Attention-based Learning for Missing Data Imputation in HoloClean

Richard Wu [1] **Aoqian Zhang** [1] **Ihab F. Ilyas** [1] **Theodoros Rekatsinas** [2]

Abstract

We study the problem of missing data imputation, a data validation task that machine learning researchers and practitioners confront regularly. We focus on mixed (discrete and continuous) data and AimNet, an attention-based learning network for missing data imputation in HoloClean, a state-of-the-art ML-based data cleaning framework. AimNet utilizes a variation of the dot product attention mechanism to learn structural properties of the mixed data distribution and relies on the learned structure to perform imputation. We perform an extensive experimental study over 14 real-world data sets to understand the role of attention and structure on data imputation. We find that the simple attention-based architecture of AimNet outperforms state-of-the-art baselines, such as ensemble tree models and deep learning architectures (e.g., generative adversarial networks), by up to 43% in accuracy on discrete values and up to 26.7% in normalized-RMS error on continuous values. A key finding of our study is that, by learning the structure of the underlying distribution, the attention mechanism can generalize better on systematically-missing data where imputation requires reasoning about functional relationships between attributes.

1 Introduction

Missing data is a common data preparation issues machine learning researchers and practitioners have to confront. In many applications and fields, including the sciences (Council et al., 2010) and data mining (Jaseena & David, 2014), missing data is present to various degrees. Missing data can reduce the statistical power of an analysis and can produce biased estimates, leading to invalid conclusions. As a result, it is imperative to handle missing data appropriately.

Many approaches have been proposed for handling missing data, with most of them focusing on either discrete or continuous data, and few focusing on mixed data (i.e., both discrete and continuous) (Soley-Bori, 2013). In some cases, simple methods may suffice: when few samples have missing values and these values are missing completely at random (MCAR) (Little & Rubin, 2019), omitting these samples may not increase the sampling error and may not introduce bias to the downstream analysis. However, these strict MCAR assumptions may not hold in most real-world data sets, hence most approaches for handling missing data focus on *data imputation*. Missing data imputation methods range from simple statistical approaches that treat each data coordinate independently of others, such as replacing missing values with the sample mode (for discrete data) or the sample mean/median (for continuous data), to more complex methods that learn a model over the input data and performs imputation by inference. Unfortunately, the performance of model-based data imputation methods can vary depending on the missing data mechanism, and they can yield poor accuracy when the missing data mechanism is systematic (Saunders et al., 2006; Kang, 2013).

We revisit deep learning architectures for data imputation and seek to answer the question: *what is a simple architecture that is interpretable and can obtain accurate results, even in the presence of systematically-missing data?* We argue that *Attention* (Bahdanau et al., 2015) should be a central component in deep learning architectures for data imputation. The attention mechanism is increasingly popular for learning structural properties of the underlying data distribution. Given a set of inputs and a query value, the attention mechanism weights the relevance of every input to the query and uses these weights to generate a query-specific representation of the inputs.

From learning autoregressive models that achieve state-of-the-art results in natural language processing (Devlin et al., 2018; Yang et al., 2019) to facilitating the discovery of causal interactions in observational time series data (Nauta et al., 2019), the attention mechanism has been shown to share commonalities with traditional approaches, such as Kernel methods, which can successfully model structural properties of complex data distributions (Tsai et al., 2019). In addition, attention mechanisms are known to be

[1]University of Waterloo [2]University of Wisconsin-Madison. Correspondence to: Theodoros Rekatsinas <thodrek@cs.wisc.edu>.

Proceedings of the 3^{rd} MLSys Conference, Austin, TX, USA, 2020. Copyright 2020 by the author(s).

computationally efficient (Vaswani et al., 2017) and interpretable (Gilpin et al., 2018). Motivated by the aforementioned benefits of Attention, we study its role in missing data imputation.

Most popular imputation methods rely on discriminative models such as decision trees (Stekhoven & Bühlmann, 2012; D'Ambrosio et al., 2012), regression models (Sentas & Angelis, 2006; Buuren & Groothuis-Oudshoorn, 2010), and neighborhood-based methods (Troyanskaya et al., 2001), which are relatively interpretable. Unfortunately, systematically-missing data may yield biased sample statistics, which causes these methods to generalize poorly as they are interpolation-based methods (Belkin et al., 2018). To address this limitation, different imputation methods turn to generative models to improve imputation accuracy; the structural assumptions encoded in the models can lead to better generalization. Generative methods include algorithms that rely on mixture-based models (e.g., mixture of Gaussians or Bernoullis) (García-Laencina et al., 2010), or deep learning models such as denoising autoencoders (Gondara & Wang, 2018) and generative adversarial nets (GANs) (Yoon et al., 2018). However, current generative methods for imputation have various drawbacks. These methods either make strict structural assumptions and fail to generalize well to mixed discrete and continuous variables, or suffer from non-convergence and mode collapse (Salimans et al., 2016). Moreover, existing deep learning based imputation methods are complex and hard to interpret.

We propose AimNet, a simple attention-based model for missing data imputation over mixed data (i.e., discrete and continuous data). AimNet is built around a new variation of the dot product attention mechanism that, given a tabular data set as input, learns (schema-level) structural properties of the data distribution. To learn flexible representations over mixed data types (discrete and continuous), AimNet ingests raw tabular data without any feature pre-processing (besides mapping discrete values to trainable embeddings and z-score normalization on continuous values) and is trained using self-supervised gradient descent-based end-to-end learning. To model mixed data distributions, AimNet's loss corresponds to a multi-task loss over regression and classification problems (targeting continuous and discrete data coordinates, respectively). AimNet is part of the HoloClean framework [1] (Rekatsinas et al., 2017), a state-of-the-art ML-based data cleaning system.

We compare AimNet against a diverse array of discriminative and generative imputation models and show that AimNet outperforms or is competitive to state-of-the-art data imputation methods under various missing data mechanisms (both random and systematic). For evaluation, we consider 14 real-world data sets with both naturally-occurring and injected missing values. For completely at random missing errors (a standard approach for evaluating imputation models), AimNet beats the next-best baselines by up to 3% in accuracy on discrete attributes and 26.7% in normalized root mean squared (NRMS) error on continuous attributes. More interestingly, AimNet exceeds the next-best baseline on systematic errors by up to 43% in accuracy on discrete attributes and 7.4% in NRMS error on continuous attributes. We find that naturally-occurring errors in a given data set follow a systematically-missing data mechanism and empirically show that AimNet is better than existing methods at handling missing data in the presence of functional dependencies between attributes.

We argue that the key module that leads to AimNet's better performance is the attention mechanism and perform a thorough experimental study on systematically-missing data, where the missing values are functionally related to the values of other attributes (i.e., missing conditionally at random or MAR). Our results show that by learning the structure of the underlying data distribution, AimNet's attention mechanism enables accurate predictions over complex mixed data domains, where imputing missing data requires extrapolating over functional relationships between attributes. In addition to producing better imputation results, AimNet achieves run times 54% (or more) lower than the run times of other baselines on data sets with large discrete domains.

2 Related Work

Missing data imputation (MDI) methods can be primarily categorized into two groups:

Discriminative Models There are several discriminative methods that rely on matrix factorization (Keshavan et al., 2010), matrix completion (Cai et al., 2010; Mazumder et al., 2010), structured prediction models (Rekatsinas et al., 2017), and vanilla autoencoders (Gondara & Wang, 2018). A popular MDI method in Bioinformatics is Multiple Imputation by Chained Equations (MICE) (Buuren & Groothuis-Oudshoorn, 2010) which solves a series of regression problems. Tree-based MDI models include MissForest (Stekhoven & Bühlmann, 2012) which uses random forests to learn an imputation model for each data coordinate and XGBoost (Chen & Guestrin, 2016), which has been shown to also be successful in dealing with missing data. Tree-based methods can learn non-linear relationships while having great in-sample and interpolation predictive power. Unfortunately, tree-based methods are known to suffer in generalization with systematically-missing data.

Generative Models A different line of work proposes the use of generative models for MDI. Early methods include Expectation Maximization algorithms (García-Laencina et al., 2010) while recent methods employ deep learning such as in Generative Adversarial Imputation Nets (GAIN)

[1]HoloClean source: `http://www.holoclean.io`

(Yoon et al., 2018), which is based on conditional GANs (Goodfellow et al., 2014). Other models such as HI-VAE (Nazábal et al., 2018) and MIWAE (Mattei & Frellsen, 2019) extend Variational Autoencoders (VAEs) (Kingma & Welling, 2014) to support mixed data distributions. While generative models yield state-of-the-art results under MCAR settings, they are difficult to train. Specifically, GANs suffer from non-convergence and mode collapse problems as a result of their loss formulation (Salimans et al., 2016). Furthermore, generative models tend to involve latent variables used in the sampling and imputation process that often do not correspond to concrete signals or structures of the data, making it difficult for a practitioner to interpret the imputation process for further understanding.

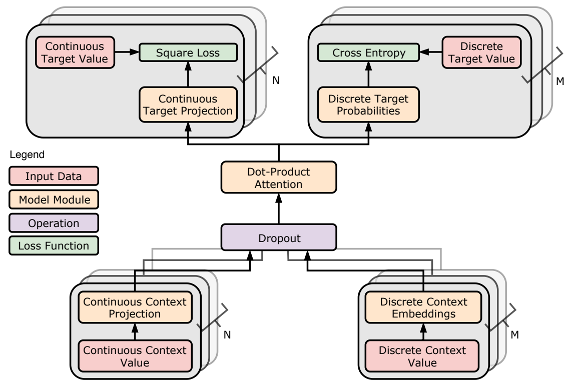

Figure 1: Architecture overview of AimNet

3 Preliminaries

We now introduce the problem of MDI and necessary background information.

Problem Consider a data set $\mathcal{D}$ with schema $\mathcal{R}$. Each j-th attribute $A_j \in \mathcal{R}$ can be either continuous or discrete. Let $N(\mathcal{R})$ and $C(\mathcal{R})$ denote the set of continuous and discrete attributes in $\mathcal{R}$. If $A_j \in N(\mathcal{R})$ then its domain is $\text{dom}(A_j) = \mathbb{R}^{d_j}$ where d_j is the dimension of A_j, and if $A_j \in C(\mathcal{R})$ then the index set over $\text{dom}(A_j)$ is $\mathcal{I} = \{1, \dots, |A_j|\}$ where $|A_j|$ is the cardinality of A_j. A missing cell value in the i-th tuple $t_i \in \mathcal{D}$ on the j-th attribute $A_j \in \mathcal{R}$ is denoted as $t_i[A_j] = \varnothing$. Let $\tilde{\mathcal{D}}$ be an imputed version of $\mathcal{D}$ where $\forall i, j$ with $t_i[A_j] \in \mathcal{D}, t_i[A_j] = \varnothing$ but for $\tilde{t}_i \in \tilde{\mathcal{D}}$ we have $\tilde{t}_i[A_j] \neq \varnothing$. We also denote $\mathcal{D}^*$ the latent ground truth sampled data set without missing data.

The goal of *missing data imputation* is: given a data set $\mathcal{D}$ with missing values obtain an imputed data set $\tilde{\mathcal{D}}$ such that for every tuple $\tilde{t}_i \in \tilde{\mathcal{D}}$ all cell values are the same as the corresponding tuple $t^*_i \in \mathcal{D}^*$. For a given data set $\mathcal{D}$ there may be missing cells in a subset of attributes $\mathcal{A} \subseteq \mathcal{R}$. When we focus on the output of a model for an attribute $A \in \mathcal{A}$, we refer to A as the *target attribute* and we refer to all attributes $A' \in \mathcal{R} \setminus \{A\}$ as the *context attributes*. The target and context terms are also used to describe cells at the tuple-level (e.g., target cell and context cells).

Error types There are three types of missing errors in tabular data (Rubin, 1976):

MCAR: Data is *missing completely at random*, regardless of the observed data in the data set. Specifically the probability that a cell $t[A_j]$ is missing in a given tuple t is conditionally independent of t and $\mathcal{D}$:

$$\Pr\left(t[A_j] = \varnothing \mid t[A_i] = v_i, \forall i; \mathcal{D}\right) = p_j \tag{1}$$

for some value p_j independent of $t[\cdot]$.

MAR: Data is *missing at random* conditioned only on the observed (non-missing) data. If we assume each tuple t is identically and independently distributed (i.i.d) according to the data generating distribution i.e., $t \overset{i.i.d}{\sim} \mathcal{D}$, then the probability that cell $t[A_j]$ is missing only depends on other observed cells in t:

$$\begin{aligned} &\Pr\left(t[A_j] = \varnothing \mid t[A_i] = v_i, \forall i; \mathcal{D}\right) \\ &= \Pr\left(t[A_j] = \varnothing \mid t[A_i], \forall i \neq j\right) \end{aligned} \tag{2}$$

MNAR: Data is *not missing at random* but is random conditioned on unobserved or latent variables. A cell value may also be missing conditioned on its own (missing ground truth) value. Any mechanism that is neither MCAR nor MAR is MNAR.

4 Architecture

At a high-level, AimNet adopts the architecture of an autoencoder model designed to handle mixed data types (discrete and continuous). An overview of AimNet's architecture is shown in Figure 1. To handle mixed data types, AimNet learns a combination of projections for continuous data and contextual embeddings for discrete data. Subsequently, AimNet relies on a new variation of the dot product attention mechanism to learn structural dependencies between different coordinates of the input data and uses the attention weights to combine the representations of different coordinate values into a unified context representation for a target attribute. Finally, the combined context passes through a non-linear transformation to perform imputation. A mixed loss function is used during training to handle mixed data types. We next describe AimNet's components in turn.

4.1 Context Embeddings

Given a data set tuple, AimNet transforms each attribute value into a vector embedding with dimension k. For continuous attributes we learn a projection to a k-dimensional vector while for discrete attributes we learn a contextual k-dimensional embedding. We describe each next.

Representation of Continuous Values For a continuous value $\vec{x} \in \mathbb{R}^{n_i}$ in attribute A_i where $n_i \leq k$ (n_i is the dimension of values in attribute A_i) we perform a *continuous context projection* up to dimension k as shown in Figure 1. We first standardize each dimension to zero mean and unit variance (z-score):

$$\underline{\vec{x}}_j = (\vec{x}_j - \vec{\mu_i}_j)/\vec{\sigma_i}_j \quad \forall j = 1, \ldots, n_i \tag{3}$$

where $\vec{\mu_i}_j$ and $\vec{\sigma_i}_j$ are the sample mean and sample variance of the j-th dimension of values in attribute A_i. We then apply a linear layer followed by a non-linear ReLU layer to obtain a non-linear transformation of the input:

$$\vec{z} = \mathbf{B}\sigma(\mathbf{A}\underline{\vec{x}} + \vec{c}) + \vec{d} \tag{4}$$

where $\mathbf{A}, \mathbf{B}, \vec{c}, \vec{d}$ are learned parameters and σ is a ReLU.

Representation of Discrete Values For each discrete attribute and each value in its domain, we associate a learnable vector of dimension k. Values that might share the same raw representation in different attributes are associated with separate vectors. We learn a lookup table of *discrete context embeddings* of the input values as shown in Figure 1.

We add a dropout layer before feeding the learned representations into the subsequent attention layer. The use of the dropout layer in effect transforms AimNet into a hybrid denoising autoencoder (Vincent et al., 2008). Given the dropout percentage specified as a hyperparameter to AimNet, each coordinate value may be stochastically dropped (zeroed out) with probability equal to the dropout percentage. To this end, dropout serves not only as a regularization tool but also as a data augmentation technique, introducing an exponential number of additional training examples with various missing patterns.

4.2 AimNet's Attention Model

AimNet uses an attention mechanism to combine the representations of different attributes into a unified context representation used for imputation. The key distinction between AimNet's attention model and prior attention models is that given a target attribute, AimNet learns how to attend to different context attributes and not to the values themselves. An analogy to the Transformer architecture (Vaswani et al., 2017) would be learning an attention model over positional encodings alone. By using attention weights that are dependent only on the attributes and not the values themselves, AimNet can learn structural dependencies between a target attribute and the remaining context attributes.

We decompose AimNet's attention layer into its fundamental components in Figure 2(a). Let N denote the total number of continuous attributes and M denoting the total number of discrete attributes. First, each of the $N + M$ attributes in the input data set is associated with a learnable encoding of dimension $N + M$. These encodings are used to form the query Q of the attention mechanism. The position of each target attribute A_j in Q are specified as a bitmask $K^{(j)} \in \{0, 1\}^{N+M}$ which acts as the key to the attention mechanism. To learn from the context only, the value of the encoding in Q selected by $K^{(j)}$ corresponding to the target attribute A_j is masked out via a leave-one-out bitmask $m \in \{0, 1\}^{N+M}$. Second, for a given sample (a single tuple in the input data, and where a mini-batch would consist of multiple tuples), let matrix V be the concatenation of all attribute-value representation embeddings described in Section 4.1. Matrix V contains $N + M$ row vectors, each corresponding to an embedding of dimension k. The vectors in V are normalized by their L2-norm, which improves learning for discrete targets by fixing the contribution in magnitude of the constituent vectors. The attention layer for a target attribute A_j is expressed as:

$$\text{Att}(Q, K^{(j)}, V) = (m \odot \text{softmax}((K^{(j)})^T Q))\text{norm}(V)$$

where the resulting output is a *context vector* of dimension k. This context vector contains the necessary information to perform imputation on the target attribute.

4.3 Mixed Target Prediction

We use a dual loss function to support mixed distributions. We also perform multi-task learning to jointly learn all target attributes whereby all model parameters are shared.

4.3.1 Continuous Target Attributes

For continuous target attributes, we perform a projection of the context vector down to dimension d_j of attribute A_j as shown in Figure 2(b). We first feed the context vector through a fully-connected ReLU layer of dimension $k \times k$. We then apply a final linear transformation from dimension k to the attribute's dimension d_j to obtain the resulting prediction value for the given cell. The (mean) squared loss is used as the loss between the predicted continuous value and the z-scored actual continuous value.

4.3.2 Discrete Target Attributes

To predict the imputation value for a target discrete cell, we learn similar vector embeddings as the contextual embeddings described in Section 4.1 but for the unique, possible target values. As shown in Figure 2(c), we compute the inner product between the context vector from the attention layer and the discrete target's vector embeddings for the given cell's domain values. A softmax is subsequently applied over the inner products to produce prediction probabilities for each domain value.

The possible values of a given discrete cell (its domain) is by default all values in its attribute. For data sets with

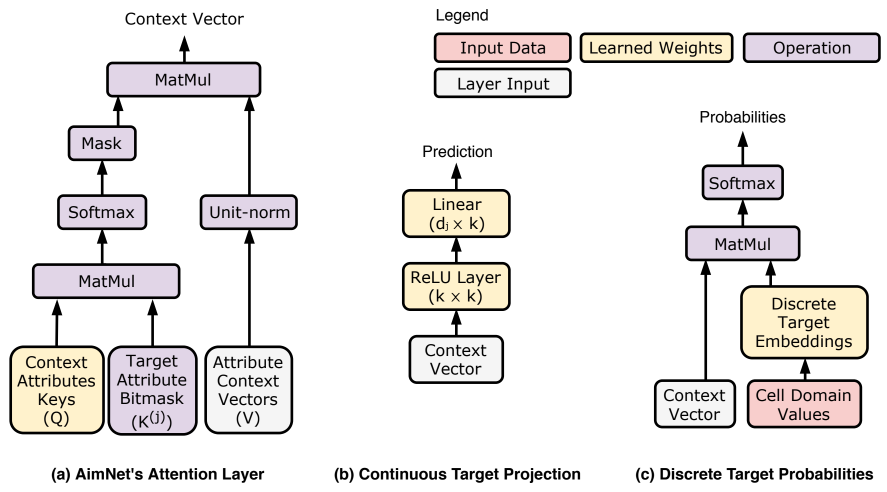
Context Vector
MatMul
Mask
Softmax
Unit-norm
MatMul
Context Attributes Keys (Q)
Target Attribute Bitmask ($K^{(j)}$)
Attribute Context Vectors (V)
Legend
Input Data
Learned Weights
Operation
Layer Input
Prediction
Linear ($d_j \times k$)
ReLU Layer ($k \times k$)
Context Vector
Probabilities
Softmax
MatMul
Discrete Target Embeddings
Context Vector
Cell Domain Values
(a) AimNet's Attention Layer
(b) Continuous Target Projection
(c) Discrete Target Probabilities

Figure 2: (a) Attention layer used in AimNet that computes attention weights based on the context-target attributes; (b) module that projects the context vector down to a continuous value for a continuous target; and (c) module that computes probabilities for the domain values of a discrete target cell.

very large discrete domains (e.g., key-like columns) and to improve performance, we use co-occurrence statistics to prune the domain to a specified maximum domain size D (Rekatsinas et al., 2017). The top D co-occurring values with the other values in the cell's tuple are pre-initialized as the cell's domain. The Categorical Cross Entropy (CCE) loss is then computed between the predicted probabilities and the actual target value as shown in Figure 1.

One subtlety is that the model separates vector embeddings for target and context values even if they are equivalent. Since context value embeddings are shared between all target attributes, their weights are constantly updated when training on every target attribute. By separating embeddings of the target values from the context value embeddings we converge quicker in practice.

5 Experimental Evaluation

5.1 Evaluation Protocol and Summary

To assess if AimNet is competitive in missing data imputation, we benchmark against competing state-of-the-art baselines on 14 data sets. We first outline the experimental setup in Section 5.2 with details of the baselines, data sets and training hyperparameters we employ. The de facto standard for evaluating data imputation is to assume missing values are distributed completely at random (MCAR). In Section 5.3, we benchmark AimNet against the other baselines on MCAR-injected missing values. We observe that AimNet outperforms or is competitive with all baselines. We then evaluate the imputation models on a data set with naturally-occurring missing values for which we have obtained the correct ground truth (see Section 5.4). We find that AimNet outperforms the next-best baseline by a large gap of $+43\%$ in accuracy. We perform a deep-dive analysis and observe that the missing values are systematically missing. We hypothesize it is due to AimNet's ability to learn functional relations between attributes of the input data set and thus counteract biases introduced due to systematically-missing data. We subsequently validate this hypothesis with both synthetic and real-world data sets in Section 5.5. We observe similar performance dominance from AimNet over competing baselines with gains of up to 11% in accuracy on discrete attributes on the real-world data sets. Our findings advocate that by learning the structure of the underlying data distribution, AimNet's attention mechanism can generalize better on systematically-missing data. We further validate that via an ablation study on AimNet's architecture. Finally, we perform micro-benchmark experiments to evaluate the sensitivity of AimNet's performance on different architectural and hyperparameter choices.

5.2 Experimental Setup

Baselines We compare our results against several state-of-the-art imputation methods (see Section 2). For discriminative models, we use XGBoost (XGB) (Chen & Guestrin, 2016), MIDAS (Gondara & Wang, 2018), MissForest (MF) (Stekhoven & Bühlmann, 2012), HoloClean (HC) (Rekatsinas et al., 2017), and MICE (Buuren & Groothuis-Oudshoorn, 2010) as representative approaches. For generative models, we focus on GAIN (Yoon et al., 2018) as it corresponds to the state-of-the-art model. For all five methods we use their open-source implementations.

Since the current open-source version of HoloClean supports only discrete data, we introduce an extension by performing quantization on continuous attributes to discretize their values and permit co-occurrence statistics to be computed across mixed attributes. We call this extended version HoloClean with Quantization (HCQ). HCQ uses k-means clustering on each continuous attribute based on a hyperparameter for the number of clusters. If a data set contains only discrete attributes then HCQ and HC are equivalent, so we report results on HCQ in place of HC for all data sets.

Data Sets We use 14 data sets, where 10 come from the UCI repository and 4 are from public and private institutions. Data sets sizes range from ~ 470 to ~ 0.5M tuples and at least a baseline accuracy of 50% (amongst discrete attributes) under a 20% MCAR injection scheme with XGBoost. This ensures repairs are not unreasonable and can be feasibly accomplished with just the observed data. The data sets containing varying proportions of discrete to continuous attributes. A detailed description of the data sets is given in Appendix A.1.

Hyperparameter Tuning and Cross-Validation We perform grid search cross-validation with a training-validation-test set split of 80-10-10 (under a 20% injection scheme such that the missing set is split evenly amongst the validation and test sets). Details on the hyperparameters tuned for each model are deferred to Appendix A.2.

Training For all experiments we employ a default set of hyperparameters for all runs with AimNet. The Adam (Kingma & Ba, 2015) optimizer is used with a default learning rate of 0.05 in conjunction with cosine annealing with warm restarts every epoch (Loshchilov & Hutter, 2017). A detailed description can be found in Appendix A.3.

Evaluation Metrics We use the following two metrics:

Accuracy: For discrete attributes, the accuracy is reported as the proportion of missing cells there were correctly imputed. The accuracy for *a given attribute* is computed and normalized by the number of missing cells in the attribute, whereas the accuracy *for a data set* is normalized by the number of missing cells across all attributes.

NRMS: For continuous attributes, the normalized Root Mean Square error by the variance of each attribute. For a given attribute A_j with n_j missing cells, ground truth values $\vec{y} \in R^{n_j}$, and predictions $\hat{\vec{y}} \in R^{n_j}$, the NRMS for A_j is $NRMS_j = (\sum_i^{n_j}((y_i - \hat{y}_i)^2)/(n_j \cdot \sigma(\vec{y})^2))^{-1/2}$ where $\sigma(\vec{y})^2$ is the sample variance. We compute the NRMS for an entire data set as $(\sum_j((NRMS_j)^2 \cdot n_j)/\sum_j n_j))^{-1/2}$. We choose to normalize RMS since continuous attributes can have different ranges.

5.3 Missing Completely at Random Data

For each data set, except Chicago which has natural errors (see Appendix A.1), we inject missing values into each attribute completely at random with $p_j = p = 0.2$. The accuracy and NRMS are reported in Table 1. As shown, AimNet outperforms all competing methods, on both discrete and continuous attributes, on almost all the data sets. AimNet does perform worse than the next-best baseline on the Eye EEG and the NYPD data sets for both discrete and continuous attributes, and CASP on continuous attributes. We draw our attention to the fact that Eye EEG consists of all continuous attributes except one binary categorical attribute and CASP consists entirely of continuous attributes. This suggests that AimNet may fall short to tree-based methods, specifically MissForest, on purely continuous data sets. The four continuous attributes in NYPD are all related to the the coordinate location where the crime occurs. After examining the NYPD data set, while there are other attributes that describe the larger regions where each crime occurs, we as human curators find it difficult to impute reasonable coordinates from the other attributes alone.

5.4 Naturally-Occurring Missing Data

To test the performance of AimNet on more complex and realistic missing cases, we benchmark AimNet and the baselines using the Chicago data set with naturally-occurring missing values. This is a data set released by the City of Chicago on taxi trips and used as a benchmark in data validation methods such as Tensorflow's TFX suite. The four attributes with missing values are `Pickup Census Tract`, `Pickup Community Area`, `Dropoff Census Tract`, and `Dropoff Community Area`. We present the imputation results and run time of each method on the Chicago data set in Table 2. For fairness, the run time for all epoch-based methods is calculated at the end of the epoch where cross-validation loss did not improve performance. We find that AimNet yields an overall accuracy 43% higher than the next-best method MissForest. Moreover, AimNet exhibits practical run time, finishing in just under one hour compared to several days for XGB and MF. Poor run time performance is an artifact of the one-hot encoding requirement of baseline methods for supporting discrete data types.

Deep-Dive Analysis We examine the Chicago data set more closely to determine why AimNet outperforms the baselines by such a huge margin. We find that the attributes with missing values in Chicago are strongly dependent on their corresponding `Latitude` and `Longitude` values. Based on the data set description, we know that there is a functional relationship between these attributes:

$$\begin{aligned} f(\texttt{Latitude}, \texttt{Longitude}) &= \texttt{Census Tract} \\ g(\texttt{Latitude}, \texttt{Longitude}) &= \texttt{Community Area} \end{aligned} \tag{5}$$

Furthermore, we look into the `Pickup Census Tract` attribute and inspect the ground truth census tracts. Ground truth was obtained by querying an external census API (see Appendix A.1 for details). We find that for certain

Table 1: Imputation accuracy and NRMS error on the test set under **MCAR** injections with $p = 0.2$ missingness across AimNet and the other baselines. The means of 10 trials per data set-method with different pseudo-random seeds are reported. Results for each method are after cross-validation on a holdout set. The best results for each data set are **bolded** as well as any results that overlap within the confidence interval ($\pm$ 1 standard deviation).

data set	Accuracy on discrete attributes (ACC $\pm$ std)						
	AimNet	HCQ	XGB	MIDAS	GAIN	MF	MICE
Tic-Tac-Toe	$\mathbf{0.61 \pm 0.01}$	0.53 ± 0.01	0.57 ± 0.02	0.46 ± 0.01	0.32 ± 0.01	0.52 ± 0.01	0.58 ± 0.02
Hospital	$\mathbf{0.99 \pm 0.0}$	$\mathbf{0.99 \pm 0.0}$	0.97 ± 0.01	0.24 ± 0.0	0.13 ± 0.01	$\mathbf{0.99 \pm 0.0}$	0.82 ± 0.01
Mammogram	$\mathbf{0.75 \pm 0.01}$	$\mathbf{0.74 \pm 0.02}$	$\mathbf{0.74 \pm 0.02}$	$\mathbf{0.74 \pm 0.01}$	0.35 ± 0.02	0.68 ± 0.02	0.64 ± 0.02
Thoracic	$\mathbf{0.86 \pm 0.01}$	0.84 ± 0.01	$\mathbf{0.85 \pm 0.01}$	0.84 ± 0.01	0.59 ± 0.09	$\mathbf{0.86 \pm 0.01}$	0.38 ± 0.4
Contraceptive	$\mathbf{0.65 \pm 0.01}$	$\mathbf{0.64 \pm 0.01}$	0.63 ± 0.01	0.63 ± 0.01	0.42 ± 0.01	0.63 ± 0.02	0.57 ± 0.01
Solar Flare	$\mathbf{0.78 \pm 0.02}$	$\mathbf{0.77 \pm 0.02}$	0.76 ± 0.01	0.65 ± 0.01	0.48 ± 0.02	0.76 ± 0.02	0.67 ± 0.02
NYPD	0.92 ± 0.0	0.89 ± 0.0	$\mathbf{0.93 \pm 0.0}$	0.79 ± 0.01	0.14 ± 0.01	0.92 ± 0.0	0.72 ± 0.0
Credit	$\mathbf{0.76 \pm 0.01}$	0.73 ± 0.02	0.75 ± 0.01	0.61 ± 0.02	0.4 ± 0.01	$\mathbf{0.76 \pm 0.01}$	0.68 ± 0.01
Australian	$\mathbf{0.72 \pm 0.02}$	0.69 ± 0.02	0.71 ± 0.02	0.61 ± 0.03	0.45 ± 0.01	$\mathbf{0.73 \pm 0.01}$	0.63 ± 0.02
Balance	$\mathbf{0.79 \pm 0.04}$	$\mathbf{0.78 \pm 0.04}$	0.75 ± 0.05	0.68 ± 0.08	0.5 ± 0.05	0.69 ± 0.05	0.72 ± 0.05
Eye EEG	0.71 ± 0.01	0.63 ± 0.01	0.81 ± 0.01	0.55 ± 0.01	0.54 ± 0.01	$\mathbf{0.87 \pm 0.01}$	0.54 ± 0.01
data set	NRMS on continuous attributes (NRMS $\pm$ std)						
	AimNet	HCQ	XGB	MIDAS	GAIN	MF	MICE
Hospital	$\mathbf{0.72 \pm 0.06}$	1.4 ± 0.36	0.87 ± 0.07	611.12 ± 129.15	1.19 ± 0.02	0.86 ± 0.07	1.13 ± 0.13
Mammogram	$\mathbf{0.91 \pm 0.04}$	1.03 ± 0.08	0.96 ± 0.06	1.12 ± 0.05	1.0 ± 0.11	0.99 ± 0.05	1.27 ± 0.11
Thoracic	$\mathbf{1.1 \pm 0.41}$	1.78 ± 2.33	1.76 ± 2.45	1.71 ± 1.17	$\mathbf{1.5 \pm 0.95}$	1.66 ± 1.61	3.62 ± 4.82
Contraceptive	$\mathbf{0.84 \pm 0.02}$	1.06 ± 0.05	0.87 ± 0.04	1.09 ± 0.03	1.13 ± 0.05	0.88 ± 0.04	1.14 ± 0.06
Solar Flare	$\mathbf{0.94 \pm 0.15}$	$\mathbf{0.94 \pm 0.13}$	1.21 ± 0.53	17698.23 ± 8959.74	$\mathbf{0.96 \pm 0.16}$	$\mathbf{0.96 \pm 0.2}$	1.22 ± 0.34
NYPD	0.15 ± 0.01	1.28 ± 0.53	0.14 ± 0.0	0.62 ± 0.04	3.19 ± 0.41	$\mathbf{0.1 \pm 0.0}$	0.37 ± 0.01
Credit	$\mathbf{0.94 \pm 0.03}$	1.84 ± 0.83	1.09 ± 0.15	1.29 ± 0.24	1.18 ± 0.09	1.04 ± 0.13	1.84 ± 0.83
Australian	$\mathbf{0.94 \pm 0.03}$	2.47 ± 2.0	1.09 ± 0.12	1.22 ± 0.13	1.24 ± 0.18	1.07 ± 0.24	1.58 ± 0.28
Balance	$\mathbf{0.92 \pm 0.02}$	1.37 ± 0.08	1.0 ± 0.03	1.02 ± 0.02	1.03 ± 0.03	1.08 ± 0.05	1.26 ± 0.07
Eye EEG	0.4 ± 0.0	0.94 ± 0.34	0.39 ± 0.0	0.84 ± 0.01	0.65 ± 0.04	$\mathbf{0.35 \pm 0.0}$	0.62 ± 0.0
Phase	$\mathbf{0.45 \pm 0.01}$	0.54 ± 0.03	$\mathbf{0.45 \pm 0.01}$	0.95 ± 0.01	0.76 ± 0.14	0.5 ± 0.01	0.63 ± 0.01
CASP	0.45 ± 0.02	1.45 ± 0.14	0.43 ± 0.02	0.82 ± 0.01	0.72 ± 0.04	$\mathbf{0.41 \pm 0.02}$	0.64 ± 0.03

Table 2: Imputation accuracy and run time on the Chicago data set. Results that could not finish are denoted as —.

Accuracy on discrete attributes for the Chicago data set						
AimNet	HCQ	XGB	MIDAS	GAIN	MF	MICE
$\mathbf{0.73 \pm 0.01}$	0.07 ± 0.0	0.27 ± 0.0	0.09 ± 0.01	0.01 ± 0.01	0.3 ± 0.0	—
Run time (minutes) for the Chicago data set						
AimNet	HCQ	XGB	MIDAS	GAIN	MF	MICE
53	124	5350	176	186	7439	—

census tract values, there is a significant difference between the latitude-longitude coordinates between (1) tuples with observed census tract values and (2) tuples with missing census tract values CT_i. This large margin also appears for `Dropoff Census Tract` and {`Pickup`, `Dropoff`} `Community Area`. Upon further inspection, the systematic separation between observed and missing samples arise from different taxi companies utilizing different centroid coordinates for census tracts while having inconsistent census tract reporting standards. More details on this analysis are reported in Appendix A.4. We hypothesis that AimNet is able to impute the aforementioned systematic errors correctly (in contrast to competing methods) because it learns the aforementioned functional relationships for the Chicago data set. Figure 3 shows the heatmap corresponding to the learned attention weights. We next perform a detailed experimental analysis to validate this hypothesis.

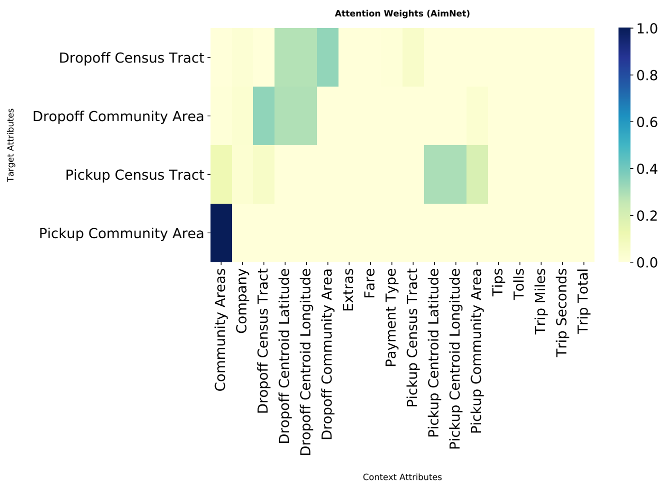

Figure 3: AimNet's learned attention weights for the cencus tract and community area attributes in the Chicago dataset.

5.5 Missing Values and Functional Relationships

Based on our findings from the Chicago dataset, we hypothesize that AimNet's attention mechanism enables our imputation solution to learn functional relationships over the attributes of the input data. We further hypothesize that this learned structure helps counteract sample biases due to systematic noise. To validate this hypothesis, we conduct a series of experiments on both synthetic data sets (where we can control the data generation and functional relationships) and real-world data sets (where we inject systematic error).

A Model for Chicago's Noise We describe a missing data mechanism that imitates how missing values were introduced in the Chicago data set. For a given target attribute Y (a random variable) and a subset of context attribute(s) $\mathcal{X}_Y \subseteq \mathcal{X}$ (random variables corresponding to each context attribute), there are three conditions that characterize the observed systematic missing values: (1) $\mathcal{X}_Y$ correspond to continuous attribute(s) and (2) there exist some function $f(\mathcal{X}_Y) \approx Y$ (we refer to the attributes corresponding to $\mathcal{X}_Y$ as the *dependent attributes* of Y), and (3) there exist a non-trivial subset of target values Y with $\hat{x}_{Y,test} \subseteq x_{Y,test}$ such that $\hat{x}_{Y,test} \cap x_{Y,train} = \varnothing$ (i.e., for a subset of target values we never observe the values of their dependent attributes in the training data). We use this model to construct a series of experiments to the above hypothesis.

Validation with synthetic data We first generate a series of synthetic data sets where the functional relationship $F(\mathcal{X}_Y) = Y$ is known. We generate synthetic data sets $\mathcal{D}_{synth}$ with attributes $\mathcal{Y} = \{Y_i\}_{i=1}^N$ and $\mathcal{X} = \{X_i\}_{i=1}^M$ such that $Y_i = f(\mathcal{X}_{Y_i})$ where $\mathcal{X}_{Y_i} \subseteq \mathcal{X}$. Y_i may be continuous or discrete. If Y_i is discrete, let $C(Y_i)$ denote the cardinality of attribute Y_i (i.e. the number of unique values or classes). For simplicity, let $C(Y_i) = c$ and $|\mathcal{X}_{Y_i}| = k$ for all i: that is the cardinality and number of dependent attributes are constant for all target attributes $\mathcal{Y}$ for a given $\mathcal{D}_{synth}$. We describe the functions $f(\mathcal{X}_Y) = Y$ for both continuous and discrete Y_i's:

Poly(n) (continuous) For a specified degree n and when $k = 1$ ($\mathcal{X}_Y = \{X\}$), $f \in \text{poly}(n)$ is defined as $Y = \prod_{i=1}^n (A_i - X)$ where $X \sim \texttt{Unif}(-1, 1)$ and $A_i \overset{i.i.d}{\sim} \texttt{Unif}(-1, 1)$. Restricting X and the roots A_i ensures that we have a reasonably complex functions with domain $[-1, 1]$ without extreme outliers had $|X| > 1$.

Interact (continuous) For $k \geq 2$, $f \in$ interact is defined as $Y = \sum_{i=1}^k A_i X_i + \sum_{i,j=1, i \neq j}^k B_{i,j} X_i X_j$ where $X_i, A_i, B_{i,j} \overset{i.i.d}{\sim} N(0, 1)$.

Linear (continuous) This is equivalent to Interact where $B_{i,j} = 0 \, \forall i, j$.

Kernel (discrete) Given k dependent attributes and c classes, we generate c class centroids $\{p_j\}_{j=1}^c$ where $p_{j,i} \sim \texttt{Unif}(\min X_i, \max X_i)$ for $i \in \{1, \ldots, k\}$. For a given tuple t we assign $Y^{(t)}$ it the class centroid closest to the k-d point in $\mathcal{X}_{Y^{(t)}}$ under the L2-norm, that is $Y^{(t)} = \arg\min_j \sum_{i=1}^k (p_{j,i} - X_i^{(t)})^2$ where $X_i \overset{i.i.d}{\sim} N(0, 1)$. Note that when $k = 2$, this generates the Voronoi diagram where $\{p_j\}_{j=1}^c$ are the points defining the c Voronoi cells which each correspond to one of the c classes.

We show the imputation performance as we vary properties of $\mathcal{D}_{synth}$ for *continuous* target attributes $\mathcal{Y}$ in Table 3. We observe the following:

1. As the data set size $|\mathcal{D}_{synth}|$ increases, the NRMS error of most methods tends to decrease, but the NRMS error of AimNet decreases at a higher rate.

2. As the number of dependent attributes (k) increases AimNet's imputation accuracy increases and we observe an increasing NRMS error gap between AimNet and other methods. The attention mechanism is able to identify the context attributes (i.e. $\mathcal{X}_Y$) for each target attribute Y as observed in Figure 4.

3. We find that AimNet always outperforms other methods on univariate functions ($k = 1$). Interestingly, as the univariate function becomes more complex and non-linear (see poly(1) vs poly(5)) AimNet's performance advantage diminishes and AimNet requires more samples to obtain performance that is similar to that of tree-based methods. This is expected as learning complex polynomial functions requires more samples. However, when the attribute relations follow simpler multivariate functions **linear** and **interact** (non-linear) AimNet is always able to perform better.

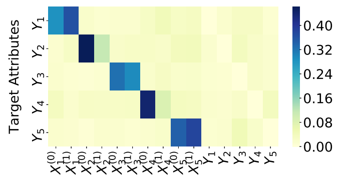

Figure 4: Attention weights of AimNet on synthetic data with $|\mathcal{D}_{synth}| = 5000, \textbf{linear}(k = 2), |Y| = 5$.

We also plot the results for discrete $\mathcal{Y}$ using the *kernel* function in Figure 5. We keep the data set size $|\mathcal{D}_{synth}| = 5000$ but vary the cardinality between $c = 5, 50, 200$ in the three plots. As the number of dependent attributes $\mathcal{X}_Y$ increase, the performance of the baseline methods decreases. For large cardinalities, AimNet fares slightly worse than XGB when $k = 1$ (most likely due to the fact that a high-capacity tree learner is capable of learning the desired 1-D half-spaces) but as k increases beyond 1, AimNet is more capable of learning reasonable Voronoi boundaries in k-D space. The fact that AimNet can learn k-D spaces and develop reasonable bounding regions for discrete classification explains why AimNet outperforms baseline methods on Chicago (see Section 5.4): *census tracts and community areas are both functions of* 2*-D latitude-longitude coordinates.*

Validation with real-world data We apply a similar study to real-world data sets. Since it is difficult to retrieve error-free ground truth in a systematic way for many

Table 3: NRMS cross-validated results on synthetic data sets for continuous target attributes $\mathcal{Y}$ across 3 generation seeds and 3 injection seeds *each* (9 trials per method and setting in total). We vary 1) $|\mathcal{D}_{synth}|$: the size of the data set to see how each method performs given fewer and more samples; 2) $|\mathcal{Y}|$: the number of other (most probably) un-correlated attributes for to see how each method adapts to noise; 3) k: number of dependent attributes $\mathcal{X}_Y$ per target Y to see how each method learns the correct function from multiple inputs; 4) $f(\mathcal{X}_Y)$: the function to see how each method learns simpler, linear functions and more complex, non-linear functions.

$\lvert\mathcal{D}_{\mathbf{synth}}\rvert$	$\lvert\mathcal{Y}\rvert$	**k**	$\mathbf{f}(\mathcal{X}_{\mathbf{Y}})$	AimNet	HCQ	XGB	MIDAS	GAIN	MF	MICE
100	1	1	poly(1)	$\mathbf{0.3 \pm 0.18}$	1.56 ± 0.37	0.9 ± 0.27	10.27 ± 4.05	2.0 ± 1.67	0.96 ± 0.35	6.89 ± 3.51
			poly(2)	2.95 ± 2.74	$\mathbf{1.57 \pm 1.12}$	$\mathbf{1.33 \pm 0.56}$	2.6 ± 2.14	24.71 ± 29.02	$\mathbf{1.42 \pm 0.56}$	4.35 ± 2.87
			poly(5)	1.64 ± 1.06	2.23 ± 2.56	$\mathbf{0.84 \pm 0.34}$	138.56 ± 174.05	38.27 ± 23.34	$\mathbf{0.89 \pm 0.32}$	1.83 ± 0.96
		2	linear	$\mathbf{0.68 \pm 0.4}$	1.51 ± 0.73	1.1 ± 0.65	1.73 ± 1.04	1.91 ± 0.98	1.08 ± 0.56	$\mathbf{0.71 \pm 0.51}$
			interact	$\mathbf{0.73 \pm 0.12}$	1.06 ± 0.25	0.88 ± 0.34	1.12 ± 0.14	1.54 ± 0.65	0.86 ± 0.26	1.26 ± 0.31
		5	linear	$\mathbf{0.59 \pm 0.36}$	1.37 ± 0.87	$\mathbf{0.85 \pm 0.67}$	2.75 ± 1.62	1.9 ± 1.55	0.93 ± 0.69	$\mathbf{0.47 \pm 0.45}$
			interact	$\mathbf{0.63 \pm 0.19}$	1.1 ± 0.09	1.03 ± 0.42	2.5 ± 0.94	1.38 ± 0.59	0.97 ± 0.21	1.35 ± 0.39
	5	1	poly(1)	$\mathbf{0.53 \pm 0.16}$	6.83 ± 0.92	1.04 ± 0.18	11.63 ± 1.89	9.78 ± 2.01	1.21 ± 0.17	7.79 ± 1.04
			poly(2)	2.37 ± 1.38	6.7 ± 4.67	$\mathbf{1.44 \pm 0.31}$	3.12 ± 0.87	17.11 ± 9.33	$\mathbf{1.65 \pm 0.43}$	9.29 ± 8.0
			poly(5)	115.1 ± 263.89	30.82 ± 23.45	$\mathbf{3.13 \pm 3.79}$	1982.04 ± 2880.58	1510.47 ± 2531.43	$\mathbf{4.17 \pm 5.01}$	24.88 ± 34.39
		2	linear	$\mathbf{0.88 \pm 0.17}$	2.06 ± 0.29	1.22 ± 0.26	3.7 ± 1.25	3.08 ± 0.86	1.25 ± 0.23	$\mathbf{0.88 \pm 0.22}$
			interact	$\mathbf{1.02 \pm 0.33}$	1.41 ± 0.26	$\mathbf{1.29 \pm 0.48}$	2.98 ± 1.38	1.76 ± 0.55	$\mathbf{1.34 \pm 0.52}$	1.45 ± 0.45
		5	linear	0.78 ± 0.19	1.49 ± 0.2	1.03 ± 0.18	3.05 ± 0.77	1.69 ± 0.31	1.05 ± 0.2	$\mathbf{0.37 \pm 0.15}$
			interact	$\mathbf{0.97 \pm 0.1}$	1.16 ± 0.09	1.21 ± 0.16	3.0 ± 0.71	1.15 ± 0.1	1.13 ± 0.09	1.69 ± 0.22
5000	1	1	poly(1)	$\mathbf{0.13 \pm 0.08}$	1.2 ± 0.2	0.98 ± 0.06	2.86 ± 1.21	1.5 ± 0.77	1.01 ± 0.05	5.85 ± 1.05
			poly(2)	$\mathbf{0.88 \pm 1.02}$	$\mathbf{1.61 \pm 0.64}$	$\mathbf{1.38 \pm 0.65}$	4.18 ± 4.13	14.63 ± 14.62	$\mathbf{1.44 \pm 0.58}$	$\mathbf{1.48 \pm 0.56}$
			poly(5)	2.38 ± 2.95	$\mathbf{1.46 \pm 0.59}$	$\mathbf{1.22 \pm 0.6}$	26.69 ± 44.08	19.73 ± 29.46	$\mathbf{1.5 \pm 0.84}$	$\mathbf{1.5 \pm 0.9}$
		2	linear	$\mathbf{0.17 \pm 0.23}$	1.46 ± 1.0	0.67 ± 0.53	1.54 ± 1.02	1.99 ± 1.26	0.68 ± 0.53	$\mathbf{0.27 \pm 0.49}$
			interact	$\mathbf{0.45 \pm 0.3}$	1.2 ± 0.39	$\mathbf{0.65 \pm 0.35}$	1.33 ± 0.81	1.25 ± 0.4	$\mathbf{0.65 \pm 0.35}$	1.34 ± 0.54
		5	linear	$\mathbf{0.05 \pm 0.03}$	1.16 ± 0.12	0.38 ± 0.14	1.11 ± 0.25	0.89 ± 0.19	0.45 ± 0.15	$\mathbf{0.04 \pm 0.03}$
			interact	$\mathbf{0.18 \pm 0.05}$	1.04 ± 0.06	0.56 ± 0.09	1.08 ± 0.11	1.15 ± 0.21	0.59 ± 0.09	1.34 ± 0.22
	5	1	poly(1)	$\mathbf{0.1 \pm 0.04}$	3.68 ± 1.27	0.98 ± 0.04	2.88 ± 1.06	7.3 ± 1.46	1.0 ± 0.03	6.28 ± 0.61
			poly(2)	$\mathbf{0.71 \pm 0.52}$	5.47 ± 8.25	1.26 ± 0.31	6.02 ± 3.73	18.83 ± 16.09	1.29 ± 0.27	1.99 ± 0.97
			poly(5)	2.03 ± 1.01	2.44 ± 0.89	$\mathbf{1.35 \pm 0.38}$	42.16 ± 48.01	169.58 ± 135.82	$\mathbf{1.57 \pm 0.5}$	2.45 ± 1.3
		2	linear	$\mathbf{0.4 \pm 0.23}$	1.89 ± 0.36	0.85 ± 0.22	2.0 ± 0.45	2.57 ± 0.56	0.88 ± 0.21	0.87 ± 0.67
			interact	$\mathbf{0.45 \pm 0.15}$	1.35 ± 0.28	0.74 ± 0.17	1.48 ± 0.4	1.67 ± 0.55	0.76 ± 0.17	1.32 ± 0.27
		5	linear	$\mathbf{0.08 \pm 0.05}$	1.23 ± 0.14	0.45 ± 0.12	1.34 ± 0.19	1.34 ± 0.14	0.55 ± 0.11	0.17 ± 0.11
			interact	$\mathbf{0.16 \pm 0.03}$	1.06 ± 0.05	0.6 ± 0.05	1.09 ± 0.07	1.09 ± 0.06	0.67 ± 0.06	1.29 ± 0.12
20000	1	1	poly(1)	$\mathbf{0.11 \pm 0.08}$	1.54 ± 0.41	1.01 ± 0.02	3.48 ± 1.01	0.57 ± 0.4	1.02 ± 0.02	5.84 ± 1.39
			poly(2)	$\mathbf{0.7 \pm 0.97}$	1.89 ± 0.68	$\mathbf{1.34 \pm 0.61}$	4.49 ± 5.48	4.63 ± 5.05	$\mathbf{1.38 \pm 0.58}$	$\mathbf{1.37 \pm 0.58}$
			poly(5)	$\mathbf{0.86 \pm 0.37}$	1.81 ± 0.96	1.27 ± 0.5	86.03 ± 194.59	49.72 ± 129.01	1.32 ± 0.5	1.3 ± 0.49
		2	linear	$\mathbf{0.22 \pm 0.31}$	1.73 ± 0.9	0.76 ± 0.52	1.73 ± 0.67	1.18 ± 0.84	0.76 ± 0.52	$\mathbf{0.29 \pm 0.42}$
			interact	$\mathbf{0.29 \pm 0.17}$	1.22 ± 0.39	0.64 ± 0.25	1.18 ± 0.24	1.28 ± 0.46	0.64 ± 0.25	1.3 ± 0.44
		5	linear	$\mathbf{0.03 \pm 0.04}$	1.29 ± 0.37	0.4 ± 0.32	1.17 ± 0.36	0.87 ± 0.41	0.44 ± 0.32	$\mathbf{0.04 \pm 0.06}$
			interact	$\mathbf{0.1 \pm 0.03}$	1.08 ± 0.08	0.46 ± 0.09	1.05 ± 0.08	0.96 ± 0.17	0.49 ± 0.09	1.14 ± 0.13
	5	1	poly(1)	$\mathbf{0.19 \pm 0.07}$	6.31 ± 1.41	0.99 ± 0.01	4.43 ± 0.53	8.43 ± 0.98	1.0 ± 0.01	7.26 ± 0.31
			poly(2)	$\mathbf{0.52 \pm 0.4}$	2.91 ± 0.75	1.15 ± 0.25	4.57 ± 3.08	11.74 ± 9.24	1.18 ± 0.24	1.57 ± 0.42
			poly(5)	1.98 ± 1.58	2.88 ± 1.49	$\mathbf{1.51 \pm 0.31}$	128.15 ± 223.35	172.29 ± 256.97	$\mathbf{1.63 \pm 0.37}$	2.34 ± 0.91
		2	linear	$\mathbf{0.28 \pm 0.16}$	1.97 ± 0.36	0.89 ± 0.22	1.99 ± 0.44	2.53 ± 0.48	0.9 ± 0.22	1.07 ± 0.83
			interact	$\mathbf{0.47 \pm 0.11}$	1.65 ± 0.41	0.87 ± 0.29	1.82 ± 0.52	2.08 ± 0.49	0.88 ± 0.29	1.39 ± 0.41
		5	linear	$\mathbf{0.06 \pm 0.04}$	1.35 ± 0.2	0.5 ± 0.16	1.36 ± 0.19	1.44 ± 0.19	0.59 ± 0.15	0.44 ± 0.38
			interact	$\mathbf{0.11 \pm 0.07}$	1.04 ± 0.03	0.5 ± 0.03	1.06 ± 0.04	1.07 ± 0.04	0.56 ± 0.03	1.26 ± 0.07

naturally-occurring errors, we turn to missing injection via MAR and MNAR (Section 3). We identify (by inspection) functional relationships with real-valued domains between attributes that satisfy conditions (1) and (2) of the hypothesis in our benchmark data sets, including those with few to no naturally-occurring errors. These relationships are outlined in Appendix A.1. Most of them in fact correspond to one of the configurations in the synthetic experiments. To simulate the conditions of our hypothesis, we inject two types of systematic errors depending on the target attribute type with 20% missing percentage (*missingness*). The details on how we inject systematic errors are provided in Appendix A.5.

Table 4 shows the imputation results of all models on the target attributes with MAR and MNAR injected errors. We see that AimNet achieves the best or tied-for-the-best results on every functional relationship within noise. We observe similar performance characteristics for the $k = 2$ discrete targets amongst the NYPD functional relationships as the results we observe in the Chicago data set. Even for simple linear real-valued relationships in the Phase and Chicago data sets, AimNet performs slightly better than XGB, except for `Trip Total`. Upon inspection, we find that AimNet's attention weights place a large weight on `Extras`, a mostly 0-valued attribute, and a small weight on `Fare`. Intuitively, there should be a large weight on `Fare` since for most cases `Trip total` $\approx$ `Fare`. XGB correctly identifies `Fare` as the most important feature for `Trip Total`.

5.6 Micro-Benchmark Experiments

We perform micro-benchmark experiments on AimNet, specifically (1) an ablation study on the attention layer (2) a sensitivity analysis on the hyperparameters of AimNet and (3) a comparison with multi-task learning versus learning a single model per attribute.

Effects of the attention layer We perform an ablation study by removing the attention layer while imputing on both synthetic and MAR and MNAR injected data sets. Here, we summarize our findings from the ablation study

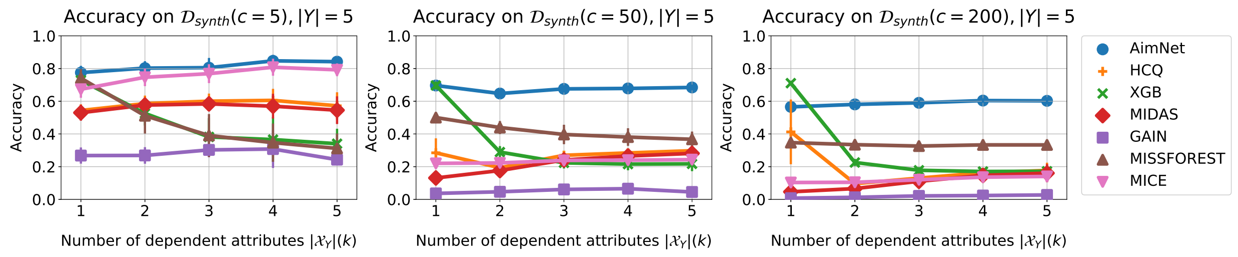

Figure 5: Accuracy on synthetic data sets with $|\mathcal{D}_{synth}| = 5000$, $|\mathcal{Y}| = 5$ varying number of dependent attributes $|\mathcal{X}_Y|$ over different class sizes $c = 5, 50, 200$.

Table 4: Imputation accuracy and NRMS error on the test set under **MAR/MNAR** error injections with $p = 0.2$ missingness across our AimNet model and the other baselines. Results that did not finish after 3 days are denoted with —.

data set	Attribute	Accuracy on discrete attributes (ACC $\pm$ std)						
		AimNet	HCQ	XGB	MIDAS	GAIN	MF	MICE
Balance	class	$\mathbf{0.83 \pm 0.07}$	0.4 ± 0.37	0.48 ± 0.32	0.7 ± 0.16	0.5 ± 0.12	0.46 ± 0.34	$\mathbf{0.78 \pm 0.15}$
	ADDR_PCT_CD	$\mathbf{0.67 \pm 0.03}$	0.19 ± 0.05	0.41 ± 0.05	0.13 ± 0.01	0.04 ± 0.04	0.59 ± 0.03	0.23 ± 0.01
NYPD	BORO_NM	$\mathbf{0.92 \pm 0.07}$	$\mathbf{0.85 \pm 0.09}$	0.58 ± 0.18	0.78 ± 0.04	0.23 ± 0.03	0.84 ± 0.09	0.58 ± 0.09
	PATROL_BORO	$\mathbf{0.83 \pm 0.07}$	0.69 ± 0.03	0.57 ± 0.17	0.6 ± 0.06	0.13 ± 0.02	0.72 ± 0.1	0.58 ± 0.07

data set	Attribute	NRMS on continuous attributes (NRMS $\pm$ std)						
		AimNet	HCQ	XGB	MIDAS	GAIN	MF	MICE
	A	$\mathbf{0.75 \pm 0.13}$	1.26 ± 0.46	$\mathbf{0.81 \pm 0.18}$	1.45 ± 0.5	2.04 ± 1.73	0.94 ± 0.18	1.41 ± 0.66
	B	$\mathbf{0.77 \pm 0.07}$	0.99 ± 0.27	$\mathbf{0.83 \pm 0.16}$	1.18 ± 0.33	1.54 ± 0.54	0.98 ± 0.13	1.32 ± 0.4
Phase	C	$\mathbf{0.79 \pm 0.11}$	1.25 ± 0.26	$\mathbf{0.81 \pm 0.13}$	1.44 ± 0.25	1.33 ± 0.34	1.09 ± 0.23	1.29 ± 0.1
	D	$\mathbf{0.62 \pm 0.09}$	1.1 ± 0.24	$\mathbf{0.65 \pm 0.09}$	1.47 ± 0.56	1.57 ± 0.63	0.85 ± 0.17	1.03 ± 0.13
	Trip Total	0.82 ± 0.21	2.38 ± 1.16	$\mathbf{0.48 \pm 0.32}$	3.84 ± 0.7	1.87 ± 0.7	$\mathbf{0.71 \pm 0.57}$	—
Chicago	Fare	$\mathbf{1.35 \pm 0.76}$	5.18 ± 1.94	$\mathbf{1.16 \pm 0.73}$	12.73 ± 7.93	58.09 ± 36.03	2.78 ± 3.0	—
	Tips	$\mathbf{0.52 \pm 0.01}$	1.29 ± 0.09	$\mathbf{0.5 \pm 0.12}$	1.59 ± 0.49	11.64 ± 6.76	0.8 ± 0.28	—

here and present all detailed figures in Appendix A.6.

First, we focus on synthetic and MAR and MNAR injected data sets. For data sets with $|Y| = 1$ (each synthetic data set consists of only one set of Y and $\mathcal{X}$) we find that the attention mechanism contributes nothing to the performance. However, once we introduce $|Y| = 5$ sets of $\mathcal{X}, Y$ we observe that as the number of classes increases, the attention mechanism eventually accounts for $> 50\%$ of the prediction accuracy. For sufficiently difficult imputation problems where the number of classes is large, and where there are other irrelevant attributes to the functional relationship in the data set, the attention mechanism helps identify the correct inputs $\mathcal{X}_Y$ for a given Y attribute.

We perform the same ablation study on the real-world MAR and MNAR injected data sets. We find that for discrete attributes, the attention mechanism accounts for $5-10\%$ of the imputation accuracy on the NYPD and Chicago attributes, all of which exhibit the *kernel* functional relationship for $k = 2$. Interestingly for the `class` attribute in the Balance data set the attention layer has a non-trivial negative impact on imputation accuracy. In fact, without the attention layer AimNet would outperform all other baselines on Balance in Table 4 after accounting for noise. For continuous attributes, the attention layer has no effect except on the Chicago attributes with the functional relationships in Equation 5.

Hyperparameter sensitivity analysis We vary the hyperparameters of AimNet to assess its sensitivity to hyperparameter perturbations. We focus on the dropout rate, max domain size, and the embedding dimension k. We summarize our findings while detailed results are presented in Appendix A.7. We find that AimNet is robust to a wide range of values for max domain size and the embedding dimension on both discrete and continuous evaluation. Dropout rate, on the other hand, can impact performance on continuous variable and should be tuned with cross-validation.

Multi-task or single-task We compare multi-task AimNet, where context parameters are shared across target attributes, with independently-trained models for each target attribute. Detailed results are shown in Appendix A.10. While there are cases where the multi-task models are more accurate we find that, on average, independently-trained models obtain results of comparable quality. That being said, independent models can trained in parallel with sub-linear speedup (usually 2-10 times faster).

6 Conclusion

We introduced AimNet a simple attention-based autoencoder network for missing data imputation for mixed continuous and discrete data. We showed that AimNet outperforms current state-of-the-art especially in the presence of systematic noise. The key component that enables AimNet to counteract sample biases due to systematic noise is a new variation of a dot product attention mechanism that can learn structural properties of the underlying data distribution.

References

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

Belkin, M., Hsu, D. J., and Mitra, P. Overfitting or perfect fitting? risk bounds for classification and regression rules that interpolate. In *NeurIPS*, pp. 2306–2317, 2018.

Buuren, S. v. and Groothuis-Oudshoorn, K. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, pp. 1–68, 2010.

Cai, J., Candès, E. J., and Shen, Z. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.

Chen, T. and Guestrin, C. Xgboost: A scalable tree boosting system. In *KDD*, pp. 785–794. ACM, 2016.

Council, N. R. et al. *The prevention and treatment of missing data in clinical trials*. National Academies Press, 2010.

D'Ambrosio, A., Aria, M., and Siciliano, R. Accurate tree-based missing data imputation and data fusion within the statistical learning paradigm. *J. Classification*, 29 (2):227–258, 2012. doi: 10.1007/s00357-012-9108-1. URL https://doi.org/10.1007/s00357-012-9108-1.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

García-Laencina, P. J., Sancho-Gómez, J., and Figueiras-Vidal, A. R. Pattern classification with missing data: a review. *Neural Computing and Applications*, 19(2): 263–282, 2010.

Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., and Kagal, L. Explaining explanations: An overview of interpretability of machine learning. In *DSAA*, pp. 80–89. IEEE, 2018.

Gondara, L. and Wang, K. MIDA: multiple imputation using denoising autoencoders. In *PAKDD (3)*, volume 10939 of *Lecture Notes in Computer Science*, pp. 260–272. Springer, 2018.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. C., and Bengio, Y. Generative adversarial nets. In *NIPS*, pp. 2672–2680, 2014.

Jaseena, K. and David, J. M. Issues, challenges, and solutions: big data mining. *NeTCoM, CSIT, GRAPH-HOC, SPTM–2014*, pp. 131–140, 2014.

Kang, H. The prevention and handling of the missing data. *Korean journal of anesthesiology*, 64(5):402, 2013.

Keshavan, R. H., Montanari, A., and Oh, S. Matrix completion from a few entries. *IEEE Trans. Information Theory*, 56(6):2980–2998, 2010.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In *ICLR*, 2014.

Little, R. J. and Rubin, D. B. *Statistical analysis with missing data*, volume 793. John Wiley & Sons, 2019.

Loshchilov, I. and Hutter, F. SGDR: stochastic gradient descent with warm restarts. In *ICLR (Poster)*. OpenReview.net, 2017.

Mattei, P. and Frellsen, J. MIWAE: deep generative modelling and imputation of incomplete data sets. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4413–4423. PMLR, 2019.

Mazumder, R., Hastie, T., and Tibshirani, R. Spectral regularization algorithms for learning large incomplete matrices. *J. Mach. Learn. Res.*, 11:2287–2322, 2010.

Nauta, M., Bucur, D., and Seifert, C. Causal discovery with attention-based convolutional neural networks. *Machine Learning and Knowledge Extraction*, 1(1):312–340, 2019.

Nazábal, A., Olmos, P. M., Ghahramani, Z., and Valera, I. Handling incomplete heterogeneous data using vaes. *CoRR*, abs/1807.03653, 2018.

Rekatsinas, T., Chu, X., Ilyas, I. F., and Ré, C. Holoclean: Holistic data repairs with probabilistic inference. *PVLDB*, 10(11):1190–1201, 2017.

Rubin, D. B. Inference and missing data. *Biometrika*, 63(3): 581–592, 1976.

Salimans, T., Goodfellow, I. J., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training gans. In *NIPS*, pp. 2226–2234, 2016.

Saunders, J. A., Morrow-Howell, N., Spitznagel, E., Doré, P., Proctor, E. K., and Pescarino, R. Imputing missing data: A comparison of methods for social work researchers. *Social work research*, 30(1):19–31, 2006.

Sentas, P. and Angelis, L. Categorical missing data imputation for software cost estimation by multinomial logistic regression. *Journal of Systems and Software*, 79(3):404–414, 2006. doi: 10.1016/j.jss.2005.02.026. URL https://doi.org/10.1016/j.jss.2005.02.026.

Soley-Bori, M. Dealing with missing data: Key assumptions and methods for applied analysis. *Boston University*, 23, 2013.

Stekhoven, D. J. and Bühlmann, P. Missforest - non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 2012. doi: 10.1093/bioinformatics/btr597. URL https://doi.org/10.1093/bioinformatics/btr597.

Troyanskaya, O. G., Cantor, M. N., Sherlock, G., Brown, P. O., Hastie, T., Tibshirani, R., Botstein, D., and Altman, R. B. Missing value estimation methods for DNA microarrays. *Bioinformatics*, 17(6):520–525, 2001. doi: 10.1093/bioinformatics/17.6.520. URL https://doi.org/10.1093/bioinformatics/17.6.520.

Tsai, Y.-H. H., Bai, S., Yamada, M., Morency, L.-P., and Salakhutdinov, R. Transformer dissection: An unified understanding for transformer's attention via the lens of kernel. *arXiv preprint arXiv:1908.11775*, 2019.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In *NIPS*, pp. 5998–6008, 2017.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P. Extracting and composing robust features with denoising autoencoders. In *ICML*, volume 307 of *ACM International Conference Proceeding Series*, pp. 1096–1103. ACM, 2008.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J. G., Salakhutdinov, R., and Le, Q. V. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237, 2019. URL http://arxiv.org/abs/1906.08237.

Yoon, J., Jordon, J., and van der Schaar, M. GAIN: missing data imputation using generative adversarial nets. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5675–5684. PMLR, 2018.

A Appendix

A.1 Detailed Data Set Descriptions

A summary of the characteristics of each data set is provided in Table 5. In detail, we use the following data sets:

- Hospital is a benchmark data set used in the data cleaning literature (Rekatsinas et al., 2017). In our experiments, we inject missing errors instead.
- NYPD [2] contains violation crimes reported to the New York City Police Department (NYPD). We focus on the latest snapshot of the data set (June 2019) and remove some of the attributes relating only to time. In addition, since we cannot obtain ground truth in a semi-automated fashion for existing missing values, we simply remove all the rows containing missing values and later inject missing errors.
- Chicago [3] consists of data on Chicago taxi rides. Chicago contains naturally-occurring missing values. As far as we know, no continuous errors exist beyond the case where the entire tuple is missing. The primary attributes with missing values are the "Census Tracts" and "Community Area" columns which denote the US census tracts and the Chicago-defined community areas the taxi ride took place, respectively. In order to judge the performance of MDI methods on the naturally-occurring errors, ground truth for census tracts may be queried by latitude-longitude from an FCC API[4] and community areas can be subsequently matched via the census tract with a join table for community areas [5]. Since this data set is the tens of gigabytes in size we sample 40000 rows from each of the top 10 taxi companies by number of rides.
- Phase is a data set on three-phase current traces collected by a third-party company. Due to privacy policies we cannot share the source in this paper.
- We use 10 data sets from the UCI repository [6] (Dua & Graff, 2017). All of them are used as-is in the experiments except the Eye EEG data set which contains outlier values we subsequently dropped (if they are beyond three standard deviations from the mean).

[2]https://data.cityofnewyork.us/Public-Safety/NYPD-Complaint-Data-Current-Year-To-Date-/5uac-w243

[3]https://data.cityofchicago.org/Transportation/Taxi-Trips/wrvz-psewurl/

[4]https://www.fcc.gov/census-block-conversions-api

[5]http://robparal.blogspot.com/2012/04/census-tracts-in-chicago-community.html

[6]http://archive.ics.uci.edu/ml/datasets/

Table 5: Data sets used in experiments sorted by the proportion of discrete to continuous attributes.

Data Set	$\lvert r \rvert$	# Continuous Attributes	# Discrete Attributes
Tic-Tac-Toe	958	0	10
Hospital	1000	2	14
Mammogram	831	1	5
Thoracic	470	3	14
Contraceptive	1473	2	8
Solar Flare	1066	3	10
NYPD	32399	4	13
Credit	653	6	10
Australian	691	6	9
Chicago	400k	11	7
Balance	625	4	1
Eye EEG	14976	14	1
Phase	9628	4	0
CASP	45730	10	0

We present functional relationships in the above data sets in Table 7. The functional relationships fall on the spectrum of the synthetic experiments (see Section 5.5): `Trip Total` from Chicago and `D` from Phase both correspond to $k = 4$ and $k = 3$ for $f(\mathcal{X}_Y) \in$ **linear**, respectively; `class` from Balance corresponds to $k = 4$ for $f(\mathcal{X}_Y) \in$ **interact**; all other functional relationships where $\mathcal{X}_Y$ is some variant of 2-D coordinates correspond to $k = 2$ and $f(\mathcal{X}_Y) \in$ **kernel**.

A.2 Hyperparameter Tuning

Table 6: Set of hyperparameters for each model over which we perform grid search cross-validation.

Method	Hyperparameter	Search Space
AimNet	dropout %	$[0, 0.25, 0.5]$
HCQ	weight decay	$[0, 0.01, 0.1]$
XGB	gamma	$[0, 0.1, 1]$
MIDAS	keep %	$[0.8, 0.65, 0.5]$
GAIN	alpha	$[0.1, 1, 10]$
MF	# trees	$[50, 100, 300]$
MICE	# iterations	$[1, 3, 5]$

Given the large number of hyperparameters in each of the baseline methods and the numerous data sets we wish to benchmark against, it is intractable to perform a thorough hyperparameter search for every baseline method. For all baselines, we begin with their default parameters as described in their corresponding papers or specified in their open-source implementations. We choose the most influential hyperparameter for each method, as shown in Table 6, to perform grid search cross-validation across.

Table 7: Functional relationships with real-valued domains for each data set where $Y = f(\mathcal{X}_Y)$. $*$: since Y and $\mathcal{X}_Y$ are both continuous and f is linear, all permutations are also valid functional relationships.

Data Set	Y	$\mathcal{X}_{\mathbf{Y}}$ (continuous)
Chicago	Pickup Census Tract	{Pickup Centroid Latitude, Pickup Centroid Longitude}
	Dropoff Census Tract	{Dropoff Centroid Latitude, Dropoff Centroid Longitude}
	Pickup Community Area	{Pickup Centroid Latitude, Pickup Centroid Longitude}
	Dropoff Community Area	{Dropoff Centroid Latitude, Dropoff Centroid Longitude}
	Trip Total*	Fare + Tips + Tolls + Extras
NYPD	ADDR_PCT_CD	{Latitude, Longitude} {X_COORD_CD, Y_COORD_CD}
	PATROL_BORO	{Latitude, Longitude} {X_COORD_CD, Y_COORD_CD}
	BORO_NM	{Latitude, Longitude} {X_COORD_CD, Y_COORD_CD}
Phase	D*	A + B + C
Balance	class	left_distance × left_weight − right_distance × right_weight

A.3 Training

For all experiments a default embedding size $k = 64$ is used with a maximum pruned domain size of $D = 50$ for AimNet. We always train AimNet with 20 epochs (although we observe in almost all data sets AimNet converges in fewer than 3), and each mini-batch consists of 1 sample if $|\mathcal{D}| \leq 2000$ or 32 samples if $|\mathcal{D}| > 2000$. Since AimNet uses a mini-batch approach, samples that have a smaller domain than D have negative values randomly sampled into its softmax loss since the softmax arguments are padded anyways to size D for mini-batch training.

A.4 Chicago Deep-Dive

In Figure 6 we plot the latitude-longitude coordinates for a region of Chicago from the Chicago data set and all samples in that region. We label each point on the plane with its `Pickup Census Tract` label. Note that since these are centroid `Latitudes` and `Longitudes` there are multiple samples per coordinate point. We display both the coordinates of the observed samples (which a model can train on) and the missing samples (which have coordinates but have missing census tracts). There is an apparent gap between missing and observed samples for a particular census tract. We demarcate the true boundaries of the census tracts in Figure 7 and notice that the observed and missing samples do indeed within their corresponding census tract boundaries. Upon further inspection, the systematic separation between observed and missing samples arise from different taxi companies utilizing different centroid coordinates for census tracts while having inconsistent census tract reporting standards.

A.5 MAR/MNAR Injection on Real Data Sets

Suppose the error percentage is x and e.g. $x = 20\%$. In either cases, let A be the target attribute in which we inject missing values. We uniform-randomly choose one if its continuous dependent attribute B. We then inject missing values into A according to its type:

Continuous Target (MAR injection)

1. Sort all tuples in ascending or descending order (randomly chosen) with respect to values of $t[B]$.
2. Uniform-randomly choose a contiguous interval $I_{A|B}$ in $t[B]$ equivalent to x of all tuples. $I_{A|B}$ cannot start nor end at the endpoints of $t[B]$.
3. Inject missing into all $t[A]$ cells whose co-occurring values in $t[B]$ is within $I_{A|B}$.

Discrete Target (MNAR injection)

1. Group tuples by the values of $t[A]$
2. For a given group g_i (for some value $v_i \in \text{dom}(A)$), randomly choose a cut-off value c. c will either be the x-th largest or smallest (randomly chosen) among all the co-occurring values in B (i.e. $g_i[B]$).
3. Inject missing into all $g_i[A] = v_i$ cells whose co-occurring values in attribute B is beyon the cut-off c. That is if c is the x-th largest then inject missing if $g_i[B] > c$ or if c is the x-th smallest then inject missing if $g_i[B] < c$.

A.6 Detailed Ablation Study Results

We perform an ablation study by removing the attention layer while imputing on both synthetic and MAR and MNAR injected data sets. We plot the results of removing the attention layer on the *kernel* data sets from Section 5.5 in Figure 8. For $|Y| = 1$ (each synthetic data set consists of only one set of Y and $\mathcal{X}$) in the top row the attention mechanism contributes nothing to the performance. However, once we introduce $|Y| = 5$ sets of $\mathcal{X}, Y$ we observe that as the number of classes increases, the attention mechanism eventually accounts for $> 50\%$ of the prediction accuracy. For sufficiently difficult imputation problems where the number of classes is large, and where there are other irrelevant

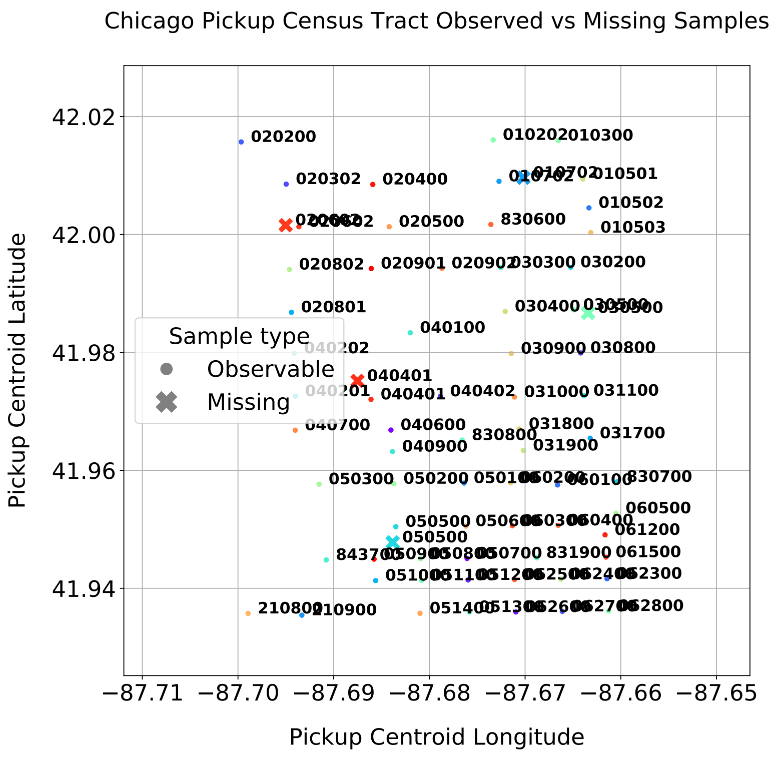

Figure 6: Observed vs missing `Pickup Census Tract` samples in the Chicago data set.

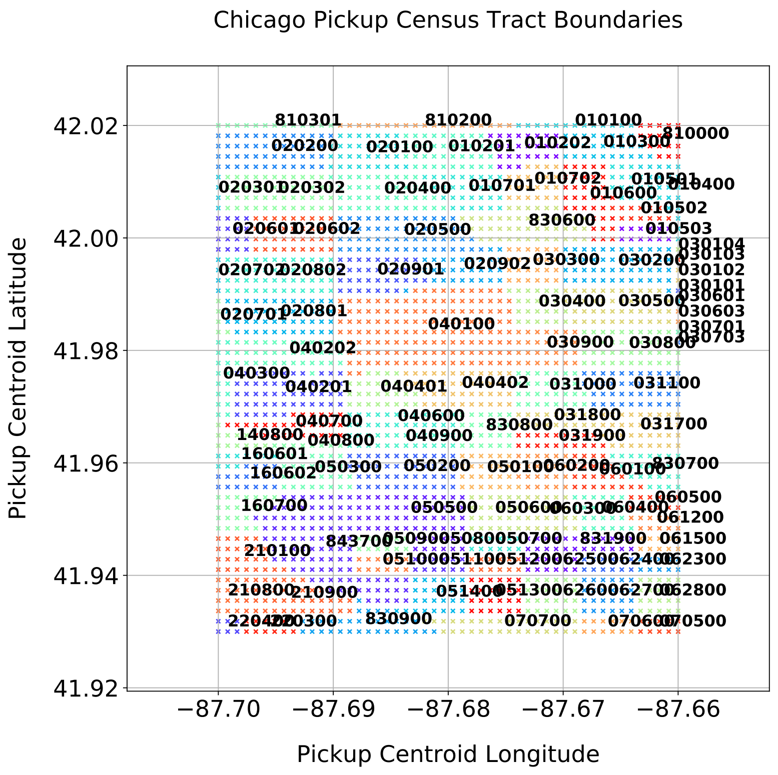

Figure 7: Approximate bounding polygons of attribute `Pickup Census Tract` for the Chicago data set.

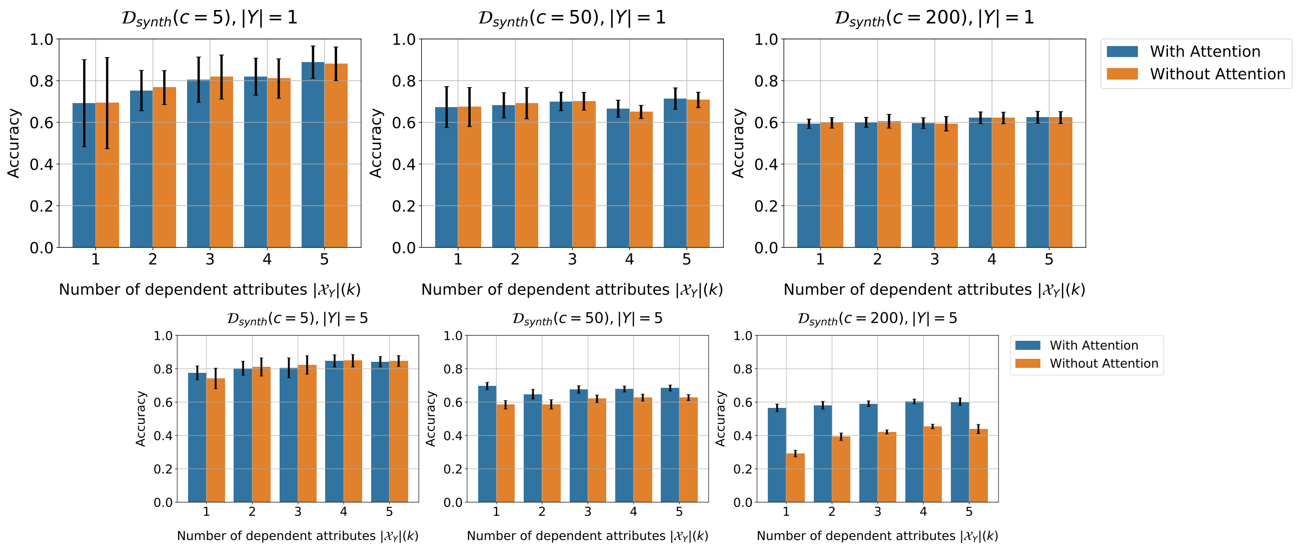

Figure 8: Accuracy on synthetic data sets of AimNet with/without attention layer with $|\mathcal{D}_{synth}| = 5000$, $|\mathcal{Y}| = 1, 5$ varying number of dependent attributes $|\mathcal{X}_Y|$ over different class sizes $c = 5, 50, 200$.

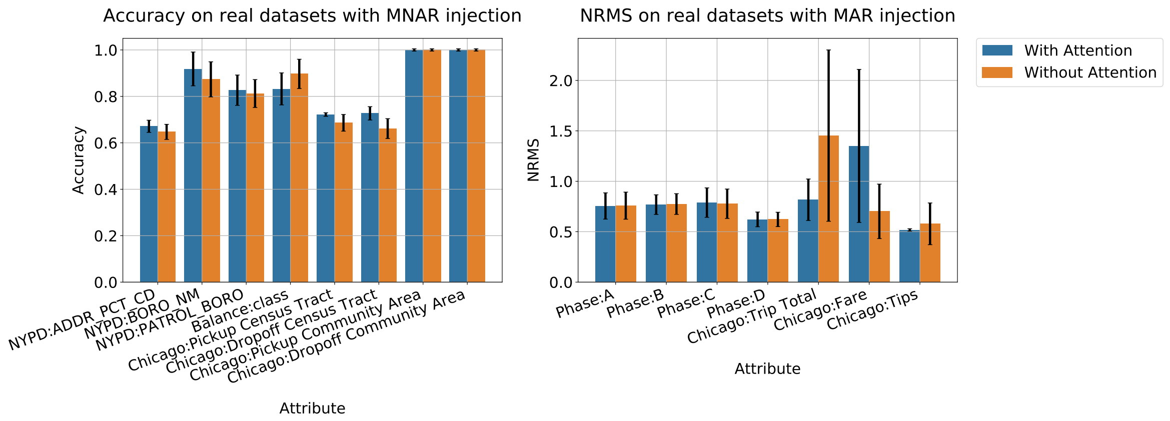

Figure 9: Imputation Accuracy and NRMS error on real data sets under MNAR/MAR injections of AimNet with and without attention layer.

attributes to the functional relationship in the data set, the attention mechanism helps identify the correct inputs $\mathcal{X}_Y$ for a given Y attribute.

We perform the same ablation study on the real-world MAR and MNAR injected data sets and plot the results in Figure 9. We find that for discrete attributes, the attention mechanism accounts for $5-10\%$ of the imputation accuracy on the NYPD and Chicago attributes, all of which exhibit the *kernel* functional relationship for $k = 2$. Interestingly for the `class` attribute in the Balance data set the attention layer has a non-trivial negative impact on imputation accuracy. In fact, without the attention layer AimNet would outperform all other baselines on Balance in Table 4 after accounting for noise. For continuous attributes, the attention layer has no effect except on the Chicago attributes with the functional relationships in Equation 5.

A.7 Sensitivity Analysis

We vary the hyperparameters of AimNet to assess the sensitivity of the model to hyperparameter perturbations, specifically dropout rate, max domain size and the embedding size k. The sensitivity analysis is performed on the NYPD data set with MCAR-injected missing values where $p = 0.2$ and the results are shown in Table 8. We observe that for discrete predictions and the corresponding imputation accuracy, the model can perform about the same regardless of the dropout rate, max domain size, and embedding size. In fact a smaller domain size and embedding size would improve run time and make AimNet even more competitive in a practical setting. We do observe that for continuous target attributes,

Table 8: Sensitivity analysis of AimNet's hyperparameters.

Accuracy on discrete attributes for the NYPD data set				
	dropout	max domain size	embedding size	AimNet
base	0.25	50	64	0.921
(dropout rate)	0.0			0.918
	0.5			0.920
(max domain size)		10		0.917
		100		0.921
(embedding size)			16	0.920
			32	0.921
			128	**0.922**
			256	0.920
NRMS on continuous attributes for the NYPD data set				
	dropout rate	max domain size	embedding size	AimNet
base	0.0	50	64	**0.150**
(dropout rate)	0.25			0.281
	0.5			0.509
(embedding size)			16	0.159
			32	**0.150**
			128	0.153
			256	**0.144**
Run time (seconds) for the NYPD data set				
base	0.0	50	64	378
(embedding size)			16	290
			32	316
			128	434
			256	636

dropout has a negative impact on NRMS. On the other hand, the embedding size has a negligible effect. It is therefore recommended to not use dropout for imputing continuous attributes: one may choose to train a separate AimNet with just the continuous target attributes. In addition, we can see that the Accuracy and NRMS have no significant change considering that the larger the embedding size is the more running time is required, albeit a sub-linear increase.

A.8 Scaling to Datasets with Large Number of Features

We run a experiments on three synthetically generated datasets with 20,000 rows each, 20% MCAR values, and 20, 50, and 100 features, respectively, for comparing runtime statistics and accuracy. Since the number of categorical features and their cardinalities are non-trivial factors in runtime we make half of them categorical and generate values such that their cardinalities are each approximately $\log(20,000) = 14$. We train each dataset for 5 epochs (which is more than sufficient in our primary experiments for a dataset of this characteristic to achieve a minimal cross-validation loss). As shown in Table 9, the wall-times are 19, 32, and 54 minutes, respectively. As for accuracy on discrete attributes, the performance is even better than the results shown in Figure 8 due to the larger data set size ($|D_{synth}| = 20000 > 5000$).

Table 9: Imputation accuracy (for discrete attributes) and run time on synthetic datasets with large number of features under **MCAR** error injections with $p = 0.2$ missingness on our AimNet model. The means of 10 trials per data set/method with different pseudo-random seeds are reported. Results are after cross-validation on a holdout set.

# of attributes	Accuracy (ACC $\pm$ std)	Run time (minutes)
20	0.86 ± 0.0	19
50	0.85 ± 0.0	32
100	0.86 ± 0.0	54

A.9 Error Percentage Analysis

We additionally vary the percentage of missing values for $p = 0.4, 0.6$ for MCAR-injected errors (in addition to $p = 0.2$ in the main experiments). The results for $p = 0.4$ and $p = 0.6$ are tabulated in Table 10 and Table 11, respectively. Unsurprisingly the accuracy and NRMS of all methods deteriorate with more missing values. We observe however that AimNet maintains its lead compared to the other baselines and in fact marginally outperforms XGB and MF on continuous imputation on the CASP data set when $p = 0.6$. This suggests that not only is AimNet competitive regardless of the missingness proportion but it in fact outpaces the baselines empirically when missingness increases.

A.10 Multi-task v.s. Single Model

We run an experiment to examine the multi-task aspect. We compare multi-task learning (parameters are shared across imputation tasks) against independently learned imputation models where results are shown in Table 12. While there are cases where multi-task models are more accurate, we find that, on average, independent models obtain results of comparable quality. That being said, independent models can be faster (2 - 10 times faster) if trained in parallel. Thus in practice we recommend running the proposed model with single target attributes in parallel if resources permit.

Table 10: Imputation accuracy and NRMS error on the test set under **MCAR** error injections with $p = 0.4$ missingness across our AimNet model and the other baselines.The means of 10 trials per data set/method with different pseudo-random seeds are reported. Results for each method are after cross-validation on a holdout set. The best results for each data set are **bolded** as well as any results that overlap within the confidence interval ($\pm$ 1 standard deviation).

Data Set	Accuracy on discrete attributes (ACC $\pm$ std)						
	AimNet	HCQ	XGB	MIDAS	GAIN	MF	MICE
Tic-Tac-Toc	$\mathbf{0.53 \pm 0.01}$	0.5 ± 0.01	0.52 ± 0.01	0.44 ± 0.02	0.35 ± 0.01	0.5 ± 0.01	0.46 ± 0.01
Hospital	$\mathbf{0.95 \pm 0.0}$	$\mathbf{0.95 \pm 0.0}$	0.91 ± 0.01	0.24 ± 0.01	0.14 ± 0.02	0.94 ± 0.01	0.7 ± 0.01
Mammogram	$\mathbf{0.73 \pm 0.02}$	$\mathbf{0.72 \pm 0.01}$	$\mathbf{0.72 \pm 0.02}$	0.71 ± 0.02	0.35 ± 0.01	0.66 ± 0.02	0.63 ± 0.02
Thoracic	$\mathbf{0.85 \pm 0.01}$	0.84 ± 0.01	0.84 ± 0.01	0.83 ± 0.02	0.52 ± 0.15	$\mathbf{0.85 \pm 0.01}$	0.75 ± 0.03
Contraceptive	$\mathbf{0.63 \pm 0.01}$	$\mathbf{0.63 \pm 0.01}$	0.62 ± 0.01	0.62 ± 0.01	0.43 ± 0.01	0.62 ± 0.01	0.55 ± 0.01
Solar Flare	$\mathbf{0.76 \pm 0.01}$	0.75 ± 0.01	0.75 ± 0.01	0.66 ± 0.01	0.46 ± 0.02	0.74 ± 0.01	0.65 ± 0.01
NYPD	0.87 ± 0.0	0.85 ± 0.0	$\mathbf{0.88 \pm 0.0}$	0.75 ± 0.0	0.15 ± 0.01	$\mathbf{0.88 \pm 0.0}$	0.58 ± 0.0
Credit	$\mathbf{0.73 \pm 0.01}$	0.7 ± 0.01	$\mathbf{0.73 \pm 0.01}$	0.6 ± 0.01	0.39 ± 0.01	$\mathbf{0.73 \pm 0.01}$	0.63 ± 0.01
Australian	$\mathbf{0.7 \pm 0.01}$	0.66 ± 0.01	0.68 ± 0.01	0.6 ± 0.01	0.46 ± 0.01	0.69 ± 0.01	0.59 ± 0.01
Balance	$\mathbf{0.73 \pm 0.03}$	$\mathbf{0.72 \pm 0.03}$	$\mathbf{0.71 \pm 0.03}$	0.64 ± 0.03	0.45 ± 0.05	0.63 ± 0.05	0.64 ± 0.04
Eye EEG	0.67 ± 0.01	0.62 ± 0.01	0.73 ± 0.01	0.55 ± 0.01	0.52 ± 0.03	$\mathbf{0.78 \pm 0.01}$	0.53 ± 0.01
Data Set	NRMS on continuous attributes (NRMS $\pm$ std)						
	AimNet	HCQ	XGB	MIDAS	GAIN	MF	MICE
Hospital	$\mathbf{0.81 \pm 0.04}$	1.1 ± 0.08	0.92 ± 0.07	440.63 ± 61.35	2.26 ± 1.18	0.89 ± 0.04	1.23 ± 0.09
Mammogram	$\mathbf{0.92 \pm 0.02}$	1.02 ± 0.04	0.98 ± 0.05	1.12 ± 0.08	1.05 ± 0.06	1.01 ± 0.03	1.25 ± 0.07
Thoracic	$\mathbf{0.94 \pm 0.01}$	1.09 ± 0.05	1.03 ± 0.11	5.64 ± 7.16	1.23 ± 0.22	0.99 ± 0.06	1.32 ± 0.12
Contraceptive	$\mathbf{0.9 \pm 0.02}$	1.12 ± 0.04	0.94 ± 0.02	1.11 ± 0.02	1.17 ± 0.05	0.99 ± 0.02	1.23 ± 0.06
Solar Flare	$\mathbf{0.93 \pm 0.09}$	$\mathbf{0.98 \pm 0.09}$	$\mathbf{1.0 \pm 0.1}$	10772.7 ± 4057.37	$\mathbf{1.0 \pm 0.09}$	1.04 ± 0.11	1.16 ± 0.07
NYPD	0.32 ± 0.01	0.44 ± 0.13	0.28 ± 0.0	0.69 ± 0.03	3.63 ± 0.17	$\mathbf{0.22 \pm 0.01}$	0.62 ± 0.01
Credit	$\mathbf{0.97 \pm 0.03}$	1.24 ± 0.03	1.26 ± 0.41	1.15 ± 0.07	1.2 ± 0.08	1.12 ± 0.18	1.34 ± 0.11
Australian	$\mathbf{0.96 \pm 0.02}$	1.23 ± 0.03	1.19 ± 0.2	1.14 ± 0.12	1.27 ± 0.16	1.07 ± 0.13	1.6 ± 0.7
Eye EEG	0.48 ± 0.0	0.71 ± 0.03	0.47 ± 0.0	0.91 ± 0.01	1.0 ± 0.28	$\mathbf{0.44 \pm 0.0}$	0.67 ± 0.01
Phase	$\mathbf{0.52 \pm 0.01}$	0.58 ± 0.0	0.53 ± 0.01	0.97 ± 0.01	1.14 ± 0.26	0.58 ± 0.01	0.73 ± 0.01
CASP	0.5 ± 0.01	1.5 ± 0.26	$\mathbf{0.49 \pm 0.01}$	0.88 ± 0.01	0.83 ± 0.09	$\mathbf{0.48 \pm 0.01}$	0.73 ± 0.03

Table 11: Imputation accuracy and NRMS error on the test set under **MCAR** error injections with $p = 0.6$ missingness across our AimNet model and the other baselines. The means of 10 trials per data set/method with different pseudo-random seeds are reported. Results for each method are after cross-validation on a holdout set. The best results for each data set are **bolded** as well as any results that overlap within the confidence interval ($\pm$ 1 standard deviation).

Data Set	Accuracy on discrete attributes (ACC $\pm$ std)						
	AimNet	HCQ	XGB	MIDAS	GAIN	MF	MICE
Tic-Tac-Toc	$\mathbf{0.48 \pm 0.01}$	$\mathbf{0.48 \pm 0.0}$	0.47 ± 0.01	0.43 ± 0.01	0.39 ± 0.01	0.44 ± 0.01	0.4 ± 0.01
Hospital	$\mathbf{0.86 \pm 0.01}$	$\mathbf{0.86 \pm 0.01}$	0.68 ± 0.01	0.24 ± 0.0	0.11 ± 0.01	0.79 ± 0.01	0.37 ± 0.01
Mammogram	$\mathbf{0.69 \pm 0.01}$	$\mathbf{0.69 \pm 0.01}$	$\mathbf{0.68 \pm 0.01}$	$\mathbf{0.68 \pm 0.01}$	0.34 ± 0.02	0.62 ± 0.02	0.58 ± 0.02
Thoracic	$\mathbf{0.85 \pm 0.01}$	0.84 ± 0.01	0.83 ± 0.0	0.84 ± 0.01	0.51 ± 0.13	0.84 ± 0.01	0.72 ± 0.04
Contraceptive	$\mathbf{0.62 \pm 0.01}$	$\mathbf{0.62 \pm 0.01}$	0.61 ± 0.01	0.61 ± 0.01	0.42 ± 0.02	0.6 ± 0.01	0.53 ± 0.01
Solar Flare	$\mathbf{0.72 \pm 0.01}$	$\mathbf{0.72 \pm 0.01}$	0.71 ± 0.01	0.66 ± 0.01	0.45 ± 0.03	0.7 ± 0.01	0.61 ± 0.01
NYPD	0.77 ± 0.0	0.76 ± 0.0	$\mathbf{0.79 \pm 0.0}$	0.67 ± 0.0	0.15 ± 0.0	0.78 ± 0.0	0.45 ± 0.0
Credit	$\mathbf{0.69 \pm 0.01}$	0.66 ± 0.01	0.68 ± 0.01	0.6 ± 0.01	0.38 ± 0.02	0.68 ± 0.01	0.57 ± 0.01
Australian	$\mathbf{0.67 \pm 0.01}$	0.65 ± 0.01	0.65 ± 0.01	0.59 ± 0.01	0.45 ± 0.02	0.65 ± 0.01	0.54 ± 0.02
Balance	$\mathbf{0.67 \pm 0.03}$	$\mathbf{0.64 \pm 0.04}$	0.63 ± 0.03	0.51 ± 0.06	0.46 ± 0.04	0.54 ± 0.03	0.55 ± 0.04
Eye EEG	0.63 ± 0.01	0.6 ± 0.01	0.66 ± 0.01	0.54 ± 0.01	0.52 ± 0.03	$\mathbf{0.67 \pm 0.01}$	0.52 ± 0.01
Data Set	NRMS on continuous attributes (NRMS $\pm$ std)						
	AimNet	HCQ	XGB	MIDAS	GAIN	MF	MICE
Hospital	$\mathbf{0.9 \pm 0.04}$	1.16 ± 0.12	1.01 ± 0.08	139.97 ± 24.15	3.79 ± 0.36	0.95 ± 0.05	1.27 ± 0.09
Mammogram	$\mathbf{0.94 \pm 0.02}$	1.05 ± 0.06	1.0 ± 0.05	1.13 ± 0.06	1.1 ± 0.11	1.01 ± 0.04	1.28 ± 0.08
Thoracic	$\mathbf{0.99 \pm 0.02}$	1.13 ± 0.05	1.17 ± 0.11	3.54 ± 5.28	1.18 ± 0.09	1.07 ± 0.04	1.43 ± 0.2
Contraceptive	$\mathbf{0.94 \pm 0.01}$	1.15 ± 0.04	1.01 ± 0.02	1.12 ± 0.02	1.31 ± 0.14	1.14 ± 0.04	1.29 ± 0.05
Solar Flare	$\mathbf{0.98 \pm 0.06}$	$\mathbf{1.01 \pm 0.06}$	1.05 ± 0.09	4454.64 ± 1610.59	1.07 ± 0.17	1.13 ± 0.14	1.24 ± 0.19
NYPD	0.56 ± 0.0	0.58 ± 0.04	0.45 ± 0.0	0.8 ± 0.01	3.51 ± 0.14	$\mathbf{0.42 \pm 0.01}$	0.98 ± 0.0
Credit	$\mathbf{0.99 \pm 0.01}$	1.33 ± 0.19	1.23 ± 0.25	1.14 ± 0.11	1.24 ± 0.11	1.13 ± 0.13	1.43 ± 0.26
Australian	$\mathbf{0.98 \pm 0.01}$	1.37 ± 0.26	1.29 ± 0.42	1.1 ± 0.04	1.25 ± 0.12	1.13 ± 0.19	1.38 ± 0.09
Eye EEG	0.59 ± 0.0	0.82 ± 0.04	$\mathbf{0.57 \pm 0.0}$	0.97 ± 0.01	1.61 ± 0.24	$\mathbf{0.57 \pm 0.0}$	0.79 ± 0.0
Phase	$\mathbf{0.64 \pm 0.0}$	0.71 ± 0.01	0.65 ± 0.0	1.0 ± 0.0	1.45 ± 0.51	0.71 ± 0.01	0.91 ± 0.01
CASP	$\mathbf{0.58 \pm 0.01}$	2.06 ± 0.51	0.59 ± 0.01	0.94 ± 0.01	1.2 ± 0.22	0.62 ± 0.0	0.88 ± 0.03

Table 12: Imputation accuracy and NRMS error on the test set under **MCAR** injections with $p = 0.2$ missingness with individually-trained AimNet models for each target attribute(Single) and shared context parameters across target attributes (Multi-Task). The means of 10 trials per data set-method with different pseudo-random seeds are reported. Results for each method are after cross-validation on a hold-out set. The best results for each data set are **bolded** as well as any results that overlap within the confidence interval (± 1 standard deviation).

Data Set	Accuracy on discrete attributes (ACC ± std)	
	Single	Multi-Task
Tic-Tac-Toc	$\mathbf{0.61 \pm 0.01}$	$\mathbf{0.61 \pm 0.01}$
Hospital	$\mathbf{0.99 \pm 0.0}$	$\mathbf{0.99 \pm 0.0}$
Mammogram	$\mathbf{0.75 \pm 0.01}$	$\mathbf{0.75 \pm 0.01}$
Thoracic	$\mathbf{0.86 \pm 0.01}$	$\mathbf{0.86 \pm 0.01}$
Contraceptive	$\mathbf{0.65 \pm 0.01}$	$\mathbf{0.65 \pm 0.01}$
Solar Flare	$\mathbf{0.78 \pm 0.02}$	$\mathbf{0.78 \pm 0.01}$
NYPD	$\mathbf{0.92 \pm 0.0}$	$\mathbf{0.92 \pm 0.0}$
Credit	$\mathbf{0.76 \pm 0.01}$	$\mathbf{0.76 \pm 0.01}$
Australian	$\mathbf{0.72 \pm 0.02}$	$\mathbf{0.72 \pm 0.01}$
Chicago	$\mathbf{0.73 \pm 0.01}$	0.7 ± 0.03
Balance	$\mathbf{0.79 \pm 0.04}$	$\mathbf{0.79 \pm 0.04}$
Eye EEG	$\mathbf{0.71 \pm 0.01}$	0.69 ± 0.01
Data Set	**NRMS on continuous attributes (NRMS ± std)**	
	Single	Multi-Task
Hospital	$\mathbf{0.72 \pm 0.06}$	$\mathbf{0.77 \pm 0.06}$
Mammogram	$\mathbf{0.91 \pm 0.04}$	$\mathbf{0.93 \pm 0.03}$
Thoracic	$\mathbf{1.1 \pm 0.41}$	$\mathbf{1.3 \pm 0.83}$
Contraceptive	$\mathbf{0.84 \pm 0.02}$	$\mathbf{0.84 \pm 0.02}$
Solar Flare	$\mathbf{0.94 \pm 0.15}$	$\mathbf{0.87 \pm 0.16}$
NYPD	$\mathbf{0.15 \pm 0.01}$	$\mathbf{0.15 \pm 0.01}$
Credit	$\mathbf{0.94 \pm 0.03}$	$\mathbf{0.94 \pm 0.03}$
Australian	$\mathbf{0.94 \pm 0.03}$	$\mathbf{0.93 \pm 0.02}$
Eye EEG	$\mathbf{0.4 \pm 0.0}$	0.44 ± 0.0
Phase	$\mathbf{0.45 \pm 0.01}$	$\mathbf{0.45 \pm 0.01}$
CASP	$\mathbf{0.45 \pm 0.02}$	0.48 ± 0.01
Data Set	**Run time (seconds)**	
	Single	Multi-Task
Tic-Tac-Toc	**9**	38
Hospital	**18**	148
Mammogram	**9**	39
Thoracic	**11**	69
Contraceptive	**15**	102
Solar Flare	**15**	131
NYPD	**378**	6320
Credit	**15**	198
Australian	**16**	145
Chicago	**3180**	462756
Balance	**11**	12
Eye EEG	**188**	5306
Phase	**66**	250
CASP	**648**	5800