Posthoc Disentanglement of Textual and Acoustic Features in Self-Supervised Speech Encoders

Anonymous TACL submission

Abstract

Self-supervised speech encoders build internal representations that are highly entangled. This entanglement makes it difficult to track how high-level characteristics (e.g., pitch, loudness, speaker identity, or semantics of an utterance) rely on textual and acoustic information when used in downstream applications. In this paper, we explore an approach to posthoc disentanglement that builds on the Information Bottleneck principle. We take representations from pre-trained neural speech models as input, and separate them into two distinct components in a cascaded fashion: first we learn a vector encoding content (i.e., what can be transcribed as text), and nothing else; second, we learn a vector encoding all complementary acoustic features relevant to a downstream task. We apply and evaluate our setup to emotion recognition and speaker identification target tasks, quantifying the relative contribution of textual and acoustic features at each model layer. Finally, we use the disentangled representations for feature attribution, allowing us to identify the most salient speech frames from both the textual and acoustic perspectives.

1 Introduction

The internal activation vectors of most modern deep learning systems, including self-supervised speech models such as Wav2Vec2 (Baevski et al., 2020) and HuBERT (Hsu et al., 2021) are highly *entangled*. This means that high-level characteristics (e.g., pitch, loudness, speaker identity, emotional state, sentence type, semantics) of a spoken utterance are not separated into specialized dimensions within the model's latent space, but are instead mixed together. Entanglement is a major obstacle to our ability to interpret and intervene. Methods to yield representations where high-level characteristics are *disentangled* as much as possible, have therefore been a major focus of research (Bengio et al., 2013; van den Oord et al., 2017; Polyak et al., 2021; Choi et al., 2021, 2022; Cho et al., 2024).

Most state-of-the-art work (e.g., Qian et al., 2019, 2020; Qu et al., 2024; Ju et al., 2024), however, involves either applying adaptations to the architecture at train time, using large amounts of finetuning data, or performing architecture-specific postprocessing, all limiting the applicability of those approaches in one way or another. In this paper we explore, instead, how to best do *posthoc distentanglement* that works with pre-existing, pre-trained models (*architecture independence*), and with limited data and compute resources (*efficiency*). Moreover, while we will separate, like existing work, 'textual representations' and 'acoustic representations', we aim for an approach that allows for the nature of the acoustic representations to depend on their relevance for a target, downstream task. This *task dependence* is important, as defining acoustic dimensions in advance, as for instance based on pre-specified static components such as pitch and timbre in SpeechSplit (Qian et al., 2020), might result in losing task-relevant information.

Because many downstream speech applications use large pre-trained speech models as starting points, the ability to disentangle representations would be societally and scientifically relevant, because it paves the way for future work that could use disentangled representations for monitoring and controlling the information contained in internal representations of speech processing pipelines.

Through a series of experiments, we find that a setup based on the Information Bottleneck (Tishby et al., 2000) and cascaded disentanglement (instead of joint optimization) delivers the efficient, effective and generally-applicable approach that we are after. In section 2 we describe this setup in more detail, and in section 3 we report that with this

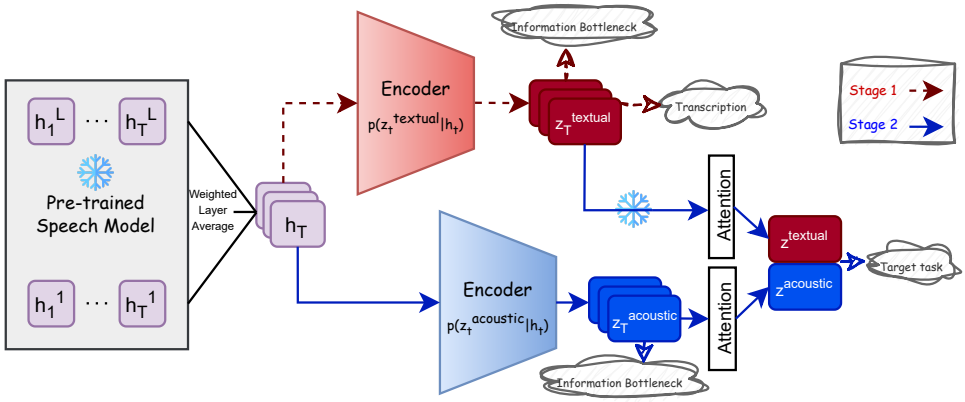

Figure 1: An overview of our cascaded setup: In stage 1, hidden states derived from a frozen pre-trained speech model are compressed using VIB to decode transcription. In stage 2, the hidden states are compressed to decode emotion, conditioned on the frozen textual latent representations learned in stage 1. This training procedure results in lightweight textual and acoustic encoders that disentangle speech frame representations into textual and task-relevant acoustic features.

setup we can successfully train a model on our target tasks (transcription, emotion classification and speaker identification). In section 4 we show we can quantify how well representations are disentangled, and that we obtain excellent scores; in this section, we also explicitly compare to related work. In the appendix we report two ablation experiments that show both the bottleneck and the cascading are crucial for these results. Finally, section 5 demonstrates how we can use the obtained disentangled representations for model interpretability.

2 Learning to Disentangle

2.1 Overview of the approach

We propose a cascaded, two-stage disentanglement setup (sketched in Figure 1) and apply it to disentangle textual and acoustic features.

In stage 1, we train a decoder with two objectives: to map the internal representation of an existing speech model to text, but also to minimize the presence of irrelevant information in these representations. This second objective is the Information Bottleneck (Tishby et al., 2000; Alemi et al., 2016); its purpose is to ensure that the latent representations preserve only the speech features necessary for accurate transcription, while filtering out any extraneous characteristics.

In stage 2, we train a second decoder on the same speech representations. This decoder also has access to the latent 'textual' representation learned in stage 1, and is again trained with 2 objectives: to predict a label relevant to a target task, and to minimize the amount of information encoded in the vector. This objective should ensure that the representation learned in stage 2 avoids encoding textual information – since the decoder already has access to it and the information minimization term discourages redundancy. Instead, it is expected to capture additional acoustic characteristics that are beneficial for the target task. In our approach, pre-trained speech encoders remain frozen as feature extractors, ensuring highly efficient training. Moreover, the textual latent representations produced in the first stage are independent of the target task and can easily be applied to new stage 2 tasks.[1]

As our first case study, we choose emotion recognition as the target task: while the content of an utterance is a strong cue for detecting emotion, the acoustic features provide additional information. For example, consider the utterances "I'm so happy" and "I'm fine," – while the former explicitly conveys happiness through its content, the latter can represent a wide range of emotions depending on prosody and tone. Our second case study closely follows the setup of the first, with speaker identification as the target task. We hypothesize that content offers only limited information, and the acoustic information will play an almost exclusive role in revealing the identity.

2.2 Variational Information Bottleneck (VIB)

To implement an Information Bottleneck, we need to learn a stochastic encoding Z that maximizes

[1]We will publicly release the code and the disentangled models.

the prediction of a target variable Y, while minimizing the information retained about the input H. Accordingly, the loss function could be defined as:

$$\mathcal{L}_{\texttt{IB}} = I(Z,Y) - \beta I(H,Z) \tag{1}$$

where $I(.,.)$ represents mutual information that measures the dependence between two variables. The coefficient $\beta \geq 0$ controls the trade-off between retaining information about either H or Y in Z. However, exact computation of IB loss is intractable, because mutual information involves a summation over all possible states. To address this, the Variational Information Bottleneck (VIB, Alemi et al., 2016) instead computes a lower bound on the IB loss in Equation 1. The VIB loss is defined as[2]:

$$\mathcal{L}_{\texttt{VIB}} = \underbrace{\frac{1}{N}\sum_{n=1}^{N} \mathop{\mathbb{E}}_{z\sim p_\theta(z|h_n)}[\log q_\phi(y_n|z)]}_{\texttt{Task loss}} - \underbrace{\beta \frac{1}{N}\sum_{n=1}^{N} \mathrm{KL}[p_\theta(z|h_n), r(z)]}_{\texttt{Information loss}} \tag{2}$$

where N denotes the sample size, $q_\phi(y|z)$ acts as the decoder, a neural network that predicts the label y from the latent representation z,[3] while $p_\theta(z|h)$ serves as a stochastic encoder, mapping the input h to the representation z; $r(z)$ is the approximate marginal $p(z)$.

2.3 Cascaded disentanglement

Consider a sequence of speech representations $(\boldsymbol{h}_1, ..., \boldsymbol{h}_T)$, produced by a pre-trained speech model, where $\boldsymbol{h}_t$ represents the speech frame representation at timestamp t, and T denotes the total number of frames in a given utterance. Given a target task, our goal is to decompose each frame representation into two distinct latent representations: $\boldsymbol{z}_t^{\texttt{textual}}$ and $\boldsymbol{z}_t^{\texttt{acoustic}}$. Our approach to disentangling these two latent representations is cascaded, and involves two stages, sketched in Figure 1. Both stages use frozen speech representations obtained from the same speech model as input.

In Stage 1, we aim to distill the textual content from speech frame representations, while discarding other non-textual features. The textual capability of a speech model is typically evaluated using automatic speech recognition (ASR), and measured by the word error rate (WER) metric, which assesses how accurately the model can transcribe spoken utterances based on their representations.

We extract all speech frame representations for a given audio from a pre-trained speech model and compute a weighted average across model layers, with the weights learned during training. These frame representations are then given as input to a bottleneck encoder, which compresses the information into low-dimensional latent frame representations. To decode transcriptions from the latent frame representations, we employ the Connectionist Temporal Classification (CTC, Graves et al., 2006) loss as the task loss term in Equation 2, thus minimizing the following loss function:

$$\mathcal{L}_{\texttt{CTC}} - \beta \mathcal{L}_{\texttt{Information}} \tag{3}$$

In this way, we force the bottleneck encoder $p(\boldsymbol{z}^{\texttt{textual}}|\boldsymbol{h})$ to retain only the information necessary for transcription (as encouraged by the CTC loss), while discarding irrelevant features in the original representation ($\boldsymbol{h}$) (constrained by the information loss[4]). We refer to the latent frame representation for frame t learned at this stage as $\boldsymbol{z}_t^{\texttt{textual}}$. Intuitively, these latent vectors capture the minimum statistics of speech representations needed for decoding transcriptions.

Our goal in the Stage 2 is to capture acoustic features that complement the textual features learned in stage 1 and contribute to the downstream task. To achieve this, we replace the task loss in Equation 2 with the Cross-Entropy loss (CE) over the target class labels, thus, minimizing the following loss function:

$$\mathcal{L}_{\texttt{CE}} - \beta \mathcal{L}_{\texttt{Information}} \tag{4}$$

Using labeled data for our target task in this stage, we extract speech frame representations from the same speech model and again learn a weighted average over layers. These representations are then fed into a bottleneck encoder $p(\boldsymbol{z}^{\texttt{acoustic}}|\boldsymbol{h})$—with

[2] See (Alemi et al., 2016) for derivation.

[3] Formally, this represents a variational approximation of the true $p(y_n|z)$, which itself is intractable as VIB assumes that the joint distribution factorizes as $p(y_n, z, h_n) = p(h_n)p(z|h_n)p(y_n|h_n)$ (Alemi et al., 2016).

[4] In an ablation experiment, we trained the encoders without the information loss. We observed a slight improvement in performance, but a failure to disentangle the textual and acoustic information as intended, which confirms the central role of the information loss in our approach.

the same architecture as the encoder in stage 1—to form the complimentary latent representations (z_t^{acoustic}). We apply the information loss to the frame-wise latent representations at the output of the encoder. Subsequently, we pass these latent representations through an attention layer[5] to have latent representations at the utterance level, since labels for our target tasks are assigned to the whole utterance. Finally, the pooled latent representation z^{acoustic} is concatenated with the frozen textual latent representations z^{textual} (previously trained in Stage 1) to decode the target task. This conditional setup encourages the trainable latent representations to retain only non-textual features, particularly those absent in the textual latent representations. During the training process in this stage, no gradient updates are directed back to the z_t^{textual} learned in stage 1.[6]

3 Experiments and Task Performance

3.1 Training details

For training the VIB, we follow (Alemi et al., 2016) and model $r(\boldsymbol{z})$ and $p(\boldsymbol{z}|\boldsymbol{h})$ as multivariate Gaussian distributions: $r(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu}=\mathbf{0}, \boldsymbol{\Sigma}=\mathbf{1})$ and $p(\boldsymbol{z}|\boldsymbol{h}) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu}(\boldsymbol{h}), \boldsymbol{\Sigma}(\boldsymbol{h}))$. The bottleneck encoders to estimate $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ consist of two shared linear layers with the same dimensionality as the original hidden representations ($\boldsymbol{h}_t$), followed by independent d-dimensional layers for each, with GELU (Hendrycks and Gimpel, 2016) activation functions in between. We experiment with different bottleneck dimensions $d = \{16, 32, 64, 128, 256\}$ as the output size of the bottleneck encoders. To estimate the gradients, we employ the reparameterization trick (Kingma and Welling, 2013): $\boldsymbol{z}_t = \boldsymbol{\mu}(\boldsymbol{h}_t) + \boldsymbol{\Sigma}(\boldsymbol{h}_t) \odot \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon}$ is sampled from the normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{1})$. During inference, we use $\boldsymbol{z}_t = \boldsymbol{\mu}(\boldsymbol{h}_t)$. In our implementation, we gradually increase the β coefficient linearly from 0.1 to 1 during training. For decoding, we use a randomly initialized linear projection into C classes, where $C = 32$ for transcription (corresponding to the number of target characters), $C = 4$ for emotion recognition, and $C = 24$ for speaker identification. The training hyperparameters are detailed in Appendix B.

3.2 Target models

To obtain speech representations, we use two prominent self-supervised speech encoders: Wav2Vec2 (Baevski et al., 2020) and HuBERT (Hsu et al., 2021) in their different sizes: Base (12-Transformer layers, 768-hidden size) and Large (24, 1024), obtained from the HuggingFace (Wolf et al., 2020) library. Our experiments include both pre-trained (on raw speech) and fine-tuned[7] (on transcribed speech) versions of these models. Both models employ the Transformer (Vaswani et al., 2017) architecture and learn speech representations through masked prediction in a self-supervised manner. Wav2Vec2 employs a contrastive loss to identify the masked speech frame among distractors, while HuBERT uses k-means clustering to create prediction targets. The models are further fine-tuned with additional labeled speech data by optimizing a linear classifier using CTC loss to decode transcription.

3.3 Data

Transcription. For training in stage 1, we use subsets of two widely used read speech corpora for ASR training: LibriSpeech (Panayotov et al., 2015) and Mozilla's Common Voice 17.0 (Ardila et al., 2019). The former is derived from audiobooks, while the latter is recorded by contributors reading sentences displayed on a screen. We randomly select 4,000 examples from each corpus, ensuring an equal representation of gender and speaker ID, with each sample having a maximum duration of 14 seconds. This results in 17.4 hours of transcribed speech training data.[8] For evaluation, we use the entire test-clean set of the LibriSpeech dataset. Following (Baevski et al., 2020), we remove non-spoken special characters (e.g., commas and periods) from transcriptions, as these are not included in our target vocabulary.

[5] We choose an attention layer over a simple average pooling to later recognize the key frame representations for interpretability.

[6] In a second ablation experiment (Appendix A), we explored the feasibility of jointly training textual and acoustic encoders as an alternative to our two-stage approach. However, we found that joint training consistently led to considerable performance degradation across all tasks and models. Additionally, training in stages offers a key advantage: the textual latent representations in stage 1 can be trained once and reused across all target tasks, while stage 2 can be trained efficiently in just a few minutes.

[7] The HuBERT model lacks a released fine-tuned checkpoint in the Base size.

[8] Similar to (Baevski et al., 2020), in our experiments we found that a few hours of transcribed data are sufficient to train a classifier for decoding transcription from pre-trained speech models.

		Stage 1 **Transcription** (WER ↓)		Stage 2 **Emotion** (Acc. ↑)		Stage 2 **Speaker Id.** (Acc. ↑)	
Model	**Size**	Original	VIB	Original	VIB	Original	VIB
HuBERT	Base	$45.4_{\pm 0.000}$	$41.6_{\pm 0.006}$	$61.8_{\pm 0.007}$	$62.7_{\pm 0.006}$	$97.8_{\pm 0.001}$	$99.1_{\pm 0.001}$
	Large	$35.0_{\pm 0.007}$	$37.6_{\pm 0.041}$	$57.6_{\pm 0.002}$	$66.1_{\pm 0.035}$	$92.6_{\pm 0.003}$	$98.4_{\pm 0.013}$
Wav2Vec2	Base	$50.0_{\pm 0.000}$	$45.0_{\pm 0.002}$	$61.7_{\pm 0.001}$	$58.9_{\pm 0.033}$	$99.8_{\pm 0.000}$	$97.9_{\pm 0.017}$
	Large	$47.0_{\pm 0.001}$	$48.7_{\pm 0.034}$	$62.2_{\pm 0.004}$	$65.9_{\pm 0.009}$	$97.9_{\pm 0.004}$	$99.8_{\pm 0.000}$
HuBERT-FT	Large	$3.1_{\pm 0.000}$	$8.2_{\pm 0.077}$	$54.6_{\pm 0.013}$	$64.8_{\pm 0.040}$	$90.5_{\pm 0.003}$	$98.2_{\pm 0.003}$
Wav2Vec2-FT	Base	$5.8_{\pm 0.000}$	$12.7_{\pm 0.008}$	$63.2_{\pm 0.006}$	$59.8_{\pm 0.056}$	$99.7_{\pm 0.000}$	$98.2_{\pm 0.002}$
	Large	$3.5_{\pm 0.000}$	$8.3_{\pm 0.002}$	$62.0_{\pm 0.004}$	$63.4_{\pm 0.007}$	$98.5_{\pm 0.001}$	$99.6_{\pm 0.000}$

Table 1: Performance of decoders trained on the *original* hidden states & the VIB latent vectors ($d = 128$). Lower WER and higher Accuracy are better. Random baselines: WER = 100 for transcription, Accuracy = 25 for emotion, and Accuracy = 4.1 for speaker identification. Scores are averaged over three independent runs with different random seeds (with ± standard deviation rounded to three decimal places).

Target tasks. For emotion data in stage 2, we use IEMOCAP (Busso et al., 2008) database as its emotion labels are influenced by both lexical content and acoustic properties. Following prior research (Li et al., 2021, 2022), we exclude utterances without transcripts and combine the *Happy* and *Excited* labels to form a 4-way classification task. We then undersample the dataset to balance emotion classes, resulting in 4,064 utterances (∼5 hours of audio, with an average duration of 4.4 seconds per segment). Each utterance is assigned one emotion from the label set: {*Angry*, *Happy*, *Neutral*, *Sad*}. For Speaker Identity as our second target task, we use a subset of Mozilla's Common Voice 17.0 dataset (Ardila et al., 2019), consisting of 4,000 training and 1,000 test samples, stratified by two genders (male and female) and 24 speaker identities. All audio files in this study are resampled to 16 kHz to match the sampling rate used for the pre-training data of the target models.

3.4 Task performance

In this section, we test the effectiveness of the disentangled representations against the original entangled representations on the downstream tasks. We will verify the disentanglement in the next section. For comparison, we also train identically structured decoders which rely on the original hidden states (h_t); we will label this condition 'Original'. The performance of these classifiers (which can be viewed as a repurposed form of *probing*; Alain and Bengio, 2016; Tenney et al., 2019) serves as a strong baseline, representing the performance achievable relying on the hidden states, without compressing them into latent representations.

Table 1 reports both VIB and Original performances, averaged over three runs with different random seeds, for transcription, emotion recognition, and speaker identification tasks. For the transcription task at stage 1, VIB demonstrates a similar or sometimes even lower word error rate (WER)[9] compared to the Original condition, implying the success of VIB training in compressing essential information for audio transcription (WER = 100 presents the performance of the random baseline).[10] As expected, representations derived from fine-tuned models show better transcription performance compared to those from pre-trained models (for both VIB and Original) as they are specifically tuned for transcription. The same success is evident in decoding our target tasks in stage 2, where we report the accuracy of both VIB and probing classifiers for emotion recognition and speaker identification tasks (the random baseline accuracy is 25 and 4.1, respectively). Overall, the classifiers benefit more from the specialized, compressed representations than from the original hidden states, as VIB encourages retaining only task-relevant information, resulting in more robust representations.

We also found VIB performance consistent across various bottleneck dimensions (details omitted). Both VIB and probing learn weights for layer

[9] We also evaluated the transcription performance using character error rate (CER), yielding a similar overall pattern.

[10] Note: In the VIB setup, unlike Original, the transcription classifier is trained on noisy representations ($\boldsymbol{z}_t = \boldsymbol{\mu}(\boldsymbol{h}_t) + \boldsymbol{\Sigma}(\boldsymbol{h}_t) \odot \boldsymbol{\epsilon}$), which explains why the VIB scores a few points worse than Original on transcription in the finetuned model. When we train a new classifier on the final, trained latent representations ($\boldsymbol{z}_t = \mu(\boldsymbol{h}_t)$) both setups yield similar results.

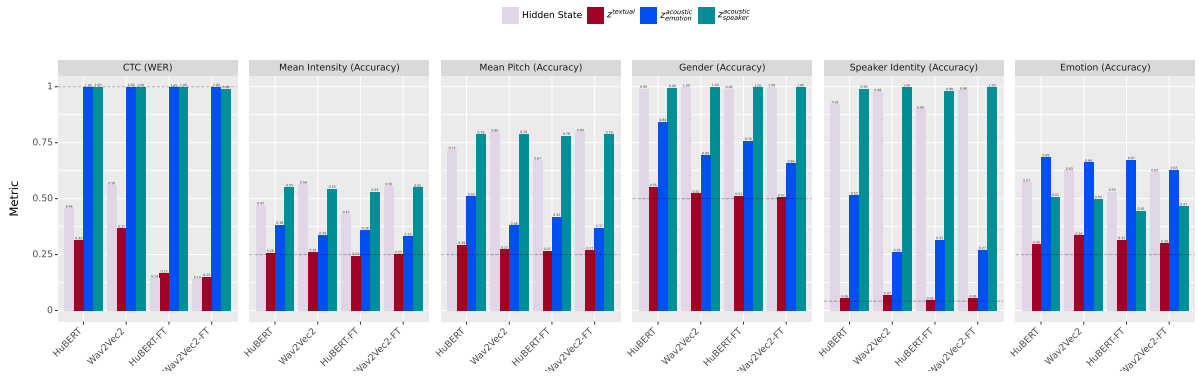

Figure 2: Probing performances of latent representations ($d = 128$) learned at stages 1 and 2, along with original hidden states derived from Large models for transcription and a set of audio features (Mean Intensity, Mean Pitch, Gender, and Speaker Identity). Probing for emotion is included only for comparison. For the WER metric, the lower score is better, while for other metrics, the higher is better. The dashed line in each plot represents the random baseline.

averaging (see Figure 1). These weights are necessary because the information is not uniformly distributed across the SSSM layers. Both approaches use these weights; the weights provide insights into the contribution of individual layers to the acoustic and textual features.

4 Evaluation of Disentangled Representations

4.1 Evaluating the degree of disentanglement

The product of our training procedure are the encoders which result in the disentangled latent representations $\boldsymbol{z}_t^{\texttt{textual}}$ and $\boldsymbol{z}_t^{\texttt{acoustic}}$ for each speech frame representation h_t. In this section, we investigate these latent representations to validate if they are truly disentangled by probing them for various types of textual and acoustic information. Given an audio input, the goal is that $\boldsymbol{z}_t^{\texttt{textual}}$, which is specialized for text, should not encode any aspects of the acoustic characteristics of the audio. In contrast, $\boldsymbol{z}_t^{\texttt{acoustic}}$, which is specialized for audio features, should not contain any information about the audio transcription.

To assess textual capability, we train a linear probing classifier on frozen latent representations to predict their transcriptions. To estimate acoustic capability, we train a set of probing classifiers followed by an attention layer on latent representations to predict various acoustic features at the utterance level, including Mean Intensity, Mean Pitch, Gender, and Speaker Identity. For comparison. we also probe the original hidden states for the same objectives with the identically-structured classifiers. We use a subset of Mozilla's Common Voice dataset (Ardila et al., 2019), consisting of 4,000 training and 1,000 test samples, stratified by two genders (male and female)[11] and 24 speakers. We extract labels for acoustic features directly from the raw audio waveforms using the Parselmouth toolkit (Jadoul et al., 2018). We then discretize them into four equally sized buckets based on quantiles to cast the task as a four-way classification problem.

Figure 2 illustrates the classification results for Large models. Dashed lines represent the random baseline performance. As we can see, acoustic latent representations (for both target tasks; $\boldsymbol{z}_{\texttt{emotion}}^{\texttt{acoustic}}$ and $\boldsymbol{z}_{\texttt{speaker}}^{\texttt{acoustic}}$) exhibit no awareness of the textual content of the audio as their performance for CTC matches the random baseline (WER $= 1$). Conversely, textual latent representations are as effective as the original hidden states and – for non-finetuned models – even outperform them at decoding transcription.

Looking into predicting acoustic features, textual latent representations consistently show random performance, suggesting no acoustic features are encoded within them. Acoustic latent representations, however, show substantial probing performance despite not having any explicit acoustic objective in their training at stage 2. Interestingly, acoustic latent representations for the task of speaker identification are better at encoding acoustic features than those of emotion recognition. It is

[11] The exclusion of other groups is due to the binary labeling in the dataset, rather than a choice by the authors.

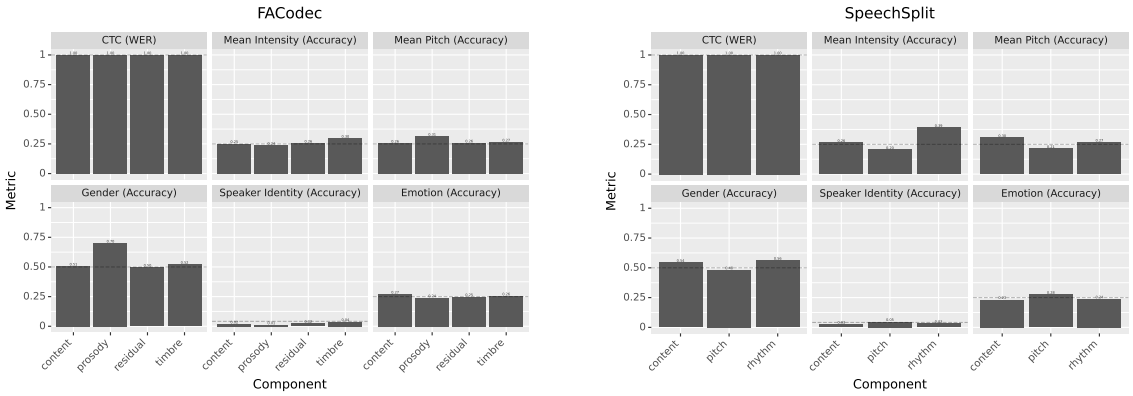

Figure 3: Probing performances of disentangled components derived from FACodec (*left*) and SpeechSplit (*right*) frameworks for transcription and a set of audio features.

likely due to acoustic information playing a greater role in revealing the speaker identity.

Additionally, in contrast to $\boldsymbol{z}^{\texttt{acoustic}}_{\texttt{speaker}}$, acoustic latent representations for emotion ($\boldsymbol{z}^{\texttt{acoustic}}_{\texttt{emotion}}$) do not match the performance of the original hidden states. For example, for the Wav2Vec2 model, the probing performance for Speaker ID based on hidden states is 0.98, while it is 0.26 for the acoustic latent representations (the random baseline is $1/24 \approx 0.04$). This disparity suggests that not all those acoustic features encoded in the hidden representations are crucial for recognizing emotion, thus, not all of those features were retained in stage 2. These findings could be important in real-world scenarios where, for privacy protection, encoding acoustic features without precisely identifying the speaker can be essential. The pattern of the probing results is also similar for Base models and for various bottleneck dimensions (detailed results omitted because of space).

4.2 Comparison with existing frameworks

Our cascaded disentanglement differs from the related work we are aware of. FACodec (Ju et al., 2024) and SpeechSplit (Qian et al., 2020) are two recent speech disentanglement frameworks with complex training architectures that decompose speech signals into predefined components, regardless of the target downstream task. In contrast to our approach, they achieve this by reconstructing the input audio during their training procedures. This enables them to evaluate their method on style transfer audio conversion, but they do not assess whether there is still overlap in the information contained in their (disentangled) components. Our probing evaluation, in contrast, is applicable to all models, and allows us to compare the quality of the disentangled internal representations obtained with different approaches.

We therefore extract the embeddings of the disentangled components from the FACodec and SpeechSplit frameworks and similarly trained a linear classifier to probe them for various types of textual and acoustic information. Figure 3 shows results. We observe that the acoustic components derived from their disentanglement framework perform poorly in linearly decoding acoustic features when compared to our acoustic latent representation (Figure 2). Also, it is not possible to linearly decode the transcription (content) of the audio from their content vectors (in contrast to our textual latent representations). Furthermore, the content vector derived from the SpeechSplit framework shows better performance than its acoustic components at predicting Mean Pitch, indicating contamination of the content vector with acoustic features. For the purposes we have in mind in this paper (see introduction), we therefore consider these representations not properly disentangled.

4.3 Qualitative evaluation

We visualize the latent representations to gain insight into how they have been encoded in the representation space for emotion recognition. We use the RAVDESS dataset (Livingstone and Russo, 2018) which contains only two identical statements spoken by 24 actors with 4 different emotions. This makes it ideal for this analysis, as the linguistic content remains constant across utterances with different emotions. We select examples with matching emotion labels from the IEMOCAP dataset, resulting in 384 utterances. We obtain the speech rep-

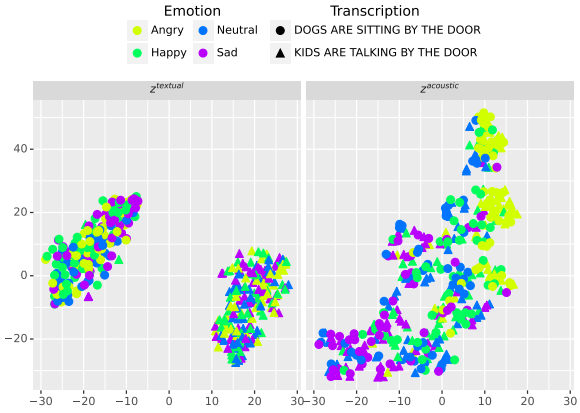

Figure 4: t-SNE of the textual and acoustic latent representations for Wav2Vec2-Large, marked and colored according to their transcription and emotion labels, respectively.

resentations from the large Wav2Vec2 model and generate their corresponding latent representations. Then, we compute the average of frame representations over all frames in each utterance and apply t-SNE. Figure 4 shows a 2D projection of these latent representations with data points marked and colored according to their transcription and emotion labels, respectively. The textual representations are clustered according to their transcriptions. In contrast, the acoustic representations are not separated by transcription, suggesting that textual information is not retained there. Instead, these acoustic representations are roughly clustered by emotion, which aligns with our desired goal. The pattern is similar for the Base model (not shown).

5 Using disentanglement for improved interpretability

5.1 Division of labor between layers

With our approach to learning disentanglement validated, we can now quantify, separately, the extent to which each layer in a self-supervised speech model contributes textually and acoustically to the target task of emotion recognition. We first train the latent representations using the same setup as we described in Section 4, but instead of using a weighted layer average in the input, we use the frame representations from a specific layer of the model. The layerwise results for stages 1 and 2 are shown in Figure 5, with the black horizontal dashed line representing random performances.

For HuBERT, the transcription ability improves as we move through the layers. In contrast, Wav2Vec2 shows a U-shaped pattern, with the best performance in the middle layers and the final layers being unable to decode transcription. For Wav2Vec2-FT, however, transcription performance improves sharply in the last layers, which is expected given that the model is fine-tuned for ASR, confirming substantial changes in the last layers during fine-tuning (Pasad et al., 2021). In stage 2 (middle panel), models perform best at emotion recognition in the middle layers. Compared to pre-trained, the fine-tuned model shows decline in emotion recognition performance in the final layers, where acoustic information is lost in favor of encoding text transcription.

We next train a probing classifier (followed by an attention layer) on top of frozen $z_t^{\texttt{textual}}$ and $z_t^{\texttt{acoustic}}$ latent representations to decode emotion. Each latent representation is specialized to preserve either textual or acoustic features from the original hidden states of each model layer. Therefore, the probing performance reveals the extent to which these textual and acoustic features contribute to emotion recognition (Figure 5, right). Comparing pre-trained and fine-tuned models, the latent acoustic representations learned from the final layers of Wav2Vec-FT contribute significantly less to emotion recognition. This is likely due to the model losing some acoustic information during fine-tuning in favor of transcription capabilities. However these layers benefit more from textual features in predicting emotion, as their representations offer more accurate transcriptions. Compared to Wav2Vec2, HuBERT shows a greater contribution to emotion recognition; acoustically in the middle layers, and textually in the last layer.

5.2 Input attributions

In this section, we use our disentangled vectors to localize salient input features for the target tasks. The attention layer in stage 2 (see Figure 1) can be used to identify those frames in the original audio input whose latent representations contribute most to our target tasks. Crucially, the disentanglement mechanism allows us to clearly separate the contributions of acoustic features from those of textual features. This *disentangled* attribution could be particularly useful in fields like psychiatry, where differences between textual and acoustic emotional expressions can aid in the diagnosis of disorders (Niu et al., 2023), or in detecting bias in speech agents' responses to user requests.

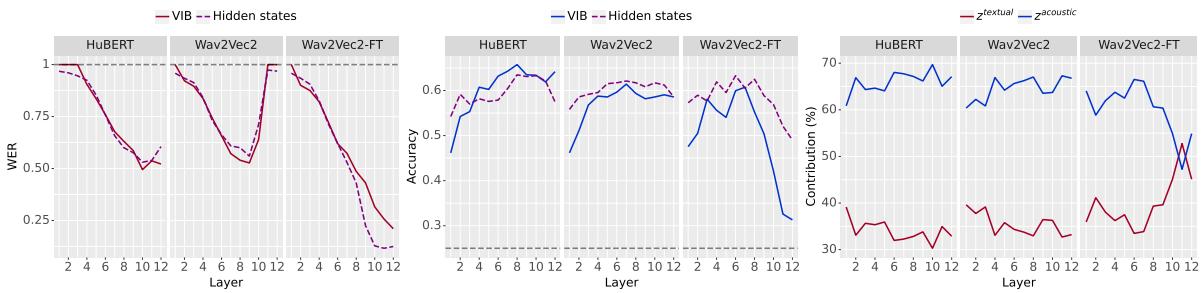

Figure 5: Layerwise performance of VIB ($d = 128$) for transcription at stage 1 (*left*) and emotion classification at stage 2 (*middle*), compared to layerwise performance using original hidden state activations. *Right*: layerwise textual and acoustic contribution to emotion classification, compared to layerwise performance using hidden states of Base models. Lower WER and higher accuracy are better. The black horizontal dashed line represents the random baseline.

Finding salient input features for model decisions is often done by computing the gradient of the model's output with respect to the inputs (Sundararajan et al., 2017; Ancona et al., 2018; Yuan et al., 2019; Samek et al., 2019). To compare our attention scores with gradient-based attribution scores, we train a classifier for the target task on the original hidden states of the Wav2Vec2 model and use Integrated Gradients (IG) to compute attribution to individual frames (Sundararajan et al., 2017). We normalize the IG scores to sum to 1.

We then obtain frame-by-frame distributions of both acoustic and sentiment features to compare them directly with attribution results. For acoustic features, we focus on intensity and pitch, identifying and representing peaks and valleys in their temporal patterns. To analyze sentiment, we use the spaCy toolkit (Honnibal and Montani, 2017) to annotate word-level polarity (positive, negative, or neutral) within the utterances. Using Montreal Forced Aligner (McAuliffe et al., 2017), we map these word-level labels to frames (Appendix B). As a result, each feature vector for a speech frame includes the following dimensions: (1) the presence of a peak or valley in intensity, (2) the same for pitch, and (3) the presence of a 'sentiment-laden' word, all normalized to the range $[0, 1]$.

We then compute the dot product between the frame-wise attribution scores and the corresponding feature vectors. Figure 6 presents these results, averaged across all examples in the IEMOCAP test set, and demonstrates that acoustic attention effectively captures peaks and valleys in acoustic features (intensity and pitch), while textual attention focuses on word polarity. Both have higher agreement with features than Integrated Gradient scores.

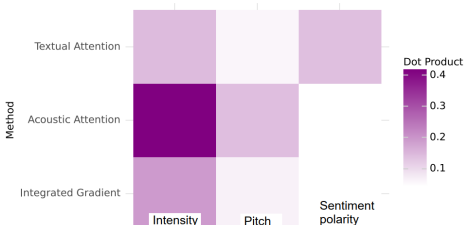

Figure 6: Dot product of attribution scores and different acoustic and textual cues.

Note that textual attention exhibits fairly high similarity with acoustic features as polar words are often pronounced with emotional emphasis.

6 Conclusions

We present an approach to posthoc disentanglement of representations from self-supervised speech models into textual and acoustic components, retaining only features relevant to the target tasks. We show that our setup using a variation information bottleneck and cascaded learning can effectively isolate key features and improve interpretability while maintaining the performance of the original models, and can identify the most important speech frames from both textual and acoustic perspectives. A unique aspect of our approach is its modality-independent architecture: in principle, any entangled multi-modal representation could be disentangled as long as appropriate target tasks are implemented. This opens up applications beyond neural speech recognition, e.g. for disentangling linguistic and visual components in representations of grounded language models.

References

Guillaume Alain and Yoshua Bengio. 2016. Understanding intermediate layers using linear classifier probes. *ArXiv*, abs/1610.01644.

Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. 2016. Deep variational information bottleneck. In *International Conference on Learning Representations*.

Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. 2018. Towards better understanding of gradient-based attribution methods for deep neural networks. In *International Conference on Learning Representations*.

Rosana Ardila, Megan Branson, Kelly Davis, Michael Henretty, Michael Kohler, Josh Meyer, Reuben Morais, Lindsay Saunders, Francis M. Tyers, and Gregor Weber. 2019. Common voice: A massively-multilingual speech corpus. *ArXiv*, abs/1912.06670.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828.

Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N Chang, Sungbok Lee, and Shrikanth S Narayanan. 2008. Iemocap: Interactive emotional dyadic motion capture database. *Language resources and evaluation*, 42:335–359.

Cheol Jun Cho, Peter Wu, Tejas S Prabhune, Dhruv Agarwal, and Gopala K Anumanchipalli. 2024. Coding speech through vocal tract kinematics. *IEEE Journal of Selected Topics in Signal Processing*, 18(8):1427–1440.

Hyeong-Seok Choi, Juheon Lee, Wan Soo Kim, Jie Hwan Lee, Hoon Heo, and Kyogu Lee. 2021. Neural analysis and synthesis: Reconstructing speech from self-supervised representations. *ArXiv*, abs/2110.14513.

Hyeong-Seok Choi, Jinhyeok Yang, Juheon Lee, and Hyeongju Kim. 2022. Nansy++: Unified voice synthesis with neural analysis and synthesis. *ArXiv*, abs/2211.09407.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, page 369–376, New York, NY, USA. Association for Computing Machinery.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv: Learning*.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdel rahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460.

Yannick Jadoul, Bill Thompson, and Bart de Boer. 2018. Introducing parselmouth: A python interface to praat. *Journal of Phonetics*, 71:1–15.

Zeqian Ju, Yuancheng Wang, Kai Shen, Xu Tan, Detai Xin, Dongchao Yang, Yanqing Liu, Yichong Leng, Kaitao Song, Siliang Tang, Zhizheng Wu, Tao Qin, Xiang-Yang Li, Wei Ye, Shikun Zhang, Jiang Bian, Lei He, Jinyu Li, and Sheng Zhao. 2024. Naturalspeech 3: Zero-shot speech synthesis with factorized codec and diffusion models. *ArXiv*, abs/2403.03100.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Yuanchao Li, Peter Bell, and Catherine Lai. 2021. Fusing asr outputs in joint training for speech emotion recognition. *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7362–7366.

Yuanchao Li, Yumnah Mohamied, Peter Bell, and Catherine Lai. 2022. Exploration of a self-supervised speech model: A study on emotional corpora. *2022 IEEE Spoken Language Technology Workshop (SLT)*, pages 868–875.

Steven R Livingstone and Frank A Russo. 2018. The ryerson audio-visual database of emotional speech and song (ravdess): A dynamic, multimodal set of facial and vocal expressions in north american english. *PloS one*, 13(5):e0196391.

Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. 2017. Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi. In *Proc. Interspeech 2017*, pages 498–502.

Minxue Niu, Amrit Romana, Mimansa Jaiswal, Melvin McInnis, and Emily Mower Provost. 2023. Capturing mismatch between textual and acoustic emotion expressions for mood identification in bipolar disorder. In *Interspeech*.

Aaron van den Oord, Oriol Vinyals, and koray kavukcuoglu. 2017. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: An asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210.

Ankita Pasad, Ju-Chieh Chou, and Karen Livescu. 2021. Layer-wise analysis of a self-supervised speech representation model. *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 914–921.

Adam Polyak, Yossi Adi, Jade Copet, Eugene Kharitonov, Kushal Lakhotia, Wei-Ning Hsu, Abdelrahman Mohamed, and Emmanuel Dupoux. 2021. Speech resynthesis from discrete disentangled self-supervised representations. In *Interspeech*.

Kaizhi Qian, Yang Zhang, Shiyu Chang, David Cox, and Mark A. Hasegawa-Johnson. 2020. Unsupervised speech decomposition via triple information bottleneck. In *International Conference on Machine Learning*.

Kaizhi Qian, Yang Zhang, Shiyu Chang, Xuesong Yang, and Mark A. Hasegawa-Johnson. 2019. Zero-shot voice style transfer with only autoencoder loss. *ArXiv*, abs/1905.05879.

Leyuan Qu, Taihao Li, Cornelius Weber, Theresa Pekarek-Rosin, Fuji Ren, and Stefan Wermter. 2024. Disentangling prosody representations with unsupervised speech reconstruction. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 32:39–54.

Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller. 2019. *Explainable AI: interpreting, explaining and visualizing deep learning*, volume 11700. Springer Nature.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3319–3328. JMLR.org.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Naftali Tishby, Fernando C. Pereira, and William Bialek. 2000. The information bottleneck method.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Hao Yuan, Yongjun Chen, Xia Hu, and Shuiwang Ji. 2019. Interpreting deep models for text analysis via optimization and regularization methods. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5717–5724.

		Transcription (WER ↓)			Emotion (Acc. ↑)		Speaker Id. (Acc. ↑)	
Model	**Size**	VIB (Stage 1)	Joint VIB (w/ Emotion)	Joint VIB (w/ Speaker Id.)	VIB (Stage 2)	Joint VIB	VIB (Stage 2)	Joint VIB
HuBERT	Base	41.6	64.1	92.6	62.7	61.9	99.1	98.0
	Large	37.6	35.6	44.2	66.1	59.1	98.4	97.7
Wav2Vec2	Base	45.0	100	100	58.9	56.1	97.9	95.3
	Large	48.7	49.8	86.1	65.9	59.1	99.8	97.7
HuBERT-FT	Large	8.2	16.7	33.5	64.8	64.1	98.2	87.8
Wav2Vec2-FT	Base	12.7	17.3	74.3	59.8	54.3	98.2	97.4
	Large	8.3	16.1	39.8	63.4	57.7	99.6	97.9

Table 2: Comparison of **cascaded VIB** with **joint VIB**. Random baselines: WER = 100 (transcription), Accuracy = 25 (emotion), Accuracy = 4.1 (speaker identification).

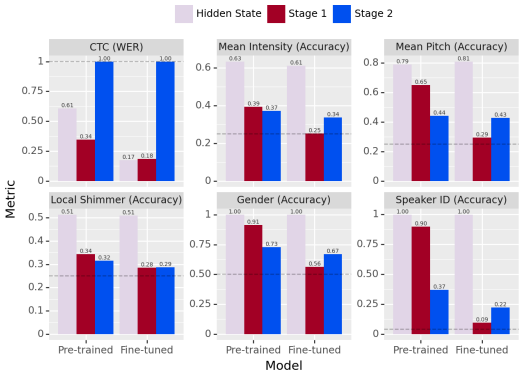

Figure 7: Probing performances of latent representations learned at stages 1 and 2 **without information loss constraint**, along with hidden states derived from **Wav2Vec2-Base** model for transcription and a set of audio features.

A Appendix: Ablation experiments

To assess the importance of the Information Bottleneck, we trained the textual and acoustic encoders without the IB loss. As shown in Figure 7, excluding information loss during training results in a failure to disentangle textual and acoustic information as intended, confirming its central role in our approach.

To assess the importance of cascading (sec. 2.3), we run experiments where textual and acoustic latent variables are trained jointly. At each training step, we feed the transcribed data into the textual encoder and the transcription classifier to calculate the CTC loss ($\mathcal{L}_{\texttt{CTC}}$), and feed the acoustic encoder and target task classifier with target task data to compute the Cross entropy loss ($\mathcal{L}_{\texttt{CE}}$). Both latent variables are also constrained by information bottleneck ($\mathcal{L}_{\texttt{Info}}$) losses. Then, we combine losses and perform backpropagation to jointly optimize both model parameters to minimize:

$$\mathcal{L}_{\texttt{CTC}} - \beta\mathcal{L}_{\texttt{Info1}} + \mathcal{L}_{\texttt{CE}} - \beta\mathcal{L}_{\texttt{Info2}} \qquad (5)$$

Incorporating transcribed data alongside the target task is necessary during training because: 1) we have limited transcribed data for downstream tasks (e.g., 5 hours for emotion recognition), which is not sufficient for learning transcription, and 2) in real-world applications transcriptions of speech are not always available. In this way, both classifiers are trained on the same data as in our two-stage VIB training, but now they are jointly optimized, making it comparable to the two-stage procedure.

Table 2 shows that joint training consistently and significantly underperforms the cascaded approach, and in two cases (for pre-trained Wav2Vec2), it completely fails to decode transcriptions, which highlights the importance of our two-stage design choice. Moreover, cascaded training offers key advantages: the textual latent representations in stage 1 can be trained once and reused across all target tasks, while stage 2 can be trained efficiently in just a few minutes.

B Appendix: Details for replicability

All models were trained for 50 epochs using the AdamW optimizer with gradient norm clipping. The learning rate was set to 1e−3 and 1e−4 for Base and Large models, respectively, with a warmup ratio of 0.1 and a cosine decay scheduler, together with weight decay. For training transcription prediction using CTC loss, we use a batch size of 1; the effective batch size here is the number of frame representations since the CTC loss is computed across all frames. For other training objectives, we use a batch size of 8. When using the Montreal Forced Aligner (McAuliffe et al., 2017, MFA), we extract the start time (t_s) and the end time (t_e) of each word in an utterance, and map them to boundary frames f_s and f_e: $f = \lceil \frac{t}{\mathcal{T}} \times T \rceil$ where $\mathcal{T}$ and T denote the total time of a given audio and the total number of frames in the frame representation, respectively.