LickNet: Software Tools Collection for Guitar Licks Networks

Matteo Martelli

University of Bologna
matteo.martelli9@studio.unibo.it

Abstract.

1 Introduction

Musicians are often intrigued on how other musicians compose or improvise their music and if there are some relations with their previous composed music or with the compositions of other musicians. Even non expert music enthusiasts often wonder what makes a musician unique, well recognisable between the others, or what makes a musician so similar to another. For example this frequently happens with guitarists, when people recognize a "Santana" guitar solo or a "Hendrix" guitar solo just listening few notes and having never heard that particular solo, or when people says that Steve Vai sounds similar to Joe Satriani.[1]. One may think that this behaviour is caused by the fact that there are some notes patterns repetition in one musician compositions history or that there are some similar notes patterns between different musicians compositions caused by their similar musical influence or musical tastes.
Thus with the aim to analyze the relations between notes patterns in guitar solos, often called guitar licks, the developing process of the software described in this document, *LickNet*, has been started. The choose of guitar solos has been taken because of the large amount of data available online but the subject may be extended to different, but similar, kind of data. For example drum patterns relations in jam sessions or other instruments executions, compositions and improvisations may be considered. It has been chosen to represent the data elements and their relations within a network, as to the means offered by the graph theory useful for a mathematical analysis.
The area of scientific research that usually is interested in the empirical examination of real-world network is the study of *Complex Networks*. Some of the study approaches and mathematical methods that usually interests the Complex Networks area, such as graph theory or scale-free network comparison[1], have been considered and will be discussed in the next sections.
In addition to the network analysis tools, LickNet provides some other features

[1] Last.fm, a collaborative website, assigns a "super" level of similarity between Steve Vai and Joe Satriani. This may be related to the fact that Steve Vai was a his student.

focused more on the non-technical use of the application, like the *Lick Classifier* and the *Lick Generator*. These functionalities are still in an early stage but can be a starting point to realize some practical application focused to end users who may want to understand and replicate some other guitarist playing style, or to realize some artificial agents that may "play" some music while interacting with human players.

2 Related Works

Several empirical studies about music have been already treated with a network analysis approach. In [2] networks have been used to represent the correlations between the musical tastes of different listener. In addition the same paper show a technique to create a network of musical groups and artists which are connected by the similarity of their audience; a similar approach has been used in other remarkable works [3][4].
The authors of [5] showed a method to create a network of human language words linking them with a correlation parameter such as their relative distance in sentence, assigning the maximum distance for forming link of 2. This means that two words can be linked together if in a sentence one word is the successor or the successor of the successor of the other word. They stated that their choice has two reason:

1. Many co-occurrences of words take place at distance of one, e.g. 'red flowers' (adjective-noun), 'stay here' (verb-adverb), 'getting dark' (verb-adjective), etc.
2. Many co-occurrences of words take place at distance of two, e.g. 'hit the ball' (verb-object), 'Mary usually cries' (subject-verb), 'Live in Boston' (verb-noun), etc..

In the section 5 it will be pointed out that a similar graph creation procedure as been used in this work but with the maximum distance for forming link of 1 as it may result non trivial to determine a multi-step correlation value between musical notes in a notes lick. Anyway a maximum distance > 1 may be tested in future works.

3 LickNet

As introduced before, LickNet offers the possibility to study the relations between notes patterns in guitar solos through the creation and analysis of guitar licks networks. Moreover it collects some software tools with which is possible to do some particular operations with guitar licks through the use of the created networks. These operations are currently two, a classifier and a generator of guitar licks.
Before getting into the software components, the next section is going to focus on how the interested data are collected, structured and processed.

4 Data Representation

As many guitarists may already know, there are plenty of websites that store guitar sheet music files. One of them is `www.ultimate-guitar.com` that counts more than 800000 sheet music files, which are mostly written with the guitar tablature notation. Guitar players usually choose this notation for its ease of use [2] but it is instrument-specific. Also guitar tablature is not standardized and different sheet-music publishers adopt different conventions. This means that a guitar tablature can be understood only by guitar players and its conversion to the standard notation or formal interpretation may result wearing. For example the semantic of a guitar tablature changes even if a guitar it is not standard tuned.
Fortunately there are various computer programs available for writing tablature. One of most frequently used is *Guitar Pro* and the *Ultimate Guitar* servers store a large amount of tablature files encoded with its format. Another interesting software is *TuxGuitar*, a free and open source tablature editor that also supports the ability to import and export Guitar Pro files[6]. Moreover its source code can be freely re-used and adapted to any other application as it has been done with LickNet with the purpose of importing the tablature files retrieved online. After that the tablature can be interpreted by software, a guitar solo is represented as a sequence of notes, where each of them is composed of the following fields:

- String: at which string of the guitar the note is played
- Value: at which fret of the fretboard the note is played
- Effects: a collections of effects such as bending, vibrato, harmonic, etc.
- Duration: the time duration of the note, represented as a fraction value

This is how the TuxGuitar library represents the notes, but some adaptations and additional informations are needed for the application purpose. It has been decided to represent a single note in LickNet as the following:

- Base Note: the value of the musical note non dependent by the octave. It's a value in the range of [0,12] as the possible notes are 11 plus the *rest* note. The note 0 is assigned to C.
- Octave: the octave of the note. Note that the note pitch p is given by the equation $p = o \times 12 + b$, where o is the octave and b is the base note.
- Time: it's a floating point value in the range [0,1] indicating the duration of the note. The dotted and double dotted notes are also considered, as are the triplets. For example a quaver dotted note would have a time value of $\frac{1}{8} + \frac{1}{16} = 0.1875$ and a semi-quaver triplet note would have the value of $\frac{1}{16} \times \frac{2}{3} \simeq 0.042$.

[2] Guitar tablature removes the requirement for the player to remember the associations between the notes and the corresponding fretboard positions, as the latter are directly represented in the tablature notation.

– Bend Distance: if a note has a bending effect, the bending distance is the offset from the base note to the bended note. For example if a F note (base note = 5) is bended to G than the distance is 2, but if is bended to F# (half step bending) than the distance is 1. It is possible that the bending effect is defined with more complex properties: it may be defined with bending and release property or with the bending, release and bending again property. Thus it has been decided to calculate the bending distance has the average bending distance.
– Node Key: it is an identifier for a note in the graph. It is essentially a string where the values of the previous fields, except the octave, are concatenated. For instance a semi-quaver triplet F note with a bending to G would have the resulting node key: "05:0.042:b2". There may be the case in which the bending should not be considered, then the bending distance offset will be added to the base note and the node key of the same note in the previous example would be: "07:0.042".

In the next section will be shown how this data model is used in the network creation.

5 Networks Generator

The network is modeled as a directed multigraph and it's created scanning the input music sheets files. Let G be a directed multigraph defined as an ordered 4-tuple (N, E, s, t) with

– N a set of nodes,
– E a set of edges,
– $s : E \to N$, assigning to each edge its source node,
– $t : E \to N$, assigning to each edge its target node.

Moreover a node $n \in N$ contains a musical note identified by its Node Key. An edge $e \in E$ from a node s to a node t exists if the musical note contained in the node t was a successor, in the input music sheet, of the musical note contained in the node s. Also, there may exists more than one edge from a node s and t as a same musical note can be reached by another note many times with different octaves jumps. Thus an edge from s to t indicates with which octave jump the note contained in s can reach t in a graph walk step. An example of a multigraph with its corresponding sheet music is shown in figure 1. The octave informations are not included in the graph nodes because similar licks can be played at different octaves and otherwise it would be harder to underline the relations between them.

It has been decided to add a weight to each edge that corresponds on the times that an edge is passed through. This means that many walks from a node a to a node b with the same distance d but different total weight exist in the graph, and the walk that more likelihood correspond to a significant lick for the graph is the one with the highest total weight. A lick is intended to be significant if it is similar to the licks that are already used in the graph creation. The process

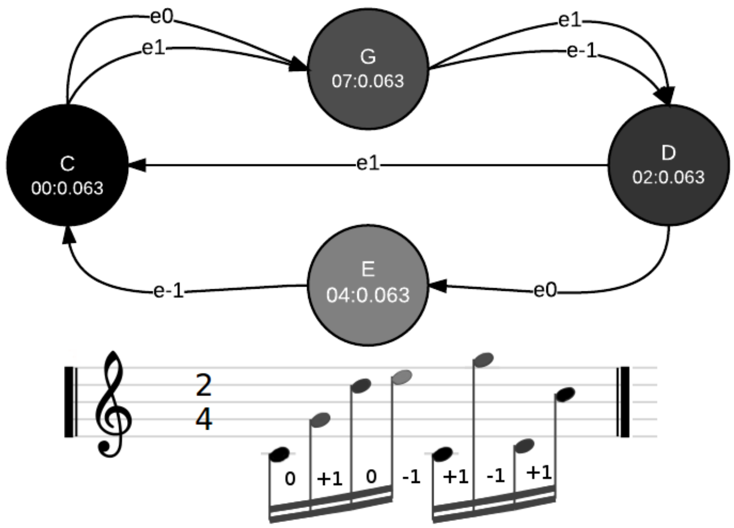

Fig. 1. A multigraph example. The corresponding input data used for the multigraph creation is represented with its sheet music. The id of each edge indicates with with octave jump its source node can reach the target node.

that determines if a lick is significant for a certain graph will be covered in the next section.

A simplification of the model has been introduced in the development process: the directed multigraph is represented as a directed graph, where an edge is an object that contains an array which each element i indicates the pass-through frequency number for the octave corresponding to i. More specifically if $f(j)$ is the frequency number related to the octave jump j, then

$$v(i) = f(i+n),$$

where n is the maximum octave increment or decrement possible. All the octaves on a standard tuned guitar are 5, thus with 6 octaves even the 7-strings guitars and most of the non-standard tuned guitars can be covered. Therefore the size of the array is 12 as a node s can reach in a single step a node t with an increment or decrement of most 6 octaves. To clarify, for example if the node s has reached with a single step the node t 5 times without any octave jump, 3 times with an octave jump of +1, 1 time with an octave jump of +2 and 2 times with an octave jump of -1, the resulting octave jumps array for the edge e that connects a and be would be:

$$v = (0, 0, 0, 0, 0, 2, 5, 3, 1, 0, 0, 0, 0)$$

A directed edge e from s to t exists if at least one element of the array is non-zero. The algorithm 1 shows how the graph creation process works.

Algorithm 1 Graph creation algorithm

```
function createGraph(sheets)
    G ← emptyGraph()
    for s in sheets do
        for n in s.notes do
            if pn then
                S_k ← genNodeKey(pn)
                T_k ← genNodeKey(n)
                if S_k ∉ G.nodes then
                    createNode(G, S_k)
                end if
                if T_k ∉ G.nodes then
                    createNode(G, T_k)
                end if
                E_k ← A_k + B_k                        ▷ Strings concatenation
                if E_k ∉ G.edges then
                    createEdge(G, E_k)
                end if
                e ← getEdge(G, E_k)
                ojump ← n.octave - pn.octave
                id ← ojump + N_OCTAVES                 ▷ N_OCTAVES is 6 for a guitar
                e.ojumps[id] += 1
            end if
            pn ← n
        end for
    end for
end function
```

5.1 Networks Analysis

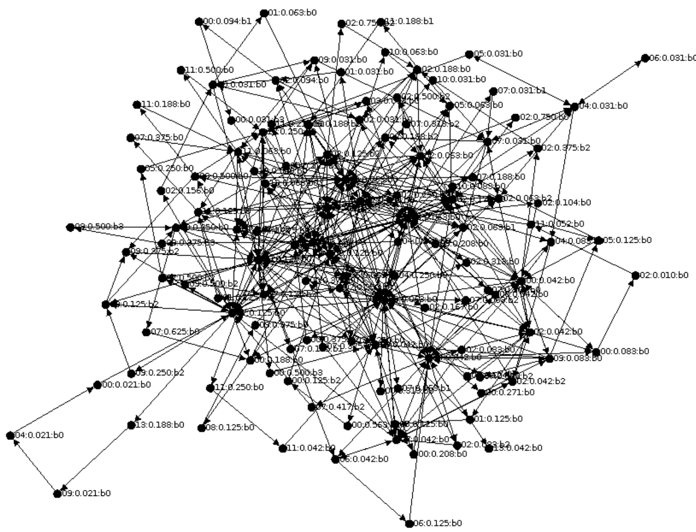

Fig. 2. A graph example created with 6 Jimi Hendrix solos. Edges of self linked nodes are not shown.

The figure 2 shows a graph created with the algorithm 1 using 6 Jimi Hendrix solo files as input. The resulting graph counts 124 nodes and 484 edges and the bending effects has been considered. Interestingly, the same graph created without the bendings consideration counts 101 nodes. This seems to happen also with other graphs and the difference is higher with those guitarists that heavily use bendings. This suggests that considering more guitar effects, such as vibratos or virtual harmonics, the vertex cardinality tends to increase. In the next sections it will be shown that considering more guitar effects may be also interesting for the licks classifier and the licks generator.

Many research articles on real network graphs, like biological, social and technological graphs showed that they share a common feature: the so-called small-world (SW) property [7]. In SW networks most nodes can be reached from every other by a small numbers of steps and furthermore they are characterized by a high clustering coefficient. In order to verify if the networks of guitar licks respects the SW properties, the average clustering coefficient C and the average shortest path length L must be determined[8]. In this analysis the network is converted to a non-directed graph as a model simplification. Let us start defining the clustering coefficient: given e_{ij} an edge from the vertex v_i to the vertex v_j, the number of cycles of length 3 (closed triplets) from/to a node i is:

$$p_{ii}^{(3)} = \sum_{jk} [Mawe_{ij}e_{jk}e_{ki} \ ,$$

thus the local clustering coefficient of a node i is

$$C_i = \frac{p_{ii}^{(3)}}{k_i(k_i - 1)/2} \ ,$$

and the average clustering coefficient of a graph is

$$C = \frac{1}{N} \sum_i C_i$$

where N is the number of nodes in the graph. Moreover the shortest path length average L, is defined as:

$$L = \frac{1}{N(N-1)} \sum_{i,j,i \neq j} d_{ij}$$

where d_{ij} is the shortest path length from a node i to a node j. Then the table 1 shows the C and L values of some graphs created with the algorithm 1 and converted to non-directed graphs, the C_{rand} and L_{rand} values that indicates C and L of the equivalent derived random networks and the small-coefficient σ defined as:

$$\sigma = \frac{C/C_{rand}}{L/L_{rand}}$$

What is remarkable is that the σ values are > 1, also $C \gg C_{rand}$ and $L \approx L_{rand}$, satisfying the SW property [8].

graph	N	C	C_{rand}	L	L_{rand}	σ
Dave Murray	102	0.329	0.074	2.864	2.602	4.039
Jimi Hendrix	124	0.429	0.064	2.675	2.649	6.63
Angus Young	122	0.266	0.068	3.092	2.888	3.65
Whole Graph	213	0.531	0.049	2.734	2.627	10.4

Table 1. Small World comparison results.

6 Lick Classifier

As introduced before, a possible application for the graphs created with the method described in the previous section may be a lick classifier. Let us introduce the scenario in which a graph is created with the solos of a particular guitarist only. Then, creating a graph for different guitarists, it may be tempting to identify to which of them an unknown guitar lick belongs to. This can be possible because the links in the graphs are weighted according to the number of times they are walked through, meaning that if a sequence of musical notes results having a higher total weight in a walk on a graph a than on a graph b, the probability that the sequence occurs more often on the graph a, and in the corresponding guitarist solos, than on the graph b is also higher. With this idea the lick classifier works: from a set of graphs and one lick it simply performs a walk of the lick on every graph collecting the corresponding total weights. The total weights are then dived by the number of notes used in the corresponding graphs creation.

7 Licks Generator

8 Tests and Results

The table 2 shows the results of 4 tests made with the lick classifier. For each experiment it has been used a guitar solo (licks set) of a particular guitarist that was not used in any of the graph creation (unknown). Each row indicates a different experiment, the first column indicates the guitarists of the unknown licks set used for each experiment and the remaining columns indicate the scores gained for each graph by the corresponding unknown licks set.

Lick guitarist	Dave Murray	Jimi Hendrix	Angus Young
Jimi Hendrix	0.285	**<u>0.356</u>**	0.196
Dave Murray	**<u>0.268</u>**	0.052	0.040
B.B. King	0.011	**<u>0.057</u>**	**0.045**
Kirk Hammett	**0.724**	**<u>0.785</u>**	0.403

Table 2. Licks classifier test results.

For convenience let us call *library* the set of graphs used in each experiment. Moreover the scores are relative to each test and should not be interpreted globally, as they are dependent on the notes sequence length of the licks. It is interesting how well the classifier recognizes that an unknown licks set of a guitarist in the library belongs to its guitarist. What is also remarkable is that the matching expectation of the experiments are satisfied: without any surprise, the licks set of B.B. King matches more the graph of Jimi Hendrix than the others. Also it is not very far the score of Angus Young while it is the score of Dave Murray. This may be caused by the fact that Jimi Hendrix and Angus Young may be more influenced by the blues playing style. Furthermore it may result weird that the licks set of Kirk Hammett, the lead guitarist of Metallica, matches more the Jimi Hendrix graph than the others. But the thing to notice is that the score matching with Dave Murray is also close while the score with Angus Young is far. This may be caused by the fact that Hammett was influenced by Jimi Hendrix [9] in his early years and that the genre played by Metallica is closer to the genre played by Iron Maiden. Anyway these are all suppositions, for a better interpretation it can be helpful considering the graph of Kirk Hammett also, in order to determine the gap between those scores and the score gained with Kirk Hammett himself. For example by the first two experiments one may interpret that Jimi Hendrix has a closer relation with Dave Murray than Angus Young, but with the limited amount of data this cannot be easy verified. A measurement method for finding the correlations between the guitarists using the classifier here proposed may be modeled in future works.

9 Development Complications

10 Future Development

11 Conclusions

References

1. Maarteen van Steen, *Graph Theory and Complex Networks: An Introduction*, 2010.
2. R. Lambiotte, M. Ausloos, Uncovering collective listening habits and music genres in bipartite networks, *Phys. Rev. E 72, 066107*, 2005
3. Dr Tamás Nepusz, Reconstructing the structure of the world-wide music scene with Last.fm, http://sixdegrees.hu/last.fm
4. Benjamin Lind, Patterns in the Ivy: The Small World of Metal, http://badhessian.org/2013/09/patterns-in-the-ivy-the-small-world-of-metal/, 2013
5. Ramon Ferrer i Cancho, Ricard V. Solè, The small world of human language, *Proc. Royal Soc. London B, DOI: 10.1098/rspb.2001.1800* 2001
6. Daniel Mantilla, TuxGuitar: Editorial review, *Software Informer*, 2014.
7. Watts, D. J. and Strogatz, S. H., Collective dynamics of 'small-world' networks, *Nature 393, 440-442*, 1998
8. Qawi K. Telesford, Karen E. Joyce, Satoru Hayasaka, Jonathan H. Burdette, Paul J. Laurienti The Ubiquity of Small-World Networks, *Brain Connect.; 1(5): 367–375*, 2011
9. Brian Ives, Hendrix At 70: "My Favorite Guitar Player Of All Time" – Kirk Hammett Of Metallica, *CBS Local*, http://wcbsfm.cbslocal.com/2012/11/25/hendrix-at-70-my-favorite-guitar-player-of-all-time-kirk-hammett-of-metallica/, 2012