

GLM-TTS is one of the stronger open-source TTS releases from late 2025 because it is framed as a production system, not just a lab demo.

The paper positions the model around three goals that matter in shipping teams: quality, controllability, and operational cost.

**Status note (as of February 14, 2026):**

The GitHub repo, inference scripts, and checkpoints are public. The README still marks RL-optimized weights and the 2D Vocos update as "coming soon," so separate what is currently runnable from what is paper-claimed.

## 60-second takeaway

- GLM-TTS uses a two-stage stack: autoregressive text-to-token generation, then flow-based token-to-waveform synthesis.
- The main technical bet is not one module; it is a bundled system: upgraded tokenizer + GRPO multi-reward RL + hybrid phoneme input + LoRA customization + vocoder upgrades.
- On Seed-TTS-eval zh (paper-reported), GLM-TTS is at CER 1.03 / SIM 76.1, and GLM-TTS\_RL improves to CER 0.89 / SIM 76.4.
- Several headline gains (phoneme-control and Vocos2D quality) are from internal evaluations, so treat them as promising, not yet independently verified.

## What GLM-TTS is trying to solve

The technical report argues that many modern zero-shot TTS systems still have five recurring production pain points:

- pronunciation control for polyphones and rare words
- emotional expressiveness without unstable tuning
- affordable voice customization without full-model finetuning
- robustness under real-world data noise
- quality retention while supporting streaming-like deployment patterns

GLM-TTS is designed as a direct response to those constraints.

# Architecture in one view

GLM-TTS follows the now-common hybrid pattern: text to discrete speech tokens, then tokens to waveform.

flowchart LR

```
A[Text] --> B[AR LLM<br/>Text to Speech Tokens]
P[Prompt Audio] --> C[Speech Tokenizer + Speaker Embedding]
C --> B
B --> D[Flow Model<br/>Tokens to Mel]
D --> E[Vocoder]
E --> F[Waveform]
```

The paper explicitly frames this as a production compromise between controllability and synthesis quality.

## The 6 design choices that matter

### 1) Tokenizer upgrades are central, not incidental

The speech tokenizer is upgraded from 12.5 Hz to 25 Hz and from a 16k to 32k vocabulary, with added pitch-estimation constraints.

The paper's own tokenizer ablation reports:

- SIM: 75.2 -> 76.1
- CER: 1.44 -> 1.03

This is important because many TTS stacks fail at the tokenizer layer before downstream modeling even has a chance.

### 2) GRPO-based RL is used as alignment, not as the whole training story

GLM-TTS applies GRPO with four rewards:

- CER (pronunciation accuracy)
- SIM (speaker similarity)
- Emotion
- Laughter

It also adds dynamic resampling and adaptive clipping to reduce reward collapse and reward hacking.

Paper-reported public benchmark impact:

- GLM-TTS: CER 1.03 / SIM 76.1
- GLM-TTS\_RL: CER 0.89 / SIM 76.4

So the RL stage looks like a measurable refinement layer rather than a complete redesign.

### **3) Phoneme-in gives explicit pronunciation control**

The system introduces a hybrid input mode where selected words/characters are replaced with phonemes while the rest stays as text.

Intended use case: polyphonic characters, rare terms, and educational reading scenarios where pronunciation precision matters more than pure free-form generation.

Internal ablation result in the paper:

- PER without Phoneme-in: 13.23
- PER with Phoneme-in: 5.14

This is one of the most practically relevant ideas for production localization and read-aloud products.

### **4) LoRA customization is positioned as cost control**

The report claims premium voice customization with approximately:

- 15% parameter adaptation
- around 1 hour single-speaker data
- around 80% lower training cost than full finetuning

For teams comparing adaptation paths, this is a strong claim, but it still needs independent replication under varied data quality.

### **5) Vocos2D is a vocoder-side quality push**

The paper proposes a 2D-convolutional Vocos variant with DiT-style residual structure.

Internal reported comparison (Vocos vs Vocos2D):

- NISQA: 3.16 -> 3.40

- UTMOS: 1.87 -> 1.91
- MOS: 3.58 -> 4.16

The qualitative claim is better frequency-subband modeling and more natural outputs.

## 6) Data pipeline discipline is treated as a first-class feature

The report emphasizes the processing stack itself: VAD, source separation, denoising, diarization, multi-ASR checks, WER filtering, punctuation optimization, and distributed processing.

For production teams, this is a reminder that much of TTS performance comes from data curation quality, not model architecture alone.

## Reported benchmark snapshot

### Seed-TTS-eval zh (paper/readme reported)

Model	CER (lower better)	SIM (higher better)	Open-source
MiniMax-Speech	0.83	78.3	No
Seed-TTS	1.12	79.6	No
VoxCPM	0.93	77.2	Yes
IndexTTS2	1.03	76.5	Yes
GLM-TTS	1.03	76.1	Yes
GLM-TTS_RL	0.89	76.4	Yes

### English split note

The report also lists test-en results for GLM-TTS at WER 2.23 / SIM 67.2 and GLM-TTS\_RL at WER 1.91 / SIM 68.1, while noting less English training data than Chinese.

## Repo reality check for engineers

If you want to run it today, the public repo includes inference and web demo entry points:

- `python glm tts_inference.py --data=example_zh --exp_name=_test --use_cache`

- `python -m tools.gradio_app`

Phoneme mode is wired in code via `--use_phoneme`.

The repo structure also exposes the major modules cleanly (llm/, flow/, grpo/, frontend/), which makes it easier to audit how paper claims map to implementation.

## **Where this fits vs your current Instavar TTS coverage**

You already have deep-dives or run reports for CosyVoice3, VoxCPM, Qwen3-TTS, IndexTTS2, and ReStyle-TTS.

GLM-TTS adds a useful new angle to that stack:

- stronger explicit emphasis on pronunciation control via hybrid phoneme input
- concrete GRPO-based reward design choices for emotional/prosodic alignment
- a clear cost-positioning narrative for voice customization

It should be treated as complementary coverage, not a replacement for your existing run-specific benchmark posts.

## **Practical caution before production decisions**

- Keep public-benchmark claims and internal-ablation claims in separate sections.
- Validate pronunciation control on your own hard-case word list before adopting Phoneme-in workflows.
- Re-test latency and long-form consistency on your actual deployment hardware.
- Track repo updates for the RL-optimized weights and Vocos2D release status.

## **Suggested evaluation checklist for next update**

1. Run the public checkpoint on your internal script set with and without --use\_phoneme.
2. Compare against your current baselines (VoxCPM, Qwen3-TTS LoRA, IndexTTS2) on the same prompts and scoring rubric.
3. Record failure taxonomy: polyphones, laughter artifacts, long-form drift, and prompt-noise inheritance.
4. Update this post once RL-optimized weights or Vocos2D release artifacts are public.

## Related Instavar TTS coverage

- [IMDA NSC Voice Cloning Finetuning Benchmark 2026](#) - run-specific benchmark summary across VoxCPM, Qwen3-TTS, IndexTTS2, and CosyVoice.
- [ReStyle-TTS and Relative Style Control in Zero-Shot TTS](#) - research briefing on relative style control workflows.
- [VoxCPM 1.5 LoRA Finetuning on IMDA NSC FEMALE 01](#) - practical LoRA run notes and checkpoint choices.
- [LoRA Fine-Tuning Qwen3-TTS for Custom Voices](#) - configuration and quality notes from our Qwen3-TTS run.
- [IndexTTS2 Finetuning on IMDA NSC FEMALE 01](#) - full-SFT baseline behavior and operational notes.

## Sources

- [GLM-TTS GitHub repository](#)
- [GLM-TTS Technical Report \(arXiv:2512.14291\)](#)
- [GLM-TTS PDF](#)
- [GLM-TTS Hugging Face model page](#)