DL LAB 2: An application with RNN

Senne Deproost and Diego Saby

November 2019

1 Introduction

In the field of machine learning we try create computational models to find solutions for a wide set of tasks. One of the more popular problems to solve are the classification of instances and function approximation with regression. Solutions like *Support Vector Machines* and *Decision Trees* have already proven their worth in the past. Techniques from the deep learning (DL) sub-field of machine learning have shown their usability when tackling on data with high dimensionality. Deep learning's ability to scale better with these kinds of data makes it useful in many application domains like machine control and decision support systems. Variations in the model's architecture could insure better performance on a domain specific task.

One DL architecture we'll focus on in this lab report is the *Recurrent Neural Network* architecture. Whereas classical *Feedforward Neural Network* (FNN) only allows the passage of activation through the neuron layers, a RNN network will remember the neuron activations from a previous step in time. This is implemented with the inclusion of memory nodes that will hold on to previous encountered activation values. An advantage of this kind of networks is the usage of *Temporal Sequences*, allowing the model to train on data like text sentences, music recordings and streams of data.

Automated content creation using DL models has known a significant rise in popularity over the past years. With *Generative Adverserial Networks* (GAN's), we can use two competing networks to train a generator for images (Goodfellow et al., 2014). In this architecture, one network, the has to mimic given image data input distribution and another adversarial network that has to estimate the loss of the predicted regression with given training data. This prediction can then be used to optimize the generative network. Other types of models like *transformers*, recently used in OpenAI's GPT-2 model (Radford et al., 2018), can generate human-like text paragraphs with the potential of passing the famous Turing test. With a rise in the exploration of the generative content field, researches already used RNN based models to generate content like images (Gregor, Danihelka, Graves, Rezende, & Wierstra, 2015), voice synthesis (van den Oord et al., 2016; Van Den Oord, Kalchbrenner, & Kavukcuoglu, 2016) and music (van den Oord et al., 2016).

1.1 Problem

In this report, we will focus on three types of models in the task of creating music from MIDI files. The data will consists of fragments of Chopin's Nocturne music composition in the MIDI file format with 30 available notes available on the tone ladder. The music consists of only one music instrument: the piano.

Apart from regular RNNs, we will experiment on *Long Short-Term Memory* (LSTM)(Hochreiter & Schmidhuber, 1997) and *Gated Recurrent Unit* (GRU) models (Cho, van Merrienboer, Bahdanau, & Bengio, 2015).

1.2 LSTM

An LSTM is a form of RNN which solves the problem of the large decay in loss function gradient over long periods of time. Specialized memory cells are used to recall on several past activations for longer periods of time. In addition, the network contains gates to control the information flow to and from memory and when it can be returned to its initial status.

1.3 GRU

GRU are similar in construction to LSTM that they have gates to control the specialized memory cells. However, these models are simplified containing less gates and dont make the distinction between memory cells and normal ones (Chung, Gulcehre, Cho, & Bengio, 2014).

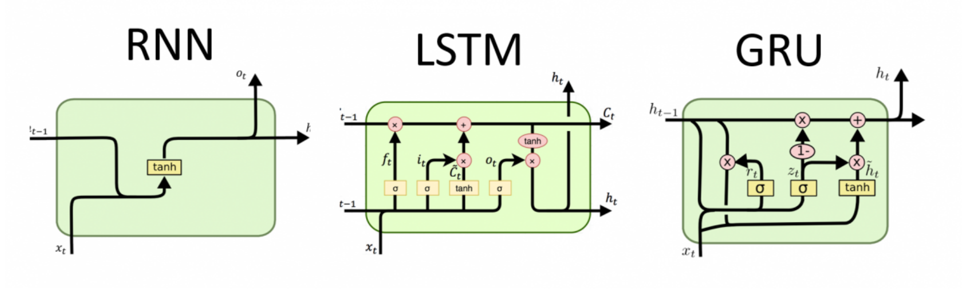

Figure 1: An overview of the three kinds of models used in the experiments. Left we have a simple RNN node containing a simple *tanh* activation function. In the middle the internal gates of the LSTM node are shown. The right contains the simplified layout of a GRU network node.

2 Preprocess the data

One of the first questions we were opposed to was how to encode music in order to train the Neural Network. Our source files are midi files. They are an useful way to encode the music where we can extract the notes and their duration from the file.
To simplify the problem we decided to abstract the notes duration and just use the notes pitch.

2.1 Enumerating the notes

Our first attempt was done inspired by some code found on the internet to produce video games music. **add reference**
To extract the notes the notes from midi file we used a library called *Music21*. It provided us with the notes names or chord names that were played. From there we have a sequence of notes names. Then we just assign the notes a number. For the prediction we used a vector of all the notes and a with a 0 for the next note that follows the sequence.

3 Experiments

In the experiments, we mainly focus on the three different kind of models as described before. We first vary the numbers of node in the hidden layers of the network. Afterwards, we change the number of layers in the network.

We train with a fixed batch size of 64 during a 200 epoch session. The results will be discussed based upon the loss function of the model and the qualitative analysis of the generated music fragments.

3.1 Vanilla RNN

3.2 LSTM

3.3 GRU

Last we run experiments with the GRU variant of RNN's.

3.3.1 Number of nodes

3.3.2 Number of layers

4 Conclusion

References

Cho, K., van Merrienboer, B., Bahdanau, D., & Bengio, Y. (2015). On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. In *Proceedings of ssst-8, eighth workshop on syntax, semantics and structure in statistical translation* (pp. 103–111). doi: 10.3115/v1/w14-4012

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. Retrieved from `http://arxiv.org/abs/1412.3555`

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). Generative Adversarial Nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems 27* (pp. 2672–2680). Curran Associates, Inc. Retrieved from `http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf`

Gregor, K., Danihelka, I., Graves, A., Rezende, D. J., & Wierstra, D. (2015). DRAW: A recurrent neural network for image generation. In *32nd international conference on machine learning, icml 2015* (Vol. 2, pp. 1462–1471).

Hochreiter, S., & Schmidhuber, J. (1997, nov). Long Short-Term Memory. *Neural Comput.*, *9*(8), 1735–1780. Retrieved from `http://dx.doi`

.org/10.1162/neco.1997.9.8.1735 doi: 10.1162/neco.1997.9.8.1735

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2018). Language Models are Unsupervised Multitask Learners.

Van Den Oord, A., Kalchbrenner, N., & Kavukcuoglu, K. (2016). Pixel recurrent neural networks. In *33rd international conference on machine learning, icml 2016* (Vol. 4, pp. 2611–2620).

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., … Kavukcuoglu, K. (2016). *WaveNet: A Generative Model for Raw Audio* (Tech. Rep.). Retrieved from http://arxiv.org/abs/1609.03499