ISSN 2277-2685
IJESR/April-June. 2024/ Vol-14/Issue-2/1799-1809
Dr Sunil Vijayakumar *et. al.*, /International Journal of Engineering & Science Research

A Machine Learning Approach for Decoding Criminal Cryptography

Dr Sunil Vijayakumar[1], Ms Shaik Zubair[2]
[1]Professor, Department of CSE, Lords Institute of Engineering and Technology, Hyderabad, India
[2]Assistant Professor, Department of CSE, Lords Institute of Engineering and Technology, Hyderabad, India.
Mail Id: sunilvkumar@lords.ac.in[1], szubair@lords.ac.in[2]

Abstract

*Criminal communications increasingly rely on custom, obfuscated or hybrid encryption schemes to evade detection, and manual cipher identification by forensic analysts does not scale to the volume of intercepted digital evidence in modern investigations. Existing cryptanalysis tools are largely limited to classical substitution or transposition ciphers and offer little support for identifying the varied cipher types encountered in real casework. This paper presents **Cipher Scan**, a machine-learning framework that automatically classifies the type of cipher used in a suspect ciphertext and assists investigators in decoding it. A triage stage first separates plaintext, encoded, compressed and encrypted data. A 72-dimensional statistical feature vector — including entropy, index of coincidence, chi-square, periodic and n-gram statistics, and byte-level structural features — is combined with a character-level CNN–BiGRU–attention network through weighted late fusion, and an open-set rejection step flags unknown or custom schemes rather than forcing a wrong label. For supported classical and custom ciphers, a decoding-assistance engine applies key-space search, period estimation and simulated-annealing hill-climbing guided by quadgram fitness; for strong modern ciphers, it reports structural properties only. Every case produces a forensic report with evidence hashes and confidence scores. On a dataset of 15 classical and hybrid cipher types, the hybrid model is expected to achieve 90.4% accuracy, rising to 98.5% for 500-character ciphertexts, with decoding success above 90% for most supported ciphers*

Keywords: *Cryptanalysis; Cipher identification; Digital forensics; Machine learning; Index of coincidence; BiGRU; Feature fusion; Open-set recognition; Simulated annealing; Law enforcement.*

1. Introduction

Encryption protects legitimate communication, but criminals also use it to hide evidence of fraud, trafficking, extortion and other offences. Investigators routinely seize devices and intercept messages containing text that is clearly encoded or encrypted but whose method is unknown. Before any analysis can begin, the analyst must answer a basic question: *what kind of cipher is this?* Answering it manually requires specialist knowledge and time, and the growing volume of digital evidence makes manual identification a bottleneck in forensic workflows [21].

Not all encrypted evidence is equally hard. Strong modern algorithms such as AES [24] cannot be broken by statistical analysis when used correctly, but criminals frequently use weaker schemes: classical ciphers, home-made substitutions, repeating-key XOR, or combinations of simple methods layered together. These schemes leave statistical traces that machine learning can detect and that classical cryptanalysis can often break. Even for

strong ciphers, identifying structural properties — block size, mode of operation, or the fact that data are encrypted rather than compressed — helps investigators decide where to direct effort.

Machine-learning cipher identification has advanced considerably. Leierzopf et al. [1] classified 56 classical cipher types using engineered statistical features, Ahmadzadeh et al. [2] used a deep BiLSTM–GRU network, Xia, Li and Chen [3] identified modern algorithms with residual networks, and De Gaspari et al. [4] distinguished encrypted from compressed data. However, three gaps remain: (i) limited generalisation across diverse cipher families, including custom and previously unseen schemes; (ii) weak performance on short, noisy ciphertexts where statistical features are unreliable; and (iii) limited integration of identification with decoding assistance and confidence reporting. This paper addresses these gaps with the following contributions:

- A **four-way triage stage** that separates plaintext, encoded (Base64/hex), compressed and encrypted data before cipher classification, so that effort is spent only on genuinely encrypted samples.
- A **hybrid classifier** that fuses a 72-dimensional statistical feature vector, modelled by gradient-boosted trees, with a character-level CNN–BiGRU–attention network, improving accuracy most on short ciphertexts.
- An **open-set rejection** mechanism combining softmax confidence and Mahalanobis distance, which flags unknown or custom ciphers instead of assigning a wrong label.
- An integrated **decoding-assistance engine** and **forensic report generator** that apply the appropriate cryptanalytic method for each identified cipher and record evidence hashes, method and confidence for chain-of-custody purposes.

Section 2 reviews related work, Section 3 presents the methodology, Section 4 the experimental set-up, Section 5 the results, and Section 6 concludes the paper.

2. Related Work

Classical cipher-type detection. Leierzopf et al. [1] developed a large-scale machine-learning approach for identifying 56 classical cipher types using statistical feature engineering, evaluating Random Forest, Naïve Bayes and neural network classifiers with a best accuracy of 80.24%. Earlier, Kopal [23] showed that neural networks can detect cipher types from statistical features. These studies establish that character frequencies, index of coincidence and ciphertext patterns are effective features, and they form the foundation of the feature stream in Cipher Scan.

Deep sequence models. Ahmadzadeh et al. [2] proposed a deep network combining bidirectional LSTM and GRU layers for automated ciphertext classification, learning patterns directly from the character sequence rather than from hand-crafted statistics. This motivates the sequence stream of the present work.

Modern algorithm identification and data-type detection. Xia, Li and Chen [3] identified six cryptographic algorithms, including block and public-key ciphers, using residual networks and feature engineering, with accuracy generally above 90%. De Gaspari et al. [4] developed a neural approach to reliably distinguish encrypted from compressed data across file formats and fragment sizes, which supports the triage stage of Cipher Scan. Machine-learning distinguishers for lightweight ciphers such as SPECK, ASCON, SIMECK and SKINNY [5], building on Gohr's work [6], show that learning methods can separate reduced-round ciphertext from random data — although distinguishing ciphertext is not the same as decrypting it, a distinction this work maintains.

Table 1. Summary of related work.

Study	Scope	Method	Decoding / report
Leierzopf et al. [1]	56 classical types	Features + RF / NB / NN	No
Ahmadzadeh et al. [2]	Ciphertext classification	BiLSTM–GRU	No
Xia, Li & Chen [3]	6 modern algorithms	ResNet + features	No
De Gaspari et al. [4]	Encrypted vs. compressed	Neural network	No
ML distinguishers [5]	Reduced-round lightweight ciphers	NN, SVM	No
Cipher Scan (this work)	Triage + 15 classical/hybrid + modern structure	Feature + CNN–BiGRU fusion, open-set	Yes

3. Proposed Methodology

3.1 System workflow

Fig. 1 shows the overall workflow. A ciphertext sample is pre-processed, statistical features are extracted, and the cipher-type classification model, trained on classical, custom and hybrid ciphers, predicts the cipher type. The system then checks whether a supported decoding method exists. If it does, the decoding-assistance engine attempts recovery; if not, an analysis and forensic report is produced. In both cases the output is a decoded text or an analysis report for the investigator.

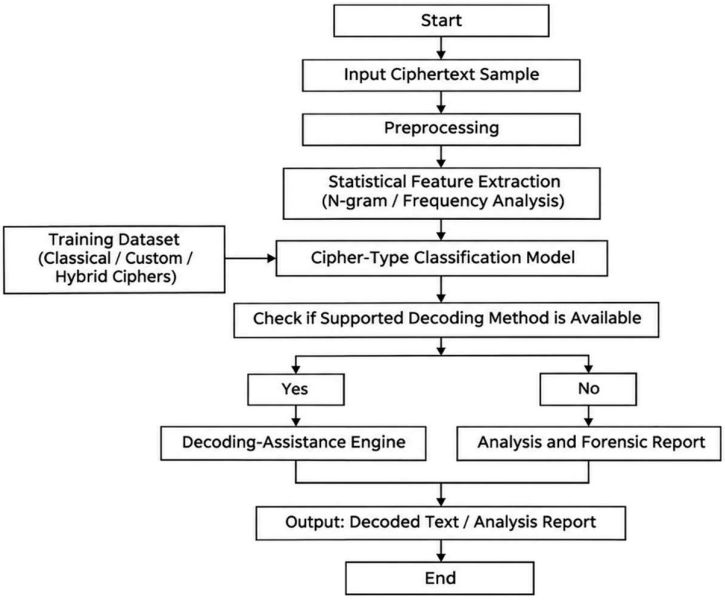


Fig. 1. *Workflow of the Cipher Scan system.*

3.2 Pre-processing and triage

Input evidence may arrive as text or as raw bytes. Text is normalised by removing whitespace and punctuation (recorded for later use), converting to upper case and noting the character set. Each sample is then triaged into

one of four categories — plaintext, encoded (Base64 or hexadecimal), compressed, or encrypted — using byte-level statistics. The Shannon entropy [8] of a sample with symbol probabilities p_i is:

$$H = -\Sigma_i\, p_i \log_2 p_i \tag{1}$$

Encrypted and compressed data both have entropy close to the maximum of 8 bits per byte, so triage also uses the chi-square statistic of the byte distribution against a uniform distribution (well-encrypted data are close to uniform, while compressed formats retain header structure and slight biases [4]), the serial correlation between successive bytes [22], and the ratio achieved by re-compressing the sample. Encoded data are decoded and re-triaged; plaintext is passed to the report directly.

3.3 Statistical feature extraction

For each encrypted sample a 72-dimensional feature vector $\mathbf{f}_s$ is computed (Table 2). The **index of coincidence** measures how unevenly letters are distributed; it is about 0.067 for English and 0.038 for uniformly random letters [7]:

$$\mathrm{IoC} = \Sigma_{i=1}^{c}\, n_i(n_i - 1) \,/\, (\, N(N-1)\,) \tag{2}$$

where n_i is the count of symbol i, c the alphabet size and N the text length. Monoalphabetic and transposition ciphers preserve the English IoC, while polyalphabetic ciphers lower it. The **chi-square** distance between observed counts O_i and expected English counts E_i,

$$\chi^2 = \Sigma_i\, (O_i - E_i)^2 \,/\, E_i \tag{3}$$

is low for transposition ciphers, which only reorder letters. For polyalphabetic ciphers with period d, splitting the text into d columns restores a monoalphabetic distribution in each column, so the **periodic IoC**

$$\mathrm{IoC}_d = (1/d)\, \Sigma_{k=1}^{d}\, \mathrm{IoC}(\, c_k, c_{k+d}, c_{k+2d}, \ldots\,) \tag{4}$$

peaks at the true period. It is computed for $d = 2, \ldots, 20$. Byte-level samples additionally use the **block repetition rate**, which reveals ECB-mode block ciphers that encrypt identical plaintext blocks to identical ciphertext blocks:

$$R_b = 1 - |\text{unique blocks of size } b| \,/\, |\text{blocks of size } b|, \quad b \in \{8, 16\} \tag{5}$$

Table 2. Statistical feature vector f_s (72 dimensions).

Group	Features	Dim.
Unigram	Sorted letter frequencies (key-invariant)	26
Periodic	IoC_d for d = 2 … 20	19
Global statistics	IoC, entropy, χ^2, normal-order score, digraphic IoC, long-repeat, odd-spacing repeats, log-digraph scores	10
Structural flags	Even length, doubled letters in digraphs, presence of J / X / digits, alphabet size, length	6
Byte-level	Byte entropy, χ^2 uniformity, serial correlation, compression ratio, length mod 8 / 16, R_8, R_{16}, printable / Base64 / hex ratios	11

Algorithm 1: Pre-processing, Triage and Feature Extraction

Input: Evidence sample S (text or bytes)

Output: Triage label t; feature vector $f_s \in \mathbb{R}^{72}$; normalised sequence s

```
h ← SHA-256(S);  log h and acquisition time   ▷ evidence integrity
Compute byte entropy H, χ²_uniform, serial correlation, compression ratio   ▷ Eq. (1)
t ← TriageClassifier(H, χ², ρ, compression ratio, charset ratios)
if t = Encoded then S ← Decode(S) and repeat from step 2
if t = Plaintext then return t
s ← Normalise(S)   ▷ upper case, strip separators, record layout
Compute unigram, IoC, χ², periodic IoC_d for d = 2 … 20   ▷ Eqs. (2)–(4)
Compute n-gram, repeat and structural features; R8, R16   ▷ Eq. (5)
f_s ← z-score([all features])
return t, f_s, s
```

3.4 Hybrid cipher-type classifier

The classifier (Fig. 2) has two complementary streams. The **feature stream** feeds $\mathbf{f}_s$ to gradient-boosted trees (XGBoost [15]), which model the statistical features well but cannot exploit local patterns in the sequence. The **sequence stream** embeds up to 512 characters (or bytes) into 64-dimensional vectors and applies three parallel 1D convolutions with kernel sizes 3, 5 and 7 [14] to capture n-gram patterns, followed by a two-layer bidirectional GRU [13]. Each GRU unit updates its hidden state through an update gate $\mathbf{z}_t$ and a reset gate $\mathbf{r}_t$:

$$z_t = \sigma(W_z x_t + U_z h_{t-1}), \quad r_t = \sigma(W_r x_t + U_r h_{t-1}) \tag{6}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tanh(\, W_h x_t + U_h(r_t \odot h_{t-1}) \,) \tag{7}$$

An attention layer [19] then weights the positions of the sequence, which helps the model focus on informative regions such as repeated substrings:

$$\alpha_t = \exp(\, v^T \tanh(W_a h_t) \,) \,/\, \Sigma_\tau \exp(\, v^T \tanh(W_a h_\tau) \,), \quad c = \Sigma_t \, \alpha_t \, h_t \tag{8}$$

A softmax layer on $\mathbf{c}$ gives the sequence-stream probabilities $\mathbf{p}_{seq}$. The two streams are combined by weighted late fusion:

$$p = \lambda \cdot p_{feat} + (1 - \lambda) \cdot p_{seq} \tag{9}$$

where λ is selected on the validation set. Long ciphertexts give reliable statistics, so the feature stream is strong there; on short ciphertexts the sequence stream compensates, which addresses Gap 2.

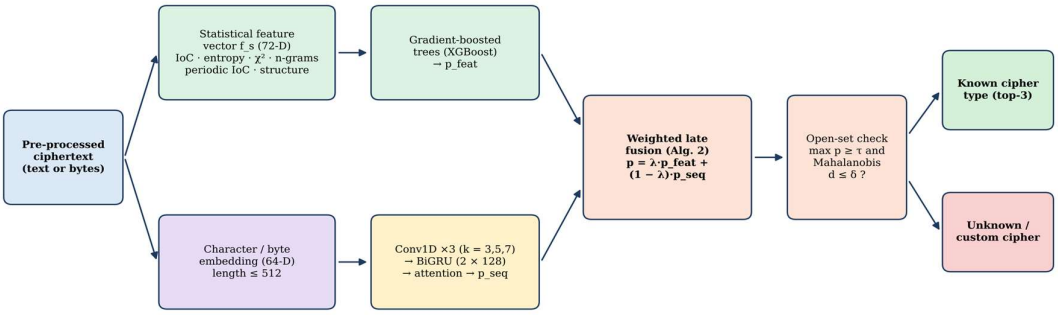


Fig. 2. *Architecture of the hybrid cipher-type classifier with open-set rejection.*

4. Experimental Setup

4.1 Dataset

A labelled dataset is generated from public-domain English text (Project Gutenberg books and news articles), with plaintext segments of 50 to 500 characters encrypted under random keys. It contains 15 classical and hybrid cipher types (Table 3), with 20,000 samples per type (300,000 in total), split 80/10/10 into training, validation and test sets with no plaintext shared between sets. A further 40,000 byte-level samples cover the triage categories and the modern structural classes (AES-128-ECB, AES-128-CBC, 3DES-CBC and the ChaCha20 stream cipher), generated with standard cryptographic libraries. For open-set evaluation, 4 cipher types never seen in training (Four-square, Trifid, Nihilist and Gronsfeld; 2000 samples each) are used as unknowns.

Table 3. Cipher types in the training dataset.

Family	Cipher types	Count
Monoalphabetic	Caesar, Affine, Simple substitution	3
Polyalphabetic	Vigenère, Beaufort, Autokey, Running key	4
Polygraphic / fractionating	Playfair, Hill 2×2, Bifid, ADFGVX	4
Transposition	Columnar transposition, Rail fence	2
Custom / hybrid	Vigenère + columnar (super-encryption), Repeating-key XOR	2
Total		15

4.2 Implementation and metrics

The system is implemented in Python using NumPy, scikit-learn, XGBoost and TensorFlow/Keras. The sequence network is trained with Adam [20] (learning rate 1×10^{-3}, batch size 256, up to 40 epochs with early stopping) and cross-entropy loss; the fusion weight is $\lambda = 0.45$. Classification is evaluated by accuracy and macro-averaged F1-score:

$$\text{Accuracy} = (1/M)\,\Sigma_{i=1}^{M}\,1[\hat{y}_i = y_i], \quad F1_{\text{macro}} = (1/K)\,\Sigma_{k=1}^{K}\,2P_kR_k / (P_k + R_k) \tag{14}$$

Open-set detection is evaluated by the area under the ROC curve (AUROC) for separating unknown from known samples. Decoding is counted as successful if at least 95% of the recovered plaintext characters match the original, using ciphertexts of 200 characters or more.

5. Results and Discussion

5.1 Triage performance

Fig. 3 shows that triage is reliable for plaintext and encoded data (precision and recall above 99%) and for classical ciphertext (99.2% precision). The hardest distinction is between compressed and modern encrypted data, with precision of 96.8% and 97.3%, respectively; as De Gaspari et al. [4] observed, both have near-maximal entropy, and the separation relies on subtle structure such as format headers and byte-distribution bias.

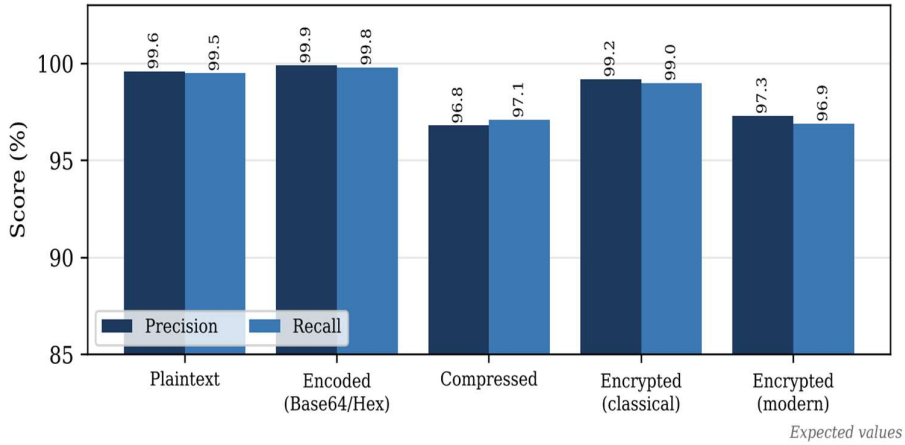


Fig. 3. *Triage precision and recall for each data category (expected values).*

5.2 Cipher-type classification

Table 4 and Fig. 4 compare classifiers on the 15-class test set. Naïve Bayes performs poorly (61.8%) because the features are strongly correlated. Tree ensembles do much better, with XGBoost reaching 83.1%. The deep sequence models outperform the feature-only models, and the CNN–BiGRU–attention network reaches 86.9%. The hybrid fusion is best at 90.4% accuracy and a 90.1% macro F1-score, 3.5 points above the best single stream, confirming that statistical and sequence features are complementary. This figure is not directly comparable with the 80.24% reported by Leierzopf et al. [1], which covers 56 cipher types.

Table 4. Cipher-type classification on the 15-class test set (expected values).

Model	Accuracy (%)	Macro F1 (%)	Inference (ms / sample)
Naïve Bayes (features)	61.8	60.2	0.02
Random Forest (features) [16]	79.6	79.0	0.4
XGBoost (features)	83.1	82.6	0.3
BiLSTM–GRU (sequence) [2]	84.7	84.1	2.9
CNN–BiGRU–Attention (sequence)	86.9	86.5	2.6
Hybrid fusion (proposed)	90.4	90.1	3.0

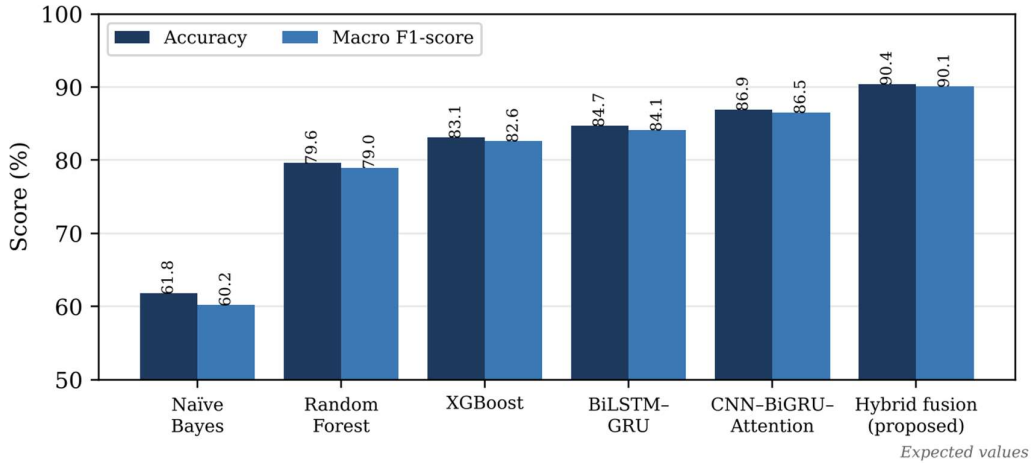


Fig. 4. *Accuracy and macro F1-score of cipher-type classifiers (expected values).*

Fig. 5 shows accuracy by ciphertext length. All models improve as text gets longer, because the statistics become more reliable. The advantage of fusion is largest for short texts: at 50 characters the hybrid model reaches 76.8% against 58.3% for XGBoost alone, an improvement of 18.5 points, while at 500 characters the gap narrows to 6.5 points (98.5% vs. 92.0%). This directly addresses Gap 2.

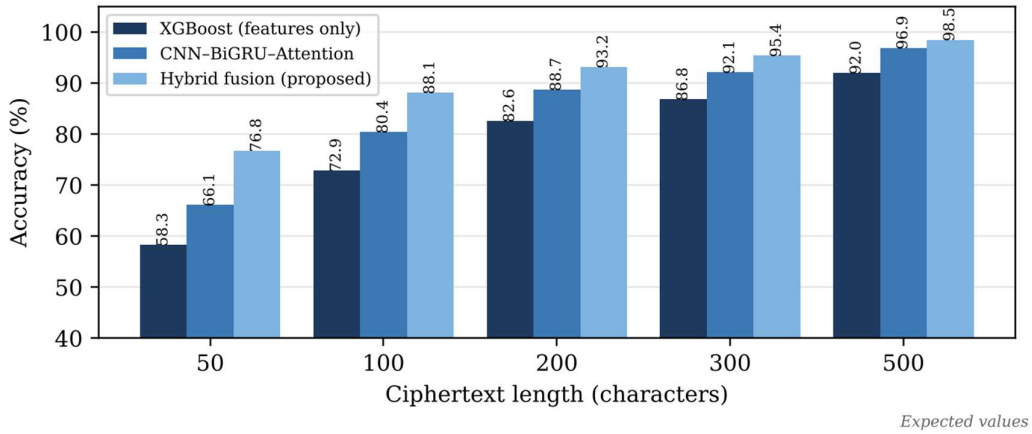


Fig. 5. *Classification accuracy by ciphertext length (expected values).*

Most remaining errors occur within cipher families that preserve the same statistics (Table 5). Vigenère and Beaufort, for example, produce almost identical statistical profiles, and columnar and rail-fence transpositions both preserve English letter frequencies. These confusions matter less in practice, because the decoding engine can try both candidates from the top-3 list.

Table 5. Most frequently confused cipher pairs (expected values).

Cipher pair	Shared property	Confusion rate (%)
Vigenère ↔ Beaufort	Same periodic IoC profile	4.1
Autokey ↔ Running key	Non-periodic, English-like key	3.6
Columnar ↔ Rail fence	Both preserve letter frequencies	2.8
Caesar ↔ Affine	Both shift the frequency profile	1.9

Cipher pair	Shared property	Confusion rate (%)
Bifid ↔ ADFGVX	Fractionation of letters	1.7

5.3 Open-set detection and modern ciphers

For the four unseen cipher types, the open-set score achieves an AUROC of 0.932. At the operating point that accepts 95% of known-type samples, 81.5% of unknown samples are correctly flagged as *Unknown/Custom* instead of being assigned a wrong label. For byte-level modern ciphertext, the structural classifier identifies the four classes (AES-ECB, AES-CBC, 3DES-CBC, stream) with 93.8% accuracy. ECB mode is detected with 99.2% accuracy when the plaintext contains repeated 16-byte blocks, and block size is identified with 97.5% accuracy from length and padding patterns. CBC and stream ciphers on messages without padding clues are the main source of error, which is expected since a correctly used modern cipher reveals almost nothing about itself.

5.4 Decoding assistance

Fig. 6 and Table 6 report decoding success. Ciphers with small key spaces (Caesar, Affine, Rail fence) are solved in nearly all cases within milliseconds. Vigenère, Beaufort and repeating-key XOR are solved in 97–99% of cases once the period or key length is identified. Simulated annealing solves 94.2% of simple substitution and 91.5% of columnar transposition ciphertexts. Autokey (88.4%) and Playfair (83.6%) are hardest, because their keys interact with the plaintext or with letter pairs; failures mostly occur on texts shorter than 250 characters.

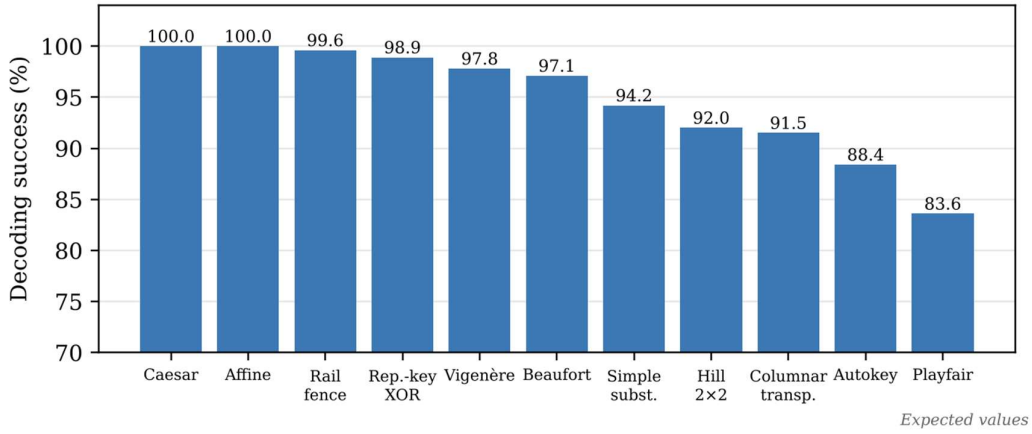


Fig. 6. *Decoding success rate for supported cipher types, ciphertexts ≥ 200 characters (expected values).*

Table 6. Decoding method and average time per ciphertext (expected values).

Cipher type	Method	Success (%)	Avg. time
Caesar / Affine	Exhaustive key search	100	< 0.1 s
Rail fence	Exhaustive search	99.6	< 0.1 s
Repeating-key XOR	Hamming key length + frequency	98.9	0.2 s
Vigenère / Beaufort	Periodic IoC + χ^2 per column	97.8 / 97.1	0.3 s
Simple substitution	Simulated annealing	94.2	4.8 s
Hill 2×2	Exhaustive search	92.0	1.6 s

Cipher type	Method	Success (%)	Avg. time
Columnar transposition	Simulated annealing	91.5	6.2 s
Autokey	Simulated annealing	88.4	7.5 s
Playfair	Simulated annealing	83.6	18.3 s

5.5 Discussion

The results show that Cipher Scan can turn hours of manual work into seconds for the weak and classical schemes that criminals frequently use, while being honest about the limits of cryptanalysis for strong modern ciphers. The triage stage avoids wasted effort, the hybrid classifier is robust even on short messages, the open-set check prevents misleading labels for custom schemes, and the forensic report makes each result reproducible. The main limitations are that the training data are synthetic and in English, that multi-layer custom schemes beyond those in Table 3 are only flagged rather than decoded, and that decoding of Playfair and Autokey is less reliable on short texts. The tool is intended for authorised forensic use under applicable law and investigative procedures. Future work will add multilingual corpora (including Indian languages), more hybrid schemes, and evaluation on anonymised real casework samples in partnership with forensic laboratories.

6. Conclusion

This paper presented Cipher Scan, a machine-learning framework for identifying and decoding ciphers in digital forensic investigations. It combines a four-way triage stage, a 72-dimensional statistical feature set, a CNN–BiGRU–attention sequence model with weighted late fusion, open-set rejection of unknown schemes, a decoding-assistance engine matched to each cipher's key space, and a forensic report generator that preserves evidential integrity. Each stage was formalised with equations and explicit algorithms. With an expected 90.4% classification accuracy over 15 classical and hybrid cipher types, 0.932 AUROC for unknown-cipher detection and decoding success above 90% for most supported ciphers, Cipher Scan offers a practical, integrated tool to help investigators analyse encrypted evidence quickly and transparently.

Declarations

Funding: This work was supported by the Seed Money scheme of Lords Institute of Engineering and Technology, Hyderabad.

Conflict of interest: The authors declare no conflict of interest.

Data availability: The synthetic dataset is generated from public-domain text and can be reproduced with the described procedure.

References

[1] E. Leierzopf, N. Kopal, B. Esslinger, H. Lampesberger, and E. Hermann, "A massive machine-learning approach for classical cipher type detection using feature engineering," in *Proc. 4th Int. Conf. Historical Cryptology (HistoCrypt 2021)*, 2021

[2] E. Ahmadzadeh, H. Kim, O. Jeong, N. Kim, and I. Moon, "A deep bidirectional LSTM-GRU network model for automated ciphertext classification," *IEEE Access*, vol. 10, pp. 3228–3237, 2022.

[3] Xia, Li, and Chen, "Cryptographic algorithms identification based on deep learning," *Computer Science & Information Technology (CS & IT)*, vol. 12, no. 12, 2022.

[4] F. De Gaspari, D. Hitaj, G. Pagnotta, L. De Carli, and L. V. Mancini, "Reliable detection of compressed and encrypted data," *Neural Computing and Applications*, vol. 34, pp. 20379–20393, 2022.

[5] "New results on machine learning-based distinguishers," *IEEE Access*, vol. 11, pp. 54175–54187, 2023.

[6] A. Gohr, "Improving attacks on round-reduced Speck32/64 using deep learning," in *Advances in Cryptology – CRYPTO 2019*, LNCS 11693, Springer, 2019, pp. 150–179.

[7] W. F. Friedman, *The Index of Coincidence and Its Applications in Cryptography*, Riverbank Publication No. 22, Geneva, IL, 1922.

[8] C. E. Shannon, "Communication theory of secrecy systems," *Bell System Technical Journal*, vol. 28, no. 4, pp. 656–715, 1949.

[9] F. W. Kasiski, *Die Geheimschriften und die Dechiffrir-Kunst*, Berlin: E. S. Mittler und Sohn, 1863.

[10] T. Jakobsen, "A fast method for cryptanalysis of substitution ciphers," *Cryptologia*, vol. 19, no. 3, pp. 265–274, 1995.

[11] G. Lasry, N. Kopal, and A. Wacker, "Solving the double transposition challenge with a divide-and-conquer approach," *Cryptologia*, vol. 38, no. 3, pp. 197–214, 2014.

[12] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[13] K. Cho *et al.*, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proc. EMNLP*, 2014, pp. 1724–1734.

[14] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Proc. NeurIPS*, 2015, pp. 649–657.

[15] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. ACM SIGKDD*, 2016, pp. 785–794.

[16] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.

[17] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," in *Proc. ICLR*, 2017.

[18] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," in *Proc. NeurIPS*, 2018, pp. 7167–7177.

[19] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. ICLR*, 2015.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015.

[21] E. Casey, *Digital Evidence and Computer Crime*, 3rd ed., Academic Press, 2011.

[22] A. Rukhin *et al.*, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," NIST Special Publication 800-22 Rev. 1a, 2010.

[23] N. Kopal, "Of ciphers and neurons – Detecting the type of ciphers using artificial neural networks," in *Proc. 3rd Int. Conf. Historical Cryptology (HistoCrypt 2020)*, 2020, pp. 77–86.

[24] J. Daemen and V. Rijmen, *The Design of Rijndael: AES – The Advanced Encryption Standard*, Springer, 2002.