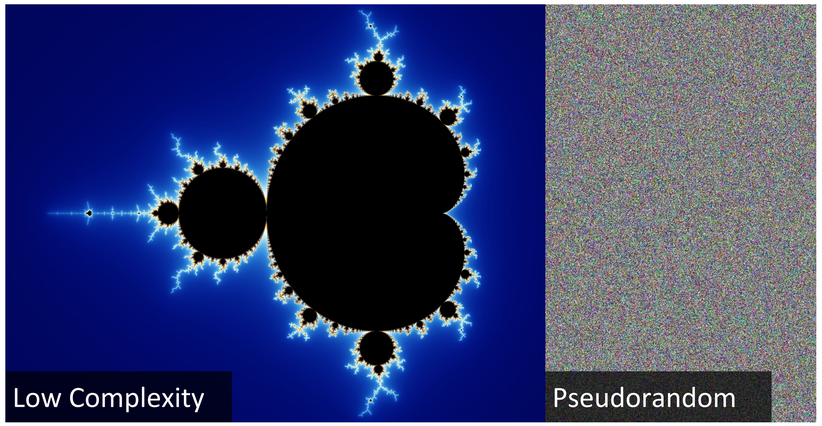
Explicit Pattern Learning and Extrapolation in Machine Intelligence

Lucas Saldyt, Computer Science
Mentor: Ajay Bansal, Assistant Professor
CIDSE

Abstract

This project studies Kolmogorov Complexity and its applications to machine intelligence. Kolmogorov Complexity measures the difficulty of describing an object, but in general is impossible to calculate. However, the minimal description of a pattern yields the best possible extrapolation, and therefore is of interest in machine learning. This project compares previous approaches to the pattern-description problem and proposes a novel type of machine learning model that complements existing approaches.



Problem

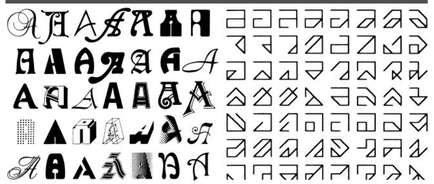
Given a partial input pattern, learn an explicit computer program that completes the pattern.

Introduction

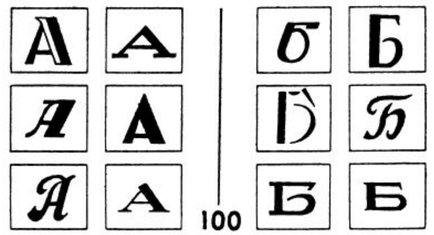
Recent ideas in machine intelligence have suggested that machine learning of explicit models may outperform currently popular "deep learning" approaches. The most influential historical work has focused on learning of integer and character sequences, font styles, music, and abstract visual patterns (Like Raven's Matrices). Modern machine learning does not solve these domains, so compositional program learning may be a superior solution.

Applications/Examples

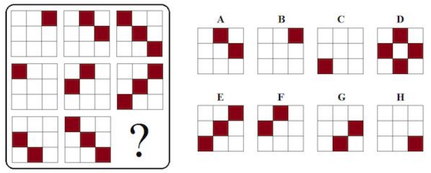
Letter variation: Extend style



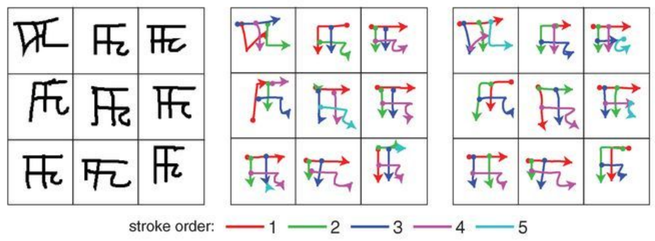
Bongard problems: Classification



Raven's Progressive Matrices: What's the next matrix?



Lake et al: Compositional learning of letters



Ongoing work

Some of the simplest and easiest studied patterns occur in integer sequences. This research proposes an algorithm similar in spirit to Bayesian Program Learning or Neural nets. The algorithm relies on mutating and crossing over sub-portions of the abstract syntax tree in a manner similar to genetic algorithms. The approach described here shows superiority on integer sequences with clear patterns. While traditional approaches can classify sequences with high accuracy, they fail to learn a proper regression in most cases.

Simple functions on integers achieve perfect accuracy

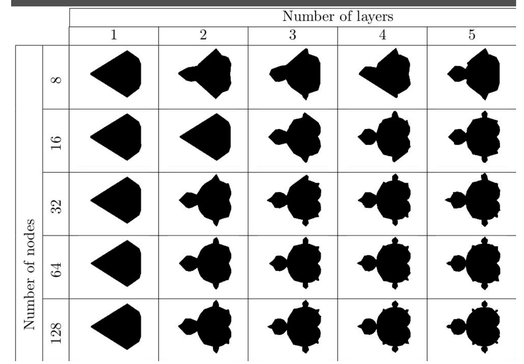
```
def fibonacci(n):
    return 1 if n in {0, 1} else f(n - 1) + f(n - 2)
```

(But not all data is so easily described: see primes)

Functions can be learned from straightforward grammar

```
function   = λ(*identifier) -> body
body       = conditional | expression
expression = (operator expression expression) | atom
atom       = function(*expression) | int | identifier
operator   = + | - | / | * | % | ..
```

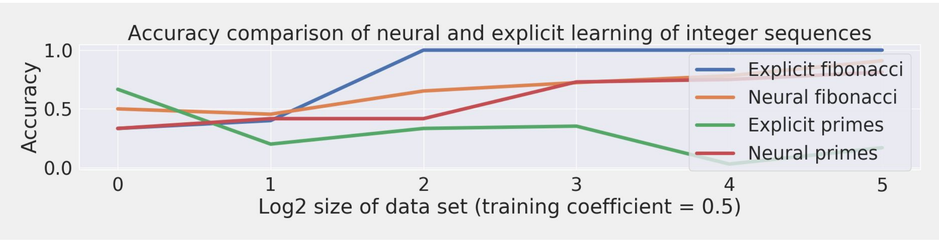
A neural net "learns" mandelbrot set, but how well?



The code describing the mandelbrot set learns it perfectly, and works at any scale, but is hard for machines to learn:

```
def mandelbrot(x0, y0, N):
    x = 0
    y = 0
    iteration = 0
    while x**2 + y**2 < 2**2 and \
          iteration < max_iterations:
        xtemp = x**2 - y**2 + x0
        y = 2*x*y + y0
        x = xtemp
        iteration += 1
    return iteration
```

Illustrative Plot



FURI

Ira A. Fulton Schools of Engineering
Arizona State University