Integrating Large Language Models in Structural Data Processing in Hungarian Civil Registers

Kata Ágnes Szűcs, Noémi Vadász, Zsolt Záros, Emese Varga, Zoltán Szatucsek
National Archives of Hungary, Budapest

Introduction

Civil registers represent the most accurate source of demographic data for the 19th and 20th centuries. By the mid-20th century, demographers had already recognized that combining census information with the individual-level data found in vital records enables analyses that would otherwise be impossible using contemporaneous statistical surveys alone. However, manual family reconstitution proved labor-intensive, limiting researchers to selected settlements and excluding migration studies.

Recent advances in neural network-based data processing provide archivists and historians with new tools to extract large volumes of information from previously unprocessable documents. Additionally, growing public interest in genealogy has created demand for more accessible archival records, motivating the application of artificial intelligence in archival practice.

The Hungarian National Archives has preserved the security copies of state civil registers since civil registration began. The collection consists of 38,353 volumes, extending over 1,918 linear meters and containing approximately 23 million pages of vital record entries from 1895 to 1980. Digitization of these documents, previously only partially microfilmed for data protection, began in 2024. In response to evolving research expectations and societal interest in family history, the Archives aims to preserve the material and enable structured, machine-readable processing of the records.

The Transkribus (Louise Seaward, 2019) platform for handwritten text recognition has accelerated the digital processing of historical documents and inspired new solutions in data extraction. While many approaches have shown promise, the heterogeneity of historical documents limits the development of universally applicable solutions (Ari Vesalainen, 2025), shifting the focus toward modular pipelines for systematic evaluation.

The National Archives of Hungary developed and successfully run a modular workflow to process civil registers of birth, marriage, and death (of four municipalities in Pest county), transferring data from digitized images to an SQL database using open-source and machine learning tools. The process includes preprocessing, layout classification, handwriting recognition, and post-processing before publishing the structured data online.

Processing historical sources presents challenges. In civil registers, difficulties extend beyond archaic expressions and inconsistent spelling. Registry entries often contain distinct pieces of information within a single continuous "composite field," [1] typically in an unstructured format. Critical data elements – such as personal names and occupations – are frequently recorded consecutively, without explicit delimiters. Semantic segmentation of these fields is essential for identifying individuals and their attributes. [2]

A composite field can be broken down into constituent elements to a certain extent through rule-based methods. However, the effectiveness of such decomposition is dependent upon both the volume and quality of the information compressed within a given cell, as well as the cell's physical dimensions and its position on the document. For example, in narrow cells located along the margins of a page – where information such as the names and places of residence of witnesses on a marriage register is recorded – the data often appears in a more fragmented form, distributed across six to eight lines. In these cases, it is difficult to determine, on a rule basis, which elements of which rows belong together and form a coherent data set. The inherently structured nature of the content (e.g., surname and given name, municipality name and address) further complicates the correct segmentation into subcomponents.

By contrast, birth registers present a different type of overloaded field in which a single cell contains the child's name, sex, and religion. Among these data elements, gender and religion and even given names are comparatively straightforward to process using rule-based approaches, partly because the larger physical space available within

[1] The nomenclature of such field varies between overloaded, complex, composite etc. referring to its state of containing several type of often complex information

[2] For example, birth registers may contain a cell with the child's first name and another with both parents' names and occupations, necessitating breakdown into individual data elements for post-processing.

these cells permits clearer separation of the entries. As a result, in practice, these three data types are typically represented in the database as three isolated rows. The latter two elements – gender and religion – are especially amenable to rule-based handling, as they are consistently defined and well delimited.

Notwithstanding the potential benefits of the rules-based approach, it is also accompanied by certain challenges and limitations. Specifically, out of a total of 17 fields in the database that are considered overloaded, 9 fields have been addressed through the application of rule-based methods, while 2 fields have been partially addressed.

One of the biggest challenge is the fragmentation of data to several rows. Specifically, each row of a multi-line field may correspond to a different data element, the content of an element may be fragmented across multiple rows, or conversely, the content of multiple rows may collectively comprise a single element. Furthermore, the complex data fields can appear in different types of civil records, such as death, birth, and marriage records, and in various layout types, making the processing more challenging. Some composite data fields can appear in multiple types of records and layouts, with different information included. To address these structural and content-related inconsistencies, we explored the potential of large language models (LLMs) to support the decomposition and structured processing of complex fields.

LLMs for Processing Structural Data

This approach leverages language models to decompose complex, multi-line fields into structured components using well-formulated prompts. Initial experiments showed that the model not only effectively decomposed data but also enhanced the output quality of the handwriting text recognition (HTR) system.

Currently, we are testing the treatment of composite fields and monitoring intermediate results using examples from the SQL database. We have remote access to the Cloud service at the HUN-REN Wigner Research Centre for three months, utilizing a setup with 520 GB SSD storage and four NVIDIA A100 40GB video cards to execute the language models. We are obtaining models from the Ollama interface and testing the latest version of Llama 3.3.

The experimental setup uses the four video cards in parallel for simultaneous processing. We are also developing a benchmark system to measure model performance, aiming to create a large benchmark corpus for each overloaded field with input from row-split HTR hypotheses and output in JSON format.

Initially, we evaluated prompt text using various models on the www.duck.ai platform, including OpenAI's GPT-4o mini and Anthropic's Claude 3.5 Haiku. The token counts for these models are lower than those from the Ollama platform, which may affect performance. The Duck.ai platform, while free, imposes daily usage limits, slowing down testing due to high token counts. Despite the limitations of web-based testing, valuable insights were gained into the models' response patterns, allowing us to assess their performance. GPT-4o mini and Llama 3.3 emerged as the top performers, highlighting the benefits of testing multiple models.

For local testing, we employed the configuration with a temperature of 0.5, a fixed top_p value, and a seed parameter set to 42to control randomness and ensure reproducibility. Three models were tested: gemma3:27b, qwen3:235b, and llama3.3:latest.

The prompt generator algorithm queries the SQL database based on a predefined statement and appends relevant prompt text. Currently executed manually, it will eventually automate the interaction between the LLM and the database by passing selected overloaded fields to the LLM and reinserting the output into the database. This will enable automated processing of large data volumes, combining predefined prompts with retrieved content and invoking the language model via an API.

Experiments with Prompt Text

The structure of the generated examples is uniform across all fields and comprises three distinct components. (1) The name of the image file and the line number on the page from which the textual data originates. (2) A list of hypotheses, broken down into their corresponding rows. (3) Bulleted text providing instructions for the LLM to handle the list of hypotheses. Additionally, a specified JSON file structure outlines the expected output for the Large Language Model.

To minimize potential misinterpretations, communication with the language model is conducted in Hungarian, ensuring a clear and unambiguous exchange of information. However as an example, in case of the mother's name, occupation and residence field the input and the prompt in English would be as follows:

First part: Data Input
File name: %fajl_mezo%HU_MNL_PVL_XXXIII_0001_a_0001_c_0007_0221—1
This is a multi-line field with the mother's data:
1. row: ['P. Bály Istvánné született Kaden-', 'V. Bály Istvánné Kaden-', 'V. Bály Istvánné született Keden-']

2. row: ['szki Anna, férje lakása', 'ski Anna, férje lakása', 'gki Anna, férje lakása', 'szli Anna, férje lakása', 'szki Anna, férjelakása']

Label the field content.

Second part: Prompting the Labeling Rules

1. The labels can be: mother_name, mother_occupation, mother_residence.

2. The labels may not appear in this order in the field, and it is possible that not all labels have corresponding data.

3. One row can contain multiple labels.

4. A label can be split across multiple rows due to line breaks or other delimiters.

4.1. The "-" and "¬" characters indicate line breaks or delimiters.

4.2. If there are line breaks or delimiters, concatenate the words without spaces before labeling.

5. The labels should be assembled sequentially based on the row numbers.

6. A label should not be moved to a subsequent row if it belongs to the current row.

7 Lists in rows represent different hypotheses for that row. Each word may be correct in one hypothesis but not in another.

7.1. Use all words, even if they are not correct in any hypothesis.

7.2. Collect the best version of each word from the hypotheses.

7.3. Do not discard any words, including those that are incorrect.

8. The mother's name can consist of multiple surnames and given names (e.g., married woman with husband's surname). All personal names belong to the "anya neve" (mother's name) label.

9. The mother's last given name and residence are often found on the same line.

10. The frequency of certain locations (e.g., Abony) may help inform the labeling.

11. The number of words in the hypotheses can help determine the original number of words in the field, which should match the number of words in the labeled output.

12. Finally, correct any typographical errors in the labels (e.g., correct given names more liberally, surnames more conservatively).

Third part: Expected Output Format

No justification is needed just reply with a JSON in this format:

'mother_name': 'data', 'mother_occupation': 'data', 'mother_residence': 'data'

Evaluation of Performance

When the model is executed with the –verbose option, detailed performance metrics are obtained, including How efficiently the model processes input prompts and generates responses and by what rate does the model evaluates tokens and generates output. In the case of our previous example, the performance of the llama3.3:latest model is as follows in Table 1:

Metric	**Value**
Total duration	5.51s
Load duration	29.54ms
Prompt eval count	935 token(s)
Prompt eval duration	2.24s
Prompt eval rate	417.56 tokens/s
Eval count	53 token(s)
Eval duration	3.24s
Eval rate	16.38 tokens/s

Table 1: Performance metrics

The model's total processing time for the input prompt and response generation was reasonable, given the task's complexity and the size of the input. The quick loading time indicates efficient memory loading. The evaluation rate suggests that the model is effective in processing tokens. In contrast, the time required for generating the response was longer than the prompt evaluation, indicating a more computationally intensive part. The evaluation rate for output generation suggests potential inefficiencies in generating responses. It appears that the response generation phase is a bottleneck in the model's performance, as the evaluation rate for this step is significantly lower than that for prompt evaluation, indicating challenges in efficiently producing output. For more details, see Table. 1

To improve the performance of the model we are currently testing models with different architectures, such as the Mixture of Experts (MoE), since it consists of multiple expert models, each specialized in a specific task

or subset of the input data, and giving it task-specific data as context information to reduce the computational intensity of generating output. Overall, the model's performance is reasonable for testing, but there may be room for improvement, particularly in the response generation process.

Key Findings and Insights

This chapter presents the initial findings and insights gained from the execution of instructions extrapolated by the model, highlighting the performance and capabilities of the large language model in processing and responding to specific prompts.

Our experiments yielded several important observations. Firstly, we found that models with higher token numbers tend to perform better, as they are less constrained by the context of the field and can handle more complex prompts. Typical errors encountered during testing included name separation, disambiguation issues and address and occupation processing difficulties. It was a significant challenge to separate personal names, especially when data was fragmented across multiple rows. This issue was exacerbated when place names, such as streets named after individuals, were present, leading to inaccurate tagging. Distinguishing between a mother's name and a municipality was a recurring problem, with models struggling to separate personal names from toponyms. While incorporating common place names into prompts provided some assistance, it is not a sustainable solution for generalization across different locales. Models also mislabelled parts of married women's names as occupations, indicating a need for improved disambiguation strategies. Models faced challenges with non-standard address formats, often focusing solely on municipality names and omitting other critical details. They struggled with historical occupations, sometimes completely omitting them instead of labeling them as blank. Incorrect hypotheses for occupations were common, and models occasionally provided inaccurate responses when questioned. Furthermore, we observed that when we had to go in to details in the prompt text and the instructions became overly long explaining minor details, it can lead to oversight and decreased performance across all models.

A viable solution to some of our challenge was to utilize locally run models with a larger number of tokens, which are more capable of handling complex tagging tasks with fewer errors processing nuanced data. However, these difficulties highlight the need for further refinement and development to improve the model's performance in processing address and occupation information.

Good practices and Conclusions

Our study demonstrates the potential of LLMs in processing structural data in multi-line and multi-hypothesis environments. Building on the success of various NLP tasks (Aakanksha Chowdhery, 2022, Hugo Touvron, 2023) our preliminary results also show that LLMs can effectively separate and tag complex structural data, making them a valuable tool for processing civil records. To further improve the performance of LLMs, we would like to carry on testing models with MoE architecture. Giving task-specific data as context information (such as occupation list, given name list, religion list, etc). Pathways, as described in the PaLM article, uses a similar concept to MoE yielding good results.

Finally, to appropriately measure the performance of the applied LLMs, we would like to create our own benchmark datasets and further research the evaluation metrics mentioned in Jiawei Chen, 2023.

References

Aakanksha Chowdhery, e. a. (2022). Palm: Scaling language modeling with pathways. https://arxiv.org/abs/2204.02311

Ari Vesalainen, e. a. (2025). Creating a historical migration dataset from finnish church records, 1800-1920. https://arxiv.org/abs/2506.07960

Hugo Touvron, e. a. (2023). Llama: Open and efficient foundation language models. https://arxiv.org/abs/2302.13971

Jiawei Chen, e. a. (2023). Benchmarking large language models in retrieval-augmented generation. https://arxiv.org/abs/2309.01431

Louise Seaward, e. a. (2019). Transforming scholarship in the archives through handwritten text recognition: Transkribus as a case study. *Journal of Documentation*, *75*(5), 954–976. https://doi.org/10.1108/JD-07-2018-0114