Deep Learning Architectures for Soil Property Prediction

Matthew Veres, Griffin Lacey, Graham W. Taylor
School of Engineering
University of Guelph
Guelph, Canada
{mveres, laceyg, gwtaylor}@uoguelph.ca

Abstract—Advances in diffuse reflectance infra-red spectroscopy measurements have made it possible to estimate a number of functional properties of soil inexpensively and accurately. Core to such techniques are machine learning methods that can map high-dimensional spectra to real-valued outputs. While previous works have considered predicting each property individually using simple regression methods, the correlation structure present in the output variables prompts us to consider methods that can leverage this structure to make more accurate predictions. In this paper, we leverage advances in deep learning architectures, specifically convolutional neural networks and conditional restricted Boltzmann machines for structured output prediction for soil property prediction. We evaluate our methods on two recent spectral datasets, where output soil properties are shown to have a measurable degree of correlation.

Keywords—deep learning; convolutional neural network; restricted Boltzmann machines; structured output; reflectance spectroscopy; soil analysis;

I. INTRODUCTION

The analysis of soil data offers many opportunities for analyzing agricultural production, monitoring land use, climate regulation, and environmental planning [1], [2]. This analysis directly affects governments who make budgets, pass laws & legislations, cities who rely on urban and rural planning, and farmers who produce food. Indirectly, these analysis affects individuals across the world. Diffuse reflectance spectroscopy is a technique frequently used to collect data for soil property prediction due to its portability and low cost, thus avoiding having to send samples to external laboratories with low turn-around times [3]. The release of two datasets, the Africa Soil Property Prediction Challenge (AFSIS), and the Land Use/Cover Area frame Statistical Survey (LUCAS)1 have created new opportunities for evaluation of machine learning approaches, while also benefiting those who rely on soil statistics.

There are several challenges associated with these two datasets, such as the collected sample scale, the number of observations, regional variability (i.e. due to climate or soil composition), interdependencies between soil characteristics, or uneven sampling across land uses.

1The LUCAS topsoil dataset was made available by the European Commission through the European Soil Data Centre, and managed by the Joint Research Centre (JRC) http://esdac.jrc.eupa.eu/

Recent developments in machine learning, known as “Representation Learning” and its close counterpart “Deep Learning”, have made well-publicized breakthroughs in transforming massive amounts of unstructured data into a number of successful applications. These methods are based on learning multiple layers of representation from raw data without resource-intensive human engineering. Applications to-date have focused on machine perception problems on rich sensory data such as images [4], [5], video [6], [7], audio [8], and natural language [9].

Although more than 20 years old, the most popular deep learning architecture in recent years is the convolutional neural network [10]. Convnets, like their standard deep neural network counterparts, perform end-to-end feature learning and are trained with the backpropagation algorithm. However, they differ in a number of architectural respects, which makes them particularly effective for modeling the types of unstructured sensory data described above. In this paper, we consider a 1-dimensional convnet, the type that has been successful for modeling speech [11], [8], applying it to soil property prediction.

Supervised deep learning methods such as convnets have traditionally considered tasks with simple, independent outputs, such as image classification [4], object detection [12], and facial expression recognition [13]. However, many problems have outputs which are high-dimensional and interdependent. The class of structured output prediction makes the treatment of complex, highly structured output explicit, though unlike feed-forward deep learning models, structured output methods must consider an exponential number of output configurations and therefore are burdened computationally. Soil analysis falls under the domain of structured output when we consider the prediction of multiple correlated properties, for example, calcium, phosphorus, pH levels, soil organic carbon, and sand content. Whereas previous approaches have predicted soil properties individually, exploiting the interdependence among the variables may improve the accuracy of the predictor.

While many options for structured output prediction methods exist [14], including those that use representation learning [15], in this paper we consider Conditional Restricted Boltzmann Machines [16], [17]. Best known for modeling time series data, they were recently explored by Mnih et
al. for structured output problems [18]. While those authors condition an RBM modeling output structure directly on the input, we explore a simple extension of this technique where we condition on the penultimate layer of a feed-forward neural network which has been trained to predict output from input. This enables us to pre-train the input-output mapping (NN) and the output structure (RBM) independently, then fine-tune the combined model.

Our main contribution is in applying structured and unstructured deep learning architectures to soil property prediction: a domain, to the best of our knowledge, that has received virtually no attention from the deep learning and structured output prediction communities. We evaluate convolutional neural networks, and variants of CRBMs for structured output prediction on two soil property prediction datasets that have been generated through reflectance spectroscopy. The first, AFSIS, which was the subject of a Kaggle competition that ended in late 2014, contains just under 2,000 soil samples that were collected from across sub-Saharan Africa [19], [20]. Second, we evaluate our method on a larger dataset, the Land Use/Cover Area frame Statistical Survey (LUCAS) which contains over 19,000 geo-referenced soil measurements that were collected from across 23 member states of the European Union [2].

II. RELATED WORK

In this section, we briefly describe work related to our main modeling frameworks: Convolutional Neural Networks and Conditional Restricted Boltzmann Machines. Details of the models are reserved for Section III. We also provide a brief review of existing work which has used diffuse reflectance infra-red spectroscopy for soil property prediction.

A. Convolutional Neural Networks

Convnets have been described as an “astounding baseline for recognition” [21] and have been applied to tasks such as object detection and recognition [4], [12], facial expression recognition [13], scene parsing [22] and pose estimation [23], [24]. Although 2-dimensional convnets for pixel-based inputs are the most frequently used, the 1-d convnet has recently made exciting progress in speech recognition [11], [8], and understanding text at the character level [25].

Part of the success of the convnet architecture relies on the significantly reduced number of parameters in the network, due to weight sharing on the convolution filters. Convolutional networks, like traditional feed-forward networks are also amenable to hardware acceleration, which has allowed for much larger, and diverse datasets to be explored.

B. Conditional Restricted Boltzmann Machines

The majority of research on CRBMs to-date, demonstrates their applicability to modeling high-dimensional time series data. Taylor et al. [16] considered the use of CRBMs in modeling distinct human motions, while Zeiler et al. successfully used them to model the complex courtship behaviours of pigeons [26]. Boulanger-Lewandowski et al. used recurrent neural networks to condition an RBM for music generation and transcription [27]. Mandel et al. applied CRBMs to music auto-tagging [28].

CRBMs have also been applied to static data. For example, Mnih et al. [18] evaluated CRBMs on prediction tasks where the output space is arbitrarily structured, such as pixel labeling on a corrupted version of the MNIST dataset, as well as multi-label classification. Recognizing that standard contrastive divergence-based learning is not optimal for training CRBMs in a predictive context, the authors proposed two alternative learning algorithms for structured output problems.

Related recent work in computer vision has used RBMs to model global shape information, and then used the model as a prior over a locally-connected conditional random field (CRF) [29], [15]. This is different than the problem we consider here, as their image-based outputs are expected to contain some spatial structure which is exploited by the local connectivity of the CRF. The downside of such approaches is that inference is computationally demanding in order to cope with an exponential number of possible outputs. This is dealt with by mean-field inference [29] or expectation-maximization [15].

C. Reflectance Spectroscopy

In soils, specific wavelengths in the visible, near-infrared, and mid-infrared regions have been demonstrated to be an indicator of several key soil properties, and a good summary of techniques for predicting these properties can be found in Rossel et al. [30]. The techniques they consider predict individual soil properties from the spectra, and there is no single model that outperforms the others across all soil properties. However, among the models they report, principal component regression (PCR), and partial least squares regression (PLSR) were frequently found to work well. Several recent works also incorporate auxiliary variables (such as sand) to improve model accuracy when predicting outputs such as organic carbon content [31], [3].

The first dataset we consider, AFSIS, contains five targets for prediction: calcium, phosphorus, pH levels, soil organic carbon, and sand content. The goal of the Kaggle competition was to score the lowest mean-columnwise root mean squared error (MCRMSE) as defined below:

\[
\text{error} = \frac{1}{m} \sum_{j=1}^{m} \sqrt{\frac{1}{n} \sum_{i=1}^{n} (Y_{ij} - \hat{Y}_{ij})^2}
\]

where \(\hat{Y}_{ij}\) is the predicted output, \(Y\) is the label of sample \(i\) in column \(j\), \(n\) is the number of observations, and \(m\) is the number of output features. We adopt this metric to compare models on both the AFSIS and LUCAS datasets.
As is common in such competitions, the three winning solutions to AFSIS took an ensemble approach at predicting the outputs listed above, using many combinations of algorithms, similar to the techniques reported in [30].

The first place solution used an advanced pre-processing pipeline to reduce noise and dimensionality, and used an ensemble of neural networks, support vector regression (SVR), Gaussian processes (GP), and multivariate regression for prediction to achieve an error rate of 0.46892. The second place competitor approached the problem by first reducing the dimensionality of the dataset by applying a Hamming window on the spectral data, then carefully removing uninformative inputs and calculating a derivative. This approach used two different training/testing sets; one for predicting the four targets Ca, P, pH, SOC and another solely dedicated to predicting sand content. This second place solution scored an MCRMSE of 0.46916. Finally, the third solution used careful pre-processing, feature selection, and a combination of neural networks, Gaussian processes, and SVRs to achieve an MCRMSE of 0.47016.

The second dataset under consideration, LUCAS, has been predominantly used for the prediction of organic carbon levels, such as that reported by Stevens et al. [31] who tested the accuracy of: PLSR, boosted regression, random forests, Support Vector Regression (SVR) and several others. Critical to the technique is a complex pre-processing pipeline which includes Savitzky-Golay smoothing, as well as first derivative filters.

III. METHODS

In this section, we give an outline of the basic architectures: the feed-forward neural network, the 1-d convolutional network, and the CRBM. We then discuss the procedure by which the CRBM is conditioned on the penultimate hidden layer of a trained NN. Details on training each of these components is reserved until Section IV.

A. Feed-Forward Neural Networks

Feed-forward NNs are the most basic deep learning architecture. The specific NN presented here is a multi-layer perceptron consisting of \( L \) hidden layers and one output layer. The network is shown in Figure 1. The units in each hidden layer \( H \) are a nonlinear function of the layer below, parameterized by weights \( W \) and biases \( b \):

\[
H^j_i = \alpha \left( \sum_{i=1}^{n} W^j_i H^i_{i-1} + b^j_i \right),
\]

where we take \( H^0 = X \). In our experiments, rectified linear units (ReLU) are used for the activation: \( \alpha(\cdot) = \max(0, \cdot) \).

We consider a regression problem, where for the final (output) layer, the outputs are a linear function of the penultimate hidden layer:

\[
Y_j = \sum_{i=1}^{n} W^L_j H^L_i + b^L_j.
\]

B. Convolutional Neural Networks

CNNs differ from simple feed-forward NNs in that convolutional filters replace the “fully-connected” weight matrices, allowing the sharing of local features and resulting in a significant decrease in the number of free parameters.

The response to each filter are “features” which can be efficiently computed by a convolution operation. In our application, the signal is 1-dimensional, so we consider a discrete, 1-d convolution of a filter \( W \) with a signal \( X \) as defined by

\[
H[n] = W[n] \ast X[n] = \sum_{u=-\infty}^{\infty} W[u] X[n - u]
\]

where \( n \) and \( u \) are discrete indices.

Similar to the weights and biases in neural networks, the filters \( W \) are updated by backpropagation. Convolutional layers process several “input maps” and produce “output maps” and there is a filter which connects every input-output map pair. The first set of input maps represent the data. Often, there are multiple input maps at the first layer (e.g. channels of a colour image), but in our case we have a single input map which represents the measured spectral bands.

Each convolutional layer is typically followed by a subsampling layer which uses a simple operation such as max average to reduce the dimensionality of the output map. In our experiments, we take the maximum over each small window within each map. The pooling operation also provides a small amount of translational invariance. In our application, this amounts to small translations across the spectral bands. Figure 2 demonstrates the difference between a traditional hidden layer, and one with a convolution/pooling stage.

C. Conditional Restricted Boltzmann Machines

A restricted Boltzmann machine is an undirected graphical model designed for unsupervised feature learning. It learns
a joint probability distribution over a vector of visible units \( V \) and vector of hidden \(^2\) units \( G \). In our application, we consider the visible units of the RBM to represent soil properties, that is, the target variables of the prediction problem. In this case, the hidden units \( G \) are intended to capture the interdependency of the target variables. We emphasize that in this model, unlike the neural network, connections between units are symmetric and undirected. To model real-valued data, the RBM used has binary logistic hidden units and real-valued Gaussian visible units. The joint probability over these units is

\[
p(V, G) = \exp(-E(V, G))/Z,
\]

where \( Z \) is the partition function which involves a sum over all possible hidden states and integration over the visible space and thus is intractable to compute. The energy function \( E \) is defined as

\[
E(V, G) = \sum_i \frac{(V_i - a_i)^2}{2\sigma_i^2} - \sum_{ij} W_{ij} \frac{V_i}{\sigma_i} G_j - \sum_j b_j G_j,
\]

where \( W \) is a pairwise weight between visible and hidden units, \( a_i \) is the bias for the \( i \)th visible unit, \( b_j \) is the bias for the \( j \)th hidden unit, and \( \sigma \) is the standard deviation of the Gaussian noise for the \( i \)th visible unit.

A conditional restricted Boltzmann machine is an extension of a restricted Boltzmann machine, where the hidden and visible units are conditioned on an additional fixed input (referred to as the conditioning layer). The conditioning layer typically contains some meaningful context regarding the visible-hidden mapping. For example, it is common to condition on a short history of a time series in sequence modeling [32]. The CRBM is depicted in Figure 3.

A CRBM, conditions both the hidden and visible layers on an additional fixed input, which we denote by \( U \). Conditioning corresponds to additional directed connection from \( U \) to the hidden layer, and from \( G \) to the visible layer. Despite this additional complexity in the architecture, inference and learning are no more complicated than for the RBM. The joint probability over this model is now conditioned on this new vector \( U \):

\[
p(V, G|U) = \exp(-E(V, G|U))/Z(U).
\]

The energy function becomes

\[
E(V, G|U) = \sum_i \frac{(V_i - a_i)^2}{2\sigma_i^2} - \sum_{ij} W_{ij} \frac{V_i}{\sigma_i} G_j - \sum_j \hat{b}_j G_j,
\]

where \( \hat{b}_j \) is a dynamic bias which includes the static bias, as well as the contribution of the conditional vector. By including additional context, conditioning should have a positive effect on the performance of the CRBM if the conditional vector contains meaningful information about the output. Though the partition function is intractable due to the exponential number of possible hidden configurations and continuous nature of \( V \), we can integrate out the hidden units since they are binary. As such, we can write the conditional probability of \( V|U \) as

\[
p(V|U) = \sum_G p(V, G|U) = \sum_G \exp(-E(V, G|U))/Z(U).
\]

The unnormalized log probability is known as the “free energy”:

\[
F(V|U) = \sum_i \frac{(V_i - a_i)^2}{2\sigma_i^2} - \sum_j \log(1+\exp(\sum_i W_{ij} V_i + \hat{b}_j))
\]

which permits tractable scoring for a fixed setting of the parameters, which can be used to find an optimal setting of \( V \) given \( U \) as opposed to Gibbs sampling.

D. Conditioning the CRBM with a Feed-Forward NN

Finally we consider a hybrid variant of the NN and CRBM, as seen in Figure 4. The CRBM is conditioned on the penultimate hidden layer of the NN, \( U = H_2 \). The intuition here is that the input-mapping can be learned first using the NN, and the output structure can be learned first with the RBM, then they can be integrated and fine-tuned to work together. The result of applying the hybrid model is prediction that can essentially be “cleaned up” to respect output structure by sampling (or optimizing free energy) within the CRBM. Details of training each of the models are presented in the following section.

\(^2\)G is used instead of \( H \) here to avoid confusion with the hidden layers of the standard NN.
In this section, we describe the steps to train each of the architectures described above.

A. Feed-Forward Neural Network and Convolutional Neural Network Learning

The feed-forward neural network and convolutional neural network were both trained using standard error back-propagation. The MCRMSE (Equation 1) was used as the loss. We used mini-batch stochastic gradient descent, (with batch size of 10) which incorporated learning rate decay (0.01% per epoch) and momentum of 0.9. We used a relatively low amount of dropout for regularization, of approximately $0 \leq p \leq 0.3$. We also used a small amount of weight decay, with $\alpha = 0.001$.

B. Conditional Restricted Boltzmann Machine Training

The free energy equation for the CRBM model is seen in Equation 10. CRBMs would ideally be trained using gradient descent in negative log conditional likelihood, denoted by $-l(\theta)$. The gradient can be obtained by first describing the log-likelihood as follows:

$$p(V|U) = \log \exp(-F(V)) - \log \sum_{V'} \exp(-F(V'|U))$$

By differentiating the log-likelihood with respect to a parameter $\theta$, the gradient is found as follows:

$$\frac{\partial -l(\theta)}{\partial \theta} = \frac{\partial F(V|U)}{\partial \theta} - \sum_{V'} \frac{\partial F(V',U)}{\partial \theta} p(V'|U)$$

The first term in Equation 12 is known as the positive gradient, and can be computed exactly. The second term in Equation 12 is known as the negative gradient, and is intractable. This is similar to the case for unconditional RBMs, as $p(V|G,U)$ and $p(G|V,U)$ are both factorial over $V$ and $G$ [18]. Since the negative gradient is intractable, an approximation to maximum likelihood is necessary.

A practical method for training RBMs, which solves the intractability problem is called contrastive divergence (CD) [33]. CD works by approximating the negative gradient by performing Gibbs sampling for a few steps. However, CD learning has the limitation of computing biased estimates of the gradient. One solution to this problem is by using persistent contrastive divergence (PCD) [34]. PCD works by maintaining a single persistent chain during the negative gradient, instead of creating a new chain during each update. PCD has the advantage of more accurately estimating gradients by better reflecting the true model distribution [18].

However, in CRBMs each conditioning vector leads to a different, unique distribution of $p(V|U)$, making sampling at each training case critical. As such, PCD is not an appropriate algorithm for CRBMs. Therefore we simply adopted standard CD training, though we admit that the more sophisticated algorithms proposed by Mnih et al. may lead to improved results. In our experiments, we used CD-1, as well as a batch size of 100 for training the RBM.

V. Datasets

A. AFSIS Dataset

The goal of the AFSIS dataset is to predict soil functional properties from diffuse reflectance infra-red spectroscopy measurements. Associated with the spectral data are five target variables: sand, pH, soil organic carbon (SOC), calcium (Ca), and phosphorus (P); the correlations between these variables can be seen in Table I. Since the variables are real-valued, prediction is a regression problem. In addition, since the targets are interdependent they are considered to be structured. We hypothesize that treating them as such in the model will improve performance.

Table I: Correlation matrix of target variables in the AFSIS training set. Pearson correlation was used, and correlations are significant at the $|p| < 0.05$ level.

<table>
<thead>
<tr>
<th></th>
<th>Sand</th>
<th>pH</th>
<th>SOC</th>
<th>Ca</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sand</td>
<td>1</td>
<td>0</td>
<td>-0.5</td>
<td>-0.29</td>
<td>0</td>
</tr>
<tr>
<td>pH</td>
<td>-1</td>
<td>0</td>
<td>0.59</td>
<td>0.15</td>
<td></td>
</tr>
<tr>
<td>SOC</td>
<td>-</td>
<td>1</td>
<td>0.42</td>
<td>0.16</td>
<td>0.16</td>
</tr>
<tr>
<td>Ca</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>P</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1</td>
</tr>
</tbody>
</table>

The training data contains 1158 samples with 3578 spectral measurements, and 16 spatial predictors from remote sensing sources. The withheld test set contains the same inputs, but 728 samples. The only pre-processing we apply to the dataset is standardization to zero mean and unit variance, for both the input and target variables. Note that we do not consider the spatial predictors in our model.

B. LUCAS Dataset

The LUCAS topsoil project was developed to provide a standardized soil dataset that spans across the European
Union [2]. The dataset has 19,036 samples spread across 23 of the 28 members of the European Union (missing Romania, Cyprus, Malta, Bulgaria, and the recently added Croatia in 2013). LUCAS contains spectral measurements in the range of 400 nm to 2500 nm, with evenly spaced readings every 0.5 nm, corresponding to 4200 features. In order to maintain a consistent labeling across datasets\(^3\), we remove samples without a measurement of sand content leaving 17,921 samples. We randomly sample 10% as a test set, and use the remaining 90% for training and cross validation. Finally, we pre-process the data by standardizing to zero mean and unit standard deviation across all spectral measurements and target variables. The target variables considered are: sand content, pH, organic carbon (OC), calcium carbonate (CaCO\(_3\)), and phosphorus (P). The correlations between these variables are shown in Table II.

Table II: Correlation matrix of target variables in the LUCAS training set. Pearson correlation was used, and correlations are significant at the \(|p| < 0.05\) level.

<table>
<thead>
<tr>
<th></th>
<th>sand</th>
<th>pH</th>
<th>OC</th>
<th>CaCO(_3)</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>sand</td>
<td>1</td>
<td>-0.41</td>
<td>0</td>
<td>-0.26</td>
<td>0</td>
</tr>
<tr>
<td>pH</td>
<td>-</td>
<td>1</td>
<td>-0.39</td>
<td>0.52</td>
<td>0.05</td>
</tr>
<tr>
<td>OC</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>-0.12</td>
<td>0</td>
</tr>
<tr>
<td>CaCO(_3)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>-0.06</td>
</tr>
<tr>
<td>P</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1</td>
</tr>
</tbody>
</table>

We recognize there may be slight discrepancies between the target variables of the LUCAS and AFSIS datasets, specifically Calcium/Calcium Carbonate. As the AFSIS pH was recorded in 1:2 parts volume water, we use the same pH category from the LUCAS dataset [19]. Similar to the AFSIS dataset, we acknowledge that the LUCAS dataset also contains auxiliary input features (including, but not limited to land use/land cover, land management and climate), but again we only consider spectral measurements in our model.

VI. EXPERIMENTS

With correlations confirmed in the target variables (Tables I & II), we now evaluate various deep learning architectures, both with structured and non-structured outputs on the AFSIS and LUCAS datasets. Due to our inputs being purely spectral wavelengths, we also leverage an additional assumption that the input features have a level of spatial continuity, allowing for the application of a 1-D convolutional neural network (CNN) in our analysis. The rationale is that the convolution filters are likely to capture rich information between subsequent spectral wavelengths. The architecture of the CNN was a convolutional layer (filter size of 50, and pooling of 15x), followed by 2 fully connected layers and a linear output.

We considered several unstructured models: a simple 2-layer NN (NN-2Hid), 3-layer NN (NN-3Hid), and CNN (Conv). The structured output models we considered were: a CRBM conditioned directly on spectral measurements (CRBM-Input), a CRBM conditioned on a 2-layer feed-forward NN (CRBM-NN), and a convnet (CRBM-Conv). Hyperparameters on all models were tuned using 5-Fold cross-validation on the training set. Our CRBM model used the parameters of the 2 hidden layer NN that achieved the lowest MCRMSE in cross-validation. Finally, in evaluating our models we report the error on the test set using the MCRMSE in Equation 1.

A. Performance comparison of the various architectures

Table III reports the average MCRMSE and standard error for both AFSIS and LUCAS datasets, where predictions were made from ten independent models trained with random weight initializations.

Table III: Error (MCRMSE) for AFSIS and LUCAS test sets. Structured output models are below the horizontal line.

<table>
<thead>
<tr>
<th>Model</th>
<th>AFSIS</th>
<th>LUCAS</th>
</tr>
</thead>
<tbody>
<tr>
<td>NN-2Hid</td>
<td>0.5318 ± 0.005</td>
<td>0.4982 ± 0.002</td>
</tr>
<tr>
<td>NN-3Hid</td>
<td>0.5476 ± 0.004</td>
<td>0.5007 ± 0.003</td>
</tr>
<tr>
<td>Conv</td>
<td>0.5232 ± 0.006</td>
<td>0.4232 ± 0.006</td>
</tr>
<tr>
<td>CRBM-Input</td>
<td>0.6235 ± 0.004</td>
<td>0.6884 ± 0.009</td>
</tr>
<tr>
<td>CRBM-NN</td>
<td>0.5426 ± 0.001</td>
<td>0.5065 ± 0.002</td>
</tr>
<tr>
<td>CRBM-Conv</td>
<td><strong>0.5209 ± 0.001</strong></td>
<td><strong>0.4216 ± 0.002</strong></td>
</tr>
</tbody>
</table>

There are two key results seen from Table III. The first is that the CRBM’s performance is improved by conditioning on the output of a feed-forward neural network (CRBM-NN) as opposed to directly conditioning on the spectral measurements (CRBM-Input). The poor performance of CRBM-Input in general demonstrates the need for the flexibility provided by multiple layers of representation between the spectral measurements and the two other sets of units in the model: the outputs and the hiddens capturing their structure.

The other key result is the effectiveness of the CNN on both datasets. With the small AFSIS dataset, the reduction in free parameters attained by weight sharing in the convnet is likely aiding generalization. However, we were surprised by how much the convnet outperforms the other methods on the LUCAS dataset, and how the MCRMSE can be further improved by conditioning a CRBM on the CNN. Another advantage of the convnet, beyond reducing parameters, is the fact that it does not need to learn the same local pattern at multiple spectral locations; this efficiency in representation may be responsible for its strong performance.

B. Individual Error Comparison

In this section, we evaluate the error that each target variable contributes to the reported MCRMSE scores. We report

\(^3\) Though the LUCAS dataset has many more target variables, in our initial study we decided to maintain consistency between the two datasets by focusing on a common set of targets. Future work will examine a higher-dimensional output space.
the root mean squared error (RMSE) against a validation set for the AFSIS dataset (as the test set was withheld), while the RMSE for LUCAS is reported against the test set.

It can be seen that the phosphorous level proved the most difficult to predict across both datasets. In the AFSIS dataset (Table IV), there does not appear to be a single model that can accurately predict each of the target outputs. The LUCAS dataset (Table V) shows similar results, however, the top performing models are all based on the convnet.

We admit that although simple, CD-1 is not designed for predictive applications. This may be responsible for the inability of the CRBMs to outshine the basic (i.e. non-structured) models. In the future, we intend to pursue predictive training algorithms for CRBMs, such as Mnih et al.’s PercLoss algorithm [18].

VII. CONCLUSIONS

In this paper, we considered several different deep learning approaches for predicting soil properties from diffuse infra-red spectroscopy measurements. We have demonstrated that the 1-d convolutional neural network was particularly effective, yet models that considered interdependencies between target variables generally performed at a sub-par level. While our test results on the AFSIS dataset are not on par with the winners of the competition who used ensemble methods, they are competitive with other single-architecture approaches. We believe there is still room for improvement in the implementation and hyper-parameter tuning of the model, perhaps aided by Bayesian Optimization or other model-based global optimization methods [35]. We are excited to present the domain of soil property prediction from spectral measurements to the deep learning community, an application that warrants further exploration.

Future work will consider predicting the full set of target outputs on the LUCAS dataset, and also consider incorporation of auxiliary inputs. For example, phosphorus (which is largely found in fertilizers) can be related to the water management practices in an area, where runoff controls the flow of particles.

REFERENCES


Table IV: RMSE & standard deviation for models in AFSIS dataset

<table>
<thead>
<tr>
<th>Model</th>
<th>Ca</th>
<th>P</th>
<th>pH</th>
<th>SOC</th>
<th>Sand</th>
</tr>
</thead>
<tbody>
<tr>
<td>NN-2Hid</td>
<td>0.2842</td>
<td>± 0.097</td>
<td>0.7528</td>
<td>± 0.408</td>
<td>0.3675</td>
</tr>
<tr>
<td>NN-3Hid</td>
<td>0.2527</td>
<td>± 0.103</td>
<td>1.0373</td>
<td>± 0.451</td>
<td>0.3903</td>
</tr>
<tr>
<td>Conv</td>
<td>0.2720</td>
<td>± 0.090</td>
<td>0.8102</td>
<td>± 0.316</td>
<td>0.3771</td>
</tr>
<tr>
<td>CRBM-Input</td>
<td>0.4312</td>
<td>± 0.072</td>
<td>0.7175</td>
<td>± 0.308</td>
<td>0.4959</td>
</tr>
<tr>
<td>CRBM-NN</td>
<td>0.2033</td>
<td>± 0.008</td>
<td>0.4758</td>
<td>± 0.007</td>
<td>0.3709</td>
</tr>
<tr>
<td>CRBM-Conv</td>
<td>0.2079</td>
<td>± 0.005</td>
<td>0.7666</td>
<td>± 0.007</td>
<td>0.3578</td>
</tr>
</tbody>
</table>

Table V: RMSE & standard deviation of models on LUCAS dataset

<table>
<thead>
<tr>
<th>Model</th>
<th>Ca</th>
<th>P</th>
<th>pH</th>
<th>SOC</th>
<th>Sand</th>
</tr>
</thead>
<tbody>
<tr>
<td>NN-2Hid</td>
<td>0.2214</td>
<td>± 0.004</td>
<td>0.8745</td>
<td>± 0.001</td>
<td>0.4013</td>
</tr>
<tr>
<td>NN-3Hid</td>
<td>0.2218</td>
<td>± 0.005</td>
<td>0.8814</td>
<td>± 0.002</td>
<td>0.4025</td>
</tr>
<tr>
<td>Conv</td>
<td>0.1972</td>
<td>± 0.005</td>
<td>0.7555</td>
<td>± 0.014</td>
<td>0.3084</td>
</tr>
<tr>
<td>CRBM-Input</td>
<td>0.5137</td>
<td>± 0.007</td>
<td>0.9897</td>
<td>± 0.017</td>
<td>0.5446</td>
</tr>
<tr>
<td>CRBM-NN</td>
<td>0.2436</td>
<td>± 0.007</td>
<td>0.8757</td>
<td>± 0.003</td>
<td>0.4078</td>
</tr>
<tr>
<td>CRBM-Conv</td>
<td>0.1978</td>
<td>± 0.002</td>
<td>0.7511</td>
<td>± 0.007</td>
<td>0.3086</td>
</tr>
</tbody>
</table>


