NumPipe: Automatic Generation, Vocalization and Evaluation of Spoken Numbers

Anonymous Authors
anonymous@review

September 26, 2026

Abstract

In the last year AI agents, meeting summarizers and similar assistants built on automatic speech recognition (ASR) have spread quickly. Their output is used directly and often unchecked, so a wrong number in a transcript is a serious error. Word error rate (WER), the fraction of words an ASR model gets wrong, reflects this problem poorly, because numbers are rare in text. In the standard test sets we count only about 13 numbers per 1000 words. Instead, we measured the pure number transcription accuracy. By this measure, two widely used open ASR models with low general error rates get only 80–88% of hard dictated numbers right. To measure and fix this at scale, we present NumPipe, an automatic pipeline with no human in the loop. It generates numbers with known values and vocalizes them with preset and zero-shot voices. Independent models then judge every clip, and the model under test is scored on the value it recognized. Because the same stages also produce training data, the pipeline can repair the weakness it measures. A 2000-step finetune on this data raises two models of different architectures from 80–88% to 98.8–99.8% on hard dictated numbers. On the public Numb3rs set they reach 97.6–98.8%. The price is 0.7 to 1.2 points of general WER. NumPipe is released as a set of NeMo Curator stages, and its test sets form a benchmark for numeric accuracy.

1 Introduction

Automatic speech recognition (ASR) has moved from transcripts that people read to transcripts that software acts on. It now sits at the front of assistants such as meeting summarizers and AI agents, whose adoption has surged over the last year [1, 22]. These systems consume the transcript without a human reading it, so recognition errors pass straight into the summary or the action [3]. A wrong amount, date, account number or percentage is therefore a serious error, and it passes silently.

The standard metric of the field is word error rate (WER), the share of transcribed words that are wrong. It does not show this problem. By this metric the two open models studied in this paper [6, 18, 28] look excellent. They reach 1.6 to 1.7 percent on LibriSpeech test-clean, a set of read audiobooks [20]. Averaged over the eight English sets of the Open ASR Leaderboard [25], the standard public comparison of English ASR models, they reach 5.5 to 6.1 percent. Yet these figures say little about numbers, since the test sets themselves contain few of them. In the reference transcripts of those eight sets, 1.69 million words in total, tokens that contain a digit amount to about 13 per 1000 words. The range runs from 7 in LibriSpeech to 26 in the company earnings calls of Earnings22 [23]. We counted the tokens after the Whisper text normalizer [21], which writes

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

spelled-out numbers as digits. Numbers therefore carry almost no weight in the aggregate error rate. A long number also counts as a single word error when one of its many words is wrong. Hearing "thirteen" as "thirty" in "thirteen million twenty five thousand four hundred two" costs one word in eight, but the number is completely wrong. Unlike an ordinary word, a wrong number cannot be recovered from context.

We therefore measure numbers on their own terms, with number transcription accuracy on dictated numbers, that is, numbers read out on their own as in an account number. A number counts as correct only if its recognized value exactly matches the spoken value. The comparison works on the value, so "four twenty" and "420" count as the same answer, and a number is either right or wrong as a whole. By this measure, the two widely used open models we study get only 80 to 88 percent of hard dictated numbers correct, despite their low general error rates. The strongest recent models, by contrast, exceed 99 percent.

NumPipe is an automatic pipeline for this problem, released as NeMo Curator stages that work out of the box [17]. A user plugs in a model and receives both a number-specific evaluation and verified training data, without any manual labeling. Finetuning on that data fixes most of the problem for dictated numbers. Its stages are modular, so a user can choose the judge and the TTS system and adjust the generated numbers to their own needs.

The paper makes three contributions, described below in order of importance.

The first contribution is the pipeline itself. It has four stages. Generation produces numbers of controlled kind, length, sign and reading style. Each comes with a spoken reading, and the scorer's parser converts the reading back to the original value. Vocalization turns the readings into speech with preset voices and with zero-shot text-to-speech (TTS), which synthesizes a new voice from a few seconds of reference audio. Judging transcribes every clip with independent models that are not told the target, and it discards the clips whose value they do not recover. Evaluation converts each hypothesis to a value and compares it with the intended one.

The second contribution is a proof of concept that the pipeline's output also serves as training data. We finetuned two models with different architectures on it. Canary-Qwen-2.5B [6, 18] has a frozen large language model (LLM) as its decoder, fed by an audio encoder through trained adapters. parakeet-tdt-0.6b-v2 [28] is a CTC/TDT model, which predicts output tokens directly from the audio encoder. On hard dictated numbers both climb from 80–88% to 98.8–99.8%, at a cost of 0.7 to 1.2 WER points on general speech.

The third contribution is a benchmark. It combines the test sets built with the pipeline with the public Numb3rs set [16], and the baseline results above, from the models as released, are its first measurement.

2 Related Work

Prior work treats numbers as a formatting problem rather than a recognition problem. The written form of a number is usually fixed after recognition by inverse text normalization (ITN), which converts spoken forms such as "four twenty" into written forms such as "420". ITN is evaluated separately from recognition [29]. NumPipe instead scores the recognized value directly, so word-form and digit-form outputs are compared on equal footing. Numb3rs [16] is a public set of 10,131 TTS utterances with human-verified labels in twelve entity categories, built for text normalization. We use its five numeric categories as an independent, human-verified test.

Synthetic speech has been used to augment training for rare words, domain adaptation and

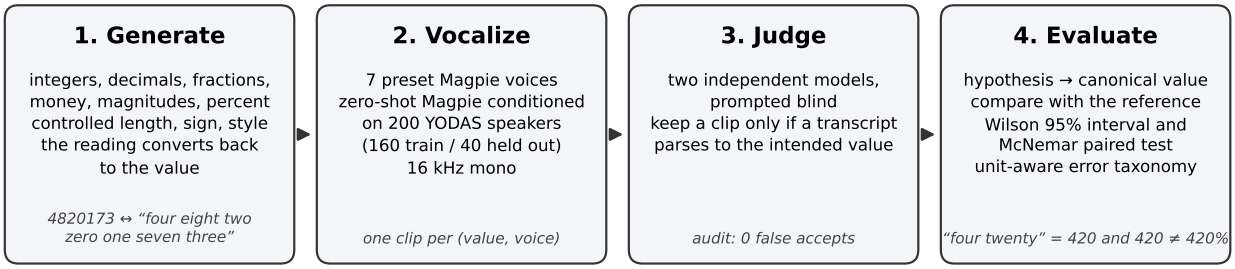

Figure 1: NumPipe. Generation produces numbers with a known written value and a spoken reading. Vocalization synthesizes each reading with preset and zero-shot voices. Judging keeps a clip only if an independent model, not told the target, recovers the intended value. Evaluation compares the recognized value with the intended one. The first three stages produce the test sets of Table 1.

low-resource languages [12, 13, 24], usually with in-domain evaluation. NumPipe adds zero-shot voices that are disjoint from the training speakers and a regression suite on public benchmarks. To limit forgetting during finetuning, our proof of concept replays general speech and labels it with the transcripts of the baseline model, following the self-training recipe of [11]. Speech-augmented language models couple an audio encoder to an LLM [6]. They are finetuned with low-rank adapters, small trainable matrices added to a frozen model [10]. Canary-Qwen-2.5B [18] is the LLM-decoder system studied here.

3 The Pipeline

The pipeline has four stages, shown in Figure 1, and the following subsections describe them in more detail. Each stage is a separate NeMo Curator component, so the pipeline is modular. A user can swap the judge models or the TTS system. Generation can also be adapted to other number formats or to another language by supplying readings and a parser grammar.

3.1 Generation

Each entity pairs a canonical written form with a deterministic English reading. Examples of written forms are "4820173", "-3.045", "3/5", "$1234.56", "472000000" and "12.5%". The generator covers integers, decimals, fractions in ordinal and "N over M" form, money with and without cents, round magnitudes such as "sixty million dollars", and percentages. Each number can be read as a digit string, "four two zero", or as a cardinal, "four hundred twenty". Zero can optionally be read as "oh". Integer parts run from one to twenty-five digits, and the long range is oversampled. Decimals and money add further digits, up to thirty-one in total. Every reading must convert back to its written form through the parser that later scores hypotheses. A set is accepted only when this holds for all rows. The generator also reports small value spaces rather than silently exhausting them. Ordinal fractions, for instance, offer only about sixty-six distinct values.

3.2 Vocalization

Readings are synthesized with seven preset English voices of NVIDIA's Magpie TTS [15] and with the Magpie zero-shot model [15]. The zero-shot model is conditioned on single-speaker clips of three

to ten seconds. These clips were cut by speaker from enhanced recordings of YODAS, a corpus of YouTube speech [14]. Of the two hundred reference speakers, one hundred and sixty are reserved for training data and forty are held out. A test with unseen voices therefore exists by construction. All audio is resampled to 16 kHz mono.

3.3 Judging

TTS systems can make mistakes too, most often on long numbers. Two independent judges therefore transcribe each clip. One is an ASR model. The other is an audio-capable language model prompted blind, that is, without being told the target. The clip is kept only if at least one transcript parses to the intended value. This design follows an audit with hard negatives. In the audit, the intended number was perturbed by one digit while the audio was left unchanged. The audit had three findings.

- A language-model judge that was told the target accepted 4.3% of these negatives. In every such case its transcript echoed the wrong number it had been given.
- The same model judging blind accepted none of them, but it rejected 9.4% of correct clips through over-strict normalization.
- A consensus of ASR models compared on values reached 99.3% accuracy, with no false accepts and 1.4% false rejects.

In a spot check of 120 random zero-shot clips, the deployed judge found 109 faithful to the intended number. The misses were mostly money and decimals of around thirty digits. On the full set of 49,959 zero-shot clips it kept 94.3%. To keep the judge independent, models that take part in it are not evaluated as systems under test.

3.4 Evaluation

A hypothesis is parsed to a canonical value. The parser reads words, digits, hybrids, magnitudes, currency and percent expressions, and it tolerates homophones such as "to" for "two". The value is then compared with the reference. Units belong to the value, so "50" is not accepted for "50%". The share of clips whose value matches is the number transcription accuracy of Section 1. The evaluation stage attaches a 95% confidence interval to every accuracy, computed with the Wilson formula for proportions. It shows how far the number could move if a different set of clips had been drawn. The tables in this paper give the point estimates. To compare two systems on the same clips, we also count the clips that only one of them gets right, which shows where a difference comes from. General-speech quality is measured as WER after the English text normalizer of the Whisper toolkit [21], which unifies spelling, numbers and punctuation before scoring. This is the convention of the Open ASR Leaderboard [25]. For the numbers themselves we do not rely on that normalizer. It handles digit strings and fully spelled-out cardinals. ASR models, however, often write long numbers in a hybrid form such as "43 quintillion 455 quadrillion", and there the normalizer expands each scale word separately and loses the value. Common word-to-number libraries stop at twelve digits. The parser of the evaluation stage reads both forms up to thirty-one digits.

4 Evaluation Sets

Table 1 lists the sets used in this paper. Two are produced by the pipeline. The first holds hard numbers spoken by the seven preset voices. The second holds fresh numbers spoken by the forty

held-out zero-shot voices. To these we add the five numeric categories of Numb3rs, whose labels were verified by human annotators. The eight English sets of the Open ASR Leaderboard guard general accuracy.

Table 1. Evaluation sets.

test set	n	measures
Hard numbers, seven preset voices, unseen values (pipeline)	868	numeric skill in the training domain
Fresh numbers, forty held-out zero-shot voices (pipeline)	1005	generalization across voices
Numb3rs [16], five numeric categories, human-verified	2792	real-world number formats
Open ASR Leaderboard, eight English sets [25]	82,185 utt.	general-speech accuracy

5 Results

5.1 Baseline systems

Table 2 shows four baseline models on the pipeline's own hard-number set, the 868 clips of Table 1, by the number of digits in the spoken value. Two recent models, Qwen3-ASR-1.7B [26] and ARK-ASR-3B [2], solve the set at 99.5% and stay above 98% even at 26 to 31 digits. The two open models finetuned later in this paper, Canary-Qwen-2.5B and parakeet-tdt-0.6b-v2, reach 85.7% and 80.5%, and their accuracy falls with length to 67.9% and 66.1% on the longest numbers. The two judge models of this set are excluded.

Table 2. Baseline models on the pipeline's hard numbers, by digit length. Number transcription accuracy in percent on the 868-clip preset-voice set of Table 1. Clips are spread roughly evenly across the length bins, with fewer above 25 digits.

system	1–6 digits	7–10	11–15	16–20	21–25	26–31	all
Qwen3-ASR-1.7B	99.5	99.2	100.0	99.4	99.3	100.0	99.5
ARK-ASR-3B	100.0	99.2	100.0	99.4	98.6	100.0	99.5
Canary-Qwen-2.5B	92.0	91.3	87.4	82.8	80.7	67.9	85.7
parakeet-tdt-0.6b-v2	94.7	85.0	77.0	79.9	69.0	66.1	80.5

Table 3 shows five baseline systems on the full Numb3rs set of 10,131 clips. The evaluation stage alone separates systems that look alike under WER. Every system reads cardinal numbers well, but digit strings are another matter. Much of the raw digit-string gap is a matter of convention, namely reading zero as "oh", and the value comparison absorbs it. WER over all twelve categories, by contrast, is a poor guide to numeric quality. It is dominated by normalizer artifacts and by the failures of small models on very short clips. Typical artifacts are "pounds" against "£" and "3 and a half" against 3.5. Models that judged this set are excluded from the table.

Table 3. Baseline systems on Numb3rs. Accuracy on the two headline categories and WER over all twelve categories.

baseline system	cardinal (n=780)	digit strings, homophones tolerated (n=771)	WER, twelve categories
ARK-ASR-3B	99.6	99.4	4.1
Riva Parakeet 1.1B	99.2	99.4	12.0
Canary-Qwen-2.5B	98.6	91.8	16.6
Granite-speech	97.8	85.7	42.0
parakeet-tdt-0.6b-v2	95.6	89.9	16.7

Table 4, first and third rows, adds the two open models on the remaining sets. parakeet-tdt recognizes 88.3% of the numbers spoken by unseen voices and Canary-Qwen 87.0%. On Numb3rs they reach 91.8% and 96.6%, with digit strings at 86.5% and 92.1%, and their leaderboard WERs are 6.05 and 5.54.

5.2 Proof of concept with finetuning on generated speech

We finetuned both models on 193 thousand utterances, about 399 hours. Generated numeric speech accounts for 88 hours in the preset voices, 93 hours in the training zero-shot voices and 9 hours of short strings and magnitudes. All of it passed the judge. The rest is replay of general speech from YODAS and from LibriSpeech train-clean-100, transcribed by the baseline model itself. The CTC/TDT model parakeet-tdt-0.6b-v2 was trained in full precision for 2000 steps at learning rate 2e-5, with targets written in words. The LLM-decoder model Canary-Qwen-2.5B kept its released configuration. Its language model is frozen, and its adapters, encoder and projection are trained on targets written in digits, for 2000 steps at four learning rates between 1e-5 and 1e-4. Appendix A lists the configurations.

Table 4. Before and after finetuning. Accuracy in percent, and leaderboard WER over eight sets with the change against the baseline model in parentheses. C1 and C6 are the configurations of Table A1. Per-set results are in Table A3.

system	preset voices	unseen speakers	Numb3rs (digit strings)	leaderboard WER
parakeet-tdt, baseline	80.5	88.3	91.8 (86.5)	6.05
parakeet-tdt, finetuned	99.7	98.8	98.8 (97.4)	6.72 (+0.67)
Canary-Qwen, baseline	85.7	87.0	96.6 (92.1)	5.54
Canary-Qwen, finetuned, lr 1e-4 (C1)	99.7	99.6	97.6 (95.0)	6.78 (+1.24)
Canary-Qwen, finetuned, lr 1e-5 (C6)	86.1	90.4	97.1 (96.7)	5.76 (+0.22)

Both models reach 98.8–99.8% on the pipeline's sets, including voices never seen in training, and 97.6–98.8% on Numb3rs. The gains are far larger than the noise. Parakeet, for example, gains 211 Numb3rs clips and loses 16, and Canary-Qwen gains 122 preset-voice clips and loses 1. Canary-Qwen also acquires a single written convention. Of its 2995 Numb3rs outputs, 2983 are plain digit strings, against 324 for the baseline model.

The evaluation also reports what the fix costs. General-speech error rises by 0.67 points of leaderboard WER for parakeet. For Canary-Qwen it rises by 0.2 to 1.2 points, depending on the learning rate. Lower learning rates reduce this cost but also the gain on dictated numbers, as Table A2 shows. The evaluation makes this trade-off explicit before deployment rather than after.

6 Limitations

Each finetuning configuration is a single run, so differences below about half a point on sets of one to three thousand clips are within noise. We therefore give the clips gained and lost for the headline comparisons in Section 5.2. Every synthetic test set, Numb3rs included, is spoken by the same TTS family that produced the training data. The zero-shot set holds out speakers, not the synthesizer, so no test in this paper uses audio from a different synthesizer. The pipeline is English-only, so other languages require readings for the generator and a parser grammar.

7 Conclusion

NumPipe turns the evaluation of spoken numbers into an automatic measurement with no human labeling in the loop. Its four stages generate numbers with known values, vocalize them with preset and zero-shot voices, judge every clip with independent models, and score a model on the value it recognized. Applied to four baseline models, the pipeline shows a gap that WER conceals. The strongest recent models read hard dictated numbers at 99.5%. Two widely used open models with low general WER manage only 80–88%, and they fail more often the longer the number gets. Because the same stages produce training data, the gap can be closed. A 2000-step finetune brings both open models to 98.8–99.8% on dictated numbers, at a cost of 0.7 to 1.2 points of general WER that the evaluation reports alongside the gain. The pipeline is released as NeMo Curator stages, and its judge, TTS system and generated numbers can be swapped or adapted. We propose it as the acceptance test for claims about numeric accuracy in speech recognition.

Reproducibility Statement

The pipeline is released as NeMo Curator stages on a public branch [17], with a README that reproduces the generation, vocalization, judging and evaluation of this paper. Appendix B gives the location. Generated sets are reproduced deterministically from the recipes and seeds. Generated training audio is not redistributed, since the pipeline regenerates it. The proof of concept used public NeMo recipes with the hyperparameters stated in Section 5.2 and Appendix A.

Ethics Statement

Synthetic clips contain no personal data. Zero-shot reference speakers are YODAS voices used only for conditioning, and no generated audio is redistributed. Numb3rs is public under CC-BY-NC-SA-4.0 [16] and is used unchanged.

Statement on AI Use

An AI assistant polished the text, fixing typos, translating and rephrasing sentences. It also checked the paper against the conference's requirements and checked the related-work references, and it helped write the README of the released branch. The pipeline itself relies on an AI model as one of its judges, as described in Section 3.3.

References

[1] S. Arora, K.-W. Chang, C.-M. Chien, Y. Peng, H. Wu, Y. Adi, E. Dupoux, H.-Y. Lee, K. Livescu, and S. Watanabe. On the Landscape of Spoken Language Models: A Comprehensive Survey. Transactions on Machine Learning Research, 2025.

[2] AutoArk. ARK-ASR-3B model card. Hugging Face model AutoArk-AI/ARK-ASR-3B, 2026.

[3] K. Binici, A. R. Kashyap, V. Schlegel, A. T. Liu, V. P. Dwivedi, T.-T. Nguyen, X. Gao, N. F. Chen, and S. Winkler. MEDSAGE: Enhancing Robustness of Medical Dialogue Summarization to ASR Errors with LLM-generated Synthetic Dialogues. AAAI, 2025.

[4] J. Carletta et al. The AMI Meeting Corpus: A Pre-announcement. MLMI, 2005.

[5] G. Chen et al. GigaSpeech: An Evolving, Multi-domain ASR Corpus with 10,000 Hours of Transcribed Audio. Interspeech, 2021.

[6] Z. Chen, H. Huang, A. Andrusenko, O. Hrinchuk, K. C. Puvvada, J. Li, S. Ghosh, J. Balam, and B. Ginsburg. SALM: Speech-augmented Language Model with In-context Learning for Speech Recognition and Translation. ICASSP, 2024.

[7] C. Cieri, D. Miller, and K. Walker. The Fisher Corpus: a Resource for the Next Generations of Speech-to-Text. LREC, 2004.

[8] J. J. Godfrey, E. C. Holliman, and J. McDaniel. SWITCHBOARD: Telephone Speech Corpus for Research and Development. ICASSP, 1992.

[9] F. Hernandez, V. Nguyen, S. Ghannay, N. Tomashenko, and Y. Estève. TED-LIUM 3: Twice as Much Data and Corpus Repartition for Experiments on Speaker Adaptation. SPECOM, 2018.

[10] E. J. Hu et al. LoRA: Low-Rank Adaptation of Large Language Models. ICLR, 2022.

[11] J. Kahn, A. Lee, and A. Hannun. Self-Training for End-to-End Speech Recognition. ICASSP, 2020.

[12] A. Laptev, R. Korostik, A. Svischev, A. Andrusenko, I. Medennikov, and S. Rybin. You Do Not Need More Data: Improving End-to-End Speech Recognition by Text-to-Speech Data Augmentation. CISP-BMEI, 2020.

[13] J. Li, R. Gadde, B. Ginsburg, and V. Lavrukhin. Training Neural Speech Recognition Systems with Synthetic Speech Augmentation. arXiv:1811.00707, 2018.

[14] X. Li, S. Takamichi, T. Saeki, W. Chen, S. Shiota, and S. Watanabe. YODAS: YouTube-Oriented Dataset for Audio and Speech. ASRU, 2023.

[15] NVIDIA. Magpie TTS Multilingual and Magpie TTS Zero-shot (NIM), 2025.

[16] NVIDIA. Numb3rs: a speech benchmark for text normalization and inverse text normalization (10,131 human-verified Magpie-TTS utterances, 12 categories, CC-BY-NC-SA-4.0). Hugging Face dataset nvidia/Numb3rs, 2025.

[17] NVIDIA. NeMo Curator: scalable data curation for speech, text and multimodal training. GitHub NVIDIA-NeMo/Curator, 2025.

[18] NVIDIA. Canary-Qwen-2.5B model card. Hugging Face, 2025.

[19] P. K. O'Neill et al. SPGISpeech: 5,000 Hours of Transcribed Financial Audio for Fully Formatted End-to-End Speech Recognition. Interspeech, 2021.

[20] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. LibriSpeech: An ASR Corpus Based on Public Domain Audio Books. ICASSP, 2015.

[21] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever. Robust Speech Recognition via Large-Scale Weak Supervision. ICML, 2023.

[22] F. Retkowski, M. Züfle, A. Sudmann, D. Pfau, S. Watanabe, J. Niehues, and A. Waibel. Summarizing Speech: A Comprehensive Survey. EMNLP, 2025.

[23] M. Del Rio, P. Ha, Q. McNamara, C. Miller, and S. Chandra. Earnings-22: A Practical Benchmark for Accents in the Wild. arXiv:2203.15591, 2022.

[24] N. Rossenbach, A. Zeyer, R. Schlüter, and H. Ney. Generating Synthetic Audio Data for Attention-Based Speech Recognition Systems. ICASSP, 2020.

[25] V. Srivastav et al. Open Automatic Speech Recognition Leaderboard. Hugging Face, 2023.

[26] Qwen Team. Qwen3-ASR-1.7B model card. Hugging Face model Qwen/Qwen3-ASR-1.7B, 2026.

[27] C. Wang et al. VoxPopuli: A Large-Scale Multilingual Speech Corpus for Representation Learning, Semi-Supervised Learning and Interpretation. ACL, 2021.

[28] H. Xu, F. Jia, S. Majumdar, H. Huang, S. Watanabe, and B. Ginsburg. Efficient Sequence Transduction by Jointly Predicting Tokens and Durations. ICML, 2023.

[29] Y. Zhang, E. Bakhturina, K. Gorman, and B. Ginsburg. NeMo Inverse Text Normalization: From Development to Production. Interspeech, 2021.

Appendix A. Finetuning configurations and full results

Each Canary-Qwen configuration changes one setting relative to the previous one. The effect of every change can therefore be read off directly. Parakeet-tdt was finetuned in full precision, because half precision broke its greedy decoding. It used AdamW at learning rate 2e-5, batch 8 with two-step accumulation, 2000 steps and SpecAugment.

Table A1. Canary-Qwen finetuning configurations. All configurations share a frozen LLM, LoRA rank 128 on the query and value projections, a trainable projection, digit targets and 2000 steps.

configuration	data and settings	learning rate
C1	released recipe on the common mix, encoder trainable	1e-4
C2	encoder frozen, short digit strings and round magnitudes added at weight 3	1e-4
C3	encoder trainable, that set at weight 6, magnitudes written as "N million"	1e-4
C4	C3 plus fifty hours of conversational replay (Fisher [7], Switchboard [8], AMI-train [4], with baseline transcripts)	1e-4
C5 / C6	C4 data	2e-5 / 1e-5
C7	C6 plus a set of percentages	1e-5
C8 / C9	C7 data	5e-5 / 2e-5

Table A2. Canary-Qwen at four learning rates on the same data. The configurations are C4, C8, C5 and C6. The leaderboard-subset change is the mean WER change against the baseline over five sets, AMI, Earnings22, LibriSpeech-other, TED-LIUM and VoxPopuli.

learning rate	preset voices	unseen speakers	Numb3rs (digit / fraction)	AMI WER (baseline 9.99)	leaderboard-subset change
1e-4	99.54	99.60	97.82 (95.9 / 98.2)	13.91	+1.32
5e-5	99.65	99.60	97.92 (95.9 / 98.2)	13.41	+1.10
2e-5	98.27	97.31	98.10 (97.6 / 87.4)	11.61	+0.66
1e-5	86.06	90.35	97.10 (96.7 / 73.5)	10.46	+0.28

Table A3. Open ASR Leaderboard, Whisper-normalized WER, for the systems of Table 4. The eight sets are AMI [4], GigaSpeech [5], LibriSpeech clean and other [20], Earnings22 [23], SPGISpeech [19], TED-LIUM [9] and VoxPopuli [27].

set	parakeet baseline	parakeet finetuned	Canary baseline	Canary lr 1e-4 (C1)	Canary lr 1e-5 (C6)
AMI	10.98	13.25	9.99	15.72	10.46
GigaSpeech	9.73	10.00	9.22	10.62	9.60
LibriSpeech clean	1.68	1.81	1.62	1.59	1.61
LibriSpeech other	3.20	3.54	3.08	3.12	3.09
Earnings22	11.29	12.59	10.30	11.55	10.50
SPGISpeech	2.14	2.49	1.90	1.98	1.91
TED-LIUM	3.41	3.73	2.65	3.98	3.26
VoxPopuli	5.96	6.33	5.59	5.73	5.67
mean, eight sets	6.05	6.72	5.54	6.78	5.76

Appendix B. Reproducibility

The pipeline is released in NeMo Curator, `https://github.com/NVIDIA-NeMo/Curator` [17], on the branch `NumPipe_Gen`. A README on that branch explains how to run it. Numb3rs is available at `https://huggingface.co/datasets/nvidia/Numb3rs` [16].