On the Limitations of Representing Functions on Sets

Edward Wagstaff [*1] **Fabian B. Fuchs** [*1] **Martin Engelcke** [*1] **Ingmar Posner** [1] **Michael Osborne** [1]

Abstract

Recent work on the representation of functions on sets has considered the use of summation in a latent space to enforce permutation invariance. In particular, it has been conjectured that the dimension of this latent space may remain fixed as the cardinality of the sets under consideration increases. However, we demonstrate that the analysis leading to this conjecture requires mappings which are highly discontinuous and argue that this is only of limited practical use. Motivated by this observation, we prove that an implementation of this model via continuous mappings (as provided by e.g. neural networks or Gaussian processes) actually imposes a constraint on the dimensionality of the latent space. Practical universal function representation for set inputs can only be achieved with a latent dimension at least the size of the maximum number of input elements.

1. Introduction

Machine learning models have had great success in taking advantage of structure in their input spaces: recurrent neural networks are popular models for sequential data (Sutskever et al., 2014) and convolutional neural networks are the state-of-the-art for many image-based problems (He et al., 2016). Recently, however, models for unstructured inputs in the form of sets have rapidly gained attention (Ravanbakhsh et al., 2016; Zaheer et al., 2017; Qi et al., 2017a; Lee et al., 2018; Murphy et al., 2018; Korshunova et al., 2018).

Importantly, a range of machine learning problems can naturally be formulated in terms of sets; e.g. parsing a scene composed of a set of objects (Eslami et al., 2016; Kosiorek et al., 2018), making predictions from a set of points forming a 3D point cloud (Qi et al., 2017a;b), or training a set of agents in reinforcement learning (Sunehag et al., 2017).

[*]Equal contribution [1]Department of Engineering Science, University of Oxford, Oxford, United Kingdom. Correspondence to: <{ed, fabian, martin}@robots.ox.ac.uk>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

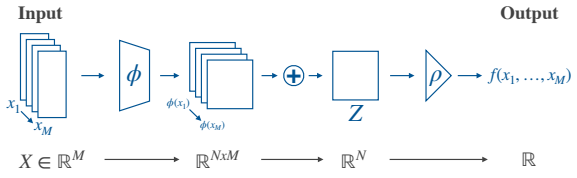


Figure 1: Illustration of the model structure proposed in several works (Zaheer et al., 2017; Qi et al., 2017a) for representing permutation-invariant functions. The sum operation enforces permutation invariance for the model as a whole. ϕ and ρ can be implemented by e.g. neural networks.

Furthermore, attention-based models perform a weighted summation of a set of features (Vaswani et al., 2017; Lee et al., 2018). Hence, understanding the mathematical properties of set-based models is valuable both in terms of set-structured applications as well as better understanding the capabilities and limitations of attention-based models.

Many popular machine learning models, including neural networks and Gaussian processes, are fundamentally based on vector inputs[1] rather than set inputs. In order to adapt these models for use with sets, we must enforce the property of *permutation invariance*, i.e. the output of the model must not change if the inputs are reordered. Multiple authors, including Ravanbakhsh et al. (2016), Zaheer et al. (2017) and Qi et al. (2017a), have considered enforcing this property using a technique which we term *sum-decomposition*, illustrated in Figure 1. Mathematically speaking, we say that a function f defined on sets of size M is *sum-decomposable via* Z if there are functions $\phi : \mathbb{R} \to Z$ and $\rho : Z \to \mathbb{R}$ such that[2]

$$f(X) = \rho\big(\Sigma_{x \in X} \phi(x)\big) \tag{1}$$

We refer to Z here as the *latent space*. Since summation is permutation-invariant, a sum-decomposition is also permutation-invariant. Ravanbakhsh et al. (2016), Zaheer et al. (2017) and Qi et al. (2017b) have also considered the idea of enforcing permutation invariance using other operations, e.g. $\texttt{max}(\cdot)$. In this paper we concentrate on a detailed analysis of sum-decomposition, but some of the limitations we discuss also apply when $\texttt{max}(\cdot)$ is used instead of summation.

[1]Or inputs of higher rank, i.e. matrices and tensors.

[2]We use $\mathbb{R}$ here for brevity – see Definition 2.2 for the fully general definition.

Our main contributions can be summarised as follows.

1. Recent proofs, e.g. in Zaheer et al. (2017), consider functions on countable domains. We explain why considering countable domains can lead to results of limited practical value (i.e. cannot be implemented with a neural network), and why considering continuity on uncountable domains such as $\mathbb{R}$ is necessary. With reference to neural networks, we ground this discussion in the universal approximation theorem, which relies on continuity on uncountable domains $[0,1]^M$.
2. In contrast to previous work (Zaheer et al., 2017; Qi et al., 2017a), which considers sufficient conditions for universal function representation, we establish a *necessary* condition for a sum-decomposition-based model to be capable of universal function representation. Additionally, we provide weaker sufficient conditions which imply a stronger version of universality. Specifically, we show that the dimension of the latent space being at least as large as the maximum number of input elements is both necessary and sufficient for universal function representation.

While primarily targeted at neural networks, these results hold for any implementation of sum-decomposition, e.g. using Gaussian processes, as long as it provides universal function approximation for continuous functions. Proofs of all novel results are available in Appendix B.

2. Preliminaries

In this section we recount the theorems and proofs on sum-decomposition from Zaheer et al. (2017). We begin by introducing important definitions and the notation used throughout our work. Note that we focus on permutation-invariant functions and do not discuss permutation equivariance which is also considered in Zaheer et al. (2017).

2.1. Definitions

Definition 2.1. A function $f(\mathbf{x})$ is *permutation-invariant* if $f(x_1,\dots,x_M) = f\big(x_{\pi(1)},\dots,x_{\pi(M)}\big)$ for all π.

Definition 2.2. We say that a function f is *sum-decomposable* if there are functions ρ and ϕ such that

$$f(X) = \rho\big(\Sigma_{x\in X}\phi(x)\big).$$

In this case, we say that (ρ,ϕ) is a *sum-decomposition* of f.

Given a latent space Z, we say that f is *sum-decomposable via* Z when this expression holds for some ϕ whose codomain is Z, i.e. $\phi : \mathfrak{X} \to Z$.

We say that f is *continuously sum-decomposable* when this expression holds for some continuous functions ρ and ϕ.

We will also consider sum-decomposability where the inputs to f are vectors rather than sets - in this context, the sum is over the elements of the input vector.

Definition 2.3. A set $\mathfrak{X}$ is *countable* if its number of elements, i.e. the cardinality, is smaller or equal to the number of elements in $\mathbb{N}$. This includes both finite and countably infinite sets; e.g. $\mathbb{N}$, $\mathbb{Q}$, and subsets thereof.

Definition 2.4. A set $\mathfrak{X}$ is *uncountable* if its number of elements is greater than the number of elements in $\mathbb{N}$, e.g. $\mathbb{R}$ and certain subsets thereof.

Notation 2.5. Denote the power set of a set $\mathfrak{X}$ by $2^{\mathfrak{X}}$.

Notation 2.6. Denote the set of *finite* subsets of a set $\mathfrak{X}$ by $\mathfrak{X}^{\mathcal{F}}$.

Notation 2.7. Denote the set of subsets of a set $\mathfrak{X}$ containing at most M elements by $\mathfrak{X}^{\leq M}$.

Remark. Throughout, we discuss expressions of the form $\Phi(X) = \Sigma_{x\in X}\phi(x)$, where X is a set. Note that care must be taken in interpreting this expression when X is not finite – we discuss this issue fully in Appendix A.1.

2.2. Background Theorems

Zaheer et al. (2017) consider the two cases where X is a subset of, or drawn from, a *countable* and an *uncountable* universe $\mathfrak{X}$. We now outline the theorems and proofs relating to these two cases.

Theorem 2.8 (Countable case)**.** *Let $f : 2^{\mathfrak{X}} \to \mathbb{R}$ where $\mathfrak{X}$ is countable. Then f is permutation-invariant if and only if it is sum-decomposable via $\mathbb{R}$.*

Proof. Since $\mathfrak{X}$ is countable, each $x \in \mathfrak{X}$ can be mapped to a unique element in $\mathbb{N}$ by a function $c(x) : \mathfrak{X} \to \mathbb{N}$. Let $\Phi(X) = \sum_{x\in X}\phi(x)$. If we can choose ϕ so that Φ is injective, then we can set $\rho = f \circ \Phi^{-1}$, giving

$$f = \rho \circ \Phi$$
$$f(X) = \rho\big(\Sigma_{x\in X}\phi(x)\big)$$

i.e. f is sum-decomposable via $\mathbb{R}$.

Now consider $\phi(x) = 4^{-c(x)}$. Under this mapping, each $X \subset \mathfrak{X}$ corresponds to a unique real number expressed in base 4. Therefore Φ is injective, and the conclusion follows. □

Remark. This construction works for any set size M, and even for sets of infinite size. However, it assumes that X is a set with no repeated elements, i.e. multisets are not supported. Specifically, the construction will fail with multisets because Φ fails to be injective if its domain includes multisets. In Appendix A.3, we extend Theorem 2.8 to also support multisets, with the restriction that infinite sets are no longer supported.

Theorem 2.9 (Uncountable case)**.** *Let $M \in \mathbb{N}$, and let $f : [0,1]^M \to \mathbb{R}$ be a continuous function. Then f is permutation-invariant if and only if it is continuously sum-decomposable via $\mathbb{R}^{M+1}$.*

The proof by Zaheer et al. (2017) of Theorem 2.9 is more involved than for Theorem 2.8. We do not include it here in full detail, but briefly summarise below.

1. Show that the mapping $\Phi : [0,1]^M \to \mathbb{R}^{M+1}$ defined by $\Phi_q(\mathbf{x}) = \sum_{m=1}^{M}(x_m)^q$ for $q = 0, \dots, M$ is injective and continuous.[3]
2. Show that Φ has a continuous inverse.
3. Define $\rho : \mathbb{R}^{M+1} \to \mathbb{R}$ by $\rho = f \circ \Phi^{-1}$.
4. Define $\phi(x) : \mathbb{R} \to \mathbb{R}^{M+1}$ by $\phi_q(x) = x^q$.
5. Note that, by definition of ρ and ϕ, (ρ, ϕ) is a continuous sum-decomposition of f via $\mathbb{R}^{M+1}$. □

Remark. Zaheer et al. (2017) conjecture that any continuous permutation-invariant function f on $2^{[0,1]}$, the power set of $[0,1]$, is continuously sum-decomposable. In Section 3, we show that this is not possible, and in Section 4 we show that even if the domain of f is restricted to $[0,1]^{\leq \mathcal{F}}$, the finite subsets of $[0,1]$, then $N \geq M$ is a *necessary condition* for arbitrary functions f to be continuously sum-decomposable. Additionally, we prove that $N = M$ is a *sufficient condition* – implying together with the above that it is not possible to do better than this.

3. The Importance of Continuity

In this section, we argue that continuity is essential to discussions of function representation, that it has been neglected in prior work on permutation-invariant functions, and that this neglect has implications for the strength and generality of existing results.

Intuitively speaking a function is continuous if, at every point in the domain, the variation of the output can be made arbitrarily small by limiting the variation in the input. Continuity is the reason that, for instance, working to machine precision usually produces sensible results. Truncating to machine precision alters the input to a function slightly, but continuity ensures that the change in output is also slight.

In Zaheer et al. (2017), the authors demonstrate that when $\mathfrak{X}$ is a countable set, e.g. the rational numbers, any function $f : 2^{\mathfrak{X}} \to \mathbb{R}$ is sum-decomposable via $\mathbb{R}$. This is taken as a hopeful indication that sum-decomposability may extend to uncountable domains, e.g. $\mathfrak{X} = \mathbb{R}$. Extending to the uncountable case may appear, at first glance, to be a mere formality – we are, after all, ultimately interested in implementing algorithms on finite hardware. Nevertheless, it is not true that a theoretical result for a countably infinite domain must be strong enough for practical purposes. In fact, considering functions on uncountably infinite domains such as $\mathbb{R}^N$ is of real importance.

[3] In the original proof, Φ is denoted E.

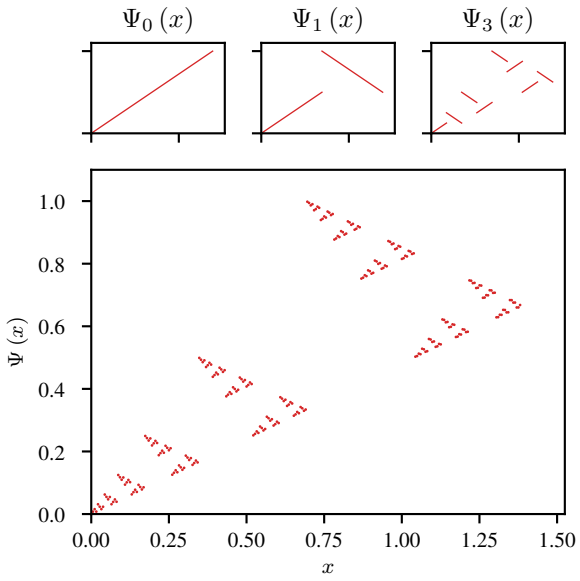


Figure 2: The function Ψ shown here is continuous at every rational point in $[0, \ln 4]$. Intuitively, this is because all jumps occur at irrational values, namely at certain fractions of $\ln 4$. It defies our intuitions for what continuity should mean, and illustrates the fact that continuity on $\mathbb{Q}$ is a much weaker property than continuity on $\mathbb{R}$. The latter property is required to satisfy the universal approximation theorem for neural networks. Ψ is defined and discussed in Appendix C.

Turning specifically to neural networks, the universal approximation theorem says that any *continuous* function can be approximated by a neural network, but not that *any* function can be approximated by a neural network (Cybenko, 1989). A similar statement is true for other approximators, such as some Gaussian processes (Rasmussen & Williams, 2006). The notion of continuity required here is specifically that of continuity on compact subsets of $\mathbb{R}^N$.

Crucially, if we wish to work mathematically with continuity in a way that closely matches our intuitions, we *must* consider uncountable domains. To illustrate this point, consider the rational numbers $\mathbb{Q}$. $\mathbb{Q}$ is dense in $\mathbb{R}$, and it is tempting to think that $\mathbb{Q}$ is therefore "all we need". However, a theoretical guarantee of continuity on $\mathbb{Q}$ is weak, and does not imply continuity on $\mathbb{R}$. The universal approximation theorem for neural networks relies on continuity on $\mathbb{R}$, and we cannot usefully take continuity on $\mathbb{Q}$ as a proxy for this

property. Figure 2 shows a function which is continuous on $\mathbb{Q}$, and illustrates that a continuous function on $\mathbb{Q}$ may not extend continuously to $\mathbb{R}$. This figure also illustrates that continuity on $\mathbb{Q}$ defies our intuitions about what continuity should mean, and is too weak for the universal approximation theorem for neural networks. We require the stronger notion of continuity on $\mathbb{R}$.

In light of the above, it is clear that continuity is a key property for function representation, and also that there is a crucially important difference between countable and uncountable domains. This raises two problems for Theorem 2.8. First, the theorem does not consider the continuity of the sum-decomposition when the domain $\mathfrak{X}$ has some non-trivial topological structure (e.g. $\mathfrak{X} = \mathbb{Q}$). Second, we still care about continuity on $\mathbb{R}$, and there is no guarantee that this is possible given continuity on $\mathbb{Q}$.

In fact, the continuity issue cannot be overcome – we can demonstrate that in general the sum-decomposition of Theorem 2.8, which goes via $\mathbb{R}$, cannot be made continuous for $\mathfrak{X} = \mathbb{Q}$:

Theorem 3.1. *There exist functions $f : 2^{\mathbb{Q}} \to \mathbb{R}$ such that, whenever (ρ, ϕ) is a sum-decomposition of f via $\mathbb{R}$, ϕ is discontinuous at every point $q \in \mathbb{Q}$.*

We can actually say something more general than the above. Our proof can easily be adapted to demonstrate that if f is injective, or if we want a fixed ϕ to suffice for any f, then ϕ can only be continuous at isolated points of the underlying set $\mathfrak{X}$, regardless of whether $\mathfrak{X} = \mathbb{Q}$. I.e., it is not specifically due to the structure of $\mathbb{Q}$ that continuous sum-decomposability fails. In fact, it fails whenever we have a non-trivial topological structure. For functions which we want to model using a neural network, this is worrying.

It is not possible to represent an everywhere-discontinuous ϕ with a neural network. We therefore view Theorem 2.8 as being of limited practical relevance and as not providing a reliable intuition for what should be possible in the uncountable case. We do however see this result as mathematically interesting, and have obtained the following result extending it to the case where the domain $\mathfrak{X}$ is uncountable. This result is slightly weaker than the countable case, in that the domain of f can contain arbitrarily large finite sets, but not infinite sets.

Theorem 3.2. *Let $f : \mathbb{R}^{\mathcal{F}} \to \mathbb{R}$. Then f is sum-decomposable via $\mathbb{R}$.*

Once again, the sum-decomposition is highly discontinuous. The limitation that f is not defined on infinite sets cannot be overcome:

Theorem 3.3. *If $\mathfrak{X}$ is uncountable, then there exist functions $f : 2^{\mathfrak{X}} \to \mathbb{R}$ which are not sum-decomposable. Note that this holds even if the sum-decomposition (ρ, ϕ) is allowed to be discontinuous.*

To summarise, we show why considering countable domains can lead to results of limited practical value and why considering continuity on uncountable domains is necessary. We point out that some of the previous work is therefore of limited practical relevance, but regard it as mathematically interesting. In this vein, we extend the analysis of sum-decomposability when continuity is not required.

4. Practical Function Representation

Having established the necessity of considering continuity on $\mathbb{R}$, we now explore the implications for sum-decomposability of permutation-invariant functions. These considerations lead to concrete recommendations for model design and provide theoretical support for elements of current practice in the area. Specifically, we present three theorems whose implications can be summarised as follows.

1. A latent dimensionality of M is *sufficient* for representing all continuous permutation-invariant functions on sets of size $\leq M$.
2. To guarantee that all continuous permutation-invariant functions can be represented for sets of size $\leq M$, a latent dimensionality of at least M is *necessary*.

The key result which is the basis of the second statement and which underpins this discussion is as follows.

Theorem 4.1. *Let $M > N \in \mathbb{N}$. Then there exist permutation invariant continuous functions $f : \mathbb{R}^M \to \mathbb{R}$ which are* **not** *continuously sum-decomposable via $\mathbb{R}^N$.*

Restated in more practical terms, this implies that for a sum-decomposition-based model to be capable of representing *arbitrary* continuous functions on sets of size M, the latent space in which the summation happens must be chosen to have dimension at least M. A similar statement is true for the analogous concept of *max-decomposition* – details are available in Appendix B.6.

To prove this theorem, we first need to state and prove the following lemma.

Lemma 4.2. *Let $M, N \in \mathbb{N}$, and suppose $\phi : \mathbb{R} \to \mathbb{R}^N$, $\rho : \mathbb{R}^N \to \mathbb{R}$ are functions such that:*

$$max(X) = \rho\left(\Sigma_{x \in X} \phi(x)\right) \tag{2}$$

Now let $\Phi(X) = \Sigma_{x \in X} \phi(x)$, and write Φ_M for the restriction of Φ to sets of size M.

Then Φ_M is injective for all M.

Proof. We proceed by induction. The base case $M = 1$ is clear.

Now let $M \in \mathbb{N}$, and suppose that Φ_{M-1} is injective. Now suppose there are sets X, Y such that $\Phi_M(X) = \Phi_M(Y)$. First note that, by (2), we must have:

$$\max(X) = \max(Y) \tag{3}$$

So now write:

$$X = \{x_{\max}\} \cup X_{\text{rem}} \; ; \; Y = \{y_{\max}\} \cup Y_{\text{rem}} \tag{4}$$

where $x_{\max} = \max(X)$, and similarly for $y_{\max}$.

But now:

$$\begin{aligned} \Phi_M(X) &= \Phi_{M-1}(X_{\text{rem}}) + \phi(x_{\max}) \\ &= \Phi_{M-1}(Y_{\text{rem}}) + \phi(y_{\max}) \\ &= \Phi_M(Y) \end{aligned}$$

From the central equality, and (3), we have:

$$\Phi_{M-1}(X_{\text{rem}}) = \Phi_{M-1}(Y_{\text{rem}})$$

Now by injectivity of Φ_{M-1}, we have $X_{\text{rem}} = Y_{\text{rem}}$. Combining this with (3) and (4), we must have $X = Y$, and so Φ_M is injective. □

Equipped with this lemma, we can now prove Theorem 4.1.

Proof. We proceed by contradiction. Suppose that functions ϕ and ρ exist satisfying (2). Define $\Phi_M : \mathbb{R}^M \to \mathbb{R}^N$ by:

$$\Phi_M(\mathbf{x}) = \Sigma_{i=1}^M \phi(x_i)$$

Denote the set of all $x \in \mathbb{R}^M$ with $x_1 < x_2 < ... < x_M$ by $\mathbb{R}^M_{\text{ord}}$, and let Φ^{ord}_M be the restriction of Φ_M to $\mathbb{R}^M_{\text{ord}}$. Since Φ^{ord}_M is a sum of continuous functions, it is also continuous, and by Lemma 4.2, it is injective.

Now note that $\mathbb{R}^M_{\text{ord}}$ is a convex open subset of $\mathbb{R}^M$, and is therefore homeomorphic to $\mathbb{R}^M$. Therefore, our continuous injective Φ^{ord}_M can be used to construct a continuous injection from $\mathbb{R}^M$ to $\mathbb{R}^N$. But it is well known that no such continuous injection exists when $M > N$. Therefore our decomposition (2) cannot exist. □

It is crucial to note that functions f for which a lower-dimensional sum-decomposition does not exist need not be "badly-behaved" or difficult to specify. The limitation extends to functions of genuine interest. For our proof, we have specifically demonstrated that even $\texttt{max}(X)$ is not continuously sum-decomposable when $N < M$.

From Theorem 2.9, we also know that for a fixed input set size M, any continuous permutation-invariant function is continuously sum-decomposable via $\mathbb{R}^{M+1}$. It is, however, possible to adapt the construction of Zaheer et al. (2017) to strengthen the result in two ways. Firstly, we can perform the sum-decomposition via $\mathbb{R}^M$:

Theorem 4.3 (Fixed set size)**.** *Let $f : \mathbb{R}^M \to \mathbb{R}$ be continuous. Then f is permutation-invariant if and only if it is continuously sum-decomposable via $\mathbb{R}^M$.*

Secondly, we can deal with variable set sizes $\leq M$:

Theorem 4.4 (Variable set size)**.** *Let $f : \mathbb{R}^{\leq M} \to \mathbb{R}$ be continuous. Then f is permutation-invariant if and only if it is continuously sum-decomposable via $\mathbb{R}^M$.*

Note that we must take some care over the notion of continuity in this theorem – see Appendix A.2.

4.1. Discussion

Theorem 4.1 does not imply *all* functions require $N = M$. Some functions, such as the mean, can be represented in a lower dimensional space. The statement rather says that if we do not want to impose any limitations on the complexity of the function, the latent space needs to have dimensionality at least M.

Theorem 4.4 suggests that sum-decomposition via a latent space with dimension $N = M$ should suffice to model any function. Neural network models in the recent literature, however, deviate from these guidelines in several ways, indicating a disconnect between theory and practice. For example, the models in Zaheer et al. (2017) and Qi et al. (2017a) are considerably more complex than Equation (1), e.g. they apply several permutation-equivariant layers to the input before a permutation-invariant layer.

In light of Theorem 4.1, this disconnect becomes less surprising. We have shown that, for a target function of sufficient complexity, $N = M$ is the bare minimum required for the model to be capable of representing the target function. Achieving this would rely on the parameterisation of ϕ and ρ being flexible enough and on the availability of a suitable optimisation method. In practice, we should not be surprised that more than the bare minimum capacity in our model is required for good performance. Even with $N > M$, the model might not converge to the desired solution. At the same time, when we are dealing with real datasets, the training data may contain noise and redundant information, e.g. in the form of correlations between elements in the input, inducing functions of limited complexity that may in fact be representable with $N < M$.

4.2. Illustrative Example

We now use a toy example to illustrate some practical implications of our results. Based on Theorem 4.1, we expect the number of input elements M to have an influence on the required latent dimension N, and in particular, we expect that the required latent dimension may increase without bound.

We train a neural network with the architecture presented in Figure 1 to predict the median of a set of values. We choose the median as a function because it is relatively simple but cannot be trivially represented via a sum in a fixed-dimensional latent space, in contrast to e.g. the mean, which is sum-decomposable via $\mathbb{R}$.[4] ϕ and ρ are parameterised by multi-layer perceptrons (MLPs). The input sets are randomly drawn from either a uniform, a Gaussian, or a Gamma distribution.

We vary the latent dimension N and the input set size M to investigate the link between these two variables and the predictive performance. The MLPs parameterising ϕ and ρ are given comparatively many layers and hidden units, relative to the simplicity of the task, to ensure that the latent dimension is the bottleneck. Further details are described in Appendix D.

Figure 3(a) shows the RMSE depending on the latent dimension for different input sizes. We make three observations.

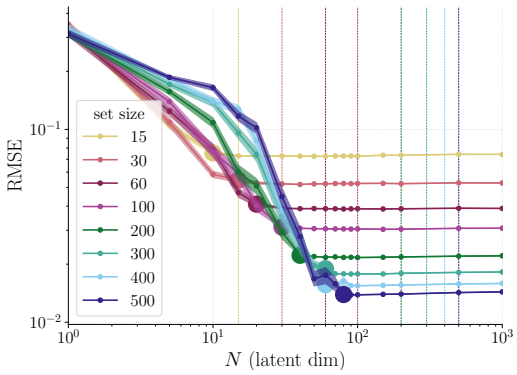
1. For each set size, the error decreases monotonically with the dimension of the latent space.

2. Beyond a certain point, increasing the dimension of the latent space does not further reduce the error. We denote this the "critical point".

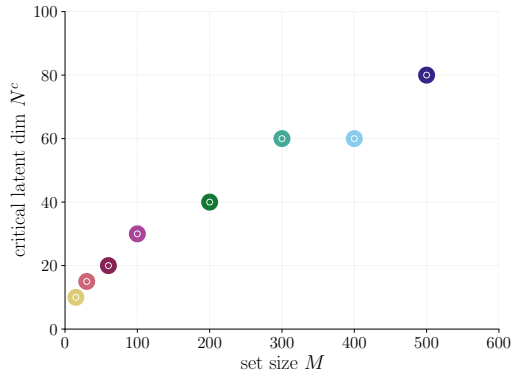
3. As the set size increases, so does the latent dimension at the critical point.

Figure 3(b) shows the critical points as a function of the input size, indicating a roughly linear relationship between the two. Note that the critical points occur at $N < M$. This can be explained by the fact that the models do not learn an algorithmic solution for computing the median, but rather to estimate it given samples drawn from the specific input distribution seen during training. Furthermore, estimating the median of a distribution, like other functions, renders some information in the input redundant. Therefore, the mapping from input to latent space does not need to be injective, allowing a model to solve the task with a smaller value of N.

[4]The construction for $Z = \mathbb{R}$ is not entirely trivial for variable set size, but going via $Z = \mathbb{R}^2$ is straightforward.



(a) Test performance on median estimation depending on latent dimension. Different colours depict different set sizes. Each data point is averaged over 500 runs with different seeds. Shaded areas indicate confidence intervals. Coloured dashed lines indicate $N = M$.



(b) Extracted 'critical points' from above graph. The coloured data points depict minimum latent dimension for optimal performance (RMSE less than 10% above minimum value for this set size) for different set sizes.

Figure 3: Illustrative toy example: a neural network is trained to predict the median of an unordered set.

5. Related Work

Much of the recent work on deep learning with unordered sets follows the paradigm discussed in (Ravanbakhsh et al., 2016), Zaheer et al. (2017), and Qi et al. (2017a) which leverage the structure illustrated in Figure 1. Zaheer et al. (2017) provide an in-depth theoretical analysis which is discussed in detail in Section 2. Qi et al. (2017a) also derive a sufficiency condition for universal function approximation. In their proof, however, they set the latent dimension N to $\lceil 1/\delta_\epsilon \rceil$ where δ_ϵ depends on the error tolerance for how closely the target function has to be approximated. As a result, the latent dimension N goes to infinity for exact representation. In similar vain, Herzig et al. (2018) consider permutation-invariant functions on graphs.

A key application domain of set-based methods is the processing of point clouds, as the constituent points do not

have an intrinsic ordering. The work by Qi et al. (2017a) on 3D point clouds, one of the first to use a permutation-invariant neural networks, is extended in Qi et al. (2017b) by sampling and grouping points in a hierarchical fashion to model the interaction between nearby points in the input space more explicitly. Qi et al. (2018) combine RGB and lidar data for object detection by using image detectors to generate bounding box proposals which are then further processed by a set-based model. Achlioptas et al. (2018) and Yi et al. (2018) show that set-based models can also be used to learn generative models of point clouds.

Vinyals et al. (2015) suggest that even though recurrent networks are universal approximators, the ordering of the input is crucial for good performance. Hence, they propose model that relies on attention to achieve permutation invariance in order to solve a sorting task. In general, it is worth noting that there exists a connection between the model in Zaheer et al. (2017) and recent attention-based models such as the one proposed in Vaswani et al. (2017). In this case, the aggregation layer includes a weighting parameter which is computed based on a key-query system which is also permutation invariant. Since the value of the weighting parameters could be learned to be 1.0, it is trivial to show that such an attention algorithm is also in principle able to approximate any permutation-invariant function, of course depending on the remaining parts of the architecture. Inspired by inducing point methods, Set Transformer (Lee et al., 2018) propose a computationally more efficient attention-module and demonstrate better performance on a range of set-based tasks. While stacking several of attention-modules can capture higher order dependencies, a more general treatment of this is offered by permutation-invariant, learnable Janossy Pooling (Murphy et al., 2018).

Similar to the methods considered here, Neural Processes (Garnelo et al., 2018b) and Conditional Neural Processes (Garnelo et al., 2018a) also rely on aggregation via summation in order to infer a distribution from a set of data points. Kim et al. (2019) add an attention mechanism to neural processes to improve empirical performance. Generative Query Networks (Eslami et al., 2018; Kumar et al., 2018) can be regarded as an instantiation of neural processes to learn useful representations of 3D scenes from multiple 2D views. Yang et al. (2018) also aggregate information from multiple views to compute representations of 3D objects.

Bloem-Reddy & Teh (2019) and Korshunova et al. (2018) consider exchangeable sequences – sequences consisting of random variables with a joint likelihood which is invariant under permutations. Bloem-Reddy & Teh (2019) provide a theorem that describes distribution-invariant models. Korshunova et al. (2018) use RealNVP (Dinh et al., 2016) as a bijective function which sequentially computes the parameters of a Student-t process.

6. Conclusions

This work derives theoretical limitations on the representation of *arbitrary* functions on sets via a finite latent space. We demonstrate why continuity requires statements on uncountable domains, as opposed to countable domains, to ensure the practical usefulness of those statements. Under this constraint, we prove that a latent space whose dimension is at least as large as the maximum input set size is both sufficient and necessary to achieve *universal* function representation. The models covered in this analysis are popular for a range of practical applications and can be implemented e.g. by neural networks or Gaussian processes. In future work, we would like to investigate the effect of constructing models with both permutation-equivariant and permutation-invariant modules on the required dimension of the latent space. Examining the implications of using self-attention, e.g. as in Lee et al. (2018), would be of similar interest.

Acknowledgements

This research was funded by the EPSRC AIMS Centre for Doctoral Training at the University of Oxford, an EPSRC DTA studentship, a Google studentship, and an EPSRC Programme Grant (EP/M019918/1). The authors acknowledge use of Hartree Centre resources in this work. The STFC Hartree Centre is a research collaboratory in association with IBM providing High Performance Computing platforms funded by the UK's investment in e-Infrastructure. The authors thank Sudhanshu Kasewa and Olga Isupova for proof reading a draft of the paper.

References

Achlioptas, P., Diamanti, O., Mitliagkas, I., and Guibas, L. Learning Representations and Generative Models for 3D Point Clouds. *International Conference on Machine Learning*, 2018.

Bloem-Reddy, B. and Teh, Y. W. Probabilistic Symmetry and Invariant Neural Networks. *arXiv preprint arXiv:1901.06082*, 2019.

Cybenko, G. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, 2(4):303–314, 1989.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using Real NVP. *arXiv preprint arXiv:1605.08803*, 2016.

Eslami, S. M. A., Heess, N., Weber, T., Tassa, Y., Szepesvari, D., Kavukcuoglu, K., and Hinton, G. E. Attend, Infer, Repeat: Fast Scene Understanding with Generative Models. *Advances in Neural Information Processing Systems*, 2016.

Eslami, S. M. A., Rezende, D. J., Besse, F., Viola, F., Morcos, A. S., Garnelo, M., Ruderman, A., Rusu, A. A., Danihelka, I., Gregor, K., Reichert, D. P., Buesing, L., Weber, T., Vinyals, O., Rosenbaum, D., Rabinowitz, N. C., King, H., Hillier, C., Botvinick, M. M., Wierstra, D., Kavukcuoglu, K., and Hassabis, D. Neural scene representation and rendering. *Science*, 360:1204–1210, 2018.

Garnelo, M., Rosenbaum, D., Maddison, C. J., Ramalho, T., Saxton, D., Shanahan, M., Teh, Y. W., Rezende, D. J., and Eslami, S. M. A. Conditional Neural Processes. *International Conference on Machine Learning*, 2018a.

Garnelo, M., Schwarz, J., Rosenbaum, D., Viola, F., Rezende, D. J., Eslami, S. M. A., and Teh, Y. W. Neural Processes. *International Conference on Machine Learning*, 2018b.

He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Herzig, R., Raboh, M., Chechik, G., Berant, J., and Globerson, A. Mapping Images to Scene Graphs with Permutation-Invariant Structured Prediction. *arXiv preprint arXiv:1802.05451*, 2018.

Kim, H., Mnih, A., Schwarz, J., Garnelo, M., Eslami, A., Rosenbaum, D., Vinyals, O., and Teh, Y. W. Attentive Neural Processes. *arXiv preprint arXiv:1901.05761*, 2019.

Kingma, D. P. and Ba, J. Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations*, 2015.

Korshunova, I., Degrave, J., Huszár, F., Gal, Y., Gretton, A., and Dambre, J. BRUNO: A Deep Recurrent Model for Exchangeable Data. *Advances in Neural Information Processing Systems*, 2018.

Kosiorek, A. R., Kim, H., Posner, I., and Teh, Y. W. Sequential Attend, Infer, Repeat: Generative Modelling of Moving Objects. *Advances in Neural Information Processing Systems*, 2018.

Kumar, A., Eslami, S., Rezende, D. J., Garnelo, M., Viola, F., Lockhart, E., and Shanahan, M. Consistent Generative Query Networks. *arXiv preprint arXiv:1807.02033*, 2018.

Lee, J., Lee, Y., Kim, J., Kosiorek, A. R., Choi, S., and Teh, Y. W. Set Transformer. *arXiv preprint arXiv:1810.00825*, 2018.

Murphy, R. L., Srinivasan, B., Rao, V., and Ribeiro, B. Janossy Pooling: Learning Deep Permutation-Invariant Functions for Variable-Size Inputs. *arXiv preprint arXiv:1811.01900*, 2018.

Qi, C. R., Su, H., Mo, K., and Guibas, L. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *IEEE Conference on Computer Vision and Pattern Recognition*, 2017a.

Qi, C. R., Yi, L., Su, H., and Guibas, L. J. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. *Advances in Neural Information Processing Systems*, 2017b.

Qi, C. R., Liu, W., Wu, C., Su, H., and Guibas, L. J. Frustum PointNets for 3D Object Detection from RGB-D Data. *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Rasmussen, C. E. and Williams, C. K. I. *Gaussian Processes for Machine Learning*. The MIT Press, 2006. ISBN 026218253X.

Ravanbakhsh, S., Schneider, J., and Poczos, B. Deep Learning with Sets and Point Clouds. *arXiv preprint arXiv:1611.04500*, 2016.

Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W. M., Zambaldi, V., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J. Z., Tuyls, K., et al. Value-Decomposition Networks For Cooperative Multi-Agent Learning. *International Conference on Autonomous Agents and MultiAgent Systems*, 2017.

Sutskever, I., Vinyals, O., and Le, Q. Sequence to Sequence Learning with Neural Networks. *Advances in Neural Information Processing Systems*, 2014.

Vaswani, A., Shazeer, N., Parmer, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L., and Polosukhin, I. Attention Is All You Need. *Advances in Neural Information Processing Systems*, 2017.

Vinyals, O., Bengio, S., and Kudlur, M. Order matters: Sequence to sequence for sets. *arXiv preprint arXiv:1511.06391*, 2015.

Yang, B., Wang, S., Markham, A., and Trigoni, N. Attentional Aggregation of Deep Feature Sets for Multi-view 3D Reconstruction. *arXiv preprint arXiv:1808.00758*, 2018.

Yi, L., Zhao, W., Wang, H., Sung, M., and Guibas, L. GSPN: Generative Shape Proposal Network for 3D Instance Segmentation in Point Cloud. *arXiv preprint arXiv:1812.03320*, 2018.

Zaheer, M., Kottur, S., Ravanbhakhsh, S., Póczos, B., Salakhutdinov, R., and Smola, A. Deep Sets. In *Advances in Neural Information Processing Systems*, 2017.