Решение задачи символьной регрессии с помощью генетических алгоритмов

Суглобов С.Н.

Российский государственный технологический университет им. К.Э.Циолковского

Символьная регрессия — метод построения регрессионных моделей путем перебора различных произвольных суперпозиций функций из некоторого заданного набора. Суперпозиция функций при этом называется «программой», а стохастический оптимизационный алгоритм построения таких суперпозиций называется генетическим алгоритмом. Так как подобные алгоритмы являются переборными и требуют значительных вычислительных ресурсов, то публикации по данной теме стали появляться в 90-х годах, а значительное развитие они получили после 2000-го года. Наиболее известными исследователями являются Джон Холланд[1] и Джон Коза[2].

Для применения генетических алгоритмов требуется описать решение задачи в “генетическом” виде, в классическом варианте им могут служить битовые строки, описывающие некие параметры решения. Как правило, такое представление решения сложно использовать, и решения могут быть заданы набором данных, некими деревьями или любыми другими структурами, которые могут описывать решение.

Также необходимо описать генетические операторы для решения, с помощью которых генетический алгоритм будет оперировать решениями.

Для генетического представления решения должны быть определены операторы мутации (производит случайное изменение решения), скрещивания(позволяет алгоритму поиска получать новые решения) и оценка приспособленности.

Для нашей задачи символьной регрессии удобно использовать представление множеством выражений (MultiExpression Programming - MEP)[3]. Суть заключается в представлении функции интерпретируемым кодом, каждая операция которого будет считаться отдельной функцией. Первой записью всегда следует инструкция переменной. Выполнение происходит сверху вниз и аргументами инструкций могут выступать лишь записи, расположенные ранее, поэтому исполнение получается линейным. Генетическое решение, представленное в таком виде, представляет сразу множество функций, что хорошо сказывается на характеристиках алгоритма поиска.

Операция мутации для подобной записи может быть определена случайным изменением инструкции, номеров аргументов или параметров в произвольной записи. В зависимости от того, какая именно запись в решении будет изменена, решение может измениться принципиально, либо незначительно.

Операция скрещивания при подобном виде записи решения реализуется достаточно просто, в последовательностях инструкций родителей берется случайная точка разрыва, и детям переходят смешанные списки.

При оценке приспособленности решения каждая запись решения рассматривается как отдельная функция, и для нее высчитывается среднеквадратическое отклонение относительно эталонных данных. Для удобства вычислений эту величину можно нормализовать.

Общая структура генетического алгоритма: (1)Создание популяции случайных решений, (2)Оценка решений, (3)Создание нового поколения, (3a)Выбор наилучших решений, (3b)Скрещивание, (3c)Мутация, (4)Проверка условий остановки эволюции, (5)Возврат к пункту 2.

Генетические алгоритмы могут реализовывать разные стратегии выбора наилучших решений, которые непосредственно сказываются на эффективности алгоритма. Как правило, решения для скрещивания выбираются случайно, а вероятность выбора зависит от его приспособленности.

Литература:
1) John H. Holland, Adaptation in Natural and Artificial Systems, The U. of Michigan Press, 1975
2) John R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press, 1992
3) Oltean Mihai, Multi Expression Programming, Technical Report, Babes-Bolyai Univ, Romania.