IMPROVED MUSIC FEATURE LEARNING WITH DEEP NEURAL NETWORKS

Siddharth Sigtia, Simon Dixon

Queen Mary University of London
Centre for Digital Music
Mile End Road, London E1 4NS, UK

ABSTRACT

Recent advances in neural network training provide a way to efficiently learn representations from raw data. Good representations are an important requirement for Music Information Retrieval (MIR) tasks to be performed successfully. However, a major problem with neural networks is that training time becomes prohibitive for very large datasets and the learning algorithm can get stuck in local minima for very deep and wide network architectures. In this paper we examine 3 ways to improve feature learning for audio data using neural networks: 1.using Rectified Linear Units (ReLUs) instead of standard sigmoid units; 2.using a powerful regularisation technique called Dropout; 3.using Hessian-Free (HF) optimisation to improve training of sigmoid nets. We show that these methods provide significant improvements in training time and the features learnt are better than state of the art hand-crafted features, with a genre classification accuracy of 83 $\pm$ 1.1% on the Tzanetakis (GTZAN) dataset. We found that the rectifier networks learnt better features than the sigmoid networks. We also demonstrate the capacity of the features to capture relevant information from audio data by applying them to genre classification on the ISMIR 2004 dataset.

Index Terms— Deep Learning, Neural Networks, MIR.

1. INTRODUCTION

Most MIR classification tasks use a common pipeline which consists of feature extraction followed by classification. This pipeline assumes that the features capture all the relevant information for a particular task. The features that are extracted are often "hand-crafted", a process that requires significant domain knowledge and engineering ingenuity. Recently, the slow progress in improving the accuracy of MIR tasks has been attributed to the use of hand-engineered features [15] and there have been several studies that explore feature learning for music audio data [17, 10, 12].

Stochastic Gradient Descent (SGD) has been used for training neural networks for the last 25 years. Although it has many advantages like ease of implementation and good convergence, there are also several drawbacks. One of the drawbacks is that the algorithm does not scale very well to large datasets. The algorithm can go through hundreds of thousands of iterations to converge to acceptable solutions on large datasets. SGD can also underfit the data if the network is very deep, a problem that had plagued training of deep neural networks up until the recent use of layer-wise unsupervised pre-training [13].

Recently, it has been found that the performance of SGD can be improved by using Rectified Linear Units (ReLUs) [20, 9]. These rectifier networks converge to similar and sometimes better solutions than the traditional sigmoid nets. In this study, we evaluate using ReLUs and directly training the network with SGD, without any unsupervised pre-training. However, ReLUs tend to overfit the data at times. Dropout is a powerful regularisation technique that helps reduce the generalization error [14]. Rectifier networks when combined with Dropout are known to generalize well.

It would be interesting to be able to compare the features learnt by rectifier nets to the features learnt by sigmoid nets. However, deep sigmoid networks are harder to optimize with SGD because of the non-linearity of the objective with respect to the parameters. SGD also takes a very large number of iterations to train sigmoid nets, making training time prohibitively large [10]. Recently, Martens proposed a powerful new method for training neural networks, called Hessian Free (HF) optimisation [18]. HF is a second order optimisation technique that has proved to be very effective for training deep and recurrent neural networks [19]. HF also offers other practical advantages like significant reduction in the number of hyper-parameters and a substantial reduction in training time.

In this paper we learn features from audio data from the GTZAN dataset using both sigmoid and rectifier neural networks. We show that SGD can be used to train ReLU nets efficiently. We also use Hessian Free optimisation to train sigmoid nets and compare their performance with rectifier networks. We find that HF is very efficient for training sigmoid networks and provides good solutions with a significant reduction in training time as compared to SGD. We also find that the rectifier nets with Dropout performed better than the sigmoid nets. We then apply the two types of learnt features to a genre classification problem on the ISMIR 2004 dataset and show that the learnt features perform better than hand-crafted features.

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

The rest of the paper is organized as follows. In section 2 we describe ReLUs, Dropout and HF. Section 3 describes the details of the experimental setup and the results. Finally, conclusions and ideas for future work are presented in section 4.

2. BACKGROUND

2.1. Rectified Linear Units

Recently, there has been a large amount of evidence demonstrating the advantages of using ReLUs over the traditional sigmoid units [20, 9]. These advantages include good convergence without the need for any pre-training, naturally sparse features in the hidden layers [9] and overcoming the problem of vanishing gradients [8, 2]. The ReLU activation function is defined as $f(x) = max(0, x)$. Unlike sigmoid units, ReLUs do not saturate at 1 and the partial derivative of the activation function with respect to the model parameters is never 0, provided the neuron is active. The hidden layer activations of ReLU nets have a hard sparsity, since the units cut off below 0. This is useful if the hidden unit activations are used as features. Neural nets with ReLUs as hidden units can reach the same error level on the training set much faster than sigmoid nets [16]. This is advantageous since it allows experimentation with much larger network architectures, which would not be possible using sigmoid nets trained with SGD.

2.2. Dropout

Although ReLUs have several properties that make them desirable as hidden units for deep neural nets, they tend to overfit the training data. Overfitting is a common problem with networks of large capacity and is generally solved by using various regularisation techniques [6]. Dropout is a regularisation technique which was introduced in [14]. Dropout when used with ReLUs reduces the problem of overfitting to the training data.

Dropout adds noise to the network during training by *dropping out* or removing a predetermined percentage of activations in each layer. The units that are dropped out are chosen at random. Therefore training a network with dropout is the same as training an ensemble of "thinned"or sub-sampled neural networks that share parameters. There are two ways of interpreting the way dropout modifies training. The first one is that Dropout avoids complex co-adaptations of the features to the training data. The second interpretation of Dropout is that it is similar to model averaging. Our implementation of Dropout follows the details outlined in [7].

2.3. Hessian Free optimisation

Sigmoid nets are hard to train with SGD because the error surface defined by the loss function is very complex due to the sigmoid non-linearities at each layer. This problem becomes worse as the depth of the network increases. Greater depth also leads to issues like vanishing gradients due to the saturation of the sigmoid function. There have been attempts to fix this by using different learning rates for different layers, adaptive schedules for the learning rate and momentum. However for sufficiently deep networks SGD proves to be inadequate. Hessian Free (HF) optimisation is a second order optimisation technique that aims to minimize an objective function $f(\theta)$ with respect to the parameters θ. The algorithm iteratively updates the parameters at every step by optimizing a local approximation to the objective. More specifically, the local approximation to the objective function at some iteration n is defined as:

$$M_{\theta_n} = f(\theta_n) + f'(\theta_n)^\intercal \delta_n + \delta_n^\intercal B \delta_n / 2 \quad (1)$$

In equation 1, B is the Gauss-Newton matrix which is used instead of the Hessian and δ_n are the search directions for updating the parameters θ_n. Although Hessian Free methods have existed in literature and have been studied for a long time, they were grossly impractical for applications to machine learning problems until recently. In [18], Martens makes several modifications to the earlier approaches and develops a version of Hessian Free that can be applied effectively to train very deep networks. HF uses the method of Conjugate Gradients (CG) to find a solution to equation 1 at each step. CG is very sensitive to the form of the curvature matrix and small batches of data are ineffective for training. In general, very large batches of data are used for each iteration of HF. Apart from better performance of CG, using large datasets significantly reduces the number of iterations required to train a network. Another advantage of using HF is that it greatly reduces the number of hyper-parameters that need to be tuned for useful solutions. In our experiments, we use HF to train sigmoid nets and compare the performance with rectifier networks.

3. EXPERIMENTS

In this section we describe the experimental details and the results. Two datasets were used for the experiments, the GTZAN dataset [24] and the ISMIR 2004 Genre dataset [1]. Although the GTZAN dataset has some shortcomings [22], it has been used as a benchmark for genre classification tasks. The GTZAN dataset consists of 1000, 30-second examples with 100 examples for each of the 10 genres. The ISMIR 2004 dataset consists of 729 examples over 6 genres for training and 729 songs for development/validation. The tracks from the ISMIR 2004 dataset were down-sampled to 22500 Hz and the first 30 seconds from each song were kept. The GTZAN dataset was split into four 50/25/25 train, validation, test splits. For all experiments with the GTZAN dataset, the training and validation sets were first used to train the neural network. The neural network was then used to extract features from the data and a classifier was trained on the features extracted from the training and validation set. The results

	No-Aggregation				Aggregation		
Hidden Units	Layer	ReLU+SGD	ReLU+SGD+Dropout	Sigmoid + HF	ReLU+SGD	ReLU+SGD+Dropout	Sigmoid + HF
50	1	75.0±1.3	75.0±1.4	71.8±0.2	75.0±1.7	76.5±1.5	78.5±2.1
	2	75.4±1.2	77.5±2.2	74.3±2.5	79.6±2.7	77.0±2.2	80.0±2.6
	3	78.3±1.1	77.0±1.2	77.8±0.7	81.3±1.8	78.0±1.0	80.8±1.1
	All	79.0±2.0	78.0±1.6	77.2±1.0	81.5±1.9	81.5±1.7	**82.1±1.7**
500	1	72.7 ±2.8	73.5±1.9	65.6±1.6	71.8±0.7	75.5±1.1	67.8±1.5
	2	78.5±1.9	78.5±2.9	70.5±1.2	79.5±1.9	82.5±1.8	74.0±2.6
	3	80.5±1.4	79.5±2.6	73.8±0.3	83.0±1.2	82.0±1.4	77.1±2.36
	All	79.0±1.4	80.5±1.8	71.6±1.5	82.5±2.3	**83.0±1.1**	76.0±1.0

Table 1. Genre classification results on GTZAN dataset (mean accuracy and standard deviation over 4 splits)

were then calculated on the test data. This was repeated for each of the 4 data splits and results are reported over these 4 splits. We then picked the model that performs best on the GTZAN dataset and applied it to a genre classification task on the ISMIR 2004 dataset.

3.1. Training

We followed a pipeline very similar to the one described in [10]. We calculate FFTs on frames of length 1024 at 22050 kHz sampling rate with an overlap of 50% and use the absolute value of each FFT frame. The output from each frame is a 513 dimensional vector. Each feature dimension was then normalized to have zero mean and unit standard deviation. Both the rectifier and sigmoid networks had the same architecture so that the features could be compared. We tried a large number of permutations of the number of hidden units, and we quote results on the two architectures that performed best. In the first architecture each hidden layer has 50 units and in the second, each has 500 units.

When training the rectifier networks with SGD, the output layers consisted of softmax units. The learning rate was tuned by a grid search and no update schedules were used. We found that a learning rate of 0.01 worked best when using ReLUs. The validation set was used to perform early stopping with a patience of 10. For the experiments with Dropout, we applied the same rate to all the hidden layers. We found the optimum dropout rate to be 0.25 for all the hidden layers. We also experimented with applying dropout to the input layer and found that we got better results when it wasn't applied to the input layer. We did not apply Dropout to the sigmoid networks because there are no accepted modifications of HF that incorporate Dropout [7].

For the experiments with Hessian Free optimisation, the output layer consisted of sigmoid units. We experimented with softmax and sigmoid output layers and found that both produced comparable results. We used the entire training data for the gradient calculation and 25% of the training data for calculation of the Gauss-Newton matrix. Increasing the number of examples provided to the curvature matrix calculation beyond this did not improve accuracies any further. An initial damping factor of 10 gave the best results on the test dataset. The Conjugate Gradient (CG) step of the algorithm was limited to a maximum of 250 iterations and the number of steps of HF was limited to 200. Apart from this, we follow all of the details of Martens' approach [19].

We used the Theano [3] python library for training the networks on a GPU.

3.2. Classification

Our aim is to use neural networks to discover features that can be used for MIR tasks. We used the activations of the hidden layers of the neural networks as features and trained a Random Forest classifier on top of these features to predict the label. Training a separate classifier on the activations is useful since it allows direct comparison with other features. There is also some evidence [23] that suggests that training a more powerful classifier, as compared to logistic regression, on the top layer of a neural net provides better results. However unlike that paper, the classifier was not trained jointly with the rest of the network. For the ReLU networks, training a separate classifier on the activations is potentially more beneficial because of the hard sparsity present in the hidden layers. We used a Random Forest classifier because the amount of training data is quite large (645,000 training examples) and Random Forest classifiers scale well to large datasets. The classifier was used to predict a label for each frame of a test example, and maximum voting was used to predict the genre for the example. We compare the results of training the classifier on the activations of each hidden layer and on all the hidden layer activations combined.

3.3. Aggregated Features

Several authors have shown significant improvements in classification accuracy by aggregating features over time [4, 10, 5, 24]. In our experiments we found that classification accuracy does not vary much over a range of aggregation times between 3 to 6 seconds. We aggregated the features over 5 seconds with an overlap of 2.5 seconds. Each aggregated feature vector was the mean and variance of the features that lie in that 5 second window.

3.4. Genre Classification on ISMIR 2004 Dataset

To evaluate the effectiveness of the features learnt in the previous section, we applied the same features to a genre classification task on the ISMIR 2004 dataset. The motivation

Hidden Units	Layer	ReLU+SGD	ReLU+SGD+Dropout	Sigmoid + HF
50	1	70.50	68.03	68.72
	2	70.80	66.94	70.23
	3	69.13	68.03	70.50
	All	72.42	69.68	71.20
500	1	68.03	70.09	68.40
	2	71.33	72.01	68.32
	3	71.46	69.41	70.37
	All	72.30	**73.46**	70.23

Table 2. Genre classification results on the ISMIR 2004 dataset

behind doing this is to test whether the features are robust to changes in the input distribution and if they capture information that can be used for tasks other than what they were trained for. We aggregated the features over 5 seconds with a 2.5 second overlap and applied them to a genre classification task. We picked the two models that performed best on the GTZAN dataset to extract features from the ISMIR data. The same procedure of classifying aggregated frames, followed by maximum voting was used to classify the test examples. To compare the learnt features with some baseline hand-crafted features, we also repeated the above experiment with MFCC features and Principal Mel-Spectrum Components (PMSC) features [11]. Previous work [10] applies the features learnt from sigmoid nets to an auto-tagging task, however we could not perform the same experiment due to unavailability of the data.

3.5. Results and discussion

Table 1 shows the results of genre classification on the GTZAN dataset. We note that the best accuracy is achieved by the rectifier net with dropout and a large number of hidden units. The system classifies the GTZAN data with an accuracy of $83 \pm 1.1\%$. These results cannot be directly compared to the results in [10], because we average results over 4 splits of the dataset. However it can be seen that results are in a similar range. We also compare our results with the results of other methods applied to the GTZAN dataset presented in [12]. MFCC features provide an accuracy of 77% while a large number of hand-crafted features provide an accuracy of 83%. We note that our method compares favourably to the current state of the art results on the GTZAN dataset. The major improvement over the system in [10], is that the network is trained from randomly initialized weights without any pre-training. The number of training epochs that SGD takes to converge is also less than with sigmoid nets. Since we use early stopping, the number of epochs varies with each trial. But on average, the system can converge to good solutions in 200 training epochs (for 50 hidden units per layer) in a training time of about 14 hours (for 200 epochs). Since SGD for rectifier nets converge faster, we were able to train networks with more hidden units which provided better results. We note that Dropout helps improve results when the number of hidden units is large. For the smaller network, the results with Dropout are worse.

From table 1, the network with 50 hidden units trained with HF gives results comparable to the network trained with SGD in [10] and better results than the rectifier network trained with the same number of units. However the sigmoid network with 500 units performs much worse, although training and validation errors were lower. This implies that the larger network overfits the training data. HF, as an optimization algorithm doesn't specifically deal with the problem of overfitting. Overfitting can be reduced with the usual regularisation techniques, which we tried to keep to a minimum to be able to compare the features learnt by the two kinds of networks (except when using Dropout). However we obtain significant improvements in training time and in reduction of hyper-parameters that need to be tuned. With the number of iterations fixed to the values described earlier, HF takes 12 hours (averaged over 4 trials) to train the network with 50 hidden units. This is a big improvement in time as compared to [10]. Another advantage of using HF is that these large networks were trained without any layer-wise unsupervised pre-training.

Table 2 shows the results on the genre classification task on the ISMIR dataset. We note that the rectifier network with 500 hidden units and Dropout provides the best classification accuracy of 73.46%. We performed tests on the train/development split provided for the Audio Description Contest [1]. We repeated the classification experiment with MFCC and PMSC features for comparison. The classification accuracy with MFCCs was 67.21% and with PMSCs was 68.45%. In [21], the authors use several hand-crafted features as input to the same classification task. Although the results can't be directly compared, our results are of the same order if not better than the results achieved by using several hand-crafted features.The above results validate the claim that the learnt features capture information from the audio data that can be used for other tasks.

4. CONCLUSIONS AND FUTURE WORK

In this paper we describe better ways to learn robust features for MIR tasks using Deep Neural Networks. We show that training time can be reduced significantly by using ReLUs and HF. Both the rectifier and sigmoid networks are trained from randomly initialized weights without any pre-training. Once the features are learnt, the feature extraction does not take very long, which makes this approach practical for large datasets. In the future, we would like to apply features learnt from neural nets to improve performance of tasks like artist identification and music transcription. We would also like to explore transfer learning and unsupervised learning algorithms to try to leverage unlabelled data to learn useful features for MIR tasks.

5. REFERENCES

[1] ISMIR 2004: Audio Description Contest. http://ismir2004.net/genre_contest/, 2004.

[2] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[3] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010. Oral Presentation.

[4] James Bergstra, Norman Casagrande, Dumitru Erhan, Douglas Eck, and Balázs Kégl. Aggregate features and AdaBoost for music classification. *Machine Learning*, 65(2-3):473–484, 2006.

[5] James Bergstra, Michael I Mandel, and Douglas Eck. Scalable genre and tag prediction with spectral covariance. In *ISMIR*, pages 507–512, 2010.

[6] Christopher M Bishop. *Neural networks for pattern recognition*. Oxford university press, 1995.

[7] George E Dahl, Tara N Sainath, and Geoffrey E Hinton. Improving deep neural networks for LVCSR using rectified linear units and dropout. In *Proc. ICASSP*, 2013.

[8] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

[9] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier networks. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics. JMLR W&CP Volume*, volume 15, pages 315–323, 2011.

[10] Philippe Hamel and Douglas Eck. Learning features from music audio with deep belief networks. In *ISMIR*, pages 339–344. Utrecht, The Netherlands, 2010.

[11] Philippe Hamel, Simon Lemieux, Yoshua Bengio, and Douglas Eck. Temporal pooling and multiscale learning for automatic annotation and ranking of music audio. In *ISMIR*, pages 729–734, 2011.

[12] Mikael Henaff, Kevin Jarrett, Koray Kavukcuoglu, and Yann LeCun. Unsupervised learning of sparse features for scalable audio classification. In *ISMIR*, pages 681–686, 2011.

[13] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[14] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. Technical report, University of Toronto, 2012.

[15] Eric J Humphrey, Juan P Bello, and Yann LeCun. Feature learning and deep architectures: new directions for music informatics. *Journal of Intelligent Information Systems*, 41:1–21, 2013.

[16] Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1106–1114, 2012.

[17] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 609–616. ACM, 2009.

[18] James Martens. Deep learning via Hessian-Free optimization. In *Proc. ICML*, pages 735–742, 2010.

[19] James Martens and Ilya Sutskever. Learning recurrent neural networks with hessian-free optimization. In *Proc. ICML*, pages 1033–1040, 2011.

[20] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

[21] Carlos N Silla Jr, Alessandro L Koerich, and Celso AA Kaestner. Improving automatic music genre classification with hybrid content-based feature vectors. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, pages 1702–1707. ACM, 2010.

[22] Bob L. Sturm. An analysis of the GTZAN music genre dataset. In *Proceedings of the second international ACM workshop on Music Information Retrieval with user-centered and multimodal strategies*, MIRUM '12, pages 7–12, New York, NY, USA, 2012. ACM.

[23] Yichuan Tang. Deep learning using linear support vector machines. In *Workshop on Representation Learning, ICML, 2013, Atlanta, USA*, 2013.

[24] George Tzanetakis and Perry Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, 10(5):293–302, 2002.