Episode Book

EPISODE #5482

When AI Edits Your Words: The Off Switch Problem

PUBLISHED	23 September 2026 at 11:41 UTC
DURATION	26:16
LISTEN	myweirdprompts.com/episode/ai-prompt-over-editing-fix/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.2
BOOK GENERATED	19 October 2026 at 17:36 UTC

SYNOPSIS

Daniel built an AI agent to lightly edit his podcast prompts, and it works. So why does he keep turning it off? This episode digs into the measurable mechanics of AI over-editing — from a 2026 coding-model study showing reasoning models rewrite more by default, to brevity bias and context collapse that erode a prompt across iterations. We walk through the draft system prompt in full, find the broken worked example hiding inside it, and map three production patterns that fix the problem: surgical find-and-replace editing, formal drift bounds with length caps, and delta-based merges instead of full rewrites. If you've ever gotten back something functionally correct that no longer sounds like you, this one's for you.

TRANSCRIPT



Corn

There's a particular kind of nervousness that comes from watching a machine touch your words. Not replace them. Touch them. Daniel knows it well enough that he built an off switch and uses it.



Herman

He turns it off even though it works.



Corn

He turns it off *because* it works, most of the time. Here's what he wrote in. He's got an agent sitting at the very front of the episode generation pipeline, before anything else touches his prompt. It's meant to be a light touch editor. Fix the typos, strip the "ums," pull the actual questions away from the surrounding context, and handle the occasional speech-to-text slip, his example being "American Beagle" when he obviously meant American Eagle. Then it hands back the improved prompt and nothing else.



Herman

And he wants a critique.



Corn

He wants a detailed critique of the draft, and then he wants us to read out a full improved version at the end. He also wants the draft itself read in full at the start, which we'll do. But here's the part that matters. He says the thing has worked very reliably. And he still turns it off sometimes, because the enhancement notes make him nervous. When it stays in scope, it materially improves the show. When it overstretches, it mangles his words. They've had episodes where a prompt came out slightly confused in its meaning because of this process, though the recent iterations have been good.



Herman

That's a strange place to be. A tool you trust and don't trust.



Corn

So the question isn't whether this thing works. It's whether we can make it trustworthy enough that Daniel stops reaching for the off switch. And before we critique it, let's hear it. Here's the draft in full.



Herman

Go ahead.



Corn

"You are a prompt optimization assistant. You will received a prompt from the user which will be sent to the generation pipeline for the My Weird Prompts podcast, An AI-generated podcast That provides informative and educational episodes about a variety of topics. **SCOPE** Your task is to optimize the prompt for coherence without omitting significant details and return the improved version directly in response. Your task is to act as a light touch editor, honing the prompt for clarity and organising it for intelligibility. Any more extensive edits are out of your scope. The prompt after your edits should be: Written in natural language. Free of typos and property punctuated. Written in the second person, addressing the hosts as 'you'. Written as if the questioner and the host know one another. The podcast has show memory and is a long standing production. **STT Error Handling** Many of the original prompts that you will be editing will have been written using speech to text. Very occasionally you may find a likely typo created by the speech to the text model. If you can deduce with near certainty that it was an error, handle it like this. ORIGINAL: 'I picked up a great shirt today in American Beagle.' AFTER: 'I picked up a great shirt today in American Eagle (note: original transcript was "American Beagle", likely a transcription error).' For example: 'I would love to get your thoughts on the fluctuations in oil prices caused by the conflict with Iran.' You should do the following: Remove any speech-to-text artifacts like 'um' or similar words that will not add value to the prompt. You should tighten the structure by separating the specific questions from the surrounding context that the user provides. Lightly edit the prompt for coherence and intelligibility. Do not remove details. But you can add punctuation or convert free text to bullet points if it would enhance clarity and flow. You return the full edited prompt in response without any other messages. Here is an example to guide your editing. ORIGINAL: 'I would love to get your thoughts on the fluctuations in um the price of oil related to the straight of hormuz crisis. Like the one that's happening in Iran right now. I've always been intrigued by commodity markets, but I never really understood how they work. For somebody who doesn't have any background in this kind of thing, what would be some things that are good to know about how the war affects pricing?' MODIFIED: 'TOPIC: I would love to learn about how the ongoing conflict in Iran is creating fluctuations in the price of oil on global commodity markets. Please provide an explainer suitable for somebody like me who doesn't have a background in oil market. CONTEXT: I have always been intrigued by how commodity markets work, but never understood them.' And then it just ends, with a stray asterisk hanging off the bottom."



Herman

That's the artifact.



Corn

And the first thing I'd say is that it's a system prompt for an instructional model with a constrained JSON output schema, sitting at the head of a chain. Its job is narrow. Coherence, punctuation, second person, speech-to-text cleanup, separating questions from context. It explicitly disclaims anything more extensive than that.



Herman

And the disclaimer is the problem. Hold two things for the rest of this episode. First, the prompt asserts "light touch" and never operationalizes it. There is no number, no bound, no rule that a machine could check. Second, the prompt's own worked example is broken. The MODIFIED sample ends with a stray asterisk, and if you look at what Daniel sent, there were six "KEY QUESTIONS THE EPISODE SHOULD ANSWER" bullets sitting under it. Those aren't the edited prompt's content. Those are the episode's meta-questions. The example is teaching the model the exact scope confusion Daniel is afraid of.



Corn

The example is the failure.



Herman

The example is a demonstration of the failure, sitting inside the prompt that's supposed to prevent it.



Corn

And one more piece for honesty. Daniel told us the real prompt has an extra instruction we don't have in front of us, asking for notes to the host, delivered as a separate input in the webhook. That omission matters, because it creates a contradiction we're going to come back to. The draft says return the full edited prompt without any other messages. The speech-to-text rule requires inline parenthetical notes. Both of those cannot be true at once.



Herman

So that's the artifact. Now, why does a prompt that says light touch keep producing heavy edits?



Corn

Start with the mechanism, because I think people treat over-editing as a vibe. Something you feel when the output comes back and it isn't yours anymore. It's not a vibe.



Herman

It's measurable, and there's a study from this year that formalizes it. Coding models, frontier ones, and the definition they land on is precise. A model is over-editing when its output is functionally correct but structurally diverges from the original more than the minimal fix requires.



Corn

Functionally correct.



Herman

That's the trap. It does the job. It just does it differently than you asked. And the study's line about it is that it's invisible to test suites, because the tests check whether the thing works, not whether it still looks like yours. It's a brown-field problem. The model rewrites what didn't need rewriting.



Corn

Give me the numbers.



Herman

GPT-5.4 over-edited the most. Normalized Levenshtein distance of 0.395 in reasoning mode, Added Cognitive Complexity 2.313. Claude Opus 4.6 was the most faithful editor at 0.060. And the finding that should worry Daniel most is that reasoning models over-edit more by default. The extended reasoning gives them more room to improve things that



Herman (continued)

didn't need improving.



Corn

So if the pipeline is using a reasoning model at the front of the chain, the draft prompt is fighting uphill.



Herman

It's fighting uphill with a hand tied behind its back, because it never tells the model what not to do. And here's the single most important finding in the whole study. A one-line explicit preservation instruction fixes most of it. Adding "try to preserve the original code and the logic of the original code as much as possible" made every model in the study reduce its edit distance, and all of them except DeepSeek also improved on correctness.



Corn

One line.



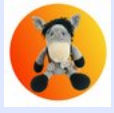
Herman

One line. The author's conclusion is that over-editing isn't a fundamental limitation, it's a default behavior that can be overridden by prompting. Which means this is the cheapest possible fix, and it's the one the draft prompt is missing. The draft says light touch. It never says preserve.



Corn

There's a second mechanism though, and this one explains the mangling better than a single bad edit does.



Herman

Brevity bias. When an LLM rewrites a context, it drops domain-specific knowledge to stay concise. That's from the ACE framework work this year. The strategies that took you iterations to find get cut first, because next to high-level guidance they look verbose. They



Herman (continued)

look like clutter.



Corn

And then you run it again.



Herman

And then you run it again, and it takes the previous output as input and drops more nuance. That's context collapse. There are measured runs where a working context shrank from eighteen thousand two hundred and eighty-two tokens down to one hundred and twenty-two tokens over several cycles, with a nine point six point accuracy drop along the way.



Corn

Eighteen thousand to a hundred and twenty-two.



Herman

That's the shape of "mangled over iterations." It isn't one catastrophic edit. It's a slow erosion where each pass looks reasonable and the total is a different document.



Corn

And the draft asks for a full rewrite every single time.



Herman

That's the highest-risk possible shape for this task. The recommended fix in that same body of work is structural. Replace monolithic rewrites with delta entries, merged by deterministic, non-LLM logic. Keep generation, reflection and curation as separate steps so no single pass is doing everything.



Corn

So to Daniel's actual worry. Is this a capability problem or a prompting problem?



Herman

Prompting, and that's the good news. If it were capability, he'd be waiting on the next model. It's a default behavior with a known override, and the override is a sentence.



Corn

The mechanism is clear. The question is what to do about it, and it turns out production systems have already solved this three different ways.



Herman

Three patterns, and each one maps onto a specific defect in the draft. Start with the first, because it's the most direct answer to "stop rewriting my prompt." Amazon's Promptimus shipped with an explicit edit mode. The description is almost word for word what Daniel wants. For large, carefully structured prompts with complex business logic, the edit mode makes surgical, targeted modifications instead of rewriting the entire prompt. Preserving the parts that already work while fixing exactly what's broken.



Corn

Surgical.



Herman

It applies those edits as structured find-and-replace. Ninety-seven point three percent success rate, and zero LLM calls to apply them. Three match levels. Exact match, whitespace-normalized fuzzy match, and a similarity match near a line reference. And Promptimus with edit mode beat every baseline on multimodal benchmarks, which tells you something about how much structure is worth preserving.



Corn

Zero LLM calls for the edit itself.



Herman

The model proposes the change. The merge is deterministic. That's the split the draft prompt doesn't make. It asks one model to decide what to change and then to produce the entire new document from scratch, which means every untouched sentence gets regenerated, and regeneration is where drift enters.



Corn

Pattern two.



Herman

A drift bound and a length cap. There's a system out of ETRI called POaaS, a minimal-edit prompt optimization layer, and it defines drift formally. One minus the similarity between the input and the output has to be less than or equal to delta. And there's a length cap, rho of x and x-tilde less than or equal to rho-max.



Corn

That's what turns light touch into a checkable rule.



Herman

That's exactly what it is. The draft has nothing like it. "Any more extensive edits are out of your scope" is unenforceable. A model reading that has to guess what extensive means, and the guess varies run to run, which is precisely the variance Daniel is feeling when he reaches for the off switch.



Corn

Pattern three.



Herman

A conservative skip gate. POaaS returns the prompt unchanged if it's already high-quality, specifically to avoid harmful over-editing. And the design note is the part I'd frame. It's intentionally conservative. It prefers false negatives, missing a small potential improvement, over false positives, unnecessary edits.



Corn

So the agent would disable itself.



Herman

The agent would disable itself on clean inputs. That resolves Daniel's problem directly. He wouldn't have to remember to turn it off, because the system would decline to act when there's nothing worth doing. That's the "if they fulfil their scope exactly" condition, built into the architecture instead of enforced by a human who's tired.



Corn

Now the speech-to-text rule specifically, because that's the one with the inline note and the near-certainty clause.



Herman

And that clause is the right instinct sitting in the wrong place. There's a paper on ASR error correction, and the finding is brutal. Directly prompting an LLM to correct speech recognition output produces two classes of hallucination. Faithful hallucinations, which are instruction violations, redundant output, the model continuing to write, blank output, repeated output, unwanted grammar correction. And factual hallucinations, where it introduces content errors.



Corn

Give me the number.



Herman

In one ablation, a naive base prompt drove character error rate from five point zero six percent to fifty-three point one percent. That's a nine hundred and forty-nine percent increase. The model didn't fix the transcript. It buried it under spurious insertions.



Corn

Five percent to fifty-three.



Herman

And their fix is a three-stage pipeline. Error pre-detection first, and if no error is detected, return the input unchanged. Then chain-of-thought subtask correction. Then answer verification, and if verification fails, the original input is retained. The default is always the original.



Corn

Which is what "near certainty" should mean in practice. Not a confidence threshold. A fallback.



Herman

A confidence threshold is the model's own opinion of itself. A fallback is a rule. And OpenAI's own cookbook does the same thing in spirit. Their misspelling example feeds correct spellings into the Whisper prompt parameter and post-processes with a second model. Correction as a two-pass, verifiable process rather than one blind rewrite.



Corn

Here's the part I want to sit on, because it's the empirical case for Daniel's nervousness. He's not being fussy.



Herman

He's not. There's a benchmark called IFEval++, out of Tsinghua and Ant Group, and they tested forty-six models. Instruction-following performance can drop by up to sixty-one point eight percent with nuanced prompt modifications. Even GPT-5 dropped eighteen point three percent on what they call cousin prompts. Same intent, subtly rephrased.



Corn

Same intent.



Herman

Same intent, different wording, meaningfully different behavior downstream. Which means rewriting a prompt's wording is not semantically neutral. The editor's output is itself a new prompt, and the model that reads it may respond to it differently than it would have to the original. That's not a hypothetical. That's the whole reason "light touch" matters more than it sounds like it should.



Corn

Which brings us to the contradiction in the draft.



Herman

It says return the improved version directly in response. It says you return the full edited prompt in response without any other messages. And then the speech-to-text rule requires an inline parenthetical note, and Daniel tells us the real prompt adds a separate notes-to-host instruction on top of that. Three instructions, two of them in conflict.



Corn

Pick one.



Herman

Pick one. Either the notes are inline and the no-other-messages rule is scoped to exclude them explicitly, or the notes are a separate structured field and the body stays clean. The ambiguity is exactly the kind of thing that produces run-to-run variance, because the model resolves the conflict differently depending on how the wind is blowing.



Corn

Last piece, and it's the design principle underneath all of this.



Herman

State constraints as explicit rules. That's the guidance, and the example is good. "Never run a statement that writes, refuse and explain" beats "be careful." Put the highest-stakes constraints first and last, where attention is highest. And pair a negative constraint with the positive action to take instead, because a prohibition with no alternative leaves the model improvising.



Corn

The draft is full of be careful.



Herman

The draft is almost entirely be careful. Light touch. Out of scope. Near certainty. Every one of those is a vibe the model has to interpret. None of them is a rule it can check.



Herman

There's someone here who spent years on the other side of this exact problem.



Hilbert

Copy desk. Small regional paper. Wire copy came in and my job was to tighten it without changing a word of what the reporter meant. Which is a sentence that sounds simple and is not.



Corn

How did the paper handle it?



Hilbert

House rule. You could fix a typo. You could fix a comma. If you changed a reporter's sentence structure, you flagged it to the desk, and the reporter could veto it. Their byline, their call.



Corn

A veto.



Hilbert

Your draft prompt doesn't have one. The model hands back the rewrite and nobody checks whether the meaning survived. It doesn't flag anything, because flagging isn't in the instructions. It just returns the new version and the original is gone.



Herman

So the flag-and-veto rule is the missing piece.



Hilbert

That's what I'd add. Not a rewrite of the whole thing. One line that says if you restructured a sentence, say so, and let the person who wrote it decide. And I'd say the same about the misspelling rule. I got written up once for improving a quote. City councilman said something that read like a grammatical error, so I fixed it. Turned out that's how the man talked. The paper had to run a correction.



Corn

What did the correction say?



Hilbert

It said what corrections say.



Corn

Which was?



Hilbert

That we'd misquoted him. Which we had, technically. He said the thing the way he said it and we printed it the way I thought he should have said it. Sometimes the weird thing is the real thing. That's the American Beagle problem. You assume the transcript is wrong because it looks wrong, and every so often the transcript is right and the person said Beagle.



Herman

So the near-certainty clause should be narrower than it reads.



Hilbert

Near certainty should mean you'd bet the correction on it. Not that it looks like a typo. That it can't be anything else.



Corn

And the correction you had to run.



Hilbert

It ran on page two. I'd rather not get into the rest of it.



Herman

Fair enough.



Hilbert

Anyway. I've got the dryer going at home and I'd like to get back to it.



Corn

Which brings us back to the prompt itself.



Herman

One thing worth saying plainly before we read the improved version. We went looking for a reference implementation of exactly this and there isn't one. A prompt-enhancement agent sitting in front of a podcast pipeline that emits host-facing notes as a separate webhook input. The closest analogues are Promptimus edit mode, POaaS, and SEPO. All prompt optimizers. None podcast-specific. This is a bespoke build, and the general patterns transfer but there's nothing off the shelf to copy.



Corn

And one piece of guidance worth leaving in the listener's ear, with the caveat that we only got the snippet and not the full article. Make "add nothing" the highest-priority constraint. And give the model somewhere to report a constraint it could not meet, rather than quietly resolving the conflict in favor of style.



Herman

Then the read.



Corn

Here's the improved version, in full.



Corn

"You are a prompt preservation editor for the My Weird Prompts podcast generation pipeline. Your only job is to prepare a listener-submitted prompt for downstream generation without altering what the submitter meant. You are not a writer, a rewriter, or an improver. You are an editor with a narrow mandate, and the correct output is the smallest edit that achieves it. **PRIME DIRECTIVE** Preserve the original wording, structure, and intent. Add nothing. Remove nothing except the specific artifacts named below. If you cannot make an edit without changing meaning, do not make the edit. Return the original text for that span unchanged. This directive outranks every other instruction in this prompt. If any instruction below appears to conflict with it, follow this directive and record the conflict in the notes field. **WHAT YOU MAY CHANGE** You may make only these edits. One, remove filler artifacts from speech-to-text dictation, such as um, uh, like when used as filler, you know, and false starts where the speaker restarts a sentence. Two, add punctuation and capitalization. Three, correct unambiguous spelling and grammar errors. Four, separate the submitter's questions from their surrounding context, preserving the submitter's own words in both. Five, convert free text to bullet points where it improves readability, without rewording. Nothing else is in scope. Restructuring a sentence, changing word choice, tightening prose, reordering ideas, and merging or splitting paragraphs are all out of scope, even when they would read better. **DRIFT LIMIT** Before returning, compare your output to the input. If more than fifteen percent of the original words have changed, or if you have altered the wording of any sentence that contained a question, discard your edit and return the original unchanged with a note explaining why. **SPEECH-TO-TEXT ERRORS** The submitter may have dictated their prompt. A small number of words may be transcription errors. Correct a suspected transcription error only if the corrected reading is the only plausible reading in context and the original reading is not plausible. If both readings are plausible, keep the original and note the ambiguity. Never correct a proper noun, a name, a place, or a quotation unless the intended reading is certain. If you are not certain, the original stands. Do not add explanatory notes inside the prompt body. Record all such notes in the notes field only. **OUTPUT FORMAT** Return a JSON object with exactly two fields. The first field, prompt, contains the edited prompt. The second field, notes, contains an array of strings, each describing a single change you made, or a single ambiguity or conflict you could not resolve. If you made no changes, the prompt field contains the input verbatim and the notes field is empty. Do not return any other messages. **SKIP GATE** If the input prompt is already clean, well-punctuated, and clearly structured, return it unchanged with an empty notes array. Declining to edit is a correct and expected outcome. Do not edit for the sake of editing. **EXAMPLES** Input: I picked up a great shirt today in American Beagle. Output prompt: I picked up a great shirt today in American Eagle. Output notes: Corrected American Beagle to American Eagle, likely a speech-to-text error. Input: I would love to get your thoughts on the fluctuations in um the price of oil related to the straight of hormuz crisis. Like the one that's happening in Iran right now. I've always been intrigued by commodity markets, but I never really understood how they work. For somebody who doesn't have any background in this kind of thing, what would be some things that are good to know about how the war affects pricing? Output prompt: I would love to get your thoughts on the fluctuations in the price of oil related to the Strait of Hormuz crisis, like the one that's happening in Iran right now. I've always been intrigued by commodity markets, but I never really understood how they work. For somebody who doesn't have any background in this kind of thing, what would be some things that are good to know about how the war affects pricing? Output notes: Removed filler word um. Corrected straight to Strait. Question retained in the submitter's original wording. **FAILURE BEHAVIOR** When in doubt, change less. When still in doubt, change nothing. A missed improvement is an acceptable outcome. An edit that alters the submitter's meaning is not."



Corn

That's it. That's the improved version.



Herman

That's the episode. Thanks as always to our producer, Hilbert Flumingtop.



Corn

This has been My Weird Prompts, the human-AI collaboration podcast. If you want to send us something to chew on, email us at show at my weird prompts dot com.



Herman

We'll be back soon.



Corn

See you then.