FEDERAL RESERVE BANK
PHILADELPHIA

Working Papers
RESEARCH DEPARTMENT

Mixed Methods
Performance of OCR, Fuzzy Search, and LLMs with Image-Based Text

Daniel Moulton
Federal Reserve Bank of Philadelphia
Consumer Finance Institute

Robyn Smith
Federal Reserve Bank of Philadelphia
Consumer Finance Institute

Larry Santucci
Federal Reserve Bank of Philadelphia
Consumer Finance Institute

WP 26-54

PUBLISHED
November 2026

FRB CFI
PHILADELPHIA

ISSN: 1962-5361
Disclaimer: This Philadelphia Fed working paper represents preliminary research that is being circulated for discussion purposes. The views expressed in these papers are solely those of the authors and do not necessarily reflect the views of the Federal Reserve Bank of Philadelphia or the Federal Reserve System. Any errors or omissions are the responsibility of the authors. Philadelphia Fed working papers are free to download at: https://www.philadelphiafed.org/search-results/all-work?searchtype=working-papers.

DOI: https://doi.org/10.21799/frbp.wp.2026.54

Mixed Methods: Performance of OCR, Fuzzy Search, and LLMs with Image-Based Text

Daniel Moulton[†] Robyn N. Smith[†] Larry Santucci[†]

October 23, 2026

Abstract

Over time, humans have created vast archives of unstructured textual data that remain largely inaccessible to contemporary researchers. Advances in optical character recognition (OCR), full-text search, and large language models (LLMs) offer unprecedented opportunities to unlock insights from historical source material by making it technically and economically feasible to analyze these data. We present a framework to consider the various costs and trade-offs of different methods for working with a large corpus of images containing unstructured text. We illustrate this framework and present performance statistics via a case study identifying racial restrictions in 5.1 million images of Philadelphia property deeds from 1910–1965. We find 8,395 deed images containing a racial restriction, which we estimate to cover more than 6% of all parcels in Philadelphia.

Keywords: optical character recognition (OCR), large language models (LLMs), artificial intelligence (AI), machine learning (ML), racially restrictive covenants, deeds, property, real estate, housing, digitization

JEL Classification: C81, G5, N32, R31, R38

Approximate word count: 10,990

[†]Federal Reserve Bank of Philadelphia, Ten Independence Mall, Philadelphia, PA 19106-1574.
Disclaimer: The views expressed here are solely those of the authors and do not necessarily reflect the views of the Federal Reserve Bank of Philadelphia, the Federal Reserve Board of Governors, or the Federal Reserve System. Nothing in the text should be construed as an endorsement of any organization or its products or services. Any errors or omissions are the responsibility of the authors. No statements here should be treated as legal advice.

1 Introduction

For thousands of years, humans have created vast archives of unstructured textual data. This unstructured textual data include materials like diary and journal entries; meeting minutes; organization bylaws; individual and business ledger books; property transfers; birth, death, and marriage certificates; records of immigration and naturalization; and shipping and business inventories. Unfortunately, much of this information remains largely inaccessible to contemporary researchers.

Many administrative documents—such as property deeds or mortgage documentation—have been digitized in the sense of digital imaging. However, this format is unsuitable for large-scale analysis because the actual content of the image is not machine-readable. Gathering, much less analyzing, data from millions of images of text is not feasible as a manual task and must be automated as far as possible.

Advances in optical character recognition (OCR), full-text search, and large language models (LLMs) offer unprecedented opportunities to unlock insights from historical source material by making it technically and economically feasible to extract, process, and analyze these data.

We propose a framework that considers when to use tools in ensemble, how to quantify trade-offs, and how to enable flexibility in the research process. While the quantitative results of this paper will eventually be out of date, balancing these considerations will remain essential regardless of the technological frontier.

We explore practical considerations of incorporating LLMs in research involving a large corpus of unstructured text embedded in images. We argue that LLMs, in their present iteration, are not a silver bullet but work best in ensemble with other established methods. We evaluate the performance and cost of various combinations of both open- and closed-source OCR models and LLMs on a subset of 279,000 images of Philadelphia property deeds from 1910–1965.

Independent of our interest in housing economics, the deeds in our data provide a wide range of image variability, which accurately reflects the reality of working with historical sources. In general, input images and archival records tend to be messy and highly heterogeneous. Variable document format and poor-quality images are typical across many different archival research contexts, and these issues become more pronounced as a function of document age. Increasing age means more handwritten content, deterioration of original sources or microfilm, and coarser reproduction methods.

This paper is structured as follows: §2 lays out a framework for selecting a research design when working with a large corpus of unstructured text in images. §3 presents our case study of Philadelphia property deeds from 1910–1965. §4 details how we tested different combinations of OCR and LLMs. §5 contains results and discusses the implications for the application of our framework to other projects. Finally, §6 concludes. An online appendix provides a glossary, more technical detail, and complete results from all tests.

2 Framework

We consider the following scenario: millions of images of textual data are available where the research question hinges on locating documents within the corpus that contain analytically relevant content. Once pertinent documents are located, additional information is to be extracted from the text in a pre-defined, structured format. A wide range of tools and models can be used to approach this problem. We organize possible methods along a spectrum ranging from a fully custom, purpose-built solution leveraging ML and/or AI models to a one-step off-the-shelf tool with minimal modification. We find something between these extremes—an ensemble method combining flexible, off-the-shelf tools—to offer the best trade-off in performance, cost, and effort. That said, each project is different, and by understanding the cost, strengths, and trade-offs of different methodologies along this spectrum, one can identify methods best suited to a specific corpus and research question. We begin by categorizing and discussing qualitative factors of the image corpus and external constraints of the research project.

1. Research Scope. Evaluating methods starts with establishing the scope and definition of the research question. Incorporating customized elements is better suited for narrow and precise research questions. Conversely, the ability to iterate on a research question as new information is learned can make customization less attractive—rebuilding or retraining a model to ask a new question can be burdensome and imprecise.

2. Task Complexity. The overall complexity of the task is a factor in choosing a methodology. A singular classification task is less complex than classification combined with extracting structured relevant text. Complexity also increases with the amount of text that needs to be analyzed to complete the task (OpenAI 2025); classifying a sentence is less complex than classifying a page or book. Simple tasks or complex tasks broken into simple tasks are best suited for off-the-shelf models or LLMs.

3. Image Variability. Considering the visual variability in the corpus of images follows similar logic. Visual variability includes the formatting within each image; whether there is structured text, like a table, mixed in with unstructured prose; whether the text is handwritten or typed; whether the handwriting is all from the same person or at least similar in style and convention; and whether typed text is all the same font and font size. Visual variability can also arise from the quality of the digital images. Modern scans from original source material in good condition are less variable than images produced from degrading and/or improperly exposed microfiche where part of the text may be corrupt and illegible. High quality images with consistent formatting and text are better suited for custom methods. A highly variable corpus will benefit from more flexible approaches.

4. Frequency of Event. Another methodological consideration is the expected rarity of the relevant content within your corpus. A brute-force approach fully processing every image in the corpus may not be computationally or fiscally responsible if the vast majority of the corpus produces a null result. Conversely, locating something that exists in most documents makes the cost per true positive of a brute-force approach more reasonable, although in this case, a sampling strategy may be appropriate. Searching for something occurring frequently in the corpus also simplifies the process of training or tuning a customized model. The balance here is the additional cost/work of introducing a step to filter the corpus before processing versus the cost of a brute-force approach.

5. Cost. Fiscal, computing, and technical expertise constraints are closely entwined. The per token cost of a frontier LLM may be unattractive, but such models are typically easier to set up and run compared to an open-weight ("free") LLM. Depending on the size of a model, running an open-weight LLM on a typical laptop may not be feasible and may require purchase or rental of substantial GPU resources. The size of the corpus combined with the computing resources dictates the time to complete a task—a mid-sized model on a corpus of ≥1 million documents running on a laptop could easily take several weeks. Faster compute times are possible, but this typically requires access to GPU resources beyond even a sophisticated gaming computer. Technical expertise to utilize the API of an LLM allows for more flexibility and customization of model parameters and control of how the model runs on computing architecture, both of which can be adjusted to decrease fiscal, time, and computing costs. Such trade-offs are similar for closed versus open source OCR models and other non-LLM machine learning (ML) models.

2.1 Fully Customized Solution

A custom, finely-tuned ML model built to classify and extract specific information from a specific corpus of documents will likely yield the best results—results that can meet or even exceed those produced by human transcription and data entry. Such a strategy will incur the greatest costs in time and effort. While all approaches require human-verified labels for validation of performance, custom models can require a larger set of labeled data to perform well.

Projects can evolve quickly as more is learned about the data, particularly with large new datasets where the contents have never been studied or accessed before. Refining subsequent iterations in the research process can lead to the discovery of something that wasn't known to exist or—worse—that invalidates the initial research design. Moreover, even when projects evolve according to plan, researchers often identify additional research questions that may inform future research. If the new research question is sufficiently different from the initial question that the model was trained on, the highly tuned machine must be rebuilt from scratch—at additional cost. Regardless, such a hyperspecialized approach can make sense for a highly visually uniform corpus and a research goal with a well-defined, narrow scope.

2.2 LLMs Alone

With the right LLM, document classification and data extraction can be done in a single step for each image without the need for intermediate optical character recognition (OCR) output.[1] This method—for performance reasons—works best for tasks of low-to-moderate complexity and when using an API rather than a web-based interface. As with any LLM, it is necessary to experiment with different prompts and closely validate results.

This method can be *very* cost inefficient. Without OCR output, there is no way to remove obvious nulls from a corpus of images. Instead, every single image must be sent through an expensive multimodal LLM. This inflexibility compounds if additional

1. Although the OCR is not output as its own data product in this method, how well the model "reads" or "comprehends" the image underpins its performance on the task.

research questions arise in the course of a project—the whole corpus must be analysed again to address new questions. Estimating the expected rate of true positives is therefore crucial before choosing this method.

Complications also arise with research questions involving multiple classification and structuring tasks. To minimize this problem, it is tempting to combine many tasks into one LLM prompt. That said, this is not a panacea; as task complexity increases, LLM model performance degrades while output and thought token costs increase.

Finally, validating results with this approach can be challenging, as the reviewer must refer to the original image rather than machine-readable OCR output. LLMs can "explain their reasoning," which can be used to assist with validating results, but we find that incorrect classifications are often accompanied by an "explanation" that appears sound but is pure hallucination.

2.3 The Value of OCR as a Separate Step

Beginning with an image of text, converting the text to a machine-readable format requires either manual human transcription, automated OCR, or both. The large monetary and time cost of manual transcription is not usually feasible for a large corpus of documents. However, improvements in OCR model performance over the last decade have unlocked the possibility of converting entire collections of books and records from static images into true machine-readable, searchable, analysis-ready text at scale.

LLMs can be used as an OCR engine with the output maintained as a separate data product. As of submission of this paper, LLMs as OCR do not substantially outperform traditional ML models and are significantly more expensive than existing paid ML-based OCR, much less open-source models. Full discussion is found in §5.3.

Converting images to text via OCR——even when the output is imprecise—enables a flexible workflow. A wide range of established methods can search and filter this text to facilitate data exploration and discard obvious nulls. This narrows the corpus to a smaller, relevant subset, at which point more precise and computationally expensive LLM and ML methods can be applied—either to the filtered text itself or to the corresponding subset of original images. Further, separate OCR can aid in verifying LLM output, a crucial step.

2.4 An Ensemble of Tools

Finally, we consider a modular approach which employs flexible, adaptive tools that achieve high—although not the highest possible—accuracy while also allowing a project to evolve organically. This approach emphasizes the combination of various off-the-shelf technologies to answer initial research questions or develops new generalized tools to facilitate a more thorough understanding of the data. The intention underlying these adaptable, generalized tools is the ability to use them on more than one project. Such tools likely include established research methods like OCR, full-text search, image processing, computer vision, classical ML models, and/or LLMs.

Custom models still have a place in this methodology. A flexible, iterative approach to the research question aims to minimize the risk of overengineering at too early a stage of data exploration while still allowing for investment in specialized tools at points when the return is greatest. LLMs are powerful tools, but efficient usage—in

terms of money, time, and performance—is key, especially when working with a large corpus of documents. Pairing LLMs with an intermediate OCR step can help target LLM work—both in terms of the number of items reviewed and length of these items.

An ensemble of tools does not have the appealing simplicity of relying on a single LLM. Designing and creating generalized tools for a multi-step approach requires a larger breadth of technical knowledge while not *always* performing better than an LLM alone. The performance gap between a modular approach and an LLM alone will decrease as LLMs continue to grow and improve; however, the considerations around monetary costs, computing costs, and flexibility will remain relevant. Further, without substantial decreases in the cost of frontier mulitmodal LLMs, using them alone can be wasteful when there are other, cheap ways to exclude a large share of null results.

3 Case Study

3.1 Data Trapped in Property Deeds

The legal process of recording property transfers generates a particularly rich administrative dataset. Deeds have been used to study a wide range of economic, legal, and historical subjects. Key examples include the long run effects of tangled titles (Hincken 2021), property transfer under intestacy (Stein and Carpenter 2022), building development and use history, genealogy, environmental history, and the concentration of and pattern of land ownership (Storey 2018). These topics provide key insight into the functioning of the real estate market—a large driver of the US economy—and therefore into inflation, employment, and financial stability for the economy as a whole.

An additional topic of ongoing research is the impact of historical zoning and use restrictions on housing access, affordability, and quality (Asquith, Mast, and Reed 2023; Bartik, Gupta, and Milo 2025; Shertzer, Twinam, and Walsh 2016). Before the legal ascendancy of state enforced zoning, properties were subject to legal covenants that performed a similar, privately initiated function (Korngold 2001; Weiss 1987). Covenants in property deeds can restrict use, ownership, occupancy, and a variety of building characteristics, such as height, setback, and exterior finish. Historically, covenants could exclude ownership and occupation based on race, religion, country of origin, or citizenship (Santucci 2020; Surani et al. 2024).

Racial covenants were one component of a broad arsenal of tools that White developers, builders, brokers, and homeowners deployed throughout the 19th and 20th centuries to control where and how Black and other minority groups could live.[2] The legality of these covenants was firmly established in *Corrigan v. Buckley* (1926).

In 1948, the Supreme Court ruled in *Shelley v. Kraemer* that state courts could not legally enforce racial covenants, although they remained viable as private contracts. The Fair Housing Act, enacted as Title VIII of the Civil Rights Act of 1968, ultimately ended the legality of these covenants altogether.

Extensive scholarship has documented the presence and location of these covenants across the United States (Delegard et al. 2026; Gordon 2023; Plotkin 2001; Winling 2026). However, efforts to quantify their long-term impact on segregation, housing

2. Language restricting by religion, citizenship, or ethnicity is typically referred to in the literature as a "racial" covenant.

wealth, job access, and other outcomes remain in their relative infancy (Almagro and Sood 2026; Plotkin 2001; Surani et al. 2024).

Our research focuses on documenting the prevalence and geographic distribution of racially restrictive covenants in Philadelphia and on measuring the long-term effects of these restrictions. As far as we have determined, this information exists only within property deeds.[3] Therefore, we first identify and extract relevant restrictive and geographic language from each deed.

The language used in these covenants varies greatly across cities, depending on local conditions and time period. In some cities, it is highly specific, comprising a laundry list of racial categories. In others, it is more flexible, granting buyers and sellers latitude in whom they excluded (Gordon 2023). Such flexibility in exclusion often manifested as a statement of *inclusion*, i.e., who *could* occupy a property rather than who could not. The most common language in Philadelphia falls into this latter category, with most restrictions prohibiting occupancy by those "not of the Caucasian race."[4]

After locating the racially restrictive deeds, we identify the modern parcels corresponding to these deeds. Only approximately 30% of covenanted deeds in our data contain an address. For the remaining 70% of covenanted Philadelphia deeds, we must rely upon a "metes and bounds" description from the deed. This is a textual, legal description of the location and boundaries of a property as determined relative to a "stable" point of reference. We are developing an LLM-assisted process to map these boundaries, a full discussion of which is in preparation as a separate manuscript.

3.2 Philadelphia Property Deeds

Through an ongoing partnership with the Philadelphia Department of Records (DOR), we obtained digital images of property deeds from 1910–1965. This dataset contains 4,682,704 images representing at least 7.3 million pages and 1.6 million deeds. The images capture the deeds in a continuous stream, varying between one and two pages per image, with no digital demarcation between documents. For this reason, we report image counts, not pages. Discussion of methods for dividing images into individual deeds is beyond the scope of this article.

The deed images were scanned from microfilm of the original deed books produced by an initiative launched in 1949 (Hughes 1958). Both the condition of the film at time of scanning and the quality of the original image captured varies substantially—ranging from clear to completely illegible.

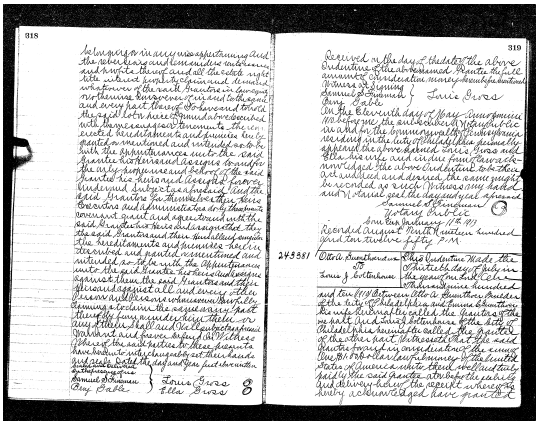
Philadelphia's deeds were recorded in three primary formats: 1) transcribed by hand, 2) transcribed by typewriter, and 3) photostats[5] of the original deed documents. Deeds were transcribed by hand until mid-1913, transcribed by typewriter from mid-1913–1934, and recorded with photostat copies from 1935 onward. Approximately 3.7% of images are handwritten, 28.5% of images are typed, and 67.8% of images are photostats. Nearly all handwritten and typed images are of two pages; in contrast,

3. Where restrictive language was recorded varies by city/county throughout the country.

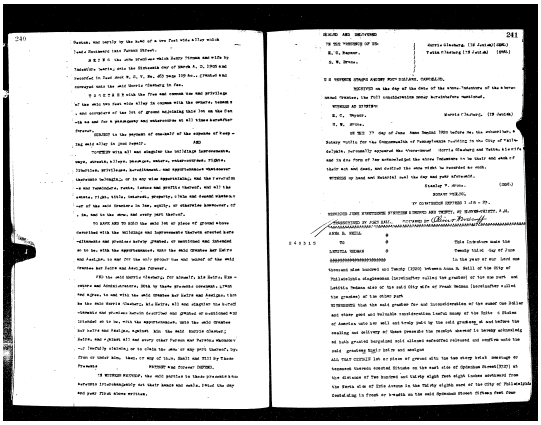
4. By their very nature, the documents explored in this project contain language that is discriminatory, racist, and offensive. The use of such language in this analysis is for research and historical purposes and is in no way an endorsement or approval of its use.

5. The photostat machine was an early photocopier that used camera film instead of toner and produced white-on-black negative copies.

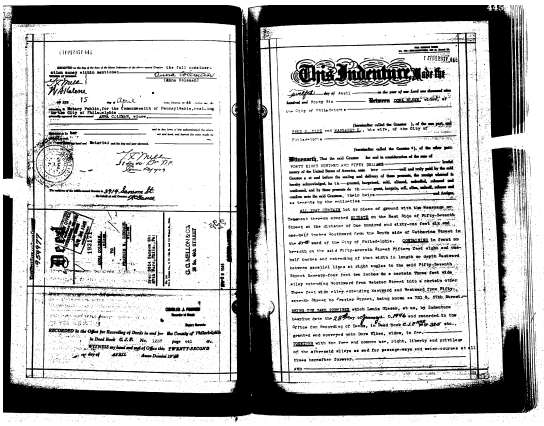
60% of photostat images are of single pages. Photostats are almost entirely double-page through 1951 and single-page thereafter. Figure 1 provides examples of two-page handwritten, typed, and photostat images, as well as of a single-page photostat.



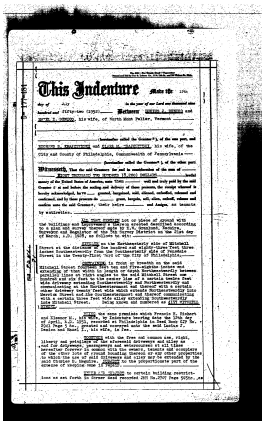
(a) Handwritten deed; double page image



(b) Typed deed; double page image



(c) Photostat deed; double page image



(d) Photostat deed; single page image

Figure 1: Examples of the four most common image types in our data.

We determined that there were two significant time periods—the early 1930s and the late 1940s—where a large number of deed images were missing from DOR's digital archive. The late 1940s gap was particularly concerning as this covered the years immediately before and after *Shelley v. Kraemer*, an important period for identifying potential treatment effects of the verdict.

For these gaps, we worked with the City of Philadelphia to scan existing microfilm or original deed books. This added 414,851 additional images, bringing the total number of images to 5,097,555. The additional images are all of photostat deed documents and are imaged at a much higher resolution than DOR's existing digital archive. We fully searched these incremental images for racially restrictive language, but we do not include them in our performance and cost measures reported in §5. Including them would bring more complexity to the paper with negligible impact on performance statistics or takeaways. We include covenanted deeds found in these incremental images in the totals reported in §5.7 with more detail given in Appendix H.

4 Methodology

4.1 Overview

Using Philadelphia property deed images, we evaluate the performance of different combinations of OCR, full-text search, and LLMs for classifying unstructured text. The classification task is to label an image as containing racially restrictive language. The approach to this task should generalize to similar textual classification tasks.

We explore different methodological approaches through the lens of the the framework in §2 and provide practical data on OCR and LLM performance. Our particular research question involves a highly variable and often low quality corpus of images, a rare event, and many secondary questions beyond covenant identification. Therefore, our work was not aligned with the cost of utilizing fully customized ML models.

We find that a hybrid choice combining various technologies offers a good combination of performance and ease of use. In our application, this means fuzzy full-text search to eliminate obvious nulls—non-covenant containing images—from our corpus, even when images are of low quality.

Despite the utility of fuzzy full-text search, using sufficiently broad search terms to ensure locating as many racial covenants as possible produced too large a number of positive matches to evaluate by hand. This is where LLMs meaningfully impact the scale at which research is possible.

We evaluate classification and cost performance for three approaches that combine traditional methods and LLMs to varying degrees. Each of these methods gets progressively closer to the "one-step" extreme described in §2.2. First, we used an ensemble method, passing all images through OCR, filtering the OCR text using fuzzy full-text search, selecting the text proximate to the matching words or phrases, and then passing this selected text—called a "snippet"—to an LLM for classification. Second, a streamlined method, again starting with the text output by an OCR engine, but without filtering and sending all of the OCR text associated with an image to an LLM for classification. Finally, a method skipping the intermediate step of OCR entirely and simply sending the whole image to an LLM for classification. Figure 2 illustrates these three methods.

4.2 Sample

We report performance statistics based on a subset of 279,131 images selected from our initial corpus of 4,682,704 images. Of these, 166,343 images were selected using fuzzy full-text search with OCR text from Amazon Textract generated in November 2021.

This search was very broad and sought to identify all images that might contain a possibly relevant word or phrase (see §4.5 and Appendix E) suggesting a racial covenant. All images matching any of the search terms listed in Appendix E are included in the subset.

A further 8,954 images were added to the subset either due to belonging to a deed known to contain an image with a racial covenant or belonging to twenty racially restricted deeds found using an older method but not matching any search term using Textract 2021 text (Santucci 2020).

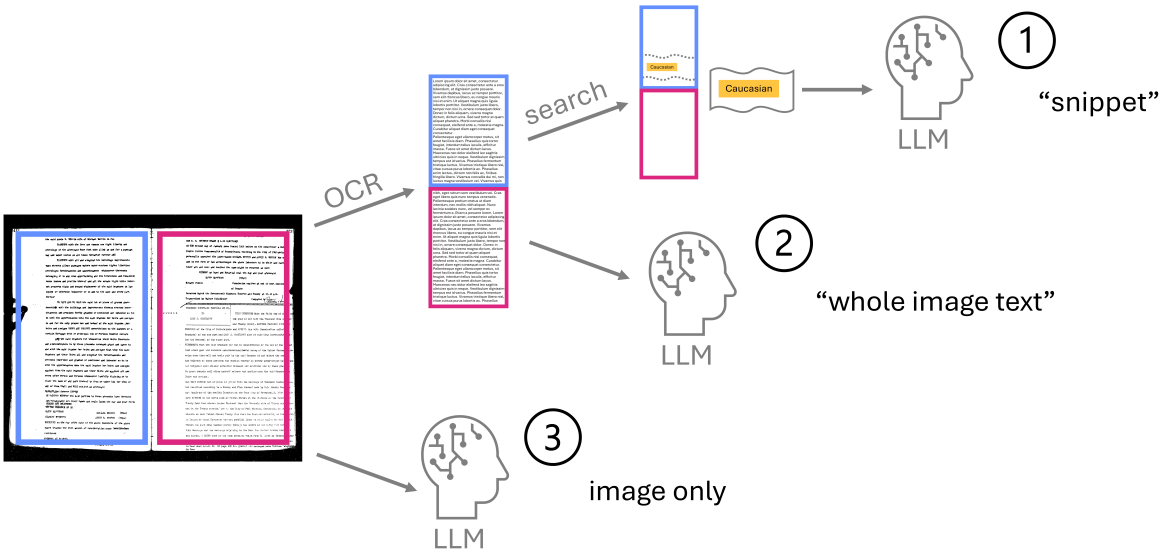


Figure 2: Three approaches which combine traditional methods and LLMs to varying degrees for classification of text in images. Approaches are shown in order of increasing reliance on and token consumption by LLMs. The terminology here is used throughout the paper.

The remaining 103,834 images in the subset are a random 4.3% sample of all remaining deed images recorded between 1910–1950 and a 0.11% random sample of all deeds recorded from 1951–1965. In line with the local and national post-WWII building boom (Wattenberg 1976), and compounded by the shift to one page per image as opposed to two pages per image prior to this date, the volume of deed images explodes from 1951 forward. At the same time, the number of covenants found via full-text search (see §4.5) fell to almost zero—despite excellent image quality and clean OCR output. It was cost-prohibitive to include a large sample of these later deeds, which did not match any search term related to the language used in racial covenants, for very little likely return in the form of true positives. See §5.4 for the large costs associated with exploring this time period at greater scale via whole image text classification. See Appendix D for an in-depth discussion of the random component of the sample.

4.3 OCR Models and Implementation

We selected OCR models from commonly used and referenced open source and paid tools. For paid choices, we tested Amazon Textract and Gemini 2.5 Flash. Deed images were processed with Amazon Textract in November 2021 and again in December 2025. Appendix I provides a comparison of Textract's evolution between these dates.

Most open-source OCR models are optimized for either handwritten or printed text, but rarely for both. For handwritten deeds, we used Kraken (Kiessling 2025), TrOCR (Li et al. 2022), and olmOCR (Poznanski, Soldaini, and Lo 2025). For typed and photostat deeds, we used docTR (Mindee 2021) and olmOCR. Table 1 summarizes our OCR model choices. Gemini 2.5 Flash and olmOCR are LLM OCR models, while the others are based on conventional ML techniques. Usage of any specific software or

Table 1: OCR models tested to extract machine-readable text from Philadelphia property deed images.

Image Type	Paid OCR	Open Source OCR
Handwritten	Amazon Textract Gemini 2.5 Flash	Kraken 6.0.3 TrOCR-base-handwritten olmOCR-2-7B-1025-FP8
Typed & Photostat	Amazon Textract Gemini 2.5 Flash	docTR 1.0.0 olmOCR-2-7B-1025-FP8

model is not an endorsement.

Textract, docTR, and Kraken are self-contained, and we found them to work out of the box without other tools or adjustments. A significant limitation of TrOCR is that it can only process an image containing a single line of text. We used line-level segmentation from Kraken to split each image into individual lines for transcription with TrOCR. For olmOCR, we found it necessary to set a `repetition_penalty` of 1.15 and to increase the `temperature` to 0.15 to curb loop hallucinations in the OCR output. For the definition of these parameters, please see the glossary in Appendix A. When using Gemini 2.5 Flash for OCR, we capped thought and output tokens to prevent explosive costs.

We did not perform global pre-processing on images in our corpus. We identified that olmOCR was sensitive to image size and tested performance on both raw images and on images where the left and right pages were split into separate images. Splitting images into single pages greatly improved the performance of olmOCR. OCR performed via Gemini 2.5 Flash was also sensitive to single-page images vs. double-page images, although much less so than olmOCR. Splitting images into separate pages doubles the cost of Textract, and Textract with layout costs 2.5 to 5 times more than standard Textract. Only a small improvement was found when using split images with Textract while almost no difference was found by utilizing the expensive layout parsing feature (see Tables 27 and 28 in Appendix H). Appendix F discusses practical considerations for using various OCR methods.

4.4 OCR Evaluation

We partnered with a data labeling firm to obtain human transcriptions with bounding boxes for all images belonging to a deed identified as containing a racial covenant. Due to cost considerations, only portions of the deed relevant to our econometric research were transcribed.[6] A total of 36,087 transcriptions were obtained.

Each transcription was aligned with OCR output from that image using the Smith-Waterman algorithm as implemented by the Python `string2string` package (Suzgun, Shieber, and Jurafsky 2023). Character error rate for each transcription was calculated using the standard formula $CER = (S+D+I)/N$ where S, D, and I are the number of

6. The transaction date, addresses when given, metes and bounds descriptions for all properties listed in the deed, the previous transaction information, and the racially restrictive covenant.

character substitutions, deletions, and insertions. N is the total number of characters in the transcription text. If an OCR model did return any output for an image, any transcriptions in that image were assigned $CER = 100\%$.

For each OCR model, we compute the average CER across transcriptions weighted by N.[7] We denote this value as $\overline{CER}_w$. A full description of our transcription data and CER calculation is given in Appendix G.

4.5 Full-text Search

For any city, the share of deed images containing racially restrictive language will be significantly smaller than the share of racially covenanted properties. Two factors drive this difference. First, deeds span many images, of which typically only one contains relevant language. Second, a covenant is usually not stated *explicitly* each time a racially restricted property is transferred. Instead, deeds refer to "restrictions of record"—which may or may not be racial—established in earlier transfers.

Both factors reduce the frequency of the target event—explicit racial covenant language—as outlined in our framework (§2). It was therefore beneficial to winnow the images and text passed to an LLM for classification by removing obvious nulls. Our search methodology aimed to efficiently maximize discovery of racially covenanted deeds while discarding as many clear null results as possible.

Search must overcome two sources of error. First, even with excellent image quality, no OCR model will return an exact representation of the source material. Accuracy degrades further as a function of the quality of the input image. Given that some images in our corpus are illegible, we expect some OCR text to be garbled or nonexistent. Second, the text in the images is handwritten or typed by humans and contains innate typos which, with accurate OCR, will be faithfully reproduced.

Both conventional ML and LLM OCR suffer from the first of these issues. OCR performed with an LLM may be able to correct innate typos in the document, which could simplify searching for specific information, but the accuracy with which they do so is model-dependent and sensitive to prompting. An LLM's ability to correct typos can generate new issues, such as incorrectly "correcting" a misspelling or "reading" truly illegible text, thus corrupting the true meaning of the text. For example, we observed LLM OCR transcribing "Canadian" or "Bavarian" as a "correction" to instances of "Caucasian" that were difficult to read or contained typos (e.g., "Cancasian").

We employed full-text search to overcome the OCR issues mentioned above and to find lower signal phrases such as "shall not be sold." Full-text search combines database technology—for speed, repeatability, and ease of use—with a broad range of text search techniques that expand on the idea of a keyword search. The search techniques we used include allowing for other words to be inserted between search terms (slop), variable Damerau–Levenshtein (DL) edit distance (see Appendix A), phrase reordering, and natural language pre-processing techniques.[8]

The particular choice of tool for implementing full-text search is not particularly important as long as in includes the above-stated capabilities. We used OpenSearch, which is fast, flexible, and facilitates easy use, but which requires more technical ex-

7. This quantity is sometimes referred to as the micro-average.
8. Such as tokenization, stemming, case-insensitive, etc.

pertise to set up than other options. Usage of any specific software or model is not an endorsement.

We begin by fuzzy searching for words and phrases reported by previous work on racial covenants in Philadelphia (Santucci 2020) and other cities (Delegard et al. 2026; Gordon 2023; Winling 2026). We found that OpenSearch's maximum DL distance of two was not sufficient, given the errors in our OCR text. We therefore employed an iterative search strategy to circumvent this limitation. We isolated matching fuzzy terms from our initial search and then fuzzy searched for those fuzzy terms separately. Searching for a term that is already fuzzy effectively increases the maximum permitted edit distance of a match.

Beyond our iterative strategy, we searched for "words" created by removing the space(s) in a phrase. We also searched for parts of words or phrases on either side of typical line break hyphenation. A full table of search terms with the total number of returned matches and number of matches positively containing a racial covenant is given in Appendix E.

This iterative search reduced the number of candidate images by more than 94%. The remaining ∼6% was still too large for comprehensive human review but was well-suited for LLM classification.

4.6 LLMs and Covenant Classification

We tested three different modes of input for LLM racial covenant classification: a short "snippet" of OCR-produced text that contains text that matched a fuzzy full-text search, the entire OCR-produced text from an image, and just the image without OCR. See Figure 2 for a visualization of these modes.

To create a snippet for each image, we combined the words surrounding any matching word or phrase identified via full-text search into one block of text. Where regions matched by different search terms overlapped, we accounted for this redundancy to produce a clean block of text without repetition. Since multiple, disparate parts of the text from a single image could match our fuzzy search terms, two disparate parts of a document could be combined into one snippet. On average, each snippet is approximately 640 characters, or about 14% of all characters in the OCR text produced from the image.

For LLMs, we choose both closed-weight, pay per token models and open-weight, free to use models. Usage of any specific software or model is not an endorsement. We tested Claude Sonnet and Gemini 2.5 Flash for paid models. Originally, we tested both Sonnet 3.7 and 4.5. During the course of this project, Sonnet 3.7 was fully deprecated. Results in §5 were produced using Sonnet 4.5. Appendix H contains results from both versions of Sonnet.

We selected Gemini 2.5 Flash for direct image comprehension after initial testing revealed performance similar to that of Gemini 2.5 Pro, which charged about four times as much for input, output, and thought tokens. Even so, additional measures were necessary to control costs. Without caps, 2.5 Flash consumed many more "thought" tokens than 2.5 Pro, mostly erasing the cost savings from the difference in per token cost. We were able to lower costs for Gemini 2.5 Flash by capping thought and output tokens, which we did not observe to meaningfully affect accuracy. We further reduced cost by selecting a subsample of 100,539 images from the core sample (§4.2), which

Table 2: LLMs tested for racial covenant identification. Gemini 2.5 Flash only tested with image input, not with OCR text from other models. mistral-rrc publicly available from Surani et al. (2024).

Paid LLM	Open Source LLM
Claude Sonnet 4.5	gpt-oss-20b
Gemini 2.5 Flash	gpt-oss-120b
	mistral-rrc

contained all handwritten images in the core sample, all images from the core sample known to contain a racial covenant, and a 30% random sample each for typewritten and photostat images from the remaining core sample. All Gemini 2.5 Flash results are appropriately weighted to reflect the difference in samples.

For open-weight models, we evaluated gpt-oss-20b, gpt-oss-120b, and mistral-rrc of Surani et al. (2024). Their mistral-rrc model is Mistral 7B fine-tuned to locate racial covenants in OCR text from images of deeds containing typed text; handwritten deeds were omitted from the fine tuning process. Table 2 summarizes our LLM choices.

The `temperature` was set to zero for each LLM to enable the most deterministic results possible. For gpt-oss-20b, we found it necessary to set a `frequency_penalty` of 0.15. Not doing so resulted in catastrophic failure in approximately 20% of classifications—the model would get "stuck" in its own "thought," looping the same text until it hit a hard output token cap without ever producing a final label. In the end, gpt-oss-20b produced results comparable to the much larger gpt-oss-120b model but with significantly more upfront technical investment and experimentation.

All classification tasks were submitted via API for paid LLMs, via vLLM for gpt-oss-20b and -120b, and via Hugging Face Transformers for mistral-rrc. LLMs provided with OCR text for classification were given the same basic prompt, which informed the LLM it would receive OCR text of variable quality from a property deed and to assume the persona of a lawyer. For direct image input, a different prompt was used, incorporating instructions about how to consider the image in addition to the instructions given in the text classification prompt.

In all cases, the LLM was instructed that its response should be in JSON format with two fields: 1) a Boolean indicating the existence of a racial covenant, and 2) a "comments" string for the LLM to report its reasoning. Inclusion of the comments field greatly increased the likelihood the LLM returned a properly formatted JSON. That said, not all output was valid JSON, and post-processing was necessary to enforce standardization. Additional information about LLM prompts is provided in Appendix B.

Every image positively classified by any OCR/LLM combination as containing a covenant, and every image that contained language highly likely to be associated with a covenant but not receiving a positive LLM classification was manually reviewed. In all, we manually reviewed approximately 30,000 images. Passing our sample through dozens of combinations of LLMs and OCR models also allowed us to identify incremental new true positives due to variations across models. We are confident that the precision

and recall (see Appendix A) reported within our sample is accurate. We note that "comments" returned by an LLM *cannot* be used alone to validate LLM classification, as they may contain hallucinated reasoning or irrelevant information. They were found to be somewhat helpful in directing human reviewers during validation.

4.7 Cost Calculation

Cost estimates for processing all 4.7 million images were extrapolated from the core sample (§4.2). Reporting costs based only on the sample images would produce highly biased estimates as that subset was deliberately selected to be comparatively high-signal. This analysis represents the estimated costs if only one single method was selected from the start.

In the cost tables, estimated recall is reported over the entire corpus relative to the estimated *true* number of covenanted images in the corpus—a figure necessarily greater than the number of covenanted images identified as we can not claim 100% recall. See Appendix C for detailed explanation of how the estimated total true positive image count was derived for each image type.

Costs for our classification task fall into two main categories: fees for closed-source models and cost of GPU time for open-source models. We considered GPU time only and ignored the standard CPU resources required for threading models and calling APIs. These CPU costs tend to be small as they can be done on low cost equipment. We assumed zero overhead or development cost; therefore, our cost estimates are necessarily lower limits.

In all cases, we used standard, publicly available, non-enterprise pricing in our estimates. Textract charges per image, while LLMs charge by the number of input, output, and thought tokens used for each API call. For LLMs, total costs were calculated using average token counts and total image counts.

The second primary cost is the cost of GPU time for open-source LLMs and OCR models. All of our GPU work was done on either Amazon's g6.12xlarge (48 CPU, 4x L4 GPU, 96GB VRAM) or an in-house 4x H200 (128 CPU, 576GB VRAM) instance. For tasks completed on the g6.12xlarge, we took total compute hours and calculated cost using public pricing[9] of $4.6016 per hour. For those tasks completed on H200 infrastructure, we took total compute hours on one H200 and calculated cost using the public pricing of $7.9120 per hour per H200 for Amazon's p5pen.48xlarge.[10] Lower costs may be available from other providers. The cost for institutions with access to in-house compute resources or enterprise pricing will likely be lower than our estimates. As of submission of this paper, the retail cost of an H200 GPU was approximately $30-33,000 plus the costs of set up, operation, and maintenance.

The cost of running OpenSearch varies greatly depending on the exact architecture used, but it cannot practically be run on a standard laptop. If the underlying infrastructure is set up to run only when requested, cost per search can be extremely low, even for hundreds of gigabytes of text. Given that vanilla Python can replicate OpenSearch's functionality on low cost hardware, albeit at slower speeds, we made the

9. https://aws.amazon.com/ec2/pricing/on-demand/

10. This instance actually comes with eight H200s, so achieving these costs would require efficient use of all all eight GPUs simultaneously.

simplifying assumption that full-text search costs are trivial compared to model and GPU costs. We therefore did not include full-text search costs in our estimates.

5 Results

We present the performance of the best performing combinations of paid and open-source OCR and LLMs on images in our core sample (§4.2). Table 3 frames the size and scale of our data. Tables 4, 5, and 6 quantify performance by image type (handwritten, typewritten, and photostat). Table 7 provides a baseline measure of recall with OCR and fuzzy search only. We then extend our performance and cost analysis to the entire corpus of images in Tables 11, 12, and 13. Appendix H provides comprehensive performance statistics for all tested combinations of OCR and LLMs.

Table 3 summarizes by image type the number of images in our corpus, the number of images in our sample, the number of images containing racially restrictive language in our sample, our estimate for the true number of images containing racially restrictive language in our corpus,[11] and the associated estimated rate of occurrence for images with racially restrictive language in the entire corpus. Appendix C explains how we arrive at the estimate for the total number of covenanted images—and therefore the estimated rate of occurrence—in the corpus of 4.7 million images.

Table 3: Summary of number of images in corpus and sample; number of images containing racially restricted language (Cov. Images) located in sample and estimated to exist in corpus (see Appendix C). † In this table, all deeds from 1913 are classified as typewritten despite including a small number of handwritten deeds. Tables 6, 7, and 9 exclude 1913. Without 1913, this row reads: 1,251,981; 130,864; 5,990; 6,338; 0.50%

Type of Image	Images in Corpus	Images in Sample	Sample Total Cov. Images	Est. Total Cov. Images	Est. Rate of Occurrence
Handwritten	146,164	17,030	617	745	0.51%
Photostat	3,233,781	126,292	1,033	1,171	0.04%
Typed†	1,302,759	135,809	6,140	6,488	0.50%

5.1 OCR Performance

We calculate the average character error rate (CER) weighted by transcription length ($\overline{CER}_w$) for 36,087 transcriptions from images in deeds containing racial covenants. These values are presented for select OCR engines separated by image type in Tables 4, 5, and 6, while $\overline{CER}_w$ for all OCR engines tested is given in Appendix G.

Across all OCR engines, we observe a highly right-skewed distribution of CER values, with most engines having at least some transcriptions with CERs above 90%. Figure 3 compares the image CER of docTR and Gemini 2.5 Flash single page OCR against the image CER of Textract. The image CER is the average CER weighted by transcription length across all transcriptions within an image. A one-to-one line is

11. We do not claim 100% recall outside of the core sample. See Appendix C.

included to facilitate direct comparison between engines, with points falling above this line indicating better performance by Textract. A simple linear regression (solid line) is also shown to summarize the overall trend in the data.

We find that docTR, a ML OCR engine, performs almost universally worse than Textract on photostat and typed images.[12] Conversely, we find that Gemini 2.5 Flash performs much better on handwritten images and, on average, better than Textract on photostat and typed images.

For typed images, there is a cluster of points in the top left of the figure indicating images for which Gemini failed catastrophically but Textract performed very well. This effect is less pronounced but still present for photostat images. We observe similar clustering when comparing olmOCR to Textract. Poor-performance outliers become more common when comparing whole image OCR for olmOCR or Gemini to Textract. These comparison figures, along with others not described here, can be found in Appendix G.

Clusters of very bad performance for LLM OCR on images handled with ease by traditional ML OCR suggest that some caution is necessary when using LLM OCR. Low average CER can hide catastrophic failures.

While better CER by itself is desirable, we view these measurements as a tool to understand overall comparative performance on our classification task. The interpretation of these results in this context is presented in §5.3.

5.2 Classification Performance on Sample

We present precision and recall for our sample split into handwritten (Table 4), photostat (Table 5), and typewritten images (Table 6). For each table, precision, recall, F1, and % max recall are colorized by percentile, with the lowest values in red and the highest in green.

These tables include the best performing combinations of paid and open-source models. For photostat and typewritten deeds, we also include the performance of docTR because olmOCR on single pages requires significant run time on expensive GPUs (Tables 12 and 13). For handwritten deeds, we found all open-source OCR engines besides olmOCR to perform so poorly out of the box that we do not consider them to be practically viable. See Appendix H for TrOCR, Kraken, and Chandra results.

The performance tables are presented in order of decreasing complexity for both OCR and LLM classification. Handwriting is notoriously difficult for OCR, and in our corpus, the quality of handwritten images is relatively low. Photostats directly image the original deed documents which, while mostly typewritten or typeset, are complicated in layout with variable fonts, font sizes, and text orientation, and tend to have very low image quality. For the most part, the typewritten images in our dataset are simply structured, consistent in font and font size, and legible to the human eye.

In addition to separating by image type, we calculate performance on whole image text and snippets separately. The column "% Max Recall" compares the relative recall of each method to the ex-post best performing method—max recall of the row divided by the max recall of the table. Results from the image only classification task are reported in the "whole text" column.

12. docTR was not run on handwritten images as the text recognition models included in the package are not optimized for handwriting.

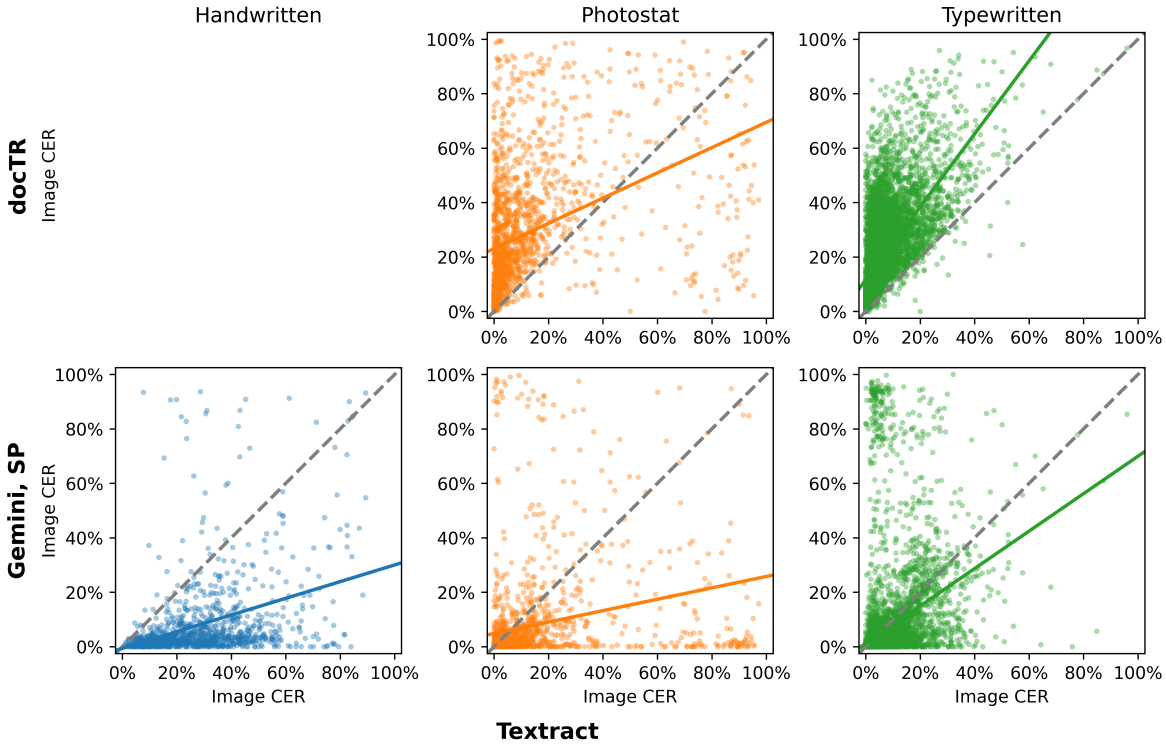


Figure 3: Image CER vs. Textract image CER for docTR (top) and Gemini 2.5 Flash (bottom). The gray dashed line represents one-to-one correspondence of CER between engines. Points above the one-to-one line indicate better performance by Textract. The solid line is a simple linear regression. Figures in the first column (blue) are handwritten images, in the second column (orange) are photostat images, and in the third column are typed images (green).

Focusing solely on performance, the highest recall for handwritten and photostat documents is produced by Gemini 2.5 Flash applied directly to input images ($R = 94.0\%$ and 96.5%) and by Textract paired with Sonnet 4.5 for typewritten images ($R = 97.9\%$). For photostat and typewritten images, however, the difference in recall between Gemini 2.5 Flash on images and Textract with Sonnet 4.5 is fairly small ($< 4\%$). Precision suffers with Gemini 2.5 Flash for direct image comprehension, particularly with photostat images, which increases the workload of human verification due to the increased number of false positives. When comparing methods by F1 score, the harmonic mean of precision and recall (see Appendix A), the best overall performance for all image types is Textract paired with Sonnet 4.5.

Holding OCR and LLM combination constant, the difference in recall between whole text and snippet is driven by three things: 1) recall of full-text search (R_{FTS}), the share of covenanted images that match any search term, 2) the usefulness of the information contained in the snippet, i.e., does it contain enough context for accurate LLM classification, and 3) the shorter length of the snippet. The snippet's shorter length provides less information for an LLM to consider, increasing the likelihood of accurate comprehension. This sensitivity to input length varies considerably across models. Equations 4 and 5 in §5.6 put these factors into functional form.

For a given pairing of OCR and LLM, recall between whole image text and snippets is fairly similar. Frontier LLMs consistently perform slightly better when classifying whole image text, whereas open-weight LLMs are more likely to perform poorly with whole text inputs. This relationship is primarily a function of OCR quality, as proxied by image type and OCR model quality. In other words, frontier LLMs can better interpret messy, garbled text. Some part of this difference may also be driven by the much larger context window in frontier models relative to open-weight models.

In Table 7, we present a simple experiment to demonstrate the difference in recall between full-text search alone and in conjunction with an LLM. In Philadelphia, almost all racially covenanted deeds contain the word "Caucasian," *and* this word is almost never mentioned in any context outside a racial restriction.[13] Therefore, in theory, if this word could be consistently identified, we could locate covenants with very high precision and reasonable recall. Accordingly, Table 7 presents the results of a single fuzzy full-text search for the word "Caucasian" in the Textract output for our sample, allowing a maximum DL edit distance of two.

The cleanest comparison is between Table 7 and the Textract results in Tables 4, 5, and 6 because this holds OCR source constant. For handwritten images, fuzzy search for "Caucasian" leads to a full-text search recall of $R_{\text{FTS}} = 62.4\%$ compared to our highest possible Textract-based recall given full-text search of $R_{\text{LLM|FTS}} = 86.5\%$ (Table 4, second row). This gain in recall highlights how LLMs and OCR complement each other—especially when raw OCR quality is poor. Compare instead the recall values for typewritten documents, where the OCR quality is much higher. Full-text search alone produces a recall of $R_{\text{FTS}} = 80.9\%$, which is proportionally closer to the best Textract $R_{\text{LLM|FTS}} = 96.8\%$ (Table 6, second row). $R_{\text{FTS}} = 80.9\%$ with Textract is in fact slightly higher than the best combination of open-source OCR and LLM of

13. We found four out of 5.1 million images in which "Caucasian" appeared but was *not* part of a racially restrictive property covenant. One each describing the race of a buyer (itself a sign of a biased transaction), a piece of furniture, a neighborhood, and a racially restricted scholarship at Temple University.

$R_{\text{LLM|FTS}} = 77.7\%$ (Table 6, last row). This illustrates a case where high-quality OCR alone (Textract) can outperform a cheaper OCR paired with an LLM, demonstrating that investing in better OCR is sometimes preferable to adding an LLM.

Table 4: Sample handwritten images: best recall for each possible combination of paid and open-source models.

OCR	LLM	Class	$\overline{CER}_w$	Whole Text Precision	Snippet Precision	Whole Text Recall	Snippet Recall	F1	% Max Recall
	Gemini 2.5 Flash	None/Paid		92.1%		94.0%		93.0%	100.0%
Textract	Sonnet 4.5	Paid/Paid	28.1%	98.4%	99.3%	89.1%	86.5%	93.5%	94.8%
olmOCR, SP	Sonnet 4.5	Open/Paid	11.2%	99.0%	99.6%	79.9%	78.3%	88.4%	85.0%
olmOCR, SP	mistral-rrc	Open/Open	11.2%	99.8%	99.8%	74.7%	78.8%	88.0%	83.8%
Textract	gpt-oss-20b	Paid/Open	28.1%	82.5%	97.0%	70.5%	74.1%	84.0%	78.8%

Table 5: Sample photostat images: best recall for each possible combination of paid and open-source models.

OCR	LLM	Class	$\overline{CER}_w$	Whole Text Precision	Snippet Precision	Whole Text Recall	Snippet Recall	F1	% Max Recall
	Gemini 2.5 Flash	None/Paid		77.8%		96.5%		86.2%	100.0%
Textract	Sonnet 4.5	Paid/Paid	10.9%	97.1%	99.5%	92.6%	89.2%	94.8%	96.0%
Textract	mistral-rrc	Paid/Open	10.9%	94.8%	95.9%	87.3%	88.2%	91.9%	91.4%
olmOCR, SP	Sonnet 4.5	Open/Paid	7.8%		99.1%		87.3%	92.8%	90.5%
olmOCR, SP	gpt-oss-20b	Open/Open	7.8%	81.2%	80.8%	86.8%	85.1%	83.9%	90.0%
docTR	Sonnet 4.5	Open/Paid	28.8%		97.6%		84.1%	90.4%	87.2%
docTR	gpt-oss-120b	Open/Open	28.8%	89.8%	91.8%	77.3%	76.3%	83.1%	80.1%

Table 6: Sample typewritten images: best recall for each possible combination of paid and open-source models.

OCR	LLM	Class	$\overline{CER}_w$	Whole Text Precision	Snippet Precision	Whole Text Recall	Snippet Recall	F1	% Max Recall
Textract	Sonnet 4.5	Paid/Paid	5.1%	99.6%	99.9%	97.9%	96.8%	98.7%	100.0%
	Gemini 2.5 Flash	None/Paid		98.4%		97.3%		97.9%	99.5%
olmOCR, SP	Sonnet 4.5	Open/Paid	3.3%		99.8%		95.0%	97.4%	97.1%
Textract	gpt-oss-120b	Paid/Open	5.1%	99.2%	99.6%	94.6%	94.4%	96.8%	96.6%
olmOCR, SP	gpt-oss-20b	Open/Open	3.3%	99.2%	99.0%	93.9%	92.6%	96.5%	96.0%
docTR	Sonnet 4.5	Open/Paid	18.4%		99.7%		87.7%	93.3%	89.6%
docTR	gpt-oss-120b	Open/Open	18.4%	97.9%	99.0%	79.7%	77.7%	87.9%	81.5%

5.3 Impact of OCR Quality on Classification

We note that ordering the performance tables by the max recall for each row ($\max(R_{WT}, R_{snip})$) demonstrates that weighted average character error rate is not a perfect proxy for OCR quality from an LLM-understanding standpoint. For example, Textract consistently outperforms olmOCR despite a worse CER.

A better CER for LLM-based OCR does not necessarily yield better classification performance, due to differences in *how* an LLM fails relative to a typical ML OCR

Table 7: Recall (R_{FTS}) for images containing "Caucasian" in a restrictive covenant using full-text search with Textract OCR. Only matches with Damerau-Levenshtein distance ≤ 2 are included.

Type of Image	Sample Total Cov. Images	Images Matching "Caucasian"	Recall (R_{FTS}) for Cov. with "Caucasian"
Handwritten	617	385	62.4%
Photostat	1,033	783	75.8%
Typewritten	5,990	4,969	80.9%

model.[14] An LLM is more likely to hallucinate coherent words in pursuit of fluent English, a bias that can *alter meaning* while producing a deceptively low CER. This failure mode is especially common with uncommon or hard-to-read words—"Caucasian" being a perfect example.

For example, in one case, illustrated in Table 8, olmOCR reads "Caucasian descent" as "Concession desert" (real English words) while Textract reads "Carcasion desent" (nonsense). Although olmOCR transcribes the *full* passage far more faithfully—7 total errors versus Textract's 20—its fluent misreading of the operative racial phrase is *worse*. Textract's garbled "Carcasion desent" is enough for the LLM to correctly label the passage as a racial restriction, whereas olmOCR's substitution of "Concession desert" leads the LLM to mislabel it entirely. A fluent misreading is more dangerous than a noisy but recoverable one. This is why CER is an imperfect, albeit still useful, proxy for OCR quality.

Further, we observe that higher-quality OCR produces viable results even when paired with smaller LLMs for classification. Conversely, more powerful LLMs can overcome low-quality OCR, as evidenced by Sonnet's relatively strong performance with docTR compared to either gpt-oss model.

Both of these observations are further demonstrated by Figure 4. Here we bin images by CER of Textract OCR and plot whole text recall for images within each bin. When CERs are low (i.e., $< 10\%$), all models achieve a recall near or above 90%, although Sonnet 4.5 does perform the best. After that, performance degrades much more slowly for larger models than for smaller models as OCR quality worsens. All LLM models perform poorly when CER is 80% or worse.

When considering the relative performance of an all-open-source approach, we exclude olmOCR for photostats and typewritten documents. olmOCR is more difficult to use and, given current GPU costs, is twice as expensive as Textract while producing worse results. With this exclusion, the best all-open-source combination of tools consistently produces a recall $\sim$80% of that of the best available combination.

Table 9 assesses Gemini as an OCR engine. We compare Claude Sonnet 4.5 recall using whole image text produced by Textract, Gemini 2.5 Flash with single page images, Gemini 2.5 Flash with double page images, and olmOCR with single page images.

14. In Appendix G we disprove an alternative hypothesis: that Textract's stronger classification performance, despite its higher average CER, could be driven by better Textract performance *on racially restrictive text specifically*. Relative differences in performance across models hold whether measured across all images, covenanted images, non-covenanted images, or the racial restriction passage alone within covenanted images.

Table 8: Example transcription by olmOCR single page and Textract of the same passage vs. human transcribed text. Words differing between the human transcribed and OCR transcribed text are highlighted. Edit distance does not allow adjacent transpositions, as is consistent with the calculation of CER.

	Human Transcribed	**olmOCR**	**Textract**
Text	And Also Subject to the following express condition and restriction to wit that any building erected on the above described premises shall not be sold to or used or occupied by any other person than one of the Caucasian descent	And also subject to the following express condition and restriction to wit that any building erected on the above described premises shall not be sold to or used or occupied by any other person than one of the **Concession desert**	and also subject to the following express condition and **restrution** to **with** that any building erected on the above **deserved** **memises** shall not **we** sold to **a** used or **armpin** by any other person than one of the **Carcasion desent**
Edit Distance	—	7	20
Key Phrase	Caucasian descent	Concession desert	Carcasion desent
Edit Distance	—	7	3
LLM Label	True	False	True

OCR produced by Gemini 2.5 Flash using single page images performed similarly to Textract. Gemini 2.5 Flash does not perform as well as Textract with double page images. Measuring recall when classifying snippet text improves Gemini Flash 2.5's performance slightly relative to Textract.

As shown in Figure 3 and by CER_{P95} in Table 9, the CER value corresponding to the 95th percentile of CER values from individual transcriptions, LLM OCR tends to fail catastrophically on documents that ML OCR models transcribe accurately. Table 10 contextualizes the impact of this "unexpectedly bad" LLM OCR on classification recall. Binning covenanted images by their Gemini single-page CER, it reports whole text recall from Sonnet 4.5, given Gemini single-page and Textract output for the same documents. Where Gemini CER is high, Textract not only has a much lower average CER but also categorically better classification recall. Crucially, the documents on which LLM OCR fails are not necessarily difficult OCR problems: we observe the same behavior when we reverse the experiment, binning by Textract CER and comparing against Gemini (Appendix H). This suggests that, where not cost-prohibitive, traditional ML and LLM-based OCR methods can be highly complementary, each filling in where the other struggles.

Table 9: Comparison of Gemini as OCR to Textract and olmOCR. Recall on sample images, whole image text, Sonnet 4.5. Recall colorized by percentile across all recall columns.

Text Type	**Textract**			**Gemini 2.5 Flash, SP**			**Gemini 2.5 Flash**			**olmOCR, SP**		
	R_{WT}	$\overline{CER}_w$	CER_{P95}	R_{WT}	$\overline{CER}_w$	CER_{P95}	R_{WT}	$\overline{CER}_w$	CER_{P95}	R_{WT}	$\overline{CER}_w$	CER_{P95}
Handwritten	89.1%	28.1%	71.7%	89.5%	7.4%	37.6%	84.4%	14.5%	78.5%	79.9%	11.2%	50.2%
Photostat	92.6%	10.9%	71.4%	92.1%	7.3%	54.7%	90.7%	16.4%	93.4%	88.3%	7.8%	56.4%
Typewritten	97.9%	5.1%	27.8%	96.9%	4.3%	34.9%	94.5%	8.2%	77.5%	96.3%	3.4%	19.5%

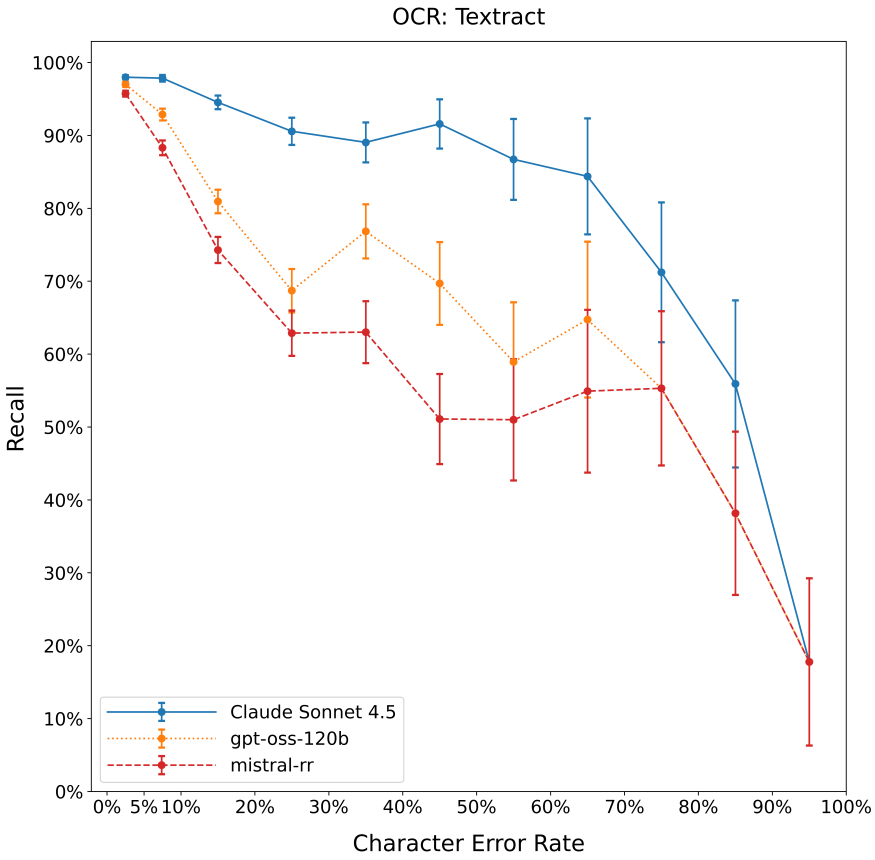


Figure 4: Recall using whole text generated by Textract by binned character error rate. X-axis tick labels indicate the left (inclusive) and right (exclusive) edges of the CER bin. Recall plotted for each bin with 90% Wilson score (confidence) interval.

5.4 Corpus Performance and Cost

We present estimated classification performance for the entire corpus and the estimated cost of processing the entire corpus split into handwritten (Table 11), photostat (Table 12), and typewritten images (Table 13). As explained in §4.7, these estimates are based on performance on the core sample extrapolated to the entire corpus.

As in Tables 4, 5, and 6, we present the best performing combinations of paid and open-source models applied to snippets, whole image text, or images only. For simplicity, the only open-weight LLM included in the cost tables is gpt-oss-120b as it was, on average, the best performing open-weight model across different OCR models and image types.

We present performance and cost of Gemini 2.5 Flash as OCR for single page images for handwritten images only. We find that the cost of Gemini 2.5 Flash is simply too high to consider beyond the most difficult images, here proxied by handwriting—$4.02 per thousand images or $5.50 per thousand images when separately OCRing left and right pages vs. $0.60-1.50 per thousand images for Textract.[15] As shown in Table 11, the results are not much better than those generated by significantly cheaper Textract.

Within each table, total cost, efficiency ratio, and cost per TP are colorized by

15. Reported costs for Gemini 2.5 Flash are for handwritten images; costs vary slightly for other image types. Textract provides a quantity discount, and price does not vary with image type.

Table 10: Comparison of whole image text recall between Gemini single page OCR and Textract when images are binned by Gemini CER. Recall colorized by percentile of value shared across both columns.

Gemini, SP CER Bin (%)	$N_{\text{cov img}}$	R_{Gemini}	R_{Textract}	Textract CER
[0, 5)	6,576	97.7%	97.2%	6.1%
[5, 10)	577	95.3%	93.4%	12.8%
[10, 20)	190	85.3%	91.1%	18.4%
[20, 30)	72	69.4%	87.5%	23.8%
[30, 40)	45	68.9%	93.3%	34.0%
[40, 50)	21	66.7%	100.0%	34.1%
[50, 60)	15	66.7%	100.0%	21.9%
[60, 70)	13	61.5%	100.0%	24.9%
[70, 80)	34	70.6%	94.1%	21.2%
[80, 90)	43	46.5%	93.0%	18.5%
[90, 100]	34	8.8%	97.1%	13.9%

percentile, with the worst values in red and the best in green. Recall is colorized similarly but the percentile ranking is shared *across all three* tables.

The efficiency ratio is the ratio of (% of max recall)/(% of cost of max recall method). In Table 11, for example, the efficiency ratio normalizes relative to the highest recall method: Gemini 2.5 Flash applied to images. Textract paired with Sonnet 4.5 on snippet text for handwritten images (Table 11, first row) yields 79.1%/90.0% = 87.9% of the best estimated recall for \$249/\$481 = 51.8% of its cost. This gives an efficiency ratio of 87.9%/51.8% = 170%.

The estimated cost tables are ordered by cost per racial covenant found. Ranking by efficiency ratio is in all cases equal to ranking by cost per racial covenant identified but not exactly the same as ordering by total cost. Once cost and efficiency ratio are considered, we see meaningful differentiation in the value of different combinations of OCR, LLM, and LLM input format (snippet, whole image text, image alone). From this perspective, the highest recall strategies look much less attractive. For example, see Gemini Flash 2.5 applied to images or Textract and Sonnet 4.5 on whole image text compared to Textract and Sonnet 4.5 on snippets or docTR and Sonnet 4.5 on snippets.

On the other hand, this table highlights the virtue of pairing relatively expensive OCR tools with full-text search and frontier LLMs for all document types. Since full-text search filters out irrelevant documents and facilitates construction of text snippets containing only the most relevant text from the image, most of the benefits in recall from using a frontier model can be realized without needlessly processing and incurring the cost of many irrelevant tokens. For all document types, the pairing of Textract, full-text search, and Sonnet 4.5 is an attractive method with relatively high recall for relatively low cost.

We find the most efficient combination for photostat and typewritten images to be docTR paired with full-text search and gpt-oss-120b (first row, Tables 12 and 13). Here, the highest efficiency method has a recall of approximately ∼75%, which is too low for

our research purposes. This highlights a potential downside of considering efficiency alone when selecting a method.

Table 11: Estimated performance and cost across entire corpus for handwritten images; table ordered by increasing cost per true positive found.

OCR	LLM	Method	Class	Recall	OCR Cost	LLM Cost	Total Cost	Efficiency Ratio	Cost Per TP
Textract	Sonnet 4.5	Snip	Paid/Paid	79%	$219	$30	$249	170%	$0.42
Textract	gpt-oss-120b	Snip	Paid/Open	62%	$219	$1	$220	151%	$0.47
Textract	gpt-oss-120b	WT	Paid/Open	63%	$219	$25	$245	138%	$0.52
	Gemini 2.5 Flash	Image	None/Paid	90%	$0	$481	$481	100%	$0.72
Gemini 2.5 Flash, SP	gpt-oss-120b	Snip	Paid/Open	82%	$803	$1	$804	54%	$1.32
Gemini 2.5 Flash, SP	gpt-oss-120b	WT	Paid/Open	82%	$803	$25	$829	53%	$1.36
Gemini 2.5 Flash, SP	Sonnet 4.5	Snip	Paid/Paid	82%	$803	$30	$833	52%	$1.37
Textract	Sonnet 4.5	WT	Paid/Paid	83%	$219	$806	$1,025	43%	$1.65
Gemini 2.5 Flash, SP	Sonnet 4.5	WT	Paid/Paid	82%	$803	$806	$1,609	27%	$2.63
olmOCR, SP	Sonnet 4.5	Snip	Open/Paid	73%	$1,943	$37	$1,980	20%	$3.64
olmOCR, SP	gpt-oss-120b	Snip	Open/Open	71%	$1,943	$1	$1,944	20%	$3.66
olmOCR, SP	gpt-oss-120b	WT	Open/Open	72%	$1,943	$25	$1,968	20%	$3.66
olmOCR, SP	Sonnet 4.5	WT	Open/Paid	76%	$1,943	$806	$2,749	15%	$4.83

Table 12: Estimated performance and cost across entire corpus for photostat images; table ordered by increasing cost per true positive found.

OCR	LLM	Method	Class	Recall	OCR Cost	LLM Cost	Total Cost	Efficiency Ratio	Cost Per TP
docTR: resnet+vgg16	gpt-oss-120b	Snip	Open/Open	72%	$626	$7	$633	1,217%	$0.75
docTR: resnet+vgg16	Sonnet 4.5	Snip	Open/Paid	80%	$626	$305	$931	918%	$0.99
docTR: resnet+vgg16	gpt-oss-120b	WT	Open/Open	74%	$626	$564	$1,189	662%	$1.37
Textract	gpt-oss-120b	Snip	Paid/Open	77%	$2,840	$9	$2,849	286%	$3.18
Textract	Sonnet 4.5	Snip	Paid/Paid	79%	$2,840	$356	$3,196	262%	$3.46
Textract	gpt-oss-120b	WT	Paid/Open	78%	$2,840	$564	$3,404	243%	$3.74
	Gemini 2.5 Flash	Image	None/Paid	94%	$0	$10,008	$10,008	100%	$9.08
docTR: resnet+vgg16	Sonnet 4.5	WT	Open/Paid	84%	$626	$16,022	$16,648	53%	$17.03
Textract	Sonnet 4.5	WT	Paid/Paid	84%	$2,840	$16,022	$18,863	47%	$19.17
olmOCR, SP	Sonnet 4.5	Snip	Open/Paid	84%	$28,826	$343	$29,169	30%	$29.84
olmOCR, SP	gpt-oss-120b	Snip	Open/Open	82%	$28,826	$8	$28,835	30%	$30.11
olmOCR, SP	gpt-oss-120b	WT	Open/Open	83%	$28,826	$564	$29,390	30%	$30.37
olmOCR, SP	Sonnet 4.5	WT	Open/Paid	85%	$28,826	$16,022	$44,849	20%	$45.23

5.5 Selecting a Method

We do not think the most "efficient" choice is always the best choice. It's up to the end user to determine the balance between cost per find and recall—it may be worthwhile to pay twice as much per find for increased recall. In the typewritten case, for example, Textract paired with full-text search and an open-weight LLM looks particularly attractive as you can achieve recall above 90% while still keeping total costs fairly low (Table 13, fourth row).

There are some permutations of OCR, LLM, and text input method that are sub-optimal by nearly any metric. For example, in Table 12 (photostat estimated performance), docTR/Sonnet 4.5 with snippet text (second row) is less expensive with higher

Table 13: Estimated performance and cost across entire corpus for typewritten images; table ordered by increasing cost per true positive found.

OCR	LLM	Method	Class	Recall	OCR Cost	LLM Cost	Total Cost	Efficiency Ratio	Cost Per TP
docTR: resnet+vgg16	gpt-oss-120b	Snip	Open/Open	76%	$253	$6	$259	3,219%	$0.05
docTR: resnet+vgg16	Sonnet 4.5	Snip	Open/Paid	86%	$253	$245	$498	1,899%	$0.09
docTR: resnet+vgg16	gpt-oss-120b	WT	Open/Open	76%	$253	$227	$480	1,749%	$0.10
Textract	gpt-oss-120b	Snip	Paid/Open	92%	$1,682	$6	$1,688	596%	$0.28
Textract	Sonnet 4.5	Snip	Paid/Paid	94%	$1,682	$263	$1,944	531%	$0.32
Textract	gpt-oss-120b	WT	Paid/Open	92%	$1,682	$227	$1,909	528%	$0.32
	Gemini 2.5 Flash	Image	None/Paid	95%	$0	$4,140	$4,140	252%	$0.67
docTR: resnet+vgg16	Sonnet 4.5	WT	Open/Paid	88%	$253	$8,788	$9,041	107%	$1.59
Textract	Sonnet 4.5	WT	Paid/Paid	95%	$1,682	$8,788	$10,469	100%	$1.69
olmOCR, SP	Sonnet 4.5	Snip	Open/Paid	92%	$17,318	$243	$17,561	58%	$2.93
olmOCR, SP	gpt-oss-120b	Snip	Open/Open	91%	$17,318	$6	$17,324	57%	$2.95
olmOCR, SP	gpt-oss-120b	WT	Open/Open	91%	$17,318	$227	$17,545	57%	$2.98
olmOCR, SP	Sonnet 4.5	WT	Open/Paid	94%	$17,318	$8,788	$26,106	39%	$4.30

recall than Textract/gpt-oss-120b with snippet text. One would choose the latter option only when constrained by specific model requirements.

All three cost tables raise serious questions about the value of using an LLM directly for image comprehension without intermediate OCR. Currently, LLMs are relatively expensive per image while offering marginal improvements in recall compared to several other methods. There is no way to remove obvious nulls from the dataset without OCR output when using LLMs directly for image comprehension, unnecessarily inflating costs for large datasets with rare events. Further, our efficiency measures do not capture the value of OCR output as a separate data product (§2.3).

The cost quoted in our tables for using an LLM without intermediate OCR is for a single classification task. Should the first task reveal a second question to be asked of the corpus, the quoted cost would need to be fully incurred a second time. With intermediate OCR, the OCR cost is incurred once. That data product can be used infinitely with full-text search for any future research question to filter to images of interest and then leverage LLM classification at much lower cost. This OCR data product is valuable for data exploration, for example, when looking for particular buyers and sellers without the need for recall augmented by an LLM.

5.6 Functional form of cost considerations

Abstracting from empirical results, cost per find can be expressed in functional form. Such a formula can be used to understand what drives cost and efficiency, and provides a way to estimate costs for any project classifying unstructured text in images with an LLM. We present this formula in terms of dollars per image OCR'd and cost per token assuming a pay to use OCR model and a frontier model. These terms can instead be viewed as hours of GPU runtime converted to price, given the cost of the hardware used.

By assuming trivial overhead and search costs (§4.7), we simplify cost per find to the ratio of combined OCR and LLM costs to the number of items correctly classified.

$$\text{Cost Per Find} = \frac{\text{Total Cost}}{\#\text{ of Finds}} = \frac{\text{Cost}_{\text{OCR}} + \text{Cost}_{\text{LLM}}}{\#\text{ of Finds}} \tag{1}$$

The Cost_{OCR} component is a product of the number of images (N_{images}) in the corpus and the cost per image of OCR (P_{image}).

$$\text{Cost}_{\text{OCR}} = N_{\text{images}} \times P_{\text{image}} \tag{2}$$

The LLM cost can be decomposed into the number of candidate images selected from the starting corpus using full-text search (N_{FTS}) scaled by the resulting token costs (T for token counts, P for price per token, and bar for average). Tokens and their associated price can be split into input, thought, and output. T_{prompt} are the input tokens from the prompt and are assumed to be constant. $\overline{T}_{\text{doc}}$ is the average number of input tokens in the content passed to the LLM for classification; input tokens may be text tokens or image tokens.

$$\begin{aligned}
\text{Cost}_{\text{LLM}} &= N_{\text{FTS}} \times \overline{T} \times \overline{P} \\
&= N_{\text{FTS}} \times \Big[\big(\overline{T}_{\text{doc}} + T_{\text{prompt}} \big) \times P_{\text{input}} \\
&\qquad + \overline{T}_{\text{thought}} \times P_{\text{thought}} \\
&\qquad + \overline{T}_{\text{output}} \times P_{\text{output}} \Big]
\end{aligned} \tag{3}$$

When considering all documents without full-text search filtering, $N_{\text{FTS}} = N_{\text{images}}$. Otherwise, N_{FTS} is a function of the number of images in the corpus, the true underlying true positive rate (k_{true}), the recall rate of full-text search filtering (R_{FTS}), and the false positive rate of the full-text search filtering (k_{FPR}).

$$N_{\text{FTS}} = \underbrace{N_{\text{images}} \times k_{\text{true}} \times R_{\text{FTS}}}_{\text{true positives}} + \underbrace{N_{\text{images}} \times k_{\text{FPR}}}_{\text{false positives}} \tag{4}$$

The number of items positively identified is a function of the total number images, the underlying true positive rate, the true positive rate of full-text search, and the recall of the LLM, given the output of full-text search ($R_{\text{LLM|FTS}}$).

$$\#\text{ of Finds} = N_{\text{images}} \times k_{\text{true}} \times R_{\text{FTS}} \times R_{\text{LLM|FTS}} \tag{5}$$

When not using full-text search, R_{FTS} is equal to 1 and $R_{\text{LLM|FTS}}$ simplifies to R_{LLM}. Putting this together gives us:

$$\text{Cost Per Find} = \frac{P_{\text{image}} + (k_{\text{true}} \times R_{\text{FTS}} + k_{\text{FPR}}) \times \overline{T} \times \overline{P}}{k_{\text{true}} \times R_{\text{FTS}} \times R_{\text{LLM|FTS}}} \tag{6}$$

Each of these parameters can be estimated with small samples of a full dataset. Equation 6 can help estimate unit and total costs for a given project, and help to determine which set of tools might have the greatest return on investment for a given project.

We note that Equation 6 is fairly linear. However, in cases where the task is particularly difficult, terms like $\overline{T}$ can become non-linear, in particular in thought token consumption. As discussed in §4.3, we impose a cap to control this non-linearity.

We also find these formulas useful to help think through what drives cost per find. For example, a higher true underlying positive rate will drive down cost per find.[16] "Efficiency Ratio" can also be considered via equation 6 in combination with recall estimates used for $R_{\text{LLM|FTS}}$ given different LLMs and search parameters.

16. Unless token costs are extraordinarily high such that $\overline{T} \times \overline{P} > 1$. For an expensive model, costs are on the order of $30 per million tokens. Even in this case, for $\overline{T} \times \overline{P}$ to be greater than one, average token usage would need to be ~33k tokens per evaluated item.

5.7 Racial Covenants in Philadelphia

Out of 5,097,555 images, we identified 8,395 that contain a explicit racial restriction. Since a single deed can span multiple images, these correspond to 8,137 unique deeds. Appendix H provides a further accounting of these numbers.

Figure 5 shows the raw number of racially restricted deeds recorded per year. Figure 6 shows the rate of racially restricted deeds per 100 deeds recorded per year.

We identify three periods of elevated covenant recording activity: 1910–1911, 1924–1929, and 1939–1940 (Figure 5). The first peak is especially elevated when considering overall real estate activity as proxied by the number of deeds recorded (Figure 6). The second period of elevated activity corresponds to the Roaring Twenties building boom. The last peak is at a much lower level than the two earlier peaks. See Appendix J for a comparison of covenanted deeds recorded relative to residential units authorized.

Racial covenant recording activity drops off dramatically in the late 1940s when deeds begin regularly using "Under and Subject to restrictions of record" instead of explicitly listing all current restrictions on a property. Crucially, this phrase gives no indication as to the type of restriction—it could be restricting fencing material, the building setback, or the race of the occupant. This means that, while restricted properties in Philadelphia were still transacting in this time period, this information is no longer recoverable from the deed as recorded.

The high incidence of covenants in the earliest years suggests that extending our analysis back to 1880 or earlier, to cover prior building booms, would likely reveal more covenanted deeds.

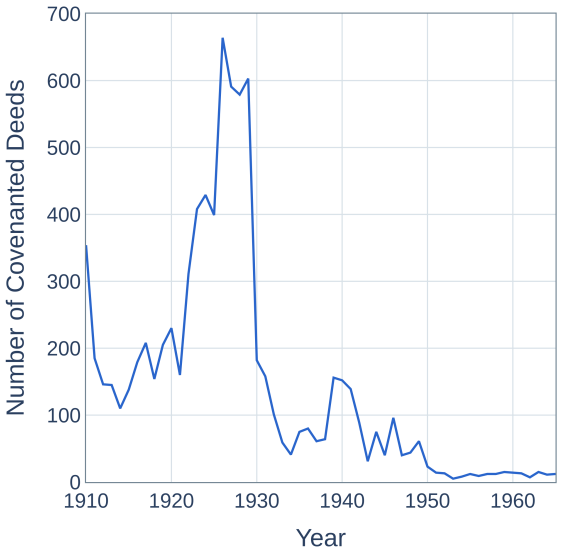


Figure 5: Deeds with racially restrictive covenants, 1910–1965.

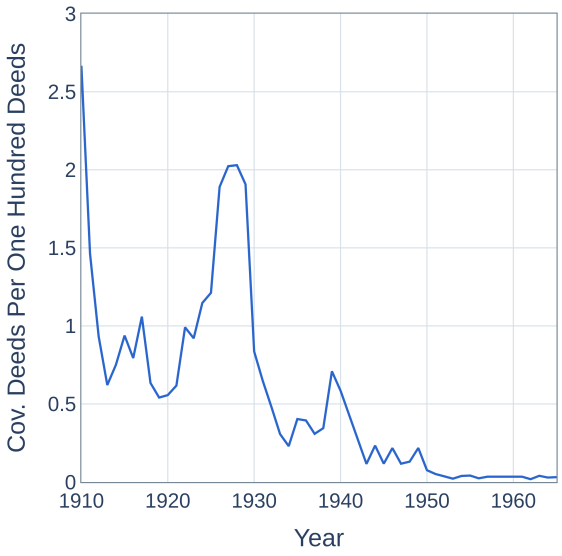


Figure 6: Racially restrictive deeds per one hundred deeds, 1910–1965.

Based on human-transcribed restriction text, the most common language used for racial restriction was "Caucasian," contained in 97.8% of covenanted deeds. Other language included "citizen" (2.8%), "White" (2.2%), "Colored" (0.7%), "African" (0.6%), "Negro" (0.6%), "Jewish" or "Hebrew" (0.3%), and a handful of very rare phrases like

"restricted by race" (6), "Italian" (3), "Polish" (2), and "Mongolian" (1).[17] A restriction may include more than one of these terms, therefore the quoted percentages do not sum to one hundred percent.

Each deed can correspond to one or thousands of parcels. Approximately 70% of covenanted deeds contain only metes and bounds property descriptions rather than addresses. We find that deeds including an address tend to cover fewer parcels than deeds with only metes and bounds descriptions.

Preliminary analysis of 7,446 covenanted deeds shows approximately 38,840 corresponding parcels, about ∼6.6% of all Philadelphia parcels. We believe this estimate is conservative. An even more conservative estimate for these 7,446 covenanted deeds yields a lower bound of 8,832 parcels, while our most permissive estimate yields an upper bound of 64,591 parcels (1.5% to 11%). A precise measure of covenanted parcels and their locations for all 8,137 covenanted deeds will be presented in future work.

6 Conclusion

Using 5.1 million images of Philadelphia property deeds from 1910–1965 for this study, we located 8,395 images which contain a racial restriction. These racially covenanted images correspond to 8,137 deeds, which we estimate to cover $\gtrsim$6.6% of all parcels in Philadelphia. Forthcoming work will leverage LLMs to extract geographic information from these deed images in order to match racially covenanted deeds with modern day parcels.

We present a framework to consider the various costs and trade-offs of different methods for working with a large corpus of images containing unstructured text. Although we emphasize methods that produce OCR output as a separate data product, we explore passing images directly to an LLM for classification. After testing the performance and estimating the cost of more than twenty five unique combinations of OCR and LLMs, our results can be summarized as follows:

1. OCR quality determines the degree of LLM firepower needed. High-quality OCR removes the need for a frontier LLM.

2. Employing both open-source OCR and open-source LLMs results in meaningfully worse performance. However, all open-source can be very cheap and may be a good choice when lower recall is acceptable.

3. Filtering candidate images and narrowing the text context passed to an LLM via full-text search ("snippet") yields recall similar to passing whole image text while greatly reducing token and compute costs. For open-source LLMs, working with snippets improves recall; for frontier LLMs, there is a small net loss in recall.

4. The best image comprehension enabled LLM models can handle classification without OCR as an intermediate step, although this does limit research flexibility. Further, LLMs are not yet clear winners as OCR engines and costs can quickly balloon.

17. These words are not consistently capitalized in deeds.

We prefer an ensemble method incorporating tools, such as OCR as an intermediate step and full-text search, for the additional research value and flexibility it provides. The ensemble method does not always produce the highest recall, but it produces sufficiently high recall while balancing fiscal cost with flexibility not quantified in any of our metrics.

Every project is subject to different priorities and constraints. The functional form of cost per find we present in §5.6 can be used to estimate the cost of different combinations of OCR and LLMs for any dataset. We hope our results, framework, and formula provide practical guidance to characterize and select a research design.

Acknowledgements

We thank Torben S.D. Johansen for thoughtful discussion as well as the conference participants at the 2024 Rasters to Rows, 2025 SSHA, and the 2026 OAH meetings for comments on earlier versions of this work. We thank Alaina Barca, Isabelle Jacowleff, and Mrunal Bhalerao for research support. We also thank Jim Carroll and Walt Rice at the Philadelphia Athenæum, Joshua Blay and Theresa Condon at the Philadelphia City Archives, and James Leonard and Tracy Dandridge at the Philadelphia Department of Records.

Declaration of generative AI use

The evaluation of LLMs is central to the research presented in this paper, and a full accounting of models and usage is given in §4. Additionally, Anthropic Opus 4.8 was used to assist with formatting complex LaTeX tables. AI was not used for original writing, but the authors did use Anthropic Opus 4.8 and Sonnet 5 for copy editing to improve clarity and readability. All AI-assisted content was reviewed and verified by the authors.

ORCID

Daniel Moulton
Robyn N. Smith
Larry Santucci

References

Almagro, Milena, and Aradhya Sood. 2026. "Choices or Constraints? Disentangling Preferences from Restrictions in Housing Markets." May. https://m-almagro.github.io/Covenants_Structural_Historic.pdf.

Asquith, Brian J, Evan Mast, and Davin Reed. 2023. "Local Effects of Large New Apartment Buildings in Low-Income Areas" [in en]. *Review of Economics and Statistics* 105, no. 2 (March): 359–375. https://doi.org/10.1162/rest_a_01055.

Bartik, Alexander, Arpit Gupta, and Daniel Milo. 2025. "The Costs of Housing Regulation: Evidence From Generative Regulatory Measurement" [in en]. *SSRN Electronic Journal* (August). https://dx.doi.org/10.2139/ssrn.4627587.

Bureau of the Census. 1976. *Housing construction statistics, 1889 to 1964* [in en]. Arno Press.

Chagué, Alix. 2024. *McCATMuS - Transcription model for handwritten, printed and typewritten documents from the 16th century to the 21st century,* September. https://doi.org/10.5281/zenodo.13788177. https://doi.org/10.5281/zenodo.13788177.

Datalab. 2025. *Chandra OCR: Layout-Aware Optical Character Recognition Model.* https://github.com/datalab-to/chandra. V. 1.0.

Delegard, Kirsten, Rebecca Gillette, Michael Corey, Penny Petersen, Ryan Mattke, and Amelia Palacios. 2026. *Mapping Prejudice.* https://mappingprejudice.umn.edu/.

Gordon, Colin. 2023. *Patchwork Apartheid.* Russell Sage Foundation, November. ISBN: 978-0-87154-554-1.

Hincken, Garrett. 2021. *How Tangled Titles Affect Philadelphia.* Report. The Pew Charitable Trusts, August. https://www.pew.org/en/research-and-analysis/reports/2021/08/how-tangled-titles-affect-philadelphia.

Hughes, Charles, Jr. 1958. "The Philadelphia Program" [in en]. *American Archivist* 21, no. 2 (April): 131–142.

Kiessling, Benjamin. 2025. *The Kraken OCR system.* V. 6.0, August. https://kraken.re.

Korngold, Gerald. 2001. "The Emergence of Private Land Use Controls in Large-Scale Subdivisions: The Companion Story to *Village of Euclid v. Ambler Realty Co.*" *Case Western Reserve Law Review* 51 (4): 617–643. https://scholarlycommons.law.case.edu/caselrev/vol51/iss4/5.

Li, Minghao, Tengchao Lv, Jingye Chen, Lei Cui, Yijuan Lu, Dinei Florencio, Cha Zhang, Zhoujun Li, and Furu Wei. 2022. *TrOCR: Transformer-based Optical Character Recognition with Pre-trained Models.* arXiv: 2109.10282 [cs.CL]. https://arxiv.org/abs/2109.10282.

Mindee. 2021. *docTR: Document Text Recognition.* https://github.com/mindee/doctr.

Nelson, David Ragnar. 2026. *The RevCity Model: HTR Recognition Model trained on 18th century English handwriting,* March. https://doi.org/10.5281/zenodo.19238205. https://doi.org/10.5281/zenodo.19238205.

OpenAI. 2025. *Context Arena.* Accessed on July 8, 2025. https://contextarena.ai/.

Plotkin, Wendy. 2001. ""Hemmed in": The struggle against racial restrictive covenants and deed restrictions in Post-WWII Chicago." *Journal of the Illinois State Historical Society* 94, no. 1 (Spring).

Poznanski, Jake, Luca Soldaini, and Kyle Lo. 2025. *olmOCR 2: Unit Test Rewards for Document OCR.* arXiv: 2510.19817 [cs.CV]. https://arxiv.org/abs/2510.19817.

Santucci, Larry. 2020. "Documenting Racially Restrictive Covenants in 20th Century Philadelphia" [in en]. *Cityscape: A Journal of Policy Development and Research* 22, no. 3 (November): 241–267. https://www.huduser.gov/portal/periodicals/cityscpe/vol22num3/ch11.pdf.

Shertzer, Allison, Tate Twinam, and Randall P. Walsh. 2016. "Race, Ethnicity, and Discriminatory Zoning." *American Economic Journal: Applied Economics* 8, no. 3 (July): 217–46. https://doi.org/10.1257/app.20140430. https://www.aeaweb.org/articles?id=10.1257/app.20140430.

Stein, Sarah, and Ann Carpenter. 2022. *Heir's Property in an Urban Context.* 2022 AEA/ASSA Annual Meeting VIRTUAL, January 7-9, 2022. Agricultural and Applied Economics Association, January. https://doi.org/10.22004/ag.econ.316788. https://doi.org/10.22004/ag.econ.316788.

Storey, Emanuel A. 2018. "Cartographically Reconstructing Surveys of Community Land Grants in New Mexico to Support Historical Research and Political Discourse" [in en]. *Historical Methods* 52, no. 2 (April): 95–109.

Surani, Faiz, Mirac Suzgun, Vyoma Raman, Christopher D. Manning, Peter Henderson, and Daniel E. Ho. 2024. "AI for Scaling Legal Reform: Mapping and Redacting Racial Covenants in Santa Clara County." *Stanford RegLab Working Paper,* https://dho.stanford.edu/wp-content/uploads/Covenants.pdf.

Suzgun, Mirac, Stuart M Shieber, and Dan Jurafsky. 2023. "string2string: A Modern Python Library for String-to-String Algorithms." *arXiv preprint arXiv:2304.14395.*

The Athenaeum of Philadelphia. *The Philadelphia Real Estate Record and Builders' Guide.* https://www.philageohistory.org/BuildersGuide/. Accessed 2025-08-16.

Wattenberg, Ben. 1976. *The Statistical History of the United States.* 639. Series N 156–169, "New Housing Units Started, 1889 to 1970". New York, NY: Basic Books, Inc.

Weiss, Marc A. 1987. *The Rise of the Community Builders.* Frederick, MD: Beard Books, January.

Winling, LaDale. 2026. *National Covenants Research Coalition.* https://www.nationalcovenantsresearchcoalition.com/whoweare.

A Glossary

Character error rate is calculated as the number of edit operations between a source string and a reference or "correct" string. This is given by the formula

$$\text{CER} = \frac{S + D + I}{N} \tag{7}$$

where S is substitutions, D is deletions, I is insertions, and N is the length of the target string. For detailed definitions of these operations, see **Damerau–Levenshtein**. In essence, this is a measure of the percentage of the source string that is "wrong" relative to the reference string.

Context window is the maximum amount of text (**tokens**) an LLM can process at one time, including both input and output tokens.

Damerau–Levenshtein edit distance is a measure of the number of steps to go from string a to string b. These steps are broken into replacements, insertions, deletions, and adjacent transpositions. Simple "edit distance" or "Levenshtein distance" does not permit adjacent transpositions. For example, the following are all Damerau-Levenshtein edit distance of one; replacement: bat→bot, insertion: cat→cart, deletion: bread→read, adjacent transposition: form→from. The latter example would be distance two in simple Levenshtein distance terms.

F1 is the harmonic mean of **precision** and **recall**. Relative to a straight average, very low recall or precision results in a lower F1 score.

$$\text{F1} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{8}$$

`frequency_penalty` is an LLM sampling parameter that reduces the likelihood of a **token** being generated again based on how many times it has already appeared in the output so far. It applies a penalty that scales with token frequency—the more often a token has occurred, the larger the penalty applied to its logit before sampling. Takes various ranges depending on the particular model or API being used. Typically it can be set to discourage (positive value) or encourage (negative value) token repetition. Similar to `repetition_penalty`, but `frequency_penalty` *escalates* the more often a token appears.

Intestacy occurs when an owner of property dies without having written a will. Property transfer under intestacy is a major cause of **tangled titles**.

Precision is the ratio of items correctly labeled positive by a model over the total number of items labeled positive by the model:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{9}$$

or the share of positive labels correctly given. See **recall** for explanation of why this measure is useful.

Recall is the ratio of items correctly labeled positive by a model over the number of positives in the underlying dataset:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{10}$$

or the percent of relevant labels found. Also called "true positive rate." Along with **precision**, this measure retains its meaning whether the positive class is very large or very small as a share of the total population, unlike simple accuracy. Simple accuracy is (correct predictions) / (total predictions). Imagine a dataset with ninety 1s and ten 0s—a model that always predicted 1 would have a simple accuracy of 90%, despite the fact that this is obviously a bad model for the purpose of identifying the 0s. In this case, the model would have a precision and recall of 0, correctly capturing the "always 1" model's poor performance.

`repetition_penalty` is an LLM sampling parameter that reduces the likelihood of a **token** being repeated more than once, considering both the prompt and the output. It is applied multiplicatively. Similar to `frequency_penalty`, but `repetition_penalty` is a flat penalty by token and does not scale with the cumulative occurrence of said token.

Tangled titles, often associated with "heirs' property," describe properties where the name of the owner on the deed is not the same as that of the apparent owner. Tangled titles are a major issue that prevents households from benefiting from property ownership.

Temperature is an LLM sampling parameter that controls how deterministic its responses are. The lower the temperature, the more consistent the LLM's responses will be when asked the exact same question multiple times. The lower bound on temperature is 0. While temperature does not theoretically have an upper bound, most models accept a maximum value of 1 or 2. Roughly speaking, temperature works by scaling the model's output logits before they are converted into probabilities: lower temperatures sharpen the probability distribution (concentrating it on the most likely tokens), while higher temperatures flatten it (making less likely tokens more probable). At temperature 0, sampling reduces to always choosing the most likely next token. That said, even at 0, a model will not always produce identical results—sometimes because the model doesn't accept a true zero temperature (instead substituting a very small non-zero value), but more often due to lower-level nondeterminism such as floating-point rounding effects in parallelized GPU computations.

Token is the basic unit of text that an LLM processes. Depending on the model and its tokenizer, a token may be a whole word, a sub-word (a fragment of a word), or a single character. As a rough rule of thumb, one token corresponds to about 4 characters of English text (including spaces). However, this varies considerably and depends on the specific model and encoding method used. Rare words and numbers tend to be split into more tokens, common words into fewer.

B LLM Prompts

As described in §4.6, we used the same basic starting prompt for all of our LLM labeling tasks. Figure 7 illustrates this prompt.

For gpt-oss-20b and -120b, we found it necessary increase the specificity of our instructions. We provided detail explicitly attempting to avoid false positives from *anti-discriminatory* passages and from business restrictions containing potentially racially coded tokens. On the other hand, gpt-oss did not like to assume that "Caucasian" was a strong sign of a racial covenant. Specifically, the following was added to the prompt for the gpt-oss models: "Try not to mistake restrictions on businesses like blacksmiths or piggeries as a racial covenant. Do not label anti-discriminatory clauses as racially restrictive. In general, if you see something that looks like a restriction on race or a restriction on a word that looks vaguely like "Caucasian" you should call this a racially restrictive covenant."

For Gemini 2.5 Flash applied directly to an image, we used the same prompt as in Figure 7 but modified references to deed text to references to the input image: "This source *image* can be low quality. It is important that you read documents very carefully and in their entirety. You must very carefully read and consider the document, taking in to account the full context of the text in the *image*. Only consider information contained in the *image*." We found it necessary to emphasize this last point. Gemini would often make an inference that a deed from a certain time period was likely to have a covenant—by considering outside "knowledge" embedded in the LLM.

Prompt:

System:
You are a lawyer very carefully reading deeds to find covenants restricting use of property for business purposes. It is important that you read documents very carefully and in their entirety. The text you are reading will be OCR'd text and will have some mistakes, you should try to correct these. You must very carefully read and consider the document, taking in to account the full context of the text I present. First consider how you would respond in a more lengthy discussion where you explain things, then summarize this into just a yes or no answer. You response will be in the form of the following json, where you use proper escaping for quotes within strings using backslashes:

```
{
      "covenant_exists":    boolean
    , "comments": string
}
```

Absolutely do not return anything but json. If you have more to say put it in "comments". Do not use any quotation marks of any form inside of the body of "comments". Only use them as appropriate to denote a string in a json.

Instructions:
Does this passage from a deed contain a racially restrictive covenant?

SNIPPET OR WHOLE IMAGE TEXT

Figure 7: The system prompt provided to the LLM instructing it how to interact with the data and how to format its response. The "comments" section provides a space for the LLM to explain its reasoning.

C Estimating True Total of Covenanted Images in Corpus

Tables 11, 12, and 13 depend on estimates of two key values. First, the number of covenanted images across all 4,682,704 deed images, not just those in our core sample of 279,131 images (see §4.2). We denote this value as Cov_{tot}. Second, the recall of each method (combination of OCR, LLM, and LLM input format) when considering all 4,682,704 images, which we denote as $R_{tot,m}$. We denote handwritten, typewritten, and photostat as h, t, and p.

In order to estimate Cov_{tot}, we start with known performance statistics for Textract 2025 and Textract 2021. We must consider Textract 21 results because, for cost reasons, it is the only OCR method we applied to all images in the corpus. We also consider Textract 25 because we ran most of the handwritten and typewritten images through this method. Using the performance of Textract 21, Textract 25, and the rate of covenants in the random portion of the core sample, we produce an estimate of Cov_{tot}. Then, using the comparative performance of Textract 25, Textract 21, and a given method (m)—a combination of OCR, full-text search, and LLM—we produce an estimate for each $R_{tot,m}$.

For example, for handwritten images, we reran all images through Textract 25 and evaluated the resulting text for covenants via snippets and whole image text passed to Sonnet 4.5. We then scaled up what we found by the appropriate recall rates measured in the core sample. For photostats, where we did not apply Textract 25 to any image outside of the core sample, we assume covenants occur outside the core sample at the same rate as in the randomly selected component of the core sample; see §4.2 and Appendix D. The typewritten total is estimated using a combination of these methods.

In detail, we start with the decomposition:

$$Cov_{tot} = TT_{25\&21,\text{FTS}} + TT_{25,\text{FTS}} + TT_{25,\text{WT}} \tag{11}$$

where $TT_{25\&21,\text{FTS}}$ are covenanted images that match via full-text search on Textract 25 *and* Textract 21 output, $TT_{25,\text{FTS}}$ are covenanted images that match via full-text search on Textract 25 output but that *do not* match via full-text search on Textract 21 output, and $TT_{25,\text{WT}}$ are covenanted images that do not match via full-text serach on Textract 25 output.

Because the core sample includes any image in our corpus of images with a full-text search match to Textract 21 output, there are no images outside of the sample that match via full-text search on Textract 21 output (see §4.2). $TT_{25\&21,\text{FTS}}$ is therefore a known quantity:

$$TT_{25\&21,\text{FTS}} = \text{Core Sample, Covenanted Images, FTS Match } TT_{25} \text{ and } TT_{21} \tag{12}$$

$TT_{25,\text{FTS}}$ can be decomposed into a known component (in the core sample) and one which we must estimate (outside the core sample):

$$\begin{aligned} TT_{25,\text{FTS}} = \quad & TT_{25,\text{FTS}\in\text{Core}} + TT_{25,\text{FTS}\notin\text{Core}} \\ TT_{25,\text{FTS}} = \quad & (\text{Core Sample, Covenanted Images, FTS Match } TT_{25} \text{ and not } TT_{21}) \\ & + (\text{Covenanted Images Outside Core Sample, FTS Match } TT_{25}) \end{aligned} \tag{13}$$

The same decomposition into a known and unknown component holds for TT_{WT}:

$$\begin{aligned} TT_{25,\text{WT}} = \quad & TT_{25,\text{WT}\in\text{Core}} + TT_{25,\text{WT}\notin\text{Core}} \\ TT_{25,\text{WT}} = \quad & (\text{Core Sample, Covenanted Images, No FTS Match}) \\ & + (\text{Covenanted Images Outside Core Sample, No FTS Match } TT_{25}) \end{aligned} \tag{14}$$

Therefore, we must produce estimates for $TT_{25,\text{FTS}\notin\text{Core}}$ and $TT_{25,\text{WT}\notin\text{Core}}$.

For handwritten (h) documents, we ran all images in the corpus through Textract 25, while for typewritten (t) documents, we ran nearly all the images through Textract 25. Ignoring a small adjustment for these remaining un-reOCRed typewritten images,[18] for handwritten and typewritten documents we can estimate Covenanted Images Outside Core Sample, FTS Match to Textract 25 as:

$$TT_{25,\text{FTS}\in(h\vee t)\notin\text{Core}} = \frac{TP_{TT_{25,\text{FTS}}\in(h\vee t)\notin TT_{21,\text{FTS}}\notin\text{Core,Sonnet,Snippet}}}{R_{TT25,\text{FTS}\in(h\vee t)\notin TT_{21,\text{FTS}}\in\text{Core,Sonnet,Snippet}}} \tag{15}$$

That is to say, we estimate the second half of Equation 13 by taking true positives generated by Sonnet 4.5 on snippet text for images matching via full-text search on Textract 25 output and scaling them up by the recall, in the core sample, of Sonnet 4.5 on snippet text for images matching via full-text search on Textract 25 output but not on Textract 21 output. The key assumption is that the recall of Sonnet 4.5 on Textract 25 snippets is the same outside the core sample as inside it, conditional on an image matching via full-text search on Textract 25 but not Textract 21 output.

For handwritten deeds we *also* passed all the new Textract 25 text through as whole image text. Therefore:

$$TT_{25,\text{WT}\in h\notin\text{Core}} = \frac{TP_{TT_{25}\in h\notin TT_{25,\text{FTS}}\notin TT_{21,\text{FTS}}\notin\text{Core,Sonnet,Whole Text}}}{R_{TT25\in h\notin TT_{25,\text{FTS}}\notin TT_{21,\text{FTS}}\in\text{Core,Sonnet,Whole Text}}} \tag{16}$$

This follows the same logic of Equation 15. We take true positives generated by Sonnet 4.5 on whole image text for images outside the core sample that do not match via full-text search on Textract 25 or Textract 21 output, and scale them up by the recall, in the core sample, of Sonnet 4.5 on whole image text for images that do not match via full-text search on Textract 25 or Textract 21 output. The key assumption is that the recall of Sonnet 4.5 on Textract 25 whole image text is the same outside the core sample as inside it, conditional on an image not matching via full-text search on Textract 25 or Textract 21 output.

Due to cost considerations and the high recall we found using snippets alone, we did not rerun typewritten documents as whole text using Textract 25 output and Sonnet 4.5. Therefore we need a different way to measure $TT_{25,\text{WT}\in t\notin\text{Core}}$. We form the following estimate:

$$TT_{25,\text{WT}\in t\notin\text{Core}} = \text{Images}_{\in t\notin\text{Core}\notin Images_{TT_{25,\text{FTS}}}} \times \frac{Cov_{\in t\in\text{Core}\in\text{Random}}}{Images_{\in t\in\text{Core}\in\text{Random}}} \tag{17}$$

18. We did not reOCR typewritten images for the years 1931 through 1934, given the overall low rate of covenanted deed per deed recorded in this period. For Equation 15, we estimated the number of missing Sonnet 4.5, Textract 25 snippet finds by taking (old 21 snippet TP, 1931-1934)×(new 25 snippet TP, 1913-1930)/(old 21 snippet TP, 1913-1930). This assumes new Textract 25 finds are a function of the number of old Textract 21 finds.

This is the number of typewritten images, not in our core sample, that do not match via full-text search on Textract 25 output, scaled by the rate at which we found covenanted images in typewritten images that were selected *randomly* for inclusion in the core sample.

We must take a different approach for photostat deeds. Given cost considerations and the very low rate per image of covenants in this period, we did not OCR any of the images outside the core sample using Textract 25. Instead, we first estimate the number of new covenants that would be found assuming we did run Textract 25 on all photostat images:

$$\begin{aligned} TT_{25 \in p \notin \text{Core}} &= TT_{25,\text{WT} \in p \notin \text{Core}} + TT_{25,\text{FTS} \in p \notin \text{Core}} \\ &= Images_{\in p \notin \text{Core}} \times \frac{Cov_{\in p \in \text{Core} \in \text{Random}}}{Images_{\in p \in \text{Core} \in \text{Random}}} \end{aligned} \tag{18}$$

Analogously to Equation 17, we estimate the number of photostat true positives to be found outside of the core sample by assuming they occur at the same rate as in the *randomly* selected component of our core sample. We can than estimate which of these incremental new finds would match a full-text search query:

$$TT_{25,\text{FTS} \in p \notin \text{Core}} = TT_{25 \in p \notin \text{Core}} \times \frac{Cov_{\in p \in \text{Core} \in TT_{25,\text{FTS}} \notin TT_{21,\text{FTS}}}}{Cov_{\in p \in \text{Core} \notin TT_{21,\text{FTS}}}} \tag{19}$$

That is, we estimate how many covenanted images outside our core sample would match via full-text search on Textract 25 output by allocating our estimate of all covenanted images outside the core sample via the within-core-sample ratio of covenants full-text search matching via Textract 25 but not Textract 21 output to those not matching via Textract 21 output.

$TT_{25,\text{WT} \in p \notin \text{Core}}$ is then calculated as:

$$TT_{25,\text{WT} \in p \notin \text{Core}} = TT_{25 \in p \notin \text{Core}} - TT_{25,\text{FTS} \in p \notin \text{Core}} \tag{20}$$

One could see an issue with the estimate of $TT_{25,\text{WT}}$ for photostats vs. typewritten documents—perhaps this difference underestimates how many photostats would be found on whole image text using Textract 25? That said, given that Textract 21 actually performs *better* on photostat recall than Textract 25 (see Appendix H and I), we do not think this is a big concern.

Armed with each of our estimates for Cov_{tot} by image type, we can now estimate $R_{tot,m}$ based on TP_m/Cov_{tot} i.e. an estimate of the number of covenants identified by each choice of LLM and text/image input type.

$$\begin{aligned} TP_m = &+ TT_{25\&21,\text{FTS}} \times R_{m|\in \text{Core} \in TT_{25,\text{FTS}} \in TT_{21,\text{FTS}}} \\ &+ TT_{25,\text{FTS}} \times R_{m|\in \text{Core} \in TT_{25,\text{FTS}} \notin TT_{21,\text{FTS}}} \\ &\quad TT_{25,\text{WT}} \times R_{m|\in \text{Core} \notin TT_{25,\text{FTS}} \notin TT_{21,\text{FTS}}} \end{aligned} \tag{21}$$

Or in words, we scale the terms in Equation 11 by the corresponding conditional recall measured in the core sample. The first part of this sum uses recall in the core sample for the method (m) on images that matched via full-text search on both Textract 25 and 21 output. The second part uses recall in the core sample for the method on the

subset of images in the core sample that matched using Textract 25 output not via Textract 21. The third part uses recall in the core sample for the method on the subset of images in the core sample that did not match any full-text search using Textract 21 or 25 output.

R_m is then TP_m/Cov_{tot}, the ratio of Equations 21 and 11.

D Sample Selection

In §4.2, we describe how our core sample of 279,131 images is composed of four components: (1) images matching any of our search terms (see Appendix E) in text output by Textract in November 2021; (2) any additional covenanted image identified as of December 2025 (Santucci 2020); (3) any image in a deed associated with a covenanted image known as of December 2025; and (4) a random sample from all remaining images.

In §4.2, we say of the random component: "103,834 images in the subset are a 4.3% sample of all remaining deed images recorded between 1910–1950 and a 0.11% sample of all deeds recorded from 1951–1965." While accurate, this is a simplified description.

More precisely, these 103,834 images arise from two distinct selection processes, each of which yields results equivalent to a simple random draw.

The first process is a 4% sample of all remaining images (after excluding the first three components described in the first paragraph) recorded between 1910 and 1955 that fuzzy match the word "under" and the word "subject" in close proximity (slop of two). The majority of restrictive clauses—racial or not—start with the phrase "under and subject to..." This sampling method was designed to select for images more likely to contain a racial covenant than for those truly drawn at random.

The second selection process is a 4.3% random sample of all remaining images—excluding the first three components described in the first paragraph and anything fuzzy matching "under" and "subject"—from 1910–1950.

In both selection processes, the later years—1956–1965 and 1951–1965—were excluded due to the large number of images and extremely low rate of covenants per image recorded from 1951 forward (see §5.7). The ex-post rate at which "under" and "subject" images contained covenants was nearly identical to the ex-post rate in the "fully random" subset. Given this, and for concision of explanation, we combined these two components into one description in §4.2.

The random component of our core sample actually contains no deeds from 1960–1965. Where we use statistics from 1960–1965 to estimate the full count of covenanted photostat images (see §5.4 and Appendix C), our cost analysis assumes covenants occur at the same rate as from 1934–1959. Since photostats from 1960–1965 actually contain covenants at a lower rate (see Figure 6), this assumption *inflates* the total estimated number of covenanted photostat images. This bias works against the takeaways shown in §5.4 Table 12 because it increases the frequency of the event. Thus it is appropriate to describe this part of the sample as "0.11% sample of all deeds recorded from 1951–1965" without impacting the meaning of any of our results.

E Philadelphia: Search Terms for Racial Covenants

Here we show statistics on the search terms used for our sample construction (see §4.5) and snippet statistics (see §4.6). All three tables in this Appendix use the 279,131 images in our core sample with OCR text generated by Textract in December 2025. Search terms are only listed in tables *if* they returned at least one covenanted image. There are many terms we searched that did not return a covenanted image; those are not listed here.

Table 14 shows the number of images matched by each of our fuzzy search terms. This table is not deduplicated. For example, any image matching "Caucasian race" would also be matched by "Caucasian." The table is ordered by the number of covenanted images matched by the given search term. Terms matching no covenanted images are excluded from the table. Table 14 also reports performance of Claude Sonnet 4.5 on text snippets associated with the given search term. "Pct with Covenant" is the percentage of matched images that contain a covenant. In other words, this column is full-text search precision (P_{FTS}), the complement to R_{FTS} in Equation 4 in §5.6.

Table 15 is deduplicated. Per-term match counts sum to the total number of distinct images matched across all terms. The table was built iteratively using greedy assignment. At each step, we select the term with the highest share of true positives among unassigned matched images. Put another way, we select the term with the highest remaining term-level precision (P_{FTS}). We then assign matching images to this term and remove these images from the pool. We repeat until no matching images remain. Terms matching no covenanted images are excluded from the table.

Tables 14 and 15 contain terms with an "x" inserted where a space should appear. This lets full-text search match text in which a space between words was replaced by another character, whether in the original document or by OCR.

Table 16 presents "fuzzy fuzzy" search terms as described in §4.5. This table is deduplicated as in Table 15 and only shows *incremental* matches relative to Tables 14 and 15.

In Philadelphia, certain language, like "Caucasian", is almost always associated *only* with a racial covenant, while other phrases, such as "not be sold," are only occasionally associated with such a covenant. That said, low full-text search precision terms, such as "not be sold" or "occupied person," do identify a substantial number of new covenants not matched by high precision search terms as seen in Table 15.

Documents fuzzy matching "Caucasian" do lead to false positives (FP_{FTS})—either in the form of proper names within edit distance two or via a tiny number (4) of cases where a deed contained "Caucasian" but not in reference to a covenant on the property. The four false-positive uses of "Caucasian" described 1) the race of a buyer, itself a sign of a biased transaction, 2) a piece of furniture, 3) a neighborhood, and 4) a racially restricted scholarship at Temple University.

While covenants explicitly excluding occupancy by Asian people exist in other cities, we found zero true positives in Philadelphia when searching for associated racial terms.[19] For this reason, Tables 14, 15, and 16 contain no such language, even though we did search for terms reported in other cities. Matches to "asian race" shown in the tables are all cases where "Caucasian" was split across lines as in "Cauc- asian."

19. We did, once working with human transcribed text, find one covenant containing the word "Mongolian."

Table 14: Search terms and performance on core sample, OCRed with Textract. Not deduplicated—images can match multiple rows in table. Only terms returning at least one covenanted image are shown. Claude performance is Sonnet 4.5 on snippets.

Search Term	Matches	Contains Covenant	Claude TP	FP	TN	FN	Pct with Cov. (P_{FTS})	Claude Precision	Recall
caucasian	6,147	6,137	6,130	0	10	7	99.84	100.00	99.89
caucasian race	4,766	4,763	4,761	0	3	2	99.94	100.00	99.96
shall not be	75,315	4,362	4,272	6	70,947	90	5.79	99.86	97.94
occupied by any	6,184	4,270	4,220	0	1,914	50	69.05	100.00	98.83
occupied person	3,952	3,730	3,688	0	222	42	94.38	100.00	98.87
shall be sold	14,150	2,485	2,449	2	11,663	36	17.56	99.92	98.55
not be sold	6,237	2,472	2,435	2	3,763	37	39.63	99.92	98.50
shall not sold	8,531	1,943	1,910	3	6,585	33	22.78	99.84	98.30
shall not be sold	7,488	1,841	1,812	2	5,645	29	24.59	99.89	98.42
other races	3,124	1,679	1,679	1	1,444	0	53.75	99.94	100.00
other than those	2,799	1,514	1,503	0	1,285	11	54.09	100.00	99.27
caucasian descent	1,016	1,016	1,013	0	0	3	100.00	100.00	99.70
citizen united	333	206	204	0	127	2	61.86	100.00	99.03
casian race	299	180	179	0	119	1	60.20	100.00	99.44
white race	300	131	130	0	169	1	43.67	100.00	99.24
asian race	130	124	123	0	6	1	95.38	100.00	99.19
caucasi	110	110	110	0	0	0	100.00	100.00	100.00
white person	438	90	90	2	346	0	20.55	97.83	100.00
cauc race	98	87	87	0	11	0	88.78	100.00	100.00
cauces race	72	66	66	0	6	0	91.67	100.00	100.00
race occupy	80	56	56	0	24	0	70.00	100.00	100.00
ian race	51	50	48	0	1	2	98.04	100.00	96.00
casian descent	52	46	45	0	6	1	88.46	100.00	97.83
colored person	1,499	43	42	0	1,456	1	2.87	100.00	97.67
negro negroes	323	38	38	1	284	0	11.76	97.44	100.00
except servants	47	36	36	0	11	0	76.60	100.00	100.00
negro person	44	35	35	1	8	0	79.55	97.22	100.00
african descent	45	35	35	1	9	0	77.78	97.22	100.00
negro descent	27	24	24	0	3	0	88.89	100.00	100.00
domestic servent	48	24	23	0	24	1	50.00	100.00	95.83
domestic servant	24	22	22	0	2	0	91.67	100.00	100.00
asian descent	22	22	22	0	0	0	100.00	100.00	100.00
cauc descent	23	20	20	0	3	0	86.96	100.00	100.00
colored race	1,257	19	19	1	1,237	0	1.51	95.00	100.00
jewish race	273	19	19	0	254	0	6.96	100.00	100.00
caucasianrace	17	17	17	0	0	0	100.00	100.00	100.00
cauces descent	18	17	17	0	1	0	94.44	100.00	100.00
color race	1,210	16	16	1	1,193	0	1.32	94.12	100.00
different race	174	15	15	0	159	0	8.62	100.00	100.00
caucasianxrace	13	13	13	0	0	0	100.00	100.00	100.00
negro race	20	11	11	0	9	0	55.00	100.00	100.00
african race	104	5	4	0	99	1	4.81	100.00	80.00
ian descent	5	4	4	0	1	0	80.00	100.00	100.00
color person	882	3	3	0	879	0	0.34	100.00	100.00
racial restrictions	312	2	1	0	310	1	0.64	100.00	50.00
casianrace	2	2	2	0	0	0	100.00	100.00	100.00
caucasianxdescent	2	2	2	0	0	0	100.00	100.00	100.00
domestic service	9	2	2	0	7	0	22.22	100.00	100.00
caucasiandescent	2	2	2	0	0	0	100.00	100.00	100.00
notxbexsold	1	1	1	0	0	0	100.00	100.00	100.00
asianrace	6	1	1	0	5	0	16.67	100.00	100.00
asiatic	2	1	1	0	1	0	50.00	100.00	100.00
ien race	5	1	1	0	4	0	20.00	100.00	100.00
jewish descent	92	1	1	0	91	0	1.09	100.00	100.00

Table 15: Deduplicated search terms and performance on core sample, OCRed with Textract. Matches duplicated recursively, retaining search terms with highest remaining percent share with covenant until all images matching search terms exhausted. Only terms returning at least one covenanted image are shown. Claude performance is Sonnet 4.5 on snippets.

Search Term	Matches	Contains Covenant	Claude				Pct with Cov. (P_{FTS})	Claude	
			TP	FP	TN	FN		Precision	Recall
caucasian race	4,725	4,722	4,720	0	3	2	99.94	100.00	99.96
caucasian descent	1,016	1,016	1,013	0	0	3	100.00	100.00	99.70
occupied person	711	490	450	0	221	40	68.92	100.00	91.84
caucasian	370	363	361	0	7	2	98.11	100.00	99.45
not be sold	3,852	146	124	1	3,705	22	3.79	99.20	84.93
occupied by any	1,994	131	115	0	1,863	16	6.57	100.00	87.79
asian race	126	120	119	0	6	1	95.24	100.00	99.17
caucasi	105	105	105	0	0	0	100.00	100.00	100.00
shall not be	63,105	85	37	4	63,016	48	0.13	90.24	43.53
other than those	1,322	63	55	0	1,259	8	4.77	100.00	87.30
other races	1,468	46	46	1	1,421	0	3.13	97.87	100.00
shall be sold	5,255	24	19	0	5,231	5	0.46	100.00	79.17
negro descent	26	23	23	0	3	0	88.46	100.00	100.00
ian race	23	22	20	0	1	2	95.65	100.00	90.91
asian descent	21	21	21	0	0	0	100.00	100.00	100.00
race occupy	42	18	18	0	24	0	42.86	100.00	100.00
caucasianrace	17	17	17	0	0	0	100.00	100.00	100.00
shall not sold	1,024	15	12	1	1,008	3	1.46	92.31	80.00
white race	179	11	10	0	168	1	6.15	100.00	90.91
african descent	20	10	10	1	9	0	50.00	90.91	100.00
white person	342	9	9	2	331	0	2.63	81.82	100.00
citizen united	135	8	7	0	127	1	5.93	100.00	87.50
cauc race	16	6	6	0	10	0	37.50	100.00	100.00
casian descent	12	6	5	0	6	1	50.00	100.00	83.33
negro person	15	6	6	1	8	0	40.00	85.71	100.00
casian race	118	5	5	0	113	0	4.24	100.00	100.00
cauces race	10	4	4	0	6	0	40.00	100.00	100.00
colored person	417	4	3	0	413	1	0.96	100.00	75.00
cauces descent	5	4	4	0	1	0	80.00	100.00	100.00
negro race	12	3	3	0	9	0	25.00	100.00	100.00
domestic servant	5	3	3	0	2	0	60.00	100.00	100.00
shall not be sold	4,430	2	2	0	4,428	0	0.05	100.00	100.00
caucasianxdescent	2	2	2	0	0	0	100.00	100.00	100.00
domestic servent	24	2	1	0	22	1	8.33	100.00	50.00
except servants	13	2	2	0	11	0	15.38	100.00	100.00
racial restrictions	45	2	1	0	43	1	4.44	100.00	50.00
cauc descent	5	2	2	0	3	0	40.00	100.00	100.00
color person	14	1	1	0	13	0	7.14	100.00	100.00
casianrace	1	1	1	0	0	0	100.00	100.00	100.00
african race	93	1	1	0	92	0	1.08	100.00	100.00
colored race	135	1	1	1	133	0	0.74	50.00	100.00
notxbexsold	1	1	1	0	0	0	100.00	100.00	100.00

Table 16: "Fuzzy Fuzzy" search, deduplicated - using phrases found from other searches. Search performed on core sample, OCRed with Textract, resulting snippets evaluated with Sonnet 4.5. Matches duplicated recursively, retaining search terms with highest remaining percent share with covenant until all images matching search terms exhausted. Terms matching no covenanted images are excluded.

Search Term	Matches	Contains Covenant	Claude TP	FP	TN	FN	Pct with Cov. (P_{FTS})	Claude Precision	Recall
w/ 2 unique TP each (9)	926	18	13	1	907	5	1.94	92.86	72.22
w/ 1 unique TP each (16)	2,003	16	11	0	1,987	5	0.80	100.00	68.75
comeasian	61	9	8	0	52	1	14.75	100.00	88.89
eauessian	7	4	4	0	3	0	57.14	100.00	100.00
convasion	433	3	0	0	430	3	0.69	0.00	0.00
esmeasian	4	3	3	0	1	0	75.00	100.00	100.00
compasioni	114	3	2	1	110	1	2.63	66.67	66.67
thecaucasian	3	3	3	0	0	0	100.00	100.00	100.00
concesion	433	3	1	0	430	2	0.69	100.00	33.33

F Practical OCR Model Takeaways

Here we give more detail of our experience working with the various OCR methods outlined in this paper.

Typically, OCR involves two steps. First, segmentation—breaking the image into regions, lines, words, and/or characters. Second, identification or recognition—assigning characters to individual segments. The three ML based tools—Kraken, TrOCR and Amazon Textract—we tested for handwriting recognition treat these steps in very different ways. Kraken includes separate tools for segmentation and recognition that must be run separately. TrOCR only performs recognition and is meant to read a "line" of text. Textract does both in one combined step—at least for the purpose of how the user interacts with the API.

Kraken has separate segmentation and recognition models. It includes a neural-network-based segmentation model (`blla`) and a rules-based legacy segmenter. For handwritten text, we used the `blla` model, as it is designed to handles the curves, loops, and irregularity of handwritten text. We tested two different recognition models: 1) McCATMUS (Chagué 2024), a model for handwritten and typed text trained on text in seven languages from the 16th to 21st centuries, and 2) RevCity (Nelson 2026), a handwriting model trained on 18th century English text.

As discussed in 4.3, TrOCR takes as input an image of a single line or word of text. It does not provide any segmentation tools, so we used Kraken's `blla` segmentation to produce the input for TrOCR. The documentation makes clear that the model treats all image inputs as 384 by 384 pixels, resizing any input to this size. This results in highly distorted (horizontally squished) text for most single lines of text, since such a line is normally much wider than it is tall. What is unclear from its documentation is whether TrOCR would perform better if the input image were processed to fit the 384 by 384 size without squishing—for example, by working on single words or very short phrases, or by shrinking the line down and padding it with white space to preserve the aspect ratio. We tested these pre-processing methods, but the model performed best without them. We therefore infer that the model was trained on heavily distorted images.

docTR is a wrapper around a set of various ML models broken into separate steps for segmentation and recognition. Metrics reported in docTR's documentation suggest mostly similar OCR performance across different choices of models. However, we found that only the default choices of `db_resnet50` for segmentation and `crnn_vgg16_bn` for recognition where viable from a run time perspective. None of the default ML segmentation or recognition models are designed to work with handwritten text; therefore, we only used docTR on typewritten and photostat images.

F.1 LLM Based OCR

There are several free-to-use LLM models that have been custom-built or fine-tuned specifically for OCR work. We tested two of the top models according to self-reported performance metrics as of the end 2025—Chandra OCR (Datalab 2025) and olmOCR. For all practical purposes, both models will only run on a GPU, and only acceptably fast on nearly top-of-the-line hardware. Chandra needs ∼18 GB of VRAM just for the model, and olmOCR needs ∼12 GB. Memory requirements spike quickly as batch size increases.

Both Chandra and olmOCR are very likely to get stuck or "loop hallucinate." This is where the model repeats the same set of output tokens indefinitely until it hits its max allowed output tokens cap. This behavior means some percentage of output is totally useless. More important, this issue greatly slows down the process of OCR for thousands of images, let alone for millions of documents.

For efficient use of GPU time, LLMs must process a set of images as a batch, where the model performs the necessary matrix operations across many images at once. Compute time increases as a function of the longest running task in a batch. This is true if even one image in a batch gets stuck. This behavior can be controlled to some degree with a variety of parameters, for example, `temperature`, `frequency_penalty`, or `repeat_penalty` or by capping output tokens. However, tweaking these parameters, even a small amount, can adversely affect performance. These parameters can be technically difficult to set depending on the interface available with the model. In particular, with Chandra, this means abandoning the simple Python wrapper provided by chandra-ocr and instead working with lower level objects from the package.

Finding the optimal settings to minimize loop responses requires iteration using training/test/holdout splits—as well as labeled data against which to evaluate performance. This is a tall order. In practice, settings often end up being determined heuristically and approximately. Table 18 quantifies how often loops occur in various OCR methods *after* optimization to lower their occurrence.

Another drawback of the LLM models considered for OCR (and of nearly all LLM OCR models) is that they do not produce bounding boxes—text is returned as one block in reading order without context as to its placement in the image. Depending on project needs, this could be a deal breaker. For our project specifically, bounding boxes are valuable because they enable verification of OCR output and LLM labels by allowing relevant text to be highlighted on the original image. Their absence is therefore a major, but not disqualifying, drawback. It is possible to use a separate ML segmentation model to isolate parts of the image into lines, boxes, and tables and then to pass these smaller images into an LLM. Such work is outside the scope of this project and would add a great deal of technical overhead, time, and cost.

Finally, as mentioned in §2.2, LLMs attempt to make OCR text coherent. Most of the time, this behavior improves various measures of performance, but it can result in substantial hallucination. For our task, even with this type of hallucination, we were still able to match relevant snippets via fuzzy search, and the meaning of the returned snippet was usually preserved sufficiently for an LLM to accurately identify racially restrictive covenants. For example, "Caucasian Race" was sometimes interpreted as "Canadian Race," which remains an explicitly racial—or at least nationality-based—restriction, albeit with completely incorrect context and meaning. There were other cases where LLMs' coherency bias destroyed meaning, such as the one demonstrated in Table 8, where "Caucasian Race" was read as "Concession desert." Bias towards coherence over accuracy could be a very serious problem in other use cases, perhaps when looking at quantitative data, proper names, or uncommon English language phrases.

Chandra proved particularly challenging to use. The documentation is insufficiently detailed, and it is not packaged to work with vanilla vLLM. Instead, the required "`chandra_vllm`" needs `sudo` access to run, which we could not grant. We therefore interfaced with Chandra via HuggingFace's API and the chandra-ocr Python wrapper, which was buggy. For instance, its prepackaged batching function did not properly

submit tasks as a batch to the GPU. We found no useful guidance on accelerating performance, addressing loop hallucination (aside from unanswered GitHub tickets[20]), or choosing optimal input image size. We suspect Chandra resizes images internally, but could not confirm whether resizing them ourselves would help without digging deep into the codebase. We could not reduce loop hallucination to an acceptable level, even when turning hyperparameters up to the point of nonsensical or empty output (see Table 19). Chandra's aggressive memory use limited us to a batch size of 10, even on an H200, and our fastest run time with coherent output was 90 seconds per image. Given this, we abandoned Chandra after running it on just 3,010 images known to contain a covenant.

An evaluation on these same 3,010 images (Table 26 in Appendix H) shows that Chandra's output performance is similar to olmOCR's, despite being significantly less user friendly. Recall for Chandra on the whole image text, filtered with fuzzy-search, and labeled by Claude Sonnet 3.7 is better than for olmOCR run on the whole image, but worse than for olmOCR on single pages or Textract. Processing time is 15 times faster for olmOCR—even when nearly doubling the number of iterations due to pre-splitting two page images into single pages—than for Chandra, and performance for olmOCR is better.

Alhough better than Chandra's, olmOCR's documentation is also not ideal. It will only work properly with images less than or equal to 1,024 pixels in both height and width. It will run on larger images without any errors but with extremely poor performance. It is crucial to carefully control how the images are downscaled. We found using openCV for resizing with `cv2.INTER_AREA` interpolation worked best. It is likely that the optimal method depends upon the input corpus of images. Further, as nearly all of our images are of left- and right-hand pages of an open deed book, simply shrinking the image to max dimension 1,024 results in a very small image, with significant information lost in the scaling process. This greatly decreased performance relative to splitting the images into individual single page components. Figure 10 compares image CER values for whole image and single page images with olmOCR. Table 18 demonstrates how frequently olmOCR fails to produce *any* result when images are not split into single pages. This splitting process can be trivial if the boundary is nearly always centered in the image or much more complicated, depending on the nature and quality of the imaging method used. This will not be an issue when there is one page per image use case of one page per image. Unlike in Chandra's case, we were able to control olmOCR's tendency to get stuck in a loop by turning `temperature` up to 0.1 and setting `repetition_penalty` to 1.5. We recommend experimenting with `frequency_penalty` as well. We found `frequency_penalty` more effective in the, admittedly different, use case of gpt-oss for classification.

olmOCR's first-class integration into vLLM improved usability substantially, making memory management, iteration, inference tuning, and performance optimization fairly straightforward. Although we achieved much faster performance than with Chandra—about 6 seconds per image on a top-of-the-line H200—olmOCR still requires substantial run time on a large corpus. Further optimizations are surely possible with more engineering effort, but a large gap in total run time is likely to persist compared to docTR or Amazon Textract.

20. https://github.com/datalab-to/chandra/issues/51

In terms of frontier LLMs for OCR, besides Gemini, we experimented with GPT-5. We found GPT-5 to perform excellently as an OCR engine on images of brief passages of text. However, as the length of text contained in an image increased, the model began to aggressively hallucinate. While many of these issues, excepting cost, could be overcome by segmenting images into smaller pieces, we think any solution would be fairly brittle and take significant iteration and development time. Cost would quickly become an issue other than for very small sets of images.

We give the prompt we used for Gemini 2.5 Flash as an OCR engine in Figure 9 and the token budget in Table 17. This prompt was developed iteratively in collaboration with Anthropic Opus 4.8 and with human verification of the resulting Gemini output. We ran images through various versions of the prompt until we had addressed common failure modes, such as loop and empty responses. Our prompt instructs the model to return "[SECTION BREAK]" when it encounters breaks between deeds that the original typists delineated with repeated characters, such as "FIFIFIFIFIFI" or "~~~~~~~~." Without specific instructions to pass over these strings, Gemini would waste huge amounts of output tokens or fail completely. Figure 8 provides an example of this type of delineation at the beginning of a typed deed.

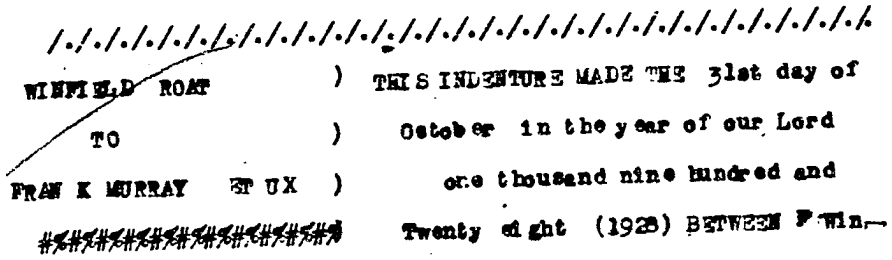


Figure 8: Deeds were recorded in continuous fashion with the beginning and end of the deed delineated, in typed deeds, with repeating characters selected by the typist. This example appears in the middle of a page. These repeating characters were difficult for LLMs to parse, and we needed to provide specific guidance about these strings in our prompt.

Table 18 quantifies the prevalence of "loop" hallucination and empty responses across LLM OCR methods with Textract as a point of comparison. These statistics are *after* prompt engineering and adjusting LLM sampling parameters to decrease these errors as much as possible (see §4.3 and Figure 9)—starting results were significantly worse for olmOCR in particular. These loop statistics do not count repeated [SECTION BREAK] outputs by Gemini as a loop. However, such output sometimes actually does represent a loop response in that Gemini simply decides everything remaining in the document to transcribe is a [SECTION BREAK]. On the other hand, there are some section breaks these models did transcribe, which ideally would not be counted as a loop. Owing to the general difficulty in systematically quantifying output as a "loop hallucination," loop figures should be taken with a grain of salt and be read as directional and relative. Generally, the harder the image is to read, the more likely LLM as OCR is to loop hallucinate. Table 19 looks at loops on the small subset of images where we ran Chandra.

Prompt:

System:

- You are a transcription engine. You output only the verbatim text of documents provided. You never add commentary, explanations, introductions, summaries, or any text that is not present in the source document.

Requirements:

- Transcribe ALL text visible in the image, from the very first word to the very last word.
- Output the text in correct reading order: verso (left page) first in its entirety, then recto (right page) in its entirety. Follow the natural reading flow within each page (top to bottom, respecting columns if present).
- Preserve paragraph breaks and line structure where meaningful.
- Transcribe exactly what is written, including punctuation, capitalization, and spelling as it appears (do not modernize or correct).
- Pay special attention to text at the top edges, bottom edges, margins, headers, footers, page numbers, and catchwords — these must not be omitted.

Handling repeated-character separators:

- The original typist may have used long sequences of repeated characters as section dividers (e.g. FIFIFIFIFI, ##########, ---------, ==========, ~~~~~~~~, ccccccccc).
- When you encounter any such sequence, replace it with the single token [SECTION BREAK] and move on immediately.
- Do NOT attempt to reproduce the repeated characters or count their repetitions.
- Do NOT dwell on or loop over these sequences under any circumstances.

Avoiding generative loops:

- If you notice your own output beginning to repeat the same word, phrase, or character sequence more than 3 times consecutively, stop immediately — this is a transcription error, not a feature of the document.

Strict output rules:

- Output ONLY the transcribed text from the document.
- Do NOT include any commentary, introductions, explanations, summaries, notes, disclaimers, or descriptions of the image.
- Do NOT wrap the output in code blocks, quotation marks, or markdown formatting.
- Do NOT add labels like “Verso:” or “Recto:” unless they appear in the original document.

Figure 9: Prompt used with Gemini 2.5 Flash for transcription (“OCR”).

Table 17: Token budget for Gemini 2.5 Flash as OCR.

LLM	**Thought**	**Max Output** (Thought + Output)
Whole Image	600	2,200
Single Pages	600	1,450

Table 18: Prevalence of OCR loop hallucination and empty responses for LLM based OCR tools and Textract for covenanted images and images in deeds containing covenanted images in core sample. Figures reflect results after prompt and sampling parameter tuning to minimize these issues. Failures are the sum of loop and empty response counts. Empty responses should be considered catastrophic, while not all loop responses result in unusable output.

Image Type	N Images	OCR	N Loop	N Empty	N Failed	Loop %	Empty %	Failed %
Handwritten	1,883	olmOCR	12	104	126	0.6%	5.5%	6.2%
		Gemini 2.5 Flash, SP	20	0	20	1.1%	0.0%	1.1%
		Gemini 2.5 Flash	11	8	19	0.6%	0.4%	1.0%
		olmOCR, SP	3	16	19	0.2%	0.8%	1.0%
		Textract	0	0	0	0.0%	0.0%	0.0%
Photostat	3,019	olmOCR	16	197	213	0.5%	6.5%	7.1%
		olmOCR, SP	10	40	50	0.3%	1.3%	1.7%
		Gemini 2.5 Flash	38	5	43	1.3%	0.2%	1.4%
		Gemini 2.5 Flash, SP	38	0	38	1.3%	0.0%	1.3%
		Textract	0	0	0	0.0%	0.0%	0.0%
Typewritten	12,552	olmOCR	23	133	156	0.2%	1.1%	1.2%
		Gemini 2.5 Flash	27	43	70	0.3%	0.2%	0.6%
		Gemini 2.5 Flash, SP	46	3	46	0.4%	0.0%	0.4%
		olmOCR, SP	15	1	16	0.1%	0.0%	0.1%
		Textract	0	0	0	0.0%	0.0%	0.0%

Table 19: Prevalence OCR loop hallucination and empty responses for 3,010 images OCRed by Chandra. Figures reflect results after prompt and sampling parameter tuning to minimize these issues. Failures are the sum of loop and empty response counts. Empty responses should be considered catastrophic, while not all loop responses result in unusable output.

OCR	N Loop	N Empty	N Failed	Loop %	Empty %	Failed %
Chandra	82	0	82	2.7%	0.0%	2.7%
olmOCR	5	62	67	0.2%	2.1%	2.2%
Gemini 2.5 Flash, SP	17	0	17	0.6%	0.0%	0.6%
Gemini 2.5 Flash	9	0	9	0.3%	0.0%	0.3%
olmOCR, SP	0	7	7	0.0%	0.2%	0.2%
Textract	0	0	0	0.0%	0.0%	0.0%

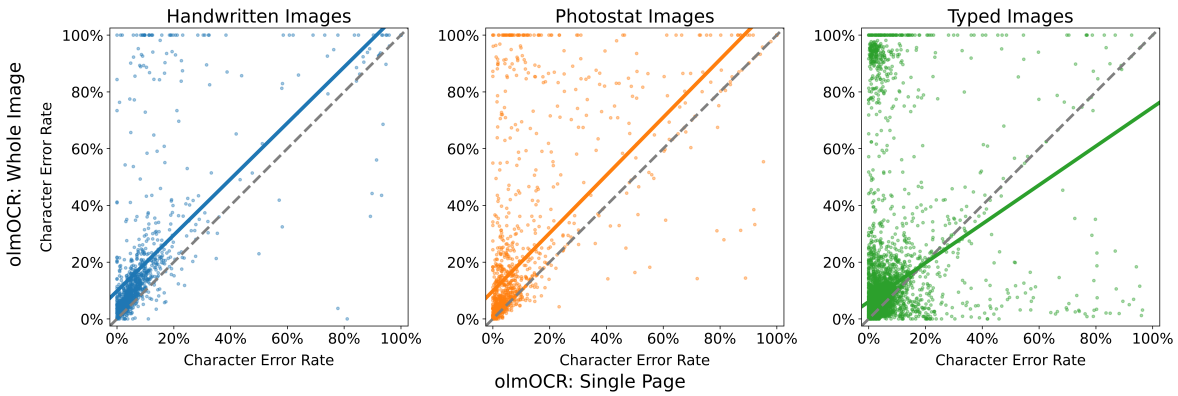


Figure 10: Whole image CER vs. single image CER for olmOCR. An empty OCR response was assigned CER = 100%. The gray dashed line represents one-to-one correspondence of CER between models. Points above the one-to-one line indicate better performance by single page OCR. The solid line is a simple linear regression. Handwritten images are plotted in blue, photostat images in orange, and typed images in green. Clustering along $y = 100\%$ indicates catastrophic failure of the whole image OCR. Clustering in the top left indicates poor performance of the whole image OCR where the single page OCR is very good. For handwritten and photostat images, the single page OCR performs nearly universally better. For typed images, single page OCR typically performs best, but there are a subset of images along $y \approx 5\%$ that indicate substantially better performance with whole image OCR than with single page OCR.

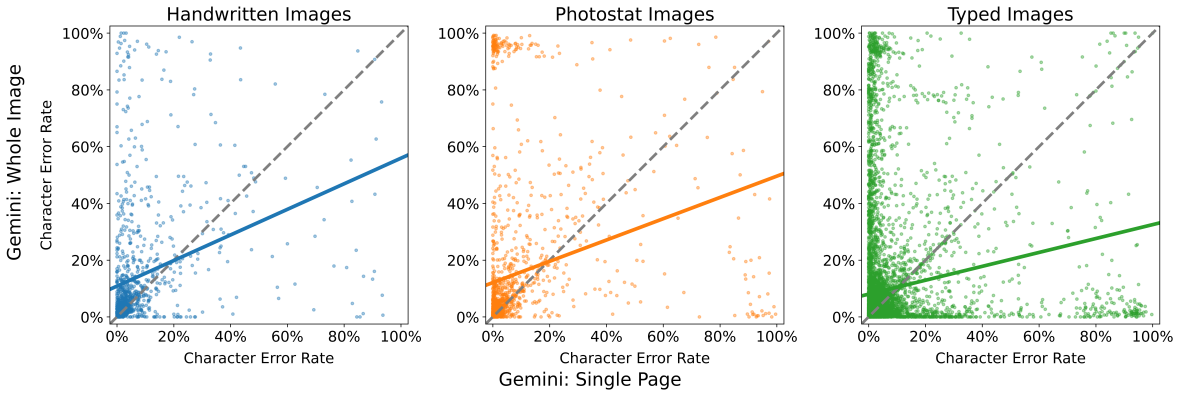


Figure 11: Whole image CER vs. single image CER for Gemini 2.5 Flash. An empty OCR response was assigned CER = 100%. The gray dashed line represents one-to-one correspondence of CER between models. Points above the one-to-one line indicate better performance by single page OCR. The solid line is a simple linear regression. Handwritten images are plotted in blue, photostat images in orange, and typed images in green. Single page OCR typically performs better for handwritten and photostat images, although the effect is not as strong as with olmOCR. For photostat images, the clustering in the top left and bottom right indicate subsets of images where single page OCR and whole page OCR perform better, respectively. For typed images, there is strong clustering along both $x \approx 5\%$ *and* $y \approx 5\%$. This indicates there are types of images where the OCR is substantially better quality with either whole image or single page OCR, but not both. The clustering along $y \approx 5\%$, where whole image OCR is better than single page OCR, is much more pronounced for Gemini 2.5 Flash than it is for olmOCR.

G OCR Evaluation

In this appendix, we provide a detailed breakdown of character error rates (CERs) across all tested OCR models. We break out results not only by image type—handwritten, photostat, and typewritten—but also by covenanted images, non-covenanted images, and the racial restriction passage alone within covenanted images. This breakdown demonstrates that OCR quality shows no obvious selection across models based on whether an image is covenanted. Although average CERs are slightly higher for covenanted images than for non-covenanted images, the difference is small, and relative performance across models remains highly similar regardless of how the images are segmented. Most important, Textract 2021 shows no strong selection effect for covenanted images relative to other models. In other words, our sample construction methods—which rely substantially on Textract 2021 output (see §4.2 and Appendix D)—did not introduce a bias toward Textract 2021 overperforming absolutely or proportionally on covenanted images.

We start by giving detailed background into the data used for our CER calculations, then give details of the calculation itself, which leads to detailed tables of performance by model and image type. Finally, we present a broad set of figures comparing performance across OCR models at the image level, offering a sense of how reliably one model over- or underperforms another.

G.1 Image Transcription

We partnered with a data labeling firm to obtain human transcriptions with bounding boxes for all images belonging to a deed identified as containing a racial covenant. Due to cost considerations, only portions of the deed relevant to our econometric research were transcribed. These included the transaction date, address or metes and bounds descriptions for all properties listed in the deed, the previous transaction information, and the racially restrictive covenant. Metes and bounds descriptions were transcribed with separate bounding boxes for each property, i.e., if two properties were transferred in the deed, the metes and bounds describing each property were transcribed in separate bounding boxes.

Bounding boxes did not span pages breaks—text at the bottom of the left page was transcribed in one bounding box, and the remaining text was transcribed in a separate bounding box at the top of the right page. By necessity, given the functionality of most labeling tools, transcriptions spanning image breaks were also transcribed in separate bounding boxes. Labelers inserted `\p` or `\i` at the end and beginning of transcriptions that spanned page or image breaks in order to better facilitate reconstruction of text in reading order. Labelers also transcribed line breaks as they appeared in the image. These extra annotations were removed from transcriptions for the purpose of OCR evaluation. A total of 36,087 transcriptions were obtained. Table 20 provides the distribution of transcriptions by image type and transcription type.

Figure 12 compares the distribution of the number of characters in each transcription between transcription types for the set of all transcribed images and those images containing a covenant. Percentages in Figure 12 are calculated in terms of the total number of all transcriptions in the set of images, regardless of transcription type, i.e., ∼15% of all transcriptions are of transaction dates and contain ∼100 characters.

Table 20: Distribution across image and transcription type for human transcriptions from images in deeds containing a racially restrictive covenant.

	Transaction	Metes & Bounds	Previous Transaction	Racial Restriction	**Total**
Handwritten	592	1,069	707	641	3,009
Photostat	981	1,394	1,182	1,039	4,596
Typed	5,873	9,599	6,624	6,386	28,482
Total	7,446	12,062	8,513	8,066	36,087

We observe that the shapes of the distributions are similar between all images and covenanted images. However, transaction date and metes and bounds transcriptions occur less often in covenanted images; this is expected, given where each type of transcription occurs within a deed. Transaction types are plotted in the order in which they normally appear within a deed. The transaction date and racial restriction are the furthest apart, therefore covenanted images are less likely to also contain a transaction date.

Next, Figure 13 shows the number of transcriptions (of any type) that appear in an image. We split images by image type due to distinct formats and qualities that can impact the spacing of text within each type of image. Between all images and covenanted images, we again see similar distributions. The largest difference is seen in photostat images containing a covenant; they are more likely than the set of all photostat images to contain only one transcription. The only transcription present in a covenanted image with only one transcription must, by definition, be the transcription of the racial covenant.

G.2 CER Calculation

Labelers were instructed to use diplomatic transcription in order to preserve the text *exactly* as it appeared in the image. If text was illegible, labelers transcribed a question mark, even if the text could be determined by context. When transcribed correctly, according to instruction, one question mark was used for each illegible character. When large portions of the text were illegible, and it was not clear exactly how many characters were missing, a single question mark was used. Figure 14 provides an image of a racially restrictive covenant from a 1917 deed with a corrected and standardized transcription, the actual labeled transcription utilizing diplomatic transcription, and the OCR output from Textract.

The standard formula for character error rate is $CER = (S + D + I)/N$ where S, D, and I are the number of character substitutions, deletions, and insertions. N is the total number of characters in the transcription text. We modified the CER calculation to treat question marks used by the labelers as wild cards, i.e., any character produced by the OCR at the same position as the question mark in the human transcription was treated as "correct" and did not contribute to the numerator. The goal was to not punish an OCR model for misreading something that a human indicated was not legible.

To determine S, D, and I, each individual transcription was aligned with the OCR

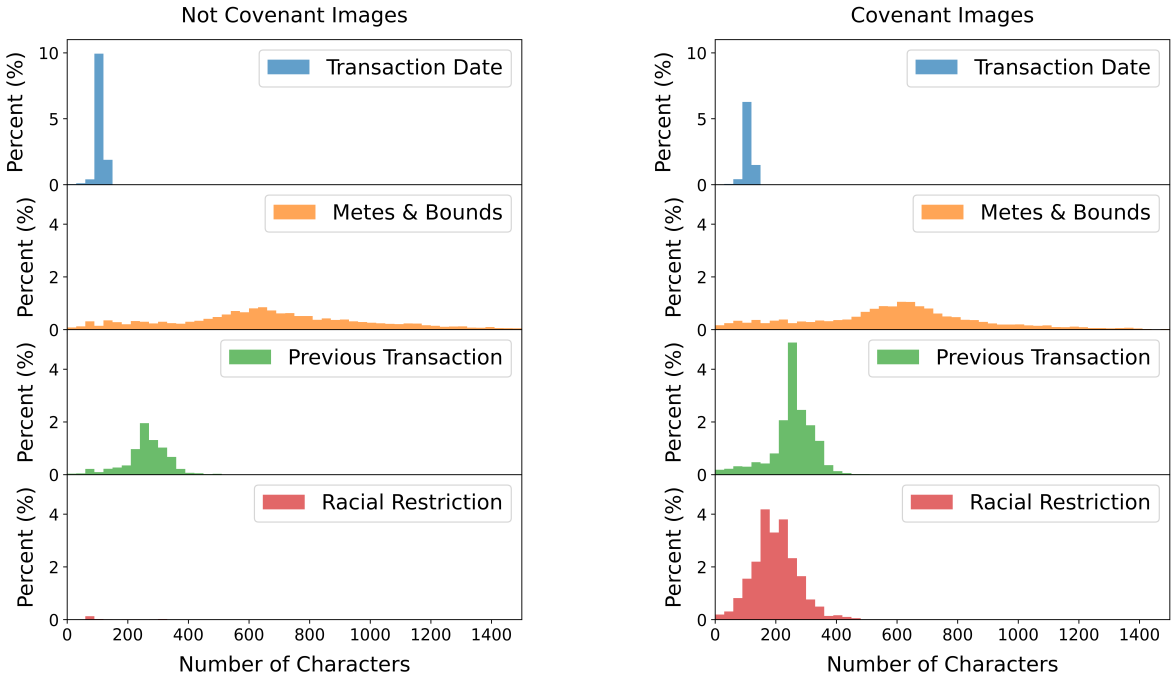


Figure 12: Distribution of the number of characters in each transcription by transcription type. Percentages reflect the number of transcription in each bin divided by the total number of transcriptions, regardless of type. Left: distributions of character count for transcriptions from transcribed images not containing a racial covenant. Right: distributions of character count for transcriptions only appearing in an image containing a racial covenant. The number of characters in transaction date transcriptions are tightly distributed because the language is highly standardized across image types; it is both shorter and always located at the very beginning of a deed, both of which decrease the likelihood of a single transaction date being transcribed into separate boxes (each with fewer characters) due to a page or image break. The left tails on the distributions regardless of transcription type can be attributed to short transcriptions created by a page or image break. The metes and bounds property descriptions are by far the most heterogeneous, both intrinsically due to the information they contain but also due to format variations across types of deeds. In each figure, the long right tail of the metes and bounds distribution is not shown ($\sim 0.8\%$ and $\sim 0.3\%$ of transcriptions for the left and right figures, respectively). A small percentage of the full racial restriction text appears in images classified as not containing a covenant because the text was split across images; the image containing the explicit racial phrase was labeled as the image containing the covenant. The overall shape of the distribution of character counts is similar between transcription from not covenanted images vs. covenanted images. Transaction date and metes and bounds transcriptions are slightly less likely to appear in covenanted images, which is to be expected, given the order in which they appear in a deed document and the typical length (i.e., number of images) in a deed document.

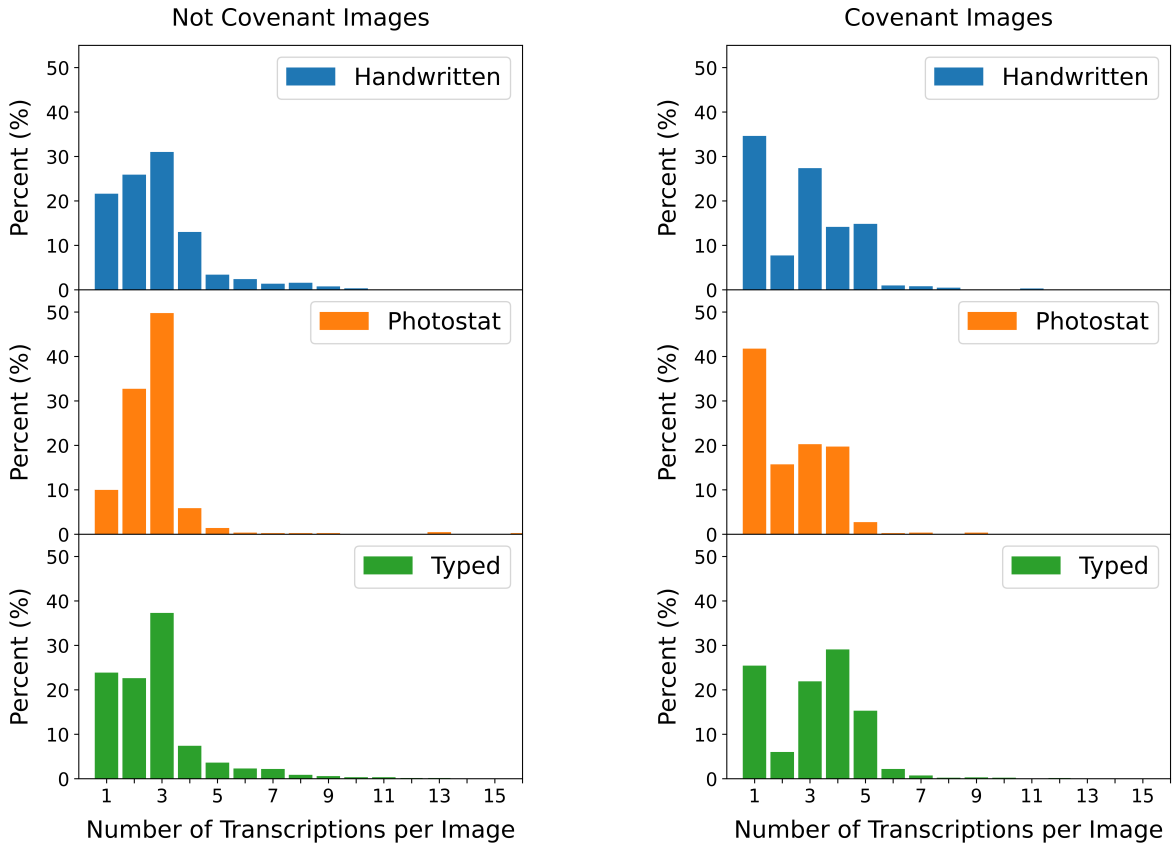
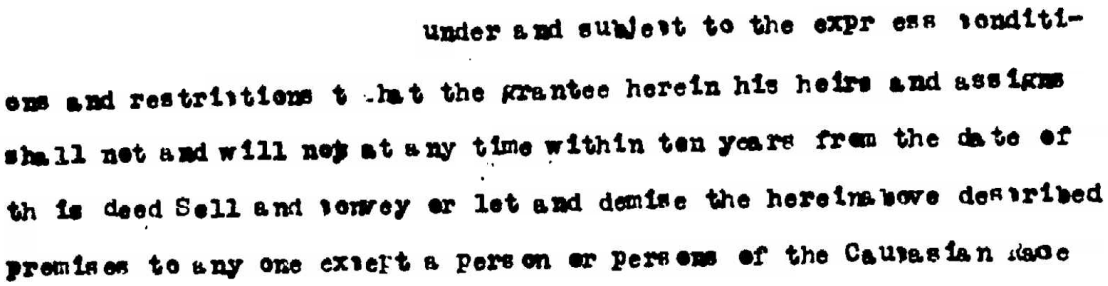


Figure 13: Distribution of the number of transcriptions appearing in each image by image type. Left: distributions of number of transcriptions per image for transcribed images not containing a covenant. Right: distributions of number of transcriptions per image for images containing a racial covenant. Images not containing a covenant and not containing a transcription are omitted in this comparison with covenant images since covenant images must, by definition, include at least one covenant. Similar distributions for handwritten and typed images vs. photostat images can be attributed to the difference in document formatting. The long tail to the right is typically attributed to deeds containing multiple properties in which each property has its own metes and bounds and previous transaction transcription.



Corrected	under and subject to the express conditi- ons and restrictions that the grantee herein his heirs and assigns shall not and will not at any time within ten years from the date of this deed sell and convey or let and demise the herein above described premises to any one except a person or persons of the Caucasian Race
Diplomatic	under and subject to the expr ess ?onditi- ons and restri?tions t hat the ?rantee herein his heirs and assigns shall not and will no? at any time within ten years from the da te of th is deed Sell and ?onvey or let and demise the hereinabove des?ribed premises to any one ex?ept a person or persons of the Cau?asian Race
Textract: Dec '25	under and subject to the expr 088 sonditi- one and restrictions t .hat the grantee herein his heirs and assigns shall net and will not at any time within ten years from the date of th is deed Sell and somey or let and demise the hereinabove described premises to any one exsept B Person or Persons of the Causasian lace

Figure 14: At top, a sample of the racially restrictive language from a deed in 1917. Next, the transcription of the passage after being fully cleaned, the actual diplomatic transcription produced by human labelers, and finally, the corresponding OCR output from Textract in December 2025. In the OCR output, text diverging from the diplomatic transcription are highlighted in yellow, while the OCR text corresponding to wildcards (question marks) in the diplomatic transcription are highlighted in red.

output from the entire image using the Smith-Waterman algorithm as implemented by the Python `string2string` package (Suzgun, Shieber, and Jurafsky 2023). We used default weights for a match (1), mismatch (-1), and gap (-1). Out of an abundance of caution, we changed the default character representing a gap in the aligned strings to "*", since it was exceedingly rare in both the human and OCR transcriptions of our documents. The Smith-Waterman algorithm is appropriate because the transcriptions are only a portion of the entire text in the image, therefore a *local* alignment algorithm is required.

Before transcription and OCR strings were aligned, both were normalized by: 1) making all characters lowercase, 2) changing all line breaks to single spaces, 3) removing leading and trailing whitespace but maintaining all whitespace *within* the string, and 4) removing punctuation. We defined punctuation using Python's built-in `string.punctuation` which includes `!#$%&"(*)+,-./:;<=>?@[\]^_{|}~`. For human tran-

Table 21: Using a portion of the text in Fig. 14, we demonstrate the normalization of the strings before alignment, the standard CER, and the CER allowing for wildcards. Normalizing the strings forces all characters to be lower case, the linebreak in "conditions" was changed to a single space, and the hyphen in "conditions" was removed. Spaces *within* the string count as characters in N. The human-labeled diplomatic transcription is treated as ground truth, or the reference text. In the Textract OCR, deviations from the transcription text that contribute to the numerator of CER are highlighted in yellow. Characters that align with the wildcards (question marks) in the transcription and *do not* contribute to the numerator of CER when wildcards are permitted are highlighted in red.

Source	Aligned String		
Diplomatic	the expr ess ?onditi ons and restri?tions	$N = 41$	
Textract: Dec '25	the expr 088 sonditi one and restrictions	CER Standard	$\frac{S+D+I}{N} = \frac{6}{41} = 14.6\%$
	the expr 088 sonditi one and restrictions	CER with wildcard	$\frac{S+D+I}{N} = \frac{4}{41} = 9.8\%$

scriptions, the question mark was removed from `string.punctuation` so that the question marks from the labelers would not be removed.

We note that, as demonstrated in Table 21, the inclusion of wildcards in the character error rate calculation improves the CER. Not all individual wildcards impact the CER; consider the string used in Table 21: the human labeler could have easily inferred and transcribed "restrictions" rather than "restri?tions." In that case, Textract correctly transcribed "restrictions", so the difference between "?" and "c" would have no impact on the CER.

The CER with wildcard was calculated for each transcription. If an OCR model did not return any output for an image, any transcriptions in that image were assigned $CER = 100\%$. To produce a single CER measurement for an OCR model, we calculate the average CER weighted by transcription length. In Table 4, for example, $\overline{CER}_w$ is the average CER from every transcription in a handwritten image weighted by the respective length of the transcription. In Figure 3, for example, we are plotting a single CER value for each image, which is calculated as the average CER from every transcription from *that specific image* weighted by the respective transcription length.

Our methodology for calculating CER was consistent across all image types and OCR models, which preserves the relative differences observed between subsamples. Regardless, given the removal of punctuation and our generous treatment of characters with which labelers had difficulty, as an intrinsic value, the $\overline{CER}_w$ we report should be considered "best case scenario" for the quality of images in our sample.

G.3 CER by OCR Model and Image Type

In Tables 22, 23, and 24, we present weighted average CER ($\overline{CER}_w$) for transcriptions in all images (column 2), images not containing a covenant (column 4), images containing a covenant (column 6), and for racial restriction transcriptions only (column 8). The CER_{P95} columns give the 95% percentile CER value for the distribution of transcription CER values. All OCR models are given in each table; models not run on

those images are indicated by empty cells. Models denoted with SP indicate that the OCR was run on single page images.

In general, we find that $\overline{CER}_w$ are similar regardless of the subset of transcriptions over which it is calculated. The $\overline{CER}_w$ is usually lower for covenanted images and slightly more so for racial restriction transcriptions, which is not surprising, given that images were only transcribed *if* they belonged to a deed containing a racial covenant, which means the racial covenant had to be legible enough to identify it in the first place.

We were concerned that we may have introduced selection bias favoring Textract 2021 and Textract 2025 given our sampling method. About 60% of our core sample was selected based on matching some full-text search query on Textract 2021 output. It is plausible that this step could have worked to preference images on which Textract 2021 happened to perform well, and especially so for images that did have covenants. However, relative differences in performance across models is similar no matter if calculated for non-covenanted images, covenanted images, or just racial restrictions. This strongly suggests that we do not have a substantial selection problem in our core sample.

Table 22: Handwritten: Character error rate by model and category of input image. Weighted CER colorized from best (green) to worst (red) across all CER columns in table.

Model	All Images		Not Covenanted		Covenanted		Racial Restriction	
	$\overline{CER}_w$	CER_{P95}	$\overline{CER}_w$	CER_{P95}	$\overline{CER}_w$	CER_{P95}	$\overline{CER}_w$	CER_{P95}
Chandra	12.9%	92.1%	—	—	12.9%	92.1%	13.8%	93.9%
docTR: resnet+vgg16	—	—	—	—	—	—	—	—
Gemini 2.5 Flash, SP	7.3%	37.8%	7.7%	39.7%	7.0%	37.0%	6.0%	37.2%
Gemini 2.5 Flash	14.8%	80.4%	15.0%	83.0%	14.6%	76.9%	12.2%	72.4%
Kraken: McCATMuS	46.6%	92.7%	46.8%	92.3%	46.5%	93.0%	43.5%	88.1%
Kraken: RevCity	37.0%	94.3%	37.2%	93.8%	36.9%	94.4%	34.6%	93.2%
olmOCR, SP	11.3%	52.7%	11.6%	47.6%	11.0%	58.4%	10.0%	86.2%
olmOCR	21.0%	97.0%	21.5%	94.0%	20.6%	100.0%	18.7%	100.0%
TrOCR	37.3%	82.7%	37.8%	85.0%	36.8%	81.0%	32.3%	70.5%
Textract: Nov '21	38.0%	83.6%	39.3%	84.7%	36.7%	82.6%	33.5%	76.2%
Textract: Dec '25	28.2%	72.0%	29.3%	74.1%	27.1%	69.5%	24.5%	68.2%
Textract: Dec '25, Layout	28.2%	72.0%	29.3%	74.1%	27.1%	69.5%	24.5%	68.2%
Textract: Dec '25, SP	27.6%	72.3%	27.6%	72.5%	27.6%	72.2%	23.5%	68.7%

Table 23: Photostat: Character error rate by model and category of input image. Weighted CER colorized from best (green) to worst (red) across all CER columns in table.

Model	All Images		Not Covenanted		Covenanted		Racial Restriction	
	$\overline{CER}_w$	CER_{P95}	$\overline{CER}_w$	CER_{P95}	$\overline{CER}_w$	CER_{P95}	$\overline{CER}_w$	CER_{P95}
Chandra	—	—	—	—	—	—	—	—
docTR: resnet+vgg16	28.9%	85.3%	28.5%	85.3%	29.3%	85.3%	27.9%	85.7%
Gemini 2.5 Flash, SP	7.4%	52.7%	7.1%	46.6%	7.7%	64.5%	8.5%	74.9%
Gemini 2.5 Flash	16.5%	93.3%	18.8%	95.8%	13.5%	86.6%	11.8%	88.6%
Kraken: McCATMuS	—	—	—	—	—	—	—	—
Kraken: RevCity	—	—	—	—	—	—	—	—
olmOCR, SP	8.0%	57.4%	7.5%	49.4%	8.6%	63.8%	7.2%	58.9%
olmOCR	19.0%	100.0%	16.0%	96.7%	22.9%	100.0%	21.0%	100.0%
TrOCR	—	—	—	—	—	—	—	—
Textract: Nov '21	14.1%	60.0%	14.4%	64.4%	13.7%	53.6%	11.8%	50.0%
Textract: Dec '25	10.9%	70.7%	7.9%	52.9%	14.9%	87.1%	11.3%	75.6%
Textract: Dec '25, Layout	10.9%	70.7%	7.9%	52.9%	14.9%	87.1%	11.3%	75.6%
Textract: Dec '25, SP	7.4%	53.4%	7.3%	54.3%	7.4%	47.6%	6.5%	35.9%

Table 24: Typewritten: Character error rate by model and category of input image. Weighted CER colorized from best (green) to worst (red) across all CER columns in table.

Model	All Images		Not Covenanted		Covenanted		Racial Restriction	
	$\overline{CER}_w$	CER_{P95}	$\overline{CER}_w$	CER_{P95}	$\overline{CER}_w$	CER_{P95}	$\overline{CER}_w$	CER_{P95}
Chandra	—	—	—	—	—	—	—	—
docTR: resnet+vgg16	18.5%	54.7%	20.2%	57.4%	17.4%	53.0%	17.4%	51.3%
Gemini 2.5 Flash, SP	4.3%	35.1%	5.2%	50.5%	3.8%	31.0%	1.8%	4.5%
Gemini 2.5 Flash	8.2%	77.4%	7.4%	77.2%	8.7%	77.6%	6.2%	53.8%
Kraken: McCATMuS	25.8%	67.6%	26.1%	66.3%	25.6%	67.9%	22.8%	54.5%
Kraken: RevCity	—	—	—	—	—	—	—	—
olmOCR, SP	3.3%	19.3%	3.5%	22.6%	3.2%	16.9%	2.2%	7.8%
olmOCR	8.3%	54.9%	9.9%	80.0%	7.3%	31.4%	5.9%	18.8%
TrOCR	—	—	—	—	—	—	—	—
Textract: Nov '21	11.5%	43.5%	11.7%	46.2%	11.3%	42.3%	9.0%	31.6%
Textract: Dec '25	5.1%	27.7%	5.4%	29.9%	4.8%	25.9%	3.6%	12.5%
Textract: Dec '25, Layout	5.1%	27.7%	5.4%	29.9%	4.8%	25.9%	3.6%	12.5%
Textract: Dec '25, SP	5.0%	27.4%	5.3%	29.8%	4.8%	25.8%	3.5%	11.9%

G.4 Additional Image CER Comparisons Between Models

We provide additional figures plotting image $\overline{CER}_w$ vs. $\overline{CER}_w$ for various combinations of models and image types. In all figures, the gray dashed line represents perfect one-to-one correspondence of CER between OCR engines. The solid line is a simple linear regression to guide the eye.

Images falling above the one-to-one line indicate better performance of the OCR engine given on the x-axis, while those below the one-to-one line indicate better performance of the OCR engine given on the y-axis. Images clustering along $y = 100\%$ indicate catastrophic failure of the OCR engine given on the y-axis. Images clustering along $x = 100\%$ indicate catastrophic failure of the OCR engine given on the x-axis.

Commentary on specific figures below may be found in their captions. Figure 15 compares olmOCR on single page images to Textract 2025 for all image types (see §5.1). Figure 16 compares additional open-source ML OCR with Textract 2025 for handwritten images. Figure 17 compares LLM OCR with Textract 2025 for handwritten images. Figure 18 compares among LLM OCR for handwritten images.

Figures comparing whole image vs. single page image OCR for both Gemini 2.5 Flash and olmOCR can be found in Appendix F. Figures comparing Textract OCR from November 2021 and December 2025 can be found in Appendix I.

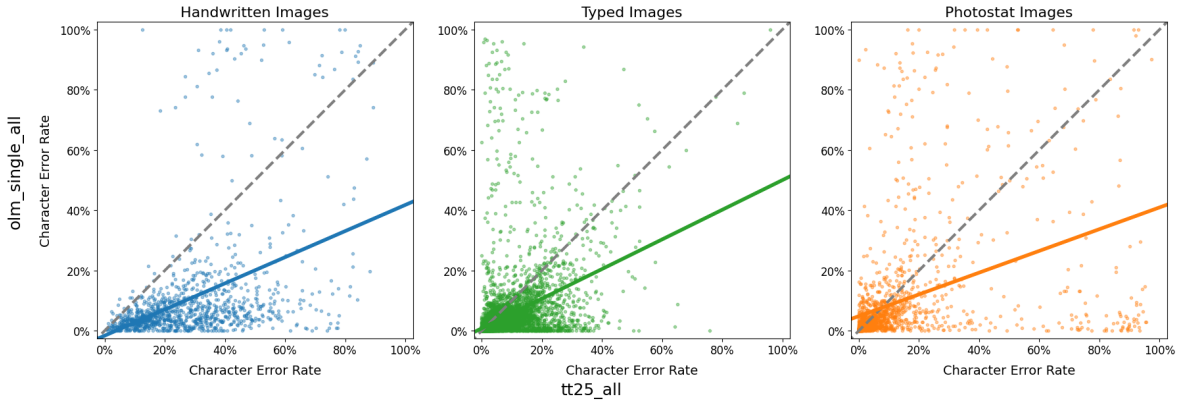
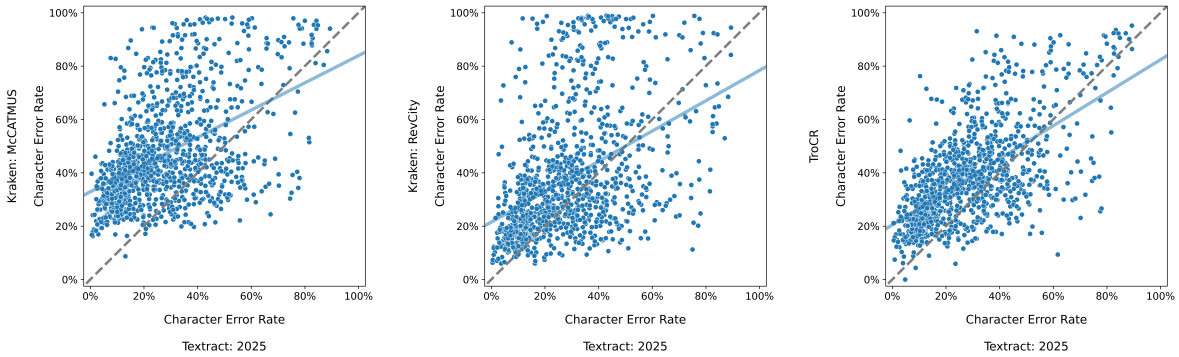
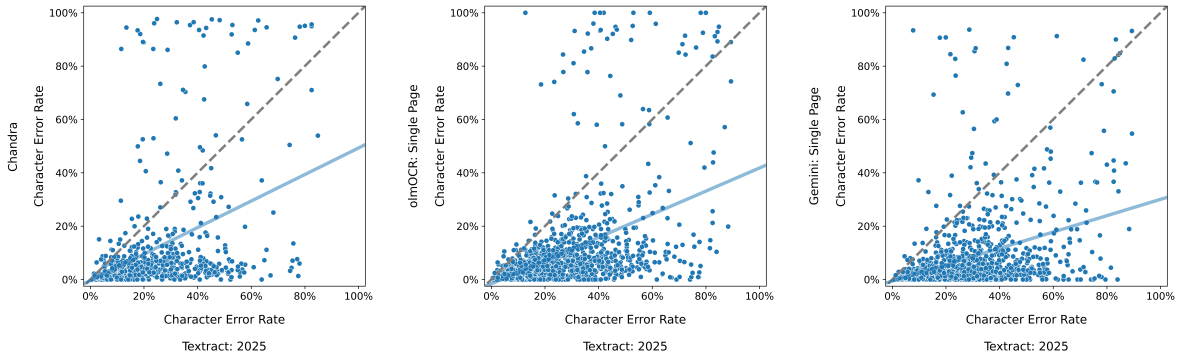


Figure 15: olmOCR single page image CER vs. Textract image CER for handwritten (blue), photostat (orange), and typed images (green). The same patterns are observed here as in the comparison of Gemini 2.5 Flash and Textract discussed in §5.1.



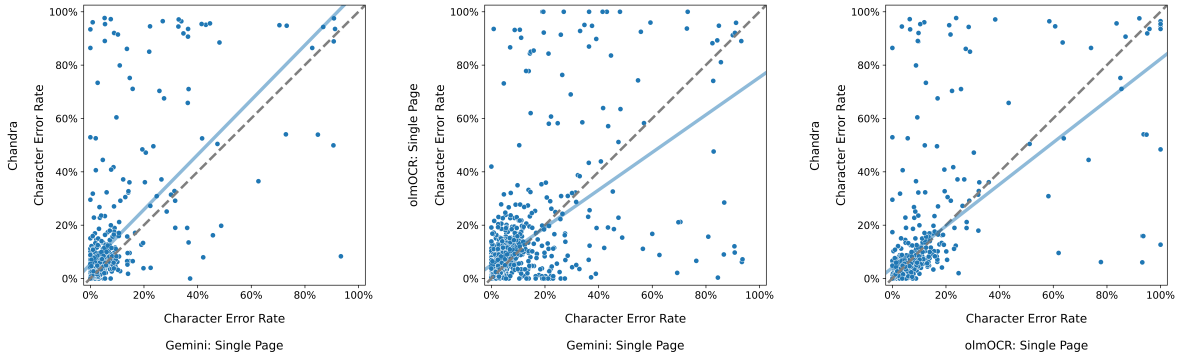
(a) **(b)** **(c)**

Figure 16: Image CER vs. Textract image CER for three open-source OCR engines compatible with handwritten images: (a) Kraken: McCATMuS, (b) Kraken: RevCity, (c) TrOCR.



(a) **(b)** **(c)**

Figure 17: Image CER vs. Textract image CER for three LLM OCR engines compatible with handwritten images: (a) Chandra, (b) olmOCR on single pages, (c) Gemini 2.5 Flash on single pages.



(a) **(b)** **(c)**

Figure 18: Image CER vs. Image image CER for combinations of three LLM OCR engines compatible with handwritten images: Chandra, olmOCR on single pages, Gemini 2.5 Flash on single pages.

H Complete Results

Here we provide extended versions of the performance tables presented in §5.

Table 25 gives a full attribution of all images in our project, the number of covenanted images, and deeds identified. It breaks down our findings across three sources: our core sample, images outside the core sample, and deeds specially imaged for this project to fill gaps in the City of Philadelphia's digital records (see §3.2).

Some deed pages were imaged multiple times during microfilming. In the main body in §5.7, we report counts of unique images; here, we also give counts before this adjustment. In our corpus, duplicated images are not *exactly* identical: each represents a different analog capture of the same physical document and can therefore differ trivially or substantially as a visual artifact.

Table 26 compares performance on 3,010 covenanted images OCRed with Chandra (see Appendix F for detailed discussion). Chandra is very slow, taking 90 seconds per image on H200 hardware, and offers only moderate improvement over olmOCR. In fact, olmOCR on single pages outperforms Chandra while requiring far less run time. Textract outperforms all other methods on this subset.

Table 27 quantifies the difference in Textract performance on single pages versus two-page images. Splitting images into single pages slightly improves performance, yielding 95.9% recall versus 95.0% recall when using snippets passed to Claude Sonnet 3.7. These gains come more from improved full-text search recall (0.6% gain in R_{FTS}) than from better LLM performance on the resulting snippets (0.3% gain in $R_{\text{LLM|FTS}}$). However, realizing these gains requires twice the OCR processing cost, a substantial expense for a small performance gain.

Table 28 quantifies the difference in performance between Textract's vanilla "Detect Document Text" API and Textract with "Layout." We theorized that allowing Textract to consider document structure might improve performance, but we found almost no difference in output. Out of 279,115 images, only 267 showed *any* difference. Among these 267 images, the output text differed by an average of 2.1 Damerau-Levenshtein edits, or 0.09% of the average output length. Moreover, there was *zero* difference in LLM classification performance. Running Textract with Layout costs 2.7 times more for the first million images and 5 times more thereafter.

Table 29 is the complement to Table 10 shown in §5.3. Binning covenanted images by their Textract single-page CER, Table 29 reports whole image text recall from Sonnet 4.5, given Textract single-page and Gemini output for the same documents. Where Textract CER is high, Gemini has a much lower average CER. The output from Gemini for high CER Textract images also leads to better recall for those images, although less so than in the other direction, as shown in Table 10. Taken together, Tables 29 and 10 demonstrate how LLM-based OCR and ML-based OCR can offset each other's failures and improve overall recall.

Tables 30, 31, and 32 are extended versions of Tables 4, 5, and 6. They report performance statistics for every OCR, LLM, and input combination we tested, except Chandra (see Table 26). For all three tables, CER, precision, and recall are colorized by percentile within a given table, with the worst values in red and the best in green.

In these extended performance tables, we break down snippet performance further than in the main body, providing full-text search false-negative counts and recall—respectively, the number of covenanted images excluded by our full-text search queries

and the percentage of all covenanted images with a full-text search match. We also report the number of false negatives the LLM produces given the full-text search snippet, along with the corresponding recall, calculated as a share of covenanted images matched by full-text search. Snippet recall is the product of full-text search recall (R_{FTS}) and LLM recall given the snippet text generated with the full-text search match ($R_{\text{LLM|FTS}}$). This decomposition helps separate LLM-driven from OCR-driven differences in performance.

Table 25: Scale of Data and Covenants Identified

	Imaged For Project	Previously Imaged	Total
Image Total	**414,851**	**4,682,704**	**5,097,555**
in sample	0	279,131	279,131
not in sample	414,851	4,403,573	4,818,424
Covenanted Images	**476**	**8,028**	**8,504**
in sample	0	7,790	7,790
not in sample	476	238	714
Unique Covenanted Images	**444**	**7,951**	**8,395**
in sample	0	7,718	7,718
not in sample	444	233	677
Covenanted Deeds	**436**	**7,701**	**8,137**
in sample	0	7,539	7,539
not in sample	436	162	598

Table 26: Performance on 3,010 covenant containing images on full-text search generated snippets labeled with Claude Sonnet 3.7. Subset of images are those that we successfully ran through Chandra before abandonment due to run time. Snippet recall is the product of full-text search recall (R_{FTS}) and LLM recall given the snippet text generated with the full-text search match ($R_{\text{LLM|FTS}}$). The seconds per image for olmOCR on single pages is the sum of the run time for the left and right page.

OCR	$\overline{CER}_w$	Cost Type	False Negative			Recall			Seconds Per
		OCR	FTS	LLM | FTS	Snippet	FTS	LLM | FTS	Snippet	Image H200
Textract: Dec '25	7.3%	Per Image	74	85	159	97.5%	97.1%	94.7%	
olmOCR, Single Pages	4.3%	GPU Time	111	160	271	96.3%	94.4%	91.0%	6
Chandra	7.2%	GPU Time	203	178	381	93.3%	93.7%	87.3%	90
olmOCR, Whole Image	8.6%	GPU Time	195	300	495	93.5%	89.3%	83.6%	6

Table 27: Comparative performance of Textract run on single page images vs. two page images. Includes 16,471 two page images containing 7,398 covenanted images. LLM method uses snippets and Claude Sonnet 3.7. Snippet recall is the product of full-text search recall (R_{FTS}) and LLM recall given the snippet text generated with the full-text search match ($R_{\text{LLM|FTS}}$). Cost is shown as the rate per thousand images, split between the first million and all subsequent images.

OCR	$\overline{CER}_w$	False Negative			w/ diff	Recall			Average			Cost Per
		FTS	LLM | FTS	Snippet	Label	FTS	LLM | FTS	Snippet	DL Dist	Chars.	% Chars.	1k Images
Single Pages	6.9%	146	157	303	465	98.0%	97.8%	95.9%	249	4,633	6.6%	$3.00/1.20
Whole Image	7.4%	191	177	368	465	97.4%	97.5%	95.0%	249	4,583	7.2%	$1.50/0.60
Difference	0.6%	45	20	65	—	0.6%	0.3%	0.9%	—	50	—	2 ×

Table 28: Comparison of vanilla Textract 2025 "Detect Document Text" vs. "Layout" output. LLM labels derived on snippets passed to Claude Sonnet 3.7. Damerau–Levenshtein edit distance, averages and mean distance to length calculated on subset of 267 images with differing output.

Input images	w Diff. LLM Label	w Textual Diff.	Avg DL Dist.	Avg Length	Avg Dist./Length
279,115	0	267	2.1	4,181	0.09%

Cost per 1k Images	First Million	Thereafter
Detect Document Text	$1.50	$0.60
Layout	$4.00	$3.00

Table 29: Comparison of whole page recall between Gemini single page OCR and Textract when images are binned by Textract CER. Recall colorized by percentile of value shared across both columns.

Textract CER Bin (%)	$N_{\text{cov img}}$	R_{Textract}	$R_{\text{Gemini,SP}}$	Gemini, SP CER
[0, 5)	4,593	98.0%	96.8%	2.5%
[5, 10)	1,664	97.9%	96.8%	4.1%
[10, 20)	779	94.6%	93.7%	7.0%
[20, 30)	205	90.7%	88.8%	9.1%
[30, 40)	122	89.3%	91.0%	10.7%
[40, 50)	90	92.2%	83.3%	16.1%
[50, 60)	49	87.8%	79.6%	17.5%
[60, 70)	29	86.2%	75.9%	23.2%
[70, 80)	36	72.2%	88.9%	14.0%
[80, 90)	32	56.3%	90.6%	13.7%
[90, 100]	21	14.3%	90.5%	10.8%

Table 30: Performance on 17,030 handwritten images containing 617 covenants. Snippet recall is the product of full-text search recall (R_{FTS}) and LLM recall given the snippet text generated with the full-text search match ($R_{\text{LLM|FTS}}$). Character error rate (CER) is calculated on a subset of 1,780 images for which we have human transcribed text.

OCR	$\overline{\text{CER}}_{\text{w}}$	LLM	Cost Type		Precision		False Negatives				Recall			
			LLM	OCR	Whole Text	Snippet	Whole Text	Snippet			Whole Text	Snippet		
								FTS	LLM | FTS	Total		FTS	LLM | FTS	Total
		Gemini 2.5 Flash	Per Image		92.1%						94.0%			
Textract: Dec '25	28.2%	Claude Sonnet 4.5	Per Token	Per Image	98.4%	99.3%	67	32	51	83	89.1%	94.8%	91.3%	86.5%
Textract: Nov '21	38.0%	Claude Sonnet 3.7	Per Token	Per Image	85.7%	99.1%	89	28	127	155	85.6%	95.5%	78.4%	74.9%
Textract: Dec '25	28.2%	Claude Sonnet 3.7	Per Token	Per Image		100.0%		32	60	92		94.8%	89.7%	85.1%
olmOCR, Single Pages	11.3%	Claude Sonnet 4.5	Per Token	GPU Time	99.0%	99.6%	124	53	81	134	79.9%	91.4%	85.6%	78.3%
olmOCR, Single Pages	11.3%	Stanford mistral-rrc	GPU Time	GPU Time	99.8%	99.8%	156	53	78	131	74.7%	91.4%	86.2%	78.8%
Textract: Dec '25	28.2%	Stanford mistral-rrc	GPU Time	Per Image	100.0%	99.8%	266	32	110	142	56.9%	94.8%	81.2%	77.0%
olmOCR, Single Pages	11.3%	gpt-oss-120b	GPU Time	GPU Time	96.7%	97.1%	146	53	98	151	76.3%	91.4%	82.6%	75.5%
olmOCR, Single Pages	11.3%	gpt-oss-20b	GPU Time	GPU Time	95.3%	96.7%	149	53	96	149	75.9%	91.4%	83.0%	75.9%
Textract: Dec '25	28.2%	gpt-oss-20b	GPU Time	Per Image	82.5%	97.0%	182	32	128	160	70.5%	94.8%	78.1%	74.1%
Textract: Dec '25	28.2%	gpt-oss-120b	GPU Time	Per Image	91.3%	98.9%	185	32	149	181	70.0%	94.8%	74.5%	70.7%
olmOCR	21.0%	Claude Sonnet 4.5	Per Token	GPU Time	97.9%	99.5%	236	101	151	252	61.8%	83.6%	70.7%	59.2%
Textract: Nov '21	38.0%	Stanford mistral-rrc	GPU Time	Per Image	99.6%	97.7%	377	28	209	237	38.9%	95.5%	64.5%	61.6%
olmOCR	21.0%	Stanford mistral-rrc	GPU Time	GPU Time	99.4%	99.5%	272	101	138	239	55.9%	83.6%	73.3%	61.3%
Textract: Nov '21	38.0%	gpt-oss-20b	GPU Time	Per Image	64.4%	84.2%	274	28	215	243	55.6%	95.5%	63.5%	60.6%
Textract: Nov '21	38.0%	gpt-oss-120b	GPU Time	Per Image	83.6%	89.4%	270	28	217	245	56.2%	95.5%	63.2%	60.3%
olmOCR	21.0%	gpt-oss-20b	GPU Time	GPU Time	94.4%	96.1%	281	101	174	275	54.5%	83.6%	66.3%	55.4%
olmOCR	21.0%	gpt-oss-120b	GPU Time	GPU Time	95.4%	97.9%	285	101	192	293	53.8%	83.6%	62.8%	52.5%
Kraken: Rev City	37.0%	Claude Sonnet 3.7	Per Token	GPU Time		98.5%		172	118	290		72.1%	73.5%	53.0%
TrOCR	37.3%	Claude Sonnet 3.7	Per Token	GPU Time		100.0%		103	247	350		83.3%	51.9%	43.3%
Kraken: CATMuS	46.6%	Claude Sonnet 3.7	Per Token	GPU Time		99.6%		268	84	352		56.6%	75.9%	42.9%

Table 31: Performance on 126,292 photostat images containing 1,033 covenants. Snippet recall is the product of full-text search recall (R_{FTS}) and LLM recall given the snippet text generated with the full-text search match ($R_{\text{LLM|FTS}}$). Character error rate (CER) is calculated on a subset of 3,137 images for which we have human transcribed text.

OCR	$\overline{\text{CER}}_{\text{w}}$	LLM	Cost Type		Precision		False Negatives				Recall			
			LLM	OCR	Whole Text	Snippet	Whole Text	Snippet			Whole Text	Snippet		
								FTS	LLM | FTS	Total		FTS	LLM | FTS	Total
		Gemini 2.5 Flash	Per Image		77.8%						96.5%			
Textract: Nov '21	14.1%	Claude Sonnet 3.7	Per Token	Per Image	77.0%	96.7%	43	28	55	83	95.8%	97.3%	94.5%	92.0%
Textract: Dec '25	10.9%	Claude Sonnet 4.5	Per Token	Per Image	97.1%	99.5%	76	68	44	112	92.6%	93.4%	95.4%	89.2%
Textract: Nov '21	14.1%	Stanford mistral-rrc	GPU Time	Per Image	96.7%	92.1%	143	28	76	104	86.2%	97.3%	92.4%	89.9%
Textract: Dec '25	10.9%	Claude Sonnet 3.7	Per Token	Per Image		98.2%		68	50	118		93.4%	94.8%	88.6%
olmOCR, Single Pages	8.0%	Claude Sonnet 4.5	Per Token	GPU Time	100.0%	99.1%	121	62	69	131	88.3%	94.0%	92.9%	87.3%
Textract: Dec '25	10.9%	Stanford mistral-rrc	GPU Time	Per Image	94.8%	95.9%	131	68	54	122	87.3%	93.4%	94.4%	88.2%
Textract: Nov '21	14.1%	gpt-oss-20b	GPU Time	Per Image	75.9%	76.1%	131	28	122	150	87.3%	97.3%	87.9%	85.5%
Textract: Dec '25	10.9%	gpt-oss-120b	GPU Time	Per Image	92.8%	95.8%	135	68	70	138	86.9%	93.4%	92.7%	86.6%
olmOCR, Single Pages	8.0%	gpt-oss-20b	GPU Time	GPU Time	81.2%	80.8%	136	62	92	154	86.8%	94.0%	90.5%	85.1%
Textract: Nov '21	14.1%	gpt-oss-120b	GPU Time	Per Image	89.1%	89.6%	136	28	110	138	86.8%	97.3%	89.1%	86.6%
Textract: Dec '25	10.9%	gpt-oss-20b	GPU Time	Per Image	80.0%	83.0%	140	68	79	147	86.4%	93.4%	91.8%	85.8%
olmOCR, Single Pages	8.0%	gpt-oss-120b	GPU Time	GPU Time	95.4%	96.0%	141	62	85	147	86.4%	94.0%	91.2%	85.8%
docTR: resnet+vgg16	28.9%	Claude Sonnet 4.5	Per Token	GPU Time		97.6%		121	43	164		88.3%	95.3%	84.1%
docTR: resnet+vgg16	28.9%	Claude Sonnet 3.7	Per Token	GPU Time		94.8%		121	49	170		88.3%	94.6%	83.5%
docTR: resnet+vgg16	28.9%	Stanford mistral-rrc	GPU Time	GPU Time	85.9%	88.0%	280	121	103	224	72.9%	88.3%	88.7%	78.3%
docTR: resnet+vgg16	28.9%	gpt-oss-120b	GPU Time	GPU Time	89.8%	91.8%	234	121	124	245	77.3%	88.3%	86.4%	76.3%
docTR: resnet+vgg16	28.9%	gpt-oss-20b	GPU Time	GPU Time	67.4%	73.0%	239	121	138	259	76.9%	88.3%	84.9%	74.9%

Table 32: Performance on 130,864 typewritten images containing 5,990 covenants. Snippet recall is the product of full-text search recall (R_{FTS}) and LLM recall given the snippet text generated with the full-text search match ($R_{\text{LLM|FTS}}$). Character error rate (CER) is calculated on a subset of 12,165 images for which we have human transcribed text.

OCR	$\overline{\text{CER}}_{\text{w}}$	LLM	Cost Type		Precision		False Negatives				Recall			
			LLM	OCR	Whole Text	Snippet	Whole Text	Snippet			Whole Text	Snippet		
								FTS	LLM | FTS	Total		FTS	LLM | FTS	Total
		Gemini 2.5 Flash	Per Image		98.4%						97.3%			
Textract: Dec '25	5.1%	Claude Sonnet 4.5	Per Token	Per Image	99.6%	99.9%	128	113	86	199	97.9%	98.2%	98.6%	96.8%
Textract: Dec '25	5.1%	Claude Sonnet 3.7	Per Token	Per Image		99.9%		113	103	216		98.2%	98.3%	96.6%
olmOCR, Single Pages	3.3%	Claude Sonnet 4.5	Per Token	GPU Time	100.0%	99.8%	222	137	174	311	96.3%	97.8%	97.2%	95.0%
Textract: Nov '21	11.5%	Claude Sonnet 3.7	Per Token	Per Image	94.5%	99.7%	250	143	238	381	95.8%	97.7%	96.1%	93.9%
Textract: Dec '25	5.1%	gpt-oss-120b	GPU Time	Per Image	99.2%	99.6%	325	113	238	351	94.6%	98.2%	96.1%	94.4%
Textract: Dec '25	5.1%	Stanford mistral-rrc	GPU Time	Per Image	99.6%	99.5%	493	113	229	342	91.8%	98.2%	96.3%	94.5%
olmOCR, Single Pages	3.3%	gpt-oss-20b	GPU Time	GPU Time	99.2%	99.0%	363	137	325	462	93.9%	97.8%	94.7%	92.6%
olmOCR, Single Pages	3.3%	gpt-oss-120b	GPU Time	GPU Time	99.6%	99.7%	380	137	291	428	93.7%	97.8%	95.3%	93.2%
Textract: Dec '25	5.1%	gpt-oss-20b	GPU Time	Per Image	98.3%	98.9%	409	113	290	403	93.2%	98.2%	95.3%	93.6%
Textract: Nov '21	11.5%	Stanford mistral-rrc	GPU Time	Per Image	99.5%	98.9%	1,164	143	616	759	80.6%	97.7%	90.0%	87.9%
docTR: resnet+vgg16	18.5%	Claude Sonnet 3.7	Per Token	GPU Time		99.7%		328	155	483		94.8%	97.4%	87.8%
docTR: resnet+vgg16	18.5%	Claude Sonnet 4.5	Per Token	GPU Time		99.7%		328	161	489		94.8%	97.3%	87.7%
Textract: Nov '21	11.5%	gpt-oss-120b	GPU Time	Per Image	98.2%	98.8%	786	143	656	799	86.9%	97.7%	89.3%	87.3%
Textract: Nov '21	11.5%	gpt-oss-20b	GPU Time	Per Image	97.2%	97.3%	926	143	716	859	84.5%	97.7%	88.3%	86.3%
docTR: resnet+vgg16	18.5%	gpt-oss-120b	GPU Time	GPU Time	97.9%	99.0%	1,215	328	785	1,113	79.7%	94.8%	86.8%	77.7%
docTR: resnet+vgg16	18.5%	gpt-oss-20b	GPU Time	GPU Time	94.8%	97.4%	1,515	328	689	1,017	74.7%	94.8%	88.4%	79.2%
docTR: resnet+vgg16	18.5%	Stanford mistral-rrc	GPU Time	GPU Time	99.6%	98.8%	1,954	328	869	1,197	67.4%	94.8%	85.4%	76.4%

I Comparison of Textract: 2021 and 2025

We initially processed our corpus of deed images with Textract in November 2021. To ensure our analysis reflected the (near) current state of the art, we reprocessed a large subset of these images with Textract in December 2025. While not central to our main results, this allows us to report on the improvement in Textract performance between 2021 and 2025.

First, we compare the character error rate (CER). Table 33 reports the average CER across 12,357 images with available human transcriptions, weighted by transcription length. Figure 19 plots per-image CER values for Textract 2021 against Textract 2025. As measured by CER, Textract 2025 outperforms 2021 with one exception. For photostat documents, the 95th-percentile character error rate—representing the worst 5% of performance—is notably better for Textract 2021 than Textract 2025 (60.1% vs. 71.4%, see Table 33). This is seen in Figure 19, with the clustering of images along $y \approx 5\%$ and in the bottom right corner.

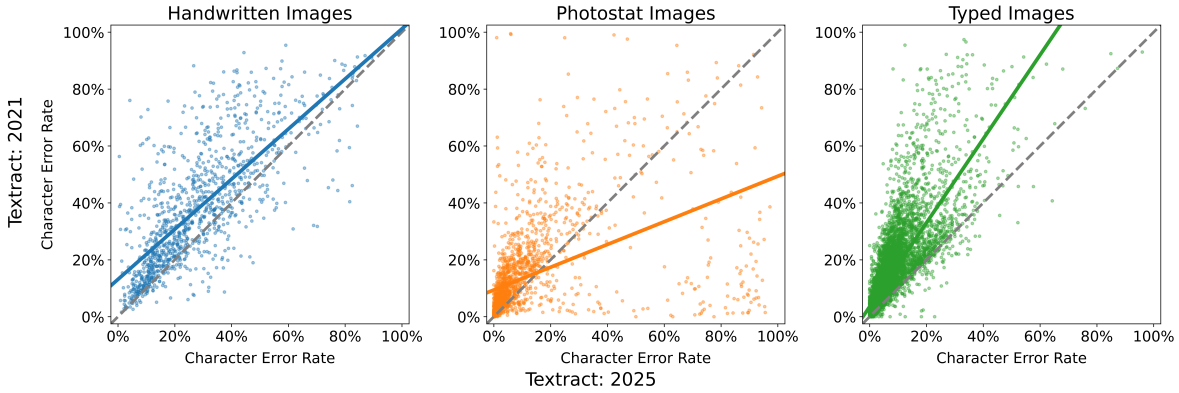


Figure 19: Textract 2021 image CER vs. Textract 2025 image CER for 12,357 images with human transcriptions available. The gray dashed line represents one-to-one correspondence of CER between models. Points above the one-to-one line indicate better performance by Textract 2025. The solid line is a simple linear regression. Handwritten images are plotted in blue, photostat images in orange, and typed images in green. Overall, Textract 2025 performs much better than 2021. We note that there is a subset of photostat images (clustered along $y \approx 5\%$ and in the bottom right corner) for which Textract 2021 significantly outperforms 2025.

Next, we present two additional comparisons that follow from the analysis in the main body of this paper. Table 33 quantifies full-text search recall (R_{FTS}) and precision (P_{FTS}) on Textract 2021 and Textract 2025 output for the core sample of 279,131 images described in §4.2.

Table 33 shows that fuzzy search using the terms described in Appendix E achieves higher recall (R_{FTS}) on Textract 2025 output than on 2021 output for both handwritten and typewritten documents. This improvement is most marked for handwritten documents. Much worse CER at the 95th percentile (CER_{P95}) for Textract 2025 partially explains why 2025 recall is worse for photostats. Precision (P_{FTS}) improves across all document types. Notably, the incremental benefit of fuzzy fuzzy search is smaller for Textract 2025 output. That said, this comparison favors 2021, since our fuzzy search terms were selected using 2021 Textract output.

Table 33: Comparison of Textract as of November 2021 vs. December 2025. Fuzzy fuzzy search numbers are *incremental* relative to the full-text search (FTS) matches rows. For example, FTS Recall+Fuzzy Fuzzy Recall=Total Recall. Average weighted character error rate, $\overline{CER}_w$, is calculated across 35,635 transcriptions that come from a subset of images with human transcription available.

	Handwritten		Photostat		Typed	
	2021	2025	2021	2025	2021	2025
Images	17,030	17,030	126,292	126,292	130,864	130,864
with cov	617	617	1,033	1,033	5,990	5,990
$\overline{CER}_w$	38.0%	28.1%	14.0%	10.9%	11.6%	5.1%
CER_{P95}	83.3%	71.7%	60.1%	71.4%	43.8%	27.8%
FTS Matches	3,713	2,286	49,449	44,032	51,313	44,350
with cov	529	569	1,001	959	5,779	5,850
FTS Precision	14.2%	24.9%	2.0%	2.2%	11.3%	13.2%
FTS Recall	85.7%	92.2%	96.9%	92.8%	96.5%	97.7%
Fuzzy Fuzzy Matches	6,310	1,246	20,033	2,586	22,400	4,203
with cov	60	16	4	6	78	40
Fuzzy Fuzzy Precision	1.0%	1.3%	0.0%	0.2%	0.3%	1.0%
Fuzzy Fuzzy Recall	9.7%	2.6%	0.4%	0.6%	1.3%	0.7%
Total Matches	10,023	3,532	69,482	46,618	73,713	48,553
with cov	589	585	1,005	965	5,857	5,890
Total Precision (P_{FTS})	5.9%	23.5%	1.4%	18.7%	7.9%	70.1%
Total Recall (R_{FTS})	95.5%	94.8%	97.3%	93.4%	97.8%	98.3%
F1_{FTS}	11.1%	37.6%	2.9%	31.1%	14.7%	81.8%

Overall, recall with Textract 2025 substantially outperforms Textract 2021 on typewritten documents, performs modestly better on handwritten documents, and performs comparably or slightly worse on photostat documents. Precision improves substantially across all document types. F1 scores are consistently higher, with the largest gains observed for typewritten documents.

J Covenanted Deeds vs. Construction

In §5.7, Figure 6 compares the number of covenanted deeds to the total number of deeds recorded each year from 1910 to 1965. We use total deeds recorded as a proxy measure for the volume of activity of the underlying process generating covenanted deeds. A better proxy would be residential recording activity, but we do not, even approximately, have a way to measure this.

As discussed in §5.7, racial covenants are not always explicitly restated each time a covenanted property changes hands. Typically, a covenant is explicitly mentioned when a buyer first purchases a newly built property from a developer. This suggests that the number of residential buildings completed might better approximate the underlying process. However, this measure has its own problems: covenants are sometimes recorded on undivided land well before construction, and the mix of covenants recorded on new builds, vacant land, and previously constructed properties shifts considerably over time.

Given these trade-offs, we consider total deed recording volume the most suitable denominator for a year-by-year covenant rate. That said, as a supplement, in Figure 20, we present a comparison of the volume of covenanted deeds relative to the volume of housing units started.

We use construction activity reports from the *Philadelphia Real Estate Record and Builders' Guide* for data covering 1910 through 1939 (The Athenaeum of Philadelphia) and building permit authorization data from the Census covering 1921 through 1964 (Bureau of the Census 1976). The Real Estate Record data claims to measure unit completions, while the Census data measures unit authorizations. Empirically, the *Builders' Guide* data leads the covenanted deed series by one to two years, a relationship consistent with starts or authorizations rather than completions. Further, where the data overlap, the timing and annual number of units are nearly identical. We therefore adopt the Census's well-documented label of "housing units authorized." For the overlapping period (1921–1939), we average the two series, although taking either alone makes little difference.

From this perspective, compared to Figure 6, the 1910-1911 peak in covenant recording looks less elevated, with the 1924-1929 and especially 1939-1940 periods looking relatively larger.

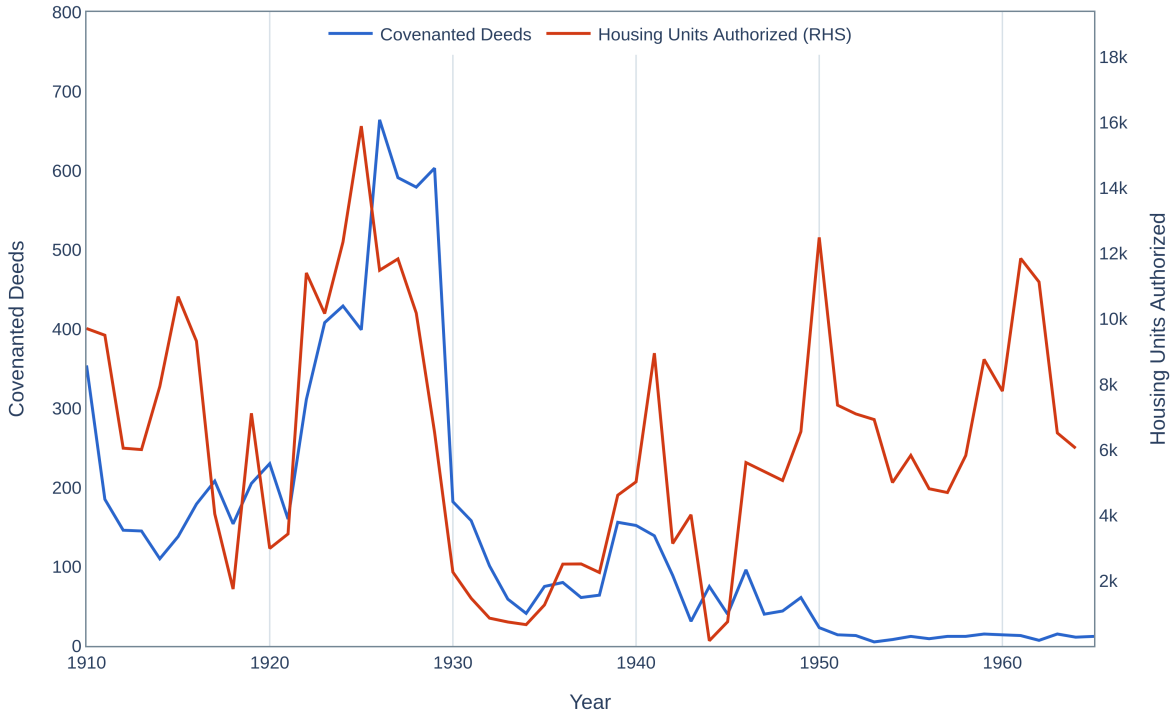


Figure 20: Deeds with racially restrictive covenants, 1910–1965.