A RECURSIVE SEARCH METHOD FOR LYRICS ALIGNMENT

Emir Demirel[1] **Sven Ahlbäck**[2] **Simon Dixon**[1]

[1] Centre for Digital Music, Queen Mary University of London, United Kingdom
[2] Doremir Music Research AB, Sweden

ABSTRACT

Audio-to-lyrics transcription and alignment requires strong acoustic and language models. Even in the presence of such models, the length of audio segments for decoding remains a challenge. In this year's MIREX submission, we present a recursive search method that splits the audio with respect to *anchoring* words for performing alignment on shorter audio segments. The recursion is applied by gradually restricting the language model and search space after each search iteration. We apply a final pass of forced alignment on the segmented audio to obtain timings for every word in the input song lyrics. According to initial experiments, our system is robust to various musical genres while being executable on local machines with low memory and computational resources.

1. INTRODUCTION

The performance of lyrics alignment in commercial music recordings with more than a few minutes of length heavily depends on how robust the model is against non-vocal segments. Especially in polyphonic music recordings, the presence of accompanying musical instruments may interrupt the alignment path and cause accumulated errors. Moreover, applying forced alignment on very large graphs may be memory exhaustive. For instance, a sequence of spectral frames spaced at 10ms intervals extracted from a 4-minute music recording would lead to 24000 state nodes in the graph. In speech recognition, typical values for the beam length when applying forced alignment does not exceed 20, as most utterances are shorter and shorter beam length is more efficient for decoding. On the other hand, long audio alignment requires a very large beam search, which can consume a huge amount of space in RAM.

In one of the latest successful systems, Stoller et al. [10] presented an end-to-end approach that aligns the input lyrics for non-overlapping audio chunks of maximum duration 10.22 seconds. Despite the great performance improvement compared to previous work, the end-to-end model requires a large training set (around 40 000 commercial music recordings, as noted in the paper). Gupta et al. [4] presented the state-of-the-art approach to lyrics alignment, which builds genre-aware pronunciation and acoustic models trained on an open source data set with polyphonic music [6]. One caveat in their approach is the large beam size required to align a full-length music recording with the corresponding complete lyrics. A large beam size may exhaust available memory during decoding in the presence of a large search space.

In our submission, we provide a system that can be run on environments with low computational resources and which also does not require a GPU. Moreover, our system uses an acoustic model trained on a publicly available data set with around 150 hours of a cappella (monophonic) recordings [1], a smaller data set compared to previously successful lyrics alignment systems [4, 10].

2. SYSTEM DETAILS

The proposed lyrics alignment pipeline is illustrated in Figure 1. In the back-end, we separate the vocal track from the original polyphonic mix and extract the vocal segments using energy-based voice activity detection (VAD). Initially beginning from these segments, we recursively search for words and their location in the vocal track. Once enough *anchoring* words are spotted, the music signal is segmented respectively and a final-pass forced alignment is applied to retrieve the timings of all the words. For robustness, the pronunciation model is constructed on the fly using a lexicon that is extended with respect to any out-of-vocabulary (OOV) words that might exist in the input lyrics.

2.1 Vocal Segmentation

First, we apply vocal source separation using *demucs* [2], a state-of-the-art open-source waveform-based music source separation tool. Compared to other music source separation tools [5, 11], which are mostly spectrogram-based models, *demucs* seemed to be more robust in retaining certain phoneme types like plosives and fricatives, even though there is recognizably more bleeding compared to other spectrogram-based approaches.

Once extracted, the vocal regions are determined based on log-energy which is calculated based on the zeroth component of MFCC features. We merge consecutive segments if the silence between them is less than 0.8 seconds

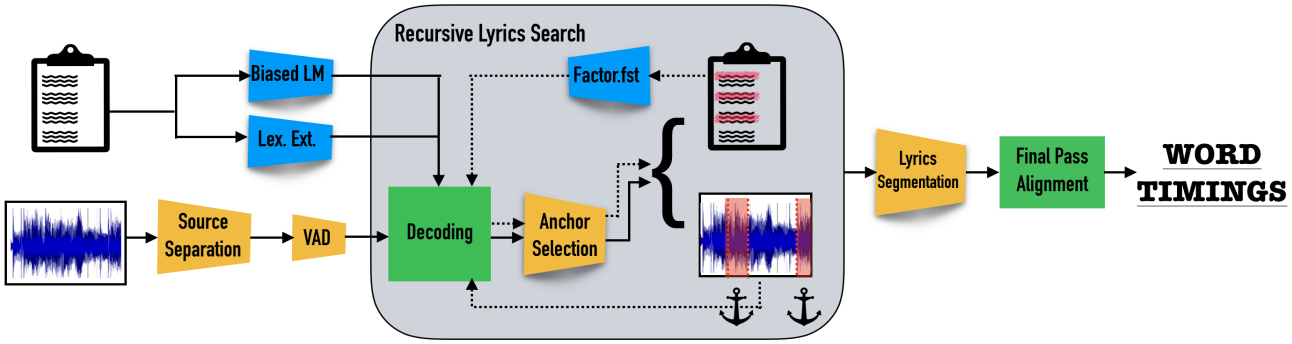

Figure 1. The pipeline for the overall system. Full lines indicate the initial and final passes, and the dashed lines are for the second and third iterations during the recursive lyrics search procedure.

although we don't merge segments that are already more than 6 seconds long.

2.2 Audio-to-lyrics Alignment

2.2.1 Pronunciation Model

For robustness during inference, all the words in the input lyrics have to be present in the vocabulary of the lyrics alignment system. In other words, the pronunciations of the input words have to be known. Therefore, the lexicon has to be extended on the fly to produce pronunciations for OOV words. From this point of view, new pronunciations are generated for OOV words using a pretrained grapheme-to-phoneme (g2p) model which learns the mapping between words and phonemes from an existing dictionary.

2.2.2 Recursive Decoding

As mentioned above, the main motivation for developing the system presented in this paper is to apply lattice decoding on smaller graphs through segmenting the whole music recording into shorter chunks. This is achieved via recursive search, a method similar to [8]. The procedure is applied through following steps:

- Initially, we transcribe the contents in VAD segments using a decoder with a biased 4-gram language model (LM) that is built using the input lyrics.
- Through text alignment, the anchoring words are spotted in the reference text and the transcript. For determining anchoring segments, a sequence of three words has to be transcribed correctly.
- The remaining parts of the audio are transcribed again, but this time using a factor transducer [7] for efficient decoding.
- The last step is applied once more, with a factor transducer with skip connections for allowing unspoken words in the vocal track to remain undetected. This is for robustness, in case there are words that are not separated clearly.
- After the last search iteration, audio is segmented with respect to anchoring segments, where each output segment is set to have at least 12 anchoring words.
- The previous step produces sentence-level audio segments, where corresponding lyrics are mapped accurately, even though the timing information is not necessarily obtained for all words. To get alignments for all words in the lyrics, we apply a final-pass alignment on these segments, given their mapped lyrics. At this step, we use a beam size of 50, even though it could be conveniently reduced to save computation time. This is important to notice in comparison to other methods that perform alignment on the complete recording.

2.3 Lyrics Transcription

In addition to the recursive lyrics alignment pipeline, we provide a transcription pipeline using the same phoneme-based acoustic model. Transcription is applied on the vocal segments extracted via Section 2.1. The decoding is first performed with the 4-gram LM presented in [3], which is trained on the lyrics of commercial English-language pop songs. After the first-pass decoding, we rescore lattices using RNNLM [12], which is trained on the same lyrics corpora.

3. EXPERIMENTAL SETUP

For all the decoding steps, we use the acoustic model presented in [3] which is based on a neural network architecture consisting of 2D convolutional layers in the back-end, which are added to provide more robust features for the following factorized time-delay layers. To refine the context dependency, a self-attention layer is added on top of the architecture. We use 40-band MFCC features as acoustic features and i-vectors [9] with a dimension of 100 as singer embeddings.

For the alignment challenge, we provide 4 different models: A GMM-HMM acoustic model is included providing a comparison with the previous year's submission to

observe how beneficial the recursive search is in the overall alignment pipeline. In addition to the phoneme-based model in [3], we also test the effectiveness of grapheme-based models trained using the same neural network architecture. Finally, a phoneme-based method is also included in the model set which does not have i-vectors in the feature space.

All of these acoustic models can be used for transcription via composition with a language and a pronunciation model. However, for simplicity, we provide composed decoding graphs for only phoneme- and grapheme-based models with i-vectors.

Running the pipeline with the different models mentioned above are explained in the *README.md* file provided in the submission.

4. CONCLUSION

Future improvements to our system would include limiting the search space during recursive decoding for achieving further efficiency in runtime, and building a more powerful acoustic model for better word-level alignments. Note that the acoustic model used for decoding is trained on monophonic singing recordings, which requires source separation in the pipeline. Hence, the performance of the source separation algorithm is crucial. On the other hand, using an acoustic model trained on polyphonic music would avoid the need for the source separation step.

In conclusion, we have presented a recursive lyrics search pipeline that could be leveraged for lyrics alignment and new sentence-level annotation generation for ALT. Our system provides a low-resource solution that is also generalizable to OOV words which could potentially exist in the input lyrics. Our initial experiments show that the system operates robustly for various genres of music.

Acknowledgments

The author E.D. received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 765068.

5. REFERENCES

[1] *Smule Sing! 300x30x2 Dataset*, (accessed August, 2020).

[2] Alexandre Défossez, Nicolas Usunier, Léon Bottou, and Francis Bach. Music source separation in the waveform domain. *arXiv preprint arXiv:1911.13254*, 2019.

[3] Emir Demirel, Sven Ahlbäck, and Simon Dixon. Automatic lyrics transcription with dilated convolutional networks with self-attention. In *International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020.

[4] Chitralekha Gupta, Emre Yılmaz, and Haizhou Li. Automatic lyrics transcription in polyphonic music: Does background music help? *arXiv preprint arXiv:1909.10200*, 2019.

[5] Romain Hennequin, Anis Khlif, Felix Voituret, and Manuel Moussallam. Spleeter: A fast and efficient music source separation tool with pre-trained models. *The Journal of Open Source Software*, 5(50):2154, 2020.

[6] Gabriel Meseguer-Brocal, Alice Cohen-Hadria, and Geoffroy Peeters. Dali: A large dataset of synchronized audio, lyrics and notes, automatically created using teacher-student machine learning paradigm. In *19th International Society for Music Information Retrieval Conference*, 2018.

[7] Pedro J Moreno and Christopher Alberti. A factor automaton approach for the forced alignment of long speech recordings. In *International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2009.

[8] Pedro J Moreno, Chris Joerg, Jean-Manuel Van Thong, and Oren Glickman. A recursive algorithm for the forced alignment of very long audio segments. In *Fifth International Conference on Spoken Language Processing*, 1998.

[9] George Saon, Hagen Soltau, David Nahamoo, and Michael Picheny. Speaker adaptation of neural network acoustic models using i-vectors. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*. IEEE, 2013.

[10] Daniel Stoller, Simon Durand, and Sebastian Ewert. End-to-end lyrics alignment for polyphonic music using an audio-to-character recognition model. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019.

[11] Fabian-Robert Stöter, Stefan Uhlich, Antoine Liutkus, and Yuki Mitsufuji. Open-Unmix: A reference implementation for music source separation. *Journal of Open Source Software*, 4(41):1667, 2019.

[12] Hainan Xu, Tongfei Chen, Dongji Gao, Yiming Wang, Ke Li, Nagendra Goel, Yishay Carmiel, Daniel Povey, and Sanjeev Khudanpur. A pruned RNNLM lattice-rescoring algorithm for automatic speech recognition. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018.