Seeded transfer learning for road roughness regression

by

Hao Jiang

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Science

Program of Study Committee:
Wensheng Zhang, Co-major Professor
Halil Ceylan, Co-major Professor
Ali Jannesari

The student author, whose presentation of the scholarship herein was approved by the program of study committee, is solely responsible for the content of this thesis. The Graduate College will ensure this thesis is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University

Ames, Iowa

2022

DEDICATION

I would like to dedicate this thesis to my parents Jianhai and Lingmei and my girlfriend Yueyi without whose support I would not have been able to complete this work.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

ACKNOWLEDGMENTS

I would like to take this opportunity to express my thanks to those who helped me with various aspects of conducting research and writing of this thesis. First and foremost, Dr. Wensheng Zhang and Dr. Halil Ceylan for their guidance, patience and support throughout the journey of IRI research. Their insights and words of encouragement have often inspired me. I would also like to thank my committee member Dr. Ali Jannesari for serving in my POS committee. Additionally, I want to thank Dr. Bo Yang, Dr. Sunghwan Kim, Dr. Yueyi Jiao, and Mr. Venkata Ashish Gorthy for their help with the IRI project.

ABSTRACT

Road roughness affects riding comfort and vehicle operational cost, and to maintain good road quality, federal and state transportation agencies must routinely survey road roughness. Since traditional methods for such surveys require specialized equipment and training, more efficient methods that collect measurement data from sensors in off-the-shelf mobile/smart devices (such as Android phones and iPhones), have recently been developed. Such sensor data, along with labelled road roughness index value, can be used to construct machine-learning models for inferring road roughness from the sensor data. Since the sensor data generated by different device types/models have different characteristics, in current practice a model is often constructed from the data of one single or small set of device type/model, using only the data from the same configuration for predictions. This means that, despite the potentially large amount of sensor data from a greater variety of devices, labelled data may still be lacking when applying machine learning to construct a model in a specific setting.

Transfer learning focuses on extracting knowledge from a source domain, then applying it to a related target domain, and this method has been used to reduce training/labeling costs for a target domain. Based on existing extensive research, in this work we propose a clustering-based seeded-transfer learning approach to address the road roughness modeling and prediction problem. We specifically develop a complete solution for transferring data from a source domain (sensor data collected from devices of type/model A) to a target domain (sensor data collected from devices of type/model B) for model training. Our contributions include: a data scaling step, an implementation to match source clusters and target seeds, and the exploration of clustering methods, optimal number of clusters, and seeds percentage of transfer learning.

We evaluate the performance of our approach using both the sensor data set collected from the field and some public data sets. The results show that: K-means clustering is less stable than

hierarchical clustering; a moderate seeds percentage is preferred; using a greater number of clusters positively affects model prediction accuracy.

CHAPTER 1. INTRODUCTION AND RELATED WORKS

1.1 Road Roughness Measurement

Transportation connects people, and road quality significantly affects passenger riding comfort. Road quality is also tightly related to operational cost of freight transporting. There are multiple standards of road-quality quantification, and road roughness or smoothness is one of the most important. "Smoothness is a measure of the level of comfort experienced by the traveling public while riding over a pavement surface" (2016 fhwa.dot.gov). Roughness is often used as an interchangeable concept with smoothness in describing a pavement surface. Transportation agencies on federal, state, county, and city levels spend significant time monitoring road roughness on a yearly basis, and there are several ways to measure it. One definition, the International Roughness Index (IRI), is most recognized and widely used globally. IRI is defined as the value of the vertical cumulative movement divided by the horizontal distance for an observed road segment. IRI computation relies on profile measurements obtained from an inertial sensor profiler, a dipstick, or a rod and level. There are two standards for IRI computation - AASHTO standard R 43M/R 43-07 (Standard Practice for Quantifying Roughness of Pavements) and ASTM standard E1926. (Standard Practice for Computing International Roughness Index of Roads from Longitudinal Profile Measurements) (2016 fhwa.dot.gov).

Since the inertial-sensor profiler is widely used, the following discussion will focus on inertial-sensor profiler based road roughness measurements. A High Speed Profiler (HSP) is a widely used device that can be mounted on a vehicle and connected to a laptop to perform road-surface roughness measurement. There are companies that manufacture HSPs and help transportation measurement engineers install and calibrate HSPs and perform measurement. However, the device is expensive and requires regular maintenance, and a malfunctioning component may take weeks to replace or repair. Because the unit is usually mounted on the front

or rear of a specialized data collection vehicle, either a vehicle dedicated to IRI measurement or a device installed in a vehicle by an experienced engineer is needed for IRI measurement. Because of such inconvenience, HSP is better suited for regular pavement monitoring or pavement measurement after new construction by State Highway Agencies, rather than making frequent measurements during pavement construction, or leaving IRI measurement to less experienced engineers.

Recent trends include using mobile phones, smart devices, or even sensors from smart cars to help measure road roughness.

1.2 Road Roughness Prediction

In contrast to simulation methods used to calculate road roughness index, there have been multiple studies that explore correlations between sensor produced data and measured road roughness index. Such studies often use machine learning or other regression methods to build a model that predicts the road roughness index when given enough sensor-collected data for training.

Bajic et. al. proposed a pipeline that processes raw sensor acceleration data and car speed to predict the road roughness index [1]. They tested several supervised machine-learning models, including multi-layer perceptron neural networks, random forest, naïve Bayes, support vector machine, linear regression and k-nearest neighbor. They also performed feature selection on features provided by raw accelerometer measurements, and achieved good prediction accuracy.

Zhang et. al. also used machine learning to estimate road roughness index and roughness category [11]. Instead of using sensor collected data, they used numerical simulation of a model that simulated in-vehicle sensors and calculated the actual road roughness index and road roughness category. Basically, the simulation produced a complex function that calculates road roughness from sensor data. The machine-learning method is used to estimate the actual function with simulation data as training data. Speed normalization and convolution of vertical

acceleration were used for feature selection. Overall, both regression and classification achieved greater than 90% accuracy.

While some studies have reported the high accuracy of machine-learning-based road-roughness prediction, most used data collected during their experiments or simulations. Because a machine-learning model is highly dependent on the data used for training, it seems likely that such models will not work well on a data set collected with different equipment.

1.3 Transfer Learning

In real-world scenarios, an adequate amount of training data is often not available, and at the same time, there may be sufficient data from a related but not identical domain. This domain may contain enough data, but since in such a case the feature space or data distribution can be different from the original real-world scenario [7], it would be desirable that knowledge transfer or transfer learning be able to utilize those data to reduce the cost of training.

One early definition of transfer learning is that it aims to extract the knowledge from one or more source tasks and apply the knowledge to a target task. It relies on the definition of domain and task.

Definition of Transfer Learning [7]: Given a source domain D_s and learning task T_s, a target domain D_t and learning task T_t, transfer learning aims to help improve the learning of the target predictive function in D_t using the knowledge in D_s and T_s, where $D_s \neq D_t$ or $T_s \neq T_t$.

According to Pan and Yang, there are four categories of transfer learning approaches: instance-based, feature-based, parameter-based, and relational-based [7]. An instance-based transfer learning approach mainly based on instance weighting uses some source-domain instances. A feature-based approach produces a new feature representation from original features, such that source and target features are matched. A parameter-based approaches focuses on source model/parameters and transfers knowledge from there to the target. Relational-based approaches transfer the logical relational knowledge or rules from source to target.

Zhuang et. al. provides a similar definition for transfer learning, but they further divide the transfer learning approaches into data and model perspectives [12]. Data-based approaches roughly include instance-based and feature-based approaches, and model-based approaches include parameter-based and relation-based approaches.

Boosting has been applied to transfer learning. *TrAdaBoost* makes it possible for users to use source data and a small portion of labelled target data to accurately learn an model [4]. *TrAdaBoost* is designed to solve classification-transfer learning problems. To work with regression problems, Pardoe and Stone proposed extension of *TrAdaBoost* and *ExpBoost*, *TrAdaBoost.R2* [8]. Several weight-adjustment strategies were explored, and a two-stage strategy was specifically emphasized. For potential multi-source problem, the authors mentioned that multi-source data should be merged into one source, after which regression transfer boosting can be performed [8].

There are other strategies for transfer learning in regression problems. Seeded transfer learning [9] used deep learning (auto encoder and decoder) to extract common features from both source and target domain, before performing a seeded transfer from source to target seeds. Most of the experiments with seeded transfer outperformed the baseline methods [9]. The thesis is an extension of this work applied to road-roughness prediction.

In the transportation research domain, there are some studies that use transfer learning to forecast road quality trends, to analyze road quality with satellite images, or to predict road roughness classification. Brewer et.al. used a convolutional neural network model previously trained on a United States satellite-image data set, to "fine-tune" the model so that it could predict road quality classification from a Nigerian data set [2]. In preliminary experiments, they achieved 80% prediction accuracy, and 99.4% were either predicted correctly or were in an adjacent class.

Marcelino et.al. adapted the widely used *TrAdaBoost* algorithm to predict the road pavement roughness at different futures times, using the Portuguese road administration database and the Long-Term Pavement Performance (LTPP) database [5]. Their results show that all the transfer learning models outperformed the baseline, especially for long-term forecasts. Tang et.al. used a

simulated environment for bridge-road surface stress, and considered different vehicle loads for vehicle-bridge interactions [10]. They transformed the bridge response power spectral density (PSD) into a three-dimensional matrix represented by a colored image. Deep convolutional neural networks and transfer learning were then applied with historical data, and the results indicated that the strategy can accurately identify road surface roughness.

1.4 Contributions

This work makes four contributions. **First**, to my knowledge, it is the first study that uses transfer learning for international roughness index regression studies, and transfer learning significantly outperforms the baseline (source only) model. **Second**, different clustering methods for comparing prediction performance were explored. **Third**, various numbers of clusters were explored to compare prediction accuracy. **Fourth**, some practical steps were applied, including scaling before distance matching, and making use of Google OR-Tools for distance matching.

CHAPTER 2. PROPOSED WORK

2.1 Proposed Framework

The proposed workflow involves data transfer from the source domain to a target domain. In this research, the first assumption is that the source domain and target domain share the same feature space, and a second assumption is that both domains have different but related data distributions. Thus, when there are insufficient target-domain data, some source data can be *transferred* as part of training data to assist with neural network training.

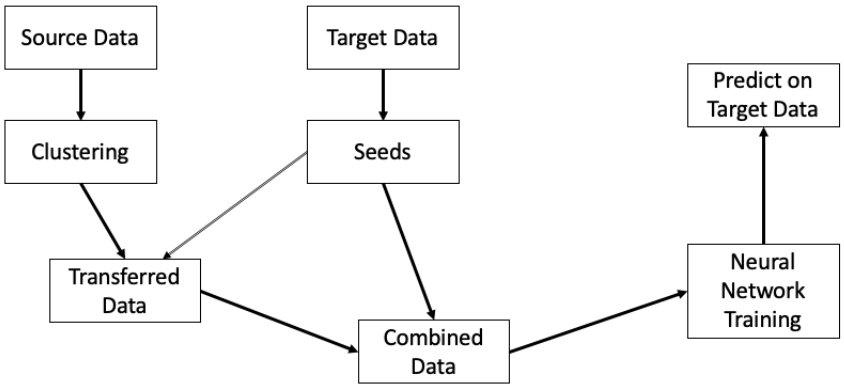

Figure 2.1 Overview of proposed workflow

Figure 2.1 depicts the major steps: clustering and seeded transfer, and neural network training on combined data.

The clustering step groups the source data into subgroups (clusters) that can be used to match target data for data transfer. The purpose of clustering is to preserve the hidden structure in source data domain, so that each cluster can be transferred to generate synthetic data that helps in the following training.

The following sections will discuss details of seeded transfer and neural network training steps.

2.2 Seeded Transfer Algorithm

Algorithm 1 performs the seeded transfer, and algorithms 2-5 are the key steps of the main algorithm, including clustering, scaling, matching, and data transferring. Note that because the Python library scikit-learn is used. the details of scaling are not given. The scaling is performed as:

$$X_{scaled} = \frac{X - Min}{Max - Min}$$

and the value can be recovered by:

$$X = X_{scaled} * (Max - Min) + Min$$

Algorithm 1: Seeded transfer algorithm

Input: Source domain data S, target domain data T, percentage of target data as seed p, number of clusters m

Output: Transferred data O

Initialize a MinMax Scaler Scl from S
Randomly select $|T| * p$ data points from T as target seeds set I
$A \leftarrow CLUSTERING(S, m)$ `// clustering source`
$B \leftarrow CLUSTERING(I, m)$ `// clustering target`
$D, U, V \leftarrow DISTANCE(A, B)$ `// calculate source-target distance`
$D' \leftarrow Scale(D, Scl)$ `// scale with MinMax scaler`
$P \leftarrow MATCH(D')$ `// find the best match`
$O \leftarrow TRANSFER(P, D, A, B)$ `// transfer source data to final data`
Return O

Algorithm 2: CLUSTERING performs clustering and regroup data by cluster

Input: A finite set $D = \{d_1, d_2, \ldots, d_n\}$ of source or target data, number of clusters m

Output: A list of sets that contains data regrouped by cluster

$L \leftarrow [l_1, l_2, \ldots, l_m]$, where $l_i \leftarrow \emptyset$, and $1 \leq i \leq m$
Call *K-means* or *Hierarchical Clustering* on D, to get labels $K = \{k_1, k_2, \ldots, k_n\}$ that indicates the cluster d_i belongs to
for $i \leftarrow 1$ ***to*** n **do**
    $j \leftarrow k_i$ `//` d_i `belongs to cluster` k_i
    $l_j \leftarrow l_j \cup \{d_i\}$
return L

Algorithm 3: DISTANCE calculates the distance between source and target cluster centers

Input: A list of set $S = \{s_1, s_2, \ldots, s_m\}$ of source data clusters, a list of set $T = \{t_1, t_2, \ldots, t_m\}$ of target data clusters

Output: A matrix D of Euclidean distance between source and target cluster centers, a list source centers U, and a list of target centers V

$D \leftarrow [[d_{1,1}, d_{1,2}, \ldots, d_{1,m}], \ldots, [d_{m,1}, d_{m,2}, \ldots, d_{m,m}]]$, where $d_{ij} = 0$, and $1 \leq i \leq m$, $1 \leq j \leq m$

for $i \leftarrow 1$ ***to*** m **do**
    $u_i \leftarrow average(s_i)$ `// center for source cluster`
    $U \leftarrow U \cup \{u_i\}$

for $i \leftarrow 1$ ***to*** m **do**
    $v_i \leftarrow average(t_i)$ `// center for target cluster`
    $V \leftarrow V \cup \{v_i\}$

for $i \leftarrow 1$ ***to*** m **do**
    for $j \leftarrow 1$ ***to*** m **do**
        $d_{i,j} \leftarrow$ Euclidean distance between u_i and v_j

return D, U, V

Algorithm 4: MATCH matches source and target centers for minimum total distance

Input: A matrix M of distance between source and target cluster centers

Output: A list of paired index from source to target P

$P \leftarrow [p_1, p_2, \ldots, p_m]$, where $p_i \leftarrow 0$, and $1 \leq i \leq m$

`/* assign pairing, such that total distance is minimized */`

Call *OR-Tools* Solver with M as input, output solution matrix R

for $i \leftarrow 1$ ***to*** m **do**
    for $j \leftarrow 1$ ***to*** m **do**
        if $r_{i,j} = 1$ **then**
            $p_i \leftarrow j$ `// source cluster` i `matches target cluster` j

return P

Algorithm 5: TRANSFER moves source data to form the transferred data

Input: A list of index of pairing P, a matrix M of distance between source and target cluster centers, A list of set $S = \{s_1, s_2, \ldots, s_m\}$ of source data clusters, a list of set $T = \{t_1, t_2, \ldots, t_m\}$ of target data clusters

Output: Transferred data O

```
1  O ← ∅
2  for i ← 1 to m do
3      j ← p_i    // source cluster i matches target cluster j
4      d ← m_{i,j}
5      foreach k ∈ s_i do
6          l ← k + d
7          O ← O ∪ {l}
8  for i ← 1 to m do
9      foreach k ∈ t_i do
10         O ← O ∪ {k}
11 return O
```

2.2.1 Clustering Method

There are multiple clustering methods, including K-means and hierarchical clustering, and performance of both clustering methods was compared, with each offering advantages and disadvantages. K-means is a more commonly used clustering method, and while the computation is relatively efficient, since the initial center is randomly selected, the K-means clustering result can be different for each iteration. Deciding the optimal number of clusters is also not straightforward, because the number of clusters is a required input parameter for K-means clustering. There are methods to decide the optimal number of clusters, e.g., the *elbow* method.

By comparison, hierarchical clustering involves constructing a tree-like structure by comparing the distance of each element. There are two ways of doing hierarchical clustering, agglomerative (bottom-up) and divisive (top-down). Because of the tree structure formed by this clustering method, sometimes it is more natural to determine the number of clusters by looking at the tree splits. Its tree structure is also relatively stable compared to K-means clustering, but hierarchical clustering computation can be slow, compared to K-means, especially when the data set is large.

Both methods for performing clustering on the source data were explored, and their performance will be discussed in the experiment chapter.

2.2.2 Number of Clusters

The initial version of seeded transfer involves performing clustering using the *size-of-seeds* data set as the number of clusters. The intuitive thinking is that breaking source data into a fixed number of clusters and performing one-to-one matching between source clusters and target seed data may be unnatural. Simply speaking, there might be a better *number of clusters*, such that, after matching, more suitable transferred source data is transferred into the final data set.

Thus, in this work, different numbers of clusters were explored. Instead of using seed data size as the number of clusters, other sizes were explored to match the potential optimal clustering size. The results for the experimental data set are discussed in experiment chapter.

2.2.3 Implementation Techniques

There are some implementation techniques that my work filled in compared to those in the original seeded-transfer paper. First, scaling was performed before looking for the optimal matching between source cluster center and target seeds data point, mainly because when the data is not scaled, the feature dimension that carry bigger values will affect the Euclidean distance more in the multi-dimensional space. Second, I used Google OR-Tools to decide on the approximately optimal matching between cluster center and target data, such that total distance is approximately minimized.

2.3 Neural Network Training

The seeded transfer algorithm will output transferred data that can then be used for subsequent machine-learning (ML) training, such that ML model can fit this transferred data. Then the ML model will be tested against the whole target data set to examine its prediction accuracy.

Multiple ML methods can be used for training on the transferred data, including multi-layer perceptron (MLP) neural networking, random forest, naïve Bayes, support vector machine, linear regression and k-nearest neighbor. In this work, the multi-layer perceptron was used because is powerful and can handle a complex data set and large or small data sets. The source data set used in the work normally had a size of less than 20, 000 that did not hinder computation of the MLP.

CHAPTER 3. DATA SET DESCRIPTION

Since this work mainly deals with regression prediction for different but related source and target data set, three real data sets from different situations were used in the experiments.

Table 3.1 summarizes the 3 data sets used.

Table 3.1 Summary of data sets used in experiment

Data set	**Source size**	**Target size**	**Input variables**	**Output variables**
Road roughness	1118	373	cIRI, speed, std(acceleration)	rIRI
California housing price	16846	154	house age, rooms, bedrooms, residents, income	house price
Wine quality	4898	1599	fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol	quality

3.1 Road Roughness Data Set

The road-roughness data set was an in-house data set collected from road roughness measurements using mobile phones, smart devices, and a High-Speed Profiler (HSP). The workflow of Figure 3.1 demonstrates the road roughness data collection process. For IRI measurement with HSP, the HSP user manual was followed, and the IRI value was calculated from commercial software that accompanied the HSP. The data was measured once for each road segment.

Smartphone: A Flutter mobile app was developed to utilize the sensor and a GPS module for recording smartphone movement. The recorded acceleration and moving speed were sent to a cloud server for raw data processing and to store the record in a database.

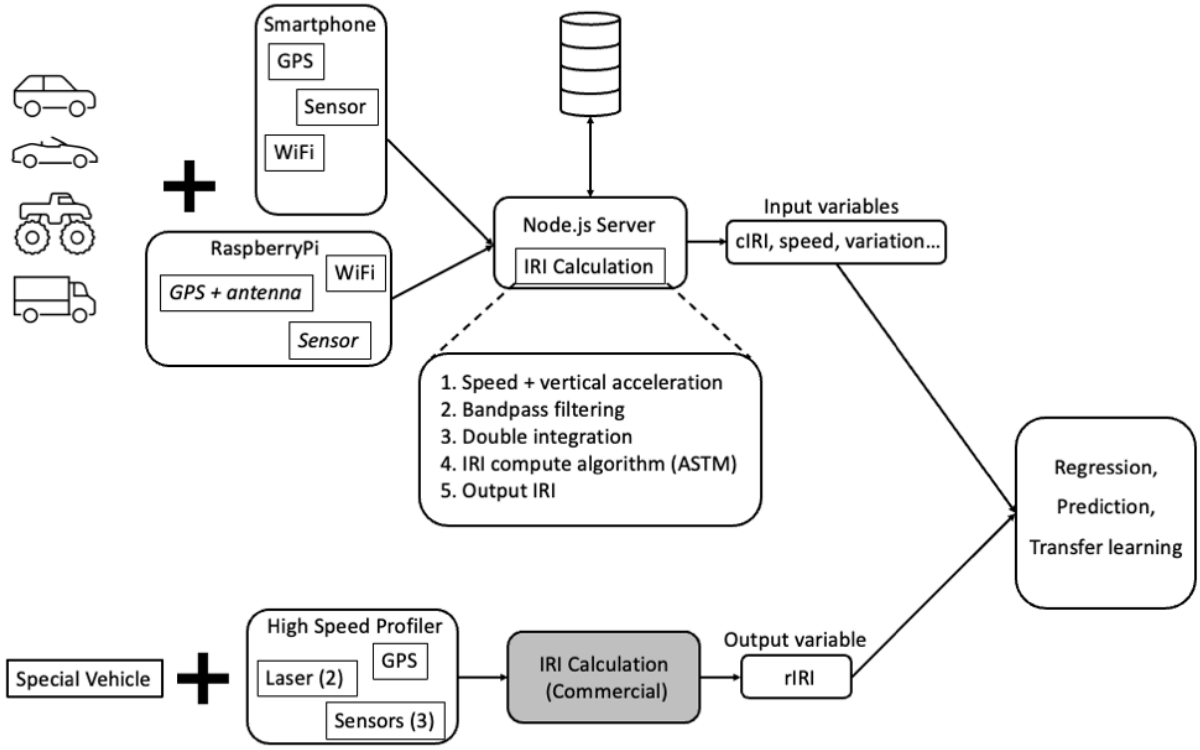

Figure 3.1 Road roughness data collection workflow

Smart box: A Raspberry Pi based smart box was developed. Some custom components, including an Inertial Motion Unit (IMU), a GPS module and a GPS antenna, were connected to the Pi through GPIO ports. To control the app, a Flutter mobile app was developed to connect the Pi through wireless communication. The Flutter app can send a start or stop signal to the Pi to record GPS data and acceleration when performing road-roughness data collection, after which the recorded file is transferred to the same cloud server and saved to a database.

Algorithm for IRI Calculation: An algorithm was developed to calculate the IRI value. First, the vertical acceleration and average speed are calculated, after which double integration with three bandpass filters is performed to calculate the movement profile. Then the widely used ASTM IRI computation algorithm implemented in Python can read the movement profile to output a calculated IRI value.

Through the smartphone, smart box, cloud server, and IRI calculation algorithm, a calculated IRI value (cIRI) for one specific vehicle and device is produced for a road segment. This IRI value is compared to the reference IRI value measured and calculated from HSP. The IRI value and other related variables are used as input variables, and the reference IRI value from HSP is used as an output variable. With data collected from 26 different sites, 2 vehicles, and 5 devices, about, 1300 data points are used for regression.

Each instance has attributes of speed, phone type, calculated International Roughness Index (IRI) value, and standard deviation of acceleration as input variables, and the HSP-measured IRI value is the output variable.

The road-roughness data set consists of source data and target data; source data is composed of data collected from several mobile phones, while target data includes data points from another house-made device. The goal of transfer learning is to use some source data to assist the machine learning model to predict a real IRI value for a new measuring device. The source and target data are not in the same domain, because with different measuring device, some specific factors may affect IRI prediction. The source and target data are related, however, because they share the same feature space and the IRI prediction mechanism is similar.

3.2 California Housing Price Data Set

The California housing price data set is a public data set from the 1990 Census for all the block groups in California [6]. A specific region comprised of about 150 instances was used for target data, while the remainder of the data were used as source data.

Each instance represents a block where thousands of people reside; each instance has attributes, including housing median age, total rooms, total bedrooms, population, households, and median income of the surveyed block as input variables. The median house price of the block is used as an output variable. The source data and target data are related because they follow a similar pattern, i.e., that the median housing price is dependent on factors like income, household

size, rooms, etc., while the exact housing price from different regions (Los Angeles vs Central Valley) can vary considerably.

3.3 Wine Quality Data Set

The wine-quality data set is a public data set from the UC Irvine Machine Learning Repository (https://archive-beta.ics.uci.edu/ml/datasets/wine+quality); the original data is from [3]. There are two sets of data for white wine and red wine quality and physicochemical features. The white wine data is used as source data and the red wine data is used as target data. The source and target data share the same feature space, but different relations between physicochemical features and wine quality. Input variables are physicochemical measurements, including pH, residual sugar, chloride, alcohol, density, sulfates, and fixed acidity, and the output variable is wine quality. White wine and red wine have a related model for predicting wine quality, although some aspects may be different (for example, one may prefer sweeter taste for white wine, but drier taste for red wine).

CHAPTER 4. EXPERIMENTS

The machine learning Python library *sci-kit learn* was mainly used for clustering, neural network, scaling, and evaluation. Specifically, *sklearn.cluster*, *sklearn.preprocessing*, *sklearn.neural-network*, and *sklearn.metrics* were used. The default parameters were used for sklearn functions, unless noted elsewhere. Python library *seaborn* and *matplotlib.pyplot* were used to produce the graphs.

The default *MLPRegressor* parameters of sci-kit was used, and Table 4.1 summarizes some key parameters. The number of input layer neurons is the same as features for each data set. Thus, the input layer neuron sizes are 3 for road roughness, 5 for California housing price, and 11 for wine quality. The output layer is one neuron that output a real value.

Table 4.1 MLP key parameters used in experiment

Activation	ReLU
Alpha	0.0001
Batch size	Auto
Hidden layer size	(100)
Learning rate	0.001
Max iteration	200

The source and target data were either naturally separated or manually selected based on some standards (physical localization or measuring device). Normally, the target data set is much smaller than the source data set, and only a portion of target data (we call seeds) are "labeled". The goal is to use source data and seeded transfer to improve prediction accuracy on the target data set. Performance on the whole target data set was measured by prediction error (RMSE).

For the transferred data, 5 independent neural network models were trained, and tested against the whole target data set. For the hierarchical-clustering experiment, clustering was performed only once, while for the K-means clustering experiment, clustering was performed 5

times, because K-means involves random initial data points (thus not stable), while hierarchical clustering is relatively stable.

4.1 Road Roughness Data Set

The source-only (SO) data was used to train the NN model, after which the models were verified with respect to the target data to shows the performance of the source-only model when no 'transfer learning' techniques are used. The average RMSE for SO model was 91.68, taken as the baseline for the following comparison on the performance of transfer-learning experiments.

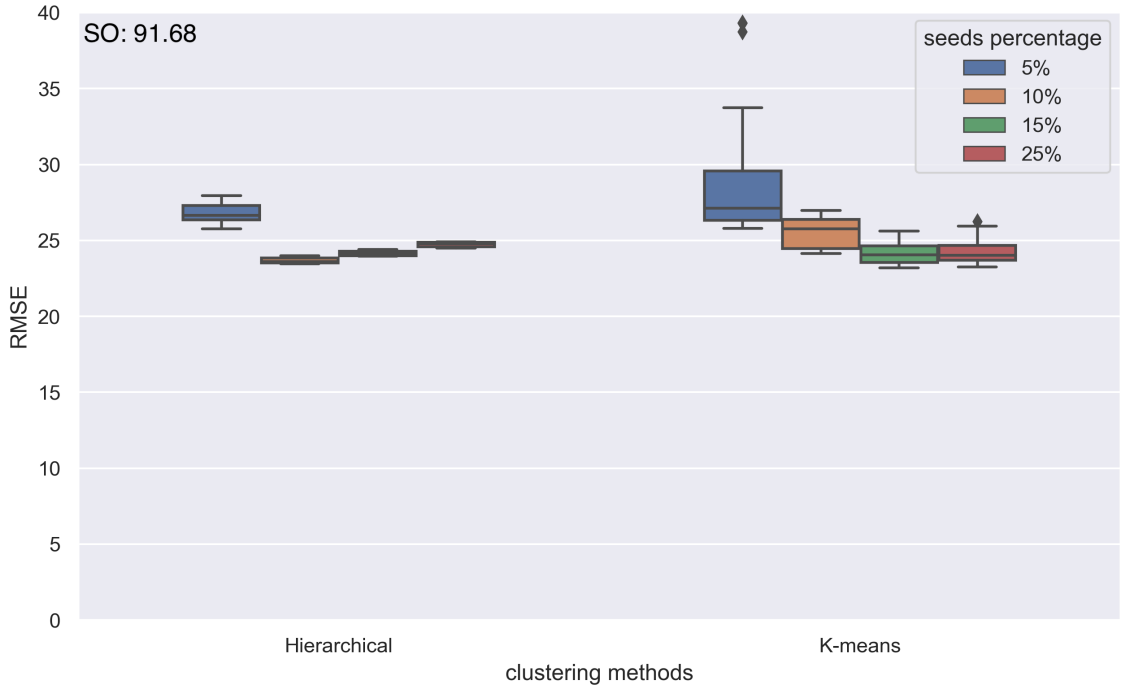

Figure 4.1 Comparison of clustering methods for road roughness data set

4.1.1 Clustering Method

Figure 4.1 shows that both K-means and hierarchical clustering exhibited significantly improved prediction accuracy. Compared to an RMSE of 91.68 from SO data set, most of transferred data generated RMSE values were less than 30. For all 4 seed percentages (5%, 10%,

15%, 25%), both clustering methods exhibited similar performance, indicating that using seeds from a target data set helps to transfer useful information from the source domain.

As for the comparison of two clustering methods, they have comparable performance for different seeds percentage. One difference is the stability. K-means clustering relies on the initial random state, while hierarchical clustering produces relatively stable clustering structure. Thus, K-means clustering method produced data has more variation than hierarchical clustering. For example, at 5% seeds, some K-means produced data has much higher RMSE (near 40). In terms of running time, both K-means and hierarchical clustering are fast (5 seconds max). K-means are normally even faster than hierarchical clustering, especially when data set size is big. However, because hierarchical clustering produces stable clustering, it only needs to be run once. In comparison, K-means may need multiple runs, even one single run is faster.

4.1.2 Number of Clusters

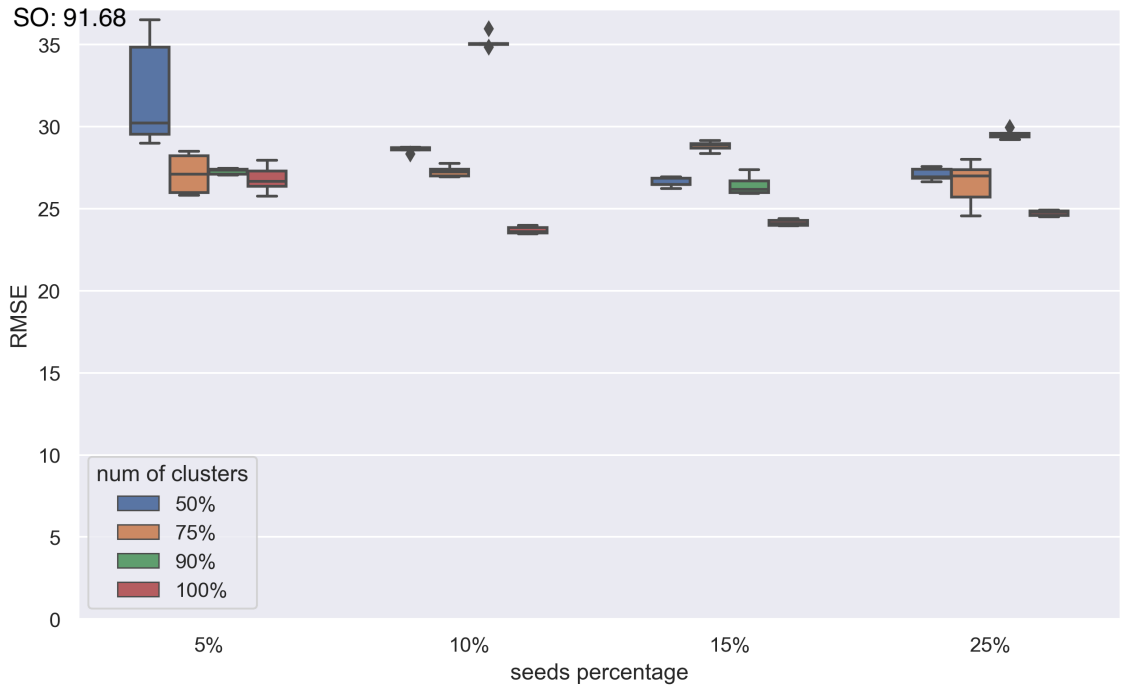

Figure 4.2 Number of clusters effects for road roughness data set (hierarchical clustering)

Let the size of target seeds data set as n. Instead of simply clustering the source data into n clusters, target data and source data are clustered into smaller number of clusters, before matching the target and source data. In the experiment, 50%, 75% and 90% are explored. As the Figure 4.2 shows, when using hierarchical clustering, the number of clusters has varying effects on prediction accuracy. Overall, the original seeds size performs the best among different seeds percentage. Similar trends can be found in K-means generated data (Figure 4.3), in which original seeds size outperforms other number of clusters. As previously observed, K-means generated data has more variations, among different number of clusters. However, in this experiment, the variation of K-means method tends to be smaller, as the seeds percentage increases (Figure 4.3).

In summary, alternative number of clusters does not seem to help improve the transfer learning prediction accuracy.

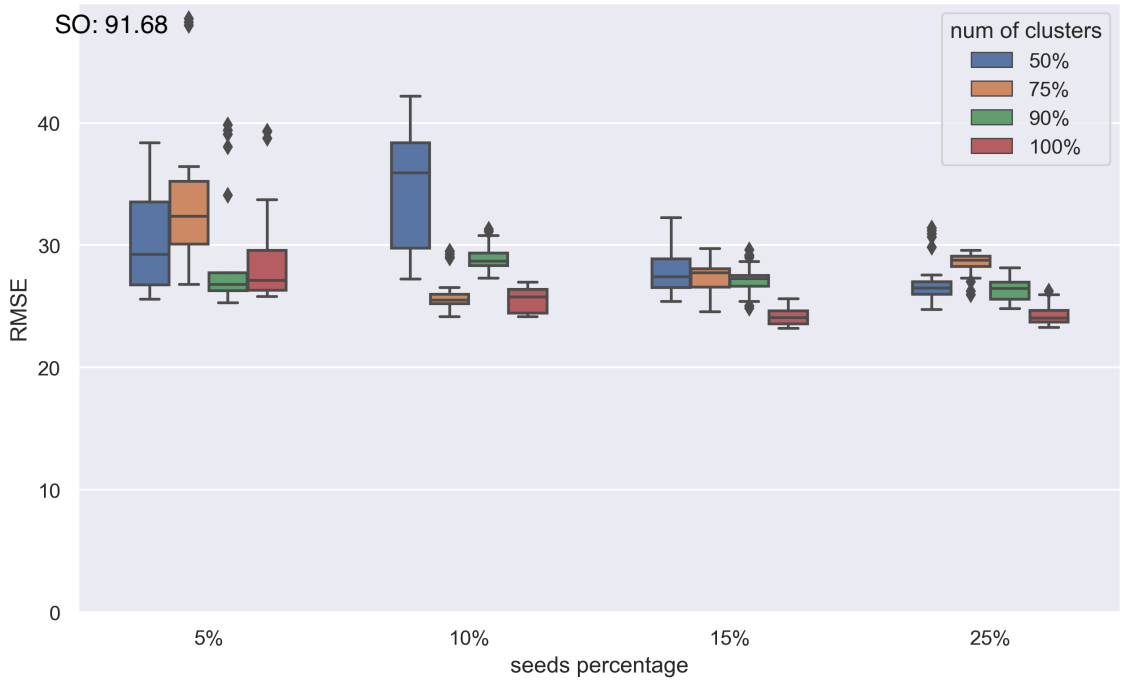

Figure 4.3 Number of clusters effects for road roughness data set (K-means)

4.1.3 Seeds Percentage

Another parameter investigated in the experiment is the seeds percentage. As Figures 4.1, 4.2, 4.3 show, with either clustering method, higher seeds percentage shows slightly improved prediction accuracy. A possible explanation is that as seeds percentage grows, more target data are involved in transfer learning. Thus, the transferred data has more original representation of the target data set. On the other hand, the seeds percentage of 10% and 15% performs similar to 25%. This indicates that a higher seeds percentage may be unnecessary. Because labeling the seeds can be expensive, it may be preferable to use smaller seeds percentage with cheaper labeling efforts, but similar performance.

4.2 California Housing Price Data Set

4.2.1 Clustering Method

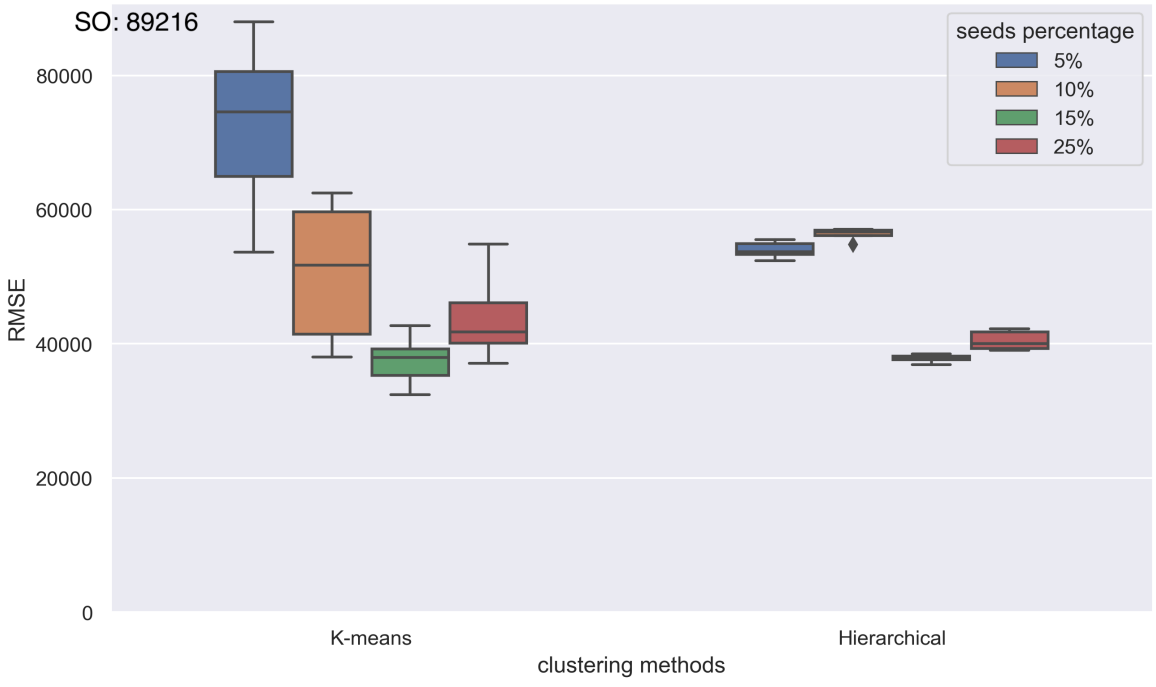

Figure 4.4 Comparison of clustering methods for California housing price data set

Similar to the road roughness data set, most of the transfer learning experiments outperform the source-only (SO) experiment. SO produced an average RMSE of 89216, while most transfer learning produced RMSE under 60000 (Figure 4.4).

As for the comparison between K-means and hierarchical clustering, K-means produced significantly more variations. With 5 different random initial state of K-means, the final RMSE varies more than 20000, while each experiment of hierarchical clustering varies in a small range (less than 3000) (Figure 4.4). The running time difference between K-means and hierarchical clustering is similar. K-means is faster for one clustering, but may need multiple repeats. The California housing price data set contains more data, so the run time difference is more obvious.

4.2.2 Number of Clusters

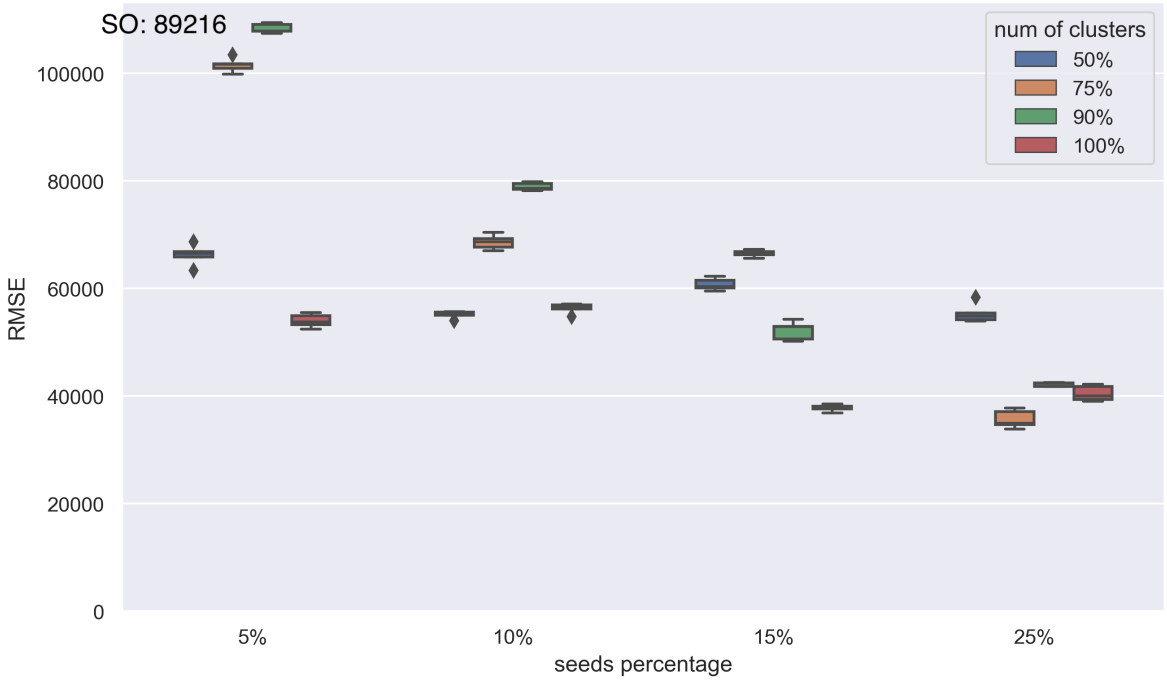

Figure 4.5 Number of clusters effects for California housing price data set (hierarchical clustering)

As Figure 4.5 shows, for hierarchical clustering, original target seeds size clustering performs the best or near the best. Some other number of clusters (75% of 25% seeds number clusters) performs slightly better than original seeds size. However, no obvious trends are found.

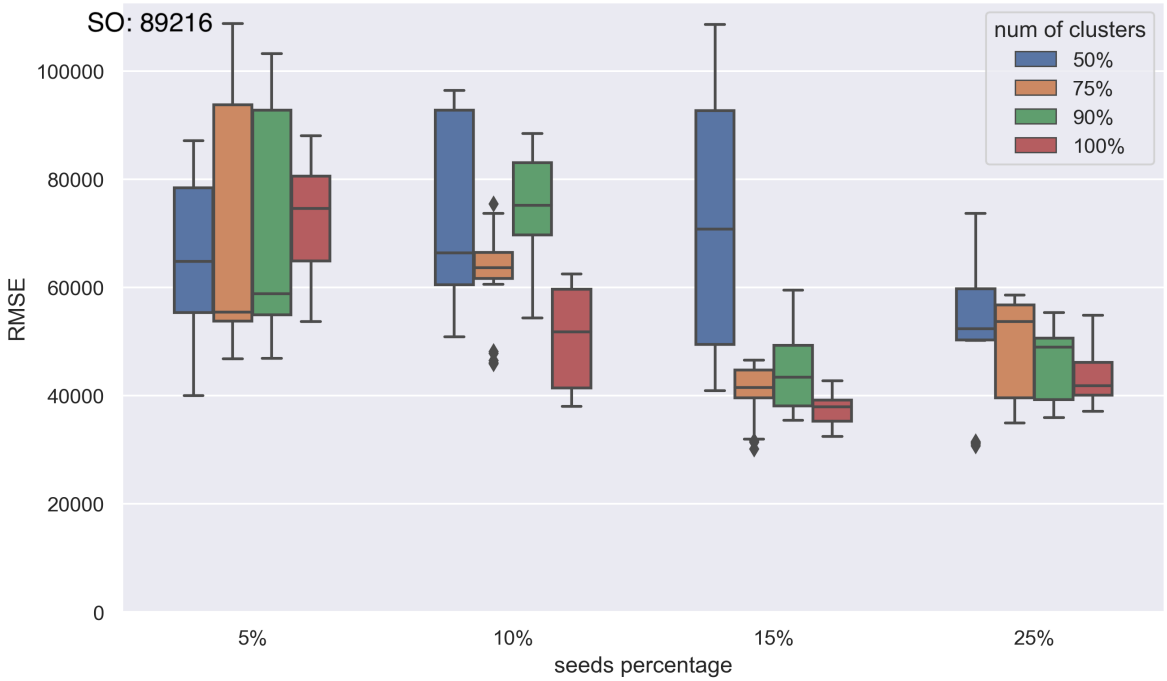

Figure 4.6 Number of clusters effects for California housing price data set (K-means)

As Figure 4.6 shows, the K-means produced big variations. Almost at all seeds percentage experiments, the in-group variation is too big to see difference among the comparison of number of clusters.

4.2.3 Seeds Percentage

Finally, the different seeds percentage experiment follow similar patterns as the road roughness experiment. As Figures 4.4, 4.5, 4.6 show, 15% and 25% performs better than 5% and 10%. This suggests that higher seeds percentage usually improves the prediction performance. However, from 15% to 25%, there is no significant improvement, but sightly higher RMSE. This

suggests that a moderate seeds percentage might be better, as it requires less labeling efforts and achieve similar results.

4.3 Wine Quality Data Set

4.3.1 Clustering Method

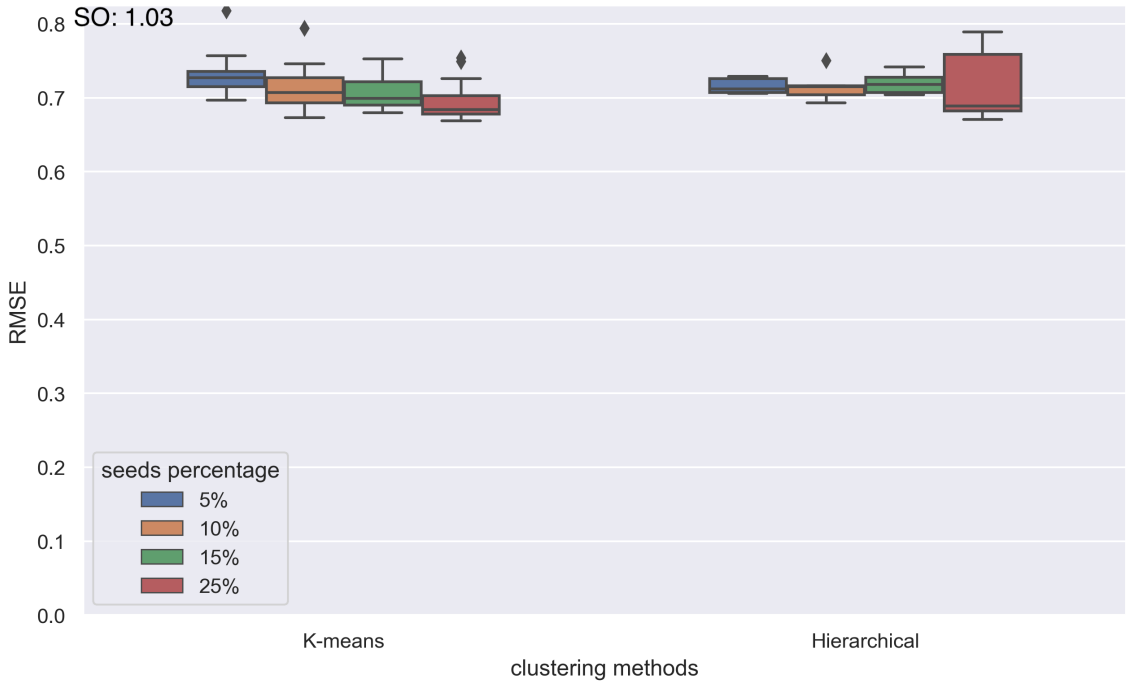

Figure 4.7 Comparison of clustering methods for wine quality data set

The baseline for source-only (SO) data is an average RMSE of 1.03. Most of the transfer learning experiments produced RMSE smaller than 0.8 (Figure 4.7). Different from road roughness and California housing price data set, K-means method produced relatively stable outcomes.

Overall, two clustering methods produced quite similar results (RMSE ranges from 0.65 to 0.75). A possible explanation is that the source and target data may share similar distribution pattern. Another educated guess is that the target data size is relatively big (1599), compared to

the other two data set (154 and 373). Thus, the target data seeds may be well represented data set, even without transfer learning.

4.3.2 Number of Clusters

As Figure 4.8 shows, the transfer learning prediction error decreases, when more clusters are applied. The difference is not very significant and consistent, though.

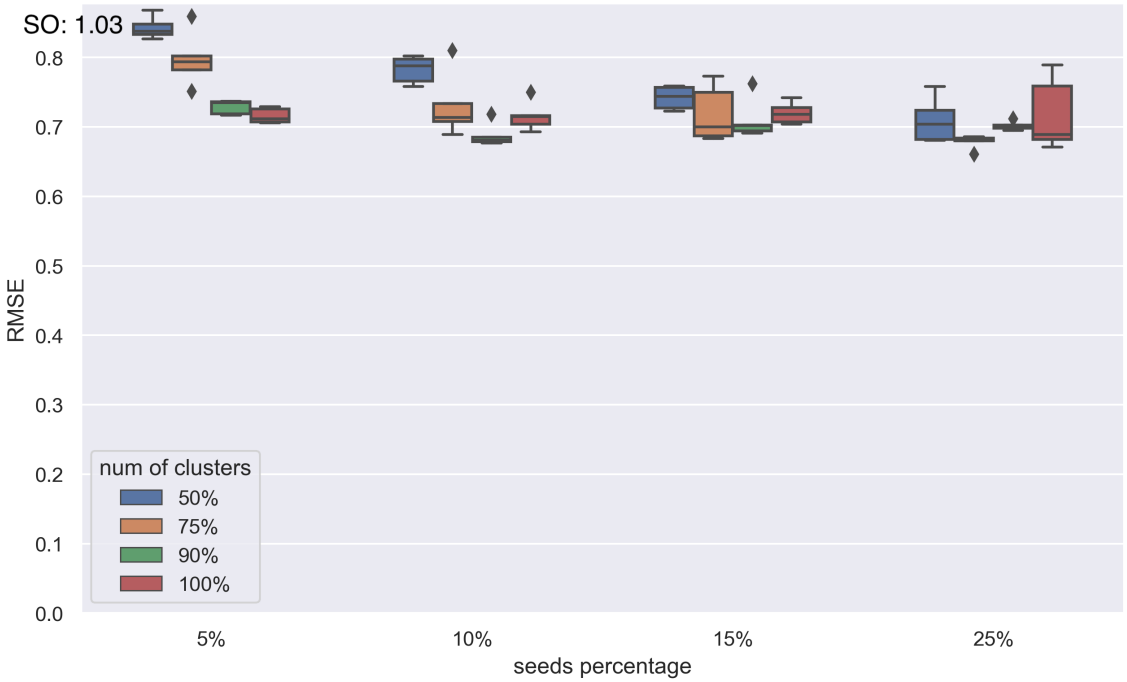

Figure 4.8 Number of clusters effects for wine quality data set (hierarchical clustering)

As Figure 4.9 shows, K-means still produced more variations than hierarchical clustering. There is no obvious pattern for number of clusters, as the variation within each group is bigger than between-group comparisons.

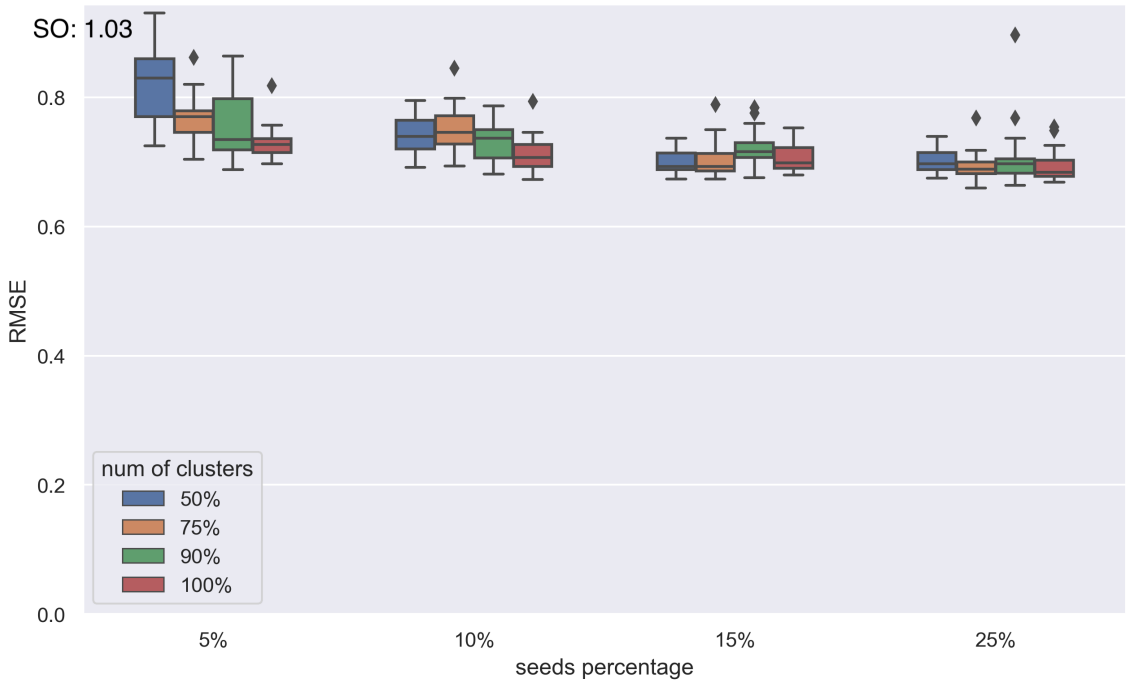

Figure 4.9 Number of clusters effects for wine quality data set (K-means)

4.3.3 Seeds Percentage

Although quite subtle, the prediction slightly increases as the seeds percentage increases (Figure 4.7). As explained previously, this could be because the target seeds contains enough data to represent the whole target domain, even without applying transfer learning.

CHAPTER 5. CONCLUSION AND RECOMMENDATIONS

The thesis described implementation of a seeded transfer learning algorithm applied to three data sets. Several algorithm components, including clustering method, number of clusters, and seeds percentage, were explored on all three experimental data sets; the findings and recommendations for practical application are summarized here.

- A transfer learning algorithm improved prediction accuracy on the target data set, compared to that using the source-only data set.

  In all three sets of experiments, except for some outliers, most transfer learning runs produced a smaller prediction error compared to the source-only run. Specifically, road roughness transfer learning reduced RMSE from 91.68 to below 30; California housing price transfer learning reduced RMSE from near 90000 to below 60000; wine quality transfer learning reduced RMSE from 1.03 to about 0.7.

- Different clustering methods have their own advantages and disadvantages. Since K-means clustering is dependent on initial random state, it tends to produce larger variations. K-means runs fast when the data that needs clustering is large. In comparison, since hierarchical clustering produces relatively stable clustering, it does not need repeats, although each single run of hierarchical clustering can be slower than that of K-means. In the three experiments, both clustering methods produce similar average prediction accuracy, but K-means exhibited relatively larger variations (for example, California housing price Figure 4.6).

  Recommendation: If the run-time for hierarchical clustering is not excessive, use hierarchical clustering. If K-means will be used, repeat several times with different initial states.

- When performing the initial clustering there still could be an optimal number of clusters, although in the three sets of experiments, it did not seem that reducing the number of clusters improved performance.

  Recommendation: For the clustering step it is preferable to use the original seeds size as the number of clusters.

- More seeds from target usually helps improve final model prediction accuracy, but to reduce labeling efforts, a moderate level of seeds percentage is preferred. In the road-roughness experiment, 10% or 15% seeds percentage performed similar to 25% (Figure 4.1). In the California housing price experiment, 15% or 25% performed the best, much better than 5% and 10% (Figure 4.4). In the wine-quality experiment, all four seeds percentages performed similarly, with a higher seeds percentage slightly better (Figure 4.7).

  Recommendation: A 15% seeds percentage is good. If labeling is a limiting factor, a 10% seeds percentage can be considered.

- *What kind of data is suitable for this transfer learning algorithm?*

  If the target domain is small enough to be labelled, then no machine learning or prediction is needed.

  If due to some limitations, there is no way to label any target data then this algorithm can't be used because this algorithm needs initial seeds to be labeled. Although the algorithm can be modified to adapt to unlabeled data (use only input space for clustering, for example), that approach requires further study.

  The source and target domain should be related but not too similar data. In the wine-quality example, since the source domain is similar to the target (white wine and red wine), the target seeds may already well represent the target domain distribution pattern. In such a case, 'transferring' data from the source domain may not improve prediction accuracy.

- In the specific discussion on road-roughness predictions, effectiveness was demonstrated by the road-roughness experiment, and this can be useful for other conditions. For example, very often, different devices (mobile phones) or different vehicles with different suspension parameters are used when measuring road roughness. Transfer learning can help utilize previously collected data from a different setting to adapt to newly collected data. With less labeling as the seeds, this transfer learning algorithm may help to achieve satisfactory prediction accuracy.

  Note that there are no measuring device or vehicle requirements. Since transfer learning is intended to solve a generalized problem, so long as the source and target domain are related, it this algorithm can be applied.

BIBLIOGRAPHY

[1] Bajic, M., Pour, S. M., Skar, A., Pettinari, M., Levenberg, E., and Alstrøm, T. S. Road Roughness Estimation Using Machine Learning. *arxiv.org* (7 2021).

[2] Brewer, E., Lin, J., Kemper, P., Hennin, J., and Runfola, D. Predicting road quality using high resolution satellite imagery: A transfer learning approach. *PLOS ONE 16*, 7 (7 2021), e0253370.

[3] Cortez, P., Cerdeira, A., Almeida, F., Matos, T., and Reis, J. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems 47*, 4 (11 2009), 547–553.

[4] Dai, W., Yang, Q., Xue, G. R., and Yu, Y. Boosting for transfer learning. *ACM International Conference Proceeding Series 227* (2007), 193–200.

[5] Marcelino, P., de Lurdes Antunes, M., Fortunato, E., and Gomes, M. C. Transfer learning for pavement performance prediction. *International Journal of Pavement Research and Technology 13*, 2 (3 2020), 154–167.

[6] Pace, K., Barry, R., Pace, K., and Barry, R. Sparse spatial autoregressions. *Statistics & Probability Letters 33*, 3 (1997), 291–297.

[7] Pan, S. J., and Yang, Q. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering 22*, 10 (2010), 1345–1359.

[8] Pardoe, D., and Stone, P. Boosting for Regression Transfer. *Proceedings of the 27th International Conference on Machine Learning* (2010).

[9] Salaken, S. M., Khosravi, A., Nguyen, T., and Nahavandi, S. Seeded transfer learning for regression problems with deep learning. *Expert Systems with Applications 115* (1 2019), 565–577.

[10] Tang, Q., Xin, J., Jiang, Y., Zhou, J., Li, S., and Chen, Z. Novel identification technique of moving loads using the random response power spectral density and deep transfer learning. *Measurement 195* (5 2022), 111120.

[11] Zhang, Z., Sun, C., Bridgelall, R., and Sun, M. Application of a Machine Learning Method to Evaluate Road Roughness from Connected Vehicles. *Journal of Transportation Engineering, Part B: Pavements 144*, 4 (12 2018), 04018043.

[12] Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., and He, Q. A Comprehensive Survey on Transfer Learning. *Proceedings of the IEEE 109*, 1 (1 2021), 43–76.