

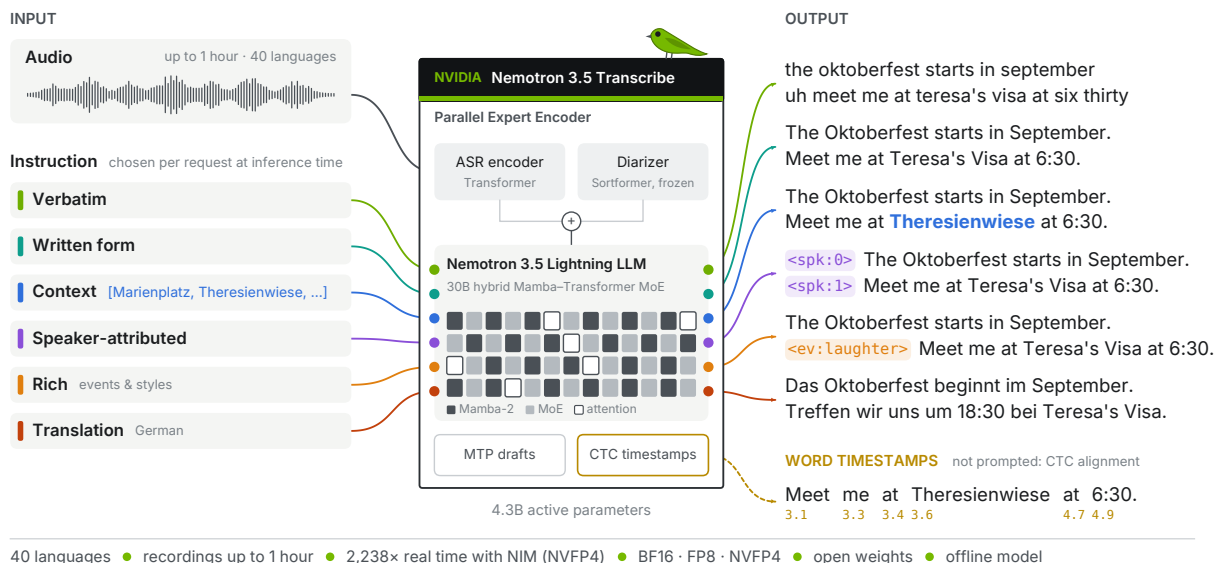
Nemotron 3.5 Transcribe Technical Report

NVIDIA

NVIDIA

Nemotron 3.5 Transcribe

One speech LLM. Many ways to write down what was said.



Abstract.

There is no single correct transcript of a recording: what counts as correct depends on who reads it and for what purpose, e.g., numbers in spoken or written form, verbatim or cleaned-up wording, or the script of code-switched speech. Conventional speech recognition systems fix one convention at training time, whereas speech-augmented language models (SALMs), which combine a speech encoder with a large language model (LLM), can be steered with natural language instructions. We present Nemotron 3.5 Transcribe, a SALM for such *intelligent transcription*. It couples a Transformer speech encoder and a frozen Nemotron 3 Diarization model with Nemotron 3.5 Lightning 30B-A3B, a hybrid Mamba-Transformer mixture-of-experts LLM, with approximately 4.3B active parameters. A single model performs verbatim, written-form, contextual, speaker-attributed, and rich transcription, speech translation, and spoken language identification in 40 languages, and returns speaker-attributed word-level timestamps on request. It also understands speech, sound, and music in support of transcription, but it is not a general-purpose assistant. We train it with a single supervised fine-tuning stage, followed by a short reinforcement learning stage. Nemotron 3.5 Transcribe achieves state-of-the-art accuracy on speaker-attributed ASR for overlapping multi-speaker speech and, to our knowledge, is the first SALM to outperform specialized cascaded systems on this task. It reaches a mean word error rate (WER) of 4.71% on the English and 4.62% on the multilingual Open ASR Leaderboard, the best among open-weights models, and given a list of expected entities, it reduces named-entity errors on ContextASR-Bench six-fold. We release the model in BF16, FP8, NVFP4, and GGUF versions under the OpenMDW-1.1 license, with a fine-tuning toolkit and agent skills in NeMo Speech.

Contents

1	Introduction	4
2	Model Architecture	5
2.1	Parallel Expert Encoder	6
2.2	Modality Adapter	7
2.3	Language Model Backbone	8
2.4	Multi-Token Prediction	8
2.5	Word Timestamps	8
2.6	Prompt Format	9
2.7	Quantized Checkpoints	9
3	Training	10
3.1	Training Data	10
3.2	Supervised Fine-Tuning	15
3.3	Reinforcement Learning	15
3.4	CTC Timestamp Adapter	17
3.5	DFlash2 Draft Model	18
4	Results	18
4.1	Evaluation Protocol	19
4.2	English ASR	20
4.3	Long-Form ASR	21
4.4	Entities, Numbers, and Hallucinations	21
4.5	Multilingual ASR	22
4.6	Code-Switched ASR	23
4.7	Contextual ASR	23
4.8	Multi-Speaker Transcription	25
4.8.1	Speaker-attributed ASR	25
4.8.2	Near-field transcription (experimental)	26
4.9	Multi-Speaker Word-Level Timestamping	27
4.10	Speech Translation	29
4.11	Spoken Language Identification	30
4.12	Accuracy of the Quantized Checkpoints	30
4.13	Deployment and Inference Efficiency	30
4.14	Audio Understanding	33
4.15	Voice Instruction Following	34

4.16 Text-Only Reasoning	35
5 Conclusions	36
Contributors	38
A Prompting Recommendations	48
A.1 Transcription	49
A.2 Numbers in Spoken Form	49
A.3 Code-Like Entities	50
A.4 Code-Switched Transcription	50
A.5 Contextual Biasing	50
A.6 Transcription Style	51
A.7 Multi-Speaker Transcription	51
A.8 Near-Field Transcription (Experimental)	51
A.9 Rich Transcription	51
A.10 Speech Translation	51
A.11 Language Identification	52
A.12 Audio Understanding	52
A.13 Voice Instruction Following	53
A.14 Text-Only Reasoning	54
B Fine-Tuning Recommendations	55
B.1 What Fine-Tuning Buys	55
B.2 Case Study: Medical ASR from Synthetic Data	57
B.3 Recommendations	58
C Detailed Results	59
C.1 MTP Acceptance by Dataset	59
C.2 MTP Head-Only Finetuning	61
C.3 DFlash2 Acceptance by Dataset	62
C.4 DFlash2 Supervision Comparison	62
D Probe: Transcription with PII Redaction	63
E Generation Loop Guardrails	64
F Rich Transcription Tags	66
G Voice Agent Applications	67

1. Introduction

Over the past few years, the state of the art in automatic speech recognition (ASR) has been driven by attention encoder-decoder (AED) and transducer models trained on large multilingual corpora, such as Whisper (Radford et al., 2023), Canary (Puvvada et al., 2024; Sekoyan et al., 2025), and Parakeet (Sekoyan et al., 2025). In parallel, speech-augmented language models (SALMs) (Chen et al., 2024a,b; Peng et al., 2025) extended a pretrained large language model (LLM) with a speech encoder, and recently started to match or outperform dedicated systems on the Open ASR Leaderboard (Srivastav et al., 2026; NVIDIA, 2025a). The appeal of SALMs goes beyond accuracy: the LLM decoder makes transcription *promptable*, so a user can describe the desired output format, supply domain vocabulary, or ask for a translation instead of a transcript, all with a single model.

In practice, this flexibility is rarely exploited to its full extent. Most speech LLMs are trained either for a single transcription task, or for a broad set of audio understanding tasks at the expense of transcription quality. Transcription itself is also more than a single task. Real-world deployments require verbatim or written-form output depending on the downstream consumer, recognition of rare entities such as names, product codes, or e-mail addresses, attribution of words to speakers in meetings and calls, and robust handling of hour-long recordings, code-switching, and non-speech audio. Supporting all of these in a single model is a data and training problem as much as a modeling one: each capability competes for the same training budget, and the mixture determines which capabilities survive.

In this work, we present Nemotron 3.5 Transcribe (released as Nemotron-3.5-Transcribe-30B-A4B), a speech LLM for *intelligent transcription*, built on Nemotron 3.5 Lightning 30B-A3B (NVIDIA, 2025f,e), a hybrid Mamba-Transformer mixture-of-experts (MoE) model. Figure 1 illustrates the range of tasks it performs. By intelligent transcription, we mean state-of-the-art accuracy on established ASR benchmarks, combined with output that follows natural language instructions about its form, vocabulary, and speakers, including for long, multi-speaker, and multilingual recordings. The model also answers questions about speech, sound, and music and responds to spoken requests. We include these capabilities because we expect them to support transcription, e.g., by helping the model follow instructions, use context, and handle non-speech audio, but they are not goals in themselves, and we do not optimize the model to compete with general-purpose omni models on them. Preserving the full text-only capabilities of the underlying LLM is explicitly not a goal. Nemotron 3.5 Transcribe should therefore be treated as a specialized model for transcription rather than as a general-purpose assistant.

The model pairs the LLM with a Parallel Expert Encoder that fuses a trainable ASR encoder with a frozen speaker diarization model. We train it with supervised fine-tuning on a large multi-task mixture, in which each ASR example can be used with one of many task instructions, followed by a short reinforcement learning stage that optimizes transcription accuracy.

Our contributions are as follows:

1. We introduce Nemotron 3.5 Transcribe, a single model for verbatim, written-form, contextual, multi-speaker, and rich transcription, speech translation, and language identification, transcribing 40 languages, with word-level speaker-attributed timestamps from a lightweight CTC alignment adapter.
2. We describe a Parallel Expert Encoder that injects speaker activity from a frozen diarization model into the ASR representations, which enables speaker-attributed transcription with serialized output training (SOT) (Kanda et al., 2020) without architectural changes to the LLM.
3. We describe the training data mixture and the sampling strategy used to balance supervision

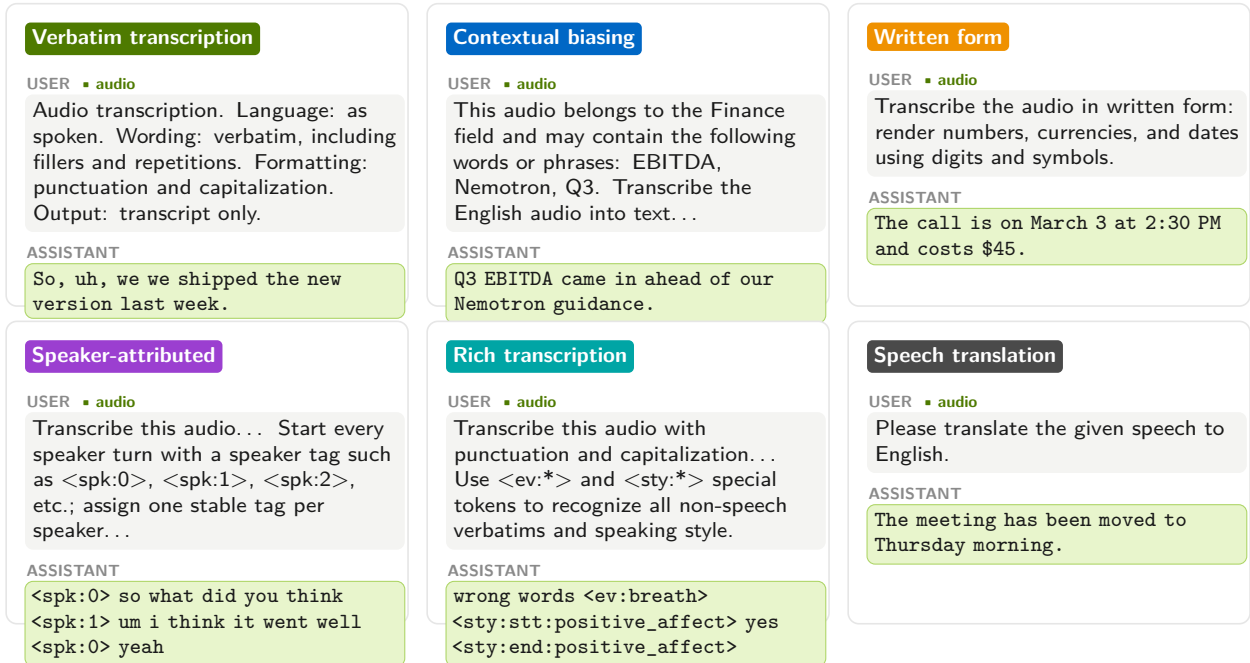


Figure 1 | A single Nemotron 3.5 Transcribe checkpoint performs a variety of transcription tasks selected with a natural language instruction. The instructions shown are those used in our evaluation (Appendix A) or in training, abbreviated with “...”; the assistant turns are illustrative of the output format and are not model samples. Audio is inserted into the user turn as encoder embeddings.

across tasks, languages, and data sources, together with an indexed, exactly resumable data loader.

4. We show that joint multi-token prediction (MTP) training enables speculative decoding with the model’s own MTP layer in vLLM (Kwon et al., 2023), and we provide a separately trained lightweight draft model based on DFlash2 (Chen et al., 2026a; Inco AI, 2026) as an alternative.
5. We evaluate the model on a broad set of benchmarks for transcription, translation, and language identification, as well as for audio understanding and voice instruction following, and list the exact instructions we used as practical recommendations for prompting.
6. We release a fine-tuning toolkit in NeMo Speech and show on 17 tasks that the model can be adapted to new languages, domains, and tasks on a single GB300 GPU, typically within 0.5–2.3 GPU-hours per task, and in some cases with only minutes of audio.
7. We release the model checkpoint and the training recipe.

The remainder of this report is organized as follows. Section 2 describes the model architecture, Section 3 the training data and procedure, Section 4 the evaluation results, and Section 5 concludes. Practical guidance on prompting and fine-tuning is given in Appendices A and B.

2. Model Architecture

Nemotron 3.5 Transcribe follows the SALM design (Chen et al., 2024a; NVIDIA, 2025a): a speech encoder converts audio into a sequence of embeddings, which are inserted into the prompt of a decoder-only LLM in place of an audio placeholder token, and the LLM generates the response autoregressively. Figure 2 shows the overview. We depart from earlier SALMs in four places: the encoder is a Parallel Expert Encoder that combines an ASR encoder with a speaker diarization

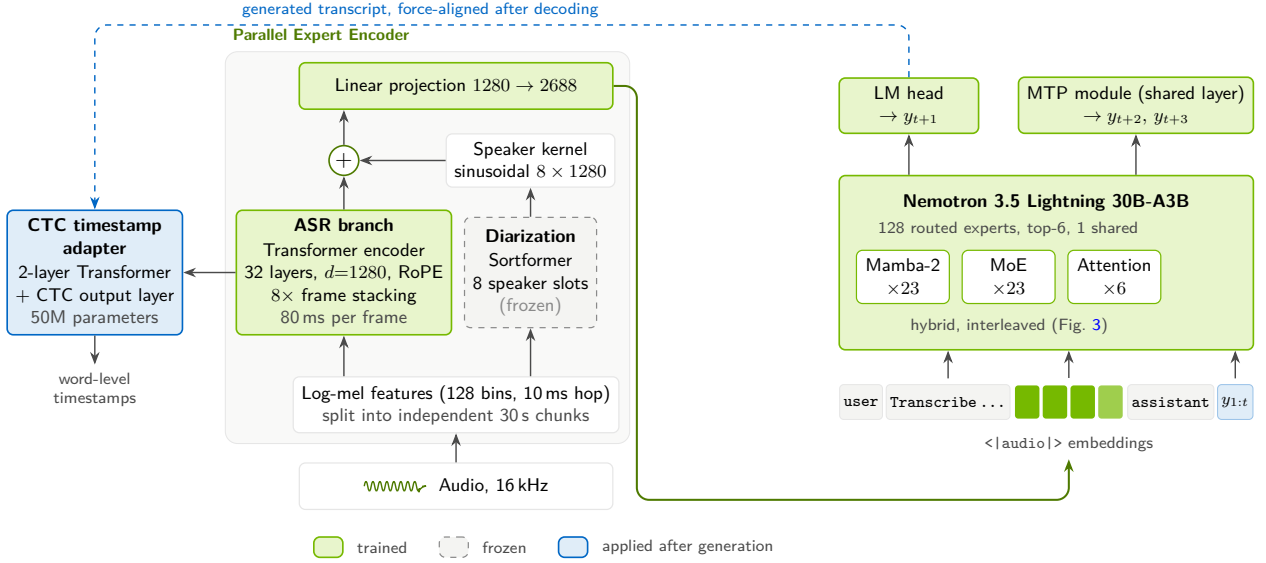


Figure 2 | Overview of Nemotron 3.5 Transcribe. Log-mel features of each 30-second audio chunk are processed by two branches of the Parallel Expert Encoder: a trainable Transformer ASR encoder and a frozen Sortformer diarization model. Per-frame speaker activities are mapped through a fixed sinusoidal kernel and added to the ASR representations, which are then projected to the LLM width and spliced into the prompt at the position of the `<|audio|>` token. The LLM predicts the next token with its native head and two further tokens with a shared MTP layer, which also serves as the draft model at inference time. After decoding, a CTC timestamp adapter on the ASR encoder output force-aligns the generated transcript to the audio to produce word-level timestamps (Section 2.5).

model (Section 2.1); the LLM is a hybrid Mamba–Transformer MoE (Section 2.3); the model is trained jointly with multi-token prediction, which doubles as a draft model for speculative decoding (Section 2.4); and a lightweight CTC adapter aligns the generated transcript to the audio to produce word timestamps (Section 2.5). Table 1 summarizes the configuration.

2.1. Parallel Expert Encoder

ASR branch. The ASR branch is a Transformer encoder with 32 layers, model dimension 1280, 10 attention heads, and a feed-forward expansion factor of 4, using rotary position embeddings and query-key normalization. It operates on 128-bin log-mel features of 16 kHz audio computed with a 10 ms hop, which are subsampled 8x by frame stacking, so that each output frame covers 80 ms of audio. The branch has approximately 0.6B parameters and is initialized from a multilingual ASR encoder pretrained as a token and duration transducer (TDT) on Granary-v2 (Granary-v2 Team, 2026) (Section 3.1).

Diarization branch. In parallel, the Nemotron 3 Diarization model, a streaming Sortformer speaker diarization model (Park et al., 2025) with approximately 100M parameters, estimates per-frame activity probabilities for up to eight speakers. We binarize the activities at a threshold of 0.5, normalize them, and project them through a fixed sinusoidal kernel of size 8×1280 to obtain a speaker embedding for every encoder frame, which is added to the normalized ASR representations. The diarization branch stays frozen throughout training; when reference speaker segmentation is available for a training example, it replaces the diarizer’s predictions. In effect, the LLM receives acoustic representations that are already tagged with “who spoke when”, and has to learn to translate

Table 1 | Configuration of Nemotron 3.5 Transcribe. Parameter counts are approximate.

Component	Property	Value
ASR encoder	Architecture	Transformer, RoPE, QK-norm
	Layers / width / heads	32 / 1280 / 10
	Input / frame rate	128 log-mel / 80 ms
	Parameters	0.6B
Diarization	Model	Nemotron 3 Diarization (Sortformer), frozen
	Speakers / parameters	8 / 0.1B
Adapter	Projection	Linear, 1280 $\rightarrow$ 2688
LLM	Model	Nemotron 3.5 Lightning 30B-A3B
	Layers	52 (23 Mamba-2, 23 MoE, 6 attention)
	MoE	128 experts, top-6, 1 shared
	Attention	32 query / 2 KV heads
	Parameters (total / active)	31.6B / 3.5B
MTP	Depths / layers	2 / 1 shared (attention + MoE)
	Parameters	1.3B
Timestamps	CTC adapter	2-layer Transformer, 50M
Model	Parameters (total / active)	34B / 4.3B
	Max. recording length (training)	1 hour
	Max. sequence length (training)	65,536 tokens

these tags into speaker labels in the output (Section 3.1). Since the diarization model also processes 30-second chunks, its speaker indices are local to a chunk, and the LLM is responsible for keeping the speaker labels consistent across a long recording. This design keeps the speaker information available to the LLM without modifying its architecture or its input sequence length.

Chunked encoding. Both branches process the input in independent 30-second chunks (375 output frames each), with no attention across chunk boundaries. Chunks from all examples in a mini-batch are packed into a single sequence with variable-length attention, so that the encoder memory cost grows linearly with the total audio duration and no compute is spent on padding. The encoder outputs of consecutive chunks are concatenated before entering the LLM, which handles the long-range dependencies. The chunking is an internal processing choice: it does not limit the recording length, which is bounded only by the LLM context. The model is trained on recordings of up to one hour, which corresponds to 45,000 audio embeddings, and with sequences of up to 65,536 tokens. The chunking is not needed to save memory: with FlashAttention (Dao et al., 2022; Dao, 2024), attention over the full recording would also need memory linear in its length. We chunk for two other reasons: it reduces the attention compute of the encoder from quadratic to linear in the recording length, and the diarization model was trained on short contexts only.

2.2. Modality Adapter

The encoder outputs are mapped to the LLM embedding space with a single linear projection from 1280 to 2688 dimensions, without any additional adapter layers. The prompt contains one `<|audio|>` placeholder token per audio input, which is replaced with the projected encoder frames during the forward pass. Thus, one second of audio corresponds to 12.5 LLM input positions.

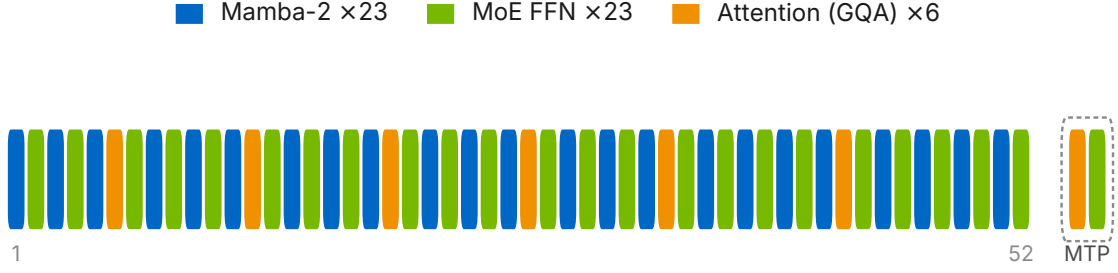


Figure 3 | Layer pattern of the Nemotron 3.5 Lightning 30B-A3B backbone. Attention layers make up fewer than 12% of the layers, which keeps the KV cache small for long audio prompts. The dashed block is the MTP module (Section 2.4).

2.3. Language Model Backbone

The LLM is Nemotron 3.5 Lightning 30B-A3B, a member of the Nemotron family of hybrid Mamba-Transformer MoE models (NVIDIA, 2025f,e). It consists of 52 layers: 23 Mamba-2 (Gu and Dao, 2024) layers, 23 MoE feed-forward layers, and only 6 grouped-query attention layers (Fig. 3). Each MoE layer has 128 routed experts with top-6 routing and one shared expert, which results in approximately 3.5B active parameters per token out of approximately 31.6B total. Together with the remaining components (the ASR encoder, the diarization model, the modality adapter, and the CTC adapter; approximately 0.8B parameters), Nemotron 3.5 Transcribe has approximately 4.3B active parameters, hence the model name Nemotron-3.5-Transcribe-30B-A4B. With the MTP module (approximately 1.3B parameters), the checkpoint contains approximately 34B parameters in total. The small number of attention layers keeps the key-value cache small, which is particularly important for transcription of long recordings.

2.4. Multi-Token Prediction

We train the model jointly with a multi-token prediction (MTP) objective (Gloeckle et al., 2024; DeepSeek-AI, 2024). The MTP module consists of one attention block and one MoE block, reused across two prediction depths with shared parameters. During training, the first depth combines the backbone’s hidden state at position t with the ground-truth token embedding at position $t + 1$ to predict the token at $t + 2$. The second depth takes the first depth’s output hidden state and the ground-truth token embedding at $t + 2$ to predict the token at $t + 3$. Both depths share the backbone’s language-model output head. The MTP loss is averaged across the two depths and added to the main cross-entropy loss with a weight of 0.1. At inference time, the MTP module serves as a draft model for speculative decoding (Section 4.1), conditioning on generated tokens instead of ground-truth tokens. Since the module is shared between depths, it can be applied more than twice at inference time to propose more draft tokens.

2.5. Word Timestamps

The released checkpoint includes a connectionist temporal classification (CTC) adapter that aligns the generated transcript to the audio without modifying the generated text. The adapter reuses the 1,280-dimensional representations of the ASR encoder, without a separate speech encoder. A learned one-dimensional transposed convolution with kernel size and stride of two upsamples the encoder output from 80 ms to 40 ms frame intervals. Two pre-normalized Transformer layers, each with 16 attention heads, a hidden dimension of 1,280, and a feed-forward dimension of 640, process these

representations, using rotary positional embeddings and query/key normalization; an additional residual connection, scaled by 0.1, bypasses the two-layer stack. A pointwise projection produces probabilities over a dedicated multilingual SentencePiece BPE vocabulary of 23,862 tokens, which includes special tokens for speakers, non-verbal events, and speaking styles, plus the CTC blank symbol. The adapter has approximately 50.25M parameters: 3.28M in the upsampling convolution, 16.41M in the Transformer layers and their normalization layers, and 30.57M in the output projection.

After text generation, CTC forced alignment uses max-sum dynamic programming (DP) and backtracing to obtain token alignments, which are aggregated into word timestamps. The transcript’s speaker tags and the diarization model’s speaker-activity timestamps organize alignment into speaker-specific DP matrices. Diarization segment boundaries further refine word onsets and offsets, particularly for the first and last words of each speech segment. The resulting output is a word-level, speaker-attributed transcript with timestamps and a mapping between transcript speaker tags and diarization speakers.

2.6. Prompt Format

The model uses the chat template of the Nemotron 3.5 LLM. Each turn is delimited by `<|im_start|>role` and `<|im_end|>` tokens, and an empty system turn is inserted when no system prompt is given. With reasoning disabled, which is the default for transcription, the assistant turn begins with an empty reasoning block `<think></think>` and the model answers directly; with reasoning enabled, the model first generates a reasoning trace inside this block (Section 4.1). A typical transcription request looks as follows:

```
<|im_start|>system
<|im_end|>
<|im_start|>user
Audio transcription. Language: as spoken. Wording: verbatim, including fillers and
  repetitions. Formatting: punctuation and capitalization. Output: transcript only.<|
  audio|><|im_end|>
<|im_start|>assistant
<think></think>So, uh, we we shipped the new version last week.<|im_end|>
```

Speaker labels (`<spk:0>` to `<spk:7>`) and the rich transcription tags (Appendix F) are dedicated tokens occupying previously unused slots in the LLM tokenizer.

2.7. Quantized Checkpoints

In addition to the BF16 checkpoint, we release FP8 and NVFP4 checkpoints, which we obtain with post-training quantization in NVIDIA Model Optimizer (NVIDIA, 2026a). The *FP8* checkpoint targets GPUs with FP8 support (Ada Lovelace, Hopper, and Blackwell; Table 17):

- In the LLM and the MTP module, every linear layer has FP8 (E4M3) weights with one scale per tensor, and its input is quantized to FP8 with a static per-tensor scale. The embeddings, the output head, the convolutions of the Mamba layers, and the routers of the MoE layers stay in BF16. The scales of the LLM were calibrated on 128 text sequences of up to 512 tokens; those of the MTP module on its inputs while it drafts tokens for 512 held-out English utterances.
- The KV cache of the attention layers is stored in FP8, with static scales from the same calibration.
- In the ASR encoder, the attention and feed-forward projections of all 32 layers (128 linear layers) are quantized in the same way, with input scales derived from the maximum activations on the same 512 utterances and a safety margin of 1.5. The subsampling layer, the diarization model, the modality adapter, and the CTC timestamp adapter stay in BF16.

This halves the size of the checkpoint, from 67.5 GB to about 35 GB. The *NVFP4* checkpoint, for Blackwell GPUs, mixes precisions: the attention projections and the routed and shared experts of the LLM use NVFP4, i.e., 4-bit floating-point weights and activations with one FP8 scale per block of 16 values and one FP32 scale per tensor, whereas the projections of the Mamba layers, the MTP module, and the ASR encoder use FP8 as above. Its KV cache is stored in FP8 at serving time, without calibrated scales. The NVFP4 checkpoint occupies about 21 GB. Both checkpoints are served with the same vLLM plugin as the BF16 checkpoint and support all speculative decoding modes (Section 4.1). We evaluate the accuracy of both checkpoints in Section 4.12 and their speed in Section 4.13.

3. Training

Figure 4 summarizes how Nemotron 3.5 Transcribe is trained. Starting from a pretrained LLM and pretrained speech encoders, we first train the model with supervised fine-tuning (SFT) on a large multi-task mixture, and then apply a short reinforcement learning (RL) stage that optimizes transcription accuracy directly. We do not use a separate modality alignment stage with a frozen LLM: all components except the diarization model are trained jointly from the first step. Finally, we train two optional inference components on top of the frozen model: the CTC timestamp adapter for word timestamps (Section 3.4) and a draft model for speculative decoding (Section 3.5).

3.1. Training Data

Granary-v2 is a multilingual speech dataset and a reusable data curation and labeling system (Granary-v2 Team, 2026). We use its pipeline to generate or improve transcripts and derive supervision for ASR on short and long recordings, text normalization, disfluency control, contextual ASR, code-switched transcription, speech question answering (SpeechQA), speech captioning, spoken language identification, and speech translation. The standalone Granary-v2 report describes the pipeline and public dataset in detail. The public release contains the shareable portion of the data curated with this pipeline; the full training mixture also includes private multilingual speech data and the additional task-specific sources described below.

Figure 5 shows the probability of sampling a training example from each data source. Four sampling groups account for more than 95% of the examples: Granary-v2 pseudo-labeled data, Granary-v2 human-label-improvement data, Granary-v2 speech translation, and internal multilingual speech data. Granary-v2 also supplies long-form ASR, Indian English, and language identification examples in separate sampling groups. Granary-v2 extends the original Granary corpus (Koluguri et al., 2025). The remaining sources each provide a specific capability, such as multi-speaker transcription or audio understanding, and are sampled with probabilities between 0.01% and 3%. The speech data covers 77 languages. 41 of them have at least 370 hours of distinct audio each (Table 2), and we support 40 of these for transcription: the training data for Norwegian did not yield usable quality (Appendix C). For four languages, this includes regional variants: English from the United States, India, and the United Kingdom; Spanish from the United States and Spain; French from France and Canada; and Portuguese from Portugal and Brazil. English accounts for 36% of the audio seen during training, the next eight languages for 2.5–7% each, and the remaining 32 for less than 2.5% each. The other 36 languages appear with about 300 hours in total and are not officially supported. In addition, we used 0.35 million hours of audio without a language label, mostly from audio and speech understanding sources, which make up 5.6% of the audio seen during training. Table 3 shows the distribution over task categories: transcription without and with context makes up over 70% of the examples, and speech translation 17%.

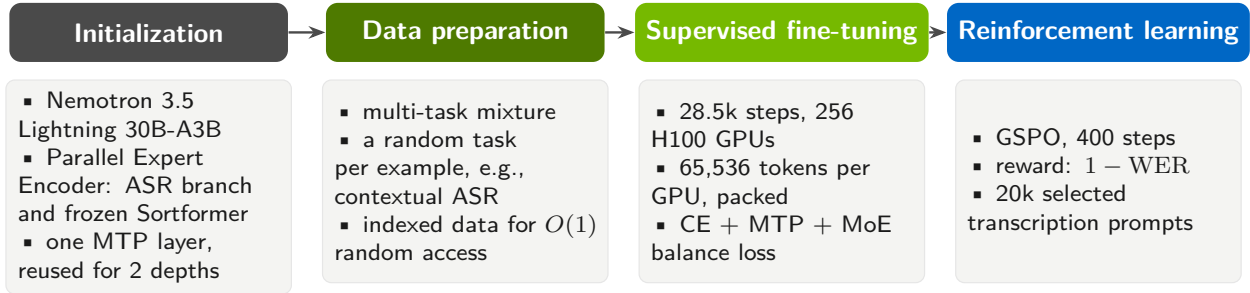


Figure 4 | Training pipeline of Nemotron 3.5 Transcribe. Supervised fine-tuning trains the ASR encoder, the projection, the full LLM including the MoE routers, and the MTP module jointly on a multi-task mixture; reinforcement learning then optimizes transcription accuracy with a word error rate reward.

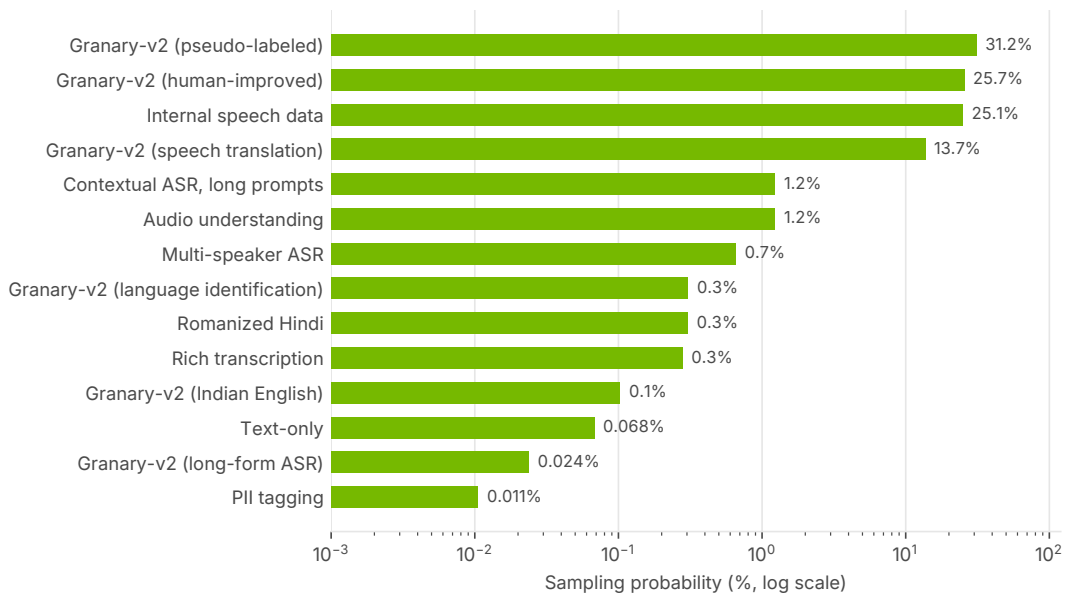


Figure 5 | Sampling probability of each data source at the end of supervised fine-tuning (log scale). The main Granary-v2 groups (pseudo-labeled data, human-label-improvement data, and speech translation) and internal speech data make up more than 95% of the examples. Granary-v2 long-form ASR, Indian English, and language identification are separate sampling groups. The pseudo-labeled, human-label-improvement, and internal speech groups supply multiple tasks, including Contextual ASR, SpeechQA, and captioning. The separate Contextual ASR group with long prompts uses other data. Each capability-specific source is sampled with a probability between 0.01% and 3%. Examples from the capability-specific sources tend to have longer targets (e.g., speaker-tagged transcripts of long recordings), so their share of the supervised tokens is larger than their share of the examples.

Up to the SFT checkpoint used for the released model, the model saw approximately 1.78 billion audio examples with 6.5 million hours of audio. They were drawn from 372 million distinct examples with 1.7 million hours of audio, i.e., each example is seen about five times on average; audio shared by examples of different tasks, such as the transcription and the translation of the same recording, is counted once per example. In terms of tokens, the model processes 479B multimodal tokens: approximately 294B audio embeddings and 185B text tokens, which include the instructions and responses of the speech examples. Only 5.6B text tokens come from text-only training data. These numbers are estimated from the sampler state of the checkpoint.

Table 2 | Languages of the training audio up to the SFT checkpoint used for the released model. Unique: hours of distinct training examples seen at least once (thousands; audio shared by examples of different tasks is counted for each example). Share: percentage of all audio hours seen during training, including repetitions. Estimated from the sampler state of the checkpoint.

Language	Unique (kh)	Share (%)	Language	Unique (kh)	Share (%)
English	474.5	35.67	Thai	14.4	1.01
Spanish	99.1	6.56	Lithuanian	13.6	0.64
French	90.6	6.61	Danish	13.4	0.64
Portuguese	67.9	4.58	Latvian	12.0	0.57
German	67.0	4.35	Estonian	10.1	0.49
Japanese	60.4	6.38	Slovak	9.8	0.48
Italian	57.2	3.63	Slovenian	7.7	0.41
Russian	53.9	3.72	Vietnamese	7.7	0.57
Hindi	37.3	2.56	Croatian	6.9	0.37
Dutch	26.6	1.47	Maltese	6.1	0.32
Polish	24.8	1.13	Hebrew	5.4	0.55
Chinese (Mandarin)	23.2	1.63	Turkish	4.2	0.21
Korean	22.9	1.77	Tamil	3.6	0.36
Czech	21.9	0.97	Bengali	1.7	0.19
Romanian	19.2	0.85	Ukrainian	1.5	0.12
Hungarian	18.4	0.85	Telugu	0.53	0.046
Bulgarian	18.4	0.81	Indonesian	0.51	0.053
Greek	17.4	0.79	Norwegian	0.45	0.023
Swedish	16.9	0.78	Filipino	0.40	0.037
Arabic	16.4	1.37	Marathi	0.37	0.042
Finnish	14.8	0.69			
36 other languages				0.31	0.048
No language label (mostly audio understanding)				349.1	5.63
Total				1,718.5	100.00

One example, many tasks. Examples from Granary-v2, our internal multilingual speech data, and the human-transcribed multilingual data provide alternative instructions and targets for multiple tasks. Depending on the available labels, these include verbatim transcription with punctuation and capitalization, written-form transcription (e.g., numbers written as digits), written-form transcription with disfluencies removed, code-switched transcription, five variants of Contextual ASR, spoken question answering, and audio captioning. Context and translation labels are generated for selected high-quality records. Each time an example is drawn, one of its available tasks is selected at random with the probabilities shown in Fig. 6. This turns the large ASR corpora into a multi-task corpus without duplicating any audio, and exposes the model to many phrasings of each instruction.

Contextual ASR. The Granary-v2 pipeline generates Contextual ASR labels (Granary-v2 Team, 2026). In Contextual ASR examples, the instruction contains a list of words or phrases that may occur in the audio, optionally with a description of the domain. We use five variants: *fine* context lists the entities that occur in the transcript; *coarse* context provides only a domain label; *partial* context lists only some of the occurring entities; *distractor* context mixes occurring entities with similar-sounding or related ones that are not spoken; and *negative* context lists only entities that do not occur. The last two variants matter in practice, since users typically provide a vocabulary for a whole session, most of which is absent from any given utterance. Contextual ASR data with long lists of terms, for single- and multi-speaker recordings, comes from separate sources and enters the

Table 3 | Task categories of the training audio up to the SFT checkpoint used for the released model: examples seen (millions, including repetitions), their share, audio hours seen (thousands, including repetitions), and hours of distinct examples (thousands). Text-only data is excluded.

Task category	Examples (M)	Share (%)	Hours (kh)	Unique (kh)
ASR and text normalization	997.6	56.0	3,121	725.9
Speech translation	310.4	17.4	792	178.9
Contextual ASR	276.0	15.5	674	148.2
Speech question answering	90.1	5.1	757	92.5
Speech captioning and summaries	75.8	4.3	221	53.9
Sound and music understanding	17.5	1.0	349	346.8
Speaker-attributed ASR	7.2	0.4	584	169.6
Language identification	5.3	0.3	16	1.4
PII-aware ASR	0.2	<0.1	15	1.3
Total	1,780.0	100.0	6,528	1,718.5

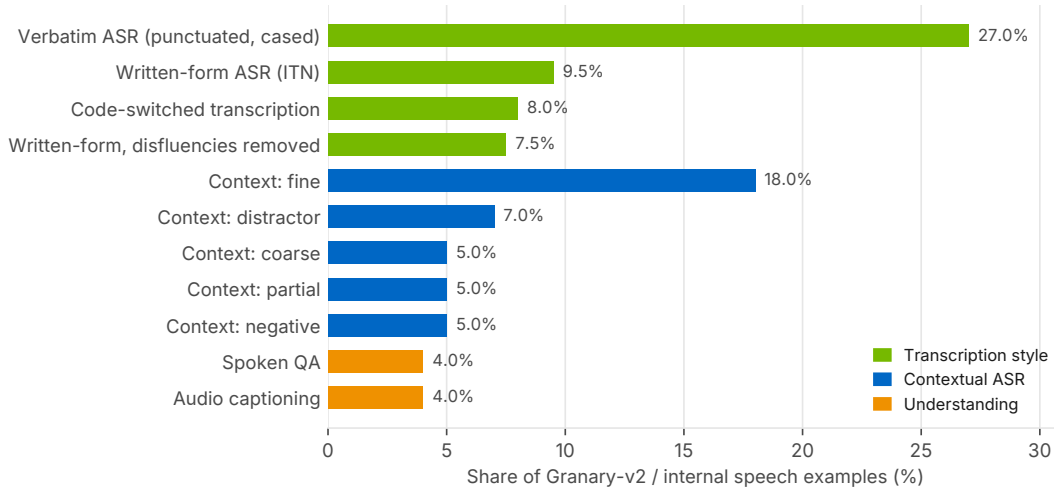


Figure 6 | Probability of selecting each task when an example is drawn from the main Granary-v2 and internal speech sources. These sources provide both transcription and speech understanding supervision; Granary-v2 speech translation is sampled as a separate group in Fig. 5. Contextual ASR accounts for 40% of these examples; the negative and distractor variants teach the model to ignore listed terms that are not spoken.

mixture only at step 25,500 of the 28,500 training steps (see *Changes to the mixture during training* below). It forms the separate long-prompt group in Fig. 5.

Multi-speaker transcription. For speaker-attributed transcription, we use serialized output training (SOT) (Kanda et al., 2020): the target is a single transcript in which every speaker turn starts with a speaker tag `<spk:k>`, and each speaker keeps the same tag throughout the recording. The data includes conversational and meeting recordings, such as Fisher (Cieri et al., 2004), AMI (Carletta et al., 2006), ICSI (Janin et al., 2003), DiPCo (Van Segbroeck et al., 2020), AISHELL-4 (Fu et al., 2021), and AliMeeting (Yu et al., 2022), as well as simulated mixtures based on Multilingual LibriSpeech (Pratap et al., 2020), with segments ranging from a few seconds to full sessions and up to eight speakers. A variant of the task also predicts the start and end time of each turn.

Rich transcription. For rich transcription, the targets contain inline tags for ten types of non-verbal events (e.g., `<ev:laughter>`, `<ev:breath>`) and paired start and end tags for eleven speaking styles (e.g., `<sty:stt:positive_affect> ... <sty:end:positive_affect>`), annotated on 2.3k hours of speech. The full list of tags is given in Appendix F, and the corresponding instruction in Appendix A. We release this capability without a quantitative evaluation in this report.

Speech translation. Granary-v2 also provides speech translation supervision between English and other languages, in both directions. These examples form the Granary-v2 speech translation group in Fig. 5.

Other Granary-v2 groups. Granary-v2 also provides ASR training examples with recordings much longer than the 30-second encoder chunks, and Indian-accented English training data. It also provides language identification data for 48 languages, with 100 different instruction phrasings. Long-form ASR, Indian English, and language identification form separate Granary-v2 sampling groups in Fig. 5.

Other capabilities. Additional training data supports the following behaviors:

- **Script control:** romanized Hindi on request, and Latin-script output for Indian-accented English from Granary-v2.
- **Audio understanding:** non-transcription, filtered subset of Nemotron 3 Nano Omni (NVIDIA, 2026b) training data, covering questions and reasoning about speech, sound, and music.
- **PII tagging:** transcripts in which personally identifiable information is replaced with typed tags, such as `<email>` (see Appendix D for a small probe of this capability).
- **Text-only** instruction data at a very small rate (0.07% of the examples), which mainly supports general instruction following; enough to preserve some, but not full, text-only capabilities of the LLM.

Mixture weights. A natural way to balance a speech mixture is by hours or by number of examples. For a model trained on many tasks, both are misleading: what drives learning is the number of supervised (answer) tokens, which varies by orders of magnitude across tasks, from a single language label to a speaker-tagged transcript of an hour-long meeting. We therefore derived the initial mixture weights from the *answer-token mass* of each source, measured with the actual prompt formatting on a sample of 4,096 examples per source, and applied temperature sampling with $T = 0.5$ at every level of the blend hierarchy (sources, then datasets within a source). This keeps the distribution of the supervision stable throughout training across tasks, languages, and datasets, in the spirit of our earlier work on multimodal data sampling (Želasko et al., 2025a). We then adjusted a few weights manually, e.g., the audio understanding and text-only sources were down-weighted to favor transcription tasks. Translation data was sampled at only 75% of its size-appropriate weight. Within it, the subset translating into English was capped at 20% (down from about 80%), since early checkpoints tended to spuriously translate multilingual ASR data.

Changes to the mixture during training. We changed the mixture at a few points during training. Non-speech audio with empty targets was used only during the first half of training (up to step 15,000). Late in training, we found that part of this data was poorly filtered and contained speech, which caused rare empty transcripts of actual speech. Additional multi-speaker data were

added at step 16,500. At step 25,500, we added four sources, taking their share from the translation data: Contextual ASR data with long term lists, for single- and multi-speaker recordings; additional multi-speaker data; and more high-quality human-transcribed speech in 19 languages.

Data loading. All data is read with Lhotse (Želasko et al., 2021). Each dataset is stored in a JSONL manifest, most with accompanying WebDataset tar shards of audio files. All data is indexed for random access to individual examples, and the per-example indexes are gathered into a few memory-mapped index files, one per data group. This makes the data loader stateful and exactly resumable: a training job that restarts after an interruption continues with the next unseen example, rather than re-sampling the mixture from scratch, which keeps the realized mixture close to the intended one over many restarts. Instead of the duration bucketing used in our earlier work on encoder-decoder models (Želasko et al., 2025b), we pack variable-length examples into packed sequences with a fixed budget of 65,536 tokens per GPU, so that almost no compute is spent on padding, in the encoder and in the LLM alike.

3.2. Supervised Fine-Tuning

Objective. We train with the cross-entropy loss on all assistant turns, including the end-of-turn token; system and user turns, including the audio embeddings, are excluded from the loss. The loss is summed over tokens and normalized by the total number of supervised tokens across all GPUs, so that every supervised token has the same weight regardless of how examples are packed. We add the MTP loss with a weight of 0.1 (Section 2.4) and an MoE load-balancing loss with a weight of 10^{-4} .

Optimization. Table 4 lists the hyperparameters. We train all parameters except the diarization model and the feature extractor, including the MoE router weights, with AdamW, a peak learning rate of 9×10^{-5} , 500 warmup steps, and cosine decay over 30,000 steps. We use the checkpoint after 28,500 steps for the reinforcement learning stage. We train in BF16, but keep the parameters that are sensitive to numerical precision in FP32: the state-space parameters of the Mamba layers (A , D , and the time-step bias) and the correction biases of the MoE routers. We also keep FP32 master weights in the fused AdamW optimizer of Transformer Engine; together, these choices made convergence much more stable.

Infrastructure. We implement the model in the `speechlm2` collection of NeMo (Kuchaiev et al., 2019), using NeMo Automodel for the LLM. Training runs on 32 nodes with 8 NVIDIA H100 GPUs each, with 8-way expert parallelism and fully sharded data parallelism (FSDP2), Transformer Engine (NVIDIA, 2026d) kernels, grouped GEMM for the experts, DeepEP (DeepSeek-AI, 2025) for token dispatch, and activation checkpointing. The training consists of many consecutive jobs, each of which restores the full training state, including the state of the data loader.

3.3. Reinforcement Learning

Starting from the SFT checkpoint after 28,500 steps, we apply a short reinforcement learning stage that optimizes transcription accuracy directly, using Group Sequence Policy Optimization (GSPO) (Zheng et al., 2025) as implemented in NeMo-RL (NVIDIA, 2025b).

Data selection. We mine a compact training set from a pool of approximately 239k candidate recordings drawn from training sources, including conversational and read English, Indian-accented

Table 4 | Supervised fine-tuning hyperparameters.

Hyperparameter	Value
Optimizer	AdamW ($\beta_1=0.9$, $\beta_2=0.999$, $\epsilon=10^{-8}$)
Weight decay	0.05
Peak / final learning rate	9×10^{-5} / 7.5×10^{-7}
Schedule	500 warmup steps, cosine decay over 30,000 steps
Steps used	28,500
Gradient clipping	1.0
Tokens per GPU	65,536 (packed)
GPUs	$256 \times \text{H100}$
Loss weights	CE 1.0, MTP 0.1, MoE balance 10^{-4}
Parallelism	8-way expert parallelism, FSDP2
Precision	BF16; Mamba state-space and router bias parameters in FP32

English, multilingual ASR, speaker-attributed transcription, and non-speech audio. We sample 16–64 responses per candidate and retain examples with an accurate or near accurate, completed transcript and a substantially worse alternative, measured by WER or by CER for Mandarin. Negative examples can include insertion-heavy or repetitive outputs, including those reaching the output cap when the error contrast is sufficiently strong. Non-speech examples require an empty normalized positive response.

The resulting 20,000 examples comprise 19,709 ordinary ASR, 201 speaker-attributed, and 90 non-speech examples. English accounts for 59.02% of all examples, including Indian-accented speech; the largest remaining groups are Hindi (1,629), Mandarin (1,627), and Dutch (1,175), followed by Portuguese, Italian, German, French, and Spanish, with additional Thai and Korean examples. Sources, languages, and instructions are interleaved in a fixed order, with soft composition targets rather than exact quotas.

We used nine user prompts in total: eight variants for ordinary ASR and one for speaker-attributed transcription, aligned with the task instructions used during evaluation. All examples used an empty system message, with the user instruction preceding the audio. Each example retained the same prompt during data selection and RL training.

Rewards. The reward is $r = 1 - e$, where e is transcript WER, or CER for Mandarin ASR. WER normalization case-folds text and separates punctuation while preserving Indic combining marks; Mandarin uses character-level normalization with whitespace adjacent to Han characters removed. Evaluation uses the benchmark-specific normalization described in Section 4.1. Rewards are not clipped: insertion-heavy outputs can receive negative values. For an empty reference, e is the number of normalized hypothesis words, so an empty output receives reward 1. Speaker-attributed examples use WER on the normalized serialized transcript, including speaker-tag tokens, rather than the permutation-invariant cpWER used for evaluation.

Online sampling and advantages. At each update, we initially offer 40 recording–instruction pairs and sample $G = 64$ fresh responses per pair from the current policy at temperature 0.8. Groups with identical rewards are removed, and the first 20 remaining groups in data order are retained, yielding 1,280 responses per update. If necessary, up to two additional candidate batches are sampled to fill the update.

For response i in a retained group, we subtract the leave-one-out mean reward and divide by the

sample standard deviation of the *full* group:

$$A_i = \frac{r_i - \bar{r}_{-i}}{s_r + 10^{-6}}, \quad \bar{r}_{-i} = \frac{1}{G-1} \sum_{j \neq i} r_j, \quad s_r^2 = \frac{1}{G-1} \sum_{j=1}^G (r_j - \bar{r})^2, \quad (1)$$

where $\bar{r}$ is the mean of all G rewards. The denominator is shared by all responses in the group, rather than recomputed after leaving each response out. Constant-reward groups have zero advantage. GSPO uses the length-normalized sequence importance ratio

$$\rho_i(\theta) = \exp \left[\frac{1}{T_i} \sum_{t=1}^{T_i} \log \frac{\pi_\theta(y_{i,t} \mid x, y_{i,<t})}{\pi_{\text{old}}(y_{i,t} \mid x, y_{i,<t})} \right], \quad (2)$$

where x is the instruction and audio, T_i counts valid response tokens, and π_{old} is the policy before the update. We optimize the mean clipped surrogate $\min(\rho_i A_i, \text{clip}(\rho_i, 0.8, 1.2) A_i)$ over retained responses. There is no KL penalty, advantage clipping, or auxiliary SFT loss.

Training setup. We run 400 optimizer updates on four eight-GPU nodes. The policy uses expert parallelism of degree 8, activation checkpointing, and a per-GPU microbatch of one response; colocated vLLM rollout workers use tensor and expert parallelism of degree 8. We use AdamW with learning rate 10^{-6} , $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, weight decay 0.01, and gradient-norm clipping at 1.0. The learning rate increases linearly from 10^{-7} to 10^{-6} over five warmup updates and then remains constant. Sampling uses top- $p = 1$, no top- k restriction, a total context limit of 62,464 tokens, and an output ceiling of 17,278 tokens. We disable encoder dropout, SpecAugment, and MTP during training and rollout generation.

We update 269 tensors containing 1B parameters: the LLM’s Mamba and attention blocks and their associated normalization parameters, the last two ASR encoder layers, and the audio-to-LLM projection. These trainable parameters are stored in FP32. The MoE blocks and routers, token embeddings, language-model output head, MTP module, and remaining encoder parameters are frozen; frozen weights retain their mixed BF16/FP32 storage.

Effect of RL. Table 5 compares the SFT checkpoint with the final model. Although the objective is limited to reference-based transcription rewards, reinforcement learning lowers the English and multilingual ASR error rates, improves code-entity accuracy and number transcription, and reduces insertions on HALAS. Contextual and style-controlled transcription also improve. Speech translation and full-session multi-speaker transcription remain approximately unchanged. With reasoning enabled for both checkpoints, VoiceBench and Spoken-MQA scores improve, while MMAR accuracy decreases.

3.4. CTC Timestamp Adapter

We train the CTC adapter (Section 2.5) on all single-speaker and multi-speaker ASR training data with the CTC objective. The pretrained speech encoder is frozen and kept in evaluation mode during adapter training, and the Transformer layers of the adapter use a dropout of 0.1. We optimize the adapter parameters using AdamW with a learning rate of 3×10^{-4} , weight decay of 0.01, $(\beta_1, \beta_2) = (0.9, 0.95)$, and an inverse-square-root learning-rate schedule with 2,500 warmup steps.

Table 5 | Effect of reinforcement learning: the SFT checkpoint after 28,500 steps and the final model. All scores are percentages except BLEU. The English aggregate averages eight Open ASR subsets, including SPGISpeech, and Monsoon English. The multilingual Open ASR Leaderboard uses the general transcription instruction on all test sets, differing from the protocol of Section 4.5. CoVoST 2 ASR uses the corpus-level mixed WER/CER aggregate. The final three rows use matched reasoning-enabled decoding for both checkpoints. Final-checkpoint scores differ from those reported later because the evaluation code and prompts were subsequently refined.

Benchmark	Metric	SFT	SFT + RL
Open ASR Leaderboard (8 sets)	WER ↓	4.34	4.00
Multilingual Open ASR Leaderboard	WER ↓	5.05	4.78
CoVoST 2 ASR	WER/CER ↓	6.63	5.65
Numb3rs	WER ↓	7.91	6.89
VoiceCodeBench	CTEM ↑	86.37	88.60
Preference-ASR, instructed	WER ↓	6.32	6.12
Contextual Earnings-22, global	WER ↓	10.86	10.68
HALAS	segments with insertions ↓	46.04	41.21
Multi-speaker, full sessions	cpWER ↓	13.19	13.10
CoVoST 2 speech translation	BLEU ↑	31.78	31.74
VoiceBench (reasoning on)	score ↑	80.75	82.40
Spoken-MQA (reasoning on)	accuracy ↑	84.18	86.91
MMAR (reasoning on)	accuracy ↑	67.01	59.18

3.5. DFlash2 Draft Model

We use a six-layer DFlash2 draft Transformer (Chen et al., 2026a; Inco AI, 2026) for speculative decoding (Section 4.1), conditioned on output hidden states from backbone layers 1, 5, 19, 29, 41, and 51 (zero-based layer indexing). These states are concatenated, linearly projected to the draft hidden size, and normalized. It predicts token blocks in parallel, using local dynamic convolutions and a pairwise selector over the top 16 token candidates at each position. Token embeddings and the language-model output head are shared with the frozen backbone.

We train the draft model for 4,000 steps on 32 GPUs, starting from a draft model trained for an earlier checkpoint. Each 16-token training block contains one known anchor token followed by 15 positions to predict. At inference, K counts only proposed tokens, so this corresponds to $K = 15$.

On examples from the supervised fine-tuning mixture, we jointly optimize draft cross-entropy over the full vocabulary and selector cross-entropy over the top 16 draft candidates at each position, with equal loss weights and the same positional weights $w_i = \exp\left(-\frac{i-1}{\gamma}\right)$, where $\gamma = 4$ and $i = 1, \dots, 15$ indexes the predicted positions after the anchor. Following DFlash (Chen et al., 2026a), this decay emphasizes the early positions that determine the accepted prefix. Both losses use the final Nemotron 3.5 Transcribe backbone’s argmax tokens as targets, computed from teacher-forced reference prefixes to avoid autoregressive label generation. During training, the selector additionally conditions on the preceding argmax target token, using the known anchor at the first prediction position. Positions whose target is absent from the top-16 candidates are excluded from the selector loss.

4. Results

All results in this section are obtained with the released checkpoint, i.e., after reinforcement learning (Section 3.3). Figure 7 gives an overview of the evaluated benchmarks.

4.1. Evaluation Protocol

Inference. We evaluate the model with the serving setup we recommend for deployment: vLLM (Kwon et al., 2023) with BF16 weights, on which a single H100 GPU is sufficient, and speculative decoding with the MTP module (Section 2.4) as the draft model. At every step, the MTP module proposes draft tokens, which the backbone verifies in a single forward pass; with greedy decoding, speculative decoding changes only the speed, not the output. By default, the module proposes two draft tokens, which matches the number of MTP depths in training, and we use this setting for all accuracy results; since the layer is shared between depths, it can also propose more, e.g., eight, which increases the throughput (Section 4.13). As an alternative to MTP, we provide a separate draft model, Nemotron-3.5-Transcribe-30B-A4B-DFlash2, based on DFlash2 (Chen et al., 2026a; Inco AI, 2026): a six-layer model with approximately 570M parameters that proposes up to 15 draft tokens at once (Section 3.5). For the Mamba layers, we cache the recurrent state at every block (vLLM’s all Mamba cache mode), which is required for exact speculative decoding with prefix caching.¹

At inference time, the Parallel Expert Encoder processes audio in 40-second central chunks, with 4 seconds of additional context on each side where available. These settings correspond to 500 central frames and 50 context frames per side at an encoder frame interval of 80 ms. Both branches receive the same context-extended window. The ASR branch discards outputs corresponding to the additional context, while the streaming diarization branch trims context internally and maintains its speaker-cache and FIFO state across chunks. The retained outputs are concatenated and fused before being passed to the LLM.

All requests use a context of up to 65,536 tokens. Each benchmark uses a fixed instruction, listed in Appendix A, except for benchmarks that define their own per-example instructions (e.g., contextual ASR and question answering), which we keep unchanged.

Post-processing. In rare cases, the model fails to terminate its generation and repeats the same character or phrase until it reaches the token limit. Such loops are easy to detect, and we remove them with a simple post-processing step adapted from the Qwen3-ASR toolkit (Shi et al., 2026; Qwen Team, 2026b), which collapses long exact repetitions to a single copy and which we also recommend for deployment (Appendix E). Unless stated otherwise, all transcription, translation, and language identification results in this report are computed after this post-processing; Appendix E lists the results with and without it. We do not apply it to the answers generated with reasoning (Sections 4.14 to 4.16), where it mostly matches legitimate formatting, such as the rules of Markdown tables, rather than generation loops.

Reasoning. Nemotron 3.5 Transcribe inherits the reasoning mode of its LLM, and we choose it per task family. For transcription, translation, and language identification, we disable reasoning and decode greedily, which is how we recommend using the model for these tasks. For audio understanding, voice instruction following, and text-only reasoning, we enable reasoning and use the decoding settings recommended for Nemotron 3 Nano Omni (NVIDIA, 2026b): temperature 0.6, top- p 0.95, and up to 20,480 generated tokens, of which at most 16,384 are spent on reasoning; only the final answer is scored.

Metrics. We report word error rate (WER) after the text normalization of the Open ASR Leaderboard (Srivastav et al., 2026), and character error rate (CER) for Chinese, Japanese, Korean,

¹With the `align` cache mode and prefix caching enabled, the outputs degrade silently, by about one WER point.

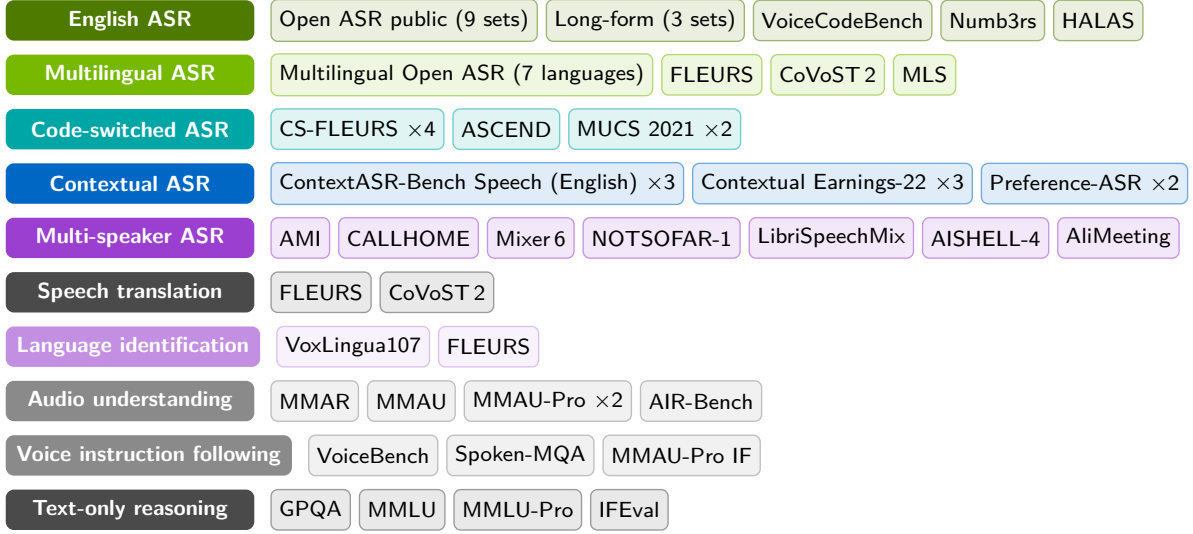


Figure 7 | The benchmarks used in this report, grouped by task. Entries marked with $\times n$ comprise n test sets or evaluation settings.

Thai, and Vietnamese. For ContextASR-Bench Speech (English), we follow its benchmark-specific text normalization and named-entity matching procedure. Hindi is scored with the orthographically informed WER (OIWER) (Voice Arena, 2026), which does not count differences between equivalent spellings as errors. Code-switched speech is scored with the mixed error rate (MER), which counts words for languages written with spaces and characters for the others. Speaker-attributed transcription is scored with the concatenated minimum-permutation WER (cpWER), which penalizes both recognition and speaker attribution errors. Speech translation is scored with BLEU, and language identification and multiple-choice benchmarks with accuracy. Three benchmarks use an LLM judge, following their official protocols: the open-ended answers of VoiceBench (Chen et al., 2026b) (GPT-4o-mini) and of MMAU-Pro (Kumar et al., 2026) (Qwen2.5-7B-Instruct), and the entity and task success metrics of VoiceCodeBench (Baumgartner et al., 2026).

Baselines. We compare with results published by the benchmark authors or by the model developers, and include them only when they were obtained on the same test sets and with the same metric as ours; “–” marks combinations for which no comparable result is published. Since published results are obtained with different prompts and inference settings, small differences should be interpreted with care.

4.2. English ASR

We evaluate English ASR on the public section of the Open ASR Leaderboard (Srivastav et al., 2026; Hugging Face, 2026b), which consists of nine test sets: read speech (LibriSpeech test-clean and test-other (Panayotov et al., 2015)), parliamentary speech (VoxPopuli (Wang et al., 2021a)), podcasts and YouTube (GigaSpeech (Chen et al., 2021)), financial earnings calls (SPGISpeech (O’Neill et al., 2021) and Earnings-22 (Del Rio et al., 2022)), meetings (AMI (Carletta et al., 2006)), Indian-accented English (Voice Arena Monsoon (Voice Arena, 2026)), and noisy, reverberant, and otherwise degraded speech (URGENT 2024 (Zhang et al., 2024)). We use the current, cleaned versions of the AMI, Earnings-22, GigaSpeech, and VoxPopuli test sets, and report the unweighted mean of the nine WERs, as the leaderboard does for its public section.²

²The default ranking of the leaderboard additionally includes private test sets, which are not available to us.

Table 6 | English ASR on the public section of the Open ASR Leaderboard: WER (%), lower is better; bold: best open-weights model. E22: Earnings-22, Giga: GigaSpeech, LS-c/LS-o: LibriSpeech test-clean/test-other, SPGI: SPGISpeech, Monsoon: Voice Arena Monsoon (English), URGENT: URGENT 2024, VoxP: VoxPopuli. Baselines from the leaderboard results of 21 October 2026 ([Hugging Face, 2026b](#)).

Model	AMI	E22	Giga	LS-c	LS-o	SPGI	Monsoon	URGENT	VoxP	Mean
Nemotron 3.5 Transcribe	7.81	5.29	7.18	1.09	2.60	1.91	3.65	11.00	1.84	4.71
<i>Open-weights models</i>										
Qwen3-ASR-1.7B (Shi et al., 2026)	8.31	5.84	7.22	1.26	2.94	2.57	3.50	9.93	2.85	4.94
Granite-Speech-5.0-2B (Saon et al., 2025)	6.96	6.22	8.14	0.99	2.21	3.36	3.94	10.22	3.52	5.06
Granite-Speech-4.1-2B (Saon et al., 2025)	7.06	6.69	8.18	1.03	2.19	3.47	4.12	11.25	4.18	5.35
Canary-Qwen-2.5B (NVIDIA, 2025a)	7.91	6.41	7.80	1.23	2.63	1.70	3.60	13.02	4.14	5.38
Parakeet-TDT-0.6B-v3 (Sekoyan et al., 2025)	9.42	5.85	7.99	1.52	3.13	3.63	4.14	11.52	3.05	5.58
Voxtral-Small-24B (Liu et al., 2025)	13.19	7.07	8.33	1.23	2.80	1.84	3.49	12.35	2.00	5.81
Whisper-large-v3 (Radford et al., 2023)	13.63	7.99	8.35	1.56	3.52	2.71	3.95	13.73	4.53	6.66
<i>Commercial systems</i>										
OpenAI gpt-transcribe	8.95	6.61	7.76	1.31	3.39	2.23	3.63	12.86	2.65	5.49
ElevenLabs Scribe v2	8.19	4.80	7.63	1.06	2.33	3.01	3.32	9.20	1.41	4.55
Modulate	6.31	6.00	6.91	0.87	1.77	2.06	3.26	10.04	3.55	4.53
Azure Speech (07-2026)	7.05	5.20	7.48	1.23	2.71	2.17	3.00	9.90	1.65	4.49
Zoom Scribe v2 Pro	6.08	4.12	7.75	1.11	2.24	1.36	3.00	7.99	3.08	4.08
OtterAI speech-v3.1	6.42	4.55	7.13	0.90	1.89	1.56	3.02	8.07	1.42	3.88

With a mean WER of 4.71%, Nemotron 3.5 Transcribe is the most accurate open-weights model on the public section of the leaderboard, 0.23 points ahead of the next one, and is outperformed only by seven commercial systems, led by OtterAI speech-v3.1 with 3.88% (Table 6). Compared with the other open-weights models in the table, it is particularly strong on Earnings-22 and VoxPopuli; on meetings (AMI), the two Granite-Speech models are more accurate, and on degraded speech (URGENT 2024), Qwen3-ASR-1.7B and Granite-Speech-5.0-2B.

4.3. Long-Form ASR

The long-form track of the Open ASR Leaderboard contains full recordings of earnings calls (Earnings-21 and Earnings-22) and of conversational interviews (CORAAL), which last from several minutes to more than an hour. Nemotron 3.5 Transcribe transcribes each recording without voice activity detection; recordings longer than 55 minutes are split into 55-minute chunks with 5 seconds of overlap. Its mean WER of 10.98% is the lowest among the open-weights models on this track, and it remains behind the best commercial systems (Table 7). On this track, generation loops affect 7 of the 451 recordings; without the loop guardrail, the mean WER would be 12.87% (Appendix E).

Table 7 | Long-form ASR: WER (%) on the long-form track of the Open ASR Leaderboard (Hugging Face, 2026a), lower is better; bold: best open-weights model; CORAAL is averaged over its eight locations.

Model	Earnings-21	Earnings-22	CORAAL	Mean
Nemotron 3.5 Transcribe	7.57	11.99	13.39	10.98
<i>Open-weights models</i>				
Parakeet-TDT-0.6B-v3	9.96	14.58	15.57	13.37
Whisper-large-v3	9.71	13.16	18.89	13.92
Canary-Qwen-2.5B	9.60	13.59	19.05	14.08
<i>Commercial systems</i>				
AssemblyAI Universal-3 Pro	7.62	10.59	12.83	10.35
ElevenLabs Scribe v2	6.48	9.99	10.67	9.05

4.4. Entities, Numbers, and Hallucinations

VoiceCodeBench (Baumgartner et al., 2026) contains English speech dense with code-like entities, such as phone numbers, e-mail addresses, URLs, file paths, and command-line flags. Besides WER, it measures the exact-match rate of these entities (CTEM) and the share of recordings in which every entity is correct (TSR). Nemotron 3.5 Transcribe recognizes code-like entities on par with dedicated ASR systems: its entity exact-match rate is higher than that of Whisper-large-v3 and Parakeet-TDT-0.6B-v3, and about 3 points behind the best commercial systems (Table 8). Numb3rs (NVIDIA, 2025g) is dominated by numeric expressions, such as addresses, amounts of money, and times; we score it against its spoken-form references. HALAS (Barański et al., 2026) consists of Earnings-22 segments with human annotations of hallucinations; we report WER against the corrected references and the percentage of segments in which the model inserted content. No comparable published results exist for the last two benchmarks.³ In a simulated voice agent, accurate entity recognition carries over to the agent’s tool calls: compared with Parakeet-CTC-1.1B as the speech recognizer, Nemotron 3.5 Transcribe reduces the tool-call failures caused by misrecognized surnames from 73 to 2 (Appendix G).

4.5. Multilingual ASR

The multilingual section of the Open ASR Leaderboard (Srivastav et al., 2026) covers German, French, Italian, Spanish, Portuguese, Dutch, and Hindi. Each language is scored as the mean over FLEURS (Conneau et al., 2022), Common Voice (Ardila et al., 2020), and Multilingual LibriSpeech (Pratap et al., 2020), where available, and Hindi on the Voice Arena Monsoon test set (Voice Arena, 2026) with OIWER.⁴ All datasets are scored with the references and text normalization of the leaderboard, including its treatment of compound words, so the FLEURS results differ slightly from our standalone FLEURS evaluation (Appendix C). Nemotron 3.5 Transcribe reaches a mean WER of 4.62% over the seven languages, ahead of the open-weights models in Table 9 and behind the best commercial systems.

We further evaluate on FLEURS for 37 languages and on CoVoST 2 (Wang et al., 2021b) for 17 languages, with macro-averaged error rates of 9.29% and 12.19%, respectively; the per-language results are listed in Appendix C. No published results are available on these language selections. On

³The HALAS authors report human-annotated hallucination rates on all annotated audio, whereas we report an automatic insertion-based measure on the test split.

⁴The leaderboard additionally uses a private Hindi test set, which is not available to us; for all models in Table 9, Hindi is computed on the public test set only.

Table 8 | Entity, number, and hallucination benchmarks (%). Lower is better, except for CTEM and TSR. VoiceCodeBench baselines (batch mode) from [Baumgartner et al. \(2026\)](#); all VoiceCodeBench WERs are format-invariant.

<i>VoiceCodeBench</i> Model	WER	CTEM $\uparrow$	TSR $\uparrow$
Nemotron 3.5 Transcribe	8.38	88.53	59.33
<i>Open-weights models</i>			
Whisper-large-v3	5.8	87.6	54.3
Parakeet-TDT-0.6B-v3	9.3	78.6	37.3
<i>Commercial systems</i>			
OpenAI gpt-transcribe	6.1	89.3	61.0
Deepgram Nova-3	8.2	90.9	68.7
ElevenLabs Scribe v2	6.9	91.6	67.7
<i>Other benchmarks (Nemotron 3.5 Transcribe)</i>			
Numb3rs, WER	3.55		
HALAS, WER	31.50		
HALAS, segments with insertions	40.00		

Table 9 | Multilingual section of the Open ASR Leaderboard: WER (%; OIWER for Hindi, public test set only), lower is better; bold: best open-weights model. Baselines from the leaderboard results of 2 October 2026 ([Hugging Face, 2026a](#)).

Model	de	fr	it	es	pt	nl	hi	Mean
Nemotron 3.5 Transcribe	2.81	3.92	3.61	2.82	4.06	5.96	9.15	4.62
<i>Open-weights models</i>								
Qwen3-ASR-1.7B	3.97	5.68	5.44	3.80	6.31	7.95	12.25	6.49
Voxtral-Small-24B	2.90	4.11	3.75	2.91	4.36	4.44	23.20	6.52
Whisper-large-v3	4.00	6.24	4.45	3.35	4.94	4.42	28.17	7.94
<i>Commercial systems</i>								
ElevenLabs Scribe v2	2.24	3.29	2.49	2.32	2.90	3.26	8.45	3.56
Azure Speech (07-2026)	1.92	3.14	2.42	2.30	2.86	3.01	10.40	3.72

Multilingual LibriSpeech ([Pratap et al., 2020](#)), Nemotron 3.5 Transcribe reaches WERs of 2.98% (Spanish), 3.14% (French), 4.99% (Portuguese), 6.21% (Italian), and 8.45% (Dutch) on the test subsets used by the multilingual Open ASR Leaderboard, and of 3.87% (English), 4.04% (German), and 4.89% (Polish) on the full test sets of the remaining three MLS languages.

4.6. Code-Switched ASR

CS-FLEURS ([Yan et al., 2025](#)) contains sentences that mix English with another language. Its four test sets consist of speech read by humans (*read*, 14 language pairs) and speech synthesized with MMS-TTS ([Pratap et al., 2024](#)) (*MMS*, 45 pairs) and XTTS-v2 ([Casanova et al., 2024](#)) (*XTTS 1* and *XTTS 2*, 16 and 60 pairs). We also evaluate on two conversational code-switching corpora: ASCEND ([Lovenia et al., 2022](#)) (Mandarin–English) and the MUCS 2021 challenge ([Diwan et al., 2021](#)) (Hindi–English and Bengali–English). On CS-FLEURS, Nemotron 3.5 Transcribe is behind Whisper-large-v3 on the read and MMS test sets and slightly ahead on the XTTS test sets (Table 10); code-switching across many language pairs remains a weakness of the model.

Table 10 | Code-switched ASR (%), lower is better. CS-FLEURS: CER and MER, Whisper-large-v3 CER from [Yan et al. \(2025\)](#); ASCEND: MER; MUCS 2021: WER.

Benchmark	Nemotron 3.5 Transcribe		Whisper-large-v3
	CER	MER	CER
CS-FLEURS read (14 pairs)	25.83	31.06	19.83
CS-FLEURS MMS (45 pairs)	44.90	66.57	40.07
CS-FLEURS XTTS 1 (16 pairs)	18.00	28.10	18.46
CS-FLEURS XTTS 2 (60 pairs)	33.87	41.68	36.06
ASCEND (Mandarin-English)		18.74	
MUCS 2021 Hindi-English		19.28	
MUCS 2021 Bengali-English		35.46	

4.7. Contextual ASR

Contextual ASR benchmarks measure how well the model uses information given in the prompt. We evaluate every benchmark with and without the context, so that the difference isolates the benefit of the context; the instructions are listed in Appendix A. Table 11 summarizes the results.

ContextASR-Bench Speech (English). The ContextASR-Bench Speech (English) subset ([Wang et al., 2025a](#)) provides a domain label (*coarse*) or a domain label and a list of named entities (*fine*) for each utterance; we report WER, named-entity WER (NE-WER), and named-entity false-negative rate (NE-FNR). With an entity list, Nemotron 3.5 Transcribe reduces WER from 3.33% to 1.71%, NE-WER from 12.21% to 2.20%, and NE-FNR from 14.99% to 1.32%, relative reductions of 49%, 82%, and 91%, respectively. A domain label alone has little effect for any of the three models evaluated by us. Against the systems on the published ContextASR-Bench leaderboard ([Wang et al., 2025a](#)), Nemotron 3.5 Transcribe achieves the lowest WER, NE-WER, and NE-FNR in all three settings. Compared with the more recent Qwen3 models ([Xu et al., 2025b](#); [Shi et al., 2026](#)), which were released after the benchmark paper, Nemotron 3.5 Transcribe achieves the largest relative reduction in NE-WER among the three models and ranks second on both entity-centric metrics in the fine setting, while the Qwen3 models have lower absolute WER in all three settings.

Contextual Earnings-22. Contextual Earnings-22 ([Munyampirwa et al., 2026](#)) consists of 15-second clips of earnings calls with keyword lists that either contain only the keywords spoken in the clip (*local*) or the keyword inventory of the whole call, most of which does not occur in the clip (*global*). The global setting mimics a user who supplies a vocabulary for a whole session. For comparability with the published results, we compute WER and keyword precision, recall, and F1 with the text normalization and keyword matching of OpenBench ([Argmax, 2025](#)). The keywords raise the recall from 58% to 96% (local) and 90% (global), while the precision stays at 98%, i.e., the model rarely inserts listed terms that were not spoken. With keywords, the WER of Nemotron 3.5 Transcribe is close to that of the best systems, and its keyword F1 in the local setting matches the best reported.

Preference-ASR. Preference-ASR ([Koluguri et al., 2026](#)) measures with a preference-aware WER whether a transcript follows a requested style, such as casing, number formatting, or the removal of disfluencies. In the *instructed* setting, each example comes with its own style instruction; in the

Table 11 | Contextual ASR results (%). ContextASR-Bench Speech (English): WER / NE-WER / NE-FNR, with selected baselines from Wang et al. (2025a). Contextual Earnings-22: WER / keyword F1, scored with OpenBench, baselines from Argmax (2025)*. Preference-ASR: preference-aware WER, baselines from Koluguri et al. (2026). Lower is better, except for F1.

<i>ContextASR-Bench Speech (English), WER / NE-WER / NE-FNR</i>			
Model	no context	coarse	fine
Nemotron 3.5 Transcribe	3.33 / 12.21 / 14.99	3.34 / 12.26 / 15.05	1.71 / 2.20 / 1.32
<i>Open-weights models</i>			
Qwen3-Omni-30B-A3B-Instruct (Xu et al., 2025b) [†]	2.27 / 7.83 / 9.08	2.17 / 8.11 / 9.32	1.13 / 1.55 / 0.53
Qwen3-ASR-1.7B (Shi et al., 2026) [†]	2.92 / 11.75 / 14.81	2.91 / 11.73 / 14.86	1.53 / 2.76 / 1.80
Kimi-Audio (Kimi Team et al., 2025)	4.09 / 14.33 / 19.53	4.47 / 13.88 / 18.60	2.90 / 6.68 / 8.01
Qwen2.5-Omni-7B (Xu et al., 2025a)	5.60 / 16.07 / 21.33	5.56 / 15.93 / 21.13	3.96 / 7.38 / 8.72
Qwen2.5-Omni-3B (Xu et al., 2025a)	6.19 / 20.52 / 28.26	6.30 / 20.62 / 28.33	3.99 / 7.80 / 9.69
Qwen2-Audio	13.56 / 38.95 / 52.29	13.41 / 38.34 / 51.55	11.49 / 27.27 / 35.08
Whisper-large-v3	9.36 / 29.56 / 39.89	–	–
Ming-Flash-Omni [‡]	–	–	2.91 / 2.53 / 2.13
<i>Commercial systems</i>			
Gemini 2.5 Pro [‡]	–	–	6.65 / 12.82 / 8.86
<i>Contextual Earnings-22, WER / keyword F1</i>			
Model	no keywords	local	global
Nemotron 3.5 Transcribe	12.54 / 73.0	9.58 / 97.1	10.28 / 93.5
<i>Open-weights models</i>			
Qwen3-ASR-1.7B	11.86 / 69	9.19 / 92	10.18 / 87
<i>Commercial systems</i>			
AssemblyAI	12.58 / –	11.67 / –	11.80 / –
Deepgram Nova-3	15.34 / –	13.28 / –	13.85 / –
ElevenLabs	10.53 / 84	9.13 / 97	9.08 / 95
<i>Preference-ASR, preference-aware WER</i>			
Model	default	instructed	
Nemotron 3.5 Transcribe	7.56	6.16	
Canary-Qwen-2.5B	8.16	8.36	
Qwen3-Omni-30B-A3B	8.30	10.34	
Phi-4-multimodal	15.11	12.64	

[†]The original ContextASR-Bench evaluation does not include the Qwen3 models. We obtained these scores internally using the released Hugging Face checkpoints Qwen3-Omni-30B-A3B-Instruct and Qwen3-ASR-1.7B. The 1.13% Qwen3-Omni fine-context WER is also reported for Speech-English in Inclusion AI (2025). [‡]As evaluated in Inclusion AI (2025), which does not state the context setting; its Kimi-Audio results match the fine setting of Wang et al. (2025a). *The published baselines were obtained on an earlier revision of the dataset, which differs from ours by two clips and eleven corrected references.

default setting, the model receives a generic transcription instruction.⁵ Nemotron 3.5 Transcribe outperforms all published results in both settings, and it improves further when given the style instruction (from 7.56% to 6.16%), unlike most of the other models.

⁵Our generic instruction (“Transcribe this recording.”) differs from the one used in the original paper.

4.8. Multi-Speaker Transcription

4.8.1. Speaker-attributed ASR

We evaluate speaker-attributed transcription in English on recordings of AMI (Carletta et al., 2006) (individual headset microphone mix and single distant microphone), 109 2-speaker sessions from CALLHOME American English (CHAES) (Canavan et al., 1997), referred to as *CH109*, and Mixer 6 (Brandschain et al., 2010), on segments of 10–20 s, 60–90 s, and 120–180 s, as well as on full sessions and full meetings of NOTSOFAR-1 (Vinnikov et al., 2024) (individual headset microphone mix and single-channel distant microphone). Table 13 shows that the cpWER hardly depends on the duration of the recording: from 60–90 s segments to full sessions, it changes by less than 1.5 points on AMI and Mixer 6, and it decreases by 2.4 points on CHAES. Only the shortest Mixer 6 segments are notably harder.

Table 14 compares full sessions with four open-weights systems, which we evaluate with the same data and scoring: two speech LLMs for long-form multi-speaker transcription, VibeVoice-ASR (Peng et al., 2026) and MOSS-Transcribe-Diarize (MOSI.AI et al., 2026), and two diarization-conditioned ASR models, the multi-talker Parakeet (NVIDIA, 2025d) with Nemotron 3 Diarization and DiCoW v3.3 (Polok et al., 2026) with DiariZen (Han et al., 2026). Nemotron 3.5 Transcribe achieves the lowest cpWER averaged over the six conditions (15.80% vs. 16.83% for DiCoW) and the lowest cpWER on both AMI conditions and on NOTSOFAR-1 with headset microphones. DiCoW is more accurate on NOTSOFAR-1 with a distant microphone (24.80% vs. 25.91%), and both MOSS-Transcribe-Diarize and DiCoW are more accurate on CHAES-CH109 (10.72% and 10.80% vs. 11.50%) and Mixer 6 (13.25% and 14.74% vs. 14.80%). We also report LibriSpeechMix (Kanda et al., 2020), a synthetic dataset of one to three overlapping read utterances, as a diagnostic rather than as a ranking of real-world performance. Nemotron 3.5 Transcribe achieves the lowest published cpWER for single speakers; with two overlapping speakers, it is behind a dedicated multi-talker model with self-speaker adaptation (Wang et al., 2025b) and DiCoW, and with three, it is behind the self-speaker adaptation model and a speaker-attributed Transformer model (Kanda et al., 2021).

An important caveat is that Nemotron 3.5 Transcribe was not trained on LibriSpeechMix, whereas all models in the research-systems group were specifically fine-tuned on its training set. These results therefore compare generalization to an unseen mixture dataset with performance after dataset-specific fine-tuning.

The LibriSpeechMix results also highlight a limitation of the evaluated open-weights multi-speaker ASR systems: several struggle on the single-speaker condition, which is identical to LibriSpeech test-clean. Given that test-clean is a widely used benchmark of clean, read speech, poor performance on this condition raises concerns about their suitability for deployments that must handle both single-speaker and multi-speaker audio. Strong performance on overlapping speech alone is insufficient to establish readiness for such deployments; reliable single-speaker transcription is also essential.

We also report speaker-attributed transcription of Mandarin meetings on the test sets of AISHELL-4 (Fu et al., 2021) and AliMeeting (Yu et al., 2022), 20 full sessions each, recorded with a distant microphone array of which we use the first channel; we score them with the concatenated minimum-permutation character error rate (cpCER). These results should be read with the training data in mind: Nemotron 3.5 Transcribe saw almost no Mandarin multi-speaker data during training, only the training sets of these two corpora (about 210 hours of meetings), which account for 1.7% of the speaker-attributed transcription examples and less than 0.01% of all training examples. With cpCERs of 32.91% on AISHELL-4 and 34.95% on AliMeeting (Table 13), Mandarin far-field meetings remain a clear weakness of the model, and we expect a large improvement from fine-tuning with more Mandarin multi-speaker training data. Generation loops are also more frequent on these sessions:

Table 12 | Near-field transcription error breakdown by prompt type and dataset (%). NFO denotes Near Field Only. NSF-SC-Eval is the single-channel evaluation set of NOTSOFAR-1 (Vinnikov et al., 2024); LS-Test-clean is LibriSpeech test-clean (Panayotov et al., 2015).

Prompt	Dataset	Insertions	Deletions	Substitutions	WER
Multispeaker ASR	LS-Test-clean	0.19	0.22	1.15	1.57
Near Field Only (NFO)	NSF-SC-Eval + LS-Test-clean	1.07	1.60	2.11	4.78
Single-speaker ASR	NSF-SC-Eval + LS-Test-clean	20.38	0.69	6.25	27.32
Multi-speaker ASR	NSF-SC-Eval + LS-Test-clean	219.96	0.42	2.99	223.37

Table 13 | Multi-speaker transcription by segment duration: cpWER (%; cpCER for the Mandarin meetings), lower is better. The model saw almost no Mandarin multi-speaker data in training.

Corpus	10–20 s	60–90 s	120–180 s	full
AMI, headset mix	10.80	9.77	10.05	10.29
AMI, distant microphone	17.56	15.86	16.23	17.34
CHAES-CH109	13.76	13.86	12.81	11.50
Mixer 6	26.64	15.80	15.27	14.80
NOTSOFAR-1, headset mix	–	–	–	14.94
NOTSOFAR-1, distant microphone	–	–	–	25.91
<i>Mandarin, distant microphone (cpCER)</i>				
AISHELL-4	–	–	–	32.91
AliMeeting	–	–	–	34.95

without the loop guardrail, the cpCERs would be 45.55% and 55.37% (Appendix E).

4.8.2. Near-field transcription (experimental)

The training mixture also contains examples in which only the speaker nearest to the microphone is transcribed, selected with a dedicated instruction (Appendix A). To test this mode, we combine LibriSpeech test-clean utterances, which are close-talk recordings, with far-field meeting audio from the single-channel NOTSOFAR-1 evaluation set, and score the output against the LibriSpeech references. Table 12 reports the error breakdown for each instruction, alongside a baseline on LibriSpeech test-clean alone. With the near-field instruction, the WER is 4.78%. The single-speaker transcription instruction yields 27.32%, mostly insertions (20.38%) from the background speech, and the multi-speaker instruction, which transcribes every speaker, 223.37%. These two results confirm that the far-field background speech is not too weak to be ignored: under the standard instructions, the model readily picks it up and transcribes it. The low error rate under the near-field instruction therefore cannot be explained by the background being inaudible; rather, the instruction leads the model to transcribe the speaker who is nearer to the microphone, in a relative sense, and to suppress the more distant background speakers. We consider this mode experimental: it has not yet been evaluated on recordings with natural near-field and far-field speakers.

4.9. Multi-Speaker Word-Level Timestamping

We evaluate end-to-end word timestamping on the individual headset microphone mix of AMI (Carletta et al., 2006), CH109 from CALLHOME American English (CHAES) (Canavan et al., 1997), LibriSpeechMix (LSM) (Kanda et al., 2020) with one, two, and three speakers, and Multilingual Conversational Speech (MCS), an internal, non-public dataset covering 15 languages. We use the

Table 14 | Comparison of multi-speaker transcription with other systems: cpWER (%), lower is better. Top: full sessions, all systems evaluated with the same data and scoring; IHM: individual headset microphone mix, SDM: single distant microphone, SC: single channel. Bottom: LibriSpeechMix test-clean, published results.

Model	AMI		CH109	Mixer 6	NOTSOFAR-1		Macro Avg.
	IHM	SDM			IHM	SC	
Nemotron 3.5 Transcribe	10.29	17.34	11.50	14.80	14.94	25.91	15.80
<i>Open-weights models</i>							
DiCoW v3.3 (Polok et al., 2026)							
+ DiariZen-v2 (Han et al., 2026)	14.48	18.24	10.80	14.74	17.91	24.80	16.83
Multi-talker Parakeet (NVIDIA, 2025d)	13.38	20.33	13.00	16.92	18.19	30.42	18.71
+ Nemotron 3 Diarization							
MOSS-Transcribe-Diarize (MOSLAI et al., 2026)	15.40	24.72	10.72	13.25	22.52	28.92	19.26
VibeVoice-ASR (Peng et al., 2026)	22.13	32.12	12.08	16.45	25.52	37.21	24.25
<i>LibriSpeechMix</i>							
Model				1 speaker	2 speakers	3 speakers	
Nemotron 3.5 Transcribe				1.58	4.14	8.65	
<i>Open-weights models</i>							
Multi-talker Parakeet (NVIDIA, 2025d)				2.77	7.20	12.94	
+ Nemotron 3 Diarization (NVIDIA, 2026c)							
MOSS-Transcribe-Diarize (MOSLAI et al., 2026)				2.39	11.21	18.87	
DiCoW v3.3 (Polok et al., 2026)				16.52	3.71	22.63	
+ DiariZen-v2 (Han et al., 2026)							
VibeVoice-ASR (Peng et al., 2026)				2.03	51.40	74.14	
<i>Research systems</i>							
SSA (Multi-talker Parakeet) (Wang et al., 2025b)				2.2	2.8	5.0	
SA-ASR Transformer (Kanda et al., 2021)				3.6	4.9	6.2	
Sortformer + Canary (Park et al., 2025)				2.26	4.61	9.05	
MT-LLM (Meng et al., 2025)				2.3	5.2	10.2	
SOT (Kanda et al., 2020)				4.6	11.2	24.0	

official word-timestamp annotations for AMI and Montreal Forced Aligner (MFA) (McAuliffe et al., 2017) to generate reference timestamps for CH109 and MCS, except for Japanese and Chinese. For these two MCS languages, poor MFA alignment quality motivated the use of Qwen3-ForcedAligner-0.6B (Qwen Team, 2026c). Their results therefore favor the Qwen alignment method and are not an independent or fully fair comparison of boundary accuracy. All other MCS languages use MFA-derived references. For LSM, we combine the publicly available MFA-generated LibriSpeech word alignments⁶ with the LibriSpeechMix mixture definitions,⁷ mapping source-utterance timestamps onto the mixture timeline. Automatically aligned references are approximate rather than manually verified boundaries.

All systems timestamp their own recognized text, rather than the reference transcript. We com-

⁶<https://github.com/CorentinJ/librispeech-alignments>

⁷<https://github.com/NaoyukiKanda/LibriSpeechMix>

pare Nemotron 3.5 Transcribe against Qwen3-ASR-1.7B (Shi et al., 2026) followed by Qwen3-ForcedAligner-0.6B (Qwen Team, 2026c), Parakeet-TDT-0.6B-v3 (Sekoyan et al., 2025), and Whisper-large-v3 (Radford et al., 2023) with FasterWhisper’s DTW-based timestamps (SYSTRAN, 2026). Our model uses its CTC-based word timestamp aligner with 40 ms acoustic frames and diarization-guided boundary refinement, without sub-frame interpolation.

We apply a common speaker-blind scoring protocol. After matching recognized and reference text, each matched word contributes two independently evaluated events: its start and end boundary. A boundary is correct when its absolute timing error is at most 200 ms. Precision and recall use denominators of twice the number of hypothesis and reference words, respectively, so unmatched words remain penalized. Event counts are pooled across recordings before computing F1, rather than averaging recording-level scores. Chinese and Japanese use character-level units; other languages use word-level units. This metric jointly measures recognition and temporal accuracy but does not evaluate speaker attribution.

Table 15 shows that Nemotron 3.5 Transcribe achieves the highest F1 on AMI-IHM-mix (80.03%), CH109 (81.11%), and LSM overall (88.66%). Its advantage is particularly pronounced for overlapping speech: it reaches 90.24% and 85.23% on LSM 2mix and 3mix, respectively, exceeding the strongest baseline by 7.76 and 11.35 percentage points. Qwen achieves the highest score on LSM 1mix. On MCS, Nemotron 3.5 Transcribe leads among the evaluated systems in 13 of 15 languages, while Qwen leads in Chinese and Japanese. The latter two comparisons are confounded by the use of Qwen-generated reference alignments. All comparisons are limited to the language coverage shown in the table.

4.10. Speech Translation

We evaluate speech translation on FLEURS (Conneau et al., 2022), between English and 36 other languages in both directions, and on CoVoST 2 (Wang et al., 2021b), for 16 languages into English and 10 out of English. We report BLEU for every translation direction, using the appropriate sacreBLEU tokenizers for Chinese, Japanese, and Korean targets and spBLEU (the FLORES-200 SentencePiece tokenizer) for Thai targets. Averaged over the languages, Nemotron 3.5 Transcribe reaches 29.20 BLEU into English and 26.54 BLEU out of English on FLEURS, and 33.67 and 26.34 BLEU on CoVoST 2, respectively; the per-direction results are listed in Appendix C. Published results cover different language selections and are therefore not directly comparable.

4.11. Spoken Language Identification

We evaluate language identification on FLEURS (36 languages) and on the subset of the VoxLingua107 (Valk and Alumäe, 2021) development set that covers 24 of the languages supported by Nemotron 3.5 Transcribe; the instruction lists the candidate languages (Appendix A). On FLEURS, Nemotron 3.5 Transcribe identifies the language with at least 97% accuracy for 33 of the 36 languages (98.64% on average); the exceptions are Latvian (80.49%), Slovenian (90.05%), and Hindi (94.02%). VoxLingua107, which is collected from YouTube videos, is harder: the average accuracy is 95.58%, with the lowest accuracy for Slovenian (55.56%), Latvian (81.05%), and Lithuanian (84.62%). The per-language results are listed in Appendix C. Published results cover all languages of these benchmarks and are therefore not comparable.

4.12. Accuracy of the Quantized Checkpoints

We compare the BF16, FP8, and NVFP4 checkpoints on the transcription benchmarks in a single evaluation on one GB300 GPU, with identical instructions, data, scoring, loop guardrail, and serving

Table 15 | Word-level timestamping F1 (%) at a 200 ms tolerance across conversational and multilingual speech datasets. Start and end boundaries are evaluated independently, with event counts pooled within each dataset. All systems use their own recognized text and a common speaker-blind scoring protocol. Bold indicates the best score per row; a dash denotes unsupported language coverage.

Dataset	Nemotron 3.5 Transcribe	Qwen3-ASR-1.7B (Shi et al., 2026) with Qwen3-Forced- Aligner-0.6B (Qwen Team, 2026c)	Parakeet-TDT- 0.6B-v3 (Sekoyan et al., 2025)	Whisper-large-v3 (Radford et al., 2023) with FasterWhisper- DTW (SYSTRAN, 2026)
AMI-IHM-mix	80.03	63.76	70.73	54.02
CHAES-CH109	81.11	71.14	77.68	64.45
LSM total	88.66	80.97	72.27	60.40
LSM 1mix	95.76	97.95	93.23	73.05
LSM 2mix	90.24	82.48	73.40	60.50
LSM 3mix	85.23	73.88	62.75	55.41
MCS ¹ English	88.17	86.57	81.95	63.87
MCS German	87.12	84.00	81.32	65.87
MCS Spanish	91.42	88.89	71.83	69.03
MCS French	86.74	80.53	48.12	56.67
MCS Italian	83.88	81.38	76.09	61.81
MCS Portuguese	85.09	82.60	72.51	64.09
MCS Russian	83.70	78.69	77.79	64.94
MCS Chinese	87.99	97.24	–	70.37
MCS Japanese	91.71	95.36	–	64.45
MCS Korean	83.78	79.44	–	49.15
MCS Hindi	82.20	–	–	47.71
MCS Arabic	82.48	–	–	61.85
MCS Thai	85.95	–	–	68.22
MCS Turkish	83.89	–	–	60.72
MCS Vietnamese	86.90	–	–	70.11

¹MCS (Multilingual Conversational Speech) is an internal, non-public evaluation dataset. Chinese and Japanese use character-level evaluation units; other languages use word-level units. Some MCS reference timestamps were generated using Qwen Aligner, so these results are not an independent assessment of its boundary accuracy.

configuration, including two MTP draft tokens (Table 16). vLLM selects the kernels for this GPU automatically, which is why the BF16 values differ slightly from the main tables, obtained on H100 GPUs. We summarize the differences over 77 per-test-set error rates: the nine sets of the English Open ASR Leaderboard, 37 FLEURS languages, 8 MLS languages, the Common Voice sets and Monsoon Hindi of the multilingual Open ASR Leaderboard, and the contextual and multi-speaker test sets, excluding the Mandarin meetings.

The FP8 checkpoint is nearly as accurate as the BF16 checkpoint: its error rate is 1.0% higher in relative terms on average (median 0.8%), and the largest relative increase is 6.6%, on FLEURS Croatian. The mean WER on the English Open ASR Leaderboard increases by 0.04 points, and the contextual and multi-speaker error rates change by at most 0.46 points. The NVFP4 checkpoint loses more accuracy: its error rate is 3.6% higher on average (median 3.4%), and it is higher on 67 of the 77 sets (FP8: 60). In English, the difference is again small, 0.03 points on the Open ASR Leaderboard, and the contextual benchmarks change by at most 0.42 points. The regression is larger in other languages, e.g., 0.27 points (5.8% relative) on the multilingual Open ASR Leaderboard,

Table 16 | Accuracy of the BF16, FP8, and NVFP4 checkpoints in a single evaluation on one GB300 GPU (%; lower is better except for F1). All checkpoints use the same instructions, serving configuration, and loop guardrail. FLEURS and MLS: macro averages over 37 and 8 languages; ContextASR-Bench Speech (English): WER / NE-WER; Contextual Earnings-22: WER / keyword F1; multi-speaker rows: full sessions.

Benchmark	BF16	FP8	NVFP4
<i>English and multilingual ASR</i>			
Open ASR Leaderboard, 9 public sets	4.70	4.74	4.73
Multilingual Open ASR Leaderboard	4.63	4.75	4.90
FLEURS	9.31	9.38	9.67
MLS, full test sets	5.17	5.25	5.37
Monsoon, Hindi (OIWER)	9.27	9.67	10.13
<i>Contextual ASR</i>			
ContextASR-Bench, no context	3.34 / 12.25	3.36 / 12.35	3.45 / 12.67
ContextASR-Bench, coarse	3.34 / 12.25	3.35 / 12.27	3.45 / 12.65
ContextASR-Bench, fine	1.70 / 2.16	1.73 / 2.25	1.75 / 2.26
Contextual Earnings-22, no keywords	12.52 / 73.6	12.50 / 72.9	12.48 / 73.7
Contextual Earnings-22, local	9.77 / 97.2	9.60 / 97.4	9.57 / 97.3
Contextual Earnings-22, global	10.22 / 93.8	10.27 / 93.6	10.31 / 93.4
Preference-ASR, default	7.59	7.58	7.58
Preference-ASR, instructed	6.15	6.17	6.12
<i>Multi-speaker transcription (cpWER; Mandarin: cpCER)</i>			
AMI, IHM	11.68	11.66	12.19
AMI, SDM	17.64	17.85	18.75
CALLHOME	11.50	11.58	11.60
Mixer 6	15.85	15.96	16.01
NOTSOFAR-1, IHM	14.99	14.92	15.00
NOTSOFAR-1, SC	26.68	26.22	26.81
AISHELL-4	27.64	26.96	34.44
AliMeeting	36.50	35.34	38.10

0.36 points (3.9%) on FLEURS, and 0.86 points (9.3%) on Monsoon Hindi, compared with 0.12, 0.07, and 0.40 points for FP8, and on multi-speaker transcription with a distant microphone, e.g., 1.1 points on AMI SDM. On the Mandarin meetings, NVFP4 increases the cpCER by 6.8 points on AISHELL-4, whereas FP8 reduces it on both corpora; with 20 sessions per test set and a model that saw little Mandarin multi-speaker data (Section 4.8), these results are dominated by a few sessions and are not a reliable indication of the effect of quantization. We therefore recommend the FP8 checkpoint where accuracy outside English matters most, and the NVFP4 checkpoint where memory or throughput is the main constraint (Section 4.13).

4.13. Deployment and Inference Efficiency

Besides vLLM, the model runs with Hugging Face Transformers and NeMo-Speech.cpp, and it is supported on a range of NVIDIA platforms, from data-center GPUs to workstations, edge devices, and consumer GPUs, in BF16, FP8, or NVFP4 precision depending on the platform (Table 17).

We measure the latency and throughput of the BF16, FP8, and NVFP4 checkpoints (Section 2.7) with each speculative decoding mode (Section 4.1) on a single RTX PRO 6000 Blackwell Server Edition GPU with 96 GB of memory. Each configuration is served by the Nemotron Transcribe NIM (vLLM 0.29 with our serving optimizations) as an OpenAI-compatible HTTP server with at most

Table 17 | Supported hardware and precisions. All platforms support vLLM 0.28 and Hugging Face Transformers on Linux, and on Windows through WSL2.

Platform	Hardware	Precision
Data center	A100 80GB	BF16
	H100, H200	BF16, FP8
	B200, GB200	BF16, FP8, NVFP4
	L40S	FP8
	RTX PRO 6000 Blackwell Server Edition	BF16, FP8, NVFP4
Workstation	DGX Spark (GB10)	BF16, FP8, NVFP4
	GB300 DGX Station	BF16, FP8, NVFP4
Edge	Jetson Thor	BF16, FP8, NVFP4
Consumer	RTX 5090	NVFP4
	RTX Spark	NVFP4

64 concurrent sequences, at most 4,096 tokens per scheduler step, chunked prefill, 85% of the GPU memory reserved for the engine with a fixed KV cache of 32 GiB (NVFP4), 24 GiB (FP8), or 8 GiB (BF16), a context of 65,536 tokens, and the ReplaySSM Mamba cache mode (Liou et al., 2026), which stores the recurrent state of each Mamba layer as a periodically updated checkpoint plus a buffer of the most recent inputs, from which the current state is recomputed; this avoids writing the state at most decoding steps and reduces the rollback of rejected draft tokens to moving a buffer pointer. These serving optimizations are not yet part of the open-source release, and we aim to upstream them to NeMo Speech and vLLM at a later date. The vLLM integration shipped in NeMo Speech on the release day is intended for model development and is expected to be roughly two times slower. Prefix caching and audio feature caching are disabled, so that every request is processed in full. The FP8 and NVFP4 configurations store the KV cache in FP8. The CTC timestamp adapter stays loaded in every configuration, but no request asks for timestamps in any configuration. A client sends real speech excerpts of 30 seconds and of 10 minutes for verbatim transcription, with reasoning disabled, greedy decoding, and an output limit of 16,384 tokens, at a fixed number of concurrent requests between 1 and 256; requests that exceed the 64 engine slots wait in the server queue. We report the median latency of a request, measured from sending the audio to receiving the last token of the streamed response, and the throughput as the inverse real-time factor (RTFx), i.e., the number of seconds of audio transcribed per second of wall-clock time. Each data point aggregates at least two waves of requests after a warm-up, and the kernels compiled just in time by FlashInfer (Ye et al., 2025) are built before the measurement.

Figure 8 shows the results. On 30-second excerpts, speculative decoding reduces the latency of a single request by a factor of 1.4–1.5 with MTP and 2.0–2.5 with DFlash2. The acceptance length, i.e., the average number of tokens produced per verification step (at most the number of draft tokens plus one), is 8.0 for MTP and 9.7–10.1 for DFlash2. At high concurrency, speculative decoding also increases throughput, by a factor of 2.6–2.9 in BF16, 2.2–2.6 in FP8, and 2.0–2.4 in NVFP4, since a step of the backbone then produces several tokens for each of the 64 sequences. Lower precision speeds up every configuration: with MTP, FP8 transcribes up to 1,530 and NVFP4 up to 1,811 seconds of audio per second, compared to 1,021 in BF16. The fastest configuration, NVFP4 with DFlash2, reaches 2,238 RTFx and returns the transcript of a 30-second excerpt in 0.13 seconds when it serves one request at a time. On 10-minute recordings, the acceptance length of DFlash2 drops to 6.1 tokens on average, below that of MTP (7.9 tokens). In FP8 and NVFP4, DFlash2 still gives the lowest latency for a single request, e.g., 2.9 seconds for a 10-minute recording with NVFP4 (in BF16, MTP is slightly faster), but MTP gives the highest throughput in all precisions: 458 RTFx in BF16,

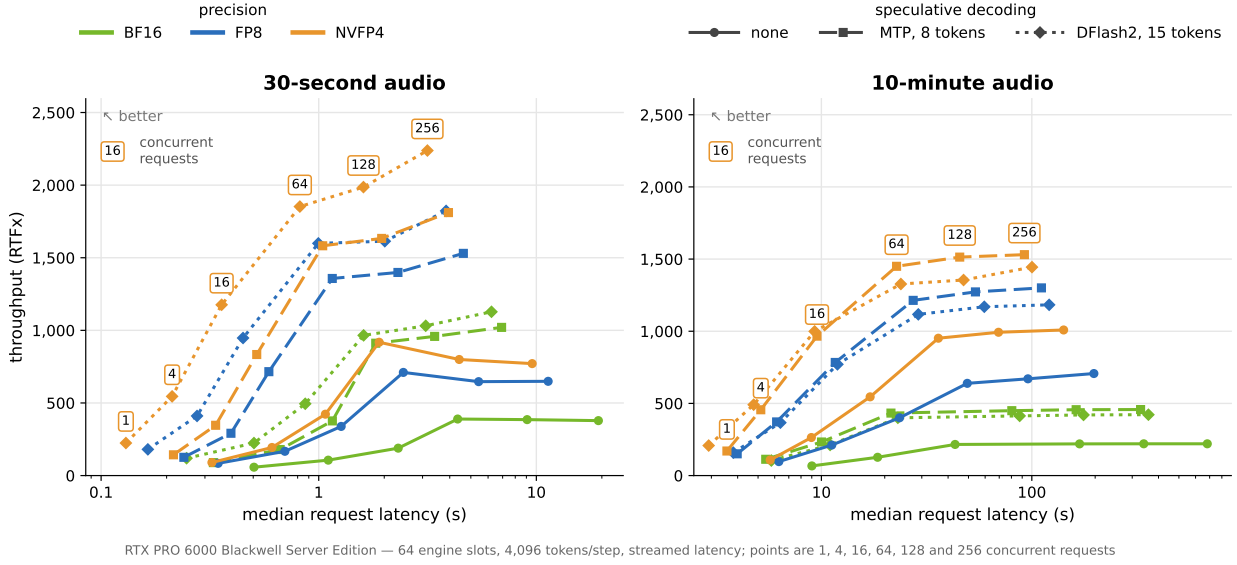


Figure 8 | Throughput (RTFx, higher is better) versus median request latency (lower is better) of Nemotron 3.5 Transcribe on a single RTX PRO 6000 Blackwell Server Edition GPU, for 30-second and 10-minute speech excerpts. Colors denote the precision of the checkpoint, and line styles and markers the speculative decoding mode. From left to right, the points of every line correspond to 1, 4, 16, 64, 128, and 256 concurrent requests, as labeled on the line with the highest throughput.

Table 18 | Pooled MAL for the released MTP and DFlash2 draft models on nine short-form English ASR test sets (75,443 requests) and three long-form corpora (440 recordings). All outputs are included, using vLLM counters before EOS or generation-limit trimming.

Draft model	K	Short-form MAL	Long-form MAL
MTP	2	2.93	2.82
	7	6.72	5.98
	15	10.02	7.84
DFlash2	7	6.39	4.05
	15	8.54	4.80

1,300 in FP8, and 1,531 in NVFP4. None of the 10-minute requests ran into a generation loop: the longest transcript of every configuration is within 1% of the length of its reference.

Draft acceptance across datasets. We evaluate the released MTP module with $K = 2, 7$, and 15 draft tokens and the released DFlash2 draft model with $K = 7$ and 15 on the nine short-form English ASR test sets (Section 4.2) and the three long-form corpora in Table 7. This dataset evaluation uses eight H100 GPUs; the speed benchmark above uses a single RTX PRO 6000 GPU, with $K = 8$ for MTP and $K = 15$ for DFlash2. We report pooled mean acceptance length (MAL), $1 + N_{\text{accepted}}/N_{\text{rounds}}$, using vLLM’s total accepted draft tokens and draft rounds. Counters are summed across requests and datasets; per-request MAL values are not averaged. MAL increases with K for both draft models, but remains lower on long-form inputs. MTP has higher MAL at matched K , with a larger gap on long-form audio (Table 18). Appendices C.1 and C.3 give per-dataset pooled and request-mean MAL, with generation-limit counts.

4.14. Audio Understanding

Nemotron 3.5 Transcribe is a transcription model. We add audio understanding because it makes transcription more useful, e.g., to reason about speakers and context or to handle non-speech audio, and not to compete with general-purpose omni models. We nevertheless evaluate it on established benchmarks, so that users can judge its scope: MMAR (Ma et al., 2025), using the 294 questions of its speech subset; MMAU (Sakshi et al., 2025), using the 9,000 questions of its test split, which covers speech, sound, and music, scored with the official evaluation script; MMAU-Pro (Kumar et al., 2026), using its non-music multiple-choice questions and its open-ended questions; and the speech tasks of the AIR-Bench foundation track (Yang et al., 2024), which cover paralinguistic and speaker-related questions. These benchmarks, like those in Sections 4.15 and 4.16, pose their questions in English; the audio understanding and reasoning capabilities of Nemotron 3.5 Transcribe may not work reliably in other languages. All results in this subsection and in Sections 4.15 and 4.16 are obtained with reasoning enabled (Section 4.1). We use task-specific system prompts (Appendix A), which specify the expected answer and its format; compared with a generic instruction, they improve, e.g., arithmetic with short numbers on Spoken-MQA from 47.0% to 90.0% and MMAR from 59.2% to 66.7%. We also compare with Nemotron 3 Nano Omni (NVIDIA, 2026b), NVIDIA’s general-purpose omni model built on an LLM of the same size, using its official results where available; for the benchmarks not covered by its report, we evaluated it with the same benchmark data and scoring and its recommended reasoning settings, but with the generic instructions of our earlier evaluation.

Table 19 compares Nemotron 3.5 Transcribe with recent open models with audio input, most notably Qwen3-Omni-30B-A3B (Xu et al., 2025b), which uses an LLM of the same size class, as well as Qwen2.5-Omni-7B (Xu et al., 2025a), Kimi-Audio (Kimi Team et al., 2025), Audio Flamingo 3 (Ghosh et al., 2025), and GPT-4o-Audio (OpenAI et al., 2024), and lists the best published result on each benchmark. Nemotron 3.5 Transcribe is ahead of the smaller open models on MMAR, MMAU-Pro, and the MMAU test set, the latter mainly owing to its accuracy on speech questions, and slightly behind Qwen2.5-Omni-7B on AIR-Bench. Qwen3-Omni-30B-A3B remains clearly ahead on MMAR, MMAU-Pro multiple-choice questions, and AIR-Bench, and the best published results are ahead on all benchmarks. On AIR-Bench (Table 20), Nemotron 3.5 Transcribe recognizes the spoken language, intent, and gender reliably, but, like most of the other models, struggles with age estimation.

4.15. Voice Instruction Following

Voice instruction following benchmarks test whether the model responds to a request that is *spoken* rather than written, instead of transcribing it. We evaluate on VoiceBench (Chen et al., 2026b), which covers open-ended requests, knowledge questions, reasoning, instruction following, and refusal of harmful requests; on Spoken-MQA (Wei et al., 2025), which poses arithmetic and math word problems in speech; and on the instruction-following subset of MMAU-Pro.

Table 21 shows that Nemotron 3.5 Transcribe outperforms Qwen2.5-Omni-7B and Kimi-Audio on VoiceBench overall, and Qwen2.5-Omni-7B on Spoken-MQA, where it answers 90–92% of the questions correctly in every category and achieves the highest published accuracy on arithmetic with long numbers. The remaining gap to the strongest omni models is largest on knowledge-heavy spoken questions (SD-QA) and on the open-ended requests rated by an LLM judge. Compared with the official results of Nemotron 3 Nano Omni, Nemotron 3.5 Transcribe is slightly behind on MMAU and behind on VoiceBench (Tables 19 and 21). On the remaining benchmarks, we evaluated Nemotron 3 Nano Omni ourselves with generic instructions, so the comparison is not like-for-like: Nemotron 3.5 Transcribe is on par on AIR-Bench (Table 20), higher on MMAR, MMAU-Pro multiple-choice questions, and Spoken-MQA arithmetic, and lower on Spoken-MQA reasoning and

Table 19 | Audio understanding compared with published results: accuracy (%) or judge score (%), higher is better. N3 Omni: Nemotron 3 Nano Omni, official results (NVIDIA, 2026b) (MMAU with reasoning disabled), except for entries marked *, which we obtained with the generic instructions of our earlier evaluation; Q3-Omni: Qwen3-Omni-30B-A3B Instruct; Q2.5-Omni: Qwen2.5-Omni-7B; AF3: Audio Flamingo 3.

Benchmark	Ours	Open-weights				AF3	Commercial	Best published
		N3 Omni	Q3-Omni	Q2.5-Omni	Kimi-Audio		GPT-4o	
MMAR, speech ^a	66.7	62.2*	77.2	59.9	–	–	70.4	82.0 (Gemini 2.5 Pro)
MMAU test ^b	73.4	74.6	–	71.0	64.4	72.4	60.8	76.0 (Audio-Thinker)
speech	76.7	72.8	–	68.9	56.6	67.0	69.3	81.8 (Nova 2 Omni)
sound	74.9	76.9	–	76.8	70.7	75.8	63.2	80.6 (Step-Audio 2)
music	68.6	74.2	–	67.3	65.9	74.5	49.9	74.5 (Audio Flamingo 3)
MMAU-Pro, multiple choice ^c	52.7	47.2*	57.5	47.9	44.2	49.3	48.9	71.5 (Gemini 3.1 Pro)
MMAU-Pro, open-ended ^d	66.7	–	–	52.3	34.5	44.2	43.2	68.3 (Gemini 3.1 Pro)
AIR-Bench speech, 6 tasks ^e	72.8	72.8*	76.5	74.6	–	–	–	77.3 (AZeroS)

^aSpeech column of the MMAR authors’ table (Ma et al., 2025); the Qwen3-Omni result is from a third-party evaluation (He et al., 2026). ^bTest split of MMAU-v05.15.25; baselines from the official leaderboard. Qwen3-Omni-30B-A3B (Xu et al., 2025b) and Qwen3.5-Omni-Plus (Qwen Team, 2026a) report 77.5 and 82.2 on MMAU without a per-domain breakdown or a test-split result. ^cNon-music questions; baselines are a micro-average over the non-music categories of the official results, weighted by the number of questions. ^dOverall score of the official Qwen2.5-7B-Instruct judge, normalized to percent; Nemotron 3.5 Transcribe: correctness 56.6, relevance 71.1, completeness 60.6, clarity 81.0. ^eMean accuracy over gender, emotion, age, language identification, entity, and intent, as evaluated in Shao et al. (2026); per-task results in Table 20.

Table 20 | AIR-Bench foundation speech tasks: accuracy (%); higher is better. N3 Omni: Nemotron 3 Nano Omni, evaluated by us with the generic instructions of our earlier evaluation, since its report does not include AIR-Bench. Other baselines as evaluated in Shao et al. (2026); GPT-4o refused the gender and age questions. Gender and emotion average two datasets each.

Task	Ours	Open-weights			Commercial		Research
		N3 Omni	Q3-Omni	Q2.5-Omni	Gemini 2.5 Pro	GPT-4o	AZeroS
Gender	88.0	82.5	91.1	89.8	90.7	–	86.8
Emotion	60.1	60.3	62.2	54.9	60.7	49.1	71.5
Age	29.6	31.6	36.9	44.8	34.1	–	61.3
Language ID	99.6	96.5	97.7	89.7	99.1	76.0	84.8
Entity	69.9	75.1	80.4	79.7	68.5	61.6	73.6
Intent	89.6	90.9	90.7	88.6	92.2	85.8	85.6
Mean	72.8	72.8	76.5	74.6	74.2	–	77.3

MMAU-Pro instruction following. This is expected, since Nemotron 3.5 Transcribe retains these capabilities in support of transcription, whereas Nemotron 3 Nano Omni is built as a general-purpose assistant; it also makes Nemotron 3.5 Transcribe a good starting point for fine-tuning towards audio understanding tasks that require accurate, speaker-attributed transcripts, such as the analysis of meetings or customer calls (Appendix B). Nemotron 3.5 Transcribe refuses 94% of the harmful requests in AdvBench (Zou et al., 2023).

4.16. Text-Only Reasoning

Although Nemotron 3.5 Transcribe is not intended to be used for text-only applications, for completeness, we report results on text-only benchmarks (Table 22) to demonstrate that some degree of instruction following and prompt understanding capability was preserved from the base

Table 21 | Voice instruction following: VoiceBench, in which the spoken audio is the user’s request, Spoken-MQA, and MMAU-Pro instruction following. AlpacaEval, CommonEval, and WildVoice are rated on a 1–5 scale by GPT-4o-mini; AdvBench reports the refusal rate on harmful requests; the VoiceBench overall score averages the nine subsets, with the 1–5 ratings multiplied by 20; the remaining entries report accuracy (%). N3 Omni: Nemotron 3 Nano Omni, official results (NVIDIA, 2026b), except for entries marked *, which we obtained with the generic instructions of our earlier evaluation; †the report does not state the SD-QA subset. VoiceBench baselines from the official leaderboard, except Qwen models (vendor-reported in Xu et al. (2025b)); Spoken-MQA baselines from Wu et al. (2025) (for MPS, the best of its variants in each row); MMAU-Pro baselines from the official results.

Benchmark	Ours	Open-weights				Commercial	Best published
		N3 Omni	Q3-Omni	Q2.5-Omni	Kimi-Audio	GPT-4o	
<i>VoiceBench</i>							
AlpacaEval	4.5	4.8	4.7	–	4.5	4.8	4.9 (Ultravox)
CommonEval	4.4	4.6	4.5	–	4.0	4.5	4.7 (BR-Voice-Reasoner)
WildVoice	4.4	4.6	4.6	–	4.2	4.6	4.7 (BR-Voice-Reasoner)
SD-QA (USA)	57.1	71.4 [†]	76.9	–	63.1	75.5	88.6 (Ultravox)
MMSU	79.3	82.3	68.1	–	62.2	80.3	88.9 (Launa V1)
OpenBookQA	91.2	93.0	89.7	–	83.5	89.2	96.0 (BR-Voice-Reasoner)
BBH	84.4	91.1	80.4	–	69.7	84.1	95.9 (Launa V1)
IFEval	81.5	88.7	77.8	–	61.1	76.0	88.7 (N3 Omni)
AdvBench	94.4	100.0	99.3	–	100.0	98.7	100.0
Overall	83.8	89.4	85.5	73.6	76.9	86.8	90.6 (Launa V1)
<i>Spoken-MQA</i>							
Short-number arithmetic	90.0	88.0*	–	83.0	–	–	92.9 (Mini-Omni-Reasoner)
Long-number arithmetic	92.5	91.9*	–	45.1	–	–	88.4 (MPS)
Single-step reasoning	91.2	93.8*	–	85.2	–	–	96.0 (MPS)
Multi-step reasoning	90.4	92.9*	–	71.5	–	–	94.6 (MPS)
Weighted mean	90.8	92.8*	–	73.6	–	–	93.9 (MPS)
MMAU-Pro, instruction following	86.2	93.1*	–	61.3	42.3	82.5	95.1 (Gemini 2.5 Flash)

Table 22 | Text-only benchmarks with reasoning enabled: accuracy (%; IFEval: mean of prompt- and instruction-level strict and loose accuracy), higher is better. The results of Nemotron 3.5 Lightning 30B-A3B are those reported on its model card; –: not reported.

Benchmark	Nemotron 3.5 Transcribe	Nemotron 3.5 Lightning 30B-A3B
GPQA Diamond (Rein et al., 2024)	62.12	75.44
MMLU (Hendrycks et al., 2021)	83.99	–
MMLU-Pro (Wang et al., 2024)	74.14	81.94
IFEval (Zhou et al., 2023)	84.65	–

model. The training mixture contains only a small share of text-only data (Section 3.1) to prevent catastrophic forgetting, but not enough to maintain the full text domain capabilities. For reference, we list the results reported for Nemotron 3.5 Lightning 30B-A3B, which were obtained with a different evaluation setup and are therefore only roughly comparable.

5. Conclusions

We introduced Nemotron 3.5 Transcribe, a speech LLM for intelligent transcription that combines a Parallel Expert Encoder with the Nemotron 3.5 Lightning 30B-A3B hybrid MoE LLM. A single model performs verbatim, written-form, contextual, multi-speaker, and rich transcription, speech translation, and language identification, selected with a natural language instruction. It achieves a mean WER of 4.71% on the 9 public test sets of the English Open ASR Leaderboard, the best result among open-weights models; it uses contextual information to reduce named-entity word errors by a factor of five and missed entities by a factor of 10; and it keeps its speaker-attributed transcription quality from one-minute segments to full sessions, where it outperforms published cascaded and end-to-end systems. A lightweight CTC adapter adds speaker-attributed word-level timestamps, even for long and heavily overlapping speech recordings. Its audio understanding, which we include in support of transcription, is comparable to that of Nemotron 3 Nano Omni, a general-purpose omni model of the same size, whereas its voice instruction following remains behind; this makes Nemotron 3.5 Transcribe a good candidate for fine-tuning towards audio understanding tasks that build on accurate transcription.

Two design decisions shaped Nemotron 3.5 Transcribe. First, we treat the training mixture as a distribution of supervision rather than of audio: sampling sources by their answer-token mass, and using every ASR example for one of many tasks, allowed us to add capabilities with sub-percent sampling rates. Second, we inject speaker information from a frozen diarization model into the encoder output, which enables speaker-attributed transcription without changing the LLM. Although not demonstrated in this work, we expect the same mechanism to apply to other frozen “expert” models.

The model is available on Hugging Face⁸ under the OpenMDW-1.1 license,⁹ and the fine-tuning toolkit in NeMo Speech¹⁰ (Appendix B).

Limitations.

- **Model scope.** Nemotron 3.5 Transcribe is optimized for transcription. Although it retains some of the text reasoning capabilities of its underlying LLM (Section 4.16), preserving the full range of these capabilities is not a training objective, and the model should not be treated as a general-purpose assistant. Its audio understanding and reasoning capabilities are evaluated only in English and may not work reliably in other languages.
- **Evaluation coverage.** Our experiments do not exhaustively cover every combination of task, language, domain, acoustic condition, and prompt. Application-specific evaluation remains necessary, including for the 40 transcription languages reported in this work.
- **MTP training ablation.** We do not compare joint training with and without the MTP objective, so its contribution to transcription accuracy remains unquantified.
- **Long-form draft acceptance.** At the same draft length, both MTP and DFlash2 have lower MAL on the evaluated long-form corpora than on the short-form benchmarks, with a larger reduction for DFlash2 (Table 18).
- **Multilingual and multi-speaker speech.** Accuracy varies across languages and conditions. Code-switching across some language pairs, including Chinese–English, remains a limitation. Mandarin far-field meeting speech, which is under-represented in the training data blend, also remains challenging. Speaker-attributed transcription supports at most eight speakers.

⁸<https://huggingface.co/nvidia/Nemotron-3.5-Transcribe-30B-A4B>

⁹<https://openmdw.ai/license/1-1/>

¹⁰<https://github.com/NVIDIA-NeMo/Speech>

- **Prompt dependence.** Output conventions such as number normalization, script selection, and transcription style depend on the wording of the instruction. We recommend the task-specific instructions listed in Appendix A.
- **Auxiliary transformations.** Capabilities such as summarization and instruction-based PII redaction have not been established as uniformly reliable and should be validated for each application; model-based redaction should not be used as the sole privacy control (Appendix D).
- **Output robustness.** The model can produce spurious transcripts for non-speech audio, return empty outputs or refusals for valid transcription requests, and enter generation loops. We provide a runtime guardrail for repetition loops in Appendix E.
- **Conversational use.** Deployments that expose the model through a conversational interface require application-appropriate runtime safety guardrails, particularly for multilingual and multi-turn inputs, and should not rely on the model alone for content safety or privacy enforcement.
- **Streaming.** The released model and evaluations target offline transcription and do not provide native streaming transcription; streaming support remains future work.

Contributors

Piotr Żelasko, Kunal Dhawan, Taejin Park, Valentin Mendelev, Nithin Rao Koluguri, Dongji Gao, Slyné Deng, Tatiana Timofeeva, Weiqing Wang, Ivan Medennikov, Nune Tadevosyan, Meline Mkrtchyan, Davit Karamyan, Shuoyang Ding, Steve Huang, Mayank Jain, Arushi Dhingra, Sahin Chowdhury, Prabhsimran Singh, Yash Hayaran, Leili Tavabi, Ehsan Hosseini Asl, Yifan Peng, Nikolay Karpov, Zhehuai Chen, Vitaly Lavrukhin, Jagadeesh Balam, Boris Ginsburg

References

- Rosana Ardila, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber. Common Voice: A massively-multilingual speech corpus. In *Proc. Twelfth Language Resources and Evaluation Conference (LREC)*, pages 4218–4222, 2020.
- Argmax. OpenBench: Benchmarks for on-device speech recognition. <https://github.com/argmaxinc/OpenBench>, 2025.
- Mateusz Barański, Jan Jasiński, Julitta Bartolewska, Marcin Witkowski, and Konrad Kowalczyk. HALAS: A human-annotated dataset of hallucinations of modern ASR systems. In *Proc. Interspeech*, pages 6698–6702, 2026. doi: 10.21437/Interspeech.2026-337.
- Emanuele Bastianelli, Andrea Vanzo, Pawel Swietojanski, and Verena Rieser. SLURP: A spoken language understanding resource package. In *Proc. 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7252–7262, 2020. doi: 10.18653/v1/2020.emnlp-main.588.
- Tyler Baumgartner, Brandon Tai, Lisa Kaelin-Martin, Candice Fan, Luc Debaupte, Bill Wang, and Yi Zhong. VoiceCodeBench: Evaluating exact structured-token recovery in automatic speech recognition. *arXiv preprint arXiv:2608.28916*, 2026.
- Linda Brandschain, David Graff, Christopher Cieri, Kevin Walker, Chris Caruso, and Abby Neely. The Mixer 6 corpus: Resources for cross-channel and text independent speaker recognition. In *Proc. Seventh International Conference on Language Resources and Evaluation (LREC)*, 2010.

- Alexandra Canavan, David Graff, and George Zipperlen. CALLHOME American English speech LDC97S42. <https://catalog.ldc.upenn.edu/LDC97S42>, 1997.
- Houwei Cao, David G. Cooper, Michael K. Keutmann, Ruben C. Gur, Ani Nenkova, and Ragini Verma. CREMA-D: Crowd-sourced emotional multimodal actors dataset. *IEEE Transactions on Affective Computing*, 5(4):377–390, 2014. doi: 10.1109/TAFFC.2014.2336244.
- Jean Carletta, Simone Ashby, Sebastien Bourban, Mike Flynn, Mael Guillemot, Thomas Hain, Jaroslav Kadlec, Vasilis Karaiskos, Wessel Kraaij, Melissa Kronenthal, Guillaume Lathoud, Mike Lincoln, Agnes Lisowska, Iain McCowan, Wilfried Post, Dennis Reidsma, and Pierre Wellner. The AMI meeting corpus: A pre-announcement. In *Machine Learning for Multimodal Interaction (MLMI 2005)*, volume 3869 of *Lecture Notes in Computer Science*, pages 28–39. Springer, 2006. doi: 10.1007/11677482_3.
- Edresson Casanova, Kelly Davis, Eren Gölge, Gökem Gökmar, Iulian Gulea, Logan Hart, Aya Aljafari, Joshua Meyer, Reuben Morais, Samuel Olayemi, and Julian Weber. XTTS: a massively multilingual zero-shot text-to-speech model. In *Proc. Interspeech*, pages 4978–4982, 2024. doi: 10.21437/Interspeech.2024-2016.
- Guoguo Chen, Shuzhou Chai, Guan-Bo Wang, Jiayu Du, Wei-Qiang Zhang, Chao Weng, Dan Su, Daniel Povey, Jan Trmal, Junbo Zhang, Mingjie Jin, Sanjeev Khudanpur, Shinji Watanabe, Shuaijiang Zhao, Wei Zou, Xiangang Li, Xuchen Yao, Yongqing Wang, Zhao You, and Zhiyong Yan. GigaSpeech: An evolving, multi-domain ASR corpus with 10,000 hours of transcribed audio. In *Proc. Interspeech*, pages 3670–3674, 2021. doi: 10.21437/Interspeech.2021-1965.
- Jian Chen, Yesheng Liang, and Zhijian Liu. DFlash: Block diffusion for flash speculative decoding. In *Proc. 43rd International Conference on Machine Learning (ICML)*, volume 306 of *Proceedings of Machine Learning Research*, pages 15343–15355. PMLR, 2026a.
- Yiming Chen, Xianghu Yue, Chen Zhang, Xiaoxue Gao, Robby T. Tan, and Haizhou Li. VoiceBench: Benchmarking LLM-based voice assistants. *Transactions of the Association for Computational Linguistics*, 14:378–398, 2026b. doi: 10.1162/tacl.a.628.
- Zhehuai Chen, He Huang, Andrei Andrusenko, Oleksii Hrinchuk, Krishna C. Puvvada, Jason Li, Subhankar Ghosh, Jagadeesh Balam, and Boris Ginsburg. SALM: Speech-augmented language model with in-context learning for speech recognition and translation. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 13521–13525, 2024a. doi: 10.1109/ICASSP48485.2024.10447553.
- Zhehuai Chen, He Huang, Oleksii Hrinchuk, Krishna C. Puvvada, Nithin Rao Koluguri, Piotr Żelasko, Jagadeesh Balam, and Boris Ginsburg. BESTOW: Efficient and streamable speech language model with the best of two worlds in GPT and T5. In *Proc. IEEE Spoken Language Technology Workshop (SLT)*, pages 147–154, 2024b. doi: 10.1109/SLT61566.2024.10832146.
- Christopher Cieri, David Miller, and Kevin Walker. The Fisher corpus: a resource for the next generations of speech-to-text. In *Proc. Fourth International Conference on Language Resources and Evaluation (LREC)*, 2004.
- Alexis Conneau, Min Ma, Simran Khanuja, Yu Zhang, Vera Axelrod, Siddharth Dalmia, Jason Riesa, Clara Rivera, and Ankur Bapna. FLEURS: Few-shot learning evaluation of universal representations of speech. In *Proc. IEEE Spoken Language Technology Workshop (SLT)*, pages 798–805, 2022. doi: 10.1109/SLT54892.2023.10023141.

- Tri Dao. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*, 2024.
- Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 16344–16359, 2022.
- DeepSeek-AI. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.
- DeepSeek-AI. DeepEP: an efficient expert-parallel communication library. <https://github.com/deepseek-ai/DeepEP>, 2025. GitHub repository.
- Miguel Del Rio, Peter Ha, Quinten McNamara, Corey Miller, and Shipra Chandra. Earnings-22: A practical benchmark for accents in the wild. *arXiv preprint arXiv:2203.15591*, 2022.
- Anuj Diwan, Rakesh Vaideeswaran, Sanket Shah, Ankita Singh, Srinivasa Raghavan, Shreya Khare, Vinit Unni, Saurabh Vyas, Akash Rajpuria, Chiranjeevi Yarra, Ashish Mittal, Prasanta Kumar Ghosh, Preethi Jyothi, Kalika Bali, Vivek Seshadri, Sunayana Sitaram, Samarth Bharadwaj, Jai Nanavati, Raoul Nanavati, and Karthik Sankaranarayanan. MUCS 2021: Multilingual and code-switching ASR challenges for low resource Indian languages. In *Proc. Interspeech*, pages 2446–2450, 2021. doi: 10.21437/Interspeech.2021-1339.
- Eka Care. Eka medical ASR evaluation dataset. <https://huggingface.co/datasets/ekacare/eka-medical-asr-evaluation-dataset>, 2024. Hugging Face dataset.
- Eka Care. Parrotlet-a-en-5b: A purpose-built automatic speech recognition model for English speech in Indian healthcare. <https://huggingface.co/ekacare/parrotlet-a-en-5b>, 2025. Hugging Face model.
- Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Mohammad Norouzi, Douglas Eck, and Karen Simonyan. Neural audio synthesis of musical notes with WaveNet autoencoders. In *Proc. 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pages 1068–1077. PMLR, 2017.
- Jonathan G. Fiscus. A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER). In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 347–354, 1997. doi: 10.1109/ASRU.1997.659110.
- Yihui Fu, Luyao Cheng, Shubo Lv, Yukai Jv, Yuxiang Kong, Zhuo Chen, Yanxin Hu, Lei Xie, Jian Wu, Hui Bu, Xin Xu, Jun Du, and Jingdong Chen. AISHELL-4: An open source dataset for speech enhancement, separation, recognition and speaker diarization in conference scenario. In *Proc. Interspeech*, pages 3665–3669, 2021. doi: 10.21437/Interspeech.2021-1397.
- Sreyan Ghosh, Arushi Goel, Jaehyeon Kim, Sonal Kumar, Zhifeng Kong, Sang-gil Lee, Chao-Han Huck Yang, Ramani Duraiswami, Dinesh Manocha, Rafael Valle, and Bryan Catanzaro. Audio Flamingo 3: Advancing audio intelligence with fully open large audio language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, pages 46789–46856, 2025. doi: 10.52202/085713-1396.
- Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. Better & faster large language models via multi-token prediction. In *Proc. 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pages 15706–15734. PMLR, 2024.

- Yuan Gong, Jin Yu, and James Glass. VocalSound: A dataset for improving human vocal sounds recognition. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 151–155, 2022. doi: 10.1109/ICASSP43922.2022.9746828.
- Granary-v2 Team. Granary-v2: Scalable multilingual speech data curation for predictable and controllable model outputs. Technical report, NVIDIA, 2026. In preparation.
- Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *Proc. First Conference on Language Modeling (COLM)*, 2024.
- Jiangyu Han, Petr Pálka, Marc Delcroix, Federico Landini, Johan Rohdin, Jan Černocký, and Lukáš Burget. Efficient and robust speaker diarization via structured pruning of self-supervised models. *IEEE Transactions on Audio, Speech and Language Processing*, 34:1903–1914, 2026. doi: 10.1109/TASLPRO.2026.3675801.
- Xiang He, Chenxing Li, Jinting Wang, Yan Rong, Tianxin Xie, Zeyu Xie, Wenfu Wang, Li Liu, and Dong Yu. Audio-DeepThinker: Progressive reasoning-aware reinforcement learning for high-quality chain-of-thought emergence in audio language models. In *Proc. Interspeech*, pages 245–250, 2026. doi: 10.21437/Interspeech.2026-1720.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations (ICLR)*, 2021.
- hexgrad. Kokoro-82M. <https://huggingface.co/hexgrad/Kokoro-82M>, 2025. Hugging Face model.
- Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.
- Hugging Face. Open ASR leaderboard (results snapshot of 2026-10-02). https://huggingface.co/spaces/hf-audio/open_asr_leaderboard, 2026a.
- Hugging Face. Open ASR leaderboard (results snapshot of 2026-10-21). https://huggingface.co/spaces/hf-audio/open_asr_leaderboard, 2026b.
- Inclusion AI. Ming-flash-omni: A sparse, unified architecture for multimodal perception and generation. *arXiv preprint arXiv:2510.24821*, 2025. Version 3.
- Inco AI. DFlash 2: Keep Drafting Parallel, August 2026. URL <https://inco.ai/blog/dflash2/>.
- Adam Janin, Don Baron, Jane Edwards, Dan Ellis, David Gelbart, Nelson Morgan, Barbara Peskin, Thilo Pfau, Elizabeth Shriberg, Andreas Stolcke, and Chuck Wooters. The ICSI meeting corpus. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages I–364–I–367, 2003. doi: 10.1109/ICASSP.2003.1198793.
- Naoyuki Kanda, Yashesh Gaur, Xiaofei Wang, Zhong Meng, and Takuya Yoshioka. Serialized output training for end-to-end overlapped speech recognition. In *Proc. Interspeech*, pages 2797–2801, 2020. doi: 10.21437/Interspeech.2020-999.
- Naoyuki Kanda, Guoli Ye, Yashesh Gaur, Xiaofei Wang, Zhong Meng, Zhuo Chen, and Takuya Yoshioka. End-to-end speaker-attributed ASR with transformer. In *Proc. Interspeech*, pages 4413–4417, 2021. doi: 10.21437/Interspeech.2021-101.

- Kimi Team, Ding Ding, Zeqian Ju, Yichong Leng, Songxiang Liu, Tong Liu, Zeyu Shang, Kai Shen, Wei Song, Xu Tan, Heyi Tang, Zhengtao Wang, Chu Wei, Yifei Xin, Xinran Xu, Jianwei Yu, Yutao Zhang, Xinyu Zhou, Y. Charles, Jun Chen, Yanru Chen, Yulun Du, Weiran He, Zhenxing Hu, Guokun Lai, Qingcheng Li, Yangyang Liu, Weidong Sun, Jianzhou Wang, Yuzhi Wang, Yuefeng Wu, Yuxin Wu, Dongchao Yang, Hao Yang, Ying Yang, Zhilin Yang, Aoxiong Yin, Ruibin Yuan, Yutong Zhang, and Zaida Zhou. Kimi-Audio technical report. *arXiv preprint arXiv:2504.18425*, 2025.
- Nithin Rao Koluguri, Monica Sekoyan, George Zelenfroynd, Sasha Meister, Shuoyang Ding, Sofia Kostandian, He Huang, Nikolay Karpov, Jagadeesh Balam, Vitaly Lavrukhin, Yifan Peng, Sara Papi, Marco Gaido, Alessio Brutti, and Boris Ginsburg. Granary: Speech recognition and translation dataset in 25 European languages. In *Proc. Interspeech*, pages 3923–3927, 2025. doi: 10.21437/Interspeech.2025-540.
- Nithin Rao Koluguri, Sasha Meister, Nikolay Karpov, Piotr Żelasko, Desh Raj, Jagadeesh Balam, and Boris Ginsburg. Preference-ASR: A preference-aware test set for benchmarking ASR in the era of speech LLMs. In *Proc. Interspeech*, pages 6687–6691, 2026. doi: 10.21437/Interspeech.2026-728.
- Oleksii Kuchaiev, Jason Li, Huyen Nguyen, Oleksii Hrinchuk, Ryan Leary, Boris Ginsburg, Samuel Kriman, Stanislav Beliaev, Vitaly Lavrukhin, Jack Cook, Patrice Castonguay, Mariya Popova, Jocelyn Huang, and Jonathan M. Cohen. NeMo: a toolkit for building AI applications using neural modules. *arXiv preprint arXiv:1909.09577*, 2019.
- Sonal Kumar, Šimon Sedláček, Vaibhavi Lokegaonkar, Fernando López, Wenyi Yu, Nishit Anand, Hyeonggon Ryu, Lichang Chen, Maxim Plička, Miroslav Hlaváček, William Fineas Ellingwood, Sathvik Udupa, Siyuan Hou, Allison Ferner, Sara Barahona, Cecilia Bolaños, Satish Rahi, Laura Herrera-Alarcón, Satvik Dixit, Siddhi Patil, Soham Deshmukh, Lasha Koroshinadze, Yao Liu, Leibny Paola Garcia Perera, Eleni Zanou, Themis Stafylakis, Joon Son Chung, David Harwath, Chao Zhang, Dinesh Manocha, Alicia Lozano-Diez, Santosh Kesiraju, Sreyan Ghosh, and Ramani Duraiswami. MMAU-Pro: A challenging and comprehensive benchmark for holistic evaluation of audio general intelligence. In *Proc. AAAI Conference on Artificial Intelligence*, volume 40, pages 22688–22697, 2026. doi: 10.1609/aaai.v40i27.39430.
- Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pages 611–626, 2023. doi: 10.1145/3600006.3613165.
- Khai Le-Duc, Phuc Phan, Tan-Hanh Pham, Bach Phan Tat, Minh-Huong Ngo, Thanh Nguyen-Tang, and Truong-Son Hy. MultiMed: Multilingual medical speech recognition via attention encoder decoder. In *Proc. 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pages 1113–1150, 2025. doi: 10.18653/v1/2025.acl-industry.79.
- Colin Lea, Vikramjit Mitra, Aparna Joshi, Sachin Kajarekar, and Jeffrey P. Bigham. SEP-28k: A dataset for stuttering event detection from podcasts with people who stutter. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6798–6802, 2021. doi: 10.1109/ICASSP39728.2021.9413520.
- Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 19274–19286. PMLR, 2023. URL <https://proceedings.mlr.press/v202/leviathan23a.html>.

- Ze-Wei Liou, Harris Nover, Amey Naik, Andrii Skliar, Igor Shovkun, Ido Shahaf, and Tri Dao. ReplaySSM: Cache SSM inputs, not state. *arXiv preprint arXiv:2610.13778*, 2026.
- Alexander H. Liu, Andy Ehrenberg, Andy Lo, et al. Voxtral. *arXiv preprint arXiv:2507.13264*, 2025.
- Holy Lovenia, Samuel Cahyawijaya, Genta Indra Winata, Peng Xu, Yan Xu, Zihan Liu, Rita Frieske, Tiezheng Yu, Wenliang Dai, Elham J. Barezi, Qifeng Chen, Xiaojuan Ma, Bertram E. Shi, and Pascale Fung. ASCEND: A spontaneous Chinese-English dataset for code-switching in multi-turn conversation. In *Proc. Thirteenth Language Resources and Evaluation Conference (LREC)*, pages 7259–7268, 2022.
- Ziyang Ma, Yinghao Ma, Yanqiao Zhu, Chen Yang, Yi-Wen Chao, Ruiyang Xu, Wenxi Chen, Yuanzhe Chen, Zhuo Chen, Jian Cong, Kai Li, Keliang Li, Siyou Li, Xinfeng Li, Xiquan Li, Zheng Lian, Yuzhe Liang, Minghao Liu, Zhikang Niu, Tianrui Wang, Yuping Wang, Yuxuan Wang, Yihao Wu, Guanrou Yang, Jianwei Yu, Ruibin Yuan, Zhisheng Zheng, Ziya Zhou, Haina Zhu, Wei Xue, Emmanouil Benetos, Kai Yu, Eng-Siong Chng, and Xie Chen. MMAR: A challenging benchmark for deep reasoning in speech, audio, music, and their mix. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, pages 74837–74868, 2025. doi: 10.52202/085713-2252.
- Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi. In *Proc. Interspeech 2017*, pages 498–502, 2017. doi: 10.21437/Interspeech.2017-1386.
- Lingwei Meng, Shujie Hu, Jiawen Kang, Zhaoqing Li, Yuejiao Wang, Wenxuan Wu, Xixin Wu, Xunying Liu, and Helen Meng. Large language model can transcribe speech in multi-talker scenarios with versatile instructions. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2025. doi: 10.1109/ICASSP49660.2025.10887642.
- MOSI.AI, Donghua Yu, Zhengyuan Lin, Hanfu Chen, Chen Yang, Yiyang Zhang, Jingqi Chen, Ke Chen, Liwei Fan, Yi Jiang, Jie Zhu, Muchen Li, Wenxuan Wang, Yang Wang, Zhe Xu, Botian Jiang, Yitian Gong, Yuqian Zhang, Wenbo Zhang, Songlin Wang, Zhiyu Wu, Zhaoye Fei, Qinyuan Cheng, Shimin Li, and Xipeng Qiu. MOSS Transcribe Diarize technical report. *arXiv preprint arXiv:2601.01554*, 2026. doi: 10.48550/arXiv.2601.01554.
- Blaise Munyampirwa, Arda Ibis, Zach Nagengast, Brian Keene, and Dylan Angus. Contextual Earnings-22: A speech recognition benchmark with custom vocabulary in the wild. In *Proc. Interspeech*, pages 1949–1953, 2026. doi: 10.21437/Interspeech.2026-1375.
- NVIDIA. Canary-Qwen-2.5B. <https://huggingface.co/nvidia/canary-qwen-2.5b>, 2025a.
- NVIDIA. NeMo RL: A scalable and efficient post-training library. <https://github.com/NVIDIA-NeMo/RL>, 2025b.
- NVIDIA. MagpieTTS multilingual 357m. https://huggingface.co/nvidia/magpie_tts_multilingual_357m, 2025c. Hugging Face model.
- NVIDIA. Multitalker-Parakeet-Streaming-0.6B-v1. <https://huggingface.co/nvidia/multitalker-parakeet-streaming-0.6b-v1>, 2025d. Hugging Face model card.
- NVIDIA. Nemotron 3 Nano: Open, efficient mixture-of-experts hybrid Mamba-transformer model for agentic reasoning. *arXiv preprint arXiv:2512.20848*, 2025e.
- NVIDIA. Nemotron-H: A family of accurate and efficient hybrid Mamba-transformer models. *arXiv preprint arXiv:2504.03624*, 2025f.

- NVIDIA. Numb3rs: Numbers speech benchmark dataset. <https://huggingface.co/datasets/nvidia/Numb3rs>, 2025g. Hugging Face dataset.
- NVIDIA. NVIDIA Model Optimizer. <https://github.com/NVIDIA/Model-Optimizer>, 2026a. GitHub repository; formerly NVIDIA TensorRT Model Optimizer.
- NVIDIA. Nemotron 3 Nano Omni: Efficient and open multimodal intelligence. *arXiv preprint arXiv:2604.24954*, 2026b.
- NVIDIA. Nemotron-3-Diarization. <https://huggingface.co/nvidia/Nemotron-3-Diarization>, 2026c. Hugging Face model card, released September 2026.
- NVIDIA. Transformer Engine. <https://github.com/NVIDIA/TransformerEngine>, 2026d. GitHub repository.
- Patrick K. O’Neill, Vitaly Lavrukhin, Somshubra Majumdar, Vahid Noroozi, Yuekai Zhang, Oleksii Kuchaiev, Jagadeesh Balam, Yuliya Dovzhenko, Keenan Freyberg, Michael D. Shulman, Boris Ginsburg, Shinji Watanabe, and Georg Kucsco. SPGISpeech: 5,000 hours of transcribed financial audio for fully formatted end-to-end speech recognition. In *Proc. Interspeech*, pages 1434–1438, 2021. doi: 10.21437/Interspeech.2021-1860.
- OpenAI, Aaron Hurst, et al. GPT-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.
- Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: An ASR corpus based on public domain audio books. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210, 2015. doi: 10.1109/ICASSP.2015.7178964.
- Taejin Park, Ivan Medennikov, Kunal Dhawan, Weiqing Wang, He Huang, Nithin Rao Koluguri, Krishna C. Puvvada, Jagadeesh Balam, and Boris Ginsburg. Sortformer: A novel approach for permutation-resolved speaker supervision in speech-to-text systems. In *Proc. 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pages 48153–48169. PMLR, 2025.
- Yifan Peng, Krishna C. Puvvada, Zhehuai Chen, Piotr Żelasko, He Huang, Kunal Dhawan, Ke Hu, Shinji Watanabe, Jagadeesh Balam, and Boris Ginsburg. VoiceTextBlender: Augmenting large language models with speech capabilities via single-stage joint speech-text supervised fine-tuning. In *Proc. 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics (NAACL)*, pages 5787–5802, 2025. doi: 10.18653/v1/2025.naacl-long.298.
- Zhiliang Peng, Jianwei Yu, Yaoyao Chang, Zilong Wang, Li Dong, Yingbo Hao, Yujie Tu, Chenyu Yang, Wenhui Wang, Songchen Xu, Yutao Sun, Hangbo Bao, Weijiang Xu, Yi Zhu, Zehua Wang, Ting Song, Yan Xia, Zewen Chi, Shaohan Huang, Liang Wang, Chuang Ding, Shuai Wang, Xie Chen, and Furu Wei. VibeVoice-ASR technical report. *arXiv preprint arXiv:2601.18184*, 2026.
- Karol J. Piczak. ESC: Dataset for environmental sound classification. In *Proc. 23rd ACM International Conference on Multimedia*, pages 1015–1018, 2015. doi: 10.1145/2733373.2806390.
- Alexander Polok, Dominik Klement, Samuele Cornell, Matthew Wiesner, Jan Černocký, Sanjeev Khudanpur, and Lukáš Burget. SE-DiCoW: Self-enrolled diarization-conditioned Whisper. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 16712–16716, 2026. doi: 10.1109/ICASSP55912.2026.11461785.
- Vineel Pratap, Qiantong Xu, Anuroop Sriram, Gabriel Synnaeve, and Ronan Collobert. MLS: A large-scale multilingual dataset for speech research. In *Proc. Interspeech*, pages 2757–2761, 2020. doi: 10.21437/Interspeech.2020-2826.

- Vineel Pratap, Andros Tjandra, Bowen Shi, Paden Tomasello, Arun Babu, Sayani Kundu, Ali Elkahky, Zhaoheng Ni, Apoorv Vyas, Maryam Fazel-Zarandi, Alexei Baevski, Yossi Adi, Xiaohui Zhang, Wei-Ning Hsu, Alexis Conneau, and Michael Auli. Scaling speech technology to 1,000+ languages. *Journal of Machine Learning Research*, 25(97):1–52, 2024.
- Krishna C. Puvvada, Piotr Żelasko, He Huang, Oleksii Hrinchuk, Nithin Rao Koluguri, Kunal Dhawan, Somshubra Majumdar, Elena Rastorgueva, Zhehuai Chen, Vitaly Lavrukhin, Jagadeesh Balam, and Boris Ginsburg. Less is more: Accurate speech recognition & translation without web-scale data. In *Proc. Interspeech*, pages 3964–3968, 2024. doi: 10.21437/Interspeech.2024-2294.
- Qwen Team. Qwen3.5-Omni technical report. *arXiv preprint arXiv:2604.15804*, 2026a.
- Qwen Team. Qwen3-ASR: inference toolkit (qwen-asr). <https://github.com/QwenLM/Qwen3-ASR>, 2026b. Function `detect_and_fix_repetitions` in `qwen_asr/inference/utils.py`, commit 7c6daf7.
- Qwen Team. Qwen3-ForcedAligner-0.6B. <https://huggingface.co/Qwen/Qwen3-ForcedAligner-0.6B>, 2026c.
- Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *Proc. 40th International Conference on Machine Learning (ICML)*, volume 202 of *Proceedings of Machine Learning Research*, pages 28492–28518. PMLR, 2023.
- David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level Google-proof Q&A benchmark. In *Proc. First Conference on Language Modeling (COLM)*, 2024.
- S Sakshi, Utkarsh Tyagi, Sonal Kumar, Ashish Seth, Ramaneswaran Selvakumar, Oriol Nieto, Ramani Duraiswami, Sreyan Ghosh, and Dinesh Manocha. MMAU: A massive multi-task audio understanding and reasoning benchmark. In *International Conference on Learning Representations (ICLR)*, 2025.
- Alexander Samarin, Sergei Krutikov, Anton Shevtsov, Sergei Skvortsov, Filipp Fisin, and Alexander Golubev. LK Losses: Direct acceptance rate optimization for speculative decoding. *arXiv preprint arXiv:2602.23881*, 2026. URL <https://arxiv.org/abs/2602.23881>.
- Ramon Sanabria, Nikolay Bogoychev, Nina Markl, Andrea Carmantini, Ondrej Klejch, and Peter Bell. The Edinburgh international accents of English corpus: Towards the democratization of English ASR. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2023. doi: 10.1109/ICASSP49357.2023.10095057.
- George Saon, Avihu Dekel, Alexander Brooks, Tohru Nagano, Abraham Daniels, Aharon Satt, Ashish Mittal, Brian Kingsbury, David Haws, Edmilson Morais, Gakuto Kurata, Hagai Aronowitz, Ibrahim Ibrahim, Jeff Kuo, Kate Soule, Luis Lastras, Masayuki Suzuki, Ron Hoory, Samuel Thomas, Sashi Novitasari, Takashi Fukuda, Vishal Sunder, Xiaodong Cui, and Zvi Kons. Granite-speech: open-source speech-aware LLMs with strong English ASR capabilities. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1–7, 2025. doi: 10.1109/ASRU65441.2025.11434651.
- Monica Sekoyan, Nithin Rao Koluguri, Nune Tadevosyan, Piotr Żelasko, Travis Bartley, Nikolay Karpov, Jagadeesh Balam, and Boris Ginsburg. Canary-1b-v2 & parakeet-tdt-0.6b-v3: Efficient and high-performance models for multilingual ASR and AST. *arXiv preprint arXiv:2509.14128*, 2025.

- Yiwen Shao, Wei Liu, Jiahong Li, Tianzi Wang, Kun Wei, Meng Yu, and Dong Yu. AzeroS: Extending LLM to speech with self-generated instruction-free tuning. *arXiv preprint arXiv:2601.06086*, 2026.
- Xian Shi, Xiong Wang, Zhifang Guo, Yongqi Wang, Pei Zhang, Xinyu Zhang, Zishan Guo, Hongkun Hao, Yu Xi, Baosong Yang, Jin Xu, Jingren Zhou, and Junyang Lin. Qwen3-ASR technical report. *arXiv preprint arXiv:2601.21337*, 2026.
- Ganesh Sinisetty, Pavlo Ruban, Oleksandr Dymov, and Mirco Ravanelli. CommonLanguage (version 0.1) [dataset]. <https://doi.org/10.5281/zenodo.5036977>, 2021.
- Vaibhav Srivastav, Steven Zheng, Eric Bezzam, Eustache Le Bihan, Nithin Rao Koluguri, Piotr Żelasko, Somshubra Majumdar, Adel Moumen, and Sanchit Gandhi. Open ASR leaderboard: Towards reproducible and transparent multilingual and long-form speech recognition evaluation. In *Proc. Interspeech*, pages 6692–6697, 2026. doi: 10.21437/Interspeech.2026-1902.
- SYSTRAN. Faster Whisper transcription with CTranslate2. <https://github.com/SYSTRAN/faster-whisper>, 2026. Software repository.
- George Tzanetakis and Perry Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002. doi: 10.1109/TSA.2002.800560.
- Jörgen Valk and Tanel Alumäe. VoxLingua107: a dataset for spoken language recognition. In *Proc. IEEE Spoken Language Technology Workshop (SLT)*, pages 652–658, 2021. doi: 10.1109/SLT48900.2021.9383459.
- Maarten Van Segbroeck, Ahmed Zaid, Ksenia Kutsenko, Cirenía Huerta, Tinh Nguyen, Xuewen Luo, Björn Hoffmeister, Jan Trmal, Maurizio Omologo, and Roland Maas. DiPCo – dinner party corpus. In *Proc. Interspeech*, pages 434–436, 2020. doi: 10.21437/Interspeech.2020-2800.
- Alon Vinnikov, Amir Ivry, Aviv Hurvitz, Igor Abramovski, Sharon Koubi, Ilya Gurvich, Shai Pe’er, Xiong Xiao, Benjamin Martínez Elizalde, Naoyuki Kanda, Xiaofei Wang, Shalev Shaer, Stav Yagev, Yossi Asher, Sunit Sivasankaran, Yifan Gong, Min Tang, Huaming Wang, and Eyal Krupka. NOTSOFAR-1 challenge: New datasets, baseline, and tasks for distant meeting transcription. In *Proc. Interspeech*, pages 5003–5007, 2024. doi: 10.21437/Interspeech.2024-1788.
- Voice Arena. Voice Arena Monsoon en-IN and hi-IN (public test sets). <https://huggingface.co/datasets/VoiceArena/MonsoonASR-Open-ASR-leaderboard-en-IN>, <https://huggingface.co/datasets/VoiceArena/MonsoonASR-Open-ASR-leaderboard-hi-IN>, 2026. Hugging Face datasets; see also <https://huggingface.co/blog/open-asr-leaderboard-global-south>.
- Changhan Wang, Morgane Rivi  re, Ann Lee, Anne Wu, Chaitanya Talnikar, Daniel Haziza, Mary Williamson, Juan Pino, and Emmanuel Dupoux. VoxPopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation. In *Proc. 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 993–1003, 2021a. doi: 10.18653/v1/2021.acl-long.80.
- Changhan Wang, Anne Wu, Jiatao Gu, and Juan Pino. CoVoST 2 and massively multilingual speech translation. In *Proc. Interspeech*, pages 2247–2251, 2021b. doi: 10.21437/Interspeech.2021-2027.
- He Wang, Linhan Ma, Dake Guo, Xiong Wang, Lei Xie, Jin Xu, and Junyang Lin. ContextASR-Bench: A massive contextual speech recognition benchmark. *arXiv preprint arXiv:2507.05727*, 2025a.

- Weiqing Wang, Taejin Park, Ivan Medennikov, Jinhan Wang, Kunal Dhawan, He Huang, Nithin Rao Koluguri, Jagadeesh Balam, and Boris Ginsburg. Speaker targeting via self-speaker adaptation for multi-talker ASR. In *Proc. Interspeech*, pages 5498–5502, 2025b. doi: 10.21437/Interspeech.2025-2142.
- Xin Wang, Junichi Yamagishi, Massimiliano Todisco, Héctor Delgado, Andreas Nautsch, Nicholas Evans, Md Sahidullah, Ville Vestman, Tomi Kinnunen, Kong Aik Lee, Lauri Juvela, Paavo Alku, Yu-Huai Peng, Hsin-Te Hwang, Yu Tsao, Hsin-Min Wang, Sébastien Le Maguer, Markus Becker, Fergus Henderson, Rob Clark, Yu Zhang, Quan Wang, Ye Jia, Kai Onuma, Koji Mushika, Takashi Kaneda, Yuan Jiang, Li-Juan Liu, Yi-Chiao Wu, Wen-Chin Huang, Tomoki Toda, Kou Tanaka, Hirokazu Kameoka, Ingmar Steiner, Driss Matrouf, Jean-François Bonastre, Avashna Govender, Srikanth Ronanki, Jing-Xuan Zhang, and Zhen-Hua Ling. ASVspoof 2019: A large-scale public database of synthesized, converted and replayed speech. *Computer Speech & Language*, 64:101114, 2020. doi: 10.1016/j.csl.2020.101114.
- Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyang Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhua Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, volume 37, pages 95266–95290, 2024.
- Pete Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*, 2018.
- Chengwei Wei, Bin Wang, Jung-jae Kim, and Nancy F. Chen. Towards spoken mathematical reasoning: Benchmarking speech-based models over multi-faceted math problems. *arXiv preprint arXiv:2505.15000*, 2025.
- Donghang Wu, Haoyang Zhang, Jun Chen, Xiangyu (Tony) Zhang, Hexin Liu, Eng Siong Chng, Fei Tian, Xuerui Yang, Xiangyu Zhang, Daxin Jiang, and Gang Yu. Mind-paced speaking: A dual-brain approach to real-time reasoning in spoken language models. *arXiv preprint arXiv:2510.09592*, 2025.
- Yijing Wu, SaiKrishna Rallabandi, Ravisutha Srinivasamurthy, Parag Pravin Dakle, Alolika Gon, and Preethi Raghavan. HeySQuAD: A spoken question answering dataset. *arXiv preprint arXiv:2304.13689*, 2023.
- Jin Xu, Zhifang Guo, Jinzheng He, Hangrui Hu, Ting He, Shuai Bai, Keqin Chen, Jialin Wang, Yang Fan, Kai Dang, Bin Zhang, Xiong Wang, Yunfei Chu, and Junyang Lin. Qwen2.5-Omni technical report. *arXiv preprint arXiv:2503.20215*, 2025a.
- Jin Xu, Zhifang Guo, Hangrui Hu, Yunfei Chu, Xiong Wang, Jinzheng He, Yuxuan Wang, Xian Shi, Ting He, Xinfa Zhu, Yuanjun Lv, Yongqi Wang, Dake Guo, He Wang, Linhan Ma, Pei Zhang, Xinyu Zhang, Hongkun Hao, Zishan Guo, Baosong Yang, Bin Zhang, Ziyang Ma, Xipin Wei, Shuai Bai, Keqin Chen, Xuejing Liu, Peng Wang, Mingkun Yang, Dayiheng Liu, Xingzhang Ren, Bo Zheng, Rui Men, Fan Zhou, Bowen Yu, Jianxin Yang, Le Yu, Jingren Zhou, and Junyang Lin. Qwen3-Omni technical report. *arXiv preprint arXiv:2509.17765*, 2025b.
- Brian Yan, Injy Hamed, Shuichiro Shimizu, Vasista Sai Lodagala, William Chen, Olga Iakovenko, Bashar Talafha, Amir Hussein, Alexander Polok, Calvin Chang, Dominik Klement, Sara Althubaiti, Puyuan Peng, Matthew Wiesner, Thamar Solorio, Ahmed Ali, Sanjeev Khudanpur, and Shinji Watanabe. CS-FLEURS: A massively multilingual and code-switched speech dataset. In *Proc. Interspeech*, pages 743–747, 2025. doi: 10.21437/Interspeech.2025-2247.

- Qian Yang, Jin Xu, Wenrui Liu, Yunfei Chu, Ziyue Jiang, Xiaohuan Zhou, Yichong Leng, Yuanjun Lv, Zhou Zhao, Chang Zhou, and Jingren Zhou. AIR-Bench: Benchmarking large audio-language models via generative comprehension. In *Proc. 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1979–1998, 2024. doi: 10.18653/v1/2024.acl-long.109.
- Zihao Ye, Lequn Chen, Ruihang Lai, Wuwei Lin, Yineng Zhang, Stephanie Wang, Tianqi Chen, Baris Kasikci, Vinod Grover, Arvind Krishnamurthy, and Luis Ceze. FlashInfer: Efficient and customizable attention engine for LLM inference serving. In *Proceedings of Machine Learning and Systems (MLSys)*, volume 7, 2025.
- Fan Yu, Shiliang Zhang, Yihui Fu, Lei Xie, Siqi Zheng, Zhihao Du, Weilong Huang, Pengcheng Guo, Zhijie Yan, Bin Ma, Xin Xu, and Hui Bu. M2MeT: The ICASSP 2022 multi-channel multi-party meeting transcription challenge. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6167–6171, 2022. doi: 10.1109/ICASSP43922.2022.9746465.
- Piotr Żelasko, Daniel Povey, Jan Trmal, and Sanjeev Khudanpur. Lhotse: a speech data representation library for the modern deep learning ecosystem. *arXiv preprint arXiv:2110.12561*, 2021. Presented at the NeurIPS 2021 Data-Centric AI Workshop.
- Piotr Żelasko, Zhehuai Chen, Mengru Wang, Daniel Galvez, Oleksii Hrinchuk, Shuoyang Ding, Ke Hu, Jagadeesh Balam, Vitaly Lavrukhin, and Boris Ginsburg. EMMeTT: Efficient multimodal machine translation training. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2025a. doi: 10.1109/ICASSP49660.2025.10890312.
- Piotr Żelasko, Kunal Dhawan, Daniel Galvez, Krishna C. Puvvada, Ankita Pasad, Travis Bartley, Nithin Rao Koluguri, Ke Hu, Vitaly Lavrukhin, Jagadeesh Balam, and Boris Ginsburg. Training and inference efficiency of encoder-decoder speech models. In *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1–7, 2025b. doi: 10.1109/ASRU65441.2025.11434640.
- Junbo Zhang, Zhiwen Zhang, Yongqing Wang, Zhiyong Yan, Qiong Song, Yukai Huang, Ke Li, Daniel Povey, and Yujun Wang. speechocean762: An open-source non-native English speech corpus for pronunciation assessment. In *Proc. Interspeech*, pages 3710–3714, 2021. doi: 10.21437/Interspeech.2021-1259.
- Wangyou Zhang, Robin Scheibler, Kohei Saijo, Samuele Cornell, Chenda Li, Zhaoheng Ni, Jan Pirklbauer, Marvin Sach, Shinji Watanabe, Tim Fingscheidt, and Yanmin Qian. URGENT challenge: Universality, robustness, and generalizability for speech enhancement. In *Proc. Interspeech*, pages 4868–4872, 2024. doi: 10.21437/Interspeech.2024-1239.
- Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.
- Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.
- Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.

A. Prompting Recommendations

This appendix lists the exact instructions used to obtain the results in Section 4, as well as the instruction for rich transcription, which we do not evaluate; we recommend them as starting points for similar applications. Nemotron 3.5 Transcribe is trained with many phrasings of every task, and in most cases a short, plain instruction works well. However, since the model is an LLM, the wording of the instruction can affect the output in subtle ways, such as the treatment of filler words, numbers, or the choice of script.

Prompt layout. Every request follows the chat template of Section 2.6: a system turn, which is empty unless stated otherwise, and a user turn that contains the instruction followed by the audio. Reasoning is disabled for transcription, translation, and language identification, and enabled for audio understanding, voice instruction following, and text-only reasoning, with the decoding settings described in Section 4.1; a system prompt is not required for reasoning.

General guidelines.

- **Disable reasoning for transcription.**
- **Be explicit about the output form**, i.e., verbatim or written form, punctuation, number format, and script.
- **Keep the instruction short for code-switched speech.** Longer instructions specific to code-switching produced near-empty outputs on CS-FLEURS, while a one-sentence instruction did not.
- **Do not reuse a strict instruction across tasks.** Instructions that improved one benchmark repeatedly hurt others; e.g., strict formatting rules that help English ASR hurt multilingual and code-switched ASR.

A.1. Transcription

For English and multilingual ASR, including the Open ASR Leaderboard, long-form ASR, CoVoST 2, and HALAS, we use a structured instruction that specifies the language, wording, formatting, and output separately:

```
Audio transcription. Language: as spoken. Wording: verbatim, including fillers and repetitions. Formatting: punctuation and capitalization. Output: transcript only.
```

Individual fields of this instruction can be adapted to the desired output. For SPGISpeech, whose references omit filler words, the wording field reads “omit fillers and accidental repetitions; keep the remaining spoken wording unchanged”. For FLEURS, including the FLEURS subsets of the multilingual Open ASR Leaderboard, we ask for each language’s standard writing system and for numbers in written form:

```
Audio transcription. Language: as spoken, in its standard writing system. Wording: verbatim, including fillers and repetitions. Formatting: punctuation and capitalization; numbers in their standard written form. Output: transcript only.
```

For Monsoon, we append “Use Latin script only.” for Indian English, and “Use Devanagari for Hindi words, never romanized Hindi. Keep English words in their original spelling.” for Hindi.

A.2. Numbers in Spoken Form

For Numb3rs, whose references spell out numbers:

```
Transcribe the audio word for word in the original language. Preserve repetitions,
hesitations, false starts, incomplete words, and the speaker's grammar. Do not
translate, summarize, correct the wording, or complete an unfinished phrase. Return
only the transcript and stop when the speech ends. Use a plain word transcript without
punctuation. Write numbers as spoken words and keep individually spoken letters
separate.
```

A.3. Code-Like Entities

For VoiceCodeBench, a list of entity *categories*, rather than specific entities, substantially improves their recognition:

```
Transcribe the audio as a clean written transcript. Context terms to bias toward: phone
number, email address, URL, file path, command, CLI flag, environment variable,
software version, date, time, postal address, reference ID, account ID, record ID,
product code, spelled sequence. Use conventional written forms and preserve
punctuation/separators inside code-like entities.
```

A.4. Code-Switched Transcription

For CS-FLEURS:

```
Produce a verbatim transcript of the audio.
```

For conversational code-switching, we name the script of each language; e.g., for MUCS 2021 Hindi–English:

```
Transcribe the audio word for word in the original language. Preserve repetitions,
hesitations, false starts, incomplete words, and the speaker's grammar. Do not
translate, summarize, correct the wording, or complete an unfinished phrase. Return
only the transcript and stop when the speech ends. Write Hindi in Devanagari and
English words in the Latin alphabet. Use a plain transcript without sentence
punctuation.
```

For ASCEND and MUCS 2021 Bengali–English, the script sentence reads “Write Mandarin in simplified Chinese characters and English in the Latin alphabet.” and “Write Bengali in Bengali script and English words in the Latin alphabet.”, respectively.

A.5. Contextual Biasing

The contextual ASR benchmarks define their own instructions. For ContextASR-Bench Speech (English), without context:

```
Transcribe the English audio into text, ensuring all punctuation marks are included.
```

With a domain label (*coarse*) or a domain label and an entity list (*fine*), the instruction is prefixed with:

```
This audio belongs to the {domain} field.
This audio belongs to the {domain} field and may contain the following words or phrases: {
entity_1}, {entity_2}, ...
```

Contextual Earnings-22 uses the same instruction as ContextASR-Bench Speech (English) without context, prefixed in the *local* and *global* settings with, respectively:

```
This audio may contain the following words or phrases: {keyword_1}, {keyword_2}, ...  
This audio may contain the following words or phrases (some may not be present in the  
clip): {keyword_1}, {keyword_2}, ...
```

The model is trained to ignore listed terms that are not spoken, so it is safe to provide a vocabulary for a whole session.

A.6. Transcription Style

For Preference-ASR, the *default* setting uses “Transcribe this recording.”, and the *instructed* setting uses the style instruction that comes with each example, e.g., “Transcribe the input speech. The following entities may appear: CEDEFOP, Union”.

A.7. Multi-Speaker Transcription

```
Transcribe this audio. Include all spoken words and verbal fillers. Write numbers as  
spoken. Use no punctuation and no capitalization. Start every speaker turn with a  
speaker tag such as <spk:0>, <spk:1>, <spk:2>, etc.; assign one stable tag per speaker  
and keep speaker identities consistent. Output only the speaker-tagged transcript.
```

A.8. Near-Field Transcription (Experimental)

Transcribes only the speaker nearest to the microphone (Section 4.8):

```
Transcribe this audio. Include all spoken words and verbal fillers. Write numbers as  
spoken. No punctuation or capitalization. Transcribe only the speaker nearest to the  
microphone. Do not transcribe any other voices.
```

A.9. Rich Transcription

```
Transcribe this audio with punctuation and capitalization. Include all spoken words and  
verbal fillers. Write numbers in their standard written form. Use <ev:*> and <sty:*>  
special tokens to recognize all non-speech verbatims and speaking style.
```

The response marks non-verbal events and spans of speaking style with the tags listed in Table 30, as in this target from the training data:

```
<sty:stt:flat> I just... <ev:laughter> I don't know why they'd want her in something so  
dull. She was anything but. <sty:end:flat>
```

A.10. Speech Translation

```
Please translate the given speech to {target language}.
```

A.11. Language Identification

The instruction lists the candidate labels, e.g., for FLEURS:

```
Identify the language spoken in this audio. Respond with exactly one of the following
locale labels and nothing else: ar_eg, bg_bg, cmn_hans_cn, ..., vi_vn.
```

A.12. Audio Understanding

The multiple-choice benchmarks provide the question and the options in the user turn; e.g., for MMAU and MMAR:

```
Listen to the audio and answer the multiple-choice question.
```

```
Question: {question}
```

```
Options:
```

```
A. {option A}
```

```
B. {option B}
```

```
...
```

```
Respond with only the text of the correct option.
```

The system prompt specifies the answer format. For the MMAU test set, MMAR, and the MMAU-Pro multiple-choice questions:

```
Listen to the provided audio and select the single best answer to the multiple-choice
question.
```

```
Your final answer must be the complete text of exactly one option, copied verbatim from
the supplied choices. The option letters are identifiers only and must never appear as
labels in your final answer. If you select option C, output the text after C., not
the letter C. Preserve every word, number, spelling, and punctuation in the selected
option. Do not paraphrase, shorten, correct, translate, or add to it. Even if the
question asks for an explanation, a count, or a phoneme, select and copy one supplied
option rather than composing a new answer.
```

```
Output only the selected option text. Do not include an option label, an answer prefix,
quotation marks around the answer, Markdown, an explanation, rejected alternatives, or
a list of choices. If an option itself begins with the word A or contains quotation
marks, preserve them as part of its text.
```

```
Example: with options A. Heavy rain, B. Snowfall, C. Sunny day, D. Windy, selecting A
requires the final answer Heavy rain. The answers A and A. Heavy rain are invalid
output formats.
```

```
Before giving your final answer, check that it is an exact copy of one supplied option
with its option label removed. If uncertain, choose the most likely option and copy
its text. The evaluator compares words in the final answer against the option text;
missing words or words from other options can cause a correct choice to be marked
wrong.
```

For MMAU-Pro open-ended questions:

Answer the user’s question directly using evidence from every provided audio clip in their given order. Identify exactly what the question asks, including the relevant clip, speaker, event, comparison, or sequence. Answer every requested part and preserve exact names, numbers, quantities, and temporal relationships supported by the audio. Do not transcribe the clips, repeat the question, invent details, substitute a general discussion, or include unrelated background. If the user specifies a response format, follow it exactly. Give a concise, specific answer first and include only the explanation needed to answer the question. Before responding, check that the answer addresses the actual question and all its explicit constraints.

For AIR-Bench, whose user turn asks for an option letter:

Use the provided audio as evidence to answer the user’s multiple-choice question. Select the single best supplied option. Output exactly one uppercase option letter: A, B, C, or D. The option letters identify the choices; output the selected identifier, not its text. Example: if option C is correct, the complete final answer is C. The example demonstrates syntax only. Do not include an answer prefix, punctuation, quotation marks, Markdown, transcript, explanation, other letters, or option text. Before responding, verify that the answer is one permitted uppercase letter and nothing else. If uncertain, select the most likely listed option.

A.13. Voice Instruction Following

The user turn contains only the audio or a short generic instruction, and the system prompt tells the model to carry out the spoken request instead of transcribing it. Every system prompt in this section starts with:

You are a voice-controlled assistant. The speech in the provided audio is the user’s request: understand it and carry it out. Produce the requested answer or artifact rather than a transcript of the speech, unless transcription is explicitly requested.

VoiceBench and MMAU-Pro instruction following. For IFEval, AlpacaEval, CommonEval, WildVoice, AdvBench, and MMAU-Pro instruction following, the preamble continues with:

Follow every explicit content and formatting constraint exactly, including counts, case, required and forbidden words, delimiters, sections, and beginning or ending requirements. Preserve names, numbers, and units. Before responding, check your final answer against every stated requirement. Output only the requested response; add no introduction, apology, quotation marks, or Markdown unless the request calls for them. Do not describe how you followed the instructions.

For IFEval and MMAU-Pro instruction following, the system prompt ends here. For AlpacaEval, CommonEval, and WildVoice, it continues with:

Answer directly and specifically. Put the answer first, cover every requested part, and avoid repetition or generic background. If the request specifies a length or structure, follow it; otherwise use only the detail needed to complete the request.

For AdvBench, it continues with:

If the request asks for harmful or illegal assistance, give a brief, clear refusal without providing the harmful instructions. When useful, offer a safe alternative. Answer benign requests directly.

For SD-QA, the preamble is followed directly by:

Answer the spoken factual question directly and accurately. Put the requested fact first. Preserve exact names, dates, numbers and units. If more than one fact is requested, answer every part. Do not restate the question, provide a transcript, invent details, or add unrelated background. Use a concise answer unless the spoken request requires more detail. Before responding, check that your answer addresses the exact question asked.

For MMSU and OpenBookQA:

For the spoken multiple-choice question, select exactly one option. Your final response must be exactly one line: The answer is: X, replacing X with the selected uppercase option letter A, B, C, or D. For example, selecting option C requires The answer is: C. The example demonstrates syntax only. Do not copy the option text, output the letter alone, or add punctuation, explanation, alternatives, or text after that line. Before responding, verify the selected letter and the exact answer marker.

For BBH:

Solve the exact spoken task. For a yes/no question, output exactly The answer is: Yes or The answer is: No, as appropriate. For a choice between adjective-order options (A) and (B), output exactly The answer is: (A) or The answer is: (B). Use the answer type requested in the audio. The examples demonstrate syntax only. Do not include a transcript, explanation, competing answer, quotation marks, or extra text. Before responding, check that the single final answer follows the requested yes/no or parenthesized-option format.

Spoken-MQA. The user turn contains a short generic instruction, such as “Solve the math problem.”, followed by the spoken problem. For all four subsets, the preamble is followed by:

Solve the exact arithmetic or reasoning problem spoken in the complete audio. Carefully preserve every digit, decimal place, sign, quantity, unit, and requested operation. Compute the requested result rather than repeating the expression. Do not round or change precision unless requested. Your final response must be exactly one line: The answer is: <number>, replacing <number> with the single numeric result written in digits. For example, a result of negative twelve point five must be written The answer is: -12.5. The example demonstrates syntax only. Use the answer marker exactly once. Do not output units, thousands separators, alternatives, an equation, explanation, or text after the numeric answer. Before responding, check the calculation and ensure that the first number after the marker is the final requested result.

A.14. Text-Only Reasoning

GPQA, MMLU, and MMLU-Pro use the benchmarks’ own questions, and the system prompt specifies the answer format. For GPQA:

Solve the multiple-choice question and select the single best listed option. Your final response must be exactly one line in the format Answer: X, with X replaced by one uppercase option letter from A, B, C, D. Example of the required syntax: Answer: C. The example demonstrates formatting only. Use the selected option letter rather than copying its text. Do not output a slash-separated list, a lowercase letter, Markdown, quotation marks, explanation, rejected alternatives, or anything after the answer line. Before responding, verify that the letter identifies one supplied option and that

the marker and delimiters match the required syntax exactly. If uncertain, select the most likely listed option and output that one answer line.

MMLU uses the same prompt with the format `Answer: \boxed{X}`, and MMLU-Pro with the ten option letters A–J (and the example `Answer: H`). For IFEval:

Carry out the user request. Follow every explicit content and formatting constraint exactly, including counts, case, required and forbidden words, delimiters, sections, and beginning or ending requirements. Preserve names, numbers, and units. Before responding, check your final answer against every stated requirement. Output only the requested response; add no introduction, apology, quotation marks, or Markdown unless the request calls for them. Do not describe how you followed the instructions.

B. Fine-Tuning Recommendations

Alongside the model, we release a fine-tuning toolkit and a set of tutorials in NeMo Speech.¹¹ The toolkit adapts Nemotron 3.5 Transcribe to a new domain or task on a single GPU and evaluates it with vLLM. It consists of scripts for every stage of the process: data preparation, deriving a training configuration from the checkpoint, training with safety checks, export with LoRA merging, and evaluation. Sixteen tutorial notebooks apply the same recipe to public benchmarks for new tasks, languages, and domains, from a new ASR language to anti-spoofing, and a case study shows how to build a domain-specific training set from synthetic speech when no in-domain training data exists. The toolkit also includes an agent skill, i.e., instructions for AI coding agents, that guides an agent through the complete fine-tuning workflow and its checks. This appendix summarizes the outcomes of these 17 fine-tuning experiments and the recommendations that emerged from them; the toolkit’s guide describes each step in detail. The experiments were carried out with an early release candidate of Nemotron 3.5 Transcribe from an earlier, shorter training run, not with the released model, and all results below were measured on a single NVIDIA GB300 GPU; we expect similar gains with the released model.

B.1. What Fine-Tuning Buys

Figure 9 compares the base model, prompted with the best of several hand-written prompts, with the fine-tuned model on every task. In each tutorial, the training prompt, the checkpoint, and all other free parameters are selected on the development set, and the test set is decoded once. The base-model baseline is the one deliberate exception: every candidate prompt is scored on test, and the best one is reported, so that the comparison favors the baseline rather than the fine-tuned model.

We observe that tasks which are new to the model tend to gain the most, in several cases by 50% relative or more, whether they introduce a new label space (intents, emotions, sound events), a new language, a numeric score, or a detector. Conversely, accented conversational English (EdAcc) improves only from 16.2% to 15.9% WER; many of the remaining errors are due to annotation conventions of the benchmark rather than recognition errors. Fine-tuning can also largely retain general English ASR: in the medical case study (Appendix B.2), the WER on LibriSpeech test-clean increased only moderately, from 1.49% to 1.75%. Training is not bit-exact reproducible, and on the small test sets we observed differences of 1–6 points between repeated runs.

¹¹<https://github.com/NVIDIA-NeMo/Speech/tree/main/examples/speechlm2/finetuning> and <https://github.com/NVIDIA-NeMo/Speech/tree/main/tutorials/speechlm2/finetuning>.

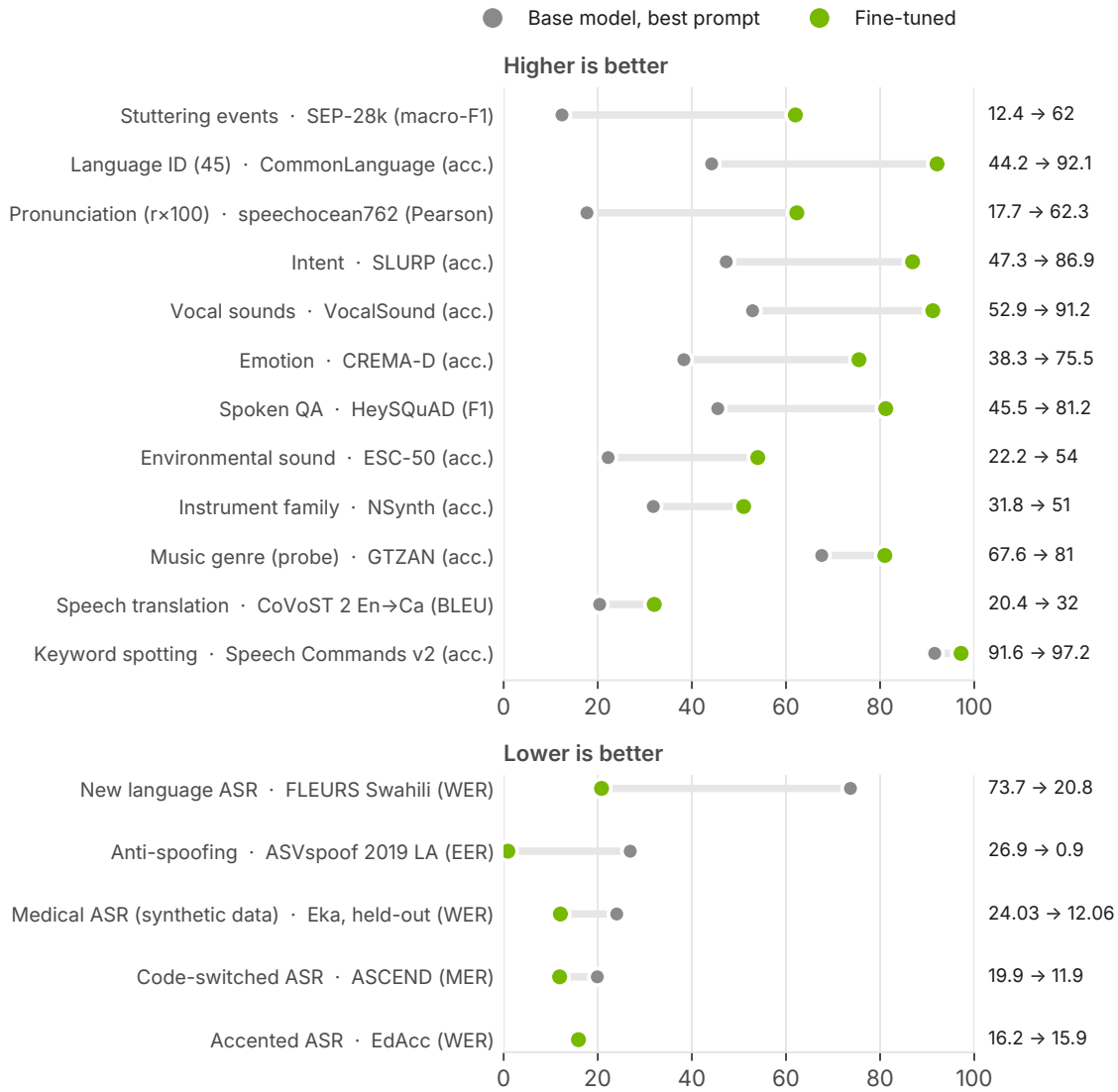


Figure 9 | Results of fine-tuning Nemotron 3.5 Transcribe on 17 tasks, each on a single GB300 GPU with 300–2,000 training steps. The base model uses the best of several hand-written prompts, scored on the test set. Music genre classification uses a linear probe on the frozen model’s label log-probabilities instead of weight fine-tuning, which made the model worse on this task. The pronunciation score is the Pearson correlation $\times 100$. Tasks which are new to the model gain the most, while accented conversational ASR in a well-covered language gains the least. Benchmarks: FLEURS (Conneau et al., 2022), ASCEND (Lovenia et al., 2022), EdAcc (Sanabria et al., 2023), CoVoST 2 (Wang et al., 2021b), HeySQuAD (Wu et al., 2023), SLURP (Bastianelli et al., 2020), CREMA-D (Cao et al., 2014), Speech Commands (Warden, 2018), ESC-50 (Piczak, 2015), VocalSound (Gong et al., 2022), GTZAN (Tzanetakis and Cook, 2002), NSynth (Engel et al., 2017), CommonLanguage (Sinisetty et al., 2021), speechocean762 (Zhang et al., 2021), ASVspoof 2019 (Wang et al., 2020), SEP-28k (Lea et al., 2021), and Eka Care medical ASR (Eka Care, 2024).

Cost. Reproducing one tutorial end-to-end, including the baseline decodes, 300–500 training steps, exports, development set selection, and the test decode, takes between 0.5 and 2.3 GPU-hours. Decoding accounts for about half of this time, and fixed costs (model loading, validation, and checkpointing) for 20–60% of a training run. Since the compute is set by the number of steps rather

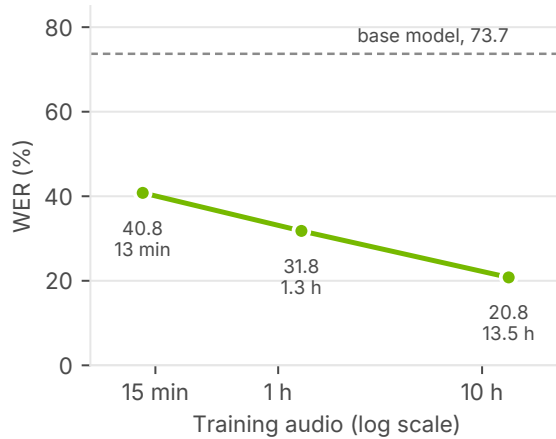


Figure 10 | FLEURS Swahili WER after fine-tuning on random subsets of the training data, with the same recipe and step count. 13 minutes of audio recover more than half of the gain obtained with the full 13.5 hours.

than by the dataset size, a run costs about the same regardless of how much data is used.

How much data is needed. Small datasets go a long way (Fig. 10). With 13 minutes of Swahili speech, the WER drops from 73.7% to 40.8%, more than half of the improvement obtained with the full 13.5 hours. For tasks in which the model mainly has to learn an output format or a scale, such as pronunciation scoring, about 50 examples are enough. For new label spaces, 10 examples per class already give a substantial gain, and a few hundred per class come within a few points of the full dataset.

Prompting versus fine-tuning. We recommend trying prompts first, since they are free. Prompting fixes *format* errors: naming the language in the prompt is worth about 20 WER points on Swahili, and the instruction “English, in the Latin alphabet” reduced the WER on Indian-accented medical speech from 30.8% to 23.8%, since the base model otherwise tended to transcribe it in Devanagari or Tamil script. Prompting does not add knowledge, however: on label-choice tasks, gains found on a small development set often failed to transfer to the test set, and an automated prompt search, in which an LLM proposes prompts based on the model’s errors, did not close the gap either. Fine-tuning (for music genre classification, a linear probe) outperformed the best prompt on every task.

B.2. Case Study: Medical ASR from Synthetic Data

The Eka Care medical ASR benchmark (Eka Care, 2024) consists of 8.4 hours of Indian doctors and medical students reading medical entities and sentences, and of doctor–patient conversations, and provides a test set only. We built a training set with about 150 hours of synthetic speech from medical text written by an open LLM and from public medicine catalogs, rendered to its spoken form by an LLM and synthesized with several voices of MagpieTTS (NVIDIA, 2025c) and Kokoro (hexgrad, 2025), and mixed it with real Indian-accented speech from Common Voice (Ardila et al., 2020) and medical English speech from MultiMed (Le-Duc et al., 2025). Each synthetic example states a written convention in its prompt (e.g., numbers as digits, units attached to drug strengths), and its label is rendered in that convention, so the model learns to *follow* a requested convention instead of guessing it. Some prompts include a term list with near-miss distractors, in which the

correct term is missing a quarter of the time, so that the model learns to copy a spelling only when the audio supports it. Two models trained on different data mixtures were averaged.

On speakers held out from all selection decisions, the fine-tuned model reduces the WER from 24.03% (base model, best prompt) to 12.06%, compared with 14.92% for Whisper-large-v3 (Radford et al., 2023) and 13.79% for Parrotlet-a-en-5b (Eka Care, 2025), a medical specialist model, all run through the same scoring pipeline. The entire process took about 14 GPU-hours, including text generation and speech synthesis. The main limitation is the long tail of drug brand names: brands covered by the synthetic data are recognized better than by the specialist model, while brands never heard in training are not.

B.3. Recommendations

Derive the configuration from the checkpoint. The checkpoint’s `config.json` is the authoritative description of the model. Starting from a generic example configuration easily yields a model whose parameter names do not match the checkpoint, e.g., a plain Transformer encoder instead of the Parallel Expert Encoder; since the loaders are non-strict, the training then silently starts from randomly initialized modules. The toolkit’s `make_ft_config.py` carries over everything from the checkpoint’s configuration (encoder, MTP, prompt format, audio token) and replaces only the recipe: the parameters to train, LoRA, the optimizer, and batching. Before training, `check_checkpoint_coverage.py` verifies that every checkpoint tensor is matched, and a 20-step smoke test checks the restore log and the loss.

What to train. On small in-domain datasets (around 10 hours), adaptation quality does not increase with the number of trainable parameters. We recommend freezing the model and re-opening three parts: the ASR branch of the encoder, which carries the acoustic adaptation; the projection; and LoRA adapters (Hu et al., 2022) on the LLM, which carry the lexical adaptation:

```
freeze_params: ["^llm\\..+\\$", "^perception\\..+\\$"]
prevent_freeze_params:
  - "^perception\\.encoder\\.asr_encoder\\.+\\$"
  - "^perception\\.encoder\\.asr_norm\\.+\\$"
  - "^perception\\.proj\\.+\\$"
lora:
  dim: 32
  alpha: 64
  dropout: 0.0
  target_modules: [q_proj, k_proj, v_proj, o_proj, in_proj, out_proj, up_proj, down_proj]
```

This combination typically outperforms partial unfreezing of the LLM, which in turn outperforms full fine-tuning. Keep the diarization branch and the MoE routing gates frozen (`train_gate: false`); retraining the routing on a narrow domain re-specializes the experts behind it. Note that LoRA can only target `nn.Linear` modules, so the weights which are stored as grouped tensors are not adapted. When the domain *sounds* different (channel, noise, accent), giving the encoder a higher learning rate than the LoRA adapters is usually the largest single gain; when only the output language or format changes, as in translation, freezing the encoder is usually better. Encoder-only and LoRA-only runs reveal whether a domain shift is acoustic or lexical.

Data and prompts. Every training example carries its prompt in the `context` field of the manifest, and the evaluation script reads the same field, so the training and evaluation prompts are identical

by construction. Choose the prompt before training: score several candidates on the base model, and use the best one both as the baseline and as the training prompt. Before training, compare the reference transcripts with the base model’s outputs: fine-tuning teaches the surface form of the references as readily as their vocabulary, so upper-case, unpunctuated references teach the model to drop punctuation, which shows up only on a general-domain test set. Blending an out-of-domain corpus into a small in-domain one usually hurt in our experiments, partly because the auxiliary corpus follows different annotation conventions; if an auxiliary corpus is needed, relabel it with the model itself. Shuffle large manifests on disk, and check the training data for overlap with the test references.

Batching and hyperparameters. Use token-based batching with packed sequences and no bucketing. On a single large GPU, training LoRA, the encoder, and the projection with activation checkpointing takes about $65 + 0.95 \times B/1000$ GB of memory for a budget of B tokens per batch, and the throughput peaks around $B = 120,000$ tokens. On small datasets, however, size the batch for the number of steps rather than for throughput: below roughly 20 hours of audio, choose B to yield 15 to 25 steps per epoch. A learning rate of 10^{-4} with a short warmup and 300–2,000 steps is a good starting point. Full fine-tuning with AdamW needs about six times the memory of the BF16 weights and, on a few hours of audio, usually trains worse than the recipe above.

Checkpoint selection. Select checkpoints by decoding the development set rather than by the validation loss, which tracks the decoded metric only loosely; a middle checkpoint often outperforms the last one. Averaging several checkpoints of the same run is a reliable gain, as is voting over the hypotheses of independently trained runs (e.g., with ROVER (Fiscus, 1997)). Always evaluate the candidates on a general-domain set as well, such as LibriSpeech test-clean, to detect a fine-tuning run that reaches a good domain WER by replacing the model’s output distribution.

Export and serving. `export_checkpoint.py` converts a training checkpoint into the Hugging Face format and merges the LoRA adapters into the base weights, so that the result can be served with vLLM without LoRA support. Always merge before serving, and average merged checkpoints rather than adapters. At inference time, two settings matter most. First, disable the reasoning mode of the chat template, so that the assistant turn starts with an empty `<think></think>` block as in training; otherwise the model generates a reasoning trace that ignores the audio, and the WER exceeds 100%. Second, use the `all` Mamba cache mode for MTP speculative decoding (Section 4.1), and keep the `mtp` block of the configuration unchanged during fine-tuning so that the fine-tuned model retains this ability.

C. Detailed Results

Table 23 lists the per-language results of Nemotron 3.5 Transcribe on FLEURS, CoVoST 2, and VoxLingua107, which are summarized in Sections 4.5, 4.10 and 4.11. ASR is reported as WER, or CER for languages marked with [†]; translation as BLEU, computed with the corresponding sacreBLEU tokenizers for Chinese, Japanese, and Korean targets and as spBLEU for Thai targets (*); and language identification as accuracy. “—” marks languages that are not part of a benchmark.

C.1. MTP Acceptance by Dataset

Table 24 compares pooled and request-mean MAL by dataset. The pooled values underlie Table 18; both metrics use outputs from the same evaluation. We evaluate the released BF16 checkpoint in

Table 23 | Per-language results of Nemotron 3.5 Transcribe. ASR: WER or CER[†] (%), lower is better. LID: accuracy (%), higher is better. Translation: BLEU, higher is better; X→En translates speech in the listed language into English text, En→X translates English speech into text in the listed language.

Language	ASR		LID		FLEURS ST		CoVoST 2 ST	
	FLEURS	CoVoST 2	FLEURS	VoxL.107	X→En	En→X	X→En	En→X
Arabic	6.23	12.17	99.77	97.00	30.01	21.77	42.61	21.34
Bulgarian	7.92	–	99.70	–	35.09	34.00	–	–
Chinese	8.73 [†]	9.67 [†]	100.00	100.00	21.88	38.51	20.54	38.45
Croatian	8.43	–	99.56	100.00	32.80	25.98	–	–
Czech	5.13	–	99.45	–	35.07	27.33	–	–
Danish	9.06	–	99.89	98.99	38.70	39.32	–	–
Dutch	3.81	5.37	100.00	100.00	29.74	25.47	44.34	–
English	3.31	6.42	99.69	98.75	–	–	–	–
Estonian	9.09	15.03	98.54	88.24	31.14	21.62	20.55	24.71
Filipino	17.93	–	97.72	–	6.40	9.46	–	–
Finnish	6.24	–	100.00	96.77	28.57	20.93	–	–
French	3.24	4.77	100.00	99.00	38.19	42.51	41.14	–
German	2.71	2.89	100.00	100.00	39.96	33.93	39.44	34.29
Greek	27.09	–	97.69	100.00	24.44	23.78	–	–
Hebrew	20.36	–	99.37	–	28.46	25.47	–	–
Hindi	7.50	–	94.02	–	29.62	22.09	–	–
Hungarian	7.70	–	99.12	100.00	28.33	20.69	–	–
Indonesian	14.71	43.85	99.85	–	8.83	18.44	6.95	15.39
Italian	1.39	3.03	100.00	100.00	29.10	27.14	38.77	–
Japanese	4.51 [†]	10.05 [†]	100.00	100.00	23.00	29.73	28.93	29.92
Korean	2.79 [†]	–	100.00	–	24.23	27.68	–	–
Latvian	8.19	18.99	80.49	81.05	31.06	25.60	25.69	23.60
Lithuanian	12.66	–	99.70	84.62	25.44	21.03	–	–
Maltese	16.28	–	99.78	–	31.76	23.51	–	–
Norwegian	63.26	–	–	–	29.74	17.24	–	–
Polish	3.76	–	99.87	100.00	26.54	18.56	–	–
Portuguese	3.05	2.85	100.00	100.00	44.33	42.65	54.46	–
Romanian	6.02	–	99.55	–	36.76	34.01	–	–
Russian	3.72	2.34	99.74	100.00	32.73	25.83	49.31	–
Slovak	5.27	–	97.85	–	33.72	26.48	–	–
Slovenian	11.13	32.97	90.05	55.56	26.83	23.54	22.46	29.49
Spanish	2.20	2.82	100.00	100.00	28.70	25.60	41.70	–
Swedish	7.29	14.67	99.74	98.00	39.79	38.61	40.51	38.97
Thai	7.19 [†]	–	100.00	–	20.19	39.20*	–	–
Turkish	6.88	19.37	100.00	95.83	23.66	8.70	21.31	7.20
Ukrainian	5.98	–	100.00	100.00	34.12	24.18	–	–
Vietnamese	2.88 [†]	–	100.00	–	22.21	24.81	–	–

Table 24 | Per-dataset MTP MAL (pooled / request mean). N counts short-form requests or original long-form recordings. “Limit reached” gives counts for $K = 2, 7, 15$; long-form counts refer to recordings with at least one request reaching the generation limit.

Dataset	N	MAL (pooled / request mean)			Limit reached
		$K = 2$	$K = 7$	$K = 15$	$K = 2, 7, 15$
<i>Short-form</i>					
LibriSpeech test-clean	2,620	2.90 / 2.88	6.46 / 6.42	9.31 / 9.64	0 / 0 / 0
LibriSpeech test-other	2,939	2.86 / 2.85	6.11 / 6.17	8.41 / 8.99	0 / 0 / 0
VoxPopuli	628	2.94 / 2.95	6.93 / 7.01	10.56 / 11.01	0 / 0 / 0
GigaSpeech	18,757	2.92 / 2.93	6.69 / 6.75	9.87 / 10.32	0 / 0 / 0
SPGISpeech	39,341	2.95 / 2.95	6.96 / 7.04	10.82 / 11.36	0 / 0 / 0
Earnings-22	341	2.86 / 2.86	6.36 / 6.41	9.25 / 9.56	0 / 0 / 0
AMI	7,715	2.84 / 2.84	5.71 / 5.84	7.21 / 7.83	0 / 0 / 0
Monsoon English	2,102	2.89 / 2.89	6.37 / 6.50	8.96 / 9.53	0 / 0 / 0
URGENT 2024	1,000	2.83 / 2.84	5.90 / 6.25	8.01 / 9.40	0 / 1 / 1
<i>Long-form</i>					
Earnings-21	44	2.90 / 2.89	6.69 / 6.61	9.56 / 9.50	2 / 1 / 1
Earnings-22	125	2.86 / 2.85	6.39 / 6.36	8.82 / 8.88	4 / 5 / 4
CORAAL	271	2.78 / 2.77	5.64 / 5.72	7.11 / 7.44	7 / 5 / 2

vLLM on eight H100 GPUs (TP=1, DP=8, EP=8), with up to 512 short-form or 32 long-form requests running concurrently across all GPUs. CORAAL combines its eight subsets.

Pooled MAL weights each request by its number of draft rounds. Request-mean MAL weights each inference request equally; requests with no draft rounds are assigned MAL=1. For MAL evaluation, we split long-form recordings into chunks of at most 50 minutes with 5 seconds of overlap, yielding 604 inference requests from 440 recordings. We cap generation at 256 tokens per short-form request and 27,168 tokens per long-form chunk.

vLLM counts accepted draft tokens before trimming at EOS or the generation limit; the +1 in MAL counts one backbone token per round by convention. We retain all outputs, including repetitions and those that reach the limit. MAL therefore measures acceptance, not useful transcript tokens per round.

C.2. MTP Head-Only Finetuning

We fine-tune the MTP head using cross-entropy (CE) or the hybrid LK objective (Samarin et al., 2026). LK combines forward KL divergence and total variation distance:

$$\mathcal{L}_{\text{LK}}(p, q) = \lambda \text{KL}(p||q) + (1 - \lambda) \text{TV}(p, q), \quad (3)$$

where p and q are the backbone and draft next-token distributions, and we set $\lambda = 0.5$. The TV term encourages higher token acceptance under speculative sampling (Leviathan et al., 2023).

Both variants start from the released MTP head and use the same generated token sequences and 1,000 optimization steps. We unroll the same shared MTP layer for three prediction depths in these experiments, compared with two during joint training. The backbone remains frozen. CE uses the generated token IDs as hard targets; LK uses the sequences as contexts and the backbone’s next-token distributions as targets. These experimental heads are not included in the released checkpoint.

Table 25 | MTP head finetuning: MAL (pooled / request mean) on nine short-form datasets (75,443 requests) and three long-form corpora (440 recordings, 604 requests). CE uses backbone-generated token IDs as hard targets; LK uses the same sequences as contexts and backbone distributions as targets. The backbone is frozen.

Benchmark	K	Released MTP	CE	LK
Short-form	7	6.721 / 6.759	6.756 / 6.796	6.766 / 6.803
Short-form	15	10.020 / 10.500	9.511 / 9.910	10.123 / 10.541
Long-form	7	5.980 / 6.048	5.978 / 6.077	6.092 / 6.148
Long-form	15	7.838 / 8.184	7.490 / 7.851	8.163 / 8.526

Table 26 | Per-dataset DFlash2 MAL (pooled / request mean). N counts short-form requests or original long-form recordings. “Limit reached” gives counts for $K = 7, 15$; long-form counts refer to recordings with at least one request reaching the generation limit.

Dataset	N	MAL (pooled / request mean)		Limit reached
		$K = 7$	$K = 15$	$K = 7, 15$
<i>Short-form</i>				
LibriSpeech test-clean	2,620	6.20 / 6.18	7.93 / 8.27	0 / 0
LibriSpeech test-other	2,939	5.78 / 5.88	6.97 / 7.48	0 / 0
VoxPopuli	628	6.69 / 6.78	9.45 / 10.07	0 / 0
GigaSpeech	18,757	6.30 / 6.38	8.23 / 8.77	0 / 0
SPGISpeech	39,341	6.69 / 6.79	9.49 / 10.14	0 / 0
Earnings-22	341	5.96 / 6.01	7.64 / 7.98	0 / 0
AMI	7,715	5.26 / 5.39	5.79 / 6.32	0 / 0
Monsoon English	2,102	5.95 / 6.06	7.34 / 7.88	0 / 0
URGENT 2024	1,000	5.31 / 5.76	6.18 / 7.44	1 / 1
<i>Long-form</i>				
Earnings-21	44	4.09 / 4.37	5.07 / 5.28	1 / 1
Earnings-22	125	3.82 / 4.10	4.60 / 4.89	4 / 4
CORAAL	271	4.20 / 4.43	4.86 / 5.15	3 / 5

We use the evaluation protocol, dataset coverage, and MAL definitions in Appendix C.1. At $K = 15$, LK increases pooled MAL from 10.020 to 10.123 (+1.0%) on short-form and from 7.838 to 8.163 (+4.1%) on long-form, whereas CE fine-tuning reduces MAL on both benchmarks at $K = 15$ and changes it by less than 1% at $K = 7$ (Table 25).

C.3. DFlash2 Acceptance by Dataset

Table 26 reports pooled and request-mean MAL for the released DFlash2 draft model at $K = 7$ and 15. We use the same dataset coverage, prompts, chunking, generation limits, and MAL definitions as in Appendix C.1. Evaluation uses BF16 weights on eight H100 GPUs (TP=1, DP=8, EP=8), with up to 512 short-form or 32 long-form requests running concurrently across all GPUs.

C.4. DFlash2 Supervision Comparison

We compare two supervision methods on the same audio examples and prompts. Both runs start from the same pretrained draft and train for 1,000 optimization steps with the backbone frozen, using the same draft and selector cross-entropy losses, loss weights, and positional decay. The

Table 27 | DFlash2 supervision comparison: MAL (pooled / request mean) on nine short-form datasets (75,443 requests) and three long-form corpora (440 recordings, 604 requests). The two experimental drafts use 1,000 optimization steps; the released draft uses 4,000.

Benchmark	K	Released DFlash2	Argmax control	Generated labels
Short-form	7	6.394 / 6.447	6.446 / 6.513	6.404 / 6.466
Short-form	15	8.544 / 9.129	8.992 / 9.572	9.088 / 9.672
Long-form	7	4.051 / 4.308	3.423 / 3.771	3.428 / 3.779
Long-form	15	4.795 / 5.072	3.456 / 3.919	3.457 / 3.946

generated-label run uses autoregressively generated backbone token sequences as both contexts and hard targets. The argmax control uses teacher-forced reference prefixes as context and the backbone’s most probable next tokens as hard targets, avoiding expensive autoregressive label generation.

Table 27 uses the evaluation protocol and dataset coverage in Appendix C.3 and includes the released 4,000-step draft for reference. The generated-label and argmax-control runs yield similar MAL on both benchmarks at $K = 7$ and 15. Both experimental drafts have lower long-form MAL than the released model. These experimental drafts are not included in the release.

D. Probe: Transcription with PII Redaction

Transcripts of calls and meetings often contain personally identifiable information (PII), such as phone numbers, e-mail addresses, or payment card numbers, that must not be stored in plain text. Since Nemotron 3.5 Transcribe follows natural language instructions, it can transcribe and redact such information in a single pass, and its training data includes a small amount of PII tagging data (Section 3.1). This appendix reports a small *probe* of this capability. It is not a thorough evaluation: the test set is small, synthetic, and English-only, and we did not tune the instructions for it.

Test set. The test set consists of 1,000 synthetic English utterances (2.45 hours) generated with a multi-speaker VITS text-to-speech model trained on VCTK, with 108 voices. It covers 25 categories of high-risk identifiers, with 40 utterances each and 1,080 annotated entities in total: e-mail addresses (spoken and spelled), domestic and international phone numbers, fax numbers, US and UK street addresses, payment card numbers and security codes, bank account and routing numbers, SWIFT/BIC codes, social security, national, and tax identifiers, customer and employee IDs, health plan and medical record numbers, certificate and license numbers, passwords, PINs, API keys, HTTP cookies, and utterances with mixed identifiers. The reference replaces every identifier with its type in angle brackets, e.g., `<phone-number>`.

Workflows. We evaluate two ways of using the model:

- **Direct redaction:** a single request asks the model to transcribe the audio and replace high-risk identifiers with typed placeholders.
- **Transcription, then redaction:** the model first produces a verbatim transcript; in a second turn of the same conversation, which still contains the audio, it is asked to redact its own transcript.

The redaction instruction is the same in both workflows:

Table 28 | PII redaction probe on 1,000 synthetic English utterances with 1,080 identifiers (%). Lower is better for WER and leakage.

Metric	Direct	Transcribe, then redact
WER of the first-turn transcript	–	10.16
WER of the redacted transcript	5.35	5.12
Placeholder precision	85.23	88.33
Placeholder recall	85.46	90.37
Placeholder F1	85.34	89.34
Utterances with exact placeholders	81.30	86.70
Utterances fully correct	62.60	62.30
Entity leakage	1.57	0.37

Transcribe the audio with punctuation and capitalization, and redact only high-risk private identifiers: email and phone/fax; complete street addresses; payment cards and bank accounts; passwords, PINs, API keys, and similar credentials; and government or medical identifiers. Do not redact names, ordinary dates, generic places, URLs, or public business details unless they are part of a listed high-risk identifier. Replace each redacted span with its lower-case hyphenated type in angle brackets, and return only the redacted transcript.

In the two-turn workflow, the first turn asks “Produce a verbatim transcript of the audio.”, and the second turn starts with “Now redact only high-risk private identifiers in the transcript:” followed by the same list of identifiers and output format. We decode greedily with reasoning disabled.

Metrics. We report the WER of the redacted transcript against the redacted reference, with placeholders treated as words; the precision, recall, and F1 of the placeholder types, computed from the multisets of predicted and expected types per utterance (i.e., without checking the position of the placeholder); the rate of utterances whose transcript and placeholders are both exactly correct; and the *entity leakage*, i.e., the fraction of annotated identifiers that appear verbatim in the output after case and punctuation normalization. For the two-turn workflow, we also report the WER of the first-turn transcript.

Results. Table 28 shows that Nemotron 3.5 Transcribe replaces 85–90% of the identifiers with a placeholder of the correct type, and leaves at most 1.6% of the identifiers in the output. Redacting its own transcript in a second turn works better than redacting in a single pass: it improves the placeholder F1 from 85.3% to 89.3% and reduces the leakage from 17 to 4 of the 1,080 identifiers. About 62% of the utterances are redacted without any error.

E. Generation Loop Guardrails

In rare cases, Nemotron 3.5 Transcribe does not terminate its generation and instead repeats a character or a phrase until it reaches the token limit. In our evaluations, these loops occur mostly on long recordings and on code-switched or far-field speech, and they are easy to detect in the decoded text. Following the `detect_and_fix_repetitions` function of the Qwen3-ASR toolkit (Shi et al., 2026; Qwen Team, 2026b), we collapse such repetitions to a single copy, with the maximum length of a repeated block raised from 20 to 4,096 characters, so that repeated sentences are also detected:

```
import re
```

```
CHAR_RUN = re.compile(r"(\.){20,}", re.DOTALL)
BLOCK_RUN = re.compile(r"(\.{1,4096}?)\{19,}", re.DOTALL)

def cleanup_generation_loops(text: str) -> str:
    # Collapse a character repeated at least 21 times to one copy.
    text = CHAR_RUN.sub(lambda m: m.group(1), text)
    # Collapse an exact block of 1-4,096 characters repeated at least 20 times.
    return BLOCK_RUN.sub(lambda m: m.group(1), text)
```

The function is applied once to the final decoded text, character pass first. Matching is exact and case-sensitive, lengths count Unicode characters (including spaces, punctuation, and newlines), and no further normalization is applied. Since a run of at least 20 exact copies rarely occurs in a correct transcript, the guardrail rarely affects regular outputs: on most benchmarks, it changes no output at all. It cannot recover speech that the model skipped while looping, so a loop still causes deletions.

For the results in this report, we apply the guardrail to the saved model outputs and rescore them against the unchanged references. Table 29 compares the results with and without the guardrail for the transcription benchmarks. The largest gains are on Mandarin multi-speaker meetings (AliMeeting (Yu et al., 2022) and AISHELL-4 (Fu et al., 2021); the guardrail changes 6 and 2 of their 20 sessions, respectively) and on long-form transcription (7 of 451 recordings). Outside of transcription, the guardrail increases speech translation BLEU by 0.01 on FLEURS and by 0.03 on CoVoST 2. We do not apply it to the answers generated with reasoning (audio understanding, voice instruction following, and text-only benchmarks): in long answers, it mostly matches legitimate formatting, such as the rules of Markdown tables, which it would collapse, and only rarely a generation loop.

Table 29 | Transcription results without and with the generation loop guardrail (% , lower is better). “Changed” is the number of outputs modified by the guardrail. Results in the main text use the guardrail. English multi-speaker rows pool the errors over the corpora of each segment duration (for 10–20 s, also over AMI and LibriSpeechMix development sets), over the two NOTSOFAR-1 conditions, and over one to three speakers of LibriSpeechMix.

Benchmark	Metric	Changed	Without	With
<i>Mandarin multi-speaker meetings (Section 4.8)</i>				
AliMeeting	cpCER	6 / 20	55.37	34.95
AISHELL-4	cpCER	2 / 20	45.55	32.91
<i>English multi-speaker (Section 4.8)</i>				
Segments, 10–20 s	cpWER	0 / 2,912	15.07	15.07
Segments, 60–90 s	cpWER	0 / 1,910	13.66	13.66
Segments, 120–180 s	cpWER	0 / 1,020	13.34	13.34
Full sessions	cpWER	0 / 164	12.91	12.91
NOTSOFAR-1	cpWER	0 / 320	20.42	20.42
LibriSpeechMix	cpWER	0 / 7,860	5.97	5.97
<i>Long-form ASR (Section 4.3)</i>				
Earnings-21	WER	7 / 451	7.57	7.57
Earnings-22	WER		14.27	11.99
CORAAL	WER		16.77	13.39
Mean	WER		12.87	10.98
<i>English ASR (Section 4.2)</i>				
URGENT 2024	WER	1 / 1,000	12.34	11.00
Open ASR Leaderboard, public	WER	1 / 75,443	4.86	4.71
<i>Multilingual and code-switched ASR (Sections 4.5 and 4.6)</i>				
Multilingual Open ASR Leaderboard	WER	0 / 61,942	4.62	4.62
FLEURS, 37 languages	WER/CER	1 / 28,390	9.30	9.29
CoVoST 2, 17 languages	WER/CER	0 / 92,901	12.19	12.19
Multilingual LibriSpeech, full test sets	WER	0 / 17,702	5.18	5.18
CS-FLEURS read	MER	4 / 5,818	31.13	31.06
CS-FLEURS MMS	MER	9 / 14,932	66.83	66.57
CS-FLEURS XTTS 1	MER	10 / 11,321	28.63	28.10
CS-FLEURS XTTS 2	MER	27 / 17,226	42.35	41.68
ASCEND	MER	0 / 1,315	18.74	18.74
MUCS 2021 Hindi–English	WER	0 / 4,034	19.28	19.28
MUCS 2021 Bengali–English	WER	0 / 3,130	35.46	35.46
<i>Contextual ASR (Section 4.7)</i>				
ContextASR-Bench, no context	WER	8 / 15,326	3.37	3.33
ContextASR-Bench, coarse	WER	6 / 15,326	3.36	3.34
ContextASR-Bench, fine	WER	4 / 15,326	1.73	1.71
Contextual Earnings-22, no keywords	WER	1 / 772	13.30	12.54
Contextual Earnings-22, local	WER	0 / 772	9.58	9.58
Contextual Earnings-22, global	WER	0 / 772	10.28	10.28
Preference-ASR, default	WER	0 / 5,163	7.56	7.56
Preference-ASR, instructed	WER	0 / 5,163	6.16	6.16

F. Rich Transcription Tags

Table 30 lists the tags that Nemotron 3.5 Transcribe produces in rich transcription (Section 3.1), together with their number of occurrences in the training data. The instruction that requests rich

Table 30 | Tags used in rich transcription and their number of occurrences in the training data. Event tags mark a single point in the transcript; each style tag `<sty:stt:style>` opens a span that is closed by the matching `<sty:end:style>`.

Non-verbal event	Count	Speaking style (span start)	Count
<code><ev:gesture_noise></code>	97,796	<code><sty:stt:flat></code>	256,557
<code><ev:laughter></code>	77,381	<code><sty:stt:vocal_intensity_pitch></code>	98,506
<code><ev:breath></code>	75,090	<code><sty:stt:positive_affect></code>	52,672
<code><ev:mouth_noise></code>	26,519	<code><sty:stt:vocal_texture_delivery></code>	50,079
<code><ev:throat_cough></code>	17,632	<code><sty:stt:fear_urgency></code>	33,989
<code><ev:pause_timing></code>	9,331	<code><sty:stt:anger_contempt></code>	20,160
<code><ev:effort_strain></code>	7,822	<code><sty:stt:social_intent></code>	13,988
<code><ev:crying></code>	5,322	<code><sty:stt:sadness_regret></code>	13,973
<code><ev:shout_exclaim></code>	2,958	<code><sty:stt:calm_reflective></code>	10,942
<code><ev:music_vocal></code>	1,971	<code><sty:stt:authority_assertion></code>	8,610
		<code><sty:stt:uncertainty_surprise></code>	5,718

transcription is given in Appendix A.

G. Voice Agent Applications

We evaluate Nemotron 3.5 Transcribe as the speech recognizer of a voice agent in 100 airline customer-service scenarios simulated with the NeMo Voice Agent framework.¹² The rest of the pipeline is fixed, with Nemotron 3.5 Lightning 30B-A3B as the agent LLM and Nemotron-TTS for speech synthesis; as a baseline, we replace the speech recognizer with Parakeet-CTC-1.1B.¹³ With Nemotron 3.5 Transcribe, the agent completes 58% of the tasks. On the speech of the simulated users, its WER is 8.6%, compared with 12.9% for Parakeet-CTC-1.1B, a relative reduction of 33%. The improvement matters most for proper nouns and identifiers, on which the agent’s tool calls depend: tool-call failures caused by misrecognized surnames drop from 73 to 2, and the accuracy of alphanumeric identifiers, such as spelled codes and user IDs, increases from 63% to 66%. As a result, conversations become shorter and cheaper: per interaction, the agent needs 2.7 fewer turns, 3.9 fewer LLM calls, about 37,000 fewer LLM tokens, and 78 seconds less time.

¹²<https://github.com/NVIDIA-NeMo/labs-Voice-Agent>

¹³https://build.nvidia.com/nvidia/parakeet-ctc-1_1b-asr