2021 IEEE 4th International Conference on Electronic Information and Communication Technology (ICEICT) | 978-1-6654-3203-0/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICEICT53123.2021.9531287

APE-GAN: A Novel Active Learning Based Music Generation Model With Pre-Embedding

1st Wenyi Su
Mars Laboratory
Whittle School
Shenzhen, China
wsu111@whittleschool.org

2nd Yixuan Fang
Mars Laboratory
Whittle School
Shenzhen, China
yfang@whittleschool.org

3rd Zheng Li*
Mars Laboratory
Whittle School
Beijing, China
zli@whittleschool.org

4th Xin Steven*
EECS
Peking University
Beijing, China
mengxinpku2018@gmail.org

***Abstract*—Being able to generate realistic musics is one of the biggest challenges for Artificial Intelligence, and the current models do not have musical descriptive ability as humans have and the musics they produce are not highly realistic. This paper proposed an active learning based music generation model with pre-embedding (APE-GAN) that can use textual inputs to generate musics, and have increased performance by active learning and a checking mechanism with the discriminator in GAN model. Through experiments, this work shows that APE-GAN only needs 5% to 10% humans labelled data to achieve relatively good music generation ability. This method uses BERT to obtain embedding vector, thus provides prior and artistic thoughts from humans so that our model can achieve a higher level of creativity than most of the popular methods. Finally, it would be too subjective to discriminate by humans whether a music sequence generated is "good" or not. This paper used an evaluation metric (Kullback–Leibler divergence, or KL divergence) based on similarity of music sequences generated by similar or different meanings textual inputs. After using Lakh Pianoroll Dataset to train APE-GAN, the similar meaning textual inputs result in outputs with KL divergence approximate to 0, while different meaning textual inputs result in outputs with KL divergence approximate to 1.**

***Keywords*—Music Generation; Generative Adversarial Networks(GAN); BERT; Active Learning; KL Divergence**

I. Introduction

The development of Artificial Intelligence is turning the impossible to possible, and one of the most exciting innovations in recent years is AIs with creativity, especially in the form of composing musics. Among productions of all sorts of arts, making musics is one of the most challenging tasks due to its unlimited varieties and somehow intrinsic patterns, making generating musics one of the major goals for AI models.

Throughout the history, several important models have been proposed by researchers to generate musics. One of the earliest papers on music generation using neural networks is published in 2013, in which J.Bayer [1] discussed generating musics by using RNNs with fast dropout, a regularization method for generalized linear models. In 2014, K.Goel [2] proposed RNN-DBNs by stacking Restricted Boltzmann Machine in RNN models to create a better representation of the data for music generation. In 2016, A.Huang [3] was able to use deep neural networks to generate musics with proper rhythm, relatively complex structures, and various lengths of music notes. In the same year, the model C-RNN-GAN is proposed in which O.Mogren [4] used a generative adversarial architecture to combine two RNNs and resulted in more complexity in the generated musics and increased intensity span similarity between the generated musics and real musics. One of the closest attempts to generate "real" musics was made in 2017 by H.W.Dong [5] in the model named MuseGAN. Notably this model can generate multi-track polyphonic musics and reasonable accompaniments. However, with all these novel methods, some problems for music generation remain unsolved.

Firstly, to the extent of our knowledge, the most common music generation models generate musics based on a random noise, or a randomly generated sequence. [6] [7] [8] [9] Apparently, these models do not exploit musics' descriptive ability, which is one of the most important functions of musics. Specifically, most of the real musics contain a certain degree of thoughts and emotions of the composers, rather than being meaningless. Secondly, there are billions of music data on the internet, but most of them remain unlabelled. Thus, models before may capture some "bad" features during training from poorly written music pieces. It is nearly impossible to neglect all the musics with unpleasant characters due to limited time and human resources, and letting experts to label them will be very costly. Finally, for many models, the generated musics are sometimes not as realistic. In other words, most of the time, a human can still effortlessly distinguish generated musics from the real ones, either from the melodic or rhythmic perspective.

In order to tackle these issues, we propose an active learning based music generation model with pre-embedding which contains contextual word vectors created by word embedding as inputs, utilizes active learning, and uses the discriminator as the checking mechanism for the output music sequences during inference. We named our model Active learning based Pre-Embeding - Generative Adversarial Network (APE-GAN), a model that can hopefully result in more realistic music generation.

There are three contributions we make:

- We use the word embedding vectors pre-processed by BERT to replace the random noises as network inputs which can improve the descriptive ability of our generation model.
- We use active learning technology to efficiently train our APE-GAN to achieve high performance under the

condition of low cost.
- We implement the trained discriminator as a checking mechanism before output to increase quality of the generated musics.

The rest of this paper is organized as follows: Section II introduces the main components of our model. Section III expatiates the complete workflow of APE-GAN. Moverover, the experiment results and conclusion of the proposed method are discussed in Section IV. Finally, Section V gives conclusion.

II. Model Components

A. Music Generative Adversarial Network

Generative adversarial network, or GAN, is a kind of deep neural network with two components, the generator (G) and the discriminator (D). During training, G attempts to produce more realistic samples, while D tries to distinguish the generated samples from the real samples. By conducting this two player game, both G and D get better, such that eventually, G can be trained to produce fairly realistic samples [10] [11]. For generation of musics, because the music sequences for either real or generated musics are represented as fixed-size matrix, we use CNNs for both G and D. The training process of GAN can be modeled by:

$$\min_G \max_D \mathbf{E}_{x\sim p_d}[\log(D(\mathbf{x}))] + \mathbf{E}_{\mathbf{z}\sim p_z}[1 - \log(D(G(\mathbf{x})))] \tag{1}$$

where z denotes the random noise sampled from a prior distribution of sample space, p_d denotes the distribution of real data, and p_z denotes the prior distribution of z.

In this work, we reference MuseGAN as our baseline architecture. In MuseGAN, the generator is divided into two sub networks, the temporal structure generator G_t and music sequence generator G_m, in which G_t maps the input vector $\mathbf{z}$ to a sequence of latent vectors $\vec{\mathbf{z}}$ that carries temporal information (the length of notes and the original information of $\mathbf{z}$):

$$\vec{\mathbf{z}} = \{\vec{\mathbf{z}}^{(t)}\}_{t=1}^{T} \tag{2}$$

where t denotes number of bars and T denotes total number of bars. Then, $\vec{\mathbf{z}}$ is used by G_m to generate melody sequences bar by bar:

$$G(\mathbf{z}) = \{G_m(\vec{\mathbf{z}})\}_{t=1}^{T} \tag{3}$$

B. Pre-Word-Embedding

For our model, instead of adding a random noise as the input to let the model generate musics by itself, we decide to add a layer of word embedding (representation of text in the form of vectors) that takes a piece of text as the input before the data enter the GAN model. The output vector $\mathbf{z}$ from the word embedding layer will be used as the input for the GAN model. Due to the temporal correlation feature for musics and the GAN model, generated musics that deviates too much from the input text will be discriminated as "fake" by the discriminator. So, through training, the generator will get better at creating musics that are coherent to the input texts.

In order to let the vector be as representative for the text information as possible, we utilize the model BERT, or Pre-training of Deep Bidirectional Transformers for word embedding. For its principle, BERT includes a special technique named Masked LM (MLM) during training: it arbitrarily masks words in the sentence and tries to predict them. "Bidirectional" means that BERT checks in both directions and it uses both left and right surroundings of the masked word to predict it. BERT is extremely powerful due to its reliance on a mechanism called the Transformer. [13]

Transformers belong to a type of neural net called Sequence-to-Sequence. [14] Seq2Seq basically transforms a given sequence of data, such as a sequence of words, into another sequence. Seq2Seq models consist of an Encoder and a Decoder. The Encoder takes the input sequence and maps it into an n-dimensional vector (which is what we need in our model). This vector is fed into the Decoder which turns the vector into an output sequence. Before Transformers, Long-Short-Term-Memory (LSTM)-based models are very popular in constructing Seq2Seq models because the LSTM model can map sequences of data to vectors while selectively remembering the parts of the sequences it finds important. [15] [16]

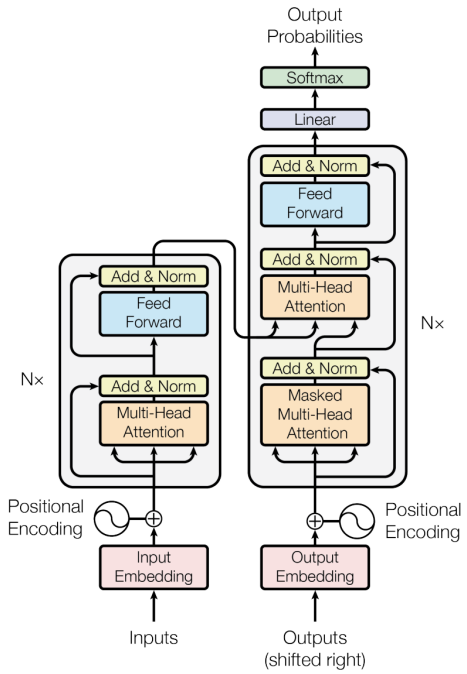

Fig. 1. Transformer Architecture [8]

Transformer is a novel architecture introduced by the paper "Attention Is All You Need". [17] As illustrated in Figure 1, both Encoder and Decoder are composed of modules that can be stacked on top of each other multiple times, which is described by Nx. The modules consist mainly of multi-head Attention and feed forward layers. The inputs and outputs (target sentences) are both first embedded into an n-dimensional

space. One important part of the model is the positional encoding of the different words in the embedding process. Since there is no recurrent networks to remember how a sequence is fed into a model, every word in the input sequence is given a relative position, since a sequence is largely dependent on the order of its elements. These positions are added to the embedded representation (n-dimensional vector) of each word. To understand Transformer better, intuitively speaking, the main mechanism of Transformer, which is Attention, can be explained as a way to look at an input sequence and decide at each step which other parts of the sequence are important. In "Attention Is All You Need", Attention is defined as:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{4}$$

where Q is a vector representation of one word in the sentence, K and V are both vector representations of all the words in the sequence, and d_k is the dimensions of Q and K. For the Encoder, Decoder, and multi-head attention modules, V consists of the same word sequence as Q. However, for the attention module which takes into account the Encoder and the Decoder sequences, V is different from the sequence represented by Q. This is because the values in V are multiplied and summed with some attention-weights a, defined as:

$$a = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right) \tag{5}$$

Going back to BERT, because its goal is to generate a language representation model, it only needs the Encoder part of the Transformer. In short, BERT is a bidirectional Encoder of many layers of Transformers stacked together. By using BERT, APE-GAN can create contextual word embedding vector $\mathbf{z}$ for any sentences and use it as the input for the GAN model as replace the random noise.

C. Active Learning

In order to let our model to use a small amount of labelled data with rich amount of unlabelled data on the internet in training and result in high precision, we incorporate pooled-based active learning in our model. In specific, there will be several rounds of training. As GAN model is learning in each round, the discriminator will label some data as "uncertain" if it finds that it deviates only by a small amount from being either "real" or "fake". Specifically, for a data, the discriminator will output a probability $p \in (0, 1)$ that the data is real or not:

$$\text{if } p \leq 0.5, \text{then the data is discriminated as "fake"} \tag{6}$$

$$\text{if } p > 0.5, \text{then the data is discriminated as "real"} \tag{7}$$

and if $p \in (0.475, 0.525)$, this data will be labelled as "uncertain". These "uncertain" data will be selected from the complete dataset and delivered to the oracle, or humans, who will label these data via online interface. After labelling, these data will be sent back to the original dataset to update it. Due to their uncertainty, these labelled data will now be very informative because by learning these data, the discriminator will be more powerful at drawing the boundary between what is "real" and what is "fake", especially for the relatively ambiguous data. Because of the selective nature of active learning, it is rather cost efficient in increasing the performance of APE-GAN by getting a relatively small amount of labelled data [18]. The complete process is illustrated Figure 2.

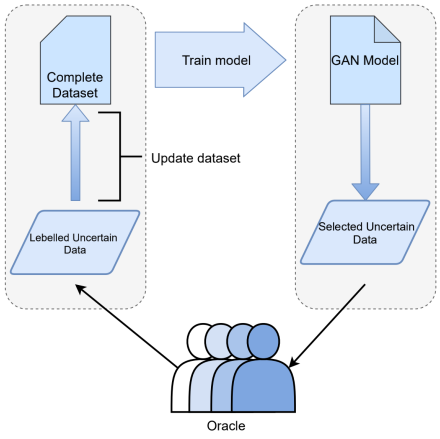

Fig. 2. Active Learning Process

D. Discriminator Checking Mechanism

For many GAN models used in music generation, though both the generator and discriminator are trained in the training process, only the generator is used to generate musics, while the discriminator's function is yet to be exploited. This may result in the case that the generator still generates unrealistic and poor quality musics by chance.

Therefore, for our final trained model that is used to generate music samples, the discriminator will be placed before the final output to check whether the generated music sequence is real or not according to Equation (6) and Equation (7). If it judges that the music is somehow fake, it will abandon the generated music and call the generator again to generate a new piece of music. This checking mechanism will increase the frequency by which the model outputs realistic musics.

III. APE-GAN Inference Workflow

A. Steps of Algorithm

With the knowledge in Section III, we can now introduce the entire process of music generation inference of APE-GAN, divided in the following steps, as illustrated in Figure 3 and Figure 4:

1) Choose a piece of text that one wants to describe with APE-GAN.
2) Use word embedding with BERT as model input.
3) Compose music sequence by the generator.
4) The discriminator determines whether the generated sequence is "real" or "fake" by using Equation (6) and Equation (7).
   - If the sequence is discriminated as "fake", then delete the sequence and repeat Step 1 to Step 3.

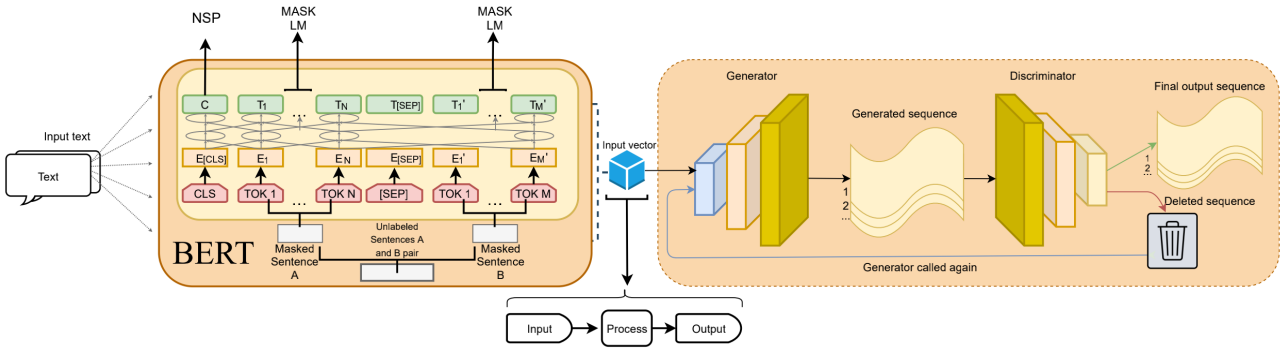

Fig. 3. APE-GAN Inference Work Flow

- If the sequence is discriminated as "real", then output the sequence.

B. Flow Chart

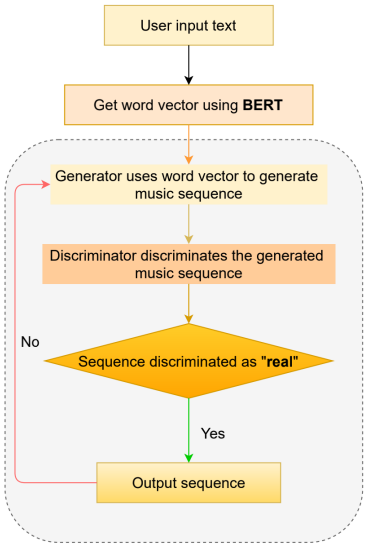

Fig. 4. APE-GAN Inference Work Flow

IV. Experiment And Result

A. Experiment Environment

In this paper the environment platform is Linux Ubuntu 18.04, with 2 RTX6000 GPUs, and 1 Inter Xeon CPU with 40 cores. The source data we used is the cleansed version of Lakh Pianoroll Dataset. All the music data are distributed in lpd_cleansed.tar.gz (636 MB). The dataset provides 21,425 pianorolls in the format of .npz files containing songs from all genres with the time signature of 4/4 and first beat starts from time zero. We hope to utilize pre-embedding with BERT, active learning, and discriminator checking mechanism to gain high performance in our model.

B. Evaluation Method

For most traditional theoretical evaluation methods for music generation models, such as computing the proportion of empty bars, they are not good at assessing the models' true creativity. Their results are usually about whether the musics conform to certain music theory or not, i.e. a binary classification problem. Another type of method, such as surveys and user studies, are often too subjective. [19] [20] Thus, we decide to use a method inspired by unsupervised learning. In detail, we input two texts with similar meanings, such as "I am happy" and "I feel good" into the model, hope that the generator's output can pass the discriminator checking mechanism, and the two output music sequences are as similar as possible. In comparison, we also input texts with opposite meanings, such as "She laughs" and "He cries" into the model, and hope that the two output music sequences are as different as possible to prove our model can truly describe different texts with different musics. To assess the difference between output sequences, we use the method Kullback–Leibler divergence (KL divergence), which intends to calculate the relative entropy between two probability distributions:

$$D_{KL}(P \,||\, Q) = \sum_{x \in \chi} P(x) \log \left(\frac{Q(x)}{P(x)} \right) \tag{8}$$

where P and Q denotes two probability distributions defined on the same probability space, χ. [21]

C. Results

For our results, Figure 5 demonstrates the pianorolls of 4 generated sequences of APE-GAN. Through analysis of the pianorolls, we can find that the sequences are valid musics, because all the samples illustrated have clear melodic lines, distinctions between voice parts, reasonable temporal structures, and variations in note lengths. They do not have excess empty bars or notes that deviate too far from the bulk of the melody.

Also, as illustrated in Figure 6 (where darker color means higher KL divergence), with textual inputs of similar meanings, for instance, "I am happy" and "I feel good", the KL divergence is near 0. In contrast, with textual inputs of different meanings, for instance, "She laughs" and "He cries", the KL divergence is near 1.

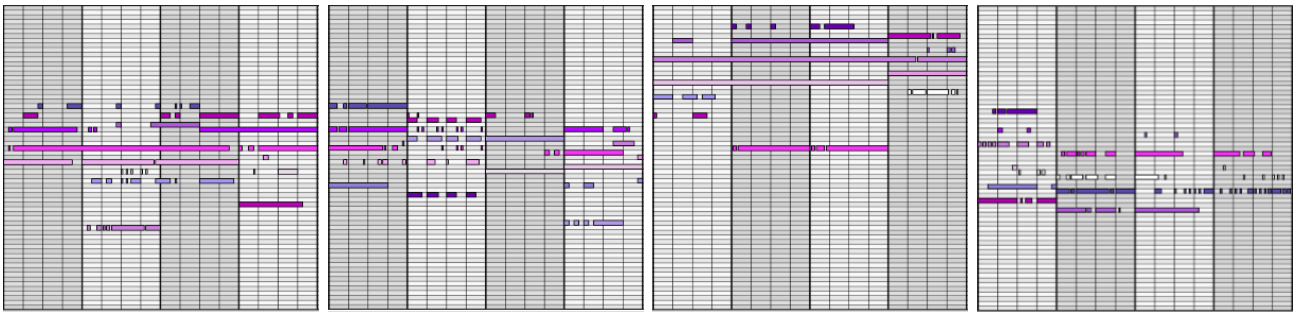

Fig. 5. Pianorolls of Generated Music

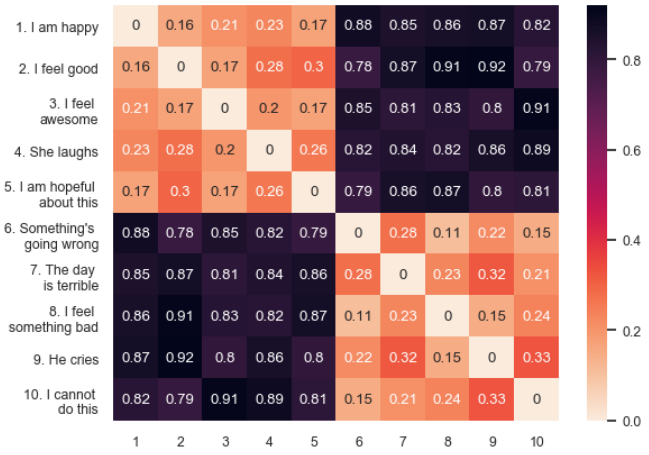

Fig. 6. KL Divergence Heatmap

V. Conclusion

Music generation models are generating increasingly realistic musics due to all those ingenious models proposed in recent years. However, the musics generated do not possess descriptive power and are not as realistic as the "true musics" written by humans. In this paper, we proposed a model named "APE-GAN" that can improve the performance of music generation models in three ways. First, we use BERT to produce word vectors as inputs for our model so that it can obtain musical descriptive ability. Second, we utilize active learning to decrease the required labels amount to 5% to 10%, which results in higher efficiency in training our model. Lastly, we implement the trained discriminator as a checking mechanism to let our model generate more realistic musics. To evaluate APE-GAN more objectively, we measure APE-GAN's descriptive ability by comparing the music sequences generated by different textual inputs with the metric of KL divergence. With pre-embedding, for similar meanings inputs, the resultant KL divergence reaches nearly 0, while for different meaning inputs, the KL divergence reaches nearly 1. Therefore, APE-GAN is able to learn semantic prior knowledge efficiently, which allows it to generate more realistic and creative musics.

VI. Acknoledgement

This work is under the support of the Mars Plan Program.

References

[1] Bayer, J., Osendorfer, C., Korhammer, D., Chen, N., Urban, S. and van der Smagt, P., 2013. On fast dropout and its applicability to recurrent networks. *arXiv preprint arXiv:1311.0701*.

[2] Goel, K., Vohra, R. and Sahoo, J.K., 2014, September. Polyphonic music generation by modeling temporal dependencies using a rnn-dbn. In *International Conference on Artificial Neural Networks* (pp. 217-224). Springer, Cham.

[3] Huang, A. and Wu, R., 2016. Deep learning for music. *arXiv preprint arXiv:1606.04930*.

[4] Mogren, O., 2016. C-RNN-GAN: Continuous recurrent neural networks with adversarial training. *arXiv preprint arXiv:1611.09904*.

[5] Dong, H.W., Hsiao, W.Y., Yang, L.C. and Yang, Y.H., 2018, April. Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 32, No. 1).

[6] Liang, X., Wu, J. and Yin, Y., 2019. MIDI-Sandwich: Multi-model Multi-task Hierarchical Conditional VAE-GAN networks for Symbolic Single-track Music Generation. *arXiv preprint arXiv:1907.01607*.

[7] Dong, H.W. and Yang, Y.H., 2018. Convolutional generative adversarial networks with binary neurons for polyphonic music generation. *arXiv preprint arXiv:1804.09399*.

[8] Malekzadeh, S., Samami, M., RezazadehAzar, S. and Rayegan, M., 2019. Classical Music Generation in Distinct Dastgahs with AlimNet ACGAN. *arXiv preprint arXiv:1901.04696*.

[9] Tokui, N., 2020. Can GAN originate new electronic dance music genres?–Generating novel rhythm patterns using GAN with Genre Ambiguity Loss. *arXiv preprint arXiv:2011.13062*.

[10] Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*.

[11] Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B. and Bharath, A.A., 2018. Generative adversarial networks: An overview. *IEEE Signal Processing Magazine, 35(1), pp.53-65*.

[12] Goodfellow, I., 2016. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*.

[13] 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*.

[14] Li, Z., Cai, J., He, S. and Zhao, H., 2018, August. Seq2seq dependency parsing. *Proceedings of the 27th International Conference on Computational Linguistics* (pp. 3203-3214).

[15] Sherstinsky, A., 2020. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D: Nonlinear Phenomena, 404*, p.132306.

[16] Staudemeyer, R.C. and Morris, E.R., 2019. Understanding LSTM–a tutorial into Long Short-Term Memory Recurrent Neural Networks. *arXiv preprint arXiv:1909.09586*.

[17] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L. and Polosukhin, I., 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

[18] Hino, H., 2020. Active Learning: Problem Settings and Recent Developments. *arXiv preprint arXiv:2012.04225*.
[19] Mao, H.H., Shin, T. and Cottrell, G., 2018, January. DeepJ: Style-specific music generation. *2018 IEEE 12th International Conference on Semantic Computing (ICSC) (pp. 377-382)*. IEEE.
[20] Chu, H., Urtasun, R. and Fidler, S., 2016. Song from PI: A musically plausible network for pop music generation. *arXiv preprint arXiv:1611.03477*.
[21] M. Vidyasagar, "Kullback-Leibler divergence rate between probability distributions on sets of different cardinalities," *49th IEEE Conference on Decision and Control (CDC), Atlanta*, GA, 2010, pp. 948-953, doi: 10.1109/CDC.2010.5716982.