

Teacher in the Loop: Reliable Distillation of Kokoro-82M into 8M-Parameter Single-Voice Students, in English and Italian

Fabio Angela

Independent researcher

fabio.angela@mithrilsolution.com

ORCID [0009-0001-1572-1041](https://orcid.org/0009-0001-1572-1041)

October 2026

DOI [10.5281/zenodo.23400040](https://doi.org/10.5281/zenodo.23400040)

Abstract

Paradee distills Kokoro-82M into an 8M-parameter single-voice text-to-speech model whose decoder is trained on 1.6-second crops, first with spectral losses and then adversarially. Re-run from scratch, the published recipe did not reach the released decoder in our runs: 1.94 and 2.89 UTMOS on 200 held-out sentences, against 4.37. We trace this gap to the target of the spectral stage. Each crop is cut from a teacher render of the whole utterance, and Kokoro’s decoder normalizes over the whole utterance and draws a random excitation, so the target cannot be predicted from the crop: regression converges to an average, and the phase-blind spectral loss barely separates good decoders from bad ones. We put the frozen teacher in the loop. It renders each crop with the excitation the student receives, so the target becomes a function of the crop, and we distil the teacher’s inverse-STFT head directly. A pool of 150k pre-rendered crops halves the cost of a step; architecture and size are unchanged. With the published stage lengths, the revised recipe scores 4.41 for `af_heart`, above the released decoder (4.37). With a 30,000-step final adversarial stage it reaches 4.47, and without per-voice tuning it trains Kokoro’s two Italian voices from scratch to 3.88 and 3.70 (teachers 4.02 and 3.87); in our runs the upstream recipe reached 2.21 on one of them. The existing text students lose nothing end to end, the original phase-locking post-filter now lowers the score, and int8 export with per-channel scales costs the final decoders at most 0.01 UTMOS at the same CPU speed. Small decoders trained with the published stage lengths keep about half of the teacher’s harmonic contrast between 2 and 4 kHz or less, a deficit UTMOS does not register; the longer adversarial stage closes 20–30% of it, and, on one voice, a generator with twice the channels most of it, at 2.2 times the CPU time. Code, weights and audio samples: <https://github.com/MithrilMan/teacher-in-the-loop>; <https://huggingface.co/MithrilMan/teacher-in-the-loop>; <https://huggingface.co/spaces/MithrilMan/teacher-in-the-loop>.

1 Introduction

Paradee [25] shows that Kokoro-82M [7], an open text-to-speech model derived from StyleTTS 2 [22], can be distilled into an 8.07M-parameter student that speaks one of its voices. The student keeps Kokoro’s two halves with much narrower layers. Each half is trained separately against the frozen teacher: a text side that predicts durations, pitch, energy and phoneme features, and a decoder that turns the teacher’s own values into audio. The released English model scores 4.41 on UTMOS [33] against the teacher’s 4.52,

and its int8 file is 9.04 MB (8.45 MB in its paper). Its code, weights and paper are openly released, which made the reproduction and the follow-up in this paper possible.

We set out to use the same recipe to build small Italian voices. Our first Italian students had a metallic, robotic timbre. To find out whether this came from the language or from the recipe, we re-ran the original English experiment with the published training scripts. The run completed, but its decoder did not reach the quality of the released one, and neither did a second run with a different seed and teacher cache (Section 3.1). On 200 held-out sentences, the two complete runs scored 1.94 and 2.89 on UTMOS, and the released decoder 4.37. The gap was already present after the spectral stage, before any adversarial training: at that point our decoder scored 1.23, and Paradee reports 2.98 (its Table 5; its final decoder scores 4.37, and the full model 4.41 with its post-filter). An independent user later reported a similar stage-one (reconstruction) score on another English corpus [30].

Our main finding concerns the target of the first decoder stage. The student is trained on 64-frame crops, and its target is the matching segment of a teacher render of the whole utterance. Kokoro’s decoder normalizes its activations with instance-normalization statistics [12, 36] computed over the whole utterance, and its harmonic excitation [38] is drawn at random. The target crop therefore depends on audio the student never sees and on random numbers it does not share. Even the teacher, given only the crop and the same excitation, differs from the stored target by a log-mel L1 of 1.24, more than twice the student’s own error (Section 3). Regressing onto such a target yields an average, and the phase-blind spectral loss used in that stage barely distinguishes good decoders from bad ones. Whether a run ends at 1.2 or near 3 then varies from run to run and, as we show for Kokoro’s two Italian voices, from voice to voice.

Our contributions are:

1. **A diagnosis of the reproducibility gap** (Section 3). The stage-one target is not a function of the training crop, the stage-one loss does not track quality, and the resulting students ignore the random part of their excitation.
2. **A teacher-in-the-loop decoder objective** (Section 4). The frozen teacher renders every training crop with the excitation that the student receives, so the target is deterministic, and the student also matches the teacher’s inverse-STFT head, its log-magnitude and the sine of its phase. In ablations, a deterministic target alone does not help, and neither does adversarial training from the first step on the stored target: the head losses are what lift the student off the floor. A pool of pre-rendered crops halves the cost of a training step.
3. **Results in English and Italian with the same 3.85M decoder** (Section 6). With a 30k-step first stage, the decoder for `af_heart` scores 4.37, equal to the released decoder, and 4.41 with the published 50k-step first stage. Kokoro’s Italian voices `if_sara` and `im_nicola` train from scratch with the same settings, where the upstream recipe reached 2.21 and 3.53 in our runs. Running the last adversarial stage for 30,000 instead of 5,000 steps improves all three voices, to 4.47 in English (teacher 4.52) and 3.88 and 3.70 in Italian (teachers 4.02 and 3.87). Paradee’s phase-locking post-filter now lowers UTMOS, so we remove it, and int8 weights with one scale per output channel cost our final decoders at most 0.01.
4. **A remaining deficit** (Section 7). Our decoders with 5,000-step adversarial stages and the released one keep the teacher’s harmonic contrast below 1 kHz but have only about half of it between 2 and 4 kHz, or less. UTMOS does not register this. An explicit harmonic input does not reduce the deficit; the longer final stage closes 20 to 30% of it, and, for `if_sara`, a generator with twice the channels most of it, at 2.2 times the CPU time.

The gain we claim is first of all reliability: with the published stage lengths, the revised recipe reaches the released model’s quality and trains new voices, and in a three-seed check on one Italian voice the final recipe’s decoders scored within 0.003 of each other. It remains bounded by the

teacher, and in English UTMOS is close to saturation. All evidence comes from automatic predictors and two blind listening checks by the author, with one training seed per voice except for that check (Section 9). Negative results and further measurements are in the appendices. The training code is available at <https://github.com/MithrilMan/teacher-in-the-loop>, the weights at <https://huggingface.co/MithrilMan/teacher-in-the-loop> and audio samples at <https://huggingface.co/spaces/MithrilMan/teacher-in-the-loop>.

2 Background

2.1 Kokoro’s decoder

Kokoro-82M [7] follows the StyleTTS 2 design [22]. A text side, built on a phoneme-level BERT [21] and a prosody predictor, turns phonemes into a duration per phoneme, a pitch contour F_0 , an energy contour N and 512-channel phoneme features. The features are stretched to one vector per acoustic frame (600 samples, 25 ms at 24 kHz), while F_0 and N have two values per frame. A 256-dimensional style vector selects the voice: its first half conditions the decoder and its second half the prosody predictor. For a given voice, Kokoro picks this vector from a table indexed by the number of phonemes in the utterance.

The decoder has two parts. A stack of residual 1-D convolutional blocks combines the features with F_0 and N . Each block is conditioned on the style through adaptive instance normalization (AdaIN) [12]: every channel is normalized to zero mean and unit variance over the time axis of the input [36], and then scaled and shifted by amounts predicted from the style. The normalization statistics are recomputed on every call over the whole input (there are no running statistics). One consequence matters throughout this paper. The output at any instant depends on the statistics of the entire input. A 1.6-second excerpt fed to the decoder is therefore normalized differently from the same excerpt inside a ten-second utterance. Our audit of the decoder found 58 such normalization operations, all computed over the input extent.

The second part is a waveform generator in the iSTFTNet family [16], built on HiFi-GAN [17]. It is driven by a harmonic excitation in the neural source-filter style [37, 38]. A sine generator produces the fundamental and its first eight overtones from the upsampled F_0 , the overtones with random initial phases, plus a little Gaussian noise. In unvoiced regions it produces Gaussian noise instead. A learned linear layer followed by tanh mixes the nine components into one excitation signal. The generator upsamples its input by a factor of 60 and ends in a convolution, `conv_post`, with 22 output channels. These form the parameters of an inverse short-time Fourier transform (iSTFT) with FFT size 20 and hop 5, which runs at 4,800 frames per second. The first 11 channels are log-magnitudes x_m and the last 11 encode the phase x_p . The waveform is

$$y = \text{iSTFT}(\exp(x_m), \sin(x_p)), \quad (1)$$

so the phase passed to the inverse transform is $\sin(x_p)$, bounded to $[-1, 1]$ radians. We call $(x_m, \sin x_p)$ the *head* of the generator.

2.2 The Paradee recipe

Paradee [25] synthesizes 12,000 English sentences with the teacher and the voice `af_heart`. For every sentence it stores the phonemes, durations, F_0 , N , the phoneme features and the waveform. The first 200 sentences are held out. Each student half is Kokoro’s own code with smaller widths. The text side (4.23M parameters) is trained to predict the teacher’s values. The decoder, preset A (3.85M parameters), keeps Kokoro’s layout but with 256 channels in the AdaIN blocks and 128 in the generator, and replaces the style vector with a learned 16-dimensional constant. Its excitation module is Kokoro’s, randomly initialized.

Because the excitation is computed without gradient, the mixer’s weights (two tensors, ten scalars) are never updated: the upstream student computes its source under `torch.no_grad()`.

The decoder is trained in three stages (Paradee §4.4; Table 9 in the appendix).

Stage one (50,000 steps, batch 16, learning rate 2×10^{-4}). The student receives the teacher’s features for a random 64-frame (1.6 s) crop. The target is the same 1.6 s of the stored full-utterance waveform. The loss is a log-mel L1 plus a multi-resolution STFT loss [40], made of spectral convergence and log-magnitude L1 at FFT sizes 512, 1024 and 2048.

GAN10, then GAN3 (5,000 steps each, batch 8). HiFi-GAN-style adversarial training [17] with a multi-period discriminator (periods 2, 3, 5, 7, 11) and a multi-resolution spectrogram discriminator [15]. It uses least-squares GAN losses [28] and feature matching [19] with weight 2, and the spectral loss weighted by 10 and then by 3.

After assembly, a parameter-free phase-locking filter corrects the phase of voiced speech between 2 and 8 kHz [20]. Weights are stored as int8 for distribution.

The paper reports UTMOS 2.98 for the decoder alone after stage one, 4.29 after GAN10 and 4.37 after GAN3. The full student scores 4.39, and 4.41 with the filter (Paradee Table 5). A 3,000-step run with spectral weight 45 (3.02) in the same paper is an ablation, not part of the published command chain.

3 Where the reproducibility gap comes from

3.1 Reproducing the recipe

We ran the published training scripts [26] (upstream revision 9c8b4d7) with their defaults, changing only the hard-coded mps device of the corpus generator to cuda, and compared against the released weights [27]. The run generated a new teacher corpus of 11,999 sentences (23.9 hours), held out the first 200, and trained the text side, decoder stage one, GAN10 and GAN3 in float32 with TF32 disabled, in about ten hours (Appendix B). We call this run *reproduction A*. An earlier run of the same recipe through our own training wrapper, *reproduction B*, differs from A in its random initialization (seed 42), its teacher cache and its floating-point policy.

Table 1 shows the outcome. On the 200 held-out sentences, with the teacher’s own features as input, reproduction A’s final decoder scores 1.94 and B’s 2.89, while the released decoder scores 4.37. The gap is already present after stage one, where A scores 1.23 and Paradee reports 2.98. The two GAN stages raise A’s score on every one of the 200 sentences but do not close the gap, and neither does Paradee’s phase-locking filter (Appendix B).

We looked for a numerical cause first. TF32 convolutions change the spectral gradient by 25% on a fixed batch, but a paired continuation that changed only this setting moved UTMOS by 0.007 (Appendix B). A user of the upstream repository independently reported a reconstruction score of about 1.2 with another software stack and another English corpus [30]. We therefore looked at the objective rather than at the arithmetic. The rest of this section uses the lab protocol of Section 5: the first 16 held-out sentences, the teacher’s features for the full utterance and fixed excitation seeds.

3.2 The stage-one loss does not track quality

Table 2 evaluates the stage-one objective itself, on full utterances. The released decoder, at 4.36, and reproduction A’s stage-one decoder, at 1.23, are almost indistinguishable by this loss, and the released decoder even has the higher log-mel error. Re-rendering the teacher with a different random excitation

Decoder	Held-out sentences	Stage one	+ GAN10	+ GAN3
Paradee paper, Table 5	200	2.98	4.29	4.37
Released decoder, our evaluation	200	–	–	4.37
Our reproduction A (published scripts)	200	1.23	–	1.94
	64	1.23	1.27	1.87
Our reproduction B (wrapper, seed 42)	200	–	–	2.89
	64	1.98	–	2.85
Teacher (Kokoro, af_heart)	200		4.52	

Table 1: UTMOS of the decoder alone (teacher conditioning) at each stage of the upstream recipe. The paper’s 2.98 is its stage-one number; its final decoder scores 4.37 (and the full model 4.41 with the post-filter). Our two reproductions score lower at every stage. “–”: not measured on that set.

Decoder (16 held-out sentences, teacher conditioning)	UTMOS	log-mel L1	MR-STFT
Teacher re-rendered with another excitation draw (floor)	–	0.097	0.233
Released decoder	4.36	0.550	0.889
Reproduction A, stage one	1.23	0.520	0.976
Reproduction B, stage one	2.05	0.530	0.979
Reproduction A, GAN3	1.84	0.642	1.055
Reproduction B, GAN3	2.84	0.607	1.021

Table 2: The two terms of the stage-one objective against the full-utterance target. The released decoder has a *worse* log-mel term than reproduction A’s stage-one decoder, and their summed loss differs by only about 4% (1.44 against 1.50), while their UTMOS differs by 3.1.

gives a log-mel distance of 0.097 to the stored target, so the students sit five to seven times above the teacher’s own floor, but within that range the loss does not order them by quality. Lowering it even degrades a good decoder. Fine-tuned with the stage-one objective (published optimizer, batch 16, 64-frame crops), the released decoder fell from 4.36 to 4.20 after 250 steps, 4.07 after 1,000 and 3.97 after 3,000, while its log-mel distance to the target *fell* from 0.550 to about 0.52. Raising the floor of the STFT logarithm from 10^{-5} to 10^{-2} changed nothing (4.07 after 1,000 steps). The objective rewards changes that the quality predictor penalizes.

3.3 Students ignore the random part of the excitation

Table 3 measures how much each decoder’s output depends on its harmonic excitation. Two new renders of the teacher with different random draws differ from each other by a log-mel L1 of 0.092. For the released decoder the difference is 0.036, and for reproduction A’s stage-one decoder only 0.015. Removing the harmonic source entirely drops the teacher from 4.52 to 1.74 and the released decoder from 4.36 to 1.29. Reproduction A’s decoder keeps its score (1.23 to 1.23): it has learned to produce its output without the excitation. This is the behaviour one expects from regression onto a target whose random part cannot be predicted. The safest output under an L1 loss is one that does not depend on the random draw, and a decoder that ignores the excitation also loses the harmonic structure that the excitation provides.

3.4 The cropped target is not a function of the crop

The stage-one target is a 1.6 s slice of a waveform that the teacher produced from the whole utterance. Two things make that slice depend on information that the student does not have. The first is that every AdaIN block normalizes over the whole input (Section 2.1), so the target was produced with statistics of

Decoder	log-mel L1 between two excitation draws	UTMOS		
		normal	no harmonic source	white-noise source
Teacher	0.092	4.52	1.74	3.47
Released decoder	0.036	4.36	1.29	2.09
Reproduction A, stage one	0.015	1.23	1.23	1.23
Reproduction B, stage one	0.013	2.05	1.69	1.38

Table 3: How much each decoder depends on its excitation (16 held-out sentences). Reproduction A’s stage-one decoder keeps the same score with no harmonic source at all, or with white noise.

Teacher rendered on	log-mel L1 to stored crop	MR-STFT	UTMOS
the full utterance (exact excitation)	0.014	0.047	4.42
the 64-frame crop only	1.24	1.36	4.19
the crop $\pm$ 16 frames of context	0.87	1.07	4.34
the crop $\pm$ 64 frames of context	0.64	0.78	4.38

Table 4: The frozen teacher, with the excitation that produced the stored target, rendered on different extents of input. Given only the crop, as the student is, the teacher misses its own stored target by 1.24, more than twice the error of any stage-one student in Table 2.

the full utterance, while the student only sees the crop. The second is that the target was rendered with one random draw of the excitation, and the student receives another.

To measure the first effect, we reconstructed the excitation that produced each stored target and ran the frozen teacher with it (Table 4). Rendered on the full utterance, the teacher reproduces its stored target almost exactly (log-mel L1 0.014). Rendered on the crop alone, with the same excitation, it misses the target by 1.24, more than twice the error of any stage-one student. The error shrinks as context is added around the crop, to 0.87 with 16 frames on each side and 0.64 with 64. The teacher’s own output on the crop is still natural speech (UTMOS 4.19). It is simply a different waveform from the stored one. The random excitation phases add a further, smaller unpredictable component.

The training target is therefore largely unpredictable from the training input, even for a model that has the teacher’s weights. An L1 regression onto such a target converges to an average. That explains the three observations above. The loss is flat with respect to perceptual quality, lowering it moves a good decoder towards the average, and the student learns to ignore the random excitation. It also explains why the outcome of stage one varies between runs. A run can settle anywhere along this flat loss, and the adversarial stages start from wherever it lands. The stage-one decoder ended at 1.23 for reproduction A and at 1.98 for B (Table 1), and Paradee reports 2.98 for its own run.

The same variation appears across voices. With the same recipe, settings and corpus, the stage-one decoder of Kokoro’s Italian voice *if_sara* scored 1.37 on its 200 held-out sentences, while that of *im_nicola* scored 3.09. After the GAN stages they reached 2.21 and 3.53 (Table 6). This matches the metallic, robotic timbre we heard in our first Italian students.

We evaluate every change to the objective by training from scratch. Any non-adversarial objective, ours included, moves the GAN-trained released decoder away from its optimum, so fine-tuning a finished decoder cannot tell which objective works (Appendix B). For the same reason the Italian decoders do not start from the English one.

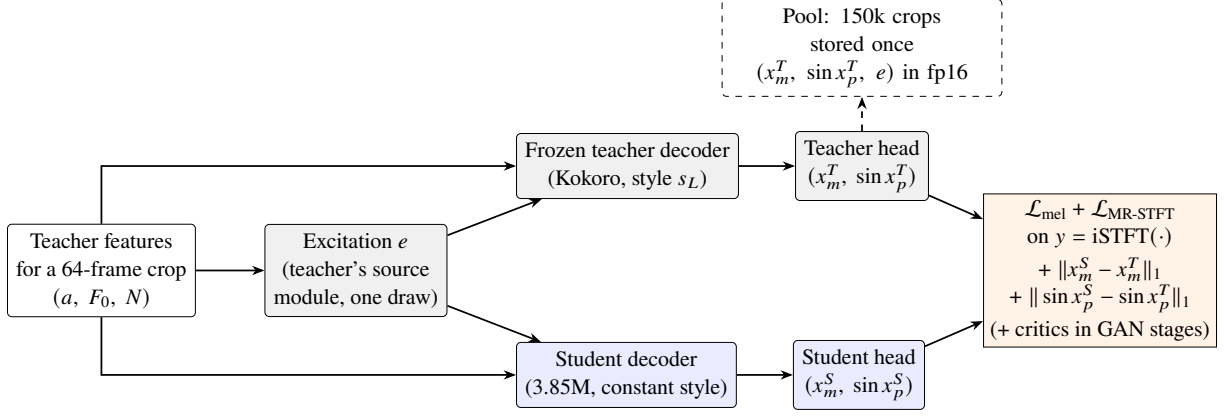


Figure 1: Teacher-in-the-loop training of the student decoder. For every training crop, the frozen teacher and the student receive the same features and the same excitation draw e , so the target is a deterministic function of the student’s input. The target is the teacher’s inverse-STFT head, and its waveform is reconstructed from that head. The losses compare the two waveforms spectrally and the two heads directly: log-magnitude x_m and the phase $\sin x_p$ that the iSTFT receives. Teacher outputs can be computed live or read from a pool of pre-rendered crops (dashed). Grey: frozen; blue: trained.

4 Method

We build on the Paradee student and keep every part that Section 3 did not point to: the architecture and size of both halves, the text side and its training, the crop length, the optimizer, the critics and the two adversarial stages. We render the decoder’s target with the teacher in the loop, add two losses on the generator head, pre-render the targets in a pool, and adjust the schedule. Figure 1 summarizes the training step, and Table 9 in the appendix lists every setting.

4.1 Teacher in the loop

For each training crop we take the teacher’s phoneme features a , pitch F_0 and energy N for the 64 frames, as in the upstream recipe. We then draw one excitation e from the teacher’s own source module on the crop’s F_0 and run the frozen teacher decoder on the crop with that excitation. The teacher uses the style vector it would use for this utterance, s_L , which Kokoro selects by the utterance’s phoneme count L . The student receives the same a , F_0 , N and e :

$$y^T = D_T(a, F_0, N, e; s_L), \quad y^S = D_S(a, F_0, N, e). \quad (2)$$

Both decoders now see exactly the same input extent and the same random numbers. The target is therefore a deterministic function of what the student sees, with one small exception discussed below. The instance-normalization statistics of the teacher are computed over the crop, as the student’s are, and the random phases of the excitation are shared.

The student’s excitation module is replaced by a copy of the teacher’s, including the mixer that the upstream student leaves at its random initialization (Section 2.2). At inference the student therefore produces excitation with the same distribution as during training and as in the teacher. The mixer, like the rest of the source module, is not trained.

The one remaining source of unpredictability is the style. The teacher’s s_L depends on the utterance’s phoneme count, while the single-voice student has one learned constant. Over the phoneme counts of the 200 Italian held-out sentences, the decoder half of Kokoro’s style vector varies by 6.5–8.0% in relative RMS for each of our three voices, so we expect this residual to be small.

4.2 Distilling the inverse-STFT head

Section 3.2 shows that magnitude losses are a weak signal. The two spectral terms discard phase, and the multi-period critic sees the waveform only adversarially. Once the target is deterministic, a stronger signal is available. The teacher’s generator ends in the head of Equation (1): per-frame log-magnitudes x_m and phases $\sin x_p$ for 11 frequency bins at 4,800 frames per second. The student’s generator has exactly the same head. We add two L1 losses that match the student’s head to the teacher’s:

$$\mathcal{L}_{\text{head}} = \|x_m^S - x_m^T\|_1, \quad \mathcal{L}_{\text{phase}} = \|\sin x_p^S - \sin x_p^T\|_1. \quad (3)$$

The phase loss compares the quantity the inverse STFT actually receives. Kokoro passes $\sin x_p$ as the phase, which is bounded to $[-1, 1]$ radians, so the target has a single representation and the loss needs no anti-wrapping treatment [1]. Both losses distil an intermediate output of the teacher, its last layer before the inverse STFT [10].

These losses only make sense with a deterministic target. In the upstream recipe, the teacher’s head for the stored target was computed with another excitation draw and with full-utterance statistics, so matching it would repeat the problem of Section 3.4. The stage-one objective becomes

$$\mathcal{L}_1 = \mathcal{L}_{\text{mel}}(y^S, y^T) + \mathcal{L}_{\text{MR-STFT}}(y^S, y^T) + \lambda_m \mathcal{L}_{\text{head}} + \lambda_p \mathcal{L}_{\text{phase}}, \quad \lambda_m = \lambda_p = 1, \quad (4)$$

where $\mathcal{L}_{\text{mel}}$ and $\mathcal{L}_{\text{MR-STFT}}$ are exactly the upstream terms (Section 2.2). A phase weight of 5 only sped up the first few thousand steps (Section 6.1).

4.3 A pre-rendered teacher pool

Running the teacher’s 53M-parameter decoder at every step dominates the cost of a training step. Because the teacher is frozen, we render its targets once. The pool holds 150,000 random 64-frame crops drawn from the training sentences. For each crop it stores the crop’s position, the excitation e (38,400 samples, fp16, 77 KB) and the teacher’s head $(x_m^T, \sin x_p^T)$ (22 channels $\times$ 7,681 frames, fp16, 338 KB), about 62 GB in total. Storing $\sin x_p$ rather than x_p loses nothing, since only $\sin x_p$ enters the inverse STFT. The target waveform is rebuilt on the GPU at training time with the teacher’s own inverse STFT. Rendering a pool takes about one hour per voice, with the teacher in fp16 (English) or in fp32 (Italian). Training from the pool took 0.27 s per step, against 0.57–0.65 s with the live teacher, on a GPU shared with two other training jobs. Each stored crop keeps its one excitation draw, while the live teacher draws a new one each time; variety comes from the 150,000 crops. We also compute the STFT of the excitation on the GPU instead of the CPU. Appendix A gives the details of both.

4.4 Schedule

Stage one. The student is trained from scratch for 30,000 steps on Equation (4), with batch 16 and AdamW [24] (learning rate 2×10^{-4} , $(\beta_1, \beta_2) = (0.8, 0.99)$, weight decay 0.01). The schedule is a 1,000-step linear warmup followed by cosine decay to zero, with gradient-norm clipping at 5. We then continue for 20,000 steps from the 30,000-step weights with a fresh optimizer, a warm restart [23] to a peak learning rate of 10^{-4} , 1,000 warmup steps and cosine decay. This brings stage one to Paradee’s 50,000 steps. The restart was added after the 30,000-step recipe had been evaluated. Section 6.2 shows that it pays after the adversarial stages.

GAN10 and GAN3. Both follow the published stages, with the same multi-period and multi-resolution critics, least-squares adversarial losses and feature matching, at batch 8. GAN10 runs for the published 5,000 steps. In the final recipe GAN3 runs for 30,000 steps instead of 5,000 (Section 6.4); with the published 5,000 steps we call the whole recipe the 50k recipe. Each stage starts from the previous generator with fresh optimizers and critics, a 1,000-step warmup and cosine decay over the whole stage. Targets come from the pool, and every reconstruction term is scaled by the stage weight λ (10, then 3), the head terms included:

$$\mathcal{L}_G = \lambda(\mathcal{L}_{\text{mel}} + \mathcal{L}_{\text{MR-STFT}} + \mathcal{L}_{\text{head}} + \mathcal{L}_{\text{phase}}) + \mathcal{L}_{\text{adv}} + 2 \mathcal{L}_{\text{fm}}. \quad (5)$$

Unlike the upstream GAN3, which uses the first eight shards of the corpus, both stages sample from the whole training pool.

Text side. We do not retrain it. The students reuse text sides already trained with the upstream text recipe (s+mlp, 4.23M parameters): in English the one from reproduction A, in Italian those of our earlier students (Section 6.5).

4.5 Export

We change two things in the upstream ONNX export. First, we remove Paradee’s phase-locking post-filter, which the export builds into the graph. It locks the 2–8 kHz phase of voiced frames to the excitation, with 33-frame smoothing, and was designed for decoders whose phase the magnitude-only objective leaves unconstrained. With head distillation it lowers UTMOS on every voice (Section 6.5). Second, we store the large weights as int8 with one scale per output channel, chosen from the operator that consumes each weight, where the upstream exporter scaled every tensor along its first axis. As upstream, the weights are dequantized in the graph, so the arithmetic stays in float32 [14, 18]. Appendix D.7 describes both schemes and the exporter’s parity check.

5 Experimental setup

Teacher and data. The teacher is Kokoro-82M v1.0 [7] (revision f3ff357). For English we reuse the teacher cache of reproduction A (Section 3.1): voice af_heart, 11,999 WikiText-103 sentences [29], the first 200 held out. The English runs of the revised recipe used 12 of the cache’s 24 shards, that is 5,799 training sentences. This is about half of what the upstream recipe trains on, a choice made for the early ablations that we kept for comparability. For Italian we built a corpus of 12,000 training and 200 held-out sentences from 1,203 articles of the Italian Wikipedia [39], with no article shared between the two splits. We synthesized it with Kokoro’s two Italian voices, if_sara and im_nicola [8], which gives 33.2 hours of audio for if_sara. Italian text is converted to phonemes as Kokoro itself does: Misaki [9] 0.9.4 with its eSpeak NG [6] backend. The only change concerns the dental stops. eSpeak writes them with a combining bridge below (U+032A), which Kokoro’s phoneme vocabulary lacks, so we map them to plain t and d. The same conversion serves the teacher corpus, the students and deployment.

Metrics. Our main metric is UTMOS [33], the utmos22_strong predictor in the SpeechMOS implementation [35], the same predictor as in the Paradee paper. Its scores are on a 1–5 mean-opinion-score scale. Unless stated otherwise, scores are for the *decoder alone* (teacher conditioning, or teacher-forced): the decoder receives the teacher’s own durations, F_0 , N and phoneme features for the full utterance, and the excitation is seeded per sentence, so two decoders compared on the same sentences receive the same random draws before their mixers. *End to end* means the student text side followed by the student decoder,

starting from phonemes. We also report the two terms of the stage-one objective against the full-utterance teacher target, where lower is closer to the target (Appendix C).

Evaluation sets and statistics. During training we score the first 16 held-out sentences every 1,000 or 2,500 steps. Final comparisons use 64 sentences or all 200, as stated. For a mean over sentences we give the half-width of its 95% confidence interval. When two systems are scored on the same sentences with the same excitation seeds, we report the mean per-sentence difference with its standard error. Its significance follows the paired t -test [34], where with 100 or 200 sentences the normal approximation is accurate. Where available we also give the number of sentences that improve, with the sign test [5]. Values of p are in `data/paired_tests.csv`; here and below, `data/` is the folder `paper/data/` of the code repository (Appendix C). The same held-out sentences were used for every diagnosis and decision in this paper. They are therefore a development set, not an untouched test set (Section 9).

Exported models. The exported graphs take phoneme ids and a speed factor and return a 24 kHz waveform. They are scored with ONNX Runtime [31] on the CPU at speed 1.0, end to end on the first 100 held-out sentences, comparing the float32 and int8 graphs on identical inputs.

Hardware and software. The students were trained in float32 on one shared server with a modified NVIDIA RTX 4080 SUPER (32 GB), with PyTorch 2.14 and CUDA 13.0. The GPU was shared with other services and usually with one to three of our own jobs, so wall-clock times are only indicative (Appendix C).

6 Results

6.1 Stage one: what lifts the student off the floor

We trained six stage-one variants from scratch for `af_heart`, all with the same seed, data and schedule (Figure 2; values in Appendix D.1). The upstream objective (`s0`) stays at 1.23, while its log-mel distance to the target falls from 1.25 to 0.75. Making the target deterministic but keeping only the spectral losses (`s1`) changes nothing: the magnitude losses are too weak a signal even when the target is predictable. Adversarial training from the first step on the stored target (`s3`) is also flat over the 3,000 steps we ran, which says nothing about much longer adversarial schedules. Only the arms with the head losses of Equation (3) leave the floor. With both at weight 1 (`s2`), the student reaches 2.04 after 4,000 steps and 2.72 after 10,000; at 6,000 steps it has a *higher* log-mel distance than `s0` (0.81 against 0.75) but scores 2.56 against 1.23. A phase weight of 5 (`s5`, which also used the fp16 teacher) speeds up the start but ends at the same level (2.65 against 2.72). With the adversarial losses added from the first step (`s6`), the student rises fastest, to 3.59, but stays below what the staged schedule reaches with the same model (Appendix F.1).

6.2 Adversarial stages and the English result

The adversarial stages build on whatever stage one delivers (Table 5, Figure 3). On top of the 10,000-step `s2` decoder (2.72), the full pipeline ended at 4.02 on 64 sentences. Trained from the pool for 30,000 steps (`s7`), stage one alone reaches 3.93, above the 2.98 that Paradee reports after its 50,000-step stage one and above both of our complete upstream reproductions. The unchanged adversarial stages then take it to 4.372 on the 200 sentences, against 4.371 for the released decoder, the same number Paradee reports for its decoder (4.37). Running GAN3 directly after stage one gives 4.358, so the two-stage schedule adds

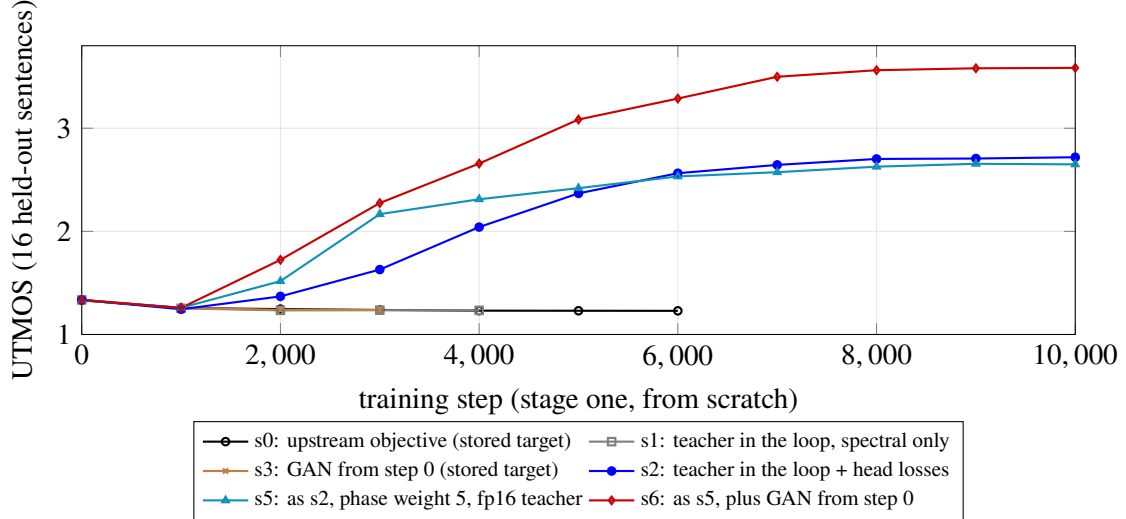


Figure 2: Stage-one arms trained from scratch for `af_heart` (seed 1, cosine schedule, half of the English corpus). Only the arms with the head losses leave the floor near 1.2. A deterministic target alone (s1) does not help, and adversarial training from the first step on the stored target (s3) does not either. Arms s0, s1 and s3 were stopped once flat.

Decoder (200 held-out sentences, teacher conditioning)	UTMOS	95% CI
Paradee paper, decoder alone (Table 5)	4.37	–
Released decoder, our evaluation	4.371	± 0.011
Ours: stage one 30k	3.930	± 0.026
Ours: stage one 30k $\rightarrow$ GAN3	4.358	± 0.011
Ours: stage one 30k $\rightarrow$ GAN10 $\rightarrow$ GAN3	4.372	± 0.012
Ours: stage one 50k	4.131	± 0.021
Ours: stage one 50k $\rightarrow$ GAN10 $\rightarrow$ GAN3 (50k recipe)	4.414	± 0.008
Ours: the same with GAN3 for 30k steps (final recipe)	4.467	± 0.006
Reproduction A, full upstream recipe	1.940	± 0.045
Reproduction B, full upstream recipe	2.889	± 0.033

Table 5: English `af_heart` on all 200 held-out sentences (Paradee’s protocol), without any post-filter. Paradee’s full model scores 4.39, or 4.41 with its phase-locking filter. All GAN stages run 5,000 steps except in the final-recipe row (Section 6.4).

only about $+0.014$. As in Section 3.2, the spectral distances do not rank these decoders: our 30k decoder after both GAN stages is further from the target than the released one, and further than our own stage-one decoder, which scores 0.43 lower (Appendix D.2).

A longer stage one. Paradee’s stage one runs for 50,000 steps. We resumed the 30k decoder with a warm restart (Section 4.4) for 20,000 more steps (s8), then ran GAN10 and GAN3 unchanged. Stage one rose from 3.930 to 4.131, and after the adversarial stages the decoder scores 4.414 against 4.372, a difference with $z = 5.9$ even when the pairing on the same sentences and seeds is ignored. This is above Paradee’s decoder (4.37) and equal to its filtered full model (4.41), without the filter, and Italian shows the same direction with paired evidence (Section 6.3). We adopt the 50k stage one and call the result, with the published adversarial stages, the *50k recipe*, and the earlier one the *30k recipe*. A blind four-sentence comparison by the author preferred s8 three times out of four (Appendix F.2).

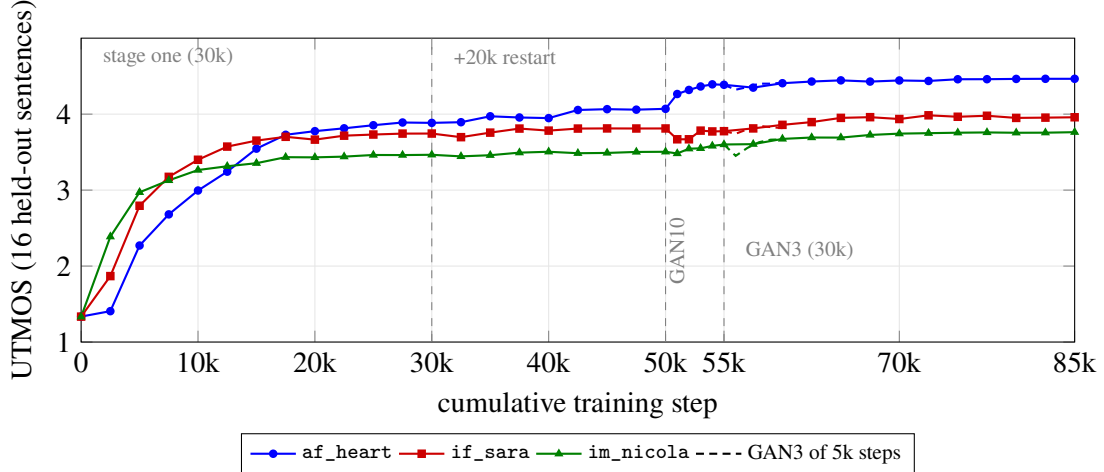


Figure 3: The final recipe for the three voices, as seen by the in-training check on the first 16 held-out sentences. Stage one runs 30k steps from scratch and continues 20k more after a warm restart. GAN10 (5k steps) and GAN3 (30k steps) follow. Dashed: the published GAN3 of 5k steps, started from the same GAN10 decoders (the 50k recipe). In English, most of the final quality is already present at the end of stage one. In Italian, the 5k-step GAN3 changes UTMOS little, while the longer one keeps improving for 12.5k to 17.5k steps. Final numbers on all 200 held-out sentences are in Tables 5, 6 and 13.

6.3 Italian voices

We trained the two Italian voices with exactly the English settings, with pools rendered by the fp32 teacher. Nothing was tuned per voice (Table 6). With the upstream recipe, *if_sara* stayed at 1.37 after stage one and reached 2.21 at the end, while *im_nicola* reached 3.09 and 3.53 (Section 3.4). With the revised recipe, the 30k stage one alone gives 3.67 and 3.39, the 50k recipe 3.75 and 3.60, and the final recipe (Section 6.4) 3.88 and 3.70. With the published stage lengths, the gain over the upstream recipe is small for *im_nicola*, whose upstream run was already a favourable one, and 1.55 for *if_sara*. Both new decoders also improve on our earlier Italian baseline, the released English decoder adapted to each voice (3.55 and 3.28), without any English initialization. Extending stage one from 30k to 50k steps improved both voices after the adversarial stages, on the same sentences and seeds: by $+0.041 \pm 0.005$ for *if_sara* (better on 153 of 200 sentences) and $+0.026 \pm 0.003$ for *im_nicola* (158 of 200), both with $p < 10^{-6}$ by the paired t -test and the sign test. Two more seeds of the final recipe for *if_sara* ended the 50k stage one lower, at 3.43 and 3.40 instead of 3.75, but reached the same final score, 3.88 (Appendix G).

In the 50k recipe the gap to the teacher is larger than in English, 0.27 for both voices against about 0.11, and the adversarial stages of the published length add much less: $+0.04$ for *if_sara* and $+0.18$ for *im_nicola* after the 30k stage one, against $+0.44$ in English, and only $+0.01$ for *if_sara* after the 50k stage one. For *if_sara*, GAN10 alone even lowers the score slightly (3.67 to 3.64), although its renders already sounded clearly better than stage one to the author. UTMOS was trained on English ratings, the Italian teachers score only 4.02 and 3.87 on it, and Kokoro’s own documentation grades both voices C [8]. We therefore read the Italian numbers mainly as a comparison between systems on the same voice (Section 9). The small gain also has a simpler cause: at 5,000 steps, the last adversarial stage is still improving (Section 6.4).

6.4 A longer final adversarial stage

For every voice, we ran the last stage of the 50k recipe again for 30,000 steps instead of 5,000, from the same GAN10 decoder and with the same losses, batch, warmup and seed; only the cosine decay was

Decoder (200 held-out sentences, teacher conditioning)	if_sara		im_nicola	
	UTMOS	95% CI	UTMOS	95% CI
Teacher target	4.022	–	3.867	–
Upstream recipe: stage one 50k	1.370	–	3.086	–
Upstream recipe: stage one $\rightarrow$ GAN10 $\rightarrow$ GAN3	2.205	–	3.535	± 0.020
Released English decoder adapted with GAN3	3.548	± 0.026	3.279	± 0.023
Ours: stage one 30k	3.674	± 0.026	3.394	± 0.023
Ours: stage one 30k $\rightarrow$ GAN10	3.640	–	3.514	–
Ours: stage one 30k $\rightarrow$ GAN10 $\rightarrow$ GAN3	3.712	± 0.026	3.575	± 0.022
Ours: stage one 50k	3.745	± 0.025	3.426	± 0.023
Ours: stage one 50k $\rightarrow$ GAN10 $\rightarrow$ GAN3 (50k recipe)	3.752	± 0.027	3.601	± 0.022
Ours: the same with GAN3 for 30k steps (final recipe)	3.882	± 0.025	3.696	± 0.021

Table 6: Kokoro’s Italian voices, trained from scratch with the English settings. In our runs, the upstream recipe trained *im_nicola* well but not *if_sara*. The revised recipe trains both. The adapted English decoder was our earlier Italian baseline: the released decoder followed by 5,000 GAN3 steps on the voice’s targets with the upstream objective. All GAN stages run 5,000 steps except in the last row (Section 6.4). “–”: interval not recorded for that run.

stretched over the longer stage, and the pools were rendered again with the same settings (Appendix D.3). On the 200 held-out sentences with the same excitation seeds (Tables 5 and 6; paired in Table 13), the longer stage improves every voice: by $+0.053 \pm 0.002$ in English, $+0.130 \pm 0.007$ for *if_sara* and $+0.094 \pm 0.003$ for *im_nicola*, with 197, 187 and 197 of the 200 sentences better ($p < 10^{-6}$ for each). The gap to the teacher falls from 0.11 to 0.05 in English and from 0.27 to 0.14 and 0.17 in Italian, and the adversarial stages now add 0.34, 0.14 and 0.27 after the 50k stage one, instead of 0.28, 0.01 and 0.18. In English the final decoder scores 4.467, against 4.371 for the released decoder. On the 16 monitored sentences (Figure 3), the Italian scores level off after 12,500 to 17,500 steps, while the English score is still rising slightly at 30,000. At the end of a 5,000-step stage the learning rate has decayed to zero, while the longer stage is still at 93% of its peak at the same step. We did not search over the length, and the longer stage costs 5.4 to 6.7 hours on the shared GPU instead of about one hour. We adopt it as the *final recipe* (Appendix A).

6.5 End to end, export and speed

Paradee joins its two halves without joint training, and so do we. Connected to the new decoders, the existing text sides lose nothing measurable (Appendix D.4): from phonemes, the final decoders score 4.474, 3.895 and 3.700 on the 200 held-out sentences, against 4.467, 3.882 and 3.696 with the teacher’s own values. For our decoders and the released one, end-to-end UTMOS is never below the teacher-forced score: in Italian it is within 1.8 standard errors of it, and in English 0.007 to 0.020 higher, as Paradee also observed for its own model (4.39 against 4.37). The predicted durations total 98.8%, 98.7% and 99.1% of the teacher’s. Only the decoders need to be replaced.

Words and voice. The word error rate of Whisper large-v3-turbo [32] shows no student differing from its teacher beyond noise, with rates of 3.1 to 4.3% against the teachers’ 3.3, 4.1 and 4.5%. The speaker similarity of WavLM x-vectors [3] to the teacher’s audio of the same sentence is 0.984 to 0.996 for every decoder, and in English ours are closer to the teacher than the released one (0.993 and 0.991 against 0.985; Appendix D.5).

The phase-locking post-filter. Paradee’s filter replaces the 2–8 kHz phase of voiced frames with the phase of the excitation plus a smoothed offset. The worse a decoder’s phase, the more it helps: +0.32 for reproduction A, +0.03 for B and +0.01 for the released decoder. It lowers the score of every decoder trained with head distillation after its adversarial stages, by 0.03 to 0.04 in English and by 0.08 to 0.22 in Italian (Appendix D.6), so we removed it from the export.

Int8, size and speed. With one scale per output channel, int8 storage costs the final decoders at most 0.01 UTMOS end to end: -0.006 ± 0.001 in English and -0.007 ± 0.011 and -0.010 ± 0.005 in Italian, on 100 held-out sentences with the same inputs and seeds. The int8 graph is 9.17 MB, against 32.8 MB in float32. It fails the parity gate inherited from our earlier exporter, which can only compare cases in which every token duration is unchanged, so we report the end-to-end change instead (Appendix D.7). Parameters (8.07M) and decoder compute (3.36 GFLOP per second of audio) are Paradee’s. With Paradee’s protocol (one CPU thread, the first 8 held-out sentences, from phonemes), on an AMD Ryzen 9 9950X3D with ONNX Runtime 1.30, the int8 graphs of the final recipe run 24 to 26 times faster than real time and the released int8 graph 23 to 25 times, within the 8% by which the same graph varied in one session; Paradee reports 17.8 on an Apple M4 Pro.

7 Harmonic definition above 2 kHz

UTMOS ranks our English decoders level with the released one or above it, but it does not say what all of them still lack. We therefore measured how well each decoder separates harmonics from the noise between them, on 40 held-out sentences rendered from the teacher’s own features. On voiced frames we take, for every harmonic in a band, the peak level near kF_0 and the mean level halfway to the next harmonic; the *harmonic contrast* is their difference in dB (Appendix C). We do not interpret it for im_nicola, whose harmonics lie too close together for the 40 ms window.

Voice	Audio	Harmonic contrast (dB)				HNR (dB)
		0–1 kHz	1–2 kHz	2–4 kHz	4–6 kHz	
af_heart	Kokoro teacher	29.0	16.5	10.5	6.2	13.6
	released decoder	28.4	14.1	4.4	4.1	13.4
	reproduction A (upstream recipe)	24.8	7.7	3.9	4.0	12.1
	ours, 30k recipe	28.2	13.6	4.2	4.1	13.1
	ours, 50k recipe	28.3	14.2	4.2	4.1	13.1
	ours, final recipe	28.6	15.0	5.5	4.1	13.2
if_sara	Kokoro teacher	29.0	17.6	11.8	6.0	12.9
	upstream recipe	26.7	11.4	5.3	3.8	12.4
	ours, stage one 50k	28.6	15.1	3.9	3.2	12.8
	ours, 50k → GAN10	28.5	15.3	5.3	3.5	12.8
	ours, 50k recipe (GAN3)	28.5	15.7	6.0	3.6	12.8
	ours, final recipe	28.6	16.5	7.7	4.0	12.8

Table 7: Harmonic contrast of voiced speech, level at the harmonics minus level halfway between them, per band, on 40 held-out sentences rendered from the teacher’s own features. Our decoders with 5,000-step adversarial stages and the released one keep the teacher’s contrast below 1 kHz but have about half of it or less between 2 and 4 kHz, and three fifths to two thirds of it between 4 and 6 kHz. The final recipe’s longer last stage recovers part of the 2–4 kHz contrast.

Table 7 shows a deficit shared by every small decoder. Below 1 kHz all but the two upstream-recipe decoders keep the teacher’s contrast (28–29 dB against 29.0), and the harmonics-to-noise ratio (HNR)

	Decoder params	Contrast 2–4 kHz	UTMOS (200)	Paired difference	End to end fp32 / int8	Int8 MB	× real time
if_sara							
Kokoro teacher	53.3M	11.81	4.022	–	–	–	–
50k recipe	3.85M	5.95	3.752		–	–	–
wider	6.54M	8.19	3.886	+0.133 ± 0.008	–	–	–
final recipe	3.85M	7.69	3.882		3.924 / 3.917	9.17	27.5
wider	6.54M	10.49	4.005	+0.123 ± 0.007	4.033 / 4.036	11.9	12.3
im_nicola							
Kokoro teacher	53.3M	–	3.867	–	–	–	–
final recipe	3.85M	–	3.696		3.726 / 3.716	9.17	–
wider	6.54M	–	3.755	+0.060 ± 0.004	3.777 / 3.777	11.9	–

Table 8: A generator with twice the channels, trained with the same pool, losses, seed and stages as the adopted decoder. Contrast as in Table 7, not measured for *im_nicola*, whose harmonics our analysis window does not resolve; UTMOS of the decoder alone on the 200 held-out sentences, paired against the adopted decoder of the same recipe (better on 182 of 200 for *if_sara* after either recipe, on 177 of 200 for *im_nicola*); end to end on the 100 sentences of the int8 check; speed with the protocol of Section 6.5, both graphs in one session.

of all but reproduction A is within 0.6 dB of the teacher’s. Between 2 and 4 kHz, however, the teachers’ harmonics stand 10.5 and 11.8 dB above the gaps between them, while the released decoder reaches 4.4 dB, reproduction A 3.9 and our 50k decoder 4.2; between 4 and 6 kHz the students reach 3.2 to 4.1 dB against 6.0 to 6.2. The students fill the space between the upper harmonics with noise, and head distillation does not fix this. In *if_sara* the adversarial stages help, from 3.9 dB after stage one to 5.3 after GAN10 and 6.0 after the 5,000-step GAN3, a change invisible to UTMOS (Table 6) but in line with the author’s informal preference for the adversarial stages.

Probably not missing information. Kokoro’s excitation contains only the fundamental and eight overtones, which end near 2 kHz for a voice at 220 Hz (Section 2.1). Above that, the generator must create the harmonics itself, which the teacher’s 53M-parameter decoder does and the 3.85M student does only in part. This would agree with Paradee’s own localization of its residual buzz to the 2–8 kHz phase of voiced speech [25], although we have not shown that the missing contrast is what listeners hear as the metallic ring of Appendix F.2. We gave the generator the positions of the upper harmonics explicitly, as a zero-initialized input with the cosine and sine of k times the phase of the fundamental for the 48 harmonics below 11 kHz, and trained 5,000 more GAN3 steps from the 50k-recipe *if_sara* decoder with and without it. No band’s contrast differed from the control by more than 0.11 dB (Appendix E). Missing information at the generator’s input does not appear to be the cause.

A longer adversarial stage. The final recipe’s 30,000-step GAN3 does move the contrast: for *if_sara* to 7.69 dB against 5.95 after the 5,000-step stage, about 30% of the gap to the teacher, and in English from 4.18 to 5.45 dB, about 20%. The contrast keeps rising after the 16-sentence UTMOS has levelled off, and two consecutive 5,000-step stages, each with fresh critics and a full cosine decay, do not reproduce the gain (Appendix E). We state the explanation as a hypothesis: what matters is the length and learning-rate schedule of the adversarial stage, including how long the same critics keep training. Even the longer stage leaves most of the gap open.

A wider decoder. Capacity accounts for much of the rest. We trained a decoder whose generator has twice the channels (256, 128 and 64 instead of 128, 64 and 32), with the AdaIN blocks narrowed from 256

to 192 channels. It has 6.54M parameters instead of 3.85M. We trained it for both Italian voices, with both recipes and the same pool, losses, seed and stages as the adopted decoder (Table 8). For `if_sara`, after the 50k recipe its 2–4 kHz contrast is already 8.19 dB, more than the adopted decoder reaches with the longer stage. After the final recipe it is 10.49 dB, which closes 77% of the gap to the teacher instead of 30%. The 1–2 kHz contrast reaches 17.46 dB (teacher 17.57), while the 4–6 kHz contrast moves little (4.42 against 3.99, teacher 6.04). UTMOS rises by 0.12 to 0.13 after either recipe, to 4.005 after the final one, level with the teacher’s 4.022, and the gain holds end to end and in int8. The cost is speed. The wider generator needs 12.6 GFLOP per second of audio instead of 3.36, and its int8 graph runs 12.3 times faster than real time on one CPU thread, against 27.5 for the adopted decoder in the same session: 2.2 times slower. For the intended use, a small model on one CPU thread, we kept the narrow decoder. We did not pursue the wider one further. On `im_nicola` the gain is smaller: $+0.030 \pm 0.004$ after the 50k recipe (better on 143 of 200 sentences) and $+0.060 \pm 0.004$ after the final one (177 of 200, to 3.755 against the teacher’s 3.867), again kept end to end and in int8. Whether it narrows that voice’s harmonic deficit we cannot say, since our contrast measure does not resolve its harmonics. We did not train it in English, and cheaper ways to add capacity where the upper harmonics are formed, for instance in the last generator stage only, remain untested.

8 Related work

Paradee and its lineage. This paper builds directly on Paradee [25, 26], which distils Kokoro-82M [7] half by half into an 8M-parameter student. Its related-work section covers distillation and compression of speech synthesis and small on-device systems; we do not repeat it. Kokoro follows StyleTTS 2 [22], with a phoneme-level BERT [21] on the text side and an iSTFTNet [16] generator in the HiFi-GAN family [17], driven by a harmonic source in the neural source-filter style [37, 38]. Our changes leave all of these components in place. They concern how the decoder’s training target is produced and which teacher signals the student is asked to match.

Distilling intermediate outputs. Knowledge distillation trains a student on outputs of a teacher [10]. Paradee already trains its text side on the teacher’s intermediate pitch, energy and phoneme features. We extend the same idea to the decoder’s last layer and supervise the inverse-STFT head, magnitude and phase. Paradee reports that distilling the teacher’s final layer did not remove its residual buzz [25]. In our experiments it is what lets stage one learn at all, but only once the target is a deterministic function of the student’s input. With the stored full-utterance target, the teacher’s head is not a valid target for the crop (Section 3.4).

Phase losses. Direct phase prediction normally has to deal with phase wrapping. Anti-wrapping losses [1] measure phase error modulo 2π . Kokoro’s head passes $\sin x_p$ to the inverse transform, which confines the phase to $[-1, 1]$ radians, so an L1 loss on that value is unambiguous. The bound is itself a limitation of the architecture. We keep it, because both teacher and student share it.

Spectral and adversarial objectives. The multi-resolution STFT loss [40] and the log-mel loss compare magnitudes only. The multi-period discriminator [17] sees waveforms, and the multi-resolution spectrogram discriminator [15] sees magnitudes. Both are trained with least-squares losses [28] and feature matching [19]. We keep all of them unchanged and show that stage one learns only once a deterministic target and head supervision are added. Paradee’s phase-locking filter belongs to the tradition of phase-locked vocoders [20]. It corrects the phase after synthesis, and we find that it lowers UTMOS once the phase is distilled.

Normalization over the input extent. Instance normalization [36] and its adaptive form [12] compute statistics over each input. They therefore make a convolutional network’s output depend on the whole excerpt it is given. Training on crops while targeting full-utterance outputs creates a train/target mismatch that larger batches do not remove. Rendering the target on the same crop removes it.

Evaluation with predicted MOS. UTMOS [33] is the standard automatic predictor in this line of work, and Paradee uses it too. Recent studies show degradations that it does not penalize and cases where reference-free metrics fail to rank modern systems [2, 11]. Our Italian results (Section 6.3) and our harmonic-contrast measurement (Section 7) give two further examples.

Quantization. We use weight-only int8 storage with per-channel scales and float arithmetic, the scheme surveyed by Krishnamoorthi [18]. Integer-arithmetic inference in the sense of Jacob et al. [14] is outside our scope.

9 Limitations

An automatic metric as the main evidence. Almost every number in this paper is a UTMOS prediction. UTMOS was trained on English ratings, yet we apply it to Italian, where the teachers score only 4.02 and 3.87 and adversarial stages that the author heard as a clear improvement barely move it (Section 6.3). In English it is close to saturation: the released decoder, the Paradee paper’s model and all our English decoders of Table 5 after the adversarial stages lie between 4.36 and 4.47, a range in which UTMOS may not order systems as listeners would [2, 11]. It also misses the harmonic deficit of Section 7. Word error rate and speaker similarity show no loss, but both are close to their ceilings and neither measures naturalness; a multilingual MOS predictor remains to be tried. We have no human MOS, only two blind four-sentence comparisons in English by the author and informal listening (Appendix F.2).

Seeds and a development set. Each configuration was trained with one seed per voice, except the final recipe for `if_sara`, whose stage one depended on the seed while its final decoders did not (Appendix G). The upstream recipe was measured with two English runs and one run per Italian voice. The 200 held-out sentences per voice served for every diagnosis and decision, and their first 16 were monitored during training, so results on them are optimistic in principle; a test set disjoint at the document level should be used for final claims.

Open questions. We did not search over the length of the final adversarial stage or separate its length from its learning-rate schedule. The cause of the harmonic deficit is only partly settled: the wider generator points to capacity, but its contrast was measured on one voice with one seed (on `im_nicola` it raises UTMOS, but our measure cannot resolve that voice’s harmonics), and we have not shown that the deficit is what listeners hear as the metallic ring (Appendix E). The teacher’s style depends on the utterance’s phoneme count, which the student’s constant style cannot represent (Section 4.1); we expect the effect to be small but have not measured it.

Scope. The students imitate Kokoro and inherit its mispronunciations, its prosody and, in Italian, the errors of its conversion from text to phonemes (Appendix D.5). Kokoro’s Italian voices are graded C in its own documentation [8], and the remaining gap to the teacher is larger there (0.14 and 0.17) than in English (0.05). In English the final decoder scores 0.10 above the released one; with one English seed and

a nearly saturated metric, we do not read this as a higher ceiling for the architecture. We trained only the two Italian voices Kokoro ships, so nothing here shows how the recipe behaves on other voices or languages. The English revised recipe used half of the English corpus (5,799 sentences) and the text side of our upstream reproduction, which we did not retrain.

Export and compute. Int8 storage passes our end-to-end check but fails the exporter’s own parity gate (Appendix D.7), and its loss varied between two trainings of the same voice (Appendix G). We measured speed on one desktop CPU only. Unlike Paradee, which trains on a laptop, the revised recipe needs the teacher on a CUDA GPU to render the pool (about one hour and 62 GB of disk per voice) and eleven to twelve hours of shared-GPU training per voice, six of them for the longer final adversarial stage (Appendix G).

10 Conclusion

Paradee showed that Kokoro-82M can be distilled into an 8M-parameter student, and its openly released code, weights and paper made this follow-up possible. Its recipe trains the decoder’s first stage on a target that the student cannot predict: a crop of a full-utterance teacher render, made with statistics and random excitation that the crop does not contain. How good a run ends up then varies between runs and between voices. Rendering each training crop with the frozen teacher and the same excitation makes the target deterministic. It also makes the teacher’s inverse-STFT head available as a target, and supervising that head is what turns stage one into a useful starting point. With the same 3.85M decoder and the same adversarial stages, the revised recipe reaches the released decoder’s English quality and trains Kokoro’s two Italian voices from scratch. A pool of pre-rendered crops keeps its cost below that of a live teacher, and the phase-locking filter is no longer needed. With a final adversarial stage of 30,000 instead of 5,000 steps, the decoders score 4.47 in English and 3.88 and 3.70 in Italian, with no loss end to end and at most 0.01 in int8. What remains is a deficit in harmonic contrast above 2 kHz that every small decoder shares and that UTMOS does not register. It does not appear to be caused by missing information at the generator’s input. The longer adversarial stage closes 20 to 30% of it, and for `if_sara` a generator with twice the channels closes 77%, at 2.2 times the CPU time, which is why we did not adopt it. Proper listening tests are still missing, and our seed check covers one voice: there, the first stage varied with the seed, but the final decoders did not. We hope the diagnosis saves others the search, and that the release makes the result easy to check. The training procedure is available at <https://github.com/MithrilMan/teacher-in-the-loop>, the Italian and English weights at <https://huggingface.co/MithrilMan/teacher-in-the-loop>, and audio samples and an in-browser demo at <https://huggingface.co/spaces/MithrilMan/teacher-in-the-loop>.

Acknowledgements

We thank Sahil Mahendrakar for releasing Paradee’s code, weights and paper openly; this work builds directly on them. We also thank the Kokoro-82M authors (hexgrad) for making the teacher model openly available.

References

- [1] Yang Ai and Zhen-Hua Ling. Neural speech phase prediction based on parallel estimation architecture and anti-wrapping losses. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023. arXiv:2211.15974.
- [2] Antonis Asonitis, Juan Pablo Zuluaga Gomez, Francesco Verdini, Aref Farhadipour, Marzieh Razavi, Pierre-Edouard Honnet, and Vijeta Avijeet. The limits of reference-free speech quality metrics as evaluators and rewards on modern text-to-speech. arXiv preprint arXiv:2609.13150, 2026.
- [3] Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, Jian Wu, Long Zhou, Shuo Ren, Yanmin Qian, Yao Qian, Jian Wu, Michael Zeng, Xiangzhan Yu, and Furu Wei. WavLM: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6): 1505–1518, 2022. arXiv:2110.13900.
- [4] Alain de Cheveigné and Hideki Kawahara. YIN, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4):1917–1930, 2002.
- [5] W. J. Dixon and A. M. Mood. The statistical sign test. *Journal of the American Statistical Association*, 41(236):557–566, 1946.
- [6] eSpeak NG contributors. eSpeak NG: an open source speech synthesizer. <https://github.com/espeak-ng/espeak-ng>, 2026. GPL-3.0-or-later; accessed October 2026.
- [7] hexgrad. Kokoro-82M. Hugging Face model card, <https://huggingface.co/hexgrad/Kokoro-82M>, 2025. Version 1.0 (released 27 January 2025), Apache-2.0. Revision used: f3ff3571791e39611d31c381e3a41a3af07b4987.
- [8] hexgrad. Kokoro-82M voices (VOICES.md). <https://huggingface.co/hexgrad/Kokoro-82M/blob/main/VOICES.md>, 2025. Lists 54 voices; the Italian voices if_sara and im_nicola are graded C overall; revision f3ff3571791e39611d31c381e3a41a3af07b4987, accessed October 2026.
- [9] hexgrad. Misaki: a G2P engine designed for Kokoro models. <https://github.com/hexgrad/misaki>, 2026. Apache-2.0; version 0.9.4 used, with its eSpeak NG backend for Italian; accessed October 2026.
- [10] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531, 2015. NIPS 2014 Deep Learning Workshop.
- [11] Wen-Chin Huang and Tomoki Toda. Attacking UTMOS: Probing the robustness of a speech quality assessment model. In *Proc. IEEE Spoken Language Technology Workshop (SLT)*, 2026. arXiv:2606.31105.
- [12] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2017. arXiv:1703.06868.
- [13] ITU-R. Recommendation ITU-R BS.1534-3: Method for the subjective assessment of intermediate quality level of audio systems. International Telecommunication Union, Radiocommunication Sector, 2015.

- [14] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. arXiv:1712.05877.
- [15] Won Jang, Dan Lim, Jaesam Yoon, Bongwan Kim, and Juntae Kim. UnivNet: A neural vocoder with multi-resolution spectrogram discriminators for high-fidelity waveform generation. In *Proc. Interspeech*, 2021. arXiv:2106.07889.
- [16] Takuhiro Kaneko, Kou Tanaka, Hirokazu Kameoka, and Shogo Seki. iSTFTNet: Fast and lightweight mel-spectrogram vocoder incorporating inverse short-time Fourier transform. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022. arXiv:2203.02395.
- [17] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. HiFi-GAN: Generative adversarial networks for efficient and high fidelity speech synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. arXiv:2010.05646.
- [18] Raghuraman Krishnamoorthi. Quantizing deep convolutional networks for efficient inference: A whitepaper. arXiv preprint arXiv:1806.08342, 2018.
- [19] Kundan Kumar, Rithesh Kumar, Thibault de Boissière, Lucas Geste, Wei Zhen Teoh, Jose Sotelo, Alexandre de Brébisson, Yoshua Bengio, and Aaron C. Courville. MelGAN: Generative adversarial networks for conditional waveform synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. arXiv:1910.06711.
- [20] Jean Laroche and Mark Dolson. Improved phase vocoder time-scale modification of audio. *IEEE Transactions on Speech and Audio Processing*, 7(3):323–332, 1999.
- [21] Yinghao Aaron Li, Cong Han, Xilin Jiang, and Nima Mesgarani. Phoneme-level BERT for enhanced prosody of text-to-speech with grapheme predictions. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023. arXiv:2301.08810.
- [22] Yinghao Aaron Li, Cong Han, Vinay S. Raghavan, Gavin Mischler, and Nima Mesgarani. StyleTTS 2: Towards human-level text-to-speech through style diffusion and adversarial training with large speech language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2306.07691.
- [23] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations (ICLR)*, 2017. arXiv:1608.03983.
- [24] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019. arXiv:1711.05101.
- [25] Sahil Mahendrakar. Paradee: Distilling Kokoro-82M into an 8M-parameter single-voice text-to-speech model. arXiv preprint arXiv:2610.06817, 2026. Submitted 5 October 2026.
- [26] Sahil Mahendrakar. paradee: an 8M-parameter English text-to-speech model distilled from Kokoro-82M. <https://github.com/sahilmahendrakar/paradee>, 2026. Apache-2.0. Training code revision 9c8b4d7504cbee7e64de2d0341bb690f0b0ab708.
- [27] Sahil Mahendrakar. Paradee-8M-v1.0. Hugging Face model card, <https://huggingface.co/sahilmahendrakar/Paradee-8M-v1.0>, 2026. Weights revision 8f34b01ef8adcb0bec470b89bf2fd62a7b2369e6.

- [28] Xudong Mao, Qing Li, Haoran Xie, Raymond Y. K. Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2017. arXiv:1611.04076.
- [29] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations (ICLR)*, 2017. arXiv:1609.07843.
- [30] MithrilMan and kaixuanluo. af_heart reproduction on CUDA: fresh original-CLI run falls well below released decoder quality. GitHub issue #3, <https://github.com/sahilmahendrakar/paradee/issues/3>, 2026. Opened 10 October 2026; independent reproduction comment 11 October 2026.
- [31] ONNX Runtime developers. ONNX Runtime. <https://onnxruntime.ai/>, 2026. Microsoft; cross-platform inference engine; version 1.30.0 used; accessed October 2026.
- [32] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *Proc. International Conference on Machine Learning (ICML)*, volume 202 of *PMLR*, 2023. arXiv:2212.04356.
- [33] Takaaki Saeki, Detai Xin, Wataru Nakata, Tomoki Koriyama, Shinnosuke Takamichi, and Hiroshi Saruwatari. UTMOS: UTokyo-SaruLab system for VoiceMOS Challenge 2022. In *Proc. Interspeech*, 2022. arXiv:2204.02152.
- [34] Student. The probable error of a mean. *Biometrika*, 6(1):1–25, 1908.
- [35] tarepan. SpeechMOS: Easy-to-use speech MOS predictors. <https://github.com/tarepan/SpeechMOS>, 2026. MIT license; utmos22_strong. Source commit ed25eachbfa42b99156c36ebec67a733b5dbb9b79, checkpoint utmos22_strong_step7459_v1.pt; accessed October 2026.
- [36] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022, 2016.
- [37] Xin Wang and Junichi Yamagishi. Neural harmonic-plus-noise waveform model with trainable maximum voice frequency for text-to-speech synthesis. In *Proc. ISCA Speech Synthesis Workshop (SSW)*, 2019. arXiv:1908.10256.
- [38] Xin Wang, Shinji Takaki, and Junichi Yamagishi. Neural source-filter-based waveform model for statistical parametric speech synthesis. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019. arXiv:1810.11946.
- [39] Wikimedia Foundation. wikimedia/wikipedia (Hugging Face dataset), configuration 20231101.it. <https://huggingface.co/datasets/wikimedia/wikipedia>, 2023. CC BY-SA 3.0 and GFDL. Revision b04c8d1ceb2f5cd4588862100d08de323dccbbaa.
- [40] Ryuichi Yamamoto, Eunwoo Song, and Jae-Min Kim. Parallel WaveGAN: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020. arXiv:1910.11480.

Setting	Upstream recipe	Revised recipe (this paper)
Teacher	Kokoro-82M v1.0	same; voices <code>af_heart</code> , <code>if_sara</code> , <code>im_nicola</code>
Training sentences	11,799 (English, our reproduction)	5,799 (English, 12 of 24 shards); 12,000 (Italian)
Held-out sentences	200	200 (first 16 monitored during training)
Decoder	preset A: 3.85M, AdaIN 256 ch., generator 128 ch., kernels 3/7/11, 16-d learned style	unchanged
Excitation mixer	random initialization, never updated	copied from the teacher, not updated
Text side	<code>s+mlp</code> , 4.23M; 8,000 steps, batch 32, lr 5×10^{-4}	unchanged (existing students reused)
Crop	64 frames (1.6 s)	same
Stage-one target	stored full-utterance teacher waveform, cropped	teacher re-rendered on the crop with the student's excitation; read from a pool
Pool	–	150,000 random crops per voice; head $(x_m, \sin x_p)$ and excitation in fp16; teacher fp16 autocast (English) or fp32 (Italian)
Stage-one loss	log-mel L1 + MR-STFT (spectral convergence + log-magnitude L1; FFT 512/1024/2048, hop FFT/4)	same + $1 \cdot L1(x_m) + 1 \cdot L1(\sin x_p)$ on the head
Stage-one schedule	50,000 steps, batch 16, lr 2×10^{-4} , 1,000-step warmup + cosine	30,000 steps from scratch (batch 16, lr 2×10^{-4} , warmup 1,000 + cosine to 0, seed 1), then 20,000 steps warm restart (fresh optimizer, peak lr 10^{-4} , warmup 1,000 + cosine)
Optimizer	AdamW, $\beta = (0.8, 0.99)$, weight decay 0.01, grad. clip 5	same
Critics	multi-period (MPD, periods 2/3/5/7/11) + multi-resolution spectrogram (MRD, FFT 512/1024/2048)	same
Adversarial losses	least-squares GAN (LSGAN) + $2 \times$ feature matching	same
GAN10	5,000 steps, batch 8, lr 2×10^{-4} , warmup 1,000 + cosine, spectral terms $\times 10$	5,000 steps, batch 8, lr 2×10^{-4} , warmup 1,000 + cosine, all reconstruction terms (incl. head) $\times 10$, pool targets
GAN3	5,000 steps, batch 8, lr 2×10^{-4} , warmup 1,000 + cosine, spectral terms $\times 3$, first 8 shards	30,000 steps (5,000 in the 50k recipe), batch 8, lr 2×10^{-4} , warmup 1,000 + cosine over the stage, all reconstruction terms $\times 3$, whole pool
Stage transitions	generator weights only; fresh critics and optimizers	same
Critic optimizer	AdamW, constant lr 2×10^{-4}	same
Excitation STFT	on CPU	on GPU, DC/Nyquist imaginary part set to +0
Precision	fp32	fp32 student; fp16 teacher (English pool)
Post-filter	phase locking, 2–8 kHz, 33 frames	none
Int8 storage	one scale along axis 0 (per LSTM direction; per MatMul input row; int8 biases)	one scale per output channel (per gate row for LSTMs); biases, norms, windows in float32

Table 9: Every setting of the upstream recipe and of the revised recipe.

A Full recipe and hyperparameters

Procedure. For one voice:

1. Synthesize the teacher corpus as in Paradee: phonemes, durations, F_0 , N , phoneme features and waveform for every sentence. Hold out 200 sentences. The waveforms are needed only for evaluation.
2. Train the text side with the upstream recipe, or reuse an existing one.
3. Render the pool. Draw 150,000 random 64-frame crops from the training sentences. For each crop, draw one excitation from the teacher’s source module, run the frozen teacher on the crop with it, and store the head $(x_m, \sin x_p)$, the excitation and the crop position.
4. Stage one: initialize the student decoder from scratch, copy the teacher’s source module into it, and train for 30,000 steps on Equation (4) with targets rebuilt from the pool. Then run 20,000 more steps after a warm restart.
5. GAN10 for 5,000 steps, then GAN3 for 30,000 steps, each from the previous generator, with fresh critics and optimizers.
6. Export to ONNX without the post-filter. Store weights as int8 per output channel. Check the int8 graph against the float32 graph end to end on held-out sentences.

The released repository provides one command per step: <https://github.com/MithrilMan/teacher-in-the-loop>.

The pool in detail. Rendering took about 15 ms per crop with the teacher under fp16 autocast (English) and about 25 ms per crop in fp32 (Italian), roughly one hour per voice. The fp16 teacher differs from fp32 by a log-mel L1 of 0.02 with the same UTMOS. A bf16 teacher was rejected, with a log-mel L1 of 0.20 and 0.08 lower UTMOS. For the English pool the rebuilt waveform matched the teacher’s output exactly on a check batch. For the Italian pools, the fp32 teacher’s head stored in fp16 rebuilds the target to within a log-mel L1 of 0.007. Each step draws its batch of crops from the pool without replacement within the batch. A 50,000-step stage one at batch 16 draws 800,000 crops, so it sees each stored crop about five times, always with the same excitation draw. The live teacher draws a new excitation each time. Variety comes from the 150,000 crops, and in our runs this did not limit quality (Section 6).

The excitation STFT. Upstream computes the STFT of the excitation on the CPU, which costs a device round trip at every step. We compute it on the GPU and set the imaginary part of the DC and Nyquist bins to +0, so that their phase is exactly 0 or π as on the CPU. The phase then agrees with the CPU version except for six values per million that fall on the branch cut. The teacher’s output differs by a log-mel L1 of 0.001, and the step is twice as fast.

B Reproduction of the upstream recipe

The run. Each command ran in a fresh process with the published defaults. The corpus came from the first 12,000 candidate sentences, of which 11,999 were accepted; the first 200 were held out and the remaining 11,799 used for training. The waveform targets were regenerated on CPU with the original script and per-row seeds. We then trained the text side (8,000 steps), decoder stage one (50,000 steps), GAN10 and GAN3 (5,000 steps each) on one NVIDIA RTX 4080 SUPER. Each GAN stage started from the previous generator with fresh critics and optimizers, as the published commands do.

Item	Value
Upstream training code	9c8b4d7504cbee7e64de2d0341bb690f0b0ab708 [26] (only edit: generator device mps→cuda)
Released weights compared	Hugging Face revision 8f34b01ef8adcb0bec470b89bf2fd62a7b2369e6 [27]
Teacher	Kokoro-82M revision f3ff3571791e39611d31c381e3a41a3af07b4987 [7]
Corpus	12,000 candidates, 11,999 accepted (23.90 h); 11,799 training, 200 held out
Software	Python 3.12.15, PyTorch 2.14.0+cu130, torchaudio 2.11.0, CUDA 13.0, kokoro/misaki 0.9.4
Precision	fp32, TF32 disabled (NVIDIA_TF32_OVERRIDE=0), no mixed precision
Commands	corpus generation; CPU target regeneration; <code>train_text.py s+mlp</code> (8,000 steps); <code>train_decoder.py A</code> (50,000 steps); GAN10 and GAN3 via <code>--init-from</code> (5,000 steps each)
Wall clock	about 10 h 03 min, including generation and CPU regeneration
200 sentences, CPU renders, teacher conditioning	released 4.371, reproduction A 1.939 (phase lock: 4.382 and 2.264)
200 sentences, CPU renders, full model	released 4.395, reproduction A 1.911 (phase lock: 4.409 and 2.232)
200 sentences, stage one only	reproduction A 1.235 (teacher conditioning), 1.234 (full)
Cached teacher targets	4.521

Table 10: Reproduction A in detail. The CPU renders use the upstream classes and the upstream UTMOS script. Their values agree with the GPU-scored lab numbers of Table 5.

CPU evaluation. In a separate CPU evaluation with the upstream code (Table 10), reproduction A scored lower than the released model on every one of the 200 sentences. This held both with the teacher’s inputs (1.94 against 4.37) and through each model’s own text side (1.91 against 4.39). Paradee’s phase-locking filter raised A to 2.26 with the teacher’s inputs and to 2.23 through its text side, which does not close the gap. The long-term average spectra of these decoders showed no single tonal artifact that could explain the gap.

Reproduction B. Reproduction B, run earlier through our own training wrapper, follows the same command sequence with the Python, NumPy and PyTorch seeds set to 42. It used its own teacher cache and the default floating-point policy, and its full pipeline scored 2.87 with its own text side. The two reproductions therefore differ in more than the seed, and we use them only to show that the outcome of the upstream recipe varies widely between runs, not to attribute the variation to one factor.

Numerics. On one fixed batch, default CUDA arithmetic with TF32 convolutions changed the spectral gradient by 25% relative to CPU, against 0.5% with TF32 disabled. About 95% of the difference in the waveform gradient came from spectral values between 10^{-5} and 10^{-4} , just above the floor of the logarithm. A paired 2,000-step continuation that changed only the TF32 setting, however, moved UTMOS by 0.007 on eight sentences (2.898 with TF32, 2.904 without). The user of the upstream repository who reported a reconstruction score of about 1.2 [30] used another software stack (PyTorch 2.4.1, CUDA) and another English corpus.

Why not fine-tune the released decoder? Any non-adversarial objective moves the GAN-trained released decoder away from its optimum, including ours. With the teacher’s mixer swapped in (start 4.19), 500 steps with the teacher in the loop and the spectral loss gave 4.08, and adding the head losses of Section 4.2 gave 3.99. Fine-tuning a finished decoder therefore cannot tell us which objective works. We evaluate every objective by training from scratch. We also do not build the Italian decoders from the

English one: the goal is a recipe that trains any voice from scratch, and UTMOS cannot detect timbre that an English initialization might leave behind.

C Evaluation details

UTMOS. SpeechMOS [35] at source commit ed25eachb. The checkpoint is utmos22_strong_step7459_v1.pt (SHA-256 38aa51ab. . .). Lab scoring ran on the GPU. The teacher target of the first 16 English held-out sentences scores 4.52 there, as it does on the CPU.

Spectral distances and periodicity. The two terms of the stage-one objective are the log-mel L1 (FFT 1024, hop 256, 80 bands) and the multi-resolution STFT term (FFT 512/1024/2048), both against the full-utterance teacher target. A periodicity measure, the normalized autocorrelation at the F_0 lag over voiced 40 ms frames, is reported as a description only.

Renderers. Decoders render each held-out sentence from Kokoro’s own features for the whole utterance, with excitation seed $1000 + i$ for sentence i . A decoder’s output is compared with the teacher’s stored waveform over their common length.

Paired statistics. For two systems on the same n sentences we report $\bar{d} \pm \text{SE}$, with $\text{SE} = s_d/\sqrt{n}$, and write $z = \bar{d}/\text{SE}$. Two-sided p -values use the normal approximation to the paired t -test. Where we ignore the pairing, SE combines the standard errors of the two means, each taken as the 95% half-width divided by 1.96, which is conservative. Sign tests are exact. All values are in `data/paired_tests.csv`, computed by `data/stats.py`.

Harmonic contrast. Each decoder renders 40 held-out sentences from the teacher’s own features, so its pitch is exact by construction. On each output we estimate F_0 with YIN [4] and keep the voiced hops within 25 dB of the loudest frames. For each hop we compute a 4096-point spectrum of a 40 ms Hann window. For every harmonic k in a band we take the peak level near kF_0 and the mean level near $(k + \frac{1}{2})F_0$, both within $\pm 10\%$ of F_0 . The contrast is the mean peak level minus the mean valley level, in dB, averaged over hops and sentences. We also report the harmonics-to-noise ratio (HNR) from the normalized autocorrelation at the pitch period. We do not interpret the measure for `im_nicola`. In a low male voice the harmonics lie about half as far apart as in the female voices, and the 40 ms window, together with pitch changes within it, blurs adjacent harmonics into each other. Even its teacher shows only 13.4 dB below 1 kHz and 2.8 dB between 2 and 4 kHz (`data/harmonic_contrast.csv`).

Hardware. All training ran on one shared server with a modified NVIDIA RTX 4080 SUPER (32 GB), eight virtual Xeon Gold 6338 cores and 43 GB of RAM. Each job ran in a container limited to four CPU cores and 20 GB of RAM. The GPU was shared with other services and usually with one to three of our own jobs, so the wall-clock times in this paper are only indicative. The software was Python 3.12, PyTorch 2.14 with CUDA 13.0, torchaudio 2.11, and kokoro and misaki 0.9.4. The students are trained in fp32.

Traceability. Every number in this paper is in a CSV file in the folder `paper/data/` of the code repository (<https://github.com/MithrilMan/teacher-in-the-loop>), called `data/` in this paper, and `data/SOURCES.md` names the run and evaluation each one comes from. Figures read their data from `data/fig_*.csv`, which `data/make_figure_data.py` derives from the in-training evaluations in `data/curves_raw.csv`.

D Additional results

D.1 Stage-one arms

Arm	Change from the upstream stage one	1k	2k	3k	6k	10k
s0	none (stored full-utterance target)	1.26	1.25	1.24	1.23	–
s1	teacher in the loop	1.25	1.23	1.24	–	–
s3	GAN from step 0, stored target	1.26	1.24	1.24	–	–
s2	teacher in the loop + head losses	1.24	1.37	1.63	2.56	2.72
s5	s2 with phase weight 5, fp16 teacher	1.26	1.52	2.17	2.53	2.65
s6	s5 with GAN from step 0	1.26	1.72	2.27	3.29	3.59

Table 11: UTMOS (16 held-out rows) of stage-one arms trained from scratch for 10,000 steps with a cosine schedule (seed 1). In s3 and s6 the reconstruction terms have weight 45. “–”: arm stopped once flat (s1 at 4k: 1.23).

Table 11 gives selected values of the curves in Figure 2 (all points are in `data/fig_stage1_arms.csv`). The adversarial arm on the stored target (s3) ran at 1.4 s per step. The phase weight of 5 (s5) gave 2.17 against 1.63 for s2 at 3,000 steps.

D.2 English on 64 sentences

Decoder (64 held-out rows, teacher conditioning)	UTMOS	log-mel L1	MR-STFT	periodicity
Teacher target	4.52	–	–	0.598
Released decoder	4.363	0.543	0.883	0.610
Ours: stage one 30k	3.934	0.548	0.883	0.594
Ours: stage one 30k → GAN3	4.343	0.600	0.933	0.597
Ours: stage one 30k → GAN10	4.344	0.583	0.919	0.600
Ours: stage one 30k → GAN10 → GAN3	4.365	0.582	0.916	0.597
Ours, short stage one: 10k (s2) → GAN10 → GAN3	4.021	0.704	1.026	0.618
Reproduction A: stage one 50k	1.235	0.518	0.976	0.574
Reproduction A: stage one → GAN10 → GAN3	1.866	0.637	1.055	0.585
Reproduction B: stage one 50k	1.976	0.528	0.980	0.572
Reproduction B: stage one → GAN10 → GAN3	2.848	0.602	1.018	0.583

Table 12: English `af_heart`, 64 held-out sentences. With the same architecture and the same adversarial stages, the revised recipe matches the released decoder. Our two reproductions of the upstream recipe remain below it. The spectral distances still do not rank decoders by quality.

Table 12 gives the English decoders of the 30k recipe on 64 held-out sentences, with the two terms of the stage-one objective and the periodicity measure of Appendix C. The spectral distances confirm Section 3.2 once more. Our complete 30k-recipe decoder has a higher log-mel distance to the target (0.582) than the released decoder (0.543), and also higher than our own stage-one decoder (0.548), which scores 0.43 lower. The adversarial stages trade spectral distance for quality. The periodicity of this decoder (0.597) is closer to the teacher’s (0.598) than the released decoder’s (0.610), although we treat this measure as descriptive only.

On the 16 monitored sentences, GAN10 raised the 10,000-step s2 decoder (2.72) to 3.59, our first decoder above the paper’s stage-one number (2.98), and GAN3 raised it to 3.84 after 2,000 steps. The 30,000-step stage one (s7) rose from 2.99 at 10,000 steps to 3.55 at 15,000 and 3.88 at 30,000. The tests behind the 50k

comparison of Section 6.2, unpaired in English and paired in Italian, are in `data/paired_tests.csv`. Before the adversarial stages, the author could not tell the 30k and 50k stage-one decoders apart by ear.

D.3 The final adversarial stage during training

Voice	Teacher	UTMOS, GAN3 of		Paired difference	Better	Wall clock, GAN3 of	
		5k steps	30k steps			5k steps	30k steps
af_heart	4.521	4.414	4.467	$+0.053 \pm 0.002$	197/200	1 h 04 min	6 h 41 min
if_sara	4.022	3.752	3.882	$+0.130 \pm 0.007$	187/200	0 h 51 min	5 h 25 min
im_nicola	3.867	3.601	3.696	$+0.094 \pm 0.003$	197/200	0 h 51 min	6 h 30 min

Table 13: The last adversarial stage run for 5,000 steps (the 50k recipe) and for 30,000 steps (the final recipe), from the same GAN10 decoder. UTMOS on the 200 held-out sentences, teacher conditioning, the same excitation seeds; paired difference as mean $\pm$ standard error. Wall clock on the shared GPU, indicative only.

Table 13 pairs the two lengths of the last stage for each voice.

The 16 monitored sentences show how the 30,000-step GAN3 of the final recipe evolves (Figure 3). For `if_sara` the score rose from 3.78 at the start of the stage to 3.86 after 5,000 steps and 3.95 after 10,000, and then stayed between 3.94 and 3.98 from 12,500 steps on. For `im_nicola` it rose from 3.60 to 3.67 after 5,000 steps and 3.74 after 15,000, and then stayed at 3.75 to 3.76. In English it rose from 4.39 to 4.41 after 5,000 steps and 4.44 after 10,000, and reached 4.46 at 20,000 and 4.47 at 30,000 steps. Thirty thousand steps is therefore past the point where the 16-sentence score levels off for the Italian voices. Each run started from the same GAN10 decoder as the 50k recipe, with fresh critics and the same losses, batch, warmup and seed. Only the cosine decay was stretched over the 30,000 steps, and the pools were rendered again with the same settings. In English the final decoder scores 4.467 against 4.371 for the released decoder on the same sentences, a difference of 0.096 with $z = 15$ even when the pairing is ignored.

D.4 End to end

Voice	Decoder	End to end	Teacher-forced	Difference	Length ratio
af_heart	released decoder	4.391	4.371	$+0.020 \pm 0.004$	0.988
af_heart	ours, 50k recipe	4.424	4.414	$+0.010 \pm 0.003$	0.988
af_heart	ours, final recipe	4.474	4.467	$+0.007 \pm 0.002$	0.988
if_sara	upstream recipe	2.201	2.205	-0.004 ± 0.011	0.987
if_sara	ours, 50k recipe	3.766	3.752	$+0.013 \pm 0.009$	0.987
if_sara	ours, final recipe	3.895	3.882	$+0.013 \pm 0.008$	0.987
im_nicola	upstream recipe	3.543	3.535	$+0.009 \pm 0.005$	0.991
im_nicola	ours, 50k recipe	3.609	3.601	$+0.007 \pm 0.005$	0.991
im_nicola	ours, final recipe	3.700	3.696	$+0.004 \pm 0.005$	0.991

Table 14: The existing text students connected to each decoder, 200 held-out sentences. “End to end” starts from phonemes; “teacher-forced” gives the same decoder Kokoro’s own values. Difference: paired mean $\pm$ standard error. Length ratio: total predicted duration over the teacher’s.

The text sides had been trained with the upstream text recipe: in English the one from reproduction A, in Italian the ones of our earlier students. Table 14 connects them to each decoder. With reproduction A’s text side, the released decoder scores 4.391, close to the 4.395 of the complete released model (Appendix B).

D.5 Intelligibility and speaker identity

Voice	Audio	WER (%)		Similarity, same sentence		Similarity, voice centroid
		teacher-forced	end to end	teacher-forced	end to end	
af_heart	Kokoro teacher	3.32		–	–	0.990
	released decoder	3.43	3.20	0.985	0.984	0.982
	ours, 50k recipe	3.40	3.12	0.991	0.989	0.986
	ours, final recipe	3.49	3.29	0.993	0.990	0.986
if_sara	Kokoro teacher	4.09		–	–	0.994
	ours, 50k recipe	3.98	4.09	0.989	0.989	0.985
	ours, final recipe	4.11	3.94	0.996	0.995	0.992
im_nicola	Kokoro teacher	4.45		–	–	0.993
	ours, 50k recipe	4.16	4.28	0.994	0.993	0.990
	ours, final recipe	4.33	4.33	0.995	0.993	0.989

Table 15: Intelligibility and speaker identity on the 200 held-out sentences. WER: Whisper large-v3-turbo with the language forced, against the normalized sentence text. Similarity: cosine of WavLM x-vectors, against the teacher’s audio of the same sentence (standard errors 0.0001 to 0.0004) and, end to end, against the centroid of 40 training sentences of the teacher. “Teacher-forced” and “end to end” as in Table 14.

Whisper large-v3-turbo [32], with the language forced, transcribes every render, and we compute the word error rate against the sentence text, both normalized. Speaker similarity is the cosine between x-vectors of a WavLM model fine-tuned for speaker verification (microsoft/wavlm-base-plus-sv) [3]. No student differs from its teacher in word error rate beyond noise: the mean per-sentence difference from the teacher is within 1.7 standard errors of zero for every decoder and condition (data/wer_similarity.csv), and the students’ rates of 3.1 to 4.3% are close to the teachers’ 3.3, 4.1 and 4.5%. Speaker similarity is close to 1 for every system, but it orders them consistently (Table 15). In English, our decoders are closer to the teacher than the released one (0.993 and 0.991 against 0.985 for the same sentence, teacher-forced). For if_sara the longer GAN3 raises it from 0.989 to 0.996, while for im_nicola the two decoders are level. End to end, similarity is at most 0.003 lower than teacher-forced.

The Italian students use the same conversion from text to phonemes as the teacher, Misaki’s eSpeak NG backend, and inherit its errors. We did not measure these errors separately. The teachers’ own word error rates, 4.1% (if_sara) and 4.5% (im_nicola) against 3.3% for af_heart, mix phoneme, synthesis, recognition and normalization errors.

D.6 The phase-locking post-filter

Table 16 measures Paradee’s filter with our implementation of the upstream filter, in the same configuration as the upstream export. It helps decoders whose phase was never supervised directly, and the worse their phase, the more it helps: +0.32 for reproduction A, +0.03 for B and +0.01 for the released decoder. In English it also helps our stage-one decoders before adversarial training. It lowers the score of every decoder trained with head distillation after its adversarial stages, by 0.03 to 0.04 in English and by 0.08 to 0.22 in Italian, and the final recipe’s decoders lose as much as the 50k ones or more. In Italian it also lowers our stage-one decoders and most earlier decoders. The single exception is the adapted English decoder for im_nicola. Since the filter is not trainable and only helps decoders that the revised recipe no longer produces, we removed it from the export. Exports made before this change still contain it.

Voice	Decoder	UTMOS change with the filter
af_heart	reproduction A, full upstream recipe	+0.320 ± 0.010
af_heart	reproduction B, full upstream recipe	+0.027 ± 0.007
af_heart	released decoder	+0.012 ± 0.003
af_heart	ours, stage one 30k (no GAN)	+0.306 ± 0.008
af_heart	ours, stage one 50k (no GAN)	+0.188 ± 0.006
af_heart	ours, 30k recipe	−0.032 ± 0.003
af_heart	ours, 50k recipe	−0.035 ± 0.003
af_heart	ours, final recipe	−0.035 ± 0.002
if_sara	released English decoder adapted with GAN3	−0.063 ± 0.005
if_sara	ours, stage one 30k (no GAN)	−0.186 ± 0.004
if_sara	ours, 50k recipe	−0.191 ± 0.005
if_sara	ours, final recipe	−0.222 ± 0.005
im_nicola	upstream recipe	−0.075 ± 0.003
im_nicola	released English decoder adapted with GAN3	+0.042 ± 0.003
im_nicola	ours, stage one 30k (no GAN)	−0.085 ± 0.003
im_nicola	ours, 50k recipe	−0.079 ± 0.003
im_nicola	ours, final recipe	−0.091 ± 0.003

Table 16: Mean paired UTMOS change from Paradee’s phase-locking filter (2–8 kHz, 33-frame smoothing, as in the upstream export), 200 held-out sentences, mean ± standard error.

Model	Stored as int8	Size (MB)	Δ UTMOS	Parity gate
af_heart, final recipe	all	9.17	−0.006 ± 0.001	fails (2.45 vs 1.31 dB)
if_sara, final recipe	all	9.17	−0.007 ± 0.011	fails (1.83 vs 1.37 dB)
im_nicola, final recipe	all	9.17	−0.010 ± 0.005	fails (1.46 vs 1.04 dB)
af_heart, 50k recipe	all	9.17	+0.006 ± 0.002	fails (2.25 vs 1.34 dB)
if_sara, 50k recipe	all	9.17	−0.000 ± 0.011	fails (1.50 vs 1.17 dB)
im_nicola, 50k recipe	all	9.17	−0.008 ± 0.005	fails (1.22 vs 0.89 dB)
if_sara, 30k recipe	all	9.17	−0.024 ± 0.011	fails (1.57 vs 1.18 dB)
if_sara, 30k recipe	text side only	20.48	−0.014 ± 0.010	fails
if_sara, 30k recipe	decoder only	21.46	−0.011 ± 0.001	passes (0.69 vs 1.18 dB)
if_sara, 30k recipe	all but 3 decoder groups	9.40	−0.016 ± 0.010	fails

Table 17: Int8 weight storage, ONNX Runtime on the CPU, end to end on 100 held-out sentences (paired, same inputs and seeds). Δ UTMOS: int8 graph minus float32 graph. The float32 graph is 32.8 MB. The parity gate compares median log-spectral distance with 1.5 times the float32 graph’s own run-to-run distance. It fails on the distance criterion. The gate compares only cases in which every token duration is unchanged, int8 rounding moves a few durations by one frame (1 to 4 of 546 tokens), and this leaves one or two of the three built-in cases to compare.

D.7 Int8 export, size and speed

The two schemes. Like upstream, we store large weights as int8 and dequantize them in the graph, so the arithmetic stays in float32 [14, 18]. The upstream exporter, however, scaled every tensor along its first axis. For ONNX LSTM weights this meant one scale per direction, shared by all four gates. For matrix-multiplication weights stored as [input, output], it meant one scale per *input* row. Biases were also stored as int8, and the filter’s overlap-add window had a single scale. Our exporter chooses one scale per output channel from the operator that consumes each weight, with one scale per gate row and direction for LSTMs. Biases, normalization parameters and windows stay in float32. The export compares the int8 graph with the float32 graph on identically seeded noise. It fails if more than 2% of token durations change, or if the median log-spectral distance exceeds 1.5 times that of the float32 graph re-run with another seed.

Per-channel scales. We first measured the per-channel int8 scheme on the earlier Italian models, which were trained with the upstream recipe. On 52 text chunks (5,441 tokens), the number of token durations that int8 storage changes fell from 27 to 9 for *if_sara* and from 35 to 27 for *im_nicola*. The median log-spectral distance to the float32 graph fell from 3.45 to 2.86 dB and from 3.74 to 3.40 dB. For comparison, re-running the float32 graph with another random excitation gives 4.70 and 3.38 dB. On chunks where both schemes keep every duration, the new scheme is closer to float32 in 123 of 127 cases, pooled over these two and three other earlier models. The graph grows from 9.06 to 9.25 MB, and one 8.85-second sentence took 410 ms instead of 395 ms with two CPU threads.

The new decoders. Table 17 shows the int8 graphs of the new students. For the final recipe the cost is at most 0.01: -0.006 ± 0.001 in English, where the paired differences are small enough for this to be significant, and -0.007 ± 0.011 and -0.010 ± 0.005 in Italian, the latter nominally significant ($p = 0.04$). For the 50k recipe it is within noise for the two Italian voices and slightly positive in English ($+0.006 \pm 0.002$). For the 30k *if_sara* model it is -0.024 ± 0.011 . To find where this comes from, we quantized one half at a time. The decoder alone costs -0.011 ± 0.001 and passes the parity gate. The text side alone costs -0.014 ± 0.010 , but its changes are noise per sentence rather than a systematic loss. Fake-quantizing the text side in PyTorch changes 34 of 17,017 token durations, F_0 by 0.21 Hz RMS and energy by 0.005, with an end-to-end UTMOS change of $+0.000 \pm 0.010$. No single layer group matters. In the decoder, by contrast, the last layers matter. All-int8 storage moves the output by 0.71 dB of log-spectral distance from float32 (re-seeding float32 moves it by 0.78 dB). Keeping only the final convolution `generator.conv_post` (4,928 parameters) in float32 halves that distance to 0.39 dB. Keeping it together with `generator.ups.1` and `generator.resblocks.4` (78,656 parameters) brings it to 0.27 dB, with a UTMOS change of -0.001 . In ONNX, keeping these three groups in float32 costs 0.23 MB and gives -0.016 ± 0.010 overall, with the remainder coming from the text side’s duration changes.

The parity gate. The parity gate inherited from our earlier exporter is too strict for these models. It only compares cases in which every token duration is unchanged, and on the three built-in cases this leaves one or two comparable cases. We therefore report the end-to-end UTMOS change instead. The released exporter reports the gate without enforcing it, and the released graphs are the ones evaluated here. A float32 keep-list for the decoder’s last layers and a gate that tolerates duration changes of one frame are left for future work.

Size and speed. The architecture is unchanged, so parameters (8.07M) and compute (3.36 GFLOP per second of audio for the decoder) are Paradee’s. The float32 graph is 32.8 MB and the int8 graph 9.17 MB,

close to the 9.04 MB of Paradee’s released int8 file (8.45 MB in its paper), which also contained the filter. We measured speed with Paradee’s protocol: one CPU thread, the first 8 held-out sentences, from phonemes. On an AMD Ryzen 9 9950X3D with ONNX Runtime 1.30, the int8 graphs of the final recipe run 24 to 26 times faster than real time (61 to 72 s of audio per voice; one run disturbed by other load was repeated, data/speed.csv). The released int8 graph, measured in the same sessions, runs 23 to 25 times faster; Paradee reports 17.8 on an Apple M4 Pro. Within one session the same released graph varied by about 8%, more than the differences between graphs (data/speed.csv). Within this variation the speeds are equal, as expected for an unchanged architecture, and the removed post-filter does not show in the timing. We measured speed on one desktop CPU only.

E Harmonic contrast: follow-up runs

if_sara	Harmonic contrast (dB)					UTMOS	
	0–1 kHz	1–2 kHz	2–4 kHz	4–6 kHz	HNR (dB)	16 rows	200 rows
Kokoro teacher	29.03	17.57	11.81	6.04	12.94	–	4.022
50k recipe (GAN3 5k)	28.54	15.70	5.95	3.57	12.81	3.82	3.752
<i>50k recipe, then 5,000 more GAN3 steps (restart)</i>							
control	28.59	15.88	6.01	3.55	12.83	3.85	3.768
with the harmonic comb	28.59	15.99	6.07	3.58	12.82	3.86	3.787
<i>50k recipe with GAN3 for 30,000 steps instead of 5,000</i>							
step 0 (GAN10 output)	28.54	15.27	5.26	3.55	12.84	3.78	–
step 5,000	28.59	15.49	6.47	3.68	12.74	3.86	–
step 10,000	28.69	16.52	7.29	3.63	12.86	3.95	–
step 20,000	28.43	16.47	7.51	3.83	12.89	3.97	–
step 30,000	28.59	16.47	7.69	3.99	12.83	3.96	3.882

Table 18: Follow-up runs on if_sara. Harmonic contrast and HNR as in Table 7 (40 held-out sentences, teacher-forced), here with two decimals. UTMOS on the 16 sentences monitored during training and on all 200 held-out sentences. Restart: 5,000 GAN3 steps from the 50k-recipe decoder with fresh critics, warmup and cosine decay, with and without the harmonic comb input. Lower block: snapshots of one GAN3 run of 30,000 steps from the recipe’s GAN10 decoder. “–”: not measured.

With the published stage lengths, our English decoders are at the level of the released one in every band of Table 7, so head distillation alone does not change the contrast. The runs below are all on if_sara.

The harmonic comb. One possible cause of the deficit is missing information: nothing in the student’s input marks where the upper harmonics lie. To test this, we added a zero-initialized input to the generator, the cosine and sine of k times the phase of the fundamental for the 48 harmonics below 11 kHz ($k = 1, \dots, 48$). It adds 77k parameters and leaves the teacher loop and the pool unchanged. Starting from the if_sara decoder of the 50k recipe, we trained 5,000 more GAN3 steps with this input and, as a control, without it (Table 18). The extra input does not change the contrast. With it, the 2–4 kHz contrast is 6.07 dB, against 6.01 for the control. No band differs from the control by more than 0.11 dB, and the HNR differs by 0.01 dB. These differences are of the same size as those between the control and the decoder it started from (up to 0.18 dB). On the 200 held-out sentences, UTMOS is 3.787 with the comb and 3.768 without. Paired on the same sentences and excitation seeds, the difference is $+0.019 \pm 0.004$, better on 137 of 200 sentences: small but consistent, and an order of magnitude below the gain of a longer adversarial stage. Giving the generator the phases of the upper harmonics explicitly therefore does not restore their structure.

The longer adversarial stage. In the 30,000-step GAN3 of the final recipe, the *if_sara* 2–4 kHz contrast rises from 5.26 dB at the start of the stage to 6.47 after 5,000 steps, 7.29 after 10,000 and 7.69 after 30,000 (Table 18). Against the 5.95 dB of the 5,000-step stage, this closes 1.74 dB of the 5.86 dB gap to the teacher, about 30%. The 1–2 kHz contrast rises from 15.70 to 16.47 dB and the 4–6 kHz contrast from 3.57 to 3.99 dB, while the HNR stays within 0.1 dB. English moves the same way, by less. Its 2–4 kHz contrast rises from 4.18 dB after the 5,000-step stage to 5.45 after 30,000 steps (4.33, 4.88 and 5.35 at 5,000, 10,000 and 20,000), about 20% of the gap to the teacher, and the 1–2 kHz contrast from 14.17 to 15.01 dB. For *if_sara*, the 16-sentence UTMOS levels off after about 12,500 steps (3.94 to 3.98 from there on), but the contrast keeps rising, from 7.29 dB at 10,000 steps to 7.51 at 20,000 and 7.69 at 30,000.

The number of steps is not the whole story. The control above has also had 10,000 GAN3 steps in total, the recipe’s 5,000 and 5,000 more, but in two stages, each with fresh critics, its own warmup and a full cosine decay, and its contrast stayed at 6.01 dB. The continuous stage had reached 7.29 dB at the same count. Its 5,000-step snapshot, taken at 93% of the peak learning rate, already has 6.47 dB, against 5.95 for the recipe’s stage of the same length, at whose end the learning rate has decayed to zero. The two runs also read separately rendered pools with the same settings, and all of these are single runs.

Open questions. We adopted the 30,000-step GAN3 after measuring it on the same development sentences, with one run per voice, and we did not search over its length. For the Italian voices the 16-sentence score levels off after 12,500 to 17,500 steps, while in English it was still rising slowly at 30,000, so a shorter stage may suffice in Italian and a longer one may still help in English. We have not separated the length of the adversarial stage from its learning-rate schedule. The wider generator’s effect on harmonic contrast was measured on *if_sara* only, with one seed (on *im_nicola* we have its UTMOS but no usable contrast), and we have not shown that the deficit is what listeners hear as the metallic ring.

F Negative results and listening checks

F.1 Negative results

A shared two-voice decoder. We trained one decoder for both Italian voices. The 16-dimensional learned style became a per-voice table, and each step drew 16 crops per voice in stage one and 8 per voice in the adversarial stages, so each voice saw as many crops as in its single-voice run. Otherwise the 30k recipe was unchanged. On each voice’s 200 held-out sentences, paired against the single-voice 30k decoders, the joint decoder loses 0.089 ± 0.008 for *if_sara* (3.622 against 3.712) and 0.086 ± 0.005 for *im_nicola* (3.489 against 3.575). Sharing 3.85M parameters and conditioning only through a 16-dimensional constant costs about 0.09 per voice. We have not tested whether the cause is capacity or the narrow conditioning.

Adversarial training from the first step. With the stored target (*s3*), the student stays at the floor for 3,000 steps. With the teacher in the loop and the head losses (*s6*), it reaches 3.59 after 10,000 steps, and its curve flattens under the cosine schedule (3.56, 3.58 and 3.59 at 8k, 9k and 10k). The staged pipeline built on the 10k *s2* decoder reaches 3.97 on the same 16 sentences, so the staged schedule wins, although the two do not use the same number of steps.

A short stage one. The adversarial stages do not make up for a weak stage one. From the 10k *s2* decoder (2.72 on the 16 monitored sentences) the full pipeline ends at 4.02, and from the 30k decoder (3.93) at 4.365, both on 64 sentences. On all 200 sentences, the 30k and 50k decoders (3.93 and 4.13) end at 4.372 and 4.414. All use the published 5,000-step GAN3.

Numerics. Neither TF32 convolutions (Appendix B) nor the device on which the excitation STFT is computed explains the gap in our upstream reproductions. A bf16 teacher was too inaccurate to serve as a target, while fp16 was acceptable (Appendix A).

F.2 Listening checks

We did not run a listening test with several listeners. The author made two blind checks on English, on headphones with a listening tool that hides the identity of each sample and matches loudness. Both used the same four held-out sentences. The first was an A/B comparison between the complete 30k-recipe (s7) and 50k-recipe (s8) decoders, after GAN10 and GAN3: s8 was preferred on three of the four sentences. That is consistent with the +0.04 UTMOS difference, but it is not evidence on its own (one-sided binomial $p = 0.31$). The second compared the teacher, s7 and s8 in one session, with a 0–3 rating of “metallic echo”. The teacher received 0 and both students 1 on all four sentences, and the teacher was judged the most natural each time. A brief buzz was marked at the same instant of the first sentence in all three versions, so it comes from the content or from the teacher. The faint metallic ring is therefore added by the small decoder and not inherited from the teacher. It is the main remaining audible defect we know of. In Italian, the author’s informal listening agreed with the ordering in Table 6 for the upstream and revised recipes, and preferred the adversarial stages to stage one alone, which UTMOS barely reflected for `if_sara`. This is enough to locate a defect, but not to rank systems. A MOS or MUSHRA-style test [13] with several listeners, in English and especially in Italian, is left to future work.

G Seeds, the released training package and cost

Seeds. Each configuration was trained once per voice, with one seed, with a single exception. We repeated the final recipe for `if_sara` with two more seeds, on the same teacher rows and with the same crop-pool settings and evaluation sentences. Stage one still depends on the seed: after its 50,000 steps the three runs score 3.745, 3.431 and 3.403 on the 200 held-out sentences. Our first run was thus a favourable one, and the stage-one rows of Table 6 should be read with this spread in mind. The spread (3.403 to 3.745) is far smaller than the gap to the upstream recipe, whose stage one scores 1.37 on this voice, and the adversarial stages remove it: the three final decoders score 3.882, 3.881 and 3.883 (95% intervals ± 0.023 to ± 0.025). For this voice, then, the final result does not rest on a favourable run. The other voices, and the intermediate 50k recipe, were trained with one seed only. The upstream recipe, conversely, was measured with only two English runs and one run per Italian voice.

The released training package. The recipe was developed in research scripts and then ported into the training package we release. We checked the port on `if_sara` only. Rerun on the paper’s teacher rows and crop pool, the package’s final decoder scores 3.878 against the paper’s 3.882, a paired difference of -0.004 ± 0.007 on the 200 held-out sentences (better on 102), and 3.883 against 3.894 end to end in the same evaluation (3.895 in Table 14, from a separate evaluation). Its int8 graph changes UTMOS by -0.030 ± 0.009 against its own float32 graph, more than any graph in Table 17 (-0.007 ± 0.011 for the paper’s final `if_sara` graph, the one we release), so the int8 loss varies between trainings and should be measured for each new voice. The weights we release are the graphs evaluated in this paper.

Cost. For `af_heart`, rendering the pool took about one hour. The 30k stage one then took 2 h 49 min, the 20k extension 1 h 20 min, GAN10 50 min and the 5,000-step GAN3 64 min, about six hours of training in total on the shared GPU. The final recipe’s 30,000-step GAN3 took 6 h 41 min instead, bringing the total

to almost twelve hours. The Italian voices took similar times, with the two voices trained concurrently. Reproduction A took about ten hours end to end, including corpus generation and the CPU regeneration of its targets. Its stage one ran at about 0.21 s per step without competing jobs. Each adversarial job needed about 6.2 GB of GPU memory, on a GPU with 32 GB. Each pool needs about 62 GB of disk. Unlike Paradee, which trained on a laptop, the revised recipe needs the teacher and a CUDA GPU while the pool is rendered. Afterwards, training reads only the pool.