Journal of Machine Learning **ISSN**: 2790-2048(e), 2790-203X(p)

Why Self-Attention is Natural for Sequence-to-Sequence Problems? A Perspective from Symmetries

Chao Ma * [1] and Lexing Ying † [1]

[1]Department of Mathematics Stanford University, Stanford, CA 94305, USA

Abstract. In this paper, we show that structures similar to self-attention are natural for learning many sequence-to-sequence problems from the perspective of symmetry. Inspired by language processing applications, we study the orthogonal equivariance of seq2seq functions with knowledge, which are functions taking two inputs – an input sequence and a knowledge – and outputting another sequence. The knowledge consists of a set of vectors in the same embedding space as the input sequence, containing the information of the language used to process the input sequence. We show that orthogonal equivariance in the embedding space is natural for seq2seq functions with knowledge, and under such equivariance, the function must take a form close to self-attention. This shows that network structures similar to self-attention are the right structures for representing the target function of many seq2seq problems. The representation can be further refined if a finite information principle is considered, or a permutation equivariance holds for the elements of the input sequence.

Keywords:
Self attention,
Symmetry,
Orthogonal equivariance,
Permutation equivariance.

Article Info.:
Volume: 2
Number: 3
Pages: 194 - 210
Date: September /2023
doi.org/10.4208/jml.221206

Article History:
Received: 06/12/2022
Accepted: 28/08/2023

Communicated by:
Zhi-Qin Xu

1 Introduction

Neural network models using self-attention, such as Transformers [47], have become the new benchmark in the fields such as natural language processing and protein folding. Though, the design of self-attention is largely heuristic, and a theoretical understanding of its success is still lacking. In this paper, we provide a perspective for this problem from the symmetries of sequence-to-sequence (seq2seq) learning problems. By identifying and studying appropriate symmetries for seq2seq problems of practical interest, we demonstrate that structures like self-attention are natural for representing these problems.

Symmetries in learning problems can inspire the invention of simple and efficient neural network structures. This is because symmetries reduce the complexity of the problems, and a network with matching symmetries can learn the problems more efficiently. For instance, convolutional neural networks (CNNs) have seen great success in vision problems, with the translation invariance/equivariance of the problems being one of the main reasons. This is not only observed in practice but also justified theoretically [21]. Many other symmetries have been studied and exploited in the design of neural network

*Corresponding author. chaoma@stanford.edu
†lexing@stanford.edu

models. Examples include permutation equivariance [57] and rotational invariance [8,17], with various applications in learning physical problems. See Section 2.1 for more related works.

In this work, we start by studying the symmetry of seq2seq functions in the embedding space, the space in which each element of the input and output sequences is represented. For a language processing problem, for example, words or tokens are usually vectorized by a one-hot embedding using a vocabulary. In this process, the order of words in the vocabulary should not influence the meaning of input and output sentences. Thus, if a permutation is applied on the dimensions of the embedding space, the input and output sequences should experience the same permutation, without other changes. This implies a permutation equivariance in the embedding space. In our analysis, we consider equivariance under the orthogonal group, which is slightly larger than the permutation group. We show that if a function f is orthogonal equivariant in the embedding space, then its output can be expressed as linear combinations of the elements of the input sequence, with the coefficients only depending on the inner products of these elements. Concretely, let $X \in \mathbb{R}^{d\times n}$ denote an input sequence with length n in the embedding space $\mathbb{R}^d$. If $f(QX)=Qf(X)$ holds for any orthogonal $Q\in\mathbb{R}^{d\times d}$, then there exists a function g such that

$$f(X) = Xg(X^TX).$$

However, the symmetry on the embedding space is actually more complicated than a simple orthogonal equivariance. In Section 3.2, we show that the target function for a simple seq2seq problem is not orthogonal equivariant, because the target function works in a fixed embedding. To accurately catch the symmetry in the embedding space, we propose to study seq2seq functions with knowledge, which are functions with two inputs, $f(X,Z)$, where $X \in \mathbb{R}^{d\times n}$ is the input sequence and $Z \in \mathbb{R}^{d\times k}$ is another input representing our knowledge of the language. The knowledge lies in the same embedding space as X and is used to extract information from X. With this additional input, the symmetry in the embedding space can be formulated as an orthogonal equivariance of $f(X,Z)$, i.e. $f(QX,QZ) = Qf(X,Z)$ for any inputs and orthogonal matrix Q. Intuitively understood, in a language application, as long as the knowledge is always in the same embedding as the input sequence, the meaning of the output sequence will not change with the embedding. Based on the earlier theoretical result for simple orthogonal equivariant functions, if a seq2seq function with knowledge is orthogonal equivariant, then it must have the form

$$f(X,Z) = Xg_1(X^TX, Z^TX, Z^TZ) + Zg_2(X^TX, Z^TX, Z^TZ).$$

If Z is understood as a parameter matrix to be learned, the following subset of this representation:

$$f(X,Z) = Xg(X^TZ)$$

is close to a self-attention used in practice, with Z being the concatenation of query and key parameters. This reveals one possible reason behind the success of self-attention-based models on language problems.

Based on the results from orthogonal equivariance, we further study the permutation equivariance on the elements of the input sequence. Under this symmetry, we show that

seq2seq functions with knowledge have a further reduced form which only involves four different nonlinear functions. Finally, discussions are made on the possible forms of g (or g_1 and g_2) in the formulations mentioned above. Based on the assumption that these functions are described by a finite amount of information (although their output sizes need to change with respect to the sequence length n), we reason that a quadratic form with nonlinearity used in usual self-attentions is one of the simplest choices of g. We also discuss practical considerations that add to the complexity of the models used in application compared with theoretical forms.

2 Background and related work

2.1 Neural networks and symmetries

Implementing symmetries in neural networks can help the models learn certain problems more efficiently. A well-known example is the success of convolutional neural networks (CNNs) on image problems due to their (approximate) translation invariance [19]. Many types of symmetries have been explored in the design of neural networks, such as permutation equivariance and invariance [15, 36, 37, 39, 57], rotational equivariance and invariance [13, 17, 45, 46], and more [23, 40, 42, 52]. Some works deal with multiple symmetries. In [49], the forms of functions with various symmetries are studied. These networks see many applications in physical problems, where symmetries are intrinsic in the problems to learn. Examples include fluid dynamics [20, 24, 29, 51], molecular dynamics [1, 43, 58], quantum mechanics [25, 26, 48], etc. Theoretical studies have also been conducted to show the benefit of preserving symmetry during learning [3, 12, 21, 30].

2.2 Self-attention

Self-attention [22,32,33,44,47] is a type of attention mechanism [2,28] that attends different elements in a same input sequence. It is the building block of a series of large language models (e.g. [6, 10, 38]), and is under extensive research. See [5, 31] for reviews.

As preparation for later studies, we briefly summarize the structure of (multihead) self-attention. A self-attention is a seq2seq operator which takes a sequence of vectors as the input, and another sequence of vectors (of the same size) as the output. Let $X \in \mathbb{R}^{d\times n}$ be the input sequence with length n. A self-attention computes the output using three parameter matrices: the query parameters $W_Q \in \mathbb{R}^{d_1\times d}$, the key parameters $W_K \in \mathbb{R}^{d_1\times d}$, and the value parameters $W_V \in \mathbb{R}^{d\times d}$. Given the input X, a query and a key is computed for every column of X by multiplying with W_Q and W_K, i.e. we compute $Q(X) = W_Q X \in \mathbb{R}^{d_1\times n}$ and $K(X) = W_K X \in \mathbb{R}^{d_1\times n}$. Then, an attention matrix is obtained by computing the inner product of all pairs of queries and keys

$$A(X) = Q(X)^T K(X) = X^T W_Q^T W_K X \in \mathbb{R}^{n\times n}.$$

Next, a weight matrix is computed by applying softmax over rows of A, and the output of the attention is obtained by a linear combination of the values, $W_V X$, using rows in the

DOI https://doi.org/10.4208/jml.221206 | Generated on 2024-12-19 03:19:39

weight matrix as coefficients. In practice, $A(X)$ is usually scaled by a factor of $1/\sqrt{d_1}$, and a residual connection is added, thus we have

$$\mathrm{Attn}(X) = X + W_V X \mathcal{S}_r \left(\frac{1}{\sqrt{d_1}} X^T W_Q^T W_K X \right)^T, \tag{2.1}$$

where $\mathcal{S}_r(\cdot)$ computes the softmax of an input matrix over rows.

Remark 2.1. In this paper, we use $X \in \mathbb{R}^{d\times n}$ to denote sequences with length n in the space $\mathbb{R}^d$. Each column of X is an element of the sequence. In many works, the same sequence is represented by an $n \times d$ matrix. The two representations are intrinsically equivalent.

The self-attention mechanism described above consists of one head, in the sense that we have one query, key, and value for each element of X. Similar to the way that we add more neurons to a layer of a fully connected neural network, we can add more heads to a self-attention, which gives a multihead attention. For a multihead attention with m heads, we have m different query, key, and value matrices, denoted by $W_Q^{(i)}, W_K^{(i)}$ and $W_V^{(i)}$. $W_Q^{(i)}$ and $W_K^{(i)}$ are still $d_1 \times d$ matrices, while $W_V^{(i)}$ are $d_2 \times d$ matrices. Besides, in order to still use the residual connection, an output parameter matrix $W_{out}^{(i)} \in \mathbb{R}^{d\times d_2}$ is added for each head to transform the value vectors in $\mathbb{R}^{d_2}$ into vectors in $\mathbb{R}^d$. With these parameters, each head is similar to a single-head self-attention

$$\mathrm{head}_i(X) = W_V^{(i)} X \mathcal{S}_r \left(\frac{1}{\sqrt{d_1}} X^T \left(W_Q^{(i)} \right)^T W_K^{(i)} X \right)^T,$$

and the output of the multihead attention is

$$\begin{aligned} \mathrm{Attn}_m(X) &= X + \sum_{i=1}^{m} W_{out}^{(i)} \mathrm{head}_i(X) \\ &= X + \sum_{i=1}^{n} W_{out}^{(i)} W_V^{(i)} X \mathcal{S}_r \left(\frac{1}{\sqrt{d_1}} X^T \left(W_Q^{(i)} \right)^T W_K^{(i)} X \right)^T. \end{aligned}$$

Remark 2.2. In a practical model like a Transformer, a fully-connected layer is sometimes added after a multihead attention. The fully-connected layer is applied to each element of the output sequence.

Self-attention-based models such as transformers have also been studied theoretically from different aspects. Many works focused on their approximation capability [16, 18, 27, 56]. Studies had also been done on the Turing completeness [34, 53], in-context learning [9, 55], and inductive bias [11] of the models.

3 Orthogonal equivariance in the embedding space

In this section, we focus on the orthogonal equivariance in the embedding space. We show that functions with such equivariance enjoy a representation that takes a similar but

more general form as self-attention. We start from a theoretical characterization for simple seq2seq functions with orthogonal equivariance (Proposition 3.1). Then, we introduce and study a class of functions called seq2seq function with knowledge, whose form is inspired by typical seq2seq learning problems.

3.1 Simple orthogonal equivariant functions

We first consider orthogonal equivariant functions given by the following definition.

Definition 3.1. *Let $\mathcal{X} = \bigcup_{n=1}^{\infty} \mathbb{R}^{d\times n}$ be the space of all sequences in $\mathbb{R}^d$ and $f : \mathcal{X} \to \mathcal{X}$ a sequence-to-sequence function. The function f is called orthogonal equivariant in the embedding space if for any $X \in \mathcal{X}$ and orthogonal matrix $Q \in \mathbb{R}^{d\times d}$, there is $f(QX) = Qf(X)$.*

For orthogonal equivariant functions, the following proposition shows that any column of the output must be a linear combination of the columns of X, with the coefficients depending only on the inner products between X's columns. A similar result has appeared in [49] and played an important role for physics applications. For the completeness of the work, we give the proof of the proposition in Appendix A.

Proposition 3.1. *Let $f : \mathcal{X} \to \mathcal{X}$ be orthogonal equivariant in the embedding space given by Definition 3.1. Then, there exists a function g taking X^TX as input and producing a matrix with appropriate shape as output such that for all $X \in \mathcal{X}$ we have*

$$f(X) = Xg(X^TX).$$

Proposition 3.1 shows that orthogonal equivariant seq2seq functions always represent a linear combination of the elements of their input sequence X, with the coefficients being orthogonal invariant.

3.2 Orthogonal equivariance with knowledge

Proposition 3.1 treats seq2seq functions that are strictly orthogonal equivariant in the embedding space. For many practical language problems or other seq2seq learning problems, the embedding indeed has some flexibility over orthogonal transformations – the information is encoded only in the relative positions between vectors in the embedding space, and an orthogonal transformation of those vectors does not change the meaning of the sequence, hence the answer of the transformed input sequence should be the transformed original answer.

However, this intuitive symmetry does not mean that the target function is orthogonal equivariant. As an example, consider a seq2seq function f that takes an arithmetic expression as the input and outputs the result of the expression, e.g $f(\text{“}2+1\text{”}) = \text{“}3\text{”}$, $f(\text{“}2-1\text{”}) = \text{“}1\text{”}$. The tokens used in the input and output sequences include single-digit numbers 0-9 and arithmetic operators. These tokens can be cast into vectors by a one-hot embedding. To be simple, suppose we only use operators “+” and “−”. Then, the embedding space has 12 dimensions. One possible embedding is

$$\text{“}+\text{”} \to \boldsymbol{e}_1, \quad \text{“}-\text{”} \to \boldsymbol{e}_2, \quad \text{“}0\text{”} \to \boldsymbol{e}_3, \quad \text{“}1\text{”} \to \boldsymbol{e}_4, \quad \cdots, \quad \text{“}9\text{”} \to \boldsymbol{e}_{12},$$

where $\boldsymbol{e}_i$ is the i-th unit vector in the standard orthonormal basis of $\mathbb{R}^{12}$. Under this embedding, $f(\text{“}2+1\text{”}) = \text{“}3\text{”}$ can be written as

$$f([\boldsymbol{e}_5, \boldsymbol{e}_1, \boldsymbol{e}_4]) = [\boldsymbol{e}_6].$$

Now, let $Q_{12} \in \mathbb{R}^{12\times 12}$ be a linear transformation that swaps the first and second entries of any vector in $\mathbb{R}^{12}$. Then, Q_{12} is orthogonal. If f is orthogonal equivariant, we will have

$$f([\boldsymbol{e}_5, \boldsymbol{e}_2, \boldsymbol{e}_4]) = f(Q[\boldsymbol{e}_5, \boldsymbol{e}_1, \boldsymbol{e}_4]) = [Q\boldsymbol{e}_6] = [\boldsymbol{e}_6].$$

This means $f(\text{“}2-1\text{”}) = \text{“}3\text{”}$, which is obviously not what we expect.

To summarize, the target function is not orthogonal equivariant because it works in a fixed embedding and cannot deal with sequences from different embeddings. The intuitive symmetry we discussed earlier can be understood as symmetry in an equivalent class of target functions. Let f be a seq2seq function in a certain embedding if an orthogonal transformation Q is applied to this embedding, then there exists another function f_Q that satisfies

$$f_Q(QX) = Qf(X).$$

f_Q does the same thing as f in a different embedding. Collecting f_Q for all orthogonal transformations Q, the set $\{f_Q\}$ is an equivalence class of f in all embeddings (obtained by orthogonal transformations).

The discussion above points out that the target function is aware of the embedding it works in. Intuitively, this is because the function contains some knowledge used to process the input sequence, and the knowledge depends on the embedding. Motivated by this point of view, we propose to study functions that take the knowledge as an explicit input. Like the input sequence, the knowledge also consists of vectors in the embedding space, showing its embedding dependence. The knowledge is used to extract information from the input sequence. Concretely, we consider functions $f : \mathcal{X} \times \mathbb{R}^{d\times k} \to \mathcal{X}$ taking two inputs, $X \in \mathcal{X}$ and $Z \in \mathbb{R}^{d\times k}$, with X being the original input sequence, and Z being the knowledge. With this additional knowledge input, the function f can be orthogonal equivariant – changing the embedding transforms X and Z simultaneously, and the true meaning of what f does is not changed. In other words, the equivalent class $\{f_Q\}$ is parameterized by the knowledge input such that $f_Q(\cdot) = f(\cdot, QZ)$.

From now on, we study orthogonal equivariant functions with knowledge, whose definition is given below.

Definition 3.2. *Let $f : \mathcal{X} \times \mathbb{R}^{d\times k} \to \mathcal{X}$ be a seq2seq function with knowledge. For any $Z \in \mathbb{R}^{d\times k}$, f is called orthogonal equivariant with knowledge Z if for any $X \in \mathcal{X}$ and orthogonal matrix $Q \in \mathbb{R}^{d\times d}$, there is $f(QX, QZ) = Qf(X, Z)$.*

As a corollary of Proposition 3.1, we have the following proposition characterizing the formulation of functions satisfying Definition 3.2.

Proposition 3.2. *Let $Z \in \mathbb{R}^{d\times k}$, and $f : \mathcal{X} \times \mathbb{R}^{d\times k} \to \mathcal{X}$ be a function that is orthogonal equivariant with knowledge Z. Then, there exist two functions g_1 and g_2 independent of Z, taking*

X^TX, Z^TX, Z^TZ as inputs, and producing matrices with appropriate shapes as outputs such that for all $X \in \mathcal{X}$, we have

$$f(X,Z) = Xg_1(X^TX, Z^TX, Z^TZ) + Zg_2(X^TX, Z^TX, Z^TZ). \tag{3.1}$$

Proof. Let $\tilde{X} = [X,Z] \in \mathbb{R}^{d\times(n+k)}$. Viewed as a function of $\tilde{X}$, f satisfies $f(Q\tilde{X}) = Qf(\tilde{X})$ for any orthogonal matrix $Q \in \mathbb{R}^{d\times d}$. Hence, by Proposition 3.1, there exists a function g depending on $\tilde{X}^T\tilde{X}$ such that

$$f(\tilde{X}) = \tilde{X}g(\tilde{X}^T\tilde{X}).$$

By the definition of $\tilde{X}$, g can be written as a function of X^TX, Z^TX and Z^TZ, i.e. $g(\tilde{X}^T\tilde{X}) = g(X^TX, Z^TX, Z^TZ)$. Noticing that $\tilde{X}$ has $n+k$ columns, g must have $n+k$ rows. Letting

$$g(X^TX, Z^TX, Z^TZ) = \begin{bmatrix} g_1(X^TX, Z^TX, Z^TZ) \\ g_2(X^TX, Z^TX, Z^TZ) \end{bmatrix}$$

with g_1 taking the first n rows and g_2 taking the next k rows, we have

$$f(X,Z) = Xg_1(X^TX, Z^TX, Z^TZ) + Zg_2(X^TX, Z^TX, Z^TZ).$$

The proof is complete. □

In practice, the knowledge Z in a function $f(X,Z)$ studied above can be treated as a parameter matrix learned during the training process. We note that the self-attention in Eq. (2.1) takes a similar form. In the self-attention, the product of X with the attention matrix has the form $Xg(Z^TX)$, with $Z = [W_Q^T, W_K^T]$ and g as the composition of a quadratic function and a softmax operation

$$g(Y) = \mathcal{S}_r\left(\frac{1}{\sqrt{d_1}} Y^T \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} Y\right).$$

Indeed, similar to our understanding of Z, the query and key parameters in self-attention are usually understood as knowledge of the language used to extract information from the input sequence. These parameters are naturally embedding-dependent. Certainly, the self-attention used in practice contains more components than merely a $Xg(Z^TX)$ form. For example, as shown in Eq. (2.1), a linear transformation in the embedding space is applied by W_V, and a residual connection is added. In Section 5, we discuss some practical considerations that may cause additional complications of the model in practice.

Broadly speaking, though, Z is not limited to the network parameters. For instance, for conditional generation tasks such as translation, question answering, and text-guided image synthesis, Z could include the representations of the source-language text, database-retrieved paragraphs, or user-input descriptions. This understanding can also help us build insights on how to use the side knowledge in these tasks efficiently.

Coming back to the formulation (3.1), if Z is understood as a parameter matrix, it is fixed after training. Then, among the three inputs of g_1 and g_2, Z^TZ is a constant, and X^TX is an identity matrix under one-hot embedding. Hence, Z^TX is the most informative input. Moreover, since Z is a constant matrix, the linear combination of its columns,

$Zg_2(X^TX, Z^TX, Z^TZ)$ becomes less important than the linear combination of X's columns, $Xg_1(X^TX, Z^TX, Z^TZ)$. Extracting the most meaningful parts in the formulation (3.1), we obtain a simpler form $f(X,Z) = Xg_1(Z^TX)$. This coincides with what appears in self-attention.

3.3 Finite information and the representation of coefficients

In formulation (3.1) or the simplified formulation $f(X,Z) = Xg_1(Z^TX)$, the coefficient functions g_1, g_2 can be quite arbitrary. They can have a highly complicated dependence on their inputs. For example, for a function $f(X,Z) = Xg(Z^TX)$ whose output has the same length as the input, when $X \in \mathbb{R}^{d\times n}$, we have $g(Z^TX) \in \mathbb{R}^{n\times n}$. In the most general case, g can have a different formulation $g_n : \mathbb{R}^{k\times n} \to \mathbb{R}^{n\times n}$ for each n. Because n can be arbitrarily large, the description of g requires an infinite amount of information. However, if these functions can be described and implemented by machine learning models, they must contain only a finite amount of information. In other words, the functions cannot get infinitely complicated when the sizes of their inputs become large. In this section, based on this finite information principle, we discuss possible forms of the g's.

For the convenience of the discussion, we focus on the form $f(X,Z) = Xg(Z^TX)$ and assume that the output of f has the same length as its input. Hence, for any input $X \in \mathbb{R}^{d\times n}$, we have $g(Z^TX) \in \mathbb{R}^{n\times n}$. In this case, we put our discussion on g under the following more specific statement of the finite information principle.

Assumption 3.1 (Finite Information Principle)**.** *g is represented by a parameterized model with a finite number of parameters not depending on n.*

This assumption concerns only one aspect of the broader idea of finite information. But it is the only aspect that we can quantify easily. A related concept is the description length, which also captures the amount of information one needs to describe an object [14,50]. For practical machine learning models, the description length should naturally be finite. The description length of deep learning models has been studied in previous works [4].

Now, we consider parameterized representations for g. Given Assumption 3.1, one of the simplest parameterizations is the composition of a nonlinear function and a quadratic form, such as $\sigma(X^TZAZ^TX)$ for some matrix $A \in \mathbb{R}^{k\times k}$. To see this, denote $Y = Z^TX \in \mathbb{R}^{k\times n}$ and consider g represented by a composition of an elementwise nonlinear function and a sum of matrix products involving Y, i.e.

$$g(Y) = \sigma\left(\sum_{i=1}^{N} W^{(i,0)} \prod_{j=1}^{K_i} \tilde{Y} W^{(i,j)}\right), \tag{3.2}$$

where $\tilde{Y}$ is either Y or Y^T, and N can be infinity. In the formulation above, $W^{(i,j)}$ are parameter matrices. By the finite information principle, the dimensions of $W^{(i,j)}$ in Eq. (3.2) should not depend on n. Then, it is easy to show that we always have $K_i \geq 2$ in Eq. (3.2), because terms with $K_i = 0$ or 1 cannot have shape $n \times n$ without n-dependent parameter matrices. Hence, there are no constant or linear terms in the sum of matrix products,

and thus the simplest terms are quadratic terms. In its simplest form, without higher order terms, we have $g(Y) = \sigma(Y^T W Y)$ for some $W \in \mathbb{R}^{k\times k}$, in which case the output always has the shape $n \times n$ for any n. Note that the self-attention matrix used in practice is very close to this form. If Z is the concatenation of the query and key matrices, i.e. $Z = [W_Q^T, W_K^T]$, then by taking

$$A = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix},$$

we have

$$X^T Z A Z^T X = X^T W_Q^T W_K X.$$

The only difference is that the softmax operation is not elementwise.

A perspective from kernels

Another perspective to create g with a finite amount of information is from the kernels. Viewing the input $Y \in \mathbb{R}^{k\times n}$ as n vectors in $\mathbb{R}^k$, g maps the n vectors into an $n \times n$ matrix, characterizing the relations between these vectors. This can naturally be achieved by a kernel function $K(\cdot,\cdot) : \mathbb{R}^k \times \mathbb{R}^k \to \mathbb{R}$. Denote $Y = [\boldsymbol{y}_1, \cdots, \boldsymbol{y}_n]$, then we can let $g(Y) = (K(\boldsymbol{y}_i, \boldsymbol{y}_j))_{n\times n}$. When K is an inner product kernel $K(\mathbf{x}, \boldsymbol{y}) = \sigma(\mathbf{x}^T \boldsymbol{y})$, which is widely used in traditional machine learning models such as the support vector machine, g takes a similar quadratic form (with an elementwise nonlinearity) as in the discussion above, i.e. $g(Y) = \sigma(Y^T Y)$. Besides, there are more kernels to choose from. For instance, a radio basis function (RBF) kernel $K(\mathbf{x}, \boldsymbol{y}) = f(\|\mathbf{x} - \boldsymbol{y}\|)$ can produce a g defined by $g_{ij}(Y) = f(\|\boldsymbol{y}_i - \boldsymbol{y}_j\|)$. These representations of coefficients may see benefits in some special applications. Actually, self-attention using kernels has already been studied in previous works such as [7, 41].

4 Permutation equivariance for sequence elements

In this section, we consider another symmetry – the permutation equivariance for the elements of the sequence. With this permutation equivariance, the form (3.1) can be further restricted. In a seq2seq problem, such as a language problem, though, the order of the input is usually important. Hence, permutation equivariance on the order of the sequence should not be expected. However, in practice, some parts of the problems or models may have permutation equivariance. For example, when self-attention-based models are used to learn seq2seq problems, a position encoding is usually added to the input sequence before being fed into the model [47].[1] In this case, the order information is included in the input sequence and the function implemented by the model can be permutation equivariant.

For any sequence $X \in \mathbb{R}^{d\times n}$, we call n the length of X, denoted by $l(X)$. We consider the following definition of permutation equivariance.

[1] When $X \in \mathbb{R}^{d\times n}$, one example of position encoding is adding a matrix $P \in \mathbb{R}^{d\times n}$ to X that contains position information. The elements of P are $P_{2i-1,j} = \sin(j/n^{2i/d})$ and $P_{2i,j} = \sin(j/n^{2i/d})$

DOI https://doi.org/10.4208/jml.221206 | Generated on 2024-12-19 03:19:39

Definition 4.1. *Let $f : \mathcal{X} \times \mathbb{R}^{d\times k} \to \mathcal{X}$ be a seq2seq function with knowledge. Assume $l(f(X,Z)) = l(X)$ always holds. For any $Z \in \mathbb{R}^{d\times k}$, f is called elementwise permutation equivariant with knowledge Z if for any permutation matrix $P \in \mathbb{R}^{l(X)\times l(X)}$, we have $f(XP,Z) = f(X,Z)P$.*

Based on the discussions in previous sections, we focus on functions with the form $f(X,Z) = Xg(Z^TX)$. Given the additional permutation equivariance in Definition 4.1, we have the following proposition that further narrows down the form of the functions. The proof of the proposition is given in Appendix B.

Proposition 4.1. *Let f be a function with form $f(X,Z) = Xg(Z^TX)$. Assume f is elementwise permutation equivariant with knowledge Z. Then, for any specific n, there exist functions $\rho_1, \rho_2, \psi_1, \psi_2$ such that for any $X \in \mathbb{R}^{d\times n}$ we have*

$$\tilde{g}_{ii}(Z^TX) = \rho_1\left(Z^T\mathbf{x}_i, \sum_{k=1,\, k\neq i}^{n} \psi_1\big(Z^T\mathbf{x}_k, Z^T\mathbf{x}_i\big)\right),$$
$$\tilde{g}_{ij}(Z^TX) = \rho_2\left(Z^T\mathbf{x}_i, Z^T\mathbf{x}_j, \sum_{k=1,\, k\neq i,j}^{n} \psi_2\big(Z^T\mathbf{x}_k, Z^T\mathbf{x}_i, Z^T\mathbf{x}_j\big)\right)$$

for $i,j = 1,2,\ldots,n$ and $j \neq i$, and $f(X,Z) = X\tilde{g}(Z^TX)$. Here, $\tilde{g} = (\tilde{g}_{ij})_{n\times n}$.

Remark 4.1. For a self-attention layer used in practice, $Z = [W_Q, W_K]$, in which case we have

$$g_{ij}(Z^TX) = \frac{e^{\mathbf{x}_i^TW_Q^TW_K\mathbf{x}_j}}{\sum_{k=1}^{n} e^{\mathbf{x}_i^TW_Q^TW_K\mathbf{x}_k}}.$$

Using the form in Proposition 4.1, this g can be obtained by taking

$$\psi_2(Z^T\mathbf{x}, Z^T\boldsymbol{y}, Z^T\boldsymbol{z}) = e^{\boldsymbol{y}^TW_Q^TW_K\mathbf{x}},$$
$$\rho_2(Z^T\mathbf{x}, Z^T\boldsymbol{y}, \psi) = \frac{e^{\mathbf{x}^TW_Q^TW_K\boldsymbol{y}}}{e^{\mathbf{x}^TW_Q^TW_K\boldsymbol{y}} + \psi},$$

and taking

$$\psi_1(Z^T\mathbf{x}, Z^T\boldsymbol{y}) = \psi_2(Z^T\mathbf{x}, Z^T\boldsymbol{y}, Z^T\boldsymbol{y}),$$
$$\rho_1(Z^T\mathbf{x}, \psi) = \rho_2(Z^T\mathbf{x}, Z^T\mathbf{x}, \psi).$$

5 Practical considerations

In previous sections, we revealed the natural forms of seq2seq functions that satisfy specific symmetries that are reasonable for many practical problems. Therefore, the structures identified can be considered when designing neural network models to learn these problems as approaches to improve learning efficiency. The self-attention, although designed

without utilizing these connections between symmetries and structures, has structures that coincide with the forms we identified. This may partially explain the success of self-attention-based models.

Usually, the models used in practice have to be more complicated than that given by the theory to address practical issues not captured in the simplified setting of the theory. For CNNs, for example, convolution layers are stacked to extract features hierarchically, and normalization layers are added to help the training. In the following, we discuss several considerations when the theories built in the previous sections are used in practical applications.

The evolution of embeddings

In our analysis for orthogonal equivariant functions, we assume the input and output are in the same embedding. In practice, this might not be true. For example, for a translation problem, the input and output sequences are in two languages, so they may not share one embedding. In this case, we need to implement a mechanism to change the embedding of the output sequence. The simplest way is to apply an elementwise linear transformation to the output, i.e., for a function f with form $f(X,Z) = Xg(Z^TX)$, we can build a new function $\tilde{f}$ by multiplying a matrix on the left of the output of f

$$\tilde{f}(X,Z,W) = WXg(Z^TX). \tag{5.1}$$

A more flexible way is to apply a general elementwise nonlinear transformation to the output, which can be achieved by a two-layer neural network, as used in many self-attention-based models

$$\tilde{f}(X,Z,U,V) = V\sigma\big(UXg(Z^TX)\big), \tag{5.2}$$

where U, V are parameter matrices, and σ is an elementwise nonlinear activation function.

Higher capacity

In applying neural networks, a higher model capacity is often desired. Giving the model more flexibility compared to the theoretical formulation can help improve the model's performance, as long as the flexibility does not impair the training efficiency. Based on the structures in Eqs. (5.1) or (5.2), more flexibility can be added to the model by considering a multihead version of such functions. For example, a multihead version for (5.1) with m heads can be

$$\tilde{f}_m(X,Z,W) = \sum_{i=1}^{m} W_iXg_i\big(Z_i^TX\big), \tag{5.3}$$

where $Z_1,\cdots,Z_m$ and $W_1,\cdots,W_m$ are different matrices, and $Z = [Z_1,\cdots,Z_m]$, $W = [W_1,\cdots,W_m]$, and $g_1,\cdots,g_m$ are different functions. This structure is similar to the multihead self-attention.

Compositions and hierarchical feature extraction

A very successful way to increase the model capacity and the learning performance is to stack several modules compositionally to form a deep model. A deep model with many

layers can hierarchically extract the information from its input. This is the intuitive reason behind the success of deep neural networks. For sequence-to-sequence applications, we can also stack structures like (5.3) into a deep model. For example, a model with L layers can be

$$h^{(0)} = X, \quad h^{(l)} = \sum_{i=1}^{m} W_i^{(l)} h^{(l-1)} g\left(\left(Z_i^{(l)}\right)^T h^{(l-1)}\right), \quad 1 \le l \le L, \quad f(X, Z, W) = h^{(L)},$$

where Z and W include all $Z_i^{(l)}$ and $W_i^{(l)}$ parameters, respectively. This structure looks similar to the successful large language models used in practice. One difference is that a residual link is added on each layer of those models to help the training. Another difference is that elementwise fully connected layers are added after some self-attentions, which can be understood as stacking structures in Eq. (5.2).

6 Summary

In this paper, we study the representations of sequence-to-sequence functions with certain symmetries and show that such functions have forms similar to self-attention. Hence, self-attention seems to be the natural structure for learning many seq2seq problems. Moreover, except for the inner product-based attention mechanism widely used nowadays, our study reveals more possibilities that may be picked in the design of attention mechanisms, such as higher-order matrix products or the RBF kernels. These forms arise from the discussion on the finite information principle. As a limitation, our discussion on the forms of g in Section 3.3 started from a simple general form (3.2). More general discussions and precise characterizations of the finite information principle are left as important future work.

Appendix A. Proof of Proposition 3.1

Proof. Consider $X = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n] \in \mathbb{R}^{d \times n} \subset \mathcal{X}$. We first show that the columns of $f(X)$ lie in the span of $\mathbf{x}_1, \cdots, \mathbf{x}_n$. Without loss of generality, we assume $f(X)$ has only one column, i.e. $f(X) \in \mathbb{R}^d$. Let $V = \text{span}(\mathbf{x}_1, \cdots, \mathbf{x}_n)$. Then, there exist $\boldsymbol{v} \in V$ and $\boldsymbol{u} \in V^\perp$ such that $f(X) = \boldsymbol{v} + \boldsymbol{u}$. Let $Q_{\boldsymbol{u}}$ be the Householder transformation

$$Q_{\boldsymbol{u}} = I - \frac{2}{\|\boldsymbol{u}\|^2} \boldsymbol{u}\boldsymbol{u}^T.$$

Then, $Q_{\boldsymbol{u}}$ is an orthogonal matrix. By its definition, we have $Q_{\boldsymbol{u}}\boldsymbol{u} = -\boldsymbol{u}$, and $Q_{\boldsymbol{u}}\boldsymbol{w} = \boldsymbol{w}$ for any $\boldsymbol{w} \perp \boldsymbol{u}$, which implies $Q_{\boldsymbol{u}}\boldsymbol{v} = \boldsymbol{v}$ and $Q_{\boldsymbol{u}}\mathbf{x}_i = \mathbf{x}_i$ for all $i = 1, 2, \ldots, n$. Since f is orthogonal equivariant, we have

$$f(Q_{\boldsymbol{u}}X) = Q_{\boldsymbol{u}} f(X) = Q_{\boldsymbol{u}}(\boldsymbol{v} + \boldsymbol{u}) = \boldsymbol{v} - \boldsymbol{u}.$$

On the other hand, since $Q_{\boldsymbol{u}}X = X$, we must have

$$f(Q_{\boldsymbol{u}}X) = f(X) = \boldsymbol{v} + \boldsymbol{u}.$$

Therefore, we have $\boldsymbol{u} = 0$, and $f(X) = \boldsymbol{v} \in V$.

Next, we show that the coefficient of the linear combinations can be taken as orthogonal invariant functions. By the analysis above, there exists a function g of input X such that

$$f(X) = Xg(X).$$

The size of g's output depends on X and $f(X)$. Because f is orthogonal equivariant, for any orthogonal matrix $Q \in \mathbb{R}^{d\times d}$ we have

$$QXg(QX) = f(QX) = Qf(X) = QXg(X),$$

which means $Xg(QX) = Xg(X)$. We can now choose g to satisfy $g(QX) = g(X)$ for any orthogonal Q.

Finally, we invoke the first fundamental theorem of invariant theory for the orthogonal group [35, 54], which states that g only depends on X via X^TX. This completes the proof. Another way to show that g can depend only on X^TX is to show that any $X, Y \in \mathbb{R}^{d\times n}$ satisfying $X^TX = Y^TY$ can be transformed to each other by orthogonal transformations. That is to say, there exists an orthogonal matrix $Q \in \mathbb{R}^{d\times d}$ that satisfies $X = QY$. Therefore, the value of g is fixed as long as its input X has fixed X^TX. □

Appendix B. Proof of Proposition 4.1

Proof. With an abuse of notations, we use $g(X, Z)$ to denote the output of g given inputs X and Z, despite that g only depends on Z^TX. By Definition 4.1, for any permutation matrix $P \in \mathbb{R}^{n\times n}$, we have

$$f(XP, Z) = XPg(XP, Z) = Xg(X, Z)P = f(X, Z)P. \tag{B.1}$$

It is possible to take a $\tilde{g}(X, Z) = \tilde{g}(Z^TX)$ such that $P\tilde{g}(XP, Z) = \tilde{g}(X, Z)P$ holds for any X and P, and

$$X\tilde{g}(X, Z) = Xg(X, Z) = f(X, Z).$$

By definition, $\tilde{g}$ satisfies $\tilde{g}(XP, Z) = P^T\tilde{g}(X, Z)P$, i.e. applying any permutation on X leads to the same permutation on the rows and column of $\tilde{g}(X, Z)$. Recall that the (i, j)-th entry of $\tilde{g}$ is given by the function $\tilde{g}_{ij}$. Denote the output of $\tilde{g}_{ij}$ given input X and Z by $\tilde{g}_{ij}(\mathbf{x}_1, \cdots, \mathbf{x}_n, Z)$. We then study the forms of $\tilde{g}_{ij}$ using $\tilde{g}(XP, Z) = P^T\tilde{g}(X, Z)P$.

First, consider a permutation P_{1i} that swaps $\mathbf{x}_1$ and $\mathbf{x}_i$. By Eq. (B.1), we have $\tilde{g}(XP_{1i}, Z)_{11} = \tilde{g}(X, Z)_{ii}$, which means

$$\tilde{g}_{ii}(\mathbf{x}_1, \cdots, \mathbf{x}_i, \cdots, \mathbf{x}_n, Z) = \tilde{g}_{11}(\mathbf{x}_i, \cdots, \mathbf{x}_1, \cdots, \mathbf{x}_n, Z).$$

Hence, all $\tilde{g}_{ii}$ can be generated by $\tilde{g}_{11}$ with a swap permutation of its inputs. For $\tilde{g}_{11}$, if we apply a permutation that is the identity on 1, the output of $\tilde{g}_{11}$ does not change, although the order of inputs is changed. This means $\tilde{g}_{11}$ is permutation invariant with the inputs $\mathbf{x}_2, \cdots, \mathbf{x}_n$. By [57, Theorem 2], viewed as a function of $\mathbf{x}_2, \cdots, \mathbf{x}_n$, $\tilde{g}_{11}$ has the form

$\rho(\sum_{k=2}^{n}\psi(\mathbf{x}_k))$ for some functions ρ and ψ. Considering the inputs $\mathbf{x}_1$ and Z, the functions ρ and ψ above depend on $\mathbf{x}_1$ and Z. Therefore, there exist functions ρ_1 and ψ_1 such that

$$\tilde{g}_{11}(X,Z)=\rho_1\left(\mathbf{x}_1,Z,\sum_{k=2}^{n}\psi_1(\mathbf{x}_k;\mathbf{x}_1,Z)\right).$$

By the relation between $\tilde{g}_{11}$ and $\tilde{g}_{ii}$, we have

$$\tilde{g}_{ii}(X,Z)=\rho_1\left(\mathbf{x}_i,Z,\sum_{k\neq i}\psi_1(\mathbf{x}_k;\mathbf{x}_i,Z)\right)$$

for any $i=1,2,\ldots,n$.

Next, we consider $\tilde{g}_{ij}$ with $i\neq j$. Without loss of generality, assume $i<j$. Let $P_{1i,2j}$ be a permutation that swaps $\mathbf{x}_1$ with $\mathbf{x}_i$, and $\mathbf{x}_2$ with $\mathbf{x}_j$. By the permutation equivariance, we have

$$\tilde{g}_{ij}(\mathbf{x}_1,\mathbf{x}_2,\cdots,\mathbf{x}_i,\cdots,\mathbf{x}_j,\cdots,\mathbf{x}_n,Z)=\tilde{g}_{12}(\mathbf{x}_i,\mathbf{x}_j,\cdots,\mathbf{x}_1,\cdots,\mathbf{x}_2,\cdots,\mathbf{x}_n,Z),$$

which means any $\tilde{g}_{ij}$ with $i\neq j$ can be generated by $\tilde{g}_{12}$. Focusing on $\tilde{g}_{12}$, similar to the arguments for $\tilde{g}_{11}$, it is easy to show that $\tilde{g}_{12}$ is permutation invariant with inputs $\mathbf{x}_3,\cdots,\mathbf{x}_n$. Therefore, there exist functions ρ_2 and ψ_2, such that

$$\tilde{g}_{12}(X,Z)=\rho_2\left(\mathbf{x}_1,\mathbf{x}_2,Z,\sum_{k=3}^{\infty}\psi_2(\mathbf{x}_k;\mathbf{x}_1,\mathbf{x}_2,Z)\right).$$

Hence,

$$\tilde{g}_{ij}(X,Z)=\rho_2\left(\mathbf{x}_i,\mathbf{x}_j,Z,\sum_{k\neq i,j}\psi_2(\mathbf{x}_k;\mathbf{x}_i,\mathbf{x}_j,Z)\right).$$

The proof is complete. □

References

[1] B. Anderson, T. S. Hy, and R. Kondor, Cormorant: Covariant molecular neural networks, *Adv. Neural Inf. Process. Syst.*, **32**, 2019.

[2] D. Bahdanau, K. Cho, and Y. Bengio, Neural machine translation by jointly learning to align and translate, *arXiv:1409.0473*, 2014.

[3] A. Bietti, L. Venturi, and J. Bruna, On the sample complexity of learning with geometric stability, *arXiv:2106.07148*, 2021.

[4] L. Blier and Y. Ollivier, The description length of deep learning models, *Adv. Neural Inf. Process. Syst.*, **31**, 2018.

[5] R. Bommasani et al., On the opportunities and risks of foundation models, *arXiv:2108.07258*, 2021.

[6] T. Brown et al., Language models are few-shot learners, *Adv. Neural Inf. Process. Syst.*, **33**:1877–1901, 2020.

[7] Y. Chen, Q. Zeng, H. Ji, and Y. Yang, Skyformer: Remodel self-attention with Gaussian kernel and Nyström method, *arXiv:2111.00035*, 2021.

[8] B. Chidester, T. Zhou, M. N. Do, and J. Ma, Rotation equivariant and invariant neural networks for microscopy image analysis, *Bioinformatics*, 35(14):i530–i537, 2019.
[9] D. Dai, Y. Sun, L. Dong, Y. Hao, S. Ma, Z. Sui, and F. Wei, Why can GPT learn in-context? Language models implicitly perform gradient descent as meta-optimizers. In: *Findings of the Association for Computational Linguistics: ACL 2023*, ACL, 4005–4019, 2023.
[10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, *arXiv:1810.04805*, 2018.
[11] B. L. Edelman, S. Goel, S. Kakade, and C. Zhang, Inductive biases and variable creation in self-attention mechanisms. In: *Proceedings of the 39th International Conference on Machine Learning*, PMLR, 5793–5831, 2022.
[12] B. Elesedy and S. Zaidi, Provably strict generalisation benefit for equivariant models. In: *Proceedings of the 38th International Conference on Machine Learning*, PMLR, 2959–2969, 2021.
[13] F. Fuchs, D. Worrall, V. Fischer, and M. Welling, SE(3)-Transformers: 3D roto-translation equivariant attention networks, *Adv. Neural Inf. Process. Syst.*, **33**:1970–1981, 2020.
[14] P. D. Grünwald, *The Minimum Description Length Principle*, MIT Press, 2007.
[15] N. Guttenberg, N. Virgo, O. Witkowski, H. Aoki, and R. Kanai, Permutation-equivariant neural networks applied to dynamics prediction, *arXiv:1612.04530*, 2016.
[16] M. Hahn, Theoretical limitations of self-attention in neural sequence models, *Trans. Assoc. Comput. Linguist.*, **8**:156–171, 2020.
[17] J. Kim, W. Jung, H. Kim, and J. Lee, CyCNN: A rotation invariant CNN using polar mapping and cylindrical convolution layers, *arXiv:2007.10588*, 2020.
[18] A. Kratsios, B. Zamanlooy, T. Liu, and I. Dokmanić, Universal approximation under constraints is possible with transformers, *arXiv:2110.03303*, 2021.
[19] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, Backpropagation applied to handwritten zip code recognition, *Neural Comput.*, **1**(4):541–551, 1989.
[20] Y. Li, J. Chang, C. Kong, and Z. Wang, Flow field reconstruction and prediction of the supersonic cascade channel based on a symmetry neural network under complex and variable conditions, *AIP Adv.*, **10**(6):065116, 2020.
[21] Z. Li, Y. Zhang, and S. Arora, Why are convolutional nets more sample-efficient than fully-connected nets? *arXiv:2010.08515*, 2020.
[22] Z. Lin, M. Feng, C. N. dos Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio, A structured self-attentive sentence embedding, *arXiv:1703.03130*, 2017.
[23] J. Ling, R. Jones, and J. Templeton, Machine learning strategies for systems with invariance properties, *J. Comput. Phys.*, **318**:22–35, 2016.
[24] J. Ling, A. Kurzawski, and J. Templeton, Reynolds averaged turbulence modelling using deep neural networks with embedded invariance, *J. Fluid Mech.*, **807**:155–166, 2016.
[25] D. Luo, G. Carleo, B. K. Clark, and J. Stokes, Gauge equivariant neural networks for quantum lattice gauge theories, *Phys. Rev. Lett.*, **127**(27):276402, 2021.
[26] D. Luo, Z. Chen, K. Hu, Z. Zhao, V. M. Hur, and B. K. Clark, Gauge invariant autoregressive neural networks for quantum lattice models, *arXiv:2101.07243*, 2021.
[27] S. Luo, S. Li, S. Zheng, T.-Y. Liu, L. Wang, and D. He, Your transformer may not be as powerful as you expect, *arXiv:2205.13401*, 2022.
[28] M.-T. Luong, H. Pham, and C. D. Manning, Effective approaches to attention-based neural machine translation, *arXiv:1508.04025*, 2015.
[29] M. Mattheakis, P. Protopapas, D. Sondak, M. Di Giovanni, and E. Kaxiras, Physical symmetries embedded in neural networks, *arXiv:1904.08991*, 2019.
[30] S. Mei, T. Misiakiewicz, and A. Montanari, Learning with invariances in random features and kernel models. In: *Proceedings of 34th Conference on Learning Theory*, PMLR, 3351–3418, 2021.
[31] Z. Niu, G. Zhong, and H. Yu, A review on the attention mechanism of deep learning, *Neurocomputing*, **452**:48–62, 2021.
[32] A. P. Parikh, O. Täckström, D. Das, and J. Uszkoreit, A decomposable attention model for natural language inference, *arXiv:1606.01933*, 2016.

[33] R. Paulus, C. Xiong, and R. Socher, A deep reinforced model for abstractive summarization, *arXiv:1705.04304*, 2017.
[34] J. Pérez, J. Marinković, and P. Barceló, On the turing completeness of modern neural network architectures, *arXiv:1901.03429*, 2019.
[35] C. Procesi, *Lie Groups: An Approach through Invariants and Representations*, Springer, 2007.
[36] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, Pointnet: Deep learning on point sets for 3D classification and segmentation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 652–660, 2017.
[37] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, Pointnet++: Deep hierarchical feature learning on point sets in a metric space, *Adv. Neural Inf. Process. Syst.*, **30**, 2017.
[38] C. Raffel et al., Exploring the limits of transfer learning with a unified text-to-text transformer, *J. Mach. Learn. Res.*, **21**(140):1–67, 2020.
[39] J. Rahme, S. Jelassi, J. Bruna, and S. M. Weinberg, A permutation-equivariant neural network architecture for auction design. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, **35**, 5664–5672, 2021.
[40] S. Ravanbakhsh, J. Schneider, and B. Poczos, Equivariance through parameter-sharing. In: *Proceedings of the 34th International Conference on Machine Learning*, PMLR, 2892–2901, 2017.
[41] D. Rymarczyk, A. Borowa, J. Tabor, and B. Zieliński, Kernel self-attention for weakly-supervised image classification using deep multiple instance learning. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 1720–1729, 2021.
[42] V. G. Satorras, E. Hoogeboom, and M. Welling, E(n) Equivariant Graph Neural Networks, In: *Proceedings of the 38th International Conference on Machine Learning*, PMLR, 9323–9332, 2021.
[43] K. Schütt, O. Unke, and M. Gastegger, Equivariant message passing for the prediction of tensorial properties and molecular spectra. In: *Proceedings of the 38th International Conference on Machine Learning*, PMLR, 9377–9388, 2021.
[44] P. Shaw, J. Uszkoreit, and A. Vaswani, Self-attention with relative position representations, *arXiv:1803.02155*, 2018.
[45] M. Shuaibi, A. Kolluru, A. Das, A. Grover, A. Sriram, Z. Ulissi, and C. L. Zitnick, Rotation invariant Graph Neural Networks using spin convolutions, *arXiv:2106.09575*, 2021.
[46] N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley, Tensor field networks: Rotation- and translation-equivariant neural networks for 3D point clouds, *arXiv:1802.08219*, 2018.
[47] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, Attention is all you need, *Adv. Neural Inf. Process. Syst.*, **30**, 2017.
[48] T. Vieijra, C. Casert, J. Nys, W. De Neve, J. Haegeman, J. Ryckebusch, and F. Verstraete, Restricted Boltzmann machines for quantum states with non-Abelian or anyonic symmetries, *Phys. Rev. Lett.*, **124**(9):097201, 2020.
[49] S. Villar, D. W. Hogg, K. Storey-Fisher, W. Yao, and B. Blum-Smith, Scalars are universal: Equivariant machine learning, structured like classical physics, *Adv. Neural Inf. Process. Syst.*, **34**:28848–28863, 2021.
[50] E. Voita and I. Titov, Information-theoretic probing with minimum description length, *arXiv:2003.12298*, 2020.
[51] R. Wang, K. Kashinath, M. Mustafa, A. Albert, and R. Yu, Towards physics-informed deep learning for turbulent flow prediction. In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1457–1466, 2020.
[52] R. Wang, R. Walters, and R. Yu, Incorporating symmetry into deep dynamics models for improved generalization, *arXiv:2002.03061*, 2020.
[53] C. Wei, Y. Chen, and T. Ma, Statistically meaningful approximation: A case study on approximating Turing machines with transformers, *Adv. Neural Inf. Process. Syst.*, **35**:12071–12083, 2022.
[54] H. Weyl, *The Classical Groups: Their Invariants and Representations*, Princeton University Press, 1946.
[55] S. M. Xie, A. Raghunathan, P. Liang, and T. Ma, An explanation of in-context learning as implicit Bayesian inference, *arXiv:2111.02080*, 2021.
[56] C. Yun, S. Bhojanapalli, A. S. Rawat, S. J. Reddi, and S. Kumar, Are transformers universal approximators of sequence-to-sequence functions? *arXiv:1912.10077*, 2019.

[57] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola, Deep sets, *Adv. Neural Inf. Process. Syst.*, **30**, 2017.
[58] L. Zhang, J. Han, H. Wang, R. Car, and W. E, Deep potential molecular dynamics: A scalable model with the accuracy of quantum mechanics, *Phys. Rev. Lett.*, **120**(14):143001, 2018.