Tree-Structure LSTM Model for Structured Jazz Solo Generation

Peizhe Gao[1], Yujia Yan[2], and Zhiyao Duan[3]

[1] Audio Information Research Lab, University of Rochester,
peizhegao321@gmail.com
[2] yyan22@ur.rochester.edu
[3] zhiyao.duan@rochester.edu

We introduce a novel tree-structured architecture aimed at modeling music pieces resembling jazz solo (lead sheet) with long-term structure using Recurrent Neural Networks (RNN). The model is based on an attention-enhanced encoder-decoder architecture [1]; the encoder is a Long Short Term Memory (LSTM) RNNs [2], and the decoder is an up-down three-level tree LSTM RNNs. We encode seed chords into vector representations, and generate section, chord and note symbols conditioned on the encoding vectors. The decoder adds a two-level restriction to the generated melody (with generated chord progression conditioned on section symbols and generated notes conditioned on generated chords). In this way, our model can generate a melody and accompanying chords with a specific structure.

Music generation has been a challenging task over the last decades. In recent years, RNN-based approaches have been applied to this task to produce various interesting results. [3] [4] [5] Many of these generated musical pieces, however, lack a consistent theme or structure. Such sequences can appear wandering and random. [6] Human music compositions adhere to relatively well-defined structural rules, making music an interesting sequence generation challenge. Here, we aim to explicitly model the section-chord-note hierarchy in music, generating structure-meaningful result.

We use a relatively concise method to represent data. Section symbols include the starting and ending symbol of each section, as well as section name symbols (like A1:, B1:, A2:etc). Chord symbols include major, minor, augmented and diminished triads; major, minor, dominant, diminished, half-diminished and augmented seventh chords, as well as major sixth chords and minor sixth chords, adding up to 144 kinds. We build hash tables to map these symbols with integer numbers. For notes, we separately represent pitch and duration, abandoning time-step rhythm representations. We choose MIDI numbers to encode pitches, while build another hash table to map categorized duration beats with integer numbers. The integers are then converted to one-hot vectors and finally embedded vectors after the embedding layer.

We choose a one-layer LSTM with 256-dimensional hidden/cell state as our encoder. The input is set to be a sequence of seed symbols, consisting of the section name symbols and the first few chord symbols in each section. After the embedding layer, the seed scalars are turned into 10-dimensional embedding vectors are fed into encoder LSTM sequentially, with hidden and cell states

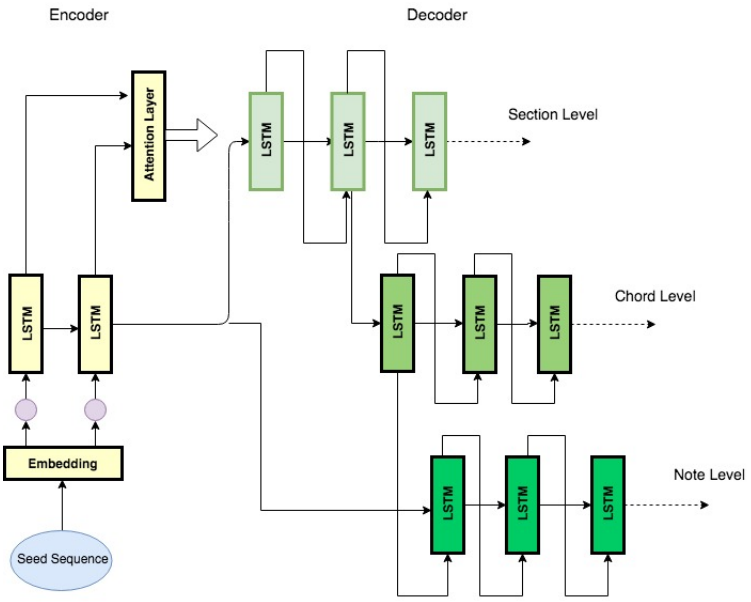

Fig. 1. Graphical representations of our model. Embedding layers, fully-connected layers and Softmax function of the decoder part are omitted. The output of attention layer is fed into all decoder LSTMs.

passed on. The last hidden and cell state is then used as the initial ones for the decoder.

In the hierarchical tree decoder, three layers from top to the bottom separately generate section, chord and note symbol, where we build a parent-feeding connection between adjacent layers [7]. When the section layer decoder generates section name symbols, its hidden state and cell state are fed into the chord layer decoder. And whenever the chord decoder generates a chord symbol, it is concatenated with the corresponding note symbol input and fed into the note decoder after embedding. We calculate accumulating note beats to match chords and their corresponding notes. All scalar symbols are fed into an embedding layer and an attention layer before the LSTM, as well as a fully-connected layer and Softmax function after the LSTM. In the note layer, embedded pitch symbol, duration symbol and chord symbol are concatenated as the input to the LSTM. When generating notes, we first generate pitch using the output of the LSTM, then generate duration conditioned on pitch.

We implemented out model using PyTorch [8]. We trained it on 341 jazz solo transcriptions from Weimar Jazz Dataset [9], with a 1e-4 learning rate. The system was trained for 500 epochs using the ADAM optimizer with Cross Entropy Loss.

Music scores are reconstructed with the help of music21 toolkit. [10] A demo page of our system is available online https://bbblues.github.io/.

Now we are performing listening test evaluation comparing out model with other baseline systems. Hopefully we will finish experiments and some statistical analyses soon.

References

1. Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In Proceedings of the ICLR, San Diego, California.
2. Sepp Hochreiter; Jrgen Schmidhuber. 1997. "Long short-term memory". Neural Computation. 9 (8): 17351780.
3. Gaetan Hadjeres and Francois Pachet. 2016. DeepBach: a Steerable Model for Bach chorales generation. arXiv preprint arXiv:1612.01010 (2016).
4. Romain Sabathe, Eduardo Coutinho, and Bjorn Schuller. 2017. Deep recurrent music writer: Memory-enhanced variational autoencoder-based musical score composition and an objective measure. In Neural Networks (IJCNN), 2017 Inter- national Joint Conference on. IEEE, 34673474.
5. Hao-Wen Dong, Wen-Yi Hsiao, Li-Chia Yang, and Yi- Hsuan Yang. Musegan: Multi-track sequential gener- ative adversarial networks for symbolic music genera- tion and accompaniment. In AAAI-18 AAAI Conference on Artificial Intelligence, 2018.
6. Hang Chu, Raquel Urtasun, and Sanja Fidler. 2017. 20 Aug. Song from PI: A musically plausible network for pop music generation. ICLR Workshop, 2017.
7. Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In Association for Computational Linguistics (ACL), 2016
8. PyTorch Homepage, https://pytorch.org/
9. Pfleiderer, Martin and Frieler, Klaus and Abeßer, Jakob and Zaddach, Wolf-Georg and Burkhart, Benjamin: Inside the Jazzomat - New Perspectives for Jazz Research. Schott Campus, 2017
10. Michael Scott Cuthbert and Christopher Ariza. mu- sic21: A toolkit for computer-aided musicology and symbolic music data.