Music Genre Classification by Lyrics using a Hierarchical Attention Network

Alexandros Tsaptsinos
ICME
Stanford University
Stanford, CA 94305
alextsap@stanford.edu

Abstract

We adapt the hierarchical attention network for the task of genre classification using lyrics. Utilising a large dataset of intact song lyrics we apply a recurrent neural network model which tries to learn importance of words, lines and sections in the genre classification task. This hierarchical structure attempts to replicate the structure of lyrics and enable learning of which sections, lines or words play importance in different genres. We explore the differences between taking layers at the line level or at the section level. We test the model over a 117 genre dataset and a smaller 20 genre dataset, classifying over a much higher number of genres than previous research. Experiments show that the hierarchical attention network outperforms basic non-neural models and in certain situations simple neural models, although high classification accuracies still present a tough challenge. We visualise the performance of the model by extracting from songs the lines and words it applies most weight to in the classification task.

1 Introduction

Automatic classification of music is an important and well researched task in music information retrieval (MIR) [McKinney and Breebaart, 2003]. Previous MIR work has primarily focused on classifying mood [Logan et al., 2004, Hu and Downie, 2010], genre [Mayer et al., 2008], annotations [Tingle et al., 2010, Nam et al., 2012], and artist [Knees et al., 2004]. Typically one or a combination of audio, lyrical, symbolic, and cultural data is used in machine learning algorithms for this task [McKay et al., 2010].

Genre classification by lyrics presents itself as a natural language processing (NLP) problem. In NLP the aim is to assign meaning and labels to text; here this equates to a genre classification of the lyrical text. Traditional approaches in text classification have utilised n-gram models and algorithms such as Support Vector Machines (SVMs), k-Nearest Neighbour (k-NN), and Naïve Bayes (NB).

In recent years the use of deep learning methods such as recurrent neural networks (RNNs) or convolutional neural networks (CNNs) has produced superior results and represent an exciting breakthrough in NLP [Kalchbrenner et al., 2014, Kim, 2014]. Whilst linear and kernel models rely on good hand selected features, deep learning architectures attempt to prevent this by letting the model learn important features themselves. However, not much research has looked into the performance of these deep learning methods with respect to the genre classification task on lyrics. Here, we attempt to remedy this situation by extending the deep learning ideas on text classification to the particular case of lyrics.

Hierarchical methods attempt to use some sort of structure on the data to improve the models and have previously been utilised in vision classification tasks [Seo et al., 2016]. Yang et al. [2016] propose a hierarchical attention network for the task of document classification. Since documents often contain

structure they introduce this knowledge to the model, resulting in superior classification results. By applying successive attention layers the model attempts to learn which words and sentences play the most important parts in the classification. It is evident that lyrics contain a very hierarchical composition: words combine to form lines, lines combine to form sections, and sections combine to form the whole song. Here, we propose application of this hierarchical attention network to lyrical content.

The paper is structured as follows. Section 2 provides a review of other research in the area. In Section 3 we described our methods, including the dataset and a description of the hierarchical attention network. In Section 4 we provide results from our experiments: Section 4.1 details results mimicking the paper of Yang et al. [2016], Section 4.2 provides results and visualisations from the lyrics classification task. We conclude with a discussion in Section 5.

2 Background

Previous research has show that lyrical data performs the weakest in the genre classification task compared to other forms of data [McKay et al., 2010]. As a consequence, this problem is not as well researched and preference has been given to other methods. Previous non-neural lyrical classifiers struggled to achieve a classification accuracy any higher than 50%.

SVMs, k-NN, and NB have been heavily used in previous lyrical classification research. In addition no research has looked into classifying more than between 10 genres despite the prevalence of clearly many more. Fell and Sporleder [2014] classify among 8 genres using n-grams along with other hand selected features to help represent vocabulary, style, structure, and semantics. Ying et al. [2012] make use of Part-of-Speech tags and classify among 10 genres using SVMs, k-NN, NB with a highest accuracy of 39.94%. McKay et al. [2010] utilise hand selected features in lyrics to produce classification accuracies of 69% amongst 5 genres and 43% amongst 10 genres.

Often the lyrical data is combined with other forms of data to produce superior classifiers. Mayer et al. [2008] combine audio and lyrical data to produce a highest accuracy of 63.50% within 10 genres via SVMs. Mayer and Rauber [2011] then use a cartesian ensemble of lyric and audio features to gain a highest accuracy of 74.08% within 10 genres.

In recent years, MIR has seen the application of deep learning in several other situations [Scaringella et al., 2006, Li et al., 2010]. In particular, neural methods have been utilised for the genre classification task on other forms of data. Sigtia and Dixon [2014] used the hidden states of a neural network as features for song on which a Random Forest classifier was built, reporting an accuracy of 83% amongst 10 genres. Similarly, Li et al. [2010] use a CNN to learn features attaining an accuracy of 84% over 10 genres. Costa et al. [2017] compare the performance of CNNs in genre classification through spectrograms with respect to results obtained through hand-selected features and SVMs. The hierarchical nature of a song has also been previously exploited in genre classification tasks with Du et al. [2016] utilising hierarchical analysis of spectrograms to help classify genre.

3 Methods

3.1 Dataset

The majority of research into this problem has suffered from copyright issues on lyrics, which have forced previous literature to mostly use bag-of-words format lyrics. Often these bag-of-words are supplemented with Part-of-Speech (POS) tags and other hand selected features. In this format you lose the structure and it has been shown that utilising intact lyrics reveals superior results in these classification tasks [Fell and Sporleder, 2014, Smith et al., 2012].

For their superior performance over bag-of-word format, an intact lyric corpus was obtained through a research agreement with LyricFind[1], as has been previously utilised in Ellis et al. [2015], and Atherton and Kaneshiro. This contained a set of 1,039,151 song lyrics. The corpus provides basic metadata, however no genre label. To add these labels we utilise the iTunes Search API[2], extracting the value for the `primaryGenreName` as our baseline truth. This unfortunately greatly reduces the size of the

[1] http://www.lyricfind.com

[2] http://apple.co/1qHOryr

dataset due to the sparse iTunes database. We then further removed any songs that were linked with a genre tag of 'Music Video'. Some genres are very sparse in the dataset and for comparison we filter via two methods. In the first we remove any genres with less than 50 instances giving a dataset of size 495,188 lyrics consisting of 117 genres. In the second we only take the top 20 genres giving a dataset of size 449,458. It must be noted that iTunes attribute genres by artist, not by track; this is a known issue of the dataset for artists whose work may cover multiple genres. Additionally noted is that the dataset originally contained various versions of the same lyrics, due to the prevalence of cover songs; we only include one of these versions. The song lyrics are split into line and segments and tokenised using the `nltk` package in Python. We split the dataset into a rough split of 80% for training, 10% for validation, and 10% for testing.

3.2 Hierarchical Attention Networks

The structure of the model follows closely to that of Yang et al. [2016]. The HAN can be thought of as applying the same layer model to each hierarchy in the model. Each layer is run through a bidirectional GRU with attention applied to the output. The attention weights are then used to create a weighted vector which is then passed as the input to the next layer. A representation of the architecture for the example song of 'Happy Birthday' can be seen in Figure 1, where the layers are applied at the word, line and song level. We briefly step through the various components of the model.

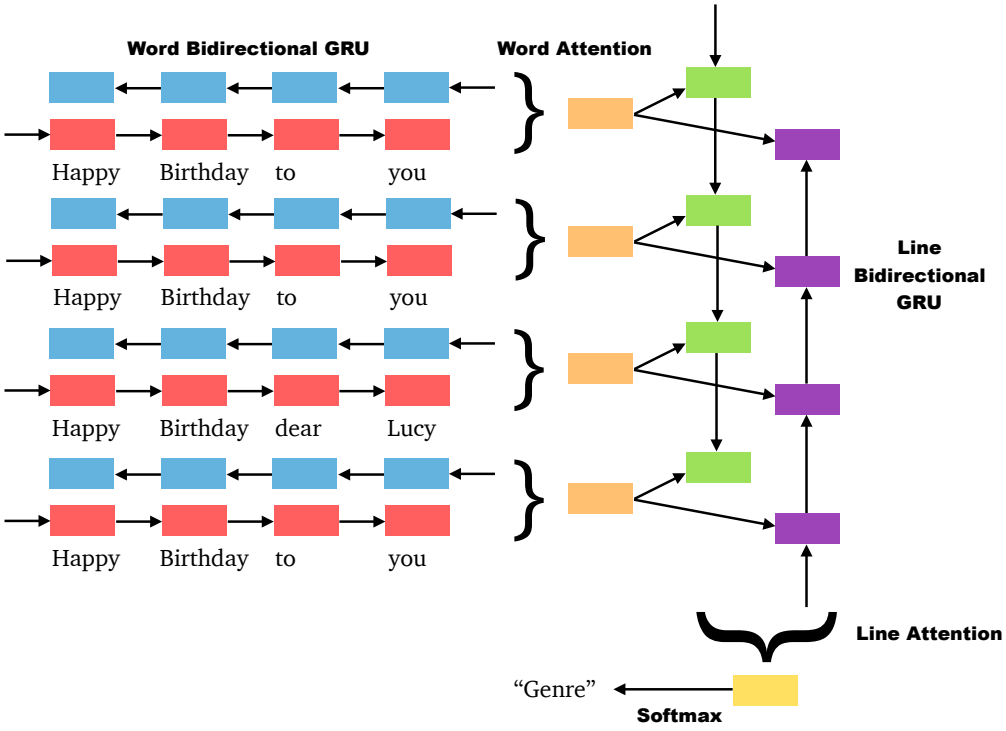

Figure 1: Representation of the HAN architecture; coloured boxes represent vectors. Blue and red vectors represent the hidden states for the forward and backward pass of the GRU at the word level, respectively. The orange line vector is then obtained from these hidden states via the attention mechanism. The green and purple vectors represent the forward and backward pass of the GRU at the line level, respectively. The yellow song vector is then obtained from these hidden states via the attention mechanism. Finally classification is performed via the softmax activation function.

3.2.1 Word Embeddings

An important idea in NLP is the use of dense vectors to represent words. To learn these word vectors a variety of methods have been proposed. A successful methodology proposes that similar words

have similar context and thus that these vectors should be learnt through their context, such as in the word2vec model from Mikolov et al. [2013]. Pennington et al. [2014] propose the GloVe method which combines global matrix factorisation and local context window methods to produce word vectors that outperformed previous word2vec and SVM based models.

We take as our vocabulary the top 30,000 words from the whole LyricFind corpus, including those we did not match with a genre. We train 100 dimensional GloVe word embeddings for these words using methods obtained from the GloVe website[3]. Previous research has shown that retraining these word vectors over the extrinsic task at hand can improve results if the dataset is large enough. In our experiments it was shown that retraining these word embeddings did improve results, and so we let our model learn superior embeddings to those provided by GloVe. For a full description of the GloVe method see Pennington et al. [2014].

3.2.2 Gated Recurrent Units

Introduced by Chung et al. [2014], Gated Recurrent Units (GRU) are a form of gating mechanism in RNNs designed to help overcome the struggle to capture long-term dependencies in RNNs. This is achieved by the introduction of intermediate states between the hidden states in the RNN. An update gate z_t is introduced to help determine how important the previous hidden state is to the next hidden state. A reset gate r_t is introduced to help determine how important the previous hidden state is in the creation of the next memory. Mathematically we describe the process as

$$z_t = \sigma\left(W_z x_t + U_z h_{t-1} + b_z\right) \tag{1}$$

$$r_t = \sigma\left(W_r x_t + U_r h_{t-1} + b_r\right) \tag{2}$$

$$\tilde{h}_t = \tanh\left(W_h x_t + r_t \circ U_h h_{t-1} + b_h\right) \tag{3}$$

$$h_t = (1 - z_t) \circ h_{t-1} + z_t \circ \tilde{h}_t, \tag{4}$$

where x_t is the word vector input at time-step t, σ is the sigmoid activation function and $\circ$ is the Hadamard product. W_z, U_z, W_r, U_r, W_h, and U_h are weight matrices randomly initialised and to be learned by the model along with the b_z, b_r, and b_h bias terms.

3.2.3 Hierarchical Attention

Attention was first proposed by Bahdanau et al. [2014] with respect to neural machine translation to allow the model to learn which words were more important in the translation objective. With regards to this paper, we would like our model to learn which words are important in classifying genre and then apply more weight to these words. Similarly, we can apply attention again on lines or song segments to let the model learn which lines or segments are more important in classification. A segment of a song is a verse, chorus or bridge of a song and typically consists of several lines. We explore the differences between applying attention at the line of segment level, as well as at the word level.

Given input vectors h_i for $i = 1, \ldots, n$ the attention mechanism can be formulated as

$$u_i = \tanh\left(W_a h_i + b_a\right) \tag{5}$$

$$\alpha_i = \frac{\exp(u_i^T u_a)}{\sum_{i=1}^{n} \exp(u_i^T u_a)} \tag{6}$$

$$s = \sum_{i=1}^{n} \alpha_i h_i, \tag{7}$$

where s is the output vector passed to the next layer consisting of the weighted sum of the current layers vectors. Parameters W_a, b_a, and u_a are learnt by the model after random initialisation. We note that the vector u_a computes relevance of the hidden representation of h_i through the cosine similarity operation.

One layer of the network takes in vectors $x_1, \ldots, x_n$, applies a bidirectional GRU to find hidden states and then uses attention mechanism to form a weighted sum of these hidden states to output as

[3] http://nlp.stanford.edu/projects/glove/

the representation. Letting GRU indicate the output of a GRU and ATT represent the output from an attention mechanism one layer is mathematically formulated as

$$\overrightarrow{h}_i = \overrightarrow{GRU}(x_i), \tag{8}$$

$$\overleftarrow{h}_i = \overleftarrow{GRU}(x_i), \tag{9}$$

$$h_i = [\overrightarrow{h}_i; \overleftarrow{h}_i], \tag{10}$$

$$s = ATT(h_1, \ldots, h_L). \tag{11}$$

Our HAN consists of two layers, one at the word level and one at the line/segment level. Consider a song of M lines or segments s_j, each consisting of n_j words w_{ij}. Let L be the pre-trained word embedding matrix. Letting LAY represent the dimension reduction operation of a layer in the network as in Equations 8 - 11 the whole HAN can be formulated for $i = 1, \ldots, n_j$ and $j = 1, \ldots, M$ as

$$x_{ij} = Lw_{ij} \tag{12}$$

$$s_j = LAY(x_{1j}, \ldots, x_{n_j j}), \tag{13}$$

$$s = LAY(s_1, \ldots, s_M). \tag{14}$$

Each layer has its own set of GRU weight matrix and bias terms to learn as well as its own attention weight matrix, bias terms and relevance vector to learn.

3.2.4 Classification

With the song vector s now obtained, classification is performed by using a final softmax layer

$$p = \text{softmax}\left(W_p s + b_p\right), \tag{15}$$

where intuitively we take the index of the largest value as the prediction for that song.

To train the model we use cross-entropy loss over K songs

$$J = -\sum_{k=1}^{K} \log(p_{d_k k}), \tag{16}$$

where d_k is the true genre label for that song.

3.2.5 Dropout

Proposed by Srivastava et al. [2014], dropout is a regularisation technique to prevent networks from overfitting. During training units in the network are randomly dropped with probability p giving the effect of training several different networks at the same time and forming an ensemble of them in classification. Not originally included in HAN we introduce dropout into the HAN at the hidden layer of each hierarchical layer in the network. Namely, we drop the output of the bidirectional GRU with probability p at each layer before continuing with attention. At testing time dropout is not included in the model.

4 Experiments

4.1 Yelp Review Classification

As an initial step we first look to mimic the classifier constructed in Yang et al. [2016]. The aim here is not to exactly recreate the results of the paper, but rather achieve a similar result using a classifier of the same structure, indicating we have built a good comparable model. We will then be able to carry this model over to the lyrics case and perform our optimisations for the set of lyrical data.

In the original paper, the authors introduce the HAN and utilise it for document classification. They perform experiments on well-known public datasets from the Yelp Dataset Challenge, IMDB reviews, Yahoo answers, and Amazon reviews. As a first objective we try to emulate the results obtained from the Yelp 2013 data in this paper, which involves classifying Yelp reviews as 1-5 stars. We obtain Yelp Reviews from the Yelp Dataset Challenge[4]. Unfortunately the archived data from 2013 is not

[4]https://www.yelp.com/dataset_challenge

Table 1: Yelp Review Classification

Paper	# Reviews	Vocabulary	Accuracy (%)
Tang et al. [2015]	335,018	211,245	65.1
Yang et al. [2016]	335,018	211,245	68.2
This paper	400,000	30,000	68.0

available and so instead we take a subsample of 400,000 reviews from the current datasets, which is of similar size to that of the Yelp 2013 dataset. As such exact replication of results may not be possible since the datasets used contain different reviews, however similar results should be achieved.

The structure of the HAN is exactly of that described in Section 3.2 however we have slight differences to the original model which we will now highlight, implemented for computational speed purposes. Each review is divided into sentences and tokenized using `nltk`[5] in Python. In the original model the tokenizing was done via Stanford's CoreNLP [Manning et al., 2014]. We retain the top 30,000 words as our vocabulary compared to the paper which keeps every word that appears at least 5 times. As described above, word embeddings are obtained via GloVe and are 100 dimensional compared the original model's use of word2vec and 200 dimensional vectors. Each direction of the GRU has a dimension of 50 and context vectors have a dimension of 100, both as in the original paper. Dropout is not included in the original model, however we introduce a dropout layer after each of the GRU layers with a dropout probability of 0.5. We utilise a mini-batch size of 64 as in the original paper but optimise using RMSprop rather than just stochastic gradient descent with momentum. We use a learning rate of 0.1, clip gradients at 1, and let the model run for 10 epochs. The results compared with that of the original paper can be seen in Table 1. Included for comparison is the best baseline model from the original paper, namely the LSTM-GRNN model from Tang et al. [2015].

From Table 1 we can see that despite the reduced vocabulary size and reduced embedding dimension near identical results have been produced that are much better than those produced by the best baseline model. Further, this accuracy was produced with minimal parameter tuning and it is hypothesised that superior accuracy would have been produced had more tuning taken place. Confident with the success of the starting model from this initial test we now continue on to the music genre classification task, where we will optimise further than has been done here.

4.2 Lyrical Genre Classification

4.2.1 Baseline Models

We compare the performance of the HAN against various baseline models.

1. (MC) 'Rock' is the most common genre in our dataset. The majority classifier simply predicts 'Rock' for every song.
2. (LR) A logistic regression run on the average song word vector produced from the GloVe word embeddings.
3. (LSTM) A Long Short-Term Memory where we treat the whole song as a single sequence of words and use max-pooling of the hidden states for classification. 50 hidden states were used in the LSTM and each song had a maximum of 600 words. For full discussion of the LSTM framework see Hochreiter and Schmidhuber [1997].
4. (HN) The hierarchical network structure in the absence of attention run at the line level. Instead at each layer all of the representations are averaged to produce the next layer input. This is equivalent to applying equal attention to each vector in that layer.

For LR, LSTM and HN we let the model retrain the word embeddings as it trains.

4.2.2 Model Configuration

Before testing the model, hyperparameters are tuned on the development set. We dropout with probability $p = 0.5$ and gradients are clipped at a maximum norm of 1. We utilise a mini-batch size

[5] http://www.nltk.org/

Table 2: Genre Classification Test Accuracies via Lyrics (%)

Model	117 Genre Dataset	20 Genre Dataset
Majority Classifier	24.71	27.17
Logistic Regression	35.21	38.13
LSTM	43.66	49.77
HN	45.85	49.09
HAN-L	46.42	49.50
HAN-S	45.05	47.60

of 64 and optimise using RMSprop with a learning rate of 0.01 [Tieleman and Hinton, 2012]. In each case 10 epochs were enough for the validation loss to stop decreasing. The baselines were run until their validation loss did not decrease for 3 successive epochs.

We test the models on the two different subsamples of the dataset and investigate the difference between applying the first layer at the line or section level. The models are padded/truncated to have uniform length. In the line model, each line has a maximum of 10 words and a maximum of 60 lines. In the section model each section has a maximum of 60 words and a maximum of 10 sections.

4.2.3 Results

For both dataset sizes we run the baseline models and the HAN ran at the section and line levels. Let HAN-L represent running over lines and HAN-S represent running over sections. The test accuracies are seen in Table 2.

From the results we see a trend between model complexity and classification accuracy. The very simple majority classifier performs weakest and is improved upon by the simple logistic regression on average bag-of-words. The neural based models perform stronger than both of the simple models. The LSTM model, which takes into account word order and tries to implement a memory of these words, gives performances of 43.66% and 49.77%, outperforming the HAN on the 20 genre dataset. Over the 117 genre dataset the best performing models were the HANs, with a highest accuracy of 46.42% when run over lines. It is observed that for the simpler 20 genre case, the more complex HAN is not required since the simpler LSTM beats it, although the LSTM took almost twice as long to train as the HAN. However for the tougher 117 genre case the HAN-L outperforms the LSTM, perhaps picking up on more of the intricacies of rarer genres.

In both cases the HAN ran at the line level produced superior results than ran over the section level giving a bump of roughly 1.4% and 1.9% in the 117 genre and 20 genre dataset, respectively. The HN, which is ran at the line level, additionally outperforms the HAN at section level. This indicates that the model performs better when looking at songs line by line rather than section by section. In the HAN-L the model can pick up on many repeated lines or lines of a similar ilk, rather than the few similar sections it attains in the HAN-S, and this may be attributive to the better performance. The network does benefit from the inclusion of attention, with the HAN-L classifying with higher accuracies than the HN.

As expected, classifying over the 20 genre dataset has given boosts of roughly 3% and 2.5% in the HAN-L and HAN-S, respectively. It is interesting to note that discarding roughly 10% of the data by only keeping roughly a sixth of the genres has not strengthened the model by much. The increase in accuracies is most likely mainly down to the removing of instances of incorrect classifications between rarer genres.

In each of the HAN models the validation loss stopped decreasing between the 5th and 8th epoch. The training and validation accuracies across the epochs for each of these models can be seen in Figure 2. The HAN-L continues to learn for longer than the HAN-S. This can be seen by the upward smile shape occurring in both of the HAN-S loss plots compared to the HAN-L. In each of the cases, whilst the training loss catapults downwards, the validation loss does not decrease greatly with no model managed to decrease the validation loss by more 1.0. For the 20 genre case the loss is, as expected, much lower than in the 117 genre case. We expect this, since classifying between many less choices becomes naturally an easier task.

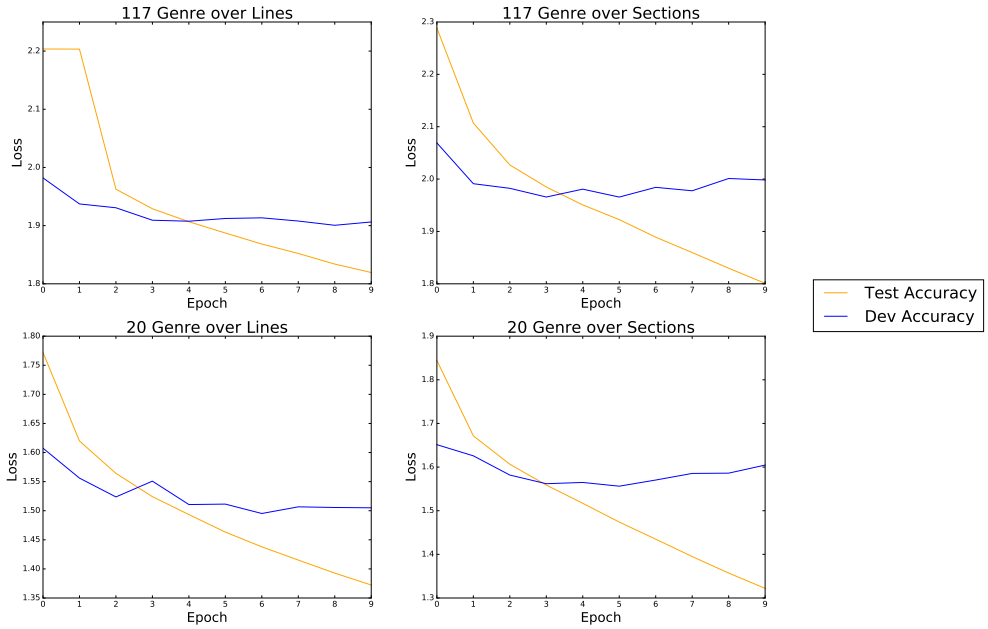

Figure 2: Learning rates of the HAN during training.

Table 3: HAN-L Confusion matrix for top 5 genres over 117 genre dataset. Rows represent true genre, whilst columns are predicted.

	Rock	**Pop**	**Alternative**	**Country**	**Hip-Hop/Rap**
Rock	6879	1198	2460	561	91
Pop	1892	2620	738	324	157
Alternative	2385	534	2866	150	67
Country	534	304	90	2199	4
Hip-Hop/Rap	71	97	63	11	2629

The confusion matrix for HAN-L run over the larger dataset for the top 5 genres can be seen in Table 3. We can see from the matrix that Rock, Pop and Alternative are all commonly confused; the model predicts Rock for Alternative almost as many times as it does Alternative. As the most common genre in the dataset by about 30,000 it is unsurprising to see the model try and predict Rock more often, and it is unclear whether a person would be able to distinguish between the lyrics of these genres. However, we see that both Country and Hip-Hop/Rap are more separated. With their distinct lyrical qualities, especially in the case of Hip-Hop/Rap, this is an encouraging result indicating that the model has learnt some of the qualities of both these genres.

4.2.4 Attention Visualisation

To help illustrate the attention mechanism, we feed song lyrics into the HAN-L and observe the weights it applies to words and lines. For each song we extract the 5 most heavily weighted lines and a visualisation of their weights and the individual word weights for a few different correctly predicted song lyrics can be seen in Figure 3. We warn that offensive language is present in these lyrics.

From these visualisations we notice that the model has placed greater weights on words we may associate with a certain genre. For example ‘baby’, and ‘ai’, are weighted heavily in the Country song and the most heavily weighted line in that song has a very strong Country feel. The model has placed great weight on a blank line, indicating the break between sections; it is unclear whether the model is learning to place importance on how songs are sectioned and the number of sections occurring. In

the Hip-Hop/Rap song the model places attention on swearwords used in the piece, as well as the incorrectly spelled 'dyin' and 'hurtin'. The model picks up the 'woh' and 'oo' in the Rock song and also heavily weights occurrences of second person determiner 'your' and pronoun and 'you'. It was found that for many Rock songs this was the case.

In addition some visualisations of lyrics that were incorrectly classified by the HAN-L can be seen in Figure 4. We observe the model predicting Country for a Pop song, applying weights to 'sin' and 'strong' which could be characteristic of Country songs. The dataset contains songs with foreign language lyrics, and it is unclear whether the iTunes genre association is truly correct. Here we observe a song with Spanish lyrics classed as Pop Latino by the model whilst iTunes deems it Pop. This seems like a fair mistake for the model to have made since it has evidently recognised the Spanish language. The model also incorrectly classifies the Hip-Hop/Rap song as Pop. In the 5 most heavily weighted lines we do not spot any instances of language that indicate a Hip-Hop/Rap song and we hypothesise the genericness of the lyrics has led the model to predict Pop.

5 Discussion

In this paper we have shown that a HAN and other neural-based methods can improve on the genre classification accuracy. We have shown by classifying over a large dataset of nearly half a million song lyrics we have been able to outperform other basic methods. In large this model has beaten all of the previously reported lyrical only genre classification model accuracies from Section 2, except for the classification amongst 5 genres. We have shown that the HAN works better with layers at the word, line and song level rather than word, section and song level.

Visualisations of the weights the HAN applies to words and lines were produced to help see what the model was learning. In a good amount of cases words and lines were heavily weighted that were cohesive with the song genre, however this was not always the case. We note that in general the model tended to let one word dominate a single line with the greatest weight. However this was not as apparent across lines, with weights amongst lines much more evenly spread. With a large amount of foreign language lyrics also present in the dataset an idea for further research would be to build a classifier that identified language, and from there classified by genre. Any such research would be inhibited, however, by the lack of such a rich dataset to train on.

From one standpoint, classification by lyrics will always be inherently flawed by vague genre boundaries with many genres borrowing lyrics and styles from one another. For example one merely need consider cover songs which utilise the same lyrics but produce songs in vastly different genres, or songs which have no lyrical content. To produce a state of the art classifier is is evident that this classifier must take into account more than just the lyrical content of the song. Audio data typically performs the strongest and further research could look into employing this hierarchical attention model to the audio and symbolic data and combining with the lyrics to build a stronger classifier. Lyrics typically perform stronger on the sentiment analysis classification task and further research could employ the HAN to this problem. In addition the HAN could be extended to include both a layer at the line and section level, or even at the character level to explore performance.

References

Jack Atherton and Blair Kaneshiro. I said it first: Topological analysis of lyrical influence networks.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Yandre MG Costa, Luiz S Oliveira, and Carlos N Silla. An evaluation of convolutional neural networks for music classification using spectrograms. *Applied Soft Computing*, 52:28–38, 2017.

Wei Du, Hu Lin, Jianwei Sun, Bo Yu, and Haibo Yang. A new hierarchical method for music genre classification. In *Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI), International Congress on*, pages 1033–1037. IEEE, 2016.

Robert J Ellis, Zhe Xing, Jiakun Fang, and Ye Wang. Quantifying lexical novelty in song lyrics. In *ISMIR*, pages 694–700, 2015.

Michael Fell and Caroline Sporleder. Lyrics-based analysis and classification of music. In *COLING*, volume 2014, pages 620–631, 2014.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Xiao Hu and J Stephen Downie. Improving mood classification in music digital libraries by combining lyrics and audio. In *Proceedings of the 10th annual joint conference on Digital libraries*, pages 159–168. ACM, 2010.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

Yoon Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

Peter Knees, Elias Pampalk, and Gerhard Widmer. Artist classification with web-based data. In *ISMIR*, 2004.

Tom LH Li, Antoni B Chan, and A Chun. Automatic musical pattern feature extraction using convolutional neural network. In *Proc. Int. Conf. Data Mining and Applications*, 2010.

Beth Logan, Andrew Kositsky, and Pedro Moreno. Semantic analysis of song lyrics. In *Multimedia and Expo, 2004. ICME'04. 2004 IEEE International Conference on*, volume 2, pages 827–830. IEEE, 2004.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*, pages 55–60, 2014.

Rudolf Mayer and Andreas Rauber. Musical genre classification by ensembles of audio and lyrics features. In *Proceedings of International Conference on Music Information Retrieval*, pages 675–680, 2011.

Rudolf Mayer, Robert Neumayer, and Andreas Rauber. Combination of audio and lyrics features for genre classification in digital audio collections. In *Proceedings of the 16th ACM international conference on Multimedia*, pages 159–168. ACM, 2008.

Cory McKay, John Ashley Burgoyne, Jason Hockman, Jordan BL Smith, Gabriel Vigliensoni, and Ichiro Fujinaga. Evaluating the genre classification performance of lyrical features relative to audio, symbolic and cultural features. In *ISMIR*, pages 213–218, 2010.

Martin McKinney and Jeroen Breebaart. Features for audio and music classification. 2003.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

Juhan Nam, Jorge Herrera, Malcolm Slaney, and Julius O Smith. Learning sparse feature representations for music annotation and retrieval. In *ISMIR*, pages 565–570, 2012.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. *ICML (3)*, 28:1310–1318, 2013.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

Nicolas Scaringella, Giorgio Zoia, and Daniel Mlynek. Automatic genre classification of music content: a survey. *IEEE Signal Processing Magazine*, 23(2):133–141, 2006.

Paul Hongsuck Seo, Zhe Lin, Scott Cohen, Xiaohui Shen, and Bohyung Han. Progressive attention networks for visual attribute prediction. *arXiv preprint arXiv:1606.02393*, 2016.

Siddharth Sigtia and Simon Dixon. Improved music feature learning with deep neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 6959–6963. IEEE, 2014.

A Smith, C Zee, and A Uitdenbogerd. In your eyes: Identifying clichés in song lyrics. In *Australasian Language Technology Workshop 2012 (ALTW 2012)*, pages 88–96. Australasian Language Technology Association, 2012.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1): 1929–1958, 2014.

Duyu Tang, Bing Qin, and Ting Liu. Document modeling with gated recurrent neural network for sentiment classification. In *EMNLP*, pages 1422–1432, 2015.

Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2), 2012.

Derek Tingle, Youngmoo E Kim, and Douglas Turnbull. Exploring automatic music annotation with acoustically-objective tags. In *Proceedings of the international conference on Multimedia information retrieval*, pages 55–62. ACM, 2010.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of NAACL-HLT*, pages 1480–1489, 2016.

Teh Chao Ying, Shyamala Doraisamy, and Lili Nurliyana Abdullah. Genre and mood classification using lyric features. In *Information Retrieval & Knowledge Management (CAMP), 2012 International Conference on*, pages 260–263. IEEE, 2012.

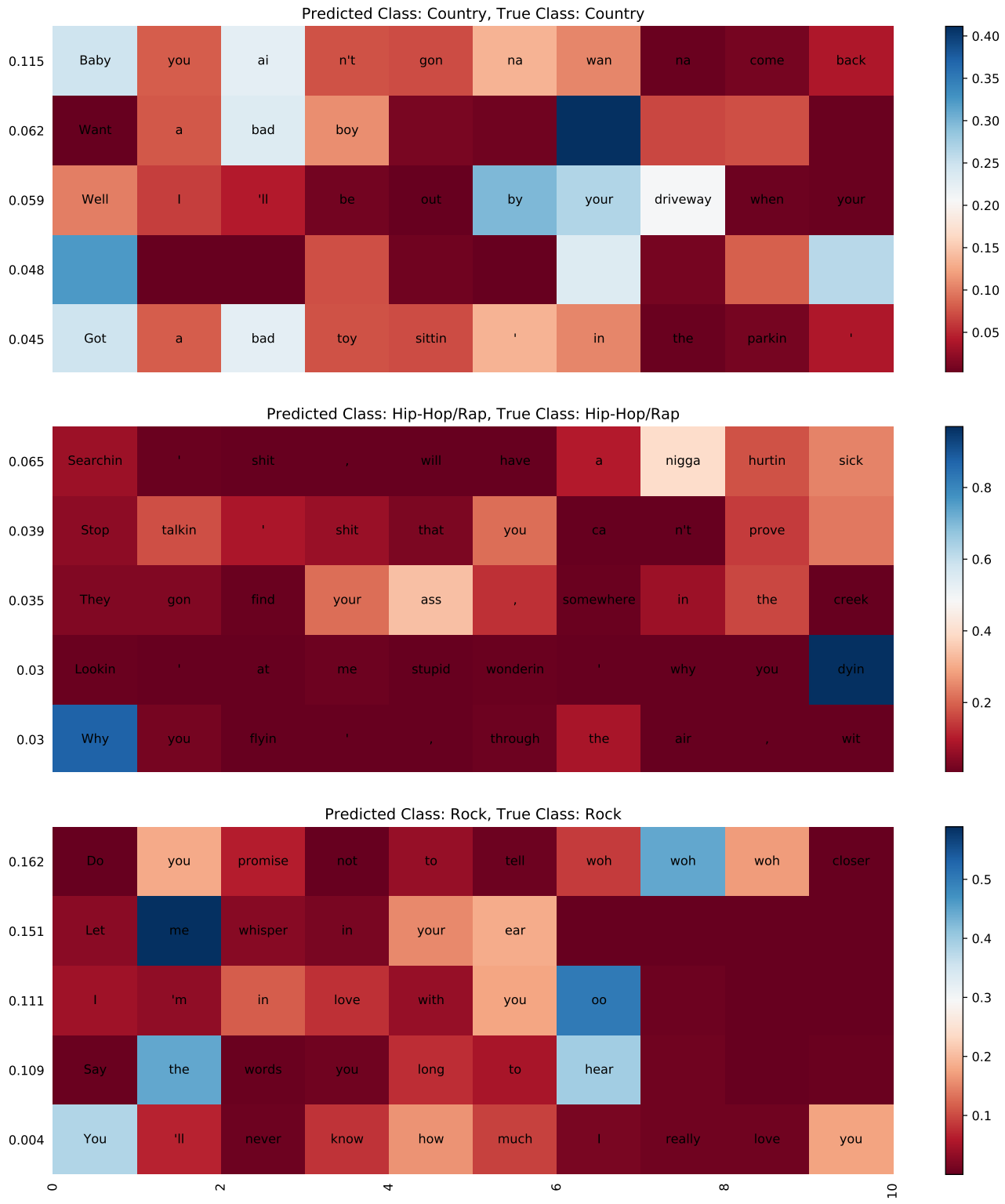

Figure 3: Weights applied by the HAN-L for song lyrics that were correctly classified. Line weights appear to the left of each line and word weights are coloured according to the respective colorbars on the right.

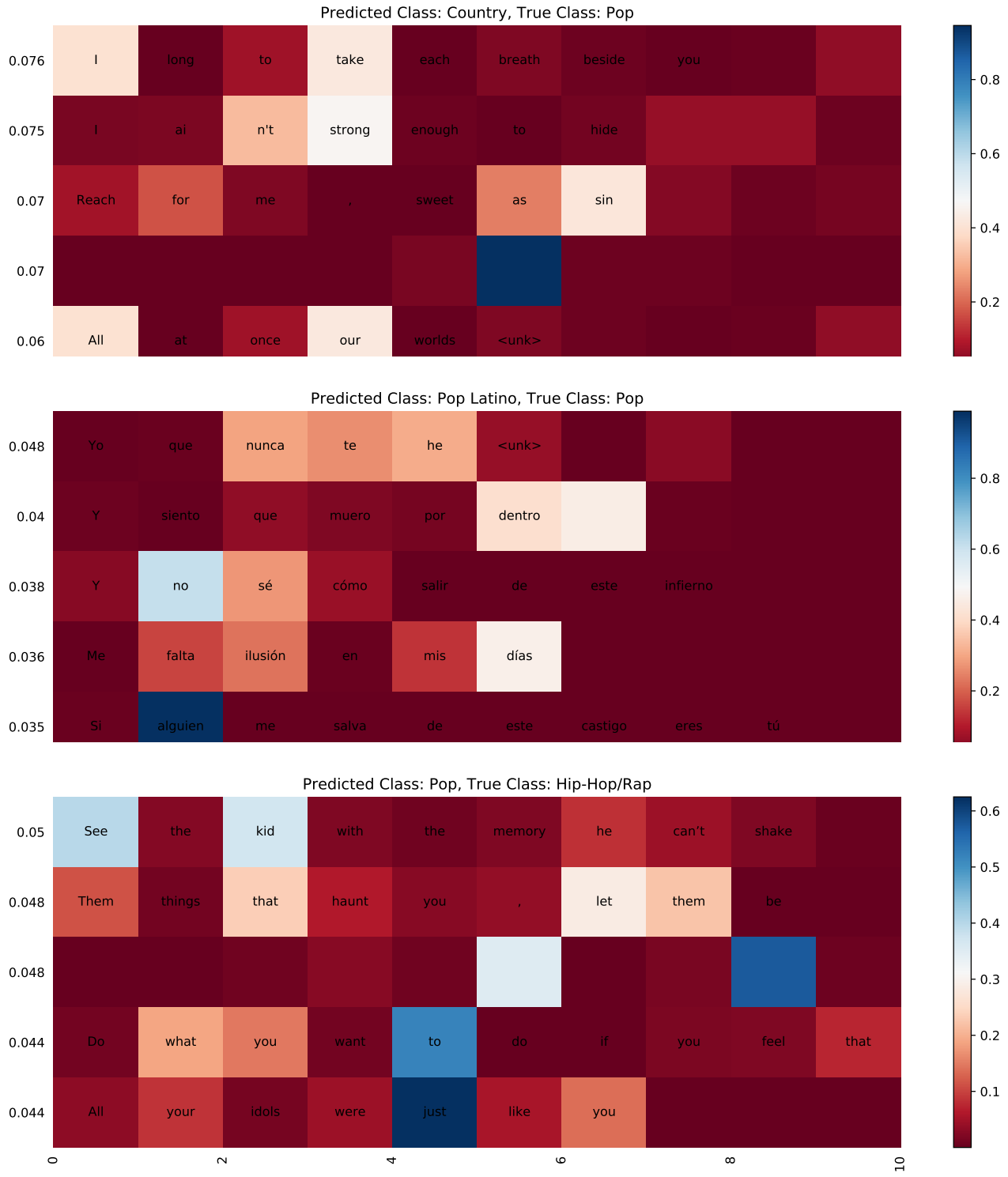

Figure 4: Weights applied by the HAN-L for song lyrics that were incorrectly classified. Line weights appear to the left of each line and word weights are coloured according to the respective colorbars on the right.