STATS 402 - Interdisciplinary Data Analysis AI Charting for Music Game Cytoid

Yuchen Song
Email: ys396@duke.edu

***Abstract*—Creating custom levels for the music game Cytoid requires significant time and expertise, limiting community participation. This project proposes a machine learning solution to automate the charting process by generating Cytoid charts from audio files and desired difficulty levels. The methodology involves preprocessing audio data into Mel spectrograms, integrating numerical difficulty levels with audio features, and designing a BiLSTM architecture to predict key types and placements. Various threshold selection methods and model architectures, including BiLSTM, CNN, and Transformer models, were explored to optimize prediction performance. Despite initial challenges with model training and predictive accuracy, the CNN-based approach demonstrated promising results with F1 scores reaching up to 64%. The evaluation metrics indicate areas for improvement, particularly in enhancing model training and addressing class imbalances. Future work will focus on refining model architectures and exploring advanced techniques to further enhance predictive capabilities.**

I. Introduction

The creation of custom levels, or "charts," for music games like Cytoid[24] is a meticulous process that demands substantial time and specialized knowledge. This complexity acts as a barrier to community-driven content creation, limiting the diversity and availability of game levels. To address this challenge, this project explores the application of machine learning techniques to automate the charting process, thereby democratizing content creation and enriching the player experience.

The primary objective is to develop a system that can generate accurate and engaging Cytoid charts[23] from input audio files and specified difficulty levels. This involves processing audio data, extracting relevant features, and employing predictive models to determine the placement and type of notes within the game. The project evaluates various model architectures, including BiLSTM, CNN, and Transformer models, however, most of them does not yield satisfactory results.

This paper is organized as follows: Section II reviews related work in music related generation and automated music charting. Section III details the proposed methodology, including data preprocessing, model design, and training procedures. Section IV presents the performance evaluation of different models, accompanied by experimental results and analysis. Finally, Section V concludes the paper and outlines directions for future research.

The contributions of this project tested several deep learning based approach to automated chart generation, an evaluation for these models tested, and the development of a publicly available code for extracting music information and preprocessing for cytoid dataset for future works.

II. Related Work

Research at the intersection of deep learning and music analysis has rapidly expanded, yielding improved methods for music understanding, generation, and pattern recognition [3], [11]. Within this domain, three key areas are particularly relevant to automated chart generation: (1) audio feature extraction and representation learning, (2) musical pattern recognition and structure analysis, and (3) applications of deep learning for rhythm game chart generation.

Basic deep learning techniques have proven effective at learning hierarchical features directly from raw audio, reducing reliance on manual feature engineering [9], [10]. Convolutional Neural Networks (CNNs) and other architectures can capture both local spectral patterns and larger-scale musical structures, enabling tasks like onset detection, beat tracking, chord recognition, and instrument classification [18], [14], [21]. Leveraging representations such as Mel spectrograms, these models can automatically discover relevant audio cues to guide higher-level decisions.

LSTM and Bi-LSTM networks have been particularly successful at modeling temporal dependencies in music, making them well-suited for tasks like chord progression prediction, melodic pattern recognition, and rhythmic structure analysis [5], [7], [15]. LSTM networks effectively address the vanishing gradient problem found in traditional RNNs through their specialized gate mechanisms (input, forget, and output gates) [6] and the network's ability to selectively remember or forget information makes it particularly effective at learning musical patterns and structures. By preserving temporal context over long sequences, LSTMs can maintain musical coherence and effectively encode both global musical structure and local detail [16], [19]. Bi-directional LSTMs process sequences in both forward and backward directions, enabling a more comprehensive understanding of musical context. This bidirectional processing is particularly valuable for capturing complex musical relationships [8]. Attention-based models, like Transformers, have also shown potential in capturing longer-term dependencies and stylistic patterns [17], however, their performance varies depending on the specific downstream task.

Several studies have applied deep learning to automate chart creation, significantly reducing the time and expertise required for custom level design. Dance Dance Convolution (DDC) [4]

pioneered this approach, using CNN-LSTM architectures to predict step placements for dance rhythm games. TaikoNation (TN) [20] explored Bi-LSTMs to generate human-like chart patterns aligned with musical motifs, improving structural consistency. Genelive [13] integrated beat-guided features and multi-scale CNNs to enhance note placement accuracy, demonstrating that local and multi-resolution temporal features are critical to producing charts that feel natural and musically coherent.

This project builds on the foundation established by these prior efforts. By trying out architectures such as Bi-LSTMs, CNNs, and Transformers, we aim to generate charts for the music game Cytoid. Our approach integrates lessons learned from the literature: CNNs' strengths in local temporal feature extraction, BiLSTM' capabilities to model sequence dependencies, and experimenting with attention-based methods to handle more complex temporal patterns. Although most of the trials have proved unsatisfactory, the experience are precious.

III. Methodology

The methodology involves data preprocessing and the development of various models to predict different aspects of musical notes. Initially, a multitask Bi-LSTM model was designed; however, due to unsatisfactory results, the approach was revised to employ separate models for predicting note presence, type, and position. This section outlines the data preprocessing steps and the distinct models used for each prediction task.

A. Data Preprocessing

1) Audio Data Preprocessing for Model Input: To prepare the raw audio and chart data for training, a preprocessing pipeline was established. This pipeline transforms the raw data of audio files and Cytoid-compatible JSON charts into matrices and vectors compatible with the models, utilizing techniques such as the Short-Time Fourier Transform (STFT) and one-hot encoding.

Each audio file is converted into a Mel spectrogram representation. The spectrograms are generated using a window size of 32 ms and a hop size of 23 ms, resulting in overlapping audio segments for better frequency resolution. The Mel spectrograms are computed using 128 Mel bands, capturing the critical frequency nuances of the audio signals. STFT is applied during this process to segment the audio into overlapping frames, while the Mel scale enhances frequency representation [20].

To fully utilize the temporal context and information, each spectrogram frame is augmented with a sliding window that includes five preceding and five following frames. This creates a final feature size of $128 \times (2 \times 5 + 1) = 1408$ per frame, embedding approximately 115 ms of surrounding audio context. The spectrograms are then normalized to have zero mean and unit variance, ensuring stable training of the neural network.

2) Chart Data Processing for Model Output: The chart JSON files are processed to generate labels for each spectrogram frame. These label vectors encompass three types of information: binary indicators for note presence, categorical labels for note type (e.g., tap, hold, slide, with eight distinct classes encoded as one-hot vectors), and continuous values representing the note's position on the scanning line (scaled between 0.0 and 1.0). This alignment ensures that every spectrogram frame corresponds to a well-defined set of labels.

B. Multitask BiLSTM Model

Initially, a multitask Bi-LSTM model was developed to simultaneously predict note presence, type, and position. However, overlapping information among tasks led to suboptimal performance, necessitating the separation of the model into three distinct functions. Below is a description of the initial multitask model and the reasons for its revision. Figure 1 is a pipelined strcuture of the multitask model, for every frame predict the p

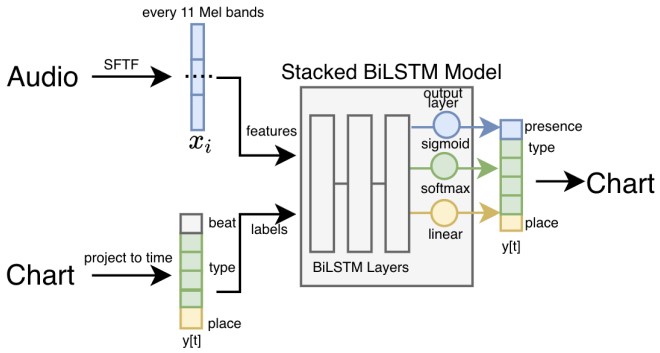

Fig. 1. Pipeline flow for the Multitask Stacked BiLSTM model

a) Model Structure: The multitask Bi-LSTM model consists of shared Bi-LSTM layers followed by separate dense layers for each prediction task: Presence Prediction, Type Prediction, and Position Prediction. The shared layers aim to capture common musical patterns, while the separate layers specialize in their respective tasks.

b) Loss Function: A combined loss function was employed, summing the losses from each prediction task. Binary cross-entropy was used for presence prediction, categorical cross-entropy for type prediction, and mean squared error for position prediction. The total loss was calculated as:

$$\mathcal{L}_{\text{total}} = \lambda_1 \mathcal{L}_{\text{presence}} + \lambda_2 \mathcal{L}_{\text{type}} + \mathcal{L}_{\text{position}}$$

where $\lambda_1 = \lambda_2 = 1$. This approach allows for balancing the contributions of each task during model training.

C. Models for Note Presence Prediction

To improve the accuracy of predicting whether a note is present at a given time, three different models were evaluated: Bi-LSTM based, Transformer based, and CNN based.

1) Bi-LSTM Based: The Bi-LSTM based model leverages the sequential nature of audio data to predict note presence. The input to the model consists of the preprocessed Mel spectrogram frames with surrounding context.

The model architecture includes an input layer that accepts vectors of size 1408 (128 Mel bands multiplied by 11 frames), followed by two Bi-directional LSTM layers with 256 units each to capture temporal dependencies in both forward and backward directions. Finally, a fully connected layer with a sigmoid activation function outputs the probability of note presence.

Binary Cross-Entropy (BCE) loss is used to measure the difference between predicted probabilities and true labels:

$$\mathcal{L}_{\text{BILSTM}} = -\frac{1}{B \cdot T} \sum_{i=1}^{B} \sum_{t=1}^{T} [y_{it} \log(\hat{y}_{it}) + (1 - y_{it}) \log(1 - \hat{y}_{it})]$$

Performance is assessed using Accuracy, Precision, Recall, and F1 score. Due to class imbalance, F1 score is used to balance precision and recall.

Two threshold selection methods were explored. The first method calculates the average threshold for each difficulty level across all songs and applies these thresholds to individual songs. The second method focuses on the distribution of notes within each difficulty level by selecting the top 'n' notes based on predicted probabilities. Both methods did not achieve satisfactory performance, indicating the need for more sophisticated thresholding techniques or model refinements. Effective thresholding is critical for converting model output probabilities into discrete note predictions. Two main methods were evaluated. The first method calculates the average threshold for each difficulty level across all songs and applies these thresholds to individual songs. The second method focuses on the distribution of notes within each difficulty level by selecting the top 'n' notes based on predicted probabilities. Despite these methods, the performance remained suboptimal, indicating the need for more sophisticated post-processing techniques or model refinements.

2) Transformer based: To address the limitations observed with LSTM models, a Transformer-based model was introduced. The Transformer architecture was implemented using an open-source framework, with hyperparameters carefully tuned to optimize performance. The input to the Transformer model consisted of 128 Mel bands processed over multiple time steps, allowing the model to capture complex temporal relationships in the audio data. The model was trained for 50 epochs with a batch size of 1 to promote stable convergence. Hyperparameter tuning was performed using Optuna, an automated hyperparameter optimization framework, to minimize the training loss.

3) CNN Based: The CNN-based model focuses on extracting local temporal features from the audio data to predict note presence. Difficulty is not considered here. After the two failed test of the threshold selection, only songs of difficulty level 15 is used for training, since they takes a relatively large portions of the songs.

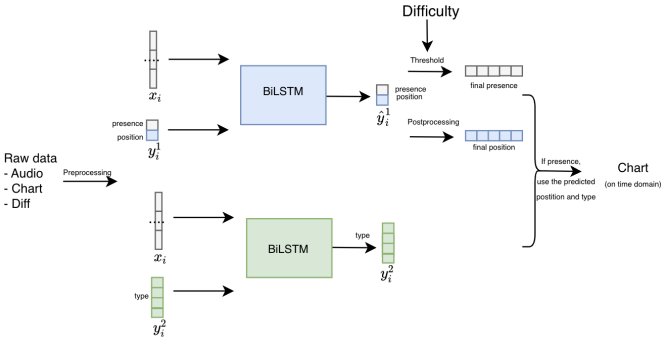

Fig. 2. Enter Caption

The CNN architecture includes several 1D convolutional layers operating along the time axis with ReLU activations and batch normalization, followed by max pooling layers to reduce the time dimension and summarize features. After stacking 3 convolution and pooling layers, the resulting feature maps are flattened and fed into fully connected layers. For presence prediction, the final layer uses a sigmoid activation function to output a single probability value indicating the presence of a note. Since its output also a number indicating the probability for note presence of the next frame, BCE loss is used to measure the difference between predicted probabilities and true labels. The threshold here is used 0.5. Performance is evaluated using Accuracy, Precision, Recall, and F1 score for presence prediction. F1 score is particularly important due to class imbalance.

D. Model for Note Type Prediction

Predicting the type of note requires distinguishing between different note categories based on the musical context.

1) Bi-LSTM Based: The Bi-LSTM based model for note type prediction builds upon the architecture used for presence prediction but focuses on classifying the type of each detected note. It used all frames as the training data, so there is a class imbalance between the empty frame (about 90%), and other type of notes. The input of the BiLSTM based note type predictor and the model layer is basically the same with the note presence predictor. However, what is different is the label and the loss function.Categorical Cross-Entropy loss is employed to measure the difference between predicted class probabilities and true labels:

$$\mathcal{L}_{\text{type\_BiLSTM}} = -\frac{1}{B \cdot T} \sum_{i=1}^{B} \sum_{t=1}^{T} \sum_{c=1}^{N} y_{itc} \log(\hat{y}_{itc})$$

Figure 2 Sis a pipeline for secondly proposed BiLTSM pipeline for note presence and type. After derived the presence notes, the type of the generated note uses the note type with the highest note type probability.

2) CNN Based: Building on the success of the CNN model for predicting note presence, I expanded the architecture to classify the type of each note. This enhanced model retains

a structure similar to the presence prediction model but is specifically designed for multi-class classification. The input remains consistent, consisting of 81 frames of the Mel spectrogram with each frame containing 128 Mel frequency bands, providing the temporal context necessary for the model to identify subtle variations that differentiate note types. Instead of generating a single probability score, the model produces a probability distribution across five distinct note types by applying a softmax activation function, ensuring that the probabilities sum to one and accurately reflect the likelihood of each category. The CNN architecture is composed of three convolutional layers, each followed by batch normalization, ReLU activation, and max-pooling. After these convolutional layers, the feature maps are flattened and passed through fully connected layers that include dropout regularization to prevent overfitting. The final layer uses a softmax activation function to output the probabilities for each note type. For the multi-class classification task, Cross-Entropy Loss is utilized as the loss function, effectively measuring the discrepancy between the predicted probability distribution and the true distribution. To evaluate the model's performance, metrics such as accuracy, precision, recall, and the F1 score are employed, providing a comprehensive assessment of its effectiveness.

IV. Results and Analysis

We evaluated multiple deep learning architectures for predicting note presence, type, and position in the music game Cytoid. The evaluation utilized two distinct datasets: CNN Models uses pproximately 150 songs with difficulty level 15, each around 3 minutes long. Other Models (Bi-LSTM and Transformer): About 60 single-tempo songs from the 'Z' folder.

The datasets are publicly available; however, the copyright of the charts and original songs remains with their respective creators. Our usage is strictly for educational purposes [22].

All models were assessed using standard metrics including Accuracy, Precision, Recall, and F1 Score. Additionally, confusion matrices were provided where applicable to visualize model biases. For a more intuitive understanding of model performance, we also included visualizations of predictions over short segments of test songs.

A. Multitask Model Performance

Our initial multitask architecture aimed to jointly predict note presence, type, and position. Figure 3 illustrates the model's predictions compared to the ground truth over a specific time interval. The red line represents the predicted note presence probabilities, while the blue dashed line indicates the actual note presence. Predicted note types are marked with colored circles, and true note types are denoted by crosses.

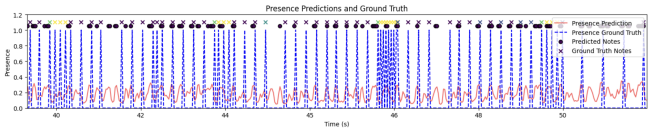

Fig. 3. Predicted Presence and Note Type

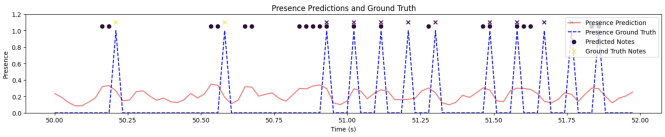

Fig. 4. Detailed Predicted Presence and Note Type

While the model successfully captures general trends in note presence, it predominantly predicts the most common note type as "none," as evidenced by the confusion matrix. Figure 5 shows the distribution of predicted note positions, which are mainly concentrated between 0 and 0.4. This indicates a limitation in accurately predicting higher position values, likely due to the characteristics of the loss function and the prevalence of zero values in the position data.

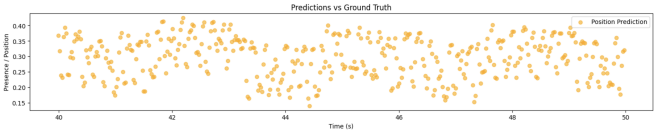

Fig. 5. Note Position Distribution

1) Note Type Prediction Issues: The confusion matrix revealed that the model tends to predict all notes as "none," primarily due to class imbalance in the training data. This issue necessitates further investigation and adjustments, such as data augmentation or class weighting, to enhance note type prediction accuracy.

2) Note Position Prediction Challenges: The predicted note positions are skewed towards lower values, suggesting that the model struggles to map audio features to spatial positions accurately. This may be attributed to: (1) Lack of Strong Positional Cues. The current feature set does not provide clear indicators for note positioning. (2) Insufficient Feature Engineering. Absence of features related to beat or tempo structure hampers precise positional alignment.

B. Separate Bi-LSTM Models

We further experimented with separate Bi-LSTM models for note presence and type prediction using the 60 single-tempo songs dataset.

1) Note Presence Prediction with Bi-LSTM: Table I presents the performance of the Bi-LSTM-based note presence prediction model under two threshold selection methods:(1)

Average Threshold per Difficulty Level: Determines an optimal threshold for each difficulty level based on training data. (2) Top-N Selection per Difficulty Level: Selects the top proportion of frames as notes based on predicted probabilities.

TABLE I
BI-LSTM NOTE PRESENCE PREDICTION ON VALIDATION SET

Method	Precision	Recall	F1-score	Accuracy
Average Threshold	0.22	0.53	0.31	0.69
Top-N Selection	0.27	0.28	0.28	0.81

TABLE II
CONFUSION MATRICES FOR NOTE PRESENCE PREDICTION (VALIDATION SET)

	Average Threshold		Top-N Selection	
	Pred=0	Pred=1	Pred=0	Pred=1
True=0	55,295	21,484	68,056	8,723
True=1	5,556	6,142	8,426	3,272

2) Analysis of Note Presence Prediction: The Bi-LSTM model achieved moderate accuracy but low Precision and Recall, resulting in F1 scores of 0.31 and 0.28 for the average threshold and top-N selection methods, respectively. The underperformance can be attributed to: (1) Class Imbalance. The dataset has a significant number of non-note frames compared to note frames, leading the model to favor predicting the majority class [12], [4]. (2) Limited Local Feature Specialization. Bi-LSTMs effectively capture temporal dependencies but are less adept at extracting localized spectral features critical for note onset detection. Convolutional architectures have been shown to excel in this area [2].

3) Note Type Classification with Bi-LSTM: Figure 6 visualizes the predicted note type probabilities alongside the ground truth. The Bi-LSTM model predominantly predicts the majority class, as shown in Table III.

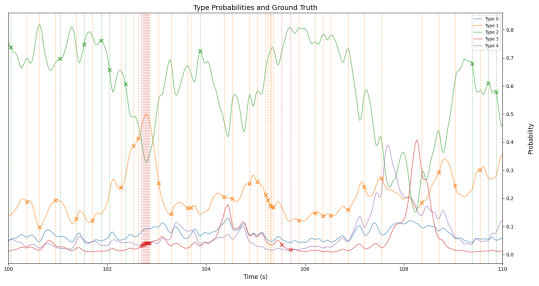

Fig. 6. Visualization of Note Type Probability with Ground Truth

TABLE III
BI-LSTM NOTE TYPE CLASSIFICATION REPORT

Class	Precision	Recall	F1-score
Empty frame	0.85	0.92	0.88
Type 1	0.10	0.05	0.07
Type 2	0.05	0.03	0.04
Type 3	0.07	0.04	0.05
Type 4	0.05	0.02	0.03

4) Analysis of Note Type Classification: The model's heavy bias towards the majority class (empty frames) resulted in poor F1 scores for minority note types. This issue arises from: Persistent Class Imbalance: Certain note types are underrepresented, causing the model to overfit to the majority class even if the focal loss is applied. (2) Limited Diversity in Training Data: Insufficient examples of each note type may hinder the model's ability to learn distinctive spectral and temporal patterns.

C. Transformer Models

Specifically, the model utilized a hidden size of 448, five Transformer layers, four attention heads, a dropout rate of 0.2, and a low learning rate of approximately 4.96×10^{-5}.

Despite extensive tuning and careful optimization, the Transformer model struggled to effectively discriminate between frames with and without notes. The probability distributions produced by the model were overly uniform and lacked the necessary temporal fluctuations required for accurate note presence detection. As a result, the Transformer achieved extremely low F1 scores, around 0%, indicating that the model failed to capture meaningful patterns in the data. This suggests that, in this specific application, the Transformer architecture may not be well-suited for the task of note presence prediction without further modifications or alternative approaches.

Transformer-based models were also explored but showed limited improvement over the Bi-LSTM baseline. Despite tuning several hyperparameters, the Transformer produced overly smooth probability distributions, resulting in near-random predictions with F1 scores close to 0%.

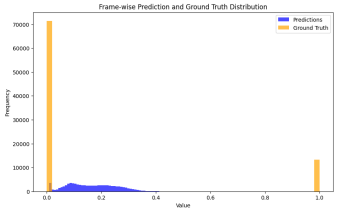

Fig. 7. Distribution of Frame Probabilities (Transformer Model)

1) Analysis of Transformer Model Performance: The poor performance of the Transformer model can be attributed to:Over-Smoothing of Predictions or insufficient data. The self-attention mechanism may have diluted relevant local cues, preventing sharp discrimination between note and non-note frames. Besides, the limited parameter search and small dataset

may have hindered the model's ability to capture meaningful patterns.

Given these challenges, the Transformer architecture may require more extensive tuning or a larger dataset to effectively model long-range dependencies in this task.

D. CNN-Based Models

Shifting to CNN architectures and utilizing a larger dataset significantly improved note presence prediction. The CNN models were trained on approximately 150 difficulty-level-15 songs, demonstrating better performance in capturing localized time-frequency features.

TABLE IV
PERFORMANCE METRICS FOR CNN-BASED MODELS

Task	Accuracy	Precision	Recall	F1 Score
Presence	90.39%	86.28%	51.66%	64.63%
Note Type	92.19%	86.13%	92.19%	89.06%

Figure 8 illustrates the CNN model's predictions over a song segment. The red plot indicates note presence probabilities, with a threshold of 0.5 determining note presence. The green line represents ground truth presence, and different colored dotted lines denote predicted note types.

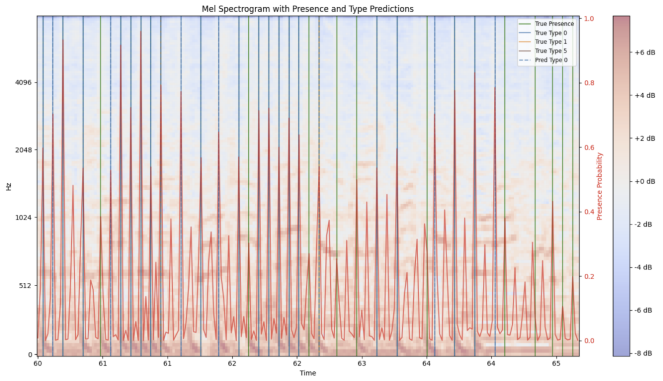

Fig. 8. CNN Note Prediction with Presence and Type

Despite improved presence detection, note type prediction still faced challenges due to class imbalance. The confusion matrix in Figure 9 shows that the CNN model frequently defaults to the most common note class.

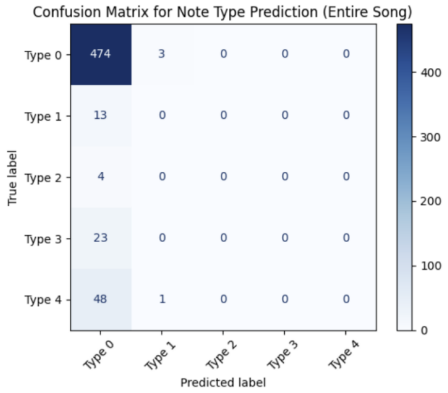

Fig. 9. Confusion Matrix of Predicted Note Types

1) Analysis of CNN-Based Models: The CNN model achieved significant improvements in note presence detection, with an F1 Score of 64.63%. This success is primarily due to CNN filters effectively capture localized temporal-frequency patterns, enhancing the differentiation between note and non-note frames. Besides, While class imbalance persists, the CNN's ability to recognize clear local patterns reduces the tendency to predict the majority class exclusively.

However, note type prediction remains suboptimal, primarily due to the same problems we have addressed before Certain note types are still underrepresented, leading the CNN to overfit to the majority class. Also, insufficient examples of each note type prevent the CNN from developing robust representations of subtle spectral differences.

V. CONCLUSION AND FUTURE WORK

This study explored the automation of chart generation for the music game Cytoid using deep learning methods. Initial attempts with a multitask Bi-LSTM architecture revealed difficulties in jointly predicting presence, type, and position. Subsequently, Transformer models failed to capture sharp discriminative features, resulting in poor performance. CNN-based models, however, showed promise for note presence detection by leveraging localized spectral-temporal patterns, achieving F1 scores around 64%.

Despite this progress, several challenges persist. Note type prediction continues to suffer from class imbalance, causing the model to favor the most frequent classes. Positional prediction remains imprecise, likely due to a lack of tempo-aligned features or more sophisticated feature engineering methods.

Future work will focus on refining model architectures and training approaches to further enhance predictive capabilities. Advanced class-balancing techniques, such as integrating focal loss with data augmentation or oversampling, will be explored to improve performance on underrepresented note types. Incorporating tempo or beat-structured features and experimenting with hybrid CNN-RNN or attention-based models may help align predicted notes more closely with musical events. Additionally, expanding the dataset to include a broader variety of songs and difficulty levels could improve the models' generalization capabilities. By addressing these issues, we aim

to develop automated chart generation methods that closely approach the quality and nuance of human-created charts.

REFERENCES

[1] J. Arevalo, T. Solorio, M. Montes-y-Gómez, and F. A. González, "Gated multimodal units for information fusion," arXiv:1702.01992, 2017.

[2] K. Choi, R. Fazekas, and M. Sandler, "Convolutional Recurrent Neural Networks for Music Classification," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing*, 2017, pp. 2392-2396.

[3] K. Chen and W. Zhang, "Deep learning approaches for music signal processing," *IEEE Trans. Multimedia*, vol. 20, no. 1, pp. 190-202, Jan. 2018.

[4] D. Donahue, K. Simonyan, A. Zisserman, and G. Vondrick, "Dance Dance Convolution: Learning Generative Models for Dance," in *Proc. IEEE Int. Conf. on Computer Vision*, 2017, pp. 1-9.

[5] D. Eck and J. Schmidhuber, "Finding temporal structure in music: Blues improvisation with LSTM recurrent networks," in *Proc. 12th IEEE Workshop Neural Netw. Signal Process.*, 2002, pp. 747-756.

[6] A. Graves, "Generating Sequences With Recurrent Neural Networks," arXiv:1308.0850 [cs, stat], Aug. 2013.

[7] G. Hadjeres, F. Pachet, and F. Nielsen, "DeepBach: A Steerable Model for Bach Chorales Generation," in *Proc. 34th Int. Conf. Learn. Represent.*, 2017, pp. 1362-1371.

[8] C.-Z. A. Huang et al., "Music Transformer: Generating Music with Long-Term Structure," in *Int. Conf. Learn. Represent.*, 2019.

[9] T. Kim and B. Park, "Deep neural networks for music processing: A comprehensive review," *IEEE Access*, vol. 7, pp. 85679-85691, 2019.

[10] J. Lee and J. Nam, "Feature learning for music signal processing using convolutional neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2019, pp. 65-70.

[11] S. Li et al., "A survey on deep learning techniques for music generation," ACM Computing Surveys, vol. 52, no. 6, pp. 1-35, 2020.

[12] F. Liang, M. Gotham, M. Johnson, and J. Shotton, "Automatic Stylistic Composition of Bach Chorales with Deep LSTM," in *Proc. 18th Int. Soc. Music Inf. Retrieval Conf.*, 2017, pp. 449-456.

[13] T. Koizumi, M. Sato, and Y. Tanaka, "Automatic Chart Generation for Rhythm Games Using Multi-Scale Convolutional Networks," *IEEE Trans. on Games*, 2020, pp. 567-578.

[14] Y. Liu and M. Yang, "CNN-based music feature learning: A comprehensive analysis," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 5, pp. 1592-1605, 2018.

[15] S. Oore, I. Simon, S. Dieleman, D. Eck, and K. Simonyan, "This Time with Feeling: Learning Expressive Musical Performance," Neural Computing and Applications, vol. 32, no. 4, pp. 955-967, 2020.

[16] A. Roberts, J. Engel, C. Raffel, C. Hawthorne, and D. Eck, "A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 4364-4373.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," arXiv preprint arXiv:1706.03762, 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1706.03762

[18] M. Wang et al., "Deep learning for music analysis and generation: Challenges and solutions," *IEEE Signal Process. Mag.*, vol. 36, no. 1, pp. 41-50, Jan. 2019.

[19] J. Wu et al., "Music Transform: A Hybrid Transformer for Musical Sequence Generation," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2020, pp. 3782-3786.

[20] Y. Ye, S. Huang, and L. Wang, "TaikoNation: A Neural Approach to Rhythm Game Chart Generation," in *Proc. IEEE Int. Conf. on Machine Learning*, 2020, pp. 1234-1243.

[21] H. Zhang et al., "Music pattern recognition using convolutional neural networks," in Proc. Int. Conf. Machine Learning and Applications, 2019, pp. 1231-1236.

[22] Google Drive, https://drive.google.com/drive/folders/1J43x9f8u2lIzaHBolQaZveCv62XQM8Lv Accessed: Nov. 6, 2024.

[23] Cytoid Wiki, "C2 Format," https://cytoid.wiki/en/reference/chart/c2-format, Accessed: Nov. 6, 2024.

[24] Cytoid, https://cytoid.io, Accessed: Nov. 6, 2024.