

[Note: This is an early draft; any comments are greatly appreciated.]

How to Use Large Language Models for Empirical Legal Research

by

Jonathan H. Choi*

1 Introduction

Legal scholars have long annotated cases by hand to summarize and learn about developments in jurisprudence. Dramatic recent improvements in the performance of large language models (LLMs) now provide a potential alternative. This Article demonstrates how to use LLMs to analyze legal documents. It evaluates best practices and suggests both the uses and potential limitations of LLMs in empirical legal research. In a simple classification task involving Supreme Court opinions, it finds that GPT-4 performs approximately as well as human coders and significantly better than a variety of prior-generation NLP classifiers, with no improvement from supervised training, fine-tuning, or specialized prompting.

2 Background and Existing Literature

LLMs are AI models that generate probability distributions over token sequences based on artificial neural networks. Modern large language models are typically created using a “transformer” architecture (Vaswani et al. 2017) trained using “reinforcement learning from human feedback” (Christiano et al. 2017). For example, OpenAI’s GPT family of models begins with an “autoregressive” transformer, meaning that it takes a body of text and predicts the next token. Given the phrase “Let’s go for a,” the model might predict that the next word is “walk” with 20% probability, “drive” with 15% probability, and so on. This model is then further trained based on feedback from human coders to make them more conversational, truthful, and inoffensive. Although LLMs were initially trained as general-purpose tools, they have been found useful in a variety of different domains. Instruction-tuned LLMs like GPT-3.5 and GPT-4 are trained as conversational partners, where the user provides a “prompt” and the model provides an appropriate “response.”

* Jonathan H. Choi: University of Southern California Gould School of Law, Los Angeles, CA. Thanks to Nicholas Bednar, Jed Stiglitz, Elissa Redmiles, and the participants at the Max Planck Institute’s Seminar on Machine Learning and the Law. For collecting the original data that I re-analyzed in this study, I thank Nina A. Mendelson and her team of research assistants: Brian Apel, Lauren Babst, Michael Bloem, Erin Chapman, Joe Celentino, Lauren Dayton, Mariel Eben, Jarrett Gross, Simmon Kim, Claire Lally, J. Parker Lee, Collin Metcalf, Emma Notis-McConarty, Julie Siegel, and Anne Yearwood.

Interest in LLMs and their applications for social science research has leapt since the introduction of ChatGPT in late 2022. Several research papers have already suggested using LLMs to annotate natural language (“coding” it, in the traditional parlance of social science)—for example, Gilardi, Alizadeh and Kubli (2023) and Tornberg (2023) find that ChatGPT outperforms crowd workers for basic annotation tasks, like having MTurk workers identify political affiliation and topic in Twitter tweets. Zhu et al. (2023) similarly find that ChatGPT performs well at a variety of benchmark tasks, like sentiment analysis and bot detection. However, these past studies have focused on simple and general tasks, completed by humans with no special training on short passages of text. In contrast, this study focuses specifically on empirical legal research. Rather than short texts (like Twitter tweets), it considers long and complex legal cases and classification tasks of much greater subtlety. In addition, this Article benchmarks the performance of LLMs not against MTurk workers, but against law students specifically trained to conduct classification.

The status quo convention in empirical legal research is for professors to hire research assistants (RAs) to analyze large quantities of legal text. However, RAs present a variety of complications. RAs are logistically complicated to hire and to supervise, they are expensive, they take a long time to complete their work, and they operate non-deterministically, such that different RAs (or even the same RA over multiple sittings) might produce different results. Thus a tool that performed just as well as human RAs might be very appealing to empirical legal scholars from the perspective of cost, convenience, and efficiency.

3 *Data and Methods*

3.1 *Data and Sample Task*

As a vignette to evaluate the abilities of LLMs, I consider a classic task for human RAs: identifying the use of judicial interpretive techniques in court opinions. Specifically, I borrow a dataset produced by Mendelson (2018), for which Mendelson hired RAs to manually identify specific canons of construction in Supreme Court opinions produced over a ten-year period. A canon of construction is a rule of thumb that a court uses when interpreting legal text. The RAs were tasked with labeling the specific canons of construction contained in the Supreme Court opinions. They identified each opinion as either containing each canon or not containing it.

For this study, I focused on a specific canon, the “rule against surplusage.”¹ The description of the rule against surplusage that I provided to the LLMs when appropriate, copied nearly verbatim from the description provided by Mendelson to her RAs, was:

¹ Cost constraints prevented me from testing the full array of canons that Mendelson addressed in her original paper.

The rule against surplusage states that we should, if possible, give every word or phrase effect. Under the rule against surplusage, no word or phrase should be ignored or needlessly given an interpretation that causes it to duplicate another or have no consequence. In other words, 'avoid a reading that renders some words altogether redundant.' For example: 18 U.S.C. § 2314 punishes anyone who 'with unlawful or fraudulent intent, transports in interstate . . . commerce any falsely made, forged, altered or counterfeited securities . . ., knowing the same to have been falsely made, forged, altered or counterfeited.' The vehicle titles in question were obviously not 'forged, altered or counterfeited,' and the defendant argued that they were not 'falsely made' either because the people who made them believed them to be accurate. The dissent (Scalia) rejected this argument because it would render 'falsely made' redundant with 'forged' and 'counterfeited.'

In Mendelson's study, the rule against surplusage was one of the most important and most frequently cited canons of construction. Her RAs found at least one instance of the rule against surplusage in 123 out of 359 cases analyzed. Crucially, Mendelson instructed her RAs not only to identify named usages of the rule against surplusage (a process that could be duplicated by simple word searches), but also to identify passages in judicial opinions that reflected the underlying *logic* of the rule against surplusage. For example, the following passage would be considered an instance where the rule against surplusage was essentially applied even though it was not named:

Under the Court's view, by contrast, the "reduction" clause merely "underscores the Secretary's discretion to allocate funds among tribes." Ante, at 196. There is, however, no reason to suppose that Congress enacted the provision simply to confirm this "ordinary" rule. Ibid. We generally try to avoid reading statutes to be so "insignificant." *TRW Inc. v. Andrews*, 534 U. S. 19, 31 (2001) (internal quotation marks omitted).²

Mirroring the task set out by Mendelson, this study considers how LLMs can conduct the same analysis as those RAs, namely correctly spotting the rule against surplusage in the Supreme Court opinions.

3.2 Prompts and Model Specifications

All analysis was conducted using OpenAI's APIs. The standard prompt that I used to analyze the rule against surplusage follows. In order to conduct exploratory analysis, I asked the model not only to answer whether it felt that the rule against surplusage appeared, but also the specific text in which it appeared and why. This

² *Salazar v. Ramah Navajo Chapter*, 132 S. Ct. 2181.

provides some model interpretability and allowed me to check whether the LLM's reasoning was valid.

You are attempting to identify whether a passage from a Supreme Court opinion includes an application of the 'rule against surplusage', a canon of construction used by judges. <canon description>

Here is the passage from a Supreme Court opinion:

<passage>

Identify whether the rule against surplusage appears in the passage from the Supreme Court opinion. Begin your answer with either 'yes' or 'no'. If the answer is yes, indicate whether the rule against surplusage was used with approval or mentioned but not applied, and quote the specific sentences where the canon appeared. If the answer is no, do not provide further explanation.

Note that the baseline prompt that I used asked the LLM to provide a yes or no answer *before* providing its reasoning. This is the opposite of chain-of-thought prompting (described below). It is possible that the model alters its responses in some way in anticipation that it will be asked to explain its reasoning, which would be an interesting question for future research.

It is possible to specify the format of LLM outputs to make them easier to process. The most common format used is JSON, which GPT-4 (and to a lesser extent GPT-3.5-turbo) can reliably provide as output. However, I found that asking for JSON output substantially degraded the quality of the LLM's analysis in exploratory analysis using GPT-3.5-turbo. To be safe, I asked for answers in well-formatted plain English, and the model almost always obliged (apart from a dozen or so responses out of thousands provided by the GPT-3.5-turbo model, which I processed by hand).

GPT's APIs allow three types of prompts: "system" prompts (specifying the type of behavior desired from the system), "user" prompts (describing specific tasks for the model to complete) and "assistant" prompts (specifying past responses provided by the model, allowing us to conduct iterative conversations through the API, as required for chain-of-thought prompting). Where a system prompt was required, I used the following simple prompt:

You are an experienced law professor evaluating court cases.

Finally, GPT's API requires the user to specify the "temperature" of the model. A higher temperature will cause the model's outputs to evince greater stochastic variation, roughly causing the outputs to be less predictable but more "creative" and prone to hallucinations. For this study, I used a temperature setting of 0, as recommended by OpenAI for classification tasks and to improve reproducibility.

3.3 Input Token Length

Each GPT model has a different token limit: roughly 2000 tokens for GPT-3, 4000 tokens for GPT-3.5-turbo, and 8000 tokens for GPT-4 8K.³ As a result, longer opinions often must be "chunked," reduced to bite-sized pieces that are small enough for GPT-4 to process. For example, a 10,000-token opinion too long for GPT-4's 8000-token limit could be split into two 5000-token chunks.⁴ I chunked opinions for this study using a variety of chunk lengths. When the token limit for a particular chunk ended in the middle of a paragraph, I included the entire paragraph in the chunk before moving to the next chunk.

Tokens are not the same as words—for example, punctuation symbols, like "." or ",", count as tokens, and a single word may be tokenized based on parts of speech, which significantly increases the token count of court opinions above their word count. (To take a random example, the opinions in the recent Supreme Court case of *Wilkins v. United States*, 598 U.S. ____ (2023), were collectively 3044 words long, but 4731 tokens long according to OpenAI's tokenizer.) When any chunk was flagged as containing an instance of the rule against surplusage, I labeled the entire opinion as containing the rule.

In addition to chunking's being technologically necessary, it might be *desirable* because LLMs perform better when given shorter passages to process. Analogous to humans, the ability of the models to hold relevant information in something akin to working memory degrades when the models are asked to consider longer pieces of text. As a result, in cases where text can be analyzed piece by piece (as is generally true for the analysis in this paper), asking GPT-4 to analyze shorter passages may produce better results.

The tradeoff is that it is more expensive and more time-consuming: more expensive because the ratio of tokens spent describing the desired task over the tokens used to present actual legal documents will be lower the longer the passage used, and more time-consuming because API calls to OpenAI take some amount of overhead to process. While paragraph-by-paragraph analysis produced good results, it was sufficiently cumbersome (primarily in the time that it took) that it was impracticable

³ I used the March 14, 2023 version of GPT-4 (gpt-4-0314).

⁴ Because token length must also include the total length of the task description and any back-and-forth, the length of the passage I was able to analyze was typically much shorter than the total token length—for example, the maximum token length I was able to analyze was 1300 tokens for GPT-3-fine-tune, 3000 tokens for GPT-3.5-turbo, and 7450 tokens for GPT-4.

at scale. In the end, I experimented with chunk lengths from 500 to roughly 8000 tokens.

3.4 Chain-of-Thought Prompting

Another technique frequently used in the literature to improve the performance of LLMs is “chain-of-thought” (CoT) prompting. CoT prompting has several variants, but one promising version (Wei et al. 2023) involves prompting the LLM to engage in step-by-step analysis before providing an answer. For this study, I initially prompted the LLM to think step by step, and then asked it to provide a conclusion based on its initial statement. Unlike baseline prompting and few-shot prompting, chain-of-thought prompting required a back-and-forth where the LLM’s prior response was re-submitted to solicit a further response. In the example below, notice how the highlighted language specifically invites the LLM to begin reasoning step by step, by drafting the beginning of the LLM’s reply in advance.

User Prompt:

Q: You are attempting to identify whether a passage from a Supreme Court opinion includes an application of the 'rule against surplusage', a canon of construction used by judges.
<canon description>

Here is the passage from a Supreme Court opinion:

<passage>

Identify whether the rule against surplusage appears in the passage from the Supreme Court opinion, whether it was used with approval or mentioned but not applied.

A: Let's think step by step.

LLM Response:

In the passage provided, the Supreme Court justices are engaged in statutory interpretation, specifically looking at how the preemption clause of 49 U.S.C. § 14501(c)(1) relates to state laws and the specific case in question. While the passage does focus on the interpretation of words and phrases in the law, it does not appear to specifically apply the rule against surplusage as it's traditionally understood.

One place where the rule against surplusage might seem relevant is in the Court's interpretation of the phrase "with respect to the transportation of property." The Court notes that this phrase "massively limits the scope of preemption" under the FAAAA, suggesting that every word or phrase in the statute is being given effect and none are redundant or meaningless. This analysis could be seen as generally in line with the rule against surplusage.

However, the justices are not explicitly addressing the rule against surplusage. The rule would imply that every word in the statute should be given separate meaning and should not duplicate another word's meaning. The Court does not express any concern about such duplication or redundancy, nor does it seem to use the rule as a guiding principle in its analysis.

Therefore, it can be said that the rule against surplusage was not explicitly mentioned or applied in this passage. If it was implicitly applied at all, it was done without any clear or direct reference to the rule.

After causing the LLM to engage in more involved step-by-step reasoning, we conclude with an additional prompt asking the LLM to summarize its thoughts in a manner more conducive to systematic analysis.

User Prompt:

Based on your prior response, determine whether the rule against surplusage was either used with approval or mentioned but not applied. Begin your answer with either 'yes' or 'no'. If the answer is yes, quote the specific sentences where the canon appeared. If the answer is no, do not provide further explanation.

LLM Response:

No.

3.5 Few-Shot Prompting

Few-shot prompting entails providing the model with examples of inputs and ideal outputs to help it understand how it is supposed to perform. While this method does not require an iterative conversation with the model, it asks the model to pick up halfway through a hypothetical conversation. I crafted the few-shot prompt by editing good responses provided by GPT-4. Here is the standard few-shot prompt that I used (note that although the prompts are presented as a sequential conversation, they are all submitted to the model at once):

User Prompt:

You are attempting to identify whether a passage from a Supreme Court opinion includes an application of the 'rule against surplusage', a canon of construction used by judges.
<canon description>

Here is the passage from a Supreme Court opinion:

<passage>

Identify whether the rule against surplusage appears in the passage from the Supreme Court opinion. Begin your answer with either 'yes' or 'no'. If the answer is yes, indicate whether the rule against surplusage was used with approval or mentioned but not applied, and quote the specific sentences where the canon appeared. If the answer is no, do not provide further explanation.

Assistant Prompt:

No.

User Prompt:

<Same as above, with a different passage>

Assistant Prompt:

Yes, the rule against surplusage appears in the passage from the Supreme Court opinion. The rule is used with approval. The sentences where the canon appears are: "But this reading would render the regulation entirely superfluous... On the dissent's reading, §402.03's reference to 'discretionary' federal involvement is mere surplusage, and we have cautioned against reading a text in a way that makes part of it redundant. See, e.g., *TRW Inc. v. Andrews*, 534 U. S. 19, 31 (2001)."

User Prompt:

<Same as above, with a different passage>

3.6 *Fine-Tuning*

All the models described above use off-the-shelf LLMs, either GPT-3.5-turbo or GPT-4. However, an alternative method—and the most common method prior to the advent of high-performance general-use LLMs late last year—was supervised learning, wherein a model would be trained specifically for a particular classification task.

Supervised classification methods abound, many of which use more traditional NLP techniques. It is also possible to fine-tune an LLM by taking a base model and updating the last layers of its neural network to adapt them to produce a particular kind of output. Thus, unlike the methods mentioned above, fine-tuning requires us to alter the actual internal weights of the model. To fine-tune the last layers of the model, we apply the same methods originally used to train the model, but specifically on a training set consisting of prompts and desired completions. The model is fine-tuned by engaging in gradient descent so that the objective function is optimized with respect to the training set.

To fine-tune the model, I generated 542 prompt-completion pairs. For fine-tuning, the prompts consisted solely of the passages to be classified, and the completions consisted solely of the word “Yes” or “No” (excluding separator sequences and stop sequences included for technical reasons). OpenAI asserts that “you should provide at least a few hundred high-quality examples, ideally vetted by human experts. From there, performance tends to linearly increase with every doubling of the number of examples.” I followed the instructions provided by OpenAI⁵ for classification in assembling the training set. I then used the trained model to classify a test set consisting of opinions that did or did not include the rule against surplusage.

4 Results

To gauge the performance of LLMs at a legal classification test, I tested each combination of parameters (model, token length, CoT, and/or few shot) on a standard set of 200 cases, roughly evenly balanced between cases where the rule against surplusage was present and where it was not. Mendelson’s original dataset included 123 cases where RAs had positively identified the rule against surplusage; I added randomly selected cases where they had not identified the rule, anticipating that some of these cases would turn out to be false negatives, bringing the true positives and true negatives roughly in line.

The primary performance metric I use in this article is F_1 score, which is the harmonic mean of precision and recall. I also report precision (also known as positive predictive value, true positives over true positives plus false positives), recall (also known as sensitivity, true positives over true positives plus false negatives), and accuracy.

To set a baseline for the performance of the GPT models, I first tested the performance of commonly used prior-generation NLP vectorizers and classifiers. For classification, I used random forest, logistic regression, multinomial naïve Bayes, and linear support vector machine models. For vectorization, I used count (specifically for multinomial naïve Bayes, which is natively better suited to count vectorization), term frequency-inverse document frequency (TF-IDF), and Doc2Vec. Doc2Vec (Le & Mikolov 2014) is especially relevant because it employs a document embedding model that was a progenitor to transformer models. Table 1 provides performance statistics for various combinations of vectorizer and classifier.⁶

⁵ This included using the gpt-3-curie model rather than the more advanced gpt-4-davinci model, which performs similarly well at roughly a tenth the cost.

⁶ For each combination, I used the default classifier and vectorizer from Scikit-learn, except for the Doc2Vec vectorizer, which was from the Gensim package. For CountVectorizer and TfidfVectorizer, I set max_features to 5000 and ngram_range to (1, 3). For Doc2Vec, I set vector_size to 50 and min_count to 2, and I used the distributed bag-of-words method. I used a balanced sample of documents in which the rule against surplusage was and was not present, with a 0.75/0.25 train/test split.

Table 1: Accuracy, Precision, Recall, and F_1 Score Results

Classifier	Vectorizer	Accuracy	Precision	Recall	F_1 Score
Random Forest	TF-IDF	0.64	0.64	0.64	0.63
Logistic Regression	Doc2Vec	0.64	0.64	0.64	0.63
Linear Support Vector Machine	Doc2Vec	0.62	0.62	0.62	0.62
Random Forest	Doc2Vec	0.60	0.60	0.60	0.60
Multinomial Naïve Bayes	CountVectorizer	0.57	0.57	0.57	0.57
Linear Support Vector Machine	TF-IDF	0.55	0.55	0.55	0.55
Logistic Regression	TF-IDF	0.49	0.49	0.49	0.48

These older NLP techniques generally did not perform particularly well (likely because of the small size of the training and test sets), with a tie for the best performance between a random forest classifier with TF-IDF vectorization and a logistic regression classifier with Doc2Vec vectorization. I use the performance of the former as a benchmark for prior-generation NLP through the remainder of this Article.

Of course, the more salient question for our purposes is the comparison between LLMs and human coders. One conventional way to evaluate the performance of LLMs relative to human coders would be to have 2 separate human coders evaluate each opinion and then calculate intercoder validity between humans, comparing it to intercoder validity between the LLM and the humans. However, for present purposes, this analysis would have been impossible because Mendelson only had one RA evaluate each case.

Instead, I manually reviewed each case of disagreement between the GPT-4 runs and the human coders to evaluate the ground truth.⁷ I generally found that human coders did not suffer from false positives, meaning that cases where they identified the rule against surplusage did indeed contain the rule against surplusage, but in many cases human coding resulted in false negatives. This was possibly because the human coders relied on heuristics in coding (like the presence of the phrase “rule against surplusage” or the word “superfluous”) that caused them to overlook more complex edge cases.

⁷ Because GPT-4 provided much more reliable descriptions of the passages it felt reflected the rule against surplusage, I found it more useful to review only these responses and not the responses of the GPT-3.5-turbo models. Fine-tuned GPT-3 did not provide any explanation at all.

Table 2 below provides accuracy, precision, recall, and F_1 scores for different combinations of model, input token length, and prompting method. As it and the following tables show, increasing the token length of inputs generally increases precision but decreases recall. This is because the longer the prompt, the more prone the model becomes to false negatives and the less prone it becomes to false positives. This makes intuitive sense, because the model is more likely to miss something important buried in a long prompt; on the other hand, the model is more likely to be overeager and to identify canon use where there was none in a short prompt.

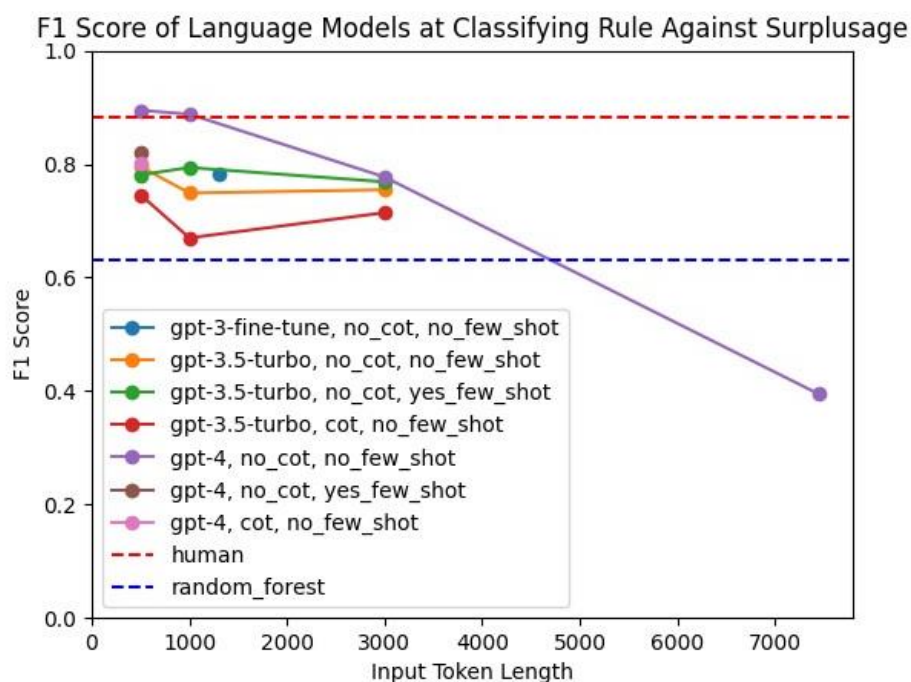
Qualitatively, the most common type of false positive occurred when the text almost but not exactly fit the description of the rule against surplusage. For example, GPT models sometimes identified as applications of the rule against surplusage situations where the Court simply emphasized the importance of the text, rather than emphasizing the need to give every word *independent* meaning.

Table 2: Accuracy, Precision, Recall, and F_1 Score Results

Classifier	Input Token Length	CoT	Few Shot	Accuracy	Precision	Recall	F_1 Score
<i>gpt-3-fine-tune</i>	1300			0.74	0.76	0.81	0.78
<i>gpt-3.5-turbo</i>	500			0.74	0.73	0.87	0.80
<i>gpt-3.5-turbo</i>	500		✓	0.70	0.69	0.91	0.78
<i>gpt-3.5-turbo</i>	500	✓		0.62	0.61	0.96	0.75
<i>gpt-3.5-turbo</i>	1000			0.72	0.78	0.72	0.75
<i>gpt-3.5-turbo</i>	1000		✓	0.73	0.72	0.88	0.79
<i>gpt-3.5-turbo</i>	1000	✓		0.66	0.77	0.59	0.67
<i>gpt-3.5-turbo</i>	3000			0.64	0.67	0.76	0.71
<i>gpt-3.5-turbo</i>	3000		✓	0.74	0.85	0.68	0.75
<i>gpt-3.5-turbo</i>	3000	✓		0.72	0.75	0.79	0.77
<i>gpt-4</i>	500			0.88	0.93	0.86	0.89
<i>gpt-4</i>	500		✓	0.86	0.85	0.79	0.82
<i>gpt-4</i>	500	✓		0.80	0.69	0.95	0.80
<i>gpt-4</i>	1000			0.88	0.94	0.84	0.89
<i>gpt-4</i>	3000			0.79	1.00	0.64	0.78
<i>gpt-4</i>	7450			0.56	1.00	0.25	0.39
<i>Random</i>				0.64	0.64	0.64	0.63
<i>Forest / TF-IDF</i>							
<i>Human</i>				0.88	1.00	0.79	0.88

Figure 1 below shows the relative F_1 score of different models, using different prompt-engineering techniques, at different input token lengths.

Figure 1



In Figures 2 and 3 below, the x -axis depicts whether CoT and few-shot prompting were used. CoT prompting systematically slightly decreases model performance; few-shot prompting generally also slightly decreases model performance, except for the GPT-3.5-turbo model with a 3000-token input length. Even then, using few-shot prompting only modestly improves performance.

Figure 2
F1 Score of Language Models at Classifying Rule Against Surplusage

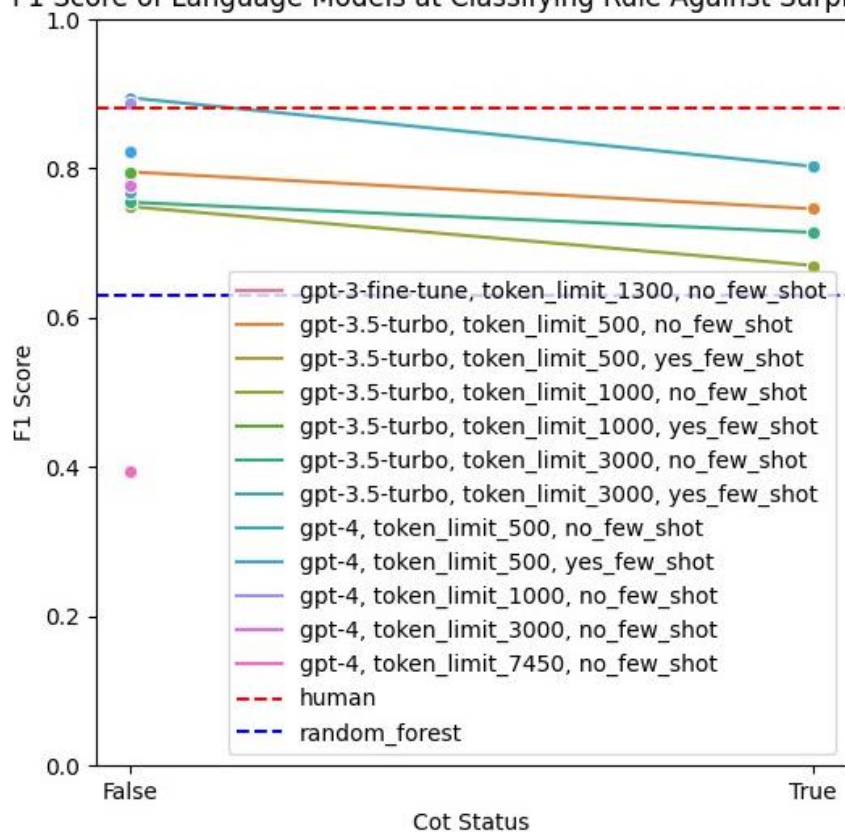
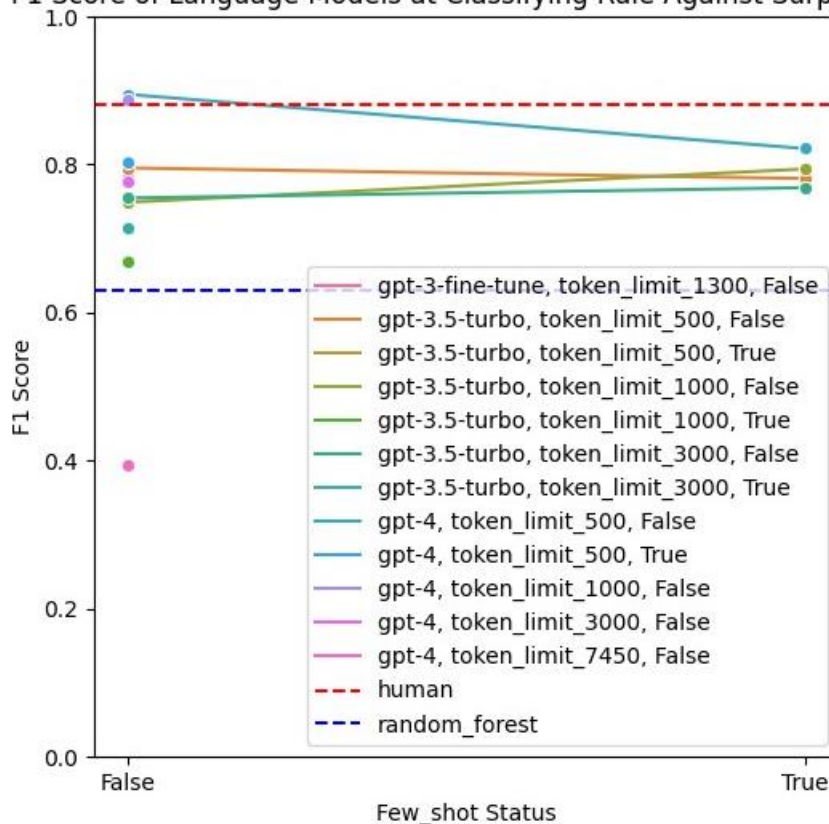


Figure 3
F1 Score of Language Models at Classifying Rule Against Surplusage



As Figures 1-3 show, there was considerable variation in the performance of GPT models at classification. However, almost all of the GPT models outperformed the best non-transformer NLP model, and the best GPT models even had performance comparable to human coders. The following Section discusses these findings in more detail.

4.7 Discussion

The prior section reveals a variety of other interesting results about the performance of LLMs at this vignette legal classification task.

First, as between different base models, GPT-4 consistently performs the best for a given set of parameters, as expected given that it is the latest and most advanced model. The GPT-3.5-turbo and GPT-3-fine-tune models demonstrate similar performance to each other at similar input token lengths. This suggests that a researcher should ideally use GPT-4, subject to the cost constraints discussed below. It also suggests that fine-tuning is not presently worth the trouble, as it does not result in performance gains and is relatively expensive to use—its cost is comparable to GPT-4, with worse performance.⁸ This may be because the GPT-3

⁸ It is possible that fine-tuning could outperform non-fine-tuned models if the size of the training set is increased. While I tested a training set size that I felt would be feasible for empirical scholars to

model available for fine-tuning has not received the additional instruction tuning that has improved the performance of the GPT-3.5-turbo model. Thus given the tradeoff between fine-tuning a worse model and zero-shot prompting a better model, the zero-shot approach currently wins.

Second, the results suggest that performance can substantially vary based on the length of the inputs. For GPT-3.5-turbo, performance was roughly consistent regardless of input token length. However, for GPT-4, which generally was the best-performing model, accuracy significantly degraded as token length increased, especially toward the limit of GPT-4's allowable token length. Indeed, I found that GPT-4, operating at its token limit, performed worse than GPT-3.5-turbo operating at its token limit.

This is an important finding in light of the considerable excitement that has surrounded increases in LLM token limits. Observers have speculated that increasing token limits (like Anthropic's recent introduction of a model with a 100,000-token limit) might allow models to process increasingly large and complex documents, analyzing lengthy legal opinions or textbooks in a single prompt. My work suggests reason to be wary of models with very large token limits. At least in the context of empirical legal research, it might be better to chunk long documents and analyze them piecemeal.

One important question for future research is whether this finding replicates to other domains, and if so, why it occurs. One explanation might be that my prompts include a standard instruction to the model at the beginning of the prompt and that the model "forgets" these instructions if the intervening quotation is too long. Researchers could use attention maps or other explainability tools to probe this issue further.

Third, I found that neither CoT nor few-shot reasoning improved model performance, and in fact both methods generally worsened performance. It should be noted that in both cases I only tested the variants of CoT and few-shot prompting that I believed were the most conventional and most likely to be useful for empirical legal research; it is possible that some other variation of these methods would produce better results. However, the fact that zero-shot off-the-shelf models had the best performance in this study is promising in terms of their ease of use for legal scholars. Zero-shot off-the-shelf models require relatively little effort other than drafting an initial prompt, and there is no minimum dataset size to use them (unlike fine-tuning).⁹

Because CoT and few-shot prompting have been found to succeed in other domains—indeed, I have personally found few-shot prompting to improve performance in other tasks involving questions about tax law, insurance law, and

generate, one important question for future research is how fine-tuned performance scales at empirical legal tasks with training set size.

⁹ On the other hand, human coding may be preferable until the test set reaches a sufficient size that human coding becomes impracticable.

American law in general (Ney et al. 2023; Choi & Schwarcz 2023)—we might wonder why CoT and few-shot prompting fail at simple classification tasks.

I hypothesize that because these prompt-engineering techniques are meant to induce domain-specific reasoning, they perform better at more complex tasks. In contrast, the simple classification exercise described in this study does not require much reasoning or orientation toward a particular mode of thinking, which might explain why elaborate prompting is unnecessary. By analogy, we can imagine that a human coder would also not improve their performance if they were asked to think step-by-step or if they were given more examples, in light of the simplicity of the task. This finding therefore contributes to an emerging literature suggesting that prompt engineering techniques are not one-size-fits-all.

In addition, it is an open secret among NLP scholars that OpenAI has trained its more recent models using reinforcement learning from human feedback (RLHF) that itself incorporates chain-of-thought prompting (i.e., by encouraging the model to engage in more reasoning before providing a final answer in the reinforcement learning step that produces the off-the-shelf GPT models). Thus it is possible that although CoT is a useful technique in general, it is no longer required when using the latest GPT LLMs.

Fourth and finally, I found that the best model—zero-shot GPT-4, with no special prompting or fine-tuning, using a relatively low token input length—performed at a level comparable to human coders, and that a simple and much cheaper model—GPT-3.5-turbo, with no special prompting or fine-tuning—performed at a level worse than human coders but which might nonetheless be acceptable. Virtually all the GPT models also outperformed all the prior-generation NLP models, possibly because the sample size was relatively small and the GPT models do not require training (with the exception of the fine-tuned GPT-3 model).

LLMs require no training, no paperwork, and no direct supervision; they work through the night and process documents much more quickly than humans, and at a much lower cost. Thus, if their performance can match human performance, it is conceivable that LLMs could replace human RAs in at least some empirical legal research tasks. Moreover, LLM performance could be augmented by subsequent human review. Because in my study false positives were more problematic than false negatives, these false positives could be reduced by having a first round of LLM analysis followed by a second round of human analysis.

A note of caution: for this study, I tested LLMs on a single, relatively straightforward legal task. The fact that the LLM was able to achieve performance comparable to human RAs on this task does not imply that the LLMs will be able to achieve similar performance to RAs on other tasks, and it especially does not suggest that LLMs can approximate human legal reasoning as a general matter. Evaluations of LLM performance are highly domain specific, and it is possible that they will perform worse even on superficially comparable future tasks.

4.8 Hallucinations

Qualitatively, chain-of-thought prompting can sometimes cause odd hallucinations when the LLM loses track of the text. CoT prompting in particular seemed to induce hallucinations: for example, it caused the model to misremember a description of the canon of construction (i.e. the example of a dissent by Justice Scalia quoted above) as part of the opinion text currently under analysis, and it sometimes entirely invented quotations that did not appear in the passage but that seemed to fit the canon under consideration.¹⁰ Notably, however, the hallucinations only occurred with the GPT-3.5-turbo model; in the sample of cases that I studied, GPT-4 produced no hallucinations.

4.9 Cost and Speed

A crucial question in comparing the use of GPT models with the use of traditional RAs will be an evaluation of the relative cost and speed of each. One key benefit of using GPT models is that they are highly scalable—whereas Mendelson (2018) was able to analyze only ten years of Supreme Court opinions, in theory computational analysis might allow us to evaluate all the Supreme Court opinions ever written. However, this benefit depends on the specific runtime and monetary cost of using GPT models. While the cost of GPT usage and the runtime of GPT models are constantly falling, I present here some very rough estimates accurate as of May 2023.

Black and Spriggs (2008) estimate that the Supreme Court has produced 26,715 opinions from 1971 to 2005, at an average length of 3828 words. Assuming a ratio of words to tokens of roughly 2:3, this suggests that all the Supreme Court opinions produced up to 2005 comprise 153,397,530 tokens. Based on OpenAI's current pricing of \$0.03 per 1000 prompt tokens in GPT-4, submitting as a prompt all Supreme Court opinions ever written would cost \$4602. The actual figure would be somewhat higher because it would include the description as part of the prompt as well as the length of the response, but this at least suggests an order of magnitude for total costs. GPT-3.5-turbo is significantly cheaper, \$0.002 per 1000 prompt tokens, resulting in a total cost of \$306 (again something of an underestimate but indicative of the order of magnitude).

The speed of GPT models is highly dependent on how overloaded OpenAI's servers are at any one time. However, with large token input lengths, rate limitations on API usage become a primary bottleneck, rather than the speed of API calls themselves. OpenAI currently imposes a 40,000-token-per-minute limitation on API access; operating at this speed would allow a GPT model to process all of the

¹⁰ For example, in the case of *Adoptive Couple v. Baby Girl*, the model (with neither CoT nor few-shot) imagined the phrase “the statutory language of § [1912(d)] indicates that congress intended the “active efforts” provision to be a separate requirement from the “beyond a reasonable doubt” provision, and we must give effect to that intent to avoid rendering either provision superfluous.” This language appears nowhere in the actual case.

Supreme Court opinions ever written in approximately 64 hours, slightly less than three days.

Thus, while GPT models are substantially cheaper and faster than human coders, there are still real costs to their use, and they cannot be scaled up indefinitely.

5 *Best Practices for LLM Use in Empirical Legal Research*

This paper represents an initial first step toward integrating LLMs into traditional legal research. While the field will likely develop norms and best practices in the months and years to come, what follows are some basic suggestions for how researchers should discipline their use of LLMs.

First, how should we validate model performance? One important problem occurs when researchers aggressively tweak prompts in order to obtain results that fail to validate out of sample. For example, Chowdhuri et al. (2023) found significant issues in a study of GPT's performance on the MIT computer science curriculum, most notably that the original study had mechanically tried new prompts for each question until the model produced the correct answer (which unsurprisingly led to an accuracy rate of 100%). While this is a particularly egregious example, the tendency to engineer prompts to perform well on a specific task is natural and often appropriate. After all, if an analogous real-world LLM user could also engineer prompts, a researcher who failed to do so could *underestimate* replicable LLM performance.

To avoid this problem, empirical legal scholars should be careful to engage in prompt engineering only with a small initial sample before applying prompts more broadly to a separate sample to evaluate performance. Best practices for out-of-sample validation apply here, including that the sample used for testing should only be used once, although the sample used for exploratory analysis can be used many times to tweak the prompt. In this study, I conducted only minimal prompt engineering, using only a couple of example cases which were then set aside before testing on the bulk of my dataset.

A second, related, issue of multiple comparisons is that scholars using LLMs to test a particular hypothesis have many degrees of freedom to tweak prompts and hyperparameters to obtain the answers they want. For example, suppose I (hypothetically) had a theory that the Supreme Court's use of the rule against surplusage has declined over time. I could analyze Supreme Court rulings using the variety of methods described above not merely to optimize the performance of my classifier, but also to select for results that validate my initial hypothesis. Fortunately, standard best research practices apply here: pre-registration can help to ameliorate researcher degrees of freedom, and scholars should be transparent about the models they select, especially if they choose models on any grounds other than performance.

Fourth and finally, LLM performance is currently insufficiently reliable and insufficiently predictable to be used without validating LLM results by comparison to human analysis. Thus researchers relying on LLMs will still need to enlist RAs or analyze text themselves in order to evaluate the reliability of their methods. While GPT models perform reasonably well on basic classification tasks, they may not be able to perform more complex legal judgments, which may still require hand-coding by RAs until LLM performance further improves. Thus before relying exclusively on LLMs, scholars should compare the performance of LLMs and humans and decide whether any decrease in coding accuracy is worth the speed and efficiency gains.

Relatedly, unexpected problems may arise when LLMs have a different balance of false positives and false negatives than human coders. For example, if LLMs are more prone to false positives than false negatives (as was true in my sample), any estimates derived from the LLM will be systematically biased. To take an extreme vignette, if LLMs never miscode false negatives but if 50% of identified positives are false positives, then the LLM's raw estimate of the frequency of the rule against surplusage will be systematically overstated by double. This problem affects human coders as well but may be more severe and opaque with LLMs. Independent review of LLM coding provides data that could be used to correct these biases, another reason why LLMs should be paired with human validation.¹¹

6 *Limitations and Future Work*

Because this work contributes to a fast-developing field, there are several limitations on the external validity of the findings in this article. First, the article focuses on a single canon of construction rather than testing a variety of canons. It is possible that other canons of construction could have unique features that would lead to different results. More broadly, the classification task that I studied was relatively straightforward. Many empirical legal studies involve more complicated judgments on which LLM performance could be completely different.

Second, due to budgetary constraints, I was not able to test every possible combination of parameters. Because it would have been exponentially expensive and time-consuming to test all possible combinations, I focused on those that I felt were most likely to be effective.

Interestingly, many of the potential issues with LLMs are also potential problems with RAs. LLMs and RAs can be sensitive to prompting, and their reasoning is not entirely transparent. That is, we can rely on their proffered explanations for their classifications, but we cannot know whether there is some underlying undisclosed heuristic (possibly an unconscious one) that actually determined their choices.

¹¹ I intend to propose a specific correction procedure for these biases in future co-authored work with Paul Connell. Our work will also address the presence of attenuation bias and suggest a method to correct for it when using an LLM.

The research does present an array of exciting questions and avenues for future investigators. Among these are the fine-tuning of additional models like BERT or LLaMA and the evaluation of an ensemble model's performance. The latter could involve running baseline, few-shot, and CoT models together, with each "voting" on the correct answer and the majority opinion being accepted as final. Future work might also consider the value of self-reflection, where a model reviews its own responses to determine their accuracy—a potentially effective method for reducing false positives, which as noted above are more problematic in my study than false negatives.

Future work could also involve testing canons other than the rule against surplusage, varying temperature, and calculating the Area Under the Receiver Operating Characteristics for a fine-tuned LLM. This could include testing various discrimination thresholds and assessing how the log probability ratios, which indicate the model's confidence in its answer, predict accuracy. Finally, it might be helpful to investigate whether there is a systematic difference in the types of false negatives and positives generated by LLMs compared to those produced by humans.

7 Bibliography

Ryan C. Black & James F. Spriggs II, *An Empirical Analysis of the Length of US Supreme Court Opinions*, *Houston Law Review* (2008)

Jonathan H. Choi & Daniel Schwarcz, *GPT-4 Goes to Law School* (2023)

Raunak Chowdhuri, Neil Deshmukh & David Koplow, *No, GPT4 can't ace MIT* (2023) <https://flower-nutria-41d.notion.site/No-GPT4-can-t-ace-MIT-b27e6796ab5a48368127a98216c76864>

Paul F. Christiano et al., *Deep Reinforcement Learning from Human Preferences*, in *Proceedings of the 31st Conference on Neural Information Processing Systems* 4302 (2017)

Fabrizio Gilardi, Meysam Alizadeh, Maël Kubli, *ChatGPT Outperforms Crowd-Workers for Text-Annotation Tasks* (2023)

Quoc Le & Tomas Mikolov, *Distributed Representations of Sentences and Documents*, *PROCEEDINGS OF THE 31ST INTERNATIONAL CONFERENCE ON MACHINE LEARNING* (2014)

Nina A. Mendelson, *Change, Creation, and Unpredictability in Statutory Interpretation: Interpretive Canon Use in the Roberts Court's First Decade*, *Michigan Law Review* (2018)

John J. Nay, David Karamardian, Sarah B. Lawsky, Wenting Tao, Meghana Bhat,

Raghav Jain, Aaron Travis Lee, Jonathan H. Choi & Jungo Kasai, Large Language Models as Tax Attorneys: A Case Study in Legal Capabilities Emergence, *PHILOSOPHICAL TRANSACTIONS OF THE ROYAL SOCIETY A: MATHEMATICAL, PHYSICAL AND ENGINEERING SCIENCES* (forthcoming 2023)

Petter Tornberg, ChatGPT-4 Outperforms Experts and Crowd Workers in Annotating Political Twitter Messages with Zero-Shot Learning (2023), <https://www.semanticscholar.org/paper/ChatGPT-4-Outperforms-Experts-and-Crowd-Workers-in-Tornberg/6354f2639d07bf8d5b08a4dcaef4c5db5fe19fdb>

Ashish Vaswani et al., *Attention Is All You Need*, in *Proceedings of the 31st Conference on Neural Information Processing Systems* 6000 (2017)

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, Denny Zhou, Chain-of-Thought Prompting Elicits Reasoning in Large Language Models (2023), <https://arxiv.org/abs/2201.11903>

Yiming Zhu, Peixian Zhang, E. Haq, Pan Hui, Gareth Tyson, Can ChatGPT Reproduce Human-Generated Labels? A Study of Social Computing Tasks (2023), <https://www.semanticscholar.org/paper/Can-ChatGPT-Reproduce-Human-Generated-Labels-A-of-Zhu-Zhang/2f45e71b2fddccf73430c8ecb1642b577a22610b>