닝 기법은 컨볼루션 신경망(Convolution Neural Network)으로 19x19 바둑판 상태를 입력하여 바둑판 모든 자리의 다음 수 선택 가능성 확률 분포를 출력 (그림 5 참조)[2)]

* 컨볼루션 신경망은 페이스북의 얼굴인식 기술인 DeepFace에 적용된 기술로 입력 이미지를 작은 구역으로 나누어 부분적인 특징을 인식하고 이것을 결합하여 전체를 인식하는 특징을 가짐

- 바둑에서는 국지적인 패턴과 이를 바탕으로 전반적인 형세를 파악하는 것이 중요하므로 컨볼루션 신경망을 활용하는 것이 적절한 선택

2) Nature paper



[그림 5] 정책 네트워크와 가치 네트워크의 구성

○ 정책 네트워크 학습

- **지도학습(supervised learning)** : KGS Go Server의 실제 대국 기보로부터 3000만 가지 바둑판 상태를 추출하여 이 중 약 2900만 개를 학습에 이용하고, 나머지 100만 가지 바둑판 상태를 시험에 이용 (정확도 57%). 이것은 사람이 다음 수를 두는 경향을 모델링 한 것

- **강화학습(reinforcement learning)** : 지도학습의 결과로 구해진 정책네트워크는 사람의 착수 선호도를 표현하지만 이 정책이 반드시 승리로 가는 최적의 선택이라고 볼 수 없음. 이것을 보완하기 위해 지도학습으로 구현된 정책 네트워크와 자체대결을 통해 결과적으로 승리하는 선택을 "강화" 학습함. 강화학습의 핵심은 정책 네트워크 간에 경기를 진행하고(self-play), 이로부터 도출된 경기결과(승패)를 바탕으로 이기는 방향으로 가도록 네트워크의 가중치를 강화(개선). 강화 후에 기존 바둑프로그램인 Pachi와 대결하여 85%의 승률

* 처음에는 지도학습의 결과를 그대로 이용하여 경기를 진행하지만 학습이 진행되면서 여러 버전의 네트워크가 생성되며 이들 간의 강화학습을 통해 실제로 승리하는 빈도가 높은 쪽으로 가중치가 학습됨

* 강화학습의 정책 네트워크 구조는 지도학습과 같으며 신경망의 가중치 값만 달라짐

○ **가치 네트워크(Value Network) : 바둑의 전체적인 형세를 파악**

- AlphaGo에서는 가치 함수도 딥러닝을 이용한 가치 네트워크(value network)로 구현

* 기존 프로그램의 가치함수는 비교적 간단한 선형결합으로 표현

- 인공신경망의 입력층과 은닉층 구조는 정책네트워크와 유사한 컨볼루션 신경망이지만 출력층은 현재의 가치(형세)를 표현하는 하나의 값(scalar)이 나오는 구조

- 특정 게임 상태에서의 승률(outcome)을 추정

* 게임에서 이길 경우의 승률을 1이라고 볼 때, 가치 네트워크의 오차는 약 0.234 수준

* 게임에서 이길 경우의 승률을 1이라고 볼 때, 가치 네트워크의 오차는 약 0.234 수준

□ MCTS에서 정책네트워크와 가치네트워크의 활용

○ 정책 네트워크에 현재 상태를 입력하여 탐색할 노드 결정 (그림 6)[3)]

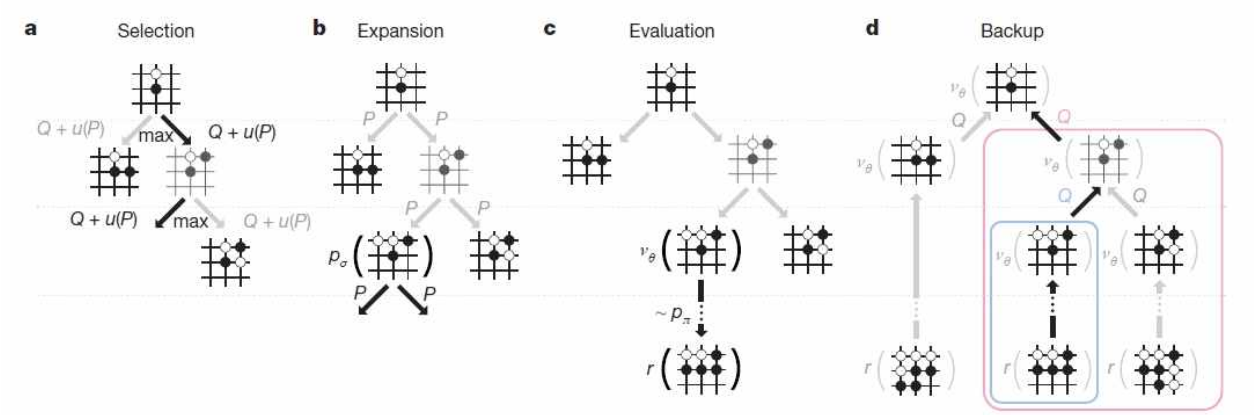
○ 시뮬레이션의 최종 노드에서 승리할 가능성을 계산하는데 초기 추정치로 가치 네트워크를 이용. 이 값은 MCTS 알고리즘에 의해 개선됨

○ 시뮬레이션의 Evaluation 단계에서 fast rollout에 이용하는 정책 네트워크는 간단한 신경망을 지도학습한 버전을 이용하여 빠르게 계산할 수 있도록 단

3) Nature

순화. 구글 보고서에 의하면 이 단계에서 단순한 버전을 이용해도 기존 정책네트워크를 이용한 경우와 유사한 성능을 보였다고 함.

○ 기존의 바둑 인공지능 프로그램이 아마추어급에 국한된 반면, AlphaGo는 위 두 가지 인공신경망 학습알고리즘을 적용하여 프로기사급 성능 달성



[그림 6] AlphaGo의 몬테카를로 트리 탐색

6. 결론 및 시사점

□ 인공지능 딥러닝 기술의 성능을 보여준 또 하나의 실증 사례

○ 바둑은 아직도 컴퓨터가 사람을 이기기 어려울 것으로 예상하고 있었으나 매우 가까워졌음

○ 이번 이세돌 9단과의 승부 결과에 관계없이 의미있는 성취임

○ 페이스북에서도 바둑 프로그램을 개발하고 있는 등 인공지능 기술을 이용하여 어려운 문제에 도전하는 사례가 급증하고 있음

□ 실험을 위한 대용량 계산 및 저장 컴퓨팅 자원을 이용할 수 있는 연구개발 환경이 기술발전에 중요한 요소

○ 구글의 업적은 인정하지만 동일한 아이디어를 가지고 있더라도 실제로 이것을 구현하여 실현할 수 있는 조직은 전 세계에 많지 않음

○ 국내 인공지능 연구개발의 지원을 위해 실험용 컴퓨팅 파워 제공 환경 조성이 필요

[참고문헌]

1. 국내문헌

2. 국외문헌

Silver, D. et al., "Mastering the game of Go with Deep neural networks and tree search," Nature vol 529, pp. 484-489, 28 Jan 2016.

3. 기타(신문기사 등)

"Google AI algorithm masters ancient game of Go," Nature.com, 27 Jan. 2016, http://www.nature.com/news/google-ai-algorithm-masters-ancient-game-of-go-1.19234

"이세돌 신의 한 수 vs 알파고 컴의 한 수," chosun.com, 2016.01.28., http://news.chosun.com/site/data/html_dir/2016/01/28/2016012800485.html

Monte Carlo tree search, Wikipedia, https://en.wikipedia.org/wiki/Monte_Carlo_tree_search

Convolutional neural network, Wikipedia, https://en.wikipedia.org/wiki/Convolutional_neural_network

Reinforcement learning, Wikipedia, https://en.wikipedia.org/wiki/Reinforcement_learning

Elo rating system, Wikipedia, https://en.wikipedia.org/wiki/Elo_rating_system

주 의

1. 이 보고서는 소프트웨어정책연구소에서 수행한 연구보고서입니다.
2. 이 보고서의 내용을 발표할 때에는 반드시 소프트웨어정책연구소에서 수행한 연구결과임을 밝혀야 합니다.