ANALYSIS OF PSEUDO-RANDOM NUMBER GENERATORS (PRNGs)

THROUGH PREDICTIVE MODELING

Priyanka Adhikari, Ruchira Banerjee, Nidhi Bendre, Simone Kaplunov, Melina Taranto

SNOWBALL
Organized by DATA Club

INTRODUCTION

Problem

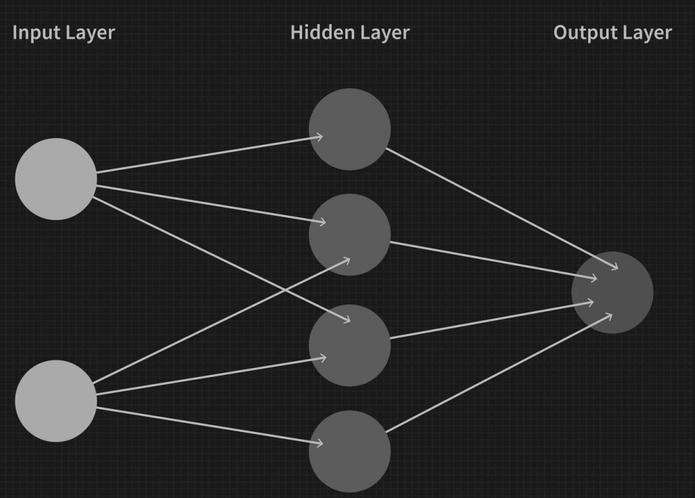
- **PRNGs** are algorithms designed to generate random sequences of numbers
  - Lack true randomness
- **Cryptography** issue: security experts are now resorting to physical sources of “randomness”
  - Example: Cloudflare currently uses a wall of lava lamps for data encryption

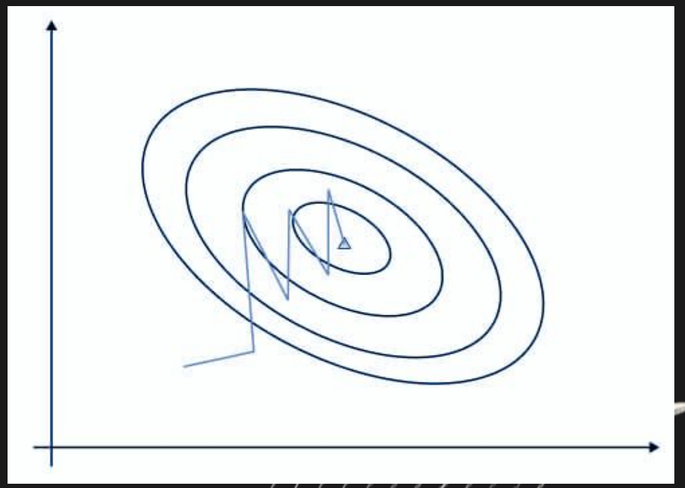
Our Goal

- Create a model to predict the output of a PRNG without giving the model any information about the PRNG algorithm
- Question: Given the last 4 generated numbers, can a neural network predict the next randomly generated number?
- How:
  - Train model on previously generated random numbers
  - Have model predict the next number
- Model of Choice:
  - Feedforward Neural Network

NEURAL NETWORKS

- Built of interconnected nodes (neurons) which are organized into layers
- **Activation function:** introduces non-linearity, making a network more intricate than standard linear regression
- **Loss (error) function:** measures the difference between the predicted output and actual output
- **Optimizer:** algorithm used to adjust the parameters of a neural network in order to minimize the loss
  - **Stochastic Gradient Descent (SGD)**





"What Is a Neural Network? | Types of Neural Networks." Cloudflare, Inc.

METHODOLOGY

- Selected PRNG: **XORShift 128**
- Dataset: Generated **~10,000** observations from algorithm
- Split data into **training/testing** datasets
- Loss functions: **BCELoss**, **L1 Loss**, **Cross Entropy Loss, Smooth L1 Loss**
- Optimization Algorithms: **Stochastic Gradient Descent (SGD)**, **ADAM**
- Framework used: **Pytorch**

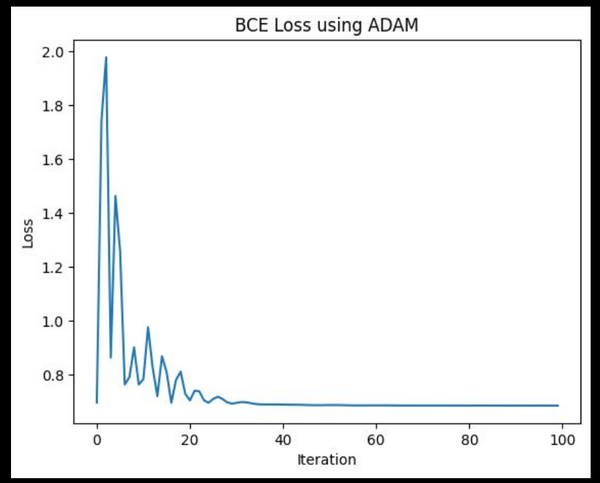
```
import torch
import torch.nn as nn

model = nn.Sequential(
    nn.Linear(in_features=128, out_features=1024),
    nn.Linear(in_features=1024, out_features=32),
    nn.Sigmoid()
)
```

Hassan, Mostaffa. "Cracking Random Number Generators Using Machine Learning – Part 1: Xorshift128." NCC Group Research Blog, 5 Apr. 2023.

RESULTS

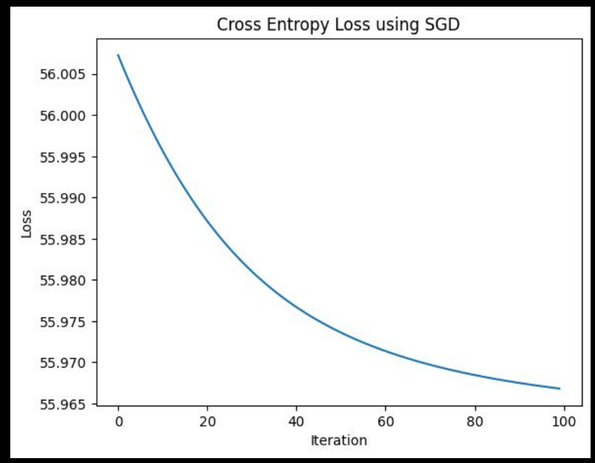
BCE Loss

- **Optimizer:** ADAM
- **Accuracy:** 0.5448



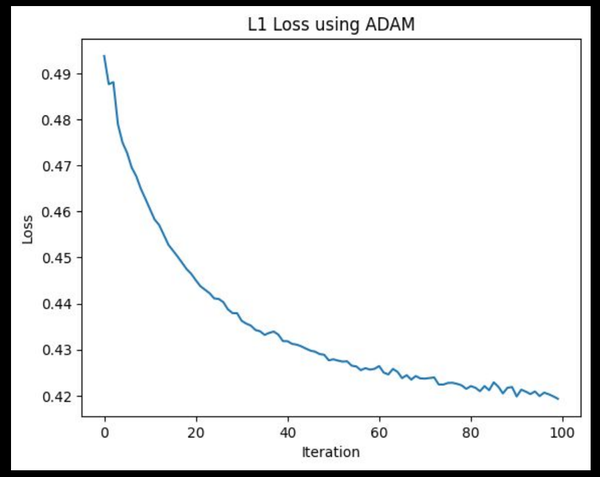
Cross Entropy Loss

- **Optimizer:** SGD
- **Accuracy:** 0.5886



L1 Loss

- **Optimizer:** ADAM
- **Accuracy:** 0.5848



Smooth L1 Loss

- **Not effective**
- **Excluded**

FUTURE DEVELOPMENT

- Optimize current model for accuracy
  - Refine neural network structure
  - Hyperparameter tuning
- Scale data
  - Train & test on larger datasets
- Add TransformerEncoderLayer
  - Incorporation of transformer
- Extend methodology to other PRNGs for comparison
  - Diverse PRNG selection
  - Cross validation across PRNGs

THANK YOU!

SNOWBALL
Organized by DATA Club