Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Eivind Nesje

Benchmarking Deployed Real-Time Voice Conversion Tools

A Unified, Reproducible Quality Comparison of Open-Source and Commercial Voice Changers

Master's thesis in Computer Science
Supervisor: Theoharis Theoharis
June 2026

NTNU
Norwegian University of Science and Technology

Eivind Nesje

Benchmarking Deployed Real-Time Voice Conversion Tools

A Unified, Reproducible Quality Comparison of Open-Source and Commercial Voice Changers

Master's thesis in Computer Science
Supervisor: Theoharis Theoharis
June 2026

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

NTNU
Kunnskap for en bedre verden

Department of Computer Science

TDT4900 - Computer Science, Master Thesis

Benchmarking Deployed Real-Time Voice Conversion Tools

A Unified, Reproducible Quality Comparison of Open-Source and Commercial Voice Changers

Author:
Eivind Nesje

Supervisor:
Theoharis Theoharis

Spring 2026

Abstract

Real-time voice conversion has moved from the research laboratory into consumer software. Anyone streaming, gaming, dubbing, or seeking anonymity online can now download or buy a tool that changes their voice as they speak. These tools advertise confident claims about naturalness, fidelity, and latency, yet those claims are almost never backed by neutral measurement. The peer-reviewed literature evaluates research systems on incompatible corpora, metrics, and hardware, while the deployed tools people actually use are assessed by their own vendors, or not at all.

This thesis builds a common standard for rigorously measuring output quality. It defines a single objective, reproducible, black-box protocol and applies it to six widely used real-time voice conversion tools in seven configurations, on two contrasting speech corpora (clean read speech from LibriSpeech and spontaneous speech from GigaSpeech) and on fixed hardware. The benchmark scores each tool on an intelligibility, naturalness, and speaker-identity portfolio, using word and character error rate, two neural mean-opinion-score predictors, and a family of speaker-embedding similarity measures.

Two findings stand out. First, the tools fall into consistent quality tiers that cut across the divide between open-source and commercial software: the commercial tools bracket the field rather than leading it, the most expensive among them is the weakest measured, and better quality tracks neither higher price nor heavier resource use. Second, the automatic measures of speaker identity prove unreliable on their own, rewarding distortion and collapsed output as readily as genuine conversion, and become meaningful only when read alongside the intelligibility and naturalness scores. Beyond the benchmark itself, the thesis contributes a structured literature review and architectural taxonomy that places the deployed tools against the peer-reviewed streaming systems, and it releases the capture harness, the scoring pipeline, and a selection of the converted audio publicly.

Sammendrag

Sanntids stemmekonvertering har beveget seg fra forskningslaboratoriet og inn i forbrukerprogramvare. Hvem som helst som direktestrømmer, spiller, dubber eller søker anonymitet på nett, kan nå laste ned eller kjøpe et verktøy som endrer stemmen deres mens de snakker. Disse verktøyene reklamerer med selvsikre påstander om naturlighet, stemmelikhet og forsinkelse, men disse påstandene er nesten aldri underbygget av nøytrale målinger. Den fagfellevurderte litteraturen evaluerer forskningssystemer på uforenlige korpus, metrikker og maskinvare, mens de utbredte verktøyene folk faktisk bruker, vurderes av leverandørene selv, eller ikke i det hele tatt.

Denne masteroppgaven etablerer en felles standard for å måle utgangskvalitet. Den definerer én objektiv, reproduserbar svartboks-protokoll og anvender den på seks mye brukte sanntids stemmekonverteringsverktøy i sju konfigurasjoner, på to kontrasterende talekorpus (ren opplest tale fra LibriSpeech og spontan tale fra GigaSpeech) og på fast maskinvare. Referansemålingen vurderer hvert verktøy på en porteføljе av forståelighet, naturlighet og taleridentitet, ved hjelp av ord- og tegnfeilrate, to nettverksbaserte prediktorer for "mean opinion score", og en gruppe likhetsmål basert på talervektorer.

To funn skiller seg ut. For det første faller verktøyene inn i konsistente kvalitetsnivåer som går på tvers av skillet mellom åpen kildekode og kommersiell programvare: de kommersielle verktøyene rammer inn feltet snarere enn å lede det, det dyreste av dem er det svakeste som ble målt, og bedre kvalitet henger verken sammen med høyere pris eller tyngre ressursbruk. For det andre viser de automatiske målene for taleridentitet seg upålitelige alene, ettersom de belønner forvrengning og kollapset lyd like gjerne som ekte konvertering, og blir meningsfulle først når de leses sammen med forståelighets- og naturlighetsmålene. Utover selve referansemålingen bidrar oppgaven med en strukturert litteraturgjennomgang og arkitektonisk taksonomi som plasserer de utbredte verktøyene opp mot de fagfellevurderte strømmesystemene, og den gjør opptaksrammeverket, poengberegningspipelinen og et utvalg av den konverterte lyden offentlig tilgjengelig.

Preface

Acknowledgments

I would like to thank my supervisor, Theoharis Theoharis, for his guidance, feedback, and support throughout this work. His feedback helped shape the direction and substance of this thesis, and his support carried me through the points where the work was most difficult.

Declaration

I declare that this thesis is my own work. Sources are acknowledged through citations and references where they are used.

Generative AI tools were used in a supporting role, for editing language, for help with LaTeX, and for assistance with debugging and research, but not to produce any of the experimental results or measured numbers. Full details are given in Appendix B and in the accompanying AI declaration form.

Eivind Nesje
Trondheim, Spring 2026

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Real-time voice conversion has moved out of the research laboratory and into everyday software. Someone streaming a game, joining a voice chat, dubbing a video, or wanting to mask their voice online can now download or buy a tool that changes their voice as they speak. These tools advertise confident claims about how natural they sound, how faithfully they reproduce a chosen voice, and how little delay they add.

These claims are almost never backed by neutral measurement. The peer-reviewed literature (Chapters 2 and 3) evaluates research systems, on research corpora, with metrics and hardware that change from paper to paper, while the deployed tools people actually use are evaluated by their own vendors, or not at all. A user choosing between two voice changers, or a researcher asking how the deployed state of the art compares to the published one, has no common standard to consult.

This thesis provides that common standard, for the one dimension that can be measured rigorously today: output quality. It defines one objective protocol, applies it to a set of widely used real-time voice conversion tools on a single corpus and a single machine, and reports the results, so the tools can be compared with each other and placed against the architectures in the research literature.

1.2 Problem Statement

The core problem is the lack of a unified, reproducible, objective comparison of the quality of deployed real-time voice conversion tools, and it has several facets. The tools vary widely, from open-source retrieval-based and zero-shot reference-conditioned tools to closed commercial software, so they can be compared only by what they produce, not by their internals. Quality is itself several things, intelligibility, naturalness, and speaker identity, so a fair comparison needs a set of measures, not one score. And for most tools there is no clean recording of the target voice to compare against. That rules out the most direct similarity measure and forces the use of reference-free proxies, whose own trustworthiness is then in question.

1.3 Research Questions

The thesis is built around two research questions, stated in full in Section 4.1 and summarised here.

RQ1. How do widely used real-time voice conversion tools compare in objective output quality,

across intelligibility, naturalness, and speaker identity, when they are all measured under one common protocol?

RQ2. Does better quality come at a higher cost to the user, whether in money or in the computing resources a tool demands, or do free and lightweight tools keep up with paid and resource-heavy ones?

1.4 Scope and Delimitations

The thesis is deliberately bounded. Three limits are important enough to state at the outset.

Quality, not latency. These are real-time systems whose users arguably care most about end-to-end latency (Section 2.2), yet this thesis measures quality, not latency. Measuring latency well is a separate engineering task, and a half-measured figure bolted onto the quality benchmark would weaken both, so it is named as the foremost future work (Chapter 8). The recorded resource use (Section 4.6.4) is a rough deployability indicator, not a latency measurement.

Deployed tools, measured directly. The objects of study are tools a real user can obtain and run, evaluated as black boxes through the audio they produce. Peer-reviewed systems are surveyed in Chapter 3 as the landscape to place the tools against, not benchmarked themselves: the contribution is to measure the deployed tools the literature leaves unmeasured.

Reference-free measurement. The metrics need no "ideal" recording to compare against, and for speaker identity they mostly need no recording of the target voice either, because for most tools none exists. How these reference-free proxies behave, and where they fail, is examined directly, and turns out to be a contribution in its own right.

1.5 Contributions

The thesis makes the following contributions.

1. A unified, reproducible, black-box benchmark for deployed real-time voice conversion tools, a capture harness and a scoring pipeline over an intelligibility, naturalness, and speaker-identity portfolio, applied to six tools in seven configurations on two contrasting speech corpora and fixed hardware to give the first common-protocol quality comparison of them (Chapters 4 to 6).
2. An analysis of quality against cost, in money and in computing resources, including a matched comparison of one RVC model run by two different hosts (Chapters 6 and 7).
3. A look at how far the common reference-free speaker-identity proxies can be trusted, showing concretely where they mislead when no clean target recording is available (Chapter 7).
4. A structured review of real-time voice conversion that places the deployed tools against the peer-reviewed literature (Chapter 3).

To support reuse and scrutiny, the benchmark is released publicly: the capture harness and scoring pipeline as an open testing environment (https://github.com/eivindnesje/Benchmarks), and the measured results together with a curated selection of the converted audio on an accompanying website (https://eivindnesje.github.io/Benchmark-website/), where the comparison can be heard as well as read.

1.6 Thesis Structure

Chapter 2 introduces the concepts and vocabulary of voice conversion, real-time operation, and evaluation. Chapter 3 reviews the literature through a documented protocol and builds the taxonomy used later. Chapter 4 specifies the benchmark, and Chapter 5 the software that runs it. Chapter 6 reports the measurements, Chapter 7 interprets them against the two research questions, and Chapter 8 concludes and sets out future work, of which a controlled latency benchmark is the most important.

Chapter 2

Background and Preliminaries

Real-time voice conversion sits where speech signal processing, applied machine learning, and audio systems engineering meet, and the rest of the thesis assumes vocabulary from all three. This chapter introduces that vocabulary: the voice conversion problem and how the field reached its current methods, what real-time operation demands, the content-encoder and vocoder pipeline behind almost every modern system, and how conversion quality is evaluated, the last being a central concern of this thesis.

2.1 The Voice Conversion Problem

2.1.1 An Informal View

Voice conversion (VC) makes one person's speech sound as if someone else had spoken it, without changing what was said. Given a recording of a *source* speaker, a VC system produces a new recording that keeps the words, their timing, and ideally their intonation, but gives them the vocal identity of a *target* speaker [42]. Humans separate these aspects effortlessly; for a machine it is hard, because a waveform carries no labels saying which parts are linguistic content and which are speaker identity. Content, timbre, pitch habits, speaking rate, and recording conditions are all tangled in one signal, and much of the history of VC is a history of better ways to pull them apart, well enough that the output is intelligible, natural, and convincingly the target [42, 4]. This thesis adds the constraint that the change must happen *live*, while the speaker is still talking (Section 2.2).

2.1.2 Formal Statement

Formally, given a source utterance $\mathbf{x}^{(s)}$ from speaker s, the goal is a synthesised utterance $\hat{\mathbf{x}}^{(t)}$ that a listener would attribute to a target speaker t while recognising the same content as in $\mathbf{x}^{(s)}$. Modern generative formulations model the conditional distribution

$$p\left(\hat{\mathbf{x}}^{(t)} \,\middle|\, \mathbf{x}^{(s)}, \mathbf{c}^{(t)}\right), \tag{2.1}$$

where $\mathbf{c}^{(t)}$ is whatever identifies the target speaker: a discrete label, a learned embedding, a reference utterance, or a set of retrieved acoustic exemplars [42, 4, 1].

Two requirements run through this thesis. First, the converted utterance must preserve the linguistic content of the source, which motivates content-preservation metrics such as word error rate (Section 2.6). Second, the converted utterance must actually resemble the target speaker, which motivates speaker-similarity metrics computed in the embedding space of speaker verification models. A system can fail at either requirement independently: it can produce a perfect imitation of the

target voice mumbling unintelligibly, or a crystal-clear utterance that still sounds like the source speaker.

2.1.3 A Brief History of Methods

This history explains why current systems look the way they do; fuller treatments are given by Sisman et al. [42] for the development up to 2020 and by Bargum et al. [4] for the deep-learning period.

The classical era of voice conversion was built on *parallel* corpora, in which the source and target speakers record the same sentences. Frame-level alignment between the two recordings, typically via dynamic time warping, lets a statistical regression model such as a Gaussian mixture model learn a mapping from source spectral features to target spectral features [42], at a clearly perceptible cost in naturalness [44]. Because recording matched sentence sets for every speaker pair does not scale, three lines of work moved beyond it:

- **Adversarial training** between unaligned speech domains. CycleGAN-VC uses cycle-consistency losses to map between two speakers with no aligned frames [17], and StarGAN-VC extends this to many-to-many conversion with a single generator [16].
- **Disentanglement.** AutoVC shows that a carefully sized autoencoder bottleneck, with speaker conditioning on the decoder, can force content and identity apart from a reconstruction loss alone [30].
- **Recognition then synthesis**, the most durable line. Sun et al. [43] turn source speech into *phonetic posteriorgrams*, frame-wise distributions over phonetic classes that are largely speaker-independent, then synthesise them in the target voice (examined in Section 2.3).

The recognition-then-synthesis line is the ancestor of today's systems. Its phonetic posteriorgram was later replaced by *self-supervised* representations such as HuBERT and WavLM [14, 7], which need no transcribed data, and its synthesis side by adversarial neural vocoders [19] and end-to-end models such as VITS [18]. A pretrained self-supervised content encoder paired with such a synthesiser is, with variations, the architecture of essentially every deployed open-source voice conversion tool today, and of most streaming systems reviewed in Chapter 3 [4, 49, 1].

One deployed tool departs from this otherwise near-universal recipe. Beatrice, a low-latency voice changer aimed at the same any-to-many setting as RVC-based tools, builds its content extractor, pitch estimator, and waveform generator in-house rather than reusing a pretrained self-supervised encoder, and ships per-target models in its own format rather than RVC's [29]. It is included in the deployed-tools benchmark of this thesis precisely because it is architecturally distinct from the retrieval-based tools that dominate the rest of the field, and is positioned alongside the peer-reviewed streaming systems in Chapter 3.

2.1.4 Taxonomic Dimensions

Several independent distinctions categorise VC systems and recur throughout the thesis.

Training data. As described above, early methods required parallel corpora with frame-level alignment. Whereas modern systems train on non-parallel data, where source and target share no common content [42, 4]. All systems studied here are non-parallel.

Mapping cardinality. A VC system is *one-to-one* when both speakers are fixed at training time, *any-to-one* when an arbitrary source maps to a fixed target, *any-to-many* when the target is chosen at inference from a fixed set, and *any-to-any* (zero-shot) when both may be unseen and the target is given by a reference utterance at inference time [4, 1, 5]. The distinction matters in practice because it determines how much work a user must do before conversion is possible. Open-source

Table 2.1: Mapping cardinalities in voice conversion. The *cost per new target* column indicates what a user must provide or compute before converting into a previously unused target voice. Deployed RVC-based tools are documented in grey literature and are mapped to the peer-reviewed landscape in Chapter 3.

Setting	Source	Target	Cost per new target	Representative systems
One-to-one	fixed, seen	fixed, seen	retrain the full model on data from the new speaker pair	classical statistical regression [42, 44]
Any-to-one	unconstrained	fixed, seen	train a dedicated model per target voice	LLVC [39]
Any-to-many	unconstrained	fixed set, seen	retrain or fine-tune so the new target joins the supported set	RVC-based tools; Beatrice [29]
Any-to-any (zero-shot)	unconstrained	unconstrained	none; a reference utterance is supplied at inference time	YourTTS [5], RT-VC [22]

consumer tools are usually any-to-one or any-to-many, with a model trained per target; recent streaming systems increasingly aim for any-to-any [49, 22]. Table 2.1 summarises the four.

Conversion target. The most common formulation transforms speaker timbre while attempting to preserve linguistic content and, where possible, source prosody. Other formulations explicitly modify prosody, accent, or emotional expression [42, 4]. This thesis is concerned with timbre conversion and treats prosody preservation as a desirable property rather than as a manipulation target.

Conversion method. Systems can also be grouped by the machine-learning method that performs the conversion. Following Bargum et al. [4], the principal groups are: pipelines that first extract linguistic content and then synthesise speech from it, sequence-to-sequence models, GANs, VAE and normalising-flow models, retrieval-based methods, and, more recently, diffusion and neural-codec language models. The groups overlap, since most deployed tools pair a self-supervised content representation with an adversarially trained synthesiser, so a single tool can belong to several groups at once [18, 1, 49]. For this reason the taxonomy developed in Chapter 3 classifies systems along several independent axes rather than assigning each to one group.

2.2 Real-Time Operation: Definitions and Constraints

This thesis studies systems meant for real-time operation. The term is used loosely, especially for consumer tools, so this section sets the definitions used later and explains why real-time operation is a different engineering problem from offline conversion, not just a faster one.

2.2.1 Two Necessary Conditions: Throughput and Latency

A voice conversion system qualifies as real-time only if two conditions hold simultaneously. First, its *real-time factor* (RTF), the ratio of processing time to processed audio duration, must stay below one, so the system keeps up with the incoming audio stream without falling progressively behind. Second, its *latency* must remain bounded and small enough not to disrupt the user interaction it supports.

The two conditions are not equivalent, and conflating them is a common source of confusion when reading tool documentation. An offline batch processor may achieve RTF $\ll 1$ on a long file while having no meaningful latency definition at all, since it requires the complete utterance before

producing any output. Conversely, a streaming system may operate at RTF just below unity and still exhibit long latency, which would make it unusable for live conversation even though it never falls behind. The peer-reviewed real-time VC literature reports both quantities explicitly; recent systems achieve RTF well below one and end-to-end latencies in the tens of milliseconds on modern consumer hardware [49, 39, 22].

Why latency matters depends on the use case: in live communication, excessive delay disrupts conversational turn-taking, which is why end-to-end latency, not throughput, is what these tools' users care about most. Latency is not a fixed property of a tool, though: it depends heavily on configuration, since most tools, open-source ones especially, expose look-ahead and buffering controls that trade latency against quality. A proper latency study would have to compare each tool across those settings. Latency against quality, and this is left to future work (Section 1.4). The contribution here is output quality, with every tool set to a comparable, usable latency (Section 4.6.2).

2.2.2 Causality and Look-Ahead

A model is *causal* if its output at time t depends only on inputs at times $\tau \leq t$. Strict causality is required for indefinite streaming operation without look-ahead-induced delay. Many high-quality offline VC architectures rely on bidirectional context, attention over the full utterance, or non-causal convolutions, none of which is compatible with strict causality [49, 42]. The principal architectural challenge of real-time VC is therefore to preserve as much as possible of the quality of bidirectional models under a causality constraint.

In practice, most streaming systems permit a small *look-ahead window* of a few tens to hundreds of milliseconds, trading a modest increase in this look-ahead delay for substantially better output quality. Reported figures from the systems most relevant to this thesis illustrate the achievable range. Yang et al. [49] report 70.8 ms end-to-end latency on a mobile platform, with the algorithmic component arising from soft speech units derived from a causally trained content encoder. The articulatory-coding system of Liu et al. [22] reports a comparable 61.4 ms end-to-end latency on a desktop CPU using a causal source extractor and a differentiable DSP vocoder. Sadov [39] reports sub-20 ms latency on a consumer CPU through a distilled GAN-based any-to-one architecture. These figures should be read with the caveat developed in Section 2.6.5: they were obtained under different measurement protocols and on different hardware, and are not directly comparable.

2.2.3 Chunked Inference, Buffer Sizes, and the Audio Stack

In deployment, audio arrives in fixed-size buffers from the host audio API (Core Audio, WASAPI or ASIO, ALSA, JACK, PulseAudio, or PipeWire), and the model runs once per buffer, which is why streaming inference is called *chunked*. A buffer of B samples at rate f_s adds B/f_s seconds of buffering on each side: at $B = 256$, $f_s = 48\,000$ Hz that is about 10.7 ms before any conversion. Smaller buffers cut latency but raise overhead and the risk of *dropouts* (glitches when a buffer misses its deadline); larger ones are safer but slower. Every tool therefore exposes a buffer-size trade-off between latency and robustness, and can behave very differently at different settings; the methodology of Chapter 4 fixes one configuration per tool and leaves buffer sweeps to the latency-focused future work of Chapter 8. A further wrinkle is that the model's internal frame rate rarely matches the buffer size, so the deployment layer must accumulate input, run the model, and smooth the output back into the stream; this queueing adds further delay that papers rarely report but users always feel.

2.3 VC Architecture

The dominant architectural pattern in modern voice conversion is a two-stage pipeline in which a *content encoder* is followed by a synthesiser, a structure also described in the literature as an encoder-decoder design. The source utterance is first encoded into an intermediate representation

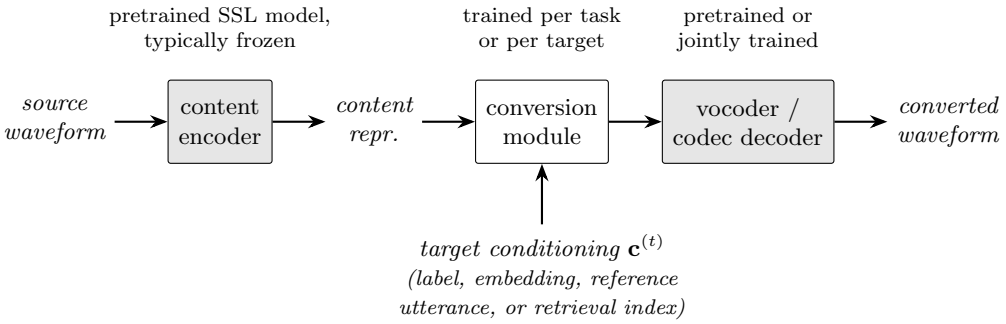

Figure 2.1: The two-stage content-encoder and vocoder pipeline behind most modern voice conversion systems. A pretrained content encoder produces a speaker-attenuated representation that a synthesis stage renders in the target voice. Shaded blocks are typically pretrained and reused across systems, the smaller conversion module carries the speaker-specific training.

intended to be invariant to speaker identity while preserving linguistic content. This representation is then re-synthesised, conditioned on the target speaker, by a separate generative model [42, 4]. Figure 2.1 shows the pattern schematically. The pattern has three motivations.

First, it puts the conceptual split between content and speaker identity into practice by enforcing it structurally: whatever speaker information survives into the intermediate representation is the only speaker information the synthesis stage can reproduce. Second, it permits the two halves of the pipeline to be trained or pretrained on different data, with large unlabelled speech corpora for the encoder and smaller speaker-specific corpora for the synthesis side. Third, it enables reuse of components: a single pretrained content encoder and a single pretrained vocoder can serve many downstream VC systems, with only a comparatively small bridging model needing speaker-specific or task-specific training [1, 49].

2.3.1 Content Encoders

A content encoder maps an input waveform or spectrogram to feature vectors at a reduced frame rate, most often 50 Hz (a 20 ms hop). In modern practice it is a pretrained self-supervised model, frozen or lightly fine-tuned [14, 7, 1], replacing the older phonetic posteriorgram [43] with richer detail and no need for transcribed data (Section 2.4).

2.3.2 Target-Speaker Conditioning

The synthesis stage must be told which voice to produce, and the form this conditioning takes determines the mapping cardinality of the resulting system (Section 2.1.4). The simplest option is a discrete speaker label embedded via a lookup table, which restricts the system to speakers seen in training. A more flexible option is a continuous speaker embedding computed by a pretrained speaker-verification model from a reference utterance, which enables zero-shot operation [5]. Retrieval-based systems take a third route: rather than conditioning a generative model on an embedding, they replace each frame of the source's content representation with the most similar frames found in a database of target-speaker features, so the target identity enters through the retrieved exemplars themselves [1]. The retrieval approach is notable in this thesis because, in modified form, it underlies the widely deployed RVC-based open-source tools that several of the benchmarked applications are built upon.

2.3.3 Vocoders and Waveform Generators

The final stage transforms the converted intermediate representation into a waveform. Two principal types dominate: adversarial mel-spectrogram vocoders exemplified by HiFi-GAN [19], and

neural audio codecs exemplified by SoundStream [50] and EnCodec [10]. The latter is increasingly favoured for streaming applications, for reasons developed in Section 2.5.

2.3.4 Disentanglement and Information Leakage

A persistent problem in this two-stage approach is incomplete disentanglement: information about the source speaker leaks through the intermediate representation and is reproduced in the output, partially compromising the conversion [30, 4]. Leakage manifests as converted speech that carries the target's broad timbre but retains recognisable traces of the source's accent, pitch habits, or vocal quality. Strategies for mitigating it include information bottlenecks [30], instance normalisation, vector quantisation, intermediate representations engineered to discard speaker information [31], and, in retrieval-based approaches, explicit replacement of source frames with target-domain exemplars [1]. Streaming systems face the additional constraint that disentanglement must be achieved causally, ruling out some of the techniques available to offline models [49].

2.4 Encoding

The largest practical change in voice conversion methodology over the past five years has been the adoption of self-supervised learning (SSL) for content encoding. Before SSL became routine, these pipelines relied on phonetic posteriorgrams from supervised acoustic models [43] or on bottleneck features from speaker-classification networks [42]. SSL representations replaced both, providing richer linguistic information without the labelled-data costs. This section introduces the SSL models that recur throughout the thesis, in roughly the order they entered the VC literature.

2.4.1 Masked Prediction on Speech: wav2vec 2.0 and HuBERT

The breakthrough demonstration that masked prediction, the pretraining principle behind large text models, transfers to raw speech came with wav2vec 2.0 [3]. The model quantises latent speech features into discrete units and trains a transformer to identify the correct unit for masked time steps via a contrastive objective, learning from unlabelled audio alone.

HuBERT [14] refined the recipe into the form most relevant to voice conversion. Its training proceeds in stages. An initial set of pseudo-labels is produced by clustering low-level acoustic features with k-means, so that each frame of audio receives a discrete cluster identifier. A transformer encoder is then trained on a masked-prediction task: a fraction of the input frames is hidden, and the encoder must predict the pseudo-labels at the masked positions from the surrounding context. Once this first pass has converged, the trained encoder is used to re-extract features from the same audio, those features are re-clustered into a more linguistically meaningful label set, and the encoder is retrained against the refined targets. The cycle can be repeated.

The resulting transformer's intermediate layers carry information correlated with phonetic content while attenuating much of the speaker- and channel-specific variation in the raw signal. The representation is not perfectly speaker-invariant, but it is content-dense enough that downstream VC systems can build directly on it with minimal further training. HuBERT features form the content backbone of many open-source VC tools, including widely deployed retrieval-based variants.

2.4.2 WavLM

WavLM [7] extends the HuBERT paradigm by augmenting the pretraining task with utterance mixing, which encourages robustness to overlapping speech and noise, and by gated relative position encoding. WavLM was shown to outperform HuBERT on a broad range of downstream tasks, and retrieval-based voice conversion has drawn on features from its sixth transformer layer, identified as a balance between phonetic content and the granularity that nearest-neighbour matching needs [1].

2.4.3 ContentVec

ContentVec [31] is a self-supervised representation built on the HuBERT framework with additional mechanisms designed to suppress speaker-related information. The authors motivate the work by observing that HuBERT and comparable encoders retain measurable speaker leakage in their intermediate features, which is precisely the failure mode described in Section 2.3. ContentVec and HuBERT-derived features form the content backbone of several major open-source VC tools, including the Soft-VC method [25] and the Retrieval-based Voice Conversion (RVC) project.

2.4.4 Discrete and Soft Speech Units

Niekerk et al. [25] compare two distinct ways of using HuBERT features for voice conversion. The *discrete units* approach quantises the features at each frame to one of a finite set of codes, so that each 20 ms of audio is summarised by a single integer cluster index. The *soft units* approach instead retains a continuous-valued vector at each frame, with one component per cluster, representing how likely the frame is to belong to each cluster. The discrete version is more compact and easier to operate on with language-model-style methods, but it discards within-cluster variation that may carry phonetic information. The soft version preserves this variation at the cost of a higher-dimensional representation.

The authors report that soft units improve naturalness and intelligibility relative to their discrete counterparts, attributing the gain to the preservation of fine-grained phonetic information that the quantisation otherwise discards. Soft units have since become the dominant content representation in streaming VC, Yang et al. [49] adopt a causally trained soft-unit encoder as the front end of their system.

2.4.5 Causal Constraints on SSL Encoders

Standard pretrained SSL encoders use bidirectional transformer layers and are therefore non-causal as published. This creates an awkward mismatch for streaming systems: the best available content representations come from models that, in their published form, cannot be run in a streaming fashion at all. Streaming systems resolve the mismatch in one of two ways. Either they retrain a causal variant of the encoder, often by distilling the bidirectional teacher into a causal student [49], or they restrict the encoder to a causal computation graph and accept a quality penalty in exchange for the latency guarantee. This is one of the principal architectural costs of moving from offline to streaming operation, and the strategies different systems use to pay it form one of the axes of the taxonomy in Chapter 3.

2.5 Neural Vocoders and Audio Codecs

The choice of waveform generator critically shapes both audio quality and latency. This section summarises the types relevant to the present thesis (Table 2.2), in roughly historical order, since each arose to fix a deficiency of its predecessor.

2.5.1 Classical Signal-Processing Vocoders

Before neural waveform generation, speech synthesis and conversion pipelines used signal-processing vocoders that decompose speech into interpretable parameters (typically a spectral envelope, a fundamental frequency track, and an aperiodicity measure) and resynthesise a waveform from modified parameters. WORLD [24] is the best-known representative and remains in use where computational budgets are tight, since it runs comfortably in real time on a CPU. The limitation of these classical vocoders is quality: the parametric decomposition introduces characteristic artefacts, and

the naturalness ceiling of classical vocoders is audibly below that of natural speech, which motivated the move to learned waveform generation.

2.5.2 Autoregressive Neural Vocoders

WaveNet [26] demonstrated that a deep autoregressive model generating audio sample by sample could exceed the quality of classical parametric synthesis by a wide margin. Its significance for this thesis is twofold. It established the neural vocoder as a component category, and it illustrated the central tension of the field: the original sample-by-sample formulation was far too slow for real-time use, requiring one full network evaluation per output sample. Much of the subsequent vocoder literature can be read as a search for WaveNet-level quality at a computational cost compatible with streaming.

2.5.3 Mel-Spectrogram GAN Vocoders: HiFi-GAN

HiFi-GAN [19] resolved the tension between quality and speed for many applications. It is a generative adversarial vocoder that maps mel-spectrograms to waveforms in a single parallel pass, introducing a multi-period discriminator that captures the periodic structure of speech and a multi-receptive-field fusion generator. Kong et al. [19] report inference rates well above real time on consumer GPUs and perceptual quality competitive with autoregressive vocoders. HiFi-GAN remains the default vocoder in many open-source VC pipelines, including retrieval-based ones [1]. A neural source-filter (NSF) variant integrates an explicit excitation signal to improve pitch handling and is used by several singing-oriented VC systems.

2.5.4 Neural Audio Codecs: SoundStream and EnCodec

SoundStream [50] is a neural audio codec made up of a convolutional encoder, a residual vector-quantisation bottleneck, and a convolutional decoder, trained end-to-end with combined adversarial and reconstruction objectives. The authors describe a streaming variant of the architecture capable of real-time inference on mobile-class processors. EnCodec [10] is a closely related codec built on a streaming convolutional encoder–decoder with a quantised latent space, developed for high-fidelity compression in real time. Although codecs were designed for compression rather than synthesis, their decoders are high-quality conditional waveform generators, and the SoundStream encoder–decoder pair was adopted as the synthesis backbone of StreamVC [49]. Similar neural-codec backbones underlie several more recent streaming VC systems reviewed in Chapter 3.

Neural codecs have two properties that make them attractive for streaming VC. First, the encoder and decoder are inherently causal, or can be made so with modest modification, aligning with the algorithmic-latency budget required for real-time operation. Second, the residual-quantiser bottleneck provides a natural site for disentangling content from speaker, since the quantisation discards information that the decoder must learn to regenerate from conditioning.

2.5.5 Vocoder Choice and Algorithmic Latency

Mel-spectrogram GAN vocoders such as HiFi-GAN inherit a latency contribution from the STFT analysis window used to compute the mel-spectrogram, typically on the order of tens of milliseconds depending on configuration. Neural codecs operating directly on waveforms can reduce this contribution by using strided convolutions in place of the STFT, at the cost of higher parameter counts in the decoder. This trade-off is one of the principal reasons that recent streaming VC systems favour codec-style decoders over mel-spectrogram GANs [49]. Table 2.2 summarises the waveform-generator types along the dimensions most relevant to this thesis.

Table 2.2: Waveform-generator types and their suitability for streaming voice conversion. Characterisations are qualitative; measured comparisons under a common protocol appear in Chapter 6.

Type (example)	Input representation	Causality and latency	Representative use in VC
Classical parametric (WORLD [24])	spectral envelope, F_0, aperiodicity	frame-based with small look-ahead; runs in real time on a CPU; quality ceiling audibly below natural speech	legacy pipelines; settings with tight compute budgets
Autoregressive neural (WaveNet [26])	conditioning features; output generated sample by sample	causal by construction, but original formulation far too slow for real-time use	historical quality reference; predecessor of modern vocoders
Mel-spectrogram GAN (HiFi-GAN, NSF variant [19])	mel-spectrogram	parallel and fast, but non-causal as published; STFT analysis window adds tens of milliseconds	retrieval-based VC [1]; default in most open-source tools
Neural codec (SoundStream [50], EnCodec [10])	waveform via quantised latents	causal or near-causal by design; low algorithmic latency; real-time decoding on mobile-class hardware	StreamVC [49]; recent streaming systems (Chapter 3)

Table 2.3: Overview of the evaluation metrics used in this thesis, the quality dimension each measures, and the underlying model or standard. Full implementation details, including model versions and checkpoints, are given in Chapter 4.

Dimension	Metric	Underlying model or standard
Content preservation	WER	Whisper ASR transcription of source and converted speech [32]
Naturalness	UTMOSv2	neural MOS predictor [2]
Naturalness	DNSMOS	non-intrusive predictor aligned with ITU-T P.835 [36, 34]
Speaker identity	SECS measures	cosine of ECAPA-TDNN embeddings [11]; reference-free variants (Ch. 4)

2.6 Evaluation of Voice Conversion

Voice conversion evaluation has several dimensions: a good conversion preserves content, sounds natural, and reaches a target identity distinct from the source. No single metric captures all three, and a system can win one by sacrificing another (an unmodified target recording scores perfect naturalness and similarity with zero content). The literature therefore uses a portfolio of objective and subjective measures [42, 4], as does this thesis (Table 2.3).

2.6.1 Subjective Evaluation

The historical gold standard is the *mean opinion score* (MOS), in which listeners rate utterances on a five-point naturalness scale. The Voice Conversion Challenge series has used MOS and similarity tests as its principal evaluation throughout, from the parallel-data systems of 2016 to the cross-lingual task of 2020 [44, 51]. Subjective testing remains the most trustworthy form of evaluation but is also the most resource-intensive: a properly powered MOS study requires tens to hundreds of listeners, careful counterbalancing, and an ethics review. The empirical chapter of this thesis therefore relies on objective surrogates, while acknowledging that they are only proxies for human judgement.

2.6.2 Objective Signal-Level Metrics

Mel-cepstral distortion (MCD) [20] computes the Euclidean distance between mel-cepstral coefficient vectors, a compact description of the short-term spectral envelope, extracted from a reference utterance and its converted counterpart, typically after aligning the two in time with dynamic time warping (DTW), an algorithm for aligning two sequences that may differ in timing [41]. MCD is a classical metric whose interpretation is well established but whose correlation with modern subjective ratings is imperfect, particularly for systems whose artefacts are not primarily spectral.

PESQ [37] is a perceptual quality estimator originally designed for telephone-band speech. Its successor **POLQA**, standardised as ITU-T Recommendation P.863 [35], addresses wideband speech but is licence-encumbered. **ViSQOL**, in its open-source v3 implementation [8], is an alternative spectro-temporal similarity metric applicable to wideband speech and music. Both PESQ and ViSQOL were developed for codec evaluation rather than for VC and should be interpreted accordingly; they are nonetheless reported in the VC literature as supplementary indicators. A structural limitation shared by all metrics in this category is that they require a reference signal to compare against, which is awkward for conversion: the ideal output (the target speaker uttering the source content) usually does not exist as a recording.

2.6.3 Objective Downstream Metrics

Two downstream-model-based metrics have become standard in the modern VC literature, and both sidestep the missing-reference problem by measuring properties of the converted audio directly.

Word error rate (WER), computed by transcribing source and converted speech with an automatic speech recognition (ASR) system and comparing the resulting transcripts, is the principal objective measure of content preservation. The empirical chapter of this thesis uses Whisper [32] for this purpose, which provides robust transcription across a wide range of acoustic conditions. The implicit assumption is that the ASR system is itself robust to conversion artefacts; where it is not, WER conflates intelligibility loss with ASR brittleness. This conflation is one of the threats to validity examined in Chapter 4 and revisited in the discussion of objective evaluation in Chapter 7.

Speaker encoder cosine similarity (SECS) measures speaker similarity in the embedding space of a pretrained automatic speaker verification (ASV) model. Embeddings are extracted from the converted utterance and from one or more reference utterances of the target speaker, and the cosine similarity of the mean vectors is reported. The empirical chapter uses ECAPA-TDNN [11] as the embedding extractor. As with WER, the metric is only as trustworthy as the underlying model: SECS measures similarity as perceived by one particular verification model, not by human listeners, and the two can diverge.

2.6.4 Neural MOS Predictors

To approximate subjective MOS scores without conducting listening tests, recent work has produced neural *MOS predictors* that map an input waveform to a predicted naturalness score, trained on corpora of human-rated synthetic speech.

MOSNet [23] was an early entry. **UTMOS** [40] combined self-supervised speech features with ensemble learning and performed strongly in the VoiceMOS Challenge 2022; its successor **UTMOSv2** [2] adds a spectrogram-based image-classification branch alongside the SSL features and achieved the top ranking in the great majority of metrics in the VoiceMOS Challenge 2024 track on high-quality synthetic speech. The VoiceMOS Challenge series itself [9] provides the systematic benchmarking evidence for how well these predictors generalise across systems and domains, and is the basis on which this thesis justifies its choice of predictor. **DNSMOS** [36] is a non-intrusive quality predictor aligned with the ITU-T P.835 standard, originally targeted at noise-suppressed speech but in routine use as a secondary quality indicator for synthesis tasks.

The empirical chapter of this thesis uses UTMOSv2 as its primary naturalness metric, with DNS-MOS reported as a secondary metric for cross-validation. The decision to rely on predicted MOS rather than subjective MOS is a deliberate scope choice driven by the absence of resources for a full subjective study; the implications, including the known caveat that MOS predictors can mis-rank systems whose artefacts differ from those in their training data, are discussed in Chapter 7.

2.6.5 The Evaluation Gap in Real-Time Voice Conversion

A consistent observation across the real-time VC literature is that latency is reported under measurement protocols that vary substantially between papers: some authors report wall-clock latency on mobile hardware [49], others on desktop CPUs [39, 22], others on server-class GPUs with unspecified buffer sizes. Different papers report different subsets of the latency components (audio buffering, look-ahead, compute, queueing, transport), usually without saying which. Quality metrics similarly vary in the choice of evaluation corpus and predictor model. This evaluation heterogeneity makes cross-paper comparison unreliable, and it is even more pronounced for deployed tools, whose quality is typically not measured at all. It motivates the unified empirical benchmark presented in Chapter 6, in which all of the deployed tools are measured with the same protocol, on the same corpus, on the same hardware. The benchmark addresses the quality dimensions of this heterogeneity, namely intelligibility, naturalness, and speaker identity, under a common protocol; closing the analogous gap for latency is left to future work.

2.7 Summary and Outlook

This chapter has introduced the vocabulary on which the remainder of the thesis depends. Voice conversion is the task of transforming a source speech signal to match the identity of a target speaker while preserving linguistic content. The field progressed from parallel-data statistical regression, through adversarial and disentanglement-based non-parallel methods, to the content-encoder and vocoder pipelines that dominate today, in which a self-supervised content encoder produces an intermediate representation that a neural vocoder or codec renders as a waveform. Real-time operation imposes two simultaneous constraints, bounded computational cost and bounded end-to-end latency, the latter depending on audio buffering, model look-ahead, compute time, internal queueing, and any network transport, of which only the look-ahead is a pure property of the model. The peer-reviewed real-time VC literature has converged on causal soft-unit content encoders paired with neural-codec or causal-GAN waveform generators, with state-of-the-art end-to-end latencies in the tens of milliseconds on consumer hardware. Evaluation is multidimensional, spanning signal-level, downstream-model-based, and subjective measures, with neural MOS predictors increasingly serving as proxies where subjective testing is infeasible.

Three implications follow for the work presented in subsequent chapters. First, the "real-time" labels attached to deployed tools conceal very different latency behaviours, since superficially similar numbers can reflect different architectural and deployment choices; this thesis therefore treats a controlled latency benchmark as future work (Chapter 8) rather than as something it measures. Second, the choice of evaluation metric is consequential: WER, the SECS measures, and predicted MOS each respond to different failure modes of a VC system, and a complete picture requires reporting all of them rather than optimising the presentation around any single one, which is exactly what the benchmark of Chapter 4 measures. Third, the gap between peer-reviewed methods and the architectures embodied in widely used open-source and commercial tools is itself the central object of study: the literature review of Chapter 3 maps the academic landscape, and the empirical benchmark of Chapter 6 situates the deployed tools within that landscape under a common quality-measurement protocol.

Chapter 3

Structured Literature Review

The background introduced voice conversion and real-time operation in the abstract; this chapter surveys the literature that applies them, through a documented protocol rather than ad-hoc reading. It has two purposes: to map the research landscape of streaming voice conversion, so the deployed tools can be placed against published methods, and to build the architectural taxonomy that Chapter 7 later uses to read the results.

Two terms recur throughout the thesis and are worth fixing here. A *system* is a published research method (such as StreamVC or RT-VC), a *tool* is a deployed piece of software an end user can download or buy and run (such as w-okada, Vonovox, or Seed-VC). This thesis benchmarks tools and places them against the systems the research community has published.

3.1 Review Protocol

3.1.1 Databases and Search Strategy

Four sources were searched. Google Scholar served as the main discovery tool, because of its broad coverage across publishers and its indexing of preprints, which matter in a field where several of the most relevant systems first appeared on arXiv. IEEE Xplore was searched directly because a large share of speech and signal-processing work, including the ICASSP and speech-workshop proceedings most relevant here, is published there. Scopus and NTNU Oria were used to reach material not openly available, using the institutional access available through the university. Backward and forward snowballing complemented the database searches: the reference lists of strongly relevant papers were followed backward, and their citing papers forward, to catch work that the keyword searches missed.

The search combined groups of terms, one naming the task and one naming the real-time constraint or the evaluation focus, in the spirit of the queries in Table 3.1. The task group named the task itself (*voice conversion*, *voice changer*). The constraint group named real-time operation (*real-time*, *streaming*, *low-latency*, *causal*). A third group, used when surveying methods and evaluation, named techniques and metrics (*self-supervised*, *vocoder*, *neural codec*, *naturalness*, *speaker similarity*, *mean opinion score*). Queries combined one term from each of two or three groups.

3.1.2 Inclusion and Exclusion Criteria

Records were judged on two dimensions: topical relevance and methodological strength. A record was *included* if it presented a voice conversion method or an evaluation of voice conversion, with priority given to work addressing real-time or streaming operation, and if it reported either an objective or a subjective evaluation or else a method described in enough detail to place it in the

Table 3.1: Representative search queries, as combinations of a task term with a constraint or evaluation term.

Focus	Example query
Real-time systems	(“voice conversion” OR “voice changer”) AND (“real-time” OR “streaming” OR “low-latency” OR “causal”)
Content representations	“voice conversion” AND (“self-supervised” OR “HuBERT” OR “WavLM” OR “content encoder”)
Waveform generation	“voice conversion” AND (“vocoder” OR “neural codec” OR “HiFi-GAN”)
Evaluation	“voice conversion” AND (“naturalness” OR “speaker similarity” OR “mean opinion score” OR “word error rate”)

taxonomy. Foundational works that are not themselves about real-time operation, such as the self-supervised representations and vocoders of Chapter 2, were included where they are necessary to understand the streaming systems that build on them. A record was *excluded* if it was not about voice conversion, if it offered neither a usable evaluation nor sufficient methodological detail, if it was superseded by a later version by the same authors, or if the full text could not be retrieved.

Preprints were treated as a special case. The real-time voice conversion literature moves quickly, and several of the systems most relevant to this thesis were, at the time of writing, available only as preprints that had not completed peer review. Excluding them would have meant excluding part of the state of the art. They were therefore admitted, but flagged as preprints and read with corresponding caution: their claims, and especially their reported latency and quality figures, are reported as the authors state them while noting that they have not been independently refereed. This treatment is applied to the low-latency and articulatory-coding systems discussed below, both of which are cited from their preprint form.

3.2 Peer-Reviewed Real-Time Voice Conversion

The offline methods that built the field, the parallel-data statistical approaches, the non-parallel adversarial and disentanglement methods, and the content-encoder and vocoder pipelines that dominate today, were already covered in Chapter 2 and are not repeated here [42, 4]. What matters now is the small, recent group of *streaming* systems, the research-literature counterpart of the deployed tools. A handful define the space: StreamVC distils a causal content encoder onto a neural-codec decoder [49]. RT-VC uses a causal articulatory representation with a differentiable DSP vocoder [22]. LLVC distils a small any-to-one network for very low latency [39]. The retrieval method kNN-VC, although offline, is the conceptual anchor for the deployed RVC tools, since it conditions on the target by matching nearest-neighbour self-supervised features rather than a learned embedding [1]. The one fact about these systems that bears on the thesis is that they do report end-to-end latency, in the tens of milliseconds, but each under its own protocol and hardware (Section 2.6.5), so even the published numbers do not compare cleanly with one another.

3.2.1 Positioning the Deployed Tools

Table 3.2 places the deployed tools alongside these research systems on a few architectural axes; the detail matters less than the pattern in its final column. The deployed tools sit in recognisable positions: the RVC tools on the retrieval branch, Seed-VC alongside the any-to-any reference-conditioned systems, and Beatrice. Yet not one of them has a published quality figure, which is the gap the benchmark of this thesis fills.

Table 3.2: Architectural taxonomy of the research streaming systems and the deployed tools that implement each design. w-okada (VC Client) hosts several model formats, so it appears on each row whose format it runs (RVC and Beatrice).

VC System	Content repr.	Cardinality	Waveform generator	Deployed tools
StreamVC [49]	causal soft units	any-to-any	neural codec decoder	—
RT-VC [22]	causal articulatory	any-to-any	DDSP vocoder	—
LLVC [39]	distilled, causal	any-to-one	compact GAN	—
kNN-VC [1]	WavLM, retrieval	any-to-any	HiFi-GAN	—
RVC [38]	SSL (HuBERT/ ContentVec) + retrieval	any-to-many	HiFi-GAN style (NSF)	RVC WebUI [38]; w-okada [48]; Applio [15]; Echo [13]; Vonovox [47]; Voice.ai [45]
Seed-VC [21, 28]	reference (in-context)	any-to-any	BigVGAN	Seed-VC (real-time) [28]
Beatrice [29]	causal phone + pitch (StreamVC-like)†	any-to-many	in-house neural	Beatrice VST/client [29]; w-okada [48]
Proprietary (closed)	undisclosed	any-to-many (fixed lib.)	undisclosed	Voicemod [46]; Dubbing AI [12]

3.3 The Gap and Its Consequence

Two gaps emerge from the review. The first is the evaluation heterogeneity already named in Section 2.6.5: even among the peer-reviewed streaming systems, latency and quality are reported on different hardware, with different corpora, and with different metrics, so that the published numbers cannot be compared across papers without heavy caveats. The second, sharper gap is the one this thesis addresses. The tools that people actually use, the RVC hosts, the zero-shot reference-conditioned tools, the low-latency designs, and the commercial tools, sit squarely in the same architectural space as the research systems, but their quality has never been measured under any common protocol at all. Their published evidence is marketing copy and demonstrations, not metrics.

The consequence is the design of the rest of this thesis. If the deployed tools cannot be compared to the peer-reviewed systems on latency, because the deployed tools report none, then a latency benchmark is a substantial undertaking in its own right and is left to future work (Chapter 8). What can be done now, and what the literature most conspicuously lacks, is a common-protocol measurement of the deployed tools' *quality*. That is the benchmark specified in Chapter 4 and reported in Chapter 6, and the taxonomy of Table 3.2 is what lets its results be read as statements about kinds of system rather than about individual tools.

Chapter 4

Methodology of Evaluation

The preceding chapters established what voice conversion is, why real-time operation is a distinct engineering problem, and how the field evaluates conversion quality. This chapter turns from background to method. It specifies the empirical study that is the central contribution of this thesis: a unified, reproducible, objective quality benchmark applied to deployed real-time voice conversion tools under one protocol, on one corpus, on one machine.

The chapter is organised as a path from intent to procedure. Section 4.1 states the research questions and the design principles that follow from them. Section 4.2 introduces the tools and the architectural distinctions that make them worth comparing. Section 4.3 describes the evaluation corpus, and Section 4.4 the conversion directions and target voices. Section 4.5 defines the metric portfolio precisely, including the reference-free speaker-identity proxies that the absence of clean target recordings forces upon the study. Section 4.6 documents the measurement apparatus and the single-configuration protocol, and Section 4.7 closes with a structured analysis of the threats to validity, several of which become the future-work agenda of Chapter 8. The software that implements the protocol is described separately in Chapter 5. This chapter is concerned with the experimental design rather than its code.

4.1 Research Questions and Design Principles

The motivation developed in the previous chapters is that "real-time voice conversion" is a label attached to a heterogeneous set of deployed tools whose output quality is, in practice, never measured under a common protocol. Vendors publish marketing claims, open-source projects publish demonstrations, and the peer-reviewed literature reports quality on incompatible corpora with incompatible metrics (Section 2.6.5). A practitioner choosing a tool, or a researcher situating a new system, has no neutral point of comparison. This thesis addresses that gap for the quality dimension. The two research questions defined in Chapter 1.3 are restated here.

RQ1 (Comparison). How do widely used real-time voice conversion tools compare in objective output quality, across intelligibility, naturalness, and speaker identity, when they are all measured under one common protocol?

RQ2 (Cost versus quality). Does better quality come at a higher cost to the user, whether in money or in the computing resources a tool demands, or do free and lightweight tools keep up with paid and resource-heavy ones?

The first question is the central one: it asks for the neutral comparison the field lacks. The second turns to what that comparison is worth to someone choosing a tool. Deployed tools differ enormously in what they ask of the user, so a natural question is whether paying more, or running the most resource-hungry program, actually buys better output. Answering RQ2 needs the resource measurements of Section 4.6.4 alongside the quality scores, and it is helped by a coincidence in the

Table 4.1: The six tools evaluated, in seven configurations. w-okada appears twice, with an RVC model and a Beatrice model, and its RVC run shares model files with Vonovox. Cardinality follows the taxonomy of Section 2.1.4.

Tool	Model	Cardinality	Year	OS	Notes
w-okada / RVC [48, 38]	retrieval-based (RVC)	any-to-many	2022	Win, macOS, Linux	open source; per-target RVC model plus retrieval index
w-okada / Beatrice [48, 29]	low-latency (Beatrice)	any-to-many	2023	Win, macOS, Linux	same host as the row above, different model type
Vonovox [47]	retrieval-based (RVC)	any-to-many	2024	Win	Source-visible (no license); loads the same RVC model files as w-okada; NVIDIA GPU optimized
Seed-VC [28]	diffusion transformer	any-to-any	2024	Win, macOS, Linux	open source; target is a reference clip at inference time; the only tool with a clean target recording
Voice.ai [45]	retrieval-based (RVC)	any-to-many	2021	Win, macOS	closed tool; loads user RVC models plus preset voices
DubbingAI [12]	commercial, preset voices	any-to-many	2023	Win, macOS	closed tool; target is a vendor preset
Voicemod [46]	commercial, preset voices	any-to-many	2014	Win, macOS	closed tool; target is a vendor preset

tool set: w-okada and Vonovox load the same RVC model files but run them very differently, so the pair shows directly what extra resources buy when the model is held fixed.

Four design principles follow from these questions and recur throughout the chapter. First, *the benchmark measures quality, not latency.* The reasons were given in Section 2.2: latency is the property users care about most, but measuring it under a controlled protocol is a separate undertaking with its own apparatus, and conflating a half-measured latency figure with a quality ranking would weaken both. Latency is therefore the foremost item of future work (Chapter 8), and the present study is explicit that its scope is output quality. Second, *the benchmark is black-box.* Several of the tools are closed commercial software that exposes no internal representations and no programmatic interface, so to treat all tools identically the protocol must observe only what every tool exposes, which is audio in and audio out. Third, *the benchmark is reference-free wherever possible.* The ideal reference for a conversion, the target speaker uttering the source content, does not exist as a recording, so the metric portfolio is built from measures that judge the converted audio directly. Fourth, *the benchmark holds the environment fixed.* One corpus, one machine, one capture path, and one representative configuration per tool remove as many confounds as a single-operator study can, at the cost of the external-validity limits discussed in Section 4.7.

4.2 Tools Under Evaluation

Six deployed tools are evaluated, in seven configurations. A *configuration* is one tool run in one specific setup, the count is seven rather than six because w-okada is run with two different models. They were selected to span the architectures surveyed in the background, to include both free open-source and paid commercial tools, and to cover a wide range of how demanding a tool is to run, which is what RQ2 turns on. None is a research prototype: every one is a tool that a real user can download or purchase and run today, which is precisely the population the thesis is about. Table 4.1 summarises them.

A few of the groupings matter later. The two RVC configurations, w-okada and Vonovox, are built on the open-source Retrieval-based Voice Conversion project [38], which pairs a self-supervised content encoder with a retrieval step over target-speaker features and a HiFi-GAN-style synthesiser, in the lineage traced in Sections 2.3 and 2.5. Each target voice is a separately trained model file with an associated feature index. Because both hosts load the identical files, any difference between them comes from the surrounding application rather than from the conversion model, which is what makes the pair useful for RQ2: it shows what a heavier host buys when the model

is held fixed. w-okada is additionally run with a Beatrice model [29], a low-latency converter of a different design, included to widen the range of tools rather than as a controlled comparison, since it shares no backend with the RVC runs. Seed-VC represents the zero-shot any-to-any approach. Unlike the retrieval step of the RVC tools, it is built on a diffusion transformer [28], conditioned on a short reference clip of the target voice supplied at inference time. It requires no per-target training, which is also the reason it is the one tool for which a clean target recording exists in this study (Section 4.5.3). The three commercial tools, DubbingAI, Voicemod, and Voice.ai, expose only a library of preset target voices and no internal representations. They stand in for the large population of consumer tools whose quality is asserted by marketing rather than measured.

4.3 Evaluation Corpus

The benchmark uses a fixed corpus of clean source utterances drawn from two public English speech datasets, chosen to contrast two acoustic registers. From **LibriSpeech** [27], a corpus of read audiobook speech recorded under clean, consistent conditions, the study draws prepared, articulate, low-noise utterances. From **GigaSpeech** [6], a multi-domain corpus that includes podcasts and online video, it draws more spontaneous and acoustically varied speech. Using both lets the benchmark report not only how the tools rank overall but how their quality degrades as the input moves from studio-clean reading to looser, real-world speech, a contrast that turns out to separate the tools more sharply than their overall means do (Chapter 6).

The working corpus comprises 69 source clips: 34 from LibriSpeech (17 female and 17 male speakers) and 35 from GigaSpeech (16 female and 19 male). Each clip carries a ground-truth transcript, used as the reference for word error rate. Utterances were selected to be between 5 and 30 seconds long, long enough to give the metrics a stable signal and the speaker embedder enough material to characterise a voice, and were kept disjoint in identity across the two corpora so that no speaker is counted twice. The corpus is deliberately small and fixed rather than large and redundant: every clip is converted by every tool in every direction, so the design is fully crossed, and a larger corpus would multiply capture time without changing the qualitative findings. The size is, nonetheless, a limit on statistical power, and Section 4.7 returns to it.

4.4 Conversion Directions and Target Voices

Every source clip is converted twice by each tool, once into a female target voice and once into a male target voice. Crossing the two source genders with the two target genders yields four directions, labelled by source-to-target gender: F_to_F, F_to_M, M_to_F, and M_to_M. The 33 female source clips populate the two female-source directions and the 36 male source clips the two male-source directions, giving 138 converted utterances per configuration and 966 across the seven configurations. Reporting per direction, rather than collapsing to a single number, exposes two effects that the background predicts should exist: cross-gender conversion is a harder identity move than same-gender conversion, and the difficulty of a direction may depend on the target voice as much as on the source.

The target voices differ by necessity across tools, because each tool expresses a target in its own terms: an RVC or Beatrice model file for w-okada and Vonovox, a reference clip for Seed-VC, and a built-in preset for the commercial tools. For the RVC pair the female and male targets are the *same* two model files, so that pair is matched not only on the conversion model but on the target identity. For the remaining tools the female and male targets are the closest available equivalent in each tool's own ecosystem. The exact target voices, model files, reference clips, and presets are recorded in the project repository. A consequence of this arrangement, important for interpreting the results, is that a direction label such as F_to_M denotes the *same source set* but a *different target identity* across tools. Within-tool comparisons across directions, and the matched RVC-pair comparison, are therefore on firmer ground than cross-tool comparisons of absolute target similarity, a point that Section 4.5.3 makes precise and that the discussion in Chapter 7 respects.

4.5 Metric Portfolio

No single number captures conversion quality, for the reason given in Section 2.6: a system can score well on one dimension by sacrificing another. The benchmark therefore reports a portfolio spanning the three quality dimensions introduced in Section 2.6, intelligibility, naturalness, and speaker identity, each with a primary metric and, where useful, a secondary cross-check. All metrics are computed on audio resampled to a common 16 kHz mono representation, so that differences between tools reflect the conversions rather than incidental differences in sample rate or channel layout. Every metric in the portfolio is reference-free in the sense that it requires no recording of the ideal output; the speaker-identity metrics additionally require no recording of the target speaker for every configuration but Seed-VC, which is the subject of Section 4.5.3.

4.5.1 Intelligibility: Word and Character Error Rate

Content preservation is measured by word error rate (WER), the standard objective proxy for intelligibility (Section 2.6). The converted utterance is transcribed by an automatic speech recognition system, and the resulting transcript is compared with the ground-truth transcript of the source clip. With S, D, and I the number of substitutions, deletions, and insertions in the optimal alignment of hypothesis to reference, and N the number of reference words,

$$\mathrm{WER} = \frac{S + D + I}{N}. \tag{4.1}$$

Character error rate (CER) is defined identically over characters, and is reported alongside WER because it is less sensitive to single-word substitutions and to tokenisation, giving a finer-grained view of intelligibility loss. Both are 0 for a perfect transcript and grow with all errors (S, D, and I), so lower is better. Both transcripts are passed through a standard English text normaliser before alignment, so that differences in casing, punctuation, and number formatting do not inflate the error rate. The ASR system is Whisper [32], chosen for its robustness across acoustic conditions, and the precise model and decoding settings are given in Chapter 5. The interpretive caveat from Section 2.6 applies: WER assumes the ASR system is itself robust to conversion artefacts, and where it is not, the metric conflates genuine intelligibility loss with ASR brittleness. This is one reason a single ASR is listed among the threats to validity in Section 4.7.

4.5.2 Naturalness: Predicted MOS

Naturalness is measured with neural mean-opinion-score predictors, which approximate the result of a listening test without conducting one (Section 2.6). The primary metric is **UTMOSv2** [2], a trained neural predictor that, unlike WER or SECS, has no closed-form definition. It was the top-ranked predictor in the great majority of metrics of the VoiceMOS Challenge 2024 track on high-quality synthetic speech, and maps an utterance to a single predicted naturalness score on the familiar five-point scale (1 to 5, higher is better). As a secondary cross-check the benchmark reports **DNSMOS** [36], a non-intrusive predictor aligned with ITU-T Recommendation P.835 [34] that returns separate speech-distortion (SIG), background-noise (BAK), and overall (OVRL) scores together with a P.808 score. DNSMOS was designed for noise suppression rather than synthesis, so it is read as corroboration rather than as an independent primary measure: where UTMOSv2 and DNSMOS agree, confidence in a naturalness ranking is higher, and where they diverge the divergence is itself informative. Both predictors are higher-is-better and reference-free, which makes them well suited to judging the artefacts a real-time converter introduces. The known limitation, that a predictor can mis-rank systems whose artefacts differ from those in its training data, is carried forward to Section 4.7 and revisited in the discussion.

4.5.3 Speaker Identity: the SECS Family

Speaker identity is measured in the embedding space of a pretrained automatic speaker verification model, following the speaker-encoder cosine similarity (SECS) approach of Section 2.6. Each utterance is mapped to a fixed-dimensional embedding by an ECAPA-TDNN model [11], and the similarity of two such embeddings $\mathbf{a}$ and $\mathbf{b}$ is their cosine similarity,

$$\mathrm{SECS}(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \, \|\mathbf{b}\|}. \tag{4.2}$$

The score lies between -1 and 1 (in practice positive for speaker embeddings), and a higher value means the two utterances are closer in speaker identity.

The canonical use of Equation 4.2 compares the converted utterance with a clean recording of the target speaker. That recording does not exist for any configuration but Seed-VC. The RVC tools ship their targets as trained model files, not as audio, and the commercial tools ship theirs as opaque presets; only Seed-VC, which is conditioned on an actual reference clip, comes with a clean target recording. This is not an oversight in the experimental design but the normal situation for a deployed any-to-one tool, and confronting it honestly is part of what the benchmark contributes. The benchmark therefore reports a *set* of identity measures, of which only one is the canonical metric.

Source similarity (src_sim**).** The cosine similarity between the converted utterance and the original source speaker, $\mathrm{SECS}(\mathbf{e}_{\mathrm{out}}, \mathbf{e}_{\mathrm{src}})$. A successful conversion moves identity *away* from the source, so for this measure *lower is better*. It is available for every tool, because the source recording always exists, and it answers the question "did the identity move at all?"

Target consistency (tgt_cons**).** The mean pairwise cosine similarity among all of a tool's outputs in a given group G (one tool, one direction),

$$\mathrm{tgt\_cons}(G) = \binom{|G|}{2}^{-1} \sum_{i<j} \mathrm{SECS}(\mathbf{e}_i, \mathbf{e}_j). \tag{4.3}$$

A tool that lands on one stable target voice regardless of which source went in will produce mutually similar outputs, so *higher is better*. Like source similarity, it needs no target recording, and it answers the question "does the tool converge on a single voice?"

Target similarity (tgt_sim**).** The canonical metric, $\mathrm{SECS}(\mathbf{e}_{\mathrm{out}}, \mathbf{e}_{\mathrm{tgt}})$, where $\mathbf{e}_{\mathrm{tgt}}$ is the embedding of a clean target recording. *Higher is better*. It is reported only for Seed-VC, the one tool for which $\mathbf{e}_{\mathrm{tgt}}$ exist.

Source similarity and target consistency are complementary but neither is a substitute for the canonical metric, and their failure modes matter for how the identity results are read. Source similarity detects whether identity moved, but a low value can be produced by a genuine conversion or by mere degradation: noise and artefacts also push an embedding away from the source. Target consistency detects convergence onto a single voice, but it is blind to *which* voice, and it is vulnerable to mode collapse, the case in which a tool maps every input toward a narrow timbre. A degenerate tool that emitted the same constant hum for every input would score a perfect consistency while converting nothing. The proxies are therefore only interpretable *together with* the intelligibility and naturalness metrics: high consistency is evidence of a stable target only when naturalness and WER confirm that the output is still speech. Whether this triangulation is sufficient to rank tools, and how the proxies line up against the canonical metric where both are available, is taken up directly in the results (Section 6.4) and discussion.

Table 4.2 consolidates the portfolio, restating for each metric its quality dimension, its direction, and whether it requires a target recording.

Table 4.2: The metric portfolio. Each quality dimension, intelligibility, naturalness, and speaker identity, with the direction of improvement ("Better") and the underlying model noted for each.

Metric	Dimension	Better	Underlying model
WER, CER	intelligibility	lower	Whisper ASR [32]
UTMOSv2	naturalness	higher	neural MOS predictor [2]
DNSMOS (SIG/BAK/OVRL/P.808)	naturalness	higher	P.835 predictor [36]
src_sim	identity (moved off source)	lower	ECAPA-TDNN [11]
tgt_cons	identity (target stability)	higher	ECAPA-TDNN [11]
tgt_sim	identity (canonical)	higher	ECAPA-TDNN [11]

4.6 Measurement Protocol and Apparatus

4.6.1 Black-box Capture

Because the tool set includes closed commercial software, the protocol observes only audio in and audio out. Each clean source clip is played out through a virtual audio cable into the running tool, and the tool's converted output is recorded through a second virtual cable, trimmed of leading and trailing silence, and saved. The arrangement treats every tool identically, whether open source or commercial, and captures exactly what a user would hear, including any artefacts introduced by the tool's own audio stack rather than by its conversion model alone. The capture harness, the cable routing, the silence trimming, and the resampling are described in detail in Chapter 5; what matters for the experimental design is that the captured signal is the tool's real output under ordinary use, not an idealised offline render.

4.6.2 Tool Configuration

Each tool is measured at a single representative configuration: the settings under which it runs correctly and is intended to be used, left at or near the vendor defaults that an ordinary user would encounter. This is a deliberate scope decision. The tools expose very different and largely incomparable configuration surfaces, an RVC host exposes pitch and index-blend controls that a commercial preset does not, and exploring each tool's configuration space would turn the comparison into several separate tuning studies. Fixing one configuration per tool keeps the comparison tractable and reproducible, at the cost of not characterising how each tool behaves across its own settings. Section 2.2.3 noted that buffer size in particular trades latency against robustness and can change a tool's behaviour. Where a tool exposed such latency-affecting controls, they were set to keep end-to-end latency in a usable range of roughly 200 to 400 ms, so the tools are compared at broadly comparable latency rather than at one fixed value. Systematically sweeping buffer sizes and other settings is left to the latency-focused future work of Chapter 8, where it belongs alongside the latency measurement it most affects.

4.6.3 Apparatus

All captures and all scoring were performed on a single Windows 11 machine with an NVIDIA GeForce RTX 5070 GPU, Intel Core Ultra 7 CPU, DDR5 6000MHz 32GB memory and the virtual-audio-cable used was VB-cable. Holding the hardware fixed removes it as a confound between tools: every tool ran on the same CPU and GPU, every metric model ran on the same machine, and the capture path was identical throughout. It also bounds external validity, since a tool's quality, and especially its real-time behaviour, can depend on the hardware it runs on, a point revisited in Section 4.7. The neural metric models, the ASR system, the MOS predictors, and the speaker

embedder, are pinned to specific checkpoints, listed in Chapter 5, so that re-running the scoring on the same audio reproduces the same numbers.

4.6.4 Resource Usage

Alongside output quality, the benchmark records how demanding each tool is to run, because a real-time tool that produces excellent audio is of little use if it saturates the machine it shares with a game or a video call. For each tool the steady-state processor, memory, and graphics-processor utilisation are observed while it converts, using the ordinary operating-system tools that any user has to hand. The exact procedure is described in Chapter 5. These figures are coarse and machine-specific, not precise measurements, and they are reported as a practical deployability indicator rather than as a quality metric. They nonetheless add a dimension the quality scores cannot capture: whether a tool reaches its quality cheaply, on the processor alone, or only by committing substantial memory and graphics-processor time.

4.7 Threats to Validity

The benchmark is a single-operator study with deliberately narrow scope, and its conclusions must be read against the threats catalogued here: the limits of the metrics, the limits of how far the results generalise, the absence of statistical inference, and the exclusion of latency.

Construct validity. Every metric in the portfolio is a proxy for a property that is ultimately perceptual. Predicted MOS is not subjective MOS, and a predictor can mis-rank systems whose artefacts differ from those in its training distribution. This is mitigated, not removed, by reporting two predictors (UTMOSv2 and DNSMOS) and treating their agreement as the unit of evidence. WER depends on a single ASR system, so a tool that happens to produce artefacts this particular recogniser is brittle to would be scored unfairly, and a second, independent ASR would cross-check this. The speaker-identity proxies are the weakest link of all: with no clean target recording for most of the tools, source similarity and target consistency stand in for true target similarity, and both have the failure modes detailed in Section 4.5.3. The identity dimension is therefore measured less directly than the other two, a point the discussion returns to.

Generalisability. Several choices bound how far the results carry. Each tool was measured at one configuration near its defaults, so the results describe each tool *as configured*, not at its unknown best. Because each tool expresses its targets in its own terms, the female and male target identities are matched only within the RVC pair, so a direction label fixes the source set but not the target identity across tools. And a single machine was used, so behaviour on weaker hardware is uncharacterised, while the commercial tools are moving targets whose behaviour can change between versions, making the measurements a snapshot pinned to the benchmarked builds.

Latency. Finally, and most importantly, the study measures quality and not latency. The tools are real-time systems whose users care most about end-to-end delay (Section 2.2), and measuring it properly, across the configurations and buffer sizes that latency depends on, is a study in its own right. A complete picture of a real-time voice conversion tool requires both quality and latency under controlled protocols; this thesis supplies the former and leaves the latter to future work. The implementation that produced the quality measurements is the subject of the next chapter.

Chapter 5

Implementation of Evaluation

The methodology of the previous chapter describes an experiment. This chapter describes the software that carries it out.[1] The benchmark is a small Python system in two halves: a *capture* harness that drives each running voice conversion tool and records its output, and a *scoring* pipeline that rates the captured audio on the metric portfolio of Section 4.5. The two halves are deliberately decoupled. Capture is slow, manual, and tied to live audio hardware, because a tool must actually be running and routed through the audio system. Scoring, by comparison, is fully automatic, repeatable, and hardware-light. Separating them means the expensive capture step is performed once per tool and its outputs can be re-scored as often as needed, for instance when a metric is added or a checkpoint is updated, without touching the tools again.

Section 5.1 gives the dataflow and the shared configuration. Section 5.2 describes the capture harness and the black-box routing it depends on. Section 5.3 explains how captured files are matched back to their source clips and transcripts. Section 5.4 covers the scoring pipeline and its caching strategy, Section 5.5 the individual metric implementations, and Section 5.6 the outputs. Section 5.8 collects the choices that make the results reproducible.

5.1 Overview and Configuration

Figure 5.1 shows the dataflow. A corpus of clean source clips feeds the capture harness, which plays each clip into a running tool and saves the converted result under a per-tool directory. The scoring pipeline then reads those converted files, rejoins each to its source clip and ground-truth transcript, computes the metrics, and writes per-file and summary tables. A single `config.toml` at the project root configures both halves: shared input and output paths, the audio devices and capture timing under a capture section, and the metric set and speaker-similarity options under a benchmark section.

The directory layout encodes just enough structure for the pipeline to interpret the files without a database:

```
config.toml                    config
record_vc.py / capture/        capture harness
score_vc.py  / benchmarks/     scoring pipeline
data/corpus/<corpus>/          clean sources: {Female,Male}/ + transcripts/
data/converted/<tool>/<cfg>/   captured outputs, named by source id
data/scores/                   per_file.csv, summary.csv, summary_by_corpus.csv
```

The corpus is organised by corpus name and then by speaker gender, with a parallel folder of

[1] The complete source code, configuration, and capture metadata are available at https://github.com/eivindnesje/Benchmarks.

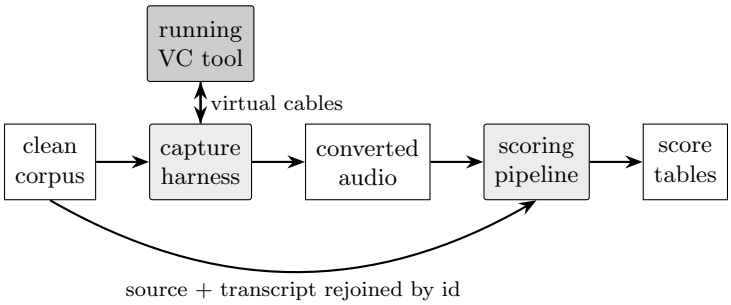

Figure 5.1: Dataflow of the benchmark. The capture harness records each tool's converted output, and the scoring pipeline rejoins it to the source clip and transcript by utterance id before writing the score tables. The two halves communicate only through the converted audio on disk.

transcripts, and the converted outputs are organised by tool and then by the configuration label used during capture. Both layouts are conventions the code reads rather than schemas it enforces, which keeps the harness usable with whatever folder structure a given tool imposes.

5.2 The Capture Harness

The capture harness is the part of the system that makes a black-box benchmark of closed tools possible. It is invoked once per tool, with the tool already running and routed, and it plays every selected source clip through the tool while recording the result.

5.2.1 Setting Up Each Tool

Before the harness can drive a tool, the tool has to be installed, routed to the virtual cables, and loaded with a target voice, and the effort this takes varies a lot. Voicemod and DubbingAI are the simplest, a standard installer followed by picking a preset voice. Voice.ai is also a commercial installer, but it is noticeably less polished, with a number of quirks and little documentation on how to drive it. w-okada and Vonovox are very similar to each other: each ships as a zip of the project that runs from a simple batch script, and both expose a frontend that, while a little more involved than the commercial installers, allows considerably more customisation than any of them. Seed-VC also installs from its source project and starts from a batch file, but that batch file opens its offline, non-real-time conversion. Running the real-time GUI instead means setting up a Python environment by hand and launching the project manually. This setup effort is noted only because it is a real cost to a user choosing a tool. It does not affect the captured audio, which is what the benchmark scores.

5.2.2 Black-box Routing

Audio is routed between the harness and the tool through two virtual audio cables. The harness plays the source clip into the first cable, whose output is selected as the tool's input device, and the tool's output is directed into the second cable, whose input the harness records. From the tool's point of view it is receiving live microphone audio and producing live output, exactly as in ordinary use, and from the harness's point of view it is playing to one device and recording from another. The two devices are resolved by name under the Windows WASAPI host API. Nothing about this arrangement depends on the tool being open or scriptable, which is what allows the same harness to drive an open-source RVC host and a closed commercial tool without modification.

```
load source clip; if mono, duplicate to stereo
resample to the playback device's native rate
append flush_s seconds of silence
start recording on the capture device for (duration + tail_s) seconds
play the prepared audio on the playback device
wait until the recording (including the tail) finishes
trim leading/trailing silence below a threshold
write the trimmed audio as 16-bit PCM, named by the source id
```

Figure 5.2: The per-clip capture procedure. The recording is opened before playback begins and runs for the clip duration plus a fixed tail, so the converter's latency is caught, and the output is then trimmed and saved under the source id.

5.2.3 Playing and Recording One Clip

For each source clip the harness performs the sequence in Figure 5.2. The source is loaded and, if mono, duplicated to two channels so that no output channel can be missed. It is resampled to the playback device's native rate by polyphase resampling, since the device, not the file, dictates the rate at which audio is clocked out. A short flush of silence is appended so that the genuine tail of the conversion is not clipped when the output stream closes. The harness then opens a recording on the capture device for the clip's duration plus a fixed tail, to catch the converter's latency, plays the prepared audio on the playback device, and blocks until the recording including the tail has completed. Capture is at 48 kHz in stereo, both to match typical device rates and to leave resampling to the analysis stage rather than the recording stage.

5.2.4 Silence Trimming

The recorded buffer begins before the tool produces sound and ends after it finishes, so the harness trims silence from both ends before saving. It computes the root-mean-square energy of the recording in short frames, marks every frame whose energy exceeds a fixed threshold as active, and keeps the span from the first to the last active frame, with a small symmetric padding so that low-energy onsets and decays are not clipped. A frame length of 20 ms, a threshold of -50 dB, and 50 ms of padding were used. If no frame exceeds the threshold the recording is kept untrimmed rather than discarded, so that a silent or failed capture is preserved for inspection rather than silently dropped. Outputs are written as 16-bit PCM and named by the source utterance id alone, and the flat naming is what lets the scoring stage rejoin each output to its source regardless of the folder structure a tool imposed.

Captures are written one tool and one configuration at a time, with a short gap of silence between clips so the tool's internal buffers can settle, and an output that already exists is skipped, so that an interrupted capture run can be resumed without repeating work or overwriting good data.

5.3 Building the Scoring Manifest

Before any metric runs, the scoring pipeline builds a manifest that pairs each converted file with the information needed to score it. The corpus is the source of truth. Every source clip and every transcript is keyed by the utterance id embedded in its filename, where the two corpora have recognisably different id shapes. The LibriSpeech clips keep their original dataset ids, each a speaker, chapter, and utterance triple such as `1462-170138-0006`. The GigaSpeech clips, which arrived from the corpus API without any usable identifier of their own, were simply numbered in sequence as `gigaspeech_00` and so on. A single regular expression recognises both, and the same expression extracts the id from each converted filename, so a converted file is joined to its source and transcript by id no matter where the tool placed it.

Each converted file becomes a sample record carrying its tool, its configuration label, the source

corpus and gender recovered from the matched source clip, and the paths to the source audio and ground-truth transcript when they are found. Two derived fields drive the later aggregation. The *program* is the top-level folder, that is, the tool. The *variant* is the folder path beneath it with any per-speaker subfolder removed, so that outputs which a tool happened to nest under a speaker id still group together. The *group*, the program and variant combined, is the unit over which the summary tables average. A converted file whose name contains no recognisable id is reported and skipped rather than scored against the wrong source, which makes a misnamed capture a visible warning instead of a silent error.

5.4 The Scoring Pipeline

The pipeline takes the manifest and the configured metric set and produces one score record per converted file. Its structure is dictated by two facts: loading a metric model is expensive and should happen once, and the analysis-ready version of each audio file is shared by several metrics and should be computed once.

For each metric named in the configuration the pipeline constructs a *scorer* object once, at startup, so that the underlying model, the ASR system, the MOS predictors, the speaker embedder, is loaded a single time and reused across every file. It then iterates over the samples. For each sample it prepares the shared analysis audio once, a 16 kHz mono signal obtained by resampling the captured output and reducing it to a single channel by averaging, and hands that same prepared signal to every scorer. The normalised audio is cached on disk, so the resampling is paid once and is reused across metrics within a run and across reruns. Each scorer returns a small dictionary of named values that is merged into the sample's record.

Robustness is handled per scorer rather than per file. A scorer that raises an exception on a particular file has its error recorded in that file's record and the pipeline moves on, so that one tool's pathological output, a corrupt capture, say, or an empty conversion, does not abort an otherwise complete run. The speaker-consistency metric of Equation 4.3 is the one quantity that cannot be computed from a single file, because it is a property of a whole group. It is therefore computed in a second pass once every output has been embedded, by collecting the per-file embeddings by group and averaging their pairwise cosine similarities, after which the raw embeddings are discarded so that they do not reach the output tables.

5.5 Metric Implementations

5.5.1 Intelligibility

Word and character error rate are computed by transcribing each converted file and comparing the result with the ground-truth transcript. Transcription uses a Whisper large-v3 model [32] through the `faster-whisper` runtime, decoded with a beam width of five and a fixed English language setting. The runtime selects half-precision on the GPU and falls back to integer precision on the CPU, and if GPU initialisation fails it retreats to the CPU automatically, so a machine without a working GPU still produces scores, only more slowly. Both the hypothesis and the reference transcripts are passed through a standard English text normaliser before the error rates of Equation 4.1 are computed, so that casing, punctuation, and number formatting do not inflate the result. When a converted file has no ground-truth transcript its error rates are left blank rather than guessed, and the raw ASR text is retained in the per-file output so that surprising error rates can be inspected by eye.

5.5.2 Naturalness

The two naturalness predictors are each wrapped in a scorer that loads its model once. UTMOSv2 [2] is run through its reference implementation and returns a single predicted mean opinion score per file. DNSMOS [36] is run through its standard implementation and returns the four P.835-aligned scores, speech distortion, background noise, overall, and the P.808 score, on the shared 16 kHz signal. Both are higher-is-better and require no reference, as required by the design.

5.5.3 Speaker Identity

The identity metrics are built on an ECAPA-TDNN speaker embedder [11] from the SpeechBrain toolkit [33]. Embeddings are cached by file path, so the source clips and any target references, which recur across many comparisons, are embedded once. For each converted file the scorer computes the source-similarity proxy against the embedding of the original source clip, and, when a target reference is configured, the canonical target-similarity metric against the embedding of the target recording. The target-consistency proxy is computed in the pipeline's second pass as described above.

Target references, where they exist, are supplied as a folder of clips, and the scorer resolves the right target for each output from most specific to least: it prefers a reference tied to the exact tool-and-direction group, then one tied to the tool, then a shared default, and within whichever it finds it prefers a clip whose filename advertises the destination gender of the conversion. A tool with no target reference at any of those levels is simply left without a target-similarity score rather than scored against some other tool's voice. This resolution is the mechanism by which Seed-VC, the one tool whose target is an actual reference clip, receives a canonical target-similarity score while the others do not, exactly as the methodology requires (Section 4.5.3).

5.6 Reporting

The scoring writes three tables. The per-file table records every metric for every converted file, together with the retained ASR text and any per-scorer errors, and is the artefact from which all later analysis is done. The summary table averages each metric over its group, that is, per tool and configuration, and records the number of files behind each mean. A third table repeats the summary broken down by source corpus, which is what surfaces the LibriSpeech versus GigaSpeech contrast reported in Chapter 6. A ranked summary is also printed to the console at the end of a run, with each metric annotated by its direction, higher-is-better or lower-is-better, so that the headline picture is legible without opening a file. Because the per-file scores are written verbatim, the summary tables are pure aggregations of them, and any further statistic, a confidence interval or a paired test between two tools, can be computed from the per-file table after the fact without re-running the metrics.

5.7 Measuring Resource Usage

The resource figures introduced in Section 4.6.4 are read off the operating system's own monitoring tools rather than measured by the benchmark code, because that is exactly how a prospective user would check the same thing. On Windows the Task Manager reports, for each running process, its processor share, its memory as a working set in megabytes, and its graphics-processor utilisation. With the rest of the machine otherwise idle, each tool is set converting a continuous stream of speech as it would be in normal use, and once its consumption has settled the average steady-state values for that tool are read off, summed across processes where a tool uses several. The numbers are reported in Chapter 6 as round, indicative averages, which is enough to separate the tools that run on the processor alone from those that also commit a graphics processor and a large memory footprint.

5.8 Reproducibility

Two properties make the scoring reproducible on the same audio. First, every metric model is pinned: the ASR model, the two MOS predictors, and the speaker embedder are fixed checkpoints, so re-scoring the same converted files reproduces the same numbers. Second, all settings live in the single configuration file, so a run is described by that file together with the pinned model identifiers rather than by command-line state. The environment is Python with the dependencies listed in the project's requirements, the speaker embedder and ASR runtime use the GPU when one is available and fall back to the CPU otherwise, so the same code produces the same scores on a CPU-only machine, only more slowly.

The capture half is reproducible in a weaker sense, and this is an honest limit rather than an oversight. Re-running the scoring on fixed audio is deterministic, but re-running the capture depends on the live tool, its configuration, and the audio devices, none of which is fully captured by the configuration file. The mitigation is to treat the captured audio itself as the reproducible artefact: once a tool's outputs are recorded they are fixed, and every number in Chapter 6 is derived from them by the deterministic scoring pipeline. The configuration used to produce each tool's captures is recorded in the project repository so that a determined reader could reconstruct the capture conditions, but the primary guarantee the system offers is that the scoring of the recorded audio is exactly repeatable.

Chapter 6

Results

This chapter reports the measurements produced by the benchmark of Chapters 4 and 5. It is deliberately descriptive: the numbers and the patterns in them are presented here, and their interpretation, the architectural and methodological reading that answers the research questions, is deferred to Chapter 7. Every quality figure is a mean over a set of converted utterances that share a tool and a conversion direction (for instance, all female-to-male conversions by one tool), where each utterance is scored individually before the scores are averaged. No confidence intervals are attached, for the reason given in Section 4.7, so small differences should be read as descriptive rather than as established orderings.

Six tools are evaluated in seven configurations, since w-okada is run with both an RVC model and a Beatrice model, and each configuration converts the same 138 utterances. Section 6.1 gives the headline comparison, and Sections 6.2 to 6.4 take the three quality dimensions in turn. Section 6.5 then reports resource usage, including a matched comparison of the same model run by two different hosts, and Section 6.6 collects the findings.

6.1 Headline Comparison

Table 6.1 reports each configuration's mean over all 138 of its converted utterances on the headline metrics. The configurations divide into roughly three tiers on intelligibility, a narrow naturalness lead for the same top tier, and a more scattered picture on the identity proxies that Section 6.4 shows to be the most treacherous to read. See Table 4.1 for each tool's model, openness, and other properties.

6.2 Intelligibility

Three tiers are visible in the word error rate. Seed-VC, Vonovox, and DubbingAI preserve content best, clustered between 2.2 % and 2.6 %. w-okada with an RVC model and Voicemod form a middle tier near 4.6 %. w-okada with a Beatrice model (6.5 %) and Voice.ai (7.9 %) trail. Character error rate orders the configurations almost identically, so the gaps reflect a consistent difference in how cleanly content survives conversion rather than an artefact of single-word substitutions. In absolute terms the top two tiers remain easy to follow; only Voice.ai, and to a lesser extent Beatrice, reach error rates at which listeners would notice dropped or altered words.

Cutting the data by source corpus, in Table 6.2, is more revealing than the overall means. Most configurations transcribe far more cleanly on the clean read speech of LibriSpeech than on the spontaneous, acoustically varied speech of GigaSpeech, but the size of that gap varies sharply, and one configuration behaves unlike all the others. Vonovox and Seed-VC degrade gently, roughly doubling an already-low error rate. w-okada / RVC and Voicemod degrade steeply, their GigaSpeech error

Table 6.1: Headline comparison across all seven configurations, each averaged over 138 converted utterances. Arrows give the better direction, and bold marks the best value in each quality column. The SECS columns are left unbolded because, as Section 6.4 argues, a lower source similarity or a higher consistency does not by itself mean a better tool.

Configuration	WER↓	CER↓	UTMOS↑	DNSMOS OVRL↑	SECS src↓	SECS cons↑	SECS tgt↑
Seed-VC	**2.2%**	**1.1%**	**3.00**	**3.36**	0.29	0.69	0.53
Vonovox	2.6%	1.3%	**3.00**	**3.36**	0.17	0.74	–
DubbingAI	2.6%	1.2%	2.99	3.30	0.21	0.73	–
w-okada / RVC	4.6%	2.5%	2.82	**3.36**	0.17	0.76	–
Voicemod	4.6%	2.5%	2.82	3.23	0.16	0.77	–
w-okada / Beatrice	6.5%	3.9%	2.89	3.13	0.11	0.77	–
Voice.ai	7.9%	4.5%	2.26	2.89	0.12	0.74	–

Table 6.2: Intelligibility and naturalness by source corpus: each configuration's mean over the 68 LibriSpeech and 70 GigaSpeech conversions.

	WER↓		UTMOS↑	
Configuration	LibriSpeech	GigaSpeech	LibriSpeech	GigaSpeech
Seed-VC	1.7%	2.8%	3.08	2.93
Vonovox	1.8%	3.4%	3.08	2.93
DubbingAI	2.2%	3.0%	3.06	2.91
w-okada / RVC	3.0%	6.1%	2.86	2.78
Voicemod	2.7%	6.5%	2.88	2.76
w-okada / Beatrice	6.5%	6.5%	2.89	2.88
Voice.ai	6.5%	9.3%	2.37	2.15

rates exceeding 6 %. w-okada / Beatrice is the exception: its error rate is essentially unchanged between the two corpora (6.5 % on both), so it is the most robust to speaking style but at a consistently high level rather than a low one. Voice.ai is both the weakest and the most style-sensitive, climbing above 9 % on GigaSpeech.

6.3 Naturalness

On predicted naturalness the top of the field is tightly packed. UTMOSv2 places Seed-VC and Vonovox level at 3.00 and DubbingAI a hair behind at 2.99. w-okada / Beatrice follows at 2.89, then w-okada / RVC and Voicemod at 2.82. Voice.ai is a clear outlier at 2.26, well below the rest. DNSMOS broadly agrees: its overall score is essentially tied near 3.36 across Seed-VC, Vonovox, and w-okada / RVC, a little lower for DubbingAI and Voicemod, lower again for Beatrice, and lowest for Voice.ai at 2.89, whose deficit is concentrated in the speech-distortion and background sub-scores. The two predictors agree on the leaders and on Voice.ai's last place, which raises confidence in the naturalness ranking even where the gaps near the top are small.

Splitting naturalness by conversion direction, in Table 6.3, exposes a strong but not universal pattern. For most configurations, converting *into* a male target yields markedly higher predicted naturalness than converting into a female target: the two to-male directions score several tenths of a point above the two to-female directions for Seed-VC, Vonovox, both w-okada configurations, and Voicemod. DubbingAI is the clear exception, scoring higher into female targets than into male ones, and Voice.ai shows only a weak version of the majority pattern. The effect is therefore not a universal law of the conversion direction: it is most naturally read as a property of the particular target voices and presets each tool provides, amplified by whatever the naturalness

Table 6.3: UTMOSv2 by conversion direction (source gender to target gender), higher is better, with each configuration's mean over to-female and to-male directions. The to-male advantage holds for most configurations but reverses for DubbingAI.

Configuration	F_to_F↑	F_to_M↑	M_to_F↑	M_to_M↑	to-F↑	to-M↑
Seed-VC	2.86	3.13	2.82	3.19	2.84	3.16
Vonovox	2.81	3.24	2.68	3.28	2.74	3.26
DubbingAI	3.07	2.80	3.18	2.89	3.13	2.85
w-okada / RVC	2.59	3.16	2.44	3.10	2.52	3.13
Voicemod	2.67	2.91	2.70	3.00	2.69	2.96
w-okada / Beatrice	2.55	3.04	2.87	3.06	2.71	3.05
Voice.ai	2.16	2.41	2.21	2.25	2.19	2.33

Table 6.4: Speaker identity. SECS-src is mean cosine to the source (lower is better), SECS-cons is output consistency (higher is better), and SECS-tgt is the canonical target similarity (higher is better), computable only for Seed-VC. Means over all 138 conversions. See Table 4.1 for each tool's model, openness, and other properties.

Configuration	SECS-src↓	SECS-cons↑	SECS-tgt↑
Seed-VC	0.29	0.69	0.53
Vonovox	0.17	0.74	–
DubbingAI	0.21	0.73	–
w-okada / RVC	0.17	0.76	–
Voicemod	0.16	0.77	–
w-okada / Beatrice	0.11	0.77	–
Voice.ai	0.12	0.74	–

predictor rewards, rather than of the source or target gender as such. The source gender, once the target is fixed, has little systematic effect.

6.4 Speaker Identity

The identity results must be read through the measures of Section 4.5.3, since only Seed-VC carries the canonical target-similarity score. All three SECS values are cosine similarities with a maximum of 1.0, which is reached only when two voices are essentially identical in the embedding space, whereas the UTMOSv2 and DNSMOS scores reported above run on the usual 1 to 5 mean-opinion-score scale. The three columns also do not share a direction of betterness: source similarity (SECS-src) is lower-is-better, because a successful conversion moves identity *away* from the source, whereas consistency (SECS-cons) and target similarity (SECS-tgt) are higher-is-better, since both reward landing on, or staying close to, the intended target. The arrows in Table 6.4 record this. The table reports the two reference-free proxies for every configuration, with Seed-VC's canonical score alongside, and it is where the proxies prove most misleading.

Taken at face value, the proxies tell a story that the quality metrics flatly contradict. By source similarity, the configuration that moves identity furthest off the source is w-okada / Beatrice (0.11), closely followed by Voice.ai (0.12); by output consistency, Voicemod and Beatrice lead at 0.77. Read alone, these two proxies would rank Voice.ai as a strong converter, one that both moves well away from the source and lands on a stable voice, when Voice.ai is in fact the weakest tool in the study on both intelligibility and naturalness. Conversely Seed-VC, a top-tier tool on every quality measure, posts the *highest* source similarity (0.29, it moves least) and the *lowest* consistency (0.69), so the proxies would rank it near the bottom on identity. The one canonical anchor available, Seed-VC's target similarity of 0.53, confirms that Seed-VC does reach its intended target, which the proxies

Table 6.5: Steady-state resource usage during conversion, read from the operating system as described in Section 5.7. Figures are indicative, not precise.

Configuration	CPU	Memory	GPU
Seed-VC	16%	2000 MB	20%
Vonovox	7%	2200 MB	10%
DubbingAI	1%	220 MB	0%
w-okada / RVC	1%	160 MB	0%
w-okada / Beatrice	1.5%	120 MB	0%
Voicemod	1%	280 MB	0%
Voice.ai	30%	2200 MB	1%

entirely fail to reflect.

Two narrower patterns are consistent with the proxies measuring genuine identity movement rather than noise. First, the fixed-target tools, which always render the same trained or preset voice, generally move further from the source and reach higher consistency than the zero-shot Seed-VC, which reconstructs a target afresh from a reference clip for each input. Second, for every configuration, same-gender conversions (F_to_F and M_to_M) retain more source similarity than cross-gender conversions, as expected when the target identity is more distant from the source. Both patterns are real, but neither rescues the proxies as a stand-alone ranking: the Voice.ai and Seed-VC cases show that source similarity and consistency can be made to look excellent by a degraded or collapsed output and poor by a faithful one. What this means for evaluating such tools is taken up in Chapter 7.

6.5 Resource Usage

Table 6.5 reports the steady-state processor, memory, and graphics-processor usage observed for each configuration, by the procedure of Section 5.7. The separation between configurations is large. Four run almost for free on the processor alone, at or below 1.5 % processor share and under 300 MB of memory with no graphics-processor use: both w-okada configurations, DubbingAI, and Voicemod. Three are heavyweight. Seed-VC and Vonovox each commit a graphics processor and around 2 GB of memory, and Voice.ai is the most demanding of all on the processor, at 30 %, while also holding around 2 GB.

The striking point is how little quality tracks cost. DubbingAI reaches the top tier on intelligibility and naturalness at the same negligible footprint as the lightest tools, while Voice.ai consumes the most processor time of any tool for the worst quality in the study. Seed-VC, a joint quality leader, reaches its quality only with a graphics processor and a large memory footprint.

The clearest illustration that resources buy quality comes from the one pair that holds the model fixed. Vonovox and w-okada both load the same RVC model files, but they run them very differently. w-okada runs the model on the processor alone, cheaply, whereas Vonovox is built to run on the graphics processor and uses it, together with around 2 GB of memory. The extra resources show up in the output: on the identical model, Vonovox roughly halves w-okada's word error rate (2.6 % against 4.6 %, widening to 3.4 % against 6.1 % on GigaSpeech) and leads naturalness by almost two tenths of a UTMOSv2 point, while the two stay matched on the identity proxies and on DNSMOS, as a shared model would suggest. Running the same conversion on the graphics processor rather than the processor, in other words, is worth a clear gain in quality here.

6.6 Summary of Findings

The measurements support five descriptive findings, carried into the discussion unchanged.

1. **Three tiers on intelligibility.** Seed-VC, Vonovox, and DubbingAI preserve content best (near 2 % word error rate); w-okada / RVC and Voicemod form a middle tier near 4.6 %; w-okada / Beatrice and Voice.ai trail at 6.5 % and 7.9 %. Character error rate agrees.

2. **A narrow naturalness lead** for the same top three, corroborated by both MOS predictors, with Voice.ai a clear outlier at the bottom.

3. **A direction effect that is strong but not universal**: converting into a male target scores higher naturalness for most configurations, but DubbingAI reverses it, which argues the effect belongs to the particular target voices rather than to gender as such.

4. **The identity proxies can mislead.** Voice.ai scores well on both reference-free proxies yet is the weakest tool overall, while the top-tier Seed-VC scores worst on them and is the only tool the canonical metric confirms reaches its target. The proxies are interpretable only together with the quality metrics.

5. **Quality does not track cost.** DubbingAI reaches the top tier at a negligible footprint, Voice.ai consumes the most processor time for the worst quality, and the heavyweight tools buy their quality with a graphics processor and large memory. On the one pair that holds the model fixed, the graphics-only Vonovox clearly beats the processor-only w-okada.

The commercial tools do not, on these quality measures, lead the field: one of them (DubbingAI) joins the best open-source tools while another (Voice.ai) is last. What these findings mean for the two research questions, and how far the reference-free identity proxies can be trusted, is the subject of Chapter 7.

Chapter 7

Discussion

The results of Chapter 6 were presented descriptively. This chapter interprets them. It works through the two research questions in turn (Sections 7.1 and 7.2), turns to what the identity proxies can and cannot tell us (Section 7.3), draws together the remaining cross-cutting observations (Section 7.4), revisits the limitations in the light of what was found (Section 7.5), and states the implications for the people who choose these tools and the people who study them (Section 7.6). Throughout, the caution of Section 4.7 applies: the benchmark reports means without confidence intervals, so the argument rests on the larger and repeated differences rather than on narrow gaps between adjacent tools.

7.1 RQ1: How the Tools Compare

The first research question asked how the deployed tools compare on objective quality under one protocol. The clearest answer is that they separate into tiers that are consistent across the quality dimensions, and that the separation does not follow the line between open-source and commercial software.

On intelligibility three configurations stand out, Seed-VC, Vonovox, and DubbingAI, all near two percent word error rate, and the same three lead naturalness. A middle group, w-okada with an RVC model and Voicemod, roughly doubles the error rate, and two configurations trail, w-okada with a Beatrice model and Voice.ai. That this ordering holds across word error rate, character error rate, and both MOS predictors is what gives it weight: a tool that led on one measure by sacrificing another would betray itself in the portfolio, and none of the leaders does. The practical message is that the best deployed tools are genuinely good. Even the middle tier is easily intelligible, and only Voice.ai, and to a lesser degree the lightweight Beatrice model, reach error rates a listener would notice.

The result that most directly answers RQ1 in the negative is that the commercial tools are not, as a group, ahead. One of them, DubbingAI, sits with the best open-source tools on every quality measure while using almost no resources at all; another, Voice.ai, is the weakest tool in the study and the most expensive to run. The two commercial tools thus bracket the entire field. Whatever the marketing of these tools claims, the measured picture is that a paid tool is neither a guarantee of quality nor an indicator of it. The open-source Seed-VC and Vonovox are as good as the best commercial tool measured, and far better than the worst.

7.2 RQ2: Cost Versus Quality

The second research question asked whether better quality comes at a higher cost to the user, in money or in computing resources. The benchmark answers with a clear no, with one instructive

exception.

Money first. The free open-source tools are not the poor relations. Seed-VC and Vonovox are joint quality leaders, and the free w-okada with an RVC model sits with the paid Voicemod in the middle tier. The paid commercial tools, far from leading, bracket the entire field: DubbingAI joins the best, while Voice.ai is the weakest tool measured. Whatever a subscription buys, it is not a guarantee of quality, and on this evidence the price of a tool tells a prospective user almost nothing about how good it will sound.

Resources tell the same story. The resource measurements add a dimension the quality scores cannot, and the headline is the absence of a relationship between the two. DubbingAI reaches the top tier on a negligible processor footprint with no graphics processor at all, while Voice.ai spends the most processor time of any tool for the worst quality. Seed-VC, a quality leader, does need a graphics processor and around two gigabytes of memory, so heavy tools are not always wasteful, but light tools are clearly not thereby compromised: a user on a modest machine is not shut out of the top tier, because DubbingAI and the RVC tools reach it on the processor alone.

The instructive exception is the one pair that holds the model fixed. Vonovox and w-okada load identical RVC files, yet Vonovox roughly halves w-okada's word error rate and leads on naturalness, while the two stay matched on the identity proxies and on DNSMOS. The resource figures explain it: Vonovox is built to run the model on the graphics processor and does, whereas w-okada runs the same model on the processor alone. Here, and only here, spending more resources did buy quality, but note what kind of cost it was. It was compute, not money, both tools are free, and it was a property of the host rather than of the model. This is a concrete instance of the point made in Section 2.2.3, that the deployment layer is not a neutral conduit, and a caution against attributing a deployed tool's quality to its model alone.

Cost to the user is not only money and compute, though, and the tools differ sharply in a third currency: the effort of getting a voice in the first place. The open-source tools are the most flexible and the least convenient. Seed-VC is zero-shot, so any target voice can be had from a short reference clip of a few seconds to half a minute, with no training at all. The RVC tools draw on a large community library of ready-made models and let a user train their own from a relatively small amount of target audio, but auditioning community models is tedious, since each must be downloaded, loaded, and configured before it can be heard. Beatrice, as run inside w-okada, instead arrives with a fixed set of around twenty-five built-in voices. The commercial tools invert the trade. Voice.ai is itself RVC-based, letting a user upload their own models or browse others much as the open-source tools do, while Voicemod and DubbingAI are more closed but offer a larger and far easier library, where a voice can be auditioned at the press of a button and a custom voice can still be built. These conveniences, and not raw quality, are what the commercial subscriptions buy.

7.3 What the Identity Proxies Can and Cannot Tell Us

Underlying both research questions is a methodological worry flagged in Section 4.5.3: whether the reference-free identity proxies are adequate for ranking tools when no clean target recording exists. The results answer it sharply, because two tools turn the proxies into a near-controlled test of their own validity.

Read on their own, the proxies would rank Voice.ai as a strong converter: it has almost the lowest source similarity, meaning its output moves far from the source speaker, and a respectable output consistency, meaning it lands on a stable voice. Yet Voice.ai is the weakest tool in the study on both intelligibility and naturalness. The explanation is the failure mode anticipated in Section 4.5.3: source similarity falls not only when a conversion genuinely changes the identity but also when the output is degraded, because noise and artefacts also push a speaker embedding away from the source. A low source similarity is therefore ambiguous, and in Voice.ai's case it reflects degradation as much as conversion. Output consistency is similarly blind, rewarding a tool that collapses many inputs toward a narrow timbre regardless of whether that timbre is the intended one or whether the result is even good speech.

The mirror image is Seed-VC. It posts the highest source similarity, so the proxies read it as moving identity the least, and the lowest consistency, so they read it as the least stable, which together would place it near the bottom on identity. Seed-VC is in fact a top-tier tool, and it is the one tool for which the canonical target-similarity metric can be computed, which confirms that it does reach its intended target. The proxies, in other words, would actively mis-rank the single tool whose true target behaviour can be checked.

The verdict is therefore a qualified negative. The proxies are not adequate as a stand-alone ranking of identity: taken alone they would invert the true order at both ends of the field. They retain a narrower usefulness. They detect coarse facts reliably, that every tool moves identity substantially off the source, that the fixed-target tools move further and are more consistent than the zero-shot one, and that cross-gender conversion moves identity further than same-gender conversion. And they become interpretable when read *together with* the intelligibility and naturalness metrics, because the combination distinguishes a genuine, stable conversion from a degraded or collapsed one. The practical recommendation that follows is twofold: never report the reference-free identity proxies without the quality metrics beside them, and obtain a clean target recording for the canonical metric wherever it is at all possible, as it was for Seed-VC. This is the methodological contribution of the thesis, and it generalises to anyone benchmarking deployed any-to-one tools without target-voice ground truth.

7.4 Cross-Cutting Observations

7.4.1 The Direction Effect Is About Voices, Not Genders

For most configurations, converting into a male target produced higher predicted naturalness than converting into a female target, and the effect was large relative to the differences between tools. It would be easy to over-read this as a property of gender, but the data caution against that reading. DubbingAI reverses the effect outright, scoring higher into female targets, and Voice.ai barely shows it. Because each tool uses its own target voices and presets, a direction label fixes the source gender but not the target identity, so the most defensible interpretation is that the effect belongs to the particular target voices each tool happens to provide, and to whatever the naturalness predictor rewards in them, rather than to the gender of the target as such. The honest statement is that target-voice choice has a strong effect on measured naturalness, of a size comparable to the differences between tools, which is itself a useful warning: a benchmark that fixed a single target gender could rank tools differently from one that balanced them.

7.4.2 Robustness to Speaking Style Separates the Tools

The split between LibriSpeech and GigaSpeech did more to separate the tools than the overall means did. On clean read speech the field is compressed; on spontaneous, acoustically varied speech it spreads out, and the tools that were best on clean speech were also the most robust to harder input. Beatrice is the instructive exception, almost unaffected by the change of register but at a uniformly higher error rate, which suggests its errors come from the model's content handling rather than from sensitivity to input conditions. The general point is that a benchmark on clean read speech alone would have understated the differences that matter, and that robustness to realistic input is a distinguishing property worth measuring explicitly.

7.4.3 Imitating the Target Was Not Tested

One effect worth raising, precisely because the benchmark could not test it, is that several of the tools advise the user to speak in a way closer to the chosen target, for instance by shifting their pitch toward it. This is the same idea, in another guise, as the finding that same-gender conversion moves identity less and tends to convert more cleanly: the closer the source already is to the target, the less work the conversion has to do. A static, pre-recorded corpus cannot capture this, since

the speakers in LibriSpeech and GigaSpeech were not trying to imitate anyone. One would expect every tool to benefit from such accommodation to a similar degree, so it is unlikely to change the ranking, but it does mean the quality a motivated live user hears may be somewhat better than the numbers here, which come from speakers making no such effort.

7.5 Limitations Revisited

The threats to validity catalogued in Section 4.7 bear directly on how far these interpretations can be pushed. Several deserve restating in the light of the results.

The naturalness findings rest on predicted MOS, not on human listening, and the direction effect in particular could be partly an artefact of what the predictors reward. The agreement between UTMOSv2 and DNSMOS on the tool ordering is reassuring, but neither is a substitute for a subjective test, and a listening study is the natural way to confirm the rankings. The intelligibility findings rest on a single ASR system, so a tool whose artefacts happen to trouble that system would be scored unfairly. The consistency of the corpus effect across tools makes a gross artefact unlikely, but a second ASR would settle it. And the absence of confidence intervals means the tiered reading of RQ1 is on firm ground only where the gaps are large. The gap between the top tier and Voice.ai is unmistakable, whereas the gap between, say, Voicemod and w-okada with an RVC model is small enough that it should not be over-interpreted without the paired tests recommended in Chapter 8.

Two further limitations shape the comparison. Because the target voices differ across tools, cross-tool comparisons of absolute identity are not on the same footing as the within-tool and matched-model ones, and the Vonovox-versus-w-okada comparison is trustworthy precisely because it holds the model fixed. And every tool was measured at a single configuration and with only two target voices, one female and one male, so the results characterise each tool as configured and with the voices chosen, not at its unknown best. A tool that trailed here might close the gap under different settings, or simply with a better-suited or higher-quality target voice, and the single-configuration decision trades that possibility for a tractable and reproducible comparison.

7.6 Implications

For a user choosing a tool, the benchmark offers concrete guidance. If intelligibility and naturalness are what matter and a graphics processor is available, Seed-VC and Vonovox are the strongest measured. If low resource use also matters, DubbingAI reaches the same tier on the processor alone, which makes it the most efficient strong tool in the study. The retrieval-based RVC model is a capable and extremely cheap option when run in a host that exploits it well, and the host matters as much as the model. The clearest negative recommendation is that price is no proxy for quality: one commercial tool leads and another trails the entire field.

For researchers, the benchmark shows that the deployed state of the art is competitive with the published one on quality, occupying the same architectural positions described in Chapter 3, and that the gap the literature leaves open is not in deployed quality, which this thesis now measures, but in deployed latency, which it does not. For evaluation methodology, the proxy result is a cautionary contribution of independent value: the reference-free identity proxies in common use can invert the true ranking of tools, and they must be read alongside intelligibility and naturalness, never alone. These threads are drawn together, and turned into a future-work agenda, in Chapter 8.

Chapter 8

Conclusion

This thesis set out to close a specific gap: the absence of a unified, reproducible, objective comparison of the quality of deployed real-time voice conversion tools. The peer-reviewed literature evaluates research systems carefully but incompatibly, and the tools that people actually use are evaluated by their vendors or not at all, so a practitioner or a researcher has had no neutral standard. The thesis built one for the dimension that can be measured rigorously today, output quality, and applied it to six tools in seven configurations on two contrasting speech corpora and fixed hardware.

8.1 Summary of Findings

The benchmark answers its two research questions as follows.

On **RQ1**, the tools separate into consistent quality tiers that do not follow the boundary between open-source and commercial software. Seed-VC, Vonovox, and DubbingAI lead on both intelligibility and naturalness. w-okada with an RVC model and Voicemod form a middle tier, and w-okada with a Beatrice model and Voice.ai trail. The ranking is stable across word and character error rate and across two MOS predictors, and the commercial tools bracket the field rather than leading it: one matches the best open-source tools, while another is the weakest measured.

On **RQ2**, better quality does not come at a higher cost. Price is no guide, since the paid commercial tools bracket the field while the free open-source tools include two of the three leaders. Resource cost is no guide either, since DubbingAI reaches the top tier on the processor alone whereas the most processor-hungry tool, Voice.ai, is the worst. The one place extra cost did buy quality was compute rather than money: on identical RVC models the graphics-processor-bound Vonovox clearly beat the processor-only w-okada, a difference owed to the host and not the model.

A methodological finding accompanies the two answers. The reference-free identity proxies are not adequate as a stand-alone ranking: taken alone they would rank the weakest tool, Voice.ai, as a strong converter, and the strong Seed-VC, the only tool whose target similarity can be checked directly, as a poor one. They confound genuine conversion with degradation and mode collapse, and are trustworthy only when read together with the intelligibility and naturalness metrics. Showing this concretely, on a tool whose true target behaviour can be verified, is a contribution in its own right.

Two further observations accompany these answers. The apparent advantage of converting into male targets is most plausibly a property of the particular target voices rather than of gender, since one tool reverses it. And the split between clean and spontaneous speech separated the tools more than the overall means did, with the tools that were best on clean speech also the most robust to harder input.

8.2 Contributions

The thesis contributes a reproducible black-box benchmark for deployed real-time voice conversion tools, combining a capture harness with a scoring pipeline over an intelligibility, naturalness, and speaker-identity portfolio; the first common-protocol quality comparison of these tools; an analysis of quality against cost in money and resources, including a matched comparison of the same model run by two hosts; a critical assessment of reference-free identity proxies that shows where they mislead; and a structured review and architectural taxonomy that places the deployed tools against the peer-reviewed literature. The complete source code for the capture harness and scoring pipeline, together with the configuration used for each tool, is available at https://github.com/eivindnesje/Benchmarks, and a selection of the converted audio behind the results of Chapter 6 can be listened to at https://eivindnesje.github.io/Benchmark-website/.

8.3 Future Work

A controlled latency benchmark is the foremost extension. The tools studied here are real-time systems whose users care most about end-to-end delay, and this thesis deliberately measured quality instead. The groundwork is in place: end-to-end latency separates cleanly into parts that belong to the model (its look-ahead) and parts that belong to the deployment environment (buffering, queueing, and transport), and the black-box capture path of Chapter 5 could be extended with timing instrumentation to measure it directly. Doing so under a stated protocol, across buffer sizes and on more than one machine, would complete the picture this thesis begins, and would let the deployed tools be compared to the peer-reviewed systems on the axis the literature reports but the tools do not.

A subjective listening study would confirm the naturalness and similarity rankings that the benchmark obtains from predicted MOS, and would test in particular whether the direction effect survives human judgement. A second ASR system would cross-check the intelligibility scores and rule out artefacts specific to one recogniser. Bootstrap confidence intervals and paired significance tests over the per-utterance scores, which the pipeline already produces, would establish which of the smaller differences between tools are reliable, and are an inexpensive and immediate improvement. A canonical target-similarity measurement wherever a clean target recording can be obtained would reduce the reliance on the reference-free proxies that this thesis found wanting.

Finally, the study could be broadened: to more tools, more configurations, and more than the two target voices run per tool here, including the buffer-size sweeps deferred from the quality benchmark; to other languages and to noisier, far-field, and more spontaneous speech; and to a larger corpus, which would sharpen the statistical power that the current corpus size limits.

8.4 Closing Remarks

Real-time voice conversion has become consumer software, and its quality has until now been a matter of vendor assertion. This thesis shows that the assertions can be replaced with measurement: that the best deployed tools are genuinely good, that price and resource cost predict quality poorly, that the host can matter as much as the model, and that the convenient reference-free proxies for speaker identity must be read with care. The benchmark is reproducible and the captured audio is fixed, so the comparison can be extended as new tools appear and, most importantly, completed along the latency axis that remains open.

Bibliography

[1] Matthew Baas, Benjamin van Niekerk and Herman Kamper. 'Voice Conversion With Just Nearest Neighbors'. In: *Proc. Interspeech*. 2023, pp. 2053–2057. DOI: 10.21437/Interspeech.2023-419.

[2] Kaito Baba et al. 'The T05 System for the VoiceMOS Challenge 2024: Transfer Learning from Deep Image Classifier to Naturalness MOS Prediction of High-Quality Synthetic Speech'. In: *Proc. IEEE Spoken Language Technology Workshop (SLT)*. 2024. arXiv: 2409.09305.

[3] Alexei Baevski et al. 'wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations'. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 33. 2020.

[4] Anders R. Bargum, Stefania Serafin and Cumhur Erkut. 'Reimagining Speech: A Scoping Review of Deep Learning-Based Methods for Non-Parallel Voice Conversion'. In: *Frontiers in Signal Processing* 4 (2024), p. 1339159. DOI: 10.3389/frsip.2024.1339159.

[5] Edresson Casanova et al. 'YourTTS: Towards Zero-Shot Multi-Speaker TTS and Zero-Shot Voice Conversion for Everyone'. In: *Proceedings of the 39th International Conference on Machine Learning (ICML)*. 2022, pp. 2709–2720.

[6] Guoguo Chen et al. 'GigaSpeech: An Evolving, Multi-Domain ASR Corpus with 10,000 Hours of Transcribed Audio'. In: *Proc. Interspeech*. 2021, pp. 3670–3674. DOI: 10.21437/Interspeech.2021-1965.

[7] Sanyuan Chen et al. 'WavLM: Large-Scale Self-Supervised Pre-Training for Full Stack Speech Processing'. In: *IEEE Journal of Selected Topics in Signal Processing* 16.6 (2022), pp. 1505–1518. DOI: 10.1109/JSTSP.2022.3188113.

[8] Michael Chinen et al. 'ViSQOL v3: An Open Source Production Ready Objective Speech and Audio Metric'. In: *Proc. 12th International Conference on Quality of Multimedia Experience (QoMEX)*. 2020. arXiv: 2004.09584.

[9] Erica Cooper et al. 'The VoiceMOS Challenge 2023: Zero-Shot Subjective Speech Quality Prediction for Multiple Domains'. In: *Proc. IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. 2023.

[10] Alexandre Défossez et al. 'High Fidelity Neural Audio Compression'. In: *Transactions on Machine Learning Research* (2023). arXiv: 2210.13438.

[11] Brecht Desplanques, Jenthe Thienpondt and Kris Demuynck. 'ECAPA-TDNN: Emphasized Channel Attention, Propagation and Aggregation in TDNN Based Speaker Verification'. In: *Proc. Interspeech*. 2020, pp. 3830–3834. DOI: 10.21437/Interspeech.2020-2650.

[12] Dubbing AI. *Dubbing AI: Real-Time Voice Changer*. TODO: confirm product page and version tested. 2024. URL: https://dubbingai.io/ (visited on 22nd June 2026).

[13] Echo. *Echo: Real-Time Voice Changer*. Native (Rust) RVC client using ONNX. TODO: confirm vendor and version. 2025. URL: https://voicechanger.live/ (visited on 30th June 2026).

[14] Wei-Ning Hsu et al. 'HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units'. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), pp. 3451–3460. DOI: 10.1109/TASLP.2021.3122291.

[15] IAHispano. *Applio: Voice Conversion Tool (RVC)*. RVC-based; includes a real-time conversion mode. 2023. URL: https://github.com/IAHispano/Applio (visited on 30th June 2026).

[16] Hirokazu Kameoka et al. 'StarGAN-VC: Non-Parallel Many-to-Many Voice Conversion Using Star Generative Adversarial Networks'. In: *Proc. IEEE Spoken Language Technology Workshop (SLT)*. 2018, pp. 266–273. DOI: 10.1109/SLT.2018.8639535.

[17] Takuhiro Kaneko and Hirokazu Kameoka. 'CycleGAN-VC: Non-Parallel Voice Conversion Using Cycle-Consistent Adversarial Networks'. In: *Proc. 26th European Signal Processing Conference (EUSIPCO)*. 2018. arXiv: 1711.11293.

[18] Jaehyeon Kim, Jungil Kong and Juhee Son. 'Conditional Variational Autoencoder with Adversarial Learning for End-to-End Text-to-Speech'. In: *Proceedings of the 38th International Conference on Machine Learning (ICML)*. 2021, pp. 5530–5540.

[19] 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)*. Vol. 33. 2020.

[20] Robert F. Kubichek. 'Mel-Cepstral Distance Measure for Objective Speech Quality Assessment'. In: *Proc. IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*. Vol. 1. 1993, pp. 125–128. DOI: 10.1109/PACRIM.1993.407206.

[21] Songting Liu. *Zero-Shot Voice Conversion with Diffusion Transformers*. 2024. arXiv: 2411.09943 [eess.AS].

[22] Yisi Liu et al. *RT-VC: Real-Time Zero-Shot Voice Conversion with Speech Articulatory Coding*. 2025. arXiv: 2506.10289 [eess.AS].

[23] Chen-Chou Lo et al. 'MOSNet: Deep Learning-Based Objective Assessment for Voice Conversion'. In: *Proc. Interspeech*. 2019, pp. 1541–1545. DOI: 10.21437/Interspeech.2019-2003.

[24] Masanori Morise, Fumiya Yokomori and Kenji Ozawa. 'WORLD: A Vocoder-Based High-Quality Speech Synthesis System for Real-Time Applications'. In: *IEICE Transactions on Information and Systems* E99-D.7 (2016), pp. 1877–1884. DOI: 10.1587/transinf.2015EDP7457.

[25] Benjamin van Niekerk et al. 'A Comparison of Discrete and Soft Speech Units for Improved Voice Conversion'. In: *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2022, pp. 6562–6566. DOI: 10.1109/ICASSP43922.2022.9746484.

[26] Aäron van den Oord et al. *WaveNet: A Generative Model for Raw Audio*. 2016. arXiv: 1609.03499 [cs.SD].

[27] Vassil Panayotov et al. 'LibriSpeech: An ASR Corpus Based on Public Domain Audio Books'. In: *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2015, pp. 5206–5210. DOI: 10.1109/ICASSP.2015.7178964.

[28] Plachta. *Seed-VC: Zero-Shot Voice Conversion*. 2024. URL: https://github.com/Plachtaa/seed-vc (visited on 21st June 2026).

[29] Project Beatrice. *Beatrice: Real-Time Voice Conversion*. TODO: confirm vendor, URL, and exact model version used. 2024. URL: https://prj-beatrice.com/ (visited on 22nd June 2026).

[30] Kaizhi Qian et al. 'AutoVC: Zero-Shot Voice Style Transfer with Only Autoencoder Loss'. In: *Proceedings of the 36th International Conference on Machine Learning (ICML)*. 2019, pp. 5210–5219.

[31] Kaizhi Qian et al. 'ContentVec: An Improved Self-Supervised Speech Representation by Disentangling Speakers'. In: *Proceedings of the 39th International Conference on Machine Learning (ICML)*. Vol. 162. Proceedings of Machine Learning Research. PMLR, 2022, pp. 18003–18017.

[32] Alec Radford et al. 'Robust Speech Recognition via Large-Scale Weak Supervision'. In: *Proceedings of the 40th International Conference on Machine Learning (ICML)*. 2023, pp. 28492–28518.

[33] Mirco Ravanelli et al. *SpeechBrain: A General-Purpose Speech Toolkit*. 2021. arXiv: 2106.04624 [eess.AS].

[34] *Recommendation P.835: Subjective Test Methodology for Evaluating Speech Communication Systems that Include Noise Suppression Algorithm*. Geneva: International Telecommunication Union. ITU-T. 2003.

[35] *Recommendation P.863: Perceptual Objective Listening Quality Prediction*. Geneva: International Telecommunication Union. ITU-T. 2018.

[36] Chandan K. A. Reddy, Vishak Gopal and Ross Cutler. 'DNSMOS: A Non-Intrusive Perceptual Objective Speech Quality Metric to Evaluate Noise Suppressors'. In: *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2021, pp. 6493–6497. DOI: 10.1109/ICASSP39728.2021.9414878.

[37] Antony W. Rix et al. 'Perceptual Evaluation of Speech Quality (PESQ) – A New Method for Speech Quality Assessment of Telephone Networks and Codecs'. In: *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Vol. 2. 2001, pp. 749–752. DOI: 10.1109/ICASSP.2001.941023.

[38] RVC-Project Contributors. *Retrieval-based Voice Conversion WebUI (RVC)*. 2023. URL: https://github.com/RVC-Project/Retrieval-based-Voice-Conversion-WebUI (visited on 21st June 2026).

[39] Konstantine Sadov. *Low-Latency Real-Time Voice Conversion on CPU*. 2023. arXiv: 2311.00873 [eess.AS].

[40] Takaaki Saeki et al. 'UTMOS: UTokyo-SaruLab System for VoiceMOS Challenge 2022'. In: *Proc. Interspeech*. 2022, pp. 4521–4525. DOI: 10.21437/Interspeech.2022-439.

[41] Hiroaki Sakoe and Seibi Chiba. 'Dynamic Programming Algorithm Optimization for Spoken Word Recognition'. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 26.1 (1978), pp. 43–49. DOI: 10.1109/TASSP.1978.1163055.

[42] Berrak Sisman et al. 'An Overview of Voice Conversion and Its Challenges: From Statistical Modeling to Deep Learning'. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), pp. 132–157. DOI: 10.1109/TASLP.2020.3038524.

[43] Lifa Sun et al. 'Phonetic Posteriorgrams for Many-to-One Voice Conversion Without Parallel Data Training'. In: *Proc. IEEE International Conference on Multimedia and Expo (ICME)*. 2016, pp. 1–6. DOI: 10.1109/ICME.2016.7552917.

[44] Tomoki Toda et al. 'The Voice Conversion Challenge 2016'. In: *Proc. Interspeech*. 2016, pp. 1632–1636. DOI: 10.21437/Interspeech.2016-1066.

[45] Voice.ai. *Voice.ai: Real-Time AI Voice Changer*. 2024. URL: https://voice.ai/ (visited on 21st June 2026).

[46] Voicemod. *Voicemod: Real-Time Voice Changer*. 2024. URL: https://www.voicemod.net/ (visited on 21st June 2026).

[47] Vonovox. *Vonovox Voice Changer*. TODO: confirm vendor and URL. 2024. (Visited on 21st June 2026).

[48] w-okada. *w-okada voice-changer*. 2023. URL: https://github.com/w-okada/voice-changer (visited on 21st June 2026).

[49] Yang Yang et al. 'StreamVC: Real-Time Low-Latency Voice Conversion'. In: *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2024, pp. 11016–11020. DOI: 10.1109/ICASSP48485.2024.10446863.

[50] Neil Zeghidour et al. 'SoundStream: An End-to-End Neural Audio Codec'. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 30 (2021), pp. 495–507. DOI: 10.1109/TASLP.2021.3129994.

[51] Yi Zhao et al. 'Voice Conversion Challenge 2020: Intra-Lingual Semi-Parallel and Cross-Lingual Voice Conversion'. In: *Proc. Joint Workshop for the Blizzard Challenge and Voice Conversion Challenge 2020*. 2020, pp. 80–98. arXiv: 2008.12527.

Appendix A

Note on Sustainability

This thesis is a methodology and evaluation study rather than a deployed system, so its relevance to sustainability, in the sense of the United Nations Sustainable Development Goals (SDGs), is mostly indirect. Two aspects are nonetheless worth stating.

Dual-use and societal impact (SDG 16). Real-time voice conversion is a dual-use technology. It supports legitimate and even pro-social uses such as privacy, accessibility, and creative expression, but the same capability enables impersonation, fraud, and non-consensual voice deepfakes. A neutral, public benchmark contributes modestly on the constructive side: transparency about what these widely available tools can and cannot do supports informed use and informed defence, where marketing claims alone do not.

Responsible and efficient computation (SDG 12). Voice conversion models and the neural metric predictors used to evaluate them are computationally non-trivial. The benchmark is designed to limit waste: the analysed audio is computed once and cached for reuse across all metrics and reruns, and the corpus is deliberately small and fixed rather than large and redundant. A benchmark that lets practitioners choose an adequate tool without each running their own redundant comparisons is, in a small way, a saving of duplicated effort and energy.

Other dimensions, such as energy use during development, were not found to be significant for a study of this scale, and are not discussed further.

Appendix B

Use of AI

Generative AI tools were used at several points in the production of this thesis. This appendix records where, so the line between the author's own work and the assistance is clear.

Planning. An AI assistant was used to brainstorm which chapters and sections to include and the order to present them in. The final structure, and every decision about what to include, was the author's.

Research. For some of the papers consulted, an AI assistant produced short notes giving a quick sense of what a paper was about and where it might be relevant. These notes were only a starting point: every paper actually used in the thesis was read by the author, who identified and chose the parts that are cited.

Code. The benchmark code was written mostly by hand. An AI assistant was used for debugging and for researching how the individual tools are intended to be run.

Writing and typesetting. The main text was written by the author. An AI assistant was then used for editing passes over language and wording, to make the text clearer, more consistent, and less repetitive, and for help with LaTeX, including formatting and the construction of some figures and tables. The author reviewed and approved all of these changes.

No AI tool was used to generate experimental results or measured numbers. All reported scores are produced by the benchmark pipeline described in Chapter 4.

NTNU

Norwegian University of
Science and Technology