

University of Queensland & Landscape Science / DES

A Report for Pattern Identification in Soil Site Data in Queensland

Submitted for Master of Data Science Capstone Project 2

Zhichuan Lu
16-NOV-2020

Table of Contents

1. Introduction	2
1.1. Motivations and Objectives.....	2
1.2. Background research	3
2. The dataset and pre-processing.....	3
2.1. The data	3
2.2. Data transformations.....	4
2.3. Missing data processing.....	7
2.4. Feature selection	7
3. ASC Analysis	8
3.1. Hierarchical Classes	8
3.2. Algorithms Analysis	9
3.2.1. MLP (Multi-Layer Perceptron).....	9
3.2.2. DLNN (Deep Learning Neural Networks).....	10
3.2.3. Decision tree	12
4. SPC Analysis.....	13
4.1. Step One: SPC Clustering	13
4.1.1 Data Exploration.....	13
4.1.2. Similarity, Centroid, and Distance Matrix	14
4.1.3. Hierarchical Clustering	15
4.2. Step Two: Supervised learning after clustering.....	16
5. Conclusions and discussions	17
References	19

1. Introduction

1.1. Motivations and Objectives

Queensland Government Soil and Land Information (SALI) database, as described by DES chief scientist Evan Thomas (personal communication, 6 May, 2020), one of my industry supervisors, "is the Queensland Government repository for soil and land information and acts as the point-of-truth for government soil and land resource data." "This information is used to support decision-making, planning and modelling goby the government and others."

The Australian Soil Classification (ASC) serves as a framework for organising our knowledge of Australian soils and provides a means of communication among scientists, and between scientists and those who use the land (Isbell, 2016).

The existing ASC labels for current SALI data have been examined and determined by many various soil experts over the years, based on the chronologically updated soil classification standards. While there is continuity between standards over time, there are changes; furthermore, even if same standards are applied on the classification process, different soil experts may have different opinions. Therefore, Task 1 of this project is to build an appropriate algorithm for machine learning to learn from existing SALI soil data as well as the discriminations of soil data classification by soil experts in diachronic periods, with the expectation of training a model that can synthesize the discriminative knowledge of soil experts and provide an effective and efficient helping tool for future soil classification matters.

Soil profiles are composite objects made up of sequences of soil layer individuals, and one of the fundamental problems in soil science is to create a classification of such profiles (McBratney & Minasny, 2017).

The existing SPC labels are decided by soil experts as well based on their knowledge and experience. Given that there are already thousands of categories in current SALI dataset's SPC column, comparing a new soil data with existing SPCs turns out to be a daunting task. consequently, it is difficult to ensure that soil data with similar properties are assigned to same SPC. As a result, it is inevitable that duplicate naming will occur in SPC over time. Hence, Task 2 of this project is aiming to correlate and merge existing SPC to reduce its size before applying similar process as in task 1.

In Task 1, machine learning models such as Multi-Layer Perceptron (MLP), Deep Learning Neural Networks, and Decision Trees (DL) are chosen to implement on ASC.

In Task 2, a Hierarchical Clustering algorithm is processed to correlate and merge existing SPC before similar methods as described in Task 1 being implemented on it.

1.2. Background research

To the best of my knowledge, this is the first time that machine learning methods have been applied to classify/cluster ASCs and SPCs in SALI data. A few published articles contain related research to this project. *Machine learning in soil classification* (B. Bhattacharya, D.P. Solomatine, 2006) uses CONstraint Clustering and Classification (CONCC) algorithm to automatically classify soil data obtained from Cone Penetration Testing (CPT) method; *Machine learning for predicting soil classes in three semi-arid landscapes* (Colby W. Brungard, 2014) compare multiple machine learning models and covariate sets for predicting soil taxonomic classes; and *Incorporating taxonomic distance into spatial prediction and digital mapping of soil classes* (Minasny & Mcbratney, 2007) investigate the advantages and challenges of incorporating taxonomic distance into soil classes prediction. This project and the studies in the above-mentioned articles have different aspects, such as: different data sources and data structures, different classification objectives, etc. Therefore, although they helped in the implementation of this project, there is no way to directly compare their findings with the present project in terms of numerical indicators. This report only provides a comparative analysis of the output of the three models mentioned above.

The main contributions of this report are: Analysing the data type and structure of the SALI database and designing appropriate input data structures for different models in order to produce meaningful results; Designing suitable deep learning neural network architectures for ASC hierarchical class fitting; Providing an auxiliary basis for further accurate classification of ASC SPC in SALI data, although the model outputs themselves are not yet accurate enough to be used directly as classification judgments.

2. The dataset and pre-processing

2.1. The data

The SALI dataset contains two levels of attributes: soil profile attributes and soil horizon attributes. Soil profile attributes, such as "DRAINAGE" and "STATUS", are attributes that represent properties of entire soil profile; soil horizon attributes, such as "HOR_MASTER", "TEXTURE_CODE", and "FTS_PH", are attributes that represent properties of multiple soil horizons, respectively. The object properties we're analysing, ASC and SPC, are both profile level properties. That means for an individual soil data, the value of horizon attributes is not a single value (scalar), but a set of values (vector).

Also, three data types are included in SALI datasets: numeric, ordinal, and nominal/categorical.

Ordinal data can be converted to numeric data.

Depending on different attribute levels and data types, soil data can be divided into the following four types:

		Data Type		
Attribute Level	Attribute Value Structure	Numeric	Ordinal	Categorical
Profile	Scalar	sn	sn	sc
Horizon	Vector	vn	vn	vc

Figure 1: SALI data construction

The SALI dataset can be presented as:

$$\chi = \begin{pmatrix} x_{1,1}^{sn} & \dots & x_{1,d_{sn}}^{sn} & x_{1,1}^{sc} & \dots & x_{1,d_{sc}}^{sc} & \begin{pmatrix} x_{1,1}^{vn} \\ \vdots \\ x_{1,d_{vn}}^{vn} \end{pmatrix} & \dots & \begin{pmatrix} x_{1,1}^{vc} \\ \vdots \\ x_{1,d_{vc}}^{vc} \end{pmatrix} & \dots & \begin{pmatrix} x_{1,1}^{vc} \\ \vdots \\ x_{1,d_{vc}}^{vc} \end{pmatrix} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ x_{n,1}^{sn} & \dots & x_{n,d_{sn}}^{sn} & x_{n,1}^{sc} & \dots & x_{n,d_{sc}}^{sc} & \begin{pmatrix} x_{n,1}^{vn} \\ \vdots \\ x_{n,d_{vn}}^{vn} \end{pmatrix} & \dots & \begin{pmatrix} x_{n,1}^{vc} \\ \vdots \\ x_{n,d_{vc}}^{vc} \end{pmatrix} & \dots & \begin{pmatrix} x_{n,1}^{vc} \\ \vdots \\ x_{n,d_{vc}}^{vc} \end{pmatrix} \end{pmatrix} \quad (1)$$

Where, χ denotes a SALI dataset of n mixed data objects; an individual data object X_i is represented by d ($d = s_n + s_c + v_n + v_c$) attributes $A_1^{sn}, \dots, A_{s_n}^{sn}, A_1^{sc}, \dots, A_{s_c}^{sc}, A_1^{vn}, \dots, A_{v_n}^{vn}, A_1^{vc}, \dots, A_{v_c}^{vc}$;

$A_1^{sn}, \dots, A_{s_n}^{sn}$ are the s_n scalar (soil profile level) numeric/ordinal attributes, $A_1^{sc}, \dots, A_{s_c}^{sc}$ are the s_c scalar (soil profile level) categorical attributes, $A_1^{vn}, \dots, A_{v_n}^{vn}$ are the v_n vector (soil horizon level) numeric/ordinal attributes, and $A_1^{vc}, \dots, A_{v_c}^{vc}$ are the v_c vector (soil horizon level) categorical

attributes; $x_{i,j}^{sn}$ denotes the j^{th} attribute of X_i^{sn} , same description applies to $x_{i,j}^{sc}$; $x_{i,j}^{vnk}$ denotes the k^{th} element in the j^{th} numeric vector attribute of X_i^{vn} , same description applies to X_i^{vc} .

2.2. Data transformations

In most cases, categorical data needs to be converted to numerical data. One hot encoding or entity embedding can be applied to conduct this converting. Therefore, one soil data input may have three dimensions: horizons, attributes, and domain values(one hot or embedding vector). The data's three-dimensional view is as follows(one hot encoding as example):

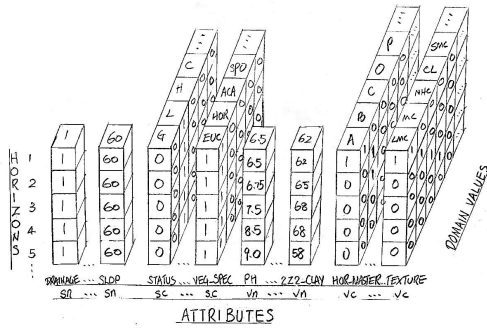


Figure 2: 3D perspective of SALI data

Of these three dimensions, Attributes are homogeneous, Domain Values are heterogeneous, Horizons are between these two, because the order of horizons is the same from top to bottom, but the number of horizon layers, as well as the depth of each horizon, varies.

Based on the analytical needs of different algorithms, soil data can be converted to one dimension, two dimensions, or three dimensions.

1. One-dimensional transformation

Three dimensions are combined into one line to generate one dimensional input data. A few options can be considered in 1D transformation.

Option one: one hot encoding to the "sc" and "vc" attributes and converts horizontal rows of "vn" and "vc" into columns. In this case, a hyper-parameter needs to be pre-defined, which is the number of horizon layers (k2) that would be retained in "vn" and "vc". In sample SALI dataset, k2 obeys Gaussian distribution, so is set to its most common value, which is 5.

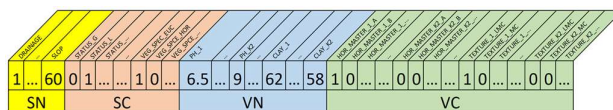
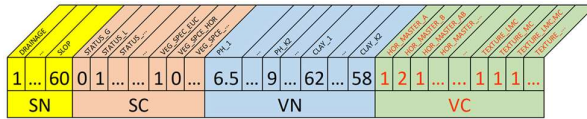


Figure 3: SALI data 1D transformation - Option one: one hot encoding + "rows to columns"

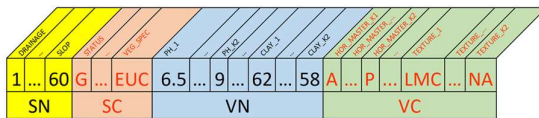
Option two: based on option one, but conduct a n-gram combination on "vc" attributes instead of "rows to column" converting to reduce the size of 1D data. An extra hyper-parameter, the number "n"(k1) of n-gram contiguous sequence for "vc" attributes needs to decide before hand. E.g., an attribute $A = ((a, b, c))$, for $k1 = 2$, then A would have value 1 at (a, b, c, ab, bc) at 1D presentation. Based on the attribute connection between the upper and the lower horizon layer that is often considered in soil analysis, k1 is set to 2.



Domain	SN	SC	VN	VC
1	...	60	0	1
...
60	0	1	...	1
...
1	0
...
6.5	9	...
...
9	62	...
...
58	1
...
1	2	1
...
1	1
...
1	1
...

Figure 4: SALI data 1D transformation - Option two: one hot encoding + n-gram + "rows to columns"

Option three: Decision Trees can work with categorical attributes, so one hot encoding is not required in this option.

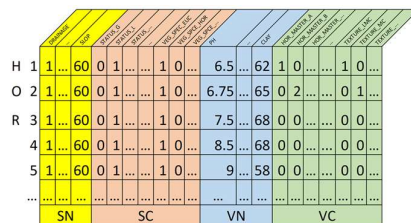


Domain	SN	SC	VN	VC
1	...	60	G	EUC
...
6.5	9	...
...
9	62	...
...
58	A
...
A	P
...
P	LMC
...
LMC
...
...	NA
...

Figure 5: SALI data 1D transformation - Option three: n-gram + "rows to columns"

2. Two-dimensional transformation

Since dimension "Domain Values" are heterogeneous, "Horizons" and "Attributes" are selected to construct 2D data input, although this choice is also imperfect: "Horizon" is not completely homogeneous. In this case, only one parameter k2 needs to be considered, but some redundant data is generated ("sn" and "sc"). This is a sparse matrix.

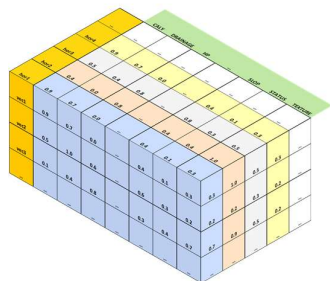


Domain	SN	SC	VN	VC
H 1	...	60	0	1
...
60	0	1	...	1
...
1	0
...
6.5	9	...
...
9	62	...
...
58	1
...
1	2	1
...
1	1
...
1	1
...

Figure 6: SALI data 2D transformation - one hot encoding

3. Three-dimensional transformation

Three-dimensional data is welcome by Deep Learning Neural Networks, and usually it will performance better then 2D inputs at the cost of more parameters. Entity Embedding is conducted for the domain values of every attributes in every level of horizons to generate below 3D input for one SALI data.



Domain	SN	SC	VN	VC
H 1	...	60	0	1
...
60	0	1	...	1
...
1	0
...
6.5	9	...
...
9	62	...
...
58	1
...
1	2	1
...
1	1
...
1	1
...

Figure 7: SALI data 3D transformation – entity embedding

2.3. Missing data processing

Based on the SALI datasets provided, the total missing rate in the attributes is calculated to be 49%.

The general consideration is : If experts can determine a label for data with missing values for certain attributes, machine learning should be able to as well. Therefore, data with some missing values are not deleted.

In this project, the general rule for missing values is, if the algorithm allows, treat them as missing values (NA), otherwise use 0 instead.

	name	EMPTY_RATE	EMPTY_	NON_EMPTY_	TOTAL
9	VALUE_13Cl_Fe	99.88%	206158	251	206409
11	VALUE_6B_GA	95.32%	196745	9664	206409
13	VALUE_ZZ2_Clay	93.27%	192516	13893	206409
10	VALUE_15N1	92.38%	190676	13733	206409
14	HOR_PREFIX	89.96%	185695	20714	206409
19	CUTAN_TYPE	89.65%	185054	21355	206409
17	HOR_SUFFIX	84.56%	174548	31861	206409
12	VALUE_2A1	82.13%	169521	36888	206409
22	NATURE	73.80%	152339	54070	206409
7	SOIL_WATER_STAT	65.50%	135202	71207	206409
20	PEDALITY_TYPE	46.47%	95908	110501	206409
16	HOR_SUBHOR	46.06%	95065	111344	206409
6	BOUND_DISTINCT	42.39%	87496	118913	206409
1	DRAINAGE	26.61%	54929	151480	206409
8	FTS_PH	21.05%	43447	162962	206409
3	STATUS	15.82%	32646	173763	206409
2	ELEM_TYPE_CODE	12.59%	25985	180424	206409
21	PEDALITY_GRADE	12.34%	25475	180934	206409
18	TEXTURE_CODE	4.83%	9975	196434	206409
15	HOR_MASTER	0.91%	1889	204529	206409
4	UPPER_DEPTH	0.00%	0	206409	206409
5	LOWER_DEPTH	0.00%	0	206409	206409
23	By_rows	100.00%	206409	0	206409
24	Total	49.80%	2261260	2279738	4540998

Figure 8: SALI data's missing data rate

2.4. Feature selection

\mathcal{X} contains hundreds of attributes. Feature selection is necessary. First of all, not all features are relevant to the study object; the relevant features vary depending on the object of study; relevant features may be highly correlated with each other and therefore redundant. Further more, curse of dimensionality is a critical issue when algorithms are applied to high-dimensional data; models tend to overfit with a large number of features, which may cause performance degradation on unseen data; increasing data dimensions can significantly increase the storage and computational requirements.

Of the two main components of dimensionality reduction (feature extraction and feature selection), only feature selection is mentioned here, because by retaining the original features, feature selection maintains the physical meaning of the original features, making the model more readable and interpretable.

Thanks to my industry supervisors, Peter Zund and Evan Thomas, expert knowledge-based feature selection is applied to extract following attributes for analysis of ASC and SPC:

NO.	Attributes	Description	Dataset	Data type	Data Structure	Attribute type	Number of attribute values
1	SALSOSSDRAINAGE	Ordinal measure of drainage behaviour at a site.	soil profile	ordinal	scalar	sc	6
2	SALSOSSITEM_TYPE_CODE	Landform element types	soil profile	categorical	scalar	sc	74
3	SALSOSSSTATUS	Soil surface condition	soil profile	categorical	scalar	sc	54
4	SALSOSS_PH	Measure of soil pH in the field	soil horizon	numeric	vector	nm	161
5	SALSHORSLIPPER_DEPTH	Upper depth of a horizon measured from the soil surface	soil horizon	numeric	vector	nm	7710
6	SALSHORSLIPPER_DEPTH	Lower depth of a horizon measured from the soil surface	soil horizon	numeric	vector	nm	771
7	SALSHORSLIPPER_DEPTH	How wet a soil horizon was when observed	soil horizon	ordinal	vector	nm	5
8	SALSHORSLIPPER_DEPTH	Ordinal measure of the sharpness of the colour changes in a mottled horizon	soil horizon	ordinal	vector	nm	7
9	SALSHORSLIPPER_DEPTH	Measure of Fe percentage in a horizon	soil horizon	numeric	vector	nm	161
10	SALSHORSLIPPER_DEPTH	Exchangeable sodium percentage in a horizon	soil horizon	numeric	vector	nm	1500
11	SALSHORSLIPPER_DEPTH	Organic carbon percentage in a horizon	soil horizon	numeric	vector	nm	1006
12	SALSHORSLIPPER_DEPTH	Measure of soil pH in lab	soil horizon	numeric	vector	nm	511
13	SALSHORSLIPPER_DEPTH	Clay percentage in a horizon	soil horizon	numeric	vector	nm	946
14	SALSHORSLIPPER_DEPTH	Horizon master designations	soil horizon	categorical	vector	nm	31
15	SALSHORSLIPPER_DEPTH	Fully describing of a horizon name	soil horizon	categorical	vector	nm	2732
16	SALSHORSLIPPER_DEPTH	Field texture grade	soil horizon	categorical	vector	nm	91
17	SALSHORSLIPPER_DEPTH	List of allowable cutan types	soil horizon	categorical	vector	nm	7
18	SALSHORSLIPPER_DEPTH	Modification of the texture, structure or fabric at natural surfaces in soil materials	soil horizon	categorical	vector	nm	14
19	SALSHORSLIPPER_DEPTH	Degree of development and distinctness of soil peds in a horizon	soil horizon	categorical	vector	nm	6
20	SALSHORSLIPPER_DEPTH	Allowable types of soil segregation (composition)	soil horizon	categorical	vector	nm	15

Figure 9: ASC features selection

NO.	Attributes	Description	Dataset	Data type	Attribute type	Adjusted attribute count
1	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
2	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
3	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
4	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
5	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
6	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
7	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
8	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
9	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
10	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
11	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
12	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
13	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
14	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
15	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
16	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
17	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
18	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
19	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18
20	SALSHORSLIPPER_DEPTH	Horizon has 18 states but for SPC definition we only use the unabbreviated states which are 0=hardsetting, 1=loose, 2=soft, 3=firm, 4=cracking, 5=soil-moisture, 6=soil-moisture, 7=cracking, 8=cracking, 9=cracking, 10=cracking, 11=cracking, 12=cracking, 13=cracking, 14=cracking, 15=cracking, 16=cracking, 17=cracking, 18=cracking	soil horizon	categorical	vector	18

Figure 10: SPC features selection

3. ASC Analysis

3.1. Hierarchical Classes

One case worth considering in ASC analysis is that its target output is hierarchically structured (Figure 11.). ASC is a four-layer classification system: ORDER---SUBORDER---GREAT_GROUP---SUBGROUP.

E.g., a SALI soil data can be 8classified88 as “CH AA AH AT”, this would decode as Bleached(AT), Eutrophic(AH), Red(AA) Chromosol(CH).

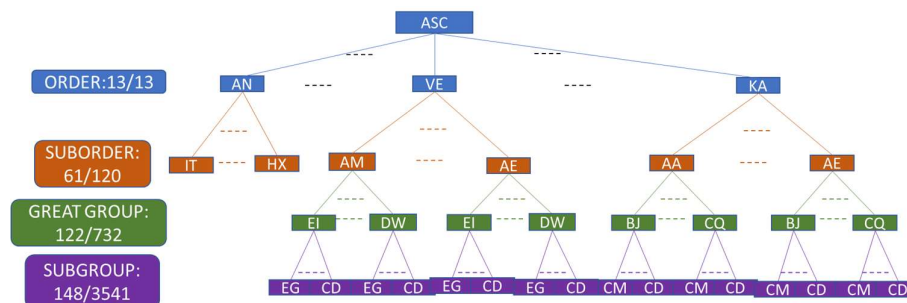


Figure 11: ASC's hierarchical structure

One of the biggest challenges in this 8 hierarchical classification structure is the so-called “rare categories” problem, that is :there are too many leaf nodes at the end of the hierarchy that most of them don’t have enough samples to train on. Figure 11. Shows that on the last layer, SUBGROUP, there are 148 classes by itself, but have 3541 hierarchical classes. Figure 12. Shows there are 83.2% of leaf nodes with less than 10 instances.

A few options can be considered to improve “rare categories” problem, First, further analysis on SALI dataset revealed that the same sub-layer classification has the same characteristics despite the different classifications of the previous layers. E.g., “CH AA AH AT” and “CH AB AH AT” should have same characteristics for 8“AT” SUBGROUP. Therefore, 4 models can be built based on 4 single layers (instead of hierarchical layers) to reduce the amount and percentage of “rare categories”. Figure 12. Shows that, the amount and percentage of “rare categories” drops from 3541 and 83% to 157 and

49% when the model is designed based on single layer class SUBGROUUP instead of hierarchical9-layer class “ORD-SO-GG-SG”. Second, sample filtering could be conducted to ignore “rare categories”. After all, there is no way to train a valid model without data. Number of classes will drop from 157 to 80 if 49% of “rare categories” are filtered out on SUBGROUP model.

ASC Rare Categories						
		NO. of Nodes		%(Sample_No.<=10)		Total Sample No.
single layer	Hierarchical layer	single layer	Hierarchical layer	single layer	Hierarchical layer	
ORD/13	ORD/13	13	13	0.0%	0.0%	42949
SO/61	ORD-SO/120	64	120	48.4%	46.0%	42949
GG/122	ORD-SO-GG/732	124	732	51.6%	66.0%	42949
SG/148	ORD-SO-GG-SG/3541	157	3541	49.0%	83.2%	42949

Figure 12: ASC’s “rare categories”

There are also other issues in hierarchical classifications. Such as:

Internal node prediction, which means some classes may not have full hierarchy subclasses, e.g., The “AN” ORDER class contains only the SUBORDER subclass, but not the GREAT GROUP and SUBGROUP subclasses. Introducing upper-layer class outputs as weights during training is one option for dealing with this matter.

Feature selection (varies by node/level), which means that different subclasses may have different requirements for data inputs. This is not an issue if models for subclasses are 9separated9 (such as Decision Trees). However, in the case of combinatorial models, consideration should be given to adding additional inputs to the model at the appropriate stage in the model design.

The algorithm design in this task is mainly focusing on how to deal with these issues.

3.2. Algorithms Analysis

3.2.1. MLP (Multi-Layer Perceptron)

Figure 13. --- Figure 15. Show the general setting and design of MLP:

MLP Algorithm settings	
Input data Format:	1D data input (Figure 4.)
Missing data Treatment:	Missing as 0
Hierarchical Clasification Method:	Local Classifier per Level (LPL) (Option 1) Local classifier per node (LCN)(Option 2)
Neural Network Architecture:	Option 1: Multi-Classes Model (Figure 14.) Option 2: Binary-Classes Model (Figure 15.)
Activation Functions:	Relu, and Sigmoid(Option 1) or softmax(Option 2)
Loss Function:	Crossentropy
Optimizer:	Rmsprop
Hyper-parameters:	k1(n-gram for sc, vc features):2 k2(No. of horizon layers): 5
Performance Evaluation Indicators:	ACC Top K matching for >90% ACC

Figure 13: MLP algorithm setting

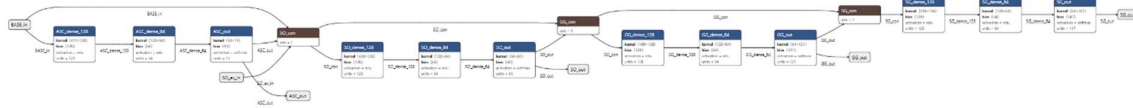


Figure 14: MLP architecture – Option 1: Multi-Classes Model (generated by NETRON)

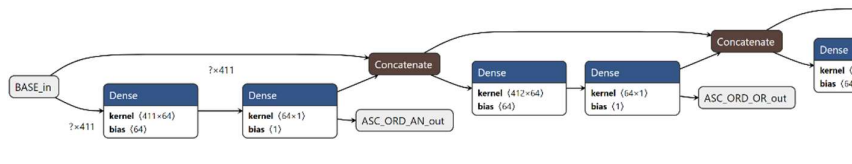


Figure 15: MLP architecture – Option 2: Binary-Classes Model (generated by NETRON)

Two designs to implement MLP. Option 1 uses 4 similar models run together to address 4 single layer subclasses. They are all multi-classes models, upper layer's outputs are concatenated as next layer's inputs, and extra features input can be added to models. Option 2 uses 344 binary models to fit every class(not layer) of all 4-layer classes. The experiment results are presented in Figure 16.:

MLP Total ACC (on test data)													
		Multi-Classes Model # of Parameters: 303,893 # of models: 4						Binary-Classes Model # of Parameters: 12,828,693 # of models: 344					
		First Matching		Top 2 Matching		Top K Matching for >90% ACC		First Matching		Top 2 Matching		Top K Matching for >90% ACC	
single layer	Hierarchical layer	single layer	Hierarchical layer	single layer	Hierarchical layer	single layer	Hierarchical layer	single layer	Hierarchical layer	single layer	Hierarchical layer	single layer	Hierarchical layer
ORD/13	ORD/13	67.9%	67.9%	86.2%	86.2%	3	3	71.5%	71.5%	87.9%	87.9%	3	3
SO/61	ORD-SO/120	64.0%	47.4%	84.9%	75.6%	3	4	64.6%	49.7%	86.3%	77.3%	3	4
GG/122	ORD-SO-GG/672	36.0%	21.3%	50.7%	42.8%	12	16	45.7%	27.8%	62.0%	52.9%	9	12
SG/148	ORD-SO-GG-SG/2389	25.8%	8.4%	33.6%	20.6%	NA	NA	36.8%	14.1%	49.0%	33.1%	NA	NA

Figure 16: MLP Total ACC on test data

Hierarchical layers classification ACCs could be calculated since single layers classification ACCs are obtained. For the first matching (only the one with the best record is nominated), MLP is performing normal in the first two layers classes, but not well in deeper subclasses. When consider the Top K matching (the best top k records are nominated as candidates), choosing 3 out of 61 options gives 95% accuracy, so MLP can be a good reference for the first two classes. Binary-classes model outperforms multi-classes model, but at the cost of training a large number of parameters.

3.2.2. DLNN (Deep Learning Neural Networks)

Figure 17. shows the general setting of DLNN:

DLNN Total ACC (on test data)													
		2D Model # of Parameters: 1,260,839 # of models: 4						2D+ Model # of Parameters: 845,357 # of models: 4					
		First Matching		Top 2 Matching		Top K Matching for >90% ACC		First Matching		Top 2 Matching		Top K Matching for >90% ACC	
single layer	Hierarchical layer	single layer	Hierarchical layer	single layer	Hierarchical layer	single layer	Hierarchical layer	single layer	Hierarchical layer	single layer	Hierarchical layer	single layer	Hierarchical layer
ORD/13	ORD/13	73.4%	73.4%	90.1%	90.1%	2	2	74.4%	74.4%	91.2%	91.2%	2	2
SO/61	ORD-SO/120	66.7%	51.0%	87.5%	79.0%	3	4	70.3%	50.0%	88.0%	78.0%	3	4
GG/122	ORD-SO-GG/672	52.9%	32.3%	71.8%	61.3%	5	7	50.0%	30.5%	70.0%	60.0%	5	7
SG/148	ORD-SO-GG-SG/2389	47.8%	19.3%	63.8%	44.2%	10	15	44.6%	17.8%	61.0%	41.4%	10	15

Figure 20: DLNN Total ACC on test data

DLNNs are performing better in deeper two classes than MLPs. 10 out of 148 options would achieve 90% accuracy, DLNN can be a sound reference for the deeper two classes. Both DLNN models perform similar, with 2D+ model uses less parameters.

3.2.3. Decision tree

The initial consideration in choosing DT as an alternative model was that neural networks are generally not very easy to interpret, while DT is just the opposite: it is easy to understand and interpret. However, this is not the case if the horizon rows are converted to columns to generate 1D input in SALI data. This would make the decision tree very large and difficult to interpret. One solution to control its interpretability is to extract some rules from the ASC book and rearrange the data to the profile level. This, however, detracts from the theme of this project: machine learning based on the information available in the data. A decision tree that drastically changes the structure of the existing SALI data is not the same as above Neural Networks. So, after consideration, it was decided not to change the data drastically and to maintain it in its original basic state, i.e., having both PROFILE and HORIZON structures. As a result, DT loses its advantages and will not be analysed in depth in this report.

Another issue of DT is revealed during the algorithm analysis process: the training model should include all 12 possible domain values of categorical attributes. If the new data to be predicted contains unknown attribute values, the model will be in error.

Figure 21. shows the general setting of DT:

DT Algorithm settings	
Input data Format:	1D data input (Figure 5.)
Missing data Treatment:	Missing as Missing (NA)
Hierarchical Classification Method:	Local Classifier per Level (LPL)
Impurity Measurement:	Entropy
Pruning Process:	Post-Pruning
Hyper-parameters:	k2(No. of horizon layers): 5
Performance Evaluation Indicators:	ACC

Figure 21: DT algorithm setting

And the results as follow:

DT Total ACC (on test data)	
ORD/13	67.2%
SO/61	62.7%
GG/122	42.5%
SG/148	37.0%

Figure 22: DT Total ACC on test data

The Figure reveals DT is not performing well in deeper two classes.

4. SPC Analysis

This task has two steps. Step one aims to correlate and merge existing SPCs; step two implements similar process as described in Task 1 to identify(or re-identify) untagged data.

4.1. Step One: SPC Clustering

The processing of this step is: first, design a proper method to measure the distance between two data; then another design to calculate the centroids for a group of data(SPCs); after that, a distance matrix of SPCs can be utilized to generate a hierarchy; and finally, a threshold could be decided to cut the tree and correlate the SPCs.

4.1.1 Data Exploration

Before above processing, a data exploration could always be helpful. There are 1848 SPCs in SALI dataset, Figure 23. Shows the histogram plot for the frequency of instances number in SPCs:

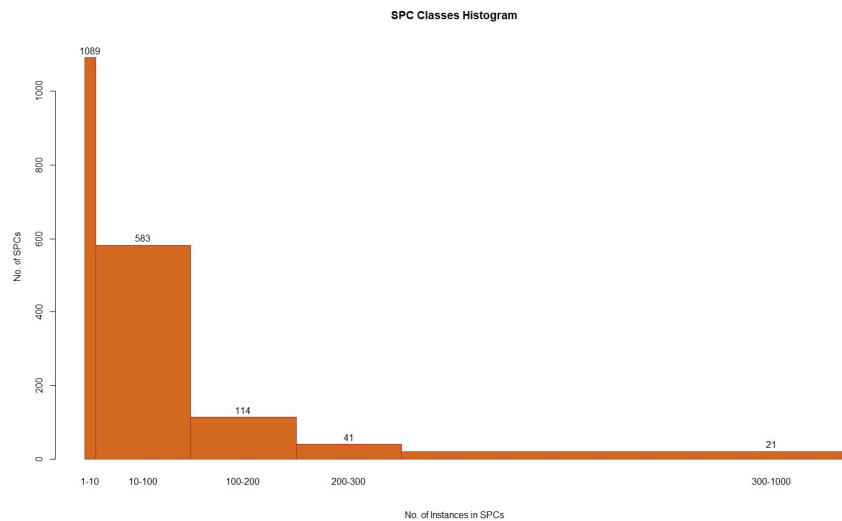


Figure 23: Histogram for the frequency of instances number in SPCs

The histogram plot shows that 1089 SPCs contains less than 10 instances each. And these 1089 SPCs only nominated 3501 instances. So, the first step to reduce the sizes of SPCs is just delete these 1089

SPCs! And these 3501 instances can be renominated with new SPC Clusters after supervised learning models are fitted (step two).

That makes 759 SPCs left. A Hierarchical Clustering processing will be conducted to further correlate and merge these 759 SPCs.

4.1.2. Similarity, Centroid, and Distance Matrix

1. Hierarchical clustering is a distance/similarity based algorithm. So, an important step is to design a suitable method for measuring the similarity between two data. In proposal report, a complex method is demonstrated. The method is effective, but not efficient. In order to improve this similarity measurement method, the SALI data is converted to one-dimensional data, as shown in figure 4. The similarity formula between the two data x_p and x_q is given as:

$$S_{x_p, x_q} = \frac{1}{d} (sim_{sn} + sim_{vn} + sim_{sc,vc}) \quad (2)$$

d : number of features in SALI data

$$sim_{sn} = \sum_{i=1}^{d_{sn}} e^{-\frac{(x_{p,i}^{sn} - x_{q,i}^{sn})^2}{2}} \quad (3)$$

$p, q \in N, N$: number of instances, d_{sn} : number of sn type features in SALI data x

$$sim_{vn} = \frac{1}{k2} \sum_{i=1}^{d_{vn}} \sum_{j=1}^{k2} e^{-\frac{(x_{p,i}^{vnj} - x_{q,i}^{vnj})^2}{2}} \quad (4)$$

$k2$: number of horizon layers, d_{vn} : number of vn type features in x

$$sim_{sc,vc} = \sum_{i=1}^{d_{sc+vc}} \left(1 - \frac{1}{2} \sum_{j=1}^{d_i} \left| \frac{x_{p,i}^{(sc/vc)j}}{\sum_{k=1}^{d_i} x_{p,i}^{(sc/vc)k}} - \frac{x_{q,i}^{(sc/vc)j}}{\sum_{k=1}^{d_i} x_{q,i}^{(sc/vc)k}} \right| \right) \quad (5)$$

d_i : number of domain values in feature i , $sc + vc$: number of sc & vc type features in x

2. The centroid of the SPC, which represents the centroid of a subset of datasets belonging to the same SPC. A centroid data C_i ($i = 1, 2, \dots, 759$, denotes the number of SPCs), its vector value is calculated as follows:

$$c_{i,j} = \frac{1}{N_i} \sum_{x \in C_i} x_{.,j} \quad (6)$$

$j = 1, 2, \dots, d'$, denotes the number of columns in SALI data, N_i : number of $x \in C_i$,

3. Based on equation (2) and (6), a distance matrix D can be created, its element value is calculated as follows:

$$dist_{i,k} = 1 - \sum_{j=1}^{d'} w_j S_{c_{i,j}, c_{k,j}} \quad (7)$$

where $w_j = \frac{1}{d \times d_j}$, is the weight for column j and $\sum_{j=1}^{d'} w_j = 1$.

This distance measurement obeys following three rules:

Nonnegativity : $dist_{i,j} \geq 0$, and $dist_{i,i} = 0$

Symmtry : $dist_{i,j} = dist_{j,i}$

TrianleInequality : $dist_{i,j} \leq dist_{i,k} + dist_{k,j}$

	2Uge	2Dyb	Md	Sp	Pg	Kp	2Ugd
2Uge	0.00000000	0.03547669	0.08036042	0.07530292	0.09575315	0.10347298	0.02430231
2Dyb	0.03547669	0.00000000	0.08369345	0.06340684	0.10594685	0.09492472	0.04566526
Md	0.08036042	0.08369345	0.00000000	0.04095939	0.06316302	0.06254663	0.08276120
Sp	0.07530292	0.06340684	0.04095939	0.00000000	0.07202928	0.05348858	0.07446782
Pg	0.09575315	0.10594685	0.06316302	0.07202928	0.00000000	0.09319240	0.09875080
Kp	0.10347298	0.09492472	0.06254663	0.05348858	0.09319240	0.00000000	0.09739572
2Ugd	0.02430231	0.04566526	0.08276120	0.07446782	0.09875080	0.09739572	0.00000000
Cl	0.07750844	0.07860633	0.03442999	0.04072802	0.07514593	0.04542483	0.07143606
Bl	0.06158128	0.07190435	0.06448292	0.07292907	0.08367826	0.10406691	0.07288497
6Dyg	0.04428675	0.02040755	0.08536344	0.06263228	0.11041046	0.08554190	0.04339812
Ko	0.10830982	0.10367393	0.08700160	0.06972719	0.10515113	0.04389028	0.10022639

Figure 24: Distance Matrix of SPCs

4.1.3. Hierarchical Clustering

A hierarchical clustering algorithm can be implemented based on above distance matrix D along with “Average” agglomeration method. A threshold can be decided depending on the number of clusters desired.

Figure 25. displays the hierarchy tree before and after the threshold cutting.

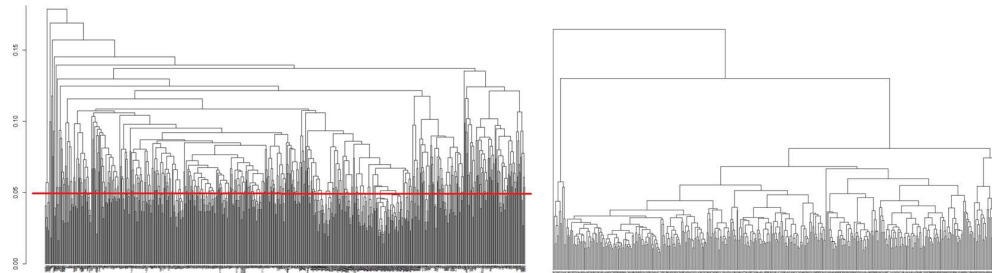


Figure 25: Hierarchy tree of SPCs before and after cutting

Also, a histogram plot for the frequency of number of instances in SPC Clusters illustrates a certain balance in the distribution of Cluster sizes:

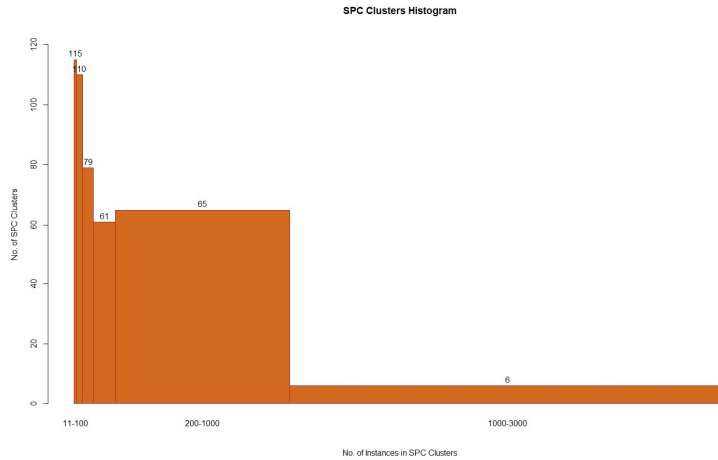


Figure 26: Histogram for the frequency of instances number in SPC Clusters

4.2. Step Two: Supervised learning after clustering

This process is similar to Task 1, only the experimental results are listed, and the process will not be repeated.

SPC Classification ACC (aft. Clustering / on test data)			
	First Matching	Top 2 Matching	Top K Matching for >90% ACC
MLP	72.3%	88.4%	3
DLNN	77.2%	91.3%	2
DT	71.5%	NA	NA

Figure 27: SPC Classification ACC after Clustering on test data

5. Conclusions and discussions

The results show some promise but also challenges in fitting an effective model for classifying/clustering ASC and SPC. Some of the conclusions from this study are:

1. The features of SALI soil dataset are characterized by a mixture of data types: numerical, ordinal, and categorical, and a mixture of data value structures: scalar values (soil profile attributes) and vector values (soil horizon attributes). Using techniques such as n-gram encoding, one-hot encoding, and entity embedding, SALI data can be transformed into one-dimensional scalar numerical data(Figure 3. & Figure 4.), one-dimensional scalar mixed data(Figure 5.), two-dimensional vector numerical data(Figure 6.), or three-dimensional sequential vector numerical data(Figure 7.), to meet the data input requirements of different models.
2. Experimental results show that deep learning neural networks(DLNN) are more suitable for hierarchical classifications.
3. Methods for calculating similarity and centroid are proposed for the characteristics of mixed data types and mixed data structures of SALI data. This provides the possibility to apply any distance/similarity based algorithm(e.g., hierarchical clustering).
4. The hierarchical classifications of ASC (ORDER---SUBORDER---GREAT_GROUP---SUBGROUP) pose challenges such as "rare categories", Internal node prediction, and Feature selection varies by node/level. This report provides some preliminary options to address these issues in neural network models. Appropriate model design (e.g., Local Classify per Level, LPL) based on the characteristics of the SALI soil dataset can effectively reduce the impact of the "rare categories". Sample filtering is another available option for ignoring extreme "rare categories". Also, in neural network models, introducing parental classes' outcomes as inputs of children model is equivalent to adding an indication signal to children model, which can make certain effective guidelines for internal node prediction.
5. When a decision tree (DT) model is used to fit SALI data with a vector data structure, it loses the advantage of interpretability due to the large number of attributes generated by one-hot encoding.

Also, the following challenges are worth discussing:

1. ASC's first class, ORDER, has an accuracy of around 70%, which is considerably lower than expected and deserves further study. The following are possible directions for improvement: missing data handling, outlier detection, model fine-tuning, and feature engineering.

2. Performing effective classification in high missing data value scenarios is a worthwhile direction of research. In this project, some preliminary studies have been done and some results have been obtained, such as missing as 0, missing as missing, and similarity calculation along with missing. However, some other attempts seem to be ineffective, such as vector value imputation.
3. This project did not conduct a comprehensive analysis of outlier detection, which may be an important factor affecting the results.

References

- Alpaydin, E., & ProQuest. (2014). Introduction to machine learning (Third ed., Adaptive computation and machine learning). Cambridge, MA: The MIT Press.
- Bhattacharya, B., & Solomatine D.P. (2006). Machine learning in soil classification. *Neural Networks*, 19, 186–195.
- Brungard, C.W., Boettinger J.L., Duniway, M.C., Wills, S.A., & Edwards, T.C. (2015). Machine learning for predicting soil classes in three semi-arid landscapes. *Geoderma*, 239-240, 68–63.
- Cao, F., Yu, L., Huang, J. Z., & Liang, J. (2017). K-mn-modes: an algorithm for clustering categorical matrix-object data. *Applied Soft Computing*, 57, 605–614.
- Carre, F. & Jacobson, M. (2009). Numerical classification of soil profile data using distance. *Geoderma*, 148, 336–345.
- Chollet, F., & Allaire, J.J. (2018). Deep learning with R. Manning Publications Co.
- Ding, S., Du, M., Sun, T., Xu, X. & Xue, Y. (2017). An entropy-based density peaks clustering algorithm for mixed type data employing fuzzy neighbourhood. *Knowledge-based Systems*, 133, 294–313.
- Hsu, C. C. (2006). Generalizing self-organizing map for categorical data. *IEEE Trans. Neural Networks*, 17(2), 294–304.
- Huang, Z. (1997). A fast clustering algorithm to cluster very large categorical data sets in data mining. *Discovery*, 1–8.
- Huang, Z. (1998). Extension to the k-modes algorithm for clustering large data sets with categorical values. *Discovery*, 2(3), 283–304.
- Isbell, R. F., & The National Committee on Soil and Terrain. (2004). The Australian soil classification (2nd ed.). Retrieved from <https://ebookcentral-proquest-com.ezproxy.library.uq.edu.au>
- Minasny, B., & McBratney, A. B. (2007). Incorporating taxonomic distance into spatial prediction and digital mapping of soil classes. *Geoderma*, 142, 285–293.
- Naik, A., & Rangwala, H. (2018). Large scale hierarchical classification : state of the art. Springer.
- National Committee on Soil Terrain, National Committee on Soil Terrain, CSIRO Publishing, & Ebooks Corporation. (2009). Australian soil and land survey : field handbook (3rd ed.). CSIRO Publishing.
- Shepard, R. W. (1987). Toward a universal law of generalization for psychological science. *Science*, 237(4820), 1317–1323.
- Strang, G. (2019). Linear algebra and learning from data. Wellesley-Cambridge Press.
- Zhang, A., Lipton, Z.C., Li, M. & Smola, A.J. (2020). Dive into Deep Learning. Retrieved from <https://d2l.ai>