Recurrent Neural Networks with Attention for Genre Classification

Jeremy Irvin, Elliott Chartock, Nadav Hollander
jirvin16@stanford.edu, elboy@stanford.edu, nadavh@stanford.edu

Motivation

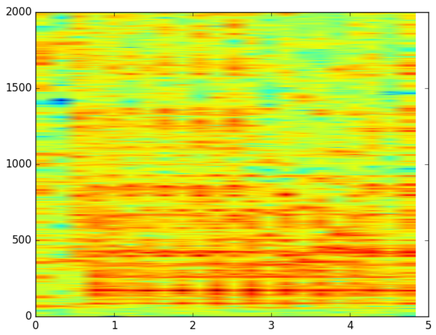
Classifying musical genre from raw audio is a problem that captures unique challenges in temporally oriented, unstructured data. Recent advances in recurrent neural networks present an opportunity to approach the traditionally difficult genre-classification problem using bleeding edge techniques [2,3,4].

Problem Definition

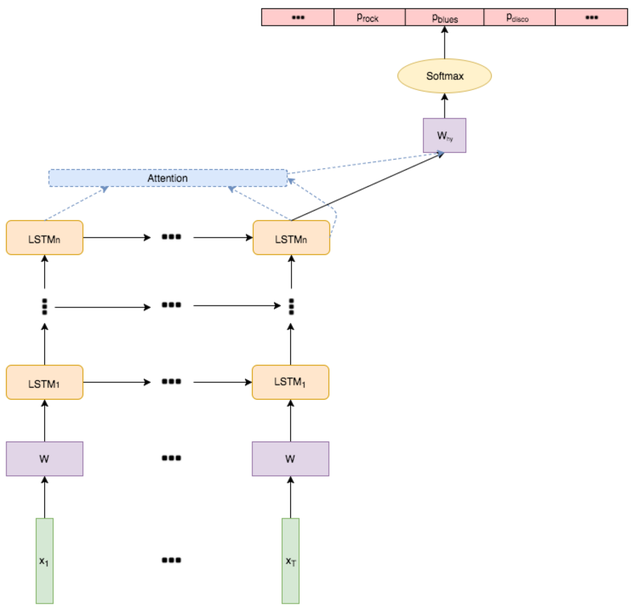
Given a variable length series of feature vectors $x_1, ..., x_T$, with each vector representing a time-step of a given song-clip, we aim to predict a class $\hat{y}$ from a set of labels $c_1, ..., c_C$, each representing a song genre.

Data

- Using the GTZAN Genre Collection [1], we start with a set of 1000 30 second song excerpts subdivided into 10 pre-classified genres: Blues, Classical, Country, Disco, Hip-Hop, Jazz, Metal, Pop, Reggae, and Rock.
- We downsampled to 4000 Hz, and further split each excerpt into 5-second clips
- For each clip, we compute a spectrogram using Fast Fourier Transforms, giving us 22 timestep vectors of dimensionality 513 for each clip.
- Spectrograms separate out component audio signals at different frequencies from a raw audio signal, and provide us with a tractable, loosely structured feature set for any given audio clip that is well-suited for deep learning techniques. (See, for example, the spectrogram produced by a jazz excerpt below)



Models



- Recurrent Neural Network (RNN)
  - Given a sequence of vectors $x_1, ..., x_T$, a Vanilla RNN with L layers computes, at a single timestep t and layer l,

$$h_t^1 = Wx_t$$

$$h_t^l = \tanh(W_{hh}^l h_{t-1}^l + W_{xh}^l h_t^{l-1})$$

$$y_t = W_{hy} h_t^L$$

where h_t^l is the hidden vector at the tth timestep and layer l, and $W, W_{hh}, W_{xh}, W_{hy}$ are parameters to the model. In our classification setting (where the output sequence is a single class), we compute

$$\hat{y} = \arg\max_{c \in C} (\tilde{y})_c$$

where $(\cdot)_c$ denotes the cth index operator, and $\tilde{y} = \text{softmax}(y_T^L)$. All vectors y_t^L are not computed except for y_T^L.

- Long Short Term Memory Network (LSTM)
  - Given a sequence of vectors $x_1, ..., x_T$, a Vanilla LSTM with L layers computes, at a single timestep t and layer l,

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \tanh \end{pmatrix} W^l \begin{pmatrix} h_t^{l-1} \\ h_{t-1}^l \end{pmatrix}$$

$$c_t^l = f \odot c_{t-1}^l + i \odot g, \quad h_t^l = o \odot \tanh(c_t^l)$$

$$y_t = W_{hy} h_t^L$$

where $\odot$ indicates element-wise multiplication, i, f, o, g, c_t^l are the H dimensional input gate, forget gate, output gate, block gate, and context vector respectively at the tth timestep and lth layer. We compute $\tilde{y}$ and make predictions as above.

- Soft Attention Mechanism [4]
  - A soft attention mechanism is an additional layer added to the LSTM defined as follows

$$\alpha_t = \frac{\exp(c_T^L \cdot h_t^L)}{\sum_{t'} \exp(c_T^L \cdot h_{t'}^L)}$$

$$c = \sum_{t=1}^{T} \alpha_t h_t^L, \quad \tilde{h} = \tanh(W_c[c; c_T^L])$$

$$\tilde{y} = \text{softmax}(W_{hy} \tilde{h})$$

where c_T^L is the memory state output output by the cell at the last timestep T and topmost layer L, h_t^L is the hidden state output by the LSTM at time t and topmost layer L, c is the context vector, $\tilde{h}$ is the attentional hidden state (where $[\cdot;\cdot]$ is concatenation and W_c are learnable parameters), and $\tilde{y}$ is the predicted distribution over classes for the input x.

Experiments and Results

Baseline	Test Accuracy
Best SVM	0.405
Best Softmax Regression	0.3809
Best 2 Layer Neural Network	0.7

Figure 1.

Model	Test Accuracy
Best Vanilla RNN	0.655
Best Vanilla LSTM	**0.79**
Best LSTM with Attention	**0.748**

Figure 2.

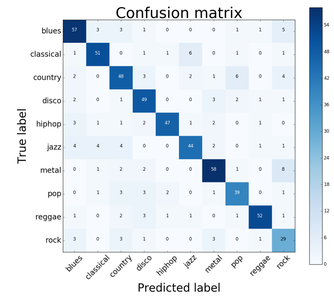


Figure 3.

Figure 1 shows a summary of baseline results. The best SVM uses a RBF kernel on a one-versus-rest scheme. The best softmax regression uses newton-cg for optimization. The best 2 layer MLP has a 125 dimensional hidden layer with a sigmoid nonlinearity.

Figure 2 shows a summary of experimental results. The best vanilla RNN has 2 layers and 125 dimensional cells. The best vanilla LSTM has 2 layers and 250 dimensional cells. The best LSTM with attention has 2 layers and 60 dimensional cells.

Figure 3 shows a confusion matrix of the best vanilla LSTM model.

Discussion

Surprisingly, the vanilla RNN model did not outperform the 2 layer MLP. As demonstrated by the success of the Vanilla LSTM, we believe this is due to the vanishing gradient problem since the sequences are relatively long. Moreover, the attention mechanism slightly hurts results, perhaps because attending to a particular part of the song provides no additional information, only increased model complexity.

Future Work

Our research has left us eager to pursue two future projects:
1. As is becoming common practice in visual recognition with Convolutional Neural Networks, we would like to develop an approach to interpret the learned features of our model, akin to [5]. We hope to auralize the learned features through music composition by transforming the weights to audio files. The greatest challenge this task poses is figuring out how to invert the spectrogram in an efficient manner.
2. Another method for understanding the learned model is to solve an optimization problem with respect to the inputs, leaving the learned parameters fixed. Consider the genre Jazz for this example. First, we find which neuron is most commonly active among all inputs when our model predicts that a clip is most likely Jazz. Then take the gradient of the chosen neuron with respect to the input vector. Using the inverse spectrogram technique described above we can convert the new input vector to an audio file and listen to the quintessential Jazz sound as learned by our LSTM.

References

[1] George Tzanetakis and Georg Essl and Perry Cook. Automatic musical genre classification Of audio signals. *Speech and Audio Processing, IEEE* pp. 293302, 2002.
[2] Alex Graves and Abdel-rahman Mohamed and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *Proc. International Conference on Acoustics, Speech and Signal Processing*, pp. 66456649, 2013.
[3] Ilya Sutskever and Oriol Vinyals and Quoc V. Le. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, pp. 3104-3112, 2014.
[4] Dzmitry Bahdanau and Kyunghyun Cho and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *Proc. International Conference on Learning Representations*, 2015.
[5] http://cs231n.github.io/understanding-cnn/