|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **교과목 명** | | 종합설계2 | | | | | | | | |
| **설계 제목** | | 인공지능을 활용한 게임 자동화 방안 | | | | | | | | |
| **설계 기간** | | 2020년도 2학기 | | | | | | | | |
| **지도교수** | | 이재환 | | | | | | | | |
| **팀원** | | **이름** | 김범준 | | **학번** | 2015124035 | **☎** | 01041781793 | **E-mail** | [tk2010kbj@naver.com](mailto:tk2010kbj@naver.com) |
| **이름** | 이태형 | | **학번** | 2015124176 | **☎** | 01053759177 | **E-mail** | rkrp15@gmail.com |
| **이름** |  | | **학번** |  | **☎** |  | **E-mail** |  |
| **목표설정** | **설계** **주제** **선정** **배경** | 게임 산업은 계속 성장하고 있고, 게임 유저들이 원하는 게임의 퀄리티는 점점 더 높아지고 있다. 유저들이 원하는 게임 퀄리티를 결정하는 요소들 중 하나는 게임내의 유저의 개입없이 자동으로 동작하는 인공지능이 얼마나 기계적이지 않고 자연스럽게 동작하는 지이다.  우리는 다양한 게임의 인공지능들 중에서, 스타크래프트2의 인공지능을 주제로 선정했다. 스타크래프트2는 10^1685가지의 많은 경우의 수를 가진, 그래서 모든 동작을 정의해주는 전통적인 소프트웨어 설계 방법론으론 풀기 어려운 문제이다. 우리는 이 어려운 문제를 푸는 과정에서 인공지능을 개선시킬 방안을 찾을 것으로 기대하고 설계 주제를 선정했다.  스타크래프트2를 주제로 선정한 다른 이유에는 스타크래프트2가 다른 게임들에 비해 학습 환경구축이 용이하고 양질의 데이터를 구할 수 있다는 장점이 있었기 때문이다. 스타크래프트2를 제작한 블리자드 엔터테인먼트사는 단순히 게임을 서비스하는 것으로 그치지 않고 게임과 통신할 수 있는 API를 제공한다. 그래서 이 API를 기반으로 딥마인드사에서 제작한 스타크래프트2 학습 환경 pysc2 라이브러리가 존재했다. 그리고 유저들의 게임 플레이를 REPLAY 파일 형식으로 제공하는데, 이를 이용하여 다양한 방법론을 시도해볼 수 있을 것으로 기대했다. | | | | | | | | |
| **설계** **목표** | 1. 스타크래프트 2 탑재 인공지능 (난이도: 아주 쉬움) 승리  2. 스타크래프트 2 탑재 인공지능 (난이도: 쉬움) 승리  3. 스타크래프트 2 탑재 인공지능 (난이도: 보통) 승리  4. 스타크래프트 2 탑재 인공지능 (난이도: 어려움) 승리  5. 일반인 유저(상위 16%, level : Silver) 승리 | | | | | | | | |
| **합성** **/**  **분석** | **관련** **기술** **조사, 분석** | - Alphastar  Alphastar는 딥마인드사에서 제작한 스타크래프트2 인공지능 플레이어다. 이 Alphastar는 지도학습과 강화학습 방법론을 모두 사용하여 상위 0.2%(Grandmaster level)을 달성하였다. 먼저 유저들의 리플레이 파일로 지도학습을 실시하여 상위 16%를 달성하였고, 그리고 A3C 기반으로 강화학습을 수행하여 상위 0.2%를 달성하였다.  - DQN  강화학습 알고리즘으로 유저의 훈련 데이터 없이 네트워크를 훈련시킬 수 있는 알고리즘이다. 보상을 기반으로 한 알고리즘으로, 어떤 환경에서 어떤 액션을 취했을 때의 보상에 대한 예측을 훈련시키고, 가장 큰 보상이 예상되는 액션을 선택하는 식으로 동작한다.  - Atari Net  Atari game을 플레이한 인공지능이 사용한 네트워크 구조로, convolution layer와 dense layer만을 사용한 간단한 구조의 네트워크이다. | | | | | | | | |
| **제작** | **설계** **내용** | 1. 게임의 단순화  스타크래프트2에는 10^1685의 수많은 경우의 수가 있기 때문에 이것을 모두 고려하여 행동을 결정하는 네트워크를 만드는 것은 어렵기 때문에 몇 가지 제약을 걸어 경우의 수를 줄여 인공지능의 성능을 향상시켰다.  [1] 게임 시작 위치고정  스타크래프트2는 게임을 시작하면 정해진 몇 개의 위치 중 임의의 위치에서 게임이 시작된다. 이것으로 인한 경우의 수를 줄이기 위해 게임 시작 위치를 한 곳 정해두었다.  [2]. 가능한 동작의 수 제한  스타크래프트2에는 573개의 동작이 있는데 이중 한 종족에서 사용할 수 있는 동작은 약 400개이다.  그리고 여기서 게임 초반에 사용할 수 있는 동작은 20~30개에 불과하다. 그래서 인공지능이 약 400개의 동작을 모두 고려하여 동작을 결정하는 것은 비효율적이므로 이를 제한하여 573개의 동작 중 4개의 동작만을 사용하여 이 중 2개를 연속적으로 수행하는 3개의 동작을 새로 정의하여 사용했다.  [3]. 게임 시작 초반 고정된 동작 수행  게임을 시작하고 얼마 지나지 않았을 때의 동작은 게임에 큰 영향을 주며 다양한 상황을 고려하여 결정해야 한다. 그래서 게임 시작 약 3분간 고정된 동작을 수행하면 인공지능이 처리할 것을 줄이고 안정적인 성능이 나온다. 그래서 게임 시작 3분간 고정된 동작을 수행하도록 제약을 걸었다.  2. 네트워크 구조 설계  EMB00004944488b  게임 시작 3분 이후부터는 인공지능을 사용하여 게임을 플레이했다. 사용한 네트워크의 구조는 ATARI 게임을 플레이하는데 사용된 ATARI 네트워크 구조를 사용하였다. 게임 지도 정보를 처리하는데 Convolution 층이 사용되었고 게임 정보 처리에 Dense 층을 사용했다. 스타크래프트2 게임의 많은 경우의 수와 게임 플레이 특성에 비해 단순한 구조인데, 이런 단순한 구조를 사용한 건 인공지능의 동작을 극적으로 단순화시켰기 때문이다.  3. 훈련 알고리즘  앞에서 설계한 네트워크를 훈련시키는 알고리즘은 DQN 알고리즘을 사용했다. 사용한 DQN 알고리즘에 대한 간단한 설명은 다음과 같다.  EMB000049444892 | | | | | | | | |
| **설계** **과정** | - 게임 플레이 단순화  [1] 게임 시작 위치고정  [2] 가능한 동작의 수 제한  [3] 게임 시작 초반 고정된 동작 수행  위 과정을 순서대로 진행하였다.  - 네트워크 설계  동작 등의 게임 플레이를 단순화시킨 이후에 인공지능의 네트워크를 설계하였다. 설계는 tensorflow keras를 사용하였다. 훈련의 용이성과 단순화된 문제에 맞춰 단순한 구조의 Atari Net 기반의 네트워크를 설계하였다.  - 훈련 알고리즘 설계  네트워크를 설계한 이후 훈련 알고리즘으로는 DQN 알고리즘을 구현했다. Replay buffer 등의 추가적인 기술은 사용하지 않고 기본 DQN 알고리즘을 사용했다.  - 네트워크 훈련(데이터 조정)  [1] 출력 발산 문제 해결  설계를 마치고 난 이후 훈련을 실시하였다. 출력이 nan(숫자가 아님)이 되는 문제가 발생했는데, 확인해본 결과 입력 데이터 중에 계속해서 증가하는 데이터가 존재하여 출력이 무한대로 발산함이 원인으로 판단하였다. 그래서 모든 데이터들을 0~1 사이의 크기를 가지도록 정규화 시켰고, 출력이 발산하는 문제를 해결하였다.  [2] 훈련  5일동안 500여개의 게임을 훈련시켰다. | | | | | | | | |
| **시험 /평가** | **설계** **결과** **검증**, **분석**, **평가** | 완성된 인공지능이 어느 정도의 성능을 내는지 확인하기 위해 스타크래프트2의 인공지능을 상대로 게임을 시켜 어느 정도의 승률을 보이는지 확인했다.  스타크래프트 인공지능의 난이도는 매우 쉬움, 쉬움, 보통, 어려움, 매우 어려움으로 단계별로 나뉘어져 있는데 매우 쉬움부터 각각의 난이도와 10판의 게임을 하고 승률을 기록했으며 같은 난이도에서 10판을 모두 지면 그 다음 난이도는 이기기 어려울 것으로 보고 테스트를 종료함   |  |  |  |  | | --- | --- | --- | --- | | 난이도 | 승리 횟수 | 패배 횟수 | 승률 | | 매우 쉬움 | 10 | 0 | 100% | | 쉬움 | 10 | 0 | 100% | | 보통 | 0 | 10 | 0% |   표1. 인공지능 상대 결과  위의 방법에 따라 테스트를 진행한 결과 표1과 같은 결과를 얻을 수 있었다.  난이도 매우 쉬움에서는 10판중 10판을 이기고 0판을 패배했고,  난이도 쉬움에서는 10판중 10판을 이기고 0판을 패배했고,  난이도 보통에서는 10판중 0판을 이기고 10판을 패배했다.  보통 난이도에서 10판중 10판 모두를 패배했기 때문에 어려움 난이도에서 이기는 것은 거의 불가능에 가깝다고 예상되어 어려움이상의 난이도를 상대로는 게임을 진행하지 않았다.  결과를 통해 설계한 인공지능이 스타크래프트2 인공지능의 쉬움 난이도와 보통 난이도의 중간 수준의 성능을 내는 것으로 보였으며 이것은 아주 높은 성능은 아니다.  스타크래프트2의 다양한 경우의 수를 줄여 네트워크의 설계를 수월하게 만들기 위해 걸었던 제약으로 인해 인공지능의 플레이에 한계가 생겨 이러한 결과가 나온 것으로 분석된다. | | | | | | | | |
| 추진 일정 대비 진행 결과 | 스타크래프트2에 제약을 주어 인공지능을 만들기 이전 다양한 방법을 시도했었고 스타크래프트2에 제약을 주어 단순하게 만드는 방법은 마지막 3주 동안 진행되었다.   |  |  |  | | --- | --- | --- | |  | 추진 일정 | 진행 결과 | | 1주차 | Pysc2코드를 수정하여 설계와 같이 동작하도록 하기  Keras를 활용하여 네트워크를 설계하고 구현하기 | 게임시작 약 3분간 고정된 동작을 할 수 있도록 따라하게 만들 리플레이 파일을 만들고 그 리플레이 파일에서 action, arg를 추출하고 그 action, arg를 사용하여 게임시작 약 4000 game step(약 3분)동안 고정된 동작을 수행하도록 하였음.  ATARI네트워크를 구현하고 설계내용에 맞추어 조정함. | | 2주차 | Pysc2와 네트워크의 연결  인공지능 학습 | 설계에 맞게 수정된 pysc2와 네트워크를 연결하여 인공지능이 게임을 플레이 할 수 있도록 함.  결과를 계속해서 확인하며 완성된 인공지능의 학습을 진행함. | | 3주차 | 인공지능 학습  결과 확인 | 결과를 계속해서 확인하며 완성된 인공지능의 학습을 진행함.  충분한 학습이 진행되고 인공지능의 다양한 난이도를 상대하여 결과를 확인함. | | | | | | | | | |
| 역할 분담 및 개인별 공헌 내용 | | 김범준  Pysc2코드를 수정하여 설계대로 동작하도록 함  이태형  ATARI네트워크를 구현하고 설계내용에 맞추어 조정함 | | | | | | | | |
| 현실적 제한 요건 반영 결과 | | 경제요건 | | 컴퓨팅 환경: Quadro RTX 6000 | | | | | | |
| 사회적 영향 | |  | | | | | | |
| 미적 요소 | |  | | | | | | |
| 보건 및 안전 | | 오픈소스 라이브러리 사용시 저작권 준수 | | | | | | |
| 내구성 | | 프로그램 안정성 향상 | | | | | | |
| 산업 표준 | |  | | | | | | |
| 토의 | | 본 설계에 대해 기타 논의 사항을 기술한다. 또한, 본 설계를 수행하면서 느꼈던 점에 대해서 공학적인 관점에서 자신의 의견을 기술한다. 종합설계 I의 결과보고서인 경우에는 종합설계 II의 계획을 기술한다.  설계 이전에는 인공지능 네트워크를 데이터만 제공해주면 알아서 동작하는 블랙박스로 생각하고 있었다. 하지만 실제로 인공지능 네트워크를 설계해서 훈련시켜보니 고려해야할 것들이 정말 많다는 걸 알게 되었다. 여러가지 하이퍼파라미터들이 존재하는데 이것을 어떻게 결정해야 하는지에 대한 근거를 찾는 게 정말 어려웠다. 데이터 중에 끝없이 증가하거나 감소하는 게 있는 등 데이터에 대해서도 고려해야할 것이 정말 많다는 걸 느꼈다.  그리고 디버깅이 힘들었는데, 지금까지 해왔던 것과는 달리 keras 라이브러리를 사용했을 때는 최종 출력을 제외하고 다른 값들을 확인하는 것이 어려웠다. 모든 값들을 확인하며 디버깅을 할 수 있었다면 훨씬 문제를 정확하고 빠르게 찾을 수 있었을 거라는 아쉬움이 남는다. 이런 디버깅을 잘하려면 결국 사용하는 라이브러리의 구조를 알고 있어야 한다는 생각이 들었다. 앞으로 조금씩 이라도 이런 라이브러리들을 공부해야 하겠다는 생각이 들었다.  이렇게 종합설계2를 진행하면서 딥러닝 기술을 활용할 때 발생하는 다양한 문제들을 직접 경험하고 발생하는 문제들을 해결하기 위한 다양한 시도를 하며 딥러닝 기술과 소프트웨어에 대해서 더 많이 이해하게 되었다. 추가로 리눅스 환경에 익숙해질 수 있어서 좋았다. | | | | | | | | |