LYRICS ALIGNMENT USING HMMS, POSTERIORGRAM-BASED DTW, AND PHONEME-BASED LEVENSHTEIN ALIGNMENT

Anna Kruspe
Fraunhofer IDMT, Ilmenau, Germany
kpe@idmt.fhg.de

ABSTRACT

We present three approaches to lyrics-to-audio alignment for MIREX.
Approach A is based on Hidden Markov Models trained on the TIMIT speech data set, and uses the Hidden Markov Toolkit (HTK) and its included alignment algorithm.
Approaches B and C employ Deep Neural Networks (DNNs) for phoneme recognition. These networks were trained on a large amount of unaccompanied singing from the DAMP karaoke data set. They are used to generate phoneme posteriorgrams from the input audio.
For approach B, one-hot templates for the expected phoneme sequences are generated from the input lyrics data. Both these templates and the posteriorgrams are input into a Dynamic Time Warping (DTW) algorithm. This algorithm then attempts to find the best alignment between the posteriorgram and the binary template, which is then converted back into timestamp information.
Approach C also uses the posteriorgrams as its input, but then employs a mapping algorithm to produce plausible phoneme strings from them. These phoneme strings are then aligned with the expected sequence using a modified Levenshtein alignment.

1. HMM-BASED ALIGNMENT

For our first approach, we used speech data for training phoneme models. This data was taken from the commonly used *Timit* data set [1]. Monophone Hidden Markov Models (HMMs) were trained using the Hidden Markov Toolkit (HTK) [5]. Mel-Frequency Cepstral Coefficients (MFCC) were employed as features.
Then, the alignment algorithm from HTK is used to align the text lyrics to the audio. The employed dictionary is the the CMU Pronouncing Dictionary [1] .

2. POSTERIORGRAM-BASED APPROACHES

Both other approaches are based on phoneme posteriorgrams. These are generated from the input audio by first extracting MFCC features, and then running them through a Deep Neural Network (DNN) for acoustic modeling.
This DNN was trained on 6000 recordings of full songs performed by amateurs without accompaniment. These recordings are from the DAMP corpus [2] [4]. Since the data set does not contain phoneme annotations, these recordings were first annotated by automatically aligning the expected phonemes using the HMM-based approach. The whole process is described in [2].
An example of one of the generated posteriorgrams is shown in figure 1a.

2.1 DTW-based alignment

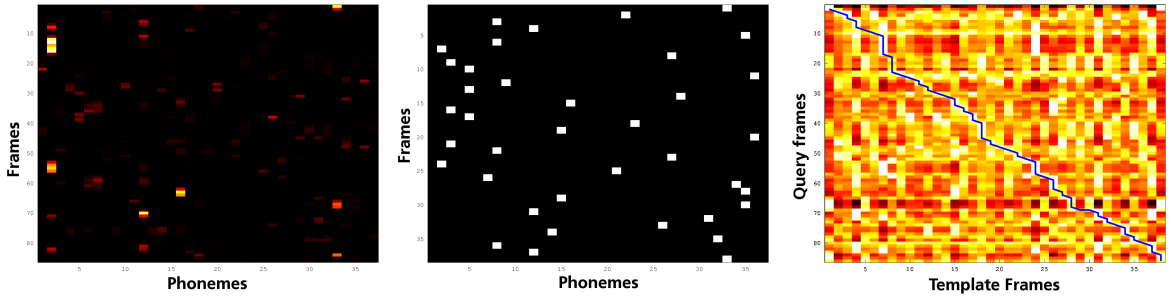
The first approach based on the phoneme posteriorgrams calculates a path between those and a binary template of the expected phoneme sequence. In order to do this, the input lyrics data is converted to phoneme sequences using the CMU Pronouncing Dictionary. Then, a one-hot matrix in the same formatting as the posteriorgram is generated from those (but, of course, with no timing information). An example of such a one-hot matrix is shown in figure 1b.
A similarity matrix between both matrices is then calculated using the cosine distance. Then, Dynamic Time Warping is performed on this matrix to obtain the cheapest warping path. An example of this is shown in figure 1c. Finally, we convert the result back into time frames.
This approach was presented in [3], where it was used to calculate the distances between a posteriorgram and a large set of lyrics templates in order to retrieve the most likely one.

2.2 Levenshtein-based alignment

The second approach based on the posteriorgrams does not operate on them directly, but rather attempts to convert them into plausible phoneme strings first.
In order to do this, the posteriorgram is first smoothed along the time axis using a short window. Then, the most likely phoneme for each frame is picked, the phonemes are grouped, and the durations and sum probabilities for each phoneme are calculated.
This commonly results in a much too long and noisy phoneme sequence, caused by mistakes that the classifier makes, and idiosyncrasies of the singer (i.e. performing a phoneme not exactly in the expected way, especially by

[1] http://www.speech.cs.cmu.edu/cgi-bin/cmudict

[2] https://ccrma.stanford.edu/damp/



(a) Phoneme posteriorgram

(b) Phoneme template

(c) Similarity matrix with cheapest path (blue)

Figure 1: Example of a DTW path calculation: Phoneme posteriorgrams are calculated for the audio recordings **(a)**. Phoneme templates are generated for the textual lyrics **(b)**. Then, a similarity matrix is calculated using the cosine distance between the two, and DTW is performed on it **(c)**.

transitioning through multiple phonemes). We therefore employ an algorithm that compresses this long sequence into a shorter one by discarding unlikely phonemes, taking into account the probabilities and the classifier's known confusions.

Ideally, this results in a phoneme sequence that is relatively similar to the expected one. However, we set the parameters in such a way that the sequence usually contains a few superfluous phonemes rather than discarding too many.

Then, we perform Levenshtein alignment between this generated phoneme sequence and the expected one. This algorithm is modified to also take the classifier's confusions into account for substitutions and insertions.

This results in a mapping from the generated phoneme string (containing time information) to the expected one, which we then convert back into timestamps.

3. REFERENCES

[1] J. S. Garofolo et al. TIMIT Acoustic-Phonetic Continuous Speech Corpus. Technical report, Linguistic Data Consortium, Philadelphia, 1993.

[2] A. M. Kruspe. Bootstrapping a system for phoneme recognition and keyword spotting in unaccompanied singing. In *17th International Conference on Music Information Retrieval (ISMIR)*, New York, NY, USA, 2016.

[3] A. M. Kruspe. Retrieval of textual song lyrics from sung inputs. In *INTERSPEECH*, San Francisco, CA, USA, 2017.

[4] J. C. Smith. *Correlation analyses of encoded music performance*. PhD thesis, Stanford University, 2013.

[5] S. J. Young, D. Kershaw, J. Odell, D. Ollason, V. Valtchev, and P. Woodland. *The HTK Book Version 3.4*. Cambridge University Press, 2006.