Exploring the Efficacy of Pre-trained Checkpoints in Text-to-Music Generation Task

Shangda Wu,[†] Maosong Sun[†‡]

[†] Department of Music AI and Information Technology, Central Conservatory of Music
[‡] Department of Computer Science and Technology, Tsinghua University
shangda@mail.ccom.edu.cn, sms@tsinghua.edu.cn

Abstract

Benefiting from large-scale datasets and pre-trained models, the field of generative models has recently gained significant momentum. However, most datasets for symbolic music are very small, which potentially limits the performance of data-driven multimodal models. An intuitive solution to this problem is to leverage pre-trained models from other modalities (e.g., natural language) to improve the performance of symbolic music-related multimodal tasks. In this paper, we carry out the first study of generating complete and semantically consistent symbolic music scores from text descriptions, and explore the efficacy of using publicly available checkpoints (i.e., BERT, GPT-2, and BART) for natural language processing in the task of text-to-music generation. Our experimental results show that the improvement from using pre-trained checkpoints is statistically significant in terms of BLEU score and edit distance similarity. We analyse the capabilities and limitations of our model to better understand the potential of language-music models.

Introduction

Creativity was once thought to be a privilege of humans, but after training on large amounts of data, transformer-based models (**?**) also exhibit this capability to some extent. Recent transformer-based models can generate human-like texts (**?**), autocomplete codes (**?**), reconstruct images (**?**), or compose music (**?**). Models designed by researchers focused on AI creativity have also shown mind-blowing generation results across modalities. For example, DALLE 2 (**?**), a text-conditional image generation model, can generate realistic images and creative art from natural language captions. On the other hand, AudioGen (**?**), a textually guided audio generation model, although trained on general audio, can also generate music clips by giving proper textual descriptions.

Symbolic music, unlike raw audio, contains explicit musical information, such as note onsets and pitch on individual tracks. With the development of deep learning technology, symbolic music generation (**?**) has shown unprecedented progress in the past few years. However, the lack of datasets has been a major limitation in the task of symbolic music generation. Due to the absence of text-music pairs data, the task of text-conditional symbolic music generation has not been given enough attention in the past.

Even without text-music datasets, a few researchers managed to achieve the conversion of input text into symbolic music. Rangarajan designed three strategies (**?**) for mapping text to notes. But as it is based on character-level mappings, the generated music is very random and does not reflect the semantic information in the text. TransProse (**?**) contains several mapping rules that generate music based on the density of emotional words (**?**) in the given text. However, TransProse does not reflect the non-emotional information in the text, and its creativity is limited by those hand-crafted mapping rules. BUTTER (**?**), a GRU-based model, can search and generate music segments given rigid text descriptions and vice versa. The dataset used by BUTTER contains 16,257 folk songs, and the paired text descriptions are synthesised from keywords (i.e., 25 keys, 6 meters, and 3 styles). Although BUTTER is a data-driven model, its flexibility is limited as the paired texts are synthesised from specified keywords, and it can only generate 16-beat (4-bar) music segments. The most recent attempt is Mubert[1], which can generate music from user-given prompts. Although its generated music is of high quality, it does not directly generate music from the input text. The input text and Mubert API tags are encoded by Sentence-BERT (**?**), and the closest tag vectors are selected and then used for music generation. All sounds are created beforehand by musicians and sound designers, and thus Mubert is more like generating a combination of sounds, instead of music.

In this paper, we model the task of text-to-music generation as a sequence-to-sequence problem, and develop a transformer-based model that is capable of generating complete and semantically consistent music scores directly from descriptions in natural language based on text[2]. To the best of our knowledge, this is the first model that achieves text-conditional symbolic music generation which is trained on real text-music pairs, and the music is generated entirely by the model without any hand-crafted rules. We further explore the efficacy of using publicly available pre-trained BERT, GPT-2, and BART checkpoints, and aim to provide empirical answers to the following research questions.

- Does using pre-trained checkpoints improve the performance of language-music models?
- To what extent can language-music models learn the relationship between natural language and symbolic music?

[1] https://github.com/MubertAI/Mubert-Text-to-Music
[2] https://github.com/sander-wood/text-to-music

We believe that researchers from the fields of symbolic music generation and natural language processing can take insights from this paper when dealing with language-music problems in the future.

Dataset

Large-scale data is the cornerstone that underpins data-driven models. For example, recent multimodal language-vision models are usually trained on hundreds of millions of image-text pairs (**?**). Likewise, large-scale paired text-music datasets are a prerequisite for language-music models.

To make models learn the relationship between natural language and symbolic music, we collected as many text-music pairs as possible, composing a text-music dataset called *Textune*. This dataset contains 282,870 English text-tune pairs, where all tunes are represented in ABC notation[3]. As ABC notation encodes music scores into sequences of ASCII characters, using this music notation system makes it easy to model the text-to-music generation task as a sequence-to-sequence problem.

All scores in Textune can be written on one stave (for vocal solo or instrumental solo) in standard classical notation, and are in a variety of styles, e.g., blues, classical, folk, jazz, pop, and world music. The scores are not all western, but also include some from other regions (e.g., Asia and Africa). In addition, all the scores have at least eight bars to present a complete musical idea.

Due to the abstract nature of music, it is much more difficult to describe music accurately than to describe images. The description of the same piece of music can vary significantly from person to person according to their respective musical backgrounds. In general, the valid text descriptions in Textune can be categorised as follows: 1) musical analysis (e.g., tonal analysis and harmonic analysis), 2) meta-information (e.g., key and meter), 3) the context in which the piece was composed (e.g., history and story), and 4) subjective perceptions (e.g., sentiment and preference).

Models

Sequence-to-sequence generation tasks typically choose from three transformer-based architectures: encoder-decoder (**?**), language model (**?**), and prefix LM (**?**). Based on previous findings (**?**), we use the encoder-decoder architecture. As using pre-trained checkpoints can improve the performance of models in various tasks (**?**), we hypothesised that it would be beneficial for the text-to-music generation task to use pre-trained checkpoints that already provide robust natural language representations. We use the following checkpoints to initialise the language-music model (see Table 1).

RND (Random): A randomly initialised encoder with a maximum input length of 1,024. We used byte-pair encoding for tokenization and only kept tokens with a minimum frequency of 100, ending up with a vocabulary size of 7,418. Setting a smaller minimum frequency would include a large number of unintelligible and meaningless tokens.

[3]https://abcnotation.com/

Table 1: The configurations of various pre-trained checkpoints. The BART checkpoints initialise both the encoder and the decoder, while the rest of the checkpoints only initialise the encoder. The number of parameters in the encoder/decoder-only checkpoints varies slightly due to different sizes of vocabulary.

Checkpoint	Layers	Hidden	Heads	Params
RND	12	768	12	91M
BERT	12	768	12	109M
GPT-2	12	768	12	117M
BART-base	6+6	768	16	139M
BART-large	12+12	1024	16	406M

BERT (?): It is an encoder-only bidirectional transformer pre-trained using a combination of masked language modelling objective and next sentence prediction on a large corpus comprising the Toronto Book Corpus and Wikipedia. We use the `bert-base-cased` checkpoint to initialise the encoder in our experiments.

GPT-2 (?): It is a decoder-only, unidirectional transformer pre-trained using language modelling on a very large corpus of ≈40 GB of text data. We use the `gpt2-small` checkpoint to initialise the encoder (not the decoder) in our experiments.

BART (?): It uses a standard encoder-decoder architecture with a bidirectional encoder (like BERT) and a unidirectional decoder (like GPT). The pre-training task involves randomly shuffling the order of the original sentences and a novel in-filling scheme, where spans of text are replaced with a single mask token. We use both checkpoints `bart-base` and `bart-large` in our experiments. The configurations of these two checkpoints do not exactly match those of others: the encoder/decoder of BART-base has only 6 layers (instead of 12), while BART-large has 1,024 units per layer (instead of 768), and they both have 16 heads (instead of 12). This is not compared apples to apples, but can provide us with baselines for initialisation using encoder-decoder checkpoints.

Except for BART, the decoder for all other models is randomly initialised with the same configuration as the RND encoder. Since almost every character in the ABC notation is semantically independent, we took character-level tokenization (but added some common notations), with a vocabulary size of 164. We trained all models using the same learning rate $\alpha = 10^{-4}$ (for BART-large, it is 5×10^{-5}), with a 1,000-step linear warmup and learning rate decay. We trained a total of 20 epochs with a batch size of 8, using the AdamW optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and a weight decay coefficient of 0.01.

Experiments

We randomly selected 2,828 (1%) pairs from Textune as the validation set, and the rest were used for training. For both training and inference, we truncated all text sequences to the maximum input length of 1,024 (for BERT, it is 512).

Table 2: Results of various pre-trained checkpoints on the validation set. We found statistically significant improvements in BLUE-N and EDS for some pre-trained checkpoints, but not all of them had such benefits (e.g., BERT and BART-large).

Checkpoint	BLEU-2	BLEU-3	BLEU-4	DIST-1	DIST-2	DIST-3	EDS
RND	44.4720.64	35.8818.12	28.8416.22	**10.614.32**	**28.4711.03**	**41.7915.24**	38.0213.10
BERT	21.7214.01	15.4410.61	10.477.90	8.715.49	19.0311.95	27.0917.03	24.8110.40
GPT-2	46.7621.23	38.2019.41	31.1418.14	10.394.39	27.8611.28	40.9315.70	39.9414.84
BART-base	**48.3421.47**	**39.8520.11**	**32.8219.22**	10.294.31	27.9811.01	41.2815.15	**40.7715.75**
BART-large	22.4114.28	15.9610.71	10.917.78	9.216.36	20.5413.28	29.0918.04	24.9810.27

In Fig. 1, we display the training and validation curves for the five pre-trained checkpoints mentioned before. Note that the vocabulary size of the BART decoder (50,265 tokens) is much larger than that of the other randomly initialised decoders (164 tokens), which leads to higher losses but does not necessarily mean that the generation quality of BART is worse. Regardless, Fig. 1 suggests that using pre-trained checkpoints to initialise the model does not guarantee a lower validation loss.

Because of the small amount of data, all models showed different degrees of overfitting. In particular, even though the number of parameters is approximately three times that of BART-base, the validation loss of BART-large is not lower. Intuitive ways to solve this problem are to collect more data, reduce the model size, or tune hyperparameters. However, due to the scarcity of symbolic music data, it is unlikely to find a human-annotated text-music dataset that is at least an order of magnitude larger (i.e., 1 million text-music pairs) than Textune for a long time. Thus, using smaller models or tuning hyperparameters are attainable solutions for now.

To verify the generation quality, we used all checkpoints with their lowest validation loss to generate tunes based on descriptions from the validation set, and using nucleus sampling with top-$p = 0.9$. We used the following metrics to evaluate the generated tunes from different models.

BLEU-N (?): An algorithm for evaluating the quality of text measures the proportion of N-grams in the reference text are reproduced by the candidate text. The higher the value, the closer the generated tunes are to ground truth. This is a common metric used in sequence-to-sequence tasks.

DIST-N (?): It evaluates the diversity of generated samples. A higher value of DIST-N means a higher proportion of distinct N-grams. We use this reference-free metric as text-to-music generation can be seen as conditional music generation, which is a creative task.

EDS: Edit **D**istance **S**imilarity is based on the Levenshtein distance $lev(a, b)$ to indicate how similar the generated tune b and the ground truth a are at the character level, ranging from 0 (no match at all) to 100 (exact match), which can be formalised as follows:

$$EDS(a, b) = (1 - \frac{lev(a, b)}{max(|a|, |b|)}) \times 100, \qquad (1)$$

where $|a|$ and $|b|$ are the length of two strings. As ABC tunes are nearly character-level sequences, EDS can effectively reflect the similarity between the generated tune and the ground truth.

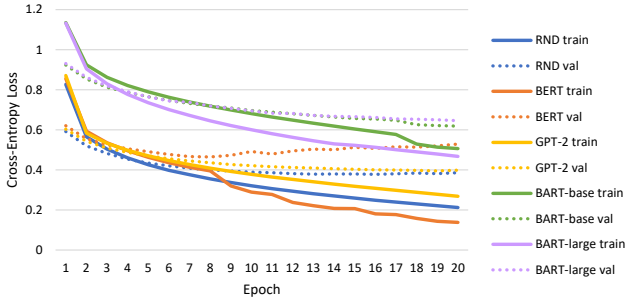

Figure 1: Training and validation curves of various models.

As shown in Table 2, RND generated more diverse tunes (higher DIST-N), but the tunes generated by GPT-2 and BART-base are closer to the ground truth (higher BLEU-N and EDS). We performed independent samples t-tests, and found statistically significant differences in the BART-base results of BLEU-N and EDS compared to that of RND, i.e., p-value < 0.05. These results show that the use of pre-trained checkpoints can improve the performance of the model on language-music tasks significantly.

For two randomly selected tunes from the Textune dataset, the average EDS is around 30%, while the results for RND, GPT-2 and BART-base on this metric are around 40%. Given the diversity of music, it indicates that these models can extract meaningful information from descriptions to generate music. However, the tunes generated by BERT and BART-large are not only low in diversity (lower DIST-N) but also far from the ground truth (lower BLEU-N and EDS). EDS suggests that the tunes generated by them are even more dissimilar to the ground truth compared to tunes randomly selected from Textune. We manually examined the tunes generated by these two models and found that there were a large number of instances of degeneration. We observed that they tend to repeatedly generate notes like `z8 | z8 | z8`. As shown in Fig. 1, we suggest that the cause of this problem is the severe overfitting of BERT and BART-large.

Discussions

To demonstrate the capabilities and limitations of language-music models, several representative examples of generated tunes are given in Fig. 2. All the text descriptions in Fig. 2 were hand-crafted by us and are not from Textune. Due to space constraints, we only show RND-generated examples. We first tested whether the model could reproduce the tunes already present in Textune. The tune chosen here is "Twin-

Input Text A:
The tune "Twinkle, Twinkle, Little Star" is a popular English lullaby.

Output Tune A:
L:1/4
M:4/4
K:C
C C G G | A A G2 | F F E E | D D C2 | G G F F | E E D2 | G G F F | E E D2 | C C G G | A A G2 |
F F E E | D D C2 |]

Input Text B:
Lead sheet of the popular English lullaby "Twinkle, Twinkle, Little Star".

Output Tune B:
L:1/4
M:4/4
K:C
"C" C C G G |"F" A A"C" G2 |"G" F F"C" E E |"G" D D"C" C2 |"C" G G F F |"Em" E E"G" D2 |
"C" G G"F" F F |"Em" E E"G" D2 |"C" C C G G |"F" A A"C" G2 |"G" F F"C" E E |"G" D D"C" C2 |]

Input Text C:
Jazz arrangement of the popular English lullaby "Twinkle, Twinkle, Little Star".

Output Tune C:
L:1/8
M:4/4
K:F
"F" A3 G"Dm7" A c2 G |"Gm7" A B2"C7" G- G4 |"Gm7" d3 c"C7" d f2 c |"Gm7" f4-"C7" f2 z c |
"F" d3 c"F#dim7" d f2 c |"Gm7" f f2 d"A7b9" f2 z c |"Dm7" d3 c"G7" d f2 c |"C7" e4-"F#dim7" e2 z2 |
"Gm7" g3 f"C7" g b2 g |"F6" c' a2 f"Bb7" c'2 z c' |"Gm7" d'3 c'"C7" d' f2 c |
"F""Bbm6" f4-"F" f2 z2 ||"Dm7" f3 d"G7" f a2 f |"Gm7" b g2 f"C7" b2 z g |
"F" a3 g"Dm7" a c'2 g |"G7" a f2 d"A7b9" f2 z c |"Dm7" d3 c"G7" d f2 c |
"C7" e4-"F#dim7" e2 z2 |"Gm7" g3 f"C7" g b2 g |"F6""Bbm6" c' a2 f"A7b9" c'2 z c' |
"Dm7" d'3 c'"G7" d' f2 c |"C7" e4-"F#dim7" e2 z2 |"Gm7" g3 f"C7" g b2 g |
"F6" c' a2 f"Bbm6" c'2 z c' |"Gm7" d'3 c'"C7" d' f2 c |"F""Bbm6" f4-"F" f2 z2 |]

Input Text D:
Note Length-1/8
Key-D
Source-Dance Music of Ireland
Meter-6/8

Output Tune D:
L:1/8
M:6/8
K:D
f/e/ | dAA AFA | BGB A2 (f/e/) | dAA AFA | g3 f2 e | dAA AFA | BGB A2 f | aba gfe | fdd d2 ::
(a/g/) | (f/g/a)a b2 a | (f/g/a)a b2 a | (f/g/a)a faa | e2 f gfe | (f/g/a)a b2 a | (f/g/a)a b2 a | gfg eag | fdd d2 :|

Input Text E:
This is a traditional Irish dance music in the key of D and its meter is 6/8.

Output Tune E:
L:1/8
Q:1/4=100
M:4/4
K:C
CDE GAB | cde d2 B | cdc edc | B3 d3 | CDE GAB | cde d2 B | cde GAB | c3 c3 :: g2 g gfe |
f2 f fed | e2 e edc | d2 d dcB | CDE GAB | cde d2 B | cdc GAB | c3 c3 :|

Figure 2: Music generation examples of RND

kle, Twinkle, Little Star", which was present in Textune a total of 11 times. We found that when top-p was set to a low value (e.g., 0.5), the model almost always reproduced the tune perfectly, as shown in Fig. 2A. This means that the model does understand the relationship between the title and the tune. However, this also indicates that it is possible for the model to directly copy the music that exists in Textune. We recommend using a higher top-p when generating tunes using this model to avoid that problem.

We then tested the creativity of the model: generating the lead sheet and the jazz arrangement of "Twinkle, Twinkle, Little Star". According to Fig. 2B, the model does understand what a lead sheet is and succeeds in placing appropriate chord symbols for this tune. It should be noted that Textune does not contain any lead sheets for this tune. This demonstrates the potential of language-music models to be applied to the melody harmonization task. However, for the more creative task, melody style transfer, the model did not perform well. The tune in Fig. 2C, although it does have a very distinctive jazz style (e.g., rhythm, harmony), has a completely different melody from "Twinkle, Twinkle, Little Star". Given that the model can perform well on the melody harmonization task, we believe that the reason for its failure on the melody style transfer comes mainly from the small amount of text-music data. If the size of text-music datasets can reach the level of text-image datasets (**?**), achieving most music generation tasks, including those requiring a high degree of creativity, should not be a challenge anymore. We finally tested whether the model can follow the objective meta-information (e.g., key, meter) given in the text to generate tunes. We specified the key (D major), the meter (6/8), and the style of the music (Irish dance music). As shown in Fig. 2D and Fig. 2E, whether or not the model can generate music that matches the meta-information given in the text description depends on its format. When describing meta-information in a list format (Fig. 2D), the model can always follow the text accurately to generate tunes. The generated tune also exhibits distinctive characteristics of Irish dance music. For example, traditional Irish music is usually in a binary form (AABB), and the music generated here is exactly composed in that way. However, when the same information is given in a more loose way (Fig. 2E), the model does not follow the description well enough, even with a low top-$p = 0.5$. Although the actual meter of this generated tune is still 6/8 and is in keeping with the characteristics of Irish music, the generated music is in the key of C major and the meter in the header is 4/4. We tested this text format on the dual task (i.e., music-to-text generation) and found that when given the prompt "... in the key of", the model can always retrieve the meta-information correctly. Theoretically, the two tasks should be of equal difficulty, i.e., correctly translating the text to the header of ABC tunes or vice versa. More investigation is needed to determine the causes of this problem.

Conclusions

In this paper, we carry out the study of language-music models trained on large-scale text-music data. According to the experimental results, the use of pre-trained checkpoints leads to generated tunes that are much more similar to ground truth, but not improved in terms of diversity. Although the model can generate tunes that matched the semantic information of the text and exhibited a certain degree of creativity on some tasks, its creativity is limited, and it is input-sensitive. With a larger dataset, it is likely to develop a language-music model that performs well in music generation tasks that require a high degree of creativity. Voluptates eligendi quas labore ex, alias eveniet itaque ducimus odio quis aperiam dolor sunt, impedit nobis libero maiores odit magnam mollitia, ad eveniet adipisci pariatur.Amet quaerat eius iure in similique esse debitis sed reprehenderit alias, dicta ratione suscipit voluptas?Delectus amet quod corporis nam molestiae aperiam voluptatibus maiores, deserunt ipsum molestias cupiditate eum dicta officiis iure, debitis atque magnam nulla.Veniam repudiandae quia vel enim, dicta similique beatae quidem reprehenderit possimus et sed