Scoring Models for Tamil Lyrics
Elanchezhiyan, TamilSelvi E, Karthikeyan, Rajapandian, Madhan Karky
Karky Research Foundation, Chennai, India.

Abstract
Lyrics are rich in features such as rhyme, pleasantness, similes, metaphors and more. Many of these features are exclusive to lyrics. We have estimated that more than two thousand Tamil lyrics are being created every year in various forms. Modeling the lyric-specific features becomes an essential task in organizing the lyrics for retrieval and analysis. In this paper, we propose three scoring models for analyzing the rhyme, pleasantness and freshness of a lyric. The paper also explains in detail on how the weights were selected for individual models. Scores were computed for 1200 Tamil lyrics based on these models. Discussing the score distribution in detail, the paper concludes with open questions for further research.

Keywords
Rhyme, Pleasantness, Freshness, Lyrics, Scoring models.

1. Introduction
Can art be quantified? We like a painting, we dislike a piece of music, and we find a certain section of a story, good. Is it possible to quantify an art form for comparing one piece of work with another? On the whole, the answer is no. But we can certainly take a few quantifiable features from the art form and compare those features. In a painting, the number of colors used is quantifiable. In music, the pitch and number of instruments are quantifiable features.

Tamil language is rich in literature. We have numerous poetic forms in Tamil. Lyrics, composed with music, are one of the most popular forms of literature. Lyrics from popular songs are being searched and quoted on many blogs, social networking platforms. A few thousands of lyrics are being created every year in form of popular music, motion picture soundtracks, private albums and more. Lyrics have unique features in them such as rhyme, similes, metaphors and many more that are not present in a prose or a news article.

In this paper, we aim to identify those specific features that can be quantified. We try to model the rhyme, pleasantness and freshness of a lyric in order to compute a score for each of these features. Algorithms for computing these scores are presented. The scores are analyzed by comparing them against human judgment for correlation.

This paper is organized into five sections. The following section presents a brief survey of literature relevant to relevant lyric feature extraction and genre classification based on lyric feature. The third section describes the scoring model for Tamil lyrics. We analyze this scoring model by determining the correlation coefficient between scoring model and human judgment in the fourth section. The final section summarizes the paper and discusses the future extension of this work.

2. Background
In Tamil, rhyme refers to the way in which successive lines are strung in a poem or lyric. The occurrence of similar syllables and words in successive lines correspond to alliteration (monai), rhyme (edhugai) and end-rhyme (iyaibu).

Lyric Freshness in this paper refers to the word freshness in lyrics. A song with words that were never used before are considered fresher.

Pleasantness of a lyric is a combined pleasantness of words used in the lyric. Pleasantness of a word does not take into account the meaning of word or the music in the song. The syllables in a word, the presence or absence of long vowels, consonant family the characters belong to and place of articulation determines the pleasantness of a word.

“Poetic Features for Poem Recognition: A Comparative Study” [1] describes the poetic features and how it helps to distinguish and extract poetic feature with which poem can be classified from other type of text.


“LaaLaLaa - A Tamil Lyric Analysis and Generation Framework” [3] generates meaningful Tamil lyrics to a given tune and domain based on the lyric features such as rhyme, meaning and flow. The Rhyme finder is used to choose words that match one or more of the three Rhyme properties (edhugai, monai, iyaibu).
“Special indices for LaaLaLaa Lyric Analysis and Generation Framework” [4] proposes a framework to construct the indices for efficient lyric generation, based on Tamil rhyme scheme such as Rhyme, Meter match and Parts of Speech for fast retrieval.

Hirjee and Brown [5] present a sophisticated tool for extracting rhymes from lyrics based on phoneme frequencies, with a focus on hip-hop styles. This work uses the basic rhyme to study the quality of lyric. A large corpus of lyrics is used for studying rhyme.

In this paper, we propose a scoring model by which a lyric is scored based on rhyming patterns, rhyme schemes (edhugai, monai, iyaibu, Ati EMI, Murru EMI, Inai EMI, Polippu EMI, Oruu EMI, Kulai EMI, Kilkkatuvey EMI, Merkatuvey EMI), Freshness (Timeline and word uniqueness) and Pleasantness (Vowel, consonants classifications & place of articulation) of lyric that are specific to Tamil language to evaluate lyrics.

3. Scoring Models for Tamil Lyric

Modeling the lyric-specific features is an essential task in organizing the lyrics for retrieval and analysis. In Rhyme scoring model based rhyme schemes and rhyme patterns in Tamil. Rhymes schemes are further classified into Ati EMI, Murru EMI, Inai EMI, Polippu EMI, Oruu EMI, Kulai EMI, Kilkkatuvey EMI, Merkatuvey EMI and Rhyme patterns are further classified into Equal Family, Consonant Family, Vowel Family, Consonant Hardness Family, and Meter Match. The following sub section explains the rhyme schemes in Tamil.

3.1 Rhyme Schemes and Rhyme Patterns in Tamil

In Tamil, the grapheme and phoneme are bound stronger than in English. There are 3 characteristic rhyme schemes in Tamil – Monai, Edhugai and Iyaibu.

Two words are said to rhyme in monai if their first letters are the same, in edhugai if their second letters are the same and in iyaibu if their last letters are the same.

Examples:

- பறைவ (paRavai) and பாசைச (pachai) rhyme in monai as they start with the same letter.
- அஞ்சை (Aruvi) and விருப்பு (viruppu) rhyme in edhugai as they share the same second letter.
- யாக்கைக (yaakkai) and வாழ்க்கை (vaazhkai) rhyme in iyaibu as they share the same last letter.

3.1.1 Rhyme Schemes

Further, rhyme can be sub-categorized according to the position of the letters which are involved in the process. Monai between words across lines have different names, if the first letter of a word matches with the first letter of a word in the subsequent line, then it is classified as ati-monai.

Example:

\[ \text{ulagathai maRandhaen uRakkathai maRandhaen} \]
\[ \text{unnudan naan ondRu kalandhaen} \]

Monai between different words within a line also has different names. The one existing between the first and second words within a line is inai monai.

Examples:

\[ \text{milavukku milavu sugam peRa ninthaen} \]

Monai between the first and third word is polippu monai.

Examples:

\[ \text{ulagathai maRandhaen uRakkathai maRandhaen} \]
**OtRu monai** is the one which exists between the first and fourth word.

Examples:

```
nilavukku nilavu sugampeRa ninaithaen
```

The **kulai monai** is the one between the first, second and third words.

Examples:

```
uyirae uyirae udambil sirandhadhu edhu vendRu thavithirandhaen
```

Monai between the first, second and fourth words is **kilkkatuvay monai**.

Examples:

```
ilavukku nilavu sugampeRa ninaithaen
```

A **merkatuvay monai** is that which occurs between the first, third and fourth words.

Examples:

```
arumpum inimayum adhanavazhi gNikoLum
```

If all the four words within a line match, then it is a **mutRu monai**.

Examples:

```
mannavaa mannavaa mannaadhi mannan allavaa
```

The positional categorization discussed above applies to edhugai and iyaibu also by considering the second and last letter of words in each line accordingly. Here, we also refer this kind of categorization as internal rhyme.

The rhyme can be further sub-categorised based on the nature of the letters. If the monai is due to only the consonants and not the vowels, then it is a **varukka monai**.

Examples:

```
pudhaikkindRa vidhaiyum muyaRsigoNdaal thaan
```

If the monai exists only due to vowels, not the consonant in the initial position, then it is a **uyir monai**.

Examples:

```
parivaana naNban thandha
```

Monai between long vowels is called as **netil monai**.

Examples:

```
vaalipathhil mannadhan
```

94
The monai due to the three classes of consonants is called an *ina monai*. Since there are hard class (vall-inam), soft class (mell-inam) and middle class (itai-inam), there exist *valina monai*, *Melina monai* and *itai-ina monai*.

**Example for valina monai**

```plaintext
thinggal enhadhu peNNaga
nejvay koavai pazhamaaga
```

**Example for Melina monai**

```plaintext
vaarukku jaanaagi kaathirundhaaL
vaNNa vizihiyin vaasalil en theevan thoandRinaan
nenjvil aedhoa aedhoa ninaithaen
madhuk kiNNathai ini naan thoda maattaen
```

The nature of letter sub-categorization discussed above applies to edhugai and iyaibu also by considering the second and last letter of words in each line accordingly. Here, we also refer this kind of categorization as an imperfect rhyme.

### 3.1.2 Rhyme Patterns

When a lyric is input to the system, rhyme scoring begins by extracting rhyme patterns. Rhyme pattern for each scheme is a string, which reflects the letter occurrences in that rhyming spot of each line in the lyric.

Separate patterns for monai, edhugai and iyaibu are extracted from the input lyric by considering the rhyming scheme as shown below.

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Rhyme Scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rhyme-AA</td>
<td>Ati EMI</td>
</tr>
<tr>
<td>Rhyme-AAAA</td>
<td>Murru EMI</td>
</tr>
<tr>
<td>Rhyme-AABB</td>
<td>Inai EMI</td>
</tr>
<tr>
<td>Rhyme-ABAB</td>
<td>Polippu EMI</td>
</tr>
<tr>
<td>Rhyme-ABBA</td>
<td>Oruu EMI</td>
</tr>
<tr>
<td>Rhyme-AAAB</td>
<td>Kulai EMI</td>
</tr>
<tr>
<td>Rhyme-AABA</td>
<td>Kilkatuvay EMI</td>
</tr>
<tr>
<td>Rhyme-ABAA</td>
<td>Merkatuvay EMI</td>
</tr>
</tbody>
</table>

**Table 1: Rhyme patterns for lyric analysis**

Here EMI represents edhugai or monai or iyaibu scheme. It is categorized according to the position of the letter involved and we try to extract all high-level rhyme features for scoring the lyric.

In order to extract the rhyme patterns for each scheme, we need to identify the occurrence of similar syllables and words in successive lines. It need not be identical alone but also based on the nature of letters (where syllable sounds belongs to same category).

Based on the nature of letters, we classify the rhyming patterns as

<table>
<thead>
<tr>
<th>Rhyme Pattern</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equal Family</td>
<td>Identical letters</td>
</tr>
<tr>
<td>Consonant Family</td>
<td>Varukka EMI</td>
</tr>
<tr>
<td>Vowel Family</td>
<td>Uyir EMI</td>
</tr>
<tr>
<td>Consonant Hardness Family</td>
<td>Valina, Melina, Idai-ina EMI</td>
</tr>
<tr>
<td>Meter Match</td>
<td>Netil EMI</td>
</tr>
</tbody>
</table>
Table 2: Classification of imperfect rhyme patterns

Given a line or stanza (of any size), we match each subsequent letter to pattern based on the category as specified in table (ii). The rhyme patterns are extracted for the three rhyming schemes (monai, edhugai, iyaibu) separately for each line or stanza.

Example:

Consider a segment of a sample lyric given below.

\[\text{ninaivae ninaivae nenjoadu kalandhvidu}
\text{nilavae nilavae viNNodu ka}\]

Extract every first, second and last letter of each word. The first letters of first line are \(ni, ni, ne, ka\) mapping to string under:

- Equal Family – AABC
- Consonant Family – AAAB
- Vowel Family – AABC
- Consonant Hardness Family – AABC
- Meter Match – AABC

Here we can observe that first two letter belongs to equal family category, while when comparing together with third letter there exists a match in the consonant family.

The second letters of first line are \(nai, nai, nj, la\) mapping to string under:

- Equal Family – AABC
- Consonant Family – AABC
- Vowel Family – AABC
- Consonant Hardness Family – AAAB
- Meter Match – AABC

The last letters of letters of first line are \(vE, vE, tu, tu\) mapping to string under:

- Equal Family – AABB
- Consonant Family – AABB
- Vowel Family – AABB
- Consonant Hardness Family – AABB
- Meter Match – AABB

Here, we extract the rhyme patterns (EMI) for each family within a line, likewise we have to extract the patterns across the lines (ati) also in the same manner.

3.1.3 Rhyme Scoring Model

The pattern string found out as described above is to be scored by the rhyme scorer. We propose a scoring model which handles high level rhyme features such as imperfect and internal rhyme.

The boundary conditions for our model are listed below:

Let,

- \(l\) be the Number of letters to be compared;
- \(l^e\) be the Number of letters matched in equal family;
- \(l^c\), be the Number of letters matched in consonant family;
- \(l^v\), be the Number of letters matched in vowel family;
- \(l^h\), be the Number of letters matched in consonant hardness family;
- \(l^m\), be the Number of letters matched in meter match;
The number of letters matched represents the character found to be equal in the pattern string. We assign weight to each category, thus the letters matched under equal family will have a higher score than the letters under consonant family and so on.

Compute score based on the pattern string

\[
\begin{align*}
EScore &= (t^e/t^f) \times w^e \\
CScore &= ((t^e - t^c )/ t^c -) \times w^c \\
VScore &= ((t^m - t^v) / t^v) \times w^v \\
HScore &= ((t^m - t^h) / t^h) \times w^h \\
MScore &= ((t^m - t^m) / t^m) \times w^m \\
Score &= \sum (EScore, CScore, VScore, HScore, MScore)
\end{align*}
\]

Where,

\[w^e = 0.5, \quad w^c = 0.2, \quad w^v = 0.2, \quad w^h = 0.05, \quad w^m = 0.05.\]

Let \(w^e, w^c, w^v, w^h, w^m\) be the weight assigned for equal, consonant, vowel, consonant hardness and meter match family respectively.

Let’s see how it works for the previous given example,

\[\text{ninaivE ninaivE nenjOt}\]

We consider first letter of each word (ni, ni, ne, ka) to calculate the score for monai as below:

\[
\begin{align*}
EScore (AABB) &= 0.5 \\
CScore (AABB) &= 0.225 \\
VScore (AABB) &= 0 \\
HScore (AABB) &= 0 \\
MScore (AABB) &= 0 \\
Score &= 0.725
\end{align*}
\]

Edhugai and Iyaibu score will also be computed in the same manner as shown above by considering the second and last letter of a word respectively. The average score of all the three rhyming schemes is the overall rhyme score.

### 3.2 Pleasantness Scoring Model

The classification of consonant family into hard, soft & mid consonants along with the place of articulation is considered for pleasantness scoring. Pleasantness is computed by applying weights to the different classes of characters. Table 1. presents the weights for pleasantness computing.

<table>
<thead>
<tr>
<th>Vowels</th>
<th>Characters</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Short Vowels</td>
<td>அ/ா, இ/ி, உ/ு, எ/ே, ஒ/ோ</td>
<td>0.25</td>
</tr>
<tr>
<td>Long Vowels</td>
<td>அ/aa, 恚/ee, க/ை, த/ை, தி/ை, தை/ை, தே/ை, தோ/ை, தொ/ோ</td>
<td>0.50</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Consonants</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Soft Consonants</td>
<td>த/nt, த/nt, ம/nt, தி/nt, தை/nt, தே/nt, தோ/nt, தொ/nt</td>
</tr>
<tr>
<td>Medium Consonants</td>
<td>தில/iy, தில/iy, தி/il, தி/il, தி/il, தி/il, தி/il, தி/il</td>
</tr>
<tr>
<td>Transformations</td>
<td>தி/da, தி/dha, தி/da, தி/da</td>
</tr>
<tr>
<td>Rough</td>
<td>ą,k, ą,ch, ąth, ąp</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Greater Rough</td>
<td>ą/t, ąR</td>
</tr>
</tbody>
</table>

Table 1: Weights for Pleasantness Score

As we can see, long vowels and nasals have higher scores where as hard consonants have a lower score.

Pleasantness of every word is computed using the values provided in the table. The average pleasantness of all words in a song would determine the pleasantness of the lyric.

3.3 Freshness Scoring Model

The word fresh has a temporal aspect associated to it. It is not possible to reason about the freshness of a lyric without knowing its time of creation. We divide the entire timeline of lyrics into small windows of 10 years. Let E1, E2,...En denote the time windows.

For each era, we find the words from lyrics that were used for the first time in that era. We associate those words to the corresponding era. E1 words are words that were used in a lyric for the first time in E1. Stopwords are removed from all the lists.

The freshness score for a new lyric is determined by the following algorithm.

Let,

\[ l \] be the input lyric.
\[ n \] be the number words in \( l \)
\[ W \] be the set of words in \( l \)
\[ E_i \] be the \( i^{th} \) time window.
\[ E(w) \] be the time window of the word \( w \).
\[ E(l) \] denote the time window of the song.
\[ F_{s_i} \] be Freshness score of a word \( i \) in the lyric.
\[ F_s(l) \] be the Freshness score of a lyric \( l \)

Freshness Score :

begin
  initialize ScoreSum := 0;
  for each \( w_i \) in \( W \)
    \[ F_{s_i} := E(w_i) / E(l) \]
    ScoreSum := ScoreSum + \[ F_{s_i} \]
  end for
  \[ F_s = \text{ScoreSum} / n; \]
end

The basic idea behind the algorithm is that if a word is created in that era and used in the same era, it has the maximum score of 1. If it is used in the next era, the score becomes half.

4. Results and Evaluation

The Rhyme, Pleasantness and freshness models were presented to a group of linguists, lyricists and language experts who were asked to judge the rhyming, pleasantness and freshness style of a list of lyrics. It is observed that the rhyme score gets a positive correlation between the manual judgment and machine judgment.
Figure 1: Rhyme Score Comparison

In the Figure 1 the x axis represents the songs and the y axis the Rhyme score yielded by the scoring model and human judgment. The correlation coefficient between scoring model and human judgment is 0.00914.

Figure 2: Pleasantness Score Comparison

In the Figure 2 the x axis represents the songs and the y axis the Pleasantness score yielded by the scoring model and that by human judgment. The correlation coefficient between scoring model and human judgment is -0.1317. Pleasantness does not have a strong correlation between human judgment and computer score. The reason could be that the human judges consider songs they are familiar with to be more pleasant, without regarding the pleasantness of individual words.

Figure 3: Freshness Score Comparison

In the Figure 3 the x axis represents the songs and the y axis the freshness score yielded by the scoring model and human judgment. The correlation coefficient between scoring model and human judgment is 0.99374. Freshness has a strong correlation between human and machine scores.

5. Conclusions and Future Work

In this paper, we proposed three scoring models for rhyme, pleasantness and freshness of a lyric. The computed scores are being used by Paadal [6] a lyric portal, for visualizing the lyrics and in advanced searches. Studying the pleasantness of words from human perspective is one area we plan to investigate. Though the proposed rhyme scoring model handles the case of internal and imperfect rhyme features, assigning different weights to different categories of rhyme styles may vary the
score. An analysis of what could be a better weight allocation would make an interesting study in the future. Adjusting the weights of rhyme score components and reducing the time window size in freshness scores will open ways for further research in this area.

6. References