Predict geographical origin of music using neural network utilizing a priori knowledge of population density

Jing Yang YANGJING@MIT.EDU
Yinan Cai YNCAI@MIT.EDU
Yixiang Liu YXLIU@MIT.EDU

Abstract

We report results of predicting geographical origin of music using neural network (NN) regression. We highlight on the customized loss function for NN which substitute mean squared error with great circle distance and also include information of global population density distribution. We proposed and experimented algorithm combining clustering and NN regression. We give a detailed comparison between the prediction performance given by linear regression and NN with different modified versions of loss function. It is clearly shown that by utilizing a priori knowledge of population density, the network suffers less from overfitting and predicting in uninhabited areas. The training accuracy is improved by 70% and test accuracy by 10% with the modified NN. We also tested the trained NN on out-of-dataset samples where we generated music features from self-collected music tracks. The prediction remains valid and confirms the quality of our model.

1. Introduction

We used the `Geographical original of music data set` [1] . This dataset contains audio features extracted from 1059 traditional music tracks and also the geographical origin of the file expressed in forms of longitude and latitude. This dataset has at least ten tracks from each country. Number of tracks from each country is summarized in *Figure* 1. The goal is to learn the relationship between the features and the geographical origins using the training data and predict the geographical origin for any held-out data.

Each geographical location is represented by its longitude and lattitude (ϕ_i,λ_i). For simplicity, the geographical distance between two points are calculated by approximating earth as a perfect sphere with radius R = 6,371 km. The great circle distance between two points (ϕ_i,θ_i) and (ϕ_j,θ_j) can be subsequently calculated as

$$d(i,j) = 2R\arcsin(\sqrt{a}), \tag{1}$$

where $a = \sin^2(\frac{\phi_j-\phi_i}{2}) + \cos\phi_j\cos\phi_i\sin^2(\frac{\theta_j}{\theta_i})$. The 68 audio features are extracted with `MARSYAS` [2] including Mel-frequency cepstral coefficients (52), zero crossing rates (4), spectral centroid (4), spectral rolloff (4) and spectral flux (4).

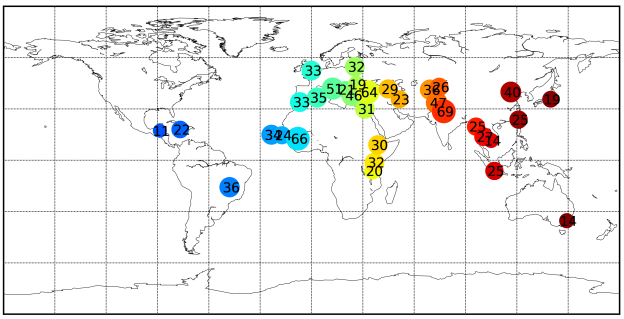


Figure 1. Spatial distribution of samples in the whole dataset. Numbers shown on point represent the number of tracks from each country.

2. Contribution

We highlight on two major contributions: (1) customized loss function penalizing predictions in low population density area; (2) combining clustering method with NN regression.

2.1. Customized loss function

The input is the feature of the music and the output is the geographical coordinate. This prediction problem can be formulated as a classification problem or a regression problem. If we choose to do classification, the model won't be general enough for an arbitrary held-out data considering that the training set doesn't

cover all the countries. Regression allows for continuous prediction and gives us a more general model. However continuous prediction means that the origin could be predicted into ocean or desert. Intuitively, the geographical origin of a music file should be in the region where the population density is large so the culture is rich enough to breed music. In this project we used a customized loss function based on the great circle distance utilizing the population density.

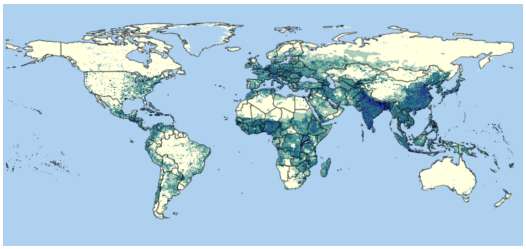


Figure 2. Gridded population density of the world in 2015.

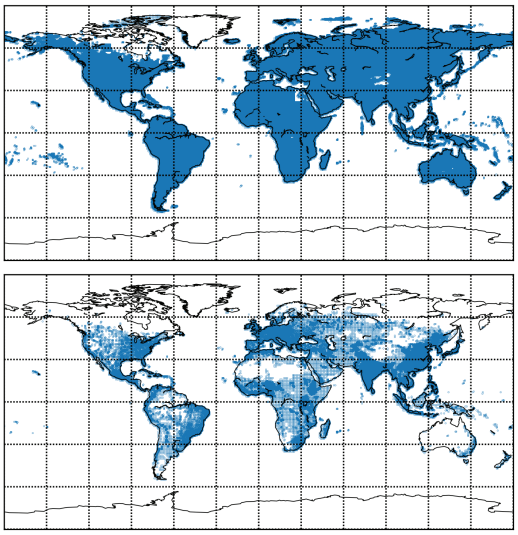


Figure 3. One-point representation of the population distribution before and after thresholding. The upper panel shows all the points in the gpw-v4 dataset. The lower panel only shows the points where the population count is larger than 10,000.

We used the Gridded Population of the World, v4 (gpw-v4) [3] (see *Figure* 2). In this dataset, the population in each $0.1^\circ \times 0.1^\circ$ square on the global raster surface is given. We simplified the gpw-v4 data using 'one-point representation' where each $0.1^\circ \times 0.1^\circ$ square with population count above a certain threshold is represented as a single point. The upper panel of *Figure* 3 shows all the points in gpw-v4 and the lower panel of *Figure* 3 only shows the points where the population count is larger than 10,000. We can see that after thresholding the density of points in *Figure* 3 (lower panel) is a good approximation of the true population density in *Figure* 2.

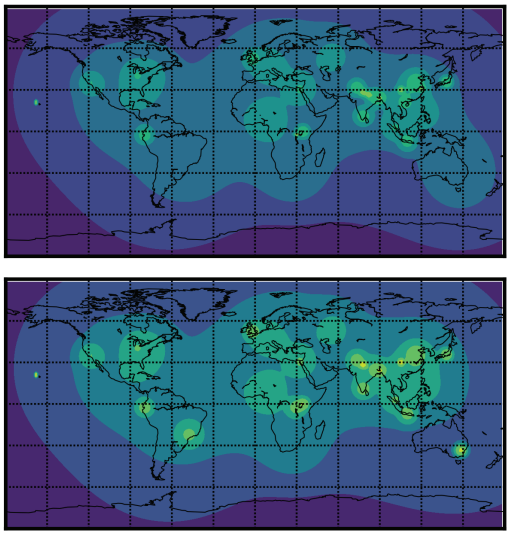


Figure 4. Probability density of the Gaussian mixture model fitted to the one-point representation after thresholding. The bright color denotes higher probability density. 60 Gaussian components were used and each component has its own single variance. The upper panel is the original Gaussian mixture model and the lower panel is the adjusted Gaussian mixture.

After the thresholding, we fitted a mixture of Gaussian model to these data points using `sklearn`[5]. 60 Gaussian components were used and each component has its own single variance. The original probability density of the Gaussian mixture model is shown in *Figure* 4 (upper panel). We can see that the probability density near Australia and South America is relatively low and it is not surprising because the density of points (see *Figure* 3 (lower panel)) at those two areas are relatively low. However as long as the population density in a certain area is above a threshold, it is possible to be the origin of music regardless of its absolute population density. To overcome this bias in Gaussian mixture model we manually adjusted the probability density near Australia and South America to make the intensity of the two peaks comparable with the other peaks. The adjusted Gaussian mixture was used in the loss function.

We tried two different loss functions using the Mixture

of Gaussian population density:

$$L = \frac{1}{N}\sum_{i=1}^{N}\left(d(\mathbf{y}_i^{\mathrm{t}}, \mathbf{y}_i^{\mathrm{p}}) - \alpha \mathrm{Prob}(\mathbf{y}_i^{\mathrm{p}})\right) \quad (2)$$

$$L = \frac{1}{N}\sum_{i=1}^{N}\left(d(\mathbf{y}_i^{\mathrm{t}}, \mathbf{y}_i^{\mathrm{p}}) \cdot (1 - \alpha \mathrm{Prob}(\mathbf{y}_i^{\mathrm{p}}))\right) \quad (3)$$

where $\mathbf{y}_i = (\phi_i, \lambda_i)$ represents the geographical coordinate. $\mathbf{y}_i^{\mathrm{p}}$ and $\mathbf{y}_i^{\mathrm{t}}$ represent the predicted and true location of the sample respectively. $d(\mathbf{y}^{\mathrm{t}}, \mathbf{y}^{\mathrm{p}})$ represents the great circle distance between $\mathbf{y}^{\mathrm{t}}$ and $\mathbf{y}^{\mathrm{p}}$ calculated using *Equation* (1). $\mathrm{Prob}(\mathbf{y}_i)$ denotes the probability density at point $\mathbf{y}_i$ in the adjusted Gaussian mixture model. The intuition behind the two loss function is when $\mathbf{y}_i^{\mathrm{p}}$ is at the region where the population density is low, i.e. Prob small, the loss will be high. α represents relative importance of low population penalty to error distance penalty.

2.2. Combination of clustering and NN regression

Clustering can serve as a preliminary step to separate data and help to reduce prediction variance because we can train different neural network for data points in different clusters, the scheme is plotted in *Figure* 5. For testing purpose, we first predict which the cluster the test point belongs to and then use the corresponding Neural Network to predict origin of the music.

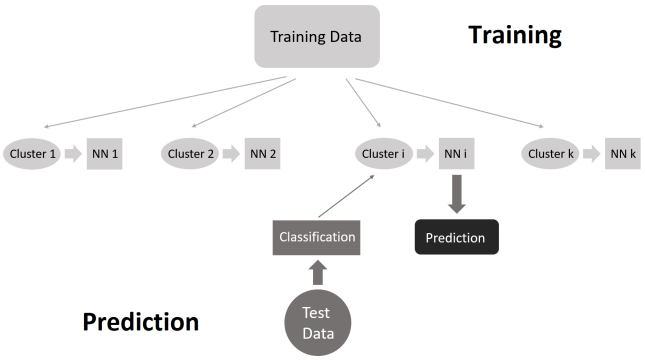


Figure 5. Scheme of combining clustering and NN regression.

3. Linear regression

We performed linear regression on the 68 features for predicting longitude and lattitude as a benchmark test using `sklearn`[5]. 80% samples from each country in the original dataset are randomly selected as training set. The remaining 20% samples are used as test set. We use this test accuracy as comparison metric between different regression method. *Figure* 6 shows the averaged distance between predicted location and true location using ridge regression and LASSO with different regularizing weight λ. Over the tested λ range, we do not observe a significant raise in test error comparing to training error, implying there is no overfitting concern. Consequently, we observe that the performance does not depend significantly on the regularization term. The test error has a minimum error of about 3,824 km.

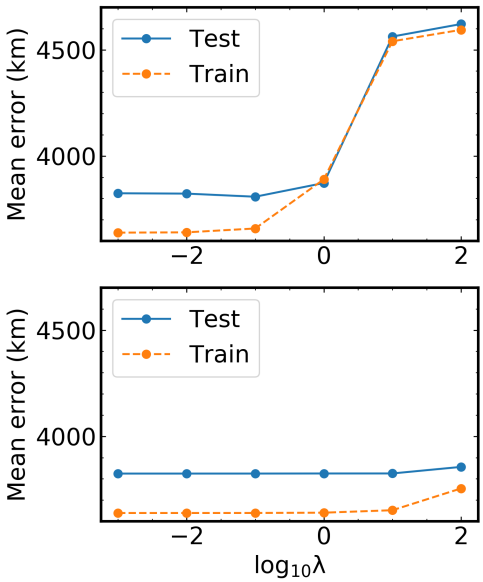


Figure 6. Averaged error of predicted location and true location by ridge regression (upper) and LASSO (lower).

In *Figure* 7 we look at the spatial distribution of predicted locations versus true locations on the test set projected onto the world map. For visualization purpose, we plot east, middle and west part of the map separately, which correspond to Americas, Europe and Africa, and Asia and Australia. Qualitatively, the linear model does capture the trend of spatial distribution from east to west. However, we can clearly see that it behaves poorly on samples located close the the margins of the map. In particular, samples in Americas are predicted towards Africa and samples in Australia are predicted towards Asia. The predicted locations show a clear clustering towards center of the map, which indicate that the simple linear model cannot capture the individual features of each country. This is a direct consequence for using mean squared distance as loss function.

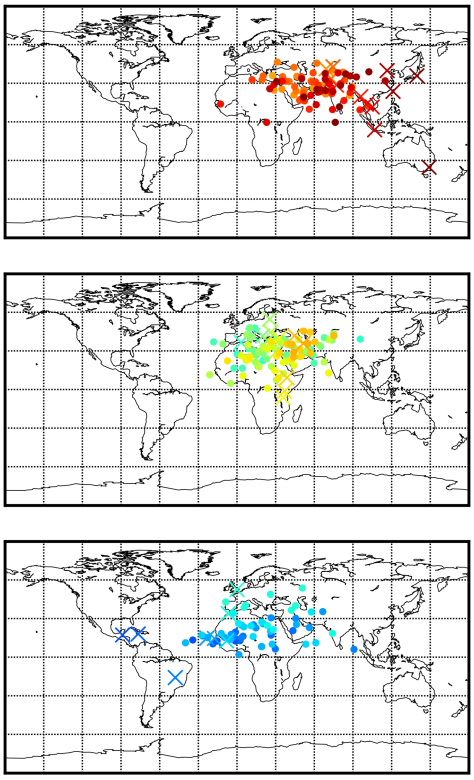


Figure 7. Spatial distribution of true locations (crosses) and predicted locations (dots) from the test set plotted with matching colors. The prediction is made using LASSO ($\lambda = 10^{-3}$). For better visualization purpose, the points are separated into three parts from west, middle to east and color-coded accordingly.

To see which features are important and which are irrelevant in the prediction, in *Figure* 8 we show the resulted weights w_1 and w_2 for the 68 features from LASSO as a function of regularization parameter λ. As λ increases, LASSO will rule out relatively unimportant features. Here it is easily observable that most weights die down with increasing penalty but a few persist. This survival analysis helps to rule out some irrelevant features. We color-coded the features to separate them into the five groups: Mel-frequency cepstral coefficients (MFCC), zero crossing rates (ZCRS), spectral centroid (CTD), spectral rolloff (RLF) and spectral flux (FLX). Out of the ten most persisting features given by survival analysis, there are one from ZCRS, one from RLF, one from FLX, and none from CTD, indicating a weak dependence on spectral centroid features. All other persisting features are Mel-frequency cepstral coefficients.

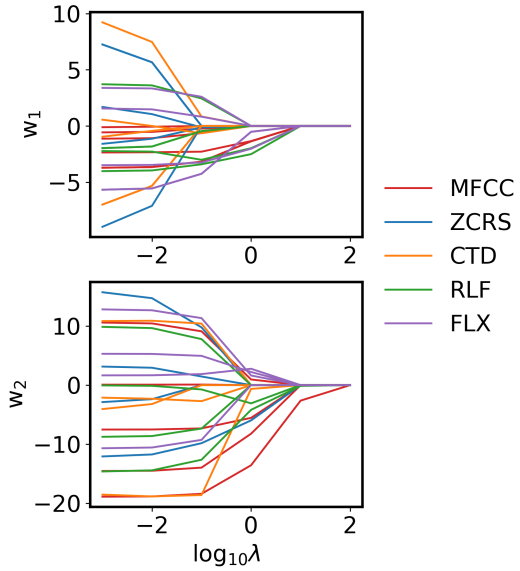


Figure 8. Evolution of weight w_1 and w_2 with regularization parameter λ. The lines are color coded so that lines with the same color correspond to features from the same group. For visual clearance only the four most persisting features from MFCC are plotted.

4. NN with customized loss function

In this section we discuss the performance of NN regression with variation in loss function. We start with simple, conventional NN with mean square error as implemented in `keras`[6] and observe that this model does not outperform linear regression by much. We customized the loss function replacing mean squared error with great circle distance and obtained better predictive power on samples lying close to the margin of the map. However, this is accompanied by increasing amount of samples being predicted in ocean. Finally, we demonstrate that by using loss function utilizing population density information, we get a more reasonable prediction with predicted locations drawn closer to land and a lower mean error distance. For comparing different loss function, NNs trained in this section are all with two hidden layers, each with 50 units and RELU activation. For penalizing uninhabited area, we used the loss function as formulated in *Equation* (2) with $\alpha = 10^4$. Loss function is implemented utilizing tensor operations in `Tensorflow`[7]. In the result section we report on the effect of varying NN structure and penalty weight. In all cases, early stopping is used where the training stops when validation loss stops to decrease for reporting the validation set prediction. A longer training of 100 epochs is also performed for demonstrating the evolution of training

and validation loss.

4.1. Conventional NN with mean squared error

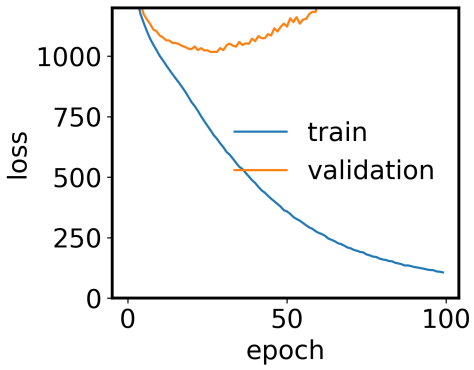


Figure 9. Evolution of training loss and validation loss with mean squared error as loss function.

We first used mean squared error as loss function for NN training. In *Figure* 9 we can observe from the evolution of training and validation loss that, though the training error can be reduced to much lower value comparing to linear regression, the model easily suffers from overfitting. We obtain mean distance error of 2,430 km on training set and 3,773 km on validation set. Looking at the spatial distribution of predicted locations from validation set in *Figure* 10, we observe that even though the training error becomes much lower comparing to linear regression, performance on validation set is not improved by much. The clustering effect is slightly reduced comparing to *Figure* 7. However, the model remains unpredictive for samples from Americas and Australia, i.e. regions isolated from the center continent. This also agrees with our interpretation on the limitation of mean squared error loss function.

4.2. NN with great circle distance as error function

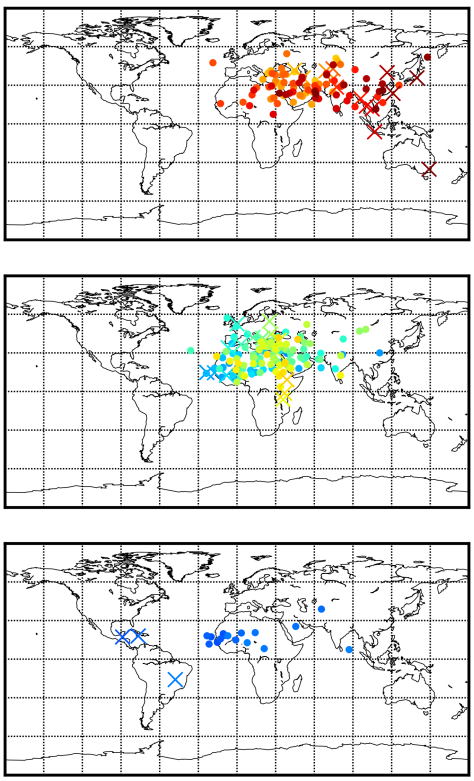


Figure 10. Spatial distribution of true locations (crosses) and predicted locations (dots) from the validation set plotted with matching colors. The prediction is made using NN with mean squared error loss function.

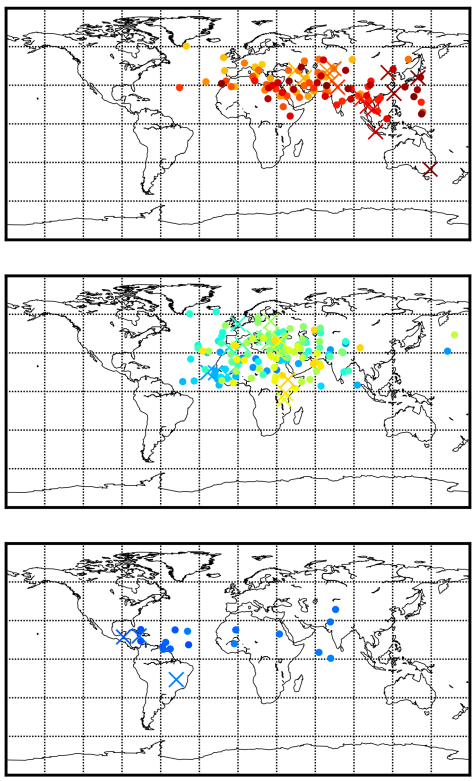


Figure 11. Spatial distribution of true locations (crosses) and predicted locations (dots) from the validation set plotted with matching colors. The prediction is made using NN with great circle distance loss function.

One straightforward modification of the conventional NN is to replace mean squared error with great circle distance, as calculated in *Equation* (1). This loss function eliminates clustering towards center of map and should behave better in predicting samples from isolated continent. Here we obtain a mean distance error of 3,639 km, with training error decreasing to less than 2,000 km. In *Figure* 11, we see that the modified loss function gives much better performance in isolated continents, providing some predictive power over Americas and Australia. This is the major source of the reduction in validation error. However, we also see a scattering of predicted locations in Africa, Europe and Asia, driving more points towards ocean area.

4.3. NN utilizing a priori knowledge of population density

In the previous two sections, we have concluded that using great circle distance can increase model's predictive power on isolated continents while increasing points predicted in uninhabited area. This observation motivates us to include a prior knowledge of population density. In *Figure* 12 we plot the evolution of loss function with population density term added. A great improvement here is that while the training loss continues to reduce with more epochs, validation performance suffers less from overfitting comparing to *Figure* 10. It can be seen that the modified model keeps the advantage of using great circle loss function in preventing overfitting, while making the evolution of validation loss even more smooth. The obtained mean error distance on the validation set is 3,533 km.

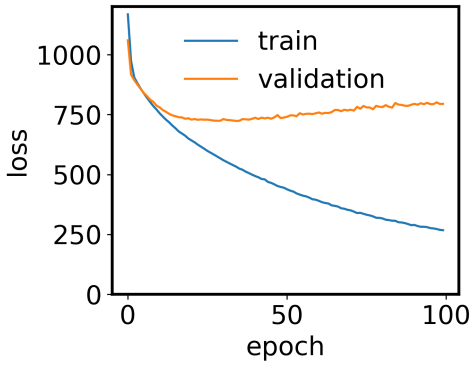


Figure 12. Evolution of training loss and validation loss with customized loss function utilizing population density.

Comparing the spatial distribution of predicted validation samples in *Figure* 13 and *Figure* 11, we see a much better prediction power in isolated islands and continents. The model has more capability of handling the complicated land shape of the world map.

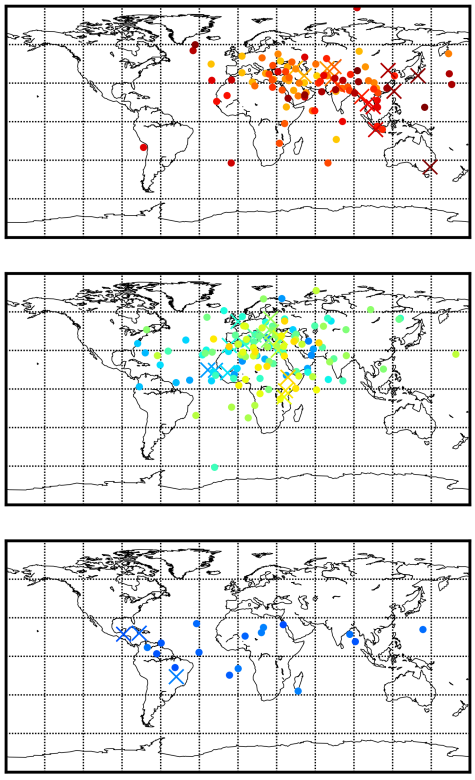


Figure 13. Spatial distribution of true locations (crosses) and predicted locations (dots) from the validation set plotted with matching colors. The prediction is made using NN with with customized loss function utilizing population density.

5. Cluster

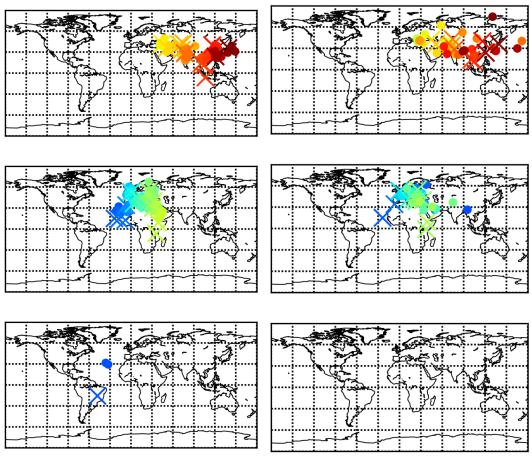
We also tried to cluster the data and train neural network individually for each cluster to decrease variance. The scheme of proposed process is shown in *Figure* 5.

Theoretically, this should work because the clustering algorithm should combine similar data points together and the neural network will be more specific for corresponding data points. However, in this particular data set, it doesn't work better than training a single neural network for the whole data set. The best average distance for unseen test set we got from this algorithm is 3,539 km, which is not significantly different from training single neural network on the whole dataset. We used 4 clusters, one set of average distance for clusters are summarized in *Table* 1, the average distance for whole test set is 3,706 km in this particular

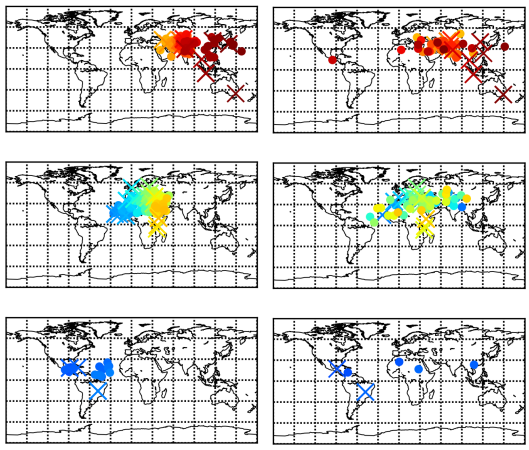
result.

Table 1. Average Distance in Each Cluster.

Cluster Index	average distance (km)
0	2,661
1	4,430
2	3,978
3	2,979



(a) Cluster 0 Train (b) Cluster 0 Test



(c) Cluster 2 Train (d) Cluster 2 Test

Figure 14. Plots of train and test data for clusters 0 and 2

This algorithm suffers a lot from lack of training data points. We only have 1059 data points in total, so each cluster will have in average 200 data points with 4 clusters. In this 200 data points, we still need 10 % for validation purpose. So the number of training data for each cluster is only about 180. The results for cluster 0 and cluster 2 training and test sets are plotted in *Figure* 14. Cluster 0 has average test set distance only 2,661 km while cluster 2 has average test set distance 4,430 km. From the plots, we can see that the prediction for music from South America in cluster 2 test sets has a very high variance and this is the reason that cluster 2 test set has higher average distance. The reason comes from two parts: first, we don't have enough music from South America in corresponding training set to capture music features of the location. Moreover, we don't have many countries in South America, so the location in South America is not continuous in training set. Therefore, we don't have enough neighbor locations as reference to compensate the lack of training points. The same situation also happens to music from Australia. The music from Australia tends to be predicted to South East Asian. Another difficulty of the this algorithm in this data set is that the clustering process didn't narrow the range of location. Music from same country can be clustered to different clusters. With all these difficulties, the performance of this algorithm is comparable to training a single neural network for all data points. This indicates that this algorithm can be more useful in other datasets where clustering can help narrow the range of output or we have enough data available for each cluster.

6. Results

6.1. Weight of population density penalty

We tested different architectures and different hyperparameters. The average distance on test set and the fraction of points predicted into ocean are summarized in *Table* 2. The hyperparameter α controls the relative importance of population density to distance error. With large α, we get predicted sample clustering in main continent and thus a significant reduction in the percentage of points predicted in ocean. However, this comes with raise in average distance error.

Table 2. Mean error distance and percentage of points predicted in ocean with varying penalty α.

α	average error (km)	ocean fraction
10^2	3,923	0.324
10^4	4,476	0.271
10^6	6,101	0.022

Here we report our best result in *Figure* 15. The best result was obtained using a NN with 4 hidden layers and each with 200 nodes and ReLU activation. Loss function *Equation* (3) was used and $\alpha = 4 \times 10^3$. The training error and validation error are 622 km and

3,496 km.

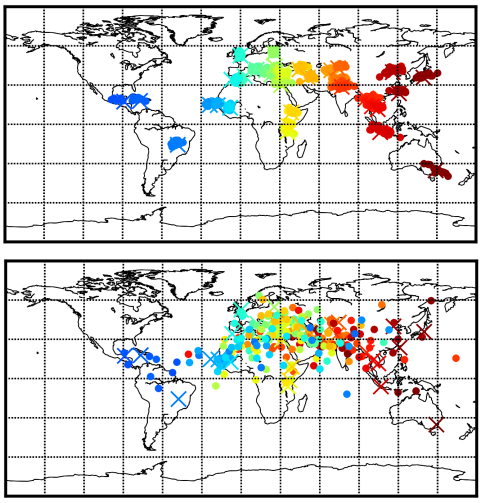


Figure 15. Spatial distribution of true locations (crosses) and predicted locations (dots) using our best model. Upper panel: train set. Lower panel: validation set.

6.2. Test on held-out data

The training set are all traditional music files from different countries. One topic of interest is to apply the trained model on modern music and study the historical influence from different areas. We tested our model on a collection of 11 jazz music tracks downloaded from free music archive [9]. We extracted their features using `MARSYAS`. To be consistent with the training set [4], we also performed feature standardization using $x' = \frac{x-\bar{x}}{\sigma}$, where $\bar{x}$, σ is the mean and standard deviation of a certain feature averaged over different data points.

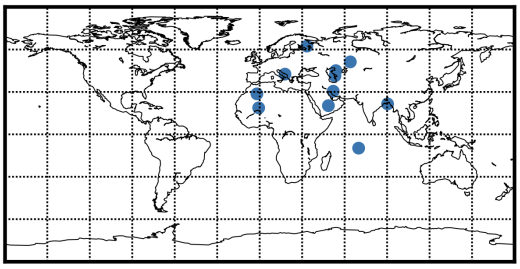


Figure 16. The predicted origin of the 11 Jazz music tracks.

The predicted locations are shown in *Figure* 16. Jazz is a music genre that originated in African-American communities in United States. It then emerged in the form of independent traditional and popular musical styles, all linked by the common bonds of African-American and European-American [8]. It is interesting to see that the predicted origins are mainly at Africa, Europe, and Middle East.

7. Conclusion and Discussion

In this report, we explored the predictive power of neural network regression on geographical origin of music. By using customized loss function with great circle distance and adding penalty for predictions in low-population-density area, we effectively reduced overfitting of the model. The best-performing model reduced training error by over 80% and validation error by over 10%. The test on held-out data also verifies the transferability of the model in revealing historical regional influence on modern music.

We would also like to discuss possible error sources and improvement of the model. First, the dataset assigns the location of capital city of each country as the true location of the sample. This might not be representative for regional culture differences, especially for large-area countries. Second, the Gaussian mixture model generated from gridded population density could lead to potential artifact in the prediction. Because of the symmetry of Gaussian model, sea area that is close to seaside city with high population density is not penalized. Third, the low population penalty term is non-convex and has a complicated landscape. We expect the current model suffers from trapping in local minima. Fine tuning of the gradient descent method could potentially help us get out of the local minima and improve the performance of the current model.

8. Division of Labor

Jing Yang performed linear regression modeling and implemented customized loss function using `keras`. Yinan Cai tested on choosing different hyperparameters and implemented clustering method. Yixiang Liu generated and tuned mixture Gaussian model of population density, collected and tested on held-out data. All authors participated in discussing ideas, training the networks, and writing the report.

The codes can be found here: `https://github.com/jyang2009/music-prediction`.

References

[1] http://archive.ics.uci.edu/ml/datasets/geographical+original+of+music

[2] MARSYAS User Manual. http://marsyas.info/doc/manual/marsyas-user/bextract.html#bextract

[3] Center for International Earth Science Information Network - CIESIN - Columbia University. 2016. Gridded Population of the World, Version 4 (GPWv4): Population Density. Palisades, NY: NASA Socioeconomic Data and Applications Center (SEDAC). http://dx.doi.org/10.7927/H4NP22DQ.

[4] F. Zhou, Q. Claire, R. D. King, Predicting the Geographical Origin of Music. 2014 IEEE International Conference on Data Mining.

[5] F. Pedregosa et al, Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12.Oct (2011): 2825-2830.

[6] https://github.com/fchollet/keras

[7] M. Abadi et al, TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[8] https://en.wikipedia.org/wiki/Jazz

[9] http://freemusicarchive.org/