Proceedings of the Sixteenth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE-20)

Computer-Generated Music for Tabletop Role-Playing Games

Lucas N. Ferreira,[1] **Levi H. S. Lelis,**[2] **Jim Whitehead,**[1]
[1]Department of Computational Media, University of California, Santa Cruz, USA
[2]Department of Computing Science, Alberta Machine Intelligence Institute (Amii), University of Alberta, Canada
lferreira@ucsc.edu, levi.lelis@ualberta.ca, ejw@soe.ucsc.edu

Abstract

In this paper we present Bardo Composer, a system to generate background music for tabletop role-playing games. Bardo Composer uses a speech recognition system to translate player speech into text, which is classified according to a model of emotion. Bardo Composer then uses Stochastic Bi-Objective Beam Search, a variant of Stochastic Beam Search that we introduce in this paper, with a neural model to generate musical pieces conveying the desired emotion. We performed a user study with 116 participants to evaluate whether people are able to correctly identify the emotion conveyed in the pieces generated by the system. In our study we used pieces generated for Call of the Wild, a Dungeons and Dragons campaign available on YouTube. Our results show that human subjects could correctly identify the emotion of the generated music pieces as accurately as they were able to identify the emotion of pieces written by humans.

Introduction

In this paper we introduce Bardo Composer, or Composer for short, a system for generating musical pieces that match the emotion of stories told in tabletop role-playing games (TRPGs). For example, if the players are fighting a dragon, Composer should generate a piece matching such an epic moment of the story. TRPG players often manually choose songs to play as background music to enhance their experience (Bergström and Björk 2014). Our goal is to develop an intelligent system that augments the players' experience with soundtracks that match the story being told in the game. Importantly, the system should allow players to concentrate on the role-playing part of the game, and not on the disruptive task of selecting the next music piece to be played. The object of our research is Dungeons and Dragons (D&D), a TRPG where players interpret characters of a story conducted by a special player called the dungeon master.

Padovani, Ferreira, and Lelis (2017; 2019) introduced Bardo, a system that automatically selects the background music of a D&D session based on the story being told by the players. This paper builds upon their system. Bardo uses a speech recognition system to transcribe voice into text, which is then classified into an emotion. Bardo then selects a song of the classified emotion from a library of labeled songs. In this work we extend Bardo to include a neural model for generating musical pieces conveying the emotions detected in the game story, instead of selecting a song from a labeled library—thus the name Bardo Composer. We expect that by generating pieces we can capture the exact emotional tone of the story, while methods that select from a set of pre-composed pieces have a more limited "emotional palette".

Language models (LMs) are able to generate coherent music pieces (Ferreira and Whitehead 2019). However, it is still challenging to generate music with a given emotion. For that we introduce Stochastic Bi-Objective Beam Search (`SBBS`), a variant of Stochastic Beam Search (Poole and Mackworth 2010) to guide the generative process while maximizing the probability given by a LM jointly with the probability of pieces matching an emotion. The emotion in the story is detected by a BERT model (Devlin et al. 2018) and is given as input to `SBBS`, which uses a GPT-2 model (Radford et al. 2019) to classify the emotion of the generated pieces.

We evaluated Composer on the Call of the Wild (CotW) dataset (Padovani, Ferreira, and Lelis 2017), which is a campaign of D&D available on YouTube. Since our primary goal is to generate music pieces conveying the current emotion of the game's story, we used Composer to generate pieces of parts of CotW that featured a transition in the story's emotion. Then, in a user study with 116 participants, we evaluated whether people correctly perceive the intended emotions in pieces generated by Composer. We also measured if the participants were able to distinguish the emotion of human-composed pieces by using Bardo's original system as a baseline. Our results show that the participants were able to identify the emotions in generated pieces as accurately as they were able to identify emotions in human-composed pieces. This is an important result towards the goal of a fully-automated music composition system for TRPGs.

Related Work

Our work is mostly related to machine learning models that generate music with a given emotion. For example, Monteith, Martinez, and Ventura (2010) trained Hidden Markov models to generate music from a corpus labeled according to a categorical model of emotion. These models are trained for each emotion to generate melodies and underlying harmonies. Ferreira and Whitehead (2019) used a genetic algorithm to fine-tune a pre-trained LSTM, controlling the LSTM to generate either positive or negative pieces. Our work differs from Monteith, Martinez, and Ventura's because we train a single LM that is controlled to generate music with different emotions. It is also different from (Ferreira and Whitehead 2019) once we control the LM at sampling time and not at training time.

Our work is also related to rule-based systems that map musical features to a given emotion (Williams et al. 2015b). For example, Williams et al. (2015a) generate soundtracks for video games using a rule-based system to transform pre-generated melodies, matching the emotion of annotated game scenes. Davis and Mohammad (2014) follow a similar approach in TransPose, a system that generates piano melodies for novels. Our work differs from these rule-based systems because we learn mappings from musical features to emotion directly from data.

Our work is also related to neural models that generate text with a given characteristic. For example, CTRL (Keskar et al. 2019) is a Transformer LM trained to generate text conditioned on special tokens that inform the LM about the characteristics of the text to be generated (e.g., style). Our work differs from CTRL because we control the LM with a search procedure and not with an extra input to the LM. Conditioning the LM requires a large amount of labeled data, which is expensive in our domain.

The Plug and Play LM (Dathathri et al. 2019) combines a pre-trained LM with a small attribute classifier to guide text generation. Although both Composer and the Plug and Play LM control the generation procedure at sampling time, we use search as a means of generation control while Plug and Play LM uses a classifier to alter the structure of the model.

Vijayakumar et al. (2018) and Kool, Hoof, and Welling (2019) proposed variations of Beam search to solve the problem of generating repetitive sentences. Our work differs from both these works because our variation of Beam search optimizes for two independent objectives.

Background

Symbolic Music Composition Symbolic music is typically generated by sampling from a LM that computes the likelihood of the next musical symbols (e.g., note) in a piece. Typically, the LM is defined as a neural network and the symbols are extracted from MIDI or piano roll representations of music (Briot, Hadjeres, and Pachet 2017). Let $x = [x_0, \cdots, x_{t-2}, x_{t-1}]$ be the first t symbols of a piece and $P(x_t|x_0, \cdots, x_{t-2}, x_{t-1})$ be the probability of the next symbol being x_t, according to a trained LM. One can sample the next symbol of the sequence according to the probability distribution P (Briot, Hadjeres, and Pachet 2017). We denote the trained language model as L and $L(x)$ is a function that returns the next symbol given a sequence x. To generate a piece with L, one provides as input a sequence of symbols $x = [x_0, x_1, \cdots, x_t]$ to bias the generation process. This input sequence is fed into L which computes $L(x) = x_{t+1}$.

Algorithm 1 Bardo Composer

```
Require: Speech recognition system $S$, Text emotion classifier $E_s$, Music emotion classifier $E_m$, LM $L$, speech signal $v$, previously composed symbols $x$, beam size $b$, number of symbols $k$
Ensure: Music piece $x$
1: $s, l \leftarrow S(v)$
2: $v, a \leftarrow E_s(s)$
3: $y \leftarrow \texttt{SBBS}(L, E_m, x, v, a, b, k, l)$ # see Algorithm 2
4: return $x \cup y$
```

Next, x_{t+1} is concatenated with x and the process repeats until a special end-of-piece symbol is found or a given number of symbols are generated.

Bardo Padovani, Ferreira, and Lelis (2017; 2019) presented Bardo, a system to select background music for tabletop games. Bardo classifies sentences produced by a speech recognition system into one of the four story emotions: Happy, Calm, Agitated, and Suspenseful. Bardo then selects a song from a library of songs corresponding to the classified emotion. The selected song is then played as background music at the game table. In this paper we use Padovani et al.'s dataset to train an emotion classifier for the story being told at a game session. Their dataset includes 9 episodes of CotW, which contains 5,892 sentences and 45,247 words, resulting in 4 hours, 39 minutes, and 24 seconds of gameplay. There are 2,005 Agitated, 2,493 Suspenseful, 38 Happy, and 1,356 Calm sentences in the dataset.

Valence-Arousal Model of Emotion We use a two-dimensional emotion model that generalizes the emotion model used in Bardo. We consider the dimensions of valence and arousal, denoted by a pair (v, a), where $v \in [0, 1]$ and $a \in [0, 1]$ (Russell 1980). Valence measures sentiment and thus $v = 0$ means a negative input and $v = 1$ means a positive input. Arousal measures the energy of the input and thus $a = 0$ means that the input has low energy whereas $a = 1$ means that the input has high energy. We use this model for classifying both the emotion of the player's speeches and the emotion of the generated music.

Bardo Composer: System Description

A general overview of Composer is shown in Algorithm 1. It receives as input a speech recognition system S, an emotion classifier for text E_s, an emotion classifier for music E_m, a LM for symbolic music generation L, a speech signal v with the last sentences spoken by the players, and a sequence x of musical symbols composed in previous calls to Composer. The algorithm also receives parameters b and k, which are used in the search procedure described in Algorithm 2. Composer returns a symbolic piece that tries to match the emotion in the players' speeches.

Composer starts by converting the speech signal v into text s with S (line 1). In addition to text, S returns the duration of the signal v in seconds, this is stored in l. Then, Composer classifies the emotion of s in terms of valence v and arousal a and it invokes our Stochastic Bi-Objective

Beam Search (SBBS) to generate a sequence of symbols y that matches the desired length l and emotion with arousal a and valence v. SBBS receives as input the models L and E_m, the current sequence x, the desired emotion values v and a, SBBS's parameter values b and k, which are explained below, and the desired length l of the piece to be generated.

In the first call to Composer, the sequence x is initialized with the the symbols of the first 4 timesteps of a random human-composed piece with the emotion v, a, as returned by E_s. Every time there is a transition from one emotion to another, we reinitialize the sequence x using the same process. This is used to bias the generative process and to emphasize emotion transitions.

To be used in real-time, Composer is invoked with the most recently captured speech signal v and returns a composed piece of music. While the most recent piece is being played at the game table, Composer receives another signal v and composes the next excerpt. One also needs to define the length of the signal v. In our implementation, similar to Padovani et al. (2017), we use YouTube's subtitle system as the speech recognition system S. Therefore, signals v are long enough to form a subtitle.

Classifying the Story's Emotion

In order to have a common model of emotion between stories and music, we use a mapping from Bardo's four emotion model to the valence-arousal model. Namely, we have Suspenseful mapping to low valence and arousal ($v = 0, a = 0$); Agitated to low valence and high arousal ($v = 0, a = 1$); Calm to high valence and low arousal ($v = 1, a = 0$); and Happy to high valence and arousal ($v = 1, a = 1$).

For example, in the context of the game Dungeons and Dragons, the sentence "Roll initiative" is normally said at the beginning of battles and it can be considered ($v = 0, a = 1$), once a battle is a negative (dangerous) moment with high energy. "Roll initiative" is normally classified as Agitated in Padovani et al.'s dataset. This mapping allows us to use the valence-arousal model with the labeled CotW dataset.

The valence-arousal mapping is based on the model used to annotate the VGMIDI dataset. When human subjects annotated that dataset, they used a continuous valence/arousal model with labels defining a fixed set of discrete basic emotions (e.g. happy or sad) (Ferreira and Whitehead 2019).

Given the limited amount of TRPG stories labeled according to emotion (there are only 5,892 sentences in the CotW dataset), we use a transfer learning approach to classify the sentences (Radford et al. 2018). We fine-tune a high-capacity BERT architecture with the CotW dataset (Devlin et al. 2018). We use BERT because it outperforms several other transformers across different NLP tasks (Devlin et al. 2018). Although in Algorithm 1 we depict the classifier for story emotions as a single E_s model, in our implementation we treat valence and arousal independently, thus we fine-tune a pre-trained BERT for each dimension.

Classifying the Music's Emotion

As was the case with the TRPG stories, given the limited amount of MIDI pieces labeled according to emotion, we also apply a transfer learning approach to classify emotion in music (E_m). However, different than the E_s model where we fine-tune a BERT architecture, for E_m we fine-tune a GPT-2 architecture (Radford et al. 2019). We use GPT-2 for E_m because it is better suited for sequence generation than BERT. Similarly to E_s, model E_m also treats valence and arousal independently. Thus, we fine-tune a pre-trained GPT-2 for each of these dimensions.

To the best of our knowledge, in the symbolic music domain, there are no publicly available high-capacity LM pre-trained with large (general) datasets. Typically, models in this domain are trained with relatively small and specific datasets. For example, the MAESTRO dataset (Hawthorne et al. 2019), the Bach Chorales (Hadjeres, Pachet, and Nielsen 2017) and the VGMIDI (Ferreira and Whitehead 2019) dataset. We pre-train a general high-capacity GPT-2 architecture as a language model (Radford et al. 2019) using a new dataset we created called ADL (Augmented Design Lab) Piano MIDI dataset [1].

The ADL Piano MIDI dataset is based on the Lakh MIDI dataset (Raffel 2016), which, to the best of our knowledge, is the largest MIDI dataset publicly available. The Lakh MIDI dataset contains a collection of 45,129 unique MIDI files that have been matched to entries in the Million Song dataset (Bertin-Mahieux et al. 2011). Among these files, there are many versions of the same piece. We kept only one version of each piece. Given that the datasets for emotion classification in music are limited to piano only, we extracted from the Lakh MIDI dataset only the tracks with instruments from the "piano family"(MIDI program numbers 1-8 in the dataset). This process generated a total of 9,021 unique piano MIDI files. These files are mainly Rock and Classical pieces, so to increase the genre diversity (e.g. Jazz, Blues, and Latin) of the dataset, we included an additional 2,065 files scraped from public sources on the Internet[2]. All files in the final collection were de-duped according to their MD5 checksum. The final dataset has 11,086 pieces.

After pre-training the high-capacity GPT-2 model, we fine-tune two independent models (one for valence and one for arousal) with an extended version of the VGMIDI dataset (Ferreira and Whitehead 2019). We extended the VGMIDI dataset from 95 to 200 labeled pieces using the same annotation method of the original dataset. All the 200 pieces are piano arrangements of video game soundtracks labeled according to the valence-arousal model of emotion.

Encoding We encode a MIDI file by parsing all notes from the NOTE_ON and NOTE_OFF events in the MIDI. We define a note as a set $z = (z_p, z_s, z_d, z_v)$, where $\{z_p \in \mathbb{Z} | 0 \leq z_p < 128\}$ is the pitch number, $\{z_s \in \mathbb{Z} | z_s \geq 0\}$ is the note starting time in timesteps, $\{z_d \in \mathbb{Z} | 0 \leq z_d \leq 56\}$ is note duration in timesteps and $\{z_v \in \mathbb{Z} | 0 \leq z_v < 128\}$ is the note velocity. Given a MIDI NOTE_ON event, we parse a note z by retrieving the starting time z_s (in seconds), the pitch number z_p and the velocity z_v from that event. To calculate the note duration z_d, we find the correspondent NOTE_OFF event of the given NOTE_ON and retrieve the NOTE_OFF end time z_e (in seconds). We discretize z_s and z_e to compute the

[1]https://github.com/lucasnfe/adl-piano-midi

[2]https://bushgrafts.com/midi/ and http://midkar.com/jazz/

Algorithm 2 Stochastic Bi-Objective Beam Search

```
Require: Music emotion classifier $E_m$, LM $L$, previously
    composed symbols $x$, valence and arousal values $v$ and
    $a$, number $k$ of symbols to consider, beam size $b$, length
    $l$ in seconds of the generated piece.
Ensure: Sequence of symbols of $l$ seconds.
 1: $B \leftarrow [x], j \leftarrow 0$
 2: while $l(y[t : t + j]) < l, \forall y \in B$ do
 3:     $C \leftarrow \{\}$
 4:     for all $m \in B$ do
 5:         $C_m \leftarrow \{m \cup s | s \in V\}$
 6:         $C_m \leftarrow k$ elements $y$ from $C_m$ with largest $p_L(y)$
 7:         $C \leftarrow C \cup C_i$
 8:     $B \leftarrow b$ sequences $y$ sampled from $C$ proportionally
        to $p_L(y)(1 - |v - E_{m,v}(y)|)(1 - |a - E_{m,a}(y)|)$
 9:     $j \leftarrow j + 1$
10: return $m \in B$ such that $p_L(m) = \max_{y \in B} p_L(y)$ and
    $l(y[t : t + j]) \geq l$
```

note duration $z_d = t \cdot z_e - t \cdot z_s$ in timesteps, where t is a parameter defining the sampling frequency of the timesteps.

We derive a sequence $x = \{z_v^1, z_d^1, z_p^1, \cdots, z_v^n, z_d^n, z_p^n\}$ of tokens for a given MIDI file by (a) parsing all notes z^i from the file, (b) sorting them by starting time z_s^j and (c) concatenating their velocity z_v^j, duration z_d^j and pitch z_p^j. We add two special tokens `TS` and `END` in the sequence x, to mark the end of a timestep and the end of a piece, respectively. This encoding yields a vocabulary V of size $|V| = 314$.

Stochastic Bi-Objective Beam Search

Next, we describe how one can use a LM and a music emotion classifier to bias the process of music generation to match a particular emotion (line 3 of Algorithm 1). For that we introduce Stochastic Bi-Objective Beam Search (`SBBS`), a search algorithm guided by the LM L and the music emotion classifiers, denoted as $E_{m,v}$ and $E_{m,a}$, for valence and arousal. The goal of `SBBS` is to allow for the generation of pieces that sound "good" (i.e., have high probability value according to the trained LM), but that also match the current emotion of the story being told by the players.

We call `SBBS` "stochastic" because it samples from a distribution instead of greedily selecting the best sequences of symbols, as a regular beam search does. The stochasticity of `SBBS` allows it to generate a large variety of musical pieces for the same values of v and a. We also call it "bi-objective" because it optimizes for realism and emotion.

The pseudocode of `SBBS` is shown in Algorithm 2. In the pseudocode we use letters x, y and m to denote sequences of musical symbols. Function $p_L(y) = \prod_{y_t \in y} P(y_t | y_0, \cdots, y_{t-1})$ is the probability of sequence y according to the LM L; a high value of $p_L(y)$ means that y is recognized as a piece of "good quality" by L. We denote as $l(y)$ the duration in seconds of piece y. Finally, we write $x[i : j]$ for $j \geq i$ to denote the subsequence of x starting at index i and finishing at index j.

`SBBS` initializes the beam structure B with the sequence x passed as input (line 1). `SBBS` also initializes variable j for counting the number of symbols added by the search. `SBBS` keeps in memory at most b sequences and, while all sequences are shorter than the desired duration l (line 2), it adds a symbol to each sequence (lines 3–9). `SBBS` then generates all sequences by adding one symbol from vocabulary V to each sequence m from B (line 5); these extended sequences, known as the children of m, are stored in C_m.

The operations performed in lines 6 and 8 attempt to ensure the generation of good pieces that convey the desired emotion. In line 6, `SBBS` selects the k sequences with largest p_L-value among the children of m. This is because some of the children with low p_L-value could be attractive from the perspective of the desired emotion and, although the resulting piece could convey the desired emotion, the piece would be of low quality according to the LM. The best k children of each sequence in the beam are added to set C (line 7). Then, in line 8, `SBBS` samples the sequences that will form the beam of the next iteration. Sampling occurs proportionally to the values of $p_L(y)(1-|v-E_{m,v}(y)|)(1-|a-E_{m,a}(y)|)$, for sequences y in C. A sequence y has higher chance of being selected if L attributes a high probability value to y and if the music emotion model classifies the values of valence and arousal of y to be similar to the desired emotion. When at least one of the sequences is longer than the desired duration of the piece, `SBBS` returns the sequence with largest p_L-value that satisfies the duration constraint (line 10).

Empirical Evaluation

Our empirical evaluation is divided into two parts. First, we evaluate the accuracy of the models used for story and music emotion classification. We are interested in comparing the fine-tuned BERT model for story emotion classification with the simpler Naïve Bayes approach of Padovani, Ferreira, and Lelis (2017). We are also interested in comparing the fine-tuned GPT-2 model for music emotion classification with the simpler LSTM of Ferreira and Whitehead (2019). In the second part of our experiments we evaluate with a user study whether human subjects can recognize different emotions in pieces generated by Composer for the CotW campaign.

Emotion Classifiers

Story Emotion The story emotion classifier we use with Composer is a pair of BERT models, one for valence and one for arousal. For both models, we use the pre-trained BERT$_{BASE}$ of Devlin et al. (2018), which has 12 layers, 768 units per layer, and 12 attention heads. BERT$_{BASE}$ was pre-trained using both the BooksCorpus (800M words) (Zhu et al. 2015) and the English Wikipedia (2,500M words).

We independently fine-tune these two BERT models as valence and arousal classifiers using the CotW dataset (Padovani, Ferreira, and Lelis 2017). Fine-tuning consists of adding a classification head on top the pre-trained model and training all the parameters (including the pre-trained ones) of the resulting model end-to-end. All these parameters were fine-tuned with an Adam optimizer (Kingma and Ba 2014) with learning rate of 3e-5 for 10 epochs. We used mini-batches of size 32 and dropout of 0.5.

Alg.	Episodes									Avg.
	1	2	3	4	5	6	7	8	9	
NB	73	88	91	85	94	81	41	74	94	80
BERT	**89**	**92**	**96**	**88**	**97**	**81**	**66**	**83**	**96**	**87**

Table 1: Valence accuracy in % of Naïve Bayes (NB) and BERT for story emotion classification.

The CotW dataset is divided into 9 episodes, thus we evaluate the accuracy of each BERT classifier using a leave-one-out strategy. For each episode e, we leave e out for testing and train in the remaining episodes. For example, when testing on episode 1, we use episodes 2-8 for training. Every sentence is encoded using a WordPiece embedding (Wu et al. 2016) with a 30,000 token vocabulary.

We compare the fine-tuned BERT classifiers with a Naïve Bayes (NB) approach (baseline), chosen because it is the method underlying the original Bardo system. NB encodes sequences using a traditional bag-of-words with tf–idf approach. Table 1 shows the accuracy of the valence classification of both these methods per episode. The best accuracy for a given episode is highlighted in bold. The BERT classifier outperforms NB in all the episodes, having an average accuracy 7% higher. For valence classification, the hardest episode for both the models is episode 7, where BERT had the best performance improvement when compared to NB. The story told in episode 7 of CotW is different from all other episodes. While the other episodes are full of battles and ability checks, episode 7 is mostly the players talking with non-player characters. Therefore, what is learned in the other episodes does not generalize well to episode 7. The improvement in accuracy of the BERT model in that episode is likely due to the model's pre-training. Episodes 5 and 9 were equally easy for both methods because they are similar to one another. The system trained in one of these two episodes generalizes well to the other.

Table 2 shows the accuracy of arousal classification of both NB and BERT. The best accuracy for a given episode is highlighted in bold. Again BERT outperforms NB in all the episodes, having an average accuracy 5% higher. In contrast with the valence results, here there is no episode in which the BERT model substantially outperforms NB.

Alg.	Episodes									Avg.
	1	2	3	4	5	6	7	8	9	
NB	82	88	75	79	82	76	98	86	84	83
BERT	**86**	**90**	**77**	**86**	**89**	**88**	**99**	**90**	**88**	**88**

Table 2: Arousal accuracy in % of Naïve Bayes (NB) and BERT for story emotion classification.

Music Emotion The music emotion classifier is a pair of GPT-2 models, one for valence and one for arousal. We first pre-trained a GPT-2 LM with our ADL Piano MIDI dataset. We augmented each piece p of this dataset by (a) transposing p to every key, (b) increasing and decreasing p's tempo by 10% and (c) increasing and decreasing the velocity of all notes in p by 10% (Oore et al. 2017). Thus, each piece generated $12 \cdot 3 \cdot 3 = 108$ different examples.

The pre-trained GPT-2 LM has 4 layers (transformer blocks), context size of 1024 tokens, 512 embedding units, 1024 hidden units, and 8 attention heads. We then fine-tuned the GPT-2 LM independently using the VGMIDI dataset, for valence and for arousal. Similarly to BERT, fine-tuning a GPT-2 architecture consists of adding an extra classification head on top of the pre-trained model and training all parameters end-to-end. Similar to the story emotion classifiers, we fine-tuned the GPT-2 classifiers for 10 epochs using an Adam optimizer with learning rate 3e-5. Different from the story emotion classifiers, we used mini-batches of size 16 (due to GPU memory constrains) and dropout of 0.25. The VGMIDI dataset is defined with a train and test splits of 160 and 40 pieces, respectively. We augmented the dataset by slicing each piece into 2, 4, 8 and 16 parts of equal length and emotion. Thus, each part of each slicing generated one extra example. This augmentation is intended to help the classifier generalize for pieces with different lengths.

Algorithm	Valence	Arousal
Baseline LSTM	69	67
Fine-tuned LSTM	74	79
Baseline GPT-2	70	76
Fine-tuned GPT-2	**80**	**82**

Table 3: Accuracy in % of both the GPT-2 and LSTM models for music emotion classification.

We compare the fine-tuned GPT-2 classifiers with LSTM models that were also pre-trained with the ADL Piano Midi dataset and fine-tuned with the VGMIDI dataset. We chose LSTMs because they are the state-of-the-art model in the VGMIDI dataset (Ferreira and Whitehead 2019). The LSTMs have same size as the GPT-2 models (4 hidden layers, 512 embedding units, 1024 hidden units) and were pre-trained and fine-tuned with the same hyper-parameters. Table 3 shows the accuracy of both models for valence and arousal. We also report the performance of these models without pre-training (i.e., trained only on the VGMIDI dataset). We call these the baseline versions of the models.

Results show that using transfer learning can substantially boost the performance of both models. The fine-tuned GPT-2 is 10% more accurate in terms of valence and 8% in terms of arousal. The fine-tuned LSTM is 5% more accurate in terms of valence and 12% in terms of arousal. Finally, the fine-tuned GPT-2 outperformed the fine-tuned LSTM by 6% and 3% in terms of valence and arousal, respectively.

User Study

In our study we measure Composer's performance at generating music that matches the emotions of a story. We use Composer to generate a piece for a snippet composed of 8 contiguous sentences of each of the first 5 episodes of the CotW dataset. Each snippet has one emotion transition that

Method	Episodes																				Average		
	e1-p1		e1-p2		e2-p1		e2-p2		e3-p1		e3-p2		e4-p1		e4-p2		e5-p1		e5-p2				
	v	a	v	a	v	a	v	a	v	a	v	a	v	a	v	a	v	a	v	a	v	a	va
Baseline	56	**65**	39	56	39	62	39	**79**	**48**	**60**	**67**	**53**	**58**	70	**63**	**75**	25	36	**72**	58	**51**	**32**	**34**
Composer	**62**	60	**44**	**65**	**82**	**68**	**53**	68	24	55	46	43	25	**87**	37	55	**81**	**86**	51	**67**	**51**	30	**34**

Table 4: The percentage of participants that correctly identified the valence and arousal (v and a, respectively) intended by the methods for the pieces parts (p1 and p2).

happens in between sentences. The sentences are 5.18 seconds long on average. To test Composer's ability to generate music pieces with emotion changes, we asked human subjects to listen to the 5 generated pieces and evaluate the transitions of emotion in each generated piece.[3]

The user study was performed via Amazon Mechanical Turk and had an expected completion time of approximately 10 minutes. A reward of USD $1 was given to each participant who completed the study. In the first section of the study, the participants were presented an illustrated description of the valence-arousal model of emotion and listened to 4 examples of pieces from the VGMIDI dataset labeled with the valence-arousal model. Each piece had a different emotion: low valence and arousal, low valence and high arousal, high valence and low arousal, high valence and arousal.

In the second section of the study, participants were asked to listen to the 5 generated pieces (one per episode). After listening to each piece, participants had to answer 2 questions: (a) "What emotion do you perceive in the 1st part of the piece?" and (b) "What emotion do you perceive in the 2nd part of the piece?" To answer these two questions, participants selected one of the four emotions: low valence and arousal, low valence and high arousal, high valence and low arousal, high valence and arousal. Subjects were allowed to play the pieces as many times as they wanted before answering the questions. The final section of the study was a demographics questionnaire including ethnicity, first language, age, gender, and experience as a musician. To answer the experience as a musician, we used a 1-to-5 Likert scale where 1 means "I've never studied music theory or practice" and 5 means "I have an undergraduate degree in music".

We compare Composer with a baseline method that selects a random piece from the VGMIDI dataset whenever there is a transition of emotion. The selected piece has the same emotion of the sentence (as given by the story emotion classifier). To compare these two methods, we used a between-subject strategy where Group A of 58 participants evaluated the 5 pieces generated by Composer and another Group B of 58 participants evaluated the 5 pieces from the baseline. We used this strategy to avoid possible learning effects where subjects could learn emotion transitions from one method and apply the same evaluation directly to the other method. The average age of groups A and B are 34.96 and 36.98 years, respectively. In Group A, 69.5% of participants are male and 30.5% are female. In Group B, 67.2% are male and 32.8% are female. The average musicianship of the groups A and B are 2.77 and 2.46, respectively.

Table 4 shows the results of the user study. We consider both parts (p1 and p2 in the table) of each episode as an independent piece. The table presents the percentage of participants that correctly identified the pieces' valence and arousal ("v" and "a" in the table, respectively), as intended by the methods. For example, 87% of the participants correctly identified the arousal value that Composer intended the generated piece for part p1 of episode 4 (e4-p1) to have. We refer to the percentage of participants that are able to identify the approach's intended emotion as the approach's accuracy. We also present the approaches' average accuracy across all pieces ("Average" in the table) in terms of valence, arousal, and jointly for valence and arousal ("va" in the table). The "va"-value of 34 for Composer means that 34% of the participants correctly identified the system's intended values for valence and arousal across all pieces generated.

Composer outperformed the Baseline in e1-p2, e2-p1, and e5-p1. Baseline outperformed Composer e3-p1, e3-p2 and e4-p2. In the other four parts, one method performed better for valence whereas the other method performance better for arousal. Overall, the average results show that both systems performed very similarly. Both of them had an average accuracy on the combined dimensions equal to 34%. The difference between these two methods and a system that selects pieces at random (expected accuracy of 25%) is significant according to a Binomial test ($p = 0.02$). These results show that the participants were able to identify the emotions in the generated pieces as accurately as they were able to identify the emotions in human-composed pieces. This is an important result towards the development of a fully automated system for music composition for story-based tabletop games.

Conclusions

This paper presented Bardo Composer, a system that automatically composes music for tabletop role-playing games. The system processes sequences from speech and generates pieces one sentence after the other. The emotion of the sentence is classified using a fine-tuned BERT. This emotion is given as input to a Stochastic Bi-Objective Beam Search algorithm that tries to generate a piece that matches the emotion. We evaluated Composer with a user study and results showed that human subjects correctly identified the emotion of the generated music pieces as accurately as they were able to identify the emotion of pieces composed by humans.

[3]Generated pieces can be downloaded from the following link: https://github.com/lucasnfe/bardo-composer

References

Bergström, K., and Björk, S. 2014. The case for computer-augmented games. *Transactions of the Digital Games Research Association* 1(3).

Bertin-Mahieux, T.; Ellis, D. P.; Whitman, B.; and Lamere, P. 2011. The million song dataset.

Briot, J.-P.; Hadjeres, G.; and Pachet, F. 2017. Deep learning techniques for music generation-a survey. *arXiv preprint arXiv:1709.01620*.

Dathathri, S.; Madotto, A.; Lan, J.; Hung, J.; Frank, E.; Molino, P.; Yosinski, J.; and Liu, R. 2019. Plug and play language models: a simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*.

Davis, H., and Mohammad, S. M. 2014. Generating music from literature. *Proceedings of the 3rd Workshop on Computational Linguistics for Literature (CLfL)* 1–10.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ferreira, L. N., and Whitehead, J. 2019. Learning to generate music with sentiment. In *Proceedings of the International Society for Music Information Retrieval Conference*, ISMIR'19.

Hadjeres, G.; Pachet, F.; and Nielsen, F. 2017. Deepbach: a steerable model for bach chorales generation. In *International Conference on Machine Learning*, 1362–1371.

Hawthorne, C.; Stasyuk, A.; Roberts, A.; Simon, I.; Huang, C.-Z. A.; Dieleman, S.; Elsen, E.; Engel, J.; and Eck, D. 2019. Enabling factorized piano music modeling and generation with the MAESTRO dataset. In *International Conference on Learning Representations*.

Keskar, N. S.; McCann, B.; Varshney, L. R.; Xiong, C.; and Socher, R. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Kingma, D., and Ba, J. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Kool, W.; Hoof, H. V.; and Welling, M. 2019. Stochastic beams and where to find them: The Gumbel-top-k trick for sampling sequences without replacement. In Chaudhuri, K., and Salakhutdinov, R., eds., *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 3499–3508. Long Beach, California, USA: PMLR.

Monteith, K.; Martinez, T. R.; and Ventura, D. 2010. Automatic generation of music for inducing emotive response. In *International Conference on Computational Creativity*, 140–149.

Oore, S.; Simon, I.; Dieleman, S.; and Eck, D. 2017. Learning to create piano performances. In *NIPS 2017 Workshop on Machine Learning for Creativity and Design*.

Padovani, R.; Ferreira, L. N.; and Lelis, L. H. S. 2017. Bardo: Emotion-based music recommendation for tabletop role-playing games. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*.

Padovani, R.; Ferreira, L. N.; and Lelis, L. H. S. 2019. Be inaccurate but don't be indecisive: How error distribution can affect user experience. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2604–2611.

Poole, D. L., and Mackworth, A. K. 2010. *Artificial Intelligence: foundations of computational agents*. Cambridge University Press.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving language understanding by generative pre-training. In *Arxiv*.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language models are unsupervised multitask learners. *OpenAI Blog* 1(8):9.

Raffel, C. 2016. *Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching*. Ph.D. Dissertation, Columbia University.

Russell, J. A. 1980. A circumplex model of affect. *Journal of personality and social psychology* 39(6):1161.

Vijayakumar, A. K.; Cogswell, M.; Selvaraju, R. R.; Sun, Q.; Lee, S.; Crandall, D.; and Batra, D. 2018. Diverse beam search for improved description of complex scenes. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Williams, D.; Kirke, A.; Eaton, J.; Miranda, E.; Daly, I.; Hallowell, J.; Roesch, E.; Hwang, F.; and Nasuto, S. J. 2015a. Dynamic game soundtrack generation in response to a continuously varying emotional trajectory. In *Audio Engineering Society Conference: 56th International Conference: Audio for Games*. Audio Engineering Society.

Williams, D.; Kirke, A.; Miranda, E. R.; Roesch, E.; Daly, I.; and Nasuto, S. 2015b. Investigating affect in algorithmic composition systems. *Psychology of Music* 43(6):831–854.

Wu, Y.; Schuster, M.; Chen, Z.; Le, Q. V.; Norouzi, M.; Macherey, W.; Krikun, M.; Cao, Y.; Gao, Q.; Macherey, K.; et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Zhu, Y.; Kiros, R.; Zemel, R.; Salakhutdinov, R.; Urtasun, R.; Torralba, A.; and Fidler, S. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, 19–27.