

Cartesia Take-home | Model Behavior

Chris Morrison's report for the Model Behavior Product Manager take-home.

Please note that all recordings linked in this report are hosted on a single static site via GH pages, to allow for easier listening ([Easily listen to all audio samples](#))

Step 1 - Spend 30 mins listening to Cartesia

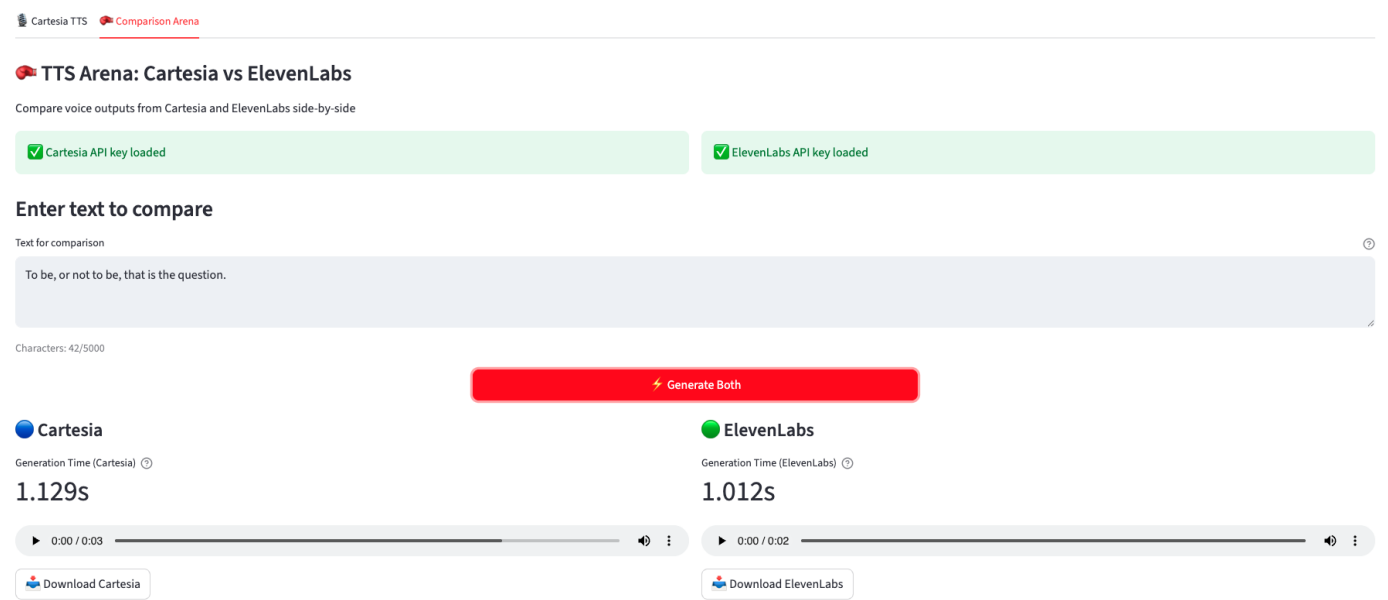
1.1. Specify how you did this?

1.1.1. Initial Investigation

Initially I used the [playground](#) and [documentation](#) to "learn by doing". My primary goal when reading the documentation and playing with the playground was to identify the different 'levers' that were available to change *how* the speech was generated.

1.1.2. Enabling more robust testing

After playing around on the playground, I wanted to be able to more easily compare the results of Cartesia with ElevenLabs side-by-side without having to worry about jumping between websites. I used Claude Code to spin up a Streamlit app that I could use to generate TTS from both APIs from the same input prompt, similar to the popular 'LM/Chatbot Arena' but for speech. I primarily used this locally, but I pushed it to the Streamlit community cloud if you want to give it a try (you may need to add your own API key) ([TTS Arena on Streamlit Cloud](#)).



1.2. What were you listening for specifically?

The first thing I was listening for was a simple 'does it work'. Specifically, does it sound like a normal person for 99.9% of the speech. This is quite a high bar because even one unnatural expression can ruin the whole experience. In this regard, I found Sonic 3 to be excellent right out of the box, without any additional prompting or tweaking to punctuation, it was able to generate an excellently 'expressed' TTS ([Example](#)). In comparison, key competitors like ElevenLabs generate a very 'flat' sounding audio output from the same prompt ([Example](#)).

Beyond basic functionality, the most important thing I wanted to test was how much the expressiveness could be influenced without losing any of the nuance of how real humans speak. From my experience working on Alexa+, I know that the way that humans (and models) generate speech is extremely nuanced and tiny tweaks on the modeling side can lead to significant changes in the customer experience. To assess this, I refer to an excellent comedy sketch I saw a few years ago, where some incredible stage actors debate how specifically to do the famous "to be or not to be" speech from Hamlet - <https://www.youtube.com/watch?v=RJXiep-yGBw>. My goal when experimenting with the Cartesia TTS model was to emulate this sketch by assessing how nuanced the changes would be in the output, for a given input. I experimented with different punctuation and noted how small differences in the input resulted in nuanced changes to the models generated output ([Example 1](#), [Example 2](#))

Step 2 - Specify 2-3 Aspects in which the model can improve

2.1. Areas of Improvement

After testing of different speech dimensions, I identified two areas to focus on analyzing for model improvement: Locale-specific pronunciation; and Acronyms, numbers & symbols. I selected these two areas because they represent fundamental capabilities that will scale across all use cases of the model. These aren't edge case capabilities, so it is critical that the model gets these things right with a high success rate.

2.1.1. Locale-specific Pronunciation (and Competitor Comparison)

Being able to pronounce words correctly within the locale / dialect context is critical for natural human-like TTS. This is a nuanced area because some words are pronounced differently depending on the dialect. For example in for example in US english, "croissant" (French loanword) is typically pronounced to rhyme with 'want', whereas in British English it would be pronounced more like the French version (although native French speakers would debate the similarity!). Other loanwords are typically pronounced using the native pronunciation, even if the word itself would be pronounced differently if written in English. An illustrative example of this is Mexican-Spanish placenames like Guadalajara and Tijuana which are pronounced with a Spanish 'hwa' in place of the hard 'j/g', even by English speakers with no Spanish language skills. The below table outlines a handful of examples illustrating how the Cartesia model performs on these tricky situations, and how a competitor model (ElevenLabs) performs on the same prompt.

Sub-dimension	Example Transcript	Improvement Required (Cartesia)	Issue Consistency?	Competitor Rating	Competitor Assessment	Cartesia Recording	Competitor Recording	Comment
Loanwords #1	Geneviève will visit Guadalajara en route to São Paulo.	None of the words were pronounced correctly	Consistent	Cartesia < Competitor	"Geneviève", "en" and "São Paulo" are pronounced poorly, but Guadalajara is spot-on for a natural American pronunciation.	Link	Link	Tested using standard US voices
Loanwords #2	Nguyễn met Siobhán at the café on Rue de la Boétie.	None of the words were pronounced correctly	Consistent	Cartesia < Competitor	"Nguyen" and "Siobhan" mispronounced, but French words were pronounced well (potentially *too* well)	Link	Link	Tested using standard US voices
Locale-specific pronunciation #1	The schedule was posted in the laboratory.	Laboratory is not quite pronounced correctly, sounds too 'American' for a British pronunciation	Consistent	Cartesia < Competitor	Both are pronounced accurately for a Brit	Link	Link	Tested using UK voice (Charlotte for Cartesia, Lily for ElevenLabs)
Locale-specific pronunciation #2	The aluminium foil was in the boot of the car.	Aluminum is pronounced halfway between American pronunciation and British pronunciation.	Consistent	Cartesia < Competitor	Pronounced accurately for a Brit	Link	Link	Tested using UK voice (Charlotte for Cartesia, Lily for ElevenLabs)

Sub-dimension	Example Transcript	Improvement Required (Cartesia)	Issue Consistency?	Competitor Rating	Competitor Assessment	Cartesia Recording	Competitor Recording	Comment
Locale-specific loanwords	Flavored croissants are very niche	Croissants and niche are pronounced strangely for a British accent (niche doesn't rhyme enough with leash).	Consistent	Cartesia < Competitor	The pronunciation of both croissants and niche are spot on.	Link	Link	Tested using UK voice (Charlotte for Cartesia, Lily for ElevenLabs)

2.1.2. Acronyms, Symbols & Numbers (and Competitor Comparison)

Being able to accurately and naturally articulate Acronyms, Symbols and Numbers is a critical skill for a TTS model, since these situations show up in almost every real-world use case imaginable (legal, financial, support calls, etc). A failure on this task (which human readers/speakers would find trivial) won't just result in an "uncanny valley" situation, it can fundamentally cause confusion or mislead the end user, breaking all trust that might have been earned. Additionally, being able to do this task accurately demonstrates that the model has a deeper understanding and contextual awareness of the information being communicated.

Sub-dimension	Example Transcript	Improvement Required (Cartesia)	Issue Consistency?	Competitor Rating	Competitor Assessment	Cartesia Recording	Competitor Recording	Comment
Filenames	The file is saved as config.yaml in the etc directory.	Voice trails off and sounds hesitant, strange pronunciation of "[...]aml" and "etc"	Consistent	Cartesia > Competitor	Performs worse on both filename and etc pronunciation	Link	Link	Tested using standard US voices
Acronyms	He got a Ph.D. from MIT and now works at OpenAI, Inc.	Mostly correct except that PHD is pronounced P-H-dot-D.	Consistent	Cartesia < Competitor	Performs better, naturally sounding out the PHD without punctuation.	Link	Link	Tested using standard US voices
Numbers	Call me at 03/04/05 at 06:07.	Doesn't correctly identify the date format so says "April fifth" instead of "Fourth of March 2005" or "Third of April 2005" (either are correct depending on context)	Consistent	Cartesia > Competitor	Performs poorly, simply reads out the numbers.	Link	Link	Tested using standard US voices

Step 3 - Pick one of the aspects to dive deeper on

Choice: Locale-specific Pronunciation

Why did you choose that one?

This is important because 20% of US households speak at least one other language at home regularly (per US Census Bureau). But it's not just those who speak another language. The vast majority of native speakers will notice if a word is not pronounced how they have heard it pronounced all their lives.

How would you operationalize data collection & evaluations?

Priority 1: Golden Set Comparison

Ultimately, the best way to get a strong signal to evaluate performance in this area is to generate a golden set of roughly 100-1,000 words or phrases (per language/locale) that represent words that have challenging or novel pronunciation. At every major model release, this golden set would be used to assess the performance of the new model, by generating a fresh TTS for each of the golden set words and phrases. This TTS (generated by the new model) would then have to be manually annotated, ideally by native speakers, along specific dimensions like accent and pronunciation. This annotation process would involve humans listening to the generated TTS and comparing it against a known-good recording of a native speaker speaking the same words/phrases, and then rating the generated TTS in comparison. For this specific area, native speakers are critical to success, since non-native speakers would miss a lot of the nuance that makes this evaluation important and valuable.

Priority 2: Introducing Automation

Because the above process is time-consuming and expensive, it is likely that it could only be performed once or twice for each major model release. This is a problem, because as the modeling team are working to improve the model they need a short feedback loop to validate if their experiments or new approaches are working, and they cannot wait months to get the information they need to make an informed decision. For this reason it is important to supplement this golden set assessment with an automated process that might not be perfect but can be run quickly and cheaply to provide a directional signal on a regular basis.

This specific task is a challenging one to solve in an automated way, but setting up a simple ASR/STT (Automatic Speech Recognition/Speech to Text) pipeline using a phoneme-based ASR (e.g. wav2vec2phoneme) could enable a rapid comparison of pronunciation at a semi-granular level. This pipeline would work by creating a known-good IPA (International Phonetic Alphabet) representation of the previously described golden set of challenging words and phrases, which would then act as the baseline to compare against ASR-generated phonemes from the output of the latest model. This setup would enable reasonably fine-grained identification of localized pronunciation, for example for UK voices, /'kwæsɔ̃/ would be the IPA groundtruth label for "croissant", whereas for a US voice it would be /kʁə'sɑnt/. This approach would enable the modeling and product team to rapidly assess the performance improvements of the model as needed, without waiting for manual annotations.

One tradeoff is that this approach is a discrete representation of a continuous variable, so small changes in how the model pronounces words could result in large changes to the measurement of this benchmark. Another tradeoff is that this approach would effectively introduce an additional variable (the ASR model) that may impact the results of the evaluation. If the ASR model had specific limitations in language or pronunciation, that could lead to incorrect or misleading results. However, through comparison with manually generated known-good labels, the correlation of this rapid offline benchmark against the manual measurement could be dialed in, and more confidence could be placed in the numbers over time.

Step 4 - A gift from the modeling team (Context Vector)

Step 4.1. What are 2-3 usages of this context feature that you could see? What is the first one you would try?

This functionality effectively enables a deeper level of customization for the model output, but without having to re-train the model (which takes time and GPUs). Below are three possible use cases for this context feature, in priority order - Cross-language performance matching would be the first use case I would investigate.

Priority 1: Cross-language performance matching in dubbing

Use **context** vector to carry the acting choices from the original performance into a localized/dubbed version. This capability has applications wherever audio and video require localization (YouTube, Podcasts, Instagram/Tik Tok, etc). This would involve extracting the context vector representation of the original voice throughout the duration of the recording, and re-inserting it during the generation of the localized dub. This is extremely powerful, and would result in a higher quality automatically generated dub, where the outputted audio mirrors the original in tone, emotional shape, phrasing changes, etc.

Priority 2: Listener-personalized voice preference

Explicit voice preference for voice assistants like Alexa and Siri - if I tell them to speak louder or faster, then I can easily load that up with each new interaction. Over time this would make the way that these assistants speak personalized, in addition to the things that they do / say. This approach would use a per-listener preference context to blend with content context at synthesis time (e.g., "serious news" and "slightly faster, less dramatic"). The preference vector could be built from behavior signals (skips/replays/ratings, "can you repeat that") or explicit requests ("a bit louder"). For the customer, the voice delivery feels personalized and tailored to their needs without adjusting sliders or updating profiles.

Priority 3: Single, dedicated support agent

Create the perception of a single, dedicated support agent for call-center tasks, that has context of how the conversation has gone previously. This agent would be the equivalent of interacting with a small business with excellent customer service - when you call up, you know who will answer and they can immediately recall who you are and the prior conversations that you have had (and adopt the appropriate tone as a result). Using a context vector is much better than trying to store the state outside the model and then trying to fake it or replicate it using discrete voice tags at generation time.

Step 4.2. Write the launch blog post for this feature.

Introducing Performance-Preserved Dubbing with Cartesia Context

Generate professional-grade dubbing and translations for a hundredth of the cost!

We're launching a new Cartesia capability that automatically generates multi-locale dubbing for your content, with up to 30% improvement in the quality of the audio in the new language (when compared to dubs created without using Cartesia Context).

Our new technology enables you to create translated audio at a quality that matches a professional dubbing actor, at a fraction of the cost, by keeping the acting choices of the original performance intact. This means you are translating not just the words, but the entire performance!

By capturing the emotional arc, loud/soft moments, and pacing of the source actor, Cartesia can carry that intent into the target language so the dubbed line feels true to the original scene.

Why it matters

Most dubbing today matches words and timing, but loses performance. The result can feel flat or off-brand, even when the translation is accurate. Performance-preserved dubbing bridges that gap: a Spanish adaptation keeps the urgency of the English lead; a TikTok voiceover holds the playful rise-and-fall of the creator; a localized game cutscene preserves tension without re-directing every line.

Available now

Performance-preserved dubbing is live via the Cartesia API. Explore the sample scripts, drop in your scenes, and share how it elevates your next localization.