FASTT: Feature-Adaptive Speech-to-Text Model Selection

Huseyin Karaca[a,1,*], A. Samil Namli[a,1], Suleyman S. Kozat[a]

[a] *Bilkent University, Ankara, Turkiye*

Abstract

Existing approaches to automatic speech recognition (ASR) either rely on a single speech-to-text (STT) model or combine multiple models through computationally expensive ensemble techniques, leading to suboptimal trade-offs between accuracy and efficiency across diverse acoustic conditions. Since different STT models exhibit complementary strengths, selecting the most suitable model for each input can significantly improve transcription accuracy. Here, we introduce FASTT, a feature-adaptive STT selection framework that learns to select the most appropriate pretrained STT model for each input. FASTT jointly learns a task-aligned feature transformation and a lightweight selection function based on acoustic and embedding-derived features, and is trained in an end-to-end manner to minimize a cumulative transcription-error objective. Due to this end-to-end training, we learn both the features that best distinguish between STT models and the selection algorithm so that FASTT outperforms static selection and standard routing methods. Across five English corpora spanning read, parliamentary, meeting, financial and crowd-sourced speech, evaluated with 5×2 cross-validation and paired significance tests, FASTT is significantly better than the best single model on four of the five corpora ($p < 0.05$) and statistically tied with it on the fifth, and recovers up to 32% of the per-utterance oracle headroom. Because FASTT selects from the members' encoder outputs and decodes only the chosen member, it avoids up to 86% of the decoding time of a decode-all ensemble with a microsecond-scale selector overhead. Furthermore, we show that the selector training procedures converge to first-order stationary points of the training objective under mild smoothness conditions for a fixed feature transformation, and we publicly release the code to support reproducibility.

Keywords: automatic speech recognition, model selection, feature adaptation, meta-learning, robust ASR

1. Introduction

Automatic speech recognition (ASR) has advanced rapidly with the emergence of large-scale pretrained models and self-supervised representation learning (Latif et al., 2025; Sun et al., 2017; Baevski et al., 2020; Hsu et al., 2021; Radford et al., 2023; Mohamed et al., 2022). Modern speech-to-text (STT) systems such as transformer encoder-decoder architectures and transducer-based models achieve strong performance across diverse domains (Latif et al., 2025; Prabhavalkar et al., 2024; Dong et al., 2018; Gulati et al., 2020). However, despite their scale and robustness, no single STT model consistently achieves optimal accuracy across all acoustic conditions (Sun et al., 2017; Zhu et al., 2024, 2023; Cornell et al., 2026). In real-world deployments, audio inputs demonstrate variability in noise, channel characteristics, speaker identity, and speaking style, leading to input-dependent performance variation across models (Cornell et al.,

*Corresponding author

Email addresses: huseyin.karaca@bilkent.edu.tr (Huseyin Karaca), samil.namli@bilkent.edu.tr (A. Samil Namli), kozat@ee.bilkent.edu.tr (Suleyman S. Kozat)

[1] H. Karaca and A. S. Namli contributed equally to this work.

2026; Watanabe et al., 2020). As a result, different STT models demonstrate complementary strengths, motivating adaptive model selection (Rice, 1976; Vanschoren, 2019; Vettoruzzo et al., 2024). While ensemble methods such as ROVER reduce errors by aggregating multiple hypotheses (Fiscus, 1997), they incur significant computational overhead. In contrast, adaptive selection aims to exploit model diversity while maintaining near single-model efficiency.

A key challenge in adaptive selection is predicting model performance from input-dependent features (Gamper et al., 2020; Ragni et al., 2022; Kastanos et al., 2020). Modern acoustic representations often use a high number of spectral features, prosodic statistics, signal-quality measures, and high-dimensional neural embeddings to distinguish between STT algorithms based on the input audio (Gamper et al., 2020; Zezario et al., 2023; Chen et al., 2022). While expressive, such feature spaces may contain substantial redundancy, correlated dimensions, and weakly informative components, especially when the amount of labeled data available to train the model selection algorithm is limited. In this regime, directly learning a selector over the raw feature space can lead to overfitting, unstable selection behavior, and poor generalization across acoustic domains. The key insight underlying this work is that selection quality depends not only on the selection architecture, but also on the geometry of the feature space in which selection decisions are made (Zezario et al., 2023; Kansizoglou et al., 2022). Here, we introduce FASTT, a feature-adaptive selection framework that learns a task-aligned feature transformation together with a selection function under a cumulative transcription-error objective. By explicitly adapting the feature representation to the selection loss and the selection function in an end-to-end manner, our approach significantly improves generalization across heterogeneous acoustic conditions while maintaining computational efficiency at inference time.

FASTT learns to identify speech characteristics that distinguish the performance regimes of different STT models and to exploit these characteristics to select the most suitable transcription model for each input. Through the learnable feature transformation module, the framework emphasizes attributes most informative for discriminating between candidate models. Because the selection is made from the members' encoder outputs and only the selected model's decoder is executed, the framework avoids the decode-every-member cost of hypothesis-fusion ensembles; the encoder passes that produce the features are part of the online cost and are measured in Section 5.4.2. In addition to the algorithmic design, we provide convergence guarantees, showing that the proposed training procedure converges to first-order stationary points under mild conditions.

1.1. Related Work

Prior work has focused on improving ASR robustness under acoustic variability, including noise, channel distortion, reverberation, and speaker diversity (Sun et al., 2017; Qian et al., 2016; Fan et al., 2021; Ghorbani & Hansen, 2023). Techniques such as multi-condition training, data augmentation, domain adaptation, and large-scale self-supervised pretraining have significantly improved model generalization across diverse environments (Escalona et al., 2025; Chen et al., 2022; Parada et al., 2022; Park et al., 2019; Paraskevopoulos et al., 2024). These approaches aim to train a single model that performs robustly across conditions by increasing representational capacity and data coverage (Baevski et al., 2020; Prabhavalkar et al., 2024; Chen et al., 2022; Ghorbani & Hansen, 2023). Rather than relying on a single robust model, traditional ensemble methods aim to reduce transcription error by aggregating hypotheses from multiple ASR models (Mangu et al., 2000; Hoffmeister et al., 2006; Goel & Byrne, 2000). ROVER (Fiscus, 1997) aligns hypotheses via word-level alignment and performs majority voting, with extensions including confusion-network alignment, consensus decoding, lattice-based combination techniques, and minimum Bayes risk decoding (Mangu et al., 2000; Hoffmeister et al., 2006; Goel & Byrne, 2000; Goel et al., 2004). More recent neural approaches have explored posterior averaging, logit-level fusion, and ensemble decoding (Liu et al., 2021; Kano et al., 2022; Gitman et al., 2023). These methods effectively exploit complementary strengths across models and often yield measurable reductions in transcription error. Building on this line of work, we consider a complementary

formulation that focuses on input-adaptive expert selection rather than hypothesis fusion. Instead of combining multiple model outputs, our framework selects a single pretrained STT expert for each input based on its acoustic characteristics. This retains model diversity at a cost close to single-model inference: the selection decision is a lightweight function of encoder-level features, so only the selected expert's decoder is executed per utterance, and the encoder passes that produce the features are shared with it (Section 5.4.2).

Adaptive expert selection is related to mixture-of-experts (MoE) models (Gan et al., 2026; Jacobs et al., 1991; Shazeer et al., 2017; Fedus et al., 2022; You et al., 2021), in which a gating network assigns input-dependent weights to multiple expert components. Classical MoE architectures jointly train both the experts and the gating function within a single end-to-end model, typically sharing intermediate representations and optimizing a unified likelihood objective. More recent sparse and conditional computation frameworks, such as sparsely gated MoE layers and Switch Transformers (Gan et al., 2026; Shazeer et al., 2017; Fedus et al., 2022), dynamically activate subsets of parameters to improve scalability and computational efficiency. However, our setting differs fundamentally from standard MoE formulations in several respects. First, the experts in our framework are independently pretrained, large-scale STT systems whose parameters remain fixed during selector training. There is no shared backbone, parameter sharing, or joint optimization between the experts and the selection mechanism. This design allows the framework to incorporate arbitrary pretrained STT systems without requiring additional joint training or fine-tuning. Second, the selection objective is not maximum likelihood or classification accuracy within a unified model, but cumulative transcription-error minimization across heterogeneous acoustic conditions. This is important because the ultimate goal of the system is to reduce transcription error in practice.

A related line of work considers confidence-based and quality estimation (QE) approaches for ASR output selection and routing (Kastanos et al., 2020; Li et al., 2021). Posterior probability-based confidence scores can guide rescoring, hypothesis selection, or fallback decisions (Kastanos et al., 2020; Mangu et al., 2000; Ravi et al., 2024; Li et al., 2021). Domain classification and environment detection have also been explored for routing utterances to domain-specialized models (Gitman et al., 2023; You et al., 2022; Kwon & Chung, 2023). Confidence-based approaches require at least partial decoder execution when the confidence is read off the decoder's posteriors, and domain classifiers only capture coarse categorical distinctions (Kastanos et al., 2020; Mangu et al., 2000; Gitman et al., 2023; Li et al., 2021). A quality-estimation router that predicts each expert's error from encoder features avoids the decoder, and we compare against one in Section 5. FASTT likewise operates on encoder-derived features without executing any STT decoder at selection time, but instead of estimating each expert's quality separately it is trained directly on the routing decision, under a cost-sensitive objective that weights every candidate by its actual transcription error.

1.2. Contributions

Our contributions are summarized as follows:

- We introduce FASTT, a modular framework for adaptive STT model selection that jointly learns a task-aligned feature transformation and a selection network under a cumulative transcription-error objective. The framework selects the most suitable STT model for each audio input from the members' encoder outputs, so that only one decoder is executed per utterance.
- We present a principled approach for feature adaptation within the framework, including diagonal gating, linear, low-rank, and nonlinear transformation variants with sparsity and stability regularization. These adaptations improve selection reliability in high-dimensional and data-limited settings.
- We provide two optimization paths for joint training of the feature transformation and the selector: (i) end-to-end backpropagation for differentiable selectors, and (ii) alternating optimization for non-differentiable selectors such as gradient-boosted trees.

- The framework is loss-agnostic, feature-agnostic, and selector-agnostic: any transcription quality metric such as word error rate (WER) or character error rate (CER) can be used (Li et al., 2014), any discriminative audio feature set can be used, and any learning algorithm can serve as the selection function.

- We evaluate FASTT on five English corpora against the best single model, the per-utterance oracle, four hypothesis-fusion ensembles, two argmin classifiers, a quality-estimation router and classical feature selection, using 5×2 cross-validation with paired significance tests, and we measure the end-to-end online latency, including feature extraction, against single-model and decode-all inference.

- We provide our implementation as open-source code to support reproducibility and enable researchers to extend the framework with custom features, selectors, and expert models.[2]

1.3. Organization

The remainder of the paper is organized as follows. Section 2 formalizes the supervised expert-selection problem. Section 3 presents the proposed feature-adaptive selection framework, including the learnable transformation and the selection function. Section 4 presents convergence guarantees to first-order stationary points for the proposed selectors. Section 5 describes the experimental setup and empirical evaluation. Finally, Section 6 concludes the paper.

2. Problem Formulation

We consider the problem of adaptive STT model selection for a collection of audio inputs $\{\mathbf{s}_i\}_{i=1}^N$, where N may be large and the acoustic characteristics may vary substantially across inputs. Each input can demonstrate diverse properties, including noise type and level, channel conditions, speaker identity, accent, speaking rate, and articulation variability, which can significantly affect the transcription accuracy of different STT systems (Yu & Deng, 2015; Li et al., 2014).

Let $\mathcal{E} = \{E_1, E_2, \ldots, E_L\}$ denote a pool of L pre-trained STT experts. For an input utterance $\mathbf{s}_i$ and expert E_j, the produced transcription is written as $y_{i,j} = E_j(\mathbf{s}_i)$. Given the reference transcription y_i^{ref}, the quality of the produced transcript is measured using an error function

$$\ell(y_{i,j},\, y_i^{\text{ref}}) \in [0, 1],$$

which may correspond to WER, CER, or another normalized transcription metric (Li et al., 2014).

In practice, no single STT model is uniformly optimal across heterogeneous acoustic conditions (Yu & Deng, 2015; Radford et al., 2023; Li et al., 2014). Instead, the optimal model achieving the lowest transcription error may vary from one utterance to another depending on the underlying speech characteristics of the input $\mathbf{s}_i$. This motivates the formulation of an expert selection problem in which the system must decide which expert should process each input utterance. Let each input utterance $\mathbf{s}_i$ be represented by a feature vector $\mathbf{z}_i \in \mathbb{R}^p$ computed using a fixed feature-extraction pipeline. A routing policy is defined as a function $\pi : \mathbb{R}^p \to \{1, \ldots, L\}$, which assigns an expert index to each feature vector. The transcription loss incurred by a routing policy π over the dataset is defined as

$$J(\pi) = \frac{1}{N} \sum_{i=1}^{N} \ell\big(y_{i,\pi(\mathbf{z}_i)},\, y_i^{\text{ref}}\big)\,.$$

The objective of adaptive expert selection is to identify the optimal routing policy within a class of admissible policies Π:

$$\pi^\dagger = \arg\min_{\pi \in \Pi} J(\pi).$$

Two policies bound this problem. A constant policy $\pi \equiv j$ commits every utterance to one expert; the best constant policy, selected on labelled data from the deployment conditions, is the single best model of the pool. The per-utterance oracle $\pi^\star(\mathbf{z}_i) = \arg\min_j \ell(y_{i,j}, y_i^{\text{ref}})$ is the lower bound on the attainable error and, like the virtual best solver

[2] Code will be released upon publication.

of a per-instance algorithm portfolio (Rice, 1976; Xu et al., 2008; Bischl et al., 2016), is not a deployable policy, since it reads the reference. Every constant policy belongs to Π, so the adaptive problem contains the static one, and a selector is assessed on two counts: it must not fall below the best constant policy, and it is useful in proportion to the share of the distance from that policy to the oracle that it recovers. Both are reported in Section 5. How much of that distance is recoverable is a property of the pool rather than of the selector. When one expert is best on nearly every utterance, the oracle gap consists of sparse and small disagreements with little acoustic signature, and the best that any policy deciding from $\mathbf{z}_i$ can do is to reproduce the constant one; when the experts are complementary, the gap is wide and structured, and per-utterance selection has something to learn from.

The main challenge in solving this problem comes from the potentially high dimensionality of the feature representation and the limited availability of labeled training examples for learning the routing policy. Furthermore, the relative performance of different STT experts may depend on subtle variations in the acoustic properties of the speech signal. We next introduce a learning-based selection framework that computes an input-dependent distribution over the available STT experts using features extracted from the input audio. To improve selection reliability in high-dimensional feature spaces with limited labeled data, the framework jointly learns a task-aligned feature transformation and a selection network that predicts expert suitability. The algorithm selects a single STT expert from this computed probability distribution to obtain the final transcription in an effective manner.

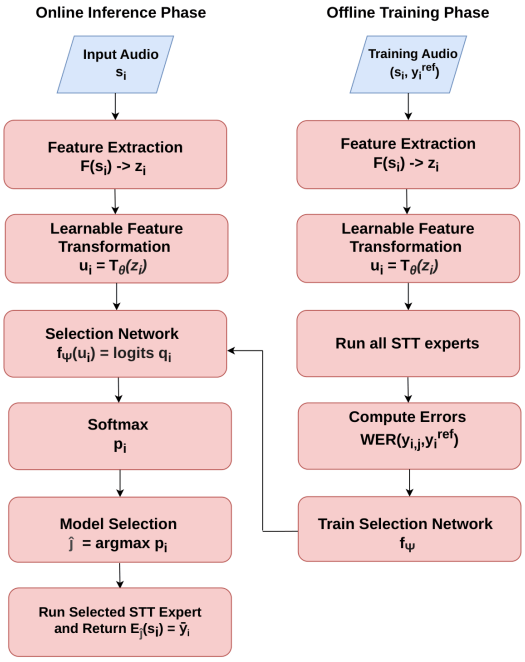


Figure 1: Overview of the FASTT framework. In the offline phase, acoustic features are extracted, transformed through a learnable feature adaptation module, and used to train the selection function by minimizing an expected transcription-error objective. At inference time, the member encoders are run once, the selector chooses from their outputs, and only the selected expert's decoder is executed.

3. FASTT: Feature-Adaptive Speech-to-Text Model Selection

3.1. Algorithm Overview

The algorithm consists of two phases: an offline training phase and an online inference phase, as illustrated in Fig. 1. In the offline phase, the system learns how to predict the relative suitability of different STT experts based on input-dependent audio characteristics. For each labeled training input $\mathbf{s}_i$, the framework extracts a feature vector $\mathbf{z}_i$ using the feature pipeline described in Section 3.2. This representation aggregates utterance-level descriptors of the audio; the descriptor families the pipeline can draw on, and the instantiation used in this work, are given in Section 3.2. To improve selection reliability in high-dimensional feature spaces, the framework learns a parametric feature transformation that maps the original feature vector $\mathbf{z}_i$ into a task-aligned representation, as shown in Fig. 1. This transformation reshapes the feature space to better capture the relationship between acoustic characteristics and expert performance. The transformed features are then passed to a selec-

tion function that produces a weight vector over the candidate STT experts $\{E_1, \ldots, E_L\}$. The parameters of the feature transformation and the selection function are learned during the offline phase using evaluations of all experts on labeled audio clips, minimizing the cumulative transcription loss defined in Section 2.

In the online inference phase, given a new input audio $\mathbf{s}_i$, the system first extracts the feature vector $\mathbf{z}_i$, as shown in Fig. 1. The trained selection model then computes a weight vector $\mathbf{p}_i$ over the candidate experts based on the transformed feature representation. A single expert $E_{\hat{j}_i}$ is selected according to the highest predicted probability, and the final transcription $\hat{y}_i = E_{\hat{j}_i}(\mathbf{s}_i)$ is obtained by executing only that expert. Because only one STT decoder is executed at inference time, the framework exploits the performance diversity of multiple pretrained experts at the cost of one decode plus the encoder passes that produce the features; the measured breakdown is given in Section 5.4.2. The complete online inference procedure is summarized in Algorithm 1, and the full pipeline is illustrated in Fig. 1.

Algorithm 1 Online STT model selection via FASTT

Inputs:

- Unlabeled input audio clip $\mathbf{s}_i$
- Set of audio feature extraction functions $\mathbb{F}(\cdot)$
- Trained FASTT selection pipeline f_{FASTT}
- STT expert pool $\mathcal{E} = \{E_1, \ldots, E_L\}$

Output:

- Final transcription $\hat{y}_i$ for audio $\mathbf{s}_i$

Inference Steps:

1: Extract utterance-level features:

$$\mathbf{z}_i \leftarrow \mathbb{F}(\mathbf{s}_i)$$

2: Apply the trained FASTT pipeline to compute selection logits:

$$\mathbf{q}_i \leftarrow f_{\text{FASTT}}(\mathbf{z}_i)$$

3: Normalize $\mathbf{q}_i$ via softmax to obtain expert probabilities $\mathbf{p}_i$

4: Select the expert index:

$$\hat{j}_i \leftarrow \arg\max_{j \in \{1,\ldots,L\}} p_{i,j}$$

5: Return final transcription:

$$\hat{y}_i \leftarrow E_{\hat{j}_i}(\mathbf{s}_i)$$

3.2. Feature Representation

For each incoming audio input $\mathbf{s}_i$, the feature pipeline $\mathbb{F}$ constructs a feature vector $\mathbf{z}_i \in \mathbb{R}^p$ that encodes acoustic and contextual information relevant to STT performance, as shown in Fig. 1. The pipeline is a fixed, selector-independent stage: any utterance-level descriptor that correlates with the relative performance of the experts can enter it, and the framework places no constraint on its composition beyond a fixed dimensionality p. We group the descriptors it can draw on into four families, which summarize signal quality, phonetic content, speaker characteristics, and temporal dynamics (Rabiner & Juang, 1993; Li et al., 2023), and then state the instantiation used in this work.

3.2.1. Spectral–Phonetic Features

Mel-frequency cepstral coefficients (MFCCs) with their first- and second-order derivatives (Δ and $\Delta\Delta$), aggregated over the utterance, are the classical spectral–phonetic descriptors (Davis & Mermelstein, 1990; Furui, 1986). Lower-order coefficients represent the coarse spectral envelope, while higher-order terms encode finer phonetic detail, and the temporal derivatives add robustness to coarticulation, prosodic variation, and speaker-specific articulation (Rabiner & Juang, 1993).

3.2.2. Prosodic Features

Prosodic features, including pitch statistics (mean, range) and silence ratios derived from voice activity detection (VAD), characterize speaking rhythm, stress, and expressiveness (Sohn et al., 1999). Such features are particularly valuable for identifying conditions where transcription accuracy may degrade under expressive, emotional, or noisy speech (Li et al., 2023).

3.2.3. Neural Speech Embeddings

High-level speech representations are obtained by pooling the hidden states of a pretrained speech encoder over the utterance. The encoder can be a

self-supervised model such as Wav2Vec2 or HuBERT (Baevski et al., 2020; Hsu et al., 2021), or the encoder of an STT expert itself. These embeddings encapsulate rich information about phonetic identity, speaker traits, and acoustic conditions, complementing conventional low-level features. They are computed online for every utterance before any decoder runs, so their cost is part of the end-to-end latency of the system and is reported as such in Section 5.4.2. In our implementation the embeddings are taken from the encoders of the member models themselves, so the encoder forward passes are shared between feature extraction and the decoding of the selected member and are paid once per utterance; only the decoder of the selected member is executed. For resource-constrained deployment scenarios, lighter embedding models or knowledge-distilled variants could be substituted without modifying the framework.

3.2.4. Signal and Temporal Features

Signal-level features including signal-to-noise ratio (SNR), zero-crossing rate (ZCR), and spectral entropy quantify acoustic clarity and spectral irregularity (Schuller & Batliner, 2013; Eyben et al., 2016). Temporal features such as utterance duration, speech rate, and pause ratio measure fluency and pacing, both of which significantly impact STT robustness (Ko et al., 2015; Tripathi et al., 2020).

3.2.5. Features Used in This Work

In the experiments of Section 5 the feature vector consists of the mean-pooled final-layer encoder states of the three member models, concatenated with the log clip duration, so that p is one plus the sum of the members' encoder widths (1,793–3,329 depending on the corpus; Table 1). This choice keeps feature extraction entirely inside the encoder passes the system pays in any case: no separate acoustic front-end runs at inference time, and the embedding of each member is available before its decoder is invoked. Because the selector reads the members' own representations of the input, the cues it can act on are those the members themselves compute, which is where their disagreements originate. The low-level descriptor families above are supported by the pipeline and can be appended to $\mathbf{z}_i$ without any change to the transformation or the selector. The resulting vector serves as the conditioning signal for the supervised selection function.

Remark 1. *The proposed framework is feature-agnostic. The selection network operates on a generic feature vector and can incorporate any set of discriminative features, such as acoustic, linguistic, prosodic, model-intrinsic, or domain-specific features that correlate with relative STT performance.*

3.3. Learnable Feature Transformation

The feature representation $\mathbf{z}_i \in \mathbb{R}^p$ described in Section 3.2 captures diverse acoustic and embedding-based properties of the input audio. While such descriptors provide informative signals for expert selection, their raw representation may not be optimally structured for predicting relative expert performance. In particular, high-dimensional feature spaces may contain correlated dimensions or redundant information that can make the selection model more difficult to train when labeled data are limited.

Classical approaches to dimensionality reduction in such settings include filter-based methods, such as mutual information ranking and variance thresholding, wrapper-based strategies, such as sequential feature selection (Guyon & Elisseeff, 2003; Kohavi & John, 1997), and embedded methods that couple feature selection with model training (Friedman, 2001). However, filter methods disregard inter-feature dependencies and their interaction with the downstream objective, wrapper approaches have combinatorial search costs that scale poorly with p, and embedded methods are typically tied to specific model families. These limitations motivate a learnable, task-aligned feature transformation that can be jointly optimized with any downstream selector.

We introduce a parametric mapping applied prior to the selection network that reshapes the geometry of the feature space without modifying the feature extraction pipeline itself, as illustrated in Fig. 1. The transformed representation is defined as

$$\mathbf{u}_i = \mathbb{T}_\theta(\mathbf{z}_i),$$

where $\mathbb{T}_\theta : \mathbb{R}^p \to \mathbb{R}^{d'}$ denotes a parametric transformation with learnable parameters θ, and $\mathbf{u}_i$ serves as the input to the downstream expert selection function. The optimization strategy for θ depends on whether the downstream selector is differentiable, as discussed in 3.5. We present several transformation variants below; all are compatible with any selector variant within the framework.

3.3.1. Diagonal Feature Gating

As a capacity-controlled transformation, we consider feature-wise scaling:

$$\mathbf{u}_i = \mathbf{g} \odot \mathbf{z}_i,$$

where $\mathbf{g} \in \mathbb{R}^p$ is a learnable gating vector and $\odot$ denotes element-wise multiplication. Each component g_j assigns an importance weight to the corresponding feature dimension j. This formulation acts as a differentiable feature-selection mechanism: dimensions with small g_j values are effectively suppressed, while informative features receive larger weights. Because the number of additional parameters is limited to p, the diagonal gating transformation provides a simple yet effective mechanism for controlling overfitting in high-dimensional, data-limited regimes. To promote sparsity and stability, we apply combined ℓ_1/ℓ_2 regularization:

$$\lambda_1 \|\mathbf{g}\|_1 + \lambda_2 \|\mathbf{g}\|_2^2.$$

3.3.2. Linear Feature Transformation

A more expressive alternative is a full linear projection:

$$\mathbf{u}_i = \mathbf{W}\mathbf{z}_i + \mathbf{b},$$

where $\mathbf{W} \in \mathbb{R}^{d' \times p}$ is a learnable projection matrix and $\mathbf{b} \in \mathbb{R}^{d'}$ is a bias vector. This transformation allows the model to reweight and mix the original feature dimensions in a task-aligned manner. By choosing $d' < p$, the transformation acts as a dimensionality-reduction mechanism. Weight decay regularization is applied to $\mathbf{W}$ to control overfitting.

3.3.3. Low-Rank Linear Transformation

While the full linear transformation allows arbitrary mixing of feature dimensions, it may introduce a large number of parameters when p is high. To provide a more capacity-controlled alternative, we consider a low-rank factorization of the projection matrix:

$$\mathbf{u}_i = \mathbf{W}_2\mathbf{W}_1\mathbf{z}_i,$$

where $\mathbf{W}_1 \in \mathbb{R}^{r \times p}$, $\mathbf{W}_2 \in \mathbb{R}^{d' \times r}$, and $r \ll \min(p, d')$ is a bottleneck dimension. This factorization constrains the rank of the projection, allowing the model to capture interactions between feature dimensions while reducing the number of learnable parameters.

3.3.4. Nonlinear Feature Transformation

To capture nonlinear feature interactions that may affect relative expert performance, we consider a nonlinear bottleneck transformation:

$$\mathbf{u}_i = \mathbf{W}_2\,\sigma(\mathbf{W}_1\mathbf{z}_i),$$

where $\sigma(\cdot)$ is an element-wise activation function such as ReLU or GELU, $\mathbf{W}_1 \in \mathbb{R}^{r \times p}$ and $\mathbf{W}_2 \in \mathbb{R}^{d' \times r}$ are learnable weight matrices, and $r \ll \min(p, d')$ is the bottleneck dimension. The bottleneck structure constrains the parameter count and helps prevent overfitting when labeled data are limited.

Remark 2. *The proposed framework is flexible with respect to the choice of feature transformation. In addition to the variants presented here, other parametric transformations such as deeper neural encoders, kernel-based mappings, or representation learning modules could be used without modifying the overall pipeline. All transformation variants are evaluated in Section 5.*

3.4. Selection Objective

Given the transformed feature vector $\mathbf{u}_i \in \mathbb{R}^{d'}$ obtained from the feature transformation stage described in 3.3, the goal of the selection function is to assign preference scores to a set of L pretrained STT experts $\mathcal{E} = \{E_1, \ldots, E_L\}$. We define the selection function as a parametric mapping

$$f_\psi(\mathbf{u}_i) = \mathbf{q}_i \in \mathbb{R}^L,$$

where ψ denotes the trainable parameters and $\mathbf{q}_i = (q_{i,1}, \ldots, q_{i,L})$ represents the logits associated with

the candidate STT experts. Each component $q_{i,j}$ reflects the relative suitability of expert E_j for the audio input $\mathbf{s}_i$. We apply a softmax operation to the logits to obtain a normalized selection probability for each expert:

$$p_{i,j} = \frac{\exp(q_{i,j})}{\sum_{m=1}^{L} \exp(q_{i,m})}, \qquad j = 1, \ldots, L, \quad (1)$$

where $p_{i,j}$ denotes the probability assigned to expert E_j. Given these probabilities, we define the expected transcription loss for the audio clip $\mathbf{s}_i$ as

$$\mathcal{L}_i = \sum_{j=1}^{L} p_{i,j} \, \mathrm{WER}(y_{i,j}, y_i^{\mathrm{ref}}), \quad (2)$$

where y_i^{ref} is the reference transcription. The WER between a hypothesis transcription $y_{i,j}$ and the reference y_i^{ref} is defined as

$$\mathrm{WER}(y_{i,j}, y_i^{\mathrm{ref}}) = \frac{S + I + D}{|y_i^{\mathrm{ref}}|},$$

where S, I, and D denote the numbers of substitutions, insertions, and deletions obtained from a standard Levenshtein alignment (Li et al., 2014).

Remark 3. *Although we use WER as the optimization objective due to its widespread use and interpretability in speech recognition research (Yu & Deng, 2015; Li et al., 2014), other transcription quality metrics such as CER or task-specific error measures could also be used within the same framework.*

The overall training objective minimizes the expected transcription error across the dataset:

$$\min_{\theta,\psi} \mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_i = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{L} p_{i,j} \, \mathrm{WER}(y_{i,j}, y_i^{\mathrm{ref}}). \quad (3)$$

During training, the WER values $\mathrm{WER}(y_{i,j}, y_i^{\mathrm{ref}})$ are treated as fixed constants obtained from offline evaluation of the STT experts. The optimization strategy for the feature transformation parameters θ and the selection function parameters ψ depends on whether the selection function is differentiable with respect to its inputs and parameters, as discussed in the following subsection.

At inference time the expert is chosen by the arg max of $\mathbf{p}_i$, whereas Eq. 3 minimizes the expected error under the soft distribution. We use the soft objective as a surrogate for two reasons. First, the error of the arg max selection is piecewise constant in ψ and θ and therefore provides no gradient, while Eq. 3 is smooth and admits both gradient-based training and the second-order boosting objective used below. Second, unlike a cross-entropy loss against the argmin expert, the surrogate is cost sensitive: it weights every candidate by its actual transcription error, so it is indifferent on utterances where the experts tie and concentrates on utterances where a wrong choice is expensive. The two objectives coincide as $\mathbf{p}_i$ approaches a one-hot vector, which is the regime the softmax moves towards during training. The gap between the soft surrogate and the hard arg max selection is measured empirically in Section 5, where the same architectures are trained as argmin classifiers with a plain and a cost-sensitive cross-entropy loss and compared with the expected-WER objective.

3.5. Joint Feature and Model Selection

A key contribution of the FASTT framework is the joint training of the feature transformation $\mathbb{T}_\theta$ and the selection function f_ψ. Instead of performing feature selection and model selection as two separate steps, FASTT learns both components together using the expected transcription error objective. This joint training aligns the feature representation directly with the model selection task.

The framework supports two optimization paths depending on the differentiability of the selection function.

3.5.1. FASTT for Differentiable Model Selectors

When the selection function $f_\psi(\cdot)$ is differentiable with respect to its parameters ψ and its inputs $\mathbf{u}_i$, the parameters θ and ψ are optimized jointly using standard backpropagation. Gradients of the expected word error rate (WER) objective flow from the softmax probabilities through the selection network and into the feature transformation. This strategy enables

end-to-end feature adaptation in a single training loop without requiring auxiliary optimization steps.

As a representative differentiable model, we use Soft Decision Trees (SDTs). SDTs are hierarchical models in which each internal node makes a soft, probabilistic routing decision based on a linear combination of features. Because these routing decisions are continuous rather than fully discrete, SDTs are differentiable and suitable for gradient-based training. We construct our selector, FASTT-SDT, using an additive ensemble of SDTs, as in Karaca & Kozat (2026). The model consists of B sequential boosting rounds. At each round b, a per-round feature transformation $\mathbb{T}_{\theta_b}(\cdot)$ is applied to the input features, followed by a soft decision tree $h_b(\cdot)$ producing L logits. The final output is the sum of logits across all boosting rounds:

$$\mathbf{q}_i = \sum_{b=1}^{B} h_b(\mathbb{T}_{\theta_b}(\mathbf{z}_i)) \,. \tag{4}$$

At each round, the current tree is fit on the gradient of the previous predictions. The per-round feature transformation $\mathbb{T}_{\theta_b}$ and the soft decision tree h_b are optimized jointly using backpropagation on the expected-WER objective in Eq. 3. A key property of the differentiable FASTT selector is that each boosting round learns its own feature transformation, allowing different rounds to focus on different subsets or combinations of the input features. The complete offline training procedure is given in Algorithm 2.

3.5.2. FASTT for Non-Differentiable Model Selectors

Tree-based ensemble methods such as gradient-boosted decision trees perform well on structured tabular data and provide strong regularization. However, they are not differentiable with respect to their input features. Gradients cannot be propagated through the discrete tree structure to update the parameters of the feature transformation $\mathbb{T}_\theta$. This creates an optimization challenge, since θ and ψ cannot be trained jointly via standard backpropagation.

To enable joint adaptation without requiring differentiability, FASTT uses an alternating optimization strategy (Bertsekas, 2016). We decompose the joint training problem into two subproblems that are optimized iteratively:

Algorithm 2 Offline training: FASTT for differentiable selectors

Inputs:

- $\{(\mathbf{s}_1, y_1^{\text{ref}}), \ldots, (\mathbf{s}_N, y_N^{\text{ref}})\}$: labeled audio clips with reference transcriptions
- $\mathbb{F}$: feature extraction pipeline
- $\mathcal{E} = \{E_1, \ldots, E_L\}$: pretrained STT expert pool
- B: number of boosting rounds

Output: Trained FASTT model $\{(\mathbb{T}_{\theta_b}, h_b)\}_{b=1}^{B}$

Offline data preparation:
for $i \leftarrow 1$ **to** N **do**
  Extract features: $\mathbf{z}_i \leftarrow \mathbb{F}(\mathbf{s}_i)$
  for each expert $E_j \in \mathcal{E}$ **do**
    Transcribe: $y_{i,j} \leftarrow E_j(\mathbf{s}_i)$
    Compute: $\text{WER}_{i,j} \leftarrow \text{WER}(y_{i,j}, y_i^{\text{ref}})$
  end for
end for
Initialize ensemble prediction: $\hat{\mathbf{q}}_i^{(0)} \leftarrow \mathbf{0}$ for all i
Boosted training:
for round $b \leftarrow 1$ **to** B **do**
  Compute residual targets from $\hat{\mathbf{q}}_i^{(b-1)}$ using Eq. 2
  Initialize per-round feature transform $\mathbb{T}_{\theta_b}$
  Fit base model h_b on $\{(\mathbb{T}_{\theta_b}(\mathbf{z}_i), \text{residual}_i)\}$
  Jointly optimize θ_b and h_b via backpropagation with ℓ_1 regularization on θ_b
  Update ensemble:

$$\hat{\mathbf{q}}_i^{(b)} \leftarrow \hat{\mathbf{q}}_i^{(b-1)} + h_b(\mathbb{T}_{\theta_b}(\mathbf{z}_i))$$

end for
Online inference: For a new audio input $\mathbf{s}_i$, compute

$$\mathbf{q}_i = \sum_{b=1}^{B} h_b\big(\mathbb{T}_{\theta_b}(\mathbb{F}(\mathbf{s}_i))\big),$$

apply softmax, and select $\hat{j}_i = \arg\max_j p_{i,j}$

1. *Selector update (fix θ, optimize ψ):* Given the current feature transformation $\mathbb{T}_\theta$, compute the transformed features $\mathbf{u}_i = \mathbb{T}_\theta(\mathbf{z}_i)$ and train the tree-based selector on these features.

2. *Transform update (fix ψ, optimize θ):* With the tree structure fixed, update the feature transformation parameters θ.

This alternating procedure is repeated for a fixed number of iterations. Because the tree ensemble f_ψ

is piecewise constant in its input, the transform update cannot backpropagate through it. We therefore fit a differentiable *surrogate* of the fixed selector: a linear map $\tilde{f}_\phi(\mathbf{u}) = \mathbf{A}\mathbf{u} + \mathbf{c}$ is fitted by least squares to the selector's own logits on the training set, $\phi = \arg\min_\phi \frac{1}{N}\sum_i \|\tilde{f}_\phi(\mathbf{u}_i) - f_\psi(\mathbf{u}_i)\|_2^2$, using a fixed number of Adam steps. With ϕ frozen, θ is then updated by gradient descent on the expected-WER objective of Eq. 3 evaluated at the composite logits $\mathbf{q}_i = \tilde{f}_\phi(\mathbb{T}_\theta(\mathbf{z}_i))$, plus the transformation's regularizer, with gradient-norm clipping. The surrogate thus provides a first-order model of how the fixed trees respond to a change in the feature geometry; the trees are refitted on the updated features in the next selector update. The number of surrogate and transform steps and their learning rates are hyperparameters (Table 2). The complete training procedure is given in Algorithm 3.

As a representative non-differentiable selector in this path, we utilize XGBoost (Chen & Guestrin, 2016). Crucially, the selector must minimize the expected-WER objective instead of conventional classification or regression losses. Standard objectives such as cross-entropy align with categorical targets, whereas FASTT requires minimizing a continuous, sequence-level transcription error over a probability distribution across candidate STT experts. Therefore, XGBoost is trained with a custom objective function. We supply the algorithm with specifically formulated first- and second-order derivatives to integrate expected-WER minimization into the tree-building process.

Let $g_{i,j}$ and $h_{i,j}$ denote the first- and second-order derivatives of $\mathcal{L}_i$ with respect to the output logit $q_{i,j}$:

$$g_{i,j} = \frac{\partial \mathcal{L}_i}{\partial q_{i,j}} = p_{i,j}\left(\mathrm{WER}(y_{i,j}, y_i^{\mathrm{ref}}) - \mathcal{L}_i\right). \quad (5)$$

To reduce computational cost and maintain stable split gains, we use a diagonal approximation to the Hessian:

$$h_{i,j} = \frac{\partial^2 \mathcal{L}_i}{\partial q_{i,j}^2} = p_{i,j}(1 - 2p_{i,j})\left(\mathrm{WER}(y_{i,j}, y_i^{\mathrm{ref}}) - \mathcal{L}_i\right). \quad (6)$$

We clip negative Hessian values to a small positive constant to ensure stable tree building (Chen & Guestrin, 2016).

Algorithm 3 Offline training: FASTT for non-differentiable selectors

Inputs:

- $\{(\mathbf{s}_1, y_1^{\mathrm{ref}}), \ldots, (\mathbf{s}_N, y_N^{\mathrm{ref}})\}$: labeled audio clips with reference transcriptions
- $\mathbb{F}$: feature extraction pipeline
- $\mathcal{E} = \{E_1, \ldots, E_L\}$: pretrained STT expert pool
- $\mathbb{T}_\theta$: learnable feature transformation
- T: number of alternating iterations

Output: Trained feature transformation $\mathbb{T}_\theta$ and selector f_ψ

Offline data preparation:
for $i \leftarrow 1$ **to** N **do**
  Extract features: $\mathbf{z}_i \leftarrow \mathbb{F}(\mathbf{s}_i)$
  for each expert $E_j \in \mathcal{E}$ **do**
    Transcribe: $y_{i,j} \leftarrow E_j(\mathbf{s}_i)$
    Compute: $\mathrm{WER}_{i,j} \leftarrow \mathrm{WER}(y_{i,j}, y_i^{\mathrm{ref}})$
  end for
end for
Alternating optimization:
for iteration $t = 1, \ldots, T$ **do**
  Transform features: $\mathbf{u}_i \leftarrow \mathbb{T}_\theta(\mathbf{z}_i)$ for all i
  Compute softmax weights $\mathbf{p}_i$ via Eq. 1
  Compute per-instance gradients $g_{i,j}$ and Hessians $h_{i,j}$ via Eq. 5, Eq. 6
  Selector update: Train XGBoost f_ψ on $\{\mathbf{u}_i\}$ using the custom WER-based objective
  Obtain updated logits: $\mathbf{q}_i \leftarrow f_\psi(\mathbf{u}_i)$
  Surrogate fit: Fit the linear surrogate $\tilde{f}_\phi$ to $\{(\mathbf{u}_i, \mathbf{q}_i)\}$ by least squares (fixed number of Adam steps)
  Transform update: Fix ϕ; for a fixed number of steps, descend Eq. 3 evaluated at $\tilde{f}_\phi(\mathbb{T}_\theta(\mathbf{z}_i))$ plus the regularizer of $\mathbb{T}_\theta$ with respect to θ (Adam, gradient clipping)
end for
Online inference: For a new audio input $\mathbf{s}_i$, compute

$$\mathbf{q}_i = f_\psi(\mathbb{T}_\theta(\mathbb{F}(\mathbf{s}_i))),$$

apply softmax, and select $\hat{j}_i = \arg\max_j p_{i,j}$

Remark 4. *The framework is flexible and can accommodate a wide range of learning algorithms for the selection function. In addition to the variants presented here, other architectures such as transformers (Vaswani et al., 2017) or alternative supervised learn-*

ing algorithms could be used without modifying the overall pipeline.

3.6. Sampling Strategy

Given the selection probabilities $\mathbf{p}_i = (p_{i,1}, \ldots, p_{i,L})$ produced by the selection function, we choose the final expert as

$$\hat{j}_i = \arg \max_{j \in \{1,\ldots,L\}} p_{i,j}.$$

The selected expert is then executed to produce the final transcription

$$\hat{y}_i = E_{\hat{j}_i}(\mathbf{s}_i),$$

as shown in Fig. 1. The online cost per utterance is therefore the encoder forward passes that produce the features, the selector, and the decoder of the single selected expert. The selector itself runs in microseconds; the encoder passes are shared with the selected expert's decoding and are paid once, whereas a hypothesis-fusion ensemble must additionally decode every expert. Section 5.4.2 reports the measured end-to-end latency of each component.

4. Mathematical Proof of Convergence

We analyze the convergence of the two selector optimization procedures presented in 3.5 for a *fixed* feature transformation $\mathbb{T}_\theta$. Under mild regularity conditions, both procedures converge to a first-order stationary point of the training objective in the selector parameters. Since the objective is non-convex, a first-order stationary point is not in general a minimizer, and the results below make no claim about optimality, about the stability of the trained selector, or about the joint problem in (θ, ψ).

The proofs hold for any smooth loss with bounded curvature, consistent with the framework's loss-agnostic design, and therefore also cover alternative admissible objectives such as cross-entropy, mean squared error, or other smooth sequence-level surrogates. In the full framework the transformation parameters are updated either jointly, for differentiable selectors, or through the alternating procedure of Section 3.5.2. In the alternating case each selector update is an instance of the subproblem analyzed here, but the surrogate-based transform update is not covered by the analysis; see Remark 5.

4.1. XGBoost-based Selector

For the XGBoost-based selector, we prove convergence under our custom WER-based objective by extending standard second-order boosting theory. Assuming bounded loss values, Lipschitz-continuous gradients, and a weak-learner edge condition, we obtain a monotone descent property. Let $L(\mathbf{q}^{(n)})$ denote the training loss evaluated at the current logits $\mathbf{q}^{(n)}$.

Under these assumptions, and for a sufficiently small step size η, the training dynamics admit the following descent inequality:

$$L_{n+1} \leq L_n - \gamma \|\nabla L(\mathbf{q}^{(n)})\|^2, \tag{7}$$

where $\gamma > 0$ is a constant that depends on the learning rate, the weak-learner edge, and the smoothness of the loss function. Summing over iterations yields an $\mathcal{O}(1/N)$ convergence rate to a first-order stationary point. Full derivations and the explicit form of γ are provided in Appendix A.

4.2. Neural Selector (MLP / FASTT-SDT)

For the neural selector, we rely on standard non-convex optimization results under smoothness of the loss function and bounded variance of the stochastic gradients. Under these assumptions, the training dynamics satisfy an $\mathcal{O}(1/N)$ convergence rate for full-batch gradient descent and an $\mathcal{O}(1/\sqrt{N})$ convergence rate for mini-batch SGD, implying convergence to a first-order stationary point. Detailed derivations are provided in Appendix A.

Remark 5. *These results characterize the selector optimization for a fixed feature transformation. For differentiable selectors the transformation parameters enter the same smooth objective and the same analysis applies to the joint parameter vector (θ, ψ), provided the transformation satisfies the smoothness assumption. For the alternating procedure of Section 3.5.2, the transform update descends a surrogate rather than the selector itself, and a convergence guarantee for the joint system remains an open question that we leave for future investigation. In practice we bound the number of alternating iterations and select it on held-out data.*

5. Experiments and Results

We evaluate the proposed FASTT framework using the selector and transformation variants described in 3.3, 3.5. The experiments are designed to (i) compare the framework variants against external baselines and classical feature selection strategies, (ii) ablate the contribution of the feature transformation and of the training objective, and (iii) measure the end-to-end inference cost.

5.1. Datasets and Feature Extraction

We evaluate on five publicly available English corpora that span a wide range of acoustic conditions: LibriSpeech (Panayotov et al., 2015) (read audiobook speech; the *dev-clean*, *dev-other*, *test-clean* and *test-other* splits), VoxPopuli's English subset (Wang et al., 2021) (European Parliament speech with accented English and varying room acoustics; the validation and test splits), the single-distant-microphone (SDM) condition of AMI (Carletta et al., 2006) (far-field meeting recordings; the validation and test splits plus a contiguous prefix of the training shards), Earnings-22 (Río et al., 2022) (earnings calls with diverse accents and telephone-quality channels; the chunked release, with disjoint shard ranges for fitting and evaluation), and the *dirty* subset of People's Speech (Galvez et al., 2021) (crowd-sourced speech with noisy, loosely aligned references; a contiguous prefix of each split). We use the segmentation shipped with each corpus and keep every clip with non-empty audio and a non-empty reference; no further clip selection is applied. Each corpus is evaluated separately, so that the selector is trained and tested under the acoustic conditions of one domain at a time, and each corpus is split once, with a fixed seed, into a 25% hyperparameter hold-out and a 75% evaluation partition that is disjoint from it (Section 5.3). The dataset statistics are summarized in Table 1.

Each clip is represented by a feature vector $\mathbf{z}_i \in \mathbb{R}^p$ with p between 1,793 and 3,329 depending on the corpus (Table 1), composed as described in Section 3.2: the mean-pooled final-layer encoder states of the three member models and the log clip duration. Since the member trios differ across corpora (Section 5.2), p varies with the members' encoder widths. This high-dimensional representation, with p of the same order as the number of training clips per fold (3,600–15,600), is the regime that motivates the learnable feature transformation: a selector trained directly on all p features has to separate the informative directions of the embedding space from the redundant ones with a comparable number of examples.

5.2. Models and Baselines

5.2.1. Individual STT Experts

For each corpus we use a pool of $L = 3$ pre-trained STT experts. On VoxPopuli, AMI-SDM, Earnings-22 and People's Speech the experts are drawn from five publicly released models that were trained or fine-tuned on LibriSpeech only: the ESPnet Transformer encoder–decoder recipe and the ESPnet Conformer–RNN-T transducer (Watanabe et al., 2018; Gulati et al., 2020; Graves, 2012), the wav2vec2-base-960h and wav2vec2-large-960h CTC models (Baevski et al., 2020), and data2vec-audio-base (Baevski et al., 2022). The trios are Conformer–RNN-T, Transformer and data2vec-audio-base on VoxPopuli; Conformer–RNN-T, Transformer and wav2vec2-large on AMI-SDM; data2vec-audio-base, Conformer–RNN-T and wav2vec2-large on Earnings-22; and data2vec-audio-base, Transformer and wav2vec2-base on People's Speech. Because every member is trained on read audiobook speech, these four corpora are out of domain for all members, and the relative ranking of the members changes from corpus to corpus, which is the regime in which adaptive selection is meaningful. LibriSpeech is evaluated with a trio of general-purpose models, Parakeet-TDT-0.6b-v2 (NVIDIA, 2025b), Canary-Qwen-2.5b (NVIDIA, 2025a) and Whisper-large-v3 (Radford et al., 2023), for which read audiobook speech is in domain; it serves as the control corpus on which one member is expected to dominate and adaptive selection has little to gain. For each clip $\mathbf{s}_i$ all three members are run offline to obtain transcriptions $\{y_{i,j}\}_{j=1}^{3}$ and their word error rates against the reference transcripts; the members' encoder states are also mean-pooled to form the embedding block of $\mathbf{z}_i$ (Section 5.1).

Table 1: Dataset statistics. Clips are those with non-empty audio and reference; the hyperparameter hold-out (25%) is disjoint from the 5×2 cv evaluation rows.

Statistic	**LibriSpeech**	**VoxPopuli**	**AMI (SDM)**	**Earnings-22**	**People's Speech**
Total clips	11,126	9,678	41,641	12,088	14,299
Features	3,329	1,793	2,049	2,305	2,049
Hyperparameter hold-out	2,782	2,420	10,410	3,022	3,575
Evaluation rows (5x2)	8,344	7,258	31,231	9,066	10,724
Train size (per fold)	4,172	3,629	15,615	4,533	5,362
Test size (per fold)	4,172	3,629	15,616	4,533	5,362

Table 2: Hyperparameter search spaces. Every arm draws 10 configurations with the same sampler and seed on the corpus's hold-out partition; fixed parameters (a single-choice grid) are omitted. The SDT and XGBoost groups collapse the near-identical grids of the FASTT, raw and SelectKBest variants of each base learner into one representative set; k is the SelectKBest feature count.

Parameter	**Search space**	**Parameter**	**Search space**
SDT		*SelectKBest*	
Transform ℓ_2 (λ_2)	1e-05, 0.001, 0.1	Num. features (k)	32, 128, 512, 2048
Bottleneck ratio	0.05, 0.15, 0.5	*ROVER (weighted)*	
Boosting rounds	1, 2, 4, 6	Vote weighting	inverse_wer, log_odds, uniform
Trees per round	4, 8, 16, 24	Null-arc penalty	0.25, 0.5
SDT ℓ_1 (λ)	1e-06, 0.0001, 0.01, 1.0	*ROVER (confidence-weighted)*	
Weight decay	1e-07, 1e-05, 0.001, 0.1	Smoothing (α)	0.0, 0.25
Batch size	128, 512, 2048	Vote weighting	inverse_wer, log_odds, uniform
Early-stopping patience	10, 20, 40	Weight by confidence	False, True
XGBoost		*CN-MBR*	
Alternating iterations	1, 3, 6, 12	Use confidence	False, True
Transform learning rate	0.0001, 0.001, 0.01	Confidence source	blend, cn, probe
Transform steps	25, 100, 400	*Score-Fusion*	
Surrogate learning rate	0.001, 0.01, 0.1	Null-arc penalty	0.25, 1.0, 4.0
Surrogate steps	50, 200, 400	*Hard-Selector (plain CE)*	
Transform ℓ_1 (λ_1)	1e-05, 0.001, 0.1	Smoothing (α)	0.0001, 0.001, ..., 1000 (8 values)
Transform ℓ_2 (λ_2)	1e-05, 0.001, 0.1	*Hard-Selector (cost-sensitive CE)*	
Learning rate	0.01, 0.03, 0.1, 0.3	Smoothing (α)	100.0, 1000.0
Max depth	3, 5, 8, 12	Objective weighting	length, uniform
Subsample	0.6, 0.8, 1.0	*QE-Router*	
Column subsample	0.1, 0.3, 0.6, 1.0	Regression target	errors, wer
Bottleneck output ratio	0.1, 0.3, 1.0	Num. α values	8, 16, 40
		Standardize features	False, True

5.2.2. External Baselines

We compare against reference points, hypothesis-fusion ensembles and learned routers that do not use the FASTT objective. Every learned baseline is tuned with the same search budget as the FASTT arms (Table 2).

- Best-Single-Model, the member with the lowest WER on the training fold, applied to every test clip. This is the static comparator every adaptive method has to beat.
- Random-Ensemble, which selects a member uniformly at random for each utterance (a seeded realization, so that it passes through the same paired test as every other arm).
- ROVER (weighted) (Fiscus, 1997), which aligns the

hypotheses of all L members and votes; the vote weighting (uniform, inverse-WER or log-odds) and the null-arc penalty are tuned.

- ROVER (confidence-weighted), which additionally weights each member's votes by its word-level posterior confidence, with tuned smoothing.
- CN-MBR, confusion-network minimum-Bayes-risk combination (Mangu et al., 2000; Goel & Byrne, 2000), which builds a confusion network over the members' hypotheses and either decodes it slot by slot or returns the candidate hypothesis of minimum expected word error under the network; the system prior, the slot temperature, the null-arc prior and an optional per-utterance confidence weighting (estimated from the encoder features by a ridge probe) are tuned.
- Score-Fusion, which scores every member's hypothesis by a tuned mixture of probe confidence, cross-member consensus and length, and either selects the top-scoring member (hard mode) or votes the aligned network with the softmax of the scores (soft mode).
- Hard-Selector (plain CE), an argmin classifier trained with cross-entropy against the per-utterance best member on exactly the same features and feature transformation as FASTT, i.e., FASTT with the expected-WER objective replaced by classification.
- Hard-Selector (cost-sensitive CE), the same classifier with cross-entropy weighted by the WER a wrong choice would incur.
- QE-Router, a quality-estimation router (Kastanos et al., 2020; Malinin & Gales, 2021) that regresses each member's error count from the features and selects the member with the lowest predicted error.
- Oracle, which selects the member with the lowest WER on each utterance and is the lower bound on the achievable WER. Because the members frequently differ by a single word, we also list the oracle at a tolerance of $k \in \{1, 2\}$ words, which keeps the best single model unless another member saves more than k words on the utterance.

5.2.3. Classical Feature Selection and Raw Baselines

To demonstrate the advantage of learned, task-aligned feature transformations within the FASTT framework, we evaluate four baselines representing traditional approaches. These methods either use the full feature set directly or apply static feature selection as a fixed preprocessing step:

- Raw-SDT / Raw-XGBoost: These baselines use the complete, high-dimensional feature set without any prior dimensionality reduction or selection. This allows us to evaluate the selector models, Soft Decision Trees and XGBoost, in their vanilla form. Raw-XGBoost trained with the expected-WER objective is exactly the AdaSTT selector of Namli et al. (2026), so the comparison with FASTT-Alternating isolates the contribution of the learned transformation over the prior state of the art.
- SelectKBest-SDT / SelectKBest-XGBoost: To assess the impact of standard dimensionality reduction, we apply a univariate statistical filter, namely ANOVA F-value ranking, to select the top k most informative features as a preprocessing step, with k tuned. The selector models are then trained exclusively on this reduced subset.

All of these baselines are optimized with the same expected-WER objective. This isolates the impact of feature processing, contrasting raw input and static feature selection against the learned, task-aligned transformations of the FASTT variants.

5.2.4. FASTT Variants

We evaluate two representative variants within the FASTT framework, corresponding to the differentiable and non-differentiable optimization paths described in 3.5:

- FASTT-Boosted (differentiable path): the boosted feature-adaptive selector of Algorithm 2, with soft decision trees as the base model, trained end-to-end with per-round feature adaptation.
- FASTT-Alternating (non-differentiable path): an XGBoost selector with a learned feature transformation, trained with the alternating procedure of Algorithm 3.

For both variants the feature transformation (diagonal, linear, low-rank or nonlinear) is a hyperparameter selected on the hold-out partition together with the other hyperparameters; the transformations are compared individually in the ablation of Section 5.4.3.

5.3. Training Setup and Evaluation

Each corpus is split once, with a fixed seed, into a 25% hyperparameter hold-out and a 75% evaluation partition (Table 1). Hyperparameters are selected on the hold-out partition only: every learned arm, including every baseline, draws the same number of configurations uniformly at random from its search space (Table 2), using Optuna's random sampler (Akiba et al., 2019) under the same seed; each configuration is fitted on 70% of the hold-out and scored on the remaining 30%, and the configuration with the lowest hold-out WER is frozen. The frozen configuration is then evaluated on the evaluation partition with 5×2 cross-validation (Dietterich, 1998): five random halvings of the partition, each used once in either direction, so that every arm is trained and tested on the same ten folds and no test clip is ever seen during hyperparameter selection or training. All numbers we report are means over these ten folds.

The primary evaluation metric is the corpus WER of the selected member on the test fold,

$$\mathrm{WER}_{\mathrm{method}} = \frac{\sum_{i \in \mathcal{D}_{\mathrm{test}}} e\big(y_{i,\,j_i},\, y_i^{\mathrm{ref}}\big)}{\sum_{i \in \mathcal{D}_{\mathrm{test}}} |y_i^{\mathrm{ref}}|}, \qquad (8)$$

where j_i is the member selected by the evaluated method and $e(\cdot,\cdot)$ is the number of word errors, so that every reference word counts once. To show how much of the achievable improvement a method captures we also report the share of the oracle gap closed, $\mathrm{GC} = (\mathrm{WER}_{\mathrm{BSM}} - \mathrm{WER}_{\mathrm{method}})/(\mathrm{WER}_{\mathrm{BSM}} - \mathrm{WER}_{\mathrm{oracle}})$, where BSM is the best single model, and the selection accuracy Acc, the share of test utterances routed to a member with the lowest WER on that utterance. GC is the gap-closed statistic with which per-instance selectors are benchmarked between the single best and the virtual best solver (Bischl et al., 2016; Kotthoff, 2014); a negative value means the method falls below the best single model, and the Acc of the BSM row is the share of utterances on which the pool has nothing to select. Statistical significance is assessed with the combined 5×2 cv F test of Alpaydın (1999), which pairs the two arms fold by fold; on every corpus each arm is tested against the better of the two FASTT variants on that corpus, which we call the corpus's FASTT anchor.

5.4. Results and Discussion

5.4.1. Transcription Performance

Table 3 reports, for every corpus, the test WER, the share of the oracle gap closed (GC) and the selection accuracy (Acc), averaged over the ten 5×2 folds, and Table 4 the paired 5×2 cv F statistic and p-value of every arm against the corpus's FASTT anchor. Against the best single model, the static reference, FASTT is significantly better on four of the five corpora: VoxPopuli ($0.1761 \rightarrow 0.1733$, $p = 0.007$), AMI-SDM ($0.6528 \rightarrow 0.6221$, $p < 0.001$), Earnings-22 ($0.4103 \rightarrow 0.3893$, $p < 0.001$) and People's Speech ($0.4011 \rightarrow 0.3860$, $p < 0.001$). In terms of the headroom that exists, FASTT recovers 29–32% of the distance from the best single model to the per-utterance oracle on the three acoustically hard corpora (AMI-SDM 32.4%, Earnings-22 32.2%, People's Speech 29.0%) and 10.5% on VoxPopuli. Much of the remaining headroom consists of utterances on which the members differ by a single word (see the tolerance oracles in the table), where a selector operating on encoder features has little to discriminate on.

Two properties of Table 3 matter more than any single cell. The first is regret. An adaptive policy is deployable only if it does not lose to the static choice it replaces, and the arms trained with the expected-WER objective are the only learned arms that never fall below the best single model on any corpus, the worst case being a statistical tie on LibriSpeech; every hypothesis-fusion ensemble, both argmin classifiers and the quality-estimation router lose to it on at least one corpus (the negative GC entries). The second is that the gain follows the diversity of the pool. The best single model is itself optimal on 88% of the utterances of LibriSpeech, on 64–66% on VoxPopuli and AMI-SDM and on 53% on Earnings-22 and People's Speech, and the share of the gap FASTT closes is 1.5%, 10.5% and 32.4%, and 32.2% and 29.0%, in the

Table 3: Test corpus WER, share of the oracle gap closed (GC, %), and selection accuracy (Acc, %), 5×2 cv mean over the 10 folds. Bold: best per dataset and metric among the competing arms (oracles excluded). †: the corpus's FASTT anchor (the better FASTT arm) is significantly better at $p < 0.05$ (paired 5×2 cv F test). Fusion arms produce a new hypothesis and have no accuracy.

	LibriSpeech			**VoxPopuli**			**AMI (SDM)**			**Earnings-22**			**People's Speech**		
Method	**WER**	**GC**	**Acc**	**WER**	**GC**	**Acc**	**WER**	**GC**	**Acc**	**WER**	**GC**	**Acc**	**WER**	**GC**	**Acc**
Reference															
Best-Single-Model	0.0226	0.0	**88.1**	0.1761†	0.0	63.8	0.6528†	0.0	66.4	0.4103†	0.0	53.2	0.4011†	0.0	53.0
External baselines															
Random-Ensemble	0.0251†	-29.5	86.2	0.1967†	-76.9	50.3	0.6710†	-19.3	61.2	0.4242†	-21.4	46.8	0.4102†	-17.3	49.3
ROVER (weighted)	**0.0214**	**13.8**	—	0.1737	9.1	—	0.6667†	-14.7	—	0.4151†	-7.3	—	0.3894†	22.4	—
ROVER (confidence-weighted)	0.0221	4.6	—	0.1840†	-29.5	—	0.6683†	-16.4	—	0.4241†	-21.2	—	0.3979†	6.3	—
CN-MBR	0.0230	-5.7	**88.1**	0.1925†	-61.3	54.9	0.6680†	-16.2	57.4	0.4295†	-29.5	47.1	0.4049†	-7.3	51.5
Score-Fusion	0.0303†	-90.9	82.8	0.2066†	-114.2	45.0	0.6796†	-28.4	60.0	0.4280†	-27.2	44.9	0.4247†	-45.3	45.6
Hard-Selector (plain CE)	0.0232†	-7.6	87.1	0.1737	8.9	66.0	0.6284†	25.9	71.3	0.3971†	20.3	57.2	0.3930†	15.7	55.4
Hard-Selector (cost-sensitive CE)	0.0233†	-8.8	86.9	0.1761†	0.0	63.8	0.6392†	14.4	69.3	0.4085†	2.8	55.2	0.3992†	3.8	53.1
QE-Router	0.0237	-13.7	86.9	0.1795†	-12.6	60.2	0.6272†	27.1	70.6	0.4056†	7.2	54.1	0.3924†	16.9	57.5
Classical feature selection															
Raw XGBoost (AdaSTT)	0.0224	1.7	**88.1**	0.1745	6.2	65.3	0.6274†	26.9	72.2	0.3926†	27.2	60.2	0.3879	25.5	60.6
Raw SDT	0.0226	0.0	**88.1**	0.1761†	0.0	63.8	0.6528†	0.0	66.4	0.4096†	1.1	54.7	0.4011†	0.0	53.0
SelectKBest XGBoost	0.0225	0.3	88.0	0.1751	3.8	64.7	0.6343†	19.5	70.4	0.3975†	19.6	58.8	0.3933†	15.0	58.2
SelectKBest SDT	0.0226	0.0	**88.1**	0.1761†	0.0	63.8	0.6528†	0.0	66.4	0.4096†	1.1	54.7	0.4011†	0.0	53.0
FASTT															
FASTT-Boosted	0.0224	1.5	88.0	0.1735	10.0	65.9	**0.6221**	**32.4**	**73.3**	0.3901	31.0	60.9	**0.3860**	**29.0**	**61.9**
FASTT-Alternating	0.0225	0.1	87.9	**0.1733**	**10.5**	**66.3**	0.6240	30.5	72.9	**0.3893**	**32.2**	**61.7**	0.3880†	25.3	60.8
Oracles															
Oracle (no tolerance)	0.0140	100.0	100.0	0.1494	100.0	100.0	0.5584	100.0	100.0	0.3452	100.0	100.0	0.3491	100.0	100.0
Oracle (1 word tolerance)	0.0184	48.9	91.6	0.1566	73.1	81.7	0.5790	78.2	84.4	0.3561	83.3	80.5	0.3602	78.7	77.7
Oracle (2 word tolerance)	0.0214	13.5	88.7	0.1647	42.6	71.3	0.6017	54.1	75.7	0.3695	62.7	68.4	0.3730	54.1	64.8

same order. Balance is not the only factor, since the members of VoxPopuli disagree less consequentially than those of AMI-SDM (the two-word-tolerance oracle closes 43% against 54% of the respective gaps), but the direction is consistent: where one expert dominates there is little to select, and where the experts are complementary the selector recovers a third of what an oracle could. The identity of the best single model is not stable either: it is the Conformer–RNN-T on VoxPopuli and AMI-SDM, data2vec-audio on People's Speech, Canary-Qwen on LibriSpeech, and on Earnings-22 the validation-selected choice alternates between the Conformer–RNN-T and wav2vec2-large across the ten folds, the two differing by 10^{-4} in corpus WER. A static policy is a bet that the ranking observed on the labelled sample holds in the field; the per-utterance policy hedges that bet, and on this evidence at no cost.

On LibriSpeech the members are general-purpose models for which read speech is in domain: the best single model is already the best member on 88.1% of the utterances and the whole oracle gap is 0.009 WER wide. Every selection arm learns to return the best single model on most utterances (Acc 88.0 vs. 88.1 for FASTT-Boosted), and FASTT and the best single model are statistically indistinguishable (0.0224 vs. 0.0226, $p = 0.737$). This is also the one corpus on which hypothesis fusion is preferable to selection: weighted ROVER (0.0214) is significantly better than FASTT ($p = 0.025$), profiting from three members that are all strong and within 0.7 WER points of each other, which is the regime in which voting is expected to work. We report the corpus because it makes the boundary of the method explicit: adaptive selection cannot help where one member dominates, and FASTT degrades gracefully to the static choice there instead of hurting it.

The hypothesis-fusion ensembles are not competitive on the out-of-domain corpora. Weighted ROVER, confidence-weighted ROVER, CN-MBR and score fu-

Table 4: Paired 5×2 cv test against the corpus's FASTT anchor on WER: the Alpaydın combined F statistic and its p-value.

	LibriSpeech		**VoxPopuli**		**AMI (SDM)**		**Earnings-22**		**People's Speech**	
Comparator	F	p	F	p	F	p	F	p	F	p
Best-Single-Model	0.65	0.737	11.69	0.007	338.55	<0.001	246.16	<0.001	110.88	<0.001
Random-Ensemble	15.93	0.003	917.77	<0.001	2937.72	<0.001	298.60	<0.001	2214.11	<0.001
ROVER (weighted)	6.60	0.025	1.40	0.373	24.94	0.001	28.17	<0.001	6.10	0.030
ROVER (confidence-weighted)	1.59	0.317	187.91	<0.001	441.08	<0.001	222.63	<0.001	66.14	<0.001
CN-MBR	3.97	0.071	370.64	<0.001	2219.25	<0.001	136.43	<0.001	169.00	<0.001
Score-Fusion	59.59	<0.001	331.28	<0.001	1642.88	<0.001	281.29	<0.001	962.80	<0.001
Hard-Selector (plain CE)	17.21	0.003	0.87	0.604	15.10	0.004	36.18	<0.001	30.00	<0.001
Hard-Selector (cost-sensitive CE)	33.93	<0.001	11.69	0.007	17.37	0.003	68.66	<0.001	10.67	0.009
QE-Router	3.86	0.075	15.82	0.004	20.77	0.002	30.77	<0.001	85.62	<0.001
Raw XGBoost (AdaSTT)	0.58	0.782	2.55	0.156	8.42	0.015	27.79	<0.001	3.69	0.081
Raw SDT	0.65	0.737	11.69	0.007	338.55	<0.001	168.06	<0.001	110.88	<0.001
SelectKBest XGBoost	0.65	0.739	4.51	0.055	46.78	<0.001	6.38	0.027	20.72	0.002
SelectKBest SDT	0.65	0.737	11.69	0.007	338.55	<0.001	168.06	<0.001	110.88	<0.001
FASTT-Boosted	-	-	1.28	0.415	-	-	1.41	0.371	-	-
FASTT-Alternating	0.66	0.730	-	-	2.85	0.130	-	-	8.16	0.016
Oracle (no tolerance)	444.73	<0.001	543.47	<0.001	3428.33	<0.001	1687.26	<0.001	3267.64	<0.001
Oracle (1 word tolerance)	133.35	<0.001	152.51	<0.001	3371.26	<0.001	1033.10	<0.001	1765.45	<0.001
Oracle (2 word tolerance)	3.98	0.070	56.24	<0.001	486.88	<0.001	306.82	<0.001	763.69	<0.001

sion all fall *below* the best single model on AMI-SDM and Earnings-22, and only weighted ROVER improves on it on both People's Speech (0.3894 vs. 0.4011) and VoxPopuli (0.1737 vs. 0.1761); FASTT is significantly better than every fusion arm on every out-of-domain corpus except weighted ROVER on VoxPopuli, where the two are statistically tied ($p = 0.373$). With three members of unequal quality the vote is decided by the two weaker members whenever they agree, which on distant-microphone and accented speech happens often enough to erase the gain from the utterances where fusion helps. Selection avoids this failure mode and also avoids decoding every member (Section 5.4.2).

The learned routers separate the contribution of the expected-WER objective from that of the features. The Hard-Selector trained with plain cross-entropy on the same features and transformation as FASTT is significantly worse on four of the five corpora (all but VoxPopuli, where the two tie at 0.1737 vs. 0.1733) and falls below the best single model on LibriSpeech (0.0232 vs. 0.0226): the argmin label carries no information about how much a wrong choice costs, so the classifier chases it on utterances where the members tie and pays for it where they do not. The cost-sensitive variant avoids the worst errors only by collapsing to a single member on three corpora, and recovers at most 14% of the headroom on any corpus. The QE-Router, which regresses every member's error count and routes to the smallest estimate, is the strongest of the three and is still significantly behind FASTT on the four out-of-domain corpora ($p \leq 0.004$). The ablation in Section 5.4.3 repeats this comparison within a single architecture.

Within the selector family, the raw XGBoost selector of AdaSTT (Namli et al., 2026) is the strongest baseline. FASTT improves on it by 0.5 WER points on AMI-SDM ($p = 0.015$) and 0.3 on Earnings-22 ($p < 0.001$); on VoxPopuli and People's Speech the improvement (0.1–0.2 points) is not significant at the 5% level ($p = 0.156$ and 0.081), and on LibriSpeech the two coincide. SelectKBest filtering is consistently worse than either: relative to FASTT, SelectKBest XGBoost loses 1.2 WER points on AMI-SDM, 0.8 on Earnings-22 and 0.7 on People's Speech ($p \leq 0.027$), and 0.2 on VoxPopuli ($p = 0.055$), because a univariate filter discards the interactions between em-

Table 5: Selector overhead (VoxPopuli, 32 clips, batch size 1, greedy decoding): time per utterance to encode all members, decode the chosen member(s), and run the selector; end-to-end is their sum. Train: fitting the selector on one 5×2 fold. ROVER decodes every member before voting. CN-MBR and Score-Fusion instead pick a single member's decode from confidence/score signals computed before decoding, like a selector, which is why their decode cost is far below ROVER's rather than the sum of all three.

Method	**encode (ms)**	**decode (ms)**	**selector (μs)**	**end-to-end (ms)**	**train selector (s)**
Members					
ESPnet Conformer-RNN-T	10.1	35.8	—	45.9	—
ESPnet Transformer	7.0	0.2	—	7.1	—
data2vec-audio-base	19.5	0.3	—	19.8	—
Reference					
Best-Single-Model	36.6	35.8	—	72.5	—
External baselines					
Random-Ensemble	36.6	12.1	—	48.8	—
ROVER (weighted)	36.6	36.3	225.6	73.1	0.00
ROVER (confidence-weighted)	36.6	36.3	42.9	73.0	0.00
CN-MBR	36.6	20.1	46.4	56.7	2.63
Score-Fusion	36.6	11.6	22.9	48.2	0.00
Hard-Selector (plain CE)	36.6	29.6	0.5	66.2	0.09
Hard-Selector (cost-sensitive CE)	36.6	35.8	0.5	72.5	0.03
QE-Router	36.6	15.3	0.7	51.9	2.63
Classical feature selection					
Raw XGBoost (AdaSTT)	36.6	21.8	5.8	58.5	10.52
Raw SDT	36.6	35.8	1.3	72.5	7.54
SelectKBest XGBoost	36.6	26.4	5.7	63.0	5.24
SelectKBest SDT	36.6	35.8	0.5	72.5	1.64
FASTT					
FASTT-Boosted	36.6	12.8	1.5	49.4	3.97
FASTT-Alternating	36.6	17.5	9.7	54.1	53.31
Oracles					
Oracle (no tolerance)	36.6	23.0	—	59.6	—
Oracle (1 word tolerance)	36.6	29.5	—	66.1	—
Oracle (2 word tolerance)	36.6	33.2	—	69.8	—

bedding dimensions that the selector relies on. The soft-decision-tree path is more sensitive to its configuration: under the same search budget, the Raw-SDT and SelectKBest-SDT arms settle on configurations that return a single member on every corpus, whereas the FASTT-Boosted search, which ranges over the transformation family with the identity as one of its values, finds a routing configuration on every corpus and is the best arm on AMI-SDM and People's Speech. The two FASTT variants are statistically indistinguishable on four corpora and differ only on People's Speech ($p = 0.016$).

5.4.2. Computational Complexity

We measure the online cost of every arm end to end on 32 VoxPopuli clips at batch size 1 on a single NVIDIA RTX PRO 6000 Blackwell Server Edition GPU, separating the three components each system pays per utterance: the encoder forward passes of the members, which produce both the features the selector reads and the states the decoders consume; the selector; and the decoder(s). Table 5 reports the three components and their sum under greedy decoding, together with the time to fit each selector on one 5×2 fold. Feature extraction is a fixed cost: encoding all three members takes 36.6 ms per utterance, comparable to the most expensive decoder, and every adaptive method pays it. The selector itself is negligible at

Table 6: Cost per utterance of the two ensemble strategies (VoxPopuli, 32 clips, batch size 1). Every strategy encodes all members once; a fusion system then decodes every member, FASTT decodes only the selected one. 'saved' is the fraction of the decode-all time FASTT avoids; the floor always decodes the cheapest member.

System	encode (ms)	decode (ms)	selector (μs)	total (ms)	× decode-all	saved
Decoding: greedy						
ESPnet Conformer-RNN-T	10.1	35.8	—	45.9	0.63	—
ESPnet Transformer	7.0	0.2	—	7.1	0.10	—
data2vec-audio-base	19.5	0.3	—	19.8	0.27	—
Decode all, then choose (fusion)	36.6	36.3	1.5	72.9	1.00	—
Choose from encoders, decode one (FASTT)	36.6	12.8	1.5	**49.4**	0.68	32%
Cheapest decoder only (floor)	36.6	0.2	1.5	36.8	0.50	—
Decoding: beam 5						
ESPnet Conformer-RNN-T	8.4	306.1	—	314.5	0.33	—
ESPnet Transformer	6.3	614.0	—	620.3	0.65	—
data2vec-audio-base	6.5	7.4	—	13.8	0.01	—
Decode all, then choose (fusion)	21.2	927.5	1.5	948.7	1.00	—
Choose from encoders, decode one (FASTT)	21.2	112.2	1.5	**133.4**	0.14	86%
Cheapest decoder only (floor)	21.2	7.4	1.5	28.6	0.03	—
Decoding: beam 20						
ESPnet Conformer-RNN-T	8.8	3196.0	—	3204.8	0.82	—
ESPnet Transformer	6.4	686.9	—	693.3	0.18	—
data2vec-audio-base	6.5	23.4	—	29.8	0.01	—
Decode all, then choose (fusion)	21.7	3906.2	1.5	3927.9	1.00	—
Choose from encoders, decode one (FASTT)	21.7	1136.4	1.5	**1158.0**	0.29	70%
Cheapest decoder only (floor)	21.7	23.4	1.5	45.0	0.01	—

Table 7: Per-member encoder and decoder time per utterance (VoxPopuli, 32 clips, batch size 1) at three decoding settings, with the WER on the timed clips. The encoder is run once; the decoder's share grows with the beam.

		decode (ms)			WER		
Member	encode (ms)	greedy	beam 5	beam 20	greedy	beam 5	beam 20
ESPnet Conformer-RNN-T	9.1	35.8	306.1	3196.0	0.1993	0.1931	0.1907
ESPnet Transformer	6.6	0.2	614.0	686.9	0.2509	0.2140	0.2165
data2vec-audio-base	10.8	0.3	7.4	23.4	0.2042	0.2054	0.2042

0.5–10 μs for the learned arms and 23–226 μs for the fusion arms. Fitting the selector is an offline, one-time cost: on one fold of this corpus FASTT-Boosted trains in 4 s and FASTT-Alternating, whose alternating loop refits the trees at every iteration, in 53 s (last column of Table 5). What distinguishes the methods is the decoding cost: a fusion ensemble must decode all three members (36.3 ms), FASTT decodes only the selected one (12.8 ms for FASTT-Boosted, 17.5 ms for FASTT-Alternating), and the best single model always decodes the most expensive member (35.8 ms). End to end, FASTT-Boosted is 32% faster than the best single model and than weighted ROVER on this corpus (49.4 vs. 72.5 and 73.1 ms), while FASTT is significantly more accurate than the best single model and statistically tied with weighted ROVER on this corpus (Table 4).

Table 6 isolates the decoding cost at three beam widths, because the decoder's share of the latency grows with the beam: a beam of 5 lowers the WER of the two ESPnet members by 0.6–3.7 points on the timed clips but multiplies their decoding time by 9–3,000 (Table 7), so at realistic decoding settings the decoders dominate. Because FASTT chooses from the encoder outputs before any decoder runs, it avoids two of the three decodes and saves 32% of the decode-all latency under greedy decoding, 86% at beam 5 (133 ms vs. 949 ms per utterance) and 70% at beam 20. A

Table 8: Ablation of the feature transformation: best validation corpus WER per transform in the hyperparameter exploration (same members, features and partition as the main search; 15–40 configurations per cell). Bold: best transform in the row. Underlined: the transform the main search selected for Table 3 under its budget of 10 configurations. The two differ where the budget landed on a transform the richer search ranks lower.

Method	Raw	Diag.	Linear	Low-rank	Nonlin.
VoxPopuli					
FASTT-Boosted	**0.1767**	0.1770	0.1770	0.1774	0.1773
FASTT-Alternating	0.1763	**0.1762**	0.1764	0.1774	0.1773
AMI (SDM)					
FASTT-Boosted	**0.6184**	0.6243	0.6230	0.6244	0.6219
FASTT-Alternating	0.6228	**0.6197**	0.6221	0.6218	0.6213
Earnings-22					
FASTT-Boosted	0.4002	0.4187	**0.3992**	0.3998	0.4003
FASTT-Alternating	0.4044	0.4025	**0.4010**	0.4016	0.4022
People's Speech					
FASTT-Boosted	**0.3908**	0.3925	0.3920	0.3935	0.3941
FASTT-Alternating	0.3912	0.3908	**0.3907**	0.3909	0.3916

Table 9: Ablation of the training objective on AMI (SDM): the same architecture, features, search budget, partition and 5×2 folds, trained with cross-entropy (CE) or the expected-WER objective (E[WER]) and evaluated on corpus WER; mean over the 10 folds, better in bold. Δ: CE minus E[WER] in WER points; p is Alpaydın's combined 5×2 cv F test. Every difference is significant at $p < 0.05$.

Architecture	CE	E[WER]	Δ	p (F)
XGBoost	0.6313	**0.6274**	+0.39	0.033
FASTT-Boosted	0.6280	**0.6221**	+0.59	0.006
FASTT-Alternating	0.6314	**0.6240**	+0.74	<0.001

decode-all ensemble, by contrast, pays every member's beam search whether or not the extra hypotheses are used. The lower bound in the same table, always decoding the cheapest member, shows the remaining cost of running the selected member rather than the cheapest one; FASTT sits between this floor and the decode-all cost in proportion to how often it selects the expensive members.

5.4.3. Ablation: Feature Transformation and Training Objective

Table 8 reports, for each corpus and FASTT variant, the best hold-out WER reached by each feature transformation in an extended hyperparameter exploration on the same members, features and partition as the main search, so that every cell is comparable with the main results. Two observations follow. First, no single transformation dominates. For FASTT-Alternating, diagonal gating is the best choice on VoxPopuli and AMI-SDM and the linear projection on Earnings-22 and People's Speech, improving on the raw features by up to 0.3 WER points; for FASTT-Boosted the linear projection is best on Earnings-22, whereas on the other three corpora the raw features are the best entry, the soft splits of the booster being themselves learned linear projections of the input. The differences among the transformations are within a few tenths of a WER point, comparable to the fold-to-fold standard deviation of the arms (0.2–0.4 points). We therefore treat the transformation as a hyperparameter, with the identity as one of its values, and let the search select it per corpus and variant rather than prescribing one; the underlined entries in the table show which transformation the main search selected under its budget of ten configurations, and where it differs from the best entry of the extended search the difference is the price of the smaller budget. Second, the transformation contributes more on the non-differentiable path, where XGBoost has no differentiable layer of its own with which to reweight its inputs, than on the

soft-decision-tree path.

Table 9 isolates the training objective on AMI-SDM, the corpus with the largest headroom. For each of three architectures (raw XGBoost, FASTT-Boosted and FASTT-Alternating) we train the identical model, features, search budget, partition and folds once with cross-entropy against the per-utterance best member and once with the expected-WER objective of Eq. 3. The expected-WER objective is better in all three cases, by 0.4 to 0.7 WER points, and every difference is significant ($p \leq 0.033$). Together with the Hard-Selector rows of Table 3, this shows that the gain of FASTT over an argmin classifier on the same features is attributable to the cost-sensitive objective and not only to the feature set or the tree model.

6. Conclusion

We presented FASTT, a feature-adaptive model selection framework for STT systems. Recognizing that no single STT expert is optimal across all acoustic conditions, our approach dynamically selects the most suitable model for each input utterance. To address the challenges posed by high-dimensional acoustic feature spaces and limited labeled data for training the selection algorithm, FASTT jointly optimizes a task-aligned feature transformation together with a lightweight selection network, directly targeting the minimization of a cumulative transcription-error objective. Across five English corpora and 5×2 cross-validation, FASTT is significantly better than the best single model on the four corpora that are out of domain for the members and recovers up to 32% of the per-utterance oracle headroom, whereas hypothesis-fusion ensembles fall below the best single model on most of them; it is also better than argmin classifiers and a quality-estimation router trained on the same features, significantly so in all but one of the out-of-domain comparisons, and the ablation attributes this margin to the expected-WER objective. Because the selection is made from the members' encoder outputs, only one decoder runs per utterance, which saves up to 86% of the decoding time of a decode-all ensemble at beam width 5. On LibriSpeech, where one member dominates, FASTT reduces to the static choice rather than degrading it. Its modular design allows flexible integration of various feature-transformation architectures, from diagonal gating to nonlinear bottlenecks, and multiple selection algorithms, including soft decision trees and gradient-boosted trees.

Four aspects delimit the scope of these results. First, every corpus is evaluated with a pool of three experts; the trios differ across corpora, but whether the margin over the best single model grows with the size of the pool has not been tested. Second, the selector is trained and evaluated within one acoustic domain at a time, so the method assumes a labelled sample of the deployment conditions; how a selector trained on one corpus transfers to another, and how its accuracy scales with the size of that sample, are not characterized here. Third, the encoder passes of all members are paid on every utterance. At realistic beam widths this fixed cost is small next to the decoding it saves, but a selector that reads a lighter front-end would trade some accuracy for it and is a natural extension. Finally, the convergence guarantees cover the selector for a fixed transformation; a guarantee for the alternating procedure remains open.

Appendix A. Convergence Analysis

Appendix A.1. Convergence of the XGBoost-Based Selector

Preconditions (A1–A3)..

A1. *Bounded cost.* Each of the L candidate STT models E_j produces a transcription $y_{i,j}$ for input $\mathbf{s}_i$, and the associated per-model cost is

$$c_{i,j} := \mathrm{WER}(y_{i,j}, y_i^{\mathrm{ref}}).$$

We assume two uniform bounds:

$$0 \leq c_{i,j} \leq B_{\max} := \max_{i,j} c_{i,j},$$
$$|c_{i,j} - c_{i,k}| \leq B_{\mathrm{rng}} := \max_{i} \max_{j,k} |c_{i,j} - c_{i,k}|.$$

The second derivative term is

$$h_{i,j} = p_{i,j}(1 - 2p_{i,j})(c_{i,j} - \mathcal{L}_i),$$

where $\mathcal{L}_i = \sum_k p_{i,k}\, c_{i,k}$ is a convex combination of the costs. Thus,

$$c_{i,j} - \mathcal{L}_i = \sum_{k \neq j} p_{i,k}(c_{i,j} - c_{i,k}),$$
$$|c_{i,j} - \mathcal{L}_i| \leq (1 - p_{i,j})\, B_{\text{rng}}.$$

Therefore,

$$|h_{i,j}| \leq B_{\text{rng}}\, p_{i,j}(1-p_{i,j})\, |1-2p_{i,j}| \leq \tfrac{\sqrt{3}}{18} B_{\text{rng}} < \tfrac{B_{\text{rng}}}{8}.$$

After clipping $h_{i,j}$ below by $\varepsilon > 0$, all diagonal entries of the approximate Hessian H^* lie in $[\varepsilon,\, B_{\text{rng}}/8]$, giving

$$\varepsilon I \preceq H^* \preceq \tfrac{B_{\text{rng}}}{8}\, I.$$

We also assume that the overall loss L is L_g-smooth, i.e., its gradient is Lipschitz with constant $L_g > 0$.

A2. *Weak-learner edge.* Let $z^{(t)} \in \mathbb{R}^{N \times K}$ be the prediction matrix at boosting iteration t, and write

$$g_t := \nabla_z L(z)\big|_{z=z^{(t)}} \in \mathbb{R}^{N \times K}.$$

Let f_t be the depth-≥ 1 regression tree obtained by least-squares fitting to the pseudo-targets $-g_t$, and define the update

$$z^{(t+1)} = z^{(t)} + \eta\, f_t,$$

with $\eta > 0$. We assume that f_t satisfies the edge condition

$$\langle -g_t, f_t \rangle \geq \rho\, \|g_t\|_2^2,$$

where $\|\cdot\|_2$ denotes the empirical ℓ_2 norm over (n, m) pairs, and the constant $\rho > 0$ depends on the tree depth and regularization (Schapire & Freund, 2012).

A3. *Quadratic fit bound.* The regularized least-squares fit in XGBoost satisfies

$$\|f_t\|_2^2 \leq \lambda^{-1} \|g_t\|_2^2,$$

where $\lambda > 0$ is the ℓ_2 regularization coefficient in the leaf-weight formula (Chen & Guestrin, 2016).

Descent lemma.. Let $L_t := L(z^{(t)})$. By the smoothness assumption in A1, $\nabla_z L$ is L_g-Lipschitz. Using a second-order Taylor expansion,

$$L_{t+1} \leq L_t - \eta\rho \|g_t\|_2^2 + \tfrac{1}{2}\eta^2 L_g \|f_t\|_2^2.$$

Substituting A3,

$$L_{t+1} \leq L_t - \eta \left(\rho - \tfrac{\eta L_g}{2\lambda}\right) \|g_t\|_2^2. \tag{A.1}$$

Define

$$\gamma := \eta\rho - \tfrac{\eta^2 L_g}{2\lambda}.$$

Choosing $\eta < \frac{2\rho\lambda}{L_g}$ ensures $\gamma > 0$ and hence

$$L_{t+1} \leq L_t - \gamma \|g_t\|_2^2.$$

Convergence rate.. Summing Eq. A.1 over t gives

$$\sum_{t=0}^{T-1} \|g_t\|_2^2 \leq \frac{L_0}{\gamma} \quad \Rightarrow \quad \min_{0 \leq t < T} \|g_t\|_2^2 \leq \frac{L_0}{\gamma T},$$

ensuring a convergence rate of $\mathcal{O}(1/T)$ toward a first-order stationary point. Since L is non-convex, this does not imply global optimality.

Appendix A.2. Convergence of the Neural Selector (MLP / FASTT-SDT)

Preconditions (A1–A3).. For $L(\theta)$ defined in Eq. 3:

A1. *Bounded loss:* $0 \leq L(\theta) \leq B := \max_{i,j} c_{i,j}$.

A2. *L_g-smooth gradient:*

$$\|\nabla L(\theta) - \nabla L(\theta')\| \leq L_g\, \|\theta - \theta'\|$$

for all θ, θ'.

A3. *Bounded stochastic-gradient variance:*

$$\mathbb{E}\big\|g_t - \nabla L(\theta_t)\big\|^2 \leq \sigma^2.$$

Let $L^\star := \inf_\theta L(\theta)$. Under A1–A3, the standard assumptions in the non-convex analyses of Bottou et al. (2018) and Ghadimi & Lan (2013) are satisfied.

Remark 6. *Our analysis assumes that $L(\theta)$ is L_g-smooth, which holds when using smooth activations such as GELU or Softplus. In practice, we train with AdamW (Loshchilov & Hutter, 2019) for efficiency, though the convergence proof is stated for stochastic gradient descent.*

One-step descent lemma.. For the full-batch update

$$\theta_{t+1} = \theta_t - \eta \nabla L(\theta_t),$$

with any fixed step size $0 < \eta \leq 1/L_g$,

$$L(\theta_{t+1}) \leq L(\theta_t) - \left(\eta - \tfrac{L_g}{2}\eta^2\right) \|\nabla L(\theta_t)\|^2. \quad \text{(A.2)}$$

Full-batch convergence.. Summing Eq. A.2 for $t = 0, \ldots, T-1$,

$$\min_{0 \leq t < T} \|\nabla L(\theta_t)\|^2 \leq \frac{L(\theta_0) - L^\star}{T\,\eta\big(1 - \frac{1}{2}\eta L_g\big)} = \mathcal{O}\big(T^{-1}\big). \quad \text{(A.3)}$$

Mini-batch SGD convergence.. With $\eta_t = \eta_0/\sqrt{t+1}$, $\eta_0 \leq 1/L_g$, and using A3,

$$\mathbb{E}\Big[\min_{0 \leq t < T} \|\nabla L(\theta_t)\|^2\Big] \leq \frac{2\big(L(\theta_0) - L^\star\big) + \eta_0 L_g \sigma^2 \log T}{\eta_0 \sqrt{T}} = \mathcal{O}\big(T^{-1/2}\big), \quad \text{(A.4)}$$

where σ^2 is the stochastic-gradient variance bound from A3. Because L is non-convex, Eq. A.3, Eq. A.4 guarantee convergence only to a first-order stationary point, which is sufficient for training neural selectors (Ghadimi & Lan, 2013).

Declaration of generative AI and AI-assisted technologies in the manuscript preparation process

During the preparation of this work, the author(s) used Claude (Anthropic) in order to improve the language and readability of the manuscript, assist with drafting and structuring text, support code and analysis used to generate tables and figures, and assist with background literature search. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the published article.

Acknowledgements

This work is supported by The Scientific and Technological Research Council of Turkiye (TUBITAK) 1515 Frontier R&D Laboratories Support Program for Turk Telekom 6G R&D Lab under project number 5249902, and by the TUBITAK 2210-A program.

References

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 2623–2631). https://doi.org/10/gf7mzz.

Alpaydın, E. (1999). Combined 5×2 cv f test for comparing supervised classification learning algorithms. *Neural Computation*, *11*, 1885–1892. https://doi.org/10/d3538r.

Baevski, A., Hsu, W.-N., Xu, Q., Babu, A., Gu, J., & Auli, M. (2022). data2vec: A general framework for self-supervised learning in speech, vision and language. In *International Conference on Machine Learning (ICML)* (pp. 1298–1312). https://doi.org/10/hcjf95.

Baevski, A., Zhou, H., Mohamed, A., & Auli, M. (2020). wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Conference on Neural Information Processing Systems (NeurIPS)* (pp. 12449–12460). https://proceedings.neurips.cc/paper/2020/file/92d1e1eb1cd6f9fba3227870bb6d7f07-Paper.pdf.

Bertsekas, D. P. (2016). *Nonlinear Programming*. (3rd ed.). Athena Scientific. http://www.athenasc.com/nonlinbook.html.

Bischl, B., Kerschke, P., Kotthoff, L., Lindauer, M., Malitsky, Y., Fréchette, A., Hoos, H., Hutter, F., Leyton-Brown, K., Tierney, K., & Vanschoren, J. (2016). ASlib: A benchmark library for algorithm selection. *Artificial Intelligence*, *237*, 41–58. https://doi.org/10/f8rs9t.

Bottou, L., Curtis, F. E., & Nocedal, J. (2018). Optimization methods for large-scale machine learning. *SIAM Review*, *60*, 223–311. https://doi.org/10/gdhpjv.

Carletta, J., Ashby, S., Bourban, S., Flynn, M., Guillemot, M., Hain, T., Kadlec, J., Karaiskos,

V., Kraaij, W., Kronenthal, M., Lathoud, G., Lincoln, M., Lisowska, A., McCowan, I., Post, W., Reidsma, D., & Wellner, P. (2006). The AMI meeting corpus: A pre-announcement. In *Lecture Notes in Computer Science* (pp. 28–39). https://doi.org/10/bf23qf.

Chen, S., Wang, C., Chen, Z., Wu, Y., Liu, S., Chen, Z., Li, J., Kanda, N., Yoshioka, T., Xiao, X., Wu, J., Zhou, L., Ren, S., Qian, Y., Qian, Y., Wu, J., Zeng, M., Yu, X., & Wei, F. (2022). WavLM: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing (JSTSP)*, *16*, 1505–1518. https://doi.org/10/gsqzxg.

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)* (pp. 785–794). https://doi.org/10/gdp84q.

Cornell, S., Boeddeker, C., Park, T., Huang, H., Raj, D., Wiesner, M., Masuyama, Y., Chang, X., Wang, Z.-Q., Squartini, S., Garcia, P., & Watanabe, S. (2026). Recent trends in distant conversational speech recognition: A review of CHiME-7 and 8 DASR challenges. *Computer Speech & Language*, *97*, 101901. https://doi.org/10/hcjf99.

Davis, S. B., & Mermelstein, P. (1990). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. In *Readings in Speech Recognition* (pp. 65–74). https://doi.org/10/hbsppq.

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, *10*, 1895–1923. https://doi.org/10/fqc9w5.

Dong, L., Xu, S., & Xu, B. (2018). Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 5884–5888). https://doi.org/10/gg6672.

Escalona, M. M., Campos-Roca, Y., & Sánchez, C. J. P. (2025). Enhancing noise robustness of automatic parkinson's disease detection in diadochokinesis tests using multicondition training. *Expert Systems with Applications*, *260*, 125401. https://doi.org/10/g8r8gs.

Eyben, F., Scherer, K. R., Schuller, B. W., Sundberg, J., André, E., Busso, C., Devillers, L. Y., Epps, J., Laukka, P., Narayanan, S. S., & Truong, K. P. (2016). The geneva minimalistic acoustic parameter set (GeMAPS) for voice research and affective computing. *IEEE Transactions on Affective Computing (TAFFC)*, *7*, 190–202. https://doi.org/10/f3sfxq.

Fan, C., Yi, J., Tao, J., Tian, Z., Liu, B., & Wen, Z. (2021). Gated recurrent fusion with joint training framework for robust end-to-end speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, *29*, 198–209. https://doi.org/10/gmxf39.

Fedus, W., Zoph, B., & Shazeer, N. (2022). Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research (JMLR)*, *23*, 1–39. https://jmlr.org/papers/v23/21-0998.html.

Fiscus, J. G. (1997). A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER). In *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)* (pp. 347–354). https://doi.org/10/cg5c35.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, *29*, 1189–1232. https://doi.org/10/fbgj35.

Furui, S. (1986). Speaker-independent isolated word recognition using dynamic features of speech spectrum. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *34*, 52–59. https://doi.org/10/fb24x3.

Galvez, D., Diamos, G., Ciro, J., Cerón, J. F., Achorn, K., Gopi, A., Kanter, D., Lam, M., Mazumder, M., & Reddi, V. J. (2021). The people's speech: A large-scale diverse english speech recognition dataset for commercial usage. In *Conference on Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*. https://doi.org/10/hcjf94.

Gamper, H., Emmanouilidou, D., Braun, S., & Tashev, I. J. (2020). Predicting word error rate for reverberant speech. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 491–495). https://doi.org/10/hcjf89.

Gan, W., Ning, Z., Qi, Z., & Yu, P. S. (2026). Mixture of experts (MoE): A big data perspective. *Information Fusion*, *127*, 103664. https://doi.org/10/hb7mtq.

Ghadimi, S., & Lan, G. (2013). Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, *23*, 2341–2368. https://doi.org/10/ggmm89.

Ghorbani, S., & Hansen, J. H. L. (2023). Domain expansion for end-to-end speech recognition: Applications for accent/dialect speech. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, *31*, 762–774. https://doi.org/10/hcjf9j.

Gitman, I., Lavrukhin, V., Laptev, A., & Ginsburg, B. (2023). Confidence-based ensembles of end-to-end speech recognition models. In *Annual Conference of the International Speech Communication Association (Interspeech)* (pp. 1414–1418). https://doi.org/10/hcjf9n.

Goel, V., & Byrne, W. (2000). Minimum bayes-risk automatic speech recognition. *Computer Speech & Language*, *14*, 115–135. https://doi.org/10/ddg4vt.

Goel, V., Byrne, W., & Khudanpur, S. (2004). Segmental minimum bayes-risk decoding for automatic speech recognition. *IEEE Transactions on Speech and Audio Processing*, *12*, 234–249. https://doi.org/10/b3hw5v.

Graves, A. (2012). Sequence transduction with recurrent neural networks. https://doi.org/10/qvvn.

Gulati, A., Qin, J., Chiu, C.-C., Parmar, N., Zhang, Y., Yu, J., Han, W., Wang, S., Zhang, Z., Wu, Y., & Pang, R. (2020). Conformer: Convolution-augmented transformer for speech recognition. In *Annual Conference of the International Speech Communication Association (Interspeech)* (pp. 5036–5040). https://doi.org/10/gjp97z.

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research (JMLR)*, *3*, 1157–1182. https://jmlr.org/papers/volume3/guyon03a/guyon03a.pdf.

Hoffmeister, B., Klein, T., Schlüter, R., & Ney, H. (2006). Frame based system combination and a comparison with weighted ROVER and CNC. In *Annual Conference of the International Speech Communication Association (Interspeech)*. https://doi.org/10/hcjgbb.

Hsu, W.-N., Bolte, B., Tsai, Y.-H. H., Lakhotia, K., Salakhutdinov, R., & Mohamed, A. (2021). HuBERT: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, *29*, 3451–3460. https://doi.org/10/grgns2.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation*, *3*, 79–87. https://doi.org/10/cnsnqg.

Kano, T., Ogawa, A., Delcroix, M., & Watanabe, S. (2022). Integrating multiple ASR systems into an NLP backend with attention fusion. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 6237–6241). https://doi.org/10/hcjf9q.

Kansizoglou, I., Bampis, L., & Gasteratos, A. (2022). Deep feature space: A geometrical perspective. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *44*, 6823–6838. https://doi.org/10/grpbxw.

Karaca, H., & Kozat, S. S. (2026). Soft gradient boosting with learnable feature transforms for sequential regression. *IEEE Signal Processing Letters (SPL)*, *33*, 186–190. https://doi.org/10/hbsppt.

Kastanos, A., Ragni, A., & Gales, M. J. F. (2020). Confidence estimation for black box automatic speech recognition systems using lattice recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 6329–6333). https://doi.org/10/hcjf9d.

Ko, T., Peddinti, V., Povey, D., & Khudanpur, S. (2015). Audio augmentation for speech recognition. In *Annual Conference of the International Speech Communication Association (Interspeech)* (pp. 3586–3589). https://doi.org/10/hbspph.

Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, *97*, 273–324. https://doi.org/10/fngm9z.

Kotthoff, L. (2014). Algorithm selection for combinatorial search problems: A survey. *AI Magazine*, *35*, 48–60. https://doi.org/10/gbf5zr.

Kwon, Y., & Chung, S.-W. (2023). MoLE : Mixture of language experts for multi-lingual automatic speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 1–5). https://doi.org/10/hcjf9s.

Latif, S., Zaidi, S. A. M., Cuayáhuitl, H., Shamshad, F., Shoukat, M., Usama, M., & Qadir, J. (2025). Transformers in speech processing: Overcoming challenges and paving the future. *Computer Science Review*, *58*, 100768. https://doi.org/10/hb2bcx.

Li, J., Deng, L., Gong, Y., & Haeb-Umbach, R. (2014). An overview of noise-robust automatic speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, *22*, 745–777. https://doi.org/10/gg8q3b.

Li, Q., Qiu, D., Zhang, Y., Li, B., He, Y., Woodland, P. C., Cao, L., & Strohman, T. (2021). Confidence estimation for attention-based sequence-to-sequence models for speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 6388–6392). https://doi.org/10/hcjf9v.

Li, Y., Zhao, Z., Klejch, O., Bell, P., & Lai, C. (2023). ASR and emotional speech: A word-level investigation of the mutual impact of speech and emotion recognition. In *Annual Conference of the International Speech Communication Association (Interspeech)* (pp. 1449–1453). https://doi.org/10/hbsppj.

Liu, X., Li, M., Chen, L., Wanigasekara, P., Ruan, W., Khan, H., Hamza, W., & Su, C. (2021). ASR n-best fusion nets. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 7618–7622). https://doi.org/10/hcjf9m.

Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*. https://doi.org/10/gqp96f.

Malinin, A., & Gales, M. (2021). Uncertainty estimation in autoregressive structured prediction. In *International Conference on Learning Representations (ICLR)*. https://doi.org/10/hcjf96.

Mangu, L., Brill, E., & Stolcke, A. (2000). Finding consensus in speech recognition: word error minimization and other applications of confusion networks. *Computer Speech & Language*, *14*, 373–400. https://doi.org/10/fbdvbv.

Mohamed, A., yi Lee, H., Borgholt, L., Havtorn, J. D., Edin, J., Igel, C., Kirchhoff, K., Li, S.-W., Livescu, K., Maaløe, L., Sainath, T. N., &

Watanabe, S. (2022). Self-supervised speech representation learning: A review. *IEEE Journal of Selected Topics in Signal Processing (JSTSP)*, *16*, 1179–1210. https://doi.org/10/grj5xq.

Namli, A. S., Celik, O. F., Yapagci, E., & Kozat, S. S. (2026). ADASTT: Adaptive speech-to-text algorithm selection via meta-learning. *IEEE Transactions on Audio, Speech and Language Processing (TASLP)*, *34*, 1787–1801. https://doi.org/10/hcb4nn.

NVIDIA (2025a). Canary-qwen 2.5b model card. (accessed: Dec. 2025). https://huggingface.co/nvidia/canary-qwen-2.5b.

NVIDIA (2025b). Parakeet-TDT 0.6b v2 model card. (accessed: Sep. 2026). https://huggingface.co/nvidia/parakeet-tdt-0.6b-v2.

Panayotov, V., Chen, G., Povey, D., & Khudanpur, S. (2015). Librispeech: An ASR corpus based on public domain audio books. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 5206–5210). https://doi.org/10/gfv84w.

Parada, P. P., Dobrowolska, A., Saravanan, K., & Ozay, M. (2022). pMCT: Patched multi-condition training for robust speech recognition. In *Annual Conference of the International Speech Communication Association (Interspeech)* (pp. 3779–3783). https://doi.org/10/hcjf9g.

Paraskevopoulos, G., Kouzelis, T., Rouvalis, G., Katsamanis, A., Katsouros, V., & Potamianos, A. (2024). Sample-efficient unsupervised domain adaptation of speech recognition systems: A case study for modern greek. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, *32*, 286–299. https://doi.org/10/hcjf9h.

Park, D. S., Chan, W., Zhang, Y., Chiu, C.-C., Zoph, B., Cubuk, E. D., & Le, Q. V. (2019). SpecAugment: A simple data augmentation method for automatic speech recognition. In *Annual Conference of the International Speech Communication Association (Interspeech)* (pp. 2613–2617). https://doi.org/10/ghbzt4.

Prabhavalkar, R., Hori, T., Sainath, T. N., Schlüter, R., & Watanabe, S. (2024). End-to-end speech recognition: A survey. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, *32*, 325–351. https://doi.org/10/gtffc6.

Qian, Y., Bi, M., Tan, T., & Yu, K. (2016). Very deep convolutional neural networks for noise robust speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, *24*, 2263–2276. https://doi.org/10/f9chkn.

Rabiner, L., & Juang, B. H. (1993). *Fundamentals of Speech Recognition*. Prentice-Hall. https://api.semanticscholar.org/CorpusID:268095015.

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., & Sutskever, I. (2023). Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning (ICML)* (pp. 28492–28518). https://doi.org/10/gt2rcg.

Ragni, A., Gales, M. J. F., Rose, O., Knill, K. M., Kastanos, A., Li, Q., & Ness, P. M. (2022). Increasing context for estimating confidence scores in automatic speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, *30*, 1319–1329. https://doi.org/10/hcjf9b.

Ravi, N., T, T. R., & Arora, V. (2024). TeLeS: Temporal lexeme similarity score to estimate confidence in end-to-end ASR. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, *32*, 4399–4408. https://doi.org/10/hcjf9w.

Rice, J. R. (1976). The algorithm selection problem. In *Advances in Computers* (pp. 65–118). https://doi.org/10/cqnhc2.

Río, M. D., Ha, P., McNamara, Q., Miller, C., & Chandra, S. (2022). Earnings-22: A practical

benchmark for accents in the wild. https://doi.org/10/hcjgbc.

Schapire, R. E., & Freund, Y. (2012). *Boosting: Foundations and Algorithms*. The MIT Press. https://doi.org/10/hbsppn.

Schuller, B., & Batliner, A. (2013). *Computational Paralinguistics: Emotion, Affect and Personality in Speech and Language Processing*. (1st ed.). Wiley. https://doi.org/10/gp79pd.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., & Dean, J. (2017). Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations (ICLR)*. https://doi.org/10/g95xv5.

Sohn, J., Kim, N. S., & Sung, W. (1999). A statistical model-based voice activity detection. *IEEE Signal Processing Letters (SPL)*, *6*, 1–3. https://doi.org/10/d64p3z.

Sun, S., Zhang, B., Xie, L., & Zhang, Y. (2017). An unsupervised deep domain adaptation approach for robust speech recognition. *Neurocomputing*, *257*, 79–87. Machine Learning and Signal Processing for Big Multimedia Analysis. https://doi.org/10/hbsppv.

Tripathi, A., Kim, J., Zhang, Q., Lu, H., & Sak, H. (2020). Transformer transducer: One model unifying streaming and non-streaming speech recognition. https://doi.org/10/qvv8.

Vanschoren, J. (2019). Meta-learning. In *Automated Machine Learning: Methods, Systems, Challenges* (pp. 35–61). https://doi.org/10/gtpv2k.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In *Conference on Neural Information Processing Systems (NeurIPS)*. https://doi.org/10/gpnmtv.

Vettoruzzo, A., Bouguelia, M.-R., Vanschoren, J., Rögnvaldsson, T., & Santosh, K. (2024). Advances and challenges in meta-learning: A technical review. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *46*, 4763–4779. https://doi.org/10/gtjm59.

Wang, C., Riviere, M., Lee, A., Wu, A., Talnikar, C., Haziza, D., Williamson, M., Pino, J., & Dupoux, E. (2021). VoxPopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation. In *Annual Meeting of the Association for Computational Linguistics (ACL)* (pp. 993–1003). https://doi.org/10/gm6jcn.

Watanabe, S., Hori, T., Karita, S., Hayashi, T., Nishitoba, J., Unno, Y., Soplin, N. E. Y., Heymann, J., Wiesner, M., Chen, N., Renduchintala, A., & Ochiai, T. (2018). ESPnet: End-to-end speech processing toolkit. In *Annual Conference of the International Speech Communication Association (Interspeech)* (pp. 2207–2211). https://doi.org/10/gprwbz.

Watanabe, S., Mandel, M., Barker, J., Vincent, E., Arora, A., Chang, X., Khudanpur, S., Manohar, V., Povey, D., Raj, D., Snyder, D., Subramanian, A. S., Trmal, J., Yair, B. B., Boeddeker, C., Ni, Z., Fujita, Y., Horiguchi, S., Kanda, N., Yoshioka, T., & Ryant, N. (2020). CHiME-6 challenge: Tackling multispeaker speech recognition for unsegmented recordings in everyday environments. In *International Workshop on Speech Processing in Everyday Environments (CHiME)* (pp. 1–7). https://doi.org/10/hcjf88.

Xu, L., Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2008). SATzilla: Portfolio-based algorithm selection for SAT. *Journal of Artificial Intelligence Research (JAIR)*, *32*, 565–606. https://doi.org/10/ggbr9s.

You, Z., Feng, S., Su, D., & Yu, D. (2021). SpeechMoE: Scaling to large acoustic models with dynamic routing mixture of experts. In *Annual Conference of the International Speech Communication Association (Interspeech)* (pp. 2077–2081). https://doi.org/10/hcb4pm.

You, Z., Feng, S., Su, D., & Yu, D. (2022). Speechmoe2: Mixture-of-experts model with improved routing. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 7217–7221). https://doi.org/10/hcb4nd.

Yu, D., & Deng, L. (2015). *Automatic Speech Recognition: A Deep Learning Approach*. Springer London. https://doi.org/10/qvvk.

Zezario, R. E., Fu, S.-W., Chen, F., Fuh, C.-S., Wang, H.-M., & Tsao, Y. (2023). Deep learning-based non-intrusive multi-objective speech assessment model with cross-domain features. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, *31*, 54–70. https://doi.org/10/hcjf9f.

Zhu, H., Cheng, G., Wang, J., Hou, W., Zhang, P., & Yan, Y. (2024). Boosting cross-domain speech recognition with self-supervision and latent feature clustering. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, *32*, 471–485. https://doi.org/10/hcjf87.

Zhu, Q.-S., Zhang, J., Zhang, Z., & Dai, L. (2023). A joint speech enhancement and self-supervised representation learning framework for noise-robust speech recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, *31*, 1927–1939. https://doi.org/10/g977mx.