Week 10 Report

Kieran Grant - 2357351G

What I've done this week

- Summarised *β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework (2016).*
  - Short summary and takeaways below.
- Began writing up skeleton of interim report with bullet points of content.
  - Will attach rough work so far as separate file.
- Started re-implementing Style Transfer code in PyTorch (Lightning).
  - Managed to load in the pre-trained weights of the Style Transfer spectrogram encoder (EfficientNet) into my implementation and verified that they give the same encoding.
- I've been trying to get Pedalboard to work with external VSTs. However, I've ran into a number of issues:
  - Library looks like it only works for VST3 (and AU on Mac) plugins, not VST2 or LV2.
  - Have tried a few different contenders of open-source suites of effects (GVST, MDA, SAFE etc), however they do not load into Pedalboard.
  - Also tried `cython-vst-loader` which was able to load in the GVST plugins, but not apply the effect to audio.
  - Possible next steps:
    - Pedalboard itself comes with a number of built-in effects (written in JUCE behind-the-scenes) - these may be much more stable and easy to use and avoids the problem of having to load in different plugin formats.
    - The Black-Box DDSP code has a library for loading in LV2 plugins - however it's very complicated. I'm not sure if it's worth the additional time to re-implement that as well.
    - I have been able to load the SAFE plugins into Pedalboard on Mac - however my MacBook is ~10 years old and might struggle with ML training.
- Read and summarised *A Feature Learning Siamese Model for Intelligent Control of the Dynamic Range Compressor (2019).*
  - Short summary and takeaways below.

Questions

- I would be interested to hear your thoughts on the issue with the Pedalboard library above, i.e. is it worth going with the easier route of using the built-in Pedalboard effects, or trying to alter the BB-DDSP LV2 loader?
- Any feedback on the skeleton of the interim report would be appreciated!

Plan for next week

- Make decision on effects to use (Pedalboard or look at different loaders).
- Continue implementation of model.
- Implement SPSA custom gradient in PyTorch for some toy examples.
- First draft of interim report complete and sent over.

Current state of project

- Still looking on track to have basic architecture complete by Christmas break.
- Made a start on interim report, it will be useful to give myself a deadline for first draft for next week - can then spend last two weeks making any changes.

β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework (2016)

Short Summary

The main idea of this paper is to modify the variational autoencoder (VAE) architecture in order to learn a disentangled latent representation of data. Disentanglement means that changes in one latent unit are sensitive to changes in one generative factor (rotation, scaling etc) while being invariant to other factors (e.g. the image doesn't rotate, scale and distort as one latent factor is changed).

Prior attempts at disentanglement have required prior knowledge about the structure of the data, or didn't scale well to unseen data. InfoGAN [1] had come closest to a scalable, unsupervised approach however it was unstable during training and was sensitive to the choice of prior distribution. Finally, there was no quantifiable measure of disentanglement which could be used to optimise the hyperparameters of the model.

The model proposed in this paper introduces a single hyperparameter, β, which scales the KL-divergence. It is shown that tuning this hyperparameter ($\beta \gg 1$) can improve disentanglement. This can be understood as constraining the model to learn the most efficient, and thus disentangled, representation of the data. The authors mention that disentanglement is not exactly the same as the factors being independent, since such a representation may not be interpretable.

To quantify the disentanglement a new metric is proposed, however it requires at least weakly labelled information. This metric uses a simple linear classifier (to account for interpretability). The idea is to have a classifier for a generative feature, say rotation, and fix the latent representation associated with that feature. All other factors are randomised and the classifier applied. In this way the aim is to minimise the amount of variance amongst the other (non-relevant) factors by improving the accuracy of the classifier.

The model is compared against vanilla VAE, InfoGAN and DC-IGN methods for disentangled factor learning using both qualitative (figures representing the change in generated data) and quantitative (using the proposed metric) benchmarks. In the latter case a synthetic dataset is generated using four independent generative factors (x position, y position, scale and rotations). In both cases, the proposed model learns more clear disentangled latent representations. In general it was found that a larger latent representation required a larger β value for efficient disentanglement.

Takeaways

- I had a much better idea of the meaning of disentanglement after reading this paper. I previously thought that disentanglement was just finding a linear independent basis, but did not account for the interoperability aspect.
- I am surprised that this is achieved by only introducing a single hyperparameter scaler, however I think that this has a benefit in being easy to implement into an existing VAE.
- The quantitative measures seem reliant on having labelled data, and otherwise means using heuristics (such as looking at the generated images). This may be more difficult with something like parameter settings.

References

[1] Chen, Xi, et al. "Infogan: Interpretable representation learning by information maximizing generative adversarial nets." Advances in neural information processing systems 29 (2016).

A Feature Learning Siamese Model for Intelligent Control of the Dynamic Range Compressor (2019)

Short Summary

In this paper the authors present a method for controlling the ratio, threshold, attack and release times of a dynamic range compressor (DRC). Three different DNNs are proposed for feature extraction from an input and reference track: the first uses CNNs on the spectrograms obtained from the audio sources, the second performs 1D-convolution on the audio waveform directly and the third uses a multi-kernel approach where different convolutions are applied to a spectrogram representation and combined. The (50-D vector) representations of the two audio signals are combined with subtraction. For predicting the parameter settings a Random Forest regression model is trained on the learned representations.

The datasets consist of 64 guitar and 64 drum loops where different compression settings are applied. The neural network is first trained end-to-end, then the final dense layer is removed to leave the vector representation as the output of the network. After some hyperparameter tuning, the CNN model (using 'regular' spectrograms with a short time-frame) achieved the best MAE score on the holdout set and performed better than using hand-crafted features. The trained feature embeddings are also shown to perform better on unseen polyphonic music compared to the rule-based features.

Takeaways

- In general I thought this paper wasn't very well written, there were a number of grammatical errors and some things were unclear. The authors also made some very general statements (e.g. saying Mel-spectrograms are worse than regular spectrograms) which may not apply out-with the scope of the paper.
- I don't see much practical use for the model as it was trained using one-to-one examples. That is an audio source and a compressed version of the same audio source. The authors don't make any attempt to show how the model performs when the reference isn't matched.
- I think it is useful that the authors compared spectrogram and waveform models for feature embedding.
- I think there could have been more work in looking at how the two feature vectors could be combined (e.g. concatenation, convolution etc.).