A Comparison of Singing Evaluation Algorithms

Partha Lal*

Centre for Speech Technology Research,
University of Edinburgh, UK
www.cstr.ed.ac.uk

Abstract

This paper describes a system that compares user renditions of short sung clips with the original version of those clips. The F_0 of both recordings was estimated and then Viterbi-aligned with each other. The total difference in pitch after alignment was used as a distance metric and transformed into a rating out of ten, to indicate to the user how close he or she was to the original singer.

An existing corpus of sung speech was used for initial design and optimisation of the system. We then collected further development and evaluation corpora — these recordings were judged for closeness to an original recording by two human judges. The rankings assigned by those judges were used to design and optimise the system. The design was then implemented and deployed as part of a telephone-based entertainment application.

Index Terms: automated singing evaluation, pitch tracking, entertainment applications

1. Introduction

This paper describes the design and evaluation of an entertainment application centred around the comparison of sung speech. The increased popularity of televised singing contests means that more and more people are interested in trying out an automated telephone-based version. Apart from the current application, this approach could perhaps be used for language learning in intonational languages, although that that possibility remains unexplored.

We produced a telephone-based system with a call-flow that can be described as:

1. The system plays a music clip containing some sung lyrics. Clips are typically around 30 seconds long and the songs used were well known pop songs.
2. After the clip has finished playing, the caller attempts to impersonate the original singer. The system plays the backing music of the clip to help the caller with tempo.
3. The system then compares the original sung clip with the caller's version and assigns a score out of ten — a higher score means the caller was closer to the original.

The problem described here is that of comparing the caller's singing with the original recording to produce a score which relates to how closely the caller impersonated the original.

The contents of this paper are as follows — Section 2 lists some related work, Section 3 describes the two main designs examined, Section 4 examines the evaluation methods employed to compare those systems and Section 5 details experiments performed using those evaluation metrics.

*This work was done whilst the author was working for Vox Generation Ltd. The author would like to thank Dr. Hans Dolfing, Kerry Robinson and Dr. Simon King for their help and advice.

2. Related Work

Saul et al. [1] demonstrate a similar application to ours, although it differs in that the user receives *real-time* feedback on their pitch. Consequently their pitch determination algorithm is also quite different to ours.

Aucouturier and Pachet [2] studied music similarity in the context of navigating through recordings in a music database — songs were compared in terms of a timbre similarity measure. With a frame size of 50ms, eight Mel Frequency Cepstral Coefficients were extracted and a Gaussian Mixture Model, consisting of three Gaussians, was formed over those features for each song. The distance between two songs was then defined using the probability of samples generated with one song model, given the other song model (the process was then reversed to make the measure symmetric). This work differs from theirs in that it operates on shorter recordings and focuses only on the pitch of vocals.

3. Approaches

There were two main designs considered, both of which involved comparing pitch estimates of the recordings taken at various intervals. Pitch estimates were made using the "Robust Algorithm for Pitch Tracking" (RAPT) [3] as implemented in SFS [4], also known as `get_f0`.

The score, s_p, returned by each system is a non-negative number such that a higher score means a greater difference between the clips. Identical clips score zero.

Another feature common to both approaches is that the reference clip was manually cleaned up to remove non-vocal audio. Ideally this stage would not be needed, given access to separate vocal and musical tracks. Failing that, source separation techniques such as in [5] could have been applied, but given the time available non-vocal regions were manually replaced with silence.

3.1. Simple Pitch Comparison

This was the first approach attempted. Each recording was represented by taking an evenly spaced selection of 50 pitch samples and linearly interpolating across unvoiced regions. Since the clips are now essentially vectors, the Euclidean distance between a caller's clip and the reference clip was defined to be the caller's score. Various parameters of that algorithm were estimated using the sung corpus collected in [6], which is described in Section 4.1.

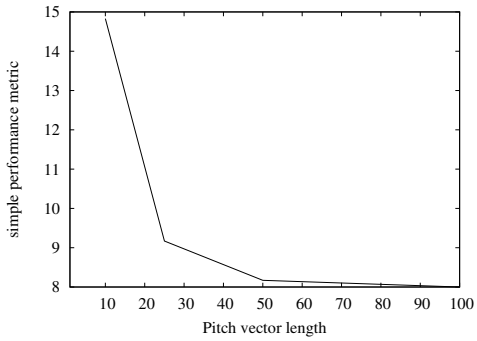

Figure 1: Plot of results in Table 1

#samples	mean $eval_1$
10	14.83
25	9.17
50	8.17
100	**8**

Table 1: The relationship between the length of the pitch vector, i.e. number of samples taken, and the simple performance metric of mean $eval_1$.

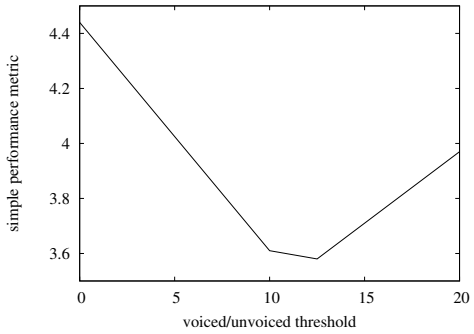

Figure 2: Plot of results in Table 2

threshold%	mean $eval_1$
0	4.44
10	3.61
12.5	**3.58**
20	3.97

Table 2: Finding the optimal voiced/unvoiced threshold.

3.2. Dynamically-aligned Pitch Comparison

One of the failings of the simple approach was that it was overly sensitive to the caller falling slightly out of time compared to the reference clip. This approach compensated for that by performing a Viterbi alignment of the caller's pitch curve and the reference pitch curve. The algorithm was modified so as to penalise substitution errors by an amount proportionate to the magnitude of the pitch difference. This was based on [7].

Given an array r of R pitch estimates of the reference recording and a similar array c , C elements in length, for the caller's recording, the score for c against reference r is $m(R,C)$. The following recursion, where $1 \leq i \leq R$ and $1 \leq j \leq C$, gives us the score

$$m(i,j) = \min \begin{cases} m(i-1,j-1) + d(i,j) \\ m(i-1,j) + p \\ m(i,j-1) + p \end{cases} \quad (1)$$

where

$$d(i,j) = |r_i - c_j| \quad (2)$$

The value of p was optimised using the evaluation metric described in Section 4.2.

There are a number of other differences between this and the first approach. In the first approach only 50 pitch samples were used (regardless of clip length) whilst here pitch estimates were made every 100ms. Also, unvoiced regions in the caller's recording are no longer interpolated over — they are retained to help align the clip to the original.

Finally, the pitch values used were normalised around the mean and standard deviation of the pitch of the reference clip, μ_r and σ_r respectively. For $1 \leq i \leq R$ and $1 \leq j \leq C$:

$$r_i \leftarrow \frac{r_i - \mu_r}{\sigma_r} \quad (3)$$

$$c_j \leftarrow \frac{c_j - \mu_r}{\sigma_r} \quad (4)$$

3.3. Penalising humming

The system scores users who hum just as highly as those who sing, since the system is based only on pitch estimates. One solution to this problem that was tested was to combine the existing pitch-based score, s_p, with the confidence score returned by a speech recogniser. The two values would then be treated as separate dimensions and the distance from the origin would represent the combined score.

To test this, we wrote an SRGS[1] grammar containing lyrics for each of the alternative songs. The recorded corpus was recognised with that grammar using a commercial off-the-shelf recogniser. The confidence scores returned ranged from 0 to 100 — sometimes nothing was recognised, i.e. a `nomatch` event as described in the VoiceXML standard[2], in which case 0 would be used. Since that score runs in the opposite direction to the pitch score (a confidence of 0 means there is very little match but $s_p = 0$ would mean an identical match for the pitch-based score) s_r was defined as $100 - confidence$. A scaling factor was applied; its value was determined using the corpus described in Section 5.3, and so the revised score is $\sqrt{(s_p^2 + 500 s_r^2)}$.

4. Evaluation

A number of different evaluation methods and corpora were used to guide decisions through the design of the final system. They are described in the following sections.

4.1. Simple Pitch Comparison

Some of the sung data in [6] were used, namely 50 recordings of subjects singing "Row, row, row your boat" and 50 recordings of

[1] Speech Recognition Grammar Specification `www.w3.org/TR/speech-grammar/`

[2] `www.w3.org/TR/voicexml20/#dml5.2.6`

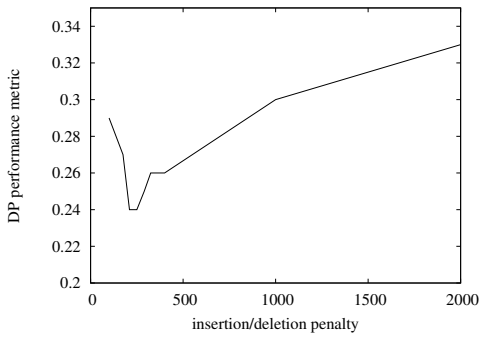

Figure 3: Plot of results in Table 3

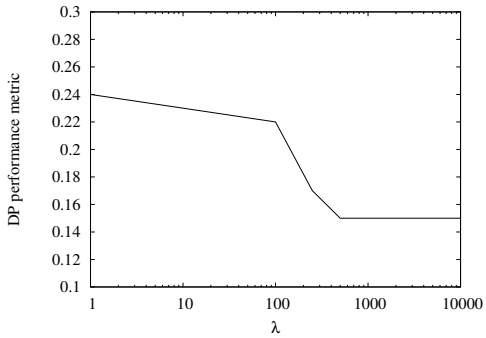

Figure 4: Plot of results in Table 4

ins. del. penalty	mean $eval_2$
100	0.29
175	0.27
210	**0.24**
250	**0.24**
290	0.25
325	0.26
400	0.26
1000	0.3
2000	0.33

Table 3: The relationship between the insertion and deletion penalty and the DP performance metric.

λ	mean $eval_2$
1	0.24
10	0.23
100	0.22
250	0.17
500	**0.15**
1000	**0.15**
10000	**0.15**

Table 4: Finding the optimum weight λ to put on the reversed confidence s_r returned by the speech recogniser.

"O Canada"[3].

Three examples were taken of each of those songs — one higher pitched female singer, a low pitched male singer and one other chosen at random. Those six samples, $R_1 \ldots R_6$ were in turn considered to be the reference that the other 49 singers were considered to be "aiming for". For each of the target renditions R_i a human judge chose a set of two to four other singers that were perceived to be singing sufficiently closely to the reference — sets $C_1 \ldots C_6$. R_i was always a member of C_i.

The system would rank all 50 singers in terms of closeness to the reference R_i and if it were performing well it would rank those singers in C_i higher. Therefore the expression to be minimised is

$$\frac{1}{N}\sum_{i=1}^{N} eval_1(i) \tag{5}$$

where $N = 6$ and

$$eval_1(i) = \frac{1}{|C_i|}\sum_{\forall c \in C_i} rank(c) \tag{6}$$

Lower values of the metric indicate better performance — the optimal value for our particular C_is is $\frac{23}{12}$. This was the first and simpler evaluation metric used.

4.2. Dynamically-aligned Pitch Comparison

In order to compare systems with each other a corpus of 35 renditions of four different songs was collected. A small VoiceXML application was setup in order to collect this data — the call flow covered the first two steps of that described in Section 1. This meant that the recordings were made with standard landline telephony equipment and conditions were similar to the publicly available system, i.e. mono recordings at an 8kHz sampling rate.

The recordings in that corpus were typically less than 10 seconds in length, whilst those in the planned deployment were to be 30 seconds long. So, at a later date, another corpus of 27 recordings of the correct length was collected. The same call-flow was used, but with two different songs.

Once the recordings had been collected, two human judges independently ranked the renditions for each song. The highest ranked recordings were those judged to be closest to the original recording the caller was attempting to impersonate. The systems being evaluated were also run on the data and provided another ranking. The outcome of this step is that for each song and for each rendition i of that song there exists

- From each judge j, a human ranking of that rendition compared to the other caller's attempts r_j^i
- A system ranking r_S^i

All rankings range from 1 through N, where N is the number of renditions of that song — no ties were allowed. Given J judges (here $J = 2$), the performance of the system on a rendition i is

$$eval_2(i) = \frac{1}{J(N-1)}\sum_{j=1}^{J} |r_j^i - r_S^i| \tag{7}$$

We need to normalise by $N - 1$ because different numbers of recordings were made for different songs. The mean $eval_2$ was the metric to be minimised — it could range from 0 (giving judgements always identical to the human judges) to 1 (always differing completely from the gold standard).

[3] In terms of the clip IDs used in the corpus, these were clips e211 through till e260 and clips g211 through till g260 respectively.

4.3. Penalising Humming

The method that penalised humming was evaluated in the same way as the dynamic programming approach of the previous section. There were some examples of humming in the development corpus.

5. Experiments

A number of parameters were estimated using the evaluation metrics described in Section 4, and these will be detailed in the following sections.

An issue that arose at this stage was that the range of the score s_p was not consistent between songs — one song may have callers scoring from 300 to 500 yet another might result in scores from 600 to 900. In both cases however, the caller had to be given a score out of ten.

To resolve this issue, the mean and standard deviation for each song's scores are maintained as callers call in — those values are used as parameters to a Gaussian distribution and subsequent scores are placed in one of ten equiprobable intervals.

A disadvantage of this approach is that it leaves open the possibility for very early callers to receive a different score to that they would have received with an identical call later on.

5.1. Simple Pitch Comparison

The simple performance metric is used in the following two sections, i.e. mean $eval_1$ as described in Equation 6.

5.1.1. Pitch vector length

The optimal number of pitch samples to take from the full pitch curve was determined using the corpus described in Section 4.1 — the search space is shown in Figure 1. The minimum, shown in bold, was **100** but 50 points were used in order to minimise the size of the feature vector.

5.1.2. Interpolation threshold

These tests were done using 50 samples, as determined in the previous section. We next looked at smoothing over unvoiced regions — pitch was used as an indicator of voicing — and here we determined the boundary between voiced and unvoiced speech. We did this by taking a given percentage of the lowest pitched samples to be unvoiced — the percentage used is determined in Figure 2 to be **12.5**.

5.2. Dynamically-aligned Pitch Comparison

The results in this section are quoted in terms of the DP performance metric, i.e. mean $eval_2$ as defined in Equation 7.

The optimal value for the insertion and deletion penalty in Equation 1 was found using the method described in Section 4.2. The details are given in Figure 3.

5.3. Penalising Humming

The coefficient λ in the "humming-sensitive" score $\sqrt{(s_p^2 + \lambda s_r^2)}$ was optimised in the same way — details are given in Figure 4. Since the range of s_p varies between songs, λ would ideally be re-estimated for each song. The optimal value resulted in an relative improvement of **37%** in mean $eval_2$.

6. Conclusions

A simple method for evaluating the singing of callers to an entertainment application was developed and evaluated. The most effective method considered involved making estimates of F_0 and performing a Viterbi alignment of the pitch curves of the clips to be compared. Measuring the total pitch difference of the aligned curves led to a distance metric between the caller and the gold standard.

The system was deployed for a popular TV talent show. Our overall aim was that the scores given were plausible to the caller, since that would encourage people to call in again to try to do better. **73%** of calls made were from repeat callers, so the system appears to have succeeded in that measure.

Further work would include verifying the effectiveness of the preliminary humming detection method on a larger corpus.

In addition, it would be preferable to use separate vocal tracks for the reference clip, either from the original recording or through source separation.

Finally, the distance measure used in the Viterbi alignment, stated in Equation 2, could be replaced with a probabilistic measure. That would mean that the distance between two pitch estimates would be the negative log probability of the pitch difference, rather than the absolute difference in pitch after normalisation (Equations 3 and 4). This is taken from [8].

7. References

[1] L. K. Saul, D. D. Lee, C. L. Isbell, and Y. LeCun, "Real time voice processing with audiovisual feedback: toward autonomous agents with perfect pitch," in *Advances in Neural Information Processing Systems 15*, S. Becker, S. Thrun, and K. Obermayer, Eds. Cambridge, MA: MIT Press, 2003, pp. 1205–1212.

[2] J.-J. Aucouturier and F. Pachet, "Music similarity measures: What's the use?" in *Proceedings of the International Symposium on Music Information Retrieval*, Paris, October 2002.

[3] D. Talkin, "A robust algorithm for pitch tracking (RAPT)," in *Speech Coding and Synthesis*. Amsterdam, Elsevier Science, 1995, pp. 495–518.

[4] M. Huckvale, "Speech Filing System," `www.phon.ucl.ac.uk/resource/sfs/`.

[5] E. Vincent, "Musical source separation using time-frequency priors," in *IEEE Transactions on Speech and Audio Processing — Special Issue on Statistical and Perceptual Audio Processing*, 2005.

[6] D. Gerhard, "Speech/song corpus," `www2.cs.uregina.ca/~gerhard/cgi-bin/corpus/`.

[7] N. Hu and R. B. Dannenberg, "A Comparison of Melodic Database Retrieval Techniques Using Sung Queries," in *Joint Conference on Digital Libraries*. Portland, OR, USA: ACM Press, 2002, pp. 301–307.

[8] N. Hu, R. B. Dannenberg, and A. L. Lewis, "A Probabilistic Model of Melodic Similarity," in *Proceedings of the International Computer Music Conference*. San Francisco: International Computer Music Association, 2002.