Prompt-to-Song Generation using Large Language Models

LLM in the Arts

Abhigyan J Singh
singh44@buffalo.edu
Electrical Engineering
University at Buffalo
Buffalo, New York, USA

Shri Harsha Adapala
sadapala@buffalo.edu
Engineering Science Data Science
University at Buffalo
Buffalo, New York, USA

ABSTRACT

This project implements a system for generating musical compositions from textual prompts by leveraging large language models (LLMs). Given a thematic prompt specifying elements like title, artist, genre, and verse ideas, our system generates song lyrics, predicts the musical genre, and composes a chord progression to produce a complete song.

By decomposing the problem into lyric generation, genre classification, and chord progression generation stages, we achieve controllable music creation that captures the semantics of the prompt. Experimental results demonstrate the viability of our LLM-based approach to creative music production.

1 INTRODUCTION

Generating complete musical compositions from high-level textual descriptions is a challenging task that requires understanding the semantics of the prompt, generating stylistically and thematically relevant lyrics, identifying the appropriate musical genre, and composing melodic and harmonic elements to create the final song. This creative process demands a high degree of musical knowledge and creative ability that is hard to capture in traditional rule-based or machine-learning approaches.

Recent advances in large language models (LLMs) have shown remarkable capabilities in natural language understanding and generation [21]. LLMs can perform complex language tasks and generate contextually relevant outputs by training on vast amounts of textual data. In this work, we leverage the power of LLMs to generate songs conditioned on textual prompts.

Our approach decomposes the text-to-music problem into three stages:

(1) Lyric generation from the textual prompt,
(2) Genre classification of the lyrics, and
(3) Chord progression generation conditioned on the lyrics and genre.

By tackling these sub-problems using dedicated models, we gain more control over the generation process and can handle the multimodal nature of lyrics and music. The overall architecture of our system is shown in Figure 1.

2 RELATED WORKS

This work explores the intersection of artificial intelligence (AI) and music composition, focusing on three key areas: lyric generation, genre classification, and chord progression generation:

NLP, May 2024, Buffalo, NY, USA
2024. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

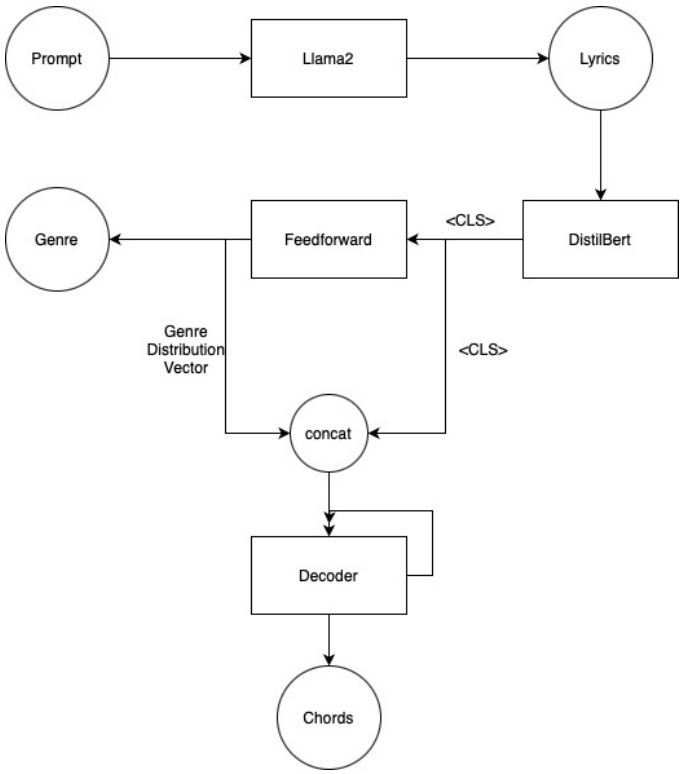

Figure 1: Design of our Prompt-to-Song system. The textual prompt is used to generate lyrics, predict genre, and compose chord progression to create the final song.

- **Transformer Based Seq-2-Seq for Chord Progression Generation** This research proposes a new method for generating chord progressions using a Seq2Seq model trained on a large music corpus. This method is claimed to be more adjustable, handle long-range dependencies, and better integrate with melodies than previous approaches.
- **An Automatic Chord Progression Generator Based On Reinforcement Learning** This research uses reinforcement learning with music theory to train an AI agent to generate chord progressions. The goal is to provide composers with a new tool to create musically interesting chord sequences.
- **An LSTM-Based Dynamic Chord Progression Generation System** This research proposes a system that uses an improved LSTM network for Interactive Music Performance, to predict future chords based on a sequence of chords played. The system

is connected to a virtual air-guitar controller that uses these predictions to change the chord options available to the performer in real time.

- **Microsoft Muzic** Muzic is a Microsoft research project that uses deep learning and AI to understand and create music. It's a suite of different models, each tackling a specific aspect of music. For understanding music, Muzic has models that can decipher music notation and even transcribe lyrics automatically. On the creative side, Muzic can generate entire songs, including lyrics and melodies, or create accompaniments or sing with your voice.
- **Chat Musician** Researchers created an open-source ChatMusician system that allows a large language model to understand and generate music. Chat Musician can compose music based on text descriptions or existing musical elements by treating music as a language. This model outperforms previous attempts in understanding and creating music, demonstrating the potential of large language models for musical tasks.

3 DATASET

We use the Chords Lyric Dataset from Kaggle [2], which contains lyrics aligned with chord labels for multiple genres. To construct input prompts, we first summarize the original lyrics using a pretrained BART model [15] to capture the key themes and ideas. We then curate these summaries into prompts containing information like song titles, artist names, genres, and verse ideas.

We clean the dataset to extract the chord progressions for each song. The final dataset contains 9 thousand curated prompts, full lyrics, genre labels, and chord progressions.

4 METHODOLOGY

Our approach decomposes the text-to-music problem into three stages:
(1) Lyric generation from the textual prompt,
(2) Genre classification of the lyrics, and
(3) Chord progression generation conditioned on the lyrics and genre.

4.1 Lyric Generation through prompting finetuned LLM

For generating lyrics from the prompts, we experiment with finetuning FlanT5 [9] and Llama [20] models. The prompts are encoded using the LLM's tokenizer and used as input to the decoder model, which generates the full lyrics token-by-token. We fine-tune the LLM for three epochs on the lyrics-prompt dataset using the LLama Factory Library [26]. An example of the finetuning dataset is shown in Figure 2.

Prompting the lyrics is done via a custom user interface as shown in Figure 3.

- **Title:** [Provide a title or theme for the song]
- **Artist or Band Name:** [Specify the name of the artist or band, real or fictional]
- **Genre:** [Specify the primary musical genre or style]
- **Sub-genres:** [List any relevant sub-genres or related styles]
- **Era or Decade:** [Indicate the era or decade that influences the song's style]

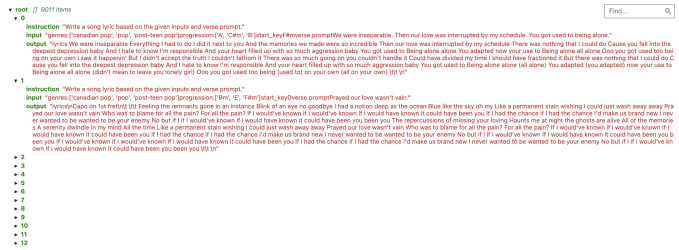

Figure 2: Fine-tuning LLM for prompt to lyrics generation

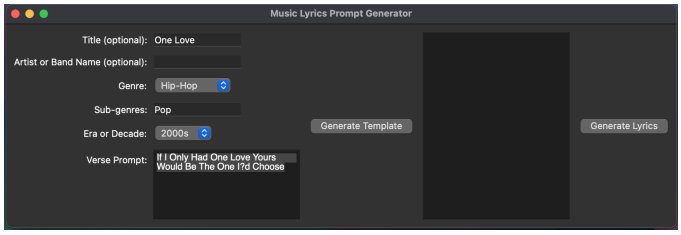

Figure 3: Prompt Generation through user preferences

- **Verse Prompt:** [Provide a detailed prompt or key points for the first verse]
- **Additional instructions:** [Add any specific requirements or guidelines for the lyrics]
- **Example Lyrics:**[Give a sample lyrics as per the chosen preferences of the user]

4.2 Genre Classification of the lyrics

To predict the genre of the generated lyrics, we compare several pretrained models, including BERT [11], DistilBERT [18], and RoBERTa [17] fine-tuned for genre classification.

Model	Loss	Accuracy
BERT	1.27	0.62
DistilBERT	1.16	0.74
veucci-lyrics-to-genre	1.38	0.49
twitter-roberta-base-sentiment-latest	1.32	0.58
emotion-english-distilroberta-base	1.36	0.52

Table 1: Comparison of different models for genre classification.

As shown in Table 1, various models achieved comparable performance on the classification task. Based on our experiments, opting for simplicity, we selected DistilBERT as the preferred model due to its lower computational complexity than other contenders.

4.3 Chord Progression

Generating chord progressions conditioned on the lyrics is challenging as it requires understanding the semantic and structural relationship between the two modalities. We explore two approaches:

Transformer-2-Sequence model

We use a two-stage encoder-decoder architecture where the encoder is a frozen DistilBERT model that generates lyric embeddings, and the decoder is an LSTM that predicts the chord progression one step at a time. We represent chords using one-hot encoding over the chord vocabulary. To help the model learn faster, we use teacher forcing [23] during training. Figure 4

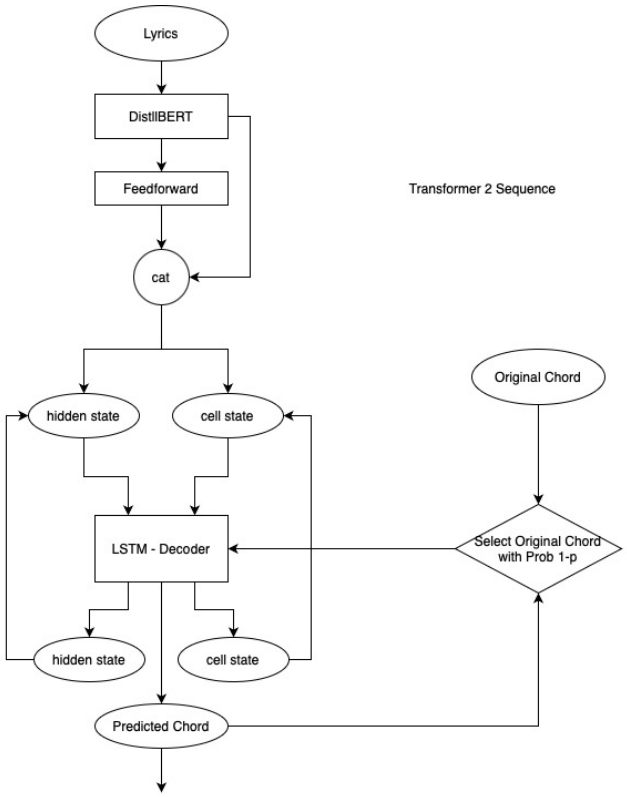

Figure 4: Transformer-to-Sequence: Model to generate chord progression to generate chord progression conditioned using Lyrics and Genre

This novel architecture for music chord prediction leverages the strengths of Long Short-Term Memory (LSTM) networks and Transformer models. The system begins with a pre-trained text-embedding model to extract meaningful representations from the input lyrics.

An LSTM decoder then processes this information, effectively capturing long-term dependencies within the lyrics that might influence chord progressions. Importantly, this Transformer encoder is frozen during training, meaning its weights are not updated. This allows the model to benefit from the Transformer's ability to consider broader musical context without altering its pre-trained knowledge. In conjunction with the LSTM decoder, the frozen Transformer enhances the prediction capabilities.

Additionally, a technique called teacher forcing is employed during training. With a certain probability, the true chord is fed back into the system as input, guiding it toward the correct mapping between lyrics and chords. The remaining probability allows the model to utilize its predictions for the next step, fostering its ability to generate novel chord sequences independently.

Overall, this hybrid architecture offers a comprehensive approach to music chord prediction by combining memory-based LSTMs with the contextual strengths of a frozen Transformer while simultaneously promoting the model's ability to learn and generate new sequences.

Song Name	Predicted Prog.	Original Prog.
Who I have	Bb, A9, B/A	G, A, D
Don't let me	Gbdim7, A/G#, Am/D	Dm, Am, Dm
Different World	F9, Dsus, Fm/C	C, D, Bm
Where did my baby	A7/G, B/C, G#m7	Em7, G, Em7

Table 2: Transformer-to-Sequence: Outputs of Chord Progression

Table 2 presents a comparison between the original chord progressions of various songs and the progressions generated by the Transformer-to-Sequence model. This table allows for visual inspection of the model's ability to capture and replicate musical patterns.

Reinforcement Learning with Human Feedback: RHLF

Reinforcement Learning from Human Feedback (RLHF) is a promising approach for training agents to perform complex tasks by incorporating human preferences and feedback [**?**]. In this work, we formulate the task of generating chord progressions conditioned on lyrics as a sequential decision-making problem and solve it using RLHF.

The key idea is to train a policy network that selects chords step-by-step based on the lyrical context and to optimize this policy using feedback from a learned reward model that encodes human preferences for chord-lyric compatibility. This setup allows us to generate chord progressions that are melodic and semantically coherent with the given lyrics.

We formulate the task as a reinforcement learning problem where the agent/policy network selects the chords conditioned on the lyrics and previous chords. Based on human feedback, the reward model is trained to predict the compatibility between the lyrics and chords. We use REINFORCE [22] to optimize the policy to maximize the expected reward.

To represent the chords as input to the reward model, we use positional encodings [21] where the position indices are added to the one-hot chord representation. This allows capturing the relative positioning of the chords. Figure 5

4.4 Chord Progression Generation using RLHF

We propose a novel approach to music chord prediction by framing it as a Reinforcement Learning (RL) problem. This formulation views chord selection as a sequence of decision-making actions.

We leverage a Markov Decision Process (MDP) to model the environment and employ vanilla Policy Gradient methods to learn the optimal policy for selecting chords.

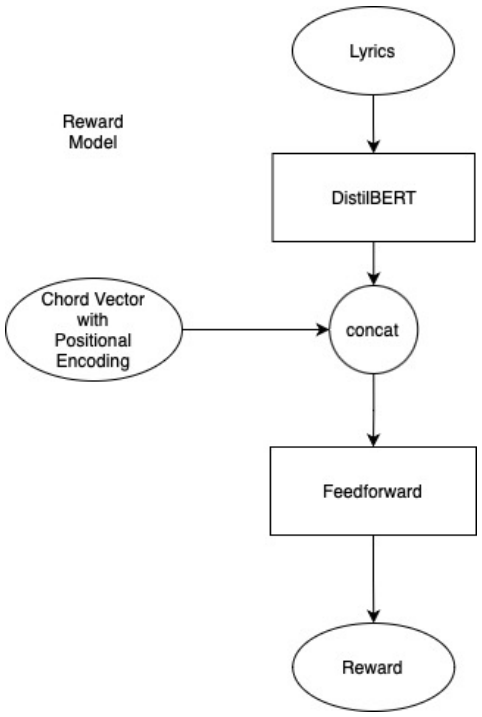

Figure 5: Reward Model: Generates Reward given the lyrics, positional encoded chord progression vector

Unlike traditional approaches with static reward functions, we introduce a neural network-based reward function inspired by the Reward Hypothesis Learning Function (RHLF) for better generalization.

This two-step process allows the model to dynamically learn and adapt the reward structure based on the encountered sequences, promoting better generalization in predicting musically appropriate chord progressions.

Constructing Reward Model

We constructed a neural network-based reward function to enhance the model's ability to generalize across diverse musical styles, drawing inspiration from techniques employed in genre classification tasks.

However, a key challenge arose when representing chords using one-hot vectors. These vectors lack inherent positional information, hindering the model's understanding of chord order within a sequence. To address this limitation, we implemented a positional encoding scheme. Here, we assigned increasing values (0 for the first chord, 0.25 for the second, and so on) to each chord in the sequence.

This encoding strategy injects positional information into the one-hot vectors, enabling the model to learn the importance of chord order in music prediction effectively.

It's important to note that during training the positional encoding network, we employed a Mean Squared Error (MSE) loss function, as this is a regression problem aiming to predict the appropriate positional value for each chord. Figure 5

RL using Policy Gradient

We leverage the previously described neural network-based reward function to train the model's policy for selecting chords. This approach utilizes a policy gradient reinforcement learning algorithm. It stands in contrast to traditional methods that minimize cross-entropy loss.

Our focus here shifts to maximizing the expected reward, aligning perfectly with the core principle of Reinforcement Learning (RL). The RL framework aims to learn a policy that leads to sequences with the highest cumulative rewards.

$$maximize \left[\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \cdot \left(\sum_{t'=t}^{T} R_\phi(s_{t'}, a_{t'}) \right) \right] \quad (1)$$

In this context, these rewards reflect musically appropriate chord progressions. Notably, the model architecture employed for the policy network mirrors the one used in the Transformer-to-Sequence section, ensuring consistency and leveraging the strengths of that architecture for effective chord selection.

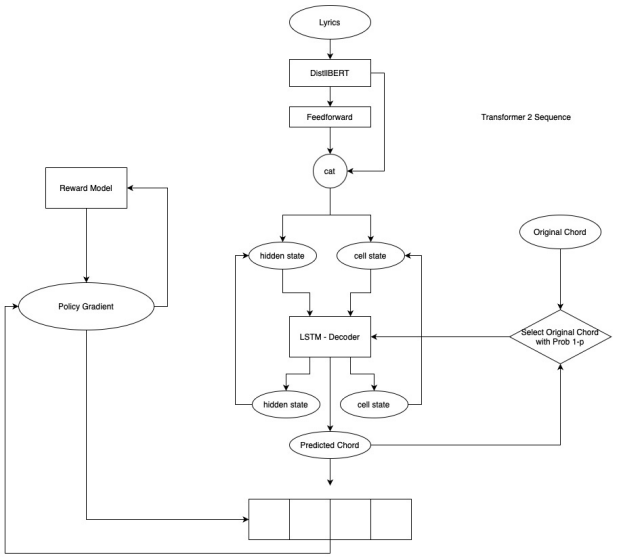

Figure 6: Policy Gradient: Training the Transformer-to-Sequence using Policy Gradient

Song Name	Predicted Prog.	Original Prog.
Ronan	Em, Am, Am7/E	Am, G, F
The Neighborhood	E11, D/E, Dsus	G, C, Am
Wild Horses	D7, E4, A7m	G7, C7, Dsus9
So, will I	F#m/B, Cm, Absus2	Gm/F, C/F, Bb/F

Table 3: RHLF: Outputs of Chord Progression

Table 3 compares the original chord progressions of various songs and the progressions generated by RHLF. This table allows for visual inspection of the model's ability to capture and replicate musical patterns.

5 AUDIO CRAFT

We have also implemented Audio Craft, a deep-learning library for audio generation. These state-of-the-art AI generative models produce high-quality audio of instruments and vocals.

5.1 AudioGen

AudioGen is an autoregressive model that generates audio samples conditioned on text prompts and/or existing audio inputs. It consists of two key components: an audio representation model and an audio language model. Figure 7

The audio representation model is a convolutional neural network that learns to encode raw audio waveforms into compact latent representations. These representations capture the salient features of the audio, such as timbre, pitch, and rhythm. The model is trained using a contrastive loss to ensure that similar audio segments are mapped to nearby points in the latent space.

The audio language model is a Transformer-based autoregressive model that generates the latent audio representations in a sequential manner. It is conditioned on text prompts and/or audio inputs, allowing for controllable audio generation. The model is trained on a large corpus of music audio and corresponding text annotations.

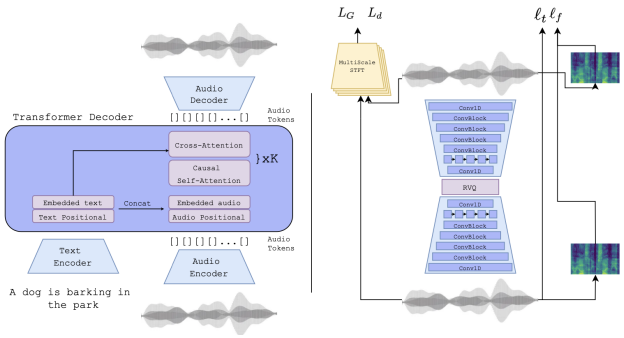

Figure 7: Left: the audio representation model. Right: the audio language model. Both text and audio embeddings are concatenated over the time dimension and fed in K causal self-attention and cross-attention blocks with the embedded text.

During inference, the text prompts and audio inputs are first encoded into a shared latent space. The audio language model then generates a sequence of latent representations based on these conditioned inputs.

Finally, the audio representation model decodes these latent representations back into the raw audio domain, producing the final generated audio samples.

5.2 EnCodec

EnCodec is an encoder-decoder model for high-fidelity audio generation and compression. It consists of an encoder network that maps raw audio into a discrete latent representation and a decoder network that reconstructs the audio from the latent codes. Figure 8

The encoder is a convolutional neural network that learns to compress the audio waveform into a compact latent space while preserving perceptually important details. It is trained using a combination of reconstruction loss and adversarial loss to ensure that the encoded representations capture the essential characteristics of the audio.

The decoder is also a convolutional network that learns to reconstruct the audio waveform from the discrete latent codes. It is trained to minimize the reconstruction error between the original and generated audio.

A key feature of EnCodec is its use of vector quantization in the latent space. The continuous latent representations are mapped to a finite set of discrete codes using a learned codebook. This quantization step allows for more efficient compression and enables the use of powerful autoregressive models in the latent space.

During training, EnCodec uses a combination of reconstruction losses, adversarial losses, and commitment loss to ensure that the encoder and decoder learn to compress and reconstruct the audio effectively while maintaining perceptual quality.

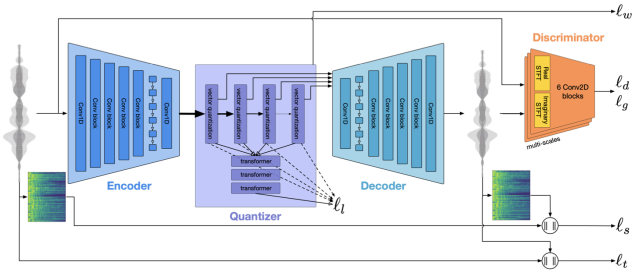

Figure 8: EnCodec: an encoder-decoder codec architecture which is trained with reconstruction (lf and lt) as well as adversarial losses (lg for the generator and ld for the discriminator). The residual vector quantization commitment loss (lw) applies only to the encoder.

6 SONG COMPOSER

For this project, we also implement Song Composer [12], an innovative LLM designed for song composition. Figure 9

Song Composer is an AI system specialized in composing, arranging, and producing complete songs. It relies heavily on Contrastive Language-Image Pre-training (CLIP) [21] to align audio and text in a shared embedding space.

The system takes as input a text prompt describing the desired genre, mood, instrumentation, and lyrics and generates several candidate audio compositions that aim to match the prompt. These candidates are ranked using a CLIP-based scoring function that measures the similarity between the prompt and the generated audio in the shared embedding space. The integration of CLIP enables Song Composer to understand and generate music that aligns with specific moods, genres, and themes, allowing for highly controllable and customizable song creation.

7 RESULTS

We evaluate our prompt-to-song generation system on a held-out test set of prompts and assess the quality of the generated lyrics,

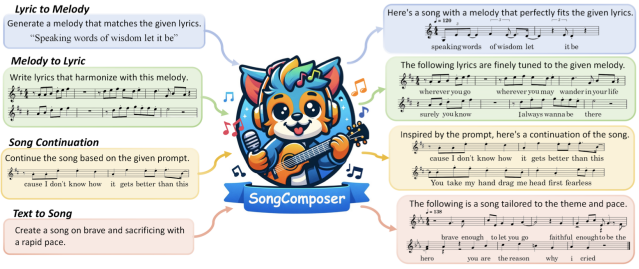

Figure 9: SongComposer is capable of composing melodies tailored to lyrics, crafting lyrics to complement melodies, extending existing songs, and generating new songs from textual prompts

genre predictions, and chord progressions using objective metrics and human evaluations.

7.1 Lyric Generation

The fine-tuned LLMs generate generally coherent lyrics, thematically relevant and stylistically consistent with the input prompts. However, the generated lyrics sometimes exhibit repetitiveness or need more narrative flow, especially for longer sequences.

7.2 Genre Classification

The DistilBERT model achieves an accuracy of 0.74 on the genre classification task, demonstrating its ability to predict musical genres based on lyrical content. The model performs well on prevalent genres like pop and rock but needs more represented or ambiguous genres. Table 2 shows some example predictions.

7.3 Chord Progression Generation

The Transformer-2-Sequence model generates chord progressions that are melodically plausible and generally align with the lyrical content. Table compares the generated progressions with the original ones, showing that the model can capture similar harmonic structures.

The RLHF approach produces chord progressions that are simpler but more harmonically coherent and better aligned with the lyrical structure. Table 3 shows some example progressions, demonstrating the model's ability to generate chord sequences that match the emotional tone of the lyrics.

7.4 Qualitative Results

Below is an example of a song generated by our system given a text prompt. The generated lyrics are thematically relevant to the prompt, the predicted genre (pop) is appropriate, and the chord progression provides a suitable harmonic structure for the lyrics.

7.4.1 *Prompt:*

I want to create a song for my natural language class, taught by Professor Rohini Srihari and Teaching assistant Sayantal Pal.

Create lyrics for such a song, include the course name into the lyrics and have a line saying that they want to grade us very highly.

7.4.2 *Lyrics:*

NLP, the class that sets us free
Prof. Srihari and Sayantal, our guiding team

Natural Language Processing, oh so fine
They want to grade us high, all the time

From words to meaning, we learn it all
In this class, we stand tall

Parsing, disambiguating, we're on a roll
NLP, you're in our soul

Natural Language Processing, oh so fine
They want to grade us high, all the time

Prof. Srihari, Sayantal Pal, we thank you
NLP, our passion true

While these results demonstrate the potential of our approach, the generated songs still need more coherence, arrangements, and emotional depth of human-composed music. Integrating the individual components into a complete and compelling song remains a challenge to be addressed in future work.

8 DISCUSSION AND ERROR ANALYSIS

Our prompt-to-song generation system showcases the potential of LLMs for creative music production. However, it also reveals several challenges and limitations that merit further analysis.

8.1 Qualitative Assessment

- **Lyric Generation:** The generated lyrics are generally thematically and stylistically consistent with the input prompts. However, they sometimes lack coherence, narrative flow, and novelty, especially for longer sequences.
- **Genre Classification:** The DistilBERT model achieves reasonable accuracy but struggles with less represented and ambiguous genres. Misclassifications often occur between similar genres.
- **Chord Progression Generation:** The Transformer-2-Sequence model generates melodically plausible chord progressions that generally align with the lyrics. However, they may lack the richness and variation of human-composed progressions. The RLHF approach produces simpler, more harmonically coherent progressions guided by human feedback.
- **Overall Song Quality:** Integrating the individual components (lyrics, chords, genre) into a cohesive and satisfying song remains a significant challenge. The generated songs may lack human-composed songs structure, dynamics, and emotional arc.

8.2 Potential Improvements

Lyric Generation:

- Incorporate techniques like diversity regularization or reinforcement learning to improve coherence and novelty.
- Explore hierarchical and global context modeling to capture long-term dependencies.

Genre Classification:

- Use data augmentation to balance genre distribution.
- Incorporate audio features for more accurate classification.
- Employ hierarchical classification to model genre relationships.

Chord Progression Generation:
- Develop musically informed chord representations (e.g., embeddings, tonal vectors).
- Integrate music theory rules and constraints into the generative process.
- Refine the RLHF approach with more sophisticated reward functions.

Overall Song Quality:
- Implement higher-level control and conditioning mechanisms for multi-scale generation.
- Explore techniques like hierarchical modeling, multi-task learning, and adversarial training.
- Conduct user studies for subjective evaluation and feedback.

8.3 Future Directions

- Incorporate additional musical elements (e.g., rhythm, arrangement).
- Develop interactive and collaborative user interfaces for human-AI co-creation.

This project demonstrates the potential of LLMs for music creation and highlights exciting research directions at the intersection of AI, creativity, and music. Addressing the identified challenges and limitations could lead to more expressive, diverse, and emotionally resonant AI-generated music.

9 CONCLUSION AND FUTURE WORK

This project demonstrates the feasibility of leveraging large language models for prompt-to-song generation. The developed system achieves a degree of controllable song creation by breaking down the process into interpretable stages of lyric generation, genre classification, and chord generation. Fine-tuning LLMs yields promising results for lyric generation, while the Transformer and RL approaches to chord generation have complementary strengths.

However, significant challenges remain in generating melodic content, integrating lyrics and chords, and imbuing the generated songs with long-term structure and coherence. Future work can explore pre-training LLMs on large-scale music corpora, cross-modal representations for lyrics and music, few-shot learning for stylistic adaptation, and reinforcement learning with music theory rewards.

Overall, this project pushes the boundaries of AI-assisted music creation and highlights the potential of language models as a powerful tool for creative musical expression. Continuing research in this direction can lead to more intuitive, interactive, and expressive music generation systems that enhance human creativity.

10 CONTRIBUTIONS

(1) **Abhigyan J Singh [50%]:**
- *Documentation:* Prepared project documentation including reports.
- *Coding:* Prompt to Lyrics Generation and Music Generation from Lyrics.

(2) **Shri Harsha Adapala [50%]:**
- *Documentation:* Prepared project documentation including reports.
- *Coding:* Genre Classification and Generating Chord Progression.

ACKNOWLEDGMENT

We want to thank Maria Pushparaj from team Pheonix for helping us understand and curate the dataset for chords and its progression.

REFERENCES

[1] [n. d.]. Chat Musician. https://huggingface.co/m-a-p/ChatMusician. Accessed: 2024-05-08.
[2] [n. d.]. Chords Lyric Dataset. https://www.kaggle.com/datasets/eitanbentora/chords-and-lyrics-dataset. Accessed: 2023-06-09.
[3] [n. d.]. Kaggle Dataset: Lyrics and Chords Dataset. https://www.kaggle.com/datasets/eitanbentora/chords-and-lyrics-dataset. Accessed: 2024-05-08.
[4] [n. d.]. Microsoft Muzic. https://github.com/microsoft/muzic. Accessed: 2024-05-08.
[5] 2018. An Automatic Chord Progression Generator Based On Reinforcement Learning. In *IEEE Explore*. https://ieeexplore.ieee.org/document/8554901
[6] 2020. An LSTM-Based Dynamic Chord Progression Generation System for Interactive Music Performance. In *IEEE Explore*. https://ieeexplore.ieee.org/document/9053992
[7] 2022. Training language models to follow instructions with human feedback. https://arxiv.org/abs/2203.02155.
[8] 2023. Transformer Based Seq-2-Seq for Chord Progression Generation. *Mathematics* 11, 5 (2023), 1111. https://doi.org/10.3390/math11051111
[9] Hyung Won Chung, Teven Le Scao, and Alexander M Rush. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416* (2022).
[10] Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. 2024. Simple and Controllable Music Generation. arXiv:2306.05284 [cs.SD]
[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2019).
[12] Shuangrui Ding, Zihan Liu, Xiaoyi Dong, Pan Zhang, Rui Qian, Conghui He, Dahua Lin, and Jiaqi Wang. 2024. SongComposer: A Large Language Model for Lyric and Melody Composition in Song Generation. arXiv:2402.17645 [cs.SD]
[13] Christos Garoufis, Athanasia Zlatintsi, and Petros Maragos. 2020. An LSTM-Based Dynamic Chord Progression Generation System for Interactive Music Performance. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 4502–4506. https://doi.org/10.1109/ICASSP40776.2020.9053992
[14] Felix Kreuk, Gabriel Synnaeve, Adam Polyak, Uriel Singer, Alexandre Défossez, Jade Copet, Devi Parikh, Yaniv Taigman, and Yossi Adi. 2023. AudioGen: Textually Guided Audio Generation. arXiv:2209.15352 [cs.SD]
[15] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 7871–7880.
[16] Shuyu Li and Yunsick Sung. 2023. Transformer-Based Seq2Seq Model for Chord Progression Generation. *Mathematics* 11, 5 (2023). https://doi.org/10.3390/math11051111
[17] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).
[18] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* (2019).
[19] Shipra Shukla and Haider Banka. 2018. An Automatic Chord Progression Generator Based On Reinforcement Learning. In *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. 55–59. https://doi.org/10.1109/ICACCI.2018.8554901
[20] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timoth'ee Lacroix, Baptiste Rozi'ere, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).
[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).
[22] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8, 3 (1992), 229–256.
[23] Ronald J Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural computation* 1, 2 (1989), 270–280.

[24] Dingyao Yu, Kaitao Song, Peiling Lu, Tianyu He, Xu Tan, Wei Ye, Shikun Zhang, and Jiang Bian. 2023. MusicAgent: An AI Agent for Music Understanding and Generation with Large Language Models. *arXiv preprint arXiv:2310.11954* (2023).

[25] Ruibin Yuan, Hanfeng Lin, Yi Wang, Zeyue Tian, Shangda Wu, Tianhao Shen, Ge Zhang, Yuhang Wu, Cong Liu, Ziya Zhou, Ziyang Ma, Liumeng Xue, Ziyu Wang, Qin Liu, Tianyu Zheng, Yizhi Li, Yinghao Ma, Yiming Liang, Xiaowei Chi, Ruibo Liu, Zili Wang, Pengfei Li, Jingcheng Wu, Chenghua Lin, Qifeng Liu, Tao Jiang, Wenhao Huang, Wenhu Chen, Emmanouil Benetos, Jie Fu, Gus Xia, Roger Dannenberg, Wei Xue, Shiyin Kang, and Yike Guo. 2024. ChatMusician: Understanding and Generating Music Intrinsically with LLM. arXiv:2402.16153 [cs.SD]

[26] Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, and Yongqiang Ma. 2024. LlamaFactory: Unified Efficient Fine-Tuning of 100+ Language Models. *arXiv preprint arXiv:2403.13372* (2024). http://arxiv.org/abs/2403.13372