Universal Approximation Theory, Neural Network and Fractal Dimension

J. Fleischman, F. Iulianelli, M. Martino, S. Pack,
C. Taliancic, N. Whybra, and K. Zhao

Tripods REU 2021
Supported by US NSF HDR TRIPODS 1934962

August 13, 2021

Outline:

- Universal approximation theorem using Neural Network
- Fractional Dimension
- Experiment
- Future research and open questions
- References

Part I: Universal Approximation Theorem

Approximation Theorem

Motivation.

Assume you've been given a wavy function, such as $f(x)$, as shown below:

One of the most impressive aspects of using a Neural Network is its ability to approximate the value of any function up to some precision, no matter how complex it is.

Universal Approximation Theorem guarantees this approximation for continuous functions.

First few definitions:

Definition

- We say that $f : \mathbb{R}^d \to \mathbb{C}$ is ρ-Lipschitz continuous if for any $x, y \in \mathbb{R}^d$ the inequality $|f(x) - f(y)| \leq \rho|x - y|$ holds.
- We say that $f : \mathbb{R}^d \to \mathbb{C}$ is Hölder continuous of order $\alpha \in (0, 1]$ if there exists a constant $\rho > 0$ such that

$$|f(x) - f(y)| \leq \rho|x - y|^{\alpha}.$$

Theorem (Universal Approximation Theorem)

Let $f : [-1,1]^d \to [-1,1]$ be a ρ-Lipschitz function. Then, for any fixed $\epsilon > 0$, there exists a Neural Network $N : [-1,1]^d \to [-1,1]$, with the sigmoid activation function, such that $|f(x) - N(x)| < \epsilon$ for any $x \in [-1,1]^d$.

We show the proof for slightly general case.

Theorem

Let $f : [-1,1]^d \to [-1,1]$ be a continuous function. Then, for any fixed $\epsilon > 0$, there exists a Neural Network $N : [-1,1]^d \to [-1,1]$, with the sigmoid activation function, such that $|f(x) - N(x)| < \epsilon$ for any $x \in [-1,1]^d$.

Corollary

The Universal Approximation Theorem also holds for Lipschitz and Hölder continuous functions.

Sketch of proof of approximation theorem:

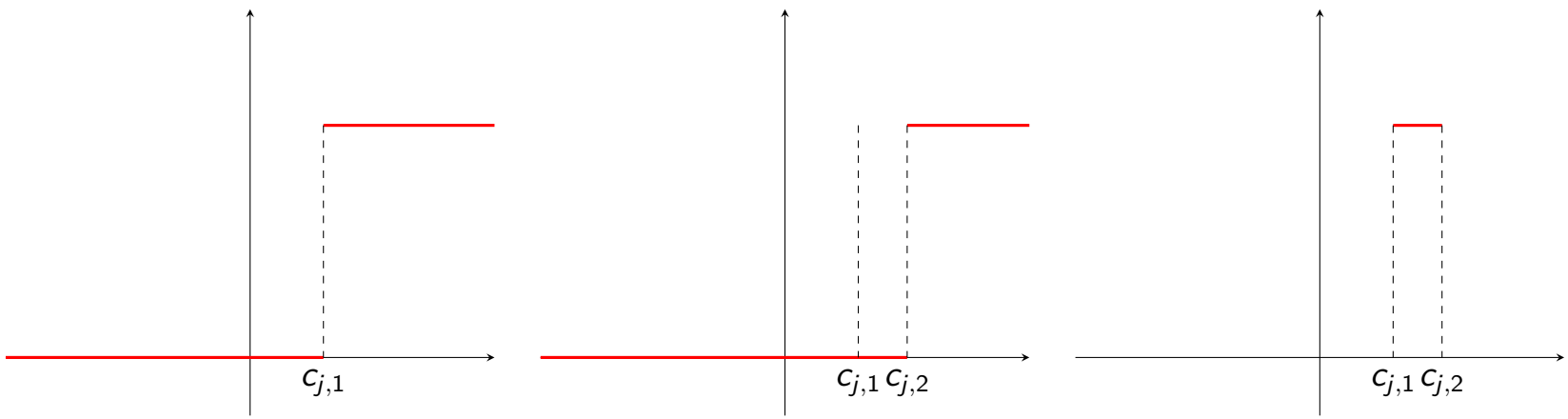
For the sake of simplicity, we prove it in dimension $d = 1$.

- Any continuous function on a compact set is a uniform continuous.
- For a given precision ϵ, we divide the interval $[-1, 1]$ into smaller sub-intervals with sufficiently small length.
- For any sub-interval, we find a "Box" function that can approximate the value of the function in that interval with error ϵ:

$$|f(x) - \alpha_i B(x, b_i)| < \epsilon$$

Sketch of proof of approximation theorem-continued:

- In place of the sigmoid function, the heaviside function has similar behavior and can be used easily for an indicator function.
- Using a translation of the function, we can calculate the distance between the leftmost and rightmost sides of an interval.



(A) Figure A (B) Figure B (C) Figure C

Figure: Box Function and Translation

Use of Sigmoid function.

Any sigmoidal function can be used in order to approximate the Heaviside function:

- The most commonly used type of functions is the sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$.
- This function is an S-shaped curve where:

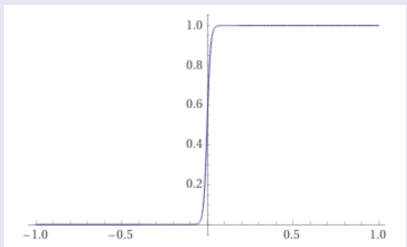
$$\lim_{x \to y} \sigma(x) = \begin{cases} 0 & y = -\infty \\ 1 & y = \infty \end{cases} \tag{1}$$

Use of sigmoid function: continued.

- We can use the following approximation for the heaviside:

$$\lim_{\alpha \to \infty} \sigma(\alpha x) = \mathbb{1}_{[0,\infty)}(x) \tag{2}$$

- In this way, choosing an adequately large value for α, the Sigmoid function approximates the heaviside function. For instance, this would work: $\sigma(x) = \frac{1}{1+e^{-100x}}$



- In the construction of Neural Networks, this approximation is used in order to have an everywhere differentiable activation function.
- This property is required in some learning algorithms

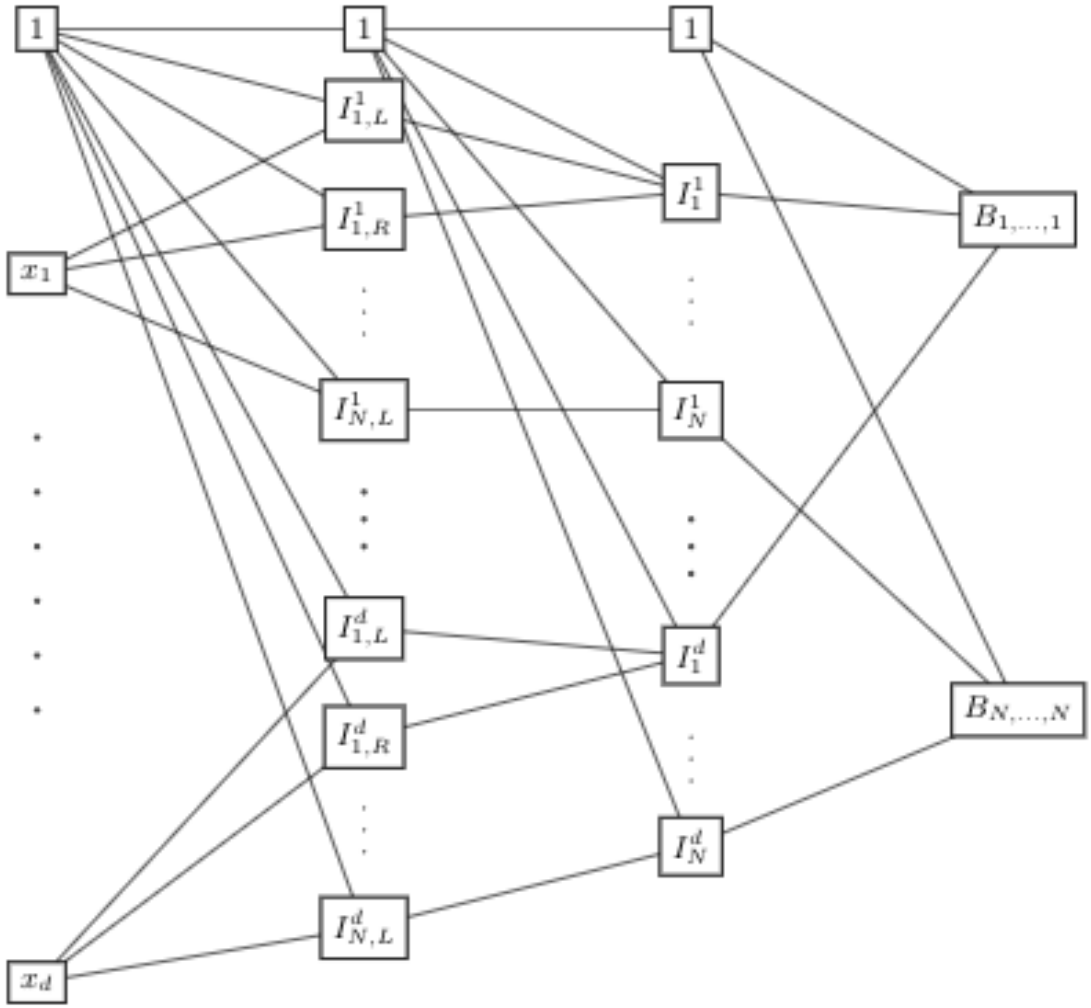


FIGURE: Neural Network in d Dimensions

Pictorial proof using Neural Net

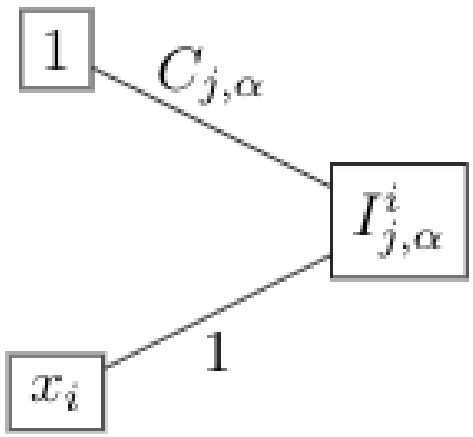


Figure: Input Layer

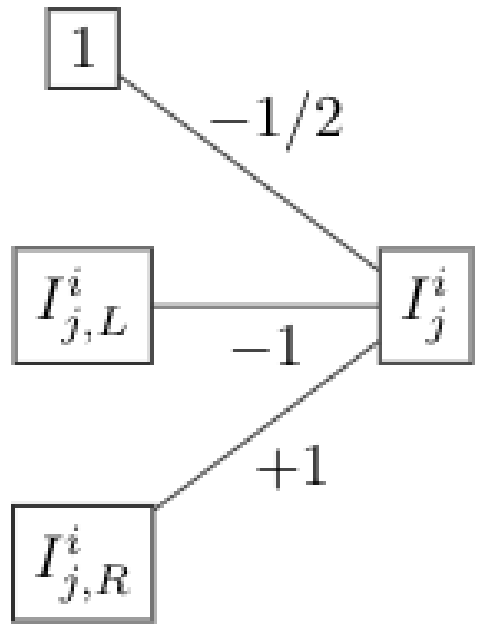


Figure: First Hidden Layer

Pictorial proof using Neural Net

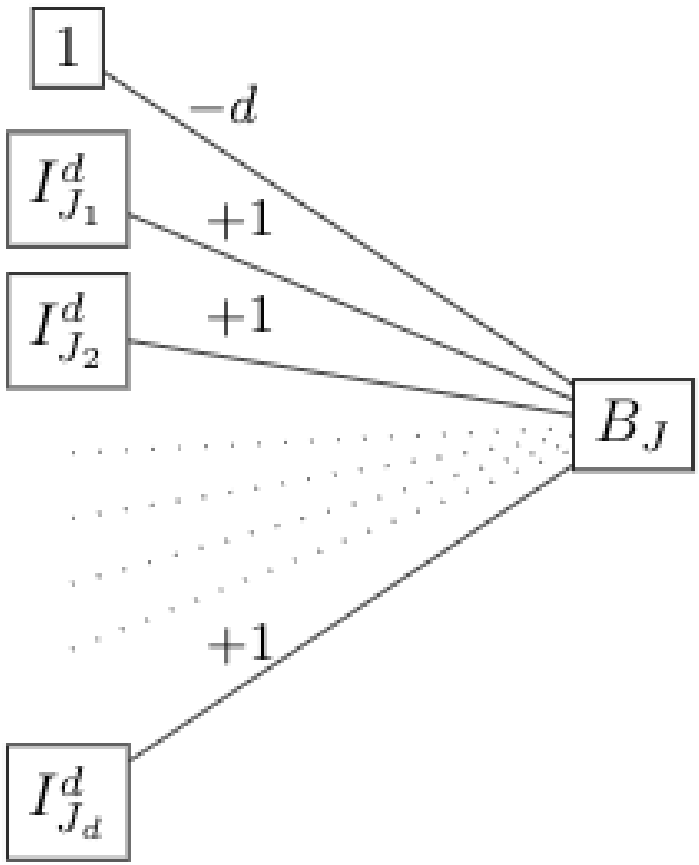


Figure: Second Hidden Layer

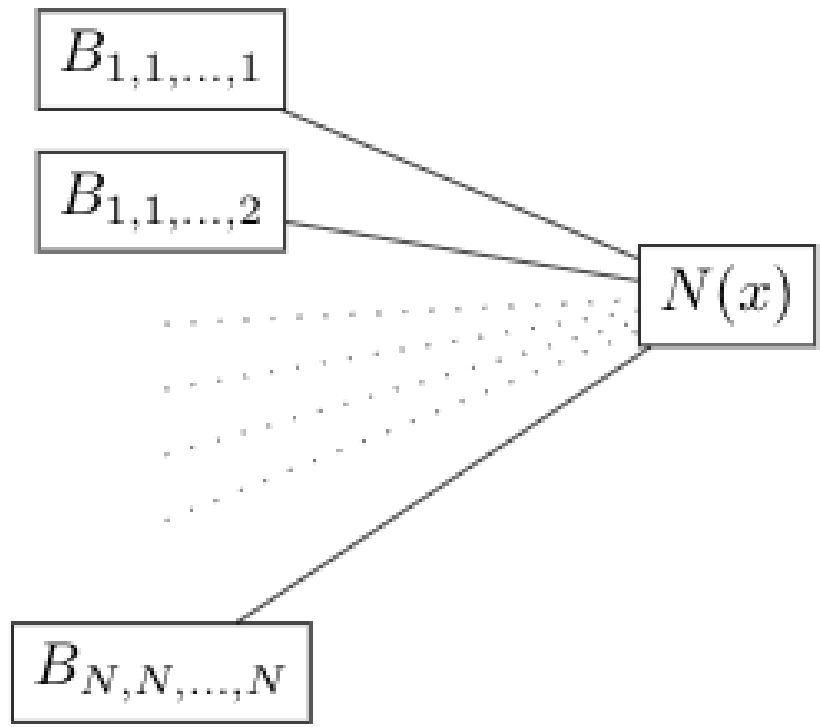


Figure: Output Layer

Summary

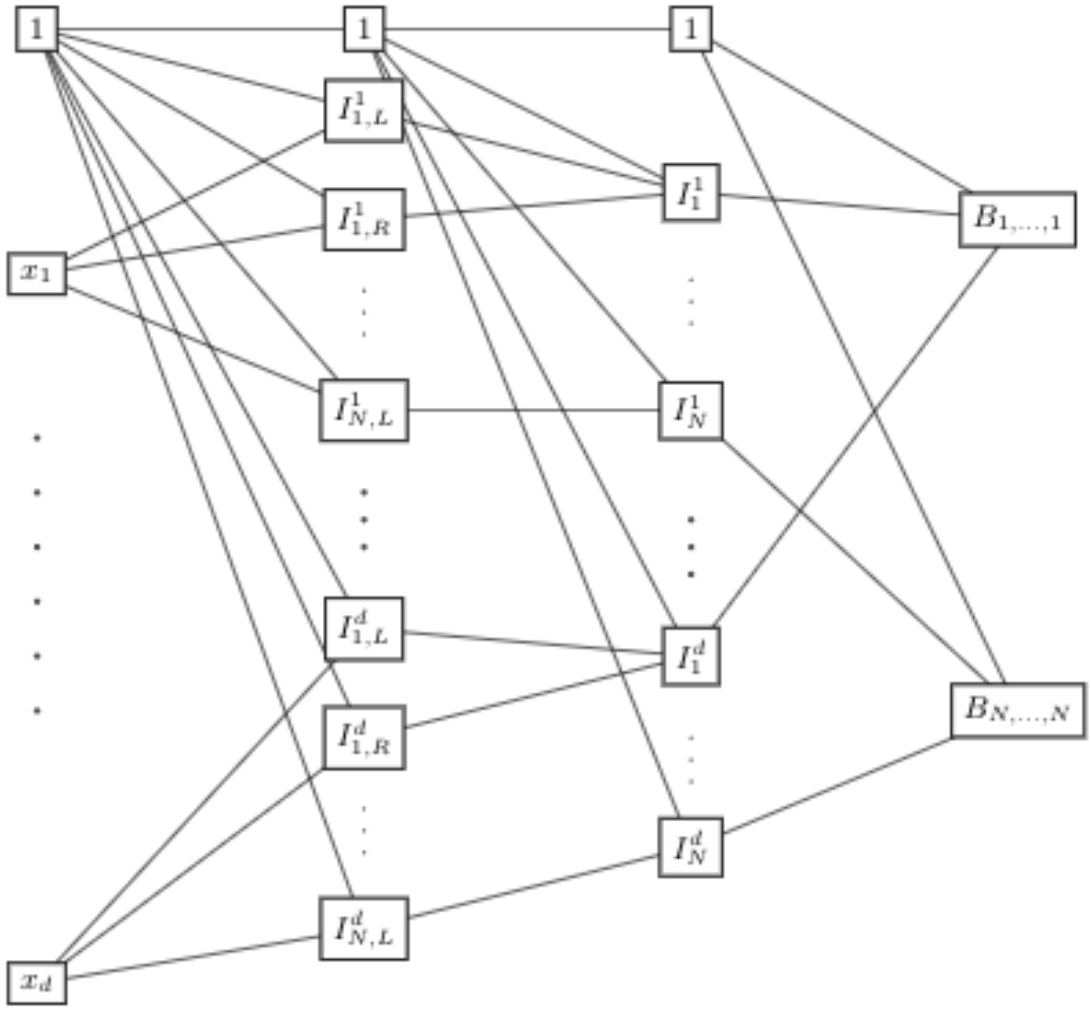


Figure: Neural Network in d Dimensions

Future research

- We are interested to know how some features of functions affect our neural network. More specifically
- How the size of the fractional dimension of graph of a function will affect the run time of a neural network?

Part II: Fractal Dimension

Is Nature Smooth?

- To model real word data sets we often try to use smooth functions like $f(x) = e^x$
- The problem is: nature is usually not smooth

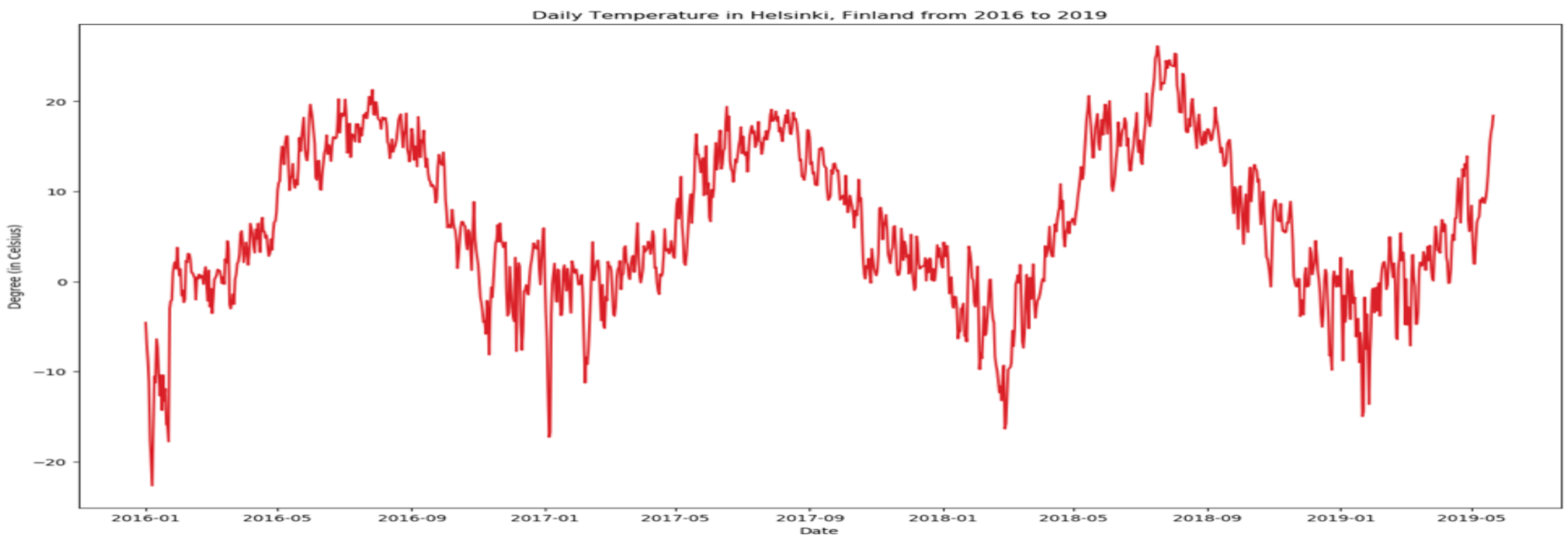


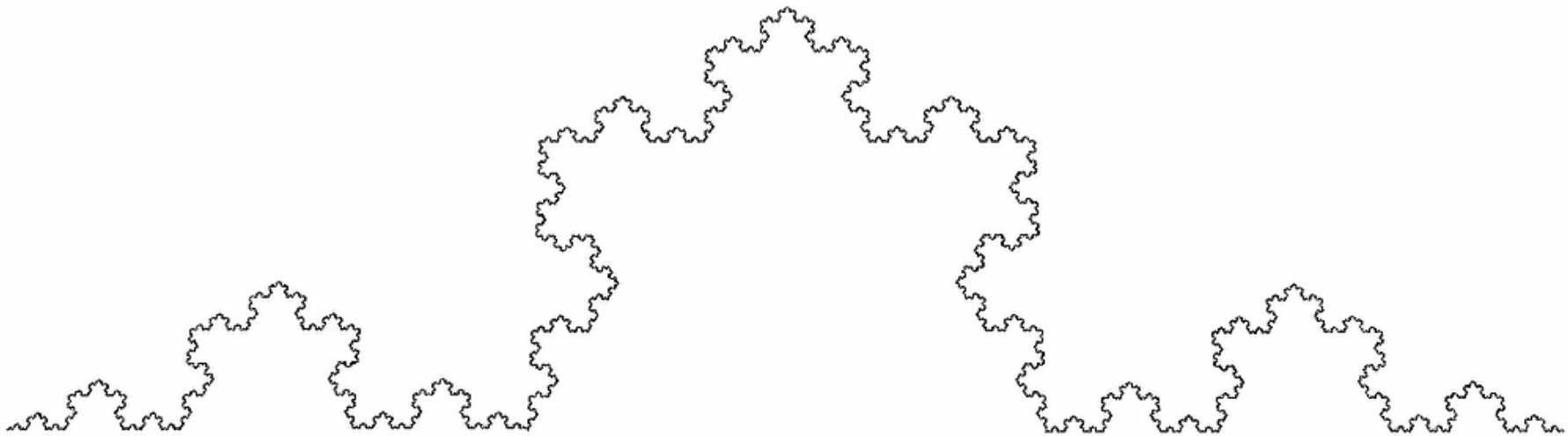
Figure: A time-series plot of the daily temperature in Helsinki, Finland over the course of a few years.

Introducing Fractal Dimension

- In linear algebra, there is a notion of vector space dimension
- For instance a point has dimension 0, a line dimension 1, a plane dimension 2, etc.
- However there are other notions of dimension (Box-counting, Hausdorff, Correlation, Information, etc.)
- These notions of dimension agree with the vector space dimension for all sets with integer valued dimensions, but expand on the idea by allowing for sets of non-integer or "fractal" dimension

Introducing Fractal Dimension

- Fractal sets often display a property called self-similarity.
- Informally this means that the graph looks the same when you zoom into it. For instance the Koch curve below has this property and has box-dimension ≈ 1.26



- Curves with fractal dimension often make appearances in time series data sets and motivates us to try modeling data using functions with fractal dimension

Our Goal

- We consider a time series, i.e. a set

$$P_n = \left\{ p_a := \left(\frac{a}{n}, f\left(\frac{a}{n} \right) \right) : 0 \leq a \leq n \right\}$$

  where $f : [0,1] \to [0,1]$.
- The notions of dimension discussed earlier really only apply to continuous curves, but we are dealing with a discrete time series
- To resolve this, we use something called the "discrete energy" of P_n which is defined as

$$I_s(P_n) = \sum_{a \neq a'} |p_a - p_{a'}|^{-s}$$

Our Goal

- Let γ be the graph of a function f. Then the discrete Hausdorff dimension of γ is the largest value of s such that $I_s(P_n)$ is bounded independently of n, or

$$dim_H(\gamma) = \sup\{s : I_s(P_n) \leq C\}$$

  where C has no dependence on n.
- Any Lipschitz (e.g. smooth) function has a Hausdorff dimension of 1. This makes sense since we are essentially plotting lines!
- It makes no sense to consider functions with dimensions greater than 2. (Intuitively: If a plane is 2 dimensional, how does one plot something of dimension bigger than 2 on the plane itself?)

Our Goal

- This means s has to be in the range $[1, 2]$.
- If our goal is to model functions with fractional dimensions, it helps to know what some of these functions are! So the natural question to ask is...
- For every s in $[1, 2]$, can we can construct a function $f_s : [0, 1] \to [0, 1]$, so that the discrete Hausdorff dimension of the graph of f_s is s?
- The answer is: Probably yes, but we haven't figured it out yet...

Possible Solutions?

- The Mandelbrot-Weierstrass function:

$$W(x) = \sum_{i=1}^{\infty} \lambda^{(s-2)i} \sin\left(\lambda^i x\right)$$

  where $\lambda > 1$.
- $W(x)$ has been shown in [A. Zaleski, 2012] to have a box-dimension of s for s in $[1, 2]$ when λ is big enough.
- The problem with $W(x)$ is that it's pretty complicated and we are ideally looking for something more simple.
- Also the result was shown for the box-dimension and not the discrete Hausdorff dimension.

Possible Solutions?

- The Koch curve is constructed by removing the middle third of a line segment and making an equilateral triangle such that there are now 4 line segments that all have length $\frac{1}{3}$. This process is repeated on all the new line segments, and this continuous forever.

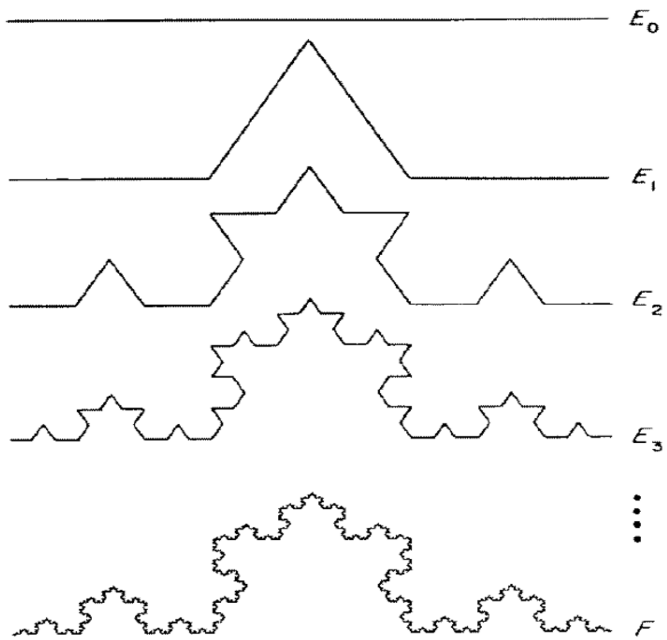


Figure: Construction of Koch curve.

Possible Solutions?

- In each iteration, the line segments shrink by a factor of $\frac{1}{3}$, and the dimension of the curve is 1.26.
- If we change the scaling factor $a = \frac{1}{3}$ to something else, can we get all dimensions between 1 and 2?
- Sadly no. [K. Falconer, 2014] has a theorem that says that a must be in $[0, \frac{1}{3}]$, and if s solves the equation $a^s + 2\left(\frac{1}{2}(1-a)\right)^s = 1$, then $dim_H(Curve) = s$.
- By changing a, we can only achieve s in $[1, 1.26...]$ this way.

Possible Solutions?

- There is still room to play with this idea. Maybe we can change the number of line segments generated in each iteration and make different shapes?
- The main problem with this approach is that it involves looking at a weirdly defined curve and not an easily written function.

Other Ideas:

- Generalization of Cantor Set?
- Generalization of Devil's Staircase?
- Weird conditions on a general function f?

Part III: Experiment and Simulations

Motivating Questions

- How does Hausdorff Dimension empirically appear to influence the learnability of a function?
- Does the chaoticness and tendency to fluctuate inherent to many functions with a non-integer Hausdorff dimension limit this learnability?
- Is it ultimately possible for a neural network to, under reasonable run-time constraints, approximate any fractal time series?

Experiment 1 Outline

Examining the Empirical Relationship between Hausdorff Dimension of a Time Series and Run-time and Test Error

- The Mandelbrot Weierstrass function returns!

$$W(x) = \sum_{i=1}^{\infty} \lambda^{(s-2)i} \sin\left(\lambda^i x\right)$$

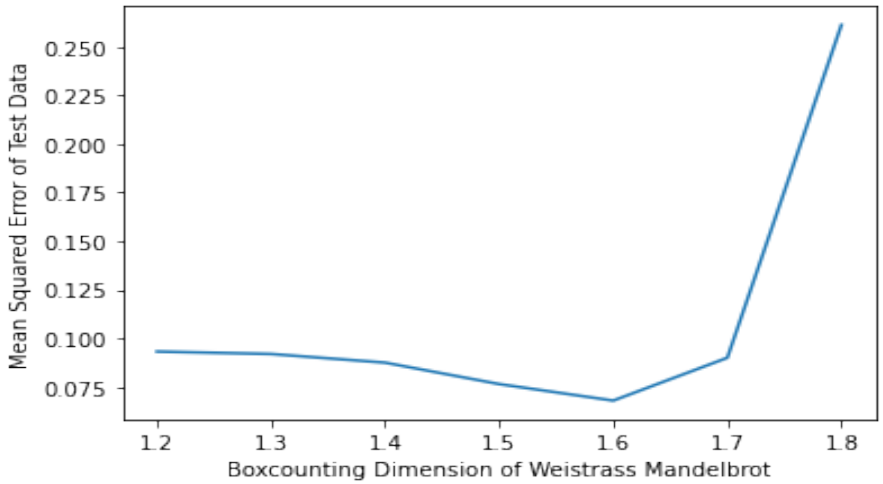
  (although s is only the box-counting dimension D_0, this is useful information as it does bound the Hausdorff dimension from above).
- Different values of s give different ranges of D_0

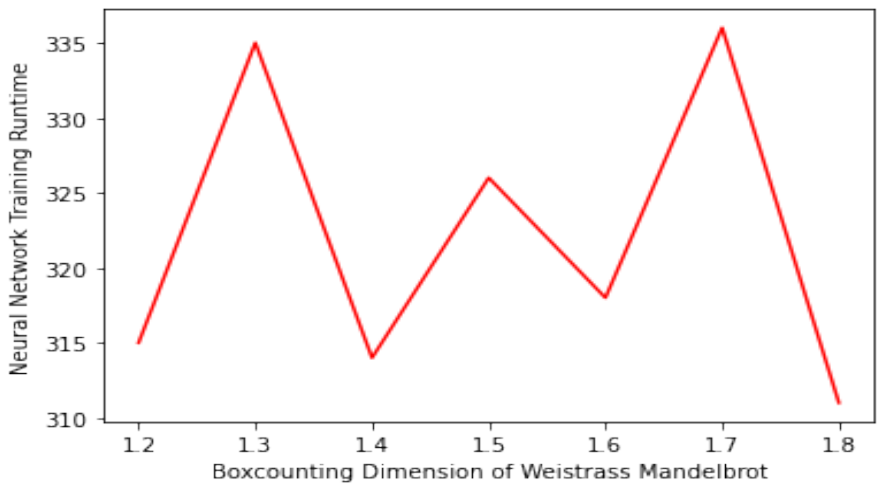
Experiment 1 Details

Neural Network training logistics

- Fractal dimensions represented: 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8
- For each dimension, the training dataset was defined $\{x_i = i/n : i \in [1, n]\}$, corresponding expected outputs were $W(x_i)$
- The test dataset was defined $\{x_j = (2j-1)j/2n : j \in [1, n]\}$, corresponding expected outputs were $W(x_j)$
- Thus the neural network was essentially being trained to interpolate between points in the time series

Experiment 1 Preliminary Results





Experiment 2 Details

- Henon Map:

$$x_{n+1} = 1 - ax_n^2 + y_n$$

$$y_{n+1} = bx_n$$

- a $= 1.4$ and b $= 0.3$ give strange attractor (Lyapunov exponents $\lambda_1 = 0.603$ and $\lambda_2 = -2.34$)
- Estimating box-counting dimension D_0 from D_1 using Kaplan Yorke ($D_1 = D_0$ because the Henon map is invertible [Young, '84]):

$$D_0 = D_1 = 1 + \frac{0.603}{2.43} \approx 1.26$$

- Network trained with degree 4 time delay embedding.
  Input: tuple of four consecutive time series values $\{x = (x_\tau, x_{\tau-1}, x_{\tau-2}, x_{\tau-3}) : \tau \in [4, n]\}$.
  Expected output: $x_{\tau+1}$, the next value of the time series.

Experiment 2 Background

Chaos and Fractal Dimension are closely related

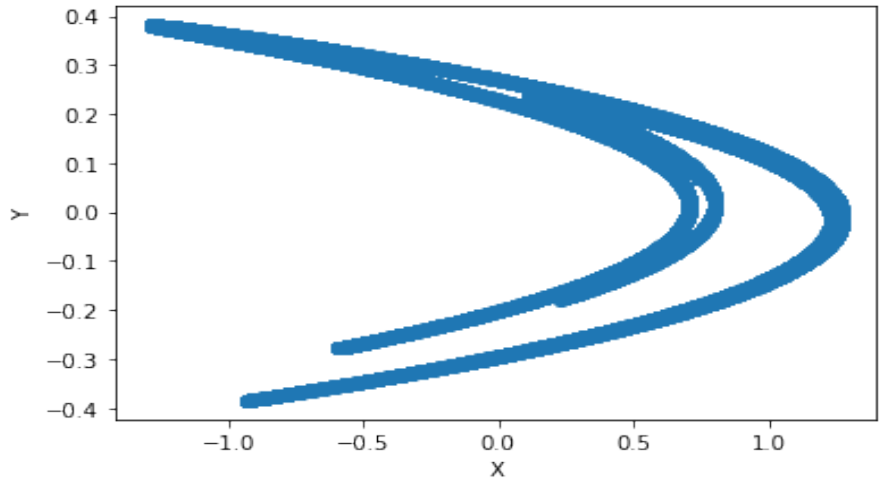
- Lyapunov exponents- divergence of initially close orbits in phase space (determinism but not predictability)
- How does dimension and chaos (quantified by Lyapunov exponents) relate?
- Kaplan-Yorke Conjecture (2D case, [Ledrappier-Young '88])

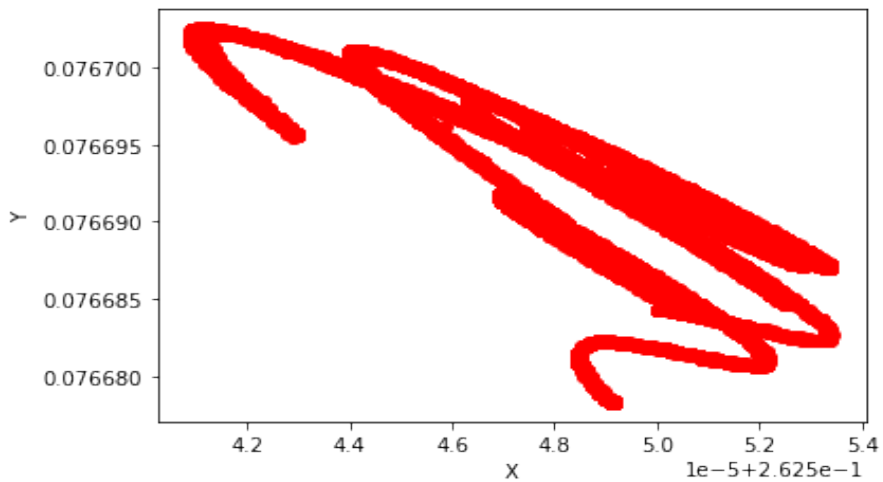
$$D_1 = 1 + \frac{\lambda_1}{|\lambda_2|}$$

(where D_1 is information dimension and satisfies $D_1 \leq D_0$)

- Empirically - how does learnability hold up in the chaotic case of a fractal dimension system?

Experiment 2 Preliminary Results





Future Work

- Ultimate Goal: Prove that a function f such that $\frac{1}{n^2}\sum_{j\neq j'}\left|f\left(\frac{j}{n}\right)-f\left(\frac{j'}{n}\right)\right|^{-s}\leq C$ independent of n for some $s\in[1,2]$ can be approximated with a neural network "on average"
- We will formalize "on average" - need some sort of alternative definition of learnability (pointwise convergence unlikely)
- Find a way to generate a function f with Hausdorff dimension s for any $s\in[1,2]$ (we're investigating the dimension on the Koch snowflake curve)

References

[Ledrappier-Young, 1988] Ledrappier, F. and Young, L. -S. *Dimension Formula for Random Transformations*, Comm. Math. Phys. 117, 529, (1988).

[Young, 1984] Young, L. S. *Dimension, Entropy, and Lyapunov Exponents in Differentiable Dynamical Systems.* Phys. A 124, 639-645, (1984).

[K. Falconer, 2014] Falconer, F. *Fractal Geometry: Mathematical Foundations and Applications. pg. 142, (2014).*

[A. Zaleski, 2012] Zaleski, A. Fractals and the Weierstrass-Mandelbrot Function. pg. 93-100, (2012).