Incorporating Domain Knowledge About XRF Spectra into Neural Networks

Matthew Dirks, David Poole
University of British Columbia, Vancouver, BC, Canada
mcdirks@cs.ubc.ca, poole@cs.ubc.ca

Abstract

This paper investigates how incorporating domain knowledge can improve the performance of machine learning algorithms. We design and evaluate an analysis-by-synthesis model that predicts the chemical composition of rocks from spectral data. Through interactions with domain experts, specific domain knowledge about the sensing instruments and general domain knowledge about spectral data is gathered. In a analysis-by-synthesis-style auto-encoder, the decoder utilizes the domain knowledge in a differentiable generative model and the encoder learns to perform model inversion of the generative model. In experimental results, we show how our model improves prediction of some element concentrations.

1 Introduction

Machine learning models use varying amounts of data and domain knowledge to provide predictions. This paper considers the effect of combining domain knowledge with data to provide predictions from spectra resulting from spectroscopy on rocks. Domain knowledge can be incorporated as a bias that attracts the excessive degrees of freedom towards reasonable regions of the parameter space and can greatly reduce model variance [1].

Specifically, the prediction task of interest is to predict the chemical composition (concentrations of 48 elements) of rock samples using spectral data from X-Ray Fluorescence (XRF) sensors in a mine. This chemical composition information is used in monitoring and improving mining operations in real-time. There is much domain knowledge from geology and physics experts about the behaviour of XRF and how X-rays interact with rocks.

Research in machine learning has gone back and forth between research that focuses on incorporating domain knowledge into learning methods and research that focuses on domain-independent learning utilizing as little human knowledge as possible [2, 3]. Towell and Shavlik describe the space of models as spanning from knowledge-intensive to knowledge-free and that "staying at one end or the other of the spectrum of possible learning systems simplifies the learning problem by allowing strong assumptions to be made about the nature of what needs to be learned. However, the middle ground is appealing; it offers the possibility that synergistic combinations of theory and data will result in powerful learning systems" [4, 5]. Our approach aims to sit in this "middle ground."

Machine learning models always have a bias, but some explicitly combine data with domain knowledge. Domain knowledge represented by propositional rules have been incorporated via the weights of a neural network [4] and via explicit rules [2, 6]. Data augmentation incorporates known invariances in the domain by generating new artificial training examples [7, 8]. Parameter sharing architectures, like Convolutional Neural Networks (CNN's) [9], also incorporate domain knowledge.

We incorporate domain knowledge into an analysis-by-synthesis-style auto-encoder, where the encoder performs "analysis" by learning how to encode the input data and the decoder performs "synthesis" by simulating how the examples were created in terms of a compact set of variables

Workshop on Perception as Generative Reasoning, NeurIPS 2019, Vancouver, Canada.

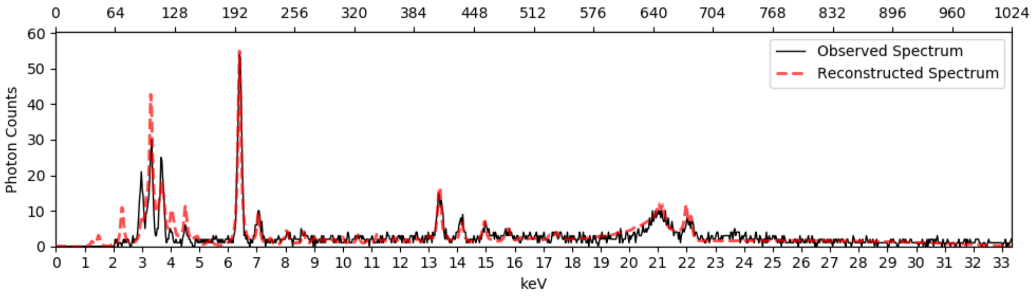

Figure 1: An example spectrum and its reconstruction through the auto-encoder with simulator.

[10]. In related work, Wu et al [11] synthesize a physical world to recover a representation from images. They use a convolutional neural network and feed the output to a physics simulator—which incorporates domain knowledge—followed by a graphics engine. Similarly, Tieleman [12] builds domain knowledge into the synthesis stage which takes as input properties of 2D graphical objects such as location and rotation and outputs an image. After training the auto-encoder, the decoder is discarded and the encoder's output is fed as input to another learning algorithm for performing some task (e.g. classification of images).

We compare standard algorithms to neural networks and to an analysis-by-synthesis-style auto-encoder. To our knowledge, analysis-by-synthesis has not been applied to spectroscopy (including XRF spectroscopy) in published literature. Our method uses a custom-built XRF simulator for synthesis and gives state-of-the-art performance on many elements.

2 Dataset

Rock Samples: We used 177 drill core rock samples (about 3 inches in length) obtained from a lithium mine, which are representative of rocks that would be seen during mining. The samples were analyzed by x-ray fluorescence (to obtain observations) and then sent for geochemical assay (to obtain ground truth).

XRF Spectra: X-Ray Fluorescence (XRF) is a technique for surface analysis that outputs a histogram of energy versus photon counts, called a spectrum (for example, Figure 1). Without any knowledge of the domain it is difficult to understand what the spectrum means, but domain experts can provide relevant knowledge about XRF and geology that give it meaning.

Geochemical Assay: For each rock sample, a geochemical assay is obtained by sending it to a commercial assay lab for destructive analysis. This method provides gold-standard estimates of 48 elements. These elements have varying ranges and detection limits—some measured in percent $(\%)$ and others in fractions of parts-per-million (ppm, 1 ppm $= 0.0001\%$). The less-abundant elements are more difficult for any sensor or algorithm to detect. Since each element varies in range, all methods in this paper normalize each element against its range.

3 Methods

LS: A standard algorithm of estimating element concentrations for a particular element from XRF spectra is to build a simple linear model from photon counts to geochemistry. For each element, the photon count of the primary peak ("KL3" peak) of the element is taken from the spectrum for every rock sample, then least squares (LS) is used to build a model from which we make predictions.

LASSO: LASSO [13] has been used in other spectroscopy methods to induce sparsity on the large feature spaces exhibited by most spectra. Least Absolute Shrinkage and Selection Operator (LASSO) regression is a least squares regression model that penalizes the sum of the absolute value of the model's coefficients (L1-norm on the coefficients). This constraint induces sparsity. LASSO is used to simultaneously select features and regress elemental composition.

FCNN: The first neural network model we evaluate is a Fully-Connected Neural Network (FCNN). We found that a single layer (i.e. no hidden layers) with RELU activations performs best. This model is very flexible, making no attempt to incorporate the domain knowledge given to us.

All neural networks are trained for 100,000 epochs or until early-stopping. Hyper-parameters (learning rate, number of layers and units, dropout probability, activation functions, and L1 regularizer scale) were tuned using a combination of trial-and-error and grid search over the parameter space and were evaluated on a subset of the data. Training and evaluation is performed using 10-fold cross-validation.

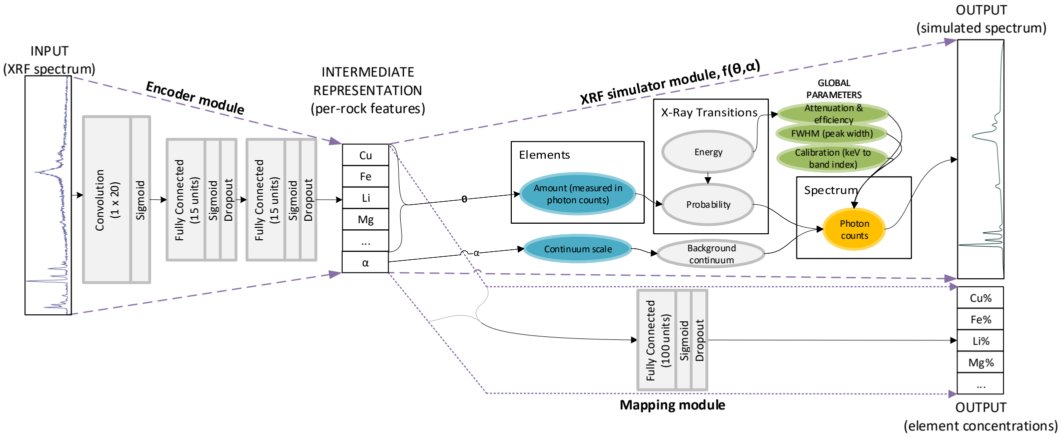

Figure 2: Our model, Analysis-by-XRF-Synthesis (AXS), for XRF spectra.

CNN: Our second neural network is a Convolutional Neural Network (CNN) [9] that incorporates very general domain knowledge about spectral data. This model has the same architecture as the encoder in the left half of Figure 2. A convolution layer, whose input is a spectrum, captures the notion that features are likely to be locally correlated, and is due to the domain knowledge that fluorescing elements create peaks whereby the photon count at the center of a peak is correlated to the photon count at neighbouring features. No pooling is used. Fully-connected layers learn the non-linearities and any dependencies between features that may be present in the data. Early-stopping, dropout, and L1 regularization are used to reduce over-fitting.

AXS: Our final model, Analysis-by-XRF-Synthesis (AXS) tries to incorporate as much of the domain knowledge as possible. The domain knowledge provided by our experts is listed in Appendix A.1 and includes general facts about spectra (e.g. peaks are important and they are shaped like Lorentzian curves) and XRF specifics (e.g. occurrence of Rayleigh peaks or that attenuation causes photons to be lost).

Auto-encoders are commonly used to build low-dimensional informative representations [14, 15]. We use a variation of an auto-encoder following the Analysis-by-Synthesis approach in which the decoder is a simulator [12, 11]. In our approach, synthesis is performed by an XRF simulator and the encoder learns to perform model inversion of the XRF simulator. By regularizing the network with a simulator, we hope to guide the learning towards basins of attraction of effective local minima that provide better generalization than a purely discriminative model [16]. **AXS** consists of 3 modules (Figure 2): encoder, XRF simulator, and mapping module. The encoder architecture is the same as the **CNN** model and outputs a low-dimensional representation whose semantics are enforced by the simulator that follows.

We built a simulator of XRF based on domain knowledge that simulates a spectrum given an elemental composition θ, and simulator parameter α. Details of the simulator, $f(\boldsymbol{\theta}, \alpha)$, are given in Appendix A.2 and is graphically depicted in Figure 2. f is completely differentiable which allows the model to be backpropagated end-to-end. Parameters of the simulator (global parameters not specific to individual rocks) are also learned during training.

By using a simulator built from domain knowledge, we effectively incorporate our domain knowledge into the system via the intermediate representation which, as input into the simulator, has defined meanings. Also, the intermediate representation is used as input to another learning module—the Mapping module—shown in Figure 2 which is a fully-connected layer with L1 regularization. The

Mapping module outputs the desired prediction targets (element concentrations in units of percent). The auto-encoder's reconstruction error is added to the prediction error induced by the Mapping module, yielding a loss function that aims to minimize both errors simultaneously. The reconstruction error term in the loss function aims to reduce variance while introducing a bias in the model. This is justified because "there may exist biased estimators that are better than the best unbiased one" [1].

4 Results & Discussion

We estimate an algorithm's generalization ability using k-fold cross-validation (CV), with $k = 10$. 10-fold CV reduces training time and provides a nearly unbiased estimate of prediction error on unseen data [17]. Prediction error for each target variable is estimated via the mean squared error (MSE) of the percentage estimate across the test sets of all cross-validation folds. Standard error, s, of the cross-validation prediction error is estimated as: $s = \frac{1}{\sqrt{k}} SD\{CV_1, ..., CV_k\}$ where k is 10, SD is the sample standard deviation, and CV_1 through CV_k are MSE's of the k cross-validation folds.

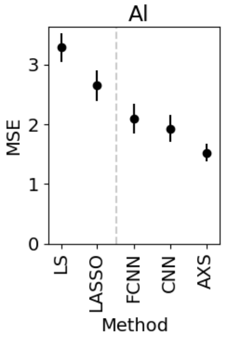

Figure 3: Example of monotonic improvement in MSE.

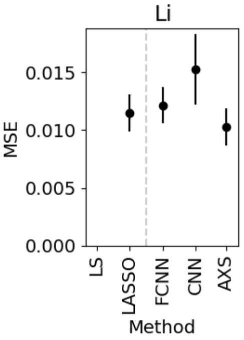

Figure 4: Example where **AXS** does well but **CNN** doesn't.

Table 1: Elements where each algorithm achieved the best MSE.

Method	Best on
LS	S
LASSO	Rb Sb
FCNN	Ca Cs Mo Sr
CNN	
AXS	Al Be Fe Ga Hf K Li Mg Pb Sn Zr

In a real-world application, some elements may be of interest and some not, so we report performance for each element individually. The best performing algorithm for each element of interest would then be deployed. For instance, for lithium (Li), **AXS** produced the lowest prediction error (Figure 4). Note that the primary peaks of some elements, including lithium, are not directly measured by the XRF sensor so **LS** is not applicable in these cases. 48 elements have ground truth, 30 of these were removed from the results because none of the methods were able to do better than predicting the average. Of the remaining 18, **AXS** achieved the best MSE on 11 elements (see Table 1). A complete table of expected prediction errors and standard errors are shown in Appendix Table 2.

Since **CNN**, and **AXS** each incorporate more domain knowledge than the model before it (where **FCNN** is the base model with the least domain knowledge), we would expect to find monotonic improvement in prediction error. 3 elements (Al, Ga, and Pb) exhibit such monotonic improvement similar to the one shown in Figure 3 (error bars are ± 1 Standard Error and graphs for all elements are shown in Appendix Figure 5). 6 elements (Ca, Cs, Mo, S, Sb, and Sr) performed oppositely—with **FCNN**'s MSE less than all other neural networks—but standard error is large, suggesting performance could have gone either way.

An example of a reconstruction from **AXS**'s auto-encoder is shown in Figure 1 (in red). Future work is to discuss with our domain experts how to fit the poor-fitting regions of the spectra better. Undoubtedly, more domain knowledge will improve the model, but we found that domain knowledge is messy and not straightforward to incorporate. Other variants of neural network architectures may also improve performance, but remain to be investigated, such as fully-convolutional networks, deconvolutional layers (in the decoder), and variational auto-encoders. The XRF simulator can generate a spectrum for any input (which approximates what would actually be observed). Future work is to generate synthetic labelled examples that may be used in model pre-training or to augment the dataset. Finally, it remains an open problem to determine whether **AXS**'s regularization power over other models is due to incorporating domain knowledge or to using a semi-supervised auto-encoding architecture.

References

[1] Johansen, T. A. On Tikhonov regularization, bias and variance in nonlinear system identification. *Automatica*, 33(3):441–446, 1997.

[2] Yu, T. *Incorporating prior domain knowledge into inductive machine learning: its implementation in contemporary capital markets*. Ph.D. thesis, University of Technology Sydney, 2007.

[3] Teng, T.-H., A.-H. Tan, J. M. Zurada. Self-organizing neural networks integrating domain knowledge and reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 26(5):889–902, 2015.

[4] Towell, G. G., J. W. Shavlik. Knowledge-based artificial neural networks. *Artificial intelligence*, 70(1-2):119–165, 1994.

[5] Mitchell, T. M. Machine learning, 1997.

[6] Yu, T., S. Simoff, T. Jan. VQSVM: A case study for incorporating prior domain knowledge into inductive machine learning. *Neurocomputing*, 73(13):2614 – 2623, 2010.

[7] Niyogi, P., F. Girosi, T. Poggio. Incorporating prior information in machine learning by creating virtual examples. *Proceedings of the IEEE*, 86(11):2196–2209, 1998.

[8] Perez, L., J. Wang. The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*, 2017.

[9] LeCun, Y., L. Bottou, Y. Bengio, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[10] Halle, M., K. Stevens. Speech recognition: A model and a program for research. *IRE transactions on information theory*, 8(2):155–159, 1962.

[11] Wu, J., E. Lu, P. Kohli, et al. Learning to see physics via visual de-animation. In *Advances in Neural Information Processing Systems*, pages 152–163. 2017.

[12] Tieleman, T. *Optimizing neural networks that generate images*. Ph.D. thesis, University of Toronto (Canada), 2014.

[13] Tibshirani, R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[14] Le, Q. V., M. Ranzato, R. Monga, et al. Building high-level features using large scale unsupervised learning. *arXiv preprint arXiv:1112.6209v5*, 2012.

[15] Bengio, Y., A. Courville, P. Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[16] Gülçehre, Ç., Y. Bengio. Knowledge matters: Importance of prior information for optimization. *The Journal of Machine Learning Research*, 17(1):226–257, 2016.

[17] Bengio, Y., Y. Grandvalet. No unbiased estimator of the variance of k-fold cross-validation. *Journal of machine learning research*, 5(Sep):1089–1105, 2004.

[18] Brouwer, P. *Theory of XRF: Getting acquainted with the principles*. PANalytical BV, 2006.

A Appendix

A.1 XRF Domain Knowledge

(K_1) Elements in the periodic table produce a set of 'peaks' centered at energies characteristic of the element.[1]

(K_2) For each element, the set of peaks occur with known ratios, given by the probability of electrons fluorescing for each transition.[2]

(K_3) Peaks are Lorentzian shaped.

(K_4) Spectra from each element and elements' transition peaks are summed in the final spectrum.

(K_5) Noise in the photon counts is due to limited exposure time. If exposure time were increased, noise would diminish by the law of large numbers. Noise will follow a Gaussian distribution because of the central limit theorem.

(K_6) Matrix effects may cause the amount of one element to increase the photon count of other elements.[3]

(K_7) 20–21 keV contains broad peaks, called Compton peaks or inelastic scattering, that depend on several factors including the energy of X-ray source, the atomic numbers of the elements in the composition, and the angle of the sensor relative to the sample [18]. Compton disappears entirely for heavy elements like lead.

(K_8) 21–24 keV contains peaks, called Rayleigh peaks or elastic scattering, that depend on several factors including the energy of the X-ray source and the atomic numbers of the elements in the composition [18].

(K_9) Peaks unrelated to composition may occur between 2–5 keV, which are due to thermal effects.

(K_{10}) Attenuation and detection efficiency will cause lower and higher keV bands to lose photon counts.

(K_{11}) Bragg scattering may occur.

(K_{12}) Background continuum is present in the spectrum which is caused by "Bremmsstrahlrung radiation" (or, backscatter in the spectrum). It is the result of outer electron shell transitions.

A.2 XRF Simulator

We built a simulator of XRF, f, that simulates a spectrum given an elemental composition based on some of the domain knowledge provided. A spectrum is a function from energy into photon count.

$$\boldsymbol{\theta} = (\theta_{\text{Cu}}, \theta_{\text{Fe}}, \text{etc}) \tag{1}$$

$$\text{spectrum} \colon \text{energy} \mapsto \text{photon count} \tag{2}$$

$$f \colon \boldsymbol{\theta} \mapsto \text{spectrum} \tag{3}$$

We approximate the background continuum (K_{12}) as a Bézier curve, b, and scale it with a single parameter, α, that depends on the rock sample:

$$b \colon p_1, p_2, \alpha \mapsto \text{spectrum} \tag{4}$$

$$b(p_1, p_2, \alpha)(i) = \alpha\big(3p_1 i(1-i)^2 + 3p_2 i^2(1-i)\big) \tag{5}$$

For attenuation and efficiency (K_{10}), we approximate them together as a sum of two sigmoids, S:

$$S \colon a_1, c_1, a_2, c_2 \mapsto \text{spectrum} \tag{6}$$

$$S(a_1, c_1, a_2, c_2)(i) = 1 + e^{a_2(c_2 - i)} + e^{a_1(c_1 - i)} \tag{7}$$

[1] These peak locations occur at specific transition energies. KL3 ($K\alpha$) is the strongest one, but there can be up to 45 others. Some energies will be out of range of our sensor, and some peaks will be too weak to see.

[2] X-ray transition energies downloaded from NIST (https://www.nist.gov/ pml/X-ray-transition-energies-database) and transition probabilities downloaded from NDS (https://www-nds.iaea.org/epdl97/).

[3] The occurrence of matrix effects depends on the minerals and physical locations of minerals. Also, elements lower on the periodic table are affected by ones above, but not the other way around.

We refer to the set of transitions for element e as T_e and the energy and probability for each transition, t, as t_{energy} and $t_{probability}$ respectively (see K_1 and K_2). The simulator, f, takes a distribution over elements, θ, and returns a spectrum over energies i:

$$g(\boldsymbol{\theta})(i) = \sum_{e \in E} \left(\theta_e \sum_{t \in T_e} Lor(t_{energy}, t_{probability}, \lambda)(i) \right) \tag{8}$$

$$f(\boldsymbol{\theta}, \alpha)(i) = S(a_1, c_2, a_2, c_2)(i) \times g(\boldsymbol{\theta})(i) + b(p_1, p_2, \alpha)(i) \tag{9}$$

where i is an energy (in units of keV), θ_e is the amount of element e, α scales the continuum, and $Lor(t_{energy}, t_{probability}, \lambda)$ is a spectrum of a Lorentzian peak (a Lorentzian has a similar shape to a Gaussian but is more narrow around the peak with longer tails):

$$Lor\colon t_{energy}, t_{probability}, \lambda \mapsto \text{spectrum} \tag{10}$$

$$Lor(t_{energy}, t_{probability}, \lambda)(i) = \frac{t_{probability} \left(\frac{\lambda}{2}\right)^2}{(i - t_{energy})^2 + \left(\frac{\lambda}{2}\right)^2} \tag{11}$$

where t_{energy} is the location of the center of the Lorentzian peak, $t_{probability}$ is the height of the Lorentzian peak, and λ is the width of the Lorentzian peak.

A.3 Full Evaluation Results

	Baselines		Neural Networks			
Element	LS	LASSO	FCNN	CNN	AXS*	AXS
Al	3.29e+0±2.4e-1	2.65e+0±2.6e-1	2.09e+0±2.5e-1	1.93e+0±2.3e-1	1.56e+0±1.7e-1	1.52e+0±1.5e-1
Be		1.25e-7±2.2e-8	1.36e-7±2.6e-8	1.33e-7±2.9e-8	1.15e-7±1.6e-8	1.19e-7±1.6e-8
Ca	8.96e+0±2.5e+0	1.11e+1±2.6e+0	6.71e+0±2.4e+0	1.44e+1±5.1e+0	1.03e+1±3.2e+0	1.01e+1±3.1e+0
Cs	1.93e-4±4.5e-5	1.42e-4±3.0e-5	1.24e-4±3.1e-5	1.43e-4±3.3e-5	1.50e-4±3.3e-5	1.34e-4±2.7e-5
Fe	7.85e-1±1.6e-1	1.01e+0±1.3e-1	7.67e-1±1.7e-1	8.33e-1±1.6e-1	7.01e-1±1.5e-1	7.21e-1±1.4e-1
Ga	3.19e-7±2.2e-8	2.31e-7±1.8e-8	2.45e-7±2.6e-8	1.95e-7±2.0e-8	1.80e-7±2.3e-8	1.72e-7±2.0e-8
Hf		1.56e-7±2.1e-8	1.52e-7±1.8e-8	2.12e-7±4.6e-8	1.58e-7±1.6e-8	1.46e-7±1.6e-8
K	1.10e+0±1.5e-1	1.28e+0±1.2e-1	1.13e+0±9.4e-2	1.29e+0±2.8e-1	1.00e+0±2.1e-1	8.70e-1±1.5e-1
Li		1.15e-2±1.6e-3	1.22e-2±1.6e-3	1.53e-2±3.0e-3	1.02e-2±1.5e-3	1.03e-2±1.6e-3
Mg	1.54e+1±1.2e+0	7.23e+0±5.7e-1	5.73e+0±8.0e-1	6.85e+0±1.9e+0	4.58e+0±9.2e-1	4.42e+0±5.4e-1
Mo	2.74e-5±5.7e-6	2.55e-5±6.1e-6	2.41e-5±6.0e-6	6.08e-5±1.5e-5	5.54e-5±1.5e-5	5.92e-5±1.4e-5
Pb		3.47e-7±6.4e-8	3.22e-7±5.3e-8	2.89e-7±4.6e-8	2.82e-7±5.1e-8	2.63e-7±4.1e-8
Rb	1.53e-4±1.9e-5	1.17e-4±1.6e-5	1.32e-4±1.8e-5	2.22e-4±4.2e-5	1.47e-4±1.8e-5	1.18e-4±1.5e-5
S	8.46e-1±1.8e-1	9.27e-1±1.4e-1	9.30e-1±2.0e-1	1.27e+0±2.0e-1	1.31e+0±2.4e-1	1.20e+0±1.9e-1
Sb	1.13e-6±2.5e-7	9.89e-7±2.3e-7	1.02e-6±2.7e-7	1.02e-6±2.5e-7	1.10e-6±2.4e-7	1.09e-6±2.7e-7
Sn	1.51e-8±1.5e-9	1.50e-8±1.6e-9	1.20e-8±1.4e-9	1.26e-8±2.0e-9	1.04e-8±1.3e-9	9.41e-9±1.1e-9
Sr	4.79e-4±2.1e-4	5.45e-4±1.5e-4	3.84e-4±1.2e-4	5.93e-4±2.6e-4	5.90e-4±2.6e-4	5.81e-4±2.4e-4
Zr	9.42e-4±4.2e-4	1.35e-3±4.8e-4	9.14e-4±2.5e-4	1.81e-3±6.1e-4	8.96e-4±2.1e-4	8.64e-4±2.2e-4

Table 2: MSE (mean squared error) and standard error reported across cross-validation test sets (except PXRF which reports training set MSE). Best scores for each element across all algorithms is highlighted. **AXS*** is a slight variation of **AXS** that is not explained in this paper. 48 elements have ground truth, 30 of these were removed from the results because none of the methods were able to do better than predicting the average, likely due to low-abundance of these elements.

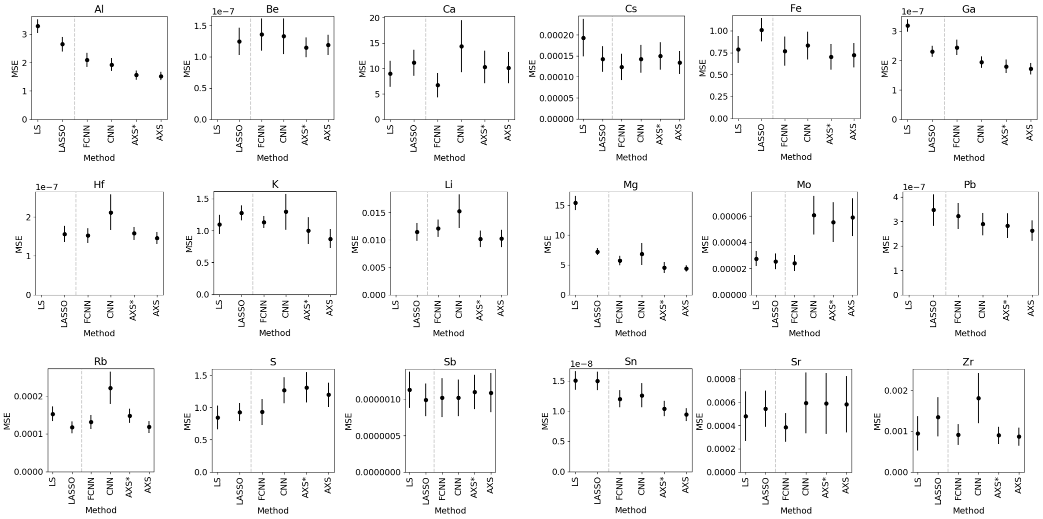

Figure 5: MSE and standard error (error bars) for each algorithm and each element. **AXS*** is a slight variation of **AXS** that is not explained in this paper. 48 elements have ground truth, 30 of these were removed from the results because none of the methods were able to do better than predicting the average, likely due to low-abundance of these elements.