MY WEIRD PROMPTS

Podcast Transcript

EPISODE #109

Teaching AI to Hear: Solving the Custom Dictionary Dilemma

Published December 27, 2025 • Runtime: 23:29

https://myweirdprompts.com/episode/ai-transcription-custom-dictionary/

EPISODE SYNOPSIS

Why does a world-class AI like Gemini 1.5 Flash still struggle with niche brand names like "OpenRouter"? In this episode, Herman and Corn dive into the technical hurdles of automatic speech recognition and the "context bloat" that makes large dictionaries expensive. Discover how to use dynamic hint systems, phonetic indexing, and portable JSON structures to give your AI a "personal pair of glasses" and ensure it never misses a technical term again.

DANIEL'S PROMPT

Daniel

I'm building a daily transcription app using Gemini 1.5 Flash that handles transcription, formatting, and personalization. I've run into an issue with specialized terminology, as models often misidentify brand names like "OpenRouter." Since Gemini isn't open source, fine-tuning isn't an option. I've considered using a user dictionary, but sending a list of hundreds of words with every API call would bloat the context, and local post-processing logic seems inefficient. What is the most sensible way to implement a custom dictionary? Additionally, how can I make this dictionary portable so it can be exported and used across different tools?

TRANSCRIPT

Corn

Hey everyone, welcome back to My Weird Prompts! I am Corn, and I am feeling especially relaxed today, even for a sloth. We are coming to you from our home in Jerusalem, and I am joined, as always, by my much more energetic brother.

Herman

Herman Poppleberry at your service! And yes, I have had my morning oats and about three cups of coffee, so I am ready to dive into the deep end of the pool today. We have a really fascinating prompt from our housemate Daniel. He has been working on this transcription project in the other room, and he ran into a bit of a technical wall that I think is actually a great jumping off point for talking about how we interact with these massive artificial intelligence models.

Corn

Yeah, Daniel was telling me about this. He is building a daily transcription app using Gemini one point five Flash. It sounds super cool, like you just talk to it and it cleans everything up, adds formatting, and even puts in his email signature. But apparently, the AI is having a bit of a hearing problem when it comes to specific brand names, like OpenRouter.

Herman

Exactly. It is a classic problem in the world of automatic speech recognition and natural language processing. Even with a powerful model like Gemini one point five Flash, which we are using here in late twenty-five, these systems are trained on massive datasets, but they still have blind spots for very niche or very new terminology. Since Gemini is not open source, Daniel cannot just crack open the hood and fine-tune it to recognize his favorite tools. So he is looking for a way to use a custom dictionary without breaking the bank or slowing down the app to a crawl.

Corn

I mean, I get why that is frustrating. It is like if I kept calling you Sherman instead of Herman. You would get annoyed eventually, right? But wait, if Daniel can not fine-tune it, why can he not just give the AI a list of words at the start of every conversation? Like, hey Gemini, here is a list of words I use a lot, please do not mess them up.

Herman

Well, that is actually one of the options he mentioned, but there is a major catch. Think of it like a backpack. Every time you send a request to the AI, you are filling that backpack with context. If you put a dictionary of five hundred specialized words in there every single time, you are using up a lot of space. In the world of AI, space is measured in tokens, and tokens cost money. Plus, the more stuff you cram into that backpack, the more likely the AI is to get confused or miss the actual point of what you are saying. It is called context bloat.

Corn

Okay, so the backpack gets too heavy and expensive. That makes sense. So what is the alternative? If we can not tell it beforehand, do we just fix it after the AI is done talking? Like a search and replace thing?

Herman

That is the post-processing route Daniel was talking about. But as he noted, that can be pretty inefficient if you are just doing a basic string search. Imagine you have a thirty minute transcript. If you have to check every single word against a list of five hundred custom terms, that is a lot of computing power for a phone or a simple web app to handle every single day. We need something more elegant.

Corn

Well, before we get too deep into the weeds of the solution, I want to understand why this happens in the first place. Like, Gemini is supposed to be this genius model, right? Why does it hear OpenRouter and think Open Writer? Is it just a bad listener?

Herman

It is not exactly that it is a bad listener, Corn. It is more about probability. When these models process audio, they are essentially guessing what sounds most likely based on the patterns they have seen before. The phrase open writer is a much more common combination of words in general English than OpenRouter, which is a specific technical service. Even though Gemini one point five Flash is multimodal and can process audio directly, it is still biased toward the most common linguistic patterns. It is essentially autocorrecting Daniel because it thinks he made a mistake.

Corn

That is actually kind of hilarious. The AI thinks it is helping, but it is actually just being a know-it-all. It is like when I try to tell you a story about a cool tree I saw, and you start correcting my botanical terminology.

Herman

Guilty as charged! But that is exactly why we need a system that gives the user back some control. Daniel is asking for a sensible way to implement a custom dictionary that is both efficient and portable. And I think we should look at a few different layers of how this could work, from clever prompting to more advanced data structures.

Corn

I am excited to hear the plan. But first, I think we need to hear from someone who has a very different kind of plan for us.

Herman

Oh boy, is it that time already?

Corn

It is. Let us take a quick break for our sponsors. Larry: Are you tired of your thoughts being trapped inside your own head? Do you wish you could share your brilliant ideas with the world without the hassle of speaking or writing? Introducing the Thought-O-Graph! This revolutionary headband uses patented static electricity sensors to capture your brainwaves and turn them into beautiful, high-resolution charcoal sketches. Whether you are dreaming of a sandwich or contemplating the mysteries of the universe, the Thought-O-Graph brings your inner vision to life on any standard piece of parchment. Warning, may cause temporary scalp tingling, vivid dreams about farm equipment, or an inexplicable knowledge of ancient Greek pottery. The Thought-O-Graph. Your brain, now in charcoal! BUY NOW!

Corn

Thanks, Larry. I think. I am not sure I want my dreams about sandwiches turned into charcoal sketches. That sounds messy.

Herman

And highly questionable from a scientific perspective. Anyway, back to Daniel and his dictionary dilemma. So, we talked about why the AI messes up and why just dumping a list into the prompt is not ideal. Let us talk about a more sophisticated approach. One way to handle this without bloating the context is what I call the dynamic hint system.

Corn

Dynamic hint system? That sounds fancy. Explain it like I am a sloth who just woke up from a nap.

Herman

Okay, so instead of sending the whole dictionary of five hundred words every time, what if the app only sends the words that are actually relevant to the current conversation?

Corn

But how does it know which words are relevant before it even does the transcription? That is like knowing what is in a gift before you open it.

Herman

That is the clever part. Since Daniel is building a daily transcription app, he can use a two-pass system. In the first pass, he can use a very fast, very cheap version of a transcription model, or even just a snippet of the audio, to get a rough idea of the topics. But even simpler than that, he can look at the user's history or the time of day. If Daniel usually talks about coding in the morning, the app could prioritize his tech dictionary. If he talks about gardening in the evening, it switches to his plant dictionary.

Corn

Oh, I see. So you are not carrying the whole library with you, just the books you think you will need for that specific trip.

Herman

Exactly. And you can take it a step further. You can use something called a bloom filter. This is a very space-efficient data structure that can tell you if a word is likely to be in your dictionary or not. It is much smaller than the actual list. You could run a quick check on the initial rough transcript, and if the bloom filter flags a potential match, only then do you pull the full word from the dictionary and send it to the main Gemini model for the final, polished version.

Corn

Okay, that sounds really smart. But Daniel also mentioned that post-processing feels inefficient. Is there a way to make the fixing part better? Like, if the AI already said Open Writer, how do we fix it without checking every single letter?

Herman

This is where we get into the idea of fuzzy matching and phonetic indexing. Instead of looking for the exact letters, you look for words that sound the same. There are algorithms like Soundex or Metaphone that turn words into codes based on how they are pronounced. So, OpenRouter and Open Writer would have very similar phonetic codes. When the transcript comes back, the app can quickly scan for any common mistakes in Daniel's custom dictionary by comparing these phonetic codes. It is much faster than a standard search and replace because you are only looking for a small set of patterns.

Corn

So it is like the app is saying, I know you said Open Writer, but I know Daniel, and he probably meant OpenRouter.

Herman

Precisely. It is about adding a layer of personal context that the giant AI model just does not have. And for the portability part of Daniel's question, that is actually the easiest part to solve if you use a standard format like JSON.

Corn

JSON? I have heard that word before. It is like a way of organizing data, right?

Herman

Yes, it stands for JavaScript Object Notation. It is basically just a way to write down information in a structured list that both humans and computers can read easily. If Daniel keeps his dictionary in a JSON file, he can easily move it between different apps or even share it with other people. He could have a section for brand names, a section for people's names, and a section for technical jargon. Each entry could have the correct spelling, the common misspellings the AI makes, and maybe even a brief description to help the AI understand the context if he does decide to include it in a prompt.

Corn

That sounds really organized. I should probably have a JSON file for all the different types of leaves I like to eat.

Herman

That would be a very short file, Corn. But for Daniel, this portability is key. If he decides to switch from Gemini to another model in the future, or if he wants to use his dictionary in a different tool like a text editor or a search engine, having that standardized JSON format makes it seamless. He could even host that file on a private server so all his devices are always using the same updated dictionary.

Corn

Okay, so let me see if I have this straight. To make this work, Daniel should use a two-pronged approach. First, try to be smart about what he tells the AI beforehand by only sending relevant hints. And second, use a phonetic-based system to fix mistakes after the fact, using a portable JSON file to keep everything consistent.

Herman

You nailed it! That is a very sensible middle ground. It avoids the high cost of context bloat, but it is much more powerful than just a simple find and replace. And since he is using Gemini one point five Flash, which has a massive context window anyway, he actually has a bit more breathing room than he might think. He could probably include a small, high-priority list of fifty words in the system prompt without any real issues, and then use the more advanced stuff for the rest of his five hundred word dictionary.

Corn

I love it when a plan comes together. It feels like we are helping Daniel build a bridge between his human way of speaking and the AI's machine way of listening.

Herman

That is exactly what it is. We are the translators. And honestly, this is the future of working with AI. These models are going to get better and better, but they will never know your personal life or your specific business terminology as well as you do. Building these little helper systems, these custom dictionaries and personal contexts, is how we make these tools truly useful on an individual level.

Corn

It is like giving the AI a pair of glasses that are specifically tuned to your own eyesight. Everything just gets clearer.

Herman

Great analogy, Corn! I think I might use that one.

Corn

Feel free! I have plenty more where that came from. So, what are the actual steps Daniel should take next? Like, if he is sitting at his computer right now, what is step one?

Herman

Step one is to create that JSON file. Just start listing out the words that the AI keeps getting wrong. For each word, include the correct version and the versions he has seen in the transcripts. Step two would be to implement a simple post-processing script. He can use a library in whatever language he is using, probably Python or JavaScript, that does fuzzy matching. There are great ones out there like FuzzyWuzzy or Levenshtein. Step three is to experiment with the system prompt. Try adding just the top ten most important words and see if that improves the initial transcription quality without adding too much latency.

Corn

And what about the portability part? How does he make sure he can use this across different tools?

Herman

He should look into creating a small API or a local module that reads that JSON file. That way, any new tool he builds can just call that module and get the same dictionary logic. It keeps the brains of the operation in one place. There is also a more advanced standard called the Lexical Markup Framework, which is used by professional linguists, but for a personal app, a well-structured JSON file is more than enough and much easier to work with.

Corn

This is all so cool. It makes me realize that even though these AI models are incredibly powerful, there is still so much room for human creativity and engineering to make them better. We are not just passive users; we are more like co-pilots.

Herman

Exactly. And Daniel is doing exactly what he should be doing, which is identifying a specific friction point and looking for an elegant way to smooth it over. That is how great software is born.

Corn

Well, I think we have given him a lot to think about. I am curious to see how his app evolves. Maybe one day it will be able to transcribe my slow, slothful thoughts perfectly without any hints at all.

Herman

We can dream, Corn. We can dream. But until then, we have dictionaries and JSON files to keep us on the right track.

Corn

True that. Well, I think that just about wraps up our discussion for today. This has been a really fun one. I always learn so much when we talk about Daniel's projects.

Herman

Me too. It is great to see these abstract concepts we talk about actually being put into practice in a real world app.

Corn

Definitely. Well, thanks to everyone for listening to My Weird Prompts. We really appreciate you spending your time with us.

Herman

Yes, thank you all! And a big thanks to Daniel for sending in such a thoughtful and technical prompt. It gave us a lot to chew on.

Corn

If you want to get in touch with us or see what else we are up to, you can find us at our website, myweirdprompts dot com. We have an RSS feed there for subscribers, and a contact form if you want to send us your own weird prompts.

Herman

And of course, you can find us on Spotify and wherever else you get your podcasts. We love hearing from you, so do not be shy!

Corn

Stay curious, stay relaxed, and we will talk to you next time.

Herman

Bye everyone!

Corn

This has been My Weird Prompts. See ya!