The evolution of computer programs

Nanna Grytzell

May 7, 2021

The use of nature as a source of inspiration in machine learning has made it possible to develop computer programs from scratch, by simulating evolution. Such simulations are fairly complex and bring possibilities, as well as challenges, into the machine learning field.

Did you know that there exist some primitive species of ferns that has more than a thousand chromosomes (subset of their genetic material)? Most organisms has 20-80 chromosomes, for instance humans have 46 chromosomes. These ferns appear to have more genetic material than they will ever need. Even more interesting is that a very similar phenomenon, called bloat, also occurs in machine learning.

There is a type of machine learning algorithm, named genetic programming (GP), that is able to design computer programs in a very powerful and effective way. GP is inspired by nature as it is designed to simulate evolution. The GP consist of a population (a set) of individuals, where each individual is representing a computer program. Each individual is supposed to solve a certain problem or task. Just as in Darwin's concept of "survival of the fittest", individuals who solves the task well, will have a high probability of persisting and spreading in the population of programs. In Darwin's terms this would have been called high relative reproductive success, or fitness. While individuals (= programs) that poorly solves the task, will have problems finding a mating program and likely die without passing their code over to the new generation. Over time, the population of programs will evolve and the amount of good computer programs increase, along with their quality.

Normally, this simulation of evolution reaches a point where the quality of the programs will increase more slowly, until improvement cease as the program appears to has hit a maximum in quality. At the same time, the size of the genetic material carried by each artificial individual continue to grow. Here, the population has reached a point where there is no gain in increasing the size of the code. This is similar to what happened during the evolution of the chromosome-rich ferns a long time ago; they reached a point where the size of their genetic material continued to grow, probably without any gain in performance or quality.

A way of combating this phenomenon is to apply another subfield of machine learning, namely artificial neural networks (ANN). These are also inspired by nature, and they have a design that resembles neuron networks in brains. Each artificial neuron is a rather simple unit, but when the neurons work together, they can find pattern in data that humans are sometimes unable to recognise.

In order to boost the evolution of computer programs, I combined GP with ANNs; thereby making two powerful machine learning algorithms to work together. The result of this, were a slower but probably more purposeful growth of the code, or in biological terms, of the genetic material carried by each individual. The aim of my master thesis has been to study the effect of combining GP with ANNs, and the bloat controlling effect were one of my discoveries.