deepGTTM-I: Local Boundaries Analyzer based on A Deep Learning Technique

Masatoshi Hamanaka[1] Keiji Hirata[2], and Satoshi Tojo[3]

[1] Kyoto University,
hamanaka@kuhp.kyoto-u.ac.jp,
[2] Future University Hakodate,
hirata@fun.ac.jp,
[3] JAIST,
tojo@jaist.ac.jp,

http://gttm.jp/

Abstract. This paper describes a method that enables us to detect the local boundaries of a generative theory of tonal music (GTTM). Although systems that enable us to automatically acquire local boundaries have been proposed such as a full automatic time-span tree analyzer (FATTA) or σGTTM, musicologists have to correct the boundaries because of numerous errors. In light of this, we propose a novel method called deepGTTM-I for detecting the local boundaries of GTTM by using a deep learning technique. The experimental results demonstrated that deepGTTM-I outperformed the previous analyzers for GTTM in an F-measure of detecting local boundaries.

Keywords: A generative theory of tonal music (GTTM), local grouping boundary, deep learning.

1 Introduction

We propose a method of automatically acquiring local grouping boundaries based on a generative theory of tonal music (GTTM) [1]. GTTM is composed of four modules, each of which assigns a separate structural description to a listener's understanding of a piece of music. These four modules output a grouping structure, metrical structure, time-span tree, and prolongational tree. As the acquisition of local grouping boundaries is the first step in GTTM, an extremely accurate analyzer makes it possible to improve the performance of all the later analyzers.

We previously constructed several analyzers or methods that enabled us to acquire local grouping boundaries such as: an automatic time-span tree analyzer (ATTA) [5], a fully automatic time-span tree analyzer (FATTA) [6], a GTTM analyzer by using statistical learning (σGTTM) [7], and a GTTM analyzer based on clustering and statistical learning (σGTTMII) [8]. However, the performance of these analyzers or methods was inadequate in that musicologists had to correct the boundaries because of numerous errors.

We propose deepGTTM-I in which we attempted to use deep learning [9] to improve the performance of acquiring local grouping boundaries to detect them.

Unsupervised training in the deep learning of deep layered networks called pre-training helps supervised training, which is called fine-tuning [10].

Our goal was to develop a GTTM analyzer that enabled us to output the results obtained from analysis that were the same as those obtained by musicologists based on deep learning by learning the results of analysis obtained by musicologists. We had to consider three issues in constructing a GTTM analyzer based on deep learning.

- Multi-task Learning
  A model or network in a simple learning task estimates the label from an input feature vector. However, local grouping boundaries can be found in many note transitions. Therefore, we consider a single learning task as estimating whether one note transition can be a boundary or not. Then, a problem in detecting local grouping boundaries can be solved by using multi-task learning.
  Subsection 4.3 explains multi-task learning by using deep learning.

- Large scale training data
  Large scale training data are needed to train a deep layered network. Labels are not needed in pre-training the network. Therefore, we collected 15,000 pieces of music formatted in musicXML from Web pages that were introduced in the MusicXML page of MakeMusic Inc. [11]. We needed labeled data to fine-tune the network. Although we had 300 pieces with labels in the GTTM database [12], this number was too small to enable the network to learn.
  Subsection 4.1 explains how we collected the data and how we got the network to learn effectively with a small dataset.

- GTTM rules
  GTTM consists of multiple rules and a note transition that is applied to many rules tends to be a local grouping boundary in the analysis of local grouping boundaries. As a result of analysis by musicologists, 300 pieces in the GTTM database were not only labeled with local grouping boundaries, but also labeled with applied positions of grouping preference rules. Therefore, the applied positions of grouping preference rules were helpful clues in detecting local grouping boundaries.
  Subsection 4.3 explains how the network learned with the grouping preference rules.

The results obtained from an experiment demonstrated that multi-task learning using the deep learning technique outperformed the previous GTTM analyzers in grouping boundaries.

The paper is organized as follows. Section 2 describes related work and Section 3 explains our method called deepGTTM-I. Section 4 explains how we evaluated the performance of deepGTTM-I and Section 5 concludes with a summary and an overview of future work.

2 Related work

We consider GTTM to be the most promising of the many theories that have been proposed [2–4], in terms of its ability to formalize musical knowledge, because GTTM captures the aspects of musical phenomena based on the Gestalt occurring in music and is presented with relatively rigid rules. We have been constructing both systems of analysis and application of GTTM for more than a decade (Fig. 1) [13]. The horizontal axis in Fig. 1 indicates years. Above the timeline are analyzers or methods that we developed.

2.1 System of Analysis for GTTM based on Full Parameterization

We first constructed a grouping structure analyzer and metrical structure analyzer (Figs. 1a and b). We developed an ATTA (Fig. 1c) [5] by integrating a grouping structure analyzer and a metrical analyzer. We extended the GTTM by full externalization and parameterization and proposed a machine-executable extension of the GTTM, exGTTM. We implemented the exGTTM on a computer that we call ATTA The ATTA had 46 adjusted parameters to control the strength of each rule. The ATTA we developed enabled us to control the priority of rules, which enabled us to obtain extremely accurate groupings and metrical structures. However, we needed musical knowledge like that which musicologists have to properly tune the parameters.

FATTA [6] (Fig. 1d) did not have to tune the parameters because it automatically calculated the stability of structures and optimized the parameters so that the structures would be stable. FATTA achieved excellent analysis results for metrical structures, but results for grouping structures and time-span trees were unacceptable.

We constructed an interactive GTTM analyzer [14] (Fig. 1e) that enabled seamless changes in the automatic analysis and manual editing processes because it was difficult to construct an analyzer that could output analysis results in the same way as musicologists. The interactive GTTM analyzer is still used to collect GTTM analysis data and everyone can download and use it for free [15].

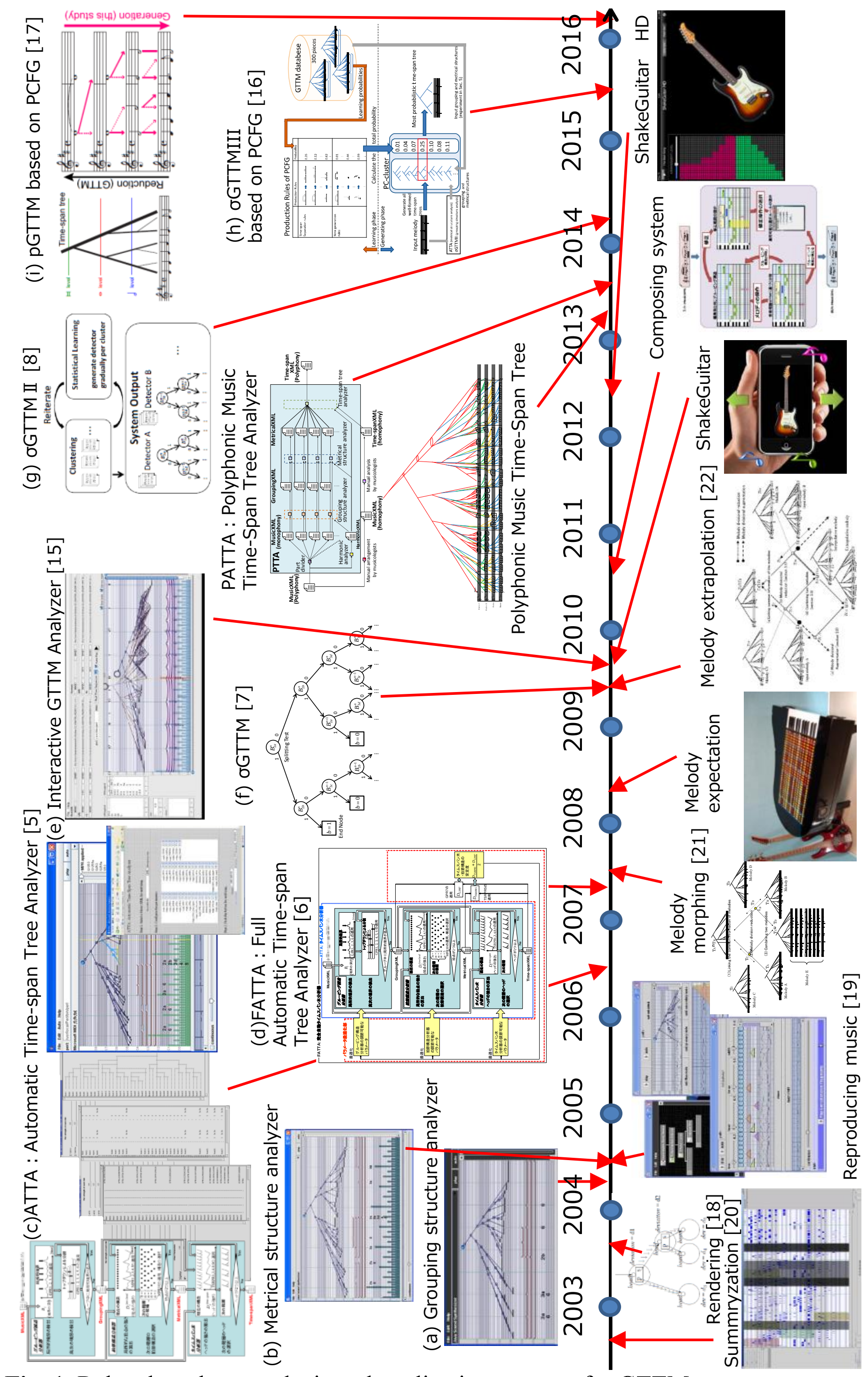
However, all these systems or methods [5, 6, 14, 15] had problems. ATTA needed musical knowledge to tune the parameters. FATTA performed poorly.

2.2 System of Analysis for GTTM based on statistical learning

σ GTTM [7] (Fig. 1f) enabled us to automatically detect local grouping boundaries by using a decision tree. Although σGTTM performed better than FATTA, it was worse than ATTA after the ATTA parameters had been tuned.

σ GTTMII [8] (Fig. 1g) had clustering steps for learning the decision tree and it outperformed ATTA if we could manually select the best decision tree. Although σGTTMII performed the best in detecting grouping boundaries, it was difficult to select the proper decision tree without musical knowledge.

σ GTTMIII [16] (Fig. 1h) enabled us to automatically analyze time-span trees by learning with a time-span tree of 300 pieces from the GTTM database [12] based on probabilistic context-free grammar (PCFG). σGTTMIII performed the best in



(i) pGTTM based on PCFG [17]
Reduction (GTTM)
Generation (this study)
(h) σGTTMIII based on PCFG [16]
(g) σGTTM II [8]
Clustering
Reiterate
Statistical Learning
System Output
PATTA : Polyphonic Music Time-Span Tree Analyzer
Polyphonic Music Time-Span Tree
(e) Interactive GTTM Analyzer [15]
(f) σGTTM [7]
(c)ATTA : Automatic Time-span Tree Analyzer [5]
(d)FATTA : Full Automatic Time-span Tree Analyzer [6]
(b) Metrical structure analyzer
(a) Grouping structure analyzer
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
ShakeGuitar HD
Composing system
ShakeGuitar
Melody extrapolation [22]
Melody expectation
Melody morphing [21]
Reproducing music [19]
Rendering [18]
Summryzation [20]

Fig. 1. Related work on analysis and application systems for GTTM.

acquiring time-span trees. pGTTM [17] (Fig. 1i) also used PCFG and we used it to attempt unsupervised learning. The main advantages of σGTTMIII and pGTTM were that the systems could learn the contexts in difference hierarchies of the structures (e.g., beats were important in the leaves of time-span trees, or chords were important near the roots of the trees.).

However, all these systems or methods [7, 8, 16, 17] had problems with detecting local grouping boundaries. σGTTM III and the pGTTM were focused on acquiring time-span trees and could not acquire local grouping boundaries.σGTTM II needed musical knowledge to select the decision tree. As σGTTM and the σGTTM II used rules that musicologists applied, they could not work as standalone analyzers. For example, information on parallel phrases is needed when detecting local grouping boundaries because parallel phrases create parallel structures in GTTM. However, σGTTM and σGTTM II do not have processes for acquiring parallel phrases.

We introduced deep learning to analyzing GTTM to solve these problems.

2.3 Application System by Using Analysis Results of GTTM

There are applications that we constructed under the time-line in Fig. 1 to use the results from analysis of GTTM. The time-span and prolongational trees provide performance rendering [18] and music reproduction [19] and provide a summarization of the music. This summarization can be used as a representation of a search, resulting in music retrieval systems [20]. It can also be used for melody morphing, which generates an intermediate melody between two melodies in systematic order [21, 22].

These systems presently need a time-span tree analyzed by musicologists because our analyzers do not perform optimally.

2.4 Melody Segmentation

As conventional methods of melody segmentation such as the Grouper of the Melisma Music Analyzer by Temperley [23] and the local boundary detection model (LBDM) by Cambouropoulos [24] require the user to make manual adjustments to the parameters, they are not completely automatic. Although Temperley [25] has also employed a probabilistic model, it has not been applied to melody segmentation. The unsupervised learning model (IDyOM) proposed by Pearce et al. makes no use of the rules of music theory with regard to melodic phrases, and it has performed as well as Grouper and LBDM [26]. However, as deepGTTM-I statistically and collectively learns all the rules for the grouping structure analysis of GTTM, we expect that deepGTTM-I will perform better than a model that only uses statistical learning.

3 GTTM and Its Implementation Problems

Figure 2 Shows local grouping boundaries, a grouping structure, a metrical structure, a timespan tree, and a prolongational tree (Fig. 2). The detection of local grouping boundaries in the grouping structure corresponds to melody segmentation.

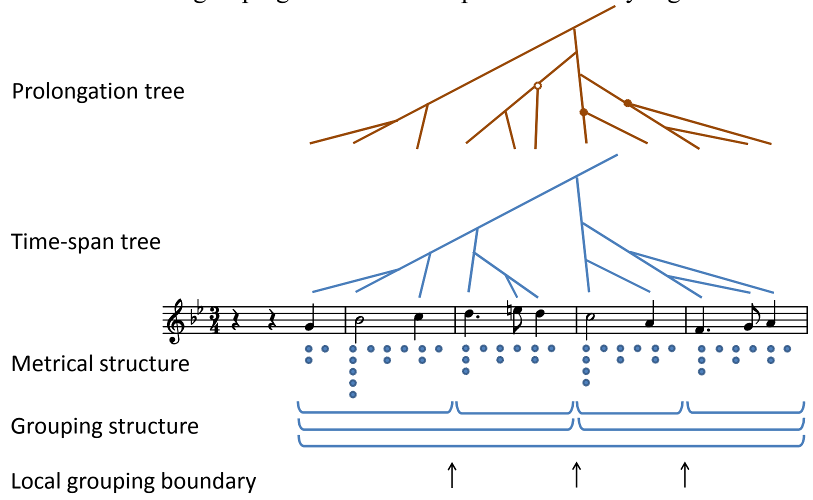


Fig. 2. Local grouping boundaries, grouping structure, metrical structure, time-span tree, and prolongational tree.

3.1 Grouping Preference Rules

The grouping structure is intended to formalize the intuitive belief that tonal music is organized into groups that are in turn composed of subgroups. These groups are presented graphically as several levels of arcs below a music staff. There are two types of rules for grouping in GTTM, i.e., grouping well-formedness rules (GWFRs) and grouping preference rules (GPRs). GWFRs are necessary conditions for the assignment of a grouping structure and restrictions on these structures. When more than one structure can satisfy the well-formedness rules of grouping, GPRs indicate the superiority of one structure over another. The GPRs consist of seven rules: GPR1 (alternative form), GPR2 (proximity), GPR3 (change), GPR4 (intensification), GPR5 (symmetry), GPR6 (parallelism), and GPR7 (time-span and prolongational stability). GPR2 has two cases: (a) (slur/rest) and (b) (attack-point). GPR3 has four cases: (a) (register), (b) (dynamics), (c) (articulation), and (d) (length).

3.2 Conflict Between Rules

Because there is no strict order for applying GPRs, a conflict between rules often occurs when applying GPRs, which results in ambiguities in analysis. Figure 3 outlines a simple example of the conflict between GPR2b (attack-point) and GPR3a (register). GPR2b states that a relatively greater interval of time between attack points initiates a grouping boundary. GPR3a states that a relatively greater difference in pitch between smaller neighboring intervals initiates a grouping boundary. Because GPR1 (alternative form) strongly prefers that note 3 alone does not form a group, a boundary cannot be perceived at both 2-3 and 3-4.

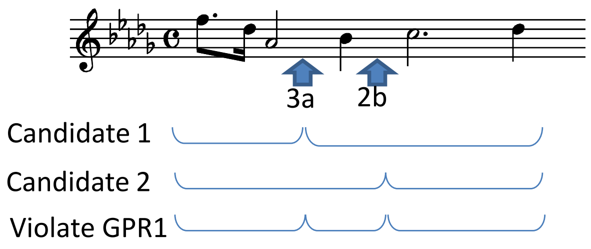


Fig. 3. Simple example of conflict between rules.

3.3 Ambiguity in defining GPR4, 5, and 6

GTTM does not resolve much of the ambiguity that exists in applying GPR4, 5, and 6. For example, GPR6 (Parallelism) does not define the decision criteria for construing whether two or more segments are parallel or not. The same problems occur with GPR4（Intensification） and GPR5 (Symmetry).

4 deepGTTM-I: local grouping boundary analyzer based on deep learning

We introduced deep learning to analyze the structure of GTTM and solve the problems described in Subsections 3.2 and 3.3. There were two main advantages of introducing deep learning.

- Learning rules applications
  We constructed a deep layered network that could output whether each rule was applicable or not on each note transition by learning the relationship between the scores and positions of applied grouping preference rules with the deep learning technique.
  Previous analysis systems based on GTTM were constructed by a human researcher or programmer. As described in Subsection 3.3, some rules in GTTM are very ambiguous and the implementations of these rules might differ depending on the person.
  However, deepGTTM-I is a learning based system where the quality of the analyzer depends on the training data and trained network.

- Learning priority of rules
  σGTTM and σGTTMII do not work well because they only determine the priority of rules from applied rules because the priority of rules depends on the

context of a piece. The input of the network in deepGTTM-I, on the other hand, is the score and it learns the priority of the rules as the weight and bias of the network based on the context of the score.

This section describes how we detected the local grouping boundaries by using deep learning.

4.1 Datasets for training

Three types of datasets were used to train the network, i.e., a non-labeled dataset for pre-training, a half labeled dataset, and a labeled dataset for fine-tuning.

(a) Non-labeled dataset. The network in pre-training learned the features of the music. A large scale dataset with no labels was needed. Therefore, we collected, 15,000 pieces of music formatted in musicXML from Web pages that were introduced on the musicXML page of MakeMusic Inc. [11] (Fig. 3a). The musicXMLs were downloaded in three steps.
1) Web autopilot script made a list of urls that probably downloaded musicXMLs in five links from the musicXML page of MakeMusic Inc.
2) The files in the url list were downloaded after urls had been omitted that were clearly not musicXML.
3) All the downloaded files were opened using the script, and files that were not musicXML were deleted.

(b) Half Labeled Dataset. The network in fine-tuning learned with the labeled dataset. We had 300 pieces with a labeled dataset in the GTTM database, which included musicXML with positions of local grouping boundaries, and positions to which the grouping preference rules were applied. However, 300 pieces were insufficient for deep learning.
Consequently, we constructed a half labeled dataset. We automatically added the labels of six applied rules of GPR2a, 2b, 3a, 3b, 3c, and 3d, because these rules could be uniquely applied as a score. We used ATTA to add labels to these rules (Fig. 3b).

(c) Labeled dataset. We artificially increased the labeled dataset, because 300 pieces in the GTTM database were insufficient for training a deep layered network. First, we transposed the pieces for all 12 keys. Then, we changed the length of note values to two times, four times, eight times, a half time, a quarter time, and an eighth time. Thus, the total labeled dataset had 25,200 (= 300x12x7) pieces (Fig. 3c).

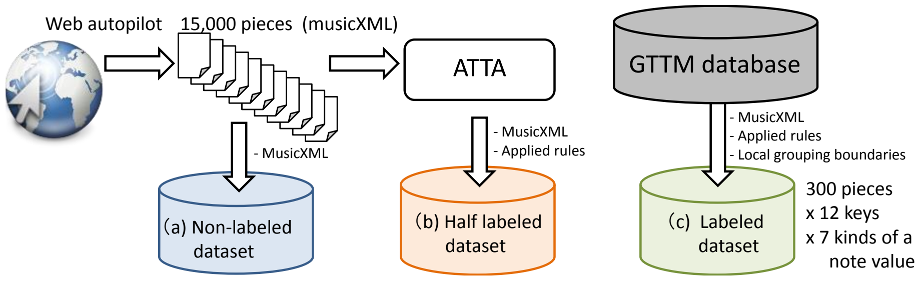


Fig. 4. Non-labeled dataset, half labeled dataset, and labeled dataset.

4.2 Deep Belief Network

We used a deep belief network (DBN) to detect the local grouping boundaries (Fig. 5). Figure 5 outlines the structure for the DBN we used. The input of DBN was the onset time, offset time, pitch, and velocity of note sequences from musicXML. The output of DBN formed multi-tasking learning, which had 11 outputs, such as 10 kinds of grouping preference rules (GPR2a, 2b, 3a, 3b, 3c, 4, 5, 6, and 7) and local grouping

4.3 Multidimensional multi-task learning

The DBN that we introduced in Subsection 4.2 was a very complex network. The fine-tuning of local grouping boundaries was a multi-task learning itself. The fine-tuning of each grouping preference rule also involved multi-task learning. Therefore, the fine-tuning of grouping preference rules involved multidimensional multi-task learning.

Multi-task learning. The processing flow for the multi-task learning of a grouping preference rule or local grouping boundaries involved four steps.
Step 1: The order of the pieces of training data was randomly shuffled and a piece was selected from top to bottom.
Step 2: The note transition of the selected piece was randomly shuffled and a note transition was selected from top to bottom.
Step 3: Back propagation from output to input was carried out in which the note transition had a boundary or the rule was applied (=1) or not (=0).
Step 4: The next note transition was repeated or the next piece in steps 2 and 1.
Multidimensional multi-task learning. The processing flow for the multidimensional multi-task learning of grouping preference rules involved three steps.
Step 1: The order of grouping preference rules was randomly shuffled and a rule was selected from top to bottom.
Step 2: Multi-task learning of the selected grouping preference rule was carried out.
Step 3: The next rules in step 1 were repeated.

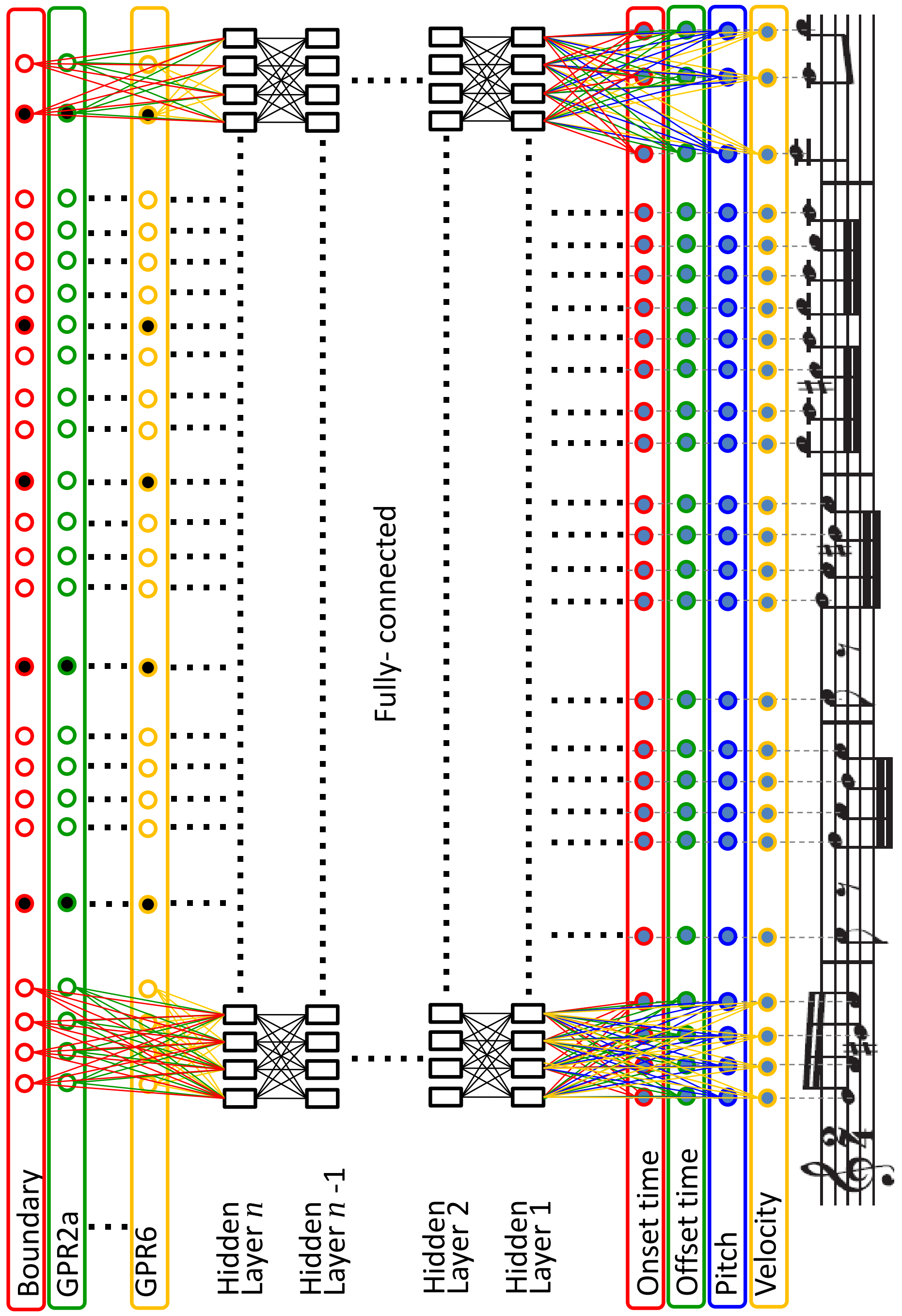


Fig. 5. Deep belief network for detect local grouping boundaries.

5 Experimental Results

We evaluated the performance of deepGTTM-I by using 100 pieces from the GTTM database where the remaining 200 pieces were used to train the network. Table 1 summarizes the results for a network that had 11 layers with 3000 units.

Table 1. Performance of ATTA, *σ* GTTM, *σ* GTTMII, and deepGTTM-I.

	Precision *P*	Recall *R*	F measure
ATTA with manual editing of parameters	0.737	0.441	0.552
σGTTM	0.467	0.736	0.571
σGTTMII with manual selection of decision tree	0.684	0.916	0.783
deepGTTM-I	0.784	0.814	0.799

The results indicate deepGTTM-I outperformed the previous analyzers in the F-measure. ATTA had adjustable parameters and σGTTMII could select the decision tree. The performance of ATTA and σGTTMII changed depending on the parameters or decision trees. Table 1 indicates the best performance was achieved by manual editing. However, as σGTTM and deepGTTM-I had no parameters for editing, deepGTTM-I performed extremely robustly.

6 Conclusion

We developed a local grouping boundaries analyzer called deepGTTM-I that was based on deep learning. We proposed multidimensional multi-task learning that efficiently learned local grouping boundaries and grouping preference rules by sharing the network. We prepared three kinds of datasets to learn the network, such as non-labeled, half labeled, and labeled datasets because labeled datasets were very limited and some labels of GPR2 and 3 could automatically acquire the previous analyzer of GTTM. After a network that had 11 layers with 3000 units had been trained, deepGTTM-I outperformed the previously developed analyzers for local grouping boundaries in the F measure.

This work was the first step in implementing GTTM by using deep learning. We plan to implement a complete analysis of GTTM by using deep learning. We also plan to analyze the network after local grouping boundaries are learned.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Number 25700036, 16H01744, 23500145.

References

1. Lerdahl, F. and Jackendoff, R.: *A Generative Theory of Tonal Music.* MIT Press (1985)
2. Cooper, G. and Meyer, L. B. *The Rhythmic Structure of Music*. The University of Chicago Press (1960)
3. Narmour, E. *The Analysis and Cognition of Basic Melodic Structure*. The University of Chicago Press (1990)
4. Temperley, D. *The Cognition of Basic Musical Structures*. MIT press, Cambridge (2001)
5. Hamanaka, M., Hirata, K., and Tojo, S.: Implementing 'a generative theory of tonal music', *Journal of New Music Research*, 35(4), 249–277 (2006)
6. Hamanaka, M., Hirata, K., and Tojo, S.: FATTA: Full automatic time-span tree analyzer, *In: Proceedings of the 2007 International Computer Music Conference* (ICMC2007), pp. 153–156 (2007)
7. Miura, Y., Hamanaka, M., Hirata, K., and Tojo, S.: Decision tree to detect GTTM group boundaries, *In: Proceedings of the 2009 International Computer Music Conference* (ICMC2009), pp. 125–128 (2009)
8. Kanamori, K. and Hamanaka, M.: Method to Detect GTTM Local Grouping Boundaries based on Clustering and Statistical Learning, *In: Proceedings of the 2014 International Computer Music Conference* (ICMC2014), pp. 125–128 (2014)
9. Hinton, G. E., Osindero, S., and The Y. W.: A fast learning algorithm for deep belief nets, *Neural computation*, Vol. 18, No. 7, pp. 1527–1554 (2006)
10. Erhan, D., Bengio, Y., Courville, A., Manzagol, A. P., Vincent, P., and Bengio, S.: Why does Unsupervised Pre-training Help Deep Learning?, Journal of Machine Learning Research, pp. 626–660 (2010)
11. MakeMusic Inc.: Music in MusicXML Format, url: http://www.musicxml.com/music-in-musicxml/, accessed on 2016-2-28.
12. Hamanaka, M., Hirata, K., and Tojo, S.: Musical Structural Analysis Database Based on GTTM, *In: Proceeding of the 2014 International Society for Music Information Retrieval Conference* (ISMIR2014), pp. 325–330 (2014)
13. Hamanaka ,M., Hirata, K., and Tojo, S.: Implementing Methods for Analysing Music Based on Lerdahl and Jackendoff's Generative Theory of Tonal Music, *Computational Music Analysis* (pp. 221–249), Springer (2016)
14. Hamanaka, M., Hirata, K., and Tojo, S.: Interactive GTTM Analyzer, *In: Proceedings of the 10th International Conference on Music Information Retrieval Conference* (ISMIR2009), pp. 291–296 (2009)
15. Hamanaka, M.: Interactive GTTM Analyzer/GTTM Database, url http://gttm.jp, see at 2016-2-28.
16. Hamanaka, M., Hirata, K., and Tojo, S.: σ GTTM III: Learning-based Time-span Tree Generator Based on PCFG, *In: Proceedings of the 11th International Symposium on Computer Music Multidisciplinary Research* (CMMR 2015), pp. 303–317 (2015)
17. Nakamura E., Hamanaka M., Hirata K., and Yoshii K.: Tree-Structured Probabilistic Model of Monophonic Written Music Based on the Generative Theory of Tonal Music, *In: Proceedings of 41st IEEE International Conference on Acousitcs, Speech and Signal Processing* (ICASSP2016), 2016.
18. Hirata, K. and Hiraga R.: Ha-Hi-Hun plays Chopin's Etude, In *Working Notes of IJCAI-03 Workshop on methods for automatic music performance and their applications in a public rendering contest* (2003)

19. Hirata, K., Matsuda, S., Kaji K., and Nagao K.: Annotated Music for Retrieval, Reproduction, and Sharing, *In: Proceedings of the 2004 International Computer Music Conference* (ICMC2004), pp. 584–587 (2004)
20. Hirata K. and Matsuda S.: Interactive Music Summarization based on GTTM, *In: Proceeding of the 2002 International Society for Music Information Retrieval Conference* (ISMIR2002), pp. 86–93 (2002)
21. Hamanaka, M., Hirata, K., and Tojo, S.: Melody Morphing Method based on GTTM. *In: Proceedings of the 2008 International Computer Music Conference* (ICMC2008), pp. 155–158 (2008)
22. Hamanaka, M., Hirata, K., and Tojo, S.: Melody Extrapolation in GTTM Approach. *In: Proceedings of the 2009 International Computer Music Conference* (ICMC2009), pp. 89–92 (2009)
23. Temperley, D. The Melisma Music Analyzer. http://www.link.cs.cmu.edu/music-analysis/ (2003)
24. Cambouropoulos, E. :The Local Boundary Detection Model (LBDM) and its application in the study of expressive timing, *In: Proceedings of the International Computer Music Conference* (ICMC2001), pp. 290–293 (2001)
25. Temperley, D. *Music and Probability*. Cambridge: The MIT Press (2007)
26. Pearce, M. T., Müllensiefen, D., and Wiggins, G. A.: A comparison of statistical and rule-based models of melodic segmentation, *In: Proceedings of the International Conference on Music Information Retrieval* (ISMIR2008), pp. 89–94 (2008)