

Solving Historical Ciphers with AI:

Analysis of GPT’s Capability in Processing and Deciphering Cryptographic Postcards

Eugen Antal

Tomáš Pavuk

Pavol Zajac

Slovak University of Technology in Bratislava

Ilkovičova 3, 841 04 Bratislava

Slovakia

eugen.antal@stuba.sk

xpavukt@stuba.sk

pavol.zajac@stuba.sk

Abstract

In this case study, we focus on the capabilities of ChatGPT for assisting in the research of historical encrypted documents. Our study is based on the dataset of historical encrypted postcards, typically with simple (substitution) ciphers. We split the task into two parts: transcription and interpretation of the postcard image, and then decipherment of the transcribed text. We summarize the current capabilities of (multimodal) AI tools in general, and specifically of GPT-5.2 / GPT-4o that were used as AI tools in our analysis. Following our cryptanalytic pipeline, we show specifically how the AI tool was capable of solving selected typical exemplars from our dataset. We show that GPT-5.2 is already capable of solving most of the required tasks with acceptable quality, and can be useful even for researchers not specifically trained in prompt engineering or artificial intelligence usage.

1 Introduction

Research in historical cryptography requires a significant effort, from collecting historical materials, through their preservation and digitization, as well as for transcription and final decipherment of historical documents. Special tools are developed for computer-assisted transcription of old manuscripts, as well as for decryption of various types of historical ciphers. On the other hand, there is a rapid development of general AI tools and assistants that help with various tasks automatically. In our case study, we focus on whether and how well these general AI tools (focusing on ChatGPT¹) can help in historical cryptography re-

search.

Classical ciphers and their cryptanalysis can represent a significant and challenging cognitive task, with significant variations across different cipher types. As such, multiple authors have proposed using cryptographic challenges for the evaluation of the AI reasoning methods. AICrypto (Wang et al., 2025) is a benchmark based on CTF-type challenges. CipherBank (Li et al., 2025) is a comprehensive benchmark designed to evaluate the reasoning capabilities of LLMs in cryptographic decryption. The dataset comprises encrypted text from specifically generated and curated plaintext. A benchmark with diverse plaintexts from different domains was also proposed and used for LLM evaluation (Maskey et al., 2025b). There is also ongoing research that tries to evaluate the capabilities of AI tools to solve classical and modern ciphers, from initial exemplification studies (Noever, 2023), to a more complex examination of AI reasoning capabilities (Akilesh et al., 2026) and benchmarking studies (Maskey et al., 2025a).

We are, however, unaware of any systematic study that uses a multimodal dataset comprised of diverse historical data. Furthermore, most of the studies seem to focus on evaluating the reasoning capabilities or direct decryption of provided examples, instead of focusing on the whole pipeline of historical cipher research.

Our aim for this paper is to study the capabilities of (multi-modal) LLMs that can support human researchers in the study of historical ciphers. In the ideal world, the AI tool should be capable of processing the provided raw material (e.g., a scan or photography of a historical document), find regions of interest, transcribe the plaintext and ciphertext parts, translate the plaintext parts, provide statistical and contextual information about the ciphertext part, and possibly even try to solve the encrypted text. Our intention is also to provide an in-

¹Currently, several multimodal AI models support extended reasoning. However, when we began our research, ChatGPT was the only model suitable for our use-case.

sight into the current capabilities of ChatGPT. We structure this paper more as a tutorial, showcasing existing capabilities, and providing some tips on how to efficiently use the existing AI tools in historical cryptography research.

In Section 2, we introduce our dataset (a collection of cryptographic postcards) and define our cryptanalytic pipeline. Section 3 briefly summarizes the AI-related technological background of this research. The achieved results are described in Section 4. The main contributions, limitations of the analysed models, and future work are discussed in Section 5.

2 The Dataset and the Cryptanalytic Pipeline

The object of our research is a collection of historical encrypted correspondence. The largest, publicly available collection of 988 cryptographic postcards (including Tobias Schrödel’s collection (Schrödel, 2021)), is accessible on the HCPortal database² of cryptograms and cipher keys (Antal and Zajac, 2020; Antal and Zajac, 2026). These postcards date mainly from the early 20th century, and the encryption systems used are relatively simple and well-known. This makes them ideal as a challenge for processing with AI assistants.

Please be aware that because the dataset is accessible online, there’s a possibility that it may have been (in some way) included during the training of the investigated large language models. However, in most cases, the transcriptions or solutions to the cryptograms are not available in the database, and it’s not easy to obtain them from the images without additional processing. Moreover, there is no available procedure for solving the problems being examined.

2.1 Analysis of the Dataset

We systematically analysed the available digitized images with a focus on the used cipher system and on the used characters (ciphertext alphabet). Because the postcards may contain the ciphertext on the picture side, address side, or on both sides, we decided to manually process the collection per individual scanned image, where at least a portion of the ciphertext is present.

From the resulting 1010 images, the vast majority (836x) were encrypted using a substitution

cipher (mostly monoalphabetic). The substitution ciphers found can be further divided based on the character sets into the following categories: numbers only³ (160x), letters only (79x), symbols only⁴ (241x) and mixed alphabet (356x). The few transposition ciphers (10x) were mainly grammatical (mirror/reverse writing) or geometrical (text written in columns). The steganographic techniques (15x) were various, such as anamorphic writing, text hidden under the stamp, etc. The shorthand category contains classical shorthand messages and other *shorthand-like* ciphers. Similar to Schrödel (2021), we will leave them out of further analysis, because we were unable to clearly identify them. Morse codes are not ciphers; however, many times presented on the auction sites as codes/ciphers. The *other* category contains 10 (possibly) abbreviation ciphers, and 1 image, where some letters were omitted from the message. In 21 cases, we were unable to clearly determine the category.

2.2 Definition of our Cryptanalytic Pipeline

According to the results of our analysis (see Table 1), the covered cipher types are relatively easy to solve. A simple substitution solver can cover the decipherment of the vast majority of the postcards.

In more detail, we selected three main categories of postcards for further experiments (based on the difficulty of the solving process):

- A numeric substitution. The cipher key is constructed based on the lexicographical order of the plaintext alphabet letters. This is the easiest cipher type/task.
- A Pigpen cipher. Medium difficulty because the cipher is graphical/geometrical, and different patterns exist, but most of them are known. If the letters are randomly assigned to the Pigpen grids, the cipher decryption task is equivalent to the hard difficulty category.
- A general simple substitution (with symbols/letters/mixed alphabets). The hardest difficulty as the analysis usually requires statistical and pattern analysis.

Our main objective is not the decryption itself, but rather the study of the interpretation, transcription, and analysis capabilities of multimodal AI

²<https://crypto.hcportal.eu/>

³In many cases, the letters were replaced with numbers based on their lexicographical order.

⁴From which 61 are Pigpen.

Cipher type	Used characters				
	Number	Letter	Symbol	Mixed	N/A (other)
Substitution	160	79	241	356	
Transposition		10			
Steganography					15
Shorthand					100
Morse code			17		
Other		11			
Unidentified					21

Table 1: Cryptographic postcards statistics.

tools. Processing of the postcards requires visual analysis and transcription before the decipherment attempt. In this regard, we focus on the following tasks:

- Transcription of numeric substitution ciphers — processing handwritten numbers.
- Transcription and interpretation of the Pigpen cipher — processing symbols.
- Transcription of substitution ciphers that use mixed alphabets.
- Interpretation of transposition and steganographic techniques — processing letters and graphic shapes.
- Analysis of the transcribed text.

We acknowledge the fact that most of the ciphers used in our dataset are trivial and relatively easy to decipher either manually or using dedicated software. This is, however, the ideal situation to use AI tools in their cryptanalysis, for offloading and centralizing the set of tasks itself. The AI-powered analysis, interpretation, and transcription could help amateur codebreakers, enthusiasts, or postcard collectors without any cryptologic background, and without the need to find or construct specific tools for specific tasks.

While we would like to automate the entire task, our experimental pipeline consists of two main parts:

- Processing the scanned image (analysis, interpretation, and transcription).
- Processing the transcribed text (decipherment).

This helps us to assess the parts separately, while in practice, it is easy to connect these two parts to form a full analytic pipeline.

3 AI Tools and How to Use Them

The term Large Language Model (LLM), in a popular sense, has become synonymous with a chat interface driven by a generative transformer implementing a statistical representation of a large language model inside a complex neural network, such as ChatGPT. Moreover, the term LLM has been conflated with the term Artificial Intelligence (AI). Even if imprecise, we will also call these and similar tools and APIs either AI tools or simply LLMs.

Recent work distinguishes between *general-purpose LLMs* and *Large Reasoning Models* (LRMs) (Srivastava and Yao, 2025). General-purpose models, such as GPT-4o, handle diverse tasks through single-pass inference without explicit multi-step explanation. In contrast, reasoning models such as OpenAI o1 and DeepSeek R1 allocate additional computation to chain-of-thought processing, decomposing complex problems into intermediate steps.

Multimodal models are capable of processing multiple input modalities: text, images, and documents within a unified architecture (Lin et al., 2025). This capability enables direct analysis of scanned materials without requiring separate pre-processing or transcription steps.

Prompt engineering (Schulhoff et al., 2024) is a technique of crafting specific prompts to the chat interface of a generative transformer in such a way that the desired results have a high(er) chance to occur as a response of the transformer. Prompt engineering can also optimize the outputs of the reasoning models (Srivastava and Yao, 2025). Moreover, LLMs themselves can be tasked to optimize prompts for a given task, a technique where the model reformulates the user’s intent into a more effective prompt for itself or another model, or even tailors it based on a prompting strategy.

Common prompting strategies include: *Zero-shot*, *Few-shot*, and *Chain-of-thought* (CoT) prompting (Schulhoff et al., 2024).

Interactions with LLMs can be *single-turn*, where the model produces a solution in response to a single prompt, or *multi-turn*, where the solution emerges through iterative refinement based on user feedback. In multi-turn interactions, feedback may include corrections, hints, or additional constraints that guide the model toward the desired output.

Modern LLM platforms extend the base text-generation capabilities through integrated *tools*. These include code interpreters for executing code, web search for retrieving external information from allowed or specified sources, file input for processing uploaded files that the model cannot handle natively, and more. Tools transform pure text inference into assisted analysis, where the model can delegate specific subtasks to specialized components and build its context from their outputs.

Visual processing in multimodal models can operate in different modes (Lin et al., 2025). *One-pass visual perception* processes the image in a single forward pass without iterative refinement. *Active visual perception* involves iterative processing where the model revisits the image, considers alternative interpretations, and integrates visual evidence with contextual reasoning, optionally leveraging tools such as a code interpreter to preprocess the image (e.g., adjusting contrast or orientation). The latter is particularly relevant, as handwritten characters may be ambiguous.

LLMs can be accessed either through a conversational *chat interface* or programmatically via an *API* (Application Programming Interface). While chat interfaces are designed mostly for interactive use, APIs additionally enable automated and batch processing of requests. APIs also enable more detailed configuration of LLM’s system prompt or its response, which allows for advanced orchestration of multiple LLMs and custom tools for the LLM to execute.

3.1 ChatGPT Overview

ChatGPT is a conversational AI system developed by OpenAI, powered by models from the GPT (Generative Pre-trained Transformer) family.

The platform provides access to multiple model variants and reasoning configurations, as well as

integrated tools relevant to the tasks considered in this paper. In particular, multimodal capabilities allow the direct analysis of scanned or photographed historical documents, while advanced reasoning modes support exploratory statistical analysis, hypothesis generation, and cryptanalytic reasoning.

This subsection provides an overview of the operating modes, available model variants, and access mechanisms relevant to our experimental setup. The description is intended to clarify the technical context in which the experiments were conducted and to support the reproducibility and interpretation of the results presented in later sections.

Prior work demonstrates that prompt formulation significantly affects the behaviour of generative transformer models (Schulhoff et al., 2024), and that prompt engineering remains effective even for models equipped with explicit reasoning capabilities (Srivastava and Yao, 2025). Consequently, model variant, operating mode, and prompt design are treated as experimentally relevant variables rather than fixed implementation details.

3.1.1 Subscription Tiers

ChatGPT is offered through multiple access tiers that differ in usage limits, availability of reasoning modes, and access to integrated tools. At the time of the experiments, the available tiers included Free (with and without login), Go, Plus, Pro, Edu and Business (OpenAI, 2026a; OpenAI, 2026b). These distinctions are relevant for reproducibility, as certain reasoning modes and tools are restricted or rate-limited depending on the access level.

In particular, logged-in Free users can manually activate extended reasoning via the input interface, whereas anonymous users rely solely on automatic mode selection. Higher-tier plans provide increased usage limits and access to advanced reasoning configurations.

Table 2 presents the available models in ChatGPT. Recent research emphasizes a distinction between general-purpose large language models and models explicitly optimized for extended reasoning (Srivastava and Yao, 2025). While all modern transformer-based models exhibit some degree of implicit reasoning, reasoning-optimized variants allocate additional computation to multi-step inference. This distinction is particularly relevant for cryptanalytic tasks, which often

require hypothesis generation, iterative refinement, and validation rather than direct pattern completion.

Model	Ctx	Out
GPT-5.2 Auto	128–196k	32k
GPT-5.2 Instant	128k	32k
GPT-5.2 Thinking	196k	32k
GPT-5.2 Pro	196k	32k
<i>Legacy models</i>		
GPT-5.1 Inst./Think.	128k/196k	16k
GPT-5 Inst./Think.	128k/200k	16k
GPT-5 Think. mini	200k	16k
GPT-4o	128k	16k
GPT-4.1	1M	32k
o3	200k	100k
o4-mini	200k	100k

Table 2: Maximum context and output tokens for ChatGPT models (not API). Ctx: Context window (tokens), Out: Maximum output tokens.

GPT-5.2 is the current flagship model, optimized for professional knowledge work including creating spreadsheets, presentations, writing code, perceiving images, and handling complex multi-step projects. It demonstrates state-of-the-art performance in long-context reasoning, vision tasks (error rates cut roughly in half on chart reasoning and software interface understanding), and tool calling (OpenAI, 2026c).

3.1.2 Operating Modes

GPT-5.2, GPT-5.1 and GPT-5 models operate in different modes (auto, instant, thinking, pro) that control reasoning depth (OpenAI, 2026d).

The choice of operating mode is particularly consequential for tasks involving ambiguity or multi-step reasoning. Cryptanalytic analysis of historical material often requires evaluating multiple plausible interpretations of partially degraded or handwritten symbols prior to decryption. Operating modes that emphasize extended reasoning are therefore more appropriate for such tasks, whereas faster modes are primarily designed for single-pass processing.

3.1.3 Integrated Tools

ChatGPT provides a set of integrated tools (code interpreter, web search, file input) that extend its base text-generation capabilities. These tools are natively available through the chat interface and may be enabled or disabled depending on the task and access tier.

These tools are conceptually distinct from the

underlying language model and can substantially alter the nature of a task by transforming it from pure inference into assisted analysis. Consequently, their usage is treated as an explicit experimental factor in this study and is reported separately where applicable.

3.1.4 Model Selection

For the experiments presented in this paper, we selected GPT-5.2 in its Thinking mode as the primary model. This choice was motivated by two principal considerations: (1) GPT-5.2 represents the current flagship model with state-of-the-art performance in vision tasks and long-context reasoning, both critical for historical document analysis. (2) The selected configuration is accessible to free-tier users, ensuring that our experimental setup can be reproduced without requiring a paid subscription.

To assess the contribution of extended reasoning capabilities, we additionally evaluated GPT-4o, a legacy non-reasoning model that lacks chain-of-thought functionality. This comparison allows us to isolate the impact of explicit reasoning on cryptanalytic performance and the overall progression of LLMs.

Furthermore, given evidence that prompt engineering can influence model performance even in reasoning-enabled systems (Srivastava and Yao, 2025), we systematically evaluate both zero-shot prompts and optimized prompt formulations. This design allows us to separate inherent model capabilities from sensitivity to task framing.

Table 3 summarizes the configurations proposed for our experiments.

Model	Reasoning	Mode	Tools
GPT-5.2	Yes	Thinking	Enabled
GPT-4o	No	—	Enabled

Table 3: Selected GPT model configurations.

4 Results of Our Experiments With AI Tools

We evaluated the two selected GPT models (see Table 3) on the two main logical parts of our pipeline, defined in Section 2.2.

4.1 Image Processing Capabilities

First, we focused on the analysis of the GPT models’ image processing capabilities of the scanned images.

4.1.1 Numeric substitution

In the first experiment, we analysed whether it is possible to successfully transcribe handwritten numbers. We evaluated the address side of the *pc_hcp_191* postcard⁵ from HCPortal. It contains twelve lines of ciphertext encrypted with a simple substitution using numbers as ciphertext characters. Moreover, each number is separated by a dash, and individual words are also separated by a comma. We prompted the models to transcribe the text from the postcard without providing any further details.

GPT-4o transcribed the text with a lot of transcription errors and without noticing the commas. Therefore, the result is only partially usable in further decipherment attempts. Communication 1 is accessible from the Appendix.

GPT-5.2 handled the transcription quite well, including the word separators. It made only a few mistakes, only in the areas where the handwriting was not clear or covered by other printed parts on the postcard. Communication 2 is accessible from the Appendix.

This experiment showed that there is no need for any prompt engineering for the transcription of handwritten numbers. Both models identified the used cipher without asking for it, and *GPT-5.2* achieved an almost correct transcription. We recommend using it for this task.

4.1.2 Substitution with symbols and mixed alphabet

The real challenge, however, remains whether we can also transcribe texts where various symbols and glyphs were used. Transcription of a ciphertext containing such a large variety of strange symbols is not trivial.

First, we focused on an easier cipher instance – on a Pigpen cipher, where a regular (known) Pigpen grid was used. We evaluated the address side of the *pc_hcp_135* postcard⁶ from HCPortal. It contains six lines of ciphertext encrypted with a classic Pigpen cipher and one additional row above the receiver’s address. To read the ciphertext correctly, it is necessary to flip the postcard upside down. We asked the models to transcribe the postcard.

While *GPT-4o* transcribed the address (handwritten plaintext) without problem, the cipher

symbols were transcribed incorrectly. The model tried to transcribe it to some Unicode graphical symbols with similar shapes as used in the Pigpen cipher; however, the transcription, the number of transcribed symbols, and the number of lines are incorrect. The postcard’s correct orientation was also not identified. Communication 3 is accessible from the Appendix. Even when we constructed a detailed prompt with specific instructions, it was unable to correctly interpret and transcribe the text. The only property identified correctly was the flipped orientation of the postcard. Communication 4 is accessible from the Appendix.

GPT-5.2 handled the transcription much better (after specially asking for the transcription of the ciphertext). The model tried to transcribe and represent the cipher symbols with Unicode shapes (differently from *GPT-4o*). It tried to construct the shapes with their base shape and with an additional bullet (if it was required). It also recognized whether the bullet is inside or after the shape. The encoding looks more precise and is almost correct (with a few errors). However, the problem with wrong image orientation is still present. Next, we specified in the prompt that only Pigpen characters were used, asked to check the image orientation, and informed the model about the transcription errors. Except for one additional bullet and one mistake in the first line, the transcription was correct. The result is satisfying and can be potentially used in cryptanalysis. Communication 5 is accessible from the Appendix.

Next, we focused on a harder cipher instance – on a substitution cipher using a mixed alphabet of letters, numbers, and symbols. We evaluated the address side of the *cpc_1919-02-07_USA_Dubois_USA_Springville* postcard⁷ from HCPortal.

GPT-4o was unable to correctly transcribe the ciphertext, whether we used simple or detailed instructions in the prompt. Communications 6, 7, and 8 are accessible from the Appendix.

GPT-5.2 handled the transcription much better, but not without errors. Mostly, it mistaken number 4 (or a symbol with a similar shape to number 4) as 7, number 3 (almost closed upper circle in number 3) as 9, etc. Despite these errors, we can use the partially correct transcription to (at least partially)

⁵<https://crypto.hcportal.eu/dashboard/cryptograms/1510>

⁶<https://crypto.hcportal.eu/dashboard/cryptograms/1454>

⁷<https://crypto.hcportal.eu/dashboard/cryptograms/944>

solve the message. Communication 9 is accessible from the Appendix.

We can conclude that a ciphertext containing uncommon symbols is problematic to transcribe in general. The transcription success depends on the symbol set used and on the quality of the input image.

4.1.3 Transposition (standard letters with unconventional text-writing direction)

In this experiment, we focused on analysing the transcription capabilities of handwritten text, which was written in an unconventional manner. We evaluated the address side of the *pc_hcp_508* postcard⁸ from HCPortal. It contains relatively easy-to-read handwritten text written in columns from left to right, instead of rows (similarly to the encryption process of the Scytale cipher), followed by three columns of similar characters, rotated by 90 degrees. The words were separated with a separator character (X). Moreover, for some columns, the writing style (orientation of the letters) at the end of the column was changed. The address and name of the recipient are also written in a special way. We prompted the models to identify what is on the input image without any additional context, followed by a second prompt, where we specified to transcribe the whole postcard and describe every component.

The *GPT-4o* transcribed most of the vertical columns correctly, including the recipient details, even the details from the post stamp. However, it was unable to read the rotated columns located at the bottom of the image. Communication 10 is accessible from the Appendix. In general, the model can handle unusual text writing directions with a couple of errors. The biggest problem for this model is to deal with the rotated part of the text.

We repeated the experiment with the *GPT-5.2* model. The result was almost perfectly correct. This model sequentially crops, adjusts, and analyses the image parts to achieve the best results. The model also identified that letter X was used as a word separator and can identify typos in the written text (made by the scribe). After we specified that some lines were rotated by 90 degrees, the model rotated and successfully read the rotated lines (correctly), too. Communication 11 is accessible from the Appendix. This model is

very suitable for analysing and transcribing handwritten text, even with different writing directions or when the text writing direction is changed halfway. There was no need for special prompts (except for the text rotation). The achieved result is acceptable with only minor typos.

4.1.4 Steganography (anamorphic writing)

In this experiment, we focused on analysing the image transformation capabilities of the models. We evaluated the picture side of the *pc_hcp_140* postcard⁹ from HCPortal. It contains a special text writing style/ text deformation technique – called anamorphic writing. The meaningful text is visible (in most cases) only if the image is stretched. The text on the postcard is also rotated by 180 degrees.

We prompted the *GPT-4o* model to transcribe the image with a hint that it may be rotated. The model was not able to identify the used technique nor interpret it correctly. Communication 12 is accessible from the Appendix. In the next step, we used a more sophisticated prompt, including detailed instructions. The model decided to use a series of geometric transforms (non-uniform scaling, rotations, and optional blur + thresholding) in an attempt to recover readable characters. The proposed result was still incorrect. Communication 13 is accessible from the Appendix.

The *GPT-5.2* model was also unable to identify the writing technique used and to transcribe the text without providing detailed instructions. Communication 14 is accessible from the Appendix. Next, we used the extended prompt with the instructions. After the model was unable to OCR the whole text successfully, it started to segment the image into word-sized chunks and repeated the OCR. We obtained only a partially correct result, despite the model presenting the result with a very high confidence. Communication 15 is accessible from the Appendix. Despite the specific instruction in the previous prompt, we were able to obtain the correct transcription only after we extracted the transformed image produced by the model during its reasoning process and used it as input (still rotated by 180 degrees). The model was able to identify and transcribe the correct text without the need for additional details. Communication 16 is accessible from the Appendix.

⁸<https://crypto.hcportal.eu/dashboard/cryptograms/1827>

⁹<https://crypto.hcportal.eu/dashboard/cryptograms/1459>

4.2 Decipherment

As per our pipeline definition, the initial step is to obtain the transcription of the ciphertext from the postcard, which will then enable us to proceed with the decipherment process. In the following, we present decipherment attempts of selected substitution ciphers (in cases where we were able to obtain a suitable transcription in the previous step).

4.2.1 Numeric substitution

We continue our experiment with the *pc_hcp_191* postcard. Because the *GPT-4o*'s transcription was not correct, we were also unable to obtain the correct decryption even if the model correctly guessed the used cipher type and key. Only part of the result (a few words) was correct.

We prompted the *GPT-5.2* model to decipher the numbers after receiving the transcription. The model successfully identified¹⁰ the used cipher type and key. The decipherment was successful, and only a few small mistakes occurred. The model was also able to correct most of its transcription errors during the decipherment (probably based on the context analysis of the resulting text). Communication 17 is accessible from the Appendix.

This experiment showed that only *GPT-5.2* is suitable for cryptanalysis of this cipher type.

4.2.2 Substitution with symbols and mixed alphabet

Because only *GPT-5.2* was able to successfully transcribe the *pc_hcp_135* postcard, we focused only on this model. First, we prompted the model to transcribe the Pigpen cipher. After the initial transcription, we asked the model to correct the (few) errors and decipher the ciphertext. It was also necessary to specify that the text was flipped. The model was unable to process our request without providing additional information. It provided us with the following options:

- Specify if the sender used the standard Pigpen alphabet.
- Provide if the stand-alone dots between groups are meant to be spaces.
- Provide some cribs.

¹⁰As an A1Z26 cipher: 1 = A, 2 = B, ..., 26 = Z (hyphens = letters, commas = word breaks).

We decided to provide the correct key as an input image, instead of selecting one from the previous option. The resulting plaintext was fully correct. Communication 18 is accessible from the Appendix. We also tried a different approach – we instructed the model to search for the key. At the beginning, the model summarized various known Pigpen cipher keys (mainly from Wikipedia); however, it was not able to find/identify the correct key and provided an incorrect result. When we asked to use a different key, it found the correct Pigpen variant from the dCode website. In this phase, it even tried to correct the misspelled words. Communication 19 is accessible from the Appendix. Then we tried a new approach with a more detailed initial prompt. When the model asked us for user input, we instructed the model to try to brute-force the cipher key. Unfortunately, it was unable to find the correct key in this way. We can deduce from our experiments that if we don't provide the possible cipher key, or it can't be found over the internet (we still need to specifically ask the model to search for it), or a randomly assigned Pigpen grid was used, this cipher type is not solvable with GPT. Communication 20 is accessible from the Appendix.

In our last experiment, we tried to decipher the substitution cipher with a mixed alphabet. The *cpc_1919-02-07_USA_Dubois_USA_Springville* postcard was transcribed only partially. Despite this fact, we decided to try to decipher it with *GPT-5.2*. We continued in the previous communication and asked the model to solve the transcribed text. It was able to correctly decrypt five words. From the result, it is possible to manually obtain the whole solution. Communication 21 is accessible from the Appendix.

5 Conclusions

In this work, we explore the capabilities of ChatGPT in assisting with the processing, interpretation, and decipherment of historical encrypted documents. We evaluated two multimodal GPT models: *GPT-4o* and *GPT-5.2*. Our case study has shown that *GPT-5.2* is suitable to process and solve various historical ciphers directly from an image input (of a cryptographic postcard).

Our approach uses visual and reasoning features of ChatGPT, and not its ability for text completion. Even though some of the used examples could

have been seen by the model during the training, it is clear from the achieved results (reasoning traces and various mistakes) that the model isn't affected by the online database and really was supported by long context reasoning, multimodality (visual processing), and tool calling.

With correct prompting, the AI tool is able to automatically apply transcription, image interpretation, and cryptanalytic tasks. We were able to successfully transcribe or interpret most of the investigated postcards, as well as decipher their contents. Moreover, in the case of easier instances, any special prompt engineering was completely unnecessary. Note that the older model, GPT-4o, is behind GPT-5.2 in every aspect, and was only rarely useful for our tasks.

One notable drawback of the GPT-5.2 model is its extended duration in achieving the required outcomes during our experiments, especially when it is performing image processing operations. For example, when processing Pigpen experiments, the initial prompt took approximately 15–25 minutes to process, with subsequent prompts often taking 30 minutes or more. This was mainly due to the model's reasoning/thinking and subsequent correction of its own mistakes (mainly image processing tasks). In some cases, the context window proved insufficient, leading to timeouts before reaching the final output.

For optimal results when using AI tools for historical cipher analysis, we recommend following these guidelines:

- **Image orientation.** Explicitly instruct the model to check and correct image rotation before processing. Printed elements (e.g., “POST CARD”, address fields) can serve as reference points, but note that the ciphertext may have a different orientation than the printed text, so the model should try multiple orientations.
- **Contextual framing.** Provide as much context as possible: specify that the input is a postcard or personal letter, mention expected conventions such as salutations, closings, period-typical abbreviations, common phrases, and likely language. The more contextual information provided, the better the transcription and decipherment accuracy. If the cipher type is known, specify it explicitly.
- **Distinguish cursive from symbols.** Stylized

cursive handwriting with flourishes is often misinterpreted as a symbolic cipher. Explicitly state that the text is handwritten in Latin script to prevent this confusion.

- **Manual reading.** The observed default behaviour of the model is to use automated OCR/computer vision, which often fails for symbolic ciphers and cursive text. Explicitly instruct the model to skip OCR and read characters manually from the start. The model can still use code tools for image enhancement, cropping, or frequency analysis.
- **Systematic processing.** For ciphertexts with many symbols or characters, instruct the model to process row-by-row or column-by-column rather than attempting one-shot transcription. Additionally, instruct the model to count and verify the number of characters per line to ensure the transcription matches the image.
- **Symbol inventory.** Request the model to create an explicit legend or inventory of all distinct symbols identified in the ciphertext. This provides the model with a consistent base for symbol interpretation throughout the transcription and allows the user to inspect and correct the model's symbol recognition. Alert the model to easily confused symbols: mirrored shapes, rotational variants (“6” vs. “9”, “d” vs. “p”), and similar corner orientations.
- **Known plaintext.** If any portion of the plaintext is known, provide it to the model — this dramatically improves decipherment accuracy.
- **Cipher-specific considerations:**
 - *Numeric substitution* (lexicographic): Minimal prompting is sufficient; transcription errors often self-correct during decoding.
 - *Pigpen*: Only standard keys are recognized; variant keys must be provided or the model instructed to search for alternatives. Verify the model's key interpretation, as symbol orientations are frequently confused.
 - *Transposition*: Explicitly specify reading direction (columnar vs row-based).

- *Symbol/mixed substitution*: Higher error rates due to glyph variety and similar-looking characters; row-by-row processing and iterative correction recommended. Context exhaustion and timeouts are common for complex instances.

Final research notice: This article was constructed as a case study, with the aim to help in automating specific tasks in historical cipher research. In addition to the examples mentioned above, we have conducted other experiments that are not included in this article. Our recommendations are based on the experience gained from all our experiments. We do not provide a full statistical evaluation of current AI capabilities, and we have not examined the capabilities of other AI models than GPT-4o and GPT-5.2. Furthermore, we are aware that it is appropriate and worthwhile to conduct a more systematic comparison with other models as well. Therefore, we are working on an expansion of our research, where we are comparing GPT with other AI systems, such as Claude or Gemini. The achieved results will be published in a future publication.

Acknowledgments

This research was supported by the project "Artificial intelligence for encrypted handwritten document processing", "09I05-03-V02-00031" of call "Support of research projects aimed at digitization of the economy in TRL levels 1-3", Call. No. 09I05-03-V02, managed by the Research Agency and funded by The Recovery and Resilience Facility of the Slovak Republic.

References

- S Akilesh, Rajeev Sekar, R Abinaya, K Palani Thannaraj, and C Christopher Columbus. 2026. Zero-shot chain-of-thought reasoning for cryptanalysis of popular ciphers. In *2026 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECs)*, pages 1–6.
- Eugen Antal and Pavol Zajac. 2020. HCPortal Overview. In *Proceedings of the 3rd International Conference on Historical Cryptology HistoCrypt 2020*, pages 18–20. Linköping University Electronic Press.
- Eugen Antal and Pavol Zajac. 2026. HCPortal: Ten Years of Development. In *Proceedings of the 9th International Conference on Historical Cryptology HistoCrypt 2026 (accepted)*.
- Yu Li, Qizhi Pei, Mengyuan Sun, Honglin Lin, Chenlin Ming, Xin Gao, Jiang Wu, Conghui He, and Lijun Wu. 2025. CipherBank: Exploring the boundary of LLM reasoning capabilities through cryptography challenge. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 5929–5965, Vienna, Austria. Association for Computational Linguistics.
- Zhiyu Lin, Yifei Gao, Xian Zhao, Yunfan Yang, and Jitao Sang. 2025. Mind with eyes: from language reasoning to multimodal reasoning. *arXiv preprint arXiv:2503.18071*.
- Utsav Maskey, ZHU Chencheng, and Usman Naseem. 2025a. Benchmarking large language models for cryptanalysis and side-channel vulnerabilities. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 19849–19865.
- Utsav Maskey, Chencheng Zhu, and Usman Naseem. 2025b. Benchmarking large language models for cryptanalysis and side-channel vulnerabilities. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 19849–19865, Suzhou, China. Association for Computational Linguistics.
- David Noever. 2023. Large language models for ciphers. *International Journal of Artificial Intelligence & Applications*, 14:1–20.
- OpenAI. 2026a. Chatgpt enterprise and edu — models & limits. Accessed: 2026-01-12.
- OpenAI. 2026b. Chatgpt pricing. Accessed: 2026-01-12.
- OpenAI. 2026c. Introducing gpt-5.2. Accessed: 2026-01-12.
- OpenAI. 2026d. Using the latest model (gpt-5.2) — api guide. Accessed: 2026-01-12.
- Tobias Schrödel. 2021. Cryptographic postcards. In *Proceedings of the 4th International Conference on Historical Cryptology HistoCrypt 2021*, pages 131–136. Linköping University Electronic Press.
- Sander Schulhoff, Michael Ilie, Nishant Balepur, Konstantine Kahadze, Amanda Liu, Chenglei Si, Yinheng Li, Aayush Gupta, Hyojung Han, Sevien Schulhoff, Pranav Sandeep Dulepet, Saurav Vidyadhara, Dayeon Ki, Sweta Agrawal, Chau Pham, Gerson C. Kroiz, Feileen Li, Hudson Tao, Ashay Srivastava, Hevander Da Costa, Saloni Gupta, Megan L. Rogers, Inna Goncareenco, Giuseppe Sarli, Igor Galynder, Denis Peskoff, Marine Carpuat, Jules White, Shyamal Anadkat, Alexander Miserlis Hoyle, and Philip Resnik. 2024. The prompt report: A systematic survey of prompting techniques. *CoRR*, abs/2406.06608.
- Saurabh Srivastava and Ziyu Yao. 2025. Revisiting prompt optimization with large reasoning models — a case study on event extraction. *arXiv preprint arXiv:2504.07357*.

Yu Wang, Yijian Liu, Liheng Ji, Han Luo, Wenjie Li, Xiaofei Zhou, Chiyun Feng, Puji Wang, Yuhan Cao, Geyuan Zhang, Xiaojian Li, Rongwu Xu, Yilei Chen, and Tianxing He. 2025. AICrypto: A Comprehensive Benchmark For Evaluating Cryptography Capabilities of Large Language Models. *CoRR*, abs/2507.09580.

Appendices

Communication 1:

<https://chatgpt.com/share/697aabad-25a0-8013-8c7f-25fd063f12ba>

Communication 2:

<https://chatgpt.com/share/697aaa83-fadc-8013-9227-2a54b85d78ff>

Communication 3:

<https://chatgpt.com/share/697bd6c1-1254-8013-a2ec-e67ec44e2e97>

Communication 4:

<https://chatgpt.com/share/697be07c-b008-8013-9a5e-28bb474db70c>

Communication 5:

<https://chatgpt.com/share/697e416f-aef4-8013-a808-e869369bfedf>

Communication 6:

<https://chatgpt.com/share/697c090f-aab0-8013-8838-0fa9da2a391f>

Communication 7:

<https://chatgpt.com/share/697c094e-c9dc-8013-a09a-ab04a603e041>

Communication 8:

<https://chatgpt.com/share/697c08c6-338c-8013-b472-cc750c2fafde>

Communication 9:

<https://chatgpt.com/share/697c1060-6540-8013-8745-7780ea71c41d>

Communication 10:

<https://chatgpt.com/share/697a91a9-e728-8013-b137-b85a43844515>

Communication 11:

<https://chatgpt.com/share/697a9a94-6b78-8013-ae27-5d2d3c32c26b>

Communication 12:

<https://chatgpt.com/share/697b53dd-9e28-8013-9e8a-ec74cd753da5>

Communication 13:

<https://chatgpt.com/share/697b513e-8658-8013-b7f7-117f8da8dc92>

Communication 14:

<https://chatgpt.com/share/697b553c-7ce0-8013-a071-82bc172cad6a>

Communication 15:

<https://chatgpt.com/share/697b5187-2fa4-8013-b3dd-8fce1c0b87e0>

Communication 16:

<https://chatgpt.com/share/697aa80f-7694-8013-8c68-c99926be0b11>

Communication 17:

<https://chatgpt.com/share/697aaa83-fadc-8013-9227-2a54b85d78ff>

Communication 18:

<https://chatgpt.com/share/697c036c-ac9c-8013-b390-8c621ccae5b0>

Communication 19:

<https://chatgpt.com/share/697c0ae7-4c0c-8013-9814-8557d24ce43c>

Communication 20:

<https://chatgpt.com/share/697c1272-f884-8013-b2af-965babeccb50>

Communication 21:

<https://chatgpt.com/share/697c1060-6540-8013-8745-7780ea71c41d>