CS492 Search Based Software Engineering

20165501 박준영

1. Symbolic Regression 구현체 실행법

1) Unzip 제출 파일
2) symbreg 디렉토리로 이동
$ cd symbreg
3) training.py 파일 실행
$ python src/training.py {train.csv} Cmd+C 와 같이 실행을 강제 종료하면 symbreg.txt 에 결과가 출력되어 있음
3) test.py 파일 실행
$ python src/test.py {test.csv} stdout 에 MSE 출력

Best solution 은 results/best.txt 입니다.

2. Symbolic Regression 문제 관찰

2.1 Linear Regression

Symbolic regression 은 non-linear 한 연산이 포함되지만 문제를 linear regression 으로 축소시키면, 간단한 행렬연산으로 MSE 를 최소로하는 상수를 찾을 수 있다.

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}$$

조금 더 문제를 단순화 시키면 하나의 symbolic variable 만을 사용한 linear regression 도 가능하다.

2.2 불필요한 동치 연산자

y - x = y + (-x)
1/x = power(x,-1)
sin(x) = cos(x+pi/2) 등 동일한 의미를 갖는 연산자들이 있다.
중복된 탐색을 피하기위해 이러한 동치 연산자들을 mutation, crossover 등의 단계에서 등장하지 않도록 제한하면 search space 를 줄일 수 있다.

3. 시도 1 - 유전자 알고리즘

3.1 Initial population

N 개의 symbolic variable 이 있을 때 2.1 에서 간략하게 설명한 1 개의 symbolic variable 을 사용한 linear regression 을 initial population 으로 사용했다. 따라서 population size 는 N 이다.

3.2 Fitness Function

Fitness 는 MSE 를 사용하였다.

3.3 Parent Selection
Fitness 순서로 오름차순 정렬한 뒤, 앞의 2 개와 나머지들을 Cross-over 하도록 선택하였다.

3.3 Cross-over
Cross-over 로는 두 expression 의 평균을 구하도록 하였다.

3.4 Mutation
현재 cross-over 의 평가를 위해 mutation 은 하지 않았다.

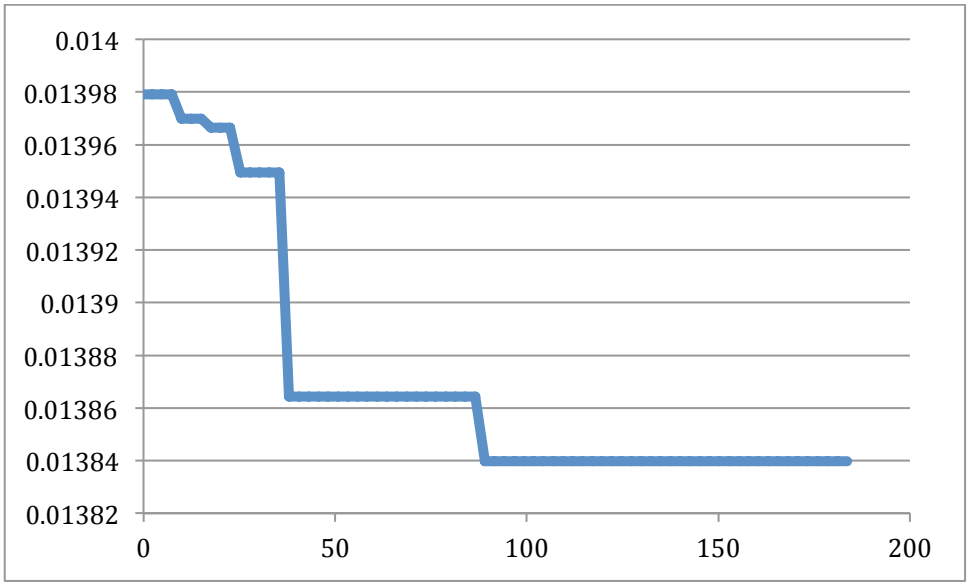
3.5 결론
Cross-over 를 3.3 대로하면, expression 의 길이가 2 배씩 증가하면서 fitness 의 계산 시간이 2 배 늘어나게 된다. population iteration 이 늘어날 수록 exponential 하게 증가하는 문제점이 있다.

4. 시도 2 - Approximation algorithm(Linear Regression)
2.1 에서 설명한 행렬연산을 통한 linear regression 을 적용해보았다.
결과는 leader board score 로 0.014702927612930999 로 좋은 결과를 얻었다.
이 linear regression 결과는 다음 시도에서 seed 로 사용한다.

5. 시도 3 - Heuristic Random Search (sin 함수)
3. 시도 1 에서 GA 의 cross-over 연산이 마땅하게 떠오르지 않아 mutation 이 탐색의 주력이 될 것으로 보았다. 현재까지의 관찰로는 linear regression 을 seed 로 non-linear term 을 mutation 으로 탐색하는 방향이 좋을 것으로 예상된다. mutation 시도 전에, a*sin(bx+c)의 꼴에서 각각의 x 에 대해 최적의 a, b, c 를 찾는 작은 search space 에 대해서 random search 를 진행해보기로 하였다.

5.1 방법
a0 + a1 * sin(a3+a4*x) 꼴로 a0~4 까지 random search 를 사용하였다.
a1 = 0 ~ 0.8
a0 = -a1 ~ a1
a3 = 0 ~ 2*pi
a4 = -100 ~ 100
으로 셋팅하였다.

5.2 결과



X 축은 시간, Y 축은 MSE 이다.
조금이나마 MSE 가 향상되는 것을 볼 수 있다.

6. 아이디어 4 - Random Mutation Search

5.시도 3 에서 생각한 아이디어인 linear regression 을 seed 로 non-linear term 을 mutation 으로 탐색하는 방향이 있었다. 하지만 시간관계상 구현과 결과를 확인하지 못 하였다.

7. 결론

Symbolic expression 은 search space 가 무한하기 때문에 동치인 연산을 없애는 것이 일이 중요하다. 동치인 연산은 fitness 를 계산하는 시간에도 영향을 주기 때문에 더욱 중요하다는 것을 다시 한 번 느꼈다. 다음에는 search space 의 landscape 를 보며 알고리즘을 구상하는 것이 좋은 결과를 얻는데 큰 도움이 될 것 같다. Linear regression 의 결과가 좋아서 더 향상시키는 일이 어려웠다.