# GENERALIZED TONAL PITCH SPACE WITH EMPIRICAL TRAINING

**Hiroyuki YAMAMOTO** (山本紘征) (yamamoto@kusuli.com) [1] and **Satoshi TOJO** (東条敏) (tojo@jaist.ac.jp) [1]

[1] **JAIST**, Ishikawa, Japan

## ABSTRACT

A chord name can be interpreted in multiple ways, so a sequence of chord names has combinatorially many interpretations though most of which are inadequate. Tonal Pitch Space (TPS) is a music model which enables us to measure the distance between two chords, and thus we can rely on the theory to find most plausible interpretations, calculating the shortest path in the network of chord sequences. Although TPS is based on classical music theory, it is not based on data in a precise sense. As a result, the distance in the original TPS is somewhat rough to achieve high prediction accuracy.

In this study, we combine empirical observations with TPS, that is, to allow users to pick arbitrary combinations of features and calculate the distance of two chord interpretations. Then we propose a path probability formula to convert a path distance to a path probability, so that we can train the parameters from annotated datasets. We illustrate several experimental distance elements and show that some combinations of them can significantly improve the prediction accuracy, which resulted in over 86% in the test set.

## 1. INTRODUCTION

A Berklee style chord name by itself can be interpreted in several ways, and we need to consider the context to determine the plausibility of each candidate. Tonal Pitch Space (TPS) [3] gives us a foundation to consider the context by defining the smoothnesss of chord connection as the numeric distance between two chords, given their keys and degrees. Based on this, Sakamoto *et al.* [4] have proposed a method to find the most plausible interpretation path for a chord sequence as the shortest path in the interpretation graph, that expresses all possible chord interpretation paths each edge is weighted by the distance on TPS. However, the prediction accuracy of this method is only around 40%. This is, we assume, partly because TPS is based on classical music theory but not on data. So its structure and coefficients are not, strictly speaking, defined in an objective manner. Therefore, the model is a little too simple to achieve high prediction accuracy.

In this study, we work through these problems by combining empirical observations with TPS. First, we rearrange the distance formula in TPS to the sum of three distance elements, then generalize it to allow us to add other distance elements. These distance elements we define are in the form of tables whose cells correspond to the specific combinations of features of two chord interpretations. Next, we propose a path probability formula which gives higher probability to a path with shorter total distance. Finally, by differentiating the cross entropy loss function, we calculate the gradient and update the parameters using it.

Our approach [1] enables us to generalize and refine TPS by learning the metric model of arbitrary combinations of features as long as they contribute to decrease the value of target (loss) function. And we demonstrate the effectiveness of our approach by showing the best model being able to significantly improve the prediction accuracy and achieve over 86%.

This paper is organized as follows. In Section 2, we review related works. Then we give the formal representation of our proposed model and the learning strategy in Section 3 and 4, respectively. Thereafter, we show the experimental results in Section 5. Finally, we conclude in Section 6.



## 2. TPS-BASED APPROACH

There have been a lot of approaches to analyze musical harmony, and nowadays, a model with Hidden Markov Model (HMM) [12–15] and that with neural networks [16–18] seem prevalent. In this paper, however, we focus on Tonal Pitch Space. The theory finds the shortest path by the sum of the smallest distances in chords, and thus it results in the most plausible interpretation of chords by keys and degrees. Therefore, the detection of the shortest path is also expected to coincide with the local key identification.

### 2.1 Tonal Pitch Space

TPS is a music model for the quantitative harmony analysis proposed by Lerdahl [3]. It is proposed to complement Lerdahl's the other music theory (the Generative Theory of Tonal Music [5]), which applies the generative grammar to extend the Schenkerian theory. A chord can be interpreted in multiple degree/key pairs (*e.g.*, interpretations of C major triad are as follows: I/C, III/a, V/F, IV/G, VI/ a, and VII/d) and TPS defines a distance between every pair of these degree/key pairs.

The distance between chord interpretations $x$ and $y$, when

[1] Source code is available at https://github.com/kusuli/smc2021/.

they are in related keys, can be calculated as equation (1)

$$\delta(x,y) = \begin{array}{r} region(x,y) + chord(x,y) \\ +basicspace(x,y) \end{array} \quad (1)$$

where $region(x,y)$ is a distance between keys, $chord(x,y)$ is a distance between degrees, and $basicspace(x,y)$ is a distance on a structure called basic space.

The calculation above is applicable only when $x$ and $y$ are in related keys which are defined as follows:

$$C(R) = \begin{cases} \{\mathrm{I}, \mathrm{i}, \mathrm{ii}, \mathrm{iii}, \mathrm{IV}, \mathrm{V}, \mathrm{vi}\} & \text{if } R \text{ is a major key} \\ \{\mathrm{i}, \mathrm{I}, \mathrm{bIII}, \mathrm{iv}, \mathrm{v}, \mathrm{bVI}, \mathrm{bVII}\} & \text{otherwise} \end{cases} \quad (2)$$

where roman numerals in this equation mean the keys with the tonic being the degree of $R$ (*e.g.*, $C(\mathrm{F})$ is F, f, g, a, Bb, C, and d). If $x$ and $y$ are not in related keys (*i.e.*, distant keys), distance between $x$ and $y$ can be calculated as :

$$\delta(x,y) = \min\left( \begin{array}{l} \delta(x, T_{R_1}) + \Delta(R_1, R_n) + \delta(T_{R_n}, y) \\ |R_1 \in C(R_x), R_n \in C(R_y) \end{array} \right)$$

$$\Delta(R_1, R_n) = \min\left(\sum_{i=1}^{n-1} \delta(T_{R_i}, T_{R_{i+1}}) | R_{i+1} \in C(R_i)\right) \quad (3)$$

where $T_R$ is key $R$'s tonic, $R_z$ is chord interpretation $z$'s key. In other words, the transition from $x$ to $y$ must be considered as a combination of transitions within related keys, and the overall distance is the shortest total distance of the transitions.

As explained above, the distance within related keys (equation (1)) is composed of the sum of three elements. Now, because equation (3) is the sum of equation (1)s, the resulting distance can also be considered as the sum of three elements. Therefore, we can rewrite the distance as follows:

$$\delta(x,y) = \begin{array}{r} totalRegion(x,y) + totalChord(x,y) \\ +totalBasicspace(x,y) \end{array} \quad (4)$$

### 2.2 Former Approaches based on TPS

Sakamoto *et al.* [4] have applied TPS to analyze chord sequences to find the most plausible interpretation as the shortest path based on the distances described above.

Given a chord sequence, first their method extends each chord to its interpretations and constrcuts a graph whose edges have weights that correspond to the distances on TPS. Then it applies the Viterbi algorithm [6] to find the shortest interpretation paths from the start to the goal. Figure 1 shows an interpretation graph for chord sequence C → F → G → C. One of the shortest interpretation path in Figure 1 is I/C → IV/C → V/C → I/C.

Catteau *et al.* [9] utilized the key profiles of Temperly [10] alongside TPS to define probabilities concerning chords, scales, and chroma vectors to estimate keys and chords from audio. Rocher *et al.* [11] used Temperly's key profiles and TPS to construct a harmonic graph then estimate individual chords and keys by finding the best path.

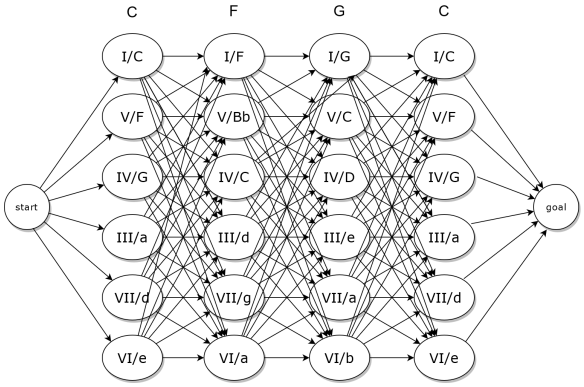


Figure 1. Interpretation graph.

In the effort to improve cadence detection, Matsubara *et al.* [7] have proposed to restrict the minor scale to harmonic one to avoid the ambiguity in chord interpretation, and to revise the candidates of chord interpretations of each chord names. Yamamoto *et al.* [8] have proposed to extend TPS and interpretation graph to consider (1) tetrads and three minor scales, (2) pivot-chord modulations, and (3) certain cadence patterns to improve the expressiveness and reduce the ambiguity mainly focusing on jazz harmony. Furthermore, there are many approaches with some kinds of metric models other than TPS. Feisthauer *et al.* [19] ,for example, defined three proximity measures based on musicologial knowledge to find the optimal path as the tonal plan.

In the following sections, we revise the structure of TPS and predict chord interpretations using the interpretation graph proposed by Sakamoto *et al.* [4].

## 3. PROPOSED MODEL

We define notations as follows:

$\mathcal{X} \triangleq \{\mathrm{I/A}, \mathrm{ii/A}, \cdots, \mathrm{VI/g\#}, \mathrm{VII/g\#}\}$ : the set of chord interpretations

$x, y \in \mathcal{X}$: individual chord interpretations

$\mathcal{I} \triangleq \{1, 2, 3, 4.1, 4.2, 5.1, \cdots, |\mathcal{I}|\}$: the set of distance element indices

$scale : \mathcal{X} \to \{0, 1\}$: the function which maps a chord interpretation to its scale [2] (*e.g.* $scale(iii/A) = 0, scale(III/c) = 1$)

$tonic : \mathcal{X} \to \{n \in \mathbb{Z} | 0 \leq n \leq 11\}$: the function which maps a chord interpretation to its tonic note [3] (*e.g.* $tonic(iii/A) = 9, tonic(III/c) = 0$)

$$majorTonic(x) \triangleq \begin{cases} tonic(x) \text{ if } scale(x) = 0 \\ (tonic(x) + 3)\ mod\ 12 \text{ otherwise} \end{cases}$$

[2] Here, we only consider major (= 0) and minor (= 1) scales.
[3] We use pitch classes to express notes.

(*e.g.* $majorTonic(iii/A) = 9$, $majorTonic(III/c) = 3$)

$root : \mathcal{X} \to \{n \in \mathbb{Z} | 0 \leq n \leq 11\}$: the function which maps a chord interpretation to its root note (*e.g.* $root(iii/A) = 1$, $root(III/c) = 3$)

$degree : \mathcal{X} \to \{n \in \mathbb{Z} | 1 \leq n \leq 7\}$: the function which maps a chord interpretation to its degree (*e.g.* $degree(iii/A) = 3$, $degree(III/c) = 3$)

$distanceElement_i : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$: the function which maps a chord interpretation pair to their distance based on the distance element of index $i \in \mathcal{I}$

$b : \mathcal{I} \to \{0, 1\}$: the function which specifies the activation of each distance element

The distance on TPS can be thought of the sum of three distance elements as in equation (4). Now we rearrange this equation as a sum of all (active) distance elements.

$$GTPS(x, y) = \sum_{i \in \mathcal{I}} b(i) \cdot distanceElement_i(x, y) \quad (5)$$

The first three distance elements are from the original TPS, namely, $totalRegion$, $totalChord$, and $totalBasicspace$ in equation (4). In addition to them, we can add arbitrary new distance elements by freely choosing which and which features to distinguish. In the following subsections, we propose in total twelve new distance elements, which are inspired by the original TPS. Finally, with $b(i)$ term in equation (5), we can use any combinations of distance elements.

### 3.1 Distance Element 4: Scale Distance

Distance elements for scale transitions. We define two variants as follows:

#### *3.1.1 DE 4.1: Symmetric Scale Distance*

$$M_{4.1} \in \mathbb{R}^2$$
$$distanceElement_{4.1}(x, y) \triangleq M_{4.1} \begin{bmatrix} (scale(x) - scale(y)) \\ mod\ 2 \end{bmatrix}^{4} \quad (6)$$

This one merely distinguishes whether the scale is changed or not.

#### *3.1.2 DE 4.2: Asymmetric Scale Distance*

$$M_{4.2} \in \mathbb{R}^{2 \times 2}$$
$$distanceElement_{4.2}(x, y) \triangleq M_{4.2} [scale(x), scale(y)] \quad (7)$$

The asymmetric version of DE 4.1 (*e.g.*, major → minor, and minor → major are considered same in DE 4.1, but not in DE 4.2).

[4] $M[i_i, i_2, \cdots, i_n]$ indicates the value in $n$ dimensional table $M$ at the index $(i_1, i_2, \cdots, i_n)$.

### 3.2 Distance Element 5: Tonic Distance

Distance elements for tonic transitions, by which we intend to generalize $totalRegion$ in equation (4). Tonic transitions can be thought of as key transitions without considering scales. We define six variants as follows:

#### *3.2.1 DE 5.1: Symmetric Relative Tonic Distance*

$$M_{5.1} \in \mathbb{R}^7$$
$$distanceElement_{5.1}(x, y) \triangleq M_{5.1} \left[ min \begin{pmatrix} (majorTonic(y) - majorTonic(x)) \\ mod\ 12, \\ (majorTonic(x) - majorTonic(y)) \\ mod\ 12 \end{pmatrix} \right] \quad (8)$$

Among all variants, this one is conceptually closest to the original $totalRegion$.

#### *3.2.2 DE 5.2: Symmetric Parallel Tonic Distance*

$$M_{5.2} \in \mathbb{R}^7$$
$$distanceElement_{5.2}(x, y) \triangleq M_{5.2} \left[ min \begin{pmatrix} (tonic(y) - tonic(x))\ mod\ 12, \\ (tonic(x) - tonic(y))\ mod\ 12 \end{pmatrix} \right] \quad (9)$$

Unlike the relative tonic distance, this one identifies parallel keys (*e.g.*, C major and C minor), instead of relative keys (*e.g.*, C major and A minor).

#### *3.2.3 DE 5.3: Asymmetric Relative Tonic Distance*

$$M_{5.3} \in \mathbb{R}^{12}$$
$$distanceElement_{5.3}(x, y) \triangleq M_{5.3} [(majorTonic(y) - majorTonic(x))\ mod\ 12] \quad (10)$$

The asymmetric version of DE 5.1 (*e.g.*, C → D and D → C are distinguished in DE 5.3, but not in DE 5.1).

#### *3.2.4 DE 5.4: Asymmetric Parallel Tonic Distance*

$$M_{5.4} \in \mathbb{R}^{12}$$
$$distanceElement_{5.4}(x, y) \triangleq M_{5.4} [(tonic(y) - tonic(x))\ mod\ 12] \quad (11)$$

The asymmetric version of DE 5.2.

### 3.3 Distance Element 6: Key Distance

Distance elements for key transitions, which can handle both scale transitions and tonic transitions at once. One can calculate those distances by the combination of DE 4.x and DE 5.x, but this assumes the independence of the transitions of scales and that of tonics. By contrast, DE 6.x can consider the interactions of scales and tonics, if any. There are two variants as follows:

#### 3.3.1 DE 6.1: Symmetric Key Distance

$$M_{6.1} \in \mathbb{R}^{2\times 7}$$
$$distanceElement_{6.1}(x, y)$$
$$\triangleq M_{6.1} \begin{bmatrix} (scale(x) - scale(y)) \\ \quad mod\ 2, \end{bmatrix} min \begin{pmatrix} (tonic(y) - tonic(x)) \\ \quad mod\ 12, \\ (tonic(x) - tonic(y)) \\ \quad mod\ 12 \end{pmatrix} \tag{12}$$

#### 3.3.2 DE 6.2: Asymmetric Key Distance

$$M_{6.2} \in \mathbb{R}^{2\times 2\times 12}$$
$$distanceElement_{6.2}(x, y)$$
$$\triangleq M_{6.2} \begin{bmatrix} scale(x), scale(y), \\ \end{bmatrix} \begin{matrix} (tonic(y) - tonic(x)) \\ mod\ 12 \end{matrix} \tag{13}$$

The asymmetric version of DE 6.1.

### 3.4 Distance Element 7: Root-Degree Distance

Distance elements for root note transitions from each degree, which roughly generalize $totalChord$ in equation (4), although with much more information [5]. We define two variants as follows:

#### 3.4.1 DE 7.1: Symmetric Root-Degree Distance

$$M_{7.1} \in \mathbb{R}^{7\times 7}$$
$$distanceElement_{7.1}(x, y)$$
$$\triangleq M_{7.1} \begin{bmatrix} degree(x), min \begin{pmatrix} (root(y) - tonic(x)) \\ \quad mod\ 12, \\ (tonic(x) - root(y)) \\ \quad mod\ 12 \end{pmatrix} \end{bmatrix} \tag{14}$$

This one calculates distance according to the relative positions of roots for each (source) degree.

#### 3.4.2 DE 7.2: Asymmetric Root-Degree Distance

$$M_{7.2} \in \mathbb{R}^{7\times 12}$$
$$distanceElement_{7.2}(x, y)$$
$$\triangleq M_{7.2} \left[degree(x), (root(y) - tonic(x))\ mod\ 12\right] \tag{15}$$

The asymmetric version of DE 7.1.

### 3.5 Distance Element 8: Key-Degree Distance

Distance elements for key and degree transitions, which can handle both key (*i.e.*, scale and tonic) transitions and degree transitions all at once. Unlike the combinations of DE 4.x, DE 5.x, and DE 7.x or DE 6.x and DE 7.x, DE 8.x can consider the interactions of scales, tonics and degrees. We define two variants as follows:

[5] A straightforward way to do this may be to take step distance between two degrees (*i.e.*, replacing $tonic$ in DE 5.2 and DE 5.4 with $degree$), but we omitted them because both of them perform very poorly.

#### 3.5.1 DE 8.1: Symmetric Key-Degree Distance

$$M_{8.1} \in \mathbb{R}^{2\times 7\times 7\times 7}$$
$$distanceElement_{8.1}(x, y)$$
$$\triangleq M_{8.1} \left[\ (scale(x) - scale(y))\ mod\ 2, degree(x),\right.$$
$$\left. degree(y), min \begin{pmatrix} (tonic(y) - tonic(x))\ mod\ 12, \\ (tonic(x) - tonic(y))\ mod\ 12 \end{pmatrix} \right] \tag{16}$$

#### 3.5.2 DE 8.2: Asymmetric Key-Degree Distance

$$M_{8.2} \in \mathbb{R}^{2\times 7\times 2\times 12\times 7}$$
$$distanceElement_{8.2}(x, y)$$
$$\triangleq M_{8.2} \left[scale(x), degree(x), scale(y), degree(y),\right.$$
$$\left. (tonic(y) - tonic(x))\ mod\ 12\right] \tag{17}$$

The asymmetric version of DE 8.1.

All the proposed distance elements and sample indices are listed in Table 1.

| DE | I/C → V/G | I/C → iv/c# |
|---|---|---|
| 4.1 Sym Scale | (0) | (1) |
| 4.2 Asym Scale | (0, 0) | (0, 1) |
| 5.1 Sym Relative Tonic | (5) | (4) |
| 5.2 Sym Parallel Tonic | (5) | (1) |
| 5.3 Asym Relative Tonic | (7) | (4) |
| 5.4 Asym Parallel Tonic | (7) | (1) |
| 6.1 Sym Key | (0, 5) | (1, 1) |
| 6.2 Asym Key | (0, 0, 7) | (0, 1, 1) |
| 7.1 Sym Root-Degree | (1, 2) | (1, 6) |
| 7.2 Asym Root-Degree | (1, 2) | (1, 6) |
| 8.1 Sym Key-Degree | (0, 1, 5, 5) | (1, 1, 4, 1) |
| 8.2 Asym Key-Degree | (0, 1, 0, 5, 7) | (0, 1, 1, 4, 1) |

Table 1. Indices for two sample transitions.

## 4. LEARNING STRATEGY

We define additional notations as follows:

$G$: an interpretation graph with $|G|$ layers

$G_{s:t}$: from $s$th layer to $t$th layer of $G$ ($G_{s:s}$ can be abbreviated as $G_s$). As a simplified notation, a node in the $s$th layer can be written as $x \in G_s$, likewise, $x \in G_{s:t}$ be a path from the $s$th layer to the $t$th layer, and $x \in G_{s:t-1}||x_t$ be a path from $s$th layer to the $(t-1)$th layer and added $x_t$ to be the last node.

$x_{s:t}$: from $s$th element to $t$th element of an interpretation path $x_{0:|G|}$ ($x_{s:s}$ can be abbreviated as $x_s$)

$x^*_{0:|G|}$: the ground truth interpretation path

$GTPS_{path}(x_{s:t}) \triangleq \sum_{u=s}^{t-1} GTPS(x_u, x_{u+1})$

We want the calculated distances to allow us to estimate the true interpretation path as a shortest path in the interpretation graph. So we need to learn the parameters to give true interpretation path a shorter total distance than the other interpretation paths.

For that purpose, we first define the path probability formula and then train the tables by using the gradients on the parameter spaces.

### 4.1 Path Probability

We define the path probability from start node to $s$th chord interpretation as below:

$$\begin{aligned}&\mathrm{P}(X_{0:s} = x_{0:s}|G_{0:s})\\ &\triangleq \begin{cases} 1 & \text{if } s = 0\,^{6} \\ \prod_{t=0}^{s-1} \frac{\exp(-GTPS(x_t, x_{t+1}))}{Z_{G,t}} & \text{otherwise} \end{cases}\end{aligned} \tag{18}$$

where

$$Z_{G,t} \triangleq \sum_{l \in G_t} \sum_{m \in G_{t+1}} \mathrm{P}(X_t = l|G_{0:t}) \exp(-GTPS(l, m))$$

We can calculate the probability for the whole interpretation path as $\mathrm{P}(X_{0:|G|} = x_{0:|G|}|G_{0:|G|})$. This probability is designed to give higher values to the interpretation paths with shorter total distances (Theorem 1).

We can calculate the node probability $\mathrm{P}(X_s = x_s|G_{0:s})$ by marginalizing path probability:

$$\begin{aligned}\mathrm{P}(X_s = x_s|G_{0:s}) &= \sum_{x_{0:s} \in G_{0:s-1}||x_s} \mathrm{P}(X_{0:s} = x_{0:s}|G_{0:s}) \\ &= \sum_{x_{0:s} \in G_{0:s-1}||x_s} \prod_{t=0}^{s-1} \frac{\exp(-GTPS(x_t, x_{t+1}))}{Z_{G,t}} \\ &= \sum_{x_{0:s} \in G_{0:s-1}||x_s} \left( \prod_{t=0}^{s-2} \frac{\exp(-GTPS(x_t, x_{t+1}))}{Z_{G,t}} \right) \\ &\quad \times \frac{\exp(-GTPS(x_{s-1}, x_s))}{Z_{G,s-1}} \\ &= \sum_{x_{0:s} \in G_{0:s-1}||x_s} \mathrm{P}(X_{0:s-1} = x_{0:s-1}|G_{0:s-1}) \\ &\quad \times \frac{\exp(-GTPS(x_{s-1}, x_s))}{Z_{G,s-1}} \\ &= \sum_{x_{s-1} \in G_{s-1}} \mathrm{P}(X_{s-1} = x_{s-1}|G_{0:s-1}) \\ &\quad \times \frac{\exp(-GTPS(x_{s-1}, x_s))}{Z_{G,s-1}}\end{aligned}$$

When $s = 0$, $x_{0:0} \in G_{0:-1}||x_0$ becomes $x_{0:0} \in x_0$ because $G_{0:-1}$ is empty. Note that, $\mathrm{P}(X_s = x_s|G_{0:s}) = \mathrm{P}(X_s = x_s|G_{0:|G|})$ is not always the case. As we can see, this process has a recursive structure, and, by calculating and memorizing in a sequential manner from the start node, we can get the node probability and path probability with the time complexity linear to $s$.

[6] 0th layer contains only one node, that is, the start node

### 4.2 Loss and Gradient

We define a cross entropy loss function as follows:

$$\begin{aligned}&Loss(x_{0:|G|}|G_{0:|G|}) \\ &\triangleq \sum_{x_{0:|G|} \in G_{0:|G|}} \begin{matrix} -\mathrm{P}^*(X_{0:|G|} = x_{0:|G|}) \\ \times \ln \mathrm{P}(X_{0:|G|} = x_{0:|G|}|G_{0:|G|}) \end{matrix}\end{aligned} \tag{19}$$

Here, $\mathrm{P}^*$ is the probability function which only responds to the ground truth:

$$\mathrm{P}^*(X_{0:|G|} = x_{0:|G|}) \triangleq \begin{cases} 1 \text{ if } x_{0:|G|} = x^*_{0:|G|} \\ 0 \text{ otherwise} \end{cases}$$

We can get the gradient by differentiating $Loss$ (19) with respect to the parameters, then apply stochastic gradient descent algorithm to update the parameters to minimize the value of $Loss$ (19), which results in maximizing the path probability for the ground truth path.

### 4.3 Accuracy

We evaluate our model based on how accurately it can predict each chord interpretation by specifying the shortest path in the interpretation graph. If there are more than one shortest paths, we calculate a weighted average for each node in proportion [7] to how many paths go through the node as is illustrated in Figure 2.

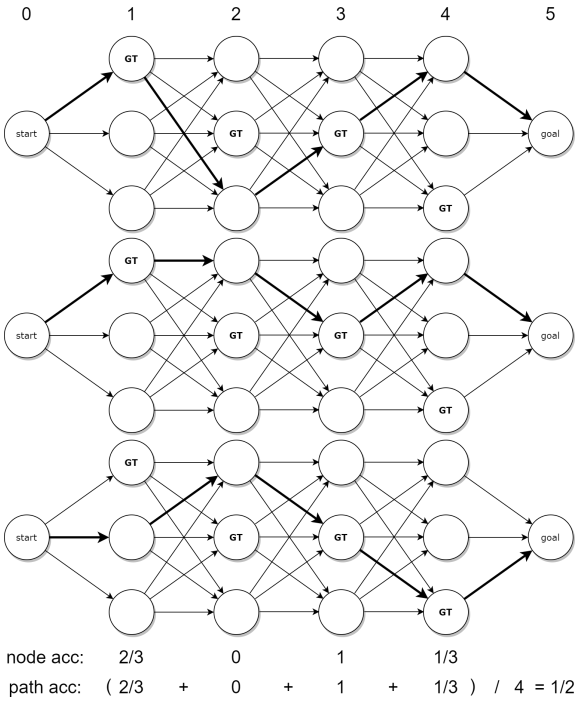


Figure 2. Accuracy calculation when there are more than one shortest paths.

[7] this proportion is different from the node probability

## 5. EXPERIMENTS

### 5.1 Data and Method

We use the dataset annotated in *rntxt* format [1], published at [1, 2]. The dataset is composed of 360 pieces (1,691 phrases, 76,341 chords) and we regard every phrase as a unit (*i.e.*, to which we predict the interpretation path) but when a phrase exceeds 50 chords we divide it into units each of which does not exceed 50 chords, resulting in 2,472 phrases. Then use 1,976 phrases to the training, and 248 phrases to the validation, and remaining 248 phrases to the test

Rntxt format contains a lot of information other than degree/key, but in this study we utilize only key and degree information. About secondary/tertiary chords, we employ local keys (*e.g.*, V/V/V on C major key is interpreted as V on D major key).

We set all initial parameter values to be zero and train the models by mini-batch stochastic gradient descent with batch size=100 and learning rate=0.001. We continue training until no accuracy update in validation set for ten epochs in a row, then pick the parameter which gives the highest validation accuracy..

### 5.2 Results

We compare the performances of each distance element and some combinations. The result is shown in the Table 2, 3, and 4.

| ex | DE 1 | DE 2 | DE 3 | mean | stdev |
|---|---|---|---|---|---|
| **0** | | | | 0.1900 | 0.0257 |
| **1** | ○ | ○ | ○ | 0.3847 | 0.1023 |
| **2** | ○ | | | 0.3780 | 0.1034 |
| **3** | | ○ | | 0.1930 | 0.0288 |
| **4** | | | ○ | 0.3842 | 0.1023 |
| **5** | ○ | ○ | | 0.3770 | 0.1052 |
| **6** | ○ | | ○ | 0.3850 | 0.1025 |
| **7** | | ○ | ○ | 0.3841 | 0.1023 |

Table 2. Performances of each distance element (and combinations) of original TPS. [9]

**ex 0** is without any distance elements, just for information.

**ex 1** is the original TPS. This one successfully double the accuracy (*i.e.*, narrow down the candidate interpretation by half) from **ex 0**. We consider this one to be the baseline.

We also conduct ablation patterns of TPS (**ex 2-7**). When used alone (**ex 2-4**), $totalBasicspace$ is the best performance (**ex4**) and achieved almost the same accuracy as the full TPS (**ex 1**). We consider the reason why $totalBasicspace$ is a little better than $totalRegion$ (**ex 2**) is that basic space can exress region distance by the diatonic level and also other levels can give additional information. Seeing the result of **ex 3**, however, $totalChord$ do not improve accuracy well. That is also the case when used two of them together (**ex 5-7**).

[9] **ex**, **DE**, **mean**, and **stdev** represent experiment, distance element, model mean accuracies, and standard deviations of accuracies respectively

| ex | DE | prms | mean | stdev |
|---|---|---|---|---|
| **8** | 4.1 Sym Scale | 2 | 0.1900 | 0.0257 |
| **9** | 4.2 Asym Scale | 4 | 0.2522 | 0.1432 |
| **10** | 5.1 Sym Relative Tonic | 7 | 0.3983 | 0.1006 |
| **11** | 5.2 Sym Parallel Tonic | 7 | 0.2908 | 0.2415 |
| **12** | 5.3 Asym Relative Tonic | 12 | 0.3974 | 0.1003 |
| **13** | 5.4 Asym Parallel Tonic | 12 | 0.2870 | 0.2408 |
| **14** | 6.1 Sym Key | 14 | 0.4249 | 0.1739 |
| **15** | 6.2 Asym Key | 48 | 0.5017 | 0.3380 |
| **16** | 7.1 Sym Root-Degree | 49 | 0.5646 | 0.1640 |
| **17** | 7.2 Asym Root-Degree | 84 | 0.5741 | 0.1628 |
| **18** | 8.1 Sym Key-Degree | 686 | 0.8625 | 0.1780 |
| **19** | 8.2 Asym Key-Degree | 2,352 | 0.8690 | 0.1717 |

Table 3. Performances of proposed distance elements.

In **ex 8-19**, we test each proposed distance elements by themselves. DE 5.1 can accomplish almost the same accuracy as the full TPS (**ex 1, 10**), although it has only seven parameters. DE 4.x cannot improve accuracy at all without distinguishing directions (**ex 8,9**), but surprisingly, for many other distance elements, it turns out that there is very little or no accuracy gain by distinguishing the direction from the comparisons **ex 10** to **ex 12**, **ex 11** to **ex 13**, **ex 16** to **ex 17**, and **ex 18** to **ex 19**. We also test tonic distances in which parallel keys are identified (**ex 11, 13**), but they are significantly worse than those of relatie keys (**ex 10, 12**). DE 8.x, being the most complex distance elements, can achieve over 86% accuracy.

In **ex 20-26**, we test some combinations of proposed distance elements. The combinations are selected so that involved distance elements complement each other though not exhaustive. The combination of **ex 23** can achieve 83% with only 58 parameters, likewise, that of **ex 25** and **ex 26** can achieve 85.0% and 86% with a little more parameters. Therefore, it seems that taking the interactions of all scale, tonic, and degree into account is not so important considering the huge parameter size. Also, it is interesting that DE 4.1 have meaningful contribution in **ex 23** here although it does not make difference at all by itself (**ex 8**).

We also test some combinations of TPS element and distance tables (**ex 27-29**). Root table can be benefited from the elements from TPS (**ex 28**), but in the other combinations, there are not so obvious accuracy gains.

From all the experiments, we can observe that the combinations which achieved over 80% (**ex 18, 19, 23-26, 29**) have all three features (*i.e.*, $scale$, $tonic/majorTonic$, and $degree$ (or $degree, root$)), but one of them (**ex 23**) does not consider interactions nor directions. Therefore, it seems that, including those three features is crucial, but considering interactions or directions have relatively small effects.

For illustrative purpose, we show some possible interpretations for a chord progression Cm $\rightarrow$ F $\rightarrow$ Bb $\rightarrow$ Eb $\rightarrow$ A$^\circ$ $\rightarrow$ D $\rightarrow$ Gm and their total distances in Table 5. We

| ex | DE 1 | DE 2 | DE 3 | DE 4.1 | DE 4.2 | DE 5.1 | DE 6.1 | DE 6.2 | DE 7.1 | DE 7.2 | prms | mean | stdev |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **20** | | | | ○ | | ○ | | | | | 9 | 0.3978 | 0.1015 |
| **21** | | | | | ○ | ○ | | | | | 11 | 0.5131 | 0.3402 |
| **22** | | | | | | ○ | | | ○ | | 56 | 0.7627 | 0.1585 |
| **23** | | | | ○ | | ○ | | | ○ | | 58 | 0.8301 | 0.1869 |
| **24** | | | | | ○ | ○ | | | ○ | | 60 | 0.8226 | 0.1814 |
| **25** | | | | | | | ○ | | ○ | | 63 | 0.8495 | 0.1775 |
| **26** | | | | | | | | ○ | | ○ | 132 | 0.8601 | 0.1681 |
| **27** | | ○ | ○ | | | | ○ | | | | 14 | 0.4210 | 0.2318 |
| **28** | ○ | | ○ | | | | | | ○ | | 49 | 0.7309 | 0.1566 |
| **29** | | | ○ | | | | ○ | | ○ | | 63 | 0.8308 | 0.1887 |

Table 4. Performances of some combinations of distance elements.

use the model trained in **ex 23** to calculate the distances. In this example, paths **b** and **c** both have only one key, but calculated distances are longer than that of **a**, which consists of two keys. But they are shorter than paths **d** and **e**. We think this order more or less matches to our musical perception.

| | Cm | F | Bb | Eb | A° | D | Gm |
|---|---|---|---|---|---|---|---|
| **a** | ii/Bb | V/Bb | I/Bb | VI/g | ii°/g | G/g | i/g |
| | - | 5.88 | 8.91 | 16.44 | 22.30 | 28.11 | 31.14 |
| **b** | ii/Bb | V/Bb | I/Bb | IV/Bb | vii°/Bb | III/Bb | vi/Bb |
| | - | 5.88 | 8.91 | 14.25 | 20.75 | 26.59 | 33.36 |
| **c** | iv/g | VII/g | III/g | VI/g | ii°/g | G/g | i/g |
| | - | 6.27 | 12.66 | 19.64 | 25.50 | 31.31 | 34.34 |
| **d** | v/f | I/F | IV/F | VII/f | vii°/Bb | VI/F | ii/F |
| | - | 8.77 | 14.11 | 26.12 | 37.46 | 46.03 | 51.89 |
| **e** | i/c | I/F | I/Bb | I/Eb | ii°/g | I/D | i/g |
| | - | 10.84 | 18.91 | 26.98 | 37.41 | 49.12 | 60.29 |

Table 5. Some possible interpretations and their total distances.

## 6. CONCLUSIONS

In this study, we have extended TPS to take in empirical observation. We generalized the distance formula in TPS and proposed a way to define distance elements that distinguish any combinations of given features and to train them with data. Our best combination achieved 86.9% accuracy in the test set, which is significantly higher than that of the baseline model (38.5%), and this result, we believe, shows that our approach successfully learns an effective metric structure from data. Also, one of our combination with only 58 parameters achieved 83%, and with 132 parameters, 86%. We hope that these simple models will help us to understand better about the structure of tonal harmony.

There are many potential directions to improve our method. First, it would be beneficial to accept sequences of chroma vectors or piano-roll as input. Second, not only TPS, it would also be meaningful to extend our approach to deal with key profiles like Krumhansl's [20]. Furthermore, the fact that distinguishing directions only makes small difference in accuracy is somewhat contradictory to our previous research [8]. This implies that there may be a better way to take directions into account.

**Acknowledgments**

This work is supported by JSPS Kakenhi 16H01744.

## 7. REFERENCES

[1] D. Tymoczko, M. Gotham, M. S. Cuthbert, and C. Ariza. "The RomanText Format: A Flexible and Standard Method for Representing Roman Numeral Analyses". *International Society for Music Information Retrieval Conference (ISMIR)*, pp.123-129, 2019

[2] M. S. Cuthbert, C. Ariza. "music21: A toolkit for computer-aided musicology and symbolic music data.", *International Society for Music Information Retrieval Conference (ISMIR)*, pp.637–642, 2010

[3] F. Lerdahl: "Tonal Pitch Space", Oxford University Press, 2001

[4] S. Sakamoto, S. Arn, M. Matsubara, S. Tojo: "Harmonic analysis based on tonal pitch space", in *Proceedings of the 8th International Conference on Knowledge and Systems Engineering (KSE)*, pp.230-233, 2016

[5] F. Lerdahl, R. Jackendoff: *"A Generative Theory of tonal music"*, Cambridge, MA, 1983

[6] A. Viterbi: *"Error bounds for convolutional codes and an asymptotically optimum decoding algorithm." IEEE transactions on Information Theory*, 13.2: pp.260-269, 1967

[7] M. Matsubara, T. Kodama, S. Tojo: *"Revisiting cadential retention in GTTM"* in *2016 Eighth international conference on knowledge and systems engineering (KSE)*, pp.218-223, 2016

[8] H. Yamamoto, Y. Uehara, S. Tojo: *"Jazz harmony analysis with $\epsilon$-transition and cadential shortcut"* in *Proceedings of 17th Sound and Music Computing Conference (SMC)*, pp.316-322, 2020

[9] B. Catteau, J. Martens, M. Leman. *"A probabilistic framework for audio-based tonal key and chord recognition"* in *Advancesin Data Analysis*, pp.637–644, 2007

[10] D Temperley. *"What's Key for Key? The Krumhansl-Schmuckler Key-Finding Algorithm Reconsidered"*. Music Perception 17, 1, pp.65–100. 1999

[11] T. Rocher, M. Robine, P. Hanna, L. Oudre. *"Concurrent estimation of chords and keys from audio"*. in *Proceedings of the 11th International Society for Music Information Retrieval Conference (ISMIR)*, pp.141–146. 2010

[12] W. Chai, B. Vercoe. *"Detection of key change in clssical piano music"*. in *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, pp.468–473. 2005

[13] L. Mearns, E. Benetos, S. Dixon. *"Automatically detecting key modulations in J. S. Bach Chorale recordings"*. in *Proceedings ofthe 8th Sound and Music Computing Conference (SMC)*, pp.25–32, 2011

[14] N. Nápoles López, C. Arthur, I. Fujinaga. *"Key-finding based on a hidden Markov model and key profiles"*. in *Proceedings of the 6th International Conference on Digital Libraries for Musicology*, pp.33–37, 2019

[15] H. Papadopoulos, G. Peeters. *"Local Key Estimation Based on Harmonic and Metric Structures"*. in *International Conference on Digital Audio Effects (DAFx)*, pp. 408–415, 2009

[16] T. Chen, L Su. *"Functional harmony recognition of symbolic music data with multi-task recurrent neural networks"*. in *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, pp.90–97, 2018

[17] T. Chen, L. Su. *"Harmony transformer: incorporating chord segmentation into harmony recognition"*. in *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*, pp.259–267, 2019

[18] G. Micchi, M. Gotham, M Giraud. *"Not all roads leadto Rome: pitch representation and model architecture for automatic harmonic analysis*, in *Transactions of the International Society for Music Information Retrieval* 3, 1 (May 2020), pp.42–54, 2020

[19] L. Feisthauer, L. Bigo, M. Giraud, F. Levé. *"Estimating keys and modulations in musical pieces"*. in *Sound and Music Computing Conference (SMC)*, pp. 323-330, 2020

[20] C. L. Krumhansl *"Cognitive foundations of musical pitch"*, Oxford University Press, 1990

## 8. APPENDIX

**Theorem 1** (order accordance)**.** *In an interpretation graph $G$, $GTPS_{path}(x_{0:s})$ is smaller than $GTPS_{path}(x'_{0:s})$ if and only if $\mathrm{P}(X_{0:s} = x_{0:s}|G_{0:s})$ is greater than $\mathrm{P}(X_{0:s} = x'_{0:s}|G_{0:s})$.*

*Proof.*

$$
\begin{aligned}
&GTPS_{path}(x_{0:s}) < GTPS_{path}(x'_{0:s}) \\
&\Leftrightarrow \exp(-GTPS_{path}(x_{0:s})) > \exp(-GTPS_{path}(x'_{0:s})) \\
&\Leftrightarrow \exp(-\sum_{t=0}^{s-1} GTPS(x_t, x_{t+1})) \\
&\quad > \exp(-\sum_{t=0}^{s-1} GTPS(x'_t, x'_{t+1})) \\
&\Leftrightarrow \prod_{t=0}^{s-1} \exp(-GTPS(x_t, x_{t+1})) \\
&\quad > \prod_{t=0}^{s-1} \exp(-GTPS(x'_t, x'_{t+1})) \\
&\#\text{divide both sides by the same (positive) value} \\
&\Leftrightarrow \frac{\prod_{t=0}^{s-1} \exp(-GTPS(x_t, x_{t+1}))}{\prod_{t=0}^{s-1} Z_{G,t}} \\
&\quad > \frac{\prod_{t=0}^{s-1} \exp(-GTPS(x'_t, x'_{t+1}))}{\prod_{t=0}^{s-1} Z_{G,t}} \\
&\Leftrightarrow \prod_{t=0}^{s-1} \frac{\exp(-GTPS(x_t, x_{t+1}))}{Z_{G,t}} \\
&\quad > \prod_{t=0}^{s-1} \frac{\exp(-GTPS(x'_t, x'_{t+1}))}{Z_{G,t}} \\
&\#\text{from equation (18)} \\
&\Leftrightarrow \mathrm{P}(X_{0:s} = x_{0:s}|G_{0:s}) > \mathrm{P}(X_{0:s} = x'_{0:s}|G_{0:s})
\end{aligned}
$$

□