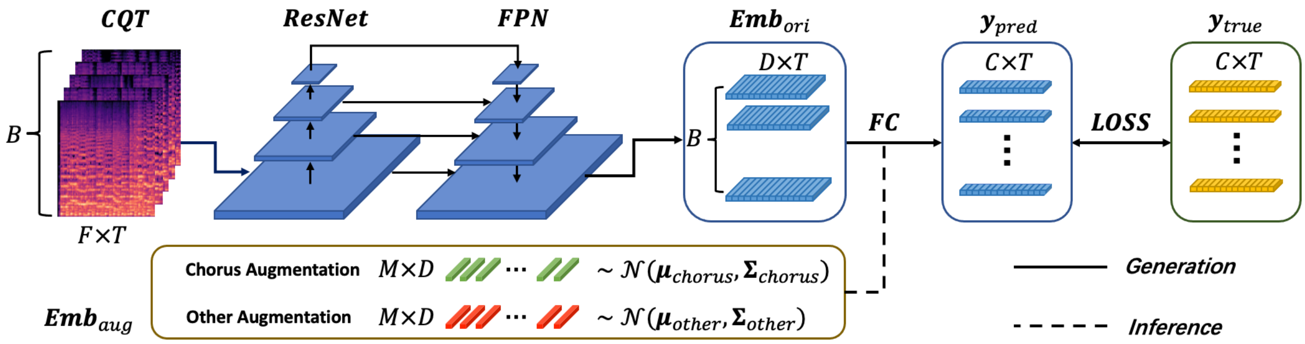

Figure 2. Model structure for LA-Chorus. The solid line represents the model generation processes in the forward pass, while the dashed line represents the inference (i.e. not explicitly generated in the forward pass).

2.2 Audio Data Augmentation

To enhance the diversity and variation in original audio features, traditional audio augmentation techniques such as time stretching, pitch shifting, noise perturbing and SpecAugment [34] are often implemented when transforming audio signals into spectrograms [14, 15]. In this paper, we take a different perspective: augmenting latent representations instead of original audio signals, motivated by the recent advancement of implicit augmentation methods in computer vision, represented by implicit semantic data augmentation (ISDA) and its variants [16, 17]. The ISDA model first used a backbone network to encode input images into latent space with semantic meaning, and then formulated a multi-variate Gaussian distribution for latent features in each class, which was estimated by their mean and covariance within the class via direct calculation. Then augmentations were sampled from the estimated distribution, and the model was later optimized by a novel cross-entropy loss tailored for latent augmentations. In this paper, we will demonstrate how this implicit augmentation method is utilized in audio to improve chorus detection.

3. PROPOSED METHOD

The structure of our proposed model is illustrated in Figure 2. We first use ResNet architecture to encode audio spectrograms into latent embeddings, Then, we apply the latent augmentation by sampling from estimated latent distributions for frames in the "chorus" class and the "non-chorus" (other) class respectively. Finally, the augmented representations are sent to a fully-connected layer to generate probability predictions. At learning and inference stages, we adopt a special cross-entropy loss that handles infinite number of latent augmentations via an upper bound, which saves the cost of sampling procedure.

3.1 ResNet-FPN

We first obtain the constant Q transform (CQT) spectrograms with F frequency bins and T frames in time domain after padding. Then a *ResNet-50* architecture is implemented as the embedding extractor $\mathcal{G}_\theta$ to extract latent embedding. Specifically, the ResNet consists of four stages that contains 3, 4, 6 and 3 residual CNN blocks respectively, where 64 convolution filters of 7×7 kernel size and a max-pooling layer of 3×3 kernel size are designed prior to these residual blocks to process the inputs.

With the size of each feature map reducing as CNN goes deeper, semantic information in deep audio features increases [8]. However, at the same time, the resolution of the feature map decreases and undermines the precision of chorus positioning. To solve this problem, we modify the backbone $\mathcal{G}_\theta$ by incorporating a Feature Pyramid Network (FPN) [8] into our ResNet architecture to construct latent features of high resolution from latent features with semantic information but of low resolution,as shown in Figure 2. The FPN design, different from the bottom-up ResNet part, takes a top-down approach that comprises four 1×1 convolutional layers that corresponds to four residual blocks of ResNet, which maps the low-dimensional outputs of ResNet to high-dimensional latent features with lateral connections.

As a result, the final latent representation $\mathbf{A}$ is of shape $T \times D$, where T is the number of frames and D is the number of latent dimensions. Because of the CNN and FPN designs in our backbone, latent embeddings not only contain both temporal and frequent information of the input audio, but also integrate feature maps of multiple resolutions to better locate chorus segments, further benefiting the latent augmentations later.

3.2 Latent Augmentations on Audio Features

To enrich variations, we apply latent augmentations on each representation $\mathbf{a}_i \in \mathbb{R}^D$ in the song, where $\mathbf{a}_i$ denotes the i_{th} row in $\mathbf{A}$. Similar to other latent variable models such as VAE variants [35] and flow-based models [36], we make a fundamental assumption that latent features within the same class follow the same latent distribution. Specifically, a latent augmentation $\tilde{\mathbf{a}}_i \in \mathbb{R}^D$ for latent feature $\mathbf{a}_i$ follows a multi-variate Gaussian distribution $\mathcal{N}(\mathbf{a}_i, \mathbf{\Sigma}_{y_i})$, where y_i indicates the label class ("chorus" or "other") for frame i, and $\mathbf{\Sigma}_{y_i} \in \mathbb{R}_+^{D \times D}$ represents the covariance matrix for class y_i. Then, we can sample from the distribution regarding to each $\mathbf{a}_i$ to get latent augmentations. In practice, a hyperparameter $\lambda > 0$ is imposed on the covariance matrix $\mathbf{\Sigma}_{y_i}$ to control the deviation of augmentations, which leads to the final distribution of augmented latent representation $\tilde{\mathbf{a}}_i$:

$$\tilde{\mathbf{a}}_i \sim \mathcal{N}(\mathbf{a}_i \,, \lambda \mathbf{\Sigma}_{y_i}). \tag{1}$$

To estimate the covariance matrix $\mathbf{\Sigma}$ for different classes, we mathematically calculate the covariance matrix estimate $\hat{\mathbf{\Sigma}}_c$ for class c (c is either "chorus" or "other") within the dataset:

$$\hat{\mathbf{\Sigma}}_c = \frac{1}{N_c} \sum_{i=1}^{N_c} (\mathbf{a}_i - \bar{\mathbf{a}})(\mathbf{a}_i - \bar{\mathbf{a}})^T, \tag{2}$$

where $\bar{\mathbf{a}} = 1/N_c \sum_{i=1}^{N_c} \mathbf{a}_i$ is the mean of latent features in class c, and N_c is the number of frames belong to class c. This covariance estimate is updated at each iteration after the generation of new latent features during training stage.

Finally, the augmented latent features $\tilde{\mathbf{A}}$ are sent to a fully connected layer with weight $\mathbf{W} \in \mathbb{R}^{D \times C}$ and bias $\mathbf{b} \in \mathbb{R}^C$ to generate the probability predictions, with $\tilde{\mathbf{a}}$ being row-vectors in $\tilde{\mathbf{A}}$:

$$\hat{\mathbf{y}} = \mathcal{F}_\phi(\tilde{\mathbf{A}}) = \tilde{\mathbf{A}}\mathbf{W} + \mathbf{b}. \tag{3}$$

In the next section, we will show an computationally efficient method for learning, which considers infinite augmentations but requires no explicit calculation of the augmented features $\tilde{\mathbf{A}}$.

3.3 Inference and Learning

Given a dataset of size N, a basic approach to formulate a loss function that treats each augmented latent features as a new sample point, and sample M augmentations for each latent feature, which results in an augmented dataset of $N \times (M+1)$ samples. Then we use cross-entropy loss function to train the model. This method is effective when M is relatively large, however, it is computationally expensive as we need to compute extra loss values for $N \times M$ augmentations.

Instead of computing the loss function with discrete augmentations, we adopt the loss function from [16] that incorporates latent augmentations from the continuous domain. For the final fully-connected layer, we denote $\mathbf{w}_c$ as the column vector in $\mathbf{W}$ and b_c as the bias element in b for class c, then the limit of the cross-entropy loss for M augmentations when M approaches to infinity equals:

$$\lim_{M \to \infty} -\frac{1}{N} \sum_{i=1}^{N} \frac{1}{M} \sum_{j=1}^{M} \log \Big(\frac{\exp(\mathbf{w}_{y_i}^T \mathbf{a}_i^{(j)} + b_{y_i})}{\sum_{c=1}^{C} \exp(\mathbf{w}_c^T \mathbf{a}_i^{(j)} + b_c)} \Big) \tag{4}$$

which is equivalent to calculating the expectation w.r.t. $\tilde{\mathbf{a}}_i$:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{\tilde{\mathbf{a}}_i} \Big[\log \Big(\frac{\exp(\mathbf{w}_{y_i}^T \tilde{\mathbf{a}}_i + b_{y_i})}{\sum_{c=1}^{C} \exp(\mathbf{w}_c^T \tilde{\mathbf{a}}_i + b_c)} \Big) \Big] \tag{5}$$

$$= \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{\tilde{\mathbf{a}}_i} \Big[\log \Big(\sum_{c=1}^{C} \exp(\tilde{\boldsymbol{\xi}}) \Big) \Big] \tag{6}$$

where $\tilde{\boldsymbol{\xi}} = (\mathbf{w}_c - \mathbf{w}_{y_i})^T \tilde{\mathbf{a}}_i + (b_c - b_{y_i})$.

Since $\log()$ is a concave function, from Jensen's inequality, we can show:

$$\mathcal{L}_{upper} = \frac{1}{N} \sum_{i=1}^{N} \log \Big(\mathbb{E}_{\tilde{\mathbf{a}}_i} \big[\sum_{c=1}^{C} \exp(\tilde{\boldsymbol{\xi}}) \big] \Big) \tag{7}$$

$$\geq \mathcal{L}. \tag{8}$$

As $\tilde{\mathbf{a}}_i$ follows $\mathcal{N}(\mathbf{a}_i \,, \lambda \mathbf{\Sigma}_{y_i})$ in Eqn. (1), and $\tilde{\boldsymbol{\xi}}$ is a linear transformation of $\tilde{\mathbf{a}}_i$, then $\tilde{\boldsymbol{\xi}}$ will also follow a Gaussian distribution:

$$\tilde{\boldsymbol{\xi}} \sim \mathcal{N}(\boldsymbol{\xi} \,, \Delta), \tag{9}$$

where $\boldsymbol{\xi} = (\mathbf{w}_c - \mathbf{w}_{y_i})^T \mathbf{a}_i + (b_c - b_{y_i})$ and $\Delta = \lambda (\mathbf{w}_c - \mathbf{w}_{y_i})^T \mathbf{\Sigma}_{y_i} (\mathbf{w}_c - \mathbf{w}_{y_i})$.

Given the moment generating equation $\mathbb{E}[\exp(tx)] = \exp(t\mu + \frac{1}{2}\sigma^2 t^2)$ for $x \sim \mathcal{N}(\mu, \sigma^2)$, we can express Eqn. (7) as follows:

$$\mathcal{L}_{upper} = \frac{1}{N} \sum_{i=1}^{N} \log \Big(\sum_{c=1}^{C} \exp(\boldsymbol{\xi} + \frac{1}{2}\Delta) \Big) \tag{10}$$

$$= -\frac{1}{N} \sum_{i=1}^{N} \log \Big(\frac{\exp(\mathbf{w}_{y_i}^T \mathbf{a}_i + b_{y_i})}{\sum_{c=1}^{C} \exp(\mathbf{w}_c^T \mathbf{a}_i + b_c + \frac{1}{2}\Delta)} \Big), \tag{11}$$

which gives us a tractable upper-bound of Eqn. (6)

Therefore, we do not need to explicitly sample augmentations from its distribution in Eqn (1). Instead, only the covariance matrix $\mathbf{\Sigma}_c$ of latent features for each class requires update at each iteration, which speeds up the convergence compared to discrete estimation.

The algorithm for learning the embedding extractor $\mathcal{G}_\theta$ and the fully-connected layer $\mathcal{F}_\phi$ is demonstrated in Algorithm 1 below. Because the model is underfitted at beginning epochs, the hyperparameter λ for controlling augmentation deviation is set to be $\lambda_0 \times \frac{\text{epoch}}{\text{total epoch}}$ to alleviate the impact of augmentation at the starting stage of training.

Algorithm 1 Algorithm for training LA-Chorus

Require: Padded CQTs batches; ResNet extractor $\mathcal{G}_\theta$; A fully connected layer $\mathcal{F}_\phi$; Initial covariance matrix $\mathbf{\Sigma}_c$ for each class; Initial λ_0 for scaling augmentation

```
for epoch = 1, 2, ..., I do
    for batch = 1, 2, ..., K do
        Encode the CQT batch into latent features
        {a_i}_{i=1}^T via ResNet-FPN extractor G_θ
        Update G_θ and F_φ by computing:
        ∇_{θ,φ} L_upper from Eqn (11)
    end for
    Update Σ_c across all batches with Eqn. (2)
    λ ← λ_0 × epoch/I
end for
return G_θ and F_φ
```

4. EXPERIMENTS

In this section, we conduct comprehensive experiments to evaluate LA-Chorus's performance against other state-of-

Dataset	#Songs	#Genres	#Lang.	#Quality
Di-Chorus	237	13	14	3

Table 1. Key statistics of Di-Chorus.

the-art (SOTA) methods in chorus detection. We first introduce the experimental setups, where details of our newly released dataset Di-Chorus and hyperparameter settings in LA-Chorus are discussed. Then we present the results of chorus detection by LA-Chorus and other methods on popular public datasets, followed by an ablation study that demonstrates the effectiveness of different modules in our proposed method.

4.1 Experimental Setup

For a fair comparison purpose, we conduct our experiment under the same settings in [5] with a cross-dataset paradigm (i.e. testing on different datasets that are not used in training). Specifically, we use 890 songs that contain chorus segments in *Harmonix* [12], along with 38 songs of Michael Jackson and 83 songs of The Beatles in *Isophonics* set [10] for training and validation. At testing time, we use three public datasets and our released dataset Di-Chorus to evaluate our model and other methods. The public datasets used for testing are: 100 "Popular" songs from *RWC* [37, 38], 210 "Popular" songs (denoted as *SP*) and 198 "Live" songs (denoted as *SL*) from *SALAMI* [11], which are chosen in consistence with the testing sets in [5].

Our newly released dataset, Di-Chorus [1] (denoted as DC), contains 237 music annotations of songs on YouTube labeled by experts. Compared to previous datasets mentioned above, songs in Di-Chorus are more easy-to-access from the appended YouTube URLs, and are more diversified since it consists of musics tracks in 14 languages as opposed to other existing datasets that are mostly in English (e.g., Harmonics) or just two or three languages (e.g., RWC and SALAMI). In addition, we also include three different recording qualities to improve the variation within dataset: Studio, Live and Original Sound Track (OST) which contains non-music segments. The key statistics are summarized in Table 1 below.

To demonstrate the performance of our proposed model on the above datasets, we compare LA-Chorus against the following methods:

- **CNMF** [39]: an unsupervised matrix factorization method from *MSAF* [40].
- **SCluster** [31]: a spectral clustering method based on frame co-occurrence matrix from *MSAF* [40].
- **Highlighter** [41]: an CNN model that takes an unsupervised approach to detect emotional highlights as chorus segments.

[1] We provide some demos of Di-Chorus in the supplementary material. Di-Chorus will be made publicly available upon acceptance for retrieval.

Models	AUC on Different Datasets			
	RWC	SP	SL	DC
CNMF	.526	.543	.478	.488
SCluster	.533	.545	.551	.568
Highlighter	.804	.703	.671	.553
Multi2021	.819	.675	.633	-
DeepChorus	.842	.780	.765	.811
LA-Chorus	**.906**	**.887**	**.831**	**.872**

Table 2. AUC results for chorus detection in various models.

Models	F1-score on Different Datasets			
	RWC	SP	SL	DC
CNMF	.403	.422	.340	.332
SCluster	.427	.448	.392	.603
Highlighter	.407	.303	.251	.283
Multi2021	.643	.473	.380	-
DeepChorus	.675	.611	.501	.662
LA-Chorus	**.728**	**.619**	**.526**	**.707**

Table 3. F1-score results for chorus detection in various models.

- **Multi2021** [3]: a CNN model based on a multi-task learning objective that jointly predicts chorus segments and their boundaries.
- **DeepChorus** [5]: a CNN model based on multi-scale networks and self-attention, which is the current state-of-the-art method for chorus detection.

Then, we validate the prediction results by AUC score (Area Under Curve) and F1 score. To evaluate these two metrics, we first create a sequence of the song length from the original annotation, with each element indicating the class of the corresponding segment. Then we can calculate AUC and F1 score for each song independently and take the average over them as the final result.

For the training details, we resample the audio at 22050 Hz and use CQT as our input feature with 12 bins per octave, where Han windowing function is applied with a hop size of 512 for extraction. The model is trained for 100 epochs with a batch size of 32 and a learning rate of 10^{-4} with a cosine decay scheduler. The code is implemented in PyTorch and run at a Tesla-V100-SXM2-32GB GPU.

4.2 Chorus Detection

We retain the experiment results for the chosen SOTAs on RWC, SP and SL from [5], and test them on Di-Chorus with the default settings in their papers, as shown in Table 2 and Table 3 by AUC score and F1-score respectively. Note we do not test Multi2021 [3] on Di-Chorus, since their code is not open-sourced.

For AUC metric, LA-Chorus outperforms other SOTAs on all datasets by a big margin. Compared to *DeepChorus* [5], which is considered as the current best method for chorus detection, our method improves the performance by over 0.06 across all datasets. The performances on F1-score also exhibit a similar pattern, where LA-Chorus generates better predictions over other models with a considerable improvement on each dataset. In particular, our model performs exceptionally well on the widely used *RWC* dataset that reaches to an AUC score of 0.906 and an F1-score 0.728. We give most of the credits to the implicit augmentation design in our model, and we illustrate this perspective in the ablation study section.

4.3 Ablation Study

To analyze the effectiveness of FPN and latent augmentation, we test our LA-Chorus by separate modules: 1) ResNet backbone only, 2) ResNet with FPN, and 3) ResNet with latent augmentation (denoted as + *LA*). To demonstrate the efficacy of applying latent augmentations over traditional audio augmentation techniques in the input space, we further show the results of applying 4) time stretching (denoted as + *TS*) and 5) pitch shifting (denoted as + *PS*) to the ResNet backbone, with the results summarized in Table 4 for AUC and Table 5 for F1-score below (Note we do not apply TS or PS in our proposed method).

Ablations	AUC on Different Datasets			
	RWC	SP	SL	DC
ResNet	.801	.773	.767	.751
ResNet + FPN	.865	.830	.767	.807
ResNet + LA	.882	.854	.824	.847
ResNet + TS	.818	.787	.765	.762
ResNet + PS	.822	.777	.789	.766
LA-Chorus	.906	.887	.831	.872

Table 4. AUC results for ablation study.

Ablations	F1-score on Different Datasets			
	RWC	SP	SL	DC
ResNet	.592	.415	.418	.588
ResNet + FPN	.648	.473	.478	.608
ResNet + LA	.692	.540	.516	.687
ResNet + TS	.576	.394	.365	.553
ResNet + PS	.590	.378	.395	.545
LA-Chorus	.728	.619	.526	.707

Table 5. F1-score results for ablation study.

From the results, we can observe that by incorporating FPN, we improve the vanilla ResNet backbone by a remarkable increase of over 0.05 on most of the datasets under both AUC and F1-score metrics, expect for *SALAMI-Live* where the result remains the same. Such findings indicate that FPN is an effective method to locate music segments by increasing the resolution of feature maps, which, without any augmentation, can already generate predictions that are comparative to *DeepChorus*.

On the other hand, when we apply implicit augmentations to latent features generated by ResNet (without FPN), significant improvements of over 0.10 are witnessed for both AUC and F1 scores on most datasets. The results are even notably better than that of the *ResNet+FPN* combination who contains important positional information, implying the dominant role of latent augmentation in the strong performance of LA-Chorus. The results from *ResNet+TS* and *ResNet+PS* further corroborate the benefit of leveraging latent augmentations for chorus detection. Although there are some effects after adopting these two traditional augmentation methods, their improvements on the original model seem incremental compared to that of *ResNet+LA*. Instead, the implicit augmentation method outperforms traditional augmentation methods by a significant margin for both metrics on each dataset, which implies a clear advantage for adopting latent augmentations.

4.4 Discussion of Limitation

Despite of the prominent performance of LA-Chorus, we believe there are still some potential limitations for future explorations. First, further investigation is needed to verify that the latent augmentations are realistic to human when transforming them back to input domain. One possible way is to train a reversed model (such as a decoder or flow-based model) that reconstructs latent features to the original inputs. Second, the AUC and F1 metrics might not measure whether the output is overfragmented or underfragmented. It's needed to design metrics that are more perceptually relevant for chorus detection task. Finally, we only focus on detecting chorus segments in this paper, whereas in MSA, there are other annotation types (e.g. verse, bridge, etc.) to be modeled [4, 33]. We believe LA-Chorus only requires minor modifications in the class dimension of latent augmentations (i.e. augmenting chorus, verse, bridge, etc.) before being applied to predict other label types in music structure analysis.

5. CONCLUSION

In this paper, we introduced a novel chorus detection model based on ResNet-FPN architecture with latent augmentations on audio features. The proposed method, different from traditional augmentation algorithms focusing on the input space, augments audio features in the latent space to explore semantic changes in audio data. Besides, we released a new diversified dataset, Di-Chorus, with expert annotations, which contains songs with 13 genres in 14 languages and 3 qualities. Comprehensive experiments have been conducted on public datasets and Di-Chorus, where LA-Chorus shows superior performance against other methods. Lastly, the effectiveness of different modules in LA-Chorus are validated by an ablation study. In the future, we plan to investigate more details on the semantic changes of audio data via latent augmentations and the extensibility of LA-Chorus to other MIR tasks.

6. REFERENCES

[1] J. van Balen, J. A. Burgoyne, F. Wiering, R. C. Veltkamp *et al.*, "An analysis of chorus features in popular song," in *Proceedings of the 14th Society of Music Information Retrieval Conference (ISMIR)*, 2013.

[2] M. Goto, "A chorus-section detecting method for musical audio signals," in *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03).*, vol. 5. IEEE, 2003, pp. V–437.

[3] J.-C. Wang, J. B. Smith, J. Chen, X. Song, and Y. Wang, "Supervised chorus detection for popular music using convolutional neural network and multi-task learning," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 566–570.

[4] J.-C. Wang, J. B. Smith, W.-T. Lu, and X. Song, "Supervised metric learning for music structure feature," in *International Society for Music Information Retrieval Conference*, 2021.

[5] Q. He, X. Sun, Y. Yu, and W. Li, "Deepchorus: A hybrid model of multi-scale convolution and self-attention for chorus detection," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022.

[6] K. Ullrich, J. Schlüter, and T. Grill, "Boundary detection in music structure analysis using convolutional neural networks," in *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, 2014.

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[8] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[10] M. Mauch, C. Cannam, M. Davies, S. Dixon, C. Harte, S. Kolozali, and D. Tidhar, "Omras2 metadata project 2009," in *In Late-breaking session at the 10th International Conference on Music Information Retrieval (ISMIR*, 2009.

[11] J. B. L. Smith, J. A. Burgoyne, I. Fujinaga, D. De Roure, and J. S. Downie, "Design and creation of a large-scale database of structural annotations." in *ISMIR*, vol. 11. Miami, FL, 2011, pp. 555–560.

[12] O. Nieto, M. McCallum, M. E. Davies, A. Robertson, A. M. Stark, and E. Egozy, "The harmonix set: Beats, downbeats, and functional segment annotations of western popular music." in *ISMIR*, 2019, pp. 565–572.

[13] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of big data*, vol. 6, no. 1, pp. 1–48, 2019.

[14] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Sixteenth annual conference of the international speech communication association*, 2015.

[15] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal processing letters*, vol. 24, no. 3, pp. 279–283, 2017.

[16] Y. Wang, X. Pan, S. Song, H. Zhang, G. Huang, and C. Wu, "Implicit semantic data augmentation for deep networks," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[17] S. Li, K. Gong, C. H. Liu, Y. Wang, F. Qiao, and X. Cheng, "Metasaug: Meta semantic augmentation for long-tailed visual recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 5212–5221.

[18] W. Liu, R. Li, M. Zheng, S. Karanam, Z. Wu, B. Bhanu, R. J. Radke, and O. Camps, "Towards visually explaining variational autoencoders," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[19] P. Agrawal and S. Ganapathy, "Interpretable representation learning for speech and audio signals based on relevance weighting," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2823–2836, 2020.

[20] Y.-J. Luo, K. Agres, and D. Herremans, "Learning disentangled representations of timbre and pitch for musical instrument sounds using gaussian mixture variational autoencoders," in *Proceedings of the 20th Society of Music Information Retrieval Conference (ISMIR)*, 2019, pp. 405–410.

[21] S. Huang, Q. Li, C. Anil, X. Bao, S. Oore, and R. B. Grosse, "Timbretron: A wavenet (cyclegan (cqt (audio))) pipeline for musical timbre transfer," in *International Conference on Learning Representations*, 2018.

[22] R. Yang, D. Wang, Z. Wang, T. Chen, J. Jiang, and G. Xia, "Deep music analogy via latent representation disentanglement," in *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*, 2019.

[23] W. Chai and B. Vercoe, "Music thumbnailing via structural analysis," in *Proceedings of the eleventh ACM international conference on Multimedia*, 2003, pp. 223–226.

[24] M. A. Bartsch and G. H. Wakefield, "To catch a chorus: Using chroma-based representations for audio thumbnailing," in *Proceedings of the 2001 IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics (Cat. No. 01TH8575)*. IEEE, 2001, pp. 15–18.

[25] M. Muller, N. Jiang, and P. Grosche, "A robust fitness measure for capturing repetitions in music recordings with applications to audio thumbnailing," *IEEE Transactions on audio, speech, and language processing*, vol. 21, no. 3, pp. 531–543, 2012.

[26] M. Cooper and J. Foote, "Automatic music summarization via similarity analysis." in *ISMIR*. Citeseer, 2002.

[27] J. Paulus, M. Müller, and A. Klapuri, "State of the art report: Audio-based music structure analysis." in *Ismir*. Utrecht, 2010, pp. 625–636.

[28] N. C. Maddage, C. Xu, M. S. Kankanhalli, and X. Shao, "Content-based music structure analysis with applications to music semantics understanding," in *Proceedings of the 12th annual ACM international conference on Multimedia*, 2004, pp. 112–119.

[29] G. Peeters, A. La Burthe, and X. Rodet, "Toward automatic music audio summary generation from signal analysis," in *ISMIR*, 2002, pp. 1–1.

[30] J. Paulus, "Improving markov model based music piece structure labelling with acoustic information." in *ISMIR*, 2010, pp. 303–308.

[31] B. McFee and D. P. Ellis, "Analyzing song structure with spectral clustering," in *Proceedings of the 15th Society of Music Information Retrieval Conference (ISMIR)*, 2014, pp. 405–410.

[32] X. Du, Z. Yu, B. Zhu, X. Chen, and Z. Ma, "Bytecover: Cover song identification via multi-loss training," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 551–555.

[33] G. Shibata, R. Nishikimi, and K. Yoshii, "Music structure analysis based on an lstm-hsmm hybrid model," in *International Society for Music Information Retrieval Conference*, 2020, pp. 15–22.

[34] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[35] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *International Conference on Learning Representations*, 2014.

[36] D. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *International conference on machine learning*. PMLR, 2015, pp. 1530–1538.

[37] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "Rwc music database: Popular, classical and jazz music databases." in *Ismir*, vol. 2, 2002, pp. 287–288.

[38] M. Goto *et al.*, "Aist annotation for the rwc music database." in *ISMIR*, 2006, pp. 359–360.

[39] O. Nieto and T. Jehan, "Convex non-negative matrix factorization for automatic music structure identification," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 236–240.

[40] O. Nieto and J. Bello, "Systematic exploration of computational music structure research," in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, 2016.

[41] Y.-S. Huang, S.-Y. Chou, and Y.-H. Yang, "Pop music highlighter: Marking the emotion keypoints," *Transactions of the International Society for Music Information Retrieval*, vol. 1, no. 1, 2018.

ACCOMONTAGE2: A COMPLETE HARMONIZATION AND ACCOMPANIMENT ARRANGEMENT SYSTEM

Li Yi[1,2] **Haochen Hu**[1,2] **Jingwei Zhao**[3] **Gus Xia**[1,2]
[1] Music X Lab, NYU Shanghai [2] MBZUAI
[3] Institute of Data Science, NUS
ly1387@nyu.edu, hh1933@nyu.edu, jzhao@u.nus.edu, gxia@nyu.edu

ABSTRACT

We propose AccoMontage2, a system capable of doing full-length song harmonization and accompaniment arrangement based on a lead melody. [1] Following AccoMontage, this study focuses on generating piano arrangements for popular/folk songs and it carries on the generalized template-based retrieval method. The novelties of this study are twofold. First, we invent a harmonization module (which AccoMontage does not have). This module generates structured and coherent full-length chord progression by optimizing and balancing three loss terms: a micro-level loss for note-wise dissonance, a meso-level loss for phrase-template matching, and a macro-level loss for full piece coherency. Second, we develop a graphical user interface which allows users to select different styles of chord progression and piano texture. Currently, chord progression styles include Pop, R&B, and Dark, while piano texture styles include several levels of voicing density and rhythmic complexity. Experimental results show that both our harmonization and arrangement results significantly outperform the baselines. Lastly, we release AccoMontage2 as an online application as well as the organized chord progression templates as a public dataset.

1. INTRODUCTION

Accompaniment arrangement is a difficult music generation task involving structured constraints of melody, harmony, and accompaniment texture. A high-quality arrangement could help with various downstream tasks and applications, such as compositional style transfer [1], automatic accompaniment [2], and score-informed source separation and music synthesis [3].

As one of the most promising arrangement systems, AccoMontage [4] uses a generalized template-based approach to first search for roughly-matched accompaniment phrases as the reference and then re-harmonize the selected reference via style transfer. It generates much more coherent results than purely learning-based algorithms, especially for full-length song arrangements.

However, AccoMontage is not yet a "complete" accompaniment generation system in the strict sense, as it still calls for chord input from users and cannot harmonize a melody. To this end, we develop AccoMontage2, a system capable of full-length song harmonization and accompaniment arrangement based on a lead melody by equipping AccoMontage with two extra components: 1) a novel harmonization module, and 2) a graphical user interface.

The main novelty of our system lies in the harmonization module. We first collect a high-quality chord progression dataset and re-organize the phrases with respect to different styles to serve as reference templates. Then, we use dynamic programming (DP) to generate structured and coherent chord progressions given a query lead melody with phrase annotation. Specifically, the DP algorithm optimizes a multi-level loss function consisting of three terms: 1) a micro-level loss for note-wise melody-chord matching, 2) a meso-level loss for phrase-template matching, and 3) a macro-level loss for the whole-piece coherency. The first term evaluates the dissonance between the melody and the candidate chords. The second term prefers chord progressions with the same length as the target melody phrases. The third term computes how well the candidate phrases connect with each other to form an organic whole. Experimental results show that both our harmonization and arrangement results significantly outperform the baselines.

In addition, we develop a graphical user interface which allows the user to select different styles of chord progression and piano texture. Currently, chord progression styles include R&B, Dark, Pop-standard, and Pop-complex. Piano texture styles include several levels of voicing density and rhythmic complexity.

We release the AccoMontage2 as an online application [2] as well as the organized chord progression templates as an open-source dataset.

In brief, the contributions of our paper are as follows:

- A complete system for full-length song harmonization and accompaniment arrangement;
- An effective harmonization algorithm with state-of-the-art performance;

[1] Codes and dataset at https://github.com/billyblu2000/accomontage2.

[2] Online GUI link at https://billyyi.top/accomontage2.

Attribution: L. Yi, H. Hu, J. Zhao, and G. Xia, "AccoMontage2: A Complete Harmonization and Accompaniment Arrangement System", in *Proc. of the 23rd Int. Society for Music Information Retrieval Conf.,* Bengaluru, India, 2022.

- A graphical user interface for controllable piano accompaniment generation.

2. RELATED WORKS

2.1 Melody Harmonization

Melody harmonization refers to the task of generating a harmonic chordal accompaniment for a given melody [5, 6]. It has been typically formulated as a prediction task, *i.e.*, to predict a sequence of chord labels conditioned on the lead melody. Recent mainstream methods range from hidden Markov models [7, 8] to deep neural networks [9–11]. Such models are typically trained to fit a groundtruth melody-chord mapping, but do not account for the fact that one melody can be harmonized with various styles in terms of genre, chord complexity, *etc*. In fact, the current state-of-the-art models [10, 11] only support simple triads and up to a few common seventh chords. Also, predictions are made locally, where neither phrase-level progression nor inter-phrase structures are explicitly considered.

In this paper, we re-formulate melody harmonization with a novel template matching approach. The usage of existing templates for music generation has been a popular idea. Existing template-based methodologies include learning based unit selection [2, 12], rule-based score matching [13, 14], and genetic algorithms [15]. In our case, we match the lead melody with chord templates from a library based on rule-based criterion and subject to user control. Such an idea is inspired by the fact that music producers tend to pick up off-the-shelf chord templates instead of harmonizing from scratch. In addition, they also have control on what style of the chords to use.

Existing template-matching attempts for harmonization typically focus on half-bar level [16]. In contrast, our model deals with phrase-level matching. We design three loss terms that measure melody-chord fitness at note-wise, intra-phrase, and inter-phrase levels, respectively. Our chord library is finely organized, supporting up to ninth chords with voice leading and various genres. Our model can therefore generate structured and coherent full-length chord progressions with different styles.

2.2 Accompaniment Arrangement

The task of accompaniment arrangement aims to generate an accompaniment conditioned on a given lead sheet (*i.e.*, a lead melody with chord progression). The quality of arrangement is related to chordal harmony, texture richness, and long-term structure. For this task, existing learning-based models often do well in harmony and texture, but are less capable of long-term generation [1, 17–21]. Previous template-matching models can easily maintain long-term structures, but suffer from fixed elementary textures and often fail to generalize [13–15].

To break such a dilemma, the AccoMontage system [4] introduces a generalized template-matching methodology, where phrase-level accompaniment templates are first searched by rule-based criterion, and then re-harmonized via deep learning-based style transfer. The search stage and the style transfer stage each optimize high-level structure and local coherency, thus guaranteeing the arrangement of high-quality accompaniment.

In this paper, we integrate our harmonization module with AccoMontage and upgrade it to a complete accompaniment generation model. An input melody is first harmonized with stylistic chord progression and then arranged with piano textures. With an additional GUI design, our model offers a flexible degree of control, including harmony styles and texture complexity.

3. METHODOLOGY

The system diagram of our AccoMontage2 system is shown in Figure 1. It can achieve full-length song harmonization and accompaniment arrangement. The input of the system is a query lead melody with phrase annotation. The harmonization module will first harmonize the melody. The generated chord progression will be sent into AccoMontage together with the original melody to arrange accompaniment. Lastly, a GUI is provided for users to adjust chord and accompaniment styles.

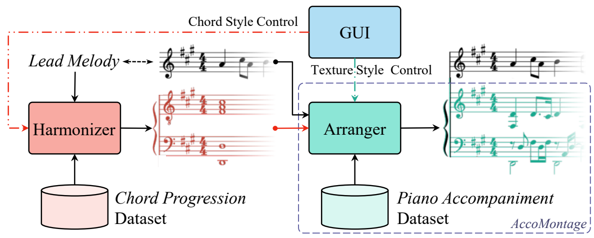

Figure 1: Diagram of AccoMontage2 system.

For the rest of this section, we first introduce the structure of the dataset and how we re-organize it in Section 3.1. Then we describe the harmonization module in Section 3.2. After that, we provide an overview of the AccoMontage system in Section 3.3. Finally, we show how the GUI is constructed to enable style controllability in Section 3.4.

3.1 Dataset Curation

A self-collected chord progression dataset is used as the reference templates for our harmonization algorithm. We create the dataset based on an existing chord progression collection [22] that contains 64,524 MIDI files, most of which are chord progression tracks with different style labels. The original dataset [22] cannot be adapted to our model directly for several reasons. First, some tracks are not pure chord progression, containing mixed melody segments. Second, many style labels are unnecessary. For example, two different styles may have similar musical elements that are hard to differentiate. Third, there are redundant progressions only different in their keys.

To solve the problems above, we process and re-organize the dataset as follows. First, we remove the MIDI files that contain melody segments and complex rhythmic textures. Second, based on our subjective assessment, we