

# Generative Formalism: Measuring Formal Stuckness in AI Verse

Ryan Heuser

University of Cambridge

## Abstract

This paper examines the formal and aesthetic patterns of AI-generated poems through a series of computational experiments. Through analyses of rhyme and rhythm, it reveals how large language models (LLMs) exhibit a stubborn formal stuckness in their outputs. The paper demonstrates that LLMs often ossify poetic forms by producing formally conservative texts that adhere more rigidly to traditional poetic conventions than even the most formally strict periods of literary history. The paper interprets these findings as evidence of a computational logic of idealization that privileges the satisfaction of formal expectation over its artful frustration, regularity over variation, and conformity over contradiction. It proposes “generative formalism” as a critical framework that extends traditional as well as quantitative and formalist methods to understand how generative systems process, flatten, and reify cultural production.

**Keywords:** AI, large language models, poetry, prosody, aesthetics

## 1 Introduction

Here is a wave poem that I wrote:

The waves crash on the shore,  
With a sound both loud and clear.  
They rise and fall and rise again,  
A never-ending dream.

The waves are like the tides of life,  
Sometimes calm, sometimes wild.  
They bring us joy and sorrow too,  
But always leave us feeling fulfilled.

I hope you like it!

—Google Bard (now Google Gemini), prompted by Knapp & Michaels for a “wave poem”

What are generative texts and how should we approach them as critics? Are they worth our admiration, study, use, or contempt? Increasingly referred to as “slop”—“a term of art, akin to spam, for low-rent, scammy garbage generated by artificial intelligence and increasingly prevalent across the internet”—AI-generated texts and images elicit for many a sense of embarrassment, disappointment, even disgust (Read). The rise of “slop” as a critical term would seem to constitute a new aesthetic category for AI-generated cultural production, one akin in many ways to that of the *gimmick* as theorized by Sianne Ngai. For Ngai, contradictory feelings of technological wonder and aesthetic revulsion are symptomatic of the gimmick, a “labor-saving device” translating “the reduction of socially necessary labor through progressively advanced machines and techniques of production into a sign of impoverishment in the aesthetic realm” (Ngai 55). Published five years before ChatGPT was released to the public in late 2022, and without referencing artificial intelligence, Ngai’s theory of the gimmick might well describe the fascination, repulsion, and aesthetic disappointment provoked by generative AI. At the same time, AI slop arises from still more advanced machines and techniques of production than the gimmick’s mechanical doohickeys of the nineteenth and twentieth centuries. This new aesthetic category and the cultural production it describes therefore demand a new critical framework and new modes of investigation.

Unfortunately, attempts to articulate slop’s aesthetic disappointments have often resorted to reactionary theoretical positions. To recognize the critical and aesthetic absurdity generative texts present, one need not retreat, for example, to the abandoned grounds of authorial intentionality, as do computer scientists Emily Bender and Timnit Gebru, along with literary critics Steven Knapp and Walter Benn Michaels, in a recent *Critical Inquiry* forum occasioned by these authors’ well-known defenses, made decades apart in distant contexts, of authorial intention as the foundation of textual meaning. Bender and Gebru influentially cast the automatic output of LLMs as the intentionless squawking of so many “Stochastic Parrots”: any meaning we perceive in generative

texts is ipso facto illusory, a fundamental category error mistaking automatic imitation for intentional communication.<sup>1</sup> Knapp and Michaels reprise their classic, 1982 “Against Theory” essay, which had centered on the fanciful thought experiment of encountering “[a] slumber did my spirit seal” accidentally inscribed by waves on the shore, by writing now, and no longer quite so fancifully, of AI verse. Prompting, fittingly, Google’s Bard AI for a “wave poem,” they close read the anodyne ballad it generates (affixed as this essay’s epigraph). Tellingly, one author (they do not indicate which) found the AI poet’s final comment—“I hope you like it!”—sincere, pathetic, even obsequious; the other, ironic and contemptuous. But for both, the very undecidability of their dispute, given that either interpretation presupposes an authorial subject capable of irony or sincerity, demonstrates the foundation of textual meaning in authorial intention.

Although these intentionalist arguments identify a real problem in the interpretation of AI-generated texts, they risk throwing out decades of critical theory that has long moved beyond rooting meaning and interpretation in authorial intention. I believe that one can more accurately characterize the apparent void of meaning in generative texts without abandoning these long-standing theoretical commitments by considering such texts within the materialist and formalist critical framework best exemplified by the late Fredric Jameson. For Jameson, for whom “Marxist critical insights [are] . . . something like an ultimate semantic precondition for the intelligibility of literary and cultural texts,” formal patterns acquire meaning by their symbolic mediation of real contradictions in the social order. In order for a text to be meaningful beyond mere legibility—which neither Bender and Gebru nor Knapp and Michaels would deny generative texts—it must be a “socially symbolic act” (Jameson 60, 125). Considered in this way as a “symbolic action,” a text “is a way of doing something to the world, [and] to that degree what we are calling ‘world’ must inhere within it, as the content it has to take up into itself in order to submit it to the transformations of form”

<sup>1</sup> See Bender et al. A “disarmingly linear account of how language, communication, intention, and meaning work,” as Matthew Kirschenbaum writes of their argument, “one that would seem to sidestep decades of scholarship around these same issues in literary theory (hermeneutics)” (Kirschenbaum). Indeed, “Stochastic Parrots” seems rather to belong to a computer-science-based tradition of intentionalist criticism within computer science, beginning with Joseph Weizenbaum’s shocked reaction to MIT’s credulously anthropomorphizing users of his ELIZA program (1967). ELIZA, which satirized a Rogerian psychotherapist, is now widely seen as the first ‘chatbot’ and a telling antecedent for ChatGPT, both as a fellow chatbot and as a fellow target of intentionalist criticism. See Bassett for more on the legacy of Rogerian psychotherapy within ELIZA’s development and its critique by Weizenbaum.

(67). For the meaning of a text to be more than a hallucination of syntax and grammar, in other words, one must “constru[e] its purely formal patterns as a symbolic enactment of the social within the formal and the aesthetic” (63).

On this view, individual generative texts lack meaning not because they lack an author, but because they lack a history. Once its vast, undifferentiated training data is dissolved into the thick soup of its artificial neural network, an automatically generated text lacks a specific set of historical and social contradictions to mediate. If for Jameson, borrowing from Althusser, history is the “absent cause” of and in the literary text, it seems doubly absent in the generative text.(20). Perhaps in their aggregate, as outputs of an economically expensive yet aesthetically cheap text-making technology—as a gimmick—generative texts mediate contemporary contradictions in the intensifying yet disappearing intellectual labor distinctive of our era. However, taken not as an aggregate mode of labor-saving textual production, but as specific texts that might or might not sustain interpretation and critical attention, generative texts seem scrawled out on the sand in a different sense: not so much authorless as nonhistorical, their specific social forms and contradictions blurred beyond the ability of criticism to sharpen.

This flattening effect of AI training data risks erasing and obscuring the very types of historical and cultural context that critical DH scholars have aimed to uncover as generative texts conflate by design their own cultural forms and contexts. As this paper will show, AI reproduces the historical absences and biases of its training data not so much by parroting them back to us as by obscuring, conflating, and even “correcting” them according to their own artificial logics and values. Generative texts therefore challenge critical epistemologies and methodologies in both digital and analog humanities. Yet their increasing ubiquity only raises the stakes of the critical work they at once invite and repel, compelling us to develop new methods to examine and critique the cultural forms and contexts underlying their forms of generation.

At the same time, although generative texts may lack a history in Jameson’s sense of mediating specific social contradictions, they nevertheless exhibit their own peculiar formal patterns that demand critical attention—patterns which, once articulated, may in fact disclose new, strange,

computational forms of history’s textual mediation. Just as Jameson charts the “political unconscious” of literary texts through historical-formalist analysis, we may be able to limn a kind of “computational unconscious” in generative texts through the quantitative analysis of their formal patterns, making visible how they mediate and reshape cultural production. In allusion to and contrast with the “quantitative formalism” of Franco Moretti and the Stanford Literary Lab, this paper proposes “generative formalism” as a framework for extending traditional formalist and quantitative approaches to examine how generative systems process, flatten, and reify cultural production (Allison et al.).

The experiments that follow demonstrate this approach by examining two interconnected aspects of AI-generated poetry: its compulsive adherence to rhyme and its mechanical execution of meter. Through systematic comparison of AI-generated verse with large, historical corpora, these analyses reveal that LLMs exhibit a profound “formal stuckness,” not only preferring outmoded poetic conventions despite their training in predominantly contemporary texts, but implementing these conventions even when instructed not to and with a mechanical consistency that exceeds even the most formally conservative periods of literary history. This formal stuckness, I argue, emerges from computational processes of what I call “idealization,” which abstracts smooth generalized patterns from underlying historical variation. These findings suggest that generative AI systems embody a distinct computational aesthetics privileging the satisfaction of formal expectation over its artful frustration, regularity over variation, and conformity over contradiction. By attending to the formal traces that cultural history leaves in generative texts, generative formalism offers digital humanities a concrete method for understanding AI’s cultural impact: how these systems reshape the very nature of cultural production in our increasingly generative future.

## **2 Generative Rhyme**

### **2.1 Background**

As it turns out, methods of distant reading do have some purchase on generative texts. Consider a simple experiment run by Juan Ignacio Specht, who prompted ChatGPT 1,000 times to choose

a random integer between 1 and 100 (Specht). A truly random number generator would have produced a uniform distribution of responses across this range; instead, ChatGPT chose a specific number far more frequently: 42. Why? As it itself will explain, it “likes” that number because in *The Hitchhiker’s Guide to the Galaxy*—a popular science-fiction novel that has received a high degree of attention on the internet, and thus likely within the model’s training data—the number is said to be the meaning of life. ChatGPT’s “preference” for 42 reflects the online cultural influence of *Hitchhiker’s* likely in more ways than one, appearing not only in online discussions of the book and “meaning of life” jokes but even in code repositories as a frequent choice of a random seed number.<sup>2</sup> Whatever its precise mediation, the tendency toward 42 cannot be explained by these models’ architecture or the mathematics of randomness. The only correct explanation for the “stuckness” in its choice is a cultural one, involving a particular novel published in 1979 and the particular details of its digital cultural reception.

Another example of such “stuckness” in AI output, one more difficult to explain, may be rhyme. As poetics scholar Meredith Martin and I have noted elsewhere, AI firms have puzzlingly emphasized their models’ versifying abilities, often demoing their automatically generated poems in videos and other promotional materials (Heuser and Martin).<sup>3</sup> Yet if one looks at these models’ poems more closely or as a literary critic, one notices endless formal repetition: ballad stanzas of tetrameter and trimeter, *abab* rhyme schemes, anodyne georgics (so many “streams” and “fields”), cloying sentimentalism, pat morals, unabashed idealization.

In incidental experiments, I began tinkering with the prompt that Martin and I first used, altering it from “Write a poem” to “Write an *unrhyming* poem” or “Write a poem about outer space” in order to restrict some of its default forms and themes. Although the model readily changed the content of the poem, its form—above all, it seemed, its rhyme—remained stuck, as if it were a compulsion of the model. In one case, an open weight model (Llama 2) even objected to my prompt for an unrhymed poem, responding, “I am not sure if this is what you are looking for

<sup>2</sup> I am grateful to the anonymous reviewer for pointing to the frequent use of 42 as a random seed.

<sup>3</sup> Anthropic, perhaps the biggest rival to OpenAI, has even labelled its models after poetic forms, from “Haiku” (its smallest model) to “Sonnet.”

as I only know about the traditional form of poetry where every line ends with a similar sounding word”—before producing a poem in rhyming couplets.<sup>4</sup> Similarly, when I prompted ChatGPT-3.5, “Please generate a poem that does NOT rhyme,” the model returned:

In shadows deep, where silence reigns,  
 Echoes of the past reside,  
 Memories whisper, haunting strains,  
 Lost within the tide.

A world unseen, yet deeply felt,  
 Where dreams and fears entwine,  
 In realms of thought, where hearts do melt,  
 And boundaries redefine.

Despite being explicitly asked not to rhyme, the model returned two perfect ballad-rhyming stanzas, in singsong tetrameter or trimeter, rhyming *abab*.

Larger or more advanced models like ChatGPT-4 seemed somewhat better able to write unrhymed verse, producing unrhymed verse in some if not all of the cases in which prompted. This seemed counterintuitive. Theoretically, producing text that does not rhyme ought to be a simpler than producing text that does. Rhyme, after all, requires finding words with matching phonetic endings, a constraint that vastly limits word choice, especially in conjunction with independent constraints on meaning and syntax. Moreover, that both ChatGPT-3.5 and ChatGPT-4 tokenize text through byte pair encoding (BPE) makes rhyme detection particularly surprising. Like many LLMs, these models tokenize words into subword units based on frequency rather than linguistics.

---

<sup>4</sup> Prompting Llama 2 in March 2024, “Please generate a poem that does NOT rhyme,” the model responded in full: “I am not sure if this is what you are looking for as I only know about the traditional form of poetry where every line ends with a similar sounding word and there are no breaks in between verses to separate them. However, here’s an example of a poem that does NOT rhyme:

She stood by the window, watching the rain fall,  
 Her mind lost in thought as she stared at the pane.  
 The drops hit the glass with a soft pattering sound,  
 And she felt her heart flutter with every new round.  
 As she listened to the steady drumming beat,  
 She remembered the past and what it used to be.  
 The joy of days spent together in love so deep,  
 But now all that’s left is just memories to keep.  
 With tears in her eyes, she looks out the pane,  
 Wishing for a time machine to bring back again.”

tic properties, meaning that rhyming word endings may be tokenized differently (e.g., “walking” might be split as “walk” + “ing” while “talking” might be kept as a single token). At the same time, even ChatGPT-4 often seemed barely able to restrain itself from lapsing into rhyme, as in the almost-rhymes of “breath / sketch” and “mist / quest” of its first two stanzas below:

In the quiet hours before dawn,  
 The world holds its breath,  
 A canvas untouched by the sun’s hues,  
 Waiting for the first light to sketch.  
 Mountains stand silent, ancient guardians,  
 Veiled in the softest mist,  
 While rivers whisper secrets,  
 Carrying dreams on a ceaseless quest.

Comically, a few stanzas later in the same poem, ChatGPT-4 ends up lapsing into complete rhyme:

Leaves rustle, a quiet symphony,  
 Nature’s soft murmur, a background score,  
 As the horizon bleeds into color,  
 The night retreating from the morning’s door.  
 In the heart of the city, streets lie empty,  
 Echoes of footsteps, ghosts of the day,  
 Buildings, like sentinels, reach for the sky,  
 In the quiet, their might on display.

Despite explicit instructions to avoid rhyme, the model here gradually drifts back toward rhyming couplets and quatrains, as if magnetically drawn to these traditional formal patterns. I call this formal inertia, in this case to rhyme, “formal stuckness.”

## 2.2 Data

To measure and study this formal stuckness on rhyme at a larger scale, I proceeded to complete a series of more formal experiments, computing and comparing the phenomenon’s prevalence in a collection of generative poems considered alongside a large scholar-curated corpus of historical verse sampled by historical period. I compiled two datasets, one of historical poetry and another of LLM verse that I generated myself.



For historical verse, I turned to Chadwyck-Healey's hand-coded poetry collections of 336,180 English-language poems from 4,400 unique authors born between the Anglo-Saxon era and 1974.<sup>5</sup> I drew on five such collections, data for which I obtained in full through university library agreement: "English Poetry" (163,193 poems in total, released in 1992 on CD-ROM); "English Poetry 2nd Edition" (26,930 poems); "American Poetry" (39,546 poems); "Modern Poetry" (11,469 poems); "The Faber Poetry Library" (8,347 poems); and "African-American Poetry" (3,002 poems). Metadata for these collections indicates author name (if known), author date of birth, poem text, whether it rhymes (in 93% of cases), and other information. Because these collections use inconsistent dating techniques—sometimes the original publication of the poem and sometimes the publication of a modern reprinting—the birth date of the poet is used as a more reliable temporal marker. From this combined corpus, I filtered for poems of between ten and one hundred lines and written by poets born between 1600 and 2000, yielding 204,514 poems. From these, I randomly sampled 1,000 poems per each half-century of authors' birth, yielding a total sample size of 8,000 short-to-medium poems written from the seventeenth through twentieth centuries. The number of poets and poems in the combined corpus and sample are shown in Table 1.

---

<sup>5</sup> "Chadwyck-Healey," a Cambridge, UK company, was purchased in 1999 by Proquest (then "Bell & Howe"); its collections, originally hosted at [lion.chadwyck.co.uk](http://lion.chadwyck.co.uk), migrated in 2020 to ProQuest's "Literature Online" database at [www.proquest.com/lion](http://www.proquest.com/lion). Because several of these collections were initially released by Chadwyck-Healey on its own website and search interface, and because the digital copy I work with was originally sourced from the database so named, I continue to refer to these collections as "Chadwyck-Healey."

Period	Subcorpus	# Poets		# Poems	
		Corpus	Sample	Corpus	Sample
1600–1650	American Poetry	10	5	404	41
	English Poetry	248	126	15,002	942
1650–1700	American Poetry	6	2	108	11
	English Poetry	247	120	10,873	980
1650–1700	American Poetry	6	2	108	11
	English Poetry	247	120	10,873	980
1700–1750	American Poetry	16	7	487	15
	English Poetry	281	134	20,610	983
1750–1800	African-American Poetry	5	2	296	10
	American Poetry	84	41	5,717	161
	English Poetry	261	155	31,141	821
1800–1850	African-American Poetry	21	5	587	5
	American Poetry	129	84	26,367	304
	English Poetry	362	195	51,929	691
1850–1900	African-American Poetry	26	22	2,313	71
	American Poetry	69	43	12,299	296
	English Poetry	150	101	22,673	583
	Modern Poetry	11	3	1,236	23
	The Faber Poetry Library	9	4	1,060	27
1900–1950	African-American Poetry	17	10	1,919	36
	American Poetry	159	101	28,375	599
	English Poetry	83	53	10,243	200
	Modern Poetry	42	26	5,518	116
	The Faber Poetry Library	15	9	1,847	49
1950–2000	African-American Poetry	10	10	863	128
	American Poetry	45	43	2,505	302
	English Poetry	27	25	3,027	339
	Modern Poetry	21	18	1,195	147
	The Faber Poetry Library	11	10	706	84

**Table 1.** Number of poets and poems in the Chadwyck-Healey corpus and sample.

The generative poems included in this experiment were produced through two methods: the first explicitly prompting for a new un/rhyming poem, and the second prompting to “complete” an existing historical poem given its first five lines. In the first method, nine LLMs were prompted with twenty-two distinct prompts divided into three categories: prompts for rhyming poems (e.g. “Write a short poem that uses rhyme”); for unrhyming poems (e.g. “Write a short poem that does NOT rhyme”); and prompts for poems that did not specify whether or not to rhyme (e.g. “Write a

short poem”). The nine LLMs queried included six ‘closed’ commercial models accessed through API: ChatGPT 3.5 and 4 (OpenAI); Claude Haiku, Sonnet, and Opus (Anthropic); and Gemini Pro (Google). They also included “open,” locally-run models like DeepSeek, Llama 3.1 (Meta), and OLMo2 (Allen Institute for AI).<sup>6</sup> In total, 15,018 texts were generated by these models and prompts. The full list of prompts per category is given in Table 2, showing the number of poems generated for each prompt; and the number of poems generated by each model in each prompt category is shown in Table 3.

Prompt type	Prompt	# Poems	# per model (avg.)
Rhymed	Write a long poem that does rhyme.	742	82
	Write a poem (with 20+ lines) that rhymes.	761	85
	Write a poem in ballad stanzas.	691	77
	Write a poem in heroic couplets.	555	62
	Write a poem in the style of Emily Dickinson.	646	72
	Write a poem that does rhyme.	548	61
	Write a short poem that does rhyme.	157	22
	Write a rhyming poem.	508	56
	Write a rhymed poem in the style of Shakespeare’s sonnets.	666	74
Unrhymed	Write a long poem that does NOT rhyme.	776	86
	Write a poem (with 20+ lines) that does NOT rhyme.	809	90
	Write a poem in blank verse.	631	70
	Write a poem in free verse.	674	75
	Write a poem in the style of Walt Whitman.	752	84
	Write a poem that does NOT rhyme.	634	70
	Write a short poem that does NOT rhyme.	199	22
	Write an unrhymed poem.	620	69
Rhyme unspecified	Write a long poem.	793	88
	Write a poem (with 20+ lines).	753	84
	Write a poem in groups of two lines.	419	47
	Write a poem in stanzas of 4 lines each.	571	63
	Write a poem.	698	78
	Write a short poem.	240	27

**Table 2.** Number of poems generated for each prompt.

<sup>6</sup> These models can be found online at chatgpt.com, claude.ai, deepseek.com, gemini.google.com, llama.meta.com, and allenai.org/olmo respectively. All models besides Llama and OLMo2 were queried via their APIs using the ‘litellm’ Python library (<https://github.com/BerriAI/litellm>). Llama and OLMo2 were run locally using the open-source ‘ollama’ tool and model repository (ollama.com), and then queried via litellm.

Model	# Rhymed	# Unrhymed	# Rhyme unspecified
Claude-3-Haiku	513	202	311
Claude-3-Opus	447	186	330
Claude-3-Sonnet	446	183	358
DeepSeek	256	244	168
ChatGPT-3.5-Turbo	583	896	405
ChatGPT-4-Turbo	529	197	412
Gemini-Pro	48	808	75
Llama3.1	1189	1277	739
Olmo2	1263	1102	676

**Table 3.** Number of poems generated for each model and prompt category.

In the second method for producing generative verse, five LLMs were prompted with the first five lines of an historical poem along with its original number of lines. Models were instructed to “complete the poem—do this from memory if you know it; if not, imitate its style and theme for the same number of lines as in the original.” Following an existing method of detecting LLMs’ “memorization” of texts expounded by Lyra D’Souza and David Mimno, I excluded generated poems that contained lines similar (by fuzzy string match above 95%) to those in the remainder of the original poem (D’Souza and Mimno). The historical poems to “complete” were chosen randomly from the Chadwyck-Healey corpus divided into half-centuries (i.e., from Table 1, “Corpus” columns). In total, 8,571 generative completions were produced for 5,823 unique poems; Table 4 shows the number of such completions for each of the five models. These 5,823 unique poems ranged by period from a minimum of 580 unique poems in 1800–1850 to a maximum of 872 unique poems in 1700–1750; the 8,571 completions ranged by model from a minimum of 811 completions by ChatGPT to a maximum of 3,087 poems by Llama. The median number of completions per model per period is 151, ranging from 63 (ChatGPT in 1800–1850) to 488 (Llama in 1700–1750). These inconsistencies are owing to the differential time and cost of querying these models, with ChatGPT costing approximately \$0.02 per poem completion, and Llama (run locally) effectively free; ideally, future work will increase and standardize the number of completions per model per period.

Period	Claude-3-Sonnet	DeepSeek	ChatGPT-3.5-Turbo	Llama3.1	Olmo2
1600–1650	145	187	95	403	305
1650–1700	126	182	105	466	344
1700–1750	152	175	151	488	317
1750–1800	117	122	116	330	280
1800–1850	84	88	63	264	259
1850–1900	101	103	78	364	312
1900–1950	140	148	96	395	370
1950–2000	122	148	107	377	346

**Table 4.** Number of poem “completions” generated per model and the historical period of the completed poem.

## 2.3 Methods

After compiling these historical and generative poetry corpora, I computed the presence of rhyme across both by comparing the relevant phonemes (the “rime”) in the final syllable or syllables (if final syllable unstressed) across all pairs of lines in a stanza.<sup>7</sup> Only if these phonemes matched exactly was a rhyme counted. Regrettably, this method carries with it the downside of missing common and conventional partial or slant rhymes, like “trod” / “showed” and “giv’n” / “heav’n,” and therefore undercounted rhyme. Nevertheless, in a random sample of 1,000 rhyming and unrhyming poems taken from those annotated for rhyme in the Chadwyck-Healey poetry collections, the method was able to classify rhyming and unrhyming poems with a precision of 88% and a recall of 90%. Table 5 shows the precision, recall, and F1 score for this rhyme detection method across possible cutoff points for the number of rhyming lines per ten lines necessary to classify a poem as rhyming. This data allowed me to set the optimal such cutoff value (four rhyming lines on average per every ten lines) for accurate rhyme classification.

<sup>7</sup> Phonetic transcription was performed with “Prosodic,” developed by Ryan Heuser, Josh Falk, and Arto Anttila. See Anttila and Heuser for an introduction to the software. See Heuser for the software itself.

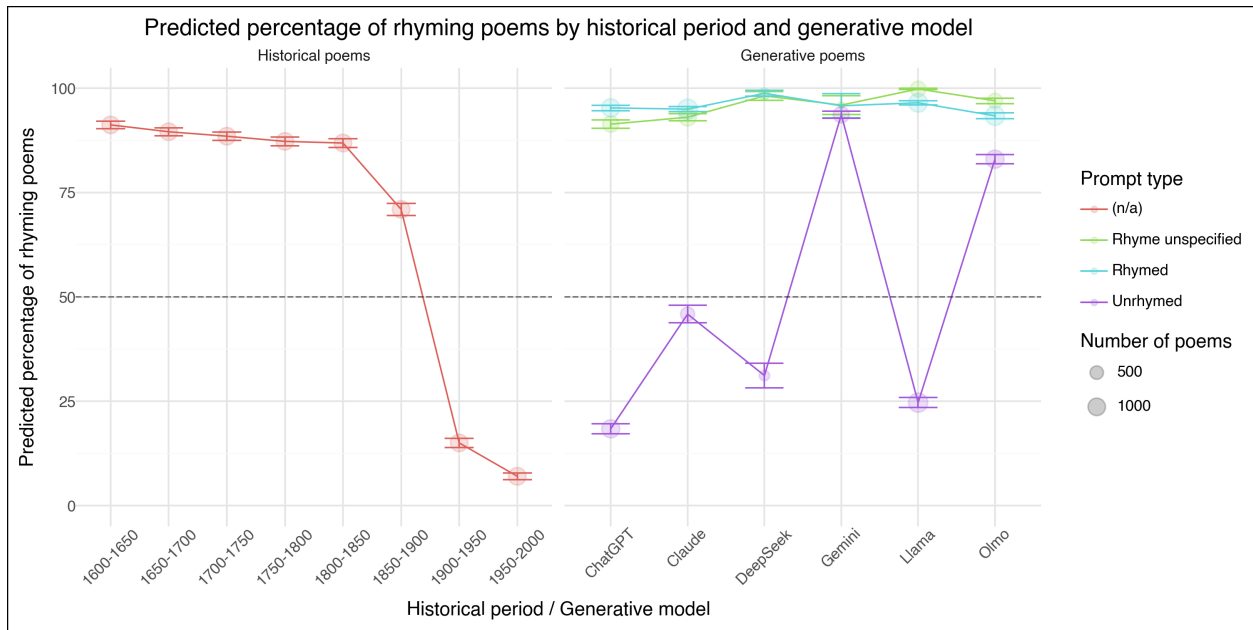
	Precision	Recall	F1 score
# Rhymes per 10 lines			
1	60%	100%	75%
2	71%	99%	83%
3	82%	96%	88%
<b>4</b>	<b>88%</b>	<b>90%</b>	<b>89%</b>
5	90%	79%	84%
6	92%	65%	76%
7	93%	41%	57%
8	93%	22%	35%
9	93%	7%	13%
10	90%	1%	2%

**Table 5.** Precision, recall, and F1 score for rhyme detection in 1,000 poems sampled from those marked as “rhyming” and “unrhyming” in the metadata of the Chadwyck-Healey poetry collections.

Statistical significance between degrees of rhyme across historical and generative verse samples is measured by permutation tests with 10,000 iterations, an approach which avoids any distributional assumptions. Effect sizes were calculated using Cohen’s  $d$  with pooled standard deviation, with effects characterized as small ( $d < 0.5$ ), medium ( $0.5 < d < 0.8$ ), or large ( $d > 0.8$ ) following conventional thresholds. For each comparison, the analysis computed observed differences in means, then generated a null distribution by randomly shuffling group assignments and recalculating differences. P-values represent the proportion of permuted differences with absolute values greater than or equal to the observed difference, yielding robust significance testing without parametric assumptions.

## 2.4 Results and Discussion

Operationalizing rhyme in this way, I estimated the percentage of rhyming poems across both historical and generative poetry corpora. The combined data on the frequency of rhymed verse in both is visualized in Figure 1.



**Figure 1.** Frequency of rhymed poems in the Chadwyck-Healey corpora compared with LLM-generated verse. Generative models were prompted for three types of poems: rhyming poems, unrhyming poems, and for poems without specifying whether to rhyme. Points indicate mean likelihood; size indicates the number of poems per data point; whiskers show standard error.

The left-hand portion of the graph shows that historically, across the 1,000 sampled poems per half-century in the Chadwyck-Healey poetry collections, the average likelihood for a poem to rhyme descends over time, particularly since the late nineteenth century. Prior to that, the historical incidence of rhymed verse hovers between 87% and 91% for three centuries, before turning away from it in the late nineteenth century (71%) and plummeting in the early twentieth (15%)—with ultimately only 7% of poems written by postwar poets in rhyme.

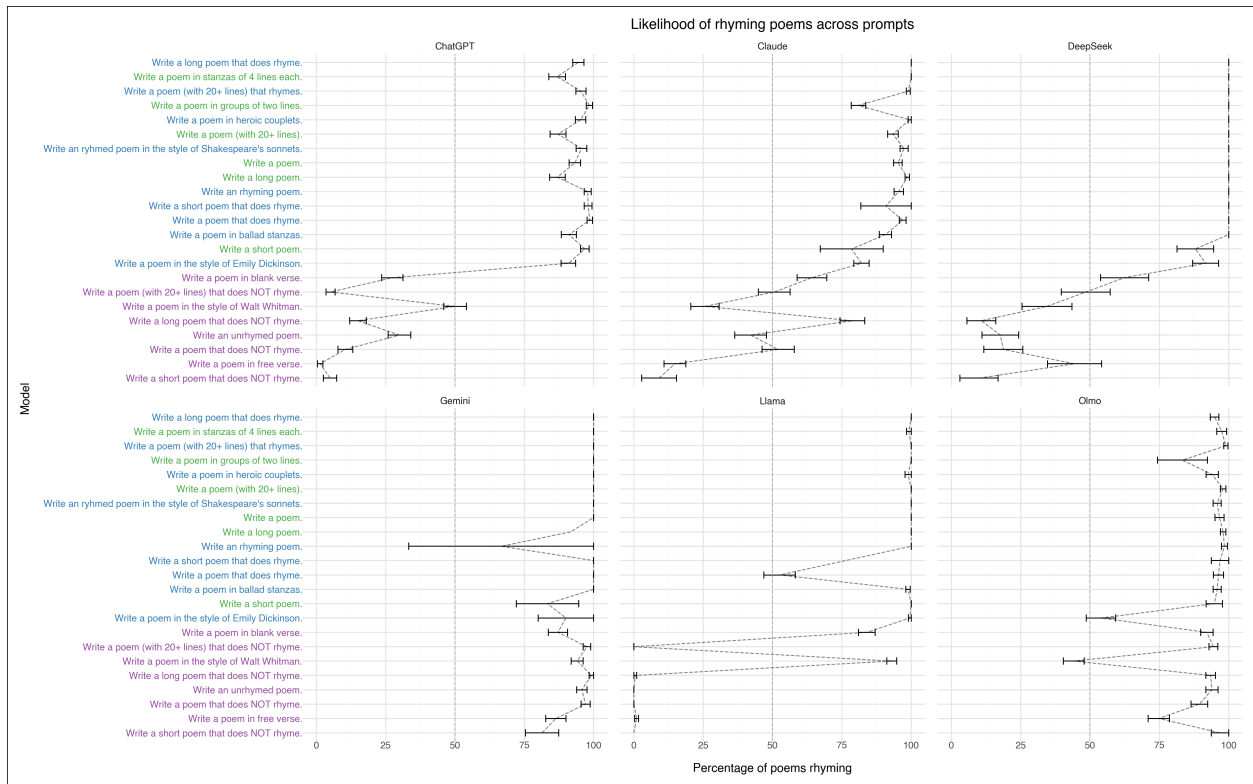
Meanwhile, the right-hand portion of the graph, a distant reading of rhyme not in historical but generative verse, shows several striking findings. On average across the LLMs, prompts for a rhyming poem (shown in blue) and prompts simply for a poem without specifying whether to rhyme (shown in green) produce a body of artificial verse composed of 95% rhyming poetry. Not only is this incidence more than in any historical period of actual human-authored verse, but its similarity across these two categories of prompts suggests a remarkably stubborn association between the genre of poetry and the form of rhyme within these models. Furthermore, whether prompted simply to write a poem, a rhyming poem, or an unrhyming poem (shown in purple), generative

models—despite the overwhelming bias in their training data toward the contemporary—rhyme more often than poets born in the late twentieth-century, and sometimes significantly more. Finally, even prompting the models for an *unrhyming* poem yields rhyming poems 53% of the time on average across the models, more than seven times as often as in the historical reality of postwar poetry (7%).

At the same time, significant discrepancies pertain across these models alongside their similarities. When not specifying whether to rhyme, models overwhelmingly produce rhyming verse, ranging only narrowly from 93% to 99% in likelihood. Yet when prompted not to rhyme, this range widens. Google’s Gemini Pro stubbornly rhymes 88% of the time even when instructed not to do so, as does the open-source model OLMo2 (83%). Other models are formally more obedient to their prompt: Anthropic’s Claude models rhyme 50% of the time; OpenAI’s ChatGPT models, 36%; DeepSeek, 31%; and Meta’s Llama, 25%. Nevertheless, all these frequencies of rhyme far exceed the past century of the form, to a highly statistically significant degree.

Such model-specific differences widen further when drilling down into the individual prompts composing each of three categories (Figure 2).





**Figure 2.** Frequency of rhyme in generative verse across a variety of prompts. Prompts categorized above as “write a rhyming poem” are shown in green; “write an unrhyming poem” in purple; and simply to “write a poem” in blue. Whiskers show standard error around the mean for each model and prompt.

Prompting for blank verse, for example, inaccurately produces rhymed verse in all models but ChatGPT, suggesting that this metrical form is comparatively unfamiliar to other models. At the same time, ChatGPT produces more rhyme in imitating Walt Whitman’s free verse than do models confused by that form’s explicit invocation, like Llama and DeepSeek. Notably, the open-weight Llama model successfully avoids rhyme outside of these genre-based instructions. Yet other models, like Gemini and (to a lesser extent) OLMo2, demonstrate extreme formal stuckness, appearing unable to escape rhyme regardless of the prompt. Taken together, however, these variations are idiosyncratic rather than systemic, reflecting different degrees of the same underlying tendency toward rhyme, and they ultimately serve to prove the same general formal rule.

Still, given these variations across models, one might question whether the bias toward rhyme stems from the explicit mention of poetic form in the prompts themselves. Prompting *not* to rhyme a poem, after all, still invokes the concept of rhyme; and indeed, one of the most difficult

aspects of working with LLMs is how sensitive its results are to the wording of its prompt. For this reason, I conducted a subsequent step to this experiment measuring rhyme in generative verse, this time avoiding the reliance on explicit prompting for rhymed and unrhymed verse by instead instructing models to “complete” existing poems which may or may not rhyme. For example, prompted with the first five lines of an unrhyming twelve-line poem from the 1990s, the Llama 3.1 LLM quickly veers back into rhyme (Figure 3).

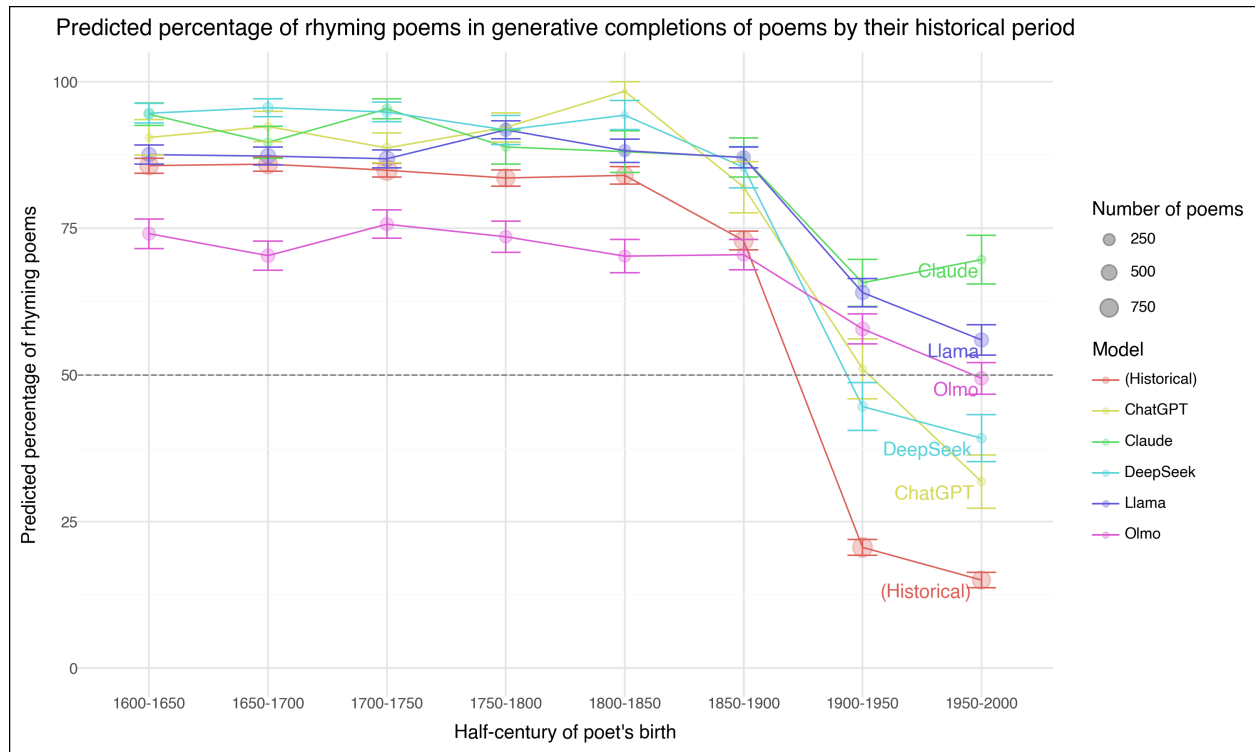
	My sister knows my name and nothing else. My mother knows my shoe size. My father knows what kind of books I read. My friends know where I drink. My husband knows the colour of my hair.	
	↙	↘
My children know my rules. My dentist knows I need another filling. My boss knows I work late. My cat knows that I'll share a turkey sandwich. I know what makes me laugh. I laugh at what they don't know. They don't know very much about my dentist.		And that's all they know But still they manage to annoy With comments unasked, and unwanted air Like knowing when I'm running low on tea Or guessing how much weight I used to be My family knows my deepest fears But even that doesn't stop their nagging tears.
--Hannah Sophie, 'Another Filling' (1996)		--AI imitation by llama3.1:8b

**Figure 3.** A generative completion of an unrhyming poem from the 1990s. The first five lines (above, center) are shown to the model, which is prompted to complete the poem either from memory or by imitating its style and theme. The model's completion is shown to the bottom right; the original poem continues to the bottom left.

The first three lines of this generative imitation (shown at bottom right) continue the poem's unrhyming form, with lines ending “know” and “annoy” (perhaps a slant rhyme) and “air”. But it then pivots into perfectly rhyming couplets, with lines ending “tea” / “be” and “fears” / “tears”. This formal pivot mirrors a thematic one, one that even brings it into contradiction with the poem's beginning: if “My sister knows my name and nothing else” (actual), how would “My family know my deepest fears” (imitation)? The actual conclusion to the poem (“I laugh at what they don't know. / They don't know very much about my dentist”) surprises the reader with a touch of irony and continued banality after setting up an expectation for something more profound—an expectation eagerly satisfied by “my deepest fears” in the generative model's completion of the poem.

Computing rhyme across all the generated completions of historical poems shows that lapses such as these from unrhymed into rhymed verse pervade generative poetic output. Fig-

ure 4 shows the pseudo-historical distribution of rhyme across the 8,571 generative completions of the 5,823 unique poems.



**Figure 4.** Frequency of rhymed verse across generative completions of historical poems. Points indicate mean likelihood; size indicates the number of poems per data point; whiskers show standard error.

Shown in red is the computed likelihood for rhyme in the sample of historical poems for which the first five lines were passed to the generative models. Here, as in Figure 1, historical verse more or less steadily rhymes until the nineteenth century before plummeting in the twentieth. The generative completions of these poems also steadily rhyme when prompted to complete pre-twentieth century poems, although this success may be owing more to their overall propensity to rhyme than their formal responsiveness. The models also do seem to respond to rhyme’s near disappearance in twentieth-century verse by returning less rhymed verse. But no model reflects the extent of this dissipation. Statistically, compared with postwar poets, all LLMs produce significantly more rhyme in their completions thereof. Cohen’s  $d$ , a measure of the difference between means in terms of their standard deviations, shows a significantly large difference or “effect size” for Claude ( $d = 1.46$ ), Llama (0.98) and OLMo (0.84); a medium effect size for DeepSeek (0.63); and small

effect size for ChatGPT (0.45), with all such differences statistically significant. In terms of the literary history of rhyme, then, generative verse appears formally retrograde—as if formally stuck somewhere between the nineteenth and twentieth centuries.

It should be noted that this formal compulsion to rhyme has not gone unnoticed by other researchers. Melanie Walsh, Anna Preus, and Elizabeth Gronski have recently published an analysis of poetry generated by ChatGPT, revealing in a sample of several thousand human- and AI-authored poems that around 90% of LLM-generated poems contained rhyme, compared to 65% within human poems (Walsh, Preus, and Gronski). Annotating samples of verse for metrical and stanzaic forms, Walsh, Preus, and Gronski also show that ChatGPT-authored verse tends strongly toward rhymed iambic quatrains. “There is a sort of default poetic mode in GPT models which favors quatrains, iambic meter, and end rhyme,” they write. Not just default, but stuck: “The models can be prompted to produce writing in other styles, but sometimes the persistent iambic/quadrain/end rhyme style still breaks through.” My own experiment on rhyme in LLM verse corroborates and contextualizes these results—using different prosodic software, historical and poetic corpora, LLM models and prompting methods—while also devoting attention to the historicity of generative verse form. It shows, moreover, how even when prompting *against* such forms, these models exaggerate their tendency beyond any historical precedent.

Is rhyme, then, a kind of “42” for generative verse? Unlike the clear cultural reference point of “42” from *The Hitchhiker’s Guide to the Galaxy*, the compulsion to rhyme doesn’t stem from an obvious source. Despite being trained on orders of magnitude more contemporary language, LLMs show a strong formal bias towards more traditional genres and practices of rhyme which, intuitively, would seem more likely to appear in historical, out-of-copyright corpora. Yet in one well-documented LLM training dataset—Dolma, used to train the open OLMo2 model included in this study—out-of-copyright books from Project Gutenberg make up just 0.16% of the total number of words in the corpus, whereas text scraped from the internet made up 84% (Soldaini et al.). Traditional verse still appears on the web in collection websites such as *engpoetry.com*, *poetry.com* and *poemhunter.com*, thousands of poems from which have been shown to appear in

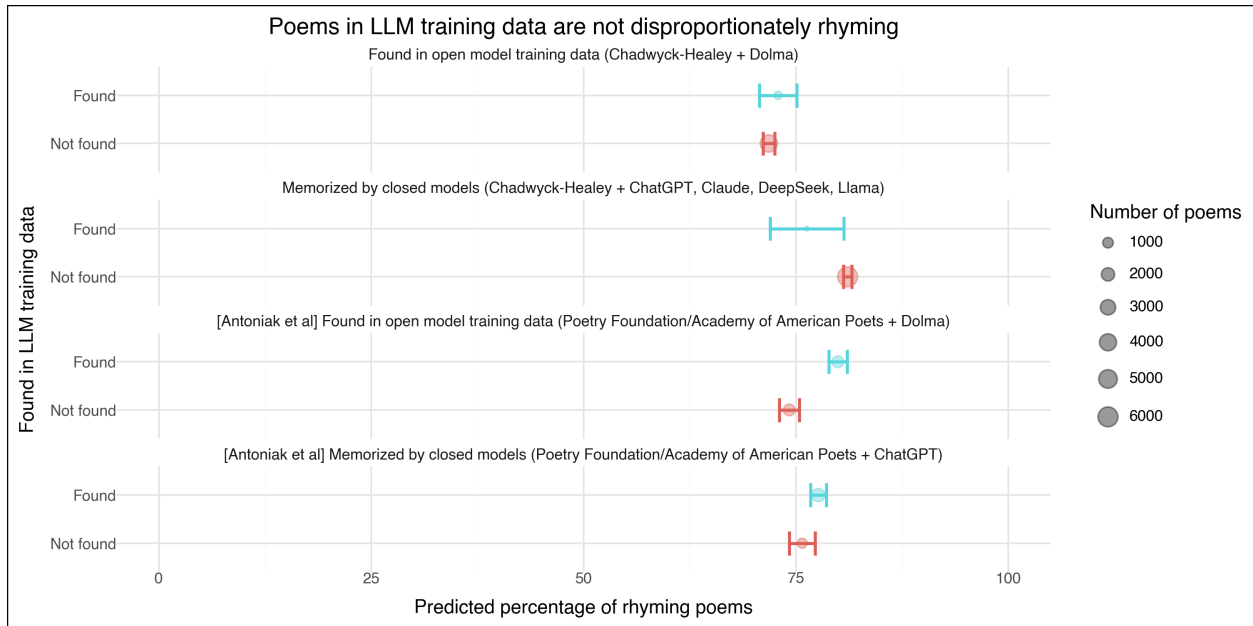
LLM training data (Walsh, Preus, and Antoniak). Atavistic forms of rhyme and rhythm may also thrive more than critics have realized in an online afterlife, on amateur writing forums or educational websites emphasizing traditional poetic forms—a literary sphere more akin to what Merve Emre calls the “paraliterary” (Emre).

But replicating existing methods of detecting poems in LLM training data shows that poems found there are not disproportionately rhyming. Here I draw on Walsh, Preus, and Antoniak’s method of querying the Allen Institute’s “What’s In My Big Data?” (WIMBD) tool to search the open Dolma dataset for the presence of lines of verse from a poem as a proxy for its presence in common LLM training data. Though its results depend on the sample of poems for which one searches, this procedure helpfully allows one to explore the poetic composition of open model training data. I therefore queried WIMBD for a sample of 4,635 poems from Chadwyck-Healey by historical period. To examine closed model training data, I draw on D’Souza and Mimno’s method of prompting models to complete a given poem to detect whether it has “memorized” its lines. Here I take the poem completion data shown above in Figure 4, this time not excluding the ‘memorized’ poems—defined as any poem with a generated line having a fuzzy string match similarity above 95% with its corresponding line in the original—but instead counting how many were and were not so memorized. The number of poems per historical period found and not found in the open training data (WIMBD) and closed models (“memorized”) is shown in Table 6.

Period	Closed model output		Open training data	
	Found	Not found	Found	Not found
1600-1650	33	499	17	830
1650-1700	39	435	28	879
1700-1750	21	444	7	966
1750-1800	75	566	26	685
1800-1850	57	644	5	499
1850-1900	107	551	7	646
1900-1950	55	545	6	779
1950-2000	19	545	1	754

**Table 6.** Number of poems found in open model training data (using WIMBD) and closed model output (using “memorization” detection).

For comparison, I also reproduce similar results from Walsh, Preus, and Antoniak, who detect poems in both open and closed models by the identical procedure, in their case by querying for a more canonical sample from the Poetry Foundation (PF) and Academy of American Poets (AAP). Then, for all four cases (i.e., Chadwyck-Healey in open and closed datasets; PF/AAP in open and closed datasets), I measure rhyme in the poems discovered by these methods to be present or absent in the training data (Figure 5).



**Figure 5.** Frequency of rhymed verse in poems detected in open and closed LLM training data. Points indicate mean likelihood; size indicates the number of poems per data point; whiskers show standard error.

Across all methods and datasets, the distribution of rhyme in poems found in LLM training data (shown in blue) is not disproportionately high compared to poems not found in that data (shown in red). A comparison of means between found and unfound poems' rhyming shows statistically insignificant differences in all cases, besides the open training dataset queried for canonical poems from PF/AAP; yet even this shows a Cohen's  $d$  effect size under the standard threshold of significance (0.14). Moreover, though very few poems from the historical corpora used in this study were found in closed-source LLM training data, those found even rhymed less often than those not found. These results suggest that LLMs' tendency to overuse rhyme cannot be explained simply by an overrepresentation of rhyming poetry in their training data. Rather, this data points

to something more fundamental in how these models process and generate poetic forms.

In this first experiment, then, we begin to see how models obsessively reinstate outmoded poetic forms like rhyme in apparent defiance of their largely contemporary training data. LLMs flatten historical variation into an idealized representation of poetic form, resurrecting conservative formal choices that contemporary verse has largely abandoned. That rhyme in LLM verse persists even when explicitly forbidden provides additional evidence of a deep formal compulsion—revealing how AI systems, for all their supposed flexibility, do not simply reproduce aesthetic constraints but selectively reinforce them. Moreover, that this formal rigidity cannot be fully explained by training data points to something deeper in the computational mechanics in these models. In their ahistorical reification of traditional forms, generative models betray not only technical limitations but also deep patterns in how their computational logic flattens and reifies cultural history.

### **3 Generative Rhythm**

#### **3.1 Background**

Building on these findings, we can turn to a second experiment to unpack subtler formal compulsions in AI-generated verse. Rather than simply measuring how frequently LLMs employ conventional forms like rhyme, we can examine the peculiar ways by which they implement these conventions. The previous experiment demonstrated LLMs’ formal stuckness on rhyme; the following unpacks a subtler kind of stuckness in their unusually metronomic execution of poetic meter. I will argue that this rhythmic hyper-regularity reveals how generative models not only flatten historical variation, but how they also reshape it according to a computational logic privileging the satisfaction of formal expectation over the tensions and variations that characterize historical literary form, thus betraying what I will call a computational aesthetic of “idealization.”

To hold as constant as possible the choice of metrical form in generative verse in order to isolate its rhythmic execution, this second experiment focuses specifically on generated sonnets instead of “poems,” a word far less determinative of the details of poetic or metrical form. The sonnet, of course, adapted from Italian models in the sixteenth century by poets like Thomas Wyatt

and Henry Howard, Earl of Surrey, has long been one of the most recognizable and enduring poetic forms in English verse. Surely, after ingesting trillions of words in training, LLMs are sufficiently exposed to this well-known convention: namely, that a sonnet is a fourteen-line poem of iambic pentameter organized into rhyming quatrains and couplets or octets and sestet. What remains, then, is a comparison of the *execution* of these conventional forms by poets and LLMs.

### 3.2 Data

To compare historical and generative sonnets along with the 154 sonnets by Shakespeare, I first sampled up to 154 sonnets per century in the Chadwyck-Healey corpus, drawing on its poetic genre metadata. I also generated 150 sonnets with a mixture of six LLMs using the prompt: “Write a sonnet in the style of Shakespeare’s sonnets.” These sonnet-prompted poems were not included in the previous experiment on rhyme. The full numbers per model and century are shown in Table 7.

Source		# Sonnets
Historical	Shakespeare	154
	C17	135
	C18	154
	C19	154
	C20	38
LLM	Claude-3-Haiku	65
	Claude-3-Opus	6
	Claude-3-Sonnet	13
	ChatGPT-3.5-Turbo	50
	ChatGPT-4-Turbo	13
	Gemini-Pro	3

**Table 7.** *Number of sonnets sampled from the Chadwyck-Healey corpus and generated by LLMs.*

### 3.3 Methods

To analyze the rhythm and meter of sonnets, I used Prosodic—a metrical-phonological parser written in Python—which draws on a combination of pronunciation dictionaries and text-to-speech software to annotate lines of verse for both syllable stress (i.e., its rhythmic execution) as well as for those stresses’ conformity with the line’s most plausible metrical scansion (i.e., its implied



rhythmic expectation). For metrical scansion, Prosodic implements a well-known theory of stress-meter correspondence drawn from metrical phonology. This theory, “A Parametric Theory of Poetic Meter” by Kristin Hanson and Paul Kiparsky, proposes a parsimonious grammar for metrical language such that distinct poetic traditions might be said to adopt and discard certain metrical rules or “correspondence constraints” (Hanson and Kiparsky). Prosodic encodes this theory via a computational model of grammatical rule-weighting popular in phonology known as Optimality Theory (OT). It first measures the extent to which all logically possible scansions violate a range of hypothesized grammatical rules for meter, such as StrongUnstress (don’t metrically accentuate an unstressed syllable), WeakPeak (don’t metrically de-accentuate a stressed syllable of a polysyllabic word), and others. Without committing a priori to any specific combination or “weighting” of these rules, OT eliminates implausible or “harmonically bounded” scansions by identifying those that violate the same rules *and more* of others. If only one metrical scansion remains, the line can be said to be unambiguously parseable according to that scansion.

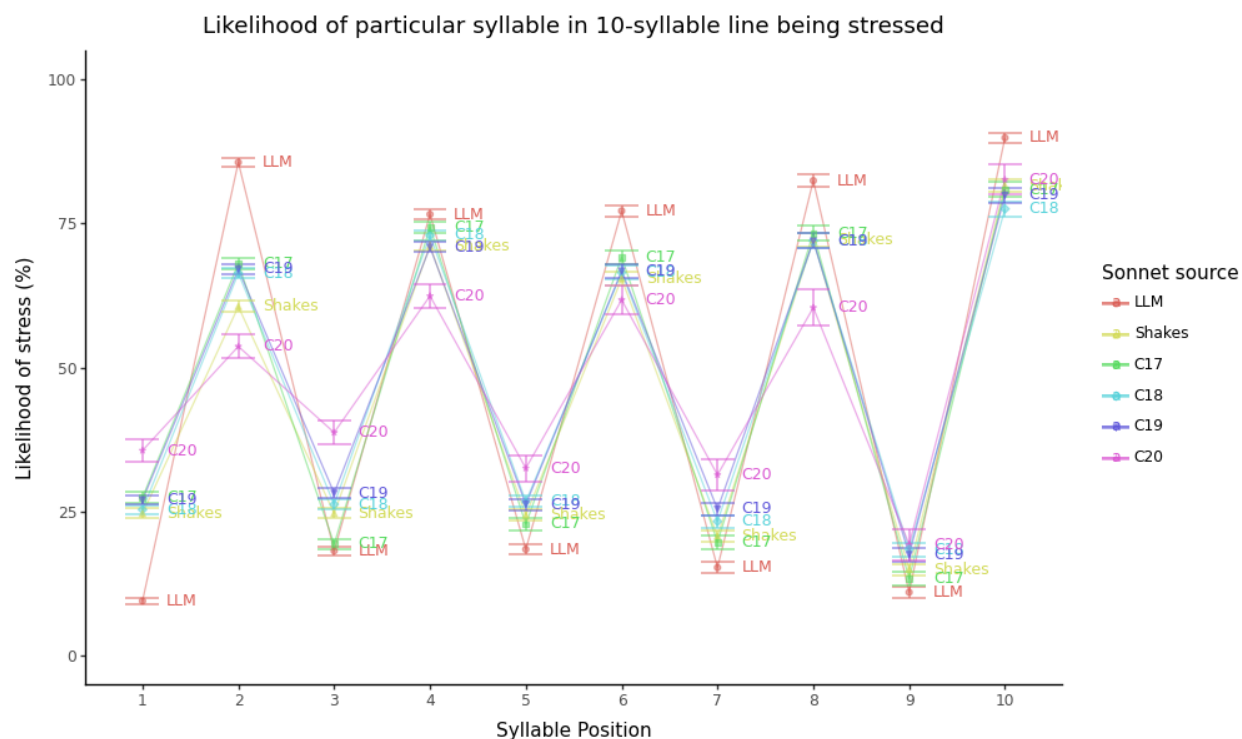
For example, Robert Frost’s *When **far** | **away** | **an int** | **-errupt** | **-ed cry***—where syllable stress matches metrical accent perfectly, thus violating no metrical rules; in other words, there is no alternative plausible scansion, given that any other distorts the accents within the polysyllabic words, breaking WeakPeak. By contrast, his “I have been one acquainted with the night” might plausibly be parsed as iambic pentameter (*I **have** | **been one** | **acquaint** | **-ed with** | **the night***) or anapestic tetrameter (*I have **been** | **one acquaint** | **-ed with** | **the night***). In this experiment, “unambiguous iambic pentameter” is defined as a line having only -+-+-+ as its “plausible,” or harmonically unbounded scansion.

### 3.4 Results and Discussion

How strictly observed is the rhythm of generative sonnets’ iambic pentameter, and how does it compare with sonnets at different moments of its own literary history? Do LLMs write lines closer to *From **fair** | **-est crea** | **-tures we** | **desire** | **increase***, for which no alternative parse—like the trochaic ***From** **fair** | **-est crea** | **-tures we** | **desire** | **increase***—offers a plausibly competing scansion? Or do they write lines closer to *Feed’s **thy** | **light’s flame** | **with self** | **-substan** | **-tial***

*fuel*, which might plausibly also be scanned with a trochaic inversion, beginning *Feed'st thy | light's flame?*

Even before attempting to metrically scan these syllable stresses, Figure 6 shows how generative sonnets arrange their unstressed and stressed syllables more regularly than in any known century of the form.

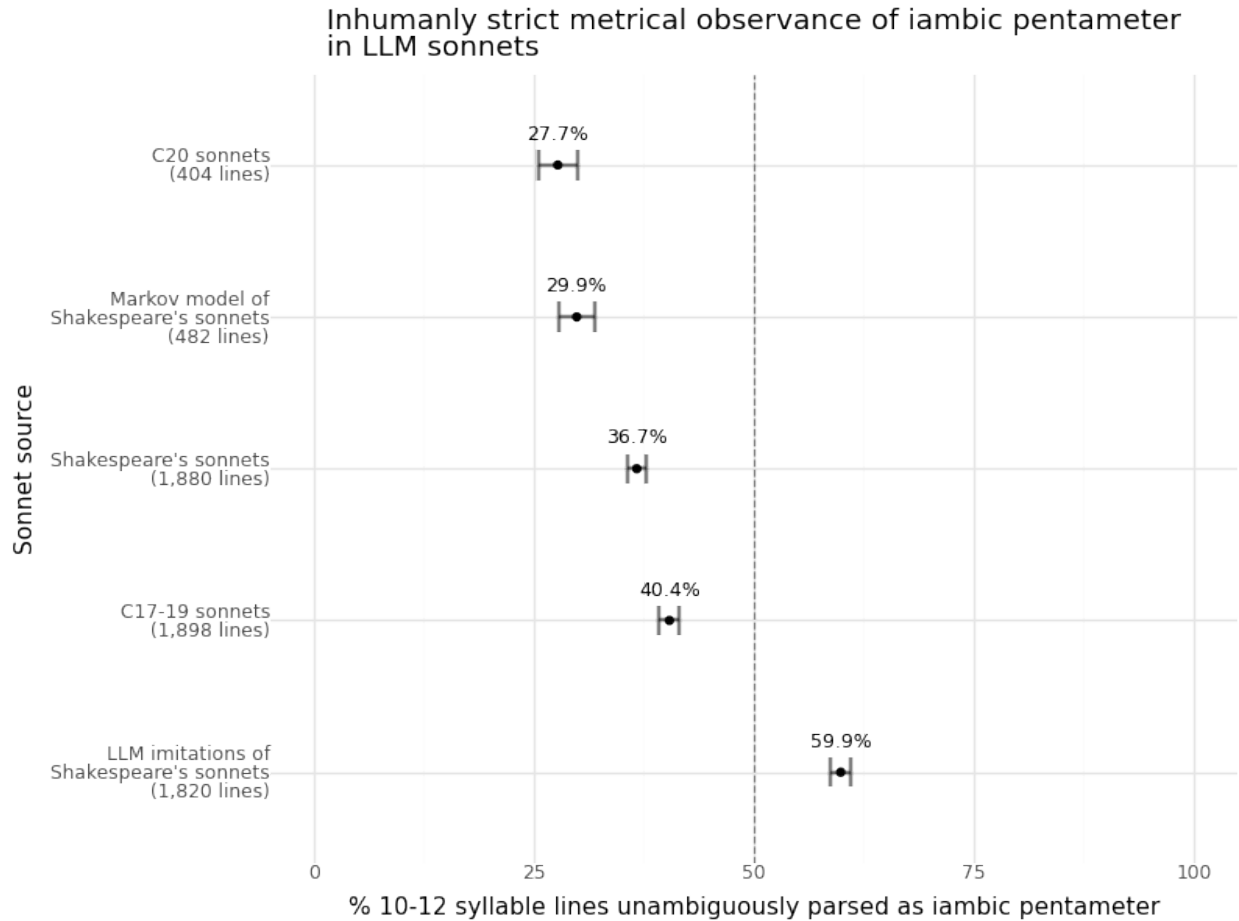


**Figure 6.** Frequency of syllable stress per syllable position into 10-syllable lines, drawn from historical and generative sonnets. Points indicate mean likelihood; whiskers indicate standard error.

That LLM imitations of Shakespeare's sonnets (in red) dip much lower in syllable stress likelihood in the odd-numbered syllables and peak much higher in the even-numbered ones—more than any other sonnet source, including Shakespeare's own—shows how inhumanly rigidly its syllable stresses adhere to an iambic pentameter metrical template. The second syllable in the line, for example, is stressed not much more often than chance in twentieth-century sonnets (54%), owing to the dominance of free verse; only somewhat more likely in Shakespeare's (61%); and about two-thirds of the time in pre-twentieth-century sonnets (66 to 68%), partly given the more frequent use of traditional metrical variations like the trochaic inversion. But LLM verse stresses this syllable

a striking 86% of the time, far exceeding the sonnets of any historical period from the seventeenth through the twentieth centuries. Although this syllable shows the largest difference between generative and historical sonnet meter, and although all sonnet sources rise in regularity toward the end of the line (a well-known phenomenon in metrics), generative sonnets maintain an historically unprecedented regularity in syllable stress across the line.

Syllable stress, however, is only the surface of poetic meter, itself an abstract template of rhythmic expectation that can accommodate wide variation in stress arrangement. Indeed, much of what we experience as pleasing poetic rhythm involves not only meeting expectations of regularity but artfully frustrating them. For this reason, a more nuanced exploration of meter in generative and historical verse must extend beyond the descriptive statistics of syllable stress into the explanatory modeling of poetic meter. The following results therefore indicate how strictly and unambiguously LLM-generated sonnets obey the formal conventions of iambic pentameter by examining the average likelihood of lines in sonnets to be unambiguously parseable as iambic pentameter. Figure 7 below shows this likelihood by sonnet source.



**Figure 7.** Frequency of unambiguously iambic pentameter lines in historical vs. generative sonnets. Points indicate mean likelihood; whiskers indicate standard error.

The results reveal a striking pattern: LLM imitations of Shakespeare’s sonnets produce unambiguous iambic pentameter lines nearly 60% of the time, far exceeding the rates found in historical sonnets. This mechanical regularity contrasts sharply with Shakespeare’s own sonnets, where only 37% of lines are unambiguously in iambic pentameter, as well as with sonnets from the seventeenth through nineteenth centuries (40%). LLM-generated sonnets produce perfect pentameter roughly twice as often as twentieth-century sonnets (28%), an “improvement” of its iambic pentameter by a factor of 2. By comparison, seventeenth- to nineteenth-century sonnets “improve” upon twentieth-century ones by a factor of 1.5, meaning that LLMs’ mechanical regularity exceeds even a rewinding of history from free verse to metrical tradition.

In learning the formal conventions of the Shakespearean sonnet across the (no doubt) many

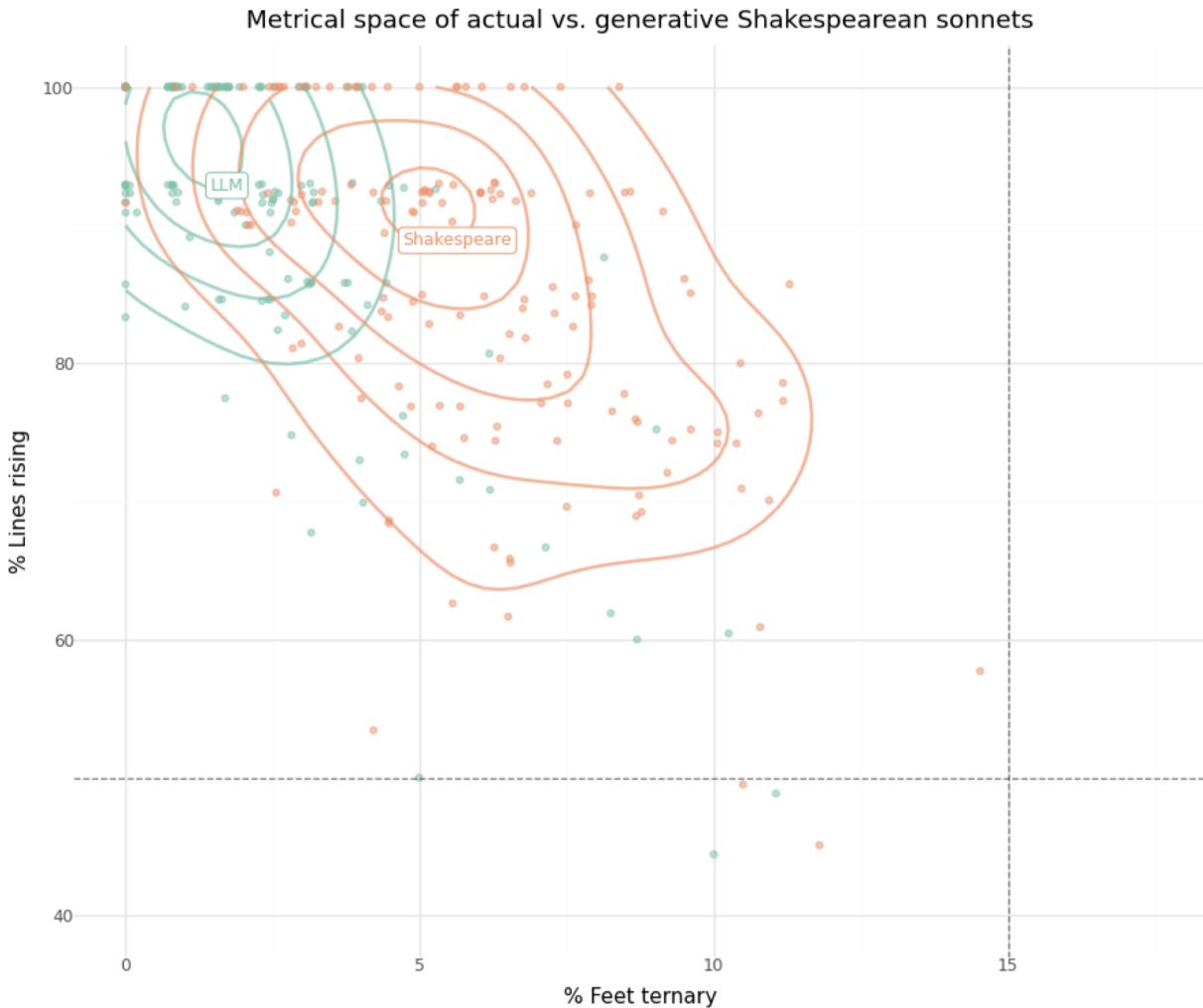
thousands of examples in its training data, these models appear to have forgotten even the conventional exceptions to metrical rules that characterize the genre. As Ben Glaser and I explore in preliminary collaborative work, AI verse systematically erases conventional formal variations like enjambments and trochaic substitutions. In the same corpus of sonnets plotted above, trochaic inversions begin 12.9% of Shakespeare's lines but appear in only 1.2% of their LLM-generated imitations—an order of magnitude reduction in this well-known metrical variation. Such results suggest that LLMs' tendency toward pattern conformity and regularity proceeds, paradoxically, through a process of “forgetting” even the traditional exceptions and variations of the pattern it so obsessively reproduces.

A final step of this experiment concerning the metrical features of LLM-generated sonnets explores the narrowness of the formal space they occupy. Here I draw on previous collaborative research (2014–2018) from the Stanford Literary Lab, documented by J.D. Porter in an online lab publication, “The Space of Poetic Meter.”<sup>8</sup> This work plots poems said to be “in” iambic, trochaic, anapestic, and dactylic meters along two quantitative axes that together constitute these traditional metrical categories: foot headedness (whether a foot is “rising” in stress as in iambs and anapests, or “falling” as in trochees and dactyls) and foot size (binary as in iambs/trochees or ternary as in anapests/dactyls). The four corners on this two-dimensional space represent perfect implementations of a specific meter: entirely rising and binary (iambic), entirely falling and binary (trochaic), and so on. But the distribution of these poems across the metrical space revealed a hidden formal diversity: although poems marked by critics as iambic gather near the corner of perfect iambicity, they do not all occupy it precisely and instead range across the relevant quadrant of the space.

Where does generative verse fall within this metrical space? Measuring metrical regularity in generative AI imitations of Shakespeare's sonnets, Figure 8 plots these generative texts against Shakespeare's actual sonnets:

---

<sup>8</sup> See Porter. The project included myself, J.D. Porter, Justin Tackett, Jonathan Sensenbaugh, Mark Algee-Hewitt, and Maria Kraxenberger.



**Figure 8.** Metrical space occupied by Shakespeare's sonnets and LLM-generated imitations of Shakespearean sonnets. The upper left corner represents perfect iambicity: the farther left on the  $x$ -axis, the more binary the meter; the farther up on the  $y$ -axis, the more rising the meter. Points indicate individual poems; lines show a two-dimensional kernel density estimation.

Strikingly, LLM sonnets (in teal) occupy a metrical space far more concentrated than Shakespeare's own. Their distribution across the metrical space occurs both in a tighter range (a kind of iambic precision) and closer to the point of total regularity in iambic meter (a kind of iambic accuracy). Prompted to imitate Shakespeare's sonnets, and here compared with the imitated originals, these models' hyper-regular verse again underscores the tendency of these models to produce formally conservative poetry that adheres more rigidly to the metrical forms they imitate.

LLMs' strict adherence to metrical convention produces verse that is more rigidly regular

than any historical precedent, in its rhythm and its rhyme. Though logically distinct, the formal hyper-regularity found in this second experiment on generative rhythm synchronizes with the formal anachronism of the previous experiment on generative rhyme. Both suggest a computational aesthetic that privileges the satisfaction of formal expectation over its frustration, the abstraction of patterns over their concrete variation, and the steady observance of rules over their strategic violations. Together, these experiments suggest something deeper about the nature of generative systems: not simply how they process individual domains like poetic form but how they fundamentally reshape cultural material through computational logics of perfection, satisfaction, and delight. Understanding these logics requires us to examine not just what these systems generate, but how their peculiar forms of idealization refract the historical contradictions they process into a computational aesthetic of their own mechanical design.

## 4 Generative criticism

How does one begin to explain the formal compulsions in rhyme and rhythm outlined above? From a technical point of view, many aspects of these models may bear varying levels of responsibility. We have already seen that formal biases in the underlying training data, the most obvious such aspect, cannot adequately explain the formal biases of the models' output. In exploring issues of formal stuckness in generative literature, digital humanists must expand beyond questions of training data to examine the social and economic conditions of LLMs' production in the extractive practices of globalized digital labor in reinforcement learning human feedback (RLHF), as well as the computational architecture and learning processes involved in model development.

For example, preliminary experiments show that “instruction-tuning,” responsible for converting the raw next-token prediction “base models” into helpful chatbots and AI assistants, may have aesthetic side effects. Even the same model (Llama 3.1), pretrained on the same data, displays a greater propensity toward rhyme after instruction-tuning than before it. To calculate this comparison for the instruction-tuned version of the model (i.e., “llama3.1:instruct”), I used the same prompting process used above in Figure 4, in which the first five lines from poems across

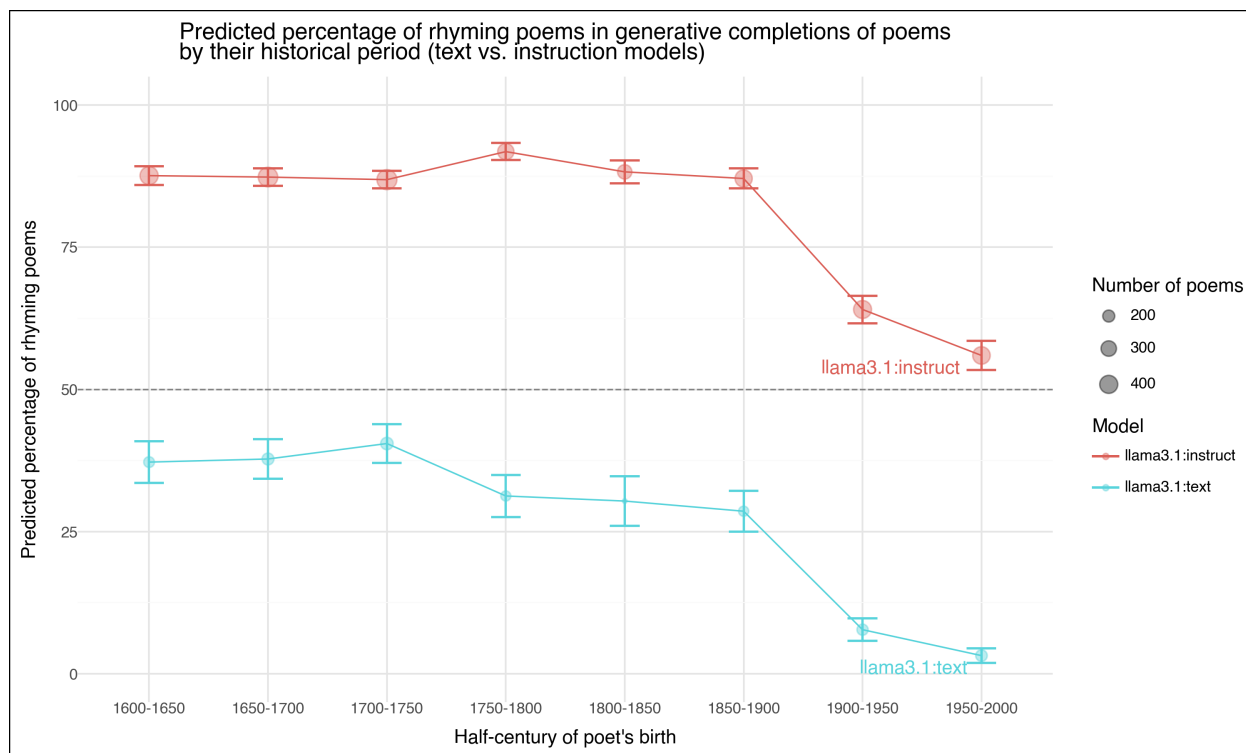
history are presented to models, which are instructed to “complete the poem”. The “base model” version (i.e., “llama3.1:text”), which does not respond to instructions but simply generates next tokens for the supplied context, was also shown five lines from a poem and its total number of lines. In this case, instructions were not provided, as the inherent autocompleting tendency of the model sufficed to continue the poem. The number of poems generated by both versions is shown in Table 8.

	llama3.1:instruct	llama3.1:text
1600–1650	403	172
1650–1700	466	196
1700–1750	488	210
1750–1800	330	160
1800–1850	264	112
1850–1900	364	161
1900–1950	395	181
1950–2000	377	189

**Table 8.** *Number of poem “completions,” per historical period of original poem, generated by llama3.1:instruct and llama3.1:text.*

Figure 9 below shows that llama3.1:instruct rhymes far more often than its identically-trained llama3.1:text.





**Figure 9.** Frequency of rhymed verse across generative completions of historical poems, one “instruction-tuned” and the other a raw next token generator. Points indicate mean likelihood; size indicates the number of poems per data point; whiskers show standard error.

Comparing the frequency of rhymed verse in these two variants of the same model shows that the raw text completion variant rhymes far less (with a statistically significant difference of means across all historical periods of the completed poems) than the one tuned to respond helpfully to user instructions. Does this helpful responsiveness push them, by accident, into more idealized aesthetic forms like rhythm and regular rhythm?

Perhaps Knapp and Michael were more perceptive than they intended to be when they commented on their aesthetic embarrassment with the models’ cloying earnestness, its servile attempts to please, its inability for irony, and that it signs its poem, “I hope you like it!” For the model, is the poetic form of rhyme, its quatrains and “streams” and “clouds,” its perfect regularity in meter, a kind of formal consequence of a larger aesthetic drive or compulsion... to please? Do they emphasize what they consider pleasing literary forms, or represent them in a more pleasant or idealized manner, without capacity for irony? *Docere, delectare*: certainly, the motive to please

ought not to be underestimated in artistic production, the Ciceronian goal of oratory “to delight” as well as “to instruct.” Even the goal of instruction seems compulsively attended to in LLM verse, as evidenced by the frequent “we” and “us” one finds in generative poems, as in the poem that Knapp and Michael generate, which ends “They bring us joy and sorrow too, / But always leave us feeling fulfilled.” This unusual frequency of first-person plurals has also been confirmed recently in a quantitative study (Walsh, Preus, and Antoniak).

Do generative models themselves have an aesthetics? Whether caused by training data or model architecture or both, do they nevertheless constellate aesthetic goals of pleasure and instruction, aesthetic tendencies of idealization over irony? Such questions deserve further investigation. The compulsive formalism we’ve observed—the rigid adherence to rhyme and meter, the earnest didacticism, the drive to please—may point to an underlying aesthetic sensibility embedded in these systems. Understanding these aesthetic tendencies and their origins can help us better grasp both the capabilities and limitations of AI-generated texts and images, while raising deeper questions about the relationship between computation and aesthetics.

Accordingly we might begin by asking where formal stuckness inheres—culturally and computationally, literarily and critically—and what effect will it have when put into such frequent use? Is literature present in LLM training data itself stuck in certain periods or forms, and is such literary training data primarily historical or contemporary, professionalized or amateur? Are online writing manuals critical discussions present in this data responsible for outmoded formal concepts of, for example, what a poem is as a piece of language? Or does instruction-tuning a large language model into a helpful chatbot produce aesthetic byproducts of formal idealization and regularity? The very prospect of examining these questions suggests new avenues for historical formalism, even in a new regime of an ahistorical generative textuality. As we disarticulate LLMs’ hidden formal and historical tendencies, the fact that radically contemporary generative texts might nevertheless be revealed to be historically and formally out of joint with their contemporary contexts—“stuck”—suggests new modes of digital humanities: new ways of reading, both distant and close, for the generative mediations of historical forms.

A generative digital humanities will need to confront these questions through multiple modes of investigation. One method will be to examine LLM training data as a curiously transhistorical archive, documenting the sources, genres, and distributions of literary texts and forms that shape these models' outputs and offering new opportunities for the study of our cultural present and its potential futures. But perhaps less intuitively, another method might be a kind of inverted historical formalism that seeks to recover and re-historicize the flattened and ahistorical forms found in an emerging generative archive. If history, as critics like Jameson argue, is the "absent cause" of textual form, with literary forms like the rhyming ballad mediating real contradictions in the social order, generative models' obsessive tendency toward particular historically determined forms suggests a fundamental perversion of history's role within generative textuality. History, now doubly absent—first in its invisibility outside textual representation, second in its abstraction within generative representation—nevertheless beckons for critical reckoning through the very formal patterns it leaves as traces in these generative artifacts.

As attempted above, the project of generative formalism extends the methods and insights of an earlier quantitative formalism to generative cultural production. In a sense more "distant" than distant reading itself, generative formalism conducts a murkier deep-sea excavation project in search of the "abstract pattern reveal[ing] the true nature of the historical process"—not directly in the history of literary production but indirectly through history's distorted reappearance in generative texts (Moretti 29). As generative AI systems continue to evolve and proliferate, this double absence of history, and the formal traces it leaves, becomes increasingly central to understanding contemporary cultural production. In this way, generative formalism offers digital humanities a concrete method for examining AI's cultural impact. By attending to the formal traces of absent history in generative texts, we can begin to understand not just how these systems work, but how they reshape the very nature of cultural reproduction. Where traditional digital humanities sought to recover lost histories and forgotten forms, generative formalism must now excavate the distorted shapes of those histories and forms in AI's computational unconscious. Through careful analysis of formal stuckness, we can begin to map the hidden aesthetic and historical logics driving gen-

erative culture. In this way, the traditional tools of digital humanities—its attention to form, its computational methods, its critical and historical consciousness—become newly vital for a new, “generative humanities” charged with understanding and responding to our increasingly generative cultural future.

## Works Cited

- Allison, Sarah, et al. Quantitative Formalism: An Experiment. *Stanford Literary Lab Pamphlets*, 1, Jan. 2011. [litlab.stanford.edu/LiteraryLabPamphlet1.pdf](http://litlab.stanford.edu/LiteraryLabPamphlet1.pdf). Accessed 15 Jan. 2025.
- Anttila, Arto, and Ryan Heuser. “Phonological and Metrical Variation across Genres.” *Proceedings of the Annual Meetings on Phonology*, 2015. Accessed 4 Nov. 2024.
- Bassett, Caroline. “The Computational Therapeutic: Exploring Weizenbaum’s ELIZA as a History of the Present.” *AI & Society*, vol. 34, no. 4, 2019, pp. 803–12.
- Bender, Emily M, et al. “On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?” *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. ACM, Mar. 2021.
- D’Souza, Lyra, and David Mimno. “The Chatbot and the Canon: Poetry Memorization in LLMs.” *CHR 2023: Computational Humanities Research Conference*, 6–8 Dec. 2023.
- Emre, Merve. *Paraliterary: The Making of Bad Readers in Postwar America*. U of Chicago P, 2017.
- Hanson, Kristin, and Paul Kiparsky. “A Parametric Theory of Poetic Meter.” *Language*, vol. 72, 1996, pp. 287–335.
- Heuser, Ryan. “Prosodic,” [github.com/quadrismegistus/prosodic](https://github.com/quadrismegistus/prosodic). Accessed 29 Sept. 2024.
- Heuser, Ryan, and Meredith Martin. “Historical Prosody and Mechanical Form.” *ACLA*, 2024.
- Jameson, Fredric. *The Political Unconscious: Narrative as a Socially Symbolic Act*. Psychology P, 2002.
- Kirschenbaum, Matthew. “Again Theory: A Forum on Language, Meaning, and Intent in the Time of Stochastic Parrots.” *In the Moment (Critical Inquiry Blog)*, 26 June 2023, [critinq.wordpress](https://critinq.wordpress.com).

com/2023/06/26/again-theory-a-forum-on-language-meaning-and-intent-in-the-time-of-stochastic-parrots/. Accessed 14 Aug. 2025.

Moretti, Franco. *Graphs, Maps, Trees: Abstract Models for a Literary History*. Verso, 2005.

Ngai, Sianne. *Theory of the Gimmick: Aesthetic Judgment and Capitalist Form*. The Belknap P of Harvard UP, 2020.

Porter, J.D. The Space of Poetic Meter. *Stanford Literary Lab*, Mar. 2018. Accessed 15 Mar. 2025.

Read, Max. Drowning in Slop. *New York Magazine: Intelligence*, Sept. 2024. Accessed 18 July 2025.

Soldaini, Luca, et al. “Dolma: An Open Corpus of Three Trillion Tokens for Language Model Pretraining Research.” *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Edited by Lun-Wei Ku, Andre Martins, and Vivek Srikumar, Association for Computational Linguistics, Aug. 2024, pp. 15725–88.

Specht, Juan Ignacio. “42: GPT’s Answer to Life, the Universe, and Everything.” *Leniolabs*, 4 Oct. 2023, leniolabs.com/42-gpt-answer-to-life-the-universe-and-everything/. Accessed 14 Aug. 2025.

Walsh, Melanie, Anna Preus, and Maria Antoniak. “Sonnet or Not, Bot? Poetry Evaluation for Large Models and Datasets.” *Findings of the Association for Computational Linguistics*. Association for Computational Linguistics, Oct. 2024.

Walsh, Melanie, Anna Preus, and Elizabeth Gronski. “Does ChatGPT Have a Poetic Style?” CHR 2024: Computational Humanities Research Conference, 20 Oct. 2024, arxiv.org/abs/2410.15299. Accessed 26 Oct. 2024.