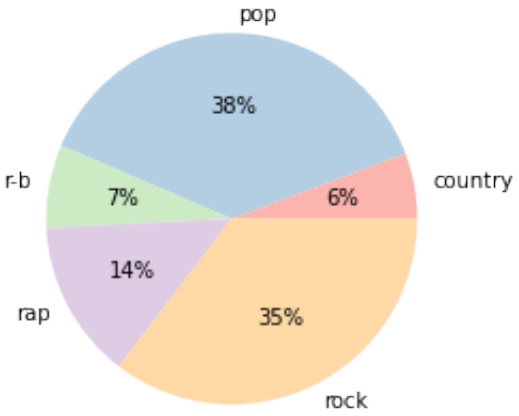

Figure 1: Genre distribution for primary genre.

2.2 Dataset Statistics

The final dataset consists of 295,416 songs created by 39,357 different artists. A summary of the dataset contents is given in Table 1. The total number of tokens (i.e., unigrams) contained in the choruses and verses for all songs is 193,696,032 and 195,567,571, respectively. This means that the amount of textual content for both chorus and verse is mostly balanced, ensuring that any difference in performance we may observe between using only chorus or verse does not stem from imbalanced training data. The distribution of primary genre labels in the dataset is given in Figure 1.

3. EXPERIMENTAL SETUP

The goal of this work is to determine whether there exists a difference in the predictive power between features extracted from different structural parts of a song's lyrics. For this, we carried out a series of experiments using different sets of textual features and different classification algorithms, each trained and evaluated once on only the *verse* and once only on the *chorus* parts of the songs in our dataset.

As explained in Section 2, we limited our final dataset to only songs which contain both at least one *chorus* and one *verse* part. This was done because, to get comparable results, we need to perform all our experiments on the same set of songs. As *chorus* and *verse* are the most common parts of a song, limiting our experiments to these two parts ensures that we have a dataset of sufficient size left to train and evaluate our models.

3.1 Feature Sets

Our experiments were performed using ten different sets of textual features commonly used in the literature (e.g., [2, 5–7, 9]). Those features together capture a wide variety of information about the lyrics, including content, grammatical structure, sound structure as well as complexity. Before using them in our experiments, we standardized them by scaling them to unit variance. We describe the feature sets used in the following:

- **Bag-of-words**: Classic bag-of-words features, counting the occurrences of word sequences of a given length, also known as n-grams, in the lyrics. In our experiments, we use unigrams, bigrams, trigrams, as well as (1,3)-grams, which is a combination of all word sequences of lengths one to three. To limit memory consumption of the resulting feature vectors, we only used the top 20,000 most common sequences appearing in the data. We also only consider sequences that appeared in at most 80% of all songs. This was done to filter out very common terms like function words. Such bag-of-words features have been frequently used in lyrics-based music classification, including for genre recognition, before (e.g. [2, 5–7, 9]).

- **Rhyme features**: A set of nine features describing the rhymes and alliterations present in the lyrics. The rhyme features (numbers of couplets, clerihews, alternating rhymes, and nested rhymes, percentage of rhymes in the text, and the number of unique rhyme words) were originally used by Mayer et al. [2] for lyrics-based genre recognition. Since alliterations play an important role in rap lyrics [10], we additionally added features describing alliterations (numbers of alliterations of length two, three, and four or longer).

- **Readability features**: A set of 13 readability metrics: Flesch reading easy, SMOG index, Flesch-Kincaid grade, automated readability index, Coleman-Liau index, Dale-Chall readability score, Linsear Write score, Gunning Fog index, Fernandez-Huerta score, Szigriszt-Pazos score, Gutierrez-Polini score, Crawford score, and the number of difficult words (i.e., words that are not on the Dale-Chall list of easy words). Readability features have been used for genre classification on normal texts, e.g. by Falkenjack et al. [11]. We included them in our experiments since we expect that they also carry useful information for lyrics.

- **Lexical features**: A set of 32 general lexical features, including token count, character count, repeated token ratio, number of unique tokens per line, average token length, average number of tokens per line, line count, unique line count, blank line count, blank line ratio, repeated line ratio, counts for specific symbols (exclamation mark, question mark, colon, etc.), count of digits, ratio of punctuation to the whole text, stop word count, stop word ratio, and hapax/dis/tris legomenon ratios. Different combinations of such lexical features have frequently been used for genre recognition (e.g., [2, 7, 9]).

- **Lexical diversity features**: A set of five lexical diversity metrics: measure of textual lexical diversity (MTLD), Herdan's C, Summer's S, Dugast's U and Maas' a.

- **Part-of-speech features**: A set of five features measuring the frequency of specific parts-of-speech within the lyrics: pronoun, adjective, adverb, noun, and verb frequency. Features describing the distribution of specific parts-of-speech within lyrics have been used for genre recognition for example by Mayer et al. [2], Mayer and Rauber [7], and Fell and Sporleder [5].

- **Morphological features**: A set of six features describing morphological properties of the lyrics: past tense ratio, -ing form ratio, comparative and superlative ratios for adjectives, and comparative and superlative ratios for adverbs. The ratio of verbs in past tense has been used for genre recognition before by Fell and Sporleder [5]. Since this was shown to be a valuable feature, we added the other five features to capture additional information about the morphological properties of a song's lyrics.

Since our goal is to determine potential differences between *verse* and *chorus* for individual feature types, we did not include a combination of all features in our experiments. It has already been shown before that different textual features carry orthogonal information, and combining them leads to increased performance [9].

3.2 Classification Algorithms

We repeat our experiments with different classification models to ensure that any difference in performance we measure between features extracted from different parts of a song is not only due to a specific property of any of the classification algorithms. Concretely, we chose the following algorithms: random forests, support vector machines, and feed-forward neural networks. All of these models are widely used in the machine learning community and have been shown to work well on many different tasks, including specifically genre recognition (e.g. [2–6, 9]).

For all three algorithms, we used the implementations provided by scikit-learn [1] version `1.0.2`. For support vector machines, we chose scikit-learn's `LinearSVC`, which uses LIBLINEAR [12], and followed the recommendation in the sklearn documentation to set `dual=False`, since in our case the number of training samples is significantly higher than the number of features. All other parameters for the used algorithms were left at their default values (100 estimators and Gini impurity for `RandomForstClassifier` and one hidden layer with 100 neurons and ReLU activation for `MLPClassifier`). Note that we did not perform a grid search to find the best hyperparameters, since our goal is only the determine the difference between features extracted from verses and choruses, not to get the best possible performance from the models.

[1] `https://scikit-learn.org/`

4. RESULTS AND DISCUSSION

We investigated every combination of song part (chorus or verse), feature set, and machine learning algorithm. Training and evaluation were done using 5-fold cross-validation. For the cross-validation, the data was shuffled before splitting it into folds. Feature extraction was done completely separately for the different song parts. For models trained and evaluated on the songs' verses, feature extraction only considered text located within the verses of the songs in the dataset, and equivalently for models trained and evaluated on the songs' choruses.

For evaluation, we chose the F_1 score to quantify the performance of each model. Since our dataset is imbalanced in terms of genres, we will focus our discussion on the macro-averaged F_1 scores, so as to not over-weigh the importance of the most common genres. However, for the sake of full transparency, we also report the micro-averaged F_1 scores for each model. Since the goal of our experiments is to determine whether there exists a performance difference between models trained on choruses and verses, we also performed statistical significance tests whenever our results indicated that the model trained on choruses performed better than the corresponding model trained on verses, or vice versa. For this, we employed a one-sided t-test ($p = .01$) on the scores of the individual cross-validation folds of both models, using the alternative hypothesis that the better overall score was indeed greater than the lower score.

We provide the results obtained in Table 2. We first take a look at the general performance of the models and feature sets. As is expected, we observe a substantial variance in performance for different machine learning models and features. The overall best performance in terms of the macro-averaged F_1 score is achieved by the support vector machine model trained on (1,3)-gram features extracted from the songs' verses, with an F_1 score of 0.5108. The best performance of the random forest and neural network model, respectively, was 0.4296 and 0.5082, both also for (1,3)-gram features trained on verses. The ten feature sets also exhibit substantial variability in performance across different machine learning models. This is especially true for part-of-speech features as well as morphological features. These two feature sets show a significantly lower performance when used with a support vector machine as opposed to random forests or neural networks. Looking at the higher score, obtained for the models trained and evaluated on the songs' verses, part-of-speech features achieved a score of 0.1872 against 0.2970 and 0.2888, and morphological features achieved a score of 0.1694 against 0.3154 and 0.2914.

Next, we examine the difference in performance between models trained and evaluated on choruses and verses, respectively. The results draw a clear picture: models trained and evaluated on verses consistently outperform the equivalent models trained and evaluated on choruses. We observe a difference in performance for every single combination of machine learning algorithm and feature set, at least in terms of the macro-averaged F_1 score. The

Feature Set	Random Forest Chorus	Random Forest Verse	Random Forest Diff.	Support Vector Machine Chorus	Support Vector Machine Verse	Support Vector Machine Diff.	Neural Network Chorus	Neural Network Verse	Neural Network Diff.
F_1 macro									
unigrams	.377 (±.002)	.428 (±.002)	.0506†	.414 (±.002)	.502 (±.002)	.0880†	.419 (±.003)	.505 (±.003)	.0858†
bigrams	.364 (±.000)	.416 (±.003)	.0522†	.396 (±.002)	.485 (±.001)	.0888†	.396 (±.003)	.479 (±.005)	.0824†
trigrams	.328 (±.002)	.385 (±.001)	.0562†	.333 (±.002)	.422 (±.001)	.0894†	.342 (±.003)	.419 (±.003)	.0772†
(1,3)-grams	.379 (±.002)	.430 (±.002)	.0502†	.432 (±.002)	.511 (±.002)	.0792†	.424 (±.002)	.508 (±.003)	.0846†
rhyme	.254 (±.002)	.318 (±.002)	.0638†	.200 (±.001)	.280 (±.001)	.0796†	.230 (±.004)	.307 (±.009)	.0776†
readability	.263 (±.001)	.371 (±.002)	.1078†	.224 (±.001)	.355 (±.001)	**.1308**†	.264 (±.006)	.360 (±.002)	.0964†
lexical	.359 (±.002)	.400 (±.002)	.0412†	.291 (±.002)	.363 (±.001)	.0720†	.360 (±.004)	.403 (±.001)	.0430†
lexical diversity	.253 (±.002)	.351 (±.001)	.0980†	.195 (±.000)	.319 (±.000)	.1242†	.245 (±.002)	.333 (±.001)	.0878†
part-of-speech	.223 (±.001)	.297 (±.001)	.0738†	.180 (±.000)	.187 (±.001)	.0068†	.207 (±.005)	.289 (±.002)	.0816†
morphological	.201 (±.002)	.315 (±.001)	**.1146**†	.164 (±.002)	.169 (±.000)	.0056†	.181 (±.003)	.291 (±.005)	**.1100**†
F_1 micro									
unigrams	.538 (±.002)	.577 (±.003)	.0396†	.526 (±.002)	.566 (±.002)	.0402†	.484 (±.003)	.548 (±.003)	.0644†
bigrams	.516 (±.000)	.559 (±.003)	.0432†	.504 (±.001)	.548 (±.000)	.0444†	.472 (±.003)	.532 (±.004)	.0602†
trigrams	.462 (±.001)	.516 (±.003)	.0534†	.460 (±.001)	.509 (±.001)	.0486†	.434 (±.004)	.490 (±.002)	.0566†
(1,3)-grams	.541 (±.002)	.581 (±.001)	.0402†	.525 (±.002)	.567 (±.002)	.0420†	.492 (±.003)	.552 (±.003)	.0598†
rhyme	.409 (±.002)	.449 (±.002)	.0404†	.425 (±.001)	.456 (±.002)	.0308†	.433 (±.002)	.461 (±.003)	.0286†
readability	.422 (±.001)	.499 (±.001)	**.0772**†	.439 (±.001)	.511 (±.002)	**.0716**†	.453 (±.001)	.513 (±.002)	.0596†
lexical	.486 (±.003)	.525 (±.001)	.0388†	.473 (±.003)	.519 (±.001)	.0464†	.493 (±.002)	.526 (±.001)	.0332†
lexical diversity	.363 (±.001)	.434 (±.002)	.0712†	.425 (±.002)	.478 (±.001)	.0522†	.376 (±.002)	.482 (±.002)	**.1068**†
part-of-speech	.392 (±.001)	.440 (±.001)	.0482†	.400 (±.001)	.406 (±.001)	.0054†	.408 (±.002)	.448 (±.002)	.0400†
morphological	.385 (±.001)	.435 (±.001)	.0502†	.388 (±.002)	.388 (±.001)	.0000	.396 (±.002)	.443 (±.002)	.0468†

Table 2: Summary of the results of our experiments. For all numbers, leading zeros were omitted for space reasons. *Diff.* is the change in F_1 score between verse and chorus. Numbers in parentheses are the standard deviation between the five cross-validation folds. Bold numbers indicate the highest and underlined numbers to lowest amount of change for each combination of feature set and machine learning algorithm. † indicates that the difference in performance is statistically significant.

micro-averaged F_1 scores show almost the same picture, with one notable exception: morphological features used with a support vector machine achieved the same score for *chorus* and *verse*. The exact difference in performance again varies depending on the machine learning algorithm and feature set. The biggest absolute difference in terms of macro-averaged scores is seen with support vector machines using readability features. Trained and evaluated on choruses, this model achieves an F_1 score of 0.2244, as opposed to a score of 0.3552 for verses. This constitutes an absolute change in score of 0.1308. The biggest relative difference is also shown by support vector machines, but for lexical diversity features. For this model, the score for verses is 0.3188 as opposed to 0.1946 for choruses, for a change of 0.1242, which is a relative change of 63.82%. The lowest difference in terms of macro-averaged F_1 scores is observed for morphological features used with support vector machines. The difference between *chorus* and *verse* for this model is 0.0056, which is also the lowest relative difference of 3.42%. For micro-averaged F_1 scores, the lowest change is 0, as mentioned before, also for morphological features using support vector machines.

To get a better sense of how the difference in performance depends on the used machine learning algorithm or feature set, we computed the average difference in macro-averaged F_1 scores between *chorus* and *verse* along both of these dimensions. The results for this are given in Table 3. We can observe from these that the performance difference

Average over ...	Average Difference
Machine Learning Algorithms	
Random Forest	0.0708
Support Vector Machine	0.0764
Neural Network	**0.0826**
Feature Sets	
unigrams	0.0748
bigrams	0.0745
trigrams	0.0743
(1,3)-grams	0.0713
rhyme	0.0737
readability	**0.1117**
lexical	0.0521
lexical diversity	0.1033
part-of-speech	0.0541
morphological	0.0767

Table 3: Average amount of change in the macro-averaged F_1 score between *chorus* and *verse*, averaged over machine learning algorithm and feature set. Bold numbers indicate the highest average differences.

	readability			lexical		
Genre	RF	SVM	NN	RF	SVM	NN
country	0.0054	0.0000	0.0011	-0.0448†	0.0001	-0.0076
pop	0.0298†	0.0367†	0.0360†	0.0110†	0.0271†	0.0193
r&b	0.0048	0.0000	-0.0021†	0.0271†	-0.0113†	-0.0583†
rap	0.4995†	0.6587†	0.4878†	0.3009†	0.3820†	0.2950†
rock	-0.0001	-0.0402†	-0.0412†	-0.0154†	-0.0383†	-0.0383

Table 4: Performance differences for the feature sets *readability* and *lexical* for individual genres in terms of F_1 scores. The values in the table are computed as the differences between chorus and verse, with positive numbers indicating that models trained on verses performed better, and negative numbers indicating that models trained on choruses performed better. † indicates that the difference in performance is statistically significant. Abbreviations: RF = random forest, SVM = support vector machine, NN = neural network.

does not seem to vary much with the machine learning algorithm. The biggest difference here is obtained by neural networks, with an average of 0.0826, while the smallest change is seen for random forests, with an average of 0.0708. In contrast, performance differences between feature sets are more substantial. The biggest average score difference there is produced by *readability*, with an average of 0.1117, while the smallest difference is seen for *lexical*, with an average of 0.0521.

Those results confirm our initial hypothesis: The predictive performance of genre recognition models is indeed significantly different depending on which parts of a song we extract the features from. Concretely, results show that features extracted from the verses of songs perform significantly better than those extracted from the choruses. As mentioned in Section 2, the amount of textual content in choruses and verses is almost identical for our dataset. We can therefore rule out that this difference is caused by different amounts of training data for the model. We also conducted our experiments with various machine learning algorithms and feature sets, showing that the changes in performance are also not due to any particular properties of specific features or algorithms. We can therefore conclude that these differences are caused by a more fundamental difference in information content between choruses and verses.

Finally, we aimed to investigate how this observed difference in performance depends on the concrete genre. To this end, we took the feature sets with the biggest and smallest difference — *readability* and *lexical* — and computed per-genre F_1 scores for them, again for the same three machine learning algorithms. We then, as before, computed the difference in performance between verse and chorus. These per-genre differences are given in Table 4. In this table, positive numbers indicate that the model trained on verses performed better, while negative numbers indicate that the models trained on choruses performed better.

Looking at these results, we can see a varied picture of the different genres. For *country*, we observe that the differences change between positive and negative, which indicates no clear tendency for whether verses or chorus perform better for this genre. The differences for countries are also not statistically significant, with one exception (random forests using *lexical* features). For *r&b*, there is also no clear tendency, with differences likewise varying between positive and negative. For *pop*, we consistently see better performance when using verse features, ranging from 0.011 to 0.0367, with five of the six observed differences being statistically significant. *Rap* is the genre for which we observe by far the largest difference in performance. Features extracted from verses clearly outperform those extracted from choruses for this genre, with differences between 0.295 and 0.6587, all of which are significant. Finally, for *rock*, we find the opposite behavior, with features extracted from choruses outperforming those extracted from verses. The statistically significant differences here range from -0.0154 to -0.0412.

5. CONCLUSION AND OUTLOOK

Building on earlier results that the structure of lyrics may play a role in genre recognition, we formulated the hypothesis that features extracted from different structural parts of a song lead to significant differences in predictive performance for genre recognition models. Through a series of experiments, we confirmed this hypothesis and showed that, depending on genre, features extracted from songs' verses can perform better than those extracted from choruses. Looking closer at how performance varies with the concrete genre, we found that this is especially true for the genres *pop* and *rap*. *Rock*, on the other hand, exhibits the opposite behavior, with classifiers using features extracted from choruses achieving better accuracy than those using features extracted from verses.

In future work, we will investigate whether we can find similar differences for other lyrics-based MIR tasks, like mood recognition or popularity prediction. Additionally, the observed differences might be used to construct better features or classification models which exploit the structure of song lyrics. Finally, we plan to incorporate audio-based features extracted from verses and choruses, respectively, and investigate whether those show similar behavior and whether they can complement lyrics-based features in a multi-modal classification setup.

6. REFERENCES

[1] B. L. Sturm, "A survey of evaluation in music genre recognition," in *International Workshop on Adaptive Multimedia Retrieval*. Springer, 2012, pp. 29–66.

[2] R. Mayer, R. Neumayer, and A. Rauber, "Rhyme and style features for musical genre classification by song lyrics." in *Proceedings of the International Conference on Music Information Retrieval*, 2008, pp. 337–342.

[3] T. C. Ying, S. Doraisamy, and L. N. Abdullah, "Genre and mood classification using lyric features," in *International Conference on Information Retrieval & Knowledge Management*. IEEE, 2012, pp. 260–263.

[4] A. Tsaptsinos, "Lyrics-Based Music Genre Classification Using a Hierarchical Attention Network." in *Proceedings of the 18th International Society for Music Information Retrieval Conference*, Oct. 2017, pp. 694–701.

[5] M. Fell and C. Sporleder, "Lyrics-based analysis and classification of music," in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, 2014, pp. 620–631.

[6] R. Neumayer and A. Rauber, "Integration of text and audio features for genre classification in music information retrieval," in *European Conference on Information Retrieval*. Springer, 2007, pp. 724–727.

[7] R. Mayer and A. Rauber, "Musical genre classification by ensembles of audio and lyrics features," in *Proceedings of the International Conference on Music Information Retrieval*, 2011, pp. 675–680.

[8] A. B. Melchiorre, N. Rekabsaz, E. Parada-Cabaleiro, S. Brandl, O. Lesota, and M. Schedl, "Investigating gender fairness of recommendation algorithms in the music domain," *Information Processing & Management*, vol. 58, no. 5, p. 102666, 2021.

[9] M. Mayerl, M. Vötter, M. Moosleitner, and E. Zangerle, "Comparing lyrics features for genre recognition," in *Proceedings of the 1st Workshop on NLP for Music and Audio (NLP4MusA)*, 2020, pp. 73–77.

[10] C. Coscarello, "The word out: A stylistic analysis of rap music," Master's thesis, 2003.

[11] J. Falkenjack, M. Santini, and A. Jönsson, "An exploratory study on genre classification using readability features," in *Proceedings of the Sixth Swedish Language Technology Conference (SLTC 2016), Umeå, Sweden*, 2016.

[12] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *The Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

TRANSFER LEARNING OF WAV2VEC 2.0 FOR AUTOMATIC LYRIC TRANSCRIPTION

Longshen Ou*, **Xiangming Gu***, **Ye Wang**
School of Computing, National University of Singapore
{longshen, xiangming, wangye}@comp.nus.edu.sg
*Both authors contributed equally to this research.

ABSTRACT

Automatic speech recognition (ASR) has progressed significantly in recent years due to the emergence of large-scale datasets and the self-supervised learning (SSL) paradigm. However, as its counterpart problem in the singing domain, the development of automatic lyric transcription (ALT) suffers from limited data and degraded intelligibility of sung lyrics. To fill in the performance gap between ALT and ASR, we attempt to exploit the similarities between speech and singing. In this work, we propose a transfer-learning-based ALT solution that takes advantage of these similarities by adapting wav2vec 2.0, an SSL ASR model, to the singing domain. We maximize the effectiveness of transfer learning by exploring the influence of different transfer starting points. We further enhance the performance by extending the original CTC model to a hybrid CTC/attention model. Our method surpasses previous approaches by a large margin on various ALT benchmark datasets. Further experiments show that, with even a tiny proportion of training data, our method still achieves competitive performance.

1. INTRODUCTION

Automatic lyric transcription (ALT) systems allow for lyrics to be obtained from large musical datasets without requiring laborious manual transcription. These lyrics can then be used for many music information retrieval (MIR) tasks, including query by singing [1], audio indexing [2], etc. Besides, because lyric alignment systems are typically built upon ALT models [3, 4], a strong-performing ALT model can lay a solid foundation for better audio-to-text alignment performance. Consequently, ALT is becoming an increasingly active topic in the recent MIR community.

One option for improving ALT performance is to incorporate knowledge obtained from studies involving the transcription of speech. Indeed, ALT is usually treated as a separate problem from automatic speech recognition (ASR), e.g., [3,5–8], eschewing large-scale speech datasets and well-developed ASR systems. However, the absence of large-scale singing datasets has impeded the construction of high-performing ALT models. While there are distinctions between sung and spoken language, e.g., sung language being less intelligible and hence harder to recognize [5, 9], they share many similarities, such as having the same vocabularies and being produced by similar physical mechanisms. Therefore, we believe it is worth investigating whether we can use knowledge and datasets from the speech domain to compensate for the inadequacy of singing datasets and bolster the performance of ALT systems.

Transfer learning methods have been found to effectively alleviate the requirement for a large amount of training data for some low-resource tasks [10,11]. For example, speech recognition for non-native speakers [12], and machine translation for low-resource languages [13]. In such scenarios, transfer learning helps mitigate the problem of insufficient data by adapting data and knowledge from related high-resource tasks or domains.

In recent years, self-supervised learning (SSL) has become a new paradigm in ASR research. Several SSL methods can perform excellently with access to only a few hours or even a few minutes of labeled data [14–16]. Among them, wav2vec 2.0 [16] has been shown to be a particularly promising model for transfer learning [12]. wav2vec 2.0 is an effective few-shot learner that only requires a small amount of data from the target domain or problem to achieve impressive results [17, 18]. This property makes wav2vec 2.0 a promising candidate to help ALT systems overcome the issue of limited training data by transferring speech representation knowledge to the singing domain.

The contributions of this paper contain four aspects:

- We propose an ALT solution that takes advantage of the similarities between spoken and singing voices. This is achieved by performing transfer learning using wav2vec 2.0 on singing data after pretraining and finetuning on speech data.
- We maximize the effectiveness of transfer learning by exploring the influence of different transfer starting points. We show that both pretraining and finetuning on speech data contribute to the high performance of our ALT system.
- We further enhance the system's performance by

extending the original connectionist temporal classification (CTC) model to a hybrid CTC/attention model for better convergence and more accurate decoding.

- Our method surpasses previous ones on various benchmark ALT datasets, including DSing [5], DALI [19, 20], Jamendo [21], Hansen [22], and Mauch [23], by about 25% relative WER reduction on average. We further show that with less than one-tenth of labeled singing data, our method can still achieve state-of-the-art results on the test split of DSing, demonstrating its effectiveness in low-resource ALT setups.

2. RELATED WORK

2.1 Automatic Lyric Transcription

Recent progress in lyric transcription has been mainly driven by three factors. First, the construction and curation of datasets containing aligned audio and lyrics, including DAMP Sing! 300x30x2 [5, 24] and DALI [19, 20], lay the foundation for data-driven ALT models. Second, the design of ALT acoustic models benefits from architectures of automatic speech recognition (ASR) models and can be further improved by adopting singing domain knowledge as inductive bias. Representative work includes TDNN-F with its variants [3, 5–7] and vanilla/convolution-augmented Transformers [8, 25, 26]. Additionally, [27] proposed to leverage the complementary information of additional modalities (video and wearable IMU sensors) for ALT systems. Third, through data augmentation methods such as adjusting speech data to make it more "song-like" [28] or synthesizing singing voice from speech voice [25, 26], more training data can be created for ALT models, thus alleviating the data sparsity problem.

2.2 Self-supervised speech representation learning

The success of deep learning methods is highly related to the power of the learned representations. Although supervised learning still dominates the speech representation learning field, it has several drawbacks. For example, substantial amounts of labeled data are required to train supervised learning ASR models [15, 16]. Moreover, representations obtained through supervised learning tend to be biased to specific problems, thus are difficult to extend to other applications [29].

To mitigate the above problems, a series of self-supervised learning (SSL) frameworks for speech representation learning have emerged, e.g., Autoregressive Predictive Coding (APC) [30], Contrastive Predictive Coding (CPC) [15], and Masked Predictive Coding (MPC) [14, 31, 32]. Moreover, wav2vec 2.0 takes advantage of both CPC and MPC to conduct self-supervised learning and has become the new paradigm for the ASR task [16]. wav2vec 2.0 has been also widely adopted as the feature extractor for other speech-related applications, e.g. speech emotion recognition [33, 34], keyword spotting [35], speaker verification and language identification [36], demonstrating that speech representations learned from wav2vec 2.0 are robust and transferable for downstream tasks.

3. METHODOLOGY

In this section, we firstly recap the structure of wav2vec 2.0 [16]. Then we elaborate on the three training stages of the proposed methods, including pretraining on speech data, finetuning on speech data, and transferring to the singing domain.

3.1 Structure of wav2vec 2.0

As shown in Fig. 1, wav2vec 2.0 is built with a CNN-based feature encoder, a Transformer-based context network, and a quantization module. For raw audio inputs $\boldsymbol{x}$ with a sampling rate of 16 kHz, the feature encoder accepts $\boldsymbol{x}$ and obtains the latent speech representations $\boldsymbol{z}$. The feature encoder has seven blocks, each of which includes a 1D temporal convolution with 512 channels followed by layer normalization [37] and GELU activation [38]. Consequently, $\boldsymbol{z} \in \mathcal{R}^{T\times 1024}$ are 2D representations with a frequency of 49 Hz. To exploit the temporal relationship among different frames of latent representations, $\boldsymbol{z}$ are further fed into the context network, which is parameterized by 12 Transformer blocks [39]. Each block has a multi-head attention module with 16 attention heads and a Feed-Forward Network (FFN) with 4,096 hidden dimensions. Resulting context representations $\boldsymbol{c} \in \mathcal{R}^{T\times 1024}$ are the features extracted from the audio signal and used for downstream tasks.

In addition to being fed into the context network, $\boldsymbol{z}$ are also accepted by a quantization module, which learns quantized speech representations $\boldsymbol{q}$, thus facilitating the self-supervised training.

3.2 Stage I: Pretraining on speech data

wav2vec 2.0 is pretrained through an SSL method [16] on large-scale unlabeled speech data, as displayed in Fig. 1(a). Before latent representations $\boldsymbol{z}$ are fed into the context network, several consecutive frame sequences are randomly masked. The masked frames are replaced by a trainable vector. wav2vec 2.0 is trained by optimizing the combination of contrastive loss and diversity loss $\mathcal{L}_m + \alpha\mathcal{L}_d$ (α refers to a balancing hyper-parameter). The contrastive objective is defined as:

$$\mathcal{L}_m = -\log \frac{\exp(sim(\boldsymbol{c}_t, \boldsymbol{q}_t)/\kappa)}{\sum_{\widetilde{\boldsymbol{q}}\sim \boldsymbol{Q}_t} \exp(sim(\boldsymbol{c}_t, \widetilde{\boldsymbol{q}}_t)/\kappa)} \quad (1)$$

where $\boldsymbol{c}_t$ is t-th frame of context representations, $\boldsymbol{Q}_t$ represents all possible quantized representations, the temperature value κ is set as 0.1 and sim refers to the cosine similarity. The diversity loss $\mathcal{L}_d$ is designed to encourage the usage of all entries in codebooks. We refer readers to [16] for more details.