Blockchain and Artificial Intelligence
Artificially blocked and chained Intelligencę

Vugranam (VC) Sreedhar
Global Blockchain Leader
IBM TSS/GTS
Vugranam@us.ibm.com
A blockchain computing model consists of:

* A sequence or ordered set of transactions
* Submitted by one or more identities
* Over one or more **channels**
* That satisfy some constraints or logic expressed as smart contracts
* Agreed upon using some distributed consensus approach by one or more **consensus** users

So that the sequences of transactions can be committed or recorded on a shared ledger

* And once the sequence of transactions are committed
* The cannot be easily modified or tampered
Why distributed system is hard?

1. Asynchrony
2. Local knowledge
3. Failure

Shared ledger = Global state or knowledge

- Perform action using local knowledge
- Compute function using local knowledge
- Finite state machine replication
- Solve crypto game or puzzle
- Mining
- Smart contract
BLOCKCHAIN MODEL

- WHICH BLOCK?
- DEMOCRACY VS FASCISM?
- HOW TO HANDLE FAILURE?

AGREED FINAL = GLOBAL STATE

\\( (m+n)! \over m! \cdot n! \)
**Consensus as Computation**

\[ y = f(x) = \begin{cases} \text{agree/accept} & \text{every node performs/computes } f(x) \\ \text{disagree/reject} & \text{solves a puzzle} \end{cases} \]
COMPUTATION MODEL

\[ X_1 = \begin{array}{cccc}
    s_1 & s_2 & t_1 & t_2 & t_3 \\
\end{array} \]

\[ f(x_1) \]

\[ y = f(x_1) \]

ACCEPT \hspace{1cm} REJECT

LEGAL SEQUENCE

SOME NODES CAN FAIL

ILLEGAL SEQUENCE
MATRICES AND TENSORS

**Matrix A is a Linear Transformation**

\[ y = Ax \]

**Matrix Decomposition or Factorization**

\[ A = UV \]

**Recommendation System**

**Curse of Dimension**

**Many ML = Matrix Decomposition**

**Many Data Can Be Flattened**

**Data**

\[ \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \]

\[ \begin{bmatrix} e_1 & e_2 & e_3 & \ldots & 4 \end{bmatrix} \text{ Features} \]
TENSORS

\[ T = \text{TENSOR} = (t_{ijk}) \]

MULTILINEAR TRANSFORMATION

\[ U \otimes V \rightarrow W \]
\[ U \otimes W \rightarrow V \]

Many "nice properties" do not generalize to tensor world

\[ \begin{array}{c}
    \text{VECTOR} \\
    i \\
    j \\
    k \\
\end{array} \]

\[ \begin{array}{c}
    \text{TENSOR} \\
    i \\
    j \\
    k \\
\end{array} \]

\[ \begin{array}{c}
    \text{MATRICE} \\
    \times \\
    \rightarrow \\
\end{array} \]

\[ \rightarrow \\
\]
Belief/Bayesian Network

\[ f = P(y|x) \]
\[ g = P(z|x) \]
\[ h = P(w|z,y) \]

Naive Bayes

Conditional Distribution

Hidden

Generated
DEEP LEARNING 101

There are only 3 things in DL:

1. INPUT
2. WEIGHTS
3. OUTPUT

Problem:
Given any two of the three, find the third.

\[ Y = F(X) \]
THREE THINGS

1. GIVEN INPUT + WEIGHTS FIND OUTPUT → PREDICTION / DISCRIMINATIVE

2. GIVEN INPUT + OUTPUT FIND WEIGHTS → LEARNING

3. GIVEN OUTPUT + WEIGHTS FIND INPUT → GENERATIVE
LEARNING AS COMPUTATION

\[ Y = f(x) \]

Learning \equiv\text{ estimating } f \text{ as } \tilde{f} \text{ or approximating } f \text{ is regular or rational or no jumps }

RIDGE FUNCTION

\[ Y = P(xw_n + b_n) \]

\[ \tilde{f}_M(x) = \sum_{n=1}^{M} \alpha_n P(w_n x + b_n) \]

\[ \lim_{M \to \infty} \|f - f_M\| = 0 \]
Piecwise Linear Approximation

\[ \tilde{f}(x) = \sum_{n} a_n P_w(n \epsilon) \]

Sampling Distance \( \epsilon \): \( |f(x) - \tilde{f}(x)| \leq C \epsilon \)

Polynomial app
Consensus as Learning

\[ y = f(x) \]

Sequence of Transactions

Accept - Legal

Or

Reject - Not Legal

Sequence

Can we use Convolution Neural Network?
Recurrent Neural Network

\( X_1 = S_1 \ldots S_n \)
\( X_2 = t_1 \ldots t_m \)

\( \text{LSTM} = \text{LONG SHORT TERM MEMORY} \)

\# OF POSSIBLE LEGAL MERGE

\[ \frac{(n+m)!}{n! \cdot m!} \]

Calculating Weights = Consensus
RECURRENT NEURAL NETWORK

RNN DYNAMIC STATE EVOLUTION DEPENDS ON INPUT + WHAT IT DID IN THE PAST (INCLUDING CURRENT STATE)

\[ Y(t+1) = Y(t) + X(t) \]

RNN = HMM + DISTRIBUTED STATE

\[ Y(t+1) = Y(t) + g(\alpha) X(t) \]

RNN CAN EASILY HANDLE VARIABLE LENGTH STRUCTURES
Recurrent Neural Net

RNN are Turing Complete \(\Rightarrow\) Simulate Arbitrary Algorithmic Task

Turing Machine = FSM + Infinite Tape

Turing NN = RNN + Addressable Memory

\(\Rightarrow\) Differentiable Computers

\(\Rightarrow\) Easy to Train
Differentiable Consensus

- No Idea 😇
- But consensus are learning procedures
- Jury Duty
- Creating PPT for CEO
- Exploring Algebraic Topology and Geometry
- Why Mathematics is important??
- Experimental Research + Mathematical Research

World Class Developers

Differentiable Geometry