SmartInsure: Blockchain and CNN Leveraged Secure and Efficient Cattle Insurance

Presenter: Sukrutha L. T. Vangipuram

Anand Kumar Bapatla¹, Aishani Gupta², S. P. Mohanty³, E. Kougianos⁴
University of North Texas, Denton, TX, USA.¹,²,³,⁴

Email: ab0841@unt.edu¹, aishanigupta@my.unt.edu², saraju.mohanty@unt.edu³, elias.kougianos@unt.edu⁴
Outline

- Introduction
- Significance of Cattle Insurance
- Centralized Insurance Management System
- Problems with Current IMS
- Novel Contributions
- Architectural Overview
- Implementation Details
- Results and Analysis
- Conclusion
Agriculture and Role of Insurance

• Agriculture: Crop cultivation + Livestock farming.

• Serves as the backbone for several country’s economies.

• Depends on unpredictable weather and climatic events, making mitigation techniques futile.

• Livestock Farming is heavily damaged by viral diseases like Bovine Respiratory Diseases (BVD), and Foot and Mouth Diseases (FMD).

• Livestock Insurance: A special type of insurance within agriculture for cattle to ensure farmers are financially protected.
Cattle Insurance

- Some viral diseases and symptoms:
  - Bovine Respiratory Diseases (BRD)
    - Coughing
    - Lack of appetite
    - Rapid and shallow breathing
    - Fever
  - Foot and Mouth Diseases (FMD)
    - Fever
    - Depression
    - Weight loss
    - Appetite loss
- Measures: Quarantine and Trade restrictions
- Farmers cannot sell produce
  - Struggle to make a profit
  - Struggle to sustain
Cattle Insurance Management System

- Entities in cattle insurance management system
  - Owners/Farmers who own the cattle and seek to insure their livestock
  - Insurance companies responsible for underwriting the policies, setting the premiums, and processing the insurance claims.
  - Insurance agents act as intermediaries who help the farmer with choosing appropriate policy.
  - Regulatory agencies to overlook operations and ensure regulations.
  - Re-insurance providers where the insurance companies insure part of their cattle insurance portfolio.
Cattle Insurance Management System

- Insurance Management System (IMS)
- Policy Management System
- Claim Processing System
- Customer Relationship Management (CRM)
- Data Analytics and Reporting
- Security and Compliance
- Billing and Payments

Cattle Owners/Farmers

- Applies for Insurance

Insurance Agents

- Choosing Policies
- Policy Information

Insurance Company

- Underwrites Policies

Livestock Inspectors

- Risk/Damage Analysis

Regulatory Authorities

- Oversight

OCIT 2023 - SmartInsure
Problems with Centralized Cattle IMS

- Centralized architectures are more prone to security threats.

- Unauthorized modification of insurance records and falsified claims.

- Significant delay in claim processing affecting the cattle owners/farmers.

- Identification of cattle using RFID tags which can be detached for falsifying information.

- Overhead costs for coordinating distributed stakeholders.

- More prone to disputes due to cumbersome paperwork involved.
Novel Solutions Proposed

- **Blockchain features** in the proposed SmartInsure help prevent many of the security threats faced by the centralized architecture.
- The **Immutability** nature of the distributed ledger ensures no modifications can be done to the transactions once they are confirmed.
- The proposed blockchain-based architecture helps create a transparent environment to increase the efficiency of insurance management functions.
- **Muzzle images** are used to identify the insured cattle, preventing falsified claims.
- Proposed SmartInsure **reduces paperwork**, which can be cumbersome and lead to many disputes, and overhead costs.
Five Components of the proposed SmartInsure

- **Distributed entities:** Includes stakeholders like Cattle owners/farmers, Insurance agents, Insurance companies, Livestock inspectors, regulatory agencies, etc.

- **On-chain component:** Smart Contracts holding the business logic and access control mechanisms.

- **Off-chain component:** Decentralized file storage system to store large muzzle image data of insured cattle.

- **Image verification service:** AI-based image identification of insured cattle to prevent falsified claims.

- **User Interface:** user-friendly interface for different stakeholders to perform insurance functions.
Architectural Overview

Participating Distributed Entities
- Cattle Owners/Farmers
- Insurance Agents
- Insurance Company
- Livestock Inspectors
- Regulatory Authorities

Entities Perform Different Actions Through DApp Interface

Decentralized Application (DApp)
- Infura
- Web3.js
- HTML, CSS

Invoking Different Smart Contract Functions

Smart Contracts

Ethereum Virtual Machine

Ethereum Blockchain

Continuous Training AI/ML model

Verify Images from off-chain Storage

Send Muzzle Images

AI/ML model

Insured Cattle Identification

Upload Muzzle Images of Insured cattle

Return CID

Return Image File

Query Muzzle Image

Distributed Data Storage

Images of Insured cattle
Implementation

- Dataset consists of 4923 images from 268 different feed yard breed cattle
- Cattle breeds:
  - Angus
  - Angus Hereford Cross
  - Continental British Cross
- Images taken with a 26MP camera
- Images resized to 180 * 180 pixels

Implementation

- Smart contract design:
  - Solidity language used for implementing smart contracts in the Ethereum platform
  - Role Based Access Control (RBAC) Mechanism is employed using smart contracts.
  - Each stakeholder will be assigned to a role and modifiers are defined to control the access to different insurance functions
  - Factory contract design pattern is used for creating and deploying cattle insurance policy smart contracts.
Implementation

- Blockchain Network:
  - Local development environment *Ganache*.
  - 10 free accounts with 100 test ETH.
  - Truffle development suite to design the decentralized App.
  - Testing is done using the node assertion library *Chai*.

- Distributed Data Storage:
  - Inter-Planetary File System (IPFS) I used for storage of cattle muzzle images.
  - Infura API to interact with the off-chain storage.
## Implementation

- **CNN Architectural details:**

<table>
<thead>
<tr>
<th>Layer</th>
<th>Type</th>
<th>Filters</th>
<th>Output Shape</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Rescaling</td>
<td>-</td>
<td>(180,180,3)</td>
</tr>
<tr>
<td>2</td>
<td>Conv2D</td>
<td>16</td>
<td>(180,180,16)</td>
</tr>
<tr>
<td>3</td>
<td>MaxPooling2D</td>
<td>-</td>
<td>(90,90,16)</td>
</tr>
<tr>
<td>4</td>
<td>Conv2D</td>
<td>32</td>
<td>(90,90,32)</td>
</tr>
<tr>
<td>5</td>
<td>MaxPooling2D</td>
<td>-</td>
<td>(45,45,32)</td>
</tr>
<tr>
<td>6</td>
<td>Conv2D</td>
<td>64</td>
<td>(45,45,64)</td>
</tr>
<tr>
<td>7</td>
<td>MaxPooling2D</td>
<td>-</td>
<td>(22,22,64)</td>
</tr>
<tr>
<td>9</td>
<td>Dense</td>
<td>128</td>
<td>(128,)</td>
</tr>
<tr>
<td>10</td>
<td>Dense</td>
<td>268</td>
<td>(268,)</td>
</tr>
</tbody>
</table>
Results

- Functional Validation
  - Test cases are designed for maximum code coverage using chai assertions.
  - Along with insurance functions, access control is also tested.

```javascript
// compiling your contracts...
===============
Consider:
Compiling /contracts/CattleInsurance.sol
Compiling /contracts/FarmerRole.sol
Compiling /contracts/InsuranceProviderRole.sol
Compiling /contracts/Migrations.sol
Compiling /contracts/Unmalleable.sol
Compiling /contracts/Regulatory.sol
Compiling /contracts/Role.sol
Artifacts written to C:\Users\anand\AppData\Local\Temp\test--20712-BMK2TYK8pRX
Compiled successfully using:
- solc: 0.8.11+commit.d7ff03943.Emscripten clang

Contract: CattleInsurance
✓ should allow insurance provider to register a farmer
✓ should allow a farmer to sign a contract with a single IPFS hash
✓ should allow a farmer to sign a contract with multiple IPFS hashes
✓ should allow a farmer to update cattle image hashes
✓ should allow a farmer to file a claim
✓ should allow insurance provider to approve a claim
✓ should allow anyone to view a contract
✓ should allow anyone to view a claim
```
Results

- CNN model performance is measured using validation accuracy and loss metrics.
- Let the number of validation samples be given as $N_{\text{correct}}$ and the total number of samples $N_{\text{val}}$. The Validation Accuracy can be calculated as follows:

$$\text{Validation Accuracy} = \frac{N_{\text{correct}}}{N_{\text{val}}}$$

- For the validation dataset of size $N_{\text{val}}$, $y_{\text{true}}$ represents the true target value, $y_{\text{pred}}$ predicted the target value and $L(y_{\text{true}}, y_{\text{pred}})$ is the loss function computed as:

$$\text{Validation Loss} = \frac{\sum_{i=1}^{N_{\text{val}}} L(y_{\text{true}i}, y_{\text{pred}i})}{N_{\text{val}}}$$
Results

Training Accuracy: 99.1%
Validation Accuracy: 94.11%

Training Loss: 0.04
Validation Loss: 0.38
Conclusions

- A **Novel cattle insurance management system** SmartInsure is proposed to solve the security and latency of IMS.
- Deep learning-based approach for **cattle identification using muzzle images** to avoid falsified insurance claims.
- **Role-Based Access Control Mechanism** is implemented for manager role-specific functions.
- Implemented DApp is **tested for functionality**.
- The deep learning model is analyzed with **validation accuracy and loss metrics**.
- Performance of the implemented CNN model is measured using validation accuracy and loss with acceptable values of **94.11% and 0.38**.
Future Work

- More complex interactions will be included in the designed Dapp.
- More tamper-proof identification mechanisms will be explored instead of muzzle images.
- A responsive user-friendly interface with multiple views with role-specific functions.
Thank You !!