Estimations of land surface temperature from urban morphology by use of different machine learning methods

Qihao HE

Abstract

Both Neural Network (NN) and Random Forest Regressor (RFR) models are common tools in the area of machine learning. Based on the result of our previous study, this study will examine the models' performance in terms of both accuracy and efficiency in predicting Land Surface Temperature (LST) of Hong Kong Islands. To improve the performance of the models, we designed a feature extraction method, tried a special way of train-test set splitting, and corrected a minor shift of the temperature image compared with other feature images using QGIS. The RFR has outperformed the NN with the optimal network structure among the tested configurations with two inner layers.

1. Introduction

As an important indicator of the Urban Heat Island (UHI) phenomenon, Land Surface Temperature (LST) has been an important topic in previous studies, many of which associated categorical variable Local Climate Zone (LCZ) urban morphology indicator with categorical LST [1]. Instead of categorical LST, continuous LST is the focus of this study. Similar to our last report, we will continue investigating the capability of machine learning models in predicting LST given various continuous features related to urban morphology. The models used in this study include the Feed-Forward Neural Network (FNN) model [2] and the Random Forest Regressor (RFR) model [3].

There are several key differences compared with our previous study. Firstly, we take a different approach to utilize the dataset. A cut that separates the train set and test set is made purposedly rather than randomly within the entire selected region so that the prediction result is more meaningful for future interpretation make improvements. Then, we extract more features for each grid unit of the map from its surroundings so that the accuracy of the RFR model is improved. Lastly, we investigated the feature maps in QGIS that discovered an anomaly where the temperature map is shifted upwards.

2. Methodology and Data

2.1. Region of Interest

This study focuses on the Hong Kong Islands Region (HK), which is part of the Pearl River Delta Region (PRD). We plan to train the models on the parts of the region excluding HK in PRD, where the exact range of this train set is $\boldsymbol{X}$ = [690000,773110] (m), $\boldsymbol{Y}$ = [2510000, 2580000] (m) in World Geodetic System (WGS) / Universal Transverse Mercator (UTM) zone 49N projection system. After the models are trained, they can be applied in the HK test set with range: $\boldsymbol{X'}$ = [790000, 850000] (m), $\boldsymbol{Y'}$= [2450000,2490000] (m) in the same projection system.

In this specially designed train-test splitting, we defined two disjoint sets $S = \{data\ point\ p\ in\ PRD \mid p \in [\boldsymbol{X}, \boldsymbol{Y}]\}$ and $S' = \{data\ point\ p\ in\ PRD \mid p \in [\boldsymbol{X'}, \boldsymbol{Y'}]\}$. The train set data points belong to set S and test set data points to belong to set S', this is significantly different from doing a random train-test splitting with points in both sets located in the same geographical range, which is common in practice, for example, the *.train_test_split()* function in the python scikit-learn (sklearn) library [4].

This way, we are in a similar scenario to that of "***Zero-Shot Learning***" more frequently seen in the area of classification problems [5]. The idea is similar in the sense that we treat the training result of our model more generally, even if the prediction result may not be as good compared with the normal way of train-test splitting, it is a more generalized result, and we can reveal more details from the prediction LST image of HK.

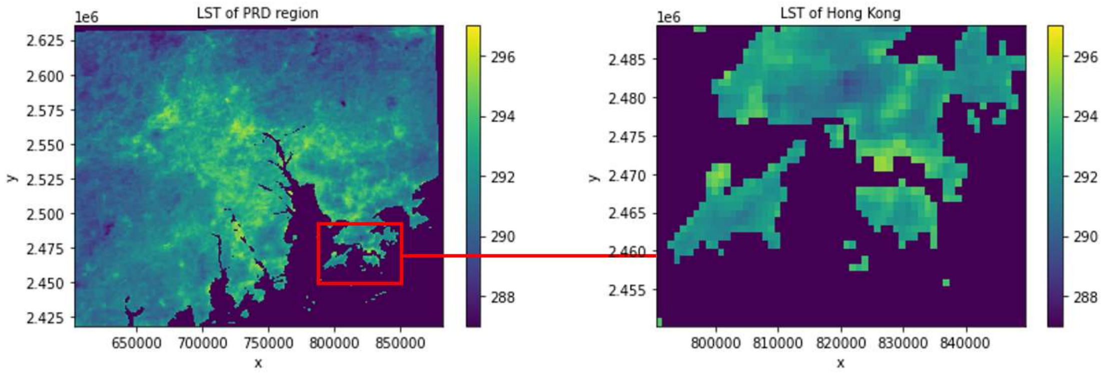

Figure 1. Area of Interest, the left image is the PRD region, with the red rectangle region the right image, which is the Hong Kong islands

2.2. Data Preparation

2.2.1. Filtering and Adjustment of Data

Similar to our previous study, the data used for the two models are represented in a large 2D map of PRD with 218*281 grids. This dataset is composed of two parts, the label variable, which is the learning target: LST, and 6 kinds of training variables: **impervious surface area (ISA)**, **Building plan area fraction ($\boldsymbol{\lambda_P, or\ lp2}$), Building surface area to plan area ratio ($\boldsymbol{\lambda_B, or\ lb2}$)**, **Area-weighted building height (AH, or ah2)**, **UN WPP-Adjusted Population Density, altitude and 2019 Global Forest Canopy Height Data**, each grid of the map contains values for all 7 variables, which is represented in a pandas data frame with 7 columns, and 218*281 rows (entries or data points).

Firstly, it can be noticed that in the dataset we used, there are invalid points out of normal ranges of the 7 variables, so we designed a filter that filters data points that satisfy any one of the following conditions: 1. Temperature < 250; 2. ISA<0 or ISA > 100; 3. Population Density < 100; 4. $\boldsymbol{\lambda_B}$ < 10000; 5. Forest Height < 0 or Forest Height >= upper bound (101). This filter is shown in our tests to be the most stable and reliable one.

Secondly, compared with our previous study, the two features X and Y coordinates values that correspond to longitude and latitude are removed from our consideration in this study. The reason is that these two features are the underlying bases for us to make the train-test splitting. Since the patterns the models have learned in set S is alien to that of S', where S and S' have disjoint X, Y ranges as introduced in part 2.1. Our experiments also observed that the predicted LST values could deviate significantly from the actual LST values if X, Y features are included. Thus, we consider it is better only to include more general features related to urban morphology in this study and discarded X, Y features.

Thirdly, by using the open-source software QGIS, we investigated the coastlines of all 7 feature images overlaid on a Google map. Among them, only the coastlines in the temperature image are observed to have shifted upwards, with the rest fitting the google map quite nicely. As a result, we will be correcting the temperature image by shifting it downwards.

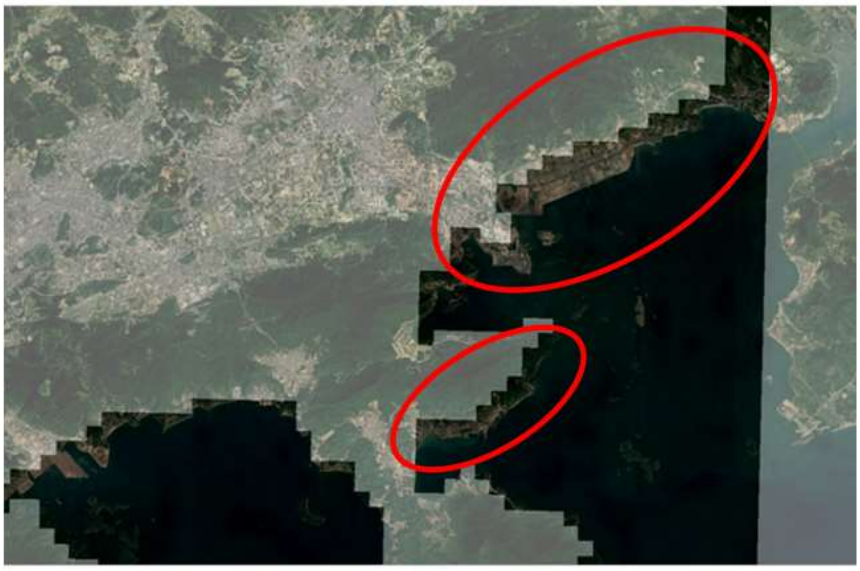

Figure 2. Temperature image on QGIS where observed shifts upwards are red circled

2.2.2. Feature Selection based on Spatial Autocorrelation

According to part 2.2.1, our dataset contains only 6 features (excluding temperature) that indicate the status of each grid in the map. However, adding 8*6 more features from the 8 surrounding grids (4 adjacent grids and 4 corner grids) to each grid being investigated may improve our model.

The idea for us to associate the surrounding grids' feature values to the center cell is natural. From the perspective of common sense, certain urban structures may significantly influence the surrounding areas' temperature, for example, skyscrapers that may cast large areas of shadow on one side when the sun is on the other side of the building. From the perspective of ***Spatial autocorrelation*** [6], an idea that indicates "similar values are located near each other, while different values tend to be scattered and further away", we may group the grids' features in this way with one grid also containing the information from the surrounding grids so that the predictions given by the model will also naturally follow the pattern that "similar values are located near each other".

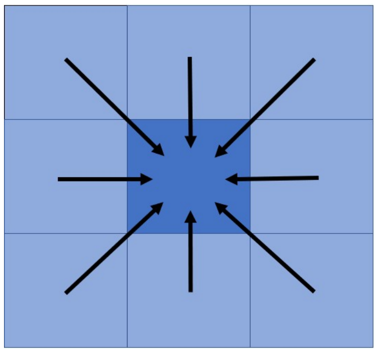

Figure 3. Extract features from surrounding grids to append to the central grid to make predictions

2.3. The Neural Network (NN) Model Structure

To produce estimated LST values, we first build a NN model with the same architecture as in our previous study, the Feedforward Neural Network (FNN) with Back Propagation (BP) Gradient Descent Based Algorithm [2] and the Mean Absolute Error (MAE) as the loss function.

In this study, since we want to examine the structure of a network more comprehensively, a set of reasonable hyper-parameters are tested out. As also described in our previous report, the number of parameters in an L-layer NN can be represented by the number of neurons in the entire network. For example, in layer l there are S_l neurons:

$$\prod_{l=1}^{L-1} S_l S_{l+1} + \sum_{l=2}^{L} S_l$$

Equation calculating the total number of parameters in a NN

Thus, this is a problem of searching the optimal configuration of an n-tuple $(S_1, S_2, \dots, S_n)$, if the model has n layers. To begin with, we may define a valid domain to conduct the searching. Firstly, according to 2.2.1, *filtering* of data reduces the size of the training set to 862, which means we can only establish 862 equations from these entries. Thus it is not reasonable to have several parameters more than 862. Combining this issue with the above ***Equation***, we cannot apply the 2.2.2 Feature Manipulation method that will easily make the model contain more than 1000 parameters. Secondly, considering efficiency, we fix our model to have only 2 dense layers and 2 **Rectifier (ReLU)** activation function (it is a nonlinear function $f(x) = \max(0, x)$ containing no additional parameters) layers that respectively follows each dense layer in the middle. The tested set of parameters for the two dense layers is in **Table 1**.

Table 1 The set of parameters to test

Type of network structure	A	B	C	D	E	F	G	H	I	J
#Neurons in First Layer	4	6	6	8	10	12	14	16	16	20
#Neurons in Second Layer	6	6	8	8	8	8	8	8	10	10
Total # of trainable parameters	57	79	95	121	147	173	199	225	261	321

2.4. The Random Forest Regressor (RFR) Model

The RFR model is a powerful ensemble ML model that provides accurate results in all sorts of regression and classification problems and is widely used in many research topics such as Temperature Downscaling [7], indoor pollutant concentration estimation [8], and Effects of Urban Form on LST [9] which is of similar purpose to our study. The RFR is a large collection of Decision Trees [10], where each decision tree is grown by random data points generated from the train set (bootstrap). The results of the collection of trees are aggregated to produce the final prediction of the forest [10], which helps the model avoid over-fitting problems, a property that helps in our case where train set and test set are disjointly designed in section 2.1.

In this study, we use the python **sklearn** library implementation of RFR [4]. Two key hyper-parameters must be defined before the model can be trained, the **n_estimators** (number of trees in the forest [4]) and **max_features** (The number of features to consider when looking for the best split [4]). From the previous study, the best recommended **max_features** value is the total number of features divided by 3 [11], which is consistent with our test results on the RFR with only 6 training features (before our Feature Extraction from surrounding grids is used). As for **n_estimators**, considering both performance and efficiency, we fix it as 500 that shows no significant difference from models using 1000 or 5000 trees in terms of prediction accuracy. But it requires much less time to complete training, which accelerates the training process and saved lots of time.

3. Result

3.1.MAE, Correlation Coefficient (CC) and Scatter Plots

Firstly, the ten NN structures' performance is compared in terms of MAE and CC. The values of MAE for all different configurations are in the range [0.738, 3.008], and the values of CC are in the range [0.282, 0.485]. The trend is that a more unstable MAE is observed when we decrease the number of parameters, where the MAE value of a model can be as big as 3.008 (8+8) and as small as 0.814 (4+6), while the larger network has relatively stable MAE values, ranging from 0.738 to 1.689.

As for CC, while middle-sized NNs perform relatively better than most smaller-sized networks (4+6 is an exception and we can see its performance compared with the best performing middle-sized network 16+8 in **Figure 4**, it can be seen that even 4+6 seems to perform well from its CC value that is 0.462, but it is incomparable with 16+8 having a marginally larger CC value 0.474 in terms of the scatter plot), while all larger networks do not have CC value greater than 0.4, and there is a trend of the increasing number of outliers in their predictions (**Figure 5** shows this trend). So we conclude from the three factors, MAE, CC, and scatter plots of prediction-real LST value, that a 16+8 structure is the optimal network structure that will be compared with the RFR.

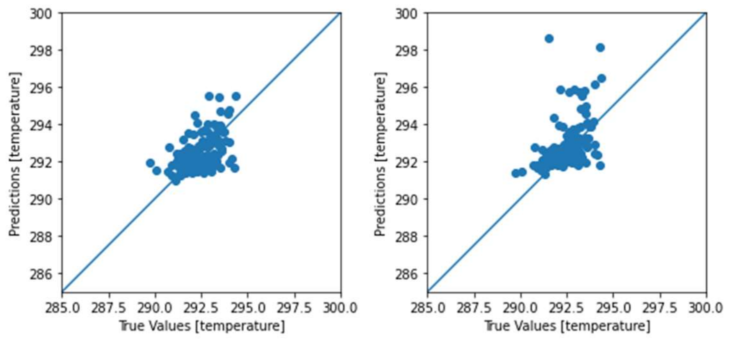

Figure 4. The left is 16+8 network's scatter plot of Prediction against Real Value, the right is 4+6 network plot

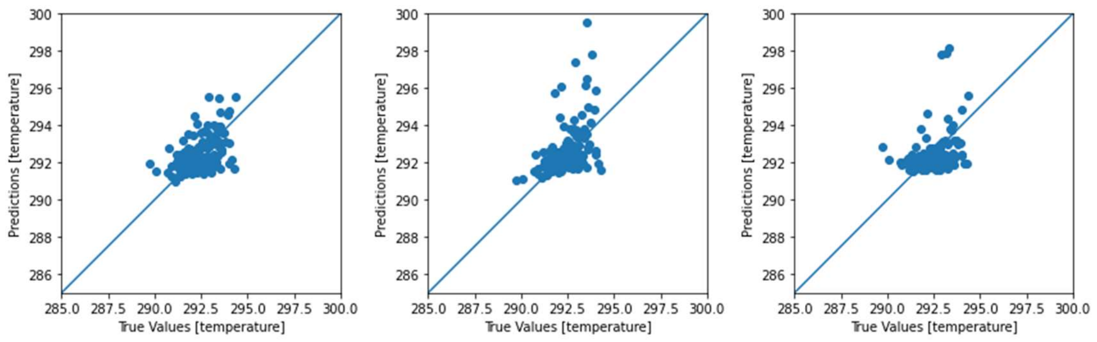

Figure 5. From left to right are respectively 16+8, 16+10, and 20+10 structures.

The RFR, in fact, outperforms the NN model not only in terms of MAE and CC, but also in terms of training time. The NN normally takes 5 mins to train, while RFR can finish within 10 seconds. The CC for the simplest RFR with the same 6 features that are used to train the 16+8 network is 0.6062106. This RFR's MAE is 0.571, also much lower compared with 0.738 of 16+8 NN. We can also observe from the scatter plots in **Figure 6** their difference.

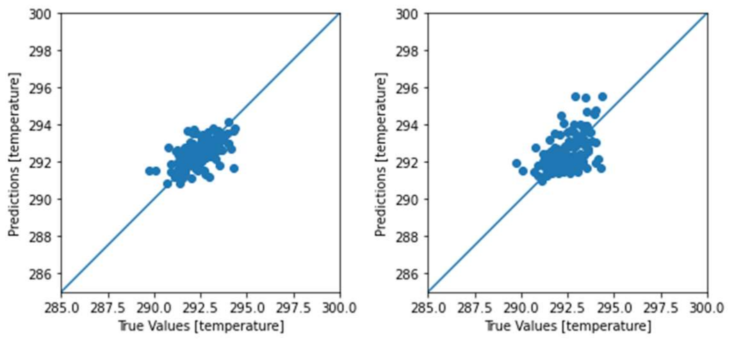

Figure 6. Left is the RFR model, right is 16+8 NN model

Now we can consider applying the feature extraction method (FE) and shift correction (SC) mentioned in section 2.2 on RFR. There are three cases: A. Without

FE and with SC; B. With FE and without SC; C. With both FE and SC. The three types of RFR models' MAE and CC values are shown in **Table 2** below:

Table 2 MAE and CC of the 3 types of RFR models

Type of RFR model	A	B	C
MAE	0.634	0.666	0.595
CC	0.617	0.670	0.645

Considering their MAE and CC values, we choose the C model which achieved a moderate CC that is smaller than only type A model, and the smallest MAE among all three adjusted models. We may see its scatter plot in **Figure 7** compared with the other two.

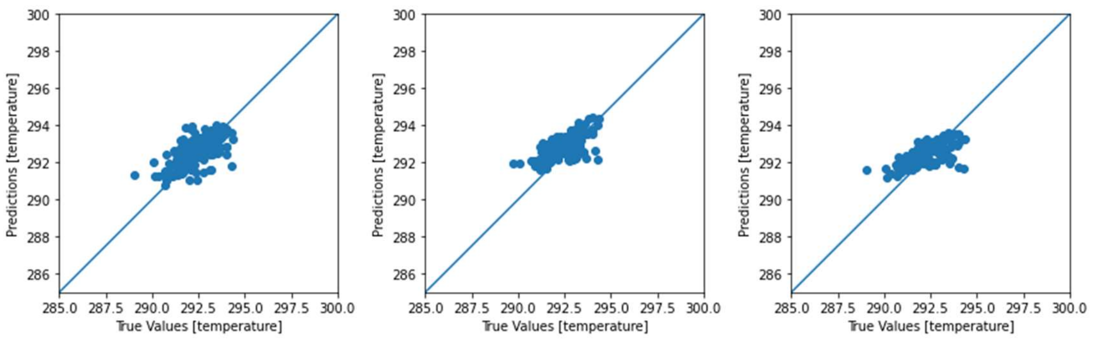

Figure 7. From left to right respectively models A, B and C, and it can be noticed that the points concentrate better to the line after applying FE, resulting in a better CC

3.2. Spatial maps of prediction

As a result of comparisons of MAE, CC, and scatter plots in section 3.1, we choose the RFR model with FE and SC applied on it to produce the final result, which is a heat map of Hong Kong Islands, containing only 155 points with valid inputs indicated by our *filtering* process specified in section 2.2.1. The result is in **Figure 8**. Different from our previous study, this one is producing predictions in the area different from the train set area as described in Section 2.1, and this map in **Figure 8** is more useful in terms that no training point is in the map.

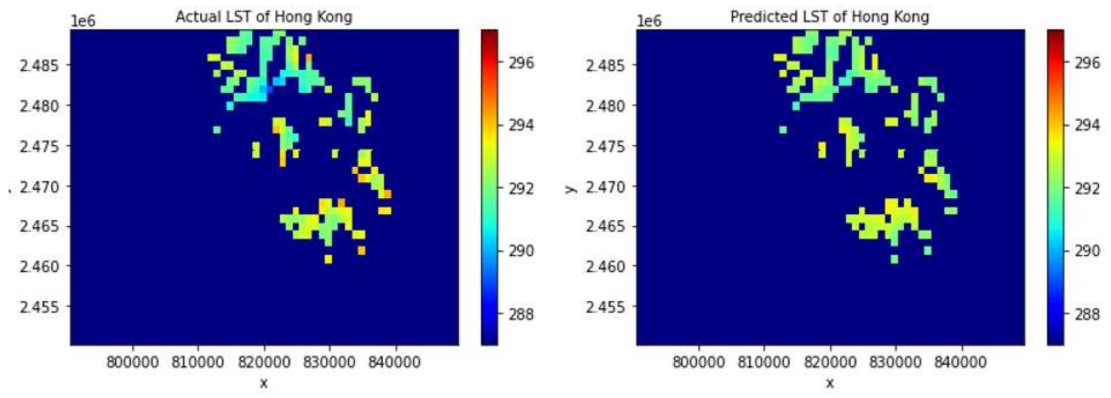

Figure 8. The left image is the actual LST of the 155 points, and the right image is the predicted LST of the 155 points

4. Conclusion and Future Works

This study attempts to go beyond our previous semester's work to make better and more reasonable predictions on continuous LST with the help of filtering invalid data points, extracting extra features from surrounding grids, and correcting the shifted temperature image. We chose the RFR model over NNs trained by data in the PRD region excluding HK to produce a prediction map in HK.

It is expected that the Feature Extraction method may be further improved, while we have made no assumptions in this study on how great the effects can be brought to the center grid by the surroundings, i.e., it is possible and reasonable to apply any transformations to the features of the surrounding grids before they are added to the feature set of the center grid, and the transformation we chose in this study is the most ordinary one $f(x) = x$, where we did not change the values for the features from surrounding grids when they are extracted.

Also, it will improve the prediction accuracy if more data points from more urban areas can be included. The model may generalize better given that limited data can lead to a biased result. It is hoped that future models will produce even more generalized and accurate results so that a more intricate relationship between urban morphology variables and LST may be unraveled.

References

[1] M. Cai, C. Ren, Y. Xu, K. K.-L. Lau, and R. Wang, "Investigating the relationship between local climate zone and land surface temperature using an improved Wudapt methodology – a case study OF Yangtze River Delta, china," *Urban Climate*, vol. 24, pp. 485–502, 2018.

[2] I. A. Basheer and M. Hajmeer, “Artificial neural networks: Fundamentals, computing, design, and application,” *Journal of Microbiological Methods*, vol. 43, no. 1, pp. 3–31, 2000.

[3] M. Belgiu and L. Drăguţ, “Random Forest in remote sensing: A review of applications and Future Directions,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 114, pp. 24–31, 2016.

[4] Pedregosa *et al.*, “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[5] P. K. Pushp en M. M. Srivastava, “Train Once, Test Anywhere: Zero-Shot Learning for Text Classification”, CoRR, vol abs/1712.05972, 2017.

[6] S. J. Rey, D. Arribas-Bel, and L. J. Wolf, “Geographic Data Science with python,” *Global Spatial Autocorrelation - Geographic Data Science with Python*. [Online]. Available: https://geographicdata.science/book/notebooks/06_spatial_autocorrelation.html. [Accessed: 20-Dec-2021].

[7] H. Ebrahimy en M. Azadbakht, “Downscaling MODIS land surface temperature over a heterogeneous area: An investigation of machine learning techniques, feature selection, and impacts of mixed pixels”, *Computers & Geosciences*, vol 124, bll 93–102, 2019.

[8] W. Che, A. T. Li, and A. K. Lau, “Estimating concentrations for particle and gases in a mechanically ventilated building in Hong Kong: Multivariate Method and Machine Learning,” *Air Quality, Atmosphere & Health*, 2021.

[9] Y. Sun, C. Gao, J. Li, R. Wang, en J. Liu, “Quantifying the Effects of Urban Form on Land Surface Temperature in Subtropical High-Density Urban Areas Using Machine Learning”, *Remote Sensing*, vol 11, no 8, 2019.

[10] L. Breiman, “Random forests”, *Machine learning*, vol 45, no 1, bll 5–32, 2001.

[11] A. Liaw, M. Wiener, en Others, “Classification and regression by randomForest”, *R news*, vol 2, no 3, bll 18–22, 2002.