Scaling Spherical CNNs

Carlos Esteves [1] **Jean-Jacques Slotine** [2] **Ameesh Makadia** [1]

Abstract

Spherical CNNs generalize CNNs to functions on the sphere, by using spherical convolutions as the main linear operation. The most accurate and efficient way to compute spherical convolutions is in the spectral domain (via the convolution theorem), which is still costlier than the usual planar convolutions. For this reason, applications of spherical CNNs have so far been limited to small problems that can be approached with low model capacity. In this work, we show how spherical CNNs can be scaled for much larger problems. To achieve this, we make critical improvements including novel variants of common model components, an implementation of core operations to exploit hardware accelerator characteristics, and application-specific input representations that exploit the properties of our model. Experiments show our larger spherical CNNs reach state-of-the-art on several targets of the QM9 molecular benchmark, which was previously dominated by equivariant graph neural networks, and achieve competitive performance on multiple weather forecasting tasks. Our code is available `https://github.com/google-research/spherical-cnn`.

1. Introduction

Spherical convolutional neural networks (Cohen et al., 2018) were introduced as a response to the convolutional neural networks (CNNs) that were central to a series of breakthroughs in computer vision (Krizhevsky et al., 2012; He et al., 2016; Simonyan & Zisserman, 2015; Ronneberger et al., 2015). Given the prevalence of spherical data across many applications, it seemed sensible to design neural networks that possess attributes analogous to those that contribute to the success of planar CNNs, such as translation equivariance, spatial weight sharing, and localized filters.

[1]Google Research, New York, NY, USA [2]Nonlinear Systems Laboratoty, MIT, Cambridge, MA, USA. Correspondence to: Carlos Esteves <machc@google.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

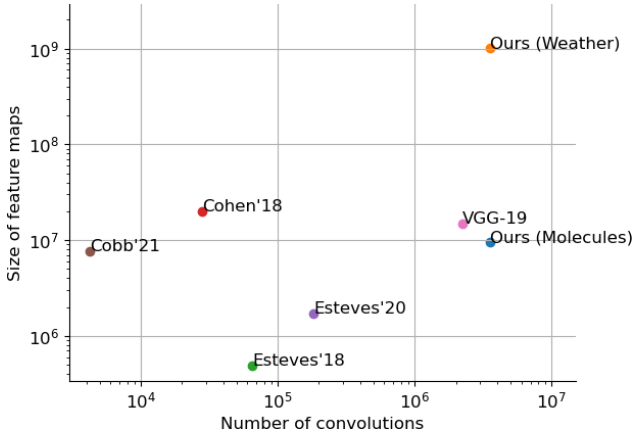

Figure 1. Previous spherical CNNs were limited to low resolutions and relatively shallow models. In this work, we scale spherical CNNs by one order of magnitude and show that they can be competitive and even outperform state-of-the-art graph neural networks and transformers on scientific applications. In the figure, the number of convolutions in a layer mapping between C_{in} and C_{out} channels is counted as $C_{in}C_{out}$, and the feature map size for a $H\times W$ feature with C channels is the number of entries HWC.

Much of the ensuing research into designing spherical CNNs (Cohen et al., 2018; Kondor et al., 2018; Esteves et al., 2020) fulfilled these objectives, providing theoretical guarantees on rotation equivariance, the ability to learn local and expressive filters, and faithful models of both scalar and vector fields on the sphere.

Nonetheless, these models have not impacted many real-world applications. One reason is that learning from large datasets requires models with adequate representational capacity, and it has not yet been shown that spherical convolution layers can be composed to construct such large models effectively. See Figure 1 – there is no spherical CNN architecture used in practice analogous to common CNN models such as VGG19 (Simonyan & Zisserman, 2015).

We are inspired by scientific applications in two areas, drug discovery and climate analysis, that have the potential for broad societal impact. Naturally, both have drawn great interest from the machine learning community – for example AlphaFold for predicting 3D structure of proteins (Jumper et al., 2021), and a litany of deep learning approaches for molecular property prediction (Wiedera et al., 2020). Prop-

erty prediction of small molecules may also be relevant in the design of drugs targeting the interaction between two proteins. For instance, current cancer drugs based on disrupting the binding of tumor suppressor p53 and ubiquitin ligase MDM2 (which targets p53 for degradation) have very low efficiency (Sun, 2006). The second area of interest is short and medium-range weather forecasting (Ravuri et al., 2021; Lam et al., 2022; Rasp et al., 2020). Climate interventions are being considered to mitigate the effects of increased greenhouse gas concentrations in the atmosphere[1]. As such interventions represent uncharted and potentially dangerous territory, climate prediction models may prove important to improve their safety and effectiveness.

Intuitively, both molecular property prediction and climate forecasting problems should benefit from spherical CNNs. The intrinsic properties of molecules are invariant to rotations of the 3D structure (atom positions), so representations that are rotation equivariant by design would provide a natural way to encode this symmetry. However, QM9 (Ramakrishnan et al., 2014), a current standard benchmark for this problem, contains 134K molecules, over 18 times larger than the dataset existing spherical CNNs can accommodate (Rupp et al., 2012). The scale of this problem necessitates models with much greater representation power and computational efficiency.

Similarly, climate forecasting datasets (Rasp et al., 2020) represent samples of the Earth's atmospheric state and thus are ideally represented as spherical signals. Furthermore, in meaningful forecasting applications, models will rely on numerous input variables and the objective is to predict at a high spatial resolution (e.g. 1° angular resolution or 64k samples). Such input and output sizes demand large models.

In this work we present a systematic and principled approach to scale spherical CNNs. Our contributions include

- a design of large scale spherical CNN models, which includes an efficient implementation of spin-weighted spherical harmonic transforms tailored to TPUs,
- general purpose layers and activations that improve expressivity and efficiency,
- application-specific modeling for molecules and weather forecasting.

As the contributions listed above hint at, we observe a naive scaling of existing spherical CNN architectures (simply increasing depth and/or width) is insufficient. Rather, our larger models required a measured design that altered multiple standard components such as the nonlinearity, batch normalization, and residual blocks – all of these improved both efficiency and test performance (see Table 1).

[1] https://www.ametsoc.org/index.cfm/ams/about-ams/ams-statements/statements-of-the-ams-in-force/climate-intervention

These advancements, along with novel domain-specific input feature representations, lead to state of the art performance on the QM9 benchmark, which has been mostly dominated by variations of graph neural networks and transformers. Our models are also competitive in multiple weather forecasting settings, showing, for the first time, that spherical CNNs are viable neural weather models.

This work shows the feasibility of, and introduces best practices for, scaling spherical CNNs. Based on our findings, we expect our JAX (Bradbury et al., 2018) implementation will provide a platform for further research with spherical CNNs targeting real world applications.

2. Related work

2.1. Spherical CNNs

Spherical CNNs have been introduced as the natural extension of standard CNNs to the sphere (Cohen et al., 2018; Esteves et al., 2018), with spherical convolutions computed via generalized Fourier transforms, where the translation equivariance is generalized to 3D rotation equivariance. Later work introduced spectral nonlinearities (Kondor et al., 2018), and extended the equivariance to conformal transformations (Mitchel et al., 2022).

One set of approaches applies filters directly to discrete samples of the spherical input. Perraudin et al. (2019) used a rotation equivariant graph CNN based on isotropic filters. Cohen et al. (2019) considered charts of an icosahedral grid, where filters are rotated and shared, yielding approximate rotation equivariance. Shakerinava & Ravanbakhsh (2021) generalized Cohen et al. (2019) to other grids, using less constrained filters that maintain equivariance.

Tangentially related are methods that operate on the sphere but are not rotation-equivariant (Coors et al., 2018; Jiang et al., 2019; Su & Grauman, 2019).

There have been attempts at improving spherical CNN's efficiency. Esteves et al. (2020) introduced spin-weighted spherical CNNs, which brought anisotropic filters at smaller cost than the full rotation group Fourier transforms. Cobb et al. (2021) counteracted the feature size expansion caused by the tensor products in Kondor et al. (2018) with heuristics to select the representation type at each layer. McEwen et al. (2022) handled high resolution spherical signals by first applying a scattering network with fixed filters (not trainable), followed by downsampling and a spherical CNN. Ocampo et al. (2022) approximated the group convolution integral using a quadrature rule, avoiding expensive generalized Fourier transforms. These previous attempts were still limited to small and sometimes contrived applications.

In this paper, we scale spherical CNNs to a number of operations and feature resolutions one order of magnitude larger

than prior work (see Figure 1), and apply them successfully to large benchmarks on molecule and weather modeling, showing that they can be comparable and sometimes surpass the state-of-the-art graph and transformer-based models.

2.2. Deep learning for molecules

There have been many flavors of message-passing graph neural networks designed specifically for molecules. See Wiedera et al. (2020) and Han et al. (2022) for relevant surveys, and a broader discussion not limited to deep learning techniques can be found in von Lilienfeld et al. (2020). Regarding the deep learning approaches, much of the recent work has focused on 3D equivariant or invariant models.

Examples of invariant models include SchNet (Schütt et al., 2017) and DimeNet++ (Klicpera et al., 2020), where the update function only uses invariant information such as bond angles or atomic distances. E(n)-equivariant networks (Satorras et al., 2021) propose an equivariant update function for node coordinates that operates on directional information (displacement between atoms), although the model instantiation for molecules skips this update leading to strict invariance. Related, Tensor Field Networks (Thomas et al., 2018) also construct an equivariant update function, this one is based on spherical harmonics. Cormorant (Anderson et al., 2019) and Steerable E(3)-equivariant GNNs (Brandstetter et al., 2022) can be seen as extensions of TFN, the former noted for using the Clebsch-Gordan non-linearity, and the latter generalizing to E(n) equivariance. PaiNN (Schütt et al., 2021) is a related model whose gated equivariant update block does not rely on spherical harmonics. This work also closely examines the loss of directional information in invariant models and finds equivariance allows for models with reduced model size. Related to the equivariant graph models are the equivariant transformer approaches such as TorchMD-Net (Thölke & Fabritiis, 2022) which updates scalar and vector node features with self-attention.

2.3. Deep learning for weather modeling

Mudigonda et al. (2017) and Weyn et al. (2019) utilize vanilla CNNs for extreme weather segmentation and short-term forecasting ("NowCasting"), respectively, while Rasp & Thuerey (2021) uses a ResNet model. Other methods for NowCasting use UNets (Agrawal et al., 2019) and conditional generative modeling (Ravuri et al., 2021).

In Weyn et al. (2020) and Lopez-Gomez et al. (2022), a cubed sphere representation (projecting the sphere onto six planar segments) is proposed. This approach enjoys some of the computational benefits of traditional CNNs while more closely observing the underlying spherical topology. Jiang et al. (2019) introduces a model for unstructured grids with experiments on extreme weather segmentation on icosahedral grids. This orientable CNN model does not offer equivariance to 3D rotations and thus expects inputs to be consistently oriented which is true of climate data.

Keisler (2022) recently introduced a graph neural network model inspired by Battaglia et al. (2018). The central component of this model is a message passing network operating on an icosahedral grid. A similar approach is taken in Lam et al. (2022), with a multi-scale mesh graph representation.

The datasets used in most of the recent deep learning research for climate modeling, such as WeatherBench (Rasp et al., 2020), consists of equiangular grids derived from reanalysis of the ERA5 data (Hersbach et al., 2020).

3. Background

Spherical CNNs. The ubiquitous convolutional neural networks (CNNs) for image analysis have convolutions on the plane as their main operation,

$$(f * k)(x) = \int_{t \in \mathbb{R}^2} f(t) k(x - t)\, dt,$$

for an input function f and learnable filter k. This operation brings filter sharing between different regions of the image via translation equivariance, which means that given a shifted input $f'(x) = f(x + h)$, the convolution output also shifts: $(f' * k)(x) = (f * k)(x + h)$. This is one of the main reasons for CNNs high performance.

Spherical CNNs generalize this notion to functions on the sphere ($f\colon S^2 \mapsto \mathbb{R}$) by using spherical convolutions,

$$(f * k)(x) = \int_{g \in \mathbf{SO}(3)} f(g\nu) k(g^{-1}x)\, dg, \qquad (1)$$

where $\mathbf{SO}(3)$ is the group of 3D rotations (which can be represented by special orthogonal 3×3 matrices), and $\nu \in S^2$ is a fixed point. Any two points on the sphere S^2 are related by a rotation in 3D (in technical terms, the sphere is a homogeneous space of the group of rotations), and the spherical convolution is equivariant to 3D rotations. Equation (1) was adopted by Esteves et al. (2018), while Cohen et al. (2018) lifts from S^2 to $\mathbf{SO}(3)$ and performs convolutions on the group, which have an almost identical expression to Equation (1) but with $x \in \mathbf{SO}(3)$.

Esteves et al. (2020) introduced spin-weighted spherical CNNs to overcome the limited expressivity of Esteves et al. (2018) and the computational overhead of Cohen et al. (2018). Spin-weighted spherical functions are complex-valued and their phase changes under rotation. They can also be interpreted as functions on $\mathbf{SO}(3)$ (Boyle, 2016) with sparse spectrum. Convolution is computed through products of spin-weighted spherical harmonics coefficients (Esteves et al., 2020).

Computing generalized convolutions. Approximating spherical and rotation group convolutions with a discrete sum is problematic because there is no arbitrarily dense self-similar grid on the sphere and on the rotation group. Hence, these convolutions are most efficiently and accurately computed in the spectral domain, via products of (generalized) Fourier coefficients (Driscoll & Healy, 1994; Kostelec & Rockmore, 2008; Huffenberger & Wandelt, 2010), which correspond to the spherical harmonics decomposition coefficients $\hat{f}_m^\ell$ for degree ℓ and order m in the case of Equation (1). Even the fastest algorithms for spherical and rotation group convolutions are still much slower than a planar convolution with a 3×3 kernel on modern devices, which has so far limited the applications of spherical CNNs.

In this paper, we adapt the algorithm of Huffenberger & Wandelt (2010) for computing spin-weighted transforms,

$$ {}_s\hat{f}_m^\ell = \int_{S^2} f(x) \overline{{}_sY_m^\ell}(x)\, dx, \tag{2} $$

$$ f(x) = \sum_\ell \sum_{|m|\leq\ell} {}_s\hat{f}_m^\ell\, {}_sY_m^\ell(x), \tag{3} $$

where ${}_sY_m^\ell$ is the spin-weighted spherical harmonic of spin s, degree ℓ, and order m, and ${}_0Y_m^\ell$ corresponds to the standard spherical harmonic Y_m^ℓ. The forward transform implementation rewrites Equation (2) as

$$ {}_s\hat{f}_m^\ell = (-1)^s i^{m+s} \sqrt{\frac{2\ell+1}{4\pi}} \sum_{m'=-\ell}^{\ell} \Delta_{m'm}^\ell \Delta_{m'(-s)}^\ell I_{m'm}, \tag{4} $$

where $\Delta_{m,m'}^\ell = d_{m,m'}^\ell(\pi/2)$ is the Wigner Δ function, d^ℓ is a Wigner (small) d matrix, and $I_{m'm}$ is an inner product with f over the sphere, computed by extending f to the torus and evaluating standard fast Fourier transforms (FFTs) on it. Symmetries of the Wigner Δ enable rewriting the sum in Equation (4) with half the number of terms as

$$ \sum_{m'=-\ell}^{\ell} \Delta_{m'm}^\ell \Delta_{m'(-s)}^\ell I_{m'm} = \sum_{m'=0}^{\ell} \Delta_{m'm}^\ell \Delta_{m'(-s)}^\ell J_{m'm}, \tag{5} $$

where $J_{m'm} = I_{m'm} + (-1)^{m+s} I_{-m'm}$ for $m' > 0$ and $J_{0m} = I_{0m}$.

The inverse transform rewrites Equation (3) as

$$ f(\theta,\phi) = \sum_{m'=-\ell}^{\ell} \sum_{m=-\ell}^{\ell} e^{im'\theta} e^{im\phi} G_{m'm}, \tag{6} $$

$$ G_{m'm} = (-1)^s i^{m+s} \sum_\ell \alpha_\ell \Delta_{(-m')(-s)}^\ell \Delta_{(-m')ms}^\ell \hat{f}_m^\ell, \tag{7} $$

Table 1. Effects of our modeling and implementation contributions. Differences are shown with respect to the results of the previous row. A model similar to the one described in Section 5.1.3 for enthalpy of atomization on QM9 was used for this analysis.

	Δ Steps/s [%] $\uparrow$	Δ RMSE [%] $\downarrow$
JAX implementation	33.7	0.0
Phase collapse	-4.6	-8.0
No Δ symmetries	16.3	0.0
Use DFT	21.4	0.0
Spectral batch norm	7.8	-1.4
Efficient residual	19.3	-2.4

where $\alpha_\ell = \sqrt{\frac{2\ell+1}{4\pi}}$. Again, the Wigner Δ symmetries imply $G_{m'm} = (-1)^{m+s} G_{(-m')m}$ so the full G is reconstructed by computing only half of its values.

The algorithm just described was adopted and implemented in TensorFlow (Abadi et al., 2016) with no changes by Esteves et al. (2020). In this work, we offer a complete rewrite in JAX, tuned for TPUs. This is by itself faster, but we also propose modifications to the algorithm to further improve its speed (see Section 4.2).

4. Method

We contribute a fast implementation of spin-weighted spherical CNNs in JAX, optimized for TPUs, that can run distributed in dozens of devices. The implementation is about $3\times$ faster than the original, and the ability to run distributed can speed it up $100\times$ or more (we use up to 32 TPUs).

Moreover, we introduce a new nonlinearity, normalization layer, and residual block architecture that are more accurate and efficient than the alternatives. Table 1 summarizes the effects on efficiency and accuracy.

4.1. Modeling

Phase collapse nonlinearity. Designing equivariant nonlinearities for equivariant neural networks containing vector or tensor features is challenging. A number of equivariant activations appear in the literature (Weiler et al., 2018; Kondor et al., 2018; de Haan et al., 2021; Xu et al., 2022) and typically the best performing one is problem-dependent. Spin-weighted spherical CNNs require specialized activations for nonzero spin features, and Esteves et al. (2020) chose a simple magnitude thresholding.

Guth et al. (2021) introduced phase collapse nonlinearities for complex-valued planar CNNs with wavelet filters, motivated by 1) translation invariance is usually desirable

for image classification, 2) in the spectral domain, translations correspond to phase shifts, 3) when applying complex wavelet filters to images, which yields complex feature maps, input translations approximately correspond to feature phase shifts, 4) using the modulus as part of the activation collapses the phase, achieving translation invariance and increasing class separability.

We adapt these ideas to spin-weighted spherical functions; in our case, we want rotation invariance (or equivariance) for functions on the sphere. The features are complex-valued and an input rotation results in phase shifts when the spin number is nonzero. Thus, using the modulus as part of the activation eliminates these shifts and brings some degree of invariance. The activation mixes all spins but only updates the zero-spin features. Since the nonzero spin features are unaffected, no information is lost by collapsing the phase.

Formally, let $x_0 \in \mathbb{C}^C$ be the stack of C channels of zero spin at some position on the sphere, and $x \in \mathbb{C}^{SC}$ be a stack of all S spins in the feature map (including zero). We apply

$$x_0 \leftarrow W_1 x_0 + W_2 |x| + b,$$

where W_1, W_2 and b are learnable parameters. This operation updates only the spin zero; subsequent convolutions propagate information to the nonzero ones. Table 6 shows this nonlinearity brings sizeable performance improvements.

Spectral batch normalization. Previous spherical CNNs computed spherical convolutions on the spectral domain and batch normalization on the spatial domain. Batch normalization requires approximating the statistics with a quadrature rule in the spatial domain. Moreover, in the spin-weighted case, zero and nonzero spins need different treatments, which is inefficient.

We propose to compute the batch normalization in the spectral domain instead. Consider that 1) the coefficient ${}_0f^\ell$ for $\ell = 0$ corresponds to the function average, and 2) the variance of the rest of the coefficients is the variance of the function. The normalization is then computed by 1) setting ${}_0f^0$ to zero and 2) dividing all coefficients by the variance. Similarly, a learnable bias is applied by directly setting ${}_0f^0$, and a learnable scale is applied to all coefficients.

The spectral batch norm is shown to be faster and more accurate than the spatial one in Table 1. It also enables a faster residual block as described next.

Spectral pooling and efficient residual block. In contrast with Esteves et al. (2020), and similarly to Cohen et al. (2018), we perform pooling in the spectral domain, which proves to be faster and more accurate. This is because the spatial pooling is sensitive to the sampling grid so it is only approximately rotation-equivariant; it also requires approximation with quadrature weights which adds to the errors. Spectral pooling is implemented by simply skipping the computation of the higher frequency coefficients within a spin-spherical Fourier transform. Spectral pooling is also conceptually different than spatial because it is a global operation while spatial pooling is localized.

One potential issue with spectral pooling is in residual layers, where the downsampling happens in the first Fourier transform, so the downsampled spatial input is never computed and hence cannot be used in the skip-connection. Our solution is to add the residual connection between Fourier coefficients, which is enabled by the spectral batch normalization described earlier. Figure 2 shows our residual block. Table 1 shows it is faster and performs better than the alternative with spatial pooling and batch norms.

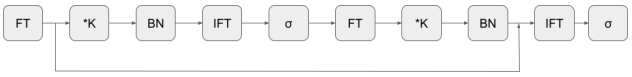

Figure 2. Our efficient residual block contains spin-weighted spherical Fourier transforms (FT) and inverses (IFT), multiplication with filter coefficients ($*K$), activation (σ) and spectral batch normalization (BN). The residual connection happens in Fourier space. Optionally, spectral pooling is performed at the first FT block.

4.2. Efficient TPU implementation

We implement the spin-weighted spherical harmonics transforms aiming for fast execution on TPUs (Jouppi et al., 2017)[2]. This drives our design decisions and sometimes departs from the optimal implementation for CPUs as introduced by Huffenberger & Wandelt (2010). The main difference is that TPUs perform matrix multiplications extremely fast, but memory manipulations like slicing and concatenating tensors may quickly become a bottleneck.

In particular, the use of Wigner Δ symmetries to reduce the number of elements computed in Equations (4) and (7) requires slicing, modifying and reconstructing the original tensors in order to cut the number of operations in half. It turns out this is slower on TPU than just computing twice as many operations without the intermediate steps for the architecture we consider, so we skip the computation of J_{nm} (Equation (5)) completely, and compute all entries of G_{nm} (Equation (7)) in a single step.

Furthermore, Huffenberger & Wandelt (2010) leverage the Fast Fourier transform (FFT) algorithm to reduce asymptotic complexity of the standard Fourier transforms (from $\mathcal{O}(n^2)$ to $\mathcal{O}(n \log n)$). While there are on-device implementations of the FFT, it turns out that in our cases it is significantly faster to compute Fourier transforms as matrix multiplications via the discrete Fourier transform (DFT) matrix. This is because, in a typical neural network pass, we will compute thousands of Fourier transforms (one for each channel for each convolution), but the resolution of

[2]The implementation is compatible with GPUs as well.

each transform is relatively small (up to $n = 256$ in our experiments), so the constant terms dominate and there is no benefit in reducing the asymptotic complexity. Table 1 quantifies the efficiency increase of these changes.

5. Experiments

5.1. Molecular property regression

We first demonstrate scaling spherical CNNs for molecular property regression from atoms and their positions in space, a task that was so far dominated by rotation equivariant graph neural networks and transformer-based models (Wiedera et al., 2020; Klicpera et al., 2020; Liao & Smidt, 2022; Thölke & Fabritiis, 2022). Previous applications of spherical CNNs (Cohen et al., 2018; Kondor et al., 2018; Cobb et al., 2021) considered only the QM7 (Rupp et al., 2012) dataset, which has 7165 molecules with up to 23 atoms, and a single regression target. However, the much larger QM9 (Ramakrishnan et al., 2014) dataset, which contains 134 000 molecules with up to 29 atoms and 12 different regression targets, has supplanted QM7 as the standard benchmark for this task. The molecules are described by their atom types and 3D positions, and labeled with geometric, energetic, electronic, and thermodynamic properties such as enthalpies and free energies of atomization.

In this section, we report the first results of spherical CNNs on QM9. The main reason to employ spherical CNNs for this task is their equivariance to 3D rotations, since the molecule properties do not change under rotations. A secondary reason is that we can design rich physically-based features when mapping from molecule to sphere.

5.1.1. Spherical representation of molecules

The first step for applying spherical CNNs is to represent the molecule as spherical functions. Cohen et al. (2018) proposed a map where spheres are placed around each atom, and points on each sphere are assigned a Coulomb-like quantity using the charge of the central atom and the distances between points on the sphere and other atoms.

We propose an alternative formulation which performs better in practice (see Table 6). Our spherical functions have no assigned radius, so they only contain directional information. The values of these functions are constructed from an inverse power law computed from pairs of atoms, spread out with a Gaussian decay. The input consists of one set of features per atom, with one channel per atom type in the dataset. We sum the contributions of all atoms of the same type. Formally, let z_i be the atomic number of atom i and r_{ij} the displacement between atoms i and j, we define the one

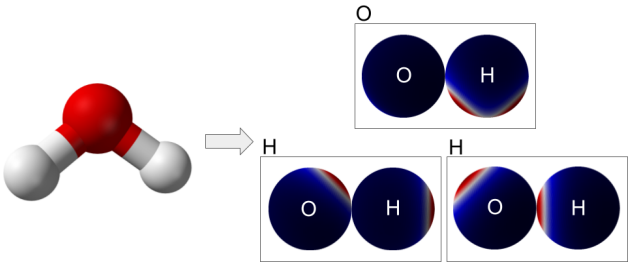

Figure 3. We represent a molecule with a set of Z functions on the sphere for each atom, where Z is the number of atom types in the dataset. Consider the H_2O molecule in the figure and let $Z = 2$; the rectangles show the two channels for each atom. The values on the sphere come from physically-based interactions between pairs of atoms, smoothed with a Gaussian kernel, and aggregated over atom types. For example, the sphere marked with an H on the top right sums up the Coulomb forces between the oxygen the two hydrogen atoms.

input channel of atom i corresponding to atom type z as

$$f_{iz}(x) = \sum_{z_j = z} \frac{z_i z_j}{|r_{ij}|^p} e^{-\frac{(x \cdot r_{ij})^2}{\beta |r_{ij}|}},$$

where β and p are hyperparameters. We set β such that the value is reduced by 95% at $45°$ away from r_{ij}. We stack the features for $p = 2$ and $p = 6$, which correspond to the power laws of Coulomb and van der Waals forces, respectively. These powers have been shown to perform well by Huang & von Lilienfeld (2016) and we confirm their findings in our setting.

Thus, a molecule with N atoms in a dataset containing Z different atom types is represented by $2NZ$ feature maps. The input representation contains global information since it aggregates interactions between all atoms, however the power law makes it biased towards nearby atoms. Figure 3 depicts the spherical representation of an H_2O molecule.

This representation is computed on-device using JAX primitives and thus is differentiable, enabling future applications such as predicting molecule deformations or interactions.

5.1.2. Architecture and training

We first apply a spherical CNN separately to the input features, at 32×32 resolution, for each atom (up to 29 on QM9). The model contains one standard spin-spherical convolutional block followed by 5 residual blocks as depicted in Figure 2 (for a total of 11 convolutional layers) with 64 to 256 channels per layer. Our method's computational cost roughly scales linearly with the number of atoms.

This first step results in one feature map per atom. We then apply global average pooling which results in a set of feature vectors, one per atom. Two different methods are used for aggregating this set to obtain per-molecule predictions. The first method, used for most of the QM9 targets, applies

Table 2. QM9 mean average errors (MAE). We scale spherical CNNs for QM9 for the first time, and show they are competitive with the previously dominant equivariant graph neural networks and transformers. We compare on two splits found in the literature, where "Split 1" has a larger training set. Our model outperforms the baselines on 8 out of 12 targets in "Split 1" and 9 out of 12 targets in "Split 2".

		μ [D]	α [a_0^3]	ϵ_{HOMO} [meV]	ϵ_{LUMO} [meV]	ϵ_{gap} [meV]	$< R^2 >$ [a_0^2]	zpve [meV]	U_0 [meV]	U [meV]	H [meV]	G [meV]	C_v [$\frac{\text{cal}}{\text{mol K}}$]
Split 1	DimeNet++ (2020)	0.030	0.044	24.6	19.5	32.6	0.331	1.21	6.32	6.28	6.53	7.56	0.023
	PaiNN (2021)	0.012	0.045	27.6	20.4	45.7	0.066	1.28	5.85	5.83	5.98	7.35	0.024
	TorchMD-Net (2022)	0.011	0.059	20.3	17.5	36.1	0.033	1.84	6.15	6.38	6.16	7.62	0.026
	Ours	0.016	0.049	21.6	18.0	28.8	0.027	1.15	5.65	5.72	5.69	6.54	0.022
Split 2	EGNN (2021)	0.029	0.071	29.0	25.0	48.0	0.106	1.55	11.00	12.00	12.00	12.00	0.031
	SEGNN (2022)	0.023	0.060	24.0	21.0	42.0	0.660	1.62	15.00	13.00	16.00	15.00	0.031
	Equiformer (2022)	0.014	0.056	17.0	16.0	33.0	0.227	1.32	10.00	11.00	10.00	10.00	0.025
	Ours	0.017	0.049	22.3	19.1	29.8	0.028	1.19	5.96	5.98	5.97	6.97	0.023

a DeepSets (Zaheer et al., 2017) or PointNet (Qi et al., 2017) aggregation, similarly to Cohen et al. (2018). The second method applies a self-attention transformer (Vaswani et al., 2017) with four layers and four heads, and is applied only to the polarizability α and electronic spatial extent $< R^2 >$, which require more refined reasoning between the atom features for accurate prediction. It is common in the literature to use different aggregation for these and other targets (Thölke & Fabritiis, 2022; Schütt et al., 2021).

We train for 2000 epochs on 16 TPUv4 with batch size 16; training runs at around 37 steps/s.

5.1.3. Results

Table 2 shows our results on the QM9 dataset. There are two different splits used in the literature, the major difference being that "Split 1" uses a training set of 110 000 elements while "Split 2" uses 100 000. We evaluate our model on both splits and compare against the relevant models. Our model outperforms the baselines on 8 out of 12 targets in "Split 1" and 9 out of 12 targets in "Split 2".

5.2. Weather forecasting

We now analyze large spherical CNNs for weather forecasting. A unique challenge here is that accurate recordings of weather data are limited to the last few decades, and thus the limited training data motivates a search for the right inductive biases for best generalization.

One potential issue to consider is that the Earth has specific topography and orientation in space which influence the weather, and input atmospheric data is always aligned, so one could argue that global rotation equivariance is unnecessary or even harmful. We claim, however, that equivariance is not harmful because we can simply include constant feature channels such as the latitude, longitude, land-sea mask and orography at each point. In fact, current neural weather models (NWMs) do include these constants (Rasp & Thuerey, 2021; Lopez-Gomez et al., 2022; Keisler, 2022).

Furthermore, we speculate that rotation equivariance can be beneficial, not in the global sense since inputs are aligned, but in the local sense where local patterns can appear at different orientations at different locations.

We evaluate large spherical CNNs on different settings using ERA5 reanalysis data (Hersbach et al., 2020), which combines meteorological observations with simulation models to provide atmospheric data such as wind speed and temperatures uniformly sampled in time and space.

5.2.1. WeatherBench

The WeatherBench (Rasp et al., 2020) benchmark is based on ERA5 data, where the data is provided in hourly timesteps for 40 years, for a total of around 350 000 examples. The dataset is accompanied by simple baseline models for predicting geopotential height at 500 hPa (Z500) and temperature at 850 hPa (T850) at $t + 3$ days and $t + 5$ days given the values at t. In follow-up work, Rasp & Thuerey (2021) applied deep residual networks to similar targets, but now taking a much larger number of predictors including geopotential, wind speed, specific humidity at 7 vertical levels, temperature at 2 m (T2M), total precipitation, and solar radiation. Each predictor is sampled at t, $t - 6$h and $t - 12$h, and the constants land-sea mask, orography, and latitude are included as features, for a total of 117 channels. In both settings, the inputs are sampled at 32×64 resolution.

Architecture and training. We follow Rasp & Thuerey (2021) and train a spherical CNN with an initial block followed by 19 residual blocks (as in Figure 2) and no pooling, for a total of 39 spherical convolutional layers, all with 128 channels. We train one model to directly predict Z500, T850 and T2M at 3 days ahead, and another to predict 5 days ahead. We train for 4 epochs on 16 TPUv4 with batch size 32; training runs at around 8.9 steps/s.

Results. Table 3 shows the results on the test set which comprises years 2017 and 2018. We outperform the baseline on all metrics in the simpler setting that takes two predictors, and show lower temperature errors on the second setting with 117 predictors. The spherical CNN even outperforms models that are pre-trained on large amounts of simulated data on some metrics. We notice that models tend to overfit, and Rasp & Thuerey (2021) employ dropout and learning rate decay based on validation loss to mitigate the issue. We did not use these methods, which might explain our underperforming the geopotential target.

Table 3. WeatherBench results. We report the RMSE on geopotential height (Z500) and temperature at two verticals (T850 and T2M). The top block follows the protocol from Rasp et al. (2020), the middle follows Rasp & Thuerey (2021). A "cont" superscript indicates a continuous model that takes the lead time as input. Spherical CNNs generally outperform conventional CNNs on this task, and even outperform models pre-trained (superscript "pre") on large amounts of simulated data on most temperature metrics.

	3 days			5 days		
	Z500 $[m^2/s^2]$	T850 [K]	T2M [K]	Z500 $[m^2/s^2]$	T850 [K]	T2M [K]
2 predictors						
Rasp et al. (2020)	626	2.87	-	757	3.37	-
Ours	531	2.38	-	717	3.03	-
117 predictors						
Rasp & Thuerey^{cont}	331	1.87	1.60	545	2.57	2.06
Rasp & Thuerey	314	1.79	1.53	561	2.82	2.32
Ours	329	1.62	1.29	601	2.57	1.89
Pretrained						
Rasp & Thuerey^{pre}	268	1.65	1.42	523	2.52	2.03
Rasp & Thuerey^{pre,cont}	284	1.72	1.48	499	2.41	1.92

5.2.2. Global temperature forecasting

Lopez-Gomez et al. (2022) proposed the task of extreme heat forecasting from short to subseasonal (up to 28 days head) timescales. Current physics-based weather models cannot forecast such long lead times, which motivates the use of machine learning. In contrast with Rasp et al. (2020) and Rasp & Thuerey (2021), Lopez-Gomez et al. (2022) does consider the spherical topology and employ an approximately uniform cubical sampling on the sphere.

Data used for this task is averaged over 24 h, and sampled daily, resulting in around 15 000 examples. Furthermore, data that is not present in WeatherBench is used, such as soil moisture, longwave radiation and vorticity. For the task we consider, Lopez-Gomez et al. (2022) used 20 predictors while we use only the 5 that are present in WeatherBench, namely temperature at 2 m (T2M), geopotential height at 300 hPa, 500 hPa and 700 hPa and incoming radiation. Lopez-Gomez et al. (2022) applied a UNet-like (Ronneberger et al., 2015) model on a 6×48×48 cubemap to forecast 28 channels corresponding to 1 to 28 days T2M.

Architecture and training. We used WeatherBench data at 128×128 resolution, which has similar number of samples to the 6×48×48 cubemap. We implement a spherical UNet with 9 spherical convolutional layers with 128 channels each. We train for 5 epochs on 16 TPUv4 with batch size 32; training runs at around 13 steps/s.

Results. Table 4 shows a comparison against 3 models introduced by Lopez-Gomez et al. (2022) over the test set (2017 to 2021). HeatNet has a loss biased towards high temperatures, ExtNet is biased towards both hot and cold extremes, while GenNet uses a standard L2 loss like our model. Our model nearly matches GenNet's performance, even when using a small subset of the predictors.

Table 4. Temperature at 2 m (T2M) prediction, following the protocol and comparing against baselines from Lopez-Gomez et al. (2022). Our model nearly matches the best baseline performance, even when using only a small subset of predictors.

		T2M RMSE [K]					
	Predictors	1 day	2 days	4 days	7 days	14 days	28 days
ExtNet	20	1.15	1.64	2.11	2.31	2.40	2.42
HeatNet	20	1.26	1.77	2.23	2.42	2.50	2.53
GenNet	20	1.13	1.60	2.03	2.22	2.31	2.34
Ours	5	1.24	1.63	2.04	2.27	2.39	2.46

5.2.3. Iterative high resolution forecasting

Keisler (2022) proposed an iterative graph neural network for weather forecasting. In this setting, predictors and targets have the same cardinality such that the model can be iterated repeatedly to forecast longer ranges, where a single iteration produces the forecast 6 h ahead. Temperature, geopotential height, specific humidity and the three components of the wind speed, are all sampled at 13 vertical levels, for a total of 78 predictors and targets, at 180×360 resolution. The dataset comprises the years 1979 to 2020 with one example every 3 h, for a total of 120 000 examples.

Architecture and training. We use the same data as Keisler (2022), but at 256×256 resolution, which has approximately the same number of samples as the baseline.

We implement a spherical UNet with 7 convolutional layers and a single round of subsampling, because a too large receptive field should not be necessary for predicting 6 h ahead iteratively. The model is repeated up to 12 steps during training, for a total of 84 convolutional layers.

We train in 3 stages. The first uses 4 rollout steps (24 h) for 50 epochs on 32 TPUv4 with batch size 32, and is followed by 10 epochs with 8 rollout steps (48 h) and 10 epochs with 12 rollout steps (72 h). Training runs at around 0.92 steps/s, 0.35 steps/s and 0.24 steps/s for each stage.

Results. Table 5 and Figure 4 show the results. Our model tends to perform better at geopotential but worse at temperature forecasts. Keisler (2022) employs a different training procedure where the resolution changes in each stage, and the results are smoothed during evaluation.

Table 5. Iterative weather forecasting, following the protocol from Keisler (2022). We compare results for 24h, 72h and 120h lead times, and report the RMSE. Our model tends to perform better at geopotential but worse at the temperature forecasts.

	1 day		3 days		5 days	
	Z500	T850	Z500	T850	Z500	T850
Keisler (2022)	64.9	0.730	175.5	1.17	344.7	1.78
Ours	58.3	0.827	167.2	1.26	340.0	1.91

5.3. Ablations

Activation, pooling, molecule representation. We train a small model with five spherical convolutional layers to regress the enthalpy of atomization H on QM9; it is trained for 250 epochs (in contrast with 2000 epochs in Section 5.1). We compare the effect of replacing each of our main contributions and show that each of them increases performance. Specifically, we compare against the magnitude activation used in Esteves et al. (2020), the gated activation introduced by Weiler et al. (2018), which we implement by learning a spin 0 feature map that is squashed and pointwise-multiplies each channel. We also compare against the spherical molecular representation introduced by Cohen et al. (2018). Table 6 shows the results.

Table 6. Effects of activation, pooling, molecule representation. We employ a phase collapse activation, compared against the gated nonlinearity of Weiler et al. (2018) and the magnitude thresholding of Esteves et al. (2020). We employ spectral pooling, compared against the spatial pooling from Esteves et al. (2020). We introduce a novel spherical representation of molecules, compared against the one by Cohen et al. (2018).

Activation	Pooling	Molecule representation	QM9/H MAE (meV)
Ours	Ours	Ours	15.25
Ours	Esteves et al.	Ours	16.13
Weiler et al.	Ours	Ours	16.70
Esteves et al.	Ours	Ours	17.01
Ours	Ours	Cohen et al.	20.90

Effects of scaling. In this experiment, we investigate how the resolutions and model capacity affect accuracy in weather forecasting. As in Section 5.2.3, we follow the protocol of Keisler (2022), but only supervising and evaluating the forecasts 6 h in the future. Table 7 shows the results; the most important factors for high performance in this task are the input and feature maps resolutions.

Table 7. Effects of scaling. We report the RMSE for geopotential at 500 hPa (Z500) and temperature at 850 hPa (T850), predicting 6 h ahead following Keisler (2022). Top row shows our base model. The next block reduces the input resolution. The following row uses separable convolutions in every other layer, which reduces the number of convolutions but keeps the feature size constant. The final block reduces the number of channels per layer, which reduces both the number of operations and feature size.

	channels	convolutions	feature size	Z500 [m^2/s^2]	T850 [K]
256×256	100%	3.0×10^5	8.4×10^7	34.93	0.62
192×192	100%	3.0×10^5	4.7×10^7	39.68	0.74
128×128	100%	3.0×10^5	2.1×10^7	46.39	0.87
64×64	100%	3.0×10^5	5.2×10^6	69.60	1.08
256×256	100%	1.6×10^5	8.4×10^7	36.24	0.65
256×256	75%	1.7×10^5	6.3×10^6	36.65	0.65
256×256	50%	7.4×10^4	4.2×10^6	41.34	0.71

6. Discussion

Limitations. The major limitation of our models is still the computational cost – our best results require training up to 4 days on 32 TPUv4, which can be expensive. As a comparison, the baseline for weather (Keisler, 2022) trains in 5.5 days on a single GPU, and the baseline for molecules (Liao & Smidt, 2022) trains in 3 days on a single GPU.

From the point of view of the applications, there are concerns of how much a model trained on reanalysis is useful for forecasting the real weather, and whether models supervised by chemical properties at some level of theory like QM9 are useful to estimate the true properties.

Conclusion. Spherical CNNs possess numerous qualities that makes them appealing for modeling spherical data or rotational symmetries. We have introduced an approach to scaling these models so they can be utilized on larger problems, and our initial results already reach state of the art or comparable performance on molecule and weather prediction tasks. We hope this work and supporting implementation will allow the research community to revisit this powerful class of models for important large scale problems.

Acknowledgments

We thank Stephan Hoyer, Stephan Rasp, and Ignacio Lopez-Gomez for helping with data processing and evaluation, and Fei Sha, Vivian Yang, Anudhyan Boral, Leonardo Zepeda-Núñez, and Avram Hershko for suggestions and discussions.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X. Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pp. 265–283, 2016. URL https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf.

Agrawal, S., Barrington, L., Bromberg, C., Burge, J., Gazen, C., and Hickey, J. Machine learning for precipitation nowcasting from radar images. *arXiv preprint arXiv:1912.12132*, 2019.

Anderson, B., Hy, T. S., and Kondor, R. Cormorant: Covariant molecular neural networks. In *Advances in Neural Information Processing Systems*, pp. 14510–14519, 2019.

Battaglia, P., Hamrick, J. B. C., Bapst, V., Sanchez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Gulcehre, C., Song, F., Ballard, A., Gilmer, J., Dahl, G. E., Vaswani, A., Allen, K., Nash, C., Langston, V. J., Dyer, C., Heess, N., Wierstra, D., Kohli, P., Botvinick, M., Vinyals, O., Li, Y., and Pascanu, R. Relational inductive biases, deep learning, and graph networks. *arXiv*, 2018. URL https://arxiv.org/pdf/1806.01261.pdf.

Boyle, M. How should spin-weighted spherical functions be defined? *Journal of Mathematical Physics*, 57(9): 092504, Sep 2016. ISSN 1089-7658. doi: 10.1063/1.4962723. URL http://dx.doi.org/10.1063/1.4962723.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Brandstetter, J., Hesselink, R., van der Pol, E., Bekkers, E. J., and Welling, M. Geometric and physical quantities improve E(3) equivariant message passing. In *International Conference on Learning Representations, ICLR*, 2022.

Cobb, O., Wallis, C. G. R., Mavor-Parker, A. N., Marignier, A., Price, M. A., d'Avezac, M., and McEwen, J. Efficient generalized spherical {cnn}s. In *International Conference on Learning Representations*, 2021.

Cohen, T., Weiler, M., Kicanaoglu, B., and Welling, M. Gauge equivariant convolutional networks and the icosahedral CNN. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, 2019.

Cohen, T. S., Geiger, M., Köhler, J., and Welling, M. Spherical CNNs. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Hkbd5xZRb.

Coors, B., Condurache, A. P., and Geiger, A. Spherenet: Learning spherical representations for detection and classification in omnidirectional images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

de Haan, P., Weiler, M., Cohen, T., and Welling, M. Gauge equivariant mesh cnns: Anisotropic convolutions on geometric graphs. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021. URL https://openreview.net/forum?id=Jnspzp-oIZE.

Driscoll, J. R. and Healy, D. M. Computing fourier transforms and convolutions on the 2-sphere. *Advances in applied mathematics*, 15(2):202–250, 1994.

Esteves, C., Allen-Blanchette, C., Makadia, A., and Daniilidis, K. Learning SO(3) equivariant representations with spherical cnns. In *The European Conference on Computer Vision (ECCV)*, September 2018.

Esteves, C., Makadia, A., and Daniilidis, K. Spin-weighted spherical CNNs. In *Advances in Neural Information Processing Systems*, 2020.

Gastegger, M., Behler, J., and Marquetand, P. Machine learning molecular dynamics for the simulation of infrared spectra. *Chem. Sci.*, 8:6924–6935, 2017. doi: 10.1039/C7SC02267K. URL http://dx.doi.org/10.1039/C7SC02267K.

Gasteiger, J., Groß, J., and Günnemann, S. Directional message passing for molecular graphs. *CoRR*, 2020. URL http://arxiv.org/abs/2003.03123v2.

Guth, F., Zarka, J., and Mallat, S. Phase collapse in neural networks. *CoRR*, 2021. URL http://arxiv.org/abs/2110.05283v1.

Han, J., Rong, Y., Xu, T., and Huang, W. Geometrically equivariant graph neural networks: A survey. *arXiv preprint arXiv:2202.07230*, 2022.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Hersbach, H., Bell, B., Berrisford, P., Hirahara, S., Horányi, A., Muñoz-Sabater, J., Nicolas, J., Peubey, C., Radu, R., Schepers, D., Simmons, A., Soci, C., Abdalla, S., Abellan, X., Balsamo, G., Bechtold, P., Biavati, G., Bidlot, J., Bonavita, M., De Chiara, G., Dahlgren, P., Dee, D., Diamantakis, M., Dragani, R., Flemming, J., Forbes, R., Fuentes, M., Geer, A., Haimberger, L., Healy, S., Hogan, R. J., Hólm, E., Janisková, M., Keeley, S., Laloyaux, P., Lopez, P., Lupu, C., Radnoti, G., de Rosnay, P., Rozum, I., Vamborg, F., Villaume, S., and Thépaut, J.-N. The era5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146(730): 1999–2049, 2020. doi: https://doi.org/10.1002/qj.3803. URL https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/qj.3803.

Huang, B. and von Lilienfeld, O. A. Communication: Understanding molecular representations in machine learning: The role of uniqueness and target similarity. *The Journal of Chemical Physics*, 145(16):161102, 2016. doi: 10.1063/1.4964627. URL https://doi.org/10.1063/1.4964627.

Huffenberger, K. M. and Wandelt, B. D. Fast and exact spin-s spherical harmonic transforms. *The Astrophysical Journal Supplement Series*, 189(2):255–260, jul 2010. doi: 10.1088/0067-0049/189/2/255.

Jiang, C. M., Huang, J., Kashinath, K., Prabhat, Marcus, P., and Nießner, M. Spherical cnns on unstructured grids. In *International Conference on Learning Representations, (ICLR)*, 2019.

Jouppi, N. P., Young, C., Patil, N., Patterson, D. A., Agrawal, G., Bajwa, R., Bates, S., Bhatia, S., Boden, N., Borchers, A., Boyle, R., luc Cantin, P., Chao, C., Clark, C., Coriell, J., Daley, M., Dau, M., Dean, J., Gelb, B., Ghaemmaghami, T. V., Gottipati, R., Gulland, W., Hagmann, R., Ho, C. R., Hogberg, D., Hu, J., Hundt, R., Hurt, D., Ibarz, J., Jaffey, A., Jaworski, A., Kaplan, A., Khaitan, H., Killebrew, D., Koch, A., Kumar, N., Lacy, S., Laudon, J., Law, J., Le, D., Leary, C., Liu, Z., Lucke, K., Lundin, A., MacKean, G., Maggiore, A., Mahony, M., Miller, K., Nagarajan, R., Narayanaswami, R., Ni, R., Nix, K., Norrie, T., Omernick, M., Penukonda, N., Phelps, A., Ross, J., Ross, M., Salek, A., Samadiani, E., Severn, C., Sizikov, G., Snelham, M., Souter, J., Steinberg, D., Swing, A., Tan, M., Thorson, G., Tian, B., Toma, H., Tuttle, E., Vasudevan, V., Walter, R., Wang, W., Wilcox, E., and Yoon, D. H. In-datacenter performance analysis of a tensor processing unit. In *ISCA*. ACM, 2017.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S. A. A., Ballard, A. J., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J., Back, T., Petersen, S., Reiman, D., Clancy, E., Zielinski, M., Steinegger, M., Pacholska, M., Berghammer, T., Bodenstein, S., Silver, D., Vinyals, O., Senior, A. W., Kavukcuoglu, K., Kohli, P., and Hassabis, D. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, 2021.

Keisler, R. Forecasting global weather with graph neural networks. *CoRR*, 2022. URL http://arxiv.org/abs/2202.07575v1.

Kingma, D. P. and Ba, J. Adam: a method for stochastic optimization. *CoRR*, 2014. URL http://arxiv.org/abs/1412.6980v9.

Klicpera, J., Giri, S., Margraf, J. T., and Günnemann, S. Fast and uncertainty-aware directional message passing for non-equilibrium molecules. *CoRR*, 2020. URL http://arxiv.org/abs/2011.14115v2.

Kondor, R., Lin, Z., and Trivedi, S. Clebsch–gordan nets: a fully fourier space spherical convolutional neural network. In *Advances in Neural Information Processing Systems*, pp. 10138–10147, 2018.

Kostelec, P. J. and Rockmore, D. N. Ffts on the rotation group. *Journal of Fourier Analysis and Applications*, 14 (2):145–179, 2008.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C., Bottou, L., and Weinberger, K. (eds.), *Advances in Neural Information Processing Systems*, volume 25, 2012.

Lam, R., Sanchez-Gonzalez, A., Willson, M., Wirnsberger, P., Fortunato, M., Pritzel, A., Ravuri, S., Ewalds, T., Alet, F., Eaton-Rosen, Z., Hu, W., Merose, A., Hoyer, S., Holland, G., Stott, J., Vinyals, O., Mohamed, S., and Battaglia, P. Graphcast: Learning skillful medium-range global weather forecasting, 2022. URL https://arxiv.org/abs/2212.12794.

Liao, Y.-L. and Smidt, T. Equiformer: Equivariant graph attention transformer for 3d atomistic graphs. *CoRR*, 2022. URL http://arxiv.org/abs/2206.11990.

Lopez-Gomez, I., McGovern, A., Agrawal, S., and Hickey, J. Global extreme heat forecasting using neural weather models. *Artificial Intelligence for the Earth Systems*, pp. 1 – 41, 2022. doi: 10.1175/AIES-D-22-0035.1. URL https://journals.ametsoc.org/view/journals/aies/aop/AIES-D-22-0035.1/AIES-D-22-0035.1.xml.

McEwen, J., Wallis, C., and Mavor-Parker, A. N. Scattering networks on the sphere for scalable and rotationally equivariant spherical CNNs. In *International Conference on Learning Representations*, 2022.

Mitchel, T. W., Aigerman, N., Kim, V. G., and Kazhdan, M. Möbius convolutions for spherical cnns. In *SIGGRAPH '22: Special Interest Group on Computer Graphics and Interactive Techniques Conference, Vancouver, BC, Canada, August 7 - 11, 2022*, pp. 30:1–30:9, 2022. doi: 10.1145/3528233.3530724. URL https://doi.org/10.1145/3528233.3530724.

Mudigonda, M., Kim, S., Mahesh, A., Kahou, S., Kashinath, K., Williams, D., Michalski, V., O'Brien, T., and Prabhat, M. Segmenting and tracking extreme climate events using neural networks. In *Deep Learning for Physical Sciences (DLPS) Workshop, held with NIPS Conference*, 2017.

Ocampo, J., Price, M. A., and McEwen, J. D. Scalable and equivariant spherical cnns by discrete-continuous (disco) convolutions. *CoRR*, 2022. URL http://arxiv.org/abs/2209.13603v1.

Perraudin, N., Defferrard, M., Kacprzak, T., and Sgier, R. Deepsphere: Efficient spherical convolutional neural network with healpix sampling for cosmological applications. *Astronomy and Computing*, 27:130–146, 2019.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 1(2):4, 2017.

Ramakrishnan, R., Dral, P. O., Rupp, M., and von Lilienfeld, O. A. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1, 2014.

Rasp, S. and Thuerey, N. Data-driven medium-range weather prediction with a resnet pretrained on climate simulations: a new model for weatherbench. *Journal of Advances in Modeling Earth Systems*, 13(2):nil, 2021. doi: 10.1029/2020ms002405. URL http://dx.doi.org/10.1029/2020MS002405.

Rasp, S., Dueben, P. D., Scher, S., Weyn, J. A., Mouatadid, S., and Thuerey, N. Weatherbench: a benchmark data set for data-driven weather forecasting. *Journal of Advances in Modeling Earth Systems*, 12(11):nil, 2020. doi: 10.1029/2020ms002203. URL http://dx.doi.org/10.1029/2020MS002203.

Ravuri, S., Lenc, K., Willson, M., Kangin, D., Lam, R., Mirowski, P., Fitzsimons, M., Athanassiadou, M., Kashem, S., Madge, S., Prudden, R., Mandhane, A., Clark, A., Brock, A., Simonyan, K., Hadsell, R., Robinson, N., Clancy, E., Arribas Herranz, A., and Mohamed, S. Skilful precipitation nowcasting using deep generative models of radar. *Nature*, 597:672–677, September 2021.

Ronneberger, O., Fischer, P., and Brox, T. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer Assisted Intervention*, 2015.

Rupp, M., Tkatchenko, A., Müller, K.-R., and Von Lilienfeld, O. A. Fast and accurate modeling of molecular atomization energies with machine learning. *Physical review letters*, 108(5):058301, 2012.

Satorras, V. G., Hoogeboom, E., and Welling, M. E(n) equivariant graph neural networks. *CoRR*, abs/2102.09844, 2021. URL https://arxiv.org/abs/2102.09844.

Schütt, K., Kindermans, P.-J., Felix, H. E. S., Chmiela, S., Tkatchenko, A., and Müller, K.-R. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. In *Advances in Neural Information Processing Systems*, pp. 992–1002, 2017.

Schütt, K., Unke, O., and Gastegger, M. Equivariant message passing for the prediction of tensorial properties and molecular spectra. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9377–9388. PMLR, 18–24 Jul 2021.

Shakerinava, M. and Ravanbakhsh, S. Equivariant networks for pixelized spheres. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, pp. 9477–9488, 2021. URL http://proceedings.mlr.press/v139/shakerinava21a.html.

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

Su, Y. and Grauman, K. Kernel transformer networks for compact spherical convolution. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 9442–9451, 2019. doi: 10.1109/CVPR.2019.00967. URL http://openaccess.thecvf.com/content_CVPR_2019/html/Su_Kernel_Transformer_Networks_for_Compact_Spherical_Convolution_CVPR_2019_paper.html.

Sun, Y. E3 ubiquitin ligases as cancer targets and biomarkers. *Neoplasia*, 8(8):645–654, 2006. ISSN 1476-5586. doi: https://doi.org/10.1593/neo.06376. URL https://www.sciencedirect.com/science/article/pii/S1476558606800035.

Thölke, P. and Fabritiis, G. D. Torchmd-net: Equivariant transformers for neural network based molecular potentials. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=zNHzqZ9wrRB.

Thomas, N., Smidt, T., Kearnes, S., Yang, L., Li, L., Kohlhoff, K., and Riley, P. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv preprint arXiv:1802.08219*, 2018.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. *CoRR*, 2017. URL http://arxiv.org/abs/1706.03762v5.

von Lilienfeld, O. A., Müller, K.-R., and Tkatchenko, A. Exploring chemical compound space with quantum-based machine learning. *Nature Reviews Chemistry*, 4(7):347–358, Jul 2020.

Weiler, M., Geiger, M., Welling, M., Boomsma, W., and Cohen, T. S. 3d steerable cnns: Learning rotationally equivariant features in volumetric data. In *Advances in Neural Information Processing Systems*, pp. 10381–10392, 2018.

Weyn, J. A., Durran, D. R., and Caruana, R. Can machines learn to predict weather? using deep learning to predict gridded 500-hpa geopotential height from historical weather data. *Journal of Advances in Modeling Earth Systems*, 11(8):2680–2693, 2019.

Weyn, J. A., Durran, D. R., and Caruana, R. Improving data-driven global weather prediction using deep convolutional neural networks on a cubed sphere. *Journal of Advances in Modeling Earth Systems*, 12(9), sep 2020.

Wiedera, O., Kohlbachera, S., Kuenemann, M., Garona, A., Ducrota, P., Seidela, T., and ThierryLanger. A compact review of molecular property prediction with graph neural networks. *Drug Discovery Today: Technologies*, 37:1–12, 2020.

Xu, Y., Lei, J., Dobriban, E., and Daniilidis, K. Unified fourier-based kernel and nonlinearity design for equivariant networks on homogeneous spaces. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, pp. 24596–24614, 2022. URL https://proceedings.mlr.press/v162/xu22e.html.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep sets. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/f22e4747da1aa27e363d86d40ff442fe-Paper.pdf.

A. Appendix

A.1. Experimental details

We use the Adam (Kingma & Ba, 2014) optimizer and a cosine decay on the learning rate with one epoch linear warmup in all experiments.

The inputs of all models are conventional spherical functions (zero spin). The first layer maps it to features of spins zero and one, which are mapped back to spin zero at the last spherical convolutional layer. This last feature is complex-valued, which we convert to real by taking the magnitude.

A.1.1. Molecular property regression

For the experiments in Section 5.1, we use five spherical residual blocks with resolutions $[32^2, 16^2, 16^2, 8^2, 8^2]$ and $[64, 128, 128, 256, 256]$ channels per layer. We minimize the L1 loss with a maximum learning rate of 10^{-4}.

Our model applies the spherical CNN independently to each atom's features, followed by global average pooling, resulting in one feature vector per atom. These are further processed by a DeepSets or transformer, as explained in Section 5.1. Finally, we map the set of atom feature vectors to the regression target in three different ways, depending on the target. The dipole moment μ relates to the displacement between atoms and the center of mass, so we use a weighted average by the displacements to aggregate the atom features (as Gastegger et al. (2017)), followed by a small MLP. We compute the electronic spatial extent $< R^2 >$ similarly, but using the distance to the center of mass squared as the weights, following Schütt et al. (2021). For the other targets, which are energy-related, we use the atom types as the weights.

Following Gasteiger et al. (2020), we estimate ϵ_{gap} as $\epsilon_{\text{HOMO}} - \epsilon_{\text{LUMO}}$, using the predictions from models the trained for ϵ_{HOMO} and ϵ_{LUMO}, without training a model specifically for the gap.

A.1.2. Iterative high resolution weather forecasting

We implement a spherical UNet similar to the one in Appendix A.1.4, with feature maps of resolutions $[256^2, 256^2, 128^2, 128^2, 128^2, 128^2, 256^2, 256^2]$ and $[128, 128, 256, 256, 256, 256, 128, 128]$ channels per layer, which are followed by batch normalization and phase collapse

Similarly to Keisler (2022), we concatenate a few constant fields to the 78 predictors; namely, the orography, land-sea mask, latitude (sine), longitude (sine and cosine), hour of the year, and hour of the day.

The maximum learning rate for the first stage is 2×10^{-4}, and we reduced it by a factor of 10 at each subsequent state.

A.1.3. WeatherBench

For the experiments in Section 5.2.1, we use 64×64 inputs and feature maps, while the baseline is at 32×64. Since the spherical harmonic transform algorithm we use requires the same number of samples along both axes, we upsample the inputs from 32×64 to 64×64. We minimize the L2 loss for this and all weather experiments.

A.1.4. Global temperature forecasting

For the experiments in Section 5.2.2, we implement a spherical UNet with feature maps of resolutions $[128^2, 64^2, 64^2, 32^2, 32^2, 32^2, 32^2, 64^2, 64^2, 128^2, 128^2]$, and 128 channels on all convolutional layers, which are followed by batch normalization and phase collapse activation. Features in the downsampling layers are concatenated to the same resolutions in the upsampling layers.

A.2. Extra experiments

FFT vs DFT. One of our perhaps surprising findings is that computing Fourier transforms via DFT matrix multiplication is faster than using the fast Fourier transform (FFT) algorithm. Here, we investigate whether this remains true for larger input resolutions. We train shallow models with only two spin-spherical convolutional layers on the protocol of Keisler (2022), with upsampled inputs to 512×512 and 768×768. Table 8 shows the results when running on 32 TPUv4 with batch size of one per device. The direct DFT method performs better than FFT on TPU even at higher resolutions, due the TPU greatly favoring computing a large matrix multiplication instead of running multiple steps on smaller inputs.

Table 8. Training time comparison of a shallow model using DFT and FFT for Fourier transform computation, varying the input resolution.

FT method	resolution	steps/s
DFT	256×256	28.5
FFT	256×256	18.7
DFT	512×512	12.6
FFT	512×512	5.8
DFT	768×768	5.1
FFT	768×768	1.8

TPUs vs GPUs While we made design decisions with TPUs in mind, the model can also run on GPUs. We evaluated our model for molecules (Section 5.1) on 8 V100 GPUs, with batch size of 1 per device, and it trains at 13.1 steps/s.

In comparison, the same model trains at 35.6 steps/s on 16 TPUv4.

Visualization Figure 4 shows a sequence of predictions of our model for a few variables, compared to the ground truth.

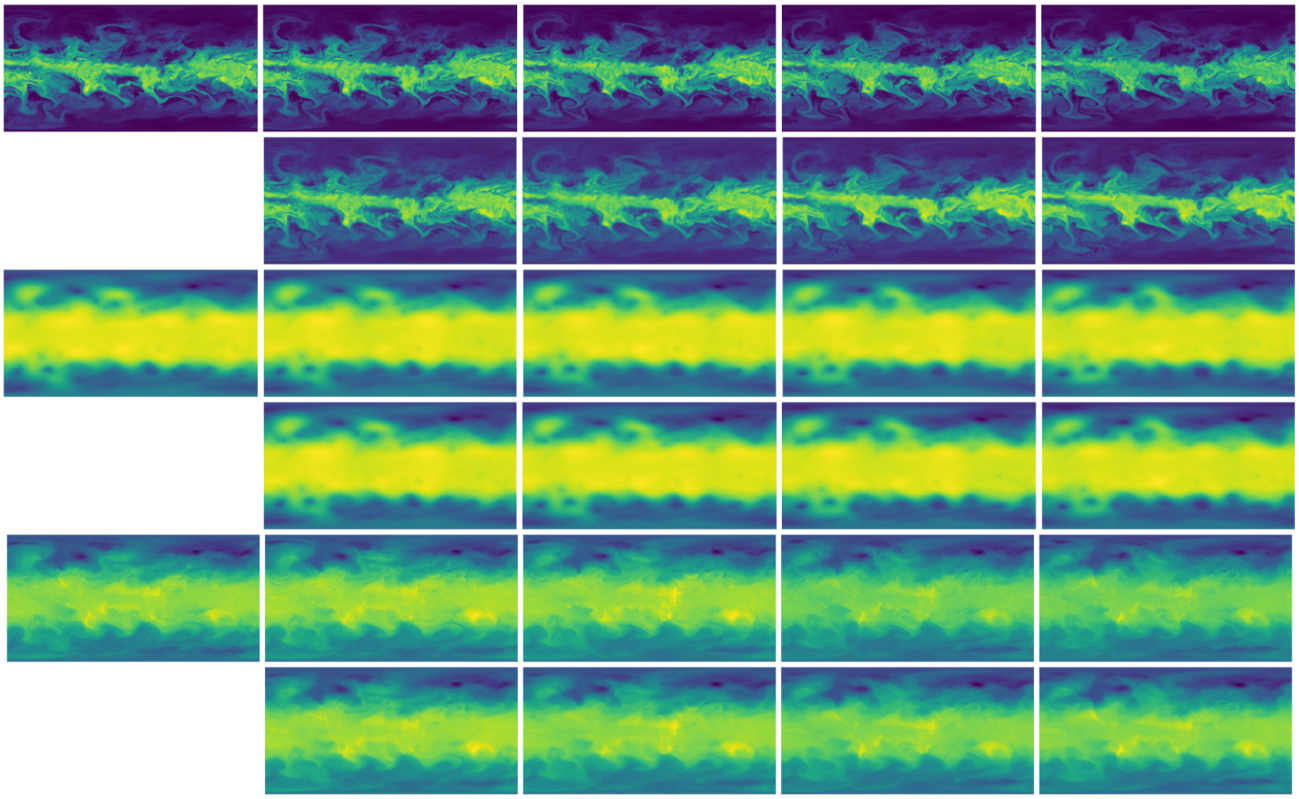

Figure 4. One day rollout of a few predictions of our model. *Top two rows:* specific humidity at $850\,\mathrm{hPa}$ (Q850). *Middle two rows:* geopotential height at $500\,\mathrm{hPa}$ (Z500). *Bottom two rows:* temperature at $850\,\mathrm{hPa}$ (T500). The first column shows the input values at $t = 0$, and subsequent columns shows $6\,\mathrm{h}$ steps. On each group of two rows, the top shows the ground truth and the bottom one shows our predictions. Our predictions show that large spherical CNNs are capable of producing high resolution outputs with high frequency details.