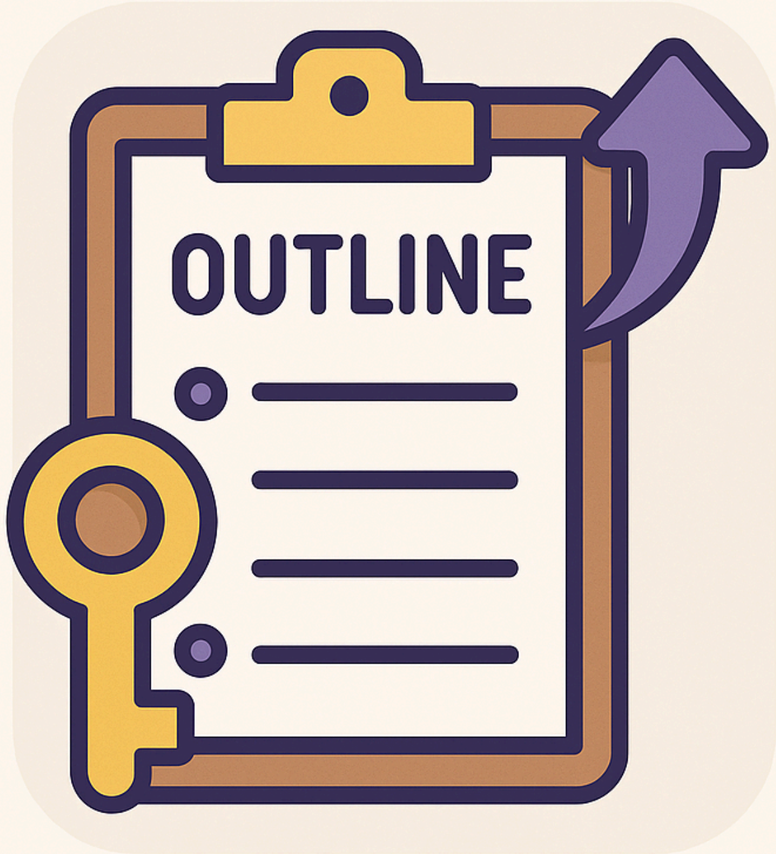
Large Language Models as Cryptanalysts: Assessing Decryption Capabilities Across Classical Ciphers

Spring 2025 – American University of Beirut

Course: CMPS396AH— Instructor: Amer Mouawad

Aline Hassan | Zeinab Saad | Hadi Tfaily

OUTLINE



- Introduction
- Background
- System Design
- Experimental Results
- Challenges and Limitations
- Future Improvements
- Conclusion and Future Work

Introduction

Why LLMs?

- Excellent at pattern recognition
- Proven in code and data tasks
- Cryptanalysis remains underexplored

Why It Matters

- Potential for new AI-driven decryption methods
- Helps anticipate cybersecurity threats
- Reveals ethical and technical limits of LLMs

Cryptanalysis Overview & History

Why Cryptanalysis Matters

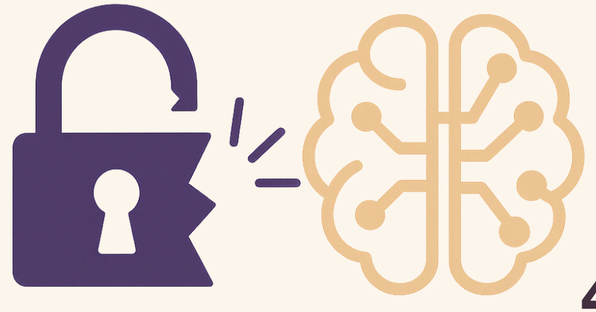
- CIA Triad: Confidentiality, Integrity, Availability
- Attacks aim to break encryption and expose vulnerabilities

Classical Cipher

- Caesar
- Vigenère
- monoalphabetic
- Rail Fence

Common Attacks

- Known/Chosen Plaintext Attacks (KPA/CPA)
- Brute Force, Side-Channel, Fault Analysis



Literature Review

ML in Classical Cryptanalysis

- LSTM, Transformers classify ciphers with ~83% accuracy
- Detect patterns in substitution/transposition ciphers

Deep Learning Advances

- ANNs, CNNs improve key prediction
- Applied in lightweight ciphers (e.g., S-AES, S-DES)

Literature Review

Generative Approaches (GANs)

- UC-GAN cracks multiple ciphers without linguistic knowledge

LLMs in Cipher Decryption

- GPT-4/GPT-4o effective in zero-shot decryption
  - High success on simple ciphers
  - Struggles with complex encryption & consistency

Gap in Existing Research

Research Gap Identified

- No existing work fine-tunes LLMs or trains transformers for cryptanalysis
- Opportunity to assess LLMs' active decryption ability
- This project bridges the gap with targeted evaluation of LLMs across classical ciphers

Algorithmic Definitions

Caesar Cipher:

$$C = E(P, k) = (P + k) mod 26$$

$$P = D(C, k) = (C - k) mod 26$$

Rail Fence Cipher:

H . . . O . . . R . .
. E . L . . . O . L .
. . L . . . W . . . D

Monoalphabetic Cipher:

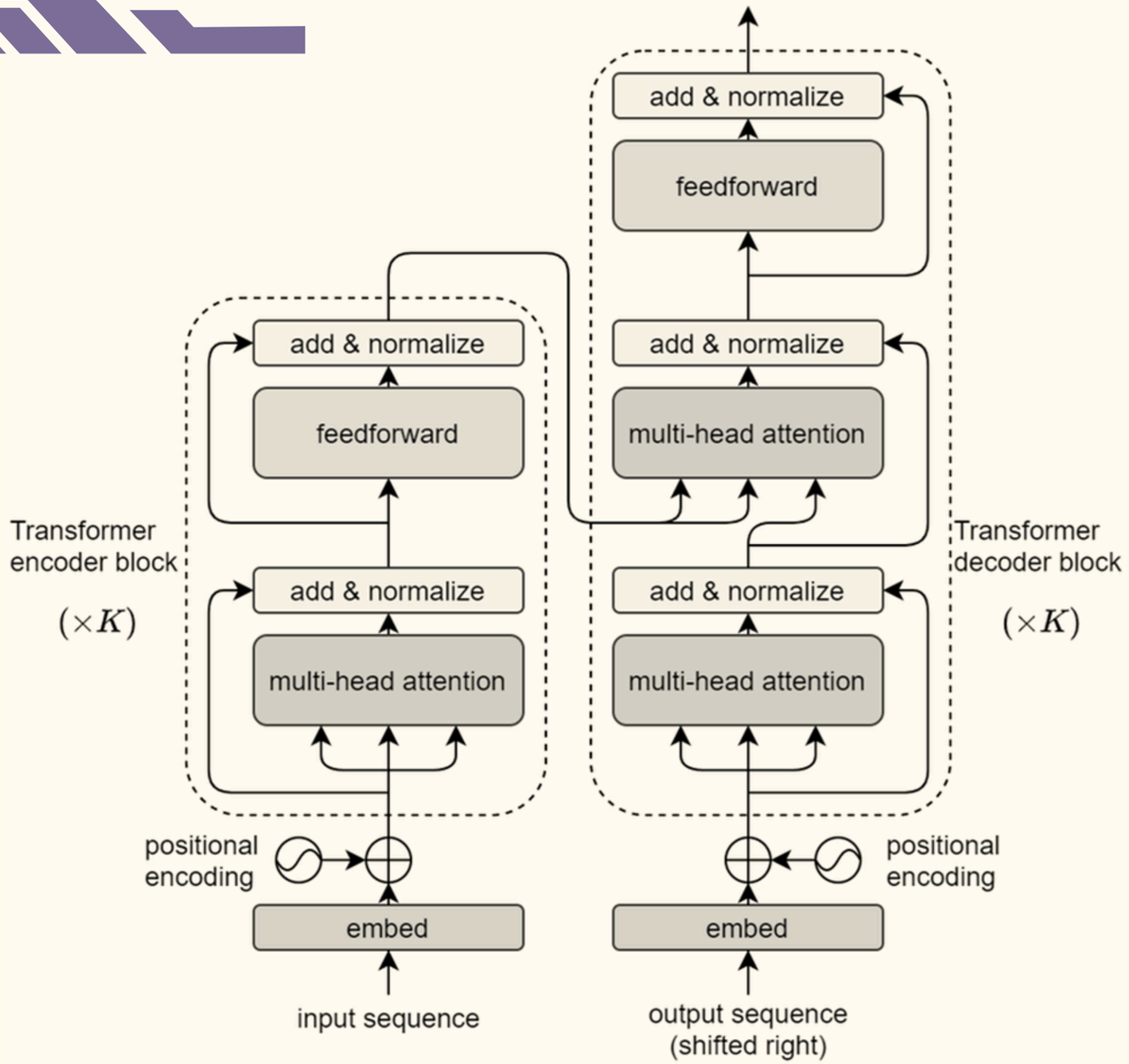
$$C = E(P, \sigma) = \sigma(P)$$
$$P = D(C, \sigma^{-1}) = \sigma^{-1}(C)$$

Vigen`ere Cipher

$$C_i = (P_i + K_i \ mod \ |K|) mod 26$$

$$P_i = (C_i - K_i \ mod \ |K|) mod 26$$

Transformer Architecture

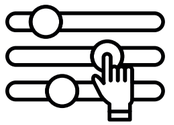


Methodology

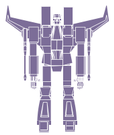
STEP 01 — Zero-shot Learning

STEP 02 — Few-shot Learning

STEP 03 — Finetuning Mistral 7B



STEP 04 — Training Vanilla Transformers

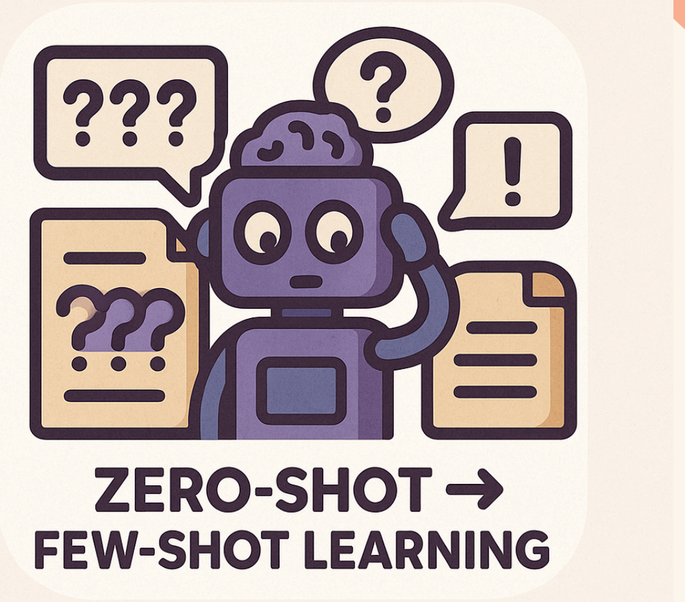


05 — Training Enhanced Transformer

Zero-shot Caesar Cipher Evaluation

Zero-shot Setup:

- Models: Mistral 7B (local), GPT-01, DeepSeek-R1
- 20 Caesar cipher test cases with random shifts (1–25)
- Prompt: "Decrypt this Caesar cipher text: [text]..."
- Hardware: A100 GPU (local), OpenAI API, web interface
- Manual verification, clean ciphertext input



Few-shot Caesar Cipher Evaluation

Few-shot Extension:

- Same models, hardware, and test set as zero-shot
- Prompt includes 3 demonstration pairs (shifts: 1, 3, 4)
- Prompt format: Examples + Target Ciphertext + Explanation
- Consistent inference parameters (temp=0, max tokens capped)
- Randomized example order, blind test ciphertexts
- Outputs cross-verified by three reviewers

Fine-tuning

Model & Architecture:

- Mistral-7B-v0.3 (decoder-only, autoregressive, 8192-token context)

Tokenization:

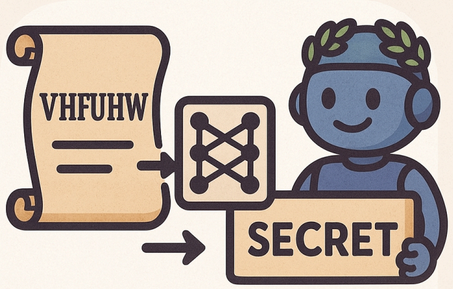
- BPE tokenizer (32k vocab), instruction-response format
- Efficient, deterministic, subword-aware

Dataset & Preprocessing

- JSONL with natural language prompts and outputs
- Preprocessed to 4096 tokens, left-aligned, truncated as needed

Training Setup

- Epochs: 5, Batch: 4 × 4, LR: 2e-5
- Optimizer: AdamW + FP16 + checkpointing
- Evaluated every epoch, logged every 10 steps



Transformer

Model Design

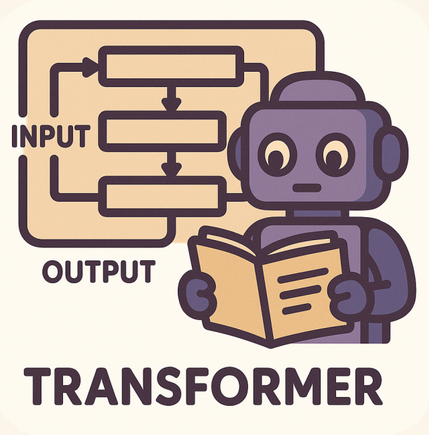
- PyTorch implementation of encoder-decoder transformer
- Applies to Caesar, Vigenère, Rail Fence, Monoalphabetic
- Uses char-level tokens with <SOS>, <EOS> markers

Architecture

- Sinusoidal position encodings
- Multi-head self & cross-attention
- 256-token fixed-length inpu

Optimization

- Hyperparameters tuned with Optuna
- Trained with Adam, cross-entropy loss
- Dropout, layer norm, checkpointing used



Enhanced Transformer for Caesar Cipher

Positional Encoding Innovation

- Replaced sinusoidal with learned embeddings
- Combined token + position using addition

Why It Works

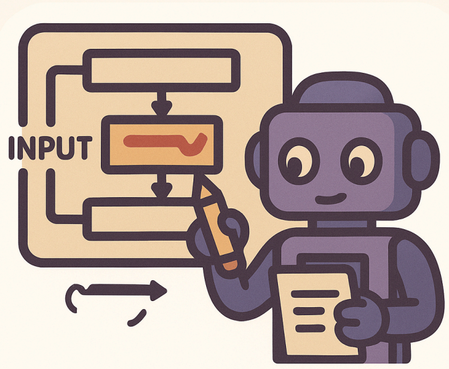
- Learns Caesar-style shift patterns efficiently
- Keeps original embedding dimensions
- Optimized for stable gradient flow

Data Strategy

- 1M Caesar samples (Google + Tatoeba)
- Augmentation: R3 → R1–5 → all 25 shifts
- Tokenized at character level (max 256)

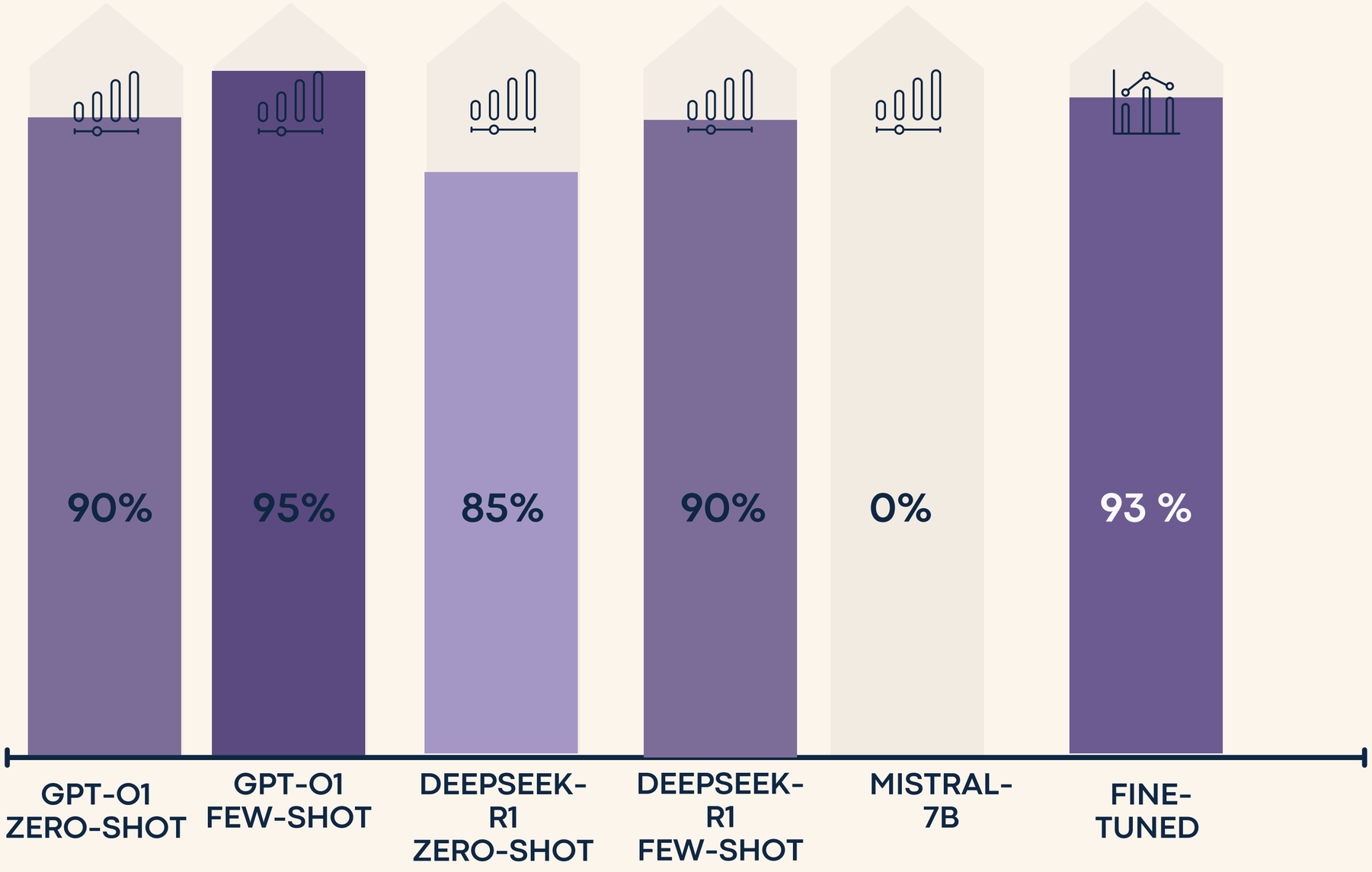
Implementation Notes

- Token + Positional + Projection layers
- Uses special tokens: <PAD>, <SOS>, <EOS>
- Stratified 80/10/10 data split



	D_MODE	HEADS	LAYERS	D_FF	MAX SEQ	DROPOUT
Caesar (Vanilla)	512	4	6	512	256	0.1103
Caesar (Updated)	512	8	3	1024	256	0.104
Monoalphabetic 1 Key	512	8	6	1024	256	0.1287
Monoalphabetic 5 Keys	512	2	6	256	256	0.2198
Monoalphabetic 100 Keys	512	8	8	256	256	0.1371
Vigen`ere 1 Key	512	8	6	256	256	0.1125
Vigen`ere 5 Keys	256	2	6	512	256	0.1021
Vigen`ere 100 Keys	512	8	8	1024	256	0.1019
Rail Fence	256	8	2	512	512	0.1861

Caesar Cipher Results



GPT-O1

GPT-O1 achieved 90% accuracy on Caesar cipher decryption with zero-shot prompting, improving to 95% using few-shot examples

DEEPSEEK-R1

DeepSeek-R1 reached 85% accuracy in zero-shot mode and increased to 90% with few-shot guidance

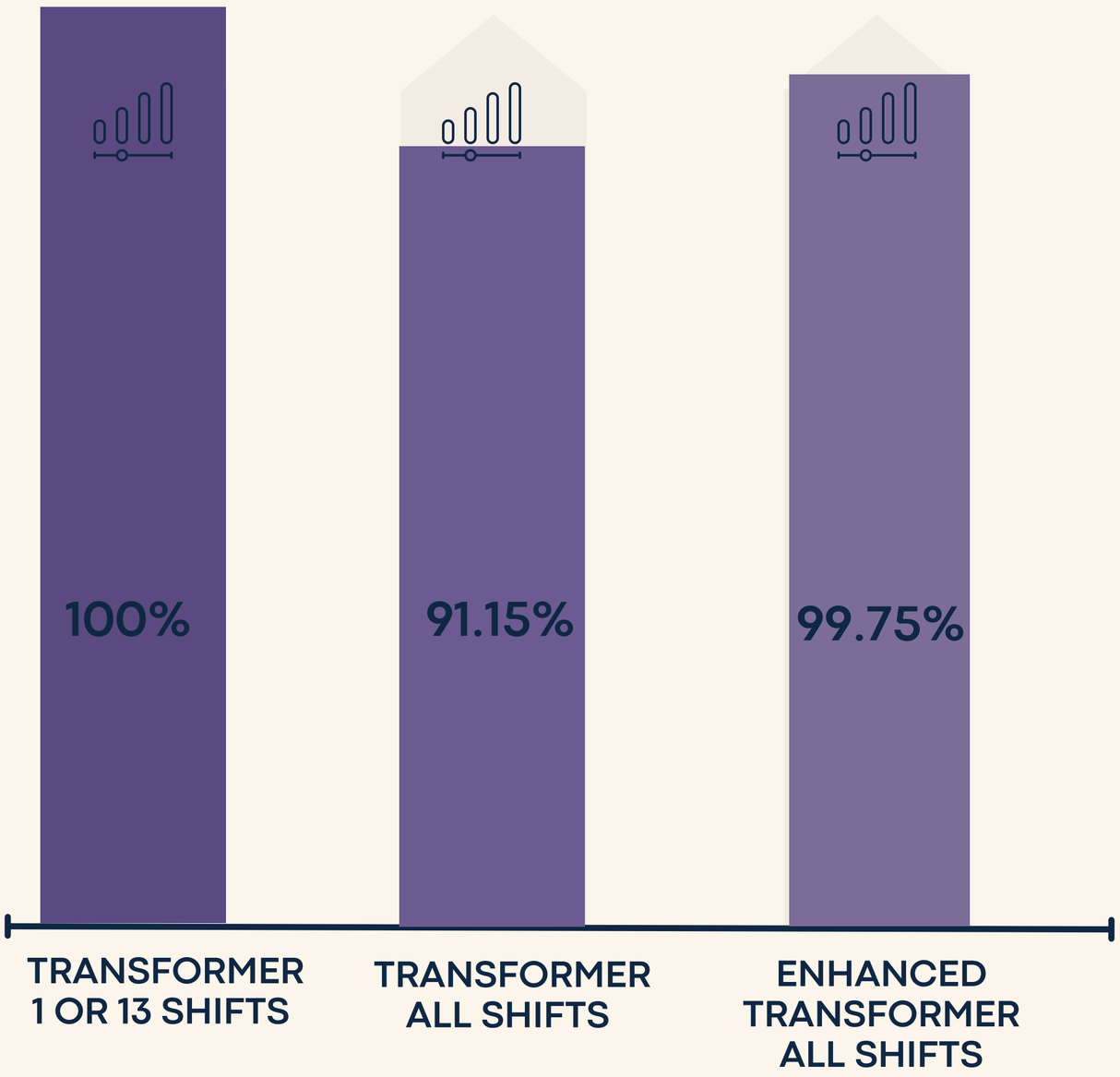
MISTRAL-7B

Mistral-7B failed to decrypt any Caesar cipher samples correctly, showing 0% accuracy across all conditions.

FINE-TUNED TRANSFORMER:

A fine-tuned transformer achieved 93% accuracy on Caesar cipher tasks, demonstrating effective adaptation through training.

Caesar Cipher Results



VANILLA TRANSFORMER (1 OR 13 SHIFTS)

The vanilla transformer perfectly decrypted Caesar cipher messages with 100% accuracy when trained on 1 or 13 fixed shifts.

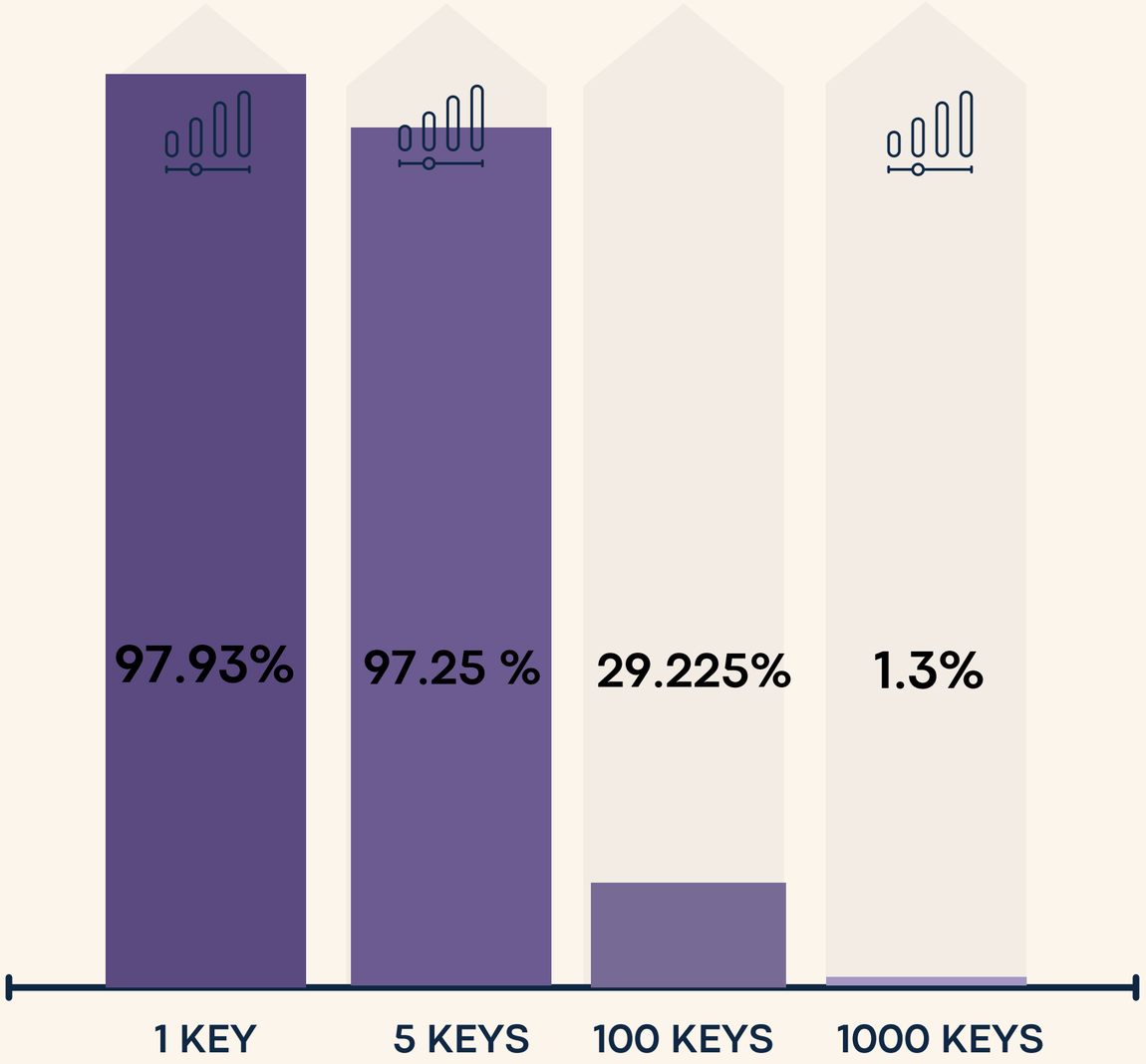
VANILLA TRANSFORMER (25 SHIFTS)

When expanded to handle all 25 shift values, the vanilla transformer's accuracy dropped slightly to 91.15%.

ENHANCED TRANSFORMER

The enhanced transformer achieved 99.75% overall accuracy and 99.9% character-level precision on Caesar cipher tasks, effectively handling all 25 shifts through improved architecture and training.

Monoalphabetic Cipher Results



1 KEY

With a single substitution key, the vanilla transformer achieved 97.93% sample accuracy and 98.87% character-level precision, indicating strong pattern learning.

5 KEYS

Expanding to five substitution keys slightly lowered sample accuracy to 97.25% while improving character precision to 99.62%, showing stable generalization.

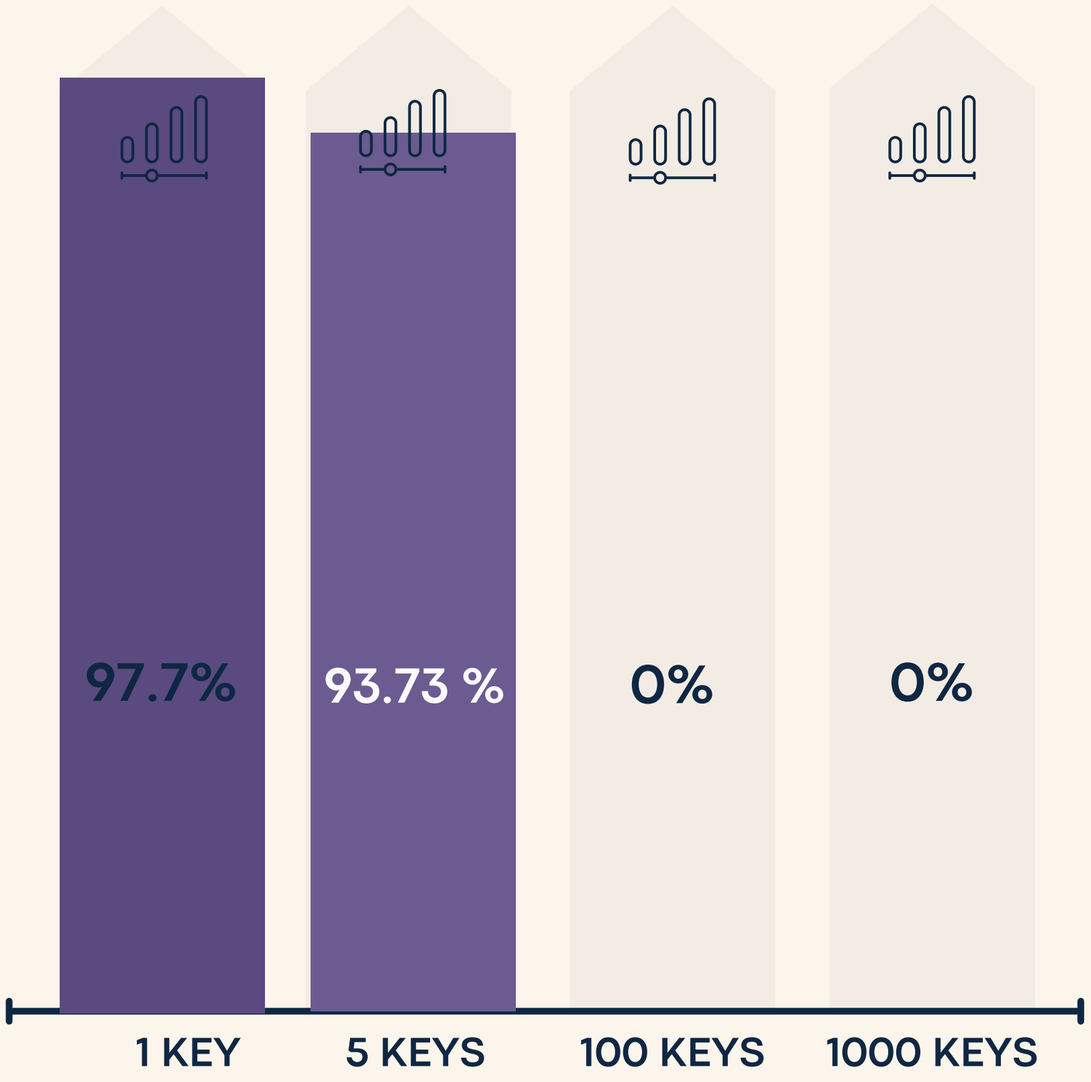
100 KEYS

A fine-tuned transformer achieved 93% accuracy on Caesar cipher tasks, demonstrating effective adaptation through training.

1000 KEYS

At 1000 random substitution keys, the model's sample accuracy collapsed to 1.3% and character accuracy to 32%, highlighting the infeasibility of learning arbitrary mappings at scale.

Vigenère Cipher Findings



1 KEY

Using a single keyword, the model achieved 97.7% sample accuracy and 99.12% character-level precision, showing strong learning of periodic shifts.

5 KEYS

With five keywords, accuracy dropped modestly to 93.725% sample-level and 98.58% character-level, maintaining reliable pattern recognition.

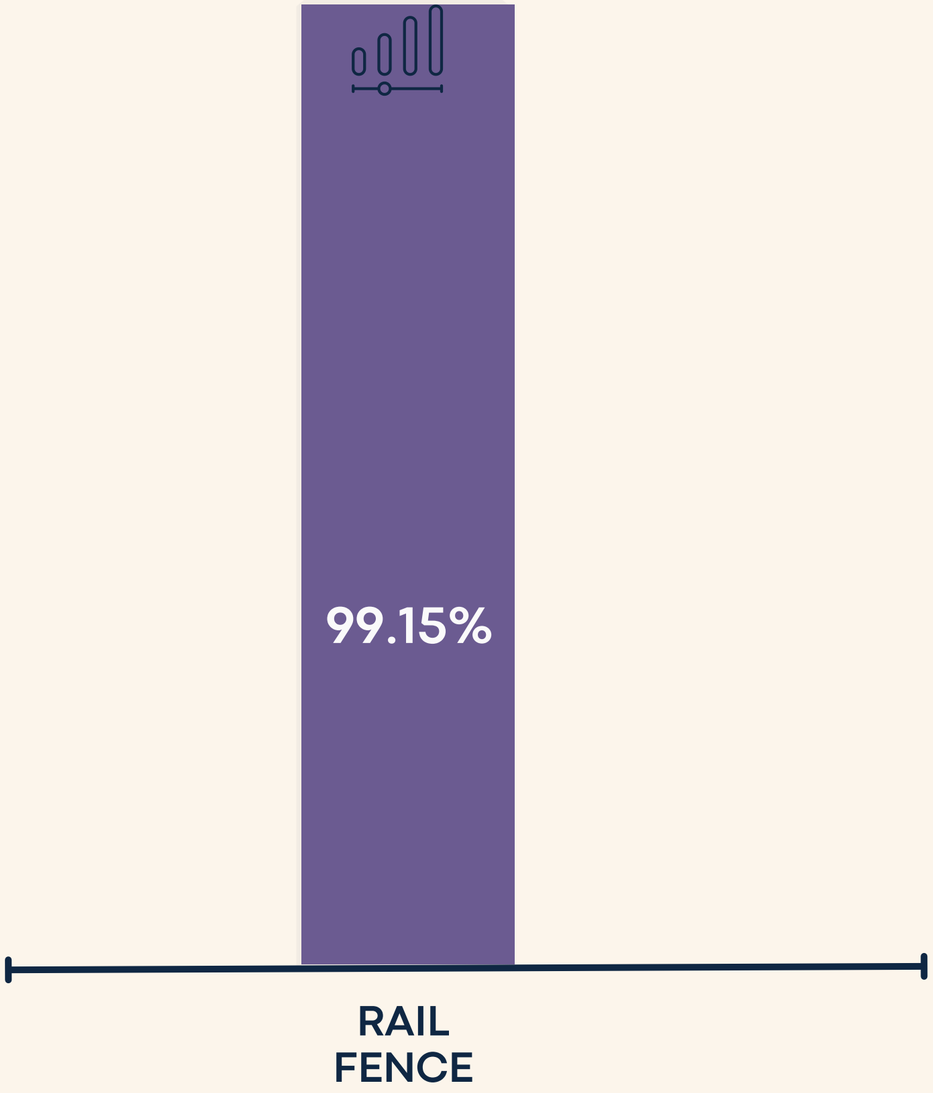
100 KEYS

With 100 keywords, the model dropped to 0% sample accuracy and 32.55% character precision, struggling with key periodicity and mapping.

1000 KEYS

At 1000 keywords, the model failed entirely, scoring 0% sample accuracy and just 2% character precision due to compounding challenges in key periodicity and mapping.

Rail Fence Cipher Results



RAIL FENCE

Across varying rail counts, the model achieved 99.15% accuracy, consistently decoding the geometric zigzag pattern due to its deterministic structure.

mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

*Equal contribution. Listing order is random. Jakob proposed replacing RNNs with self-attention and started the effort to evaluate this idea. Ashish, with Illia, designed and implemented the first Transformer models and has been crucially involved in every aspect of this work. Noam proposed scaled dot-product attention, multi-head attention and the parameter-free position representation and became the other person involved in nearly every detail. Niki designed, implemented, tuned and evaluated countless model variants in our original codebase and tensor2tensor. Llion also experimented with novel model variants, was responsible for our initial codebase, and efficient inference and visualizations. Lukasz and Aidan spent countless long days designing various parts of and implementing tensor2tensor, replacing our earlier codebase, greatly improving results and massively accelerating our research.

[1]Work performed while at Google Brain.

[3]Work performed while at Google Research.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

1 Introduction

Recurrent neural networks, long short-term memory [13] and gated recurrent [7] neural networks in particular, have been firmly established as state of the art approaches in sequence modeling and transduction problems such as language modeling and machine translation [35, 2, 5]. Numerous efforts have since continued to push the boundaries of recurrent language models and encoder-decoder architectures [38, 24, 15].

Recurrent models typically factor computation along the symbol positions of the input and output sequences. Aligning the positions to steps in computation time, they generate a sequence of hidden states h_t, as a function of the previous hidden state h_{t-1} and the input for position t. This inherently sequential nature precludes parallelization within training examples, which becomes critical at longer sequence lengths, as memory constraints limit batching across examples. Recent work has achieved significant improvements in computational efficiency through factorization tricks [21] and conditional computation [32], while also improving model performance in case of the latter. The fundamental constraint of sequential computation, however, remains.

Attention mechanisms have become an integral part of compelling sequence modeling and transduction models in various tasks, allowing modeling of dependencies without regard to their distance in

Challenges and Limitations:

Computational Constraints

- Runtime limits: 6h HPC / 24h Colab capped training duration.
- Intractable key spaces: 26! (monoalphabetic), 26^n (Vigenère) exceeded compute capacity.
- Small batch sizes: Limited to 128 due to GPU memory constraints
- Long training times: 3.2 hours/epoch for 1000-key tasks made full tuning infeasible.
- Sequence limit: 256 tokens max due to $O(n^2)$ attention scaling.

FUTURE IMPROVEMENTS

Hardware Scaling:

Use 4×A100 GPUs for high-key ciphers.

Algorithmic Enhancements:

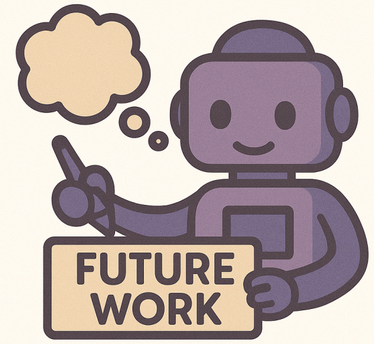
Update Embedding, & Attention Type

Data Pipeline Optimization

On-the-fly cipher gen, binary format, cached attention.

Combined Strategy

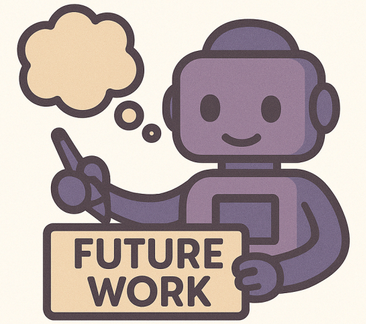
Hardware + smarter algorithms + faster data = breakthrough capability



CONCLUSION

CONCLUSION AND FUTURE WORK

- Transformers excel at structured ciphers: Rail Fence (98.97%), Caesar (99.75%).
- Fail on complex substitution: Monoalphabetic (1.3%), Vigenère (0%).
- Limitations: No arithmetic reasoning, poor long-range pattern handling.
- Future: Extends to more complex encryption techniques



Thank You!!