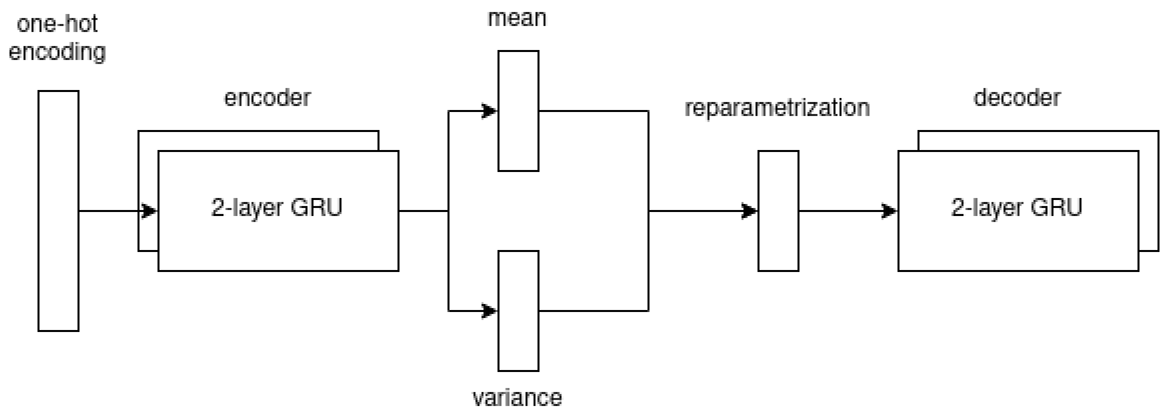
The Ai Music Generation Challenge 2022 - Technical Document

Marco Amerotti

folktune-VAE: a VAE for Irish Traditional music

folktune-VAE is a Variational Autoencoder developed as part of the author's bachelor thesis "Latent Representations For Traditional Music Analysis and Generation". The primary goal of this work was to investigate the musical properties of the learned latent space, correlating them with the explicit knowledge and understanding we have of music in general; in addition, some generative results are evaluated. It seemed appropriate to adapt this model for this challenge.

Technical details



The architecture of folktune-VAE.

The model consists of a two-layer GRU encoder and a two-layer GRU decoder.

Each tune in ABC notation is tokenized and transformed into a one-hot encoded sequence; we only consider sequences than 256 tokens long. This sequence is then fed through the encoder and the resulting vector is used to obtain the mean and variance for that tune; we then sample a latent vector using said mean and variance and use that to decode back the original sequence. We also introduce an embedding dropout of 0.2.

Our model employs a 32 dimensional latent space and a hidden size of 256 for the encoder and decoder.

An extensive explanation of the model and implementation choices can be found in the original work.

Training

The model has been trained for 450 epochs on the same dataset used to train folkrnn (which counts more than 20.000 tunes in ABC notation).

We then fine-tuned it using the 350 O'Neill's reels for 100 additional epochs.

During training we use a learning rate of `0.001` and annealing of the KL-Divergence.

Sampling

We sample using top-p sampling with `p = 0.9` and `temperature = 1`. The only reason behind this choice is experimental results obtained by playing around with the model.

The model generates tune in the key of C and in one of the following modes: Major, Minor, Dorian or Mixolydian. To generate tunes in different keys, we use the following procedure:

- we count for each mode how many reels in the O'Neill collection have a certain key;
- for each generated reel, we choose a random key between those allowed by its mode using said counts as weights;
- we transpose the reel accordingly.

To generate the 1000 tunes, we seeded the model with `0` and generated 1100 tunes. We then processed them as explained above and kept the first 1000. The need to generate more tunes lies in the occasional not well-formed reels that the model produces; in this way, we ensure to have enough reels to discard a few.