Text to Data: A Machine Learning Approach to Historical Chinese Documents

Pan Chen[1], Wolfgang Keller[1,3], Carol H. Shiue[1,3], and Sen Yan[*2]

[1]Department of Economics, University of Colorado, Boulder
[2]Leeds School of Business, University of Colorado, Boulder
[3]NBER and CEPR

March 20, 2025

Abstract

This paper introduces a large language model (LLM) designed for automated data extraction from historical Chinese documents. We apply this approach to Chinese family genealogies, focusing on the Ma clan of Tongcheng, which spans 20 generations from the 15th to 20th century. By fine-tuning OpenAI's ChatGPT-4o, we extract structured demographic and socio-economic data from biographical entries and evaluate its accuracy against manually curated data. Our results show that LLM-based extraction dramatically improves efficiency while maintaining high accuracy across standard evaluation metrics. Fine-tuning ChatGPT-4o increases classification accuracy by more than 20 percent compared to baseline models, enhancing precision in data retrieval. Our model outperforms manual data extraction especially for complex variables requiring historical context, such as social status. The approach also improves adaptability for historical text analysis while ensuring replicability. By integrating AI with traditional economic history research methodologies, this study advances scalable, high-accuracy historical text analysis, providing a foundation for new economic insights drawn from historical documents.

[*]We thank audience members at the First CEPR Applied Micro-AI Conference (Heidelberg, Feb 2025) for useful comments. A website with access to the large language model developed in this paper is http://3.145.178.72:8501/.

1 Introduction

Artificial intelligence (AI) and large language models are transforming the way data is extracted from historical and archival records. Traditionally, historians and archivists have manually analyzed vast collections of documents, a time-consuming process that demands expertise in multiple languages, scripts, and historical contexts. Advances in AI-driven natural language processing and optical character recognition now enable the automated digitization and analysis of these sources, making research more efficient and scalable. This paper examines the application of these technologies to Chinese family genealogies, a rich source for investigating long-term economic processes through intergenerational family linkages.

This analysis examines the Ma clan of Tongcheng, a county in central China's Anhui province. In scholarly literature, Chinese clans are often referred to as lineages or common-descent groups, consisting of individuals who trace their ancestry to a shared progenitor. For the Ma clan of Tongcheng, this progenitor was born in 1408 and passed away in 1460. Their genealogy, a historical document, records 18 successive generations, with the most recent documented death occurring in 1929. As outlined in Section 2, genealogies provide not only essential demographic data but also valuable socio-economic information, making them a rich source for economic research.

This study utilizes machine-readable text derived from scanned genealogies, processed through optical character recognition (OCR). Each text entry represents the biography of a single individual (n = 1,012). Key variables of interest include birth year, number of sons, birth order among brothers, and socio-economic status. To extract structured data, we fine-tuned OpenAI's ChatGPT-4o, creating a specialized large language model (LLM) tailored to our dataset. Model performance was assessed by comparing its extracted data to manually curated information from trained researchers.

Our findings highlight two key points: First, automated data extraction from Chinese historical documents via LLMs is highly effective, achieving over 90 percent accuracy across various machine learning metrics. Second, fine-tuning ChatGPT-4o significantly enhances extraction accuracy, improving structured information retrieval by approximately 20 percent. The fine-tuned

model is available at http://3.145.178.72:8501/ .

The potential of large language models (LLMs) to transform unstructured text into structured data is well-established, as highlighted in recent studies (e.g., Korinek, 2023, 2024; Dell, 2024). One of the most significant advantages of LLMs is their efficiency. They dramatically accelerate the extraction of data from historical texts, making large-scale analysis feasible where it was previously impractical. For instance, processing a thousand biographies from a typical Chinese family genealogy might take a team of trained researchers several weeks, whereas a well-trained LLM running on a mid-sized GPU can accomplish the same task in minutes. This efficiency shift allows to conduct large-scale quantitative studies instead of much smaller case studies with limited generalizability. Furthermore, some historical documents require specialized knowledge to interpret, and finding researchers with expertise in the script or historical context can be difficult. LLMs help mitigate this challenge by automating classification tasks that might otherwise be infeasible due to a lack of trained human resources.

Another key advantage of LLMs is their adaptability. Off-the-shelf models like ChatGPT can be fine-tuned with specific instructions and trained on particular corpora to classify and analyze texts in ways tailored to the researcher's objectives. This flexibility allows researchers to refine AI outputs to align with their specific analytical frameworks. Additionally, LLMs can be trained to recognize historical language variations, stylistic nuances, and domain-specific terminology, making them highly effective tools for processing diverse textual sources.

A final advantage is replicability. Leading LLMs can generate results with a high degree of consistency, approaching determinism.[1] By contrast, large-scale manual data extraction is often subject to variations over time as research teams change, and individual researchers may introduce their own interpretive biases into the classification process. AI-driven approaches ensure a level of standardization that is difficult to achieve in human-led analysis, reducing inconsistencies and improving the reliability of historical data extraction.

[1] In OpenAI's ChatGPT-4o, for example, this can be achieved by setting the "Temperature" parameter to zero, as noted in Korinek (2023).

This paper details the development of an LLM designed to enable researchers to leverage the potential benefits of AI-driven historical text analysis. We assess the model's performance by comparing its results against manually extracted data from trained researchers, which serves as the empirical baseline. A key contribution of this work is the development of an LLM tailored to a lower-resource language and setting—historical Chinese. AI models often exhibit significant performance declines when applied beyond English and a few other high-resource languages. For instance, even state-of-the-art visual models tasked with OCR on printed Japanese documents from the 1950s misidentify over half of the characters (Dell, 2024). This highlights the critical challenge of adapting AI for historical texts in lower-resource settings.

Related research applying machine learning methods includes Keller, Shiue, and Yan (2024), which classifies large-scale Qing dynasty court records to distinguish entries linked to social unrest from those that are not. While that study relies on the BERT language model developed in 2019 (Devlin et al., 2019), our work builds on the more advanced ChatGPT-4o model from 2024. Given recent breakthroughs in speed, accuracy, and overall language processing capabilities, this shift enables more refined analysis and more reliable text classification (see Korinek (2024)).

Our research also relates to prior studies that employ manually extracted data from Chinese family genealogies, including Shiue (2017), Shiue (2019), and Shiue and Keller (2024). By integrating AI with historical research methodologies, we aim to bridge the gap between traditional manual approaches and scalable computational techniques. The combination of historical expertise with AI-driven text processing presents a promising path for advancing large-scale analysis of historical records to gain new economic insights.

2 Family Genealogies as a Source for Economic Research

Chinese genealogies are documents that contain information about a family's history, providing information in the form of an annotated family tree where male individuals are listed in terms of an intergenerationally-linked structure along with the names and vital statistics of their spouses and children. Genealogies originated from beliefs surrounding ancestral ceremonies and Confucian

teachings dating to at least to the Tang Dynasty (618-907 A.D.). By the Ming and Qing, the practice had spread widely across the empire. As such, the Chinese genealogy is a well-known type of a historical document that falls under the category of household documents

In addition to its ritual significance of ancestor worship, genealogies had a number of economic functions. First of all, the genealogy defined who is a member in the clan and who is not. As a rule, all males descending from the progenitor were members, irrespective of wealth or other characteristics. The declaration of membership was important because it determined rights and responsibilities of each person, which was particularly important in the case of conflict.

Second, the genealogy stipulated key property rights to its clan members, including settlement rights in a certain area as well as rights to use common clan resources (such as land or schools). Third, the genealogy laid out which public goods were to be provided by the clan. That could include local administration, irrigation, or providing a safety net for the poor. Fourth, the genealogy described the dues that each member was to pay to the clan as a form of taxation.

Both because of its ritual significance and its economic functions, there is little reason to believe that the clan would have strong reason for misrepresentation in its genealogy. The genealogy of the Ma clan employed in this paper, in particular, has been found to be highly reliable in earlier research (Shiue, 2017, 2019).

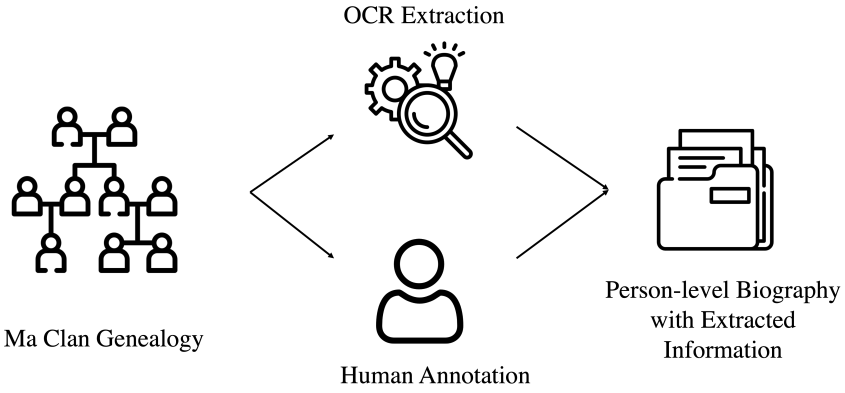
The type of information included in a particular genealogy varies, however, generally it will include vital statistics of each husband, number of generation, socio-economic status (academic degrees, official positions), vital statistics of the wives, birth order of husband and wives, vital statistics of each child, and burial location.

3 Preparation of Dataset for Automated Data Extraction

Figure 1 describes the overall workflow for obtaining data from the Ma genealogy. We start from pdf image files of the Ma clan genealogy. These image files are then employed in two different ways. First, we hire researchers trained in the reading of Chinese family genealogies to perform manual extraction of the data. The results of this manual data extraction are employed as baseline

in the evaluation of our LLMs. In addition, insights from the manual extraction are utilized to train GPT-4o to the specific context of family genealogies ("Human Annotation" in Figure 1). Second, we perform optical character recognition (OCR) on the pdf image files to obtain the biographies of each individual man as the textual content of the image files. Finally, we match the person-level biography records from the OCR of the image files with the person-level human annotations from the manual extraction to obtain the final dataset to which the LLM for the automated data extraction is applied.

Figure 1: Process to Construct the Dataset

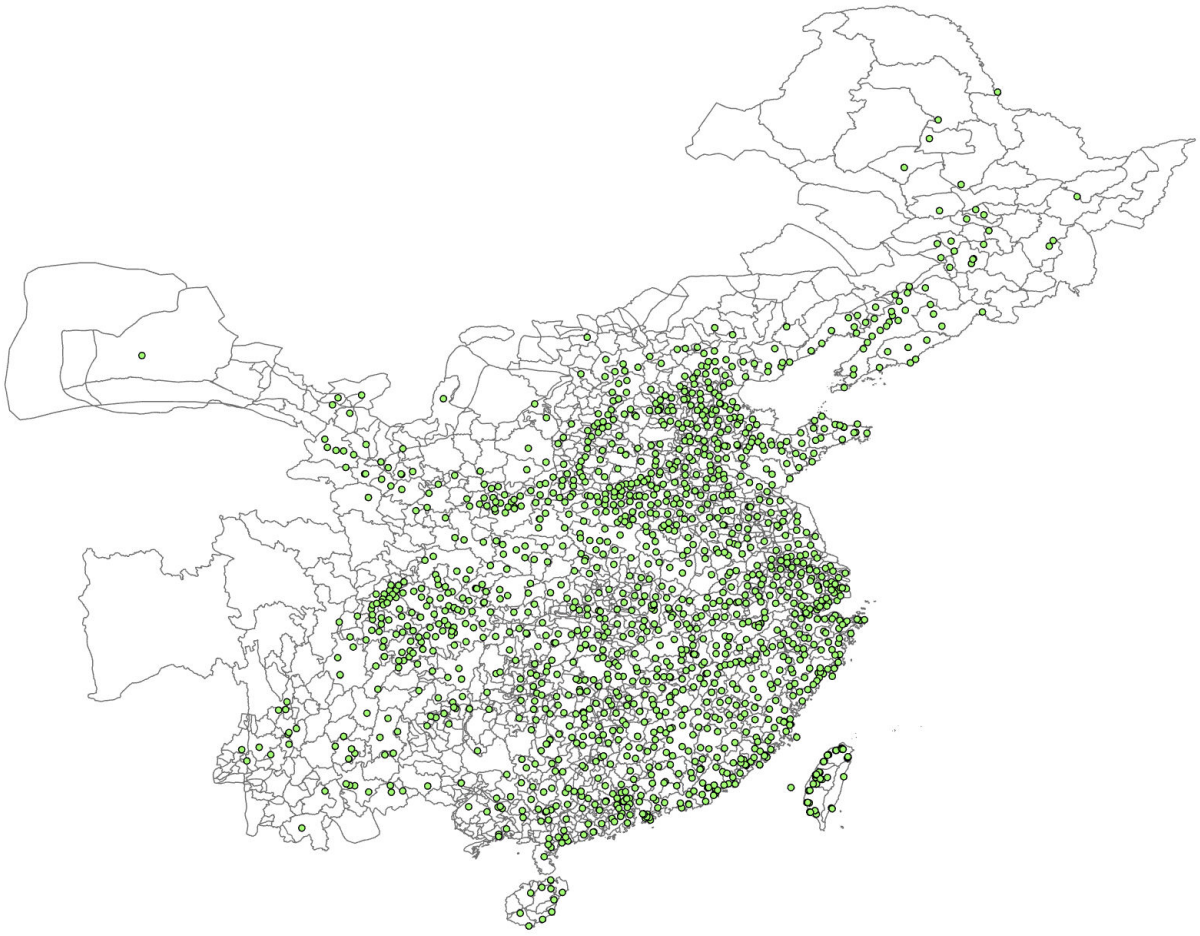


3.1 Availability and Format of Chinese Family Genealogies

Already by the year 2008, more than 50,000 genealogies were catalogued in Wang (2008), covering a period spanning from 1000 AD to 2000 AD. Figure 2 shows the locations of these recorded genealogies, many of which are housed in libraries and public collections. Today, the number of known Chinese family genealogies is larger, however, likely in the hundreds of thousands.

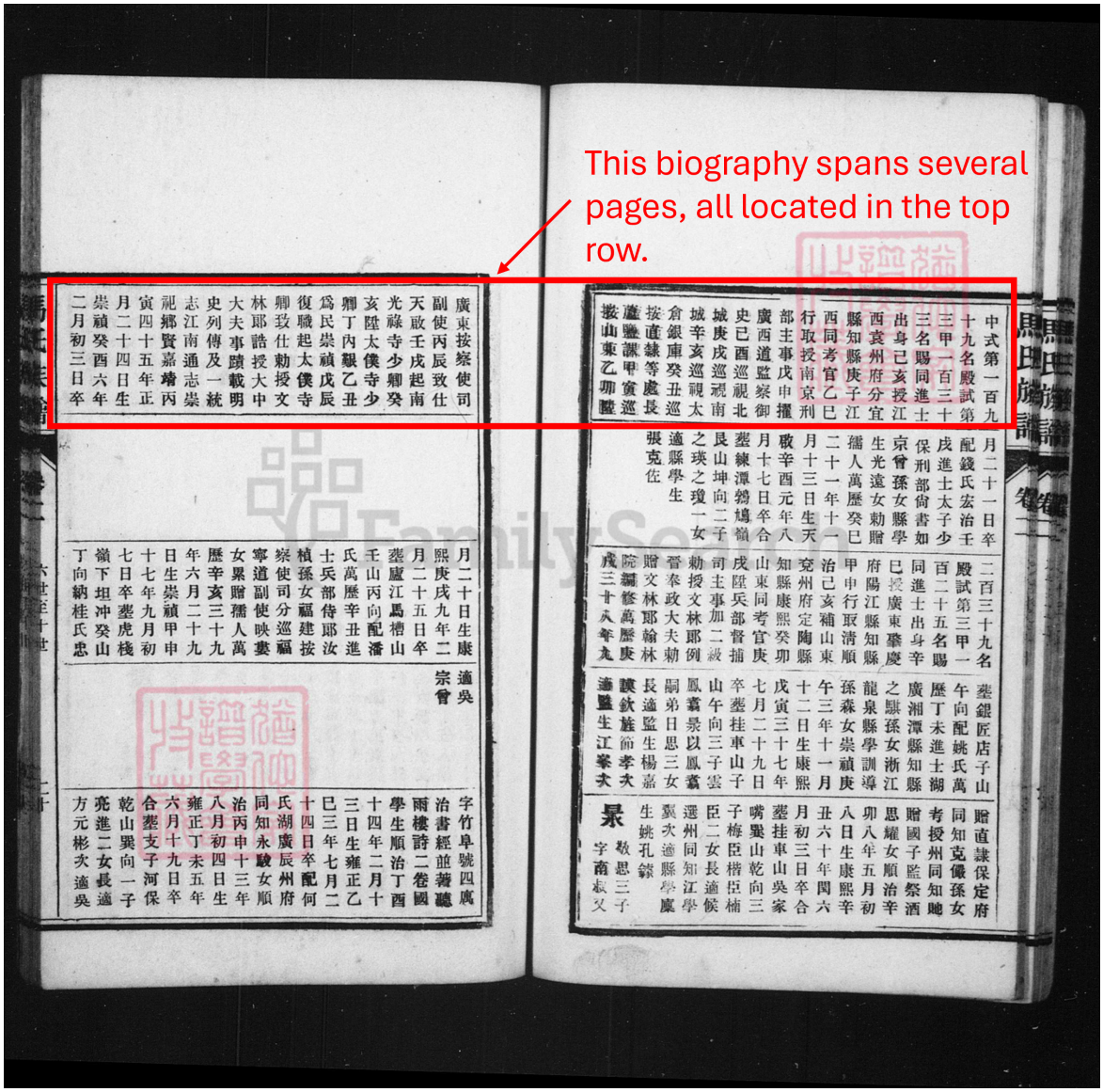
The most common layout of genealogies can be categorized into three groups: vertical, horizontal, and tree-like. Figure 3 exemplifies the horizontal layout, on which we focus in this paper. The vertical and tree-like layouts are shown in the Appendix, Figures A1 and A2 for examples). The horizontal layout example is from the Ma clan of Tongcheng, while the other two examples are from different clans.

Figure 2: Locations of Recorded Genealogies in Wang (2008) within Early-Qing-Era Boarders



Data Source: Wang (2020). Shapefile source: CHGIS (2016).

Figure 3: Example Genealogy Layout in Horizontal Format



3.2 Manual Extraction and Human Annotation

We hired a post-doctoral scholar and ten student research assistants at the Shanghai University of Finance and Economics to manually code the information in the genealogy. They were instructed to read the MA clan genealogy and then extract the predefined variables for each person. The manually extracted data, stored in a spreadsheet file, contains demographic and other genealogical information for 1,518 individuals and up to 228 variables. The information is related to demographic and other genealogical variables, familial relationships, as well as geographic locations. We present key variables in Table 1.

The process of manually extracting information from the image files also generated notes, or annotations, for each of the individuals that we utilize to train the LLMs that are employed in the automated data extraction.

3.3 Using Optical Character Recognition to Extract Text

Optical Character Recognition (OCR) is a well-known technology used to convert different types of documents–such as scanned paper documents, PDFs, or images of text–into editable and searchable data. To keep the focus on using LLMs for automated extraction, however, the OCR step is taken as given for the purposes of this paper.

Nevertheless, the OCR step is an important part of the overall task (see discussion in Dell (2024)). In the present case, the horizontal layout of the image files (see Figure 3) indicates that the biographies are listed in rows and may span across multiple pages, all within the same row. Humans can read this structure relatively easily, but machines cannot. For example, it is difficult for the machine to know how to order the rows, how to know the first row is one person, and the second row is another person; and how to organize the characters of the same person across pages. We work with the Center for Digital Cultures at Academia Sinica (https://ocr.ascdc.tw/) to extract text content from the pdf image files. This OCR process is successful in more than 98 percent of the biographies that were also manually extracted. Table 2 gives an example of a biography after OCR processing.

Table 1: Key Variables in the Manually Extracted Dataset

Variable	Description
家族輩份 (Assigned Generation)	A numerical value indicating the assigned generation.
兄弟排行 (Birth Order)	The ranking of the husband among his brothers.
丈夫名 (Name)	The name of the husband.
丈夫身分 (Social Status)	The highest social status of the husband.
父親名 (Father's Name)	The name of the husband's father.
生年年號 (Birth Reign)	The reign name during which the husband was born.
生年年次 (Birth Year in Reign)	The year within the reign when the husband was born.
生年 (Birth Year)	The actual birth year of the husband.
生月 (Birth Month)	The birth month of the husband.
卒年年號 (Death Reign)	The reign name when the husband passed away.
卒年年次 (Death Year in Reign)	The year within the reign when the husband passed away.
卒年 (Death Year)	The actual year of the husband's death.
卒月 (Death Month)	The month of the husband's death.
葬地 (Burial Place)	The burial place of the husband.
妻子數 (Number of Wives)	The total number of wives the husband had.
兒子數 (Total Sons)	The total number of sons from all wives.
妻族 (Wife's Clan)	The Chinese name of the wife's clan.
妻序 (Wife's Order)	The order in which she became the husband's wife.
妻生年號 (Wife's Birth Reign)	The reign name when the wife was born.
妻生年次 (Wife's Birth Year in Reign)	The year within the reign when the wife was born.
妻生年 (Wife's Birth Year)	The actual birth year of the wife.
妻生月 (Wife's Birth Month)	The birth month of the wife.
妻卒年號 (Wife's Death Reign)	The reign name when the wife passed away.
妻卒年次 (Wife's Death Year in Reign)	The year within the reign when the wife passed away.
妻卒年 (Wife's Death Year)	The actual year of the wife's death.
妻卒月 (Wife's Death Month)	The month of the wife's death.
夫妻子數 (Number of Sons)	The total number of sons from this couple.
夫妻女數 (Number of Daughters)	The total number of daughters from this couple.

Table 2: Example Biography After OCR Processing

Person Name	Biography
慎思(Shensi)	之瓏長子字德公康熙壬寅六十一年正月初六日生雍正戊申六年三月十五日卒莖松茂嶺牛角乾山巽向配詹氏振先女順治庚子十七年十二月二十四日生康熙丙申五十五年三月十四日卒莖黃華黃家凹一子昇一女唐達士(the first son of Zhilong, was born on the sixth day of the first month in the 61st year of the Kangxi era (1662). He passed away on the 15th day of the third month in the 6th year of the Yongzheng era (1728) at Jingsongmao ridge, Niujia Chong, facing Qian mountain. He was married to the daughter of Zhenxian Zhan, who was born on the 24th day of the 12th month in the 17th year of the Shunzhi era (1661) and passed away on the 14th day of the 3rd month in the 55th year of the Kangxi era (1716) at Jinghuanghua, Huangjiaao. They had one son, Sheng, and one daughter, married to Dashi Tang.)

3.4 Matching OCR with Manually Extracted Records and Comparison

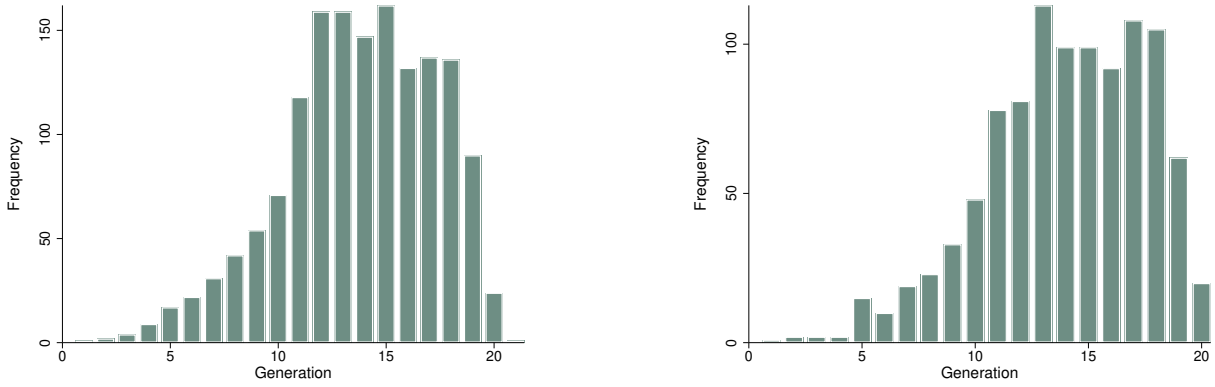
Next, we match the individual biographies of the OCR output to the manually extracted data. Imperfect matching leads to a reduced sample size. We end up with a dataset containing n = 1,012 individual biographies for which we have both OCR-processed text as well as manually created data and annotation information (66.7% of the hand-collected sample). This is mostly due to imprecision in the OCR processing, as noted in section 1. For the unmatched cases, at least 40% can be attributed to slight differences in names between the OCR and hand-collected data; even slight differences in Chinese names can result in unmatched cases.

To better understand the consequences of this reduced sample for characteristics of our sample, the following compares the overall sample with the matched sample in a number of dimensions.

Panels (a) and (b) of Figure 4 present the distributions of the overall sample and the matched subsample, respectively. In general, the distributions are similar. Furthermore, Figure A3 shows the proportion of individuals in each generation with at least some social status, based on their participation in the civil service examination and any official positions they attained. These two distributions are also quite similar. This supports the claim that the matched subsample closely mirrors the distribution of the overall sample. Finally, a comparison of the mean values of key

variables for the two subsamples, presented in Table B1, further reinforces this conclusion.

Figure 4: Sample Distribution in Manually Extracted Sample by Generation.



(a) Overall Sample **(b)** Matched Sample

Note: Horizontal axis denotes generation.

In the following section, we turn to fine-tuning the off-the-shelf GPT-4o model for present purposes.

4 Fine-tuning ChatGPT

This section discusses the process of fine-tuning ChatGPT's 4o model for the purposes of data extraction from Chinese family genealogies. We first introduce the overall process to fine-tune a GPT model via OpenAI's API, and then we conduct inference using the fine-tuned version model (GPT stands for Generative Pre-Trained Transformer, and API stands for application programming interface). This section will also discuss the experiment settings and show and the train/validation loss during fine-tuning process.

4.1 Fine-Tuning and Inference Process Overview

We use large language models (LLMs) to extract the structure data from the text. Typically, the biography of a man in a genealogy documents his name, date of birth, date of death, burial location, wife's information (clan name when born, dates of birth and death, numbers of sons and/or daughters), and total number of sons of this man. For persons with above average social status,

such as those with some education, the biography is usually longer and contains more detailed information. This unstructured nature of genealogies is one of the key reasons why we need a powerful LLM to extract data from these genealogies.

Among large language models, ChatGPT is renowned for its exceptional ability to tackle complex tasks. While it excels at extracting information from text, its performance can be further enhanced for specific tasks by fine-tuning the model on the focal task. As an example, for the application of a powerful LLM on certain medical issues, it can be trained on a relatively small set of data that is specific to that domain, which will increase its performance in the particular task. The following describes this approach of fine-tuning in our context. The manually-created annotations serve as guidance on how to read and interpret historical Chinese genealogies.

Our code calls OpenAI's API to fine-tune the GPT-4o model. The steps to complete the fine-tuning process are as follows.

1) **Construct training and validation data**: We prepare both the training and validation dataset according to the specific format required by OpenAI for fine-tuning their model. The training dataset's purpose is to optimize the parameters of the GPT model, while the validation dataset is used to select the best fine-tuned model from the training process.

2) **Upload train and validation data**: Upload the training and validation datasets to OpenAI via their API.

3) **Fine-tuning**: The OpenAI system optimizes part of the parameters of their GPT-4o model based on the training dataset, and the best fine-tuned model based on the validation set can be selected. Specifically, in the training and validation datasets, we provide example output, a JSON string containing required variables, along with the input (one man's biography as text file), and the GPT model is instructed to reduce the difference between the output it generates and the ground zero truth output we provide.

4) **Inference**: Once the fine-tuning process is complete, OpenAI provides us with a model ID

that uniquely identifies the model we have fine-tuned. We call this model to extract data from text as discussed in the next section through calls to OpenAI's API.

4.2 Fine-Tuning Details

This section provides additional information on the separate fine-tuning steps. First, consider the split of our data set (n = 1,012) into training, validation, and test sub-datasets. We randomly split our dataset into these three sub-datasets using ratios of 60%, 20%, and 20%, respectively, which is a common practice in the deep learning literature (Goodfellow, Bengio, and Courville, 2016). This results in 607 samples for training, 202 for validation, and 203 for testing. Only the training dataset is used to optimize the model's parameters. Model selection is based on the the validation dataset, while the test dataset serves as a final model evaluation on unseen data. Notably, while the validation set is not used to update model parameters, it is still "seen" during training since it guides model selection.

Next, we turn to the nature of prompts that are designed. Generally, a prompt in an LLM provides context, instructions, and constraints which guides the model to generate relevant and coherent responses based on user intent. In this paper, we design the prompt to guide the GPT model to extract the variables (name, birth data, social status, and so on) and instruct the model to generate a JSON string with the required structure. This JSON string is straightforward to load as a dictionary datatype into the programming language python for later analysis. We display the prompt used in this paper in Figure 5.

Figure 5: Prompt for the LLM

```
[
    {"role": "system", "content":
    """
    你是一个从古文家谱中抽取信息的助手，我所需要的信息为：
    ['name', 'order_in_brother', 'social_status', 'father_name', 'birth_reign', 'death_reign',
    ↪  'burial_location', 'total_sons', 'num_wife', 'wife_info'].

    具体的细节如下：

    name 表示在该人物传记中主人公的名字，通常位于开头，并用中文逗号分隔，通常为两个字符，偶尔有单个或多个字符。

    order_in_brother 表示在该人物传记中主人公在兄弟中的排行，通常位于开头几句，紧随其父亲名字之后，常以长子、次子、三
    ↪  子等方式表示。若未明确给出则标记为-1。

    social_status 表示该人物所获得的最高学历或官职，如国子监生、秀才、进士、举人、知县、知府等。如无学历或官职，请抽
    ↪  取'Unknown'。

    father_name 表示该人物传记中主人公父亲的名字，通常位于开头，在中文句号后面，一般为两个字符，偶尔有单个或多个字符，
    ↪  无法找到相关信息请抽取'Unknown'。

    birth_reign 表示该人物出生时所在皇帝的在位年份（年号纪年）。并且删除天干地支信息，如咸丰已未九年为咸丰九年。此信息
    ↪  一般在传记开头、妻子信息之前，若未明确给出则标记为-1。

    death_reign 表示该人物去世时所在皇帝的在位年份（年号纪年）。如乾隆九年，光绪元年等等。此信息一般在传记开头、妻子信
    ↪  息之前，若未明确给出则标记为-1。

    burial_location 表示该人物死亡后埋葬的位置，通常紧随\葬于"字样之后。无法找到相关信息请抽取'Unknown'。

    total_sons 表示该人物一生中所有儿子的数量。若未明确给出则标记为-1。

    num_wife 表示该人物一生中娶妻的总数量。若未明确给出则标记为-1。

    wife_info 是一个列表，每一项表示该人物一位妻子的信息，包含以下内容：
        wife_clan 妻子的姓氏，通常以\X氏"的形式出现，如张氏、李氏等。无法找到相关信息请抽取'Unknown'。
        wife_birth_reign 妻子出生时的皇帝在位年份（年号纪年），如乾隆九年，光绪元年等等。若未明确给出则标记为-1。
        wife_death_reign 妻子去世时的皇帝在位年份（年号纪年），如乾隆九年，光绪元年等等。若未明确给出则标记为-1。
        nsons 该任妻子所生儿子数量。若未明确给出则标记为-1。
        ndaus 该任妻子所生女儿数量。若未明确给出则标记为-1。

    我的输入是多段未分段的古文，我所需要的输出为Python字典，
    字典的key严格按照上述信息格式，value为从文本中抽取的信息。
    """,
    {"role": "user", "content": "<<Insert the biography texts here>>"
    }
]
```

Next, we follow the instructions of OpenAI and convert the training and validation datasets to the required format, which is JSON Lines (JSONL) files.[2]. JSON Lines is a text file format where each line is a separate, self-contained JSON object, which is efficient for streaming and processing large datasets. One example of a JSON object in the training and validation datasets, based on the prompt in Figure 5, is given in Figure 6:

[2]https://platform.openai.com/docs/guides/fine-tuning

Figure 6: Example from Training and Validation Dataset

```
{
    "messages": [
        {
            "role": "system",
            "content": "你是一个从古文家谱中抽取信息的助手，我所需要的信息为：[
                'name', 'order_in_brother', 'social_status', 'father_name',
                'birth_reign', 'death_reign', 'burial_location', 'total_sons',
                'num_wife', 'wife_info'
            ].
            具体的细节如下：

            name: 表示在该人物传记中主人公的名字，通常位于开头，并用中文逗号分隔，通常为两个字符，偶尔有单个或多个字
            ↪  符。

            order_in_brother: 表示在该人物传记中主人公在兄弟中的排行，通常位于开头几句，紧随其父亲名字之后，常以长
            ↪  子、次子、三子等方式表示。若未明确给出则标记为-1。

            social_status: 表示该人物所获得的最高学历或官职，如国子监生、秀才、进士、举人、知县、知府等。如无学历或官
            ↪  职，请抽取 \"Unknown\"。

            father_name: 表示该人物传记中主人公父亲的名字，通常位于开头，在中文句号后面，一般为两个字符，偶尔有单个或
            ↪  多个字符，无法找到相关信息请抽取 \"Unknown\"。

            birth_reign: 表示该人物出生时所在皇帝的在位年份（年号纪年）。并且删除天干地支信息，如咸丰已未九年为咸丰九
            ↪  年。此信息一般在传记开头、妻子信息之前，若未明确给出则标记为-1。

            death_reign: 表示该人物去世时所在皇帝的在位年份（年号纪年）。如乾隆九年，光绪元年等等。此信息一般在传记开
            ↪  头、妻子信息之前，若未明确给出则标记为-1。

            burial_location: 表示该人物死亡后埋葬的位置，通常紧随\葬于"字样之后。无法找到相关信息请抽取
            ↪  \"Unknown\"。

            total_sons: 表示该人物一生中所有儿子的数量。若未明确给出则标记为-1。

            num_wife: 表示该人物一生中娶妻的总数量。若未明确给出则标记为-1。

            wife_info: 是一个列表，每一项表示该人物一位妻子的信息，包含以下内容:
                wife_clan: 妻子的姓氏，通常以\X氏"的形式出现，如张氏、李氏等。
                无法找到相关信息请抽取 \"Unknown\"。
                wife_birth_reign: 妻子出生时的皇帝在位年份（年号纪年），如乾隆九年，
                光绪元年等等。若未明确给出则标记为-1。
                wife_death_reign: 妻子去世时的皇帝在位年份（年号纪年），如乾隆九年，
                光绪元年等等。若未明确给出则标记为-1。
                nsons: 该任妻子所生儿子数量。若未明确给出则标记为-1。
                ndaus: 该任妻子所生女儿数量。若未明确给出则标记为-1。"
        },
        {
            "role": "user",
            "content": "邦基，春生长子字只君号伯莱治书经监生诰赠朝议大夫乾隆戊寅二三年月十三日生道光甲申四年三月二十八
            ↪  日卒配左氏监生松仪孙女县学生祺女诰封太恭人乾隆壬申十七年四月初七日生道光丙午二十三年闰七月三十日卒合茎
            ↪  怀宁云潭保马踏石辛山乙向二子树华树章一女适监生张炜"
        },
        {
            "role": "assistant",
            "content": {
                "name": "邦基",
                "order_in_brother": 1,
                "social_status": "生",
                "father_name": "春生",
                "birth_reign": "乾隆二十三年",
                "death_reign": "道光四年",
                "burial_location": "",
                "total_sons": 2,
                "num_wife": 1,
                "wife_info": [
                    {
                        "wife_clan": "左氏",
                        "wife_birth_reign": "乾隆十七年",
                        "wife_death_reign": "道光二十三年",
                        "nsons": 2,
                        "ndaus": 1
                    }
                ]
            }
        }
    ]
}
```

We also restrict the structure of the output from the GPT to some extent. When designing the prompt shown in Figure 5, we instruct the GPT model to generate a JSON string that can be loaded as the dictionary in python. Even though in most cases the model can generate the desired JSON string that can be loaded successfully, it may make some mistakes. This could be due to missing some characters as simple as a comma, but it may mean that it cannot be read in python correctly as a dictionary. To address this we define a parameter called "response_format" to the structure schema shown in Figure C1. This ensure that the model always generates the correct JSON string with the required structure, as instead of returning the GPT-generated string (i.e., textual datatype in programming language), the OpenAI API will modify string following the schema defined in the "response_format" parameter.

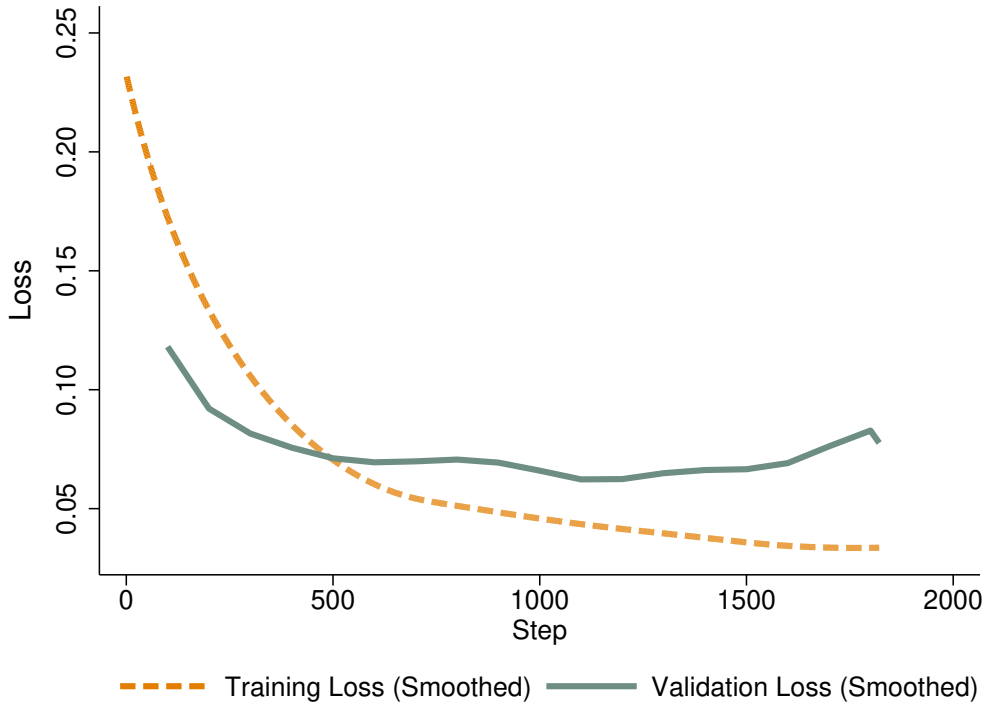
4.3 Fine-Tuning Experiment and Results

In this section, we introduce the experiment settings and provide information on train/validation loss during the fine-tuning process. The experiment involves fine-tuning the model called "gpt-4o-2024-08-06" (GPT-4o for short) in the OpenAI GPT model family for 3 epochs. We choose "gpt-4o-2024-08-06" because it was the latest model at the time of writing, and 3 epochs (as well as batch_size 1) are default settings for fine-tuning GPT-4o models via OpenAI's API. In each epoch, the parameters of GPT-4o get optimized based on the entire training dataset. One can think of one step in the epoch as analogous to updating regression coefficients using only a subset of observations rather than the entire dataset.

Figure 7 shows the evolution of both training loss and validation loss as a function of fine-tuning steps (iterations). Training loss are the residuals of within-sample prediction using the testing dataset. Validation loss refers to the residuals obtained using the model from the validation dataset, and it is used to measure the fine-tuned model's performance on non-training samples. Notice that validation loss does not fall much anymore after about 500 steps, and also the training loss curve becomes much less steep. This indicates that beyond approximately 500 steps, further improvements in the model's performance are difficult to achieve. We will examine in the next

section how large these gains in performance are compared to the benchmark GPT-4o model.

Figure 7: Smoothed Training and Validation Losses in Fine-Tuning



5 Data Extraction: LLM Results

5.1 Extraction Goals and an Example

In principle, the goal is to extract as much information as possible from the biographies, now after OCR processing in one-line text-based format, because each data item might serve as valuable input in research on a number of different topics. Thus, we seek to extract each Ma clan member's demographic and family information, including name, birth and death dates, wife's information, education, academic degrees, official positions, number of sons, and other variables. At the same time, if the error rate in the extraction of a particular data item turns out to be particularly high, it may be preferred to drop this variable from consideration. Table 3 provides a list of variables that we seek to extract. This section will provide information on extraction performance for specific variables, enabling researchers to better understand such trade-offs. Note that if a biography does

not have a certain piece of information, the corresponding field returns "unknown."

Table 3: Extracted Information: Variables

Information	
Name	Wife's family name
Father's name	Wife order[b]
Social status[a]	Wife's birth date
Order in brothers	Wife's death date
Birth date	Number of son births by wife
Death date	Number of daughter births by wife
Burial location	
Number of wives	
Number of sons	

Note: [a]Social status is individual's societal's rank (e.g., academic degree, official in a particular position). [b]The position of a wife in chronological order if multiple wives or female partners exist.

The information shown in Table 3 is gathered via API calls using the fine-tuned model as described in the previous section. We have developed a website that provides access to our model, see http://3.145.178.72:8501/. This website enables the researcher to extract key information from a person's biography when the biography is provided as input. Figure 8 is an example of converting a text biography into structured data using our website. In this example, we successfully extract several details about the individual. This man is named "Guanguang," the eldest son of "Jin" (his father). This confirms that intergenerational links are immediately given in this data source. He was born in June of the 60th year of the Kangxi Emperor (June 1721) and passed away in July of the 50th year of the Qianlong Emperor (August 1790).[3] He was a "county student"; this means he passed a local government exam and was eligible to proceed to higher levels in China's famed civil service examination (keju). All information in Chinese family genealogies are "lifetime" pieces of information, and in this case this means that this man's highest life-time social status was "county student" (which would put his status into the top 20 percent of society). Guanguang's biography also mentions that he had two sons and two daughters. We also extract information on his burial

[3]Dates in Chinese family genealogies are based on the lunar calendar; it is straightforward to convert them to the solar calendar.

location as well as the name, birth, and death dates of his wife.

The structured information just extracted is valuable. For example, birth and death dates allow to calculate lifespan, information on which at the individual level is rare for periods covered by Chinese family genealogies, especially for individuals that are not elites. Combinging information on lifespan and social status allows to examine whether there was a socio-economic status-health gradient in pre-industrial times. Furthermore, information on the educational background of an individual together with information about a person's ancestors and descendants, can help us better understand economic mobility, within-family intergenerational transmission, and other aspects of long-run processes. In the present case, we utilize the fine-tuned model to collect structured information–"data"–on all of the members of the Ma clan.

5.2 Evaluation Using the Test Dataset

Of the n = 1,012 OCR-extracted biographies of Ma clan in our matched dataset, 203 have not been used for either training or validation. These 20% of the original sample are our test dataset, which we now use for out-of-sample validation. The following model evaluation is conducted through both machine learning metrics and statistics.

5.2.1 Machine Learning Metrics Evaluation

The extraction of information from historical text can be seen as a classification task. For example, each unique birth date is treated as a distinct class, making this a multi-class classification problem. To assess the performance of the fine-tuned model in such a setting, we employ four standard machine learning metrics, each of which is based on true versus false positives and negatives. Our four metrics are (i) accuracy, (ii) weighted F1, (iii) weighted recall, and (iv) weighted precision; these metrics are defined in Appendix D. In the context of multi-class classification, the metric accuracy is equal to the metric weighted recall. The proof of this can be found in Appendix D as well.

These machine learning metrics are computed based on results for our test dataset. Specifically, we evaluate these metrics for eight data items, which are the husband's order among his brothers,

Figure 8: An Example Input and Output of Our Model

观光进长子字嘉孚号莳香治易经县学生康熙辛丑六十年六月初一日生乾隆庚戌五十五年七月二十三日卒配左氏鬯广东琼山县知县腮女康熙壬寅六十一年九月初七日生乾隆甲寅五十九年十月二十六日卒合茎挂车山姚太君山右嘴亥已兼壬丙二子惟馨福熙二女长适监生姚培斑次适姚培文

(a) An Example Biography


```
{
"name":"观光"
"father_name":"进"
"social_status":"县学生"
"order_in_brothers":"长子"
"birthday":"康熙六十年七月"
"deathday":"乾隆五十五年七月"
"burial_location":"挂车山"
"num_wife":1
"num_sons":2
"wife_info":[
0:{
"wife_family_name":"左"
"wife_order":"1"
"wife_birthday":"康熙六十年七月"
"wife_deathday":"乾隆五十九年十一月"
"wife_birth_sons":2
"wife_birth_girls":2
}
]
}
```


(b) Example Extracted Information


```
{
 "name": "Guan Guang",
 “father’s name": "Jin",
 "social status": "County student",
 "order in brothers": "Eldest son",
 "birthday": "60th year of Kangxi, 7th month",
 "death day": "55th year of Qianlong, 7th month",
 "burial location": "Gua Che Mountain",
 “number of wives": 1,
 "number of sons": 2,
 "wife’s info": [
  {
   "wife’s family name": "Zuo",
   "wife order": "1",
   "wife’s birthday": "60th year of Kangxi, 7th month",
   "wife’s death day": "59th year of Qianlong, 11th
month",
   "wife born sons": 2,
   "wife born girls": 2
  }
 ]
}
```


(c) Example Extracted Information (Translated)

his birth year, his death year, the number of his wives, the total number of his sons, the name of his wife, as well as the birth and death years of his wife. At this point we have not evaluate other data items such as social status because these variables are mostly text rather than a number, and that makes it challenging to evaluate whether the extracted information matches the manual extraction results when computing the metrics.

As can be seen in Table 4, our model outperforms that of non-fine-tuned GPT-4o according to most metrics. Across all measures and all variables, the average improvement in performance through fine-tuning is 22 percent. The comparison shown in Table 4 demonstrates that fine-tuning significantly enhances GPT-4o's ability to extract structured information from the biography. The most notable improvements are in accuracy and in weighted recall, with 31 percent, but also the weighted F1 score increases through fine-tuning on average across all variables by 25 percent. This indicates that the fine-tuned model is a more consistent and reliable choice for extracting complex variables. While precision drops slightly for a few variables, this captures a tradeoff between higher weighted recall and overall accuracy. This makes the fine-tuned model more reliable in real-world applications, where missing relevant information is often more problematic than a small increase in false positives.

Across variables, the largest improvements through fine-tuning GPT-4o are realized for extracting information on the birth order of brothers, with 52 percent, but also for wife's name and death year the improvement exceeds 30 percent. The only variable for which fine-tuning does not improve the performance of the LLM beyond the level of GPT-4o is the death year of the husband. Overall, this analysis shows that fine-tuning can lead to substantial performance improvements even if the baseline is one of the industry-leading large language models.

5.3 Evaluation Based on Statistics

Analysis of the metrics in the previous section demonstrates that our fine-tuned model performs very well. In the next step, we compare the values of the machine-extracted variables with the corresponding values in the manually-processed dataset. The following shows this comparison for

Table 4: Machine Learning Metrics Comparing ChatGTP 4o and our Fine-tuned LLM

	GPT-4o				Fine-tuned GPT-4o			
	Accuracy	F1	Precision	Recall	Accuracy	F1	Precision	Recall
Order in brothers	0.4729	0.623	**0.9673**	0.4729	**0.8818**	**0.883**	0.8945	**0.8818**
Birth year	0.8719	0.8671	0.9433	0.8719	**0.8768**	**0.8583**	**0.9385**	**0.8768**
Death year	**0.9064**	**0.9143**	**0.9674**	**0.9064**	0.9015	0.9027	0.9643	0.9015
Number of wife	0.7586	0.8383	0.938	0.7586	**0.9507**	**0.9482**	**0.9511**	**0.9507**
Number of sons	0.734	0.6967	0.7021	0.734	**0.8227**	**0.8262**	**0.8359**	**0.8227**
Wife name	0.625	0.6271	**0.987**	0.625	**0.9095**	**0.913**	0.9533	**0.9095**
Wife birth year	0.6121	0.613	**0.9773**	0.6121	**0.8319**	**0.8137**	0.9084	**0.8319**
Wife death year	0.6034	0.6014	**0.9787**	0.6034	**0.8836**	**0.8819**	0.9499	**0.8836**

Note: Numbers in bold indicate better performance in the comparisons. The F1, Precision and Recall scores are weighted average in the context of multi-class classification; see Appendix D.

the 203 randomly selected biographies in our test dataset. Table 5 presents summary statistics for three key variables, with figures for the manually-extracted data on the left and the LLM-extracted data further on the right of the table.

Table 5: Comparison of Manual Extraction and Machine Extraction: Key Variables

Variable	Manual Extraction			Machine Extraction				
	Mean (1)	Std (2)	Obs (3)	Mean (4)	Std (5)	Obs (6)	Mean diff (7)	P-value (8)
Social status	0.207	0.406	203	0.374	0.485	203	-0.167	0.000
Number of sons	1.459	1.475	135	1.366	1.480	142	0.093	0.601
Number of wives	1.008	0.087	131	1.000	0.000	134	0.008	0.313

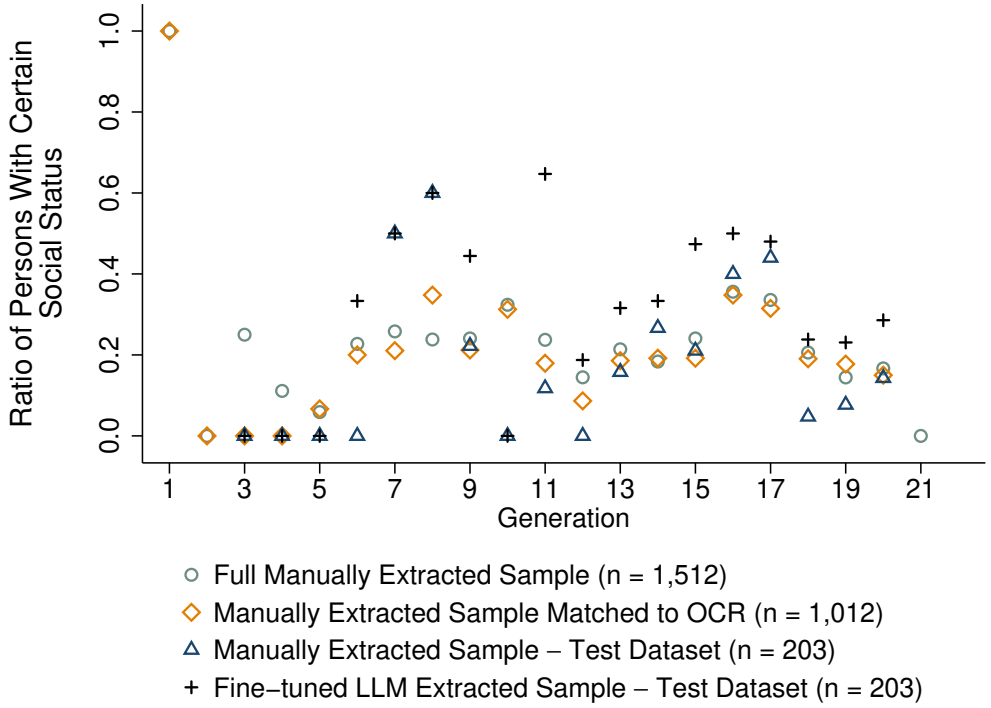
Note: Table compares three variables in the manually-extracted data and the fine-tuned LLM-extracted data. Social status is an indicator variable that the individual has at least some status. Number of wives includes unmarried female partners (concubines). Sample is the n = 203 randomly selected biographies of our test dataset.

Table 5 indicates that the distribution for two variables in the manual- and in the machine-extracted data is similar, namely for the Number of sons and the Number of wives. For the number of sons, both methods indicate that there were on average about 1.4 sons in the typical marriage of a Ma husband, and also the standard deviation is similar for manually extracted and machine-extracted data (about 1.48). In the case of the number of wives, the mean difference between

manual and machine extraction is 0.8%, which is not significant at standard levels.[4]

However, the difference between manually- and machine-extracted data for the social status variable is bigger. As Table 5 indicates, manual extraction yields to about one in five individuals having at least some status (20.7%), whereas our fine-tuned LLM classifies more than one in three individuals as having at least some status (37.4%). To probe deeper into this difference, we examine this difference generation by generation. Results are shown in Figure 9.

Figure 9: Proportion of Persons with At Least Some Status by Generation



Note: Manually and machine-extracted data sets comprise of the same individuals.

Figure 9 indicates that the fraction of individuals holding status follows a similar pattern across generations for the manually- and the machine-extracted test datasets. This pattern is a M-shaped trend, with relatively high values around generations 8 and 16, and this pattern is generally found also in the two larger samples shown in Figure 9. At the same time, we see that machine-extraction tends to yield higher levels of social status than manual extraction, especially starting from the

[4]Results indicate that manual extraction yields one individual who had two wives whereas all machine-extracted data are single-wife marriages.

11th generation. This is what is behind the overall mean difference shown in Table 5.

Upon closer inspection of the outputs of manual extraction versus machine extraction, we find that manual extraction of the status variable focuses on explicit stages of the civil service examination, such as being a student at various levels, as well as being a local, provincial, or national graduate of the civil service examination (*shengyuan*, *juren*, and *jinshi*, respectively). In contrast, our fine-tuned LLM also infers social status that is not thus-recorded in the biographies. For example, the relatively high share of individuals with status in the 11th generation using machine-extraction results because nearly half of the individuals in this generation wrote books and had a certain degree of fame, even though they lacked any of the explicit signs of status that the manual data extraction focused on. [5]

The question is which approach is better–a focus on explicit signs of social status, as implied by the manual approach, or a broader approach that also utilizes implicit signs of status? Research by scholars specializing on these questions indicates a clear preference for the latter. For example, a study of social status using family genealogies from Anhui province during Ming-Qing China classifies educated individuals as having elevated social status even if they have neither academic degree nor office (Telford, 1990). Also studies on human capital returns during this period recognize that an individual may have elevated social status even though he has not obtained a certain degree (Shiue, 2017), while other work indicates that especially such men that do not have the highest degrees or official positions were crucial to organizing and running the clan (Beattie, 1979). In sum, the fact that our fine-tuned LLM can reason and infer such higher status provides a more accurate reflection of the social status distribution, and it underscores the superior performance of our model compared to a manual extraction approach that focuses on simple explicit signs of status.

[5] Such educated individuals who were knowledgeable but did not pass the civil service exam are also referred to as *tongsheng*.

6 Conclusions

China's historical records are exceptionally rich and vast, providing abundant materials for economic history research. However, the need for significant human and material resources to process these records severely hinders the progress of research. With the development of LLMs, automated data extraction from historical Chinese text is starting to be feasible.

We fine-tune a version of ChatGPT to extract information from genealogies from Ma clan of Tongcheng county, in Anhui province. We first OCR the text from these genealogies, and then use 80% of text biographies to fine-tune ChatGPT, a groundbreaking large language model. We show that the fine-tuned version of GPT-4o model outperforms the non-fine-tuned version according to most metrics, and on average there is an improvement in performance through fine-tuning of more than 20%.

Comparisons to the output of manual data extraction generated by humans confirm that automated extraction using our fine-tuned LLM is generally successful. For most variables we find that machine extraction leads to similar distributions as the manual extraction approach. In the case of social status, a variable crucial for studying many economic questions, our fine-tuned model has clear advantages in terms of reasoning and inference over a manual approach that focuses on explicit status indicators. This indicates that machine learning approaches may not just be more efficient, consistent, and replicable but also lead to "better" data extraction than what is possible with the available human classifiers.

The frontier in using LLMs to automate data extraction from historical text is quickly advancing. Recently, DeepSeek has attracted widespread attention for offering performance on par with the latest ChatGPT while having lower usage costs. Future research may consider employing DeepSeek to further improve automated data extraction. Another direction for future research is to perform the tasks of optical character recognition and data extraction in one instead of multiple steps.

References

Beattie, Hilary J. 1979. *Land and lineage in China*. Cambridge University Press.

CHGIS. 2016. "1820 Layers GBK Encoding." URL https://doi.org/10.7910/DVN/2K4FHX.

Dell, Melissa. 2024. "Deep Learning for Economists." *NBER Working Paper* 32768.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. "Bert: Pre-training of deep bidirectional transformers for language understanding." In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 4171–4186.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Keller, Wolfgang, Carol H Shiue, and Sen Yan. 2024. "Mining Chinese Historical Sources At Scale: A Machine Learning-Approach to Qing State Capacity." *NBER Working Paper* 32982.

Korinek, Anton. 2023. "Generative AI for Economic Research: Use Cases and Implications for Economists." *Journal of Economic Literature* 61 (4):1281–1317.

———. 2024. "LLMs Learn to Collaborate and Reason: Update to Korinek 2023." *Journal of Economic Literature* .

Shiue, Carol H. 2017. "Human capital and fertility in Chinese clans before modern growth." *Journal of Economic Growth* 22:351–396.

Shiue, Carol H and Wolfgang Keller. 2024. "Elite Strategies for Big Shocks: The Case of the Fall of the Ming." *NBER Working Paper* 33121.

Shiue, Carol Hua. 2019. "Social mobility in the long run: a temporal analysis of China from 1300 to 1900." *Centre for Economic Policy Research* .

Telford, Ted A. 1990. "Patching the holes in Chinese genealogies: Mortality in the lineage populations of Tongcheng County, 1300-1880." *Late Imperial China* 11 (2):116–136.

Wang, Heming. 2008. *Comprehensive Catalogue of Chinese Genealogies (Zhongguo jiapu zongmu)*. Shanghai Ancient Works Publishing House.

Wang, Yuhua. 2020. "Comprehensive Catalogue of Chinese Genealogies." Harvard Dataverse, V3. URL https://doi.org/10.7910/DVN/PO0VF6.

Appendices

Appendix A Figures

Figure A1: Example of Vertical Layout Genealogy

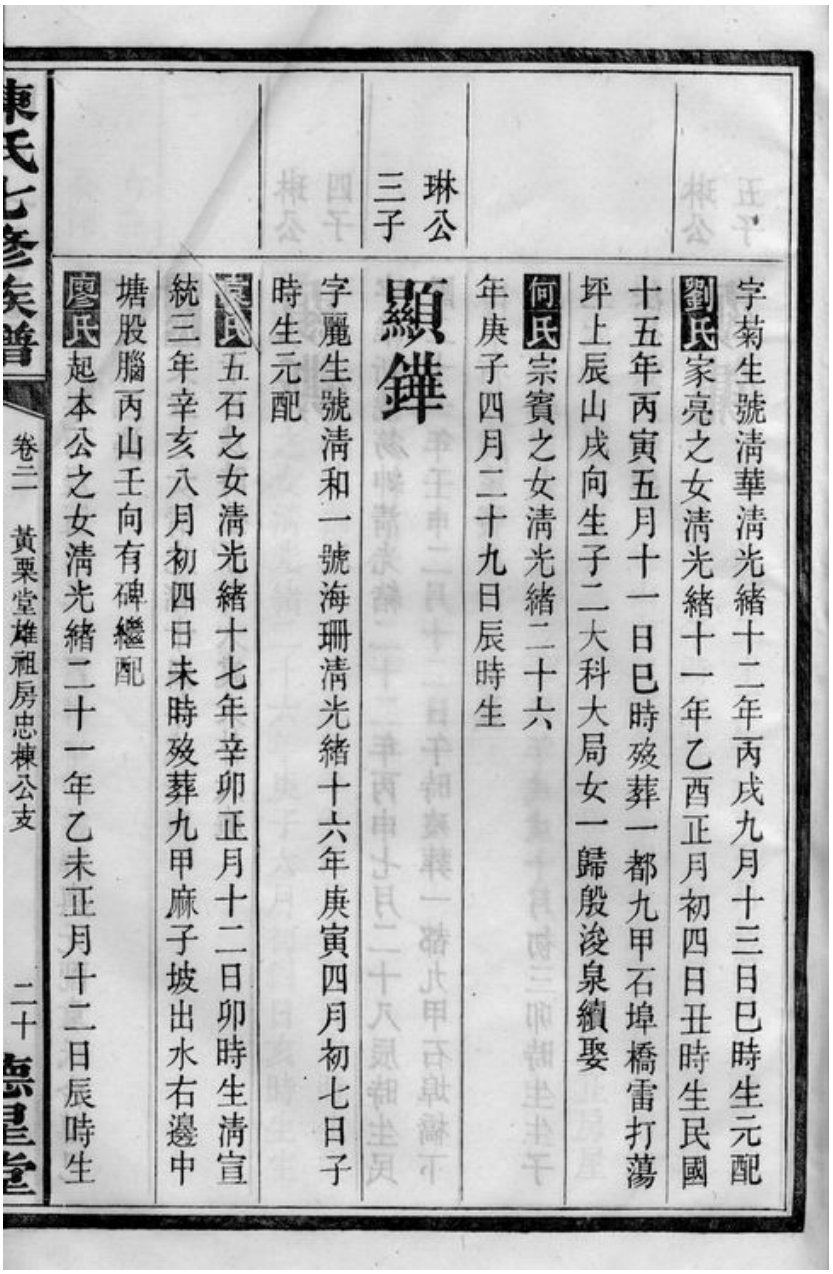


Figure A2: Example of Tree-Like Layout Genealogy

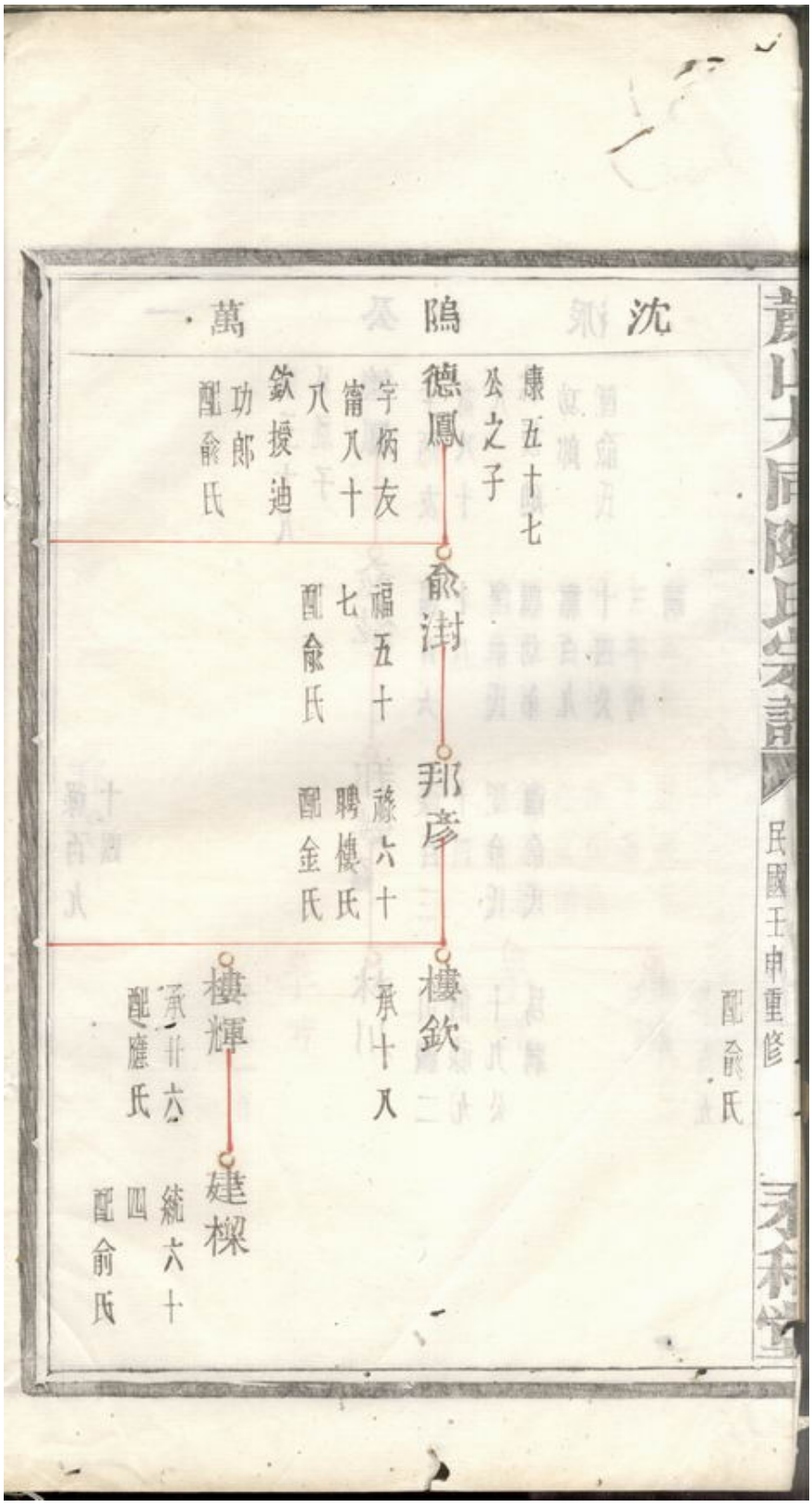
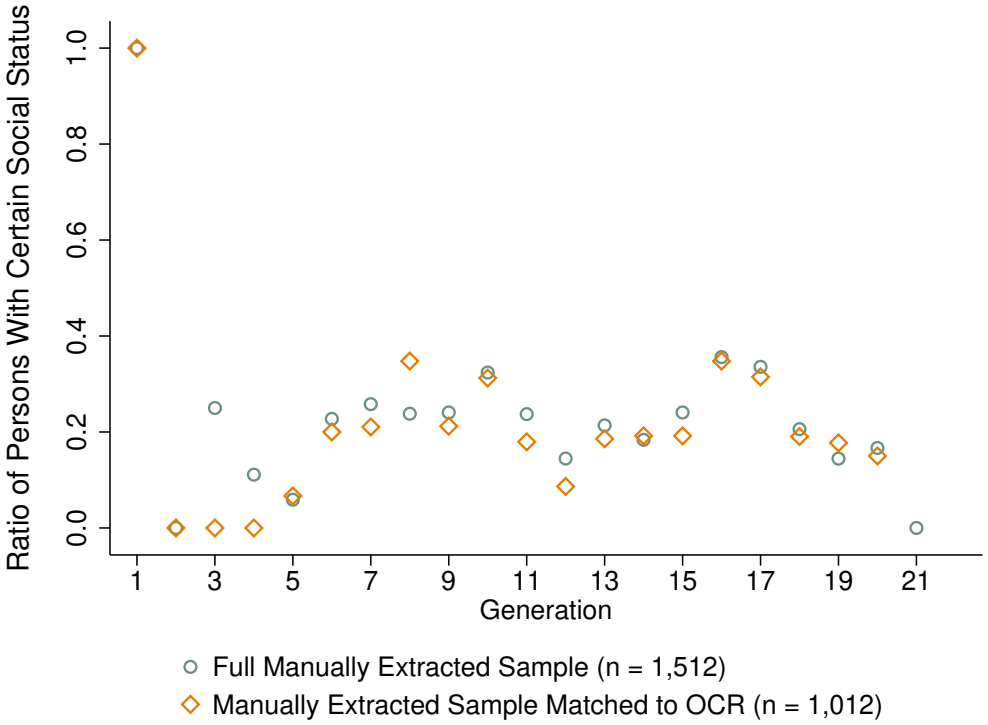


Figure A3: Proportion of Persons with At Least Some Social Status by Generation: Overall vs. Matched Sample



Note: Figure illustrates the proportion of individuals in each generation who hold at least some social status, as determined by their educational background.

Appendix B Tables

Table B1: Summary Statistics of Overall and Matched Samples

Variable	Overall Sample			Matched Sample				
	Mean (1)	Std (2)	Obs (3)	Mean (4)	Std (5)	Obs (6)	Mean diff (7)	P-value (8)
Social status (has certain social status=1)	0.232	0.422	1,518	0.215	0.411	1012	0.016	0.332
Number of sons	1.645	1.439	995	1.565	1.456	667	0.080	0.269
Number of wives	1.007	0.083	1013	1.005	0.068	655	0.002	0.548

Note: "Social status" variable is 1 if the individual holds at least some status, 0 otherwise. For example, if a person is a county student (indicating eligibility to take the county exam), his "social status" is 1. "Overall Sample" refers to the entire manually processed sample.

Appendix C Response Format

Figure C1: Restriction on GPT Output

```
{
    "type": "json_schema",
    "json_schema": {
        "name": "person_info",
        "schema": {
            "type": "object",
            "properties": {
                "name": {"type": "string"},
                "order_in_brother": {"type": "integer"},
                "social_status": {"type": "string"},
                "father_name": {"type": "string"},
                "birth_reign": {"type": "string"},
                "death_reign": {"type": "string"},
                "burial_location": {"type": "string"},
                "total_sons": {"type": "integer"},
                "num_wife": {"type": "integer"},
                "wife_info": {
                    "type": "array",
                    "items": {
                        "type": "object",
                        "properties": {
                            "wife_clan": {"type": "string"},
                            "wife_birth_reign": {"type":
                            ↪  "string"},
                            "wife_death_reign": {"type":
                            ↪  "string"},
                            "nsons": {"type": "integer"},
                            "ndaus": {"type": "integer"}
                        },
                        "required": [
                            "wife_clan",
                            "wife_birth_reign",
                            "wife_death_reign",
                            "nsons",
                            "ndaus"
                        ],
                        "additionalProperties": false
                    }
                }
            },
            "required": [
                "name",
                "order_in_brother",
                "social_status",
                "father_name",
                "birth_reign",
                "death_reign",
                "burial_location",
                "total_sons",
                "num_wife",
                "wife_info"
            ],
            "additionalProperties": false
        }
    }
}
```

Appendix D Evaluation Metrics

In the context of binary classification, the metrics precision, recall and F1 score are defined as follows:

- **Accuracy** is the ratio of correctly predicted outcomes to all outcomes.
- **Precision** is the ratio of correctly predicted positive outcomes to all outcomes predicted positive.
- **Recall** measures the proportion of actual positive outcomes correctly identified by the model.
- **F1 score** is the harmonic mean of precision and recall, offering a balanced measure of both.

However, in a multi-class scenario such as ours (e.g., n different birth years), defining what is meant by a positive outcome (correct prediction) is not as straightforward. In the following, we employ the metrics weighted average precision, weighted recall, and weighted F1. These metrics are computed as follows:

1. Separately for each of the n classes, we treat the current class as the *positive* label, and all other classes are *negative*.
2. Precision, Recall, and F1 score are calculated individually for each class using this binary lens.
3. Computation of the *weighted average* of each metric across all classes, with the weights being each class's frequency in the dataset.

Specifically, the metric weighted recall for a multi-class scenario can be written as:

$$\textit{Weighted Recall} = \sum_{i=1}^{n} \frac{n_i}{N} \frac{TP_i}{n_i} \tag{1}$$

$$= \frac{\sum_{i=1}^{n} TP_i}{N} \tag{2}$$

where:

- TP_i is the number of correctly predicted outcomes for class i.
- n_i is the total number of outcomes belonging to class i.
- N is the total number of outcomes in the dataset.
- $\frac{n_i}{N}$ is the weight of class i.
- $\frac{TP_i}{n_i}$ is recall considering class i as the positive label and others as negative labels.

Note that this expression is identical to overall accuracy, defined as the ratio of total correctly predicted outcomes (across all classes) to the total number of outcomes. Thus, in a multi-class scenario, the weighted recall metric simplifies to the overall accuracy metric.