A. Proof to Theorem 1

The proof is divided into five parts.

Lemma 1.1. *Under the assumptions in theorem 1, the global minimum of equation (4) is 0.*

Proof. Construct the encoders that satisfy equation (2), which is a feasible choice. Construct the decoder as follows:

$$\begin{aligned} &D(\boldsymbol{Z}_c, \boldsymbol{Z}_r, \boldsymbol{Z}_f, \boldsymbol{V}) \\ =&g_s(h_c^{-1}(\boldsymbol{Z}_c), h_r^{-1}(\boldsymbol{Z}_r), h_f^{-1}(\boldsymbol{Z}_f), g_u^{-1}(\boldsymbol{V})) \\ =&g_s(\boldsymbol{C}, \boldsymbol{R}, \boldsymbol{F}, U) \\ =&\boldsymbol{S}, \end{aligned} \tag{12}$$

which achieves 0 reconstruction loss in equation (4). □

Lemma 1.2. *Equation (8) implies*

$$I(\boldsymbol{R}; A(\boldsymbol{S}), f(\boldsymbol{R})) < H(\boldsymbol{R}), \quad \forall f(\cdot) \text{ s.t. } H(\boldsymbol{R}|f(\boldsymbol{R})) > 0. \tag{13}$$

Proof. We will prove this by contradiction. If there exists an $f(\cdot)$ s.t. $H(\boldsymbol{R}|f(\boldsymbol{R})) > 0$ but $I(\boldsymbol{R}; A(\boldsymbol{S}), f(\boldsymbol{R})) = H(\boldsymbol{R})$, then there exist $\boldsymbol{r}_1 \neq \boldsymbol{r}_2$, which $f(\cdot)$ cannot distinguish but $A(\boldsymbol{S})$ can, *i.e.*

$$A(g_s(\boldsymbol{C}_1, \boldsymbol{r}_1, \boldsymbol{F}_1, \boldsymbol{V}_1)) \neq A(g_s(\boldsymbol{C}_2, \boldsymbol{r}_2, \boldsymbol{F}_2, \boldsymbol{V}_2)), \quad \text{w.p. } 1. \tag{14}$$

which contradicts with (8). □

Lemma 1.3. *Under the assumptions in theorem 1, in order to achieve the global minimum of equation (4), $\boldsymbol{Z}_r$ must satisfy equation (2).*

Proof. We will prove this by contradiction. If

$$H(\boldsymbol{R}|\boldsymbol{Z}_r) > 0, \tag{15}$$

then we have

$$\begin{aligned} H(\boldsymbol{R}|\hat{\boldsymbol{S}}) &\geq H(\boldsymbol{R}|\boldsymbol{Z}_r, \boldsymbol{Z}_c, \boldsymbol{Z}_f, \boldsymbol{V}) \\ &\geq H(\boldsymbol{R}|\boldsymbol{Z}_r, \boldsymbol{Z}_c, \boldsymbol{Z}_f) \\ &\geq H(\boldsymbol{R}|\boldsymbol{Z}_r, \boldsymbol{A}(\boldsymbol{S}), \boldsymbol{A}(\boldsymbol{P})) \\ &> 0, \end{aligned} \tag{16}$$

where the first and third lines are due to the data processing inequality; the second line is given by equation (1) and the independence assumption among the aspects; the last line is given by equation (13). Equation (16) essentially means $\hat{\boldsymbol{S}}$ cannot reconstruct $\boldsymbol{R}$, and thereby cannot reconstruct $\boldsymbol{S}$, which contradicts with the optimal loss being 0.

Moreover, if

$$H(\boldsymbol{Z}_r|\boldsymbol{R}) > 0, \tag{17}$$

then

$$H(\boldsymbol{R}|\boldsymbol{Z}_r) = H(\boldsymbol{Z}_r|\boldsymbol{R}) + H(\boldsymbol{R}) - H(\boldsymbol{Z}_r) \geq H(\boldsymbol{Z}_r|\boldsymbol{R}) > 0, \tag{18}$$

where the '$\geq$' inequality is from equation (11). This will again lead to a contradiction. □

Lemma 1.4. *Under the assumptions in theorem 1, and assuming $\boldsymbol{Z}_r$ satisfies equation (2), in order to achieve the global minimum of equation (4), $\boldsymbol{Z}_c$ must satisfy equation (2).*

Proof. We will prove this by contradiction. If

$$H(\boldsymbol{C}|\boldsymbol{Z}_c) > 0, \tag{19}$$

then we have

$$\begin{aligned} H(\boldsymbol{C}|\hat{\boldsymbol{S}}) &\geq H(\boldsymbol{C}|\boldsymbol{Z}_r, \boldsymbol{Z}_c, \boldsymbol{Z}_f) \\ &= H(\boldsymbol{C}|f_r(\boldsymbol{R}), \boldsymbol{Z}_c, \boldsymbol{Z}_f) \\ &= H(\boldsymbol{C}|\boldsymbol{Z}_c, \boldsymbol{Z}_f) \\ &\geq H(\boldsymbol{C}|\boldsymbol{Z}_c, \boldsymbol{F}) \\ &= H(\boldsymbol{C}|\boldsymbol{Z}_c, g_p(\boldsymbol{F}, \boldsymbol{R})) \\ &= H(\boldsymbol{C}|\boldsymbol{Z}_c) > 0, \end{aligned} \tag{20}$$

where the first line is similar to equation (16); the second line is given by $\boldsymbol{R}$ satisfying equation (2); the third and last lines are due to the independence assumption among the aspects; the fourth line is given by the data processing inequality; the fifth line is given by equation (10). Equation (20) essentially means $\hat{\boldsymbol{S}}$ cannot reconstruct $\boldsymbol{C}$, and thereby cannot reconstruct $\boldsymbol{S}$, which contradicts with the optimal loss being 0.

Moreover, if

$$H(\boldsymbol{Z}_c|\boldsymbol{C}) > 0, \tag{21}$$

then

$$H(\boldsymbol{C}|\boldsymbol{Z}_c) = H(\boldsymbol{Z}_c|\boldsymbol{C}) + H(\boldsymbol{C}) - H(\boldsymbol{Z}_c) \geq H(\boldsymbol{Z}_c|\boldsymbol{C}) > 0, \tag{22}$$

where the '$\geq$' inequality is from equation (11). This will again lead to a contradiction. □

Lemma 1.5. *Under the assumptions in theorem 1, and assuming $\boldsymbol{Z}_r$ and $\boldsymbol{Z}_c$ satisfy equation (2), in order to achieve the global minimum of equation (4), $\boldsymbol{Z}_f$ must satisfy equation (2).*

Proof. We will prove this by contradiction. If

$$H(\boldsymbol{F}|\boldsymbol{Z}_f) > 0, \tag{23}$$

then we have

$$\begin{aligned} H(\boldsymbol{F}|\hat{\boldsymbol{S}}) &\geq H(\boldsymbol{F}|\boldsymbol{Z}_r, \boldsymbol{Z}_c, \boldsymbol{Z}_f) \\ &= H(\boldsymbol{C}|f_r(\boldsymbol{R}), f_c(\boldsymbol{C}), \boldsymbol{Z}_f) \\ &= H(\boldsymbol{C}|\boldsymbol{Z}_f) > 0, \end{aligned} \tag{24}$$

where the first line is similar to equation (16); the second line is given by $\boldsymbol{R}$ and $\boldsymbol{C}$ satisfying equation (2); the third is due to the independence assumption among the aspects. Equation (24) essentially means $\hat{\boldsymbol{S}}$ cannot reconstruct $\boldsymbol{F}$, and thereby cannot reconstruct $\boldsymbol{S}$, which contradicts with the optimal loss being 0.

Moreover, if

$$H(\boldsymbol{Z}_f|\boldsymbol{F}) > 0, \tag{25}$$

then

$$H(\boldsymbol{F}|\boldsymbol{Z}_f) = H(\boldsymbol{Z}_f|\boldsymbol{F}) + H(\boldsymbol{F}) - H(\boldsymbol{Z}_f) \geq H(\boldsymbol{Z}_f|\boldsymbol{F}) > 0. \tag{26}$$

where the '$\geq$' inequality is from equation (11). This will again lead to a contradiction. □

Theorem 1 can be implied by combining lemmas 1.3, 1.4 and 1.5.

B. Additional Implementation Details

B.1. Input Features

The input and output spectrograms are 80-dimensional mel-spectrograms computed using 64 ms frame length and 16 ms frame hop. For each speaker, the input pitch contour is first extracted using a pitch tracker (Yamamoto et al., 2019), and then normalized by its mean and four times its standard deviation. This operation roughly limits the pitch contour to be within the range of 0-1. After that, we quantize the range 0-1 into 256 bins and turn it into one-hot representations. Finally, we add another bin to represent unvoiced frames producing 257 one-hot encoded feature $\boldsymbol{P}$.

B.2. Information Bottleneck Implementations

As discussed, SPEECHFLOW adopts two methods to restrict the information flow. The first is random resampling, and the second is the constraints on the physical dimensions, which include the downsampling operations in frequency and time dimensions.

The random resampling is implemented as follows. First, the input signal is divided into segments, whose length is randomly uniformly drawn from 19 frames to 32 frames (Polyak & Wolf, 2019). Each segment is resampled using linear interpolation with a resampling factor randomly drawn from 0.5 (compression by half) to 1.5 (stretch). For each input utterance, the random sampling operations at the input layers of the content encoder and pitch encoder share the same random sampling factors. We find that by having the same random sampling factors, we can reduce the remaining rhythm information after the random sampling, and thus achieving better disentanglement.

We follow the downsampling implementation in AUTOVC. Suppose the downsampling factor is k and we use zero-based indexing of the frames. For the forward direction output of the bidirectional-LSTM, $t = kn + k - 1$, $n \in \{0, 1, 2 \cdots\}$ are sampled; for the backward direction, $t = kn$ are sampled. In this way, we can ensure the frames at both ends are covered by at least one forward code and one backward code.

B.3. Converting Pitch

During training, the input speech to the content encoder and the input pitch contour to the pitch encoder are always aligned (due to the shared random sampling factors), *i.e.* they share the same (contaminated) rhythm information $A(\boldsymbol{R})$. During pitch conversion, however, such alignment no longer exists, because the pitch contour is replaced with that of another utterance. To restore the temporal alignment, before we perform the pitch conversion, we first perform a rhythm-only conversion to the new pitch contour, where the conversion target is the input speech to the content encoder.

The rhythm-only conversion on pitch contour is essentially the same as the rhythm-only conversion on speech, except that we need to use a mini SPEECHFLOW variant that operates on pitch contour, not speech. Specifically, there are two major differences between the variant and the original SPEECHFLOW. First, the variant comes with only two encoders, the rhythm encoder and the pitch encoder, whose inputs are spectrograms and the corresponding pitch contours. The content encoder is removed because there is no content information in pitch contour. Second, rather than reconstructing speech, the decoder reconstructs pitch contour from the outputs of the encoders. The output dimension of the decoder at each time is the one-hot encoding dimension of the pitch contour (257), and the cross-entropy loss is applied. The hyperparameter settings are the same as in the original SPEECHFLOW. Following the same argument as in SPEECHFLOW, it can be shown that this variant can disentangle the pitch and rhythm information of pitch contour, and thus can perform the rhythm-only conversion.

B.4. General Guide on Tuning the Bottlenecks

Although tuning the information bottleneck dimensions is the most difficult part of getting SPEECHFLOW to work properly, there are some straightforward guidelines on how to choose the correct physical dimension of each code. The basic idea is that removing one of the four codes should reproduce the results in figure 8.

Specifically, when the rhythm code is set to zero, the output should be almost blank, as shown in the top-left spectrogram in figure 8. If the output still preserves significant speech energy, it means that the rhythm code dimension is too small. Consider increasing the dimension. Throughout this section, by increasing the dimension, we refer to two operations. The first operation is to increase the channel dimension of the encoder output. The second operation is to increase the sampling rate of the down-sampled code. Accordingly, by decreasing the dimension, we refer to the two opposite operations.

When the content code is set to zero, the output should become a set of slots with uninformative spectral shapes,

as shown in the top-right spectrogram in figure 8. If the output preserves *the same* formant structure as the input speech, it means that the content code dimension is too small, and that the rhythm code dimension is too large (the case where the rhythm bottleneck is too wide has been discussed in section 5.5). In this case, consider increasing the content code dimension, and decreasing the rhythm code dimension. Please note that the key is to compare the formant structure with that of the input speech. In some cases, removing the content would produce a speech-like spectrogram with clear harmonic and formant structures, instead of the aforementioned empty slots of uninformative spectral shapes. However, as long as the formant structure is drastically different from the input speech, the bottleneck setting is appropriate.

When the pitch code is set to zero, the output should become either a voiced spectrogram with a constant pitch and harmonic structure, as shown is the bottom-left spectrogram in figure 8, or an unvoiced spectrogram with no harmonic structure at all. If the output does not fall in either case, *i.e.* the output preserves the same pitch or voiced/unvoiced states as the input speech, it means that either the rhythm code or the content code is too wide (both cases have been discussed in section 5.5). If this happens, determine which case it is by setting the content code to zero, and then make adjustments accordingly. It is also worth mentioning that it does not harm to set a relatively large bottleneck dimension for pitch, because the information conveyed by the pitch contour is already very limited.

Finally, if the speaker identity is changed to another speaker, the output should sound like the target speaker. If it does not sound like the target speaker, it means either the rhythm code or the content code is too wide. Follow the aforementioned procedures to identify the problem. However, if there are no anomalies in the aforementioned diagnosis, they may be both too wide. Try decreasing them simultaneously. Conversely, if the converted speech is of very poor quality, it implies that both the rhythm code and the content code are too narrow. Try increasing them simultaneously.

B.5. Test Token Selection

The samples for subjective evaluations on pitch are hand-picked by authors. Authors listen to all the pairs in the test set and find the parallel pairs that are perceptibly different in pitch, e.g. rise vs fall tones, or high vs low tones, in at least one word. We then sub-select 20 pairs that maintain speaker diversity. For rhythm, we identify top 40 parallel pairs with greatest differences in time length, and then sub-select 20 pairs that maintain speaker diversity. For timbre, we simply randomly pick pairs from different speakers. Please note that the selection is based on the original speech only. They are not based on any conversion results.

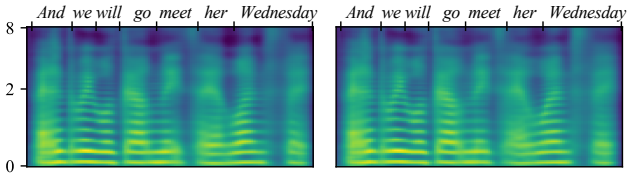

Figure 9. Reconstructed speech produced by AUTOVC with a random resampling module. The ground truth speech for the left column is in figure 7. The word boundaries and labels are copied from that of the ground truth.

C. Additional Experiment Results

C.1. Does Random Resampling Remove All Rhythm?

In figure 2 and section 4.4, we assume that the random resampling only contaminates rhythm information, but does not completely remove it. To verify this assumption, we train a single autoencoder for speech, where the encoder and decoder are the SPEECHFLOW content encoder and decoder respectively. If random resampling only removes a portion of the rhythm information, the output reconstruction can still roughly temporally aligned with the ground truth speech. Otherwise, the reconstruction will be completely misaligned.

Figure 9 shows two reconstruction results with different randomly drawn resampling factors, whose ground truth utterances are both the top-left panel of figure 7. To assist our judgment of the alignment, we directly copy the word boundaries and labels from the ground truth. As can be observed, the two reconstructions are very alike, even though their random resampling factors are different. Furthermore, both reconstructions can recover the ground truth speech decently, only with some minor blurring, which verifies that random resampling performs an *incomplete disentanglement* of rhythm. In other words, SPEECHFLOW shows that we can build a complete disentanglement mechanism even if we only have a partial disentanglement technique.

C.2. Do Rhythm Labels Exist?

In section 1, we have discussed that one motivation for designing SPEECHFLOW is that rhythm labels are not directly available. If they were, the rhythm aspect could be disentangled in much simpler ways. In this section, we would like to explore if there exist any rhythm labels.

We have identified two promising candidate rhythm labels, short-time energy and unvoiced-voiced (UV) label. The short-time energy is computed by taking the moving average of the squared waveform. The UV labels are derived from pitch contour, which equals one if the corresponding frame is voiced, and zero otherwise. Both candidates are informative of the syllable boundaries, and neither contains other information such as content and pitch. To test if

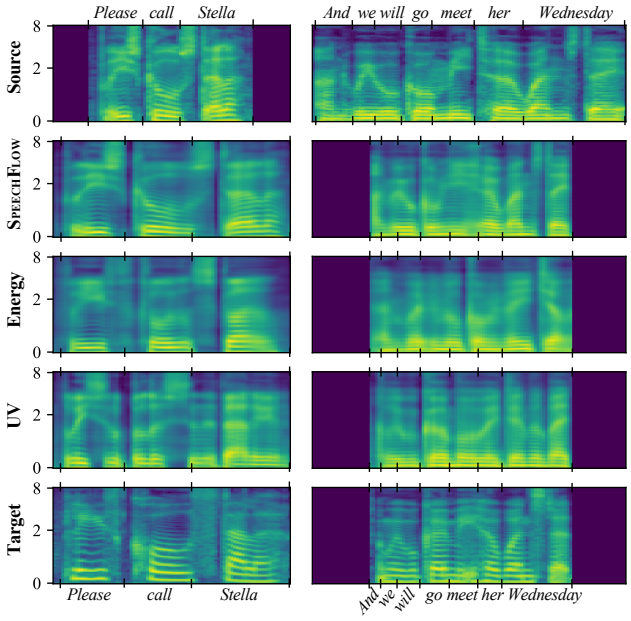

Figure 10. Rhythm-only conversion using the rhythm feature in SPEECHFLOW (second row) compared with that using candidate rhythm features, including short-time energy (third row) and UV label (fourth row).

these candidates are equally effective as the SPEECHFLOW rhythm encoder, we train two variants of SPEECHFLOW, one replacing the rhythm code with the short-time energy, and the other with the UV label. We then perform the rhythm-only conversion using SPEECHFLOW and the two variants, by replacing the rhythm code/label with that of the target speech. If the candidates are effective, the corresponding rhythm-only conversions should be successful.

Figure 10 shows the rhythm-only conversion results on two utterances, *'Please call Stella'* and *'And we will go meet her Wednesday'*, produced by these three algorithms. At first glance, all the conversion results are temporally aligned with the target speech, which seems to suggest that the rhythm aspect has been successfully converted. However, a close inspection into the formant structure of the candidate conversion results reveals that the content within each syllable is completely incorrect.

With the 'fill in the blank' perspective discussed in section 5.4, we can better understand why the candidate rhythm labels fail. Both candidates can accurately provide the temporal information of the syllable boundaries, and thus the blanks are correctly located in time. However, the candidates fail to provide the anchor information of what to fill in each blank, and that is why the conversion algorithms put the wrong content in the blanks. In summary, obtaining a rhythm label is a nontrivial task, because the rhythm label should contain some anchor information to associate each syllable with the correct content, while excluding excessive content to ensure content disentanglement. SPEECHFLOW, with a triple information bottleneck design, manages to obtain such an effective rhythm code, which contributes to a successful rhythm conversion.

C.3. Additional Conversion Spectrograms

In figure 11, we augment the spectrogram visualization results in section 5.2 (figure 4) with two additional utterances, *'One showing mainly red and yellow'* and *'Six spoons of fresh snow peas'*, and with all the conversion types (not just the single-aspect conversions) displayed. Consistent with the results shown in section 5.2, these additional results show that SPEECHFLOW can successfully convert the intended aspects to match those of the target speech, while keeping the remaining aspects matching the source speech. Remarkably, when all three aspects are converted, the converted speech becomes very similar to the target speech.

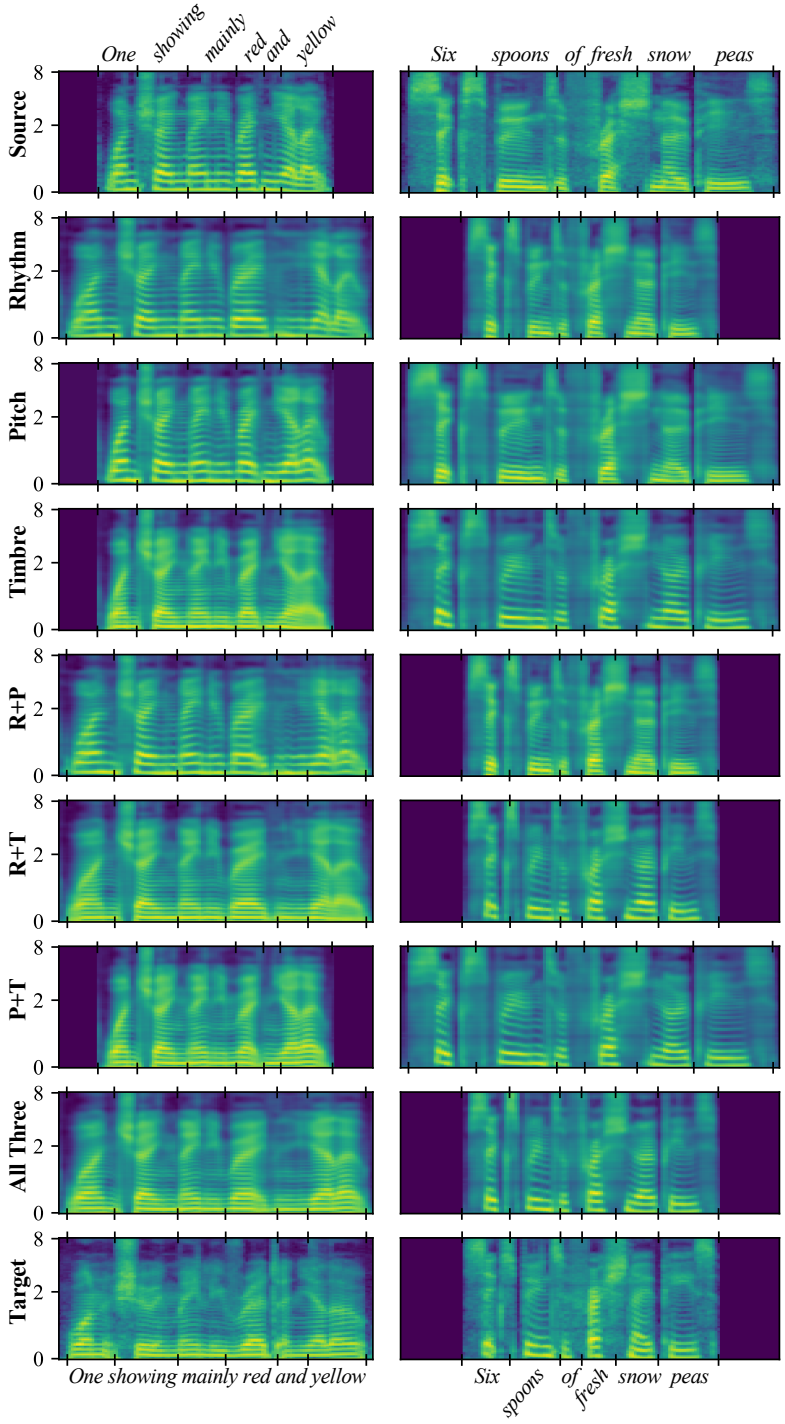

Figure 11. Spectorgrams of aspect-specific conversion results on two utterances, *'One showing mainly red and yellow'* (left) and *'Six spoons of fresh snow peas'* (right). R+P denotes rhythm+pitch conversion; R+T denotes rhythm+timbre conversion; P+T denotes pitch+timbre conversion.