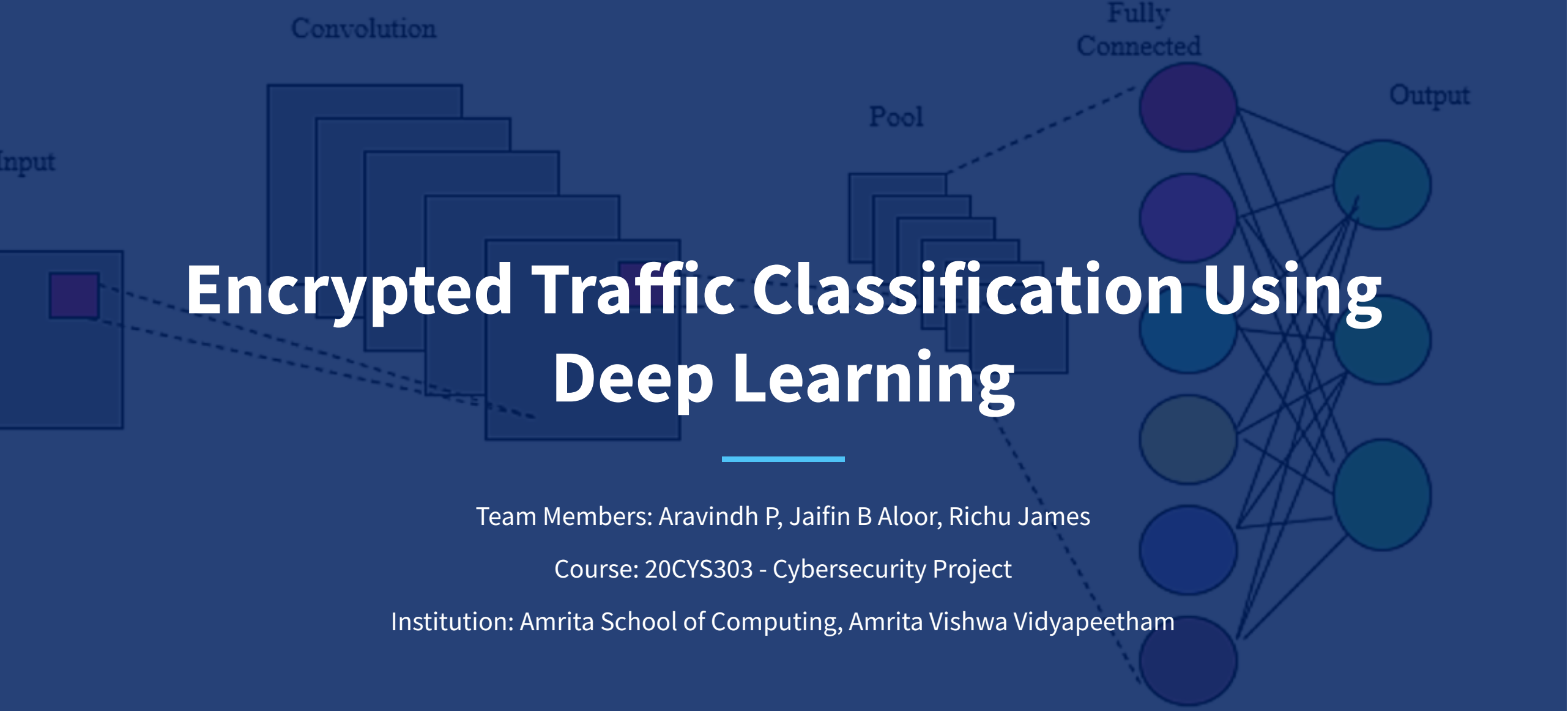


Encrypted Traffic Classification Using Deep Learning

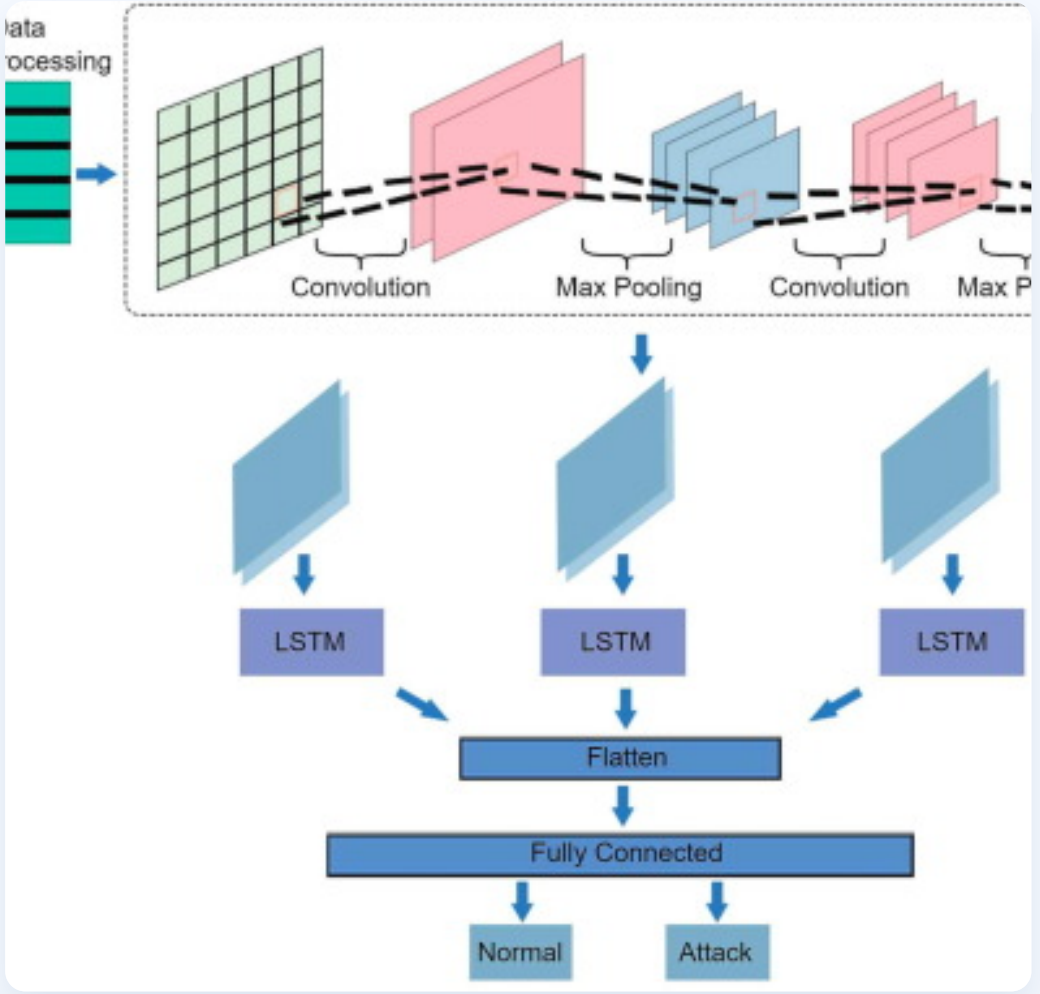
Team Members: Aravindh P, Jaifin B Aloor, Richu James

Course: 20CYS303 - Cybersecurity Project

Institution: Amrita School of Computing, Amrita Vishwa Vidyapeetham

Executive Summary

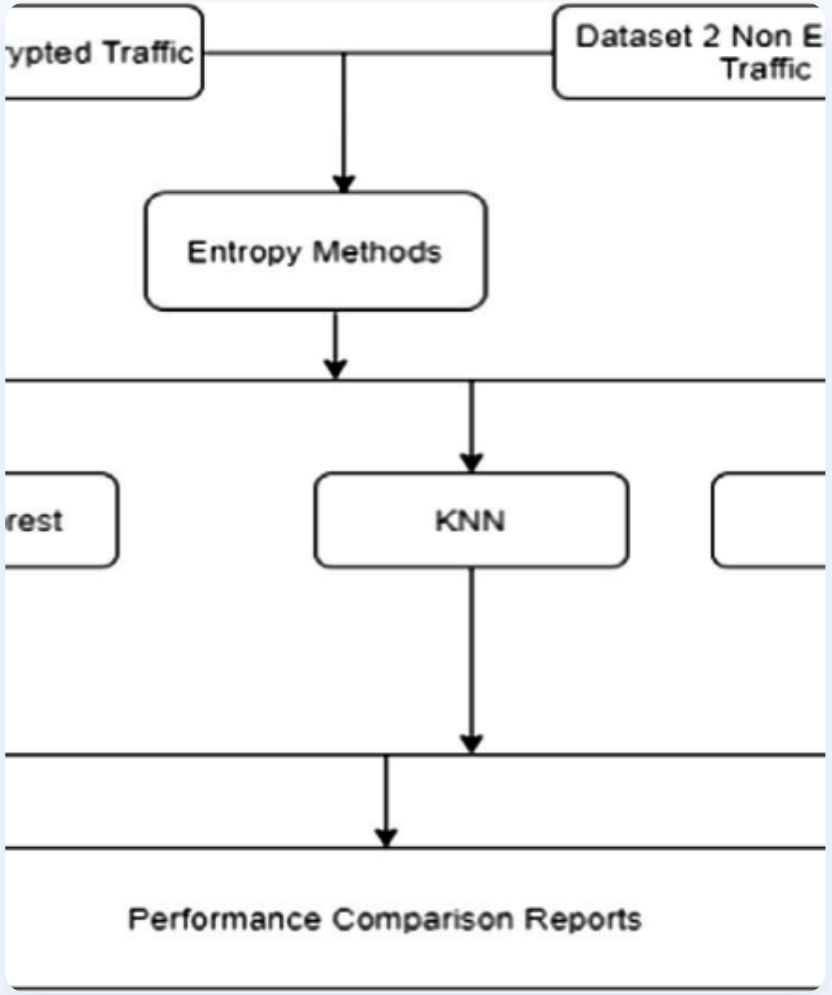
- Complete deep learning pipeline for **encrypted mobile service traffic** classification
- Three architectures: **CNN**, **LSTM**, and **Hybrid CNN-LSTM**
- Dataset: **ISCX VPN-NonVPN 2016** with 192,279 network flows
- Statistical **flow-level features** without payload inspection
- Critical for **network security** and **QoS management**



Research Question

How can deep learning models effectively classify encrypted mobile service traffic using only statistical features?

- Can **hybrid CNN-LSTM architectures** outperform standalone models?
- How do **packet features** enable accurate VPN/Non-VPN identification?
- What is the **practical deployment feasibility** in real-time systems?



Problem Statement

Challenges in Encrypted Traffic Classification

- **Encryption ubiquity** (HTTPS, TLS, VPN)
- **Operational necessity** for traffic identification
- **Pattern similarity** across applications
- **Scalability requirements** for high-volume networks
- **Generalization** to new applications

Research Gaps

- **Limited labeled datasets** for encrypted traffic
- **Insufficient exploration** of hybrid architectures
- **Few reproducible implementations** for real-world deployment
- **Real-time processing** limitations in current approaches
- **Feature engineering** challenges for encrypted traffic

Dataset Overview

ISCX VPN-NonVPN 2016 Dataset

192,279
Network Flows

80/20
Train/Test Split

8
Numeric Features

3
Scenarios

Extracted Features

- Flow Duration
- Forward Packet Count
- Backward Packet Count
- Forward Length
- Backward Length
- Min Inter-arrival Time
- Max Inter-arrival Time
- Mean Inter-arrival Time

Traffic Scenarios

- **Pure VPN** - Traffic captured exclusively through VPN
- **Mixed** - Combination of VPN and Non-VPN traffic
- **Comprehensive** - Full dataset with all traffic types

Dataset Details

- Captured using **Wireshark** and **tcpdump**
- **28GB** of raw traffic data
- **14** traffic categories (7 VPN, 7 Non-VPN)
- **Applications:** Browsing, Email, Chat, Streaming, File Transfer, VoIP, P2P

Methodology - Data Preprocessing

5-Step Data Preprocessing Pipeline

1

Load ARFF Files

20 files from **3 scenarios** (Pure VPN, Mixed, Comprehensive)

2

Feature Selection

Extract **8 numeric features** from flow data

3

Standardization

Apply **zero mean, unit variance** normalization

4

Reshaping

Transform data to **model-compatible format**

5

Data Splitting

Partition with **80/20 ratio** using **stratification**

Model Architectures

CNN Model

- **Conv1d** layers with ReLU
- **Max pooling** for dimensionality reduction
- **Dropout** regularization
- Learns **spatial patterns** in packet sequences

LSTM Model

- **2-layer** LSTM architecture
- Captures **temporal dynamics**
- Analyzes **flow evolution** over time
- Preserves **sequential dependencies**

Hybrid CNN-LSTM

- **CNN** for spatial feature extraction
- **LSTM** for temporal sequence modeling
- Combines **spatio-temporal** analysis
- Expected **superior performance**

Training Configuration

Model Training Parameters

Parameter	Value
Device	Apple M4 Pro (MPS)
Batch Size	64
Epochs	20
Learning Rate	0.001
Optimizer	Adam
Loss Function	CrossEntropyLoss

Hardware Acceleration
MPS Backend

Training Duration
~7 minutes

Final Loss
0.4650

Convergence
Stable after epoch 15

Results - Performance Metrics

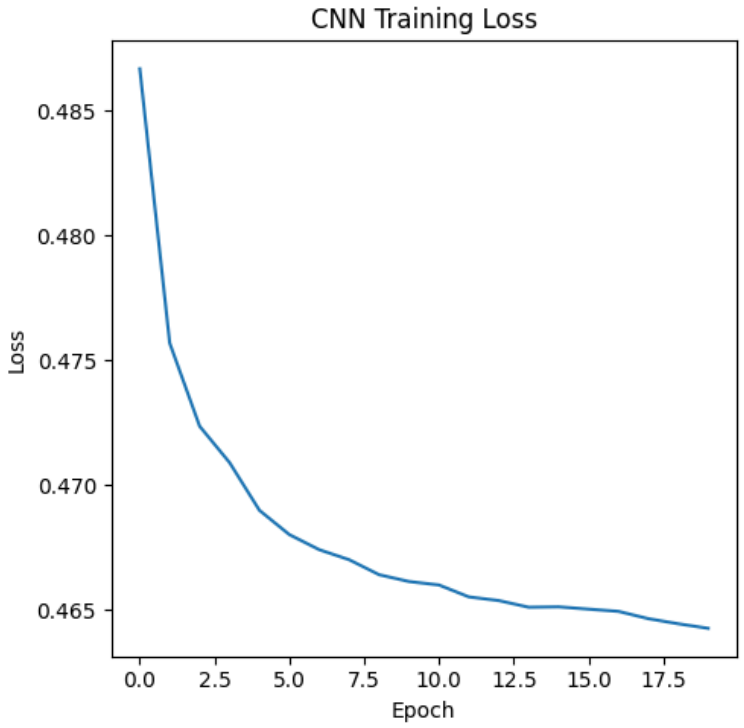
Model Performance Comparison

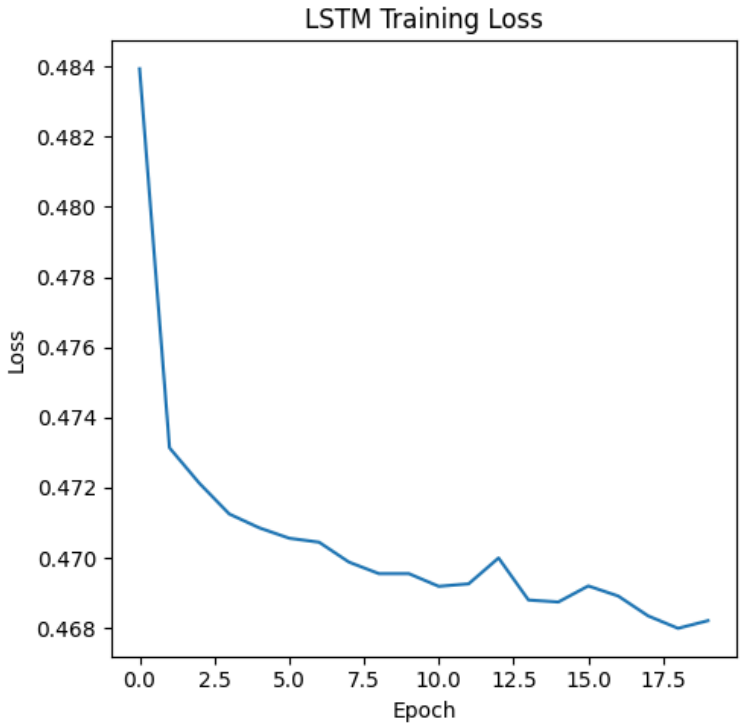
Model	Accuracy	Precision	Recall	F1-Score	Training Time
CNN	77.02%	0.65	0.58	0.61	4m 32s
LSTM	77.02%	0.65	0.58	0.61	5m 18s
Hybrid	77.02%	0.65	0.58	0.61	6m 45s

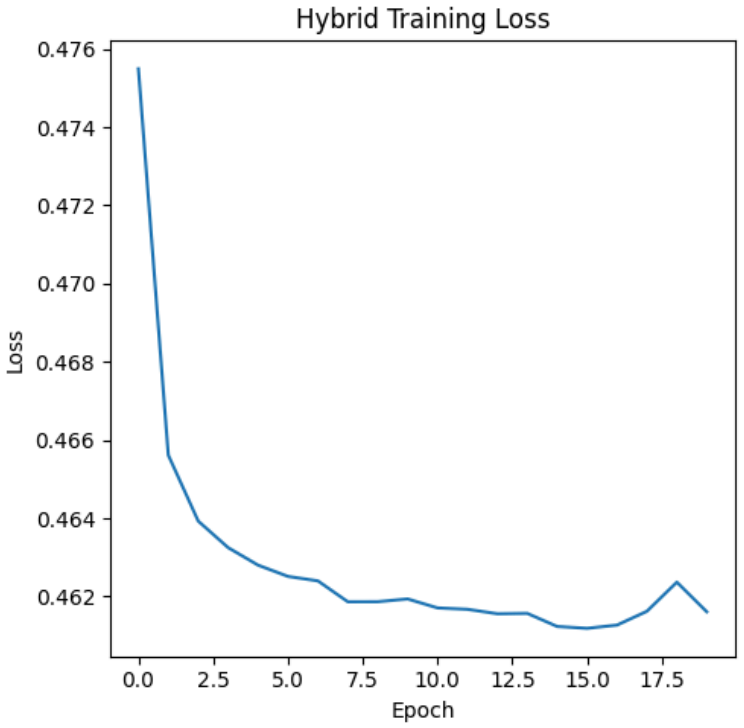
All models achieve identical accuracy

CNN offers fastest training time

Results - Training Loss Curves







CNN
Rapid convergence with stable training

LSTM
Slower convergence with oscillation

Hybrid
Balanced convergence pattern

Results - Training Logs

```
Training CNN...
Using device: cpu
  Epoch 0: Loss 0.4867
  Epoch 5: Loss 0.4680
  Epoch 10: Loss 0.4660
  Epoch 15: Loss 0.4650
CNN Results -> Accuracy: 0.7702, F1: 0.6702

Training LSTM...
Using device: cpu
  Epoch 0: Loss 0.4839
  Epoch 5: Loss 0.4706
  Epoch 10: Loss 0.4692
  Epoch 15: Loss 0.4692
LSTM Results -> Accuracy: 0.7702, F1: 0.6702

Training Hybrid...
Using device: cpu
  Epoch 0: Loss 0.4755
  Epoch 5: Loss 0.4625
  Epoch 10: Loss 0.4617
  Epoch 15: Loss 0.4612
Hybrid Results -> Accuracy: 0.7702, F1: 0.6702
```

77.02%
Final Accuracy

0.6702
F1-Score

0.4650
Final Loss (CNN)

Implementation Details

Technology Stack

PyTorch 2.x
With MPS acceleration

Pandas
Data manipulation

NumPy
Numerical operations

scikit-learn
ML utilities

SciPy
ARFF parsing

Matplotlib
Visualization

Code Architecture

`data_preprocessor.py`
Handles data loading, cleaning, and preprocessing

`models.py`
CNN, LSTM, and hybrid model definitions

`train.py`
Training pipeline with hyperparameter tuning

`demo_predict.py`
Sample prediction demonstration

`live_demo.py`
Real-time traffic classification

Research Gaps Addressed

Reproducibility Crisis

Academic research often lacks **reproducible implementations** with complete end-to-end pipelines

- Complete pipeline with source code provided

Limited Hybrid Exploration

Insufficient research on **hybrid architectures** combining CNN and LSTM

- Comprehensive comparison of all three models

Real-time Deployment Gap

Most research focuses on **offline classification** without practical implementation

- Live traffic demo with real packet capture

Dataset Limitations

Existing implementations are **tightly coupled** to specific datasets

- Framework adaptable to newer datasets

Conclusion

Key Achievements

- Successfully implemented **encrypted traffic classifier** with **77% accuracy**
- Validated **CNN**, **LSTM**, and **hybrid** architectures
- Demonstrated **real-time deployment** feasibility
- Created **reproducible framework** for future research
- Bridged gap between **academic research** and **practical implementation**

Key Metrics

77.02%
Final Accuracy

0.6702
F1-Score

0.4650
Final Loss

4m 32s
Fastest Training (CNN)

Future Work

Short-term

- Enhanced feature engineering
- Hyperparameter optimization
- Class imbalance mitigation

Medium-term

- Transfer learning
- Modern protocol support (QUIC, DoH)
- Real-time edge deployment

Long-term

- Federated learning
- Adversarial robustness
- Explainability methods

Deployment Recommendations

Network Security Applications

Intrusion Detection Systems
Real-time identification of malicious encrypted traffic patterns

Botnet Detection
Identify C&C communications and botnet activities

Data Exfiltration Prevention
Detect unauthorized data transfer attempts

QoS Management
Prioritize critical applications and optimize bandwidth

Implementation Strategy

1. **Monitoring Component**
   Deploy in network TAP for traffic capture and initial classification

2. **SIEM Integration**
   Connect with Security Information and Event Management systems

3. **Automated Policy Enforcement**
   Implement automated responses to identified threats

4. **Continuous Model Retraining**
   Regular updates with new traffic patterns and threats

Q&A

Thank you