Emotion-Controlled Symbolic Music Generation via Prompt Tuning with Structured Constraints

Anonymous ACL submission

Abstract

Controlling emotional expressions while creating symbolic music is still considered an huge problem. Most of the current methods have generated music of limited lengths or do not provide precise control over the generated piece's emotion. In the following text, we propose a hybrid approach that utilizes prompt tuning with a structured music generation process. To do this, we have trained prompt embeddings using a pre-trained SkyTNT model, keeping all base parameters frozen. We then use these learned prompt embeddings to generate music while using various constraints, increasing the music length from approximately 32 seconds to 148 seconds per piece. Our experiments, conducted using three different music pieces (sad, happy, and calm), have shown significant differentiation between each of them. Happy music had a note density of 12.0 notes per second, sad music had a note density of 5.21 notes per second, and calm music had a note density of 7.81 notes per second. We propose this work as a parameter-efficient prompt tuning application, which is currently gaining popularity in the natural language processing community, and discuss its benefits, as well as its limitations.

1 Introduction

In recent years, transformer models have become much better at producing symbolic music. They have become able to produce longer and more coherent music compared to previous RNN models. However, controlling what is produced by the models, especially the emotional qualities, is extremely difficult. When it comes to text generation, a prompt can be given to the model to generate text in a certain style or about a certain topic. However, in the case of music, the emotional qualities are based on the combined effect of tempo, harmony, melody, and rhythm.

The most common ways to perform controlled music generation are to train a new model from scratch with conditioning or to fine-tune an entire pre-trained model. Both are costly, and the latter risks catastrophic forgetting of the musical structure that the model was pre-trained on. Additionally, most systems are limited to producing short music, less than 30 seconds long.

We use a different approach. Our method utilizes prompt tuning (Lester et al., 2021), a parameter-efficient technique developed within NLP, to learn emotion-specific representations on labeled MIDI data. Specifically, we fine-tune 12 prompt tokens for each emotion type (sad, happy, calm) on top of SkyTNT (Radford et al., 2019), a GPT-2 variant pre-trained on MIDI data. This only required 36,864 trainable parameters, which is only a small part of the millions of parameters in SkyTNT. The learned prompt is then used to inform a structured generation layer, which used constraints on scales, chord progressions, and rhythmic patterns to generate multi-track music pieces that can last anywhere from 30 seconds to 2.5 minutes.

This work is at the boundary of NLP methodology and music generation. Prompt tuning is originally a text-based language model technique. We believe the applicabillity of prompt tuning to the music world is an extension worth researching and using. Hence, we hope it is of interest to the NLP community, despite the output domain of music rather than text.

In our evaluation, there is clear differentiation in the emotions in the generated output: happy music has a note density of 12.0 notes per second compared to 5.21 for sad and 7.81 for calm, as well as note durations and scales that differ in ways that are consistent with what research on how people perceive emotions in music would suggest (Juslin and Sloboda, 2010).

2 Related Work

2.1 Symbolic Music Generation

Many transformer-based architectures have been successfully used in symbolic music generation. Music Transformer (Huang et al., 2018) was the first to use relative position representation into music sequences. SkyTNT, which we have used, is a GPT-2-based model (Radford et al., 2019), pre-trained on a large dataset of MIDI files and available online as a community-supported model through Hugging Face. It was also available in pre-trained form for easy use.

2.2 Controllable Generation

Previous work in controlled music generation has explored various methods to control the output, such as selecting a genre and assigning labels based on emotions given by the music. These methods usually required a change in the model architecture or tuning of the entire parameter space, which may possibly hurt the model's music generation abilities.

2.3 Prompt Tuning

The technique of prompt tuning was first proposed by Lester et al. (2021) as a method to fine-tune frozen language models by training only a few learnable input vectors. This technique is highly effective on a variety of text processing tasks while using many fewer parameters than fine-tuning. We extend this approach to symbolic music by learning a few tokens specific to each emotion instead of fine-tuning the entire model.

2.4 Music and Emotion

Research on how people emotionally respond to music has shown that there are significant correlations between musical characteristics and the emotions felt by the people who listen to the music (Juslin and Sloboda, 2010). For instance, the tempo of the music has been shown to have one of the highest correlations with the emotions felt by the listeners. Faster tempos are associated with positive or high-energy emotions, while slower tempos are associated with sad or relaxing emotions. Major keys are associated with happiness, and minor keys are associated with sad emotions. Our generation has followed this research.

3 Method

3.1 Base Model: SkyTNT

Our base model is SkyTNT, a GPT-2-style transformer with 12 layers, 16 attention heads, and a hidden dimension of 1024. It was pre-trained on MIDI data and uses a vocabulary of 3,239 tokens representing musical events (note onsets, offsets, time shifts, etc.). The model is available via Hugging Face (`skytnt/midi-model`). We freeze all of its parameters during our training.

3.2 Prompt Tuning for Emotion Learning

For each emotion $e \in \{\text{sad, happy, calm}\}$, we introduce a learnable prompt matrix $\mathbf{P}_e \in \mathbb{R}^{n \times d}$, where $n = 12$ tokens and $d = 1024$ is the hidden dimension. During training, the prompt is prepended to the input token sequence:

$$\mathbf{h}_0 = [\mathbf{P}_e; \mathbf{E}(\mathbf{x})] \tag{1}$$

where $\mathbf{E}(\mathbf{x})$ is the token embedding of input sequence $\mathbf{x}$.

We minimize cross-entropy loss over emotion-labeled MIDI sequences while keeping all base model parameters θ frozen:

$$\mathcal{L} = -\sum_{i=1}^{T} \log p(x_i \mid x_{<i}, \mathbf{P}_e; \theta) \tag{2}$$

The total number of trainable parameters is $3 \times 12 \times 1024 = 36{,}864$, compared to the millions of parameters in the full model.

3.3 Prompt-Guided Structured Generation

Then, after the prompts are trained, we use the statistics they have learned on to inform a process of structured generation. The idea is to combine the prompts' learnings with the explicit constraints of music.

We define base parameters for each emotion based on what is known about the ways in which listeners map musical features onto emotions:

- **Scales:** Natural minor for sad, major for happy, pentatonic for calm.
- **Chord progressions:** i-iv-VI-III (sad), I-IV-V-I (happy), I-ii-iii-I (calm).
- **Base tempo:** 65 BPM (sad), 120 BPM (happy), 85 BPM (calm).
- **Rhythm:** Longer note values for sad, shorter for happy, mixed for calm.

The learned prompt embeddings influence generation through tempo adjustment. We compute the mean activation μ_p of the prompt embedding and scale the base tempo:

$$\text{tempo}_{\text{final}} = \text{tempo}_{\text{base}} \times (1 + \mu_p \times \alpha) \quad (3)$$

where $\alpha = 0.3$. The prompt variance also modulates rhythmic complexity: higher variance produces more varied rhythm patterns. The structured generator then produces 32 and 64 bar pieces with melody, chord, and bass tracks. Because tempo varies by emotion, piece durations range from roughly 32 seconds with happy at 120 BPM to roughly 148 seconds of sad at 64 BPM.

3.4 Multi-Track Generation

The generator has three tracks: melody, which uses step motion 70% of the time, which is motion to an adjacent scale degree, 30% of the time it has leaps, and sometimes it has octave transpositions; chords change every two measures, three-note clusters in an octave below the melody; bass has alternating chord roots and fifths, two octaves below the melody.

4 Experiments

4.1 Dataset

We use 58 jazz MIDI files that we manually labeled by emotion: 12 sad (e.g., "Georgia," "Days of Wine and Roses"), 14 happy (e.g., "Caravan," "Take the A Train"), and 32 calm (e.g., "Autumn Leaves," "Stella by Starlight"). Labels were asigned based on the established emotional associations of each piece in the jazz literature. The dataset is admittedly small, which we discuss in the limitations.

4.2 Training Details

We train the emotion prompts for 60 epochs with AdamW (learning rate 0.0008, weight decay 0.01) and gradient clipping at 0.5. Input sequences are truncated to 256 tokens. On a single GPU, training takes several hours given the small dataset size.

4.3 Generation

For each of the three emotions, we create 64-bar pieces. Although we also created 32-bar versions during development, we only report results for the 64-bar versions, as one of our objectives is extended generation. Because of the different base tempos for each of the emotions, the resulting lengths are also different: for happy, it is about 64 seconds, for calm, it is about 90 seconds, and for sad, it is about 148 seconds. The learned prompt embedding is then used to adjust the base parameters before the structured generator is applied.

Metric	Sad	Happy	Calm
Note Density (notes per second)	5.21	12.00	7.81
Avg Duration (seconds)	0.81	0.40	0.64
Pitch Range (semitones)	46	46	45
Unique Pitches	26	24	21
Polyphony	4.38	4.93	5.00
Pitch Entropy	4.53	4.40	4.19
Avg Velocity	63.5	64.1	64.4
Velocity Standard Deviation	11.6	11.7	11.6

Table 1: Evaluation metrics for the 64-bar generated pieces for each emotion.

4.4 Evaluation Metrics

We try to assess the generated pieces with respect to standard symbolic music metrics, as suggested by recent surveys on evaluating music generation (Ji et al., 2023). We started out trying to use the mgeval toolkit (Dong et al., 2020), but had some compatibility problems with its Python 2 codebase, so we ended up calculating all metrics with the pretty_midi library (Raffel and Ellis, 2014) instead.

The metrics we use are: note density (notes per second), pitch range (semitones between highest and lowest pitch), polyphony (average number of notes played at the same time), pitch entropy (a measure of how spread out the pitch decisions are), average note duration, and velocity statistics (mean and standard deviation).

5 Results

Table 1 shows metrics for each emotion from the 64-bar generated pieces.

5.1 Emotional Differentiation

The first thing to notice is the note density. For happy music, it is 12.0 notes per second, sad music has 5.21 notes per second, while calm music has a note density of 7.81 notes per second. This is what we would expect given their difference in terms of tempo and rhythm, as a faster tempo and smaller note values mean more notes are packed into a given second. However, note duration shows a very different result, as sad music has an average note duration of 0.81 seconds, while happy music has an average note duration of 0.40 seconds, with calm music sitting in between at 0.64 seconds.

Pitch range is very similar between the three emotions, at 45-46 semitones, which is what we would expect given that all three emotions sit in a similar range. However, note that the number of unique pitches does differ between each emotion, as sad music uses 26 pitches, while happy music uses 24 pitches, with calm music sitting at 21 pitches. Sad and happy music used seven-note scales while calm music used a five-note pentatonic scale. Pitch entropy follows the same pattern.

Polyphony is around 4.4 to about 5.0 for all three emotions, supporting that the chord voicings are creating a similar level of chord level for all three emotions. The differences between the emotions are in timing and melody, but not in the number of notes being played at the same time.

5.2 Consistency Across Lengths

We have also created 32-bar, which is the medium-sized output, versions of all pieces during development. The metrics for the 32-bar and 64-bar versions were similar, with standard deviation less than 0.05 for all metrics. This says that the generation does not get impacted by the issue of worse quality when the output gets longer, which is a problem that is seen in single-token generation. The 64-bar versions of the pieces are presented in the table from earlier.

6 Discussion

The hybrid approach is an effective approach in the sense that it creates musically coherent music with different characteristics per emotion. However, we would like to see what exactly differences are between the outputs.

The majority of the emotional differences in the output is a result of the generation parameters, for example, the scale that is used, the tempo it is set to, and the rhythms that are included. These are set by hand according to what we know about how humans perceive emotions in music; these parameters are not learned. The P-tuning aspect does play a role in this as well, as it adjusts tempo according to what it has learned about prompt statistics. In practice, however, this is a small change, usually in the range of 0-1 BPM. With a larger training set, the prompts could have a greater effect. However, with a set of 58 examples, there is only so much signal to be learned.

Of course, the system has a few things that benefit it. First, the parameter efficiency is real, where we are only using 36K parameters compared to full fine-tuning, which uses millions. Second, the use of a structured layer means that we can generate music that is a few minutes long without it completely falling apart, which is hard to create when one is only trying to generate the next word. Finally, it's all interpretable, meaning a researcher and musician can look at the parameters and see what scale and tempo are being used, and change it based on what they need to change it to.

We view this as a decent starting point. The prompt tuning adds a learned component that is not just based on the rules, while the use of a structured layer provides the musical unity that neural generation has difficulty with on a dataset of this size. However, it remains a question whether this type of hybrid approach will be useful as datasets grow in size and generation becomes easier.

7 Conclusion

We introduced a hybrid method for emotion-controlled symbolic music generation with prompt tuning and a generation based off of a rule that was set. Emotion prompts are learned from a frozen SkyTNT model and used to control parameters in a rule-based music generation component. The model generates music pieces with 30 seconds to 2.5 minutes of length on multiple music tracks in MIDI format. Happy music has 2.3 times the density of sad music, while other metrics such as note lengths, pitch, and scale vary consistently amongst emotions.

The model can create music pieces of any length without a significant drop in quality. Future work includes testing the model on a larger set of emotion-labeled music, exploring the relationship between learned prompts and music generation, and most importantly, testing the model on human listeners to ensure that the metric differences correspond to actual differences in music perception.

Limitations

We used only 58 labeled MIDI files, which are from the jazz genre. This is a small data set and may affect the prompt tuning performance. If there is a more diverse data set, prompt tuning may perform better and may result in a better output.

Furthermore, we use only three basic emotions: sad, happy, and calm. Emotions are more complex and diverse, but we used the three most basic ones. Thus, using more diverse emotions could result in

different outputs.

The influence of the P-tuning component is also small. The tempo can only change by 0-1 BPM. The majority of the influence is from the hand-designed structured parameters. It is worthy to restate this one more time to stress the importance of this.

On top of that, we did not use any human listening study. Our evaluation is objective and thus, we use only the objective metrics from the MIDI output. Note density and note duration relate to tempo-related emotions, but we did not test whether humans can really hear the emotions. This is the most important thing we did not do, and it is the most important thing we missed.

Finally, the generation method that we used is less flexible but more coherent. A fully end-to-end learning method may discover more ways of expressing emotions than our predefined rules can.

References

Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Ian Simon, Curtis Hawthorne, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Douglas Eck. Music Transformer: Generating Music with Long-Term Structure. *arXiv preprint arXiv:1809.04281*, 2018.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3045-3059, 2021.

Patrik N. Juslin and John A. Sloboda. *Handbook of Music and Emotion: Theory, Research, Applications*. Oxford University Press, 2010.

Shulei Ji, Jing Luo, and Xinyu Yang. A comprehensive survey on deep music generation: Multi-level representations, algorithms, evaluations, and future directions. *arXiv preprint arXiv:2506.05104*, 2023.

Hao-Wen Dong, Ke Chen, Julian McAuley, and Taylor Berg-Kirkpatrick. MusPy: A toolkit for symbolic music generation. In *Proceedings of the 21st International Society for Music Information Retrieval Conference (ISMIR)*, pages 142-149, 2020.

Colin Raffel and Daniel P. W. Ellis. Intuitive analysis, creation and manipulation of MIDI data with pretty_midi. In *Proceedings of the 15th International Society for Music Information Retrieval Conference (ISMIR)*, Late Breaking and Demo Papers, 2014.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.