Memory Is All You Need: Testing How Model Memory Affects LLM Performance in Annotation Tasks

Joan C. Timoneda*   Sebastián Vallejo Vera†

Abstract

Generative Large Language Models (LLMs) have shown promising results in text annotation using zero-shot and few-shot learning. Yet these approaches do not allow the model to retain information from previous annotations, making each response independent from the preceding ones. This raises the question of whether model memory—the LLM having knowledge about its own previous annotations in the same task—affects performance. In this article, using OpenAI's GPT-4o and Meta's Llama 3.1 on two political science datasets, we demonstrate that allowing the model to retain information about its own previous classifications, irrespective of their accuracy, yields significant performance improvements between 5 and 25% when compared to zero-shot and few-shot learning. Moreover, memory reinforcement, a novel approach we propose that combines model memory and reinforcement learning, yields additional performance gains in three out of our four tests. These findings show that memory-based approaches outperform few-shot learning in annotation tasks and have important implications for applied researchers looking to improve the performance and efficiency of LLMs.

---

*Assistant Professor, Department of Political Science, Purdue University
†Assistant Professor, Department of Political Science, University of Western Ontario

Introduction

Recent work in the social sciences explores the benefits and limitations of using Generative Large Language Models (LLMs) as text annotators, with promising results. LLMs often outperform human coders (Gilardi, Alizadeh and Kubli, 2023) and provide comparable accuracy when labeling political text (Overos et al., 2024; Ornstein, Blasingame and Truscott, 2023), even across multiple languages (Heseltine and Clemm von Hohenberg, 2024). Most of these studies craft their LLM prompts carefully, using zero-shot or few-shot learning with chain of thought (CoT) reasoning (Wei et al., 2022; Chen et al., 2023). However, there is one dimension to LLM prompting that remains understudied: how does *model memory* affect performance in annotation tasks? While few-shot learning provides some examples to the LLM to improve its performance, it does not allow the model to learn from its own previous task-specific classifications. Doing so, we find, significantly improves performance over zero-shot and few-shot learning with CoT.

A brief survey of extant research reveals the novelty of our memory-based approach. Table 1 lists publications that use LLMs as annotators alongside their prompting strategy. It shows that most of the focus has been on the prompt and its design. Many authors use zero-shot learning, where the LLM is given a prompt without any examples to guide the annotation. Some use few-shot learning, where the LLM receives a small set of examples in the prompt to illustrate how to annotate new text (Alizadeh et al., 2023;

Article	Prompting Strategy
Ornstein, Blasingame and Truscott (2023)	Few-Shot + **No memory**
Alizadeh et al. (2023)	Few-Shot + **No memory** + CoT
Mellon et al. (2024)	Few-Shot (25) + **No memory**
Gilardi, Alizadeh and Kubli (2023)	Zero-Shot + **No memory**
Kuzman, Mozetič and Ljubešić (2023)	Zero-Shot + **No memory**
Huang, Kwak and An (2023)	Zero-Shot + **No memory**
Törnberg (2023)	Zero-Shot + **No memory**
Nguyen and Rudra (2024)	Zero-Shot + **No memory**
Laskar et al. (2023)	Zero-Shot + **No memory**
Ostyakova et al. (2023)	Zero-Shot + **No memory** + Multi-Step
Reiss (2023)	Zero-Shot + **No memory**
Le Mens et al. (2023)	Zero-Shot + **No memory**
Müller and Proksch (2024)	Zero-Shot + **No memory**
Steinert and Kazenwadel (2024)	Zero-Shot + **No memory**
Rathje et al. (2024)	Zero-Shot + **No memory**
Lee et al. (2024)	Zero-Shot + **No memory**
Dickson and Yildirim (2025)	Zero-Shot + **No memory**
Heseltine and Clemm von Hohenberg (2024)	Zero-Shot + **No memory**

Table 1: Prompting Strategies of Articles using LLMs as Annotators

Mellon et al., 2024; Ornstein, Blasingame and Truscott, 2023). The highest-performing few-shot approaches use chain-of-thought prompting, 'where large LLMs are provided with both the question and a step-by-step reasoning answer as examples' (Alizadeh et al., 2023).[1] Yet one aspect remains consistent across these works: authors use a no-memory strategy, sending each new text to the LLM as an independent user message. None of the approaches reviewed allow the model to retain previous information, gain knowledge and improve its performance –as human annotators do. Furthermore, few-shot learning techniques only provide *correct* answers to the models, rather than allowing the model to understand both successes and failures.[2]

In this article, we analyze how model memory affects annotation performance by testing four different LLM *interaction approaches*: (1) zero-shot learning with no memory, (2) few-short learning (CoT) with no memory, (3) memory prompting and (4) memory reinforcement. The first two approaches reflect common state-of-the-art practices in the literature. The latter two are novel approaches we introduce in this article that, we find, lead to significant performance gains in applied political science tasks. For (1), we follow the literature and provide a system prompt[3] without any examples, and submit each new classification request as a user message independent from previous ones. We give no examples to the LLM. Importantly, the model keeps no explicit memory of the task at hand.[4] For (2), the current state-of-the-art approach, we apply few-shot learning with CoT reasoning by adding 10 examples to the system prompt in (1).[5] In separate sessions, we ask the LLM to classify each example and provide a reasoning. We then add the text, the true classification, and the full reasoning to the few-shot learning with CoT prompt.[6]

For (3), our memory prompting approach, the model keeps a memory of previous text instances it has classified before it classifies a new one. We accomplish this by sending the conversation history to the model for every new annotation request. The model thus has explicit knowledge of previous texts and how it

---

[1] Another approach is multi-step approach, where researchers use a tree-like annotation scheme that breaks labeling tasks into multiple steps with simpler questions that build up to the desired annotation task (Ostyakova et al., 2023).

[2] Chia et al. (2023) propose *contrastive CoT prompting*, where they add examples with incorrect and correct explanations to the CoT prompt in order to show the LLM how not to reason. They find improvements in arithmetic reasoning and factual QA. We think this logic supports our memory-based approaches, which take all classifications regardless of accuracy.

[3] System prompts are parts of the overall LLM prompt used to provide general instructions for a task.

[4] While generative LLMs do not store previous text instances in an API session, the model session itself remains active through the annotation task. OpenAI has stated that GPT models do not retain memory between API calls (Achiam et al., 2023). Since no textual information is kept, the model cannot improve its classification based on patterns it identifies in previous annotations and behavior. However, there is a chance that processing optimizations within model may subtly influence subsequent queries.

[5] The number of examples for few-shot learning ranges from 1 to 25, but a majority lie within the 1-10 range.

[6] See (Alizadeh et al., 2023) for further details on applying few-shot learning with CoT.

classified them, and can use this knowledge to classify new text instances – as a human annotator would. It does not, however, know if its previous annotations were correct or not.[7] The logic behind this approach is that the LLM may improve its performance over time as it gains more information on the task at hand.

Finally, memory reinforcement –(4)– is also a new method we propose that is similar in logic to reinforcement learning (Ouyang et al., 2022). This approach requires a ground truth annotated training set whose observations are sent sequentially to the LLM for classification. We set aside 20 percent of observations in the dataset for training, similar to the common 20/80 percent train-test split in machine learning. After each training annotation, the researcher informs the model whether it was correct or not, sending a punishment or reward message. We record the interaction, update the history of interactions, and send it back to the model together with the next textual observation to annotate. After classifying the full set of training examples, the model is asked to annotate the rest of the sample with knowledge of the full interaction during training. As with the second approach, the model always has a memory of previous performance. Since reinforcement learning is part of LLM model training, we hypothesize that this approach should yield the best performance.[8] By way of an example, if our dataset has 600 observations, we set aside 120 (20%) as the ground truth training set. We generate a system prompt for classification and, after sending each text for annotation, we respond with a message rewarding the model if correct, and punishing it if not, together with the correct answer.[9] We save the LLM's classification, our response to it, and its response. When we submit the request to classify the second observation in the training set, we send the saved data plus the new text to classify (the system prompt is always at the top with the task instructions). We follow this procedure with each new text from the training set. Once the full set of 120 has been classified, we save that full conversation history, and submit it together with every new text instance for the rest of the unlabeled sample (480 observations, or 80%).

---

[7]Note that in the tests below we send only a partial history with the most recent 200 classifications to avoid API rate limit errors and to limit costs for applied researchers. While sending the full conversation history would be ideal, our tests (especially with GPT-4o) showed that API errors increased significantly above 200 previous classifications. If texts are long, errors could emerge sooner, so the researcher may need to adjust this number downward to fit their task. Moreover, while GPT-4o is not an expensive model, costs can quickly accumulate when passing long conversation histories to the model for every single classification request.

[8]OpenAI's GPT-4, for instance, is fine-tuned using Reinforcement Learning from Human Feedback, which is especially relevant in filtering answers to make them more helpful and less harmful and biased (Achiam et al., 2023).

[9]In Appendix A we provide the reward and punishment prompts used in our examples.

To test the performance of these four strategies, we use state-of-the-art versions of OpenAI's and Meta's generative models at the time of writing – GPT-4o and Llama 3.1. We apply these models to two political science classification tasks, political nostalgia in party manifestos (Müller and Proksch, 2024) and incivility on Twitter (Rheault, Rayment and Musulan, 2019). Our main result and contribution is to show that memory matters: the memory prompting and memory reinforcement approaches significantly improve performance in annotation tasks, using both GPT-4o and Llama 3.1. This finding has major implications for applied researchers seeking to use generative LLMs in annotation tasks. Improved performance translates into better annotations and higher-quality data, while a reduced need for few-shot examples puts less emphasis on the prompt and improves efficiency.

Data and Methods

Our first set of tests uses data on political nostalgia from Müller and Proksch (2024). The authors coded specific sentences from party manifestos in 24 European countries into a binary measure. Four human annotators coded the data into either nostalgic or not nostalgic,[10] and the authors aggregated these into a variable where a sentence is classified as nostalgic if three of the four coders agreed.[11] The dataset contains 1,200 human-annotated sentences, with 151 coded as nostalgic and 1,049 as not nostalgic. To maintain consistency in the comparison between our two samples, and considering the significant imbalance of these data, we created a subsample of 600 observations, keeping all 151 nostalgic observations and drawing a random sample of 449 non-nostalgic ones. Sentences from the sample were 20.83 words long on average, with a minimum of 3 and a maximum of 174 words, similar to the original sample.

Our second set of tests uses Rheault, Rayment and Musulan's (2019) data on incivility on Twitter. The authors collected 2.2 million tweets about US and Canadian public officials between April and July, 2017, and manually coded a sample of 10,000 tweets as civil or uncivil through crowd-sourcing. A tweet was deemed uncivil if it contained "(1) swear words; (2) vulgarities; (3) insults; (4) threats; (5) personal attacks

---

[10]The authors define nostalgia as a 'predominately positive emotion that is associated with recalling memories of important or momentous events, usually experienced with close others' (Müller and Proksch, 2024).

[11]The authors translated each non-English sentence into English before classification. They also used different dictionary and machine learning approaches for classification, but we will use the human-annotated data for our purposes.

on someone's private life; or (6) attacks targeted at groups (hate speech)" (Rheault, Rayment and Musulan, 2019). For our tests, we drew a balanced random sample of 600 tweets from the authors' training data, with 300 civil and 300 uncivil tweets.[12] Texts were 16.76 words long on average, with a minimum of 3 and a maximum of 31 words, all similar to the original sample. We selected this sample to provide a test with a different type of text data, one with less formal and less specialized political language than data from political party manifestos.

For each of these, we use two LLMs and compare their performance – GPT-4o and Llama 3.1-70B, the latest releases by OpenAI and Meta.[13] GPT-4o is an LLM trained on a vast corpus to produce text based on a prompt by the user. It is faster and more efficient than GPT-4 while retaining similar performance.[14] These models are then optimized for dialogue using Reinforcement Learning with Human Feedback (Christiano et al., 2017), where the model produces multiple answers from a single prompt, and a human evaluates the best response. This response is then used as a reward function to improve model's output (Achiam et al., 2023). Similar to OpenAI's GPT LLMs, models from the Llama family are fine-tuned and then improved by combining techniques that use preference ranking to signal to the model the best response from several LLM responses (Wu et al., 2024). Given these learning mechanisms, we expect that retaining previous iterations of the annotation task, even if correct labels remain unknown, should improve overall model performance.

For both LLMs, we run the following tests on each of the two datasets. First, we ask the LLMs to annotate the data *without memory*, i.e. we submit each new classification request as an independent user message, with the model storing no memory of previous interactions. We do this for both the zero-shot and few-shot CoT approaches. We classify the entire dataset 10 times and calculate and report the average performance over all the runs. Second, we follow *memory prompting*, keeping the most recent 200 annotations in the conversation history when submitting a new classification request. We also generate 10 different classification runs. Lastly,

---

[12] We decided to draw a sample that had a sufficiently large N to maintain power but also would keep costs down, especially when using GPT-4o.

[13] We focus on two widely available models that are relatively easy to implement through cloud-based servers. GPT-4o can be implemented on a pay-per-token basis using OpenAI's API. Meta's Llama family is open-source and free, an important element when considering our models of choice (Palmer, Smith and Spirling, 2024). Note also that ChatGPT is the publicly available version of OpenAI's models, and uses GPT-4o in its 'pro' version.

[14] GPT-4o is also significantly cheaper than GPT-4, which is one of the reasons we selected it over GPT-4 considering its similar performance (See Table B1 in Appendix B for more details on costs).

we apply *memory reinforcement* over 10 runs.[15] For each of these tests, we use the OpenAI library in Python to query both LLMs and obtain an annotation.[16]

For the political nostalgia tests, the system prompt is: "Please classify the following sentence as containing 'nostalgic' or 'not nostalgic' ideas or subtexts about politics, society, or the past. [New line] Return only the name of the category." For the incivility classification task, we use the following base system prompt: "Please classify the following sentence as 'civil' or 'incivil'. [New line] Return only the name of the category." Each sentence is then submitted to the LLM as a new user message. For the few-shot CoT approach, we add ten examples to the prompt (i.e., text, answer, and reasoning), and then ask the model to classify each new text into one of two categories.[17] For memory prompting, we save the conversation history after every annotation and run the model again asking it to classify a new text, keeping a set of previous annotations in the conversation history.[18] For *memory reinforcement*, we follow the same procedure as with memory prompting with the hold-out training set, but add a reinforcement step where the model is told whether it was the correct or incorrect classification (see Appendix A for the reward and punishment messages). We also save the reinforcement interaction and submit it to the model before it classifies the next text. We use a temperature of 0.7 across all models to maintain consistency. We keep the rest of the hyperparameters at their default values.

Each test results in a set of classifications that we compare with the human-annotated ground truth to evaluate model performance across the various prompting approaches. In the results section, we report three key performance metrics: $F1$ scores, precision and recall, following Timoneda and Vallejo Vera (2025a,b). $F1$ is the harmonic mean between precision and recall and is a better metric than accuracy in unbalanced samples. While our incivility sample is balanced (300 civil and 300 uncivil), the models often produced unbalanced classifications, in which case $F1$ is also the best metric for true model accuracy. We report

---

[15]In Appendix C, we also provide results for both the memory prompting and memory reinforcement approaches using the few-shot CoT prompt. Adding the few-shot examples does not yield additional performance benefits for these two approaches.

[16]For Llama, we use the OpenAI library and code paired with a server that hosts the Llama model. The code is available in the replication materials.

[17]Each example sentence is indicated with 'text:'; each correct classification with 'Answer:', and each reasoning text with 'Reasoning:'. All these elements are always separated by a new line.

[18]That is, first we append the previous classification to the conversation history. Then, we append a new text to classify to the conversation history as a new user prompt. Lastly, we make a new request to the model passing the conversation history, which includes both previous interactions plus the latest text to classify. When interacting with the API this way, the researcher can have a conversation with the model through the API. The code to implement this approach is in the replication materials.

precision and recall to understand the extent of the imbalance between false positives and false negatives, which will be an important part of our results discussion below. All three statistics range between 0 and 1.

Results: Nostalgia

The results for the nostalgia dataset are in Table 2, and generally show larger differences between current no-memory approaches and our two memory-based alternatives. Table 2a displays the results for GPT-4o. Again, as with incivility, few-shot CoT bests zero-shot learning (both without memory) by 12.66 percent on average when comparing $F1$ scores. The improvement is 33.2 percent with Llama. These results confirm the well-known finding in the literature that few-shot CoT significantly improves on zero-shot performance. More importantly, however, both memory prompting and memory reinforcement show significant improvements. With GPT-4o, memory prompting yields performance gains of 5.57 percent on average when compared to few-shot CoT with no memory. Especially relevant is the gain in the nostalgic category, which goes from 0.673 to 0.753 with memory prompting, an 11.23 percent improvement. With Llama, the gain is 9.08 percent on average, with the nostalgic category's $F1$ score also improving substantially from 0.626 to 0.755, or 20.61 percent. All the $F1$ score differences between few-shot CoT and memory prompting are statistically significant at the 0.01 level in a two-sample t-test.

(a) GPT-4o

	Zero-Shot No-Memory				Few Shot No-Memory (CoT)				Memory Prompting				Memory Reinforcement			
	Prec.	Rec.	$F1$	SD_{F1}	Prec.	Rec.	$F1$	SD_{F1}	Prec.	Rec.	$F1$	SD_{F1}	Prec.	Rec.	$F1$	SD_{F1}
Not Nostalgic	0.825	0.943	0.880	0.004	0.890	0.894	0.892	0.004	0.934	0.885	0.908	0.008	0.981	0.863	**0.918**	0.008
Nostalgic	0.707	0.407	0.516	0.015	0.681	0.673	0.677	0.011	0.704	0.812	0.753	0.024	0.700	0.948	**0.804**	0.013
Overall	0.796	0.808	0.789	0.006	0.837	0.838	0.838	0.006	0.876	0.867	0.869	0.012	0.910	0.884	**0.889**	0.009

(b) Llama 3.1

	Zero-Shot No-Memory				Few Shot No-Memory (CoT)				Memory Prompting				Memory Reinforcement			
	Prec.	Rec.	$F1$	SD_{F1}	Prec.	Rec.	$F1$	SD_{F1}	Prec.	Rec.	$F1$	SD_{F1}	Prec.	Rec.	$F1$	SD_{F1}
Not Nostalgic	0.787	0.975	0.871	0.002	0.869	0.892	0.881	0.002	0.961	0.837	0.894	0.016	0.998	0.831	**0.903**	0.010
Nostalgic	0.741	0.215	0.334	0.011	0.653	0.602	0.626	0.009	0.655	0.897	0.755	0.022	0.658	0.969	**0.783**	0.015
Overall	0.776	0.784	0.736	0.003	0.815	0.819	0.817	0.004	0.884	0.853	0.859	0.017	0.905	0.866	**0.873**	0.011

Table 2: Nostalgia results for the three different interaction approaches, by LLM. Performance metrics averaged over 10 runs.

The results for the memory reinforcement approach are even stronger. Reinforcement yields an average performance improvement of 9.26 percent with GPT-4o and 11.48 percent with Llama when compared to few-shot CoT. Performance gains in the nostalgic category are especially noteworthy, as the reinforcement $F1$ score increases from 0.677 in few-shot CoT with GPT-4o to 0.804 with reinforcement, or 18.76 percent. With Llama, these numbers are 0.626 and 0.783 respectively, or a difference of 25.08 percent. Again, all these differences are statistically significant at the 0.01 level in a two-sample t-test.

Results: Incivility

Table 3 reports the main results on the incivility dataset for GPT-4o (a) and Llama 3.1 (b). Comparing across prompting strategies, performance is lowest for the zero-shot no-memory approach for both LLMs. With GPT-4o, this method yields an $F1$ score of 0.751 for civil tweets, 0.822 for uncivil tweets, and 0.786 overall. As expected, few-shot CoT with no memory significantly improves classifier performance over zero-shot by 3.8 percent, on average. All F1 gains for civil, uncivil and the overall model are statistically significant at the 0.01 level.[19] Importantly, memory prompting improves on few-shot CoT significantly, especially for civil tweets, where the $F1$ score jumps to 0.820, a 9.19 percent gain. For uncivil tweets, the $F1$ score remains

(a) GPT-4o

	Zero-Shot No-Memory				Few Shot No-Memory (CoT)				Memory Prompting				Memory Reinforcement			
	Prec.	Rec.	$F1$	SD_{F1}	Prec.	Rec.	$F1$	SD_{F1}	Prec.	Rec.	$F1$	SD_{F1}	Prec.	Rec.	$F1$	SD_{F1}
Civil	0.935	0.627	0.751	0.015	0.943	0.680	0.790	0.004	0.882	0.766	**0.820**	0.008	0.923	0.723	0.810	0.012
Uncivil	0.720	0.960	0.822	0.007	0.750	0.960	0.842	0.003	0.793	0.897	0.842	0.005	0.772	0.939	**0.848**	0.007
Overall	0.828	0.792	0.786	0.011	0.846	0.820	0.816	0.004	0.838	0.832	**0.831**	0.006	0.848	0.831	0.829	0.010

(b) Llama 3.1

	Zero-Shot No-Memory				Few Shot No-Memory (CoT)				Memory Prompting				Memory Reinforcement			
	Prec.	Rec.	$F1$	SD_{F1}	Prec.	Rec.	$F1$	SD_{F1}	Prec.	Rec.	$F1$	SD_{F1}	Prec.	Rec.	$F1$	SD_{F1}
Civil	0.957	0.480	0.639	0.007	0.947	0.627	0.755	0.005	0.874	0.722	0.795	0.012	0.909	0.763	**0.830**	0.013
Uncivil	0.653	0.978	0.783	0.003	0.724	0.966	0.828	0.003	0.764	0.896	0.828	0.005	0.797	0.920	**0.855**	0.007
Overall	0.805	0.729	0.711	0.005	0.835	0.798	0.792	0.004	0.819	0.809	0.811	0.008	0.853	0.843	**0.842**	0.010

Table 3: Incivility results for the three different interaction approaches, by LLM. Performance metrics averaged over 10 runs.

---

[19]We run two-sample t-tests with both sets of 10x runs to determine statistical significance between two F1 scores. This applies to all results below where we mention a statistically significant difference.

at 0.842, but for the overall model it increases to 0.832, an increase of 5.85 percent. These improvements are statistically significant and show that model memory can yield important performance gains, especially considering ceiling effects.[20] Gains for memory reinforcement are slightly more modest and statistically indistinguishable from those with the memory prompting approach. They are, however, equally significant on average when compared to both approaches without memory (zero- and few-shot), especially in the civil category and overall model performance.

With Llama (Table 3b), the results show more marked improvements when using either the memory prompting or the memory reinforcement approaches. For civil tweets, memory prompting improves performance by 24.41 percent –0.639 vs. 0.795– when compared to zero-shot with no memory, and 4.8 percent over few-shot CoT with no memory. For uncivil tweets, the gain is 5.3 percent over zero-shot. The $F1$ score is the same as with few-shot CoT, but overall model performance improves by 14.1 percent over zero-shot and 2.4 percent over few-shot CoT, both statistically significant differences. The largest performance gains, however, are in the memory reinforcement strategy with Llama. For civil tweets, the reinforcement $F1$ score stands at 0.830, 4.4 percent better than the memory prompting approach (0.795), 9.94 percent higher than few-shot CoT (0.755), and 29.89 percent higher than the zero-shot no-memory approach. With uncivil tweets, the gains are also significant. Reinforcement bests zero-shot no-memory by 9.19 percent (0.783 vs 0.855) and memory prompting by 3.26 percent (0.828 vs 0.855). Overall performance is also significantly improved using memory reinforcement, with gains of 18.42 percent when compared to the zero-shot no-memory approach (0.711 vs 0.842) and 4.2 percent when compared to the memory prompting strategy (0.811 vs 0.842).

Two additional findings merit mention. First is the significant increase in recall scores for both LLMs, especially with the memory prompting approach. Differences are significant in both datasets but stronger with the nostalgia data. With GPT-4o, no-memory approaches lead to a large imbalance between false negatives and false positives. Zero-shot has an average of 13 for the former and 112 for the latter, while few-shot CoT has averages of 12 and 96, respectively. This improves significantly with memory prompting, which averages 30 false negatives and 70 false positives. With Llama, large imbalances are also reduced with

---

[20] Most scores are around or above 0.8, leaving less room for improvement.

memory prompting, which has 32 false negatives and 83 false positives. As a consequence, for both models, memory prompting (GPT-4o) and memory reinforcement (Llama) produce recall scores that are significantly higher than no-memory approaches. Notice that precision also increases for the uncivil category with memory prompting and memory reinforcement, due to fewer false positives. With the nostalgia data, recall scores are also significantly higher when using memory prompting and memory reinforcement, indicating a greater balance between false negatives and false positives.

Second, there are important differences across the two LLMs. On the one hand, while both memory prompting and memory reinforcement perform consistently better than either of the no-memory approaches, it is not always the case that reinforcement yields better results than memory prompting. In the nostalgia task, reinforcement outperforms all other approaches with both LLMs, but memory prompting is generally better than reinforcement with GPT-4o in the incivility task. We suspect this may be due to task complexity: classifying incivility might be easier than nostalgia, as evinced by the lower zero-shot $F1$ scores in the nostalgia task, so the reinforcement step may lead to greater improvements with more complex texts.

Lastly, we want to note a few smaller findings that may be relevant to applied researchers. First is Llama's greater capacity to process larger amounts of tokens more rapidly and with less API errors and rate limitations. In our tests (see Table B1 in Appendix B), we find that Llama tends to run faster than GPT-4o and rarely yields an API error. Conversely, GPT-4o has limits on how many tokens per minute one can send to the API.[21] These limits may affect researchers who spend smaller amounts on the platform, especially with tasks that require sending long texts or conversation histories to the API. The reinforcement approach is such a task. Other important considerations for applied researchers are cost and time of running these models, which we report for all models in Appendix B. Llama is generally cheaper (or free) and faster than GPT-4o.[22]

---

[21]See Appendix B for a complete breakdown of these limits.

[22]While Llama is open-source, a server is required to run the model through the API, which is the most convenient way to run it. Users can also look to run it for free either in a local computer or through institutional compute facilities.

Implications and Conclusion

In this article we address an important and understudied question in LLM annotation prompting: does model *memory* affect classification performance? We test four approaches: zero-shot and few-shot CoT with no memory, the most common approaches currently in the literature; memory prompting, where we allow the model to have a memory of previous classifications; and memory reinforcement, where the model is told about its performance using a training set (20% of the sample) in an interactive session, and then allowed to classify the rest of the sample. We test these approaches using OpenAI's GPT-4o and Meta's Llama 3.1 on two political science datasets (incivility on Twitter and political nostalgia). Our main finding is that memory matters: allowing the model to keep a history of previous interactions significantly affects future performance, as does reinforcing whether a subset of previous classifications was correct or not. With both GPT-4o and Llama, the common no-memory approaches perform significantly below the memory prompting and memory reinforcement approaches.

Our findings have major implications for applied researchers using LLMs for annotation tasks, who may want to use either a memory prompting or memory reinforcement approach rather than the more common zero-shot or few-shot CoT no-memory strategies. More consistent and accurate annotations have a positive impact on downstream tasks, in particular when using the resulting data as a training set. Fine-tuned encoder-decoder models like RoBERTa may still outperform generative models in some text classification tasks (e.g., sentiment), while being considerably faster and cheaper Bucher and Martini (2024). However, transformers-based models require high-quality annotated training sets for optimal performance (Timoneda and Vallejo Vera, 2025*b*), which can be costly. With generative models, researchers can implement *memory prompting* and *memory reinforcement* to improve-task specific performance efficiently.

Given the promising results, there are various lines of inquiry to further explore. The more balanced ratio between false positives and false negatives obtained through memory prompting and memory reinforcement suggests a more consistent reasoning from LLMs that avoids falling into repetitive labeling patterns (e.g., predicting the same label for most texts). Future research can explore the extent to which this is true for tasks with varying degree of complexity. Additionally, a model that relies on memory and information for

prediction, is likely to be less reliant on the grammatical architecture of the prompt itself. This can have important implications in terms of replicability and consistency.

References

Achiam, Josh, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat et al. 2023. "Gpt-4 technical report." *arXiv preprint arXiv:2303.08774* .

Alizadeh, Meysam, Maël Kubli, Zeynab Samei, Shirin Dehghani, Juan Diego Bermeo, Maria Korobeynikova and Fabrizio Gilardi. 2023. "Open-source large language models outperform crowd workers and approach ChatGPT in text-annotation tasks." *arXiv preprint arXiv:2307.02179* 101.

Bucher, Martin Juan José and Marco Martini. 2024. "Fine-Tuned'Small'LLMs (Still) Significantly Outperform Zero-Shot Generative AI Models in Text Classification." *arXiv preprint arXiv:2406.08660* .

Chen, Jiuhai, Lichang Chen, Heng Huang and Tianyi Zhou. 2023. "When do you need chain-of-thought prompting for chatgpt?" *arXiv preprint arXiv:2304.03262* .

Chia, Yew Ken, Guizhen Chen, Luu Anh Tuan, Soujanya Poria and Lidong Bing. 2023. "Contrastive chain-of-thought prompting." *arXiv preprint arXiv:2311.09277* .

Christiano, Paul F, Jan Leike, Tom Brown, Miljan Martic, Shane Legg and Dario Amodei. 2017. "Deep reinforcement learning from human preferences." *Advances in neural information processing systems* 30.

Dickson, Zachary P and Tevfik Murat Yildirim. 2025. "The effects of COVID-19 infection on opposition to COVID-19 policies: evidence from the US congress." *Political Communication* 42(1):127–150.

Gilardi, Fabrizio, Meysam Alizadeh and Maël Kubli. 2023. "ChatGPT outperforms crowd workers for text-annotation tasks." *Proceedings of the National Academy of Sciences* 120(30):e2305016120.

Heseltine, Michael and Bernhard Clemm von Hohenberg. 2024. "Large language models as a substitute for human experts in annotating political text." *Research & Politics* 11(1):20531680241236239.

Huang, Fan, Haewoon Kwak and Jisun An. 2023. Is ChatGPT better than Human Annotators? Potential and Limitations of ChatGPT in Explaining Implicit Hate Speech. In *Companion Proceedings of the ACM Web Conference 2023*. WWW '23 ACM p. 294–297.

Kuzman, Taja, Igor Mozetič and Nikola Ljubešić. 2023. "ChatGPT: Beginning of an End of Manual Linguistic Data Annotation? Use Case of Automatic Genre Identification.".

Laskar, Md Tahmid Rahman, Mizanur Rahman, Israt Jahan, Enamul Hoque and Jimmy Huang. 2023. "CQSumDP: a ChatGPT-annotated resource for query-focused abstractive summarization based on debatepedia." *arXiv preprint arXiv:2305.06147* .

Le Mens, Gaël, Balázs Kovács, Michael T Hannan and Guillem Pros. 2023. "Uncovering the semantics of concepts using GPT-4." *Proceedings of the National Academy of Sciences* 120(49):e2309350120.

Lee, Cinoo, Kristina Gligorić, Pratyusha Ria Kalluri, Maggie Harrington, Esin Durmus, Kiara L Sanchez, Nay San, Danny Tse, Xuan Zhao, MarYam G Hamedani et al. 2024. "People who share encounters with racism are silenced online by humans and machines, but a guideline-reframing intervention holds promise." *Proceedings of the National Academy of Sciences* 121(38):e2322764121.

Mellon, Jonathan, Jack Bailey, Ralph Scott, James Breckwoldt, Marta Miori and Phillip Schmedeman. 2024. "Do AIs know what the most important issue is? Using language models to code open-text social survey responses at scale." *Research & Politics* 11(1):20531680241231468.

Müller, Stefan and Sven-Oliver Proksch. 2024. "Nostalgia in European party politics: a text-based measurement approach." *British Journal of Political Science* 54(3):993–1005.

Nguyen, Thi Huyen and Koustav Rudra. 2024. Human vs ChatGPT: Effect of Data Annotation in Interpretable Crisis-Related Microblog Classification. In *Proceedings of the ACM on Web Conference 2024*. pp. 4534–4543.

Ornstein, Joseph T, Elise N Blasingame and Jake S Truscott. 2023. "How to train your stochastic parrot: Large language models for political texts." *Political Science Research and Methods* pp. 1–18.

Ostyakova, Lidiia, Veronika Smilga, Kseniia Petukhova, Maria Molchanova and Daniel Kornev. 2023. Chatgpt vs. crowdsourcing vs. experts: Annotating open-domain conversations with speech functions. In *Proceedings of the 24th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. pp. 242–254.

Ouyang, Long, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray et al. 2022. "Training language models to follow instructions with human feedback." *Advances in neural information processing systems* 35:27730–27744.

Overos, Henry David, Roman Hlatky, Ojashwi Pathak, Harriet Goers, Jordan Gouws-Dewar, Katy Smith, Keith Padraic Chew, Jóhanna K Birnir and Amy H Liu. 2024. "Coding with the machines: machine-assisted coding of rare event data." *PNAS nexus* 3(5):pgae165.

Palmer, Alexis, Noah A Smith and Arthur Spirling. 2024. "Using proprietary language models in academic research requires explicit justification." *Nature Computational Science* 4(1):2–3.

Rathje, Steve, Dan-Mircea Mirea, Ilia Sucholutsky, Raja Marjieh, Claire E Robertson and Jay J Van Bavel. 2024. "GPT is an effective tool for multilingual psychological text analysis." *Proceedings of the National Academy of Sciences* 121(34):e2308950121.

Reiss, Michael V. 2023. "Testing the reliability of chatgpt for text annotation and classification: A cautionary remark." *arXiv preprint arXiv:2304.11085* .

Rheault, Ludovic, Erica Rayment and Andreea Musulan. 2019. "Politicians in the line of fire: Incivility and the treatment of women on social media." *Research & Politics* 6(1):2053168018816228.

Steinert, Christoph Valentin and Daniel Kazenwadel. 2024. "How user language affects conflict fatality estimates in ChatGPT." *Journal of Peace Research* p. 00223433241279381.

Timoneda, Joan C and Sebastián Vallejo Vallejo Vera. 2025*a*. "Behind the mask: Random and selective masking in transformer models applied to specialized social science texts." *PloS one* 20(2):e0318421.

Timoneda, Joan C. and Sebastián Vallejo Vera. 2025*b*. "BERT, RoBERTa or DeBERTa? Comparing Performance Across Transformer Models in Political Science Text." *The Journal of Politics* .

Törnberg, Petter. 2023. "Chatgpt-4 outperforms experts and crowd workers in annotating political twitter messages with zero-shot learning." *arXiv preprint arXiv:2304.06588* .

Wei, Jason, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou et al. 2022. "Chain-of-thought prompting elicits reasoning in large language models." *Advances in neural information processing systems* 35:24824–24837.

Wu, Tianhao, Weizhe Yuan, Olga Golovneva, Jing Xu, Yuandong Tian, Jiantao Jiao, Jason Weston and Sainbayar Sukhbaatar. 2024. "Meta-rewarding language models: Self-improving alignment with llm-as-a-meta-judge." *arXiv preprint arXiv:2407.19594* .

Appendix for: "Memory Is All You Need"

Appendix A: Reward and Punishment Prompts

In the main manuscript we describe memory reinforcement, a prompting approach where the model is shown a text to classify, and then the researcher informs the model whether it was correct or not, sending a punishment or reward message. After the model provides a correct response to the classification task, it would see the following reward message:

> Congratulations, your classification was correct! Remember this for future classifications.

After the model provides an incorrect response to the classification task, it would see the following reward message:

> Unfortunately, your classification was incorrect. It should be: [expected label]. Try again to classify it, and remember to update your knowledge for future classifications.

Appendix B: The Costs, Limits and Time of Running LLMs

In our tests, we find that Llama 3.1 tends to run faster than GPT-4o and rarely yields an API error. It is also cheaper when deployed through a web server like `deepinfra.com`.[1] Llama 3.1 is free when deployed locally, yet the hardware required to do so makes it an unrealistic option for many researchers. As Table B1 shows, the most expensive technique is *memory reinforcement*, as the number of input tokens increases with every run at a higher rate than with *memory prompting* (both OpenAI and DeepInfra charge by the input and output token). The differences in cost between Llama 3.1 and GPT-4o are stark. For example, with the 600 observations from the incivility dataset, running memory reinforcement would cost $12.63 with GPT-4o and $2.67 with Llama 3.1. Similarly, to run the same dataset with the same technique would take GPT-4o around 31 minutes, while it only takes Llama around 7 minutes. In both cases, however, the time to label the data is considerably lower than what it would take a human coder. Overall, considering costs, time, and performance, Llama seems to be the best option. However, testing the performance of both models with both memory approaches when using them in applied research should be within reach for many researchers.

As for rate limitations, Llama 3.1 has none while GPT-4o varies according to usage tiers – usage based on the amount of money spent on the platform. At Tier 1 ($5 paid), 500 calls per minute and 30,000 tokens per minute; at Tier 2 ($50 paid), 5,000 calls per minute and 450,000

Model	Prompting Strategy	Case	Avg. Time (minutes)	Cost USD$ (per run)
GPT-4o	Zero-Shot No-Memory	Incivility	6.15	$0.10
GPT-4o	Few-Shot No-Memory (CoT)	Incivility	5.85	$1.01
GPT-4o	Memory Prompting	Incivility	14.87	$14.52
GPT-4o	Memory Reinforcement	Incivility	30.53	$12.63
Llama 3.1-70b	Zero-Shot No-Memory	Incivility	2.2	<$0.01
Llama 3.1-70b	Few-Shot No-Memory (CoT)	Incivility	2.42	<$0.01
Llama 3.1-70b	Memory Prompting	Incivility	5.58	$0.45
Llama 3.1-70b	Memory Reinforcement	Incivility	6.86	$2.67
GPT-4o	Zero-Shot No-Memory	Nostalgia	5.326	$0.11
GPT-4o	Few-Shot No-Memory (CoT)	Nostalgia	6.72	$1.09
GPT-4o	Memory Prompting	Nostalgia	12.12	$15.87
GPT-4o	Memory Reinforcement	Nostalgia	29.13	$13.8
Llama 3.1-70b	Zero-Shot No-Memory	Nostalgia	4.48	<$0.01
Llama 3.1-70b	Few-Shot No-Memory (CoT)	Nostalgia	5.03	<$0.01
Llama 3.1-70b	Memory Prompting	Nostalgia	5.92	$0.49
Llama 3.1-70b	Memory Reinforcement	Nostalgia	7.98	$2.89

Table B1: Cost and time for each run, per model and dataset.

---

[1]This is the service we used to run our Llama models. The model is free but the server charges by the token at very low rates, as Table B1 shows. Researchers can use Llama locally for free or through a service such as this one.

tokens per minute; at Tier 3 ($100 paid), 5,000 calls per minute and 800,000 tokens per minute. When running *memory reinforcement*, we used ∼90M input tokens 7.921 output tokens over 10 runs and 310 minutes. We found that the rate limitations from OpenAI did not affect our pipeline after Tier 3. However, these limits may affect researchers who spend smaller amounts on the platform, especially with tasks that require sending long texts or conversation histories to the API.

Appendix C: Few-shot Learning vs. No-Restart and Reinforcement

This appendix further compares few-shot learning CoT with memory prompting and memory reinforcement. First, we add the few-shot CoT prompt to both the original prompts for memory prompting and memory reinforcement. We do this to rule out that few-shot could further improve model performance. The most relevant result in Table C1, using the incivility dataset, is that including a few-shot prompt does not improve on the main results from the memory prompting and memory reinforcement approaches in Tables 2 and 3. In the case of Llama 3.1, memory reinforcement with zero-shot remains the clear best performer (see Table 2b in the main article).

(a) GPT-4o

	Few-shot Memory Prompting				Few-shot Memory Reinforcement			
	Prec.	Rec.	$F1$	SD_{F1}	Prec.	Rec.	$F1$	SD_{F1}
Civil	0.886	0.775	0.827	0.008	0.911	0.724	0.807	0.009
Uncivil	0.800	0.900	0.847	0.005	0.771	0.929	0.843	0.003
Overall	0.843	0.837	0.837	0.006	0.841	0.826	0.825	0.006

(b) LLaMA 3.1

	Few-shot Memory Prompting				Few-shot Memory Reinforcement			
	Prec.	Rec.	$F1$	SD_{F1}	Prec.	Rec.	$F1$	SD_{F1}
Civil	0.873	0.733	0.796	0.010	0.915	0.731	0.813	0.010
Uncivil	0.770	0.893	0.827	0.006	0.776	0.932	0.847	0.005
Overall	0.821	0.813	0.811	0.007	0.846	0.832	0.830	0.007

Table C1: Few-shot learning incivility results for the three different interaction approaches, by LLM.

Second, we also tested a variant of our reinforcement approach stripped of the reinforcement step, which would make the test equivalent to a few-shot learning approach with 120 examples. The results, as expected, are significantly worse than the original memory reinforcement approach, as few-shot learning is known to decrease performance with larger numbers of examples (**?**). Average performance losses range between 20 and 40 percent when compared to memory reinforcement, depending on model and dataset. This result also strengthens the value of the reinforcement step, as it does not appear to induce any of the issues related to adding too many examples to a few-shot prompt.