# Textless Syllable Alignment of Japanese Songs

**Alexander Gu** [1] **Alexander Siracusa** [1] **Khoi Pham** [1]

## Abstract

Transcribing Japanese song vocals into time-aligned Hiragana syllables is essential for applications like karaoke but remains labor-intensive. This project tackles textless syllable-level transcription by leveraging Japanese phonetics. We created a novel dataset from syllable-timed karaoke transcriptions of Japanese music videos and implemented a pipeline using Wav2Vec2 embeddings with a transformer-based model for syllable segmentation and LSTM for syllable prediction. Despite timing inaccuracies in the dataset affecting the segmentation model, the syllable prediction model achieved a micro F1 score of 0.535 across 110 Hiragana classes. Compared to prior methods that rely on transcripts or complex language models, our approach aimed for an adaptable solution tailored to Japanese phonetics. These results demonstrate the potential of textless transcription methods in specialized domains and guide future improvements in dataset precision and model architecture.

## 1. Introduction

We are interested in transcribing speech in Japanese songs to Hiragana script timed at the syllable level, and our motivation is two-fold. Firstly, this is a task that has been traditionally done manually in culturally-significant applications such as karaoke (Mazen, 2018), and we think this can be further automated for these use cases.

Secondly, we aimed to venture the relatively uninvestigated textless transcription approach in the context of general speech transcription (known as Automated Speech Recognition, or ASR). Our work aimed to align Japanese vocal audio with precise syllable-level Hiragana transcriptions. Unlike existing approaches that require transcripts or flexible language models, we sought to exploit the inherent structure of Japanese phonetics to simplify the transcription process.

### 1.1. Research contributions

An attempt has been made to apply some of the techniques used by Zhu et al. with a very simplified token set of size 110 and relatively simple neural architectures, for reasons described in Section 3.1.3. As directly attempting phone alignment has traditionally been hindered by the availability of time-precise data for the relevant domain, we are in a unique position upon coming across reasonably well-timed syllable-level karaoke transcriptions (as described in Section 3.1.2). Consequently, the dataset we curate for this task is novel, compared to well-known Japanese speech datasets that are not timed at the syllable-level (Maekawa, 2003; The Speech Database Committee et al., 2006).

Though the attempt was not successful, we gained insight into the implementation-specific limitations that contributed to the challenges faced.

## 2. Related Work

Though we are trying to align speech by syllable, within the context of ASR, the closest mapping to this task is phone alignment, which–as noted by Zhu et al. (2021)–has received relatively little attention, due to loss functions like CTC (Graves et al., 2006) rendering this task obsolete for downstream ASR tasks when paired with end-to-end (E2E) models by removing the requirement to have time-precise labels. It seems that the task of phone alignment itself is very difficult to tackle, as the time-precise labels it requires is very time-costly to produce from scratch (similar statement noted by Alharbi et al., p.131871). Within phone alignment, though, approaches like Forced Alignment exist (Bain, 2022; Japan7, 2024), but the reliance on transcripts hinders their adaptability to when the singer goes "off script" and to when transcripts don't exist. Approaches like textless (i.e., transcript-less) phone alignment, on the other hand, have been relatively unexplored (as noted by Zhu et al. and from personal observations). Zhu et al. uses this approach, but for Chinese and English (not Japanese). Adapting ASR techniques across problem domains (like from Chinese to Japanese) is known to be nontrivial (Alharbi et al., 2021, p.131867).

# 3. Proposed Method

## 3.1. Dataset

### 3.1.1. Structure

An audio file can be thought of as a sequence of $K \in \mathbb{Z}$ segments, denoted $\{s_k\}_{k=0}^{K}$. Let segment $s_k = (t_k, s_k, e_k)$, where $t_k \in T$ (the set of token classes) and $s_k, e_k$ non-negative real numbers denoting the start and end times of segment $k$. The set of tokens $T$ consists of 109 hiragana-based tokens and a dedicated silence token (see Appendix).

Some assumptions are made to simplify the problem. Firstly, the audio file contains just Japanese vocals (english is not supported), and the corresponding labels are in just Hiragana (see Section 3.1.3). Secondly, at any point in the audio file, there are no overlapping vocals (e.g., two people singing different verses at the same time).

### 3.1.2. Data Collection and Munging

With YT-DLP (yt dlp, 2024), we scraped 66 YouTube Japanese music videos with their syllable-level subtitle data (as WebVTT files). The subtitle data was then parsed into CSV files, with some entries filtered to meet the mentioned assumptions.

### 3.1.3. Domain-Specific Information

Japanese has three writing systems: hiragana, katakana, and kanji. Though all three see regular use and are intermixed in writing, for our purposes, we can transcribe all Japanese speech with just hiragana, since it is syllabary (each character represents a distinct syllable) and katakana and kanji characters can be mapped to hiragana characters. Through this view, the characters have a one-to-one mapping with their transcriptions (assuming clear enunciation), unlike with written English and Chinese, which suffer from homophones even with clear pronunciation. This suggests that Japanese syllable-level transcription should be achievable with simpler models and a smaller token set, compared to tried implementations with different languages.

### 3.1.4. Data Augmentation for Sparse Breaks

A significant challenge during training was the sparsity of syllable breaks, with only 4-6% of the 20-ms intervals corresponding to breaks. This imbalance made it easy for the model to achieve high accuracy by predicting "no break" for every timestep, a result that was practically useless.

To address this, we augmented the training data by adding "padding" around each syllable break. This approach created a gradient of values around the actual break points, as illustrated below:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Original: | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Padded: | 0 | 0.2 | 0.35 | 1 | 1 | 0.8 | 0.25 | 0 |

This augmentation encouraged the model to predict breaks more frequently and penalized it less for near-miss predictions. However, it introduced complications when breaks were close together, and necessitated a post-processing step to merge clustered predictions into single breaks. Despite these challenges, this method significantly improved practical performance.

Once the model was trained to predict break probabilities, we selected a threshold to classify breaks. While 0.5 is the default choice, we opted for a slightly lower threshold of 0.4 to further encourage break predictions. Any prediction exceeding 0.4 was classified as a break.

## 3.2. Pipeline

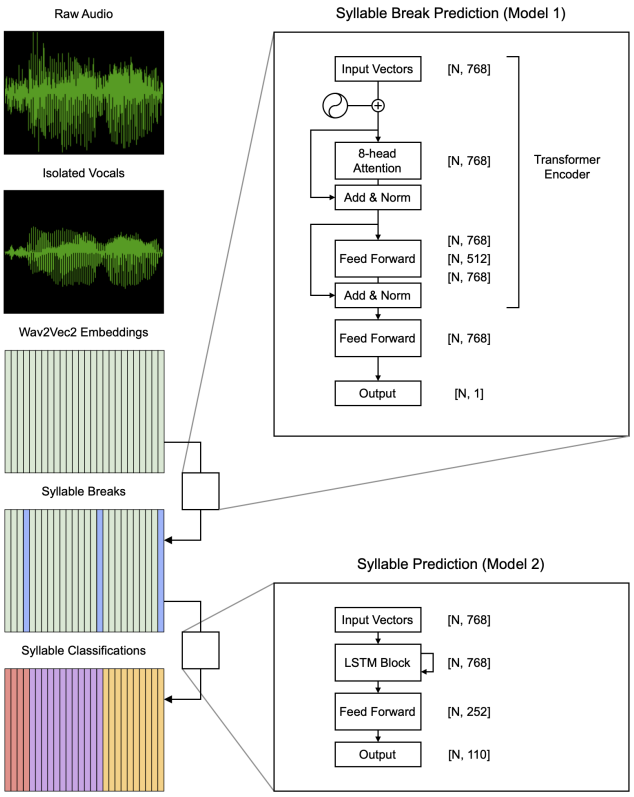


*Figure 1.* Pipeline Diagram
Model 1 diagram adapted from Vaswani et al.

### 3.2.1. Voice Isolation

We began by using an off-the-shelf voice isolation model (Anjok07, 2023) to extract vocals from the songs in our dataset. After manually inspecting the isolated tracks, we confirmed that no syllables were inadvertently erased, ensuring the integrity of our labels.

#### 3.2.2. WAV2VEC2 ENCODING

Next, we applied Wav2Vec2 (Baevski et al., 2020) to embed each 20-ms slice of the audio files as a 768-dimensional vector. Although 20-ms is the default interval for embedding, we experimented with smaller intervals by upsampling the original waveform as described by Zhu et al.. However, smaller intervals proved to be computationally expensive, and made identifying segment breaks more difficult. We chose to abandon them.

#### 3.2.3. SYLLABLE SEGMENTATION (MODEL 1)

For syllable segmentation, tried a number of different models using the same structure: For all elements in a series of 20-ms timesteps ( represented as a Wav2Vec2 vector) predict either "break" or "no break" to indicate the boundary between syllables. All of our models were trained and evaluated using binary cross-entropy (BCE) loss.

$$f(x_1, x_2, ...x_T) = (y_1, y_2, ...y_T) \tag{1}$$

Where $x_t$ represents a 20ms interval of a song, and $y_t$ is either 1 or 0 for "break" or "no break".

We tried many approaches using this structure. Detailed below are some of the ones we tried.

1. **LSTM:** Based on the assumption that identifying syllable breaks requires only a local context–typically a few seconds of audio–the limited long-term memory of the LSTM appeared sufficient.
2. **Fixed Offset Prediciton LSTM:** Instead of predicting breaks at the current timestep, the LSTM was tasked with predicting breaks occurring at a fixed interval (e.g., 0.5 seconds) earlier. This allowed the model to incorporate information from both before and after a break.
3. **Transformer Encoder:** Although this approach could be considered overkill given the local nature of syllable breaks, it outperformed the LSTM regardless.
4. **Spectrogram:** Upon visual inspection, we found syllable breaks aligned somewhat with spectrogram patterns. We tried most of the previous approaches using a CNN encoded spectrogram, with each column of pixels corresponding to 20-ms, the same as Wave2Vec2. This is inspired by the approach taken by (Zhu et al., 2021).

**Final Approach** We opted for Wave2Vec2 fed into a transformer encoder, as it performed the best in practice. While the spectrogram into transformer model performed similarly, we preferred the consistency offered from the fact that Model 2 also uses Wave2Vec2.

By iteratively refining our model architecture and data preparation techniques, we developed a robust approach to syllable segmentation. The combination of transformer encoders and augmented training data proved to be the most effective solution, significantly improving the model's ability to identify syllable breaks.

#### 3.2.4. SYLLABLE PREDICTION (MODEL 2)

Our syllable prediction model utilized an LSTM to take in the series of vectors that corresponded to a given syllable. Given a specific time interval, we feed in each of the Wav2Vec2 vectors that are contained within it in order, then at the last timestep predict the final classification. During training and evaluation, we use the ground truth segment breaks from our dataset, not those generated by model 1. In this way we are able to train and test our syllable prediction model independently of our segmentation model.

## 4. Experiment

In order to build and evaluate our models, we split up our dataset into a training set of the first fifty-nine songs and a test set of the last seven. Each of the proposed model architectures found in 3.2.3 and 3.2.4 where subsequently trained for ten epochs with a batch size of 1 song and 128 syllables for model 1 and 2 respectively. The syllable prediction model specifically was trained using teacher forcing from the ground truth syllable breaks, not those generated by the segmentation model.

We used binary and multi-class cross entropy loss to perform gradient descent for segmentation and syllable prediction respectively. However, as both the syllable breaks and syllable classes are heavily imbalanced across all timesteps, we chose to pick our final models based on F1 score calculated using the test set.

## 5. Results

### 5.1. Model 1 Results

Our segmentation model performed poorly despite trying all of our approaches mentioned in Section 3.2.3. This is especially problematic as when generating new subtitles without teacher forcing, our syllable prediction model relies on accurate enough segments to perform well. Our best model, the transformer encoder, was chosen using F1 score, which can be seen below along with other metrics. ("break" is positive, and "no break" is negative.)

$$\begin{bmatrix} 5228 & 8328 \\ 6854 & 49446 \end{bmatrix} \quad \begin{bmatrix} TP & FP \\ FN & TN \end{bmatrix}$$

$$\text{Precision} = 0.3857 \qquad \text{Recall} = 0.4327$$

$$\text{Accuracy} = 0.7827 \qquad \text{F1 Score} = 0.4078$$

### 5.2. Model 2 Results

Our syllable prediction model achieved the following results during testing.

$$\text{Accuracy} = 0.5362 \quad \text{F1 Score (Micro)} \quad = 0.5351$$

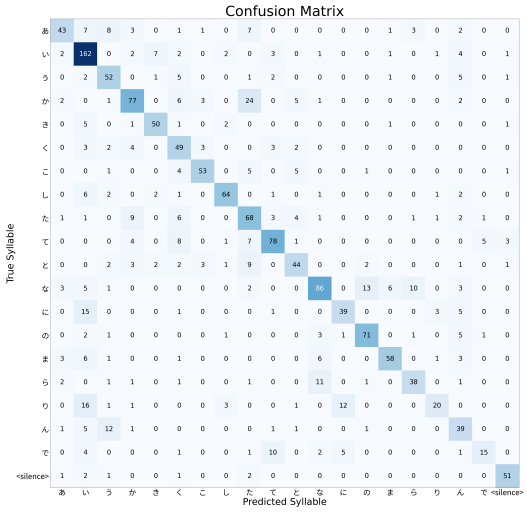


*Figure 2.* Syllable Classification Confusion Matrix

Considering the high number of total classes at 110, we believe this is a relatively successful result. Additionally, many of the mistakes are between similar sounding syllables such as ”wo” and ”o”, or ”ka” and ”ga”. Given how certain sounds are often left out or slurred while singing, some of this error may be unavoidable. Figure 2 shows a confusion matrix of the twenty most common syllables in our test set, as including more would impede readability.

## 6. Discussion

The below details the primary challenges stemmed from our dataset and approach.

### 6.1. Inherent Dataset Flaws

Though we initially found our dataset’s timings reasonable, during experimentation, we found that (1) the timings were not precise enough for our needs (at times up to 50 ms off) and (2) our implemented approach–basing predictions on 20-ms intervals–was very sensitive to these misalignments.

The labels also did not account for breathing, which was not filtered by our off-the-shelf vocal separation tool. As seen in Figure 3, this resulted in cases where, say, segments labeled “silent” would contain unlabeled vocal data.

We tried to address these limitations with both automated and manual data munging approaches, but have since deemed this too time-costly for the limited time-frame of this project.

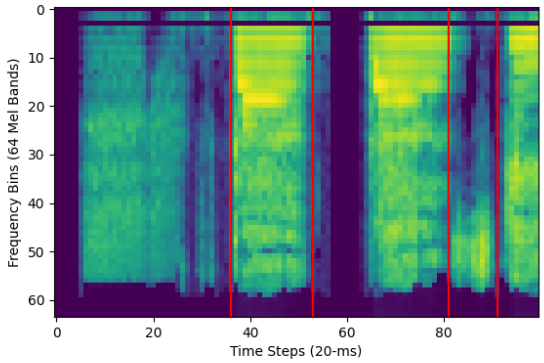


*Figure 3.* 64-Mel Spectrogram, where red lines indicate the labeled start and end times. Note the vocal signal before the leftmost red line corresponds to breathing.

### 6.2. Limitations of Textless Transcription

Even with a perfect dataset, there are still limitations inherent in our chosen approach due initially unaccounted linguistic phenomena. For example, singers will sometimes truncate or even skip the pronunciation of certain syllables, which further mismatches the labels and audio signals.

## 7. Conclusions and Future Work

### 7.1. Wav2Vec2 Fine-tuning using Contrastive Learning

We started, but did not finish, work on fine-tuning the Wav2Vec2 model (for syllable prediction) with contrastive learning. Doing so would allow us to leverage the much more abundant unlabeled audio data.

### 7.2. Advanced Loss Function

Based on our findings for Model 2, the BCE loss we used to evaluate its performance seems to undersell how well it does. It might be worth considering giving it “partial credit” for reasonable guesses, especially when at times a human might not be able to tell due to distortions in the audio.

### 7.3. Data Augmentation

Besides manually hacking away at the dataset’s timings, we could also leverage the many renditions (“covers”) of the more popular songs by different people, as the timing should more or less be the same.

## References


Alharbi, Sadeen, Alrazgan, Muna, Alrashed, Alanoud, Alnomasi, Turkiayh, Almojel, Raghad, Alharbi, Rimah, Alharbi, Saja, Alturki, Sahar, Alshehri, Fatimah, and Almojil, Maha. Automatic speech recognition: Systematic literature review. *IEEE Access*, 9:131858–131876, 2021. doi: 10.1109/ACCESS.2021.3112535.

Anjok07. ultimatevocalremovergui, 2023. URL https://github.com/Anjok07/ultimatevocalremovergui.

Baevski, Alexei, Zhou, Henry, Mohamed, Abdelrahman, and Auli, Michael. wav2vec 2.0: A framework for self-supervised learning of speech representations. *CoRR*, abs/2006.11477, 2020. URL https://arxiv.org/abs/2006.11477.

Bain, Max. Whisperx: Automatic speech recognition with word-level timestamps (& diarization), 2022. URL https://github.com/m-bain/whisperX.

Graves, Alex, Fernández, Santiago, Gomez, Faustino, and Schmidhuber, Jürgen. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pp. 369376, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595933832. doi: 10.1145/1143844.1143891. URL https://doi.org/10.1145/1143844.1143891.

Japan7. Forced alignment for karaokes, 2024. URL https://github.com/Japan7/yohane.

Maekawa, Kikuo. Corpus of spontaneous japanese: its design and evaluation. In *ISCA/IEEE Workshop on Spontaneous Speech Processing and Recognition*, pp. paper MMO2, 2003.

Mazen. Lyrics mad violet evergarden op full sincerely, 2018. URL https://www.youtube.com/watch?v=rh-xfHTJp6M.

The Speech Database Committee, The Acoustical Society of Japan, (Newspaper article data : the Mainichi Newspapers Co., Ltd ), The Acoustical Society of Japan, and The Mainichi Newspapers Co., Ltd. 日本音響学会新聞記事読み上げ音声コーパス（JNAS）, September 2006.

Vaswani, Ashish, Shazeer, Noam, Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan N., Kaiser, Lukasz, and Polosukhin, Illia. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

yt dlp. A feature-rich command-line audio/video downloader, 2024. URL https://github.com/yt-dlp/yt-dlp.

Zhu, Jian, Zhang, Cong, and Jurgens, David. Phone-to-audio alignment without text: A semi-supervised approach. *CoRR*, abs/2110.03876, 2021. URL https://arxiv.org/abs/2110.03876.


## A. Appendix - Token set

The token set T = {あ,い,う,え,お,か,き,く,け,こ,さ,し,す,せ,そ,た,ち,つ,て,と,な,に,ぬ,ね,の,は,ひ,ふ,へ,ほ,ま,み,む,め,も,や,ゆ,よ,ら,り,る,れ,ろ,わ,を,ん,が,ぎ,ぐ,げ,ご,ざ,じ,ず,ぜ,ぞ,だ,ぢ,づ,で,ど,ば,び,ぶ,べ,ぼ,ぱ,ぴ,ぷ,ぺ,ぽ,きゃ,きゅ,きょ,しゃ,しゅ,しょ,ちゃ,ちゅ,ちょ,にゃ,にゅ,にょ,ひゃ,ひゅ,ひょ,みゃ,みゅ,みょ,りゃ,りゅ,りょ,ぎゃ,ぎゅ,ぎょ,じゃ,じゅ,じょ,びゃ,びゅ,びょ,ぴゃ,ぴゅ,ぴょ,でぃ,ふぁ,ふぃ,ふぇ,ふぉ, ⟨silence⟩}.