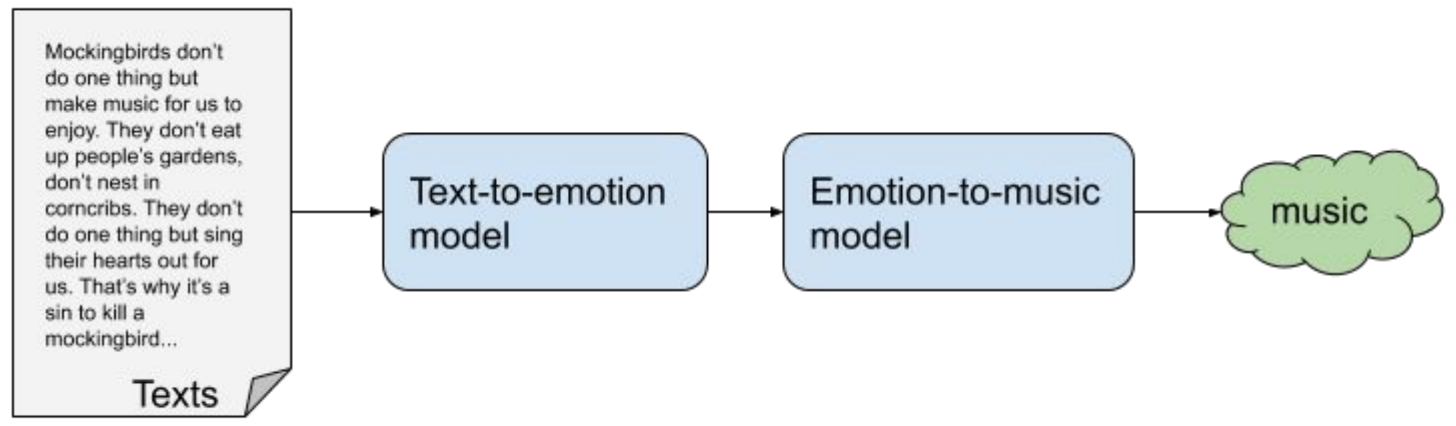
Literary Muzak

Chuan He, Liang Ping Koh, Qiyin Wu

1. Problem Statement and overall approach

- Text to music involves two difficult sub problems: text to emotion and emotion to music.
- Text to emotion is difficult due to the complexities of language and the ambiguity of emotional labels.
- Emotion to music is difficult because we are mapping low dimensional input to complex and highly patterned output.
- We use transformers to translate text to emotional labels, and then use GAN to train a generator to generate a music based on an emotional label:

End-to-end workflow


Mockingbirds don't do one thing but make music for us to enjoy. They don't eat up people's gardens, don't nest in corncribs. They don't do one thing but sing their hearts out for us. That's why it's a sin to kill a mockingbird...
Texts
Text-to-emotion model
Emotion-to-music model
music

2. Dataset and Pre-processing

- We use the SemEval2018 dataset of 6857 tweets with binary labels for eight emotional classes for our text to emotion model.
- We use 307 pokemon soundtracks, 92 final fantasy soundtracks, 88 pop songs and 200 emotionally labeled MIDI files which we process into sequences of notes for our emotion to music model.

3. Text to Emotion method

- We tested the Transformer and multiple LSTM models against each other in emotional classification of tweets
- The following is an illustration of a transformer model:

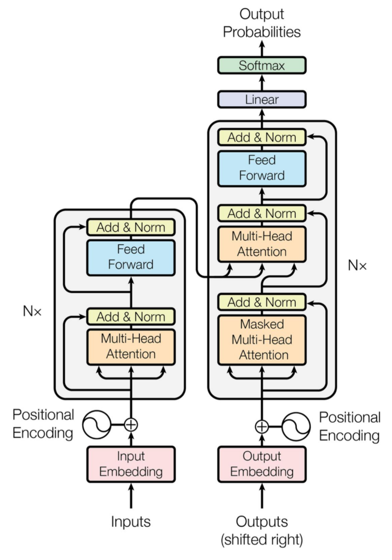


Figure 1: The Transformer - model architecture.

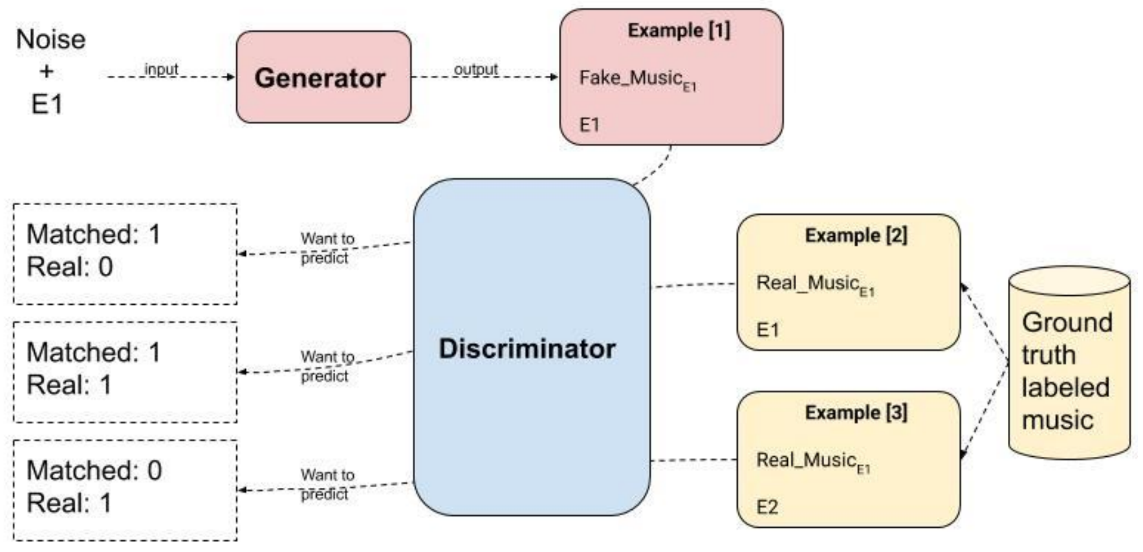
- We were able to replicate state of the art results, and the transformer beat other models on emotional classification:

Table 8: F1-score of finetuned language models on SemEval plutchik classification, with IBM Watson as a baseline.

		Anger	Anticipation	Disgust	Fear	Joy	Sadness	Surprise	Trust	**Average**
Semeval	Transformer	**.771**	**.403**	**.764**	**.765**	**.818**	**.691**	**.400**	**.271**	**.610**
	mLSTM	.548	.275	.576	.319	.651	.491	.122	.168	.394
	ELMo	.614	.294	.662	.388	.734	.531	.154	.181	.445
	Watson	.498	-	.331	.149	.684	.359	-	-	-

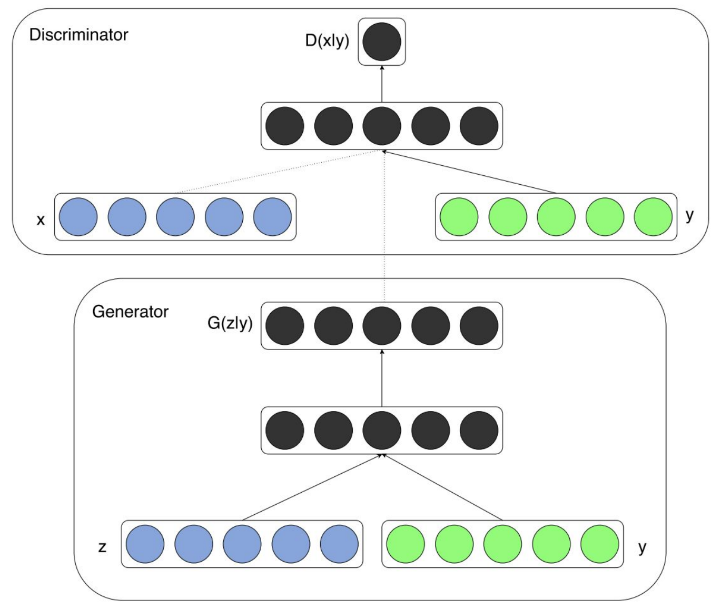
4. Emotion to Music Method

Emotion-styled music GAN



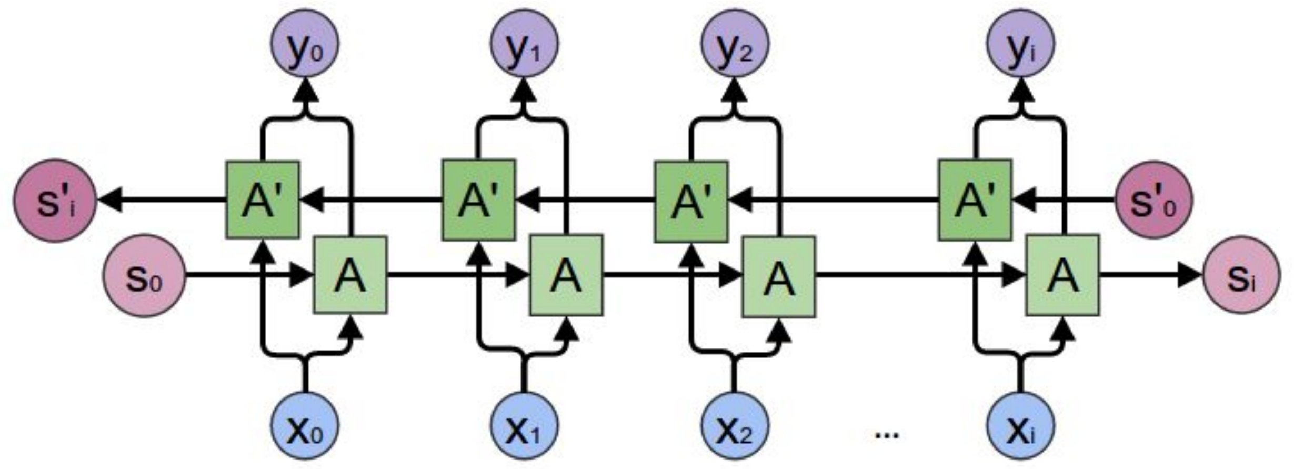
Notation:
- E1: emotion 1
- Fake_Music$_{E1}$: fake music with emotion 1

- We use a GAN to train a generator to generate music from an emotional label and a noise of fixed dimension.
- We want output to both sound like real music but also have an emotional quality; thus, our output is conditioned on emotional label; this is a conditional GAN:

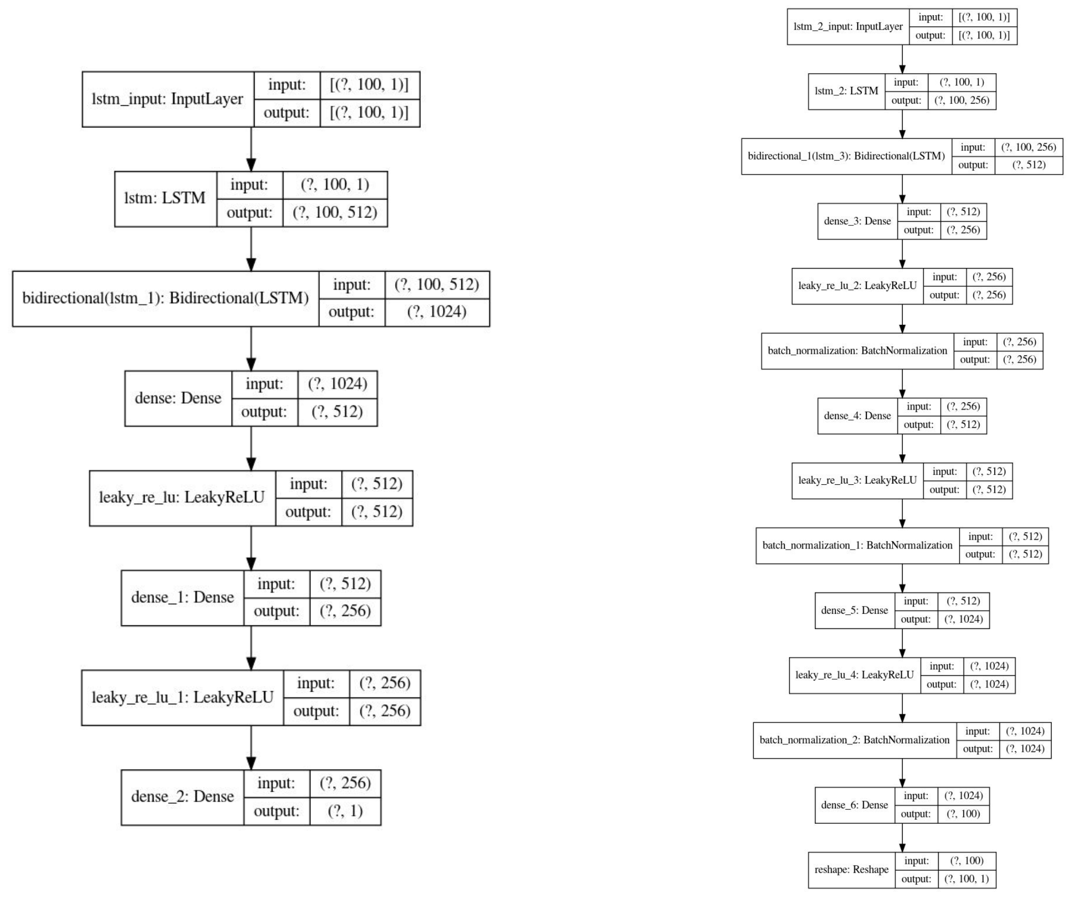


- In a CGAN, conditional labels are embedded into inputs as representations. In our model, we concatenate sequences repeating the emotional classes to our note sequences. We experimented with several lengths.
- However, training a GAN with only 200 emotionally labelled music examples, there was not enough data to learn the natural music representation. We thus undertook transfer learning. You can hear the bad soundtrack **(Music Demo).**

- We use 307 Pokemon soundtracks, 88 pop songs and 92 final fantasy soundtracks to train our generator to generate realistic sounding music, without emotional labels, and then transfer these weights and retrain on our 200 emotionally labelled examples.
- We used a bidirectional LSTM architecture for the generator and discriminator. This allows the output to obtain information from both past and forward states simultaneously.



- This is our architecture:


lstm_input: InputLayer | input: [(?, 100, 1)] | output: [(?, 100, 1)]
lstm: LSTM | input: (?, 100, 1) | output: (?, 100, 512)
bidirectional(lstm_1): Bidirectional(LSTM) | input: (?, 100, 512) | output: (?, 1024)
dense: Dense | input: (?, 1024) | output: (?, 512)
leaky_re_lu: LeakyReLU | input: (?, 512) | output: (?, 512)
dense_1: Dense | input: (?, 512) | output: (?, 256)
leaky_re_lu_1: LeakyReLU | input: (?, 256) | output: (?, 256)
dense_2: Dense | input: (?, 256) | output: (?, 1)
lstm_2_input: InputLayer | input: [(?, 100, 1)] | output: [(?, 100, 1)]
lstm_2: LSTM | input: (?, 100, 1) | output: (?, 100, 256)
bidirectional_1(lstm_3): Bidirectional(LSTM) | input: (?, 100, 256) | output: (?, 512)
dense_3: Dense | input: (?, 512) | output: (?, 256)
leaky_re_lu_2: LeakyReLU | input: (?, 256) | output: (?, 256)
batch_normalization: BatchNormalization | input: (?, 256) | output: (?, 256)
dense_4: Dense | input: (?, 256) | output: (?, 512)
leaky_re_lu_3: LeakyReLU | input: (?, 512) | output: (?, 512)
batch_normalization_1: BatchNormalization | input: (?, 512) | output: (?, 512)
dense_5: Dense | input: (?, 512) | output: (?, 1024)
leaky_re_lu_4: LeakyReLU | input: (?, 1024) | output: (?, 1024)
batch_normalization_2: BatchNormalization | input: (?, 1024) | output: (?, 1024)
dense_6: Dense | input: (?, 1024) | output: (?, 100)
reshape: Reshape | input: (?, 100) | output: (?, 100, 1)

- The combination of tuning our embedding style, transfer learning and bidirectional LSTMs allowed us to produce more natural sounding music, with subtle differences between our different emotion classes **(Music Demo)**

Conclusion

- Two key takeaways:
  - Bidirectional LSTM, transfer learning and GANs with bidirectional RNNs can be successful even with shallow networks and small data
- Next steps (For final report):
  - Rather than manually tuning the embeddings, we can learn an embedding of emotion into musical input. We can also try deeper networks and bigger more diverse datasets.