ROMAN NUMERAL ANALYSIS WITH GRAPH NEURAL NETWORKS: ONSET-WISE PREDICTIONS FROM NOTE-WISE FEATURES

Emmanouil Karystinaios[1] **Gerhard Widmer**[1,2]
[1] Institute of Computational Perception, Johannes Kepler University Linz, Austria
[2] LIT AI Lab, Linz Institute of Technology, Austria
firstname.lastname@jku.at

ABSTRACT

Roman Numeral analysis is the important task of identifying chords and their functional context in pieces of tonal music. This paper presents a new approach to automatic Roman Numeral analysis in symbolic music. While existing techniques rely on an intermediate lossy representation of the score, we propose a new method based on Graph Neural Networks (GNNs) that enable the direct description and processing of each individual note in the score. The proposed architecture can leverage notewise features and interdependencies between notes but yield onset-wise representation by virtue of our novel edge contraction algorithm. Our results demonstrate that *ChordGNN* outperforms existing state-of-the-art models, achieving higher accuracy in Roman Numeral analysis on the reference datasets. In addition, we investigate variants of our model using proposed techniques such as NADE, and post-processing of the chord predictions. The full source code for this work is available at `https://github.com/manoskary/chordgnn`

1. INTRODUCTION

Automatic Chord Recognition is one of the core problems in Music Information Retrieval. The task consists of identifying the harmonies or chords present in a musical piece. Various methods have been proposed to address this task using either an audio or symbolic representation of the music [1]. In the symbolic domain, most approaches focus on the related and arguably more complex problem of Automatic Roman Numeral Analysis, which is a functional harmony analysis problem that has its roots in musicological research of Western classical music.

Roman Numeral Analysis is a notational system used in music theory to analyze chord progressions and identify the relationship between chords in a given key. In this system, each chord in a piece of music is assigned a Roman numeral based on its position within the key's scale. For example, in the key of C major, the I chord is C major, the IV chord is F major, and the V chord is G major. Roman Numerals are an important tool for understanding and analyzing the harmonic structure of music, and they are a valuable resource for musicians, composers, and arrangers alike.

In Music Information Retrieval, a lot of work has been done to automate Roman Numeral analysis. However, current approaches still face significant challenges. Some of these are related to the large chord symbol vocabulary. A common way to address this problem is to divide a Roman Numeral into several components (e.g., key, degree, inversion) and transform the analysis into a multitask learning scenario. However, multitask approaches themselves face challenges with interdependencies among tasks. Lastly, Roman Numeral analysis faces a score representation problem related to existing models such as CNNs whose inputs must be in fixed-sized chunks. Recent state-of-the-art approaches follow an audio-inspired strategy, dividing a musical score into fixed-length time frames ("windows") which are then processed by a Convolutional Recurrent Neural Network (CRNN). However, such a representation is unnatural for scores and has the added practical disadvantage of being time-limited (for example regarding notes extending beyond the current window) and, due to the fixed-length (in terms of score time) constraint, capturing varying amounts of musically relevant context.

In this paper, we propose a new method for automatic Roman Numeral analysis based on Graph Neural Networks that can leverage note-wise information to address the score representation issue. Our model, *ChordGNN*, builds on top of existing multitask approaches but introduces several novel aspects, including a graph convolutional architecture with an edge contraction pooling layer that combines convolution at the note level but yields the learned representation at the onset level.

Our proposed method, *ChordGNN*, is evaluated on a large dataset of Western classical music, and the experimental results demonstrate that it outperforms existing state-of-the-art methods, in terms of the commonly used Chord Symbol Recall measure. To address the interdependencies among tasks we investigate the effect of post-processing and other proposed techniques such as NADE and gradient normalization. Finally, we look at a qualitative musical example and compare our model's predictions with other state-of-the-art models.

Attribution: E. Karystinaios and G. Widmer, "Roman Numeral Analysis with Graph Neural Networks: Onset-wise Predictions from Note-wise Features", in *Proc. of the 24th Int. Society for Music Information Retrieval Conf.,* Milan, Italy, 2023.

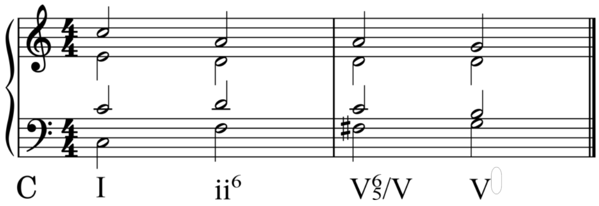

Figure 1. A Roman Numeral analysis for two bars for four-part harmony in C major. Capital letters stand for major quality and lowercase for minor quality. The third chord has a dominant seven as its primary degree and the dominant of C major as its secondary degree. The V_5^6 indicates a major with a seven quality in second inversion. The bass (lowest chord note) of that chord is F sharp, the root is D, and the local key is C major.

2. RELATED WORK

There is a big body of literature covering the topic of Automatic Chord Recognition applied in the audio domain; however, in our work, we focus on the problem of automatic Roman Numeral Analysis in the symbolic domain. It consists of labeling the chords and harmonic progressions in a piece of music using Roman Numerals, where each numeral represents a chord built on a particular scale degree. Numerous approaches have tried to automate Roman Numeral analysis or infer harmonic relations between chords. Notable work includes statistical models such as *Melisma* [2], HMM-based models [3], and grammar-based approaches [4].

In recent years, research has shifted towards a deep learning and data-driven approach. Due to the large vocabulary of possible Roman Numerals, the problem has been divided into several component subtasks, thus resulting in a multitask learning setting [5]. As a multitask problem, a Roman Numeral is characterized by the following components: the primary and secondary degree (as illustrated in Figure 1), the local key at the time point of prediction, the root of the chord, the inversion of the chord, and the quality (such as major, minor, 7, etc.). Although the root can be derived from the other components, it was pointed out by [6] that redundancy is assisting Roman Numeral analysis systems to learn. An example of Roman Numerals and their components can be viewed in Figure 1. Recent state-of-the-art approaches decompose the numeral prediction task to the simultaneous prediction of those 6 components [5–9].

Most deep learning approaches to Roman Numeral analysis are inspired by work in audio classification, cutting a score into fixed-size chunks (in terms of some constant score time unit; e.g., a 32nd note) and using these as input to deep models. Using this quantized time frame representation, [6] introduced a CRNN architecture to predict Roman Numerals. Other work has continued to build on the latter by introducing more tasks to improve performance such as the *AugmentedNet* model [7], or introducing intradependent layers to inform in an orderly fashion the prediction of one task with the previously predicted task, such as the model introduced by [8]. Other architectures, such as the CSM-T model, have demonstrated good results by introducing modular networks which treat a score as a sequence of notes ordered first by onset and then by pitch [9].

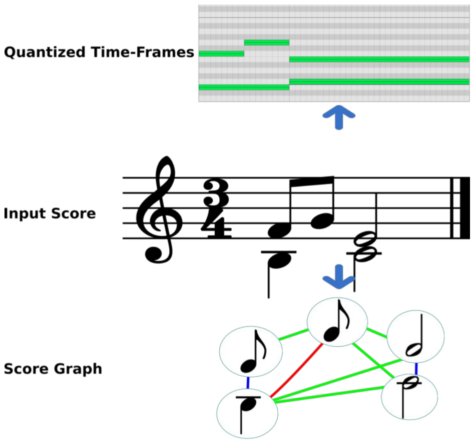

Figure 2. Different representations of the score excerpt shown in the middle. Top: quantized time frame representation, bottom: graph representation.

Should a musicologist perform music analysis on a piece of music, they would consider the individual notes existing in the score. Thus, a time frame representation would come across as unnatural for symbolic music and in particular for such an analysis task. In this paper, we present a method that no longer treats the score as a series of quantized frames but rather as a partially ordered set of notes connected by the relations between them, i.e., a graph. A visual comparison of the two representations is shown in Figure 2. Recently, modeling scores as graphs has also been demonstrated to be beneficial for problems such as expressive performance generation [10], cadence detection [11], voice separation [12], or boundary detection [13].

Automatic Roman Numeral analysis, as a multitask problem, is mostly tackled with hard parameter-sharing models. These models share part of the model across all tasks as an encoder, and then the common embeddings are branched to a classification model per task [6–8]. However, some approaches separate tasks from this paradigm to a more modular or soft parameter sharing approach [9].

In the field of multitask learning, a lot of research has been done on the problem of conflicting gradients during backpropagation in hard parameter-sharing models. Issues with multi-objective optimization have been early addressed by Zhang et al. [14] and recent solutions have been proposed for the multitask setting in the form of dynamic task prioritization [15], gradient normalization [16], rotation matrices [17], or even game-theoretic approaches [18]. In our work, we experimentally evaluate some of these techniques in the multitask setting to investigate whether Roman Numeral analysis subtasks conflict with each other (see Section 5.2).

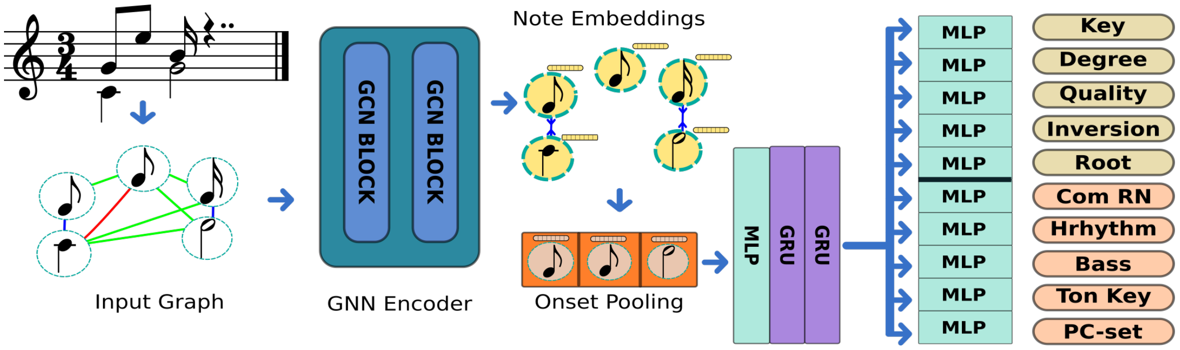

Figure 3. The proposed Architecture Chord-GNN

3. METHODOLOGY

3.1 Roman Numeral Analysis

We already discussed, in Section 2, how Roman Numeral analysis can be viewed as a multi-task problem. In this section, we describe in detail the additional tasks introduced by [7] that we also use for training and prediction. First, let us assume that the prediction can be broken down into specific time points, and each time point is attributed to a unique onset in the score.

The Roman Numeral prediction can be viewed as a simultaneous prediction of the local key, degree (primary and secondary), quality, inversion, and root. Each one of these tasks is a categorical, multiclass classification problem. However, [7] indicated that only three tasks would be sufficient for 98% of the Roman Numeral annotations in our dataset (detailed in Section 4.1). These three tasks comprise the prediction of a restricted vocabulary of common Roman Numeral symbols in combination with the local key and the inversion. We refer to Roman Numeral prediction involving the 5 tasks as *conventional RN*, and the combined prediction of key, inversion, and restricted RN vocabulary *alternative RN*, as RN_{alt}, in accordance with [7].

Several other tasks have been introduced that have been shown to improve the performance of related models [7]. These include the Harmonic Rhythm, which is used to infer the duration of a Roman Numeral at a given time point; the Tonicization task, a multiclass classification task that refers to a tonicized key implied by the Roman Numeral label and is complementary to the local key; the Pitch Class Sets task, which includes a vocabulary of different pitch class sets, and the Bass task, which aims to predict the lowest note in the Roman Numeral label.

3.2 Graph Representation of Scores

Our approach to automatic Roman Numeral analysis no longer treats the score as a sequence of quantized time frames but rather as a graph, which permits us to specify note-wise information such as pitch spelling, duration, and metrical position. We use graph convolution to model interdependencies between notes. We model our score generally following Karystinaios and Widmer [11], but we opt for a heterogeneous graph convolution approach, i.e., including different edge relations/types. Furthermore, we develop an edge contraction pooling layer that learns onset-wise representations from the note-wise embeddings and therefore yields a sequence.

After the edge contraction, we follow [6–8] by adding to the graph convolution a sequence model for the hard-sharing part of our model, and simple shallow multi-layer perceptron heads for each task. In essence, we replace the CNN encoder that works on quantized frames of the score in previous approaches, with a graph convolutional encoder followed by an edge contraction layer. Our proposed architecture is shown in Figure 3.

The input to the GNN encoder is an attributed graph $G = (V, E, X)$ where V and E denote its node and edge sets and X represents the node feature matrix, which contains the features of the notes in the score. For our model, we used pitch spelling, note duration, and metrical position features.

Given a musical piece, the graph-building process creates a set of edges E, with different relation types $\mathcal{R}$. A labeled edge (u, r, v) of type r between two notes u, v belongs to E if the following conditions are met:

- notes starting at the same time: $on(u) = on(v) \rightarrow r = \text{onset}$
- note starting while the other is sounding: $on(u) > on(v) \wedge on(u) \leq on(v) + dur(v) \rightarrow r = \text{during}$
- note starting when the other ends: $on(u) + dur(u) = on(v) \rightarrow r = \text{follow}$
- note starting after a time frame when no note is sounding: $on(u) + dur(u) < on(v) \wedge \nexists v' \in V,\ on(v') < on(v) \wedge on(v') > on(u) + dur(u) \rightarrow r = \text{silence}$

3.3 Model

In this section, we introduce and describe *ChordGNN*, a Graph Convolutional and Recurrent Neural Network. The structure of the network is visually outlined in Figure 3. *ChordGNN* uses heterogeneous graphSAGE [19] convolutional blocks defined as:

$$\begin{aligned} \mathbf{h}^{(l+1)}_{\mathcal{N}_r(v)} &= \text{mean}\left(\{\mathbf{h}^{l}_{u}, \forall u \in \mathcal{N}_r(v)\}\right) \\ \mathbf{h}^{(l+1)}_{v_r} &= \sigma\left(W \cdot \text{concat}(\mathbf{h}^{l}_{v}, \mathbf{h}^{l+1}_{\mathcal{N}_r(v)})\right) \\ \mathbf{h}^{(l+1)}_{v} &= \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} \mathbf{h}^{(l+1)}_{v_r} \end{aligned} \tag{1}$$

where $\mathbf{h}_v^{(0)} = \mathbf{x}_v$ and $\mathbf{x}_u$ is the input features for node u, $\mathcal{N}(u)$ are the neighbors of node u, and σ is a ReLU activation function. We name the output representations of all nodes after graphSAGE convolution $H = \{h_u^{(L)} \mid u \in V\}$ where L is the total number of convolutional layers.

Given the hidden representation H of all nodes, and onset edges $E_{\text{On}} = \{(u, v) \mid on(u) = on(v)\}$, the onset edge contraction pooling is described by the following equations: first, we update the hidden representation with a learned weight, $H' = HW^{(\text{cpool})}$. Subsequently we need to unify the representations for every node u, such that $\forall v \in \mathcal{N}_{\text{On}}(v),\ h_u^{(\text{cp})} = h_v^{(\text{cp})}$:

$$h_u^{(\text{cp})} = h_u + \sum_{v \in \mathcal{N}_{\text{On}}(v)} h_v \tag{2}$$

where, h_u and h_v belong to H'. Subsequently, we filter the vertices:

$$V' = \{v \in V \mid \forall u \in V,\ (v, u) \in E_{\text{On}} \implies u \notin V'\} \tag{3}$$

Therefore, $H^{(cp)} = \{h_u^{(cp)} \mid \forall u \in V'\}$ are the representations obtained. Sorting the representations by the onset on which they are attributed we obtain a sequence $S = [h_{u_1}^{(cp)}, h_{u_2}^{(cp)}, \dots h_{u_k}^{(cp)}]$ such that $on(u_1) < on(u_2) < \dots < on(u_k)$.

The sequence S is then passed through an MLP layer and 2 GRU layers. This concludes the hard-sharing part of our model. Thereafter, an MLP head is attached per task, as shown in Figure 3.

For training, we use the dynamically weighted loss introduced by [20]. The total loss $\mathcal{L}_{tot}$ of our network is calculated as a weighted sum of the individual losses for every task, where the weights are learned during training:

$$\mathcal{L}_{\text{tot}} = \sum_{t \in \mathcal{T}} \mathcal{L}_t * \frac{1}{2\gamma_t^2} + \log(1 + \gamma_t^2) \tag{4}$$

where $\mathcal{T}$ is the set of tasks; $\mathcal{L}_t$ is the cross-entropy loss relating to task t; the γ_t are learned scalars that give the weight for each task t; and the log expression is a regularization term [20].

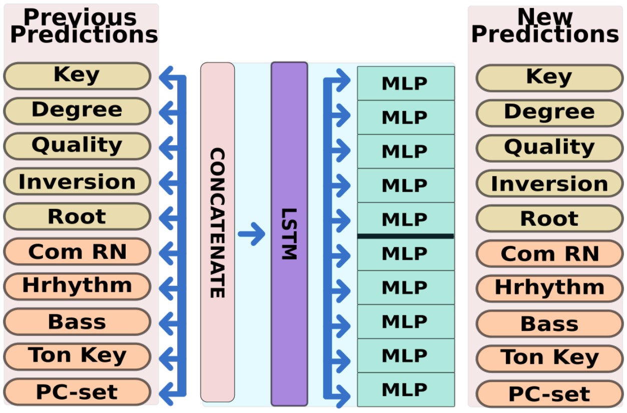

Figure 4. Post-processing of Roman Numeral predictions.

3.3.1 *Post-processing*

We enhance our model with a post-processing phase after the model has been trained. The post-processing phase combines the logits of all tasks' predictions by concatenating them and, then, feeds them to a single-layer bidirectional LTSM block. Then, again the embeddings of the sequential block are distributed to 11 one-layer MLPs, one for each task. The post-processing block is sketched in Figure 4.

4. EXPERIMENTS AND CORPORA

In the experiments, we compare our model, *ChordGNN*, with other recent models for automatic Roman Numeral analysis. We run experiments with our model in the exact same way as described in the paper [7], including the specific data splits, so that our results are directly comparable to the figures reported there. A detailed comparison of the results will be given in Table 1. Furthermore, we develop variants of our model using proposed techniques such as NADE [8], and post-processing of the chord predictions. We report a configuration study of our model on the use of gradient normalization techniques and NADE that should improve results on Multi-Task learning scenarios and avoid common Multi-Task Learning problems such as conflicting gradients. Lastly, we compare our model with the updated version *v1.9.1* of the state-of-the-art model Augmented-Net [21] and datasets.

4.1 Datasets

For training and evaluation, we combined six data sources into a single "Full" Dataset of Roman Numeral annotations in accordance with [7]: the Annotated Beethoven Corpus (ABC) [22]; the annotated Beethoven Piano Sonatas (BPS) dataset [5]; the Haydn String Quartets dataset (HaydnSun) [23]; the TAVERN dataset [24]; a part of the When-in-Rome (WiR) dataset [25, 26]; and the Well-Tempered-Clavier (WTC) dataset [25] which is also part of the WiR dataset.

Training and test splits for the full dataset were also provided by [7]. It is worth noting that the BPS subset splits were already predefined in [5]. In total, approximately 300 pieces were used for training, and 56 pieces were used for testing, proportionally taken from all the different data sources. We draw a distinction for the BPS test set, which includes 32 Sonata first movements and for which we ran an additional experiment. The full test set also includes the 7 Beethoven piano sonatas.

In addition to the above datasets, we include data augmentations identical to the ones described in [7]: texturization and transposition. The texturization is based on a dataset augmentation technique introduced by [27]. The transposition augmentation boils down to transposing a score to all the keys that lie within a range of key signatures that have up to 7 flats or sharps. It should be noted that the augmentations are only applied in the training split.

For our last experiment (to be reported on in Section 5.3 below), we add additional data that were recently introduced by [21]. The additional data include the annotated Mozart

	Model	Key	Degree	Quality	Inversion	Root	RN	RN (Onset)	RN_{alt}
BPS	Micchi (2020)	82.9	68.3	76.6	72.0	-	42.8	-	-
	CSM-T (2021)	69.4	-	-	-	75.4	45.9	-	-
	AugNet (2021)	**85.0**	**73.4**	**79.0**	73.4	**84.4**	45.4	-	49.3
	ChordGNN (Ours)	79.9	71.1	74.8	75.7	82.3	46.2	46.6	48.6
	ChordGNN+Post (Ours)	82.0	71.5	74.1	**76.5**	82.5	**49.1**	**49.4**	**50.4**
Full	AugNet (2021)	**82.9**	67.0	**79.7**	78.8	83.0	46.4	-	51.5
	ChordGNN (Ours)	80.9	70.1	78.4	78.8	84.8	48.9	48.4	50.4
	ChordGNN+Post (Ours)	81.3	**71.4**	78.4	**80.3**	**84.9**	**51.8**	**51.2**	**52.9**

Table 1. Model comparison on two different test sets, the Beethoven Piano Sonatas (BPS), and the full test set. RN stands for Roman Numeral, RN_{alt} for the alternative Roman Numeral computations discussed in Section 3.1. $RN(Onset)$ refers to onset-wise prediction accuracy, all other scores use the CSR score (see Section 5). Note that model *CSM-T* reports *Mode* instead of *Quality*.

Piano Sonatas (MPS) dataset [28] for which we also applied the aforementioned augmentations.

4.2 Configuration

For all our experiments, we train our network with the AdamW optimizer. We fix our architecture with a hidden size of 256, a learning rate of 0.0015, a weight decay of 0.005, and a dropout of 0.5 which is applied to each learning block of our architecture.

5. RESULTS

As an evaluation metric, we use Chord Symbol Recall (CSR) [29] where for each piece, the proportion of time is collected during which the estimated label matches the ground truth label. We apply the CSR at the 32nd note granularity level, in accordance with [6, 7, 9].

5.1 Quantitative Results

In the first experiment, which compares our *ChordGNN* to existing state-of-the-art approaches, we evaluate the full dataset, but also the annotated Beethoven Piano Sonatas (BPS) [5] subset, which many previous approaches had also used. The results are shown in Table 1. We present the CSR scores (where they are applicable) for Local Key, Degree, Quality, Inversion, Root, conventional Roman Numeral, and Alternative Roman Numeral (see Section 3). Furthermore, we include the onset-wise accuracy score for our models' conventional Roman Numeral predictions.

On the BPS subset, we compare our model *ChordGNN* with the Micchi (2020) model [6], the *CSM-T* (2021) model [9] and the *AugmentedNet* 2021 model [7]. Our results on Roman Numeral prediction surpass all previous approaches. Note that the *AugmentedNet* model exhibits higher prediction scores on the individual Key, Degree, Quality, and Root tasks, which are used jointly for the prediction of the Roman numeral. These results indicate that our model obtains more meaningfully interrelated predictions, with respect to the Roman numeral prediction, resulting in a higher accuracy score.

Moreover, we compare *ChordGNN* to *AugmentedNet* on the full test dataset. Our model surpasses *AugmentedNet* with and without post-processing in all fields apart from local key prediction and quality. Our model obtains up to 11.6% improvement in conventional Roman Numeral prediction.

In both experiments, post-processing has been shown to improve both RN and RN_{alt}. However, *ChordGNN* without post-processing already surpasses the other models.

Variant	RN	RN_{alt}
ChordGNN (Baseline)	46.1 ± 0.003	47.8 ± 0.007
ChordGNN + WLoss	$\mathbf{48.9} \pm 0.001$	$\mathbf{50.4} \pm 0.010$
ChordGNN + Rotograd	45.5 ± 0.003	47.1 ± 0.005
ChordGNN + R-GradN	45.2 ± 0.006	46.7 ± 0.005
ChordGNN + NADE	48.2 ± 0.005	49.9 ± 0.005

Table 2. Configuration Study: Chord Symbol Recall on Roman Numeral analysis on the full test set. RN stands for Roman Numeral, RN_{alt} refers to the alternative Roman Numeral computations discussed in section 3.1. WLoss stands for the dynamically weighted loss described in Section 3, and R-GradN stands for Rotograd with Gradient Normalization. Every experiment is repeated 5 times with the same ChordGNN model as Table 1 without post-processing.

5.2 Configuration Study

For a systematic study of multitask training, we investigated the effects of extension modules, gradient normalization techniques, and learnable weight loss. In detail, we test 5 configurations using as baseline the *ChordGNN* model (without post-processing) with standard CE loss and no weighing. Furthermore, we test our proposed architecture using the dynamically weighted loss described in Section 3.3 (same as the model in Table 1), Rotograd [17] and GradNorm [16] for Gradient Normalization, and NADE [8]. The models are run on the Full data set described above and averaged over five runs with random initialization. The results, summarized in Table 2, suggest that using the dynamically weighted loss yields better results compared to other methods such as the Baseline or Gradient Normalization techniques. Furthermore, the dynamically weighted loss is comparable to NADE but also more robust on Conventional Roman Numeral prediction on our datasets.

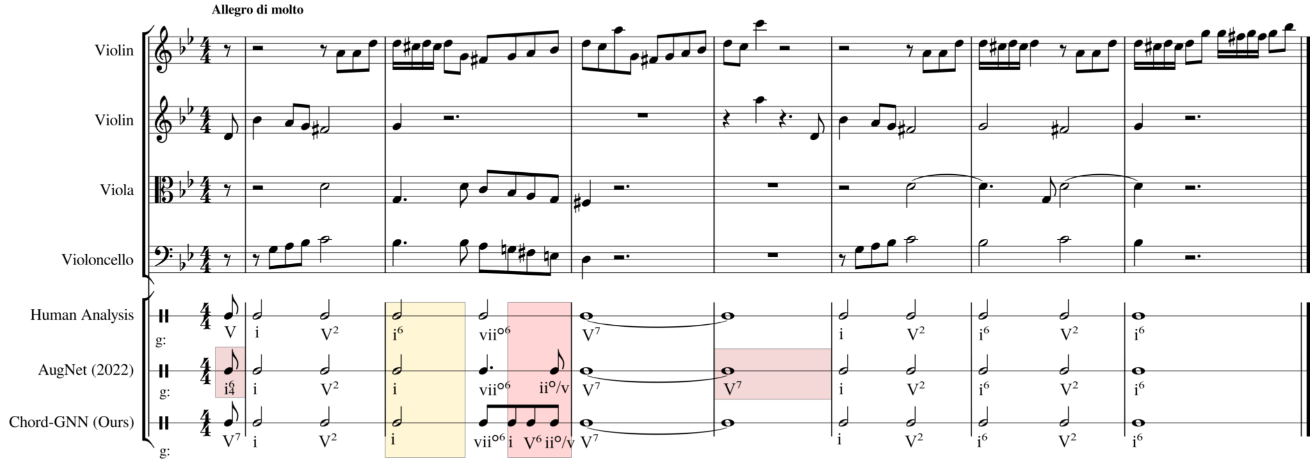
Allegro di molto
Violin
Violin
Viola
Violoncello
Human Analysis
AugNet (2022)
Chord-GNN (Ours)
g:
V i V2 i6 vii°6 V7 i V2 i6 V2 i6
i6/4 i V2 i vii°6 ii°/v V7 V7 i V2 i6 V2 i6
V7 i V2 i vii°6 i V6 ii°/v V7 i V2 i6 V2 i6

Figure 5. A comparison between the human annotation, AugmentedNet, and ChordGNN on a passage of Haydn's string quartet op.20 No.3 movement 4. The red (wrong) markings on Human Analysis and AugNet (2022) are from [21]

5.3 Latest developments

Our last experiment focuses on specific developments that have very recently been published in Nápoles López's Ph.D. thesis [21]. In the thesis, three additional tasks, related to predicting the components of a canonical representation of the current chord, as implied by the Roman Numeral, were proposed and the dataset was extended with the Annotated Mozart Piano Sonatas (MPS) corpus [28], as mentioned in Section 4.1 above.

To test the relevance of these updates, we trained an adapted version of our model, now with 11+3=14 individual tasks and including the Mozart data. It turns out that the updated model improves significantly in performance, achieving a 53.5 CSR score on conventional Roman Numeral (compare this to row "ChordGNN (Ours)" in Table 1). Furthermore, post-processing can improve the results by up to two additional percentage points. [1]

5.4 A Musical Example

In Figure 5, we look at a comparison between the human annotations, *AugmentedNet* and *Chord-GNN* predictions (The musical excerpt is taken from Nápoles López's thesis [21], and the predictions relate to the new models trained as described in the previous section.). Marked in red are false predictions, and marked in yellow are correct predictions of the model with wrong ground-truth annotations. Both models' predictions are very similar to the human analysis. However, our model correctly predicts the initial pickup measure annotation. In measure 2, the ground truth annotation marks a tonic in first inversion; however, the viola at that point is lower than the cello and therefore the chord is actually in root position. Both models obtain a correct prediction at that point. Subsequently, our model predicts a harmonic rhythm of eighth notes, which disagrees with the annotator's half-note marking. Analyzing the underlying harmony in that passage, we can justify our model's choices.

The human annotation suggests that the entire second half of the 2nd measure represents a vii^o chord. However, it should not be in the first inversion, as the cello plays an F# as the lowest note (which is the root of vii^o). The AugNet analysis faces the same issue, in contrast with the predictions of ChordGNN. However, there are two conflicting interpretations of the segment. First, the vii^o on the third beat is seen as a passing chord between the surrounding tonic chords, leading to a dominant chord in the next measure. Alternatively, the vii^o could already be part of a prolonged dominant harmony (with passing chords on the offbeats) leading to the V^7. The ChordGNN solution accommodates both interpretations as it doesn't attempt to group chords at a higher level, treating each eighth note as an individual chord rather than a passing event. The other two solutions prefer the second option.

6. CONCLUSION

In this paper, we presented *ChordGNN*, a model for automatic Roman Numeral analysis in symbolic music, based on a note-level, graph-based score representation. We showed that *ChordGNN* improves on other state-of-the-art models, and that post-processing can further improve the accuracy of the predictions. A configuration study suggests that gradient normalization techniques or techniques for carrying prediction information across tasks are not particularly beneficial or necessary for such a model.

Follow-up work will focus on strengthening the robustness of our models by pre-training with self-supervised methods on large corpora. We believe that such pre-training can be beneficial for learning helpful intrinsic musical information. Such a step is crucial since more data improves predictions but Roman Numeral annotations are hard to find or produce. Moreover, we aim to enrich the number of tasks for joint prediction by including higher-level analytical targets such as cadence detection and phrase boundary detection. Finally, we aim to extend our method to the audio domain.

[1] Unfortunately, we cannot directly compare these numbers to [21], as their results are not reported in comparable terms.

7. ACKNOWLEDGEMENTS

We gratefully acknowledge the musical analysis of the vii^o passage in Fig. 5 (Section 5.4) that was offered by an anonymous reviewer, and which we took the liberty of adopting for our text. This work is supported by the European Research Council (ERC) under the EU's Horizon 2020 research & innovation programme, grant agreement No. 101019375 ("Whither Music?"), and the Federal State of Upper Austria (LIT AI Lab).

8. REFERENCES

[1] J. Pauwels, K. O'Hanlon, E. Gómez, M. Sandler *et al.*, "20 years of Automatic Chord Recognition from Audio," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2019.

[2] D. Temperley, *The cognition of basic musical structures*. MIT press, 2004.

[3] C. Raphael and J. Stoddard, "Functional Harmonic Analysis Using Probabilistic Models," *Computer Music Journal*, vol. 28, no. 3, pp. 45–52, 2004.

[4] J. P. Magalhaes and W. B. de Haas, "Functional Modelling of Musical Harmony: an experience report," *ACM SIGPLAN Notices*, vol. 46, no. 9, pp. 156–162, 2011.

[5] T.-P. Chen, L. Su *et al.*, "Functional Harmony Recognition of Symbolic Music Data with Multi-task Recurrent Neural Networks." in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2018.

[6] G. Micchi, M. Gotham, and M. Giraud, "Not all roads lead to Rome: Pitch representation and model architecture for automatic harmonic analysis," *Transactions of the International Society for Music Information Retrieval (TISMIR)*, vol. 3, no. 1, pp. 42–54, 2020.

[7] N. Nápoles López, M. Gotham, and I. Fujinaga, "AugmentedNet: A Roman Numeral Analysis Network with Synthetic Training Examples and Additional Tonal Tasks." in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[8] G. Micchi, K. Kosta, G. Medeot, and P. Chanquion, "A deep learning method for enforcing coherence in Automatic Chord Recognition." in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[9] A. P. McLeod and M. A. Rohrmeier, "A modular system for the harmonic analysis of musical scores using a large vocabulary," in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[10] D. Jeong, T. Kwon, Y. Kim, and J. Nam, "Graph Neural Network for Music Score Data and Modeling Expressive Piano Performance," in *Proceedings of International Conference on Machine Learning (ICML)*, 2019.

[11] E. Karystinaios and G. Widmer, "Cadence Detection in Symbolic Classical Music using Graph Neural Networks," *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2022.

[12] E. Karystinaios, F. Foscarin, and G. Widmer, "Musical Voice Separation as Link Prediction: Modeling a Musical Perception Task as a Multi-Trajectory Tracking Problem," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2023.

[13] C. Hernandez-Olivan, S. R. Llamas, and J. R. Beltran, "Symbolic Music Structure Analysis with Graph Representations and Changepoint Detection Methods," 2023.

[14] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "Facial Landmark Detection by Deep Multi-task Learning," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.

[15] M. Guo, A. Haque, D.-A. Huang, S. Yeung, and L. Fei-Fei, "Dynamic Task Prioritization for Multitask Learning," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[16] Z. Chen, V. Badrinarayanan, C.-Y. Lee, and A. Rabinovich, "GradNorm: Gradient Normalization for Adaptive Loss Balancing in Deep Multitask Networks," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

[17] A. Javaloy and I. Valera, "RotoGrad: Gradient Homogenization in Multitask Learning," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022.

[18] A. Navon, A. Shamsian, I. Achituve, H. Maron, K. Kawaguchi, G. Chechik, and E. Fetaya, "Multi-task Learning as a Bargaining Game," *Proceedings of the International Conference on Machine Learning (ICML)*, 2022.

[19] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.

[20] L. Liebel and M. Körner, "Auxiliary tasks in multi-task learning," *arXiv preprint arXiv:1805.06334*, 2018.

[21] N. Nápoles López, "Automatic roman numeral analysis in symbolic music representations," Ph.D. dissertation, Schulich School of Music McGill University, December 2022.

[22] M. Neuwirth, D. Harasim, F. C. Moss, and M. Rohrmeier, "The Annotated Beethoven Corpus (ABC): A dataset of harmonic analyses of all Beethoven string quartets," *Frontiers in Digital Humanities*, vol. 5, p. 16, 2018.

[23] N. Nápoles López, "Automatic Harmonic Analysis of Classical String Quartets from Symbolic Score," Ph.D. dissertation, Master's thesis, Universitat Pompeu Fabra, 2017.

[24] J. Devaney, C. Arthur, N. Condit-Schultz, and K. Nisula, “Theme and Variation Encodings with Roman Numerals (TAVERN): A new data set for symbolic music analysis,” in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2015.

[25] M. Gotham, D. Tymoczko, and M. S. Cuthbert, “The RomanText Format: A Flexible and Standard Method for Representing Roman Numeral Analyses.” in *Proceedings of the International Society for Music Information Retrieval Conference (ISMIR)*, 2019.

[26] M. R. H. Gotham and P. Jonas, “The Openscore Lieder Corpus,” in *Proceedings of the Music Encoding Conference (MEC)*, 2021.

[27] N. Nápoles López and I. Fujinaga, “Harmonic Reductions as a Strategy for Creative Data Augmentation,” in *Late-Breaking Demo at International Society for Music Information Retrieval Conference (ISMIR)*, 2020.

[28] J. Hentschel, M. Neuwirth, and M. Rohrmeier, “The Annotated Mozart Sonatas: Score, Harmony, and Cadence,” *Transactions of the International Society for Music Information Retrieval (TISMIR)*, vol. 4, no. ARTICLE, pp. 67–80, 2021.

[29] C. Harte, “Towards automatic extraction of harmony information from music signals,” Ph.D. dissertation, Queen Mary University of London, 2010.