Towards Isotropic Deep Learning A New Default Inductive Bias

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

This position paper explores the geometric implications of current functional forms in deep learning and argues for the adoption of a new paradigm to be termed: *Isotropic Deep Learning*. The existing inductive bias for functional forms is a discrete rotational symmetry — an often underappreciated *choice*. This symmetry is promoted to a continuous rotationally symmetric framework in this paper. It has been demonstrated that the current functional forms influence the activation distributions. Through training, the discrete symmetries in current forms can induce similar broken symmetries in embedded representations [1]. This produces a geometric artefact in representations that is not task-necessitated, solely due to human-imposed choices. There appears to be no strong a priori justification for why such a representation or functional form is universally desirable, and this paper proposes several detrimental effects of this current formulation. As a result, this modified framework for functional forms is explored with the goal of unconstraining representations and improving network performance. This framework is encouraged to be developed substantially so it can be adopted as a new default in time. A variety of preliminary functions are proposed within the paper, including several activation functions. Since this overhauls almost all functional forms characterising modern deep learning, it is suggested that this shift may constitute a novel category of deep learning.

1 Introduction

Elementwise functional forms singularly dominate current deep learning [2, 3, 4, 5, 6, 7, 8, 9, 10, 11]. This is particularly evident in activation functions, sometimes referred to as 'ridged' [12] activation functions. Activation functions are often displayed univariately [13, 14, 15], generally characterised in form by *Eqn.* 1, with σ being a placeholder activation function, e.g. ReLU ($f(x) = \max(0, x)$), Tanh ($f(x) = \tanh(x)$), etc.

$$f : \mathbb{R} \to \mathbb{R}, \quad x \mapsto f(x) = \sigma(x) \tag{1}$$

However, this display choice obfuscates a crucial (standard) basis dependence. This is explicitly displayed in *Eqn.* 2 as a multivariate functional form, which should be considered a more implementation-correct form[1]. This reveals the functional form's usually hidden $\hat{e}_i$ basis dependence. The multivariate form is depicted for an n neuron layer, with activation vector $\vec{x} \in \mathbb{R}^n$. This standard basis dependence is arbitrary and appears as a historical precedent, rather than appropriate inductive bias, discussed further in *App.* G.

$$\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n, \quad \vec{x} \mapsto \mathbf{f}(\vec{x}) = \sum_{i=1}^{n} \sigma(\vec{x} \cdot \hat{e}_i)\, \hat{e}_i \tag{2}$$

[1]Softmax has an extra denominator term, but still displays the basis-dependent nature of elementwise forms.

Due to this basis dependence, non-linear transformations differ angularly in effect [16, 1]. Therefore, this will be termed an *anisotropic function*, indicating this rotational asymmetry. Due to the pervasive use of these functional forms, including optimisers, normalisers, regularisers, activation functions etc., current deep learning as a paradigm may consequently be termed '*anisotropic deep learning*'. Despite its implications, this *choice* of basis-dependent anisotropic form appears underappreciated and incidental in the development of most contemporary models, with anisotropic forms are almost treated as axiomatic to deep learning rather than a considered choice. Hence, re-evaluating and systematically reformulating this foundational aspect of modern deep learning, with such widereaching consequences, is felt to constitute a new branch: '*Isotropic Deep Learning*'.

This asymmetry in the non-linear transform is particularly about the standard (Kronecker) basis vectors, and their negative, $\{+\hat{e}_i, -\hat{e}_i\}_{\forall i}$, due to the elementwise application. Therefore, it can be said to distinguish the standard basis - a '*distinguished basis*'[2]. This is an often unappreciated implicit inductive bias in the representational geometry. For example, the standard basis' activation space distortions are visible in *Fig.* 1 showing the mapping of elementwise-$\tanh$ on a variety of test shapes. Most generally, this *choice* of functional forms can be considered to break *continuous*

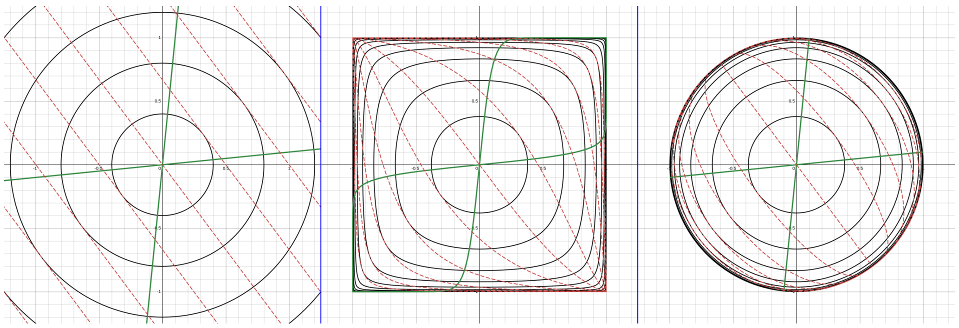

Figure 1: Left shows a 2-dimensional plane, $\mathbb{R}^2$, populated with various shapes: black concentric circles, green lines through the origin, red parallel lines and in faint black the standard (cartesian) coordinate axes $\hat{e}_1$ and $\hat{e}_2$ (which are kept untransformed). If this space is then imaged through elementwise-$\tanh$, the individual pointwise coordinates making up the shapes are passed through the standard $\tanh$ activation. The resultant shapes are shown in the centre plot. Right shows a similar plot, but for the so-called isotropic-$\tanh$ presented in *Sec:* 3.1. One can see that the objects in the centre plot are distorted around the basis directions, whilst in the right-most plot, they are not distorted due to the basis directions. An interactive demonstration of these functions is available [Removed for Anonymity].

rotational symmetry, and reduce it to a *discrete* rotational symmetry — a permutation symmetry of the standard basis. In effect, if the function is treated in its multivariate form, in deep learnings current formulation, it is equivariant to a permutation of the components of its vector decomposed in the standard basis. For an element of the permutation group $\mathbf{P} \in \mathcal{S}_n$, the following equivariance relation holds $f(\mathbf{P}\vec{x}) = \mathbf{P}f(\vec{x})$. However, permutation symmetry is a discrete rotational symmetry which is a subgroup of the special-orthogonal symmetry proposed: $\mathcal{S}_n \subset \mathrm{SO}(n)$ — the permutation symmetry can be said to be a broken continuous rotational symmetry.

Non-linearities are usually pivotal to the network's ability to achieve a desired computation, as seen through the universal approximation theorem's [17] explicit dependence on the form of the activation function [18]. The non-linearities produce differing local transformations, such as stretching, compressing, and generally reshaping a manifold - displayed elegantly in Olah [19] (whom also stated disillusionment on the elementwise form for different reasons). Consequently, the network may be expected to adapt by moving representations to geometries about these distinguished directions, to use specific local transforms to achieve the desired computation. Hence, an anisotropy about a distinguished basis is induced into the activation distribution shown by.

[2]This is suggested generalisation from a *'privileged basis'* discussed in Elhage et al. [16]. *'Distinguished basis'* is felt to better reflect how representations may be more-or-less aligned to a basis; whereas *'privileged basis'* is felt to suggest a greater alignment. The term 'basis' will be retained despite the set of '*distinguished vectors*' potentially being under-/over complete for spanning the activation space, as demonstrated by Bird [1].

This has been empirically demonstrated by Bird [1]: training results in the discrete symmetry of the functional forms, inducing a broken symmetry in the activations. Since these non-linear zones are centred around the distinguished bases, the embedded representations are expected to move to beneficial angular arrangements about the arbitrarily imposed privileged basis's geometry. For example, they appear to move towards the non-linearities' extremums, aligned, anti-aligned or other geometries, through training [1]. This may correspond to a local, dense or sparse coding [20] or superposition [16], respectively.

Therefore, the network has adapted its representations through training due to these functional form choices, probably for various optimisation reasons. The causal hypothesis aids in explaining the observed tendency of privileged-basis alignment. This is the hypothesis underlying the author's position: **functional forms should be a deliberate and considered decision, with a suitably optimal, and minimally harmful, default**. Currently, anisotropy results in a human-caused representational collapse onto the privileged basis, and it is suggested that this is frequently not a task-necessitated collapse. There appears to be little justification for why this is universally desirable, with several key negative implications discussed in *Sec.* 2. Without a priori justification, this inductive bias may be detrimental to computation, so unconstraining the activation appears generally preferable.

Overall, historic and frequently overlooked functional forms for modern deep learning directly influence the models' activations and therefore behaviour. A functional form has been entrenched, which is basis-dependent on an arbitrary basis, typically obsfuscated, neglected, and seldom questioned [21]. Yet, a causal link between this arbitrary basis and activations has been empirically demonstrated [1]. Hence, a resultant effect on the final performance of the model is hypothesised and should be explored. These are often underappreciated choices due to decades of largely unquestioned usage, which appear to have significant consequences for representations. Therefore, this should be well-justified as a default and studied. This is the position of the author.

Throughout the rest of this position paper, **it is argued that a departure from this anisotropic functional form paradigm towards the isotropic paradigm may be generally preferable as an inductive bias**, unless otherwise justified. It encourages the reader to be conscious of these choices when designing a model, as well as the usual architectural tool kit. Particularly, isotropic choices, equivalent to basis independence, may be thought to unconstrain the representations into more optimal arrangements for general tasks and architectures. There are some instances where isotropy may be particularly beneficial, such as the amendments discussed for self-attention, speculated in *App.* E.1. At least the tenets of this paradigm of reviewing such functional form choices are encouraged, and comparisons at least should reveal which characteristics of functional forms most significantly contribute to performance.

2 Hypothesised Problems of Anisotropy

This section argues that current functional forms impose unintended anisotropic performance detriments, indicating that *Isotropic Deep Learning*, once substantially developed, are proposed as the default inductive bias unless an alternative is task-necessitated.

This section lays out a non-exhaustive set of arguments discussing some implications that anisotropic functional forms may cause. These mainly centre on the role of the activation functions, since this is the area the author has primarily explored in their PhD thus far. To the author's knowledge, some of these failure modes are newly characterised phenomena, such as the so-called '*neural refractive problem*'.

2.1 The Neural Refractive Problem

The '*neural refractive problem*' describes how linear and origin-intersecting trajectories of activations may converge or diverge from their initial path after an activation function is applied. This is analogous to a light ray refracting through an optically varying medium or boundary.

It appears to be a phenomenon in all anisotropic activation functions to date. The 'refraction effect' typically occurs more significantly at larger magnitudes — potentially a failure mode under network extrapolation. Neural refraction is demonstrated by curvature of previously straight origin-intersecting lines in the centre plot of *Fig.* 1, but not in the rightmost plot of the same figure.

Mathematically, this has several representations, a magnitude-varying 'dynamic refraction' shown in *Eqn.* 3 or differentially in *Eqn.* 4. Also defined is a 'static refraction' definition shown in *Eqn.* 5. These are described for a multivariate activation function $\mathbf{f}$ and vector $\vec{x} = \alpha\hat{x}$ where $\hat{x}$ is a unit vector. This relation may be satisfied for a single direction, a subset of the space or all directions $\hat{x} \in \mathcal{X} \subseteq S^n$. The relations generally show how the activation function alters the direction of its input vector in an anisotropic manner.

$$\exists\hat{x} \in \mathcal{S}^{n-1}, \exists\alpha_1 \neq \alpha_2 > 0 : \frac{\mathbf{f}\left(\alpha_1\hat{x}\right)}{\left\|\mathbf{f}\left(\alpha_1\hat{x}\right)\right\|} \neq \frac{\mathbf{f}\left(\alpha_2\hat{x}\right)}{\left\|\mathbf{f}\left(\alpha_2\hat{x}\right)\right\|} \tag{3}$$

$$\exists\hat{x} \in \mathcal{S}^{n-1}, \exists\alpha_0 : \frac{\partial}{\partial\alpha} \left.\frac{\mathbf{f}\left(\alpha\hat{x}\right)}{\left\|\mathbf{f}\left(\alpha\hat{x}\right)\right\|}\right|_{\alpha_0} \neq \vec{0} \tag{4}$$

$$\exists\hat{x} \in \mathcal{S}^{n-1}, \exists\alpha : \frac{\mathbf{f}\left(\alpha\hat{x}\right)}{\left\|\mathbf{f}\left(\alpha\hat{x}\right)\right\|} \neq \hat{x} \tag{5}$$

It can be seen that along a straight-line trajectory in direction $\hat{x}$, the result of the activation function is a curved line if dynamically refracted. Therefore, if the linear feature hypothesis is followed, then *every* linear feature, in refracted directions, becomes curved following the activation function. The network may exploit some of this curvature to construct new linear features in the subsequent layers; however, there may be many instances where this curvature is detrimental to established semantics. The network may lose semantic separability, produce magnitude-based semantic inconsistency or compensatory maladaptations in later layers, which fail for out-of-distribution samples. This may hinder the generalisation performance of the network, which is resolved by isotropic choices.

Particularly detrimental, in both refraction cases, can be the loss of semantic separability. If two distinct trajectories, representing different semantics, are transformed into curves which intersect or converge, then the separability of these concepts is lost or misrepresented. For example, suppose one direction is a linear feature for the presence of a dog in an image, whilst the other is for a horse. In that case, if these activations are of a magnitude where the activation function causes convergence, the identity of the activation's meaning can be misconstrued.

This may be particularly consequential for functions such as Sigmoid and Tanh, since large magnitude inputs end up at particular limit points (discussed as trivial representational alignments in Bird [1]). For example, Tanh produces the limit points shown in *Eqn.* 6 when $\hat{x} \cdot \hat{e}_i \neq 0$ for all i. If there exists an i, s.t. $\hat{x} \cdot \hat{e}_i = 0$, then the transformed vector has a 0 in the corresponding index. Therefore, all vectors end up at limit points with sufficient magnitude when using elementwise-Tanh or -Sigmoid. A fully-connected layer can only effectively separate two such converging directions at a time, which are then further curved by a subsequent activation function.

$$\lim_{\substack{\forall i, \hat{x}\cdot\hat{e}_i \neq 0 \\ \alpha\to\infty}} \mathbf{f}\left(\alpha\hat{x}\right) = \sum_{i=1}^{N} \tanh\left(\alpha\hat{x}\cdot\hat{e}_i\right)\hat{e}_i \approx \sum_{i=1}^{N} \pm\hat{e}_i = \left(\pm1, \cdots, \pm1\right)^T \tag{6}$$

Consequently, semantic separability is lost for large magnitudes except for 3^n discrete limit points for Tanh and Sigmoid. Therefore, embedded activations may be expected to align with these limit points. This explains some results empirically observed by Bird [1]. Similarly, ReLU has one distinct limit point, $\vec{0}$, but otherwise an orthant unaffected by neural refraction. The author speculates whether this is an additional reason for the success of ReLU, due to only a subset of directions experiencing the neural refraction phenomena. Furthermore, this would suggest an advantage of Leaky-ReLU: despite featuring static-refraction, directions do not become overlapped, so semantic separability is retained. Otherwise, the network may expend training time on producing robust semantic separability, a needless compensatory adaptation, which may lower representational capacity or extend training as a result.

More generally, dynamic deflection of trajectories may cause semantic ambiguity for the network, where only samples interpolable from training samples are reliably semantically identifiable. Particularly, *the more significant the deflection, the greater the semantic ambiguity may be expected.* Therefore, a magnitude-dependent semantic inconsistency may arise due to such deflections. A deflection function can be a trivial diagnostic measure, defined by *Eqn.* 7 for a particular activation function.

$$\theta\left(\alpha; \hat{x}, \mathbf{f}\right) = \arccos\left(\frac{\mathbf{f}\left(\alpha\hat{x}\right)\cdot\hat{x}}{\left\|\mathbf{f}\left(\alpha\hat{x}\right)\right\|}\right) \tag{7}$$

This may explain why the network may perform excessively poorly on out-of-training-distribution samples. For example, suppose a linear feature roughly represents the quantity of cows in a field. In that case, the network may fail to extrapolate its function when an anomalous amount of cows are present, as this would be a considerable magnitude of the linear feature, which is typically deflected the most significantly. Therefore, the deflection is unprecedented and becomes uninterpretable. The activation function would result in a loss of semantic consistency. Consequently, a network seeking to preserve linear features may constrain activation magnitudes, through training, to regions where the non-linear response is approximately predictable and stable to avoid the damaging consequences of neural refractions.

Angular anisotropies fundamentally cause the refraction phenomenon. If compression and rarefaction of certain angular regions occur, linear features will be deflected in various ways. A fix for this is introducing isotropy — the initial motivation for developing the paradigm. This does not prevent compression and rarefaction of activation distributions in general, as a bias can be added to reintroduce these useful phenomena predictably. It is argued that these are only an issue when they affect linear, not affine, features in a potentially unpredictable and thus semantically uninterpretable way.

The phenomenon is eliminated from networks by rearranging *Eqn.* 5 shown in *Eqn.* 8, then applying the simplification $\|\mathbf{f}\left(\alpha\hat{x}\right)\| = \sigma\left(\alpha\right)$ in *Eqn.* 9.

$$\mathbf{f}\left(\alpha\hat{x}\right) = \|\mathbf{f}\left(\alpha\hat{x}\right)\|\,\hat{x}' \tag{8}$$

$$\mathbf{f}\left(\alpha\hat{x}\right) = \sigma\left(\alpha\right)\hat{x}' \tag{9}$$

Finally choosing $\hat{x}' = \mathbf{R}\hat{x}$ for isotropy and $\mathbf{R}\hat{x} = \mathrm{I}_n\hat{x} = \hat{x}$ for simplicity, shown in *Eqn.* 10.

$$\mathbf{f}\left(\alpha\hat{x}\right) = \sigma\left(\alpha\right)\hat{x} \tag{10}$$

In standard notation, *Eqn.* 10 can be rewritten into the *final functional form for isotropic activation functions* shown in *Eqn.* 11. This is later compared to other functional forms in *Tab.* 3.1.

$$\mathbf{f}\left(\vec{x}\right) = \sigma\left(\|\vec{x}\|\right)\hat{x} \tag{11}$$

This can be generalised as a result of rotational equivariance of the function, which is used to generalise the principle under a symmetry. This can be expressed as a condition in *Eqn.* 12, which uses a commutator bracket for convenience, with $\forall\mathbf{R} \in \mathrm{SO}\left(n\right)$. This bracket can be used to similarly define the current anisotropic discrete rotational (permutation) paradigm, by using the transform $\forall\mathbf{P} \in \mathcal{S}_n$ instead of the rotation. This may be recognised as superficially similar to equivariant neural networks, due to an analogous equivariance relation; however, the differences in both implementation and motivations are substantial and discussed further in *App.* F.

$$\left[\mathbf{R}, \mathbf{f}\right] = \left(\mathbf{R}\mathbf{f} - \mathbf{R}\mathbf{f}\right) = \vec{0} \tag{12}$$

The relation may be more familiar as $\mathbf{f}\left(\mathbf{R}\vec{x}\right) = \mathbf{R}\mathbf{f}\left(\vec{x}\right)$. This relation only applies to single-argument functions and requires generalising to more circumstances. A preliminary condition may be $\mathbf{f}\left(\mathbf{R}\vec{a}_1, \cdots, \mathbf{R}\vec{a}_N\right) = \mathbf{R}\mathbf{f}\left(\vec{a}_1, \cdots, \vec{a}_N\right)$ for $\mathbf{f}: \bigotimes_N \mathbb{R}^n \to \mathbb{R}^n$.

This introduces the general isotropic functional form for activation functions given in *Eqn.* 13. This should be a piecewise function, defined using the identity at $\vec{x} = \vec{0}$, but this is suppressed for simplicity. This functional form is $\mathcal{O}\left(n\right)$ time for $\mathbb{R}^n$. Future work is establishing a universal approximation theorem for this functional form, as this is ongoing research for the author's PhD.

$$\mathbf{f}: \mathbb{R}^n \to \mathbb{R}^n, \quad \vec{x} \mapsto \mathbf{f}\left(\vec{x}\right) = \sigma\left(\|\vec{x}\|\right)\hat{x} \tag{13}$$

This is not to be confused with the radial-basis functional form displayed in *Eqn.* 14.

$$\mathbf{f}: \mathbb{R}^n \to \mathbb{R}^n, \quad \vec{x} \mapsto \mathbf{f}\left(\vec{x}\right) = \sum_{i=1}^{N} \sigma\left(\|\vec{x} - \vec{c}_i\|\right)\hat{e}_i \tag{14}$$

The 'neural refractive problem' outlines how semantic meanings may become intertwined or ambiguous due to current functional forms consistently skewing linear features in undesirable ways. A hypothesis is developed that this may be especially detrimental for out-of-distribution activations, which are likely to be most deflected and hence most semantically corrupted. Thus, the network's generalisation may then fail excessively. It may be expected that the network produces compensatory adaptations for the phenomena, which might be narrow in the scope of their corrections. Since neural refraction is a non-linear and anisotropic phenomenon, it cannot be inverted by a single subsequent layer, potentially wasting training time on these corrections to the activation distributions due to unintended refraction.

2.2 Emergence of Linear Features and Semantic Interpolatability

As previously mentioned, symmetry-broken functional forms induce symmetry-broken representations. Thus, *approximately* discrete embedding directions are tended towards [16, 1]. Since embedded activations are often discretised, so too may be semantically meaningful directions since these are conjectured to align with these anisotropic embeddings. This generally appears to be the case [22, 23, 1]. Reversing this causality would suggest that a continuous rotational symmetry allows a continuous embedding. Functional forms would not induce direction-based symmetry breaking in their embeddings through training. It is hoped this will enable networks to acquire a more naturally continuous, and hence optimal, representation in their embedding for the given task.

However, many real-life semantics are continuums: colours, positions of objects, broad morphology, even within a single species. Representational collapse onto a single discrete semantic may lose this vital nuance. It appears a poor inductive bias to have functional forms encourage discretised representations. Isotropic functions do not prevent discrete semantics, which can be clustered through bias terms. However, they do not promote discretisation either, enabling continuous representations since they remove arbitrary basis-dependencies and limit points. Therefore, moving towards isotropy is hypothesised to encourage embeddings to be more smoothly distributed, taking on intermediate values between typically discrete linear features and substantially enlarging the expressivity and representation capacity of networks — only limited by concept interferences.

In this case, the discrete concept of 'representation capacity' may become irrelevant; each layer may express different continuous arrangements, where differing concepts are angularly suppressed and expressed in analogy to the linear features hypothesis [24]. Instead, the '*magnitude-direction hypothesis*' is proposed as a continuous extension, magnitudes indicating the amount of stimulus present, direction indicating the particular concept. Activations then populate this more continuous manifold.

This continuous-semanticity may also produce a better organised semantic map at each network layer since intermediate representations may now relate otherwise discrete features. This connectivity can bring them continuously into proximity (which 'weight locking' discussed in *Sec.* 2.3 may typically prevent). This may additionally aid researchers in comparing representational alignment between models and biology, discussed further in *App.* E.4.

Therefore, in general settings, the inductive bias of isotropy appears more appropriate as a default, due to many real-world semantics being continuous. However, anisotropy may also be a good inductive bias if universal discretisation of concepts at all scales and abstraction levels is expected. Isotropy can be thought of as adding an inductive bias that enables continuous and interpolatable semantics while retaining discrete semantics when task-necessitated — as opposed to human imposition. Hence, it generalises the discrete linear features paradigm into a more continuous setting.

2.3 Weight Locking, Optimisation Barriers and Disconnected Basins

'*Weight locking'* is a term to describe how particulary the weight parameter[3] may suffer from being stuck in local-minima found further into loss valleys, encountered only after a sufficient amount of training. This arises only due to the anisotropic functional form's *discrete* permutation symmetry. This discussion relies upon the aforementioned observations of discrete representations due to functional form symmetry breaking.

Qualitatively, this is because the semantically meaningful linear features tend to become discretised and aligned with geometric positions about the distinguished bases. Hence, any small perturbation to a parameter may misalign activations to the network's existing 'understood' semantics, once these semantics have developed. Consequently, this may largely halt further progress shortly after the formation of discrete semantic directions.

In effect, further small perturbations to the parameters may move activations from a semantically aligned to a semantically dislocated state, making the activation's meaning ambiguous. The ambiguity may negatively affect its corresponding output, forfeiting performance. This would create an emergent 'false' local minima due to the discretised semantics. This is hypothesised to be due to the discrete rotational symmetry of functional forms. Consequently, creating a plethora of architectural local

[3]Though similarly applies to a 'locking' of the bias to $\vec{0}$.

minima in the space. Only sufficiently large perturbations to a parameter may move activations between two differing semantically aligned directions.

It may also suggest that the optimisation barrier may be some function of the angular separation of the semantically meaningful directions. Angularly closer linear features would be less likely to suffer semantic dislocation if an activation is nudged towards another close semantic direction. By increasing the number of geometric positions representations occupy through varying anisotropy, this effect may be controlled. Extremising this would suggest that isotropy is beneficial since it is expected to produce continuous and interpolable representations.

This semantic dislocation is an emergent consequence of breaking the continuous symmetric forms. The dual of this argument is basin connectivity. Without continuous rotational symmetry, the loss landscape loses connectivity of many of its minima. This is a $n!$ local-minima degeneracy in weights due to the permutation symmetry of an $\mathbb{R}^n$ layer. However, enforcing the isotropy constraints results in sets of continuously connected local minima which can be smoothly transformed into one another, by corresponding parameter rotations shown in *Eqn.* 15, a consequence of *Eqn.* 11.

$$\forall \mathbf{R} \in \mathrm{SO}(n) : \underbrace{\mathbf{W}^l \mathbf{R}^\top}_{\mathbf{W}'^l} \mathbf{f}\left(\underbrace{\mathbf{R}\mathbf{W}^{l-1}}_{\mathbf{W}'^{l-1}} \vec{x} + \underbrace{\mathbf{R}\vec{b}}_{\vec{b}'}\right) = \mathbf{W}^l \mathbf{f}\left(\mathbf{W}^{l-1}\vec{x} + \vec{b}\right) \tag{15}$$

If this is downgraded to discrete rotational symmetry (i.e. permutation symmetry), then artificial optimisation barriers may reemerge in these basins. In this case, only a sufficiently large perturbation to the parameters may dislodge the network into a more optimal minima, while more minor perturbations are insufficient. Effectively, the discrete permutation symmetry may result in overlaid sets of discretised lattice solutions for the parameters, much like how it breaks the symmetry of activations through training too [1].

This is a qualitative intuition since this hypothesis remains challenging to verify until robust methods to determine semantically meaningful directions are produced. Nevertheless, in either case, steps can be taken immediately to counteract the problem, and this is to introduce isotropy to connect these minima.

3 Isotropic Implementations

This position paper argues for implementing isotropic functional forms into neural networks as a default inductive bias. Near-term adoption may be rate-limited due to the development of suitably optimal functions, particularly since *anisotropic deep learning* has a substantial head start and analogues to existing functions are not so trivial to produce. Despite this, several preliminary implementations are outlined in this section as a starting point; however, these functions are likely far from optimal and substantial research and development are required to bring isotropic deep learning into practicality. *It is hoped that the arguments of this position paper will encourage the field to begin a directed search for more suitable functions with this guiding principle of isotropy.*

Nevertheless, below is a non-exhaustive list of activation functions and some of their consequences. A summary of other functions, including optimisers, regularisers, and normalisers, is also discussed in *App.* C. In *App.* E, several suggested initial applications for Isotropy are discussed, including a modification to self-attention in *App.* E.1 and a proposal for training-time dynamic network topologies *App.* E.2.

3.1 Activation Functions

As stated, the isotropic functional form for activation functions is given in *Eqn.* 13. In *Tab.* 3.1, it is compared with the other common functional forms. It is clear in this comparison that the isotropic functional form is basis-independent and relatively simple. Further criteria, in addition to isotropy, are also a performance necessity. However, these will be explored in future work.

Beginning from this functional form, familiar analogous to elementwise functions can be developed: isotropic-Tanh, isotropic-Relu, and isotropic-Leaky-Relu. However, it is hoped that the development of the paradigm will produce further activation functions that are not just analogues of existing activation functions but exploit the novel properties of isotropy for optimal performance.

Radial Basis Form	**Elementwise Form**	**Isotropic Form**
$\mathbf{f}(\vec{x}) = \sum_{i=1}^{N} \sigma\left(\|\vec{x} - \vec{c}_i\|\right) \hat{e}_i$	$\mathbf{f}(\vec{x}) = \sum_{i=1}^{n} \sigma\left(\vec{x} \cdot \hat{e}_i\right) \hat{e}_i$	$\mathbf{f}(\vec{x}) = \sigma\left(\|\vec{x}\|\right) \hat{x}$

Isotropic-Tanh is described in *Eqn.* 16. In basis directions, $\hat{e}_i$, it is equal in function to standard elementwise-$\tanh$, as indicated by its name. It is bounded, up to a norm of one, but does not angularly saturate like standard $\tanh$, allowing activation to continue semantically shifting.

$$\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n, \quad \vec{x} \mapsto \mathbf{f}\left(\vec{x}\right) = \tanh\left(\|\vec{x}\|\right) \hat{x} \tag{16}$$

This function is reasonably cheap, computation of $r = \|\vec{x}\|$, $\tanh(r)$ and $\operatorname{sech}^2(r)$, need only be computed once (including for backward-pass) rather than per-component like the anisotropic functional forms. The vector norms are naturally constrained to $[0, 1)$ acting as an implicit normaliser. Around the origin, the transform is approximately the identity: $\lim_{r \to 0} \mathbf{J}\left(r\hat{x}\right) = \mathrm{I}_n$, justifying $\mathbf{f}\left(\vec{0}\right) = \vec{0}$, to preserve a smooth gradient. It is also globally 1-Lipschitz.

Isotropic-ReLU is shown in *Eqn.* 17, an analogue to its traditional implementation.

$$\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n, \quad \vec{x} \mapsto \mathbf{f}\left(\vec{x}; R_0\right) = \max\left(\|\vec{x}\| - R_0, 0\right) \hat{x} \tag{17}$$

Effectively, all activations are reduced by a threshold magnitude, R_0, with negative resultant magnitudes set to zero. Variations can be made to this activation function as shown in *Eqns.* 18 and 19, which include a maximum magnitude, R_∞ or do not reduce magnitudes except for below R_0, respectively. These activation functions continue to use $\mathbf{f}\left(\vec{0}\right) = \vec{0}$ property.

$$\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n, \quad \vec{x} \mapsto \mathbf{f}\left(\vec{x}; R_0, R_\infty\right) = \min\left(\max\left(\|\vec{x}\| - R_0, 0\right), R_\infty\right) \hat{x} \tag{18}$$

$$\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n, \quad \vec{x} \mapsto \mathbf{f}\left(\vec{x}; R_0\right) = \begin{cases} \vec{0} & : \quad \|\vec{x}\| < R_0 \\ \vec{x} & : \quad \|\vec{x}\| \geq R_0 \end{cases} \tag{19}$$

Isotropic-Leaky-ReLU follows a similar form to ReLU; however, it linearly rescales the magnitudes below the threshold, forming a ball of smaller rescaled magnitudes. It is displayed in *Eqn.* 20, with a small value $0 < \alpha \ll 1$.

$$\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n, \quad \vec{x} \mapsto \mathbf{f}\left(\vec{x}; \alpha, R_0\right) = \begin{cases} \alpha\vec{x} & : \quad \|\vec{x}\| < R_0 \\ \vec{x} - (1 - \alpha) R_0 \hat{x} & : \quad \|\vec{x}\| \geq R_0 \end{cases} \tag{20}$$

Isotropic-Soft-ReLU uses $\alpha = 0$, and 'Isotropic-SoftLeaky-ReLU', $\alpha \in (0, 1)$, are left in the derivative form of the radial part $\sigma'(r)$ in *Eqn.* 21, where $\phi(r)$ is a monotonically increasing function. There are very many suitable candidates fulfilling this ϕ and one may be selected which has a suitable balance between performance, computation-cost and desirable properties. Imposed is $0 < \delta < R_0$, where $\delta < R_0$ is the centre-point of the interpolation window and 2δ is the width of this window. Consequently, the function blends smoothly between two linear regions of differing scaling.

$$\sigma : \mathbb{R} \to \mathbb{R}, \quad r \mapsto \sigma'\left(r; R_0, \delta, \alpha, \phi\right) = \begin{cases} \alpha & : \quad \|\vec{x}\| \leq R_0 - \delta \\ \dfrac{\phi\left(\frac{r - R_0 + \delta}{2\delta}\right)}{\phi\left(\frac{r - R_0 + \delta}{2\delta}\right) + \phi\left(\frac{R_0 + \delta - r}{2\delta}\right)} & : \quad R_0 - \delta < \|\vec{x}\| < R_0 + \delta \\ 1 & : \quad \|\vec{x}\| \geq R_0 + \delta \end{cases} \tag{21}$$

Isotropic-Sinusoids are a possibility which do not appear to have an analogue within the current anisotropic paradigm — demonstrating a broad array of new possibilities. With the aforementioned isotropic-Relu-like functions, the networks may normalise magnitudes such that the distributions 'escape' the non-linear regions of the function, due to utilising the non-linearity constructively, which can initially be an unpredictable learning hurdle. Therefore, a function that introduces non-linearity throughout the space may be desirable. Proposed is 'isotropic-Sinusoids', which allows for distributions to be compressed, rarefied and folded (for $|\lambda_m| > 1$) in a predictable manner. It is hoped the network can utilise this for effective computation. This activation function is demonstrated in *Eqn.* 22. It may be helpful because it includes a monotonicity-violating parameter $\lambda_m \in \mathbb{R}$, enabling folding of embeddings.

$$\mathbf{f}\left(\vec{x}\right) = \vec{x} + \lambda_m \sin\left(\|\vec{x}\|\right) \hat{x} \tag{22}$$

4 Alternative View Against Isotropy

It is argued that anisotropies result in an activation distribution shift, which may be detrimental to the network's performance. This is because this inductive bias is typically introduced universally, and if no justification exists for this particular distribution, then it may be a suboptimal imposition by the network designer. However, it could also be argued that some symmetry-breaking anisotropy may be beneficial, particularly discretised semantics. By clustering parts of the activation distribution, redundant information can be usefully lost, leading to classifications that develop more quickly. In classification, one of the most common applications of deep learning, this clustering may be a suitable a priori justification for anisotropy. Therefore, introducing isotropy may limit the network's performance in this case.

Despite this, current activation functions produce anisotropies along a Cartesian grid, due to their standard basis dependence. This particular arrangement does not seem justified through classification, with Papyan et al. [25] showing a phenomenon of 'Neural Collapse' onto an equiangular tightframe for classification networks, which does not align with the standard basis. However, the works of Logan and Shepp [12] suggest that decomposition onto the standard basis can aid with the curse of dimensionality, though this is only indirectly associated with deep learning. In addition, Elhage et al. [16], demonstrate the phenomenon of '*Superposition*', which appears about the privileged basis. However, using this as support for anisotropy or studies finding local coding [] would be circular in logic, since it is a phenomenon of anisotropic networks and is affected by rotations to the anisotropy shown by Bird [1].

Nevertheless, anisotropies could be reintroduced in a more general arrangement and in such a way as to mitigate the aforementioned problems. This could consist of a more uniform distribution of anisotropic directions from which semantics could then, more densely, develop. This is outlined in *App.* B, which the author feels is one of the most crucial developments in this framework.

5 Conclusion

In this position paper, the isotropic functional form is proposed as a hypothesised better default inductive bias for deep learning, when developed. Current forms have been demonstrated in the literature to produce task-unmotivated representational artefacts [1], which this work hypothesised may limit the networks' semantic expressibility. It is further argued that the current anisotropic functional forms may have detrimental effects on performance and learning, through the 'neural refractive problem', 'discrete semantics' and 'weight locking'. Hence, removing such constraints from the model is also argued to unconstrain the representations from any particular basis. The network is then expected to produce a more natural activation representation based upon task necessities, rather than by human-imposed functional forms. It is expected to improve the semantic structure and produce high-capacity embeddings. It is proposed that the breadth of reformulation may constitute a new direction and branch of deep learning: *Isotropic deep learning* to distinguish it from existing paradigms.

The adjustment to functional forms is through promoting the existing discrete rotational symmetry (permutation symmetry) of modern deep learning to a continuous special-orthogonal symmetry, or perhaps the orthogonal symmetry. This has substantial consequences for the form of almost every function in modern-day deep learning and a connected roadmap for future work in these directions is proposed, especially within the appendices. This position paper showcases several examples of isotropic and quasi-isotropic functions as a starting point. Yet, these are analogues of anisotropic functions and not expected to be inherently optimal or better just because they display superficial similarity to existing functions.

Instead, it is the author's position that this change to isotropic deep learning is generally advantageous, but may need substantial time for development as a paradigm, such that better optimised implementations are discovered which suitably leverage the isotropy. Therefore, empirical work on these placeholder functions will be presented in the following papers to not distract from the primary motivation for this shift to Isotropic deep learning. The proposed ideas are hoped to stimulate the communities' interest in beginning a directed search for such isotropic functions, which will hopefully bring the paradigm into widespread applicability and adoption.

References

[1] George Bird. The spotlight resonance method: Resolving the alignment of embedded activations. In *Second Workshop on Representational Alignment at ICLR 2025*, 2025. URL `https://openreview.net/forum?id=alxPpqVRzX`.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL `https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf`.

[3] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. URL `http://jmlr.org/papers/v15/srivastava14a.html`.

[4] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions, 2014. URL `https://arxiv.org/abs/1409.4842`.

[5] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. URL `https://arxiv.org/abs/1409.1556`.

[6] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015. URL `https://arxiv.org/abs/1502.03167`.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. URL `https://arxiv.org/abs/1512.03385`.

[8] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL `https://arxiv.org/abs/1412.6980`.

[9] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks, 2018. URL `https://arxiv.org/abs/1608.06993`.

[10] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020. URL `https://arxiv.org/abs/1905.11946`.

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL `https://arxiv.org/abs/1706.03762`.

[12] Benjamin F Logan and Larry A Shepp. Optimal reconstruction of a function from its projections. 1975.

[13] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[14] Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions, 2017. URL `https://arxiv.org/abs/1710.05941`.

[15] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus), 2023. URL `https://arxiv.org/abs/1606.08415`.

[16] Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition, 2022. URL `https://arxiv.org/abs/2209.10652`.

[17] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

[18] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991. ISSN 0893-6080. doi: https://doi.org/10.1016/0893-6080(91)90009-T. URL `https://www.sciencedirect.com/science/article/pii/089360809190009T`.

[19] Chris Olah. Neural networks, manifolds, and topology — colah.github.io. `https://colah.github.io/posts/2014-03-NN-Manifolds-Topology/`, April 2014. [Accessed 15-05-2025].

[20] Peter Foldiak and Dominik Endres. Sparse coding, Jan 2008. URL `http://www.scholarpedia.org/article/Sparse_coding#:~:text=Sparse%20coding%20is%20the%20representation,subset%20of%20all%20available%20neurons`.

[21] David Lowe and D Broomhead. Multivariable functional interpolation and adaptive networks. *Complex systems*, 2(3):321–355, 1988.

[22] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization, Nov 2017. URL `https://distill.pub/2017/feature-visualization/`.

[23] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations, 2017. URL `https://arxiv.org/abs/1704.05796`.

[24] Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. Linear algebraic structure of word senses, with applications to polysemy. *Transactions of the Association for Computational Linguistics*, 6:483–495, 2018.

[25] Vardan Papyan, X. Y. Han, and David L. Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117 (40):24652–24663, September 2020. ISSN 1091-6490. doi: 10.1073/pnas.2015509117. URL `http://dx.doi.org/10.1073/pnas.2015509117`.

[26] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization?, 2019. URL `https://arxiv.org/abs/1805.11604`.

[27] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016. URL `https://arxiv.org/abs/1607.06450`.

[28] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization, 2017. URL `https://arxiv.org/abs/1607.08022`.

[29] Yuxin Wu and Kaiming He. Group normalization, 2018. URL `https://arxiv.org/abs/1803.08494`.

[30] Sergey Ioffe. Batch renormalization: Towards reducing minibatch dependence in batch-normalized models, 2017. URL `https://arxiv.org/abs/1702.03275`.

[31] Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

[32] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL `https://proceedings.mlr.press/v9/glorot10a.html`.

[33] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(61):2121–2159, 2011. URL `http://jmlr.org/papers/v12/duchi11a.html`.

[34] Matthew D. Zeiler. Adadelta: An adaptive learning rate method, 2012. URL `https://arxiv.org/abs/1212.5701`.

[35] Aston Zhang, Zachary C Lipton, Mu Li, and Alexander J Smola. *Dive into deep learning*. Cambridge University Press, 2023.

[36] Jiaxuan Wang and Jenna Wiens. Adasgd: Bridging the gap between sgd and adam, 2020. URL https://arxiv.org/abs/2006.16541.

[37] Charles George Broyden. The convergence of a class of double-rank minimization algorithms 1. general considerations. *IMA Journal of Applied Mathematics*, 6(1):76–90, 1970.

[38] Roger Fletcher. A new approach to variable metric algorithms. *The computer journal*, 13(3): 317–322, 1970.

[39] Donald Goldfarb. A family of variable-metric methods derived by variational means. *Mathematics of computation*, 24(109):23–26, 1970.

[40] David F Shanno. Conditioning of quasi-newton methods for function minimization. *Mathematics of computation*, 24(111):647–656, 1970.

[41] Jorge Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782, 1980.

[42] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9: 1735–80, 12 1997. doi: 10.1162/neco.1997.9.8.1735.

[43] Ward Cheney and David Kincaid. Linear algebra: Theory and applications. *The Australian Mathematical Society*, 110:544–550, 2009.

[44] G. W. Stewart. The efficient generation of random orthogonal matrices with an application to condition estimators. *SIAM Journal on Numerical Analysis*, 17(3):403–409, 1980. ISSN 00361429. URL http://www.jstor.org/stable/2156882.

[45] Francesco Mezzadri. How to generate random matrices from the classical compact groups, 2007. URL https://arxiv.org/abs/math-ph/0609050.

[46] Kai Hu and Barnabas Poczos. Rotationout as a regularization method for neural network, 2020. URL https://openreview.net/forum?id=r1e7M6VYwH.

[47] Wallace Givens. Computation of plain unitary rotations transforming a general matrix to triangular form. *Journal of the Society for Industrial and Applied Mathematics*, 6(1):26–50, 1958. doi: 10.1137/0106004. URL https://doi.org/10.1137/0106004.

[48] Adelaide P Yiu, Valentina Mercaldo, Chen Yan, Blake Richards, Asim J Rashid, Hwa-Lin Liz Hsiang, Jessica Pressey, Vivek Mahadevan, Matthew M Tran, Steven A Kushner, Melanie A Woodin, Paul W Frankland, and Sheena A Josselyn. Neurons are recruited to a memory trace based on relative neuronal excitability immediately before training. *Neuron*, 83(3):722–735, August 2014.

[49] Lingxuan Chen, Kirstie A Cummings, William Mau, Yosif Zaki, Zhe Dong, Sima Rabinowitz, Roger L Clem, Tristan Shuman, and Denise J Cai. The role of intrinsic excitability in the evolution of memory: Significance in memory allocation, consolidation, and updating. *Neurobiol. Learn. Mem.*, 173(107266):107266, September 2020.

[50] John Bridle. Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 2. Morgan-Kaufmann, 1989. URL https://proceedings.neurips.cc/paper_files/paper/1989/file/0336dcbab05b9d5ad24f4333c7658a0e-Paper.pdf.

[51] Ilia Sucholutsky, Lukas Muttenthaler, Adrian Weller, Andi Peng, Andreea Bobu, Been Kim, Bradley C. Love, Christopher J. Cueva, Erin Grant, Iris Groen, Jascha Achterberg, Joshua B. Tenenbaum, Katherine M. Collins, Katherine L. Hermann, Kerem Oktar, Klaus Greff, Martin N. Hebart, Nathan Cloos, Nikolaus Kriegeskorte, Nori Jacoby, Qiuyi Zhang, Raja Marjieh, Robert Geirhos, Sherol Chen, Simon Kornblith, Sunayana Rane, Talia Konkle, Thomas P. O'Connell, Thomas Unterthiner, Andrew K. Lampinen, Klaus-Robert Müller, Mariya Toneva, and Thomas L. Griffiths. Getting aligned on representational alignment, 2024. URL https://arxiv.org/abs/2310.13018.

[52] Alex H Williams, Erin Kunz, Simon Kornblith, and Scott Linderman. Generalized shape metrics on neural representations. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 4738–4750. Curran Associates, Inc., 2021. URL `https://proceedings.neurips.cc/paper_files/paper/2021/file/252a3dbaeb32e7690242ad3b556e626b-Paper.pdf`.

[53] Taco S. Cohen and Max Welling. Group equivariant convolutional networks, 2016. URL `https://arxiv.org/abs/1602.07576`.

[54] Taco S. Cohen and Max Welling. Steerable cnns, 2016. URL `https://arxiv.org/abs/1612.08498`.

[55] Daniel E. Worrall, Stephan J. Garbin, Daniyar Turmukhambetov, and Gabriel J. Brostow. Harmonic networks: Deep translation and rotation equivariance, 2017. URL `https://arxiv.org/abs/1612.04642`.

[56] Taco S. Cohen, Mario Geiger, Jonas Koehler, and Max Welling. Spherical cnns, 2018. URL `https://arxiv.org/abs/1801.10130`.

[57] Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges, 2021. URL `https://arxiv.org/abs/2104.13478`.

[58] Shan Carter, Zan Armstrong, Ludwig Schubert, Ian Johnson, and Chris Olah. Activation atlas, Jun 2020. URL `https://distill.pub/2019/activation-atlas/`.

[59] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65 6:386–408, 1958. URL `https://api.semanticscholar.org/CorpusID:12781225`.

[60] R. Quian Quiroga, L. Reddy, G. Kreiman, C. Koch, and I. Fried. Invariant visual representation by single neurons in the human brain. *Nature*, 435(7045):1102–1107, Jun 2005. ISSN 1476-4687. doi: 10.1038/nature03687. URL `https://doi.org/10.1038/nature03687`.

[61] Katharine M Cammack, Thomas R Reppert, and Denise R Cook-Snyder. The simpsons neuron: A case study exploring neuronal coding and the scientific method for introductory and advanced neuroscience courses. *J Undergrad Neurosci Educ*, 20(1):C1–C10, December 2021.

[62] G Kreiman, C Koch, and I Fried. Category-specific visual responses of single neurons in the human medial temporal lobe. *Nat Neurosci*, 3(9):946–953, September 2000.

[63] Charles Sherrington. *Man on his nature*. 1940.

[64] Daniel J Graham and David J Field. Sparse coding in the neocortex. *Evolution of nervous systems*, 3:181–187, 2006.

[65] Chris Olah, Arvind Satyanarayan, Ian Johnson, Shan Carter, Ludwig Schubert, Katherine Ye, and Alexander Mordvintsev. The building blocks of interpretability. `https://distill.pub/2018/building-blocks/`, March 2018. [Accessed 02-08-2024].

[66] Charles G. Gross. Genealogy of the "grandmother cell". *The Neuroscientist*, 8(5):512–518, 2002. doi: 10.1177/107385802237175. URL `https://doi.org/10.1177/107385802237175`. PMID: 12374433.

[67] Charles E Connor. Neuroscience: friends and grandmothers. *Nature*, 435(7045):1036–1037, June 2005.

[68] Jerzy Konorski. Learning, perception, and the brain: Integrative activity of the brain. an interdisciplinary approach. *Science*, 160(3828):652–653, 1968. doi: 10.1126/science.160.3828.652. URL `https://www.science.org/doi/abs/10.1126/science.160.3828.652`.

[69] H B Barlow. Summation and inhibition in the frog's retina. *J Physiol*, 119(1):69–88, January 1953.

[70] Simon Thorpe. Local vs. distributed coding. *Intellectica*, 8(2):3–40, 1989.

[71] J. Y. Lettvin, H. R. Maturana, W. S. McCulloch, and W. H. Pitts. What the frog's eye tells the frog's brain. *Proceedings of the IRE*, 47(11):1940–1951, 1959. doi: 10.1109/JRPROC.1959.287207.

[72] H. K. Hartline. The response of single optic nerve fibers of the vertebrate eye to illumination of the retina. *American Journal of Physiology-Legacy Content*, 121(2):400–415, 1938. doi: 10.1152/ajplegacy.1938.121.2.400. URL `https://doi.org/10.1152/ajplegacy.1938.121.2.400`.

[73] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, Dec 1943. ISSN 1522-9602. doi: 10.1007/BF02478259. URL `https://doi.org/10.1007/BF02478259`.

[74] Frank Rosenblatt. *The perceptron: a theory of statistical separability in cognitive systems (Project Para)*. Cornell Aeronautical Laboratory, 1958.

[75] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[76] Andrew L Maas, Awni Y Hannun, Andrew Y Ng, et al. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3. Atlanta, GA, 2013.

[77] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks, 2013. URL `https://arxiv.org/abs/1311.2901`.

[78] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks, 2014. URL `https://arxiv.org/abs/1312.6199`.

[79] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization, 2015. URL `https://arxiv.org/abs/1506.06579`.

A Taxonomy of Functional Forms

Isotropic deep learning is centred around the equivariance to special-orthogonal group actions. This is formalised through an equivariance relation, such as in *Eqn.* 23 defining a functional form for $\mathbf{f}:\mathbb{R}^n\rightarrow\mathbb{R}^n$.

$$\forall\mathbf{R}\in\mathrm{SO}\left(n\right):\left[\mathbf{R},\mathbf{f}\right]=0 \tag{23}$$

It has already been shown how current deep learning's functional forms can be connected through a similar relations, as an equivariance of functional forms to the permutation group, as shown in *Eqn.* 24.

$$\forall\mathbf{P}\in\mathcal{S}_n:\left[\mathbf{P},\mathbf{f}\right]=0 \tag{24}$$

Similarly, quasi-isotropic functional forms (though not the specific instance in *Eqn.* 27) can be connected through a discrete rotational symmetry group ψ_n, with set cardinality $|\mathcal{S}_n|\ll|\psi|$ and extending this to $\mathcal{S}_n\subset\psi$. This can be represented through *Eqn.* 25.

$$\forall\mathbf{\Psi}\in\psi_n:\left[\mathbf{\Psi},\mathbf{f}\right]=0 \tag{25}$$

This taxonomic approach could be used as both a unifying perspective and generalised approach to deep learning. Considering the standard permutation, a specific discrete-rotational, continuous-rotational and general linear symmetries for an $\mathbb{R}^n$ space associated with a single layer, can be constructed such that they are unified under a group heirarchy: $\mathcal{S}_n\subset\psi_n\subset\mathrm{SO}\left(n\right)\subset\mathrm{GL}\left(n\right)$. Applying each of these to functional forms produces, current deep learning, quasi-isotropic deep learning, isotropic deep learning and linear approximators respectivly. Hence, several forms of machine learning can be said to be unified together under such approach, including linear approximators.

Hence, it is natural to generalise this functional form equivariance relationship more broadly. For a symmetry family $\mathcal{G}$, one could impose a functional form equivariance as shown in *Eqn.* 26. In general this would need to be a task-motivated inductive bias. This could produce a functional form taxonomy for deep learning approaches.

$$\forall\mathbf{G}\in\mathcal{G}:\left[\mathbf{G},\mathbf{f}\right]=0 \tag{26}$$

These symmetries can all be associated to symmetries of directed graphs, endowed with continuous activations, as briefly discussed in *App.* F. In effect, one can construct an directed graph which represents the neuron connectivity of an arbitrary architecture. If nodes within this graph are organised and divided into groups of neuron layers, then the continuous activations assigned to these nodes can be said to form an $\mathbb{R}^n$ activation space. Prior to parameter initialisation and specific functional form implementation, these layers could be considered to feature symmetries of their $\mathbb{R}^n$ space. Choosing such a symmetry group, which leaves these spaces invariant, can then be used with the general equivariance relation of *Eqn.* 26 to produce functional forms and parameter initialisations obeying the symmetry. It proposed that this could be a powerful unifying direction for deep learning, which is may be worth considerable exploration and may generate and categories what could be considered novel forms of deep learning.

B Quasi-Isotropic Functional Forms

A middleground, balancing the aforementioned problems whilst enabling representation compression not just through the bias, is to relax the hard isotropy condition and introduce slight symmetry breaking in many directions. This can be achieved using many small perturbations to the direction unit vector only, producing a softer symmetry breaking. Then the network has many distinguished vectors, a subset with which it may align its representations in a task-dependent manner. Therefore, it does not favour a particular basis, but still introduces some desirable consequences of anisotropy, for problems such as classification. If one further restricts the functions from featuring dynamic refraction, it limits detrimental anisotropic effects.

One method is to apply a non-linearity based on rounding the vector's directions. This is shown in *Eqn.* 27, where $[\cdot]$ indicates the rounding operation and $\phi\left(\vec{x}\right) \neq \vec{x}$. In fact, the anisotropic perturbation may be implemented as simply as: $\phi\left(\vec{x}\right) = \beta\vec{x}$ for $\beta \neq 1$. The overall angular term is approximately unit-normed, but can be trivially modified to be exactly norm-1.

$$\Phi\left(\hat{x};\alpha\right) = \frac{[\alpha\hat{x}]}{\alpha} + \phi\left(\hat{x} - \frac{[\alpha\hat{x}]}{\alpha}\right) \approx \hat{x} \tag{27}$$

This produces a quasi-isotropic functional form shown in *Eqn.* 28, with an isotropy-breaking parameter α. Slight anisotropic refraction is added, independent of magnitude, so it is predictable and thus extrapolatable to the network. Due to the angular rarefaction and compression by the proposed non-linearity, representation over- and underdensities may then occur, where semanticity may begin to be assigned. However, for $\alpha \to \infty$, isotropy is continuously reintroduced and could be an optimisable parameter.

$$\mathbf{f}\left(\vec{x}\right) = \sigma\left(\|\vec{x}\|\right)\Phi\left(\hat{x}\right) \tag{28}$$

C Beyond Isotropic Activation Functions

In this position paper, activation functions are predominantly explored, showing how the isotropic functional form opens up a wealth of new functions to explore and design. However, the isotropic deep learning principles are not limited to activation functions; a network is not isotropic until all constituent functions are reformulated using the provided conditions. Despite [1]'s results empirically demonstrating representational anisotropies due to activation functions, it is also hypothesised that other transforms also incur a similar basis-dependent representational alignment. This hypothesis is used to broaden the scope of isotropic deep learning, encouraging an overhaul to almost all functional forms within modern-day deep learning.

A non-exhaustive list of functional forms requiring reformulation are: initialisers, normalisers, regularisers, operations, optimisers and gradient clipping. This section provides a brief summary of some of these, highlighting anisotropies present in current forms. Isotropic reformulations will follow. These directions are so far incomplete, and future work will be required to both develop and add to these functional forms with empirical benchmarking.

C.1 Normalisers:

Normalisation of the activations within deep learning is implemented frequently. This may be to prevent exponential gradient growth or decay, produce faster solution convergence, smooth the loss landscape [26], or initially thought to primarily aid in reducing internal covariate shift [6]. In this section, their Isotropic properties are analysed. Throughout this section, it is essential to stress that it is functional form anisotropy defined over an arbitrary distribution as opposed to anisotropy in any particular activation distribution — the latter is expected to still occur in isotropic networks.

There have been many proposed normalisation techniques, including but not limited to "Batch Normalisation" by Ioffe and Szegedy [6], "Layer Normalisation" by Ba et al. [27], "Instance Normalisation" by Ulyanov et al. [28] and "Group Normalisation" by Wu and He [29]. The mathematics of each of these will be briefly outlined below. Similar approaches can be taken for other normalisations not listed. When the functions do not meet the isotropic requirements, this is not a suggestion that the functions are inherently suboptimal, as they were designed for differing purposes than Isotropic deep learning is. This is simply an analysis of whether current normalisations can be classified as Isotropic deep learning methods, and whether they introduce further incidental inductive biases in the representations besides the intended distributional shifts.

Batch-normalisation [6] consists of normalising every element of the activation vector based upon mean and standard deviation statistics across a (mini-)batch, to produce a consistent activation distribution to train from. This was initially argued to reduce covariate shift, which otherwise may reduce learning due to saturation of activation functions, whilst acting as a regulariser due to noise within the statistics, and enabling a higher learning rate alongside less constraints on parameter initialisation. Expressed in multivariate form, *Eqn.* 29 shows the batch-normalisation operation, with trainable parameters $\vec{\gamma}$ and $\vec{\beta}$, where $\mathbb{E}_B\left[\cdots\right]$ indicates an average over the batch.

$$\mathbf{f}\left(\vec{x}\right)=\sum_{i=1}^{N}\left(\vec{\gamma}\cdot\hat{e}_i\right)\frac{\vec{x}\cdot\hat{e}_i-\mathbb{E}_B\left[\vec{x}\cdot\hat{e}_i\right]}{\sqrt{\epsilon+\mathbb{E}_B\left[\left(\vec{x}\cdot\hat{e}_i-\mathbb{E}_B\left[\vec{x}\cdot\hat{e}_i\right]\right)^2\right]}}\hat{e}_i+\vec{\beta} \tag{29}$$

Despite the appearance of many basis terms, $\hat{e}_i$, under strong assumptions, Batch normalisation could be isotropic. For example, if the activation distributions were a perfect normal distribution, with $\vec{\gamma}=\alpha\vec{1}$ and $\vec{\beta}=\vec{0}$ then batch-normalisation across multiple activations would produce an isotropic distribution, due to the product of zero-mean normal distributions being isotropic. However, even within isotropic deep learning, the activations are not expected to be isotropic nor constrained to be a zero-mean normal distribution, so a per-coordinate rescaling would counterintuitivly incur anisotropy. This approach also makes Batch-normalisation sensitive to the mini-batch size and *not* isotropic.

The dependence on mini-batch can be reformulated in an alternative manner. One can represent batch normalisation through a matrix of activations: $\mathbf{X}\in\mathbb{R}^{b\times n}$, for batch size b and number of features n. The mean-statistic consequently becomes $\vec{\mu}_j=\mathbf{X}_{ij}\vec{1}_i$, in the standard basis and using

Einstein summation convention for compactness. With subtraction of the mean, $\mathbf{X}'_{ij} = \mathbf{X}_{ij} - \vec{\mu}_j \vec{1}_i$, and then normalisation using $\sigma_j^2 = \mathbf{X}'_{ij}\mathbf{X}'_{ij} + \epsilon$. One can see that the resultant manifold of possible representations is first constrained to a plane orthogonal to $\vec{1}_i$, a $\mathbb{R}^{(b-2)\times n}$ space, then normalisation approximatly produces an $\mathcal{S}^{(b-2)} \times \mathbb{R}^n$ space (if ϵ is ignored), embedded in the original $\mathbb{R}^{b\times n}$ space. The parameters $\vec{\gamma}$ amd $\vec{\beta}$ produce a different embedding. Consequently, one can see that a single sample activation space is preserved at $\mathbb{R}^n$, but across the batch the space is $\mathcal{S}^{(b-2)} \times \mathbb{R}^n$, showing how the stochasticity in the batch sampling, results in a change to batched-activation space, producing the regularising stochasticity affecting gradients. Despite batch normalisation being a function $\mathbf{f}: \mathbb{R}^{b\times n} \to \mathbb{R}^{b\times n}$ for its extrinsic space, the effect on its intrinsic manifolds will be denoted $\mathbf{f}: \mathbb{R}^{b\times n} \to \mathcal{S}^{b-2} \times \mathbb{R}^n \hookrightarrow \mathbb{R}^{b\times n}$.

Layer-norm [27] consists of similar approach in normalising activations; however, the statistics are calculated in a different mannor. Layer-norm acts sample-wise within a batch, so removes the dependence on the mini-batch stochasticity and batch-size. Alongside other properties, this has made it a popular choice of normaliser. It is given by the function in *Eqn.* 30, where $\mathbb{E}_i[\cdots]$ is the expectation over index i occuring in the basis vectors $\hat{e}_i$.

$$\mathbf{f}(\vec{x}) = \sum_{i=1}^{N} \left(\gamma \frac{\vec{x}\cdot\hat{e}_i - \mathbb{E}_i[\vec{x}\cdot\hat{e}_i]}{\sqrt{\epsilon + \mathbb{E}_i\left[(\vec{x}\cdot\hat{e}_i - \mathbb{E}_i[\vec{x}\cdot\hat{e}_i])^2\right]}} + \beta \right) \hat{e}_i \qquad (30)$$

Despite this function producing $\mathbf{f}: \mathbb{R}^n \to \mathbb{R}^n$, the intrinsic dimensionality of an $\mathbb{R}^n$ dimensional object is mapped to an $(n-2)$-sphere, $\mathcal{S}^{n-2}$, embedded within an $\mathbb{R}^n$ space: $\mathbf{f}: \mathbb{R}^n \to \mathcal{S}^{n-2} \hookrightarrow \mathbb{R}^n$. This is similar to batch-norm, but directly affects the representational geometry of individual samples. This loses two degrees-of-freedom in the *activation* vectors which are not reintroduced simply by the two degrees-of-freedom in *parameters*: γ and β. The latter affect the manifold globally by altering the embedding: $\mathbf{f}(\vec{x}) \cdot \vec{1} = n\beta$ (an equation for a hyperplane with normal $\hat{1}$) and $\left\|\mathbf{f}(\vec{x}) - \beta\vec{1}\right\|_2 = \gamma$ (constraining the vector norm within the hyperplane). Two parameter degrees-of-freedom do therefore not constitute a replacement of representational degrees-of-freedom. By definition, this cannot be Isotropic due to a breakdown of rotational equivariance when rotating into plane's normal direction. Furthermore, the implementation of β and γ are basis-dependent.

The success of layer-norm could be reinterpreted as a map with the loss of information in two directions, which can be optimised to remove redundant data. This may explain an extra benefit of layer-norm in classification networks. This principle could be generalised by reinterpreting layer-norm as a sequence of three isotropic layers producing a small bottleneck: $\mathbb{R}^n \to \mathbb{R}^{n-2} \to \mathbb{R}^n$. If an isotropic activation function is applied on $\mathbb{R}^{n-2}$, such as $\mathbf{f}(\vec{x}) = \vec{x}/\sqrt{\vec{x}\cdot\vec{x}} = \hat{x}$, or the implicit normalisation properties of isotropic-$\tanh$, then one can largely recreate the form of layer-norm architecturally. However, the weight and bias parameters of $\mathbb{R}^{n-2} \to \mathbb{R}^n$ layer now act as a basis-free substitute for γ and β. This basis-independence is essential under the isotropic framework's approach to reducing unintentional representational inductive biases. One can also further constrict the bottleneck if desirable. This will be termed *isotropic layer-norm*, and could be used as a drop-in replacement within existing classification models which may especially benefit from removing redundant directions.

Instance normalisation [28], arose in convolutional neural networks for style transfer. It is defined in *Eqn.* 31, where the basis-dependence is expressed through indices for brevity, with notation aligning with Ulyanov et al. [28].

$$\mathbf{f}(\vec{x})_{tijk} = \gamma_i \frac{x_{tijk} - [\mathbb{E}_{hw}[x_{tihw}]]_{ti}}{\sqrt{\epsilon + \left[\mathbb{E}_{hw}[x_{tihw} - [\mathbb{E}_{hw}[x_{tihw}]]_{ti}]^2\right]_{ti}}} + \beta_i \qquad (31)$$

Analysing the Isotropic properties of this normalisation is made difficult due to the use of convolution. Isotropy is argued to arise from architectural symmetries of the model, discussed briefly in *Apps.* F and A. This approach would suggest that the isotropic equivariance relation would be restricted to a subset of rotations within a linear subspace, indexed per spatial pixel, of the entire activation space. In effect, for a given spatial location, specific height and width index, there is a resultant vector in $\mathbb{R}^C$, for C channels, which the rotations equivariance applies to. This would suggest that Instance normalisation, when $\beta_i = 0$ and $\gamma_i = 1$, is isotropic for convolutional neural networks. However, there may be some ambiguity on the restrictions of β_i and γ_i for isotropy, since this could be interpreted as a single linear layer with restricted parameters. However, such restrictions amount to a linear transform on the aforementioned subspace of $\mathbf{W} = \mathrm{diag}\left(\vec{\gamma}\right)$, which is a standard-basis aligned restriction on the parameters, indicating this is truly anisotropic.

Group normalisation [29], can be intuited as a middleground between Layer normalisation and Instance normalisation, since it is argued that channels are not statistically independent. Hence, the statistics are computed along these groups of channels. Since it is a middleground between both Instance normalisation and Layer normalisation, arguments for its anisotropy are trivially extended.

C.1.1 Potential Isotropic Normalisers:

Alongside the isotropic layer-norm proposed above, several other formulations can be produced. These consist of a batch-norm-like approach, to be termed 'Chi-normalisation', a time-like normalisation and a simpler layer-norm than above. Normalisations are the most analogous to activation functions in terms of symmetry equivariance constraints: $[\mathbf{R}, \mathbf{f}] = 0$, per $\mathbf{f}: \mathbb{R}^n \to \mathbb{R}^n$. This makes them a reasonably straight forward candidate for reformulation. Despite this, due to the presence of parameters, the equivariance must apply across their initialisation distributions too. This is discussed further in *App.* C.2. The full possibilities of normalisation within isotropic deep learning is not limited to these few suggestions.

Isotropic deep learning is proposed to make continuous directions meaningful and magnitudes indicate the degree of that meaning. Large growing magnitudes may destablise training in the networks, so it is sensible to normalise over the magnitude — analogous to existing normalisation techniques.

A simplified layer-normalisation than the one stated above is simply: $\mathbf{f}\left(\vec{x}\right) = \gamma\hat{x} + \vec{\beta}$, where $\hat{x}$ is the unit-normalised vector of $\vec{x}$. Though, the increased parameters and collapse of redundant directions may make the prior form better.

Second, is term a 'Chi-normaliser' and is computed over a batch, which is acknowledged to have its drawbacks. After a linear transform is performed on prior activations, it could be assumed that an activation distribution is approximatly a multivariate normal distribution. Therefore, one could subtract the mean of the distribution across a batch, and then normalise the magnitudes. The magnitudes of an n-dimensional, standard multivariate normal distribution follow the Chi distribution, shown in *Eqn.* 32 with its mean and variance, for $x \geq 0$ and Γ being the gamma-function.

$$f(x;n) = \frac{x^{n-1}e^{-\frac{x^2}{2}}}{2^{\frac{n}{2}-1}\Gamma\left(\frac{n}{2}\right)}, \quad \mathbb{E}\left(f(x;n)\right) = \sqrt{2}\frac{\Gamma\left(\frac{n+1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)}, \qquad \mathrm{Var}\left(f(x;n)\right) = n - 2\mu. \tag{32}$$

Therefore, the mean and standard deviation statistics could be estimated over the batch, and the magnitude distribution normalised using these. Then the scale of the resultant chi-like distribution must be made standard. This is achieved by dividing by the mean of the distribution. The extra factor of $\sqrt{n}$, is then absorbed into the scaling parameter γ. For batched activations $\mathbf{X} \in \mathbb{R}^{b\times n}$, with elements indexed and using Einstein summation convention, the Chi-normalisers is given by *Eqn.* 33 and *Eqn.* 34. Ideally *Eqn.* 33, should be removed since it can break isotropy through the statistic and consequently change activation directions.

$$\mathbf{X}'_{ij} = \mathbf{X}_{ij} - \frac{\mathbf{X}_{sj}\mathbf{1}_{si}}{n} \tag{33}$$
$$\mathbf{f}\left(\mathbf{X}'\right) = \frac{\gamma\mathbf{X}'_{ij}}{\sqrt{\|\mathbf{X}'_{sm}\mathbf{X}'_{sm} + \epsilon\|}} + \vec{1}_i\vec{\beta}_j \tag{34}$$

One may reshape the width of the chi-like distribution, using a further statistic, such that it follows more closely to the intended standard chi-distribution. Though further normalisations may be non-trivial due to ensuring positive domained operations, which subtraction breaks. This direction change would be highly undesirable, but the magnitudes may be rectified: $\max\left(0, x\right)$.

Finally, one may estimate magnitude-normalising statistics by a moving average over training steps [30]. This is a time-like normalisation, which could also be computed over the batch if desirable. This can arguably result in stale statistics, due to the changing representations by optimisation, but results may differ from previous attempts by normalising strictly the magnitude. It is *not* proposed that gradients should be propagated through time; statistics could be assumed to be approximatly constants for efficiency. Storing such statistics is neglible in memory requirements, since they would be only scalars per normalisation layer. These can be updated, such that they follow an exponential weight average. Similar stale statistics are found to be beneficial in optimisers such as in *App.* C.3. Consequently, more recent activations would contribute more to estimating the statistics, reducing the stale-statistics problem. This direction would need considerably further exploration.

These preliminary normalisers could serve as a starting point to test and build new isotropic normalisers from.

C.2 Initialisers:

Initialisation of parameters may also have unintended consequences for representational geometry. At an extreme, if one initialised all weights as rank-1 matrices, then we may expect both worse performing and slower learning networks, due to lower expressivity and gradient flow difficulties. Therefore, initialisation considerations are essential for representational geometric inductive biases.

To inform the Isotropic constraints on initialisation, requires analysis of how the isotropic symmetry is said to 'derive' from the architecture. This is discussed briefly within *Apps.* F and A. In effect, under the construction discussed, an arbitrary architecture is said to be invariant to continuous rotations of its nodes, after fields are assigned to them, but before functional forms are introduced to the architecture. The choice of symmetry then informs the functional form choices.

The set of n nodes with field, $\mathbb{F}$, assigned to them, is said to form a $\mathbb{F}^n$ representation space. This space is chosen to be invariant to $\mathrm{SO}\left(n\right)$ group actions. This manifests as $[\mathbf{R}, \mathbf{f}] = 0$ constraints on functions within the space. Following this approach, the initialisers producing parameters which transform between sequential spaces should too be invariant to rotations of the underlying spaces. Therefore, a relation such as $\forall \mathbf{R} \in \mathrm{SO}\left(n\right)$ and $\forall \mathbf{T} \in \mathrm{SO}\left(m\right)$ then: $\mathbf{R}\mathbb{P}_{\mathbf{W}}\mathbf{T} = \mathbb{P}_{\mathbf{W}}$ is desirable, for parameter $\mathbf{W} \in \mathbb{F}^{n \times m}$ and probability distribution $\mathbb{P}_{\mathbf{W}}$ over $\mathbb{F}^{n \times m}$. In this case, the initialiser is invariant to the same transforms as the nodes, and does not induce anisotropic artefacts by an unintended inducive bias.

Many such initialisers could be formulated, in particular two will be suggested: rectangular-orthogonal [31] and multivariate normal. If the product measure for $\mathbb{R}^{nm}$, is composed from independent standard normal distributions, $\mathbb{R}$, then the overall distribution is rotationally symmetric, meeting requirements. Orthogonally, one could draw two elements *uniformly* from groups $\mathrm{SO}\left(n\right)$ and $\mathrm{SO}\left(m\right)$ and use their $U \in \mathbb{R}^{n \times n}$ and $V \in \mathbb{R}^{m \times m}$ matrix representations respectivly. These distributions could then be joined together using a $\Lambda \in \mathbb{R}^{m \times n}$ matrix drawn uniformly from rectangular-diagonal matrices — however, this could be restricted to the δ_{nm} matrix, an identity matrix padded with appropriate zeros to be $n \times m$ in shape. The distributions joined using $\mathbf{W} = U \Lambda V$ would then be isotropic. This list of possible initialisers is not exhaustive.

This leaves some free parameters available within these isotropic distributions: σ for the multivariate normal and a 'gain' factor for orthogonal. These could be chosen using a similar approach to Glorot and Bengio [32] for isotropic deep learning. This could be ensuring the expectation of magnitudes between layers stays relativly constant and an appropriate constraint on gradients in isotropic networks. Future work would need to establish how these values should be set in-line with the principles of Isotropic deep learning.

C.3 Optimisers:

Both stochatic gradient descent and momentum variants do not contain a basis dependence in their formulations; however, many adaptive optimisers do. For example, Adagrad by Duchi et al. [33] shown in *Eqn.* 35, AdaDelta by Zeiler [34] shown in *Eqn.* 36 and ADAM by Kingma and Ba [8] shown in *Eqn.* 37, all use approximations which are basis-dependent. A basis dependence within the optimiser is likely to optimise preferentially in specific directions, producing an anisotropic effect on parameters. This anisotropic preference in parameters then shapes the distribution of activations through the network. Hence, it is expected that anisotropy in an optimiser results in a form of representational bias. Since this is an indirect effect, such a bias may be non-trivial, obsfuscating its effect. An isotropic optimiser would remove these biases.

In the following equations, parameters at time-step t are indicated by θ, a per-coordinate gradient at time-step by g_t, learning rate by η — largely consistent with Zhang et al. [35] notation when comparing the algorithms.

Adagrad optimiser:

$$\begin{aligned} \theta_{t+1} &= \theta_t - \frac{\eta g_t}{\sqrt{s_t+\epsilon}} \\ s_{t+1} &= s_t + g_t^2, \quad s_0 = 0 \end{aligned} \tag{35}$$

AdaDelta optimiser:

$$\begin{aligned} \theta_{t+1} &= \theta_t - g_t' \\ g_t' &= \frac{\sqrt{\Delta x_{t-1}+\epsilon}}{\sqrt{s_t+\epsilon}} g_t \\ \Delta x_t &= \rho \Delta x_{t-1} + (1-\rho) g_t'^2, \quad \Delta x_0 = 0 \\ s_t &= \rho s_{t-1} + (1-\rho)\, g_t^2, \quad s_0 = 0 \end{aligned} \tag{36}$$

ADAM optimiser:

$$\begin{aligned} \theta_{t+1} &= \theta_t - \eta g_t' \\ g_t' &= \frac{\tilde{v}_t}{\sqrt{\tilde{s}_t}+\epsilon} \\ v_t &= \beta_1 v_{t-1} + (1-\beta_1)\, g_t, \quad v_0 = 0 \\ s_t &= \beta_2 s_{t-1} + (1-\beta_2)\, g_t^2, \quad s_0 = 0 \\ \tilde{v}_t &= \frac{v_t}{1-\beta_1^t} \\ \tilde{s}_t &= \frac{s_t}{1-\beta_2^t} \end{aligned} \tag{37}$$

Within *Eqn.* 35, this can be seen through the coordinate-wise accumulated squared-gradient value. Similar is true for *Eqn.* 36 and *Eqn.* 37. Thus a decomposition along the standard basis is used in this accumulation, which results in an anisotropy through its implementation.

However, in Wang and Wiens [36] their optimiser is ADAM-like with rotational equivariance. This AdamSGD algorithm is displayed in *Eqn.* 38. This may be a starting point for a more optimal Isotropic adaptive optimiser.

$$\begin{aligned} \theta_{t+1} &= \theta_t - \eta_t m_t \\ \eta_t &= \eta \sqrt{\frac{1-\beta_2^t}{v_t / \dim \theta}} \\ v_t &= \beta_2 v_{t-1} + (1-\beta_2)\, \|g_t\|_2^2, \quad v_0 = 0 \\ m_t &= \beta_1 m_{t-1} + (1-\beta_1)\, g_t, \quad m_0 = 0 \end{aligned} \tag{38}$$

Alternativly, a return to an inverse-Hessian approximating quasi-Newton algorithms such as BFGS [37, 38, 39, 40] or L-BFGS [41], may be another approach. Gradient clipping is another operation requiring consideration, since anisotropy could produce a representational bias. Both directions are being activly researched by the author.

C.4 Operations:

It is possible to also define several new operations within the Isotropic framework, these include min-like and max-like functions and a new multiplication.

The standard minimum function and maximum function are displayed in *Eqns.* 39 and 40 respectivly, for a function $\mathbf{f} : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}^N$.

$$\mathbf{f}\left(\vec{x}, \vec{n}\right) = \min\left(\vec{x}, \vec{n}\right) = \sum_{i=1}^{N} \min\left(\vec{x} \cdot \hat{e}_i, \vec{n} \cdot \hat{e}_i\right) \hat{e}_i \tag{39}$$

$$\mathbf{f}\left(\vec{x}, \vec{n}\right) = \max\left(\vec{x}, \vec{n}\right) = \sum_{i=1}^{N} \max\left(\vec{x} \cdot \hat{e}_i, \vec{n} \cdot \hat{e}_i\right) \hat{e}_i \tag{40}$$

These functions are applied elementwise to components of $\vec{x}$ and $\vec{n}$ as indicated by the sum over standard basis directions, $\hat{e}_i$. This basis dependency is an example of an inductive bias which affects representations. These functions have two arguments, which makes the initial isotropy equivariance relation inapplicable. One way to generalise such a equivariance is by applying the group action to both arguments in a modifed equivariance relation: $\mathbf{R}\mathbf{f}\left(\vec{x}, \vec{n}\right) = \mathbf{f}\left(\mathbf{R}\vec{x}, \mathbf{R}\vec{n}\right)$, for all $\mathbf{R} \in \mathrm{SO}\left(n\right)$. This form may need revision, as it has not yet been connected to the architectural appraoch to symmetries, discussed in *Apps.* A and F. Accordingly, one can define *Eqns.* 41 and 42 as functional forms which follow this relation. In the case where $\vec{n} \cdot \vec{x} = 0$, the operation should be defined as the identity map.

$$\mathbf{f}\left(\vec{x}, \vec{n}\right) = \min\left(\mathrm{sign}\left(\vec{x} \cdot \vec{n}\right), \left|\frac{\vec{n} \cdot \vec{n}}{\vec{x} \cdot \vec{n}}\right|\right) \mathrm{sign}\left(\vec{x} \cdot \vec{n}\right) \vec{x} \tag{41}$$

$$\mathbf{f}\left(\vec{x}, \vec{n}\right) = \max\left(\mathrm{sign}\left(\vec{x} \cdot \vec{n}\right), \left|\frac{\vec{n} \cdot \vec{n}}{\vec{x} \cdot \vec{n}}\right|\right) \mathrm{sign}\left(\vec{x} \cdot \vec{n}\right) \vec{x} \tag{42}$$

These operations are a suggestion for such an isotropic function, but may not be optimal. They clip vectors which across a hyperplane boundary defined by the choice of $\vec{n} \in \mathbb{R}^N$, the hyperplane equation is: $\vec{x} \cdot \vec{n} = \|\vec{n}\|_2^2$. The minimum function preserves the coordinates of all samples on the origin side of the hyperplane, and projects all other coordinates onto a hyperplane. This projection is carefully chosen such that the projected point continues to be on a line passing through the origin and to the original coordinates, whilst intersecting the plane. As a consequence, neural refraction does not occur on the plane boundary for origin intersecting lines. The maximum function computes the opposing case, where points are kept constant on the far side of the hyperplane which does not include the origin. In this case, points within the other sector are projected onto the hyperplane in similar manner.

Similarly, Hadamard mutliplication, also termed elementwise multiplication, is inherently basis-dependent. This can be seen through its basis dependence, through $\hat{e}_i$, in *Eqn.* 43. This is used in settings such as the LSTM blocking-gates [42], dropout [3], alongside many more.

$$\mathbf{f}\left(\vec{x}, \vec{n}\right) = \vec{x} \otimes \vec{n} = \sum_{i=1}^{N} \left(\vec{x} \cdot \hat{e}_i\right) \left(\vec{n} \cdot \hat{e}_i\right) \hat{e}_i \tag{43}$$

One interpretation of *Eqn.* 43, is that it scales each component of $\vec{x}$ by each component of $\vec{n}$ such that each axis in the standard basis is rescaled. This is equivilant to premultiplying $\vec{x}$ with a diagonal matrix $\mathbf{W} = \mathrm{diag}\left(\vec{n}\right) \in \mathbb{R}^{N \times N}$, where diag decomposes $\vec{n}$ along the standard basis and puts these components along the diagonal of a square matrix. This axis-wise rescaling is anisotropic.

To reproduce a similar behaviour, one could use Isotropic multiplication shown in *Eqn.* 44. This rescales the component of $\vec{x}$ which lies in the $\vec{n}$ direction by an amount determined by $\|\vec{n}\|$.

$$\mathbf{f}\left(\vec{x}, \vec{n}\right) = \vec{x} + \left(\left(\|\vec{n}\| - 1\right) \vec{x} \cdot \hat{n}\right) \hat{n} \tag{44}$$

These are just examples and may not be a simple drop-in replacement for existing operations, due to the differences in the way the operations act. Particularly, initial anisotropic operations rescale in multiple axes/hyperplanes at once, whereas the isotropic operations only act in single directions. This may make them suboptimal as blocking-gates, such as in LSTMS, since they can only collapse the representations in one direction at a time: $\mathbf{f}:\mathbb{R}^N\times\mathbb{R}^N\to\mathbb{R}^{N-1}\hookrightarrow\mathbb{R}^N$. These are single direction gates which may be limited in how they reshape the representations and have greater computational cost; therefore, further development is certainly needed. In general, these specific implementations may each require a custom isotropic operation which fully replicates their desired effect in an isotropic and basis-independent manner.

Finally, there may be some instances where representations are required to be clipped into a hypercube-shape. This is *not* an isotropic operation, nor basis independent; however, this formulation does reduce neural refraction, so is included. Using *Eqn.* 45 results in substantial neural refraction along the boundary of the hypercube: lines passing through the origin are projected across the boundary, substantially changing direction.

$$\mathbf{f}\left(\vec{x}\right)=\max\left(\min\left(\vec{x},1\right),-1\right)=\sum_{i=1}^{N}\max\left(\min\left(\vec{x}\cdot\hat{e}_i,1\right),-1\right)\hat{e}_i \tag{45}$$

Using *Eqn.* 46 achieves the same result without neural refraction at the boundary. Any coordinate which is outside the boundary is projected back onto the boundary, along a line between the origin and the original coordinate. As a result, direction is preserved. An affine transform can recentre this box at arbitrary coordinates and reshape the box due to the linear transform.

$$\mathbf{f}\left(\vec{x}\right)=\min\left(\frac{1}{\max_i\left(\vec{x}\cdot\hat{e}_i\right)},1\right)\vec{x} \tag{46}$$

D Stochastic Isotropy — Producing Immediate Anisotropic Analogues

One method to approximate isotropy with current functions is to stochastically choose a basis on which the anisotropic function operates. This enables anisotropic functions to be used in an isotropic network, without inducing a representational alignment to an arbitrary basis.

For example, current (anisotropic-)dropout by Srivastava et al. [3], would appear to privilege the basis anti-aligned with the standard basis, to maximally preserve information when a direction of the standard basis is collapsed. So it is expected to incur an arbitrary basis dependence on the representations. However, anisotropic dropout can be applied to a stochastically chosen basis. This randomness is hypothesised to prevent a representational-anisotropy induced by an arbitrarily chosen basis.

This can be achieved by producing a basis, uniformly drawn from the layer's special-orthogonal symmetry: $\mathbf{B} \sim \mathrm{SO}\left(n\right)$. There are many such methods to produce such a uniform random matrix, with varying computational costs, such as via exponentiation of Lie generator scaled by an appropriately drawn random variable, the Gram-Schmidt procedure [43], and many others [44, 45]. The existing matrix-multiplication procedure is computationally cumbersome so that simpler formulations may be desirable. The following proposed forms are only a starting point for converting anisotropic functions directly to stochastically-isotropic forms. In practice, isotropic functions should be constructed from the ground up rather than merely analogous functions converted from existing anisotropic ones.

For the example of standard dropout, shown in *Eqn.* 47, it can be made stochastically-isotropic by including the basis-transform shown in *Eqn.* 48. Where $\vec{x}$ is the activation vector, with normalisation factor S_a and S_{si}, dropout-mask $M_i = \vec{M} \cdot \hat{e}_i$ and standard-basis vectors $\hat{e}_i$.

$$\vec{x}' = S_a \sum_{i=1}^{N} M_i \left(\vec{x} \cdot \hat{e}_i\right) \hat{e}_i \tag{47}$$

$$\vec{x}' = S_{si} \sum_{i=1}^{N} \left(\mathbf{B}\vec{M} \cdot \hat{e}_i\right) \left(\vec{x} \cdot \hat{e}_i\right) \hat{e}_i \tag{48}$$

Similar formulations, such as RotationOut by Hu and Poczos [46], showed generally improved performance when the basis is effectively stochastically rotated. However, this method remains stochastically anisotropic as the rotations are generated through Given's rotations [47], which is not uniform over the space of special-orthogonal matrices — a necessity for full stochastic-isotropy. Nevertheless, the implementation by Hu and Poczos [46] is somewhat encouraging.

This procedure can be generalised and applied to any existing anisotropic function. Yet, it is generally preferable to construct an isotropic function from first principles rather than relying on stochastic isotropy, which may be computationally costly.

D.1 Considering Correlating the Stochastic-Isotropy

A curious extension, particularly to stochastically isotropic dropout, would be to correlate the random bases in time. This may produce a time-like structure in a network's embedded activation distribution.

If one imagines a random walk of the rotation matrices: $\mathrm{SO}\left(n\right) \ni \mathbf{R^{(t+dt)}} = \mathbf{R^{(t)}}\delta\mathbf{R}$, with $\delta\mathbf{R} = e^{\mathfrak{r}\cdot\vec{n}}$, with $\mathfrak{r}$ being the corresponding (normalised) anti-symmetric generators for rotations and $\vec{n} \sim \mathcal{N}(\vec{0}, \sigma \mathrm{I}_n)$ with $0 < \sigma \ll 1$. This procedure results in a random walk of the rotation matrix at each time step.

Following this, a time-correlated Bernoulli distribution can be defined. Beginning with $\vec{D}^{(0)}$, divide up the layer of neurons into two sets: inactive $I_n = \left\{i | \vec{D}_i^{(n-1)} = 0\right\}$ and active $A_n = \left\{i | \vec{D}_i^{(n-1)} = 1\right\}$. Then we have two hyper-parameters: the standard dropout probability λ and an overlap probability Γ, such that $\left|A_n\right| q + \left|I_n\right| \Gamma = \left(\left|A_n\right| + \left|I_n\right|\right) \lambda$ - where q is not a free parameter. If $\left|A_n\right| = 0$ or $\left|I_n\right| = 0$, then temporarily define $q = \Gamma = \lambda$. If not, one must prevent unnormalised probabilities as shown in *Eqn.* 49.

$$\Gamma = \max\left(0, \max\left(\lambda + \frac{\left|A_n\right|}{\left|I_n\right|}\left(\lambda - 1\right), \min\left(1, \min\left(\lambda + \frac{\left|A_n\right|}{\left|I_n\right|}\lambda, \Gamma\right)\right)\right)\right) \tag{49}$$

Leading to $q = \lambda + \frac{|I_n|}{|A_n|}(\lambda - \Gamma)$. Then use one Bernoulli function across all active neurons using $\mathbb{R}^{|A_n|} \ni \vec{D}^{(n)}_{(A)} \sim \mathrm{BernoulliDist.}^{|A_n|}(q)$ likewise for inactive neurons $\mathbb{R}^{|I_n|} \ni \vec{D}^{(n)}_{(I)} \sim \mathrm{BernoulliDist.}^{|I_n|}(\Gamma)$. Therefore, correlating the inactive neurons across the time steps, whilst still introducing a degree of random dropout. Thus, the 'basis of dropout' undergoes a random walk at every time step, and neurons are randomly chosen to be dropped from the network, with a differing likelihood if they were just previously dropped. The coherence time can be adjusted through Γ, for the specific time-dependent task needed.

This creates a link between the stimulus's presentation time to the network and the neurons it alters, such that stimuli presented in a smaller time window perturb a similar subset of the network's neurons. This may produce an encoding similar to that found in human cognition, where neurons are thought to go through excitability cycles of slightly differing frequencies and phases. When the excitability is higher, information (engrams) preferentially encodes upon those neurons [48, 49]. As groups of neurons begin to decohere, some overlap remains, such that memories are interlaced if they occur within a temporal window of coherence. *This potentially gives neural networks using isotropic dropout an advantage in time-series data.*

E Potential Applications

Besides the proposed general applicability of the isotropic modifications, below are some places where they may yield significant benefit in performance or enable desirable behaviours in networks.

E.1 Isotropy In Transformers

It is argued that isotropic deep learning may be a more appropriate inductive bias for deep learning. However, there may also be some architectures which especially benefit from its inclusion. One of these is the self-attention step of transformers [11], where isotropic-$\tanh$ may be of particular benefit, in replacing the softmax operation [50].

Softmax is defined through elements being bounded between zero and one, $\mathbf{f}\left(\vec{x}\right) \cdot \hat{e}_i \in [0, 1]$ and summing to one. As a consequence, it is non-negative, and there are regimes where this may be limiting.

It has been shown that representations can exist in an antipodal superposition [16], particularly when stimuli do not tend to coexist in samples, thus antipodal arrangements can exist with minimal interference. Such a stimuli may be a continuous quantity, but its two extremes are mutually exclusive. Many of these semantics exist in the real world: daytime-to-nighttime. These could be represented through a zero-to-one scale; however, perhaps a $[-1, 1]$ scale may be a better representations, with zero better assigned as a neutral middle point. This is because in the linear features hypothesis, often magnitude is indicative of the strength of presence of a stimuli. In this case, the negative of a semantic direction may be equally meaningful and present in varying amounts. It may be expected that enabling this behaviour within the self-attention step is favourable.

Moreover, the sum-to-one case may not always be desirable: it always encourages a change to the semantics when considering the residual-step-modification. **This may force a semantic correction to an activation in transformers even when it is inappropriate or force the existance of a near-zero value vector.**

The self-attention step compares the pairwise similarities between several vectors grouped into the so-called ‘keys’ and ‘queries’. The degree of similarity then affects how much of another semantic is expressed: the ‘values’. However, the softmax layer is basis-dependent and prevents a negative expression of these value semantics.

A more suitable choice may be isotropic-$\tanh$. In analogy of its sum-to-one constraint, its vector-magnitude is at maximum one, $0 \leq \|\mathbf{f}\left(\vec{x}\right)\| \leq 1$, whilst elementwise its values are $-1 \leq \mathbf{f}\left(\vec{x}\right) \cdot \hat{e}_i \leq 1$. Hence, it can express a negative of the value semantic, or any scaling of it between -1 and 1. This suggests that isotropic-$\tanh$ may be quite an appealing drop-in replacement for softmax in the attention step, at least conceptually. Its continuous rotational symmetry may also offer an advantage, since the underlying pariwise similarity of self-attention $QK^T = \vec{x}^T W_Q^T W_K \vec{x} \hat{=} \vec{x}^T W'_{kq} \vec{x}$, is also basis-independent in $\vec{x}$, which aligns with the principles of Isotropic deep learning. Hence, removing further bases may enable a more even interpolation between, and perturbation to, the value vectors. Hence, an isotropic adaptation to a self-attention may appear as shown in *Eqn.* 50, which will be explored in future work.

$$\text{Attention}\left(Q, K, V\right) = \text{Isotropic-Tanh}\left(\frac{QK^T}{\sqrt{d_k}}\right) V \tag{50}$$

However, this does not make transformers ‘isotropic’ as a whole, since there are many further anisotropic steps present. Nevertheless, single-layer isotropic adaptations remain compatible with a larger anisotropic network or radial-basis network. So, one may hybridise these approaches if the task necessitates.

E.2 Real-Time Dynamical Network Topology

An appealing feature of isotropic deep learning is the relation displayed in *Eqn.* 15, showing that due to rotational equivariance, a rotation to one weight matrix can be counteracted with the inverse-rotation of another, preserving the network’s function. Consequently, a particular gauge can be chosen that expresses the weights in a beneficial basis.

One such basis may be a magnitude ordering of the singular values for the matrices. One could then set a threshold for the singular value to determine if each corresponding direction in such a matrix has a meaningful contribution to the overall functionality. It can be pruned with little adverse effect on the network if it is deemed to have negligible value.

Moreover, ζ latent neurons can be included, with zero-initialised singular values fully connected to existing neurons. Since the Jacobians of the isotropic activation functions are not strictly diagonal, these latent neurons may be rapidly trained if required. Therefore, the otherwise static fully-connected network is now dynamic, growing and shrinking with task-necessitated demand, with minimal impact to performance with these actions. Due to the continuous rotational symmetry available, this is enabled through an isotropic functional form. It poses an interesting research direction, where transfer learning and task-swapping may become more straightforward. Output and input neurons could also be appended and removed in such a way, allowing for real-time changes to a dataset, or even training on multiple datasets. Such a procedure could be trivially extended to convolutional networks, allowing a dynamic number of kernels.

This could offer substantial insight into how parameters may be shared between tasks in real-time. For example, we may postulate that if a new dataset is introduced partway through training on a different dataset, there might be a short-term parameter increase until the network parameter-sharing begins, followed by a phase of pruning until a more compact architecture is reached. These network dynamics may be incredibly insightful.

It appears it may side-step the lottery ticket hypothesis in choosing optimal network size before training. Due to the computational cost, this does not need to be computed at every step, only periodically, and can be performed layerwise.

E.3 Multi-Headed Layers

Although not limited to isotropic deep learning, producing ‘multi-head’ feed-forward layers may be desirable enabling purturbative-like corrections to activations at each layer. This could be achieved, by summing over a series of activation functions each layer. An example of this is shown in *Eqn.* 51 or a more general construction in *Eqn.* 52, for weight matrices $\mathbf{W}^j$ and $\mathbf{W}'^j$, biases $\vec{b}^j$ and $\vec{b}'^j$ and an activation function $\mathbf{f}:\mathbb{R}^n\rightarrow\mathbb{R}^n$.

$$\vec{x}^{l+1}=\sum_j \mathbf{f}\left(\mathbf{W}^j\vec{x}^l+\vec{b}^j\right) \tag{51}$$

$$\vec{x}^{l+1}=\sum_j \mathbf{W}'^j\mathbf{f}\left(\mathbf{W}^j\vec{x}^l+\vec{b}^j\right)+\vec{b}'^j \tag{52}$$

This is effectively a sum over several a feed-forward layer, which could increase the expressibility of a layer. It may be especially beneficial for isotropic functions, such as isotropic-$\tanh$, where each sum produces a further purturbative ‘correction’ to an output vector.

E.4 Isotropic Representations May Aid Semantic Alignment

An emerging interdisciplinary field of semantic alignment [51] are trying to produce comparable representations between deep learning and the brain. The author feels it is worth investigating how the representations produced by the Isotropic Deep Learning approach may aid this. This is because anisotropic functional forms have been shown to create representational structure due to functional forms, as opposed to being naturally emerging from the data. Such artificial structure may be detrimental to representational alignment tasks. However, this connection is largely speculative, but is included as a point of discussion and potential research avenue for isotropic deep learning. For isotropy, this may provide a testable route for the hypothesis of isotropic deep learning forming more ‘natural’ semantic structure in representations.

Current anisotropic networks are imparting a significant effect on the representations [1], producing activation alignment with the discrete distinguished directions produced by the anisotropic functional forms. This is producing an artificial, and possibly unnatural, anisotropic distortion onto the distribution. This may representation basis-dependence may jeopardise efforts to produce representational alignment. Particularly removing anisotropy may help with alignment methods which use continuous rotation like in Williams et al. [52]'s work, since the inductive bias of isotropy is equivariance to continuous rotation.

Despite this, the brain is unlikely to be isotropic due to the approach's delocalised functional forms. In isotropic networks, neurons instead act as a collective and are arbitrarily decomposable into any set of individual neurons, due to the gauge invariance. Consequently, there is likely to be no single identifable and agreed upon definition of a neuron in an isotropic network. This is fundamentally incompatible with the structures seen in the brain. However, despite the incompatibility of functional forms with biological neuron behaviour, representations may differ. Representational distortion in deep learning does not imply that the brain also produces anisotropic and discrete features when time-averaging the firings of neurons.

A similar approach may be extendable to inferring meaning from undeciphered languages. If isotropy produces a continuous representation, free from basis-distortions, then one may expect a more structured and interpolatable semantic structure. One may speculate whether an approximatly language-agnostic structure may develop in similar analogy to representational alignment, which is either assuming or encouraging model-agnostic representation structure. Hence, there may be a chance that a representation free from artificial anisotropies may aid in deducing a representational alignment between known and unknown vocabulary for decipherment. It is speculated that isotropic networks may be especially beneficial for this due to their removal of artificial geometric inductive biases on the representations.

Despite this, the success of ensemble models may limit this alignment. Ensemble models use a diverse set of individual models to collectively approximate a task solution. The model diversity would suggest that there are more minima than those connected through a permutation or continuous symmetry, which would not yield diverse individaul solutions as are functionally identical. Therefore, this may challenge any assumption of a universal and comparable representational structure for semantics. Despite this, they may produce different approximations to a universal representation. Substantial testing will illucidate this, and the basis-undistorted representations produced by isotropic networks may be particularly beneficial in this research.

F Distinction from Equivariant Networks of Geometric Deep Learning

Both equivariant networks and the proposed isotropic deep learning use an equivariant relation in their definition, which may make them appear similar due to a shared symmetry formalism. However, they differ substantially in how this relation is implemented and motivated, as well as in its consequences and the origin of the symmetry. This section will review those differences. Starting with a discussion of the closest key-concepts within geometric deep learning, namely Equivariant Group-Convolutions [53] and discussion of Steerable-CNNs [54], Harmonic networks [55] and Spherical-CNNs [56]. Following this is a summary of similarities to isotropic deep learning between these methods. Finally, crucial differences will be made clear to show isotropic deep learning's distinction from these methods and the geometric deep learning paradigm as a whole.

The approach of Cohen and Welling [53]'s Group Equivariant Networks is to utilise a specific symmetry, particularly the one expressed in the underlying task's data-structure domain, and ensure the network as a whole respects the task-relevant symmetry through use of a modified convolution operation: Group Convolutional Neural Networks (G-CNNs). This is generalising the traditional translation equivariance of the convolution operation (ignoring edge effects) to instead be equivariant to a discrete group $\mathcal{G}$. This symmetry group is chosen a priori, *defined by the known symmetries of the task as an inductive bias* and the symmetry-respecting constraint is applied to the *whole* network architecture. This can have a wealth of benefits: increased weight-sharing efficiency, physically accurate modelling and a resultant increased expressive capacity.

If the task initially has its data distributed over a particular linear base space and obeys a symmetry of that space, then the data can be 'lifted' to a symmetry group acting on that space in the equivariant networks framework. This symmetry is displayed by data and feature maps being modelled as a map: $f : \mathcal{G} \to \mathbb{R}^n$. Every element within the group is assigned an n-dimensional vector by f such that the vectors are interrelated through group action. This may be intuited as an enlarged representation space for the activations due to these interrelated copies. The convolution operation then preserves this discrete symmetry in its transform to produce new features for the group. These activation spaces remain enlarged to accommodate the various poses due to the symmetry. So the geometric structure is increased, and possibly semantic information is enriched by better capturing the geometrical group structure.

The group-convolution is then implemented as a modification of the classic discrete convolution operation: by applying the filter over the group, as shown in *Eqn.* 53 — for more details of precise implementation see Cohen and Welling [53]. Where k indexes the filter ψ. Note that the sum is over the base space $\mathcal{X}$ in the first layer: $h \in \mathcal{X}$, not $h \in \mathcal{G}$. Then the resultant equivariant group-convolution respects the symmetries of the task at every layer, given by the aforementioned data structure $f : \mathcal{G} \to \mathbb{R}^n$. This can also be viewed as augmenting the number of filters within the convolution with each action. Crucially, this is achieved more efficiently in practice through indexing, exploiting the group's structure. Overall, this adapts the convolutional operation and padding to respect the symmetries in the underlying data structure, but preserves the functional form of surrounding operations like the activation functions, which is demonstrated to be unaffected by such symmetry constraints. Particularly, the non-linearities must remain elementwise in functional form to preserve this equivariance of the convolution in G-CNNs. This is because the elementwise nature commutes with the group action, so preserving equivariance.

$$[f \star \psi](g) = \sum_{h \in \mathcal{G}} \sum_{k} f_k(h)\, \psi_k\left(g^{-1}h\right) \tag{53}$$

In the subsequent works of [54], [55] and [56], they make considerable progress in developing this paradigm by extending the architectures and tools. In [54], the authors build upon earlier work by creating steerable capsules, in which vectors transform under irreducible representations of the discrete group. These steerable filters are constructed as linear combinations of base filters, producing a more parameter-efficient construction. In [55], the authors then use the steerable filters to construct equivariance to continuous patch rotation using finite filters and spherical harmonic functions exhibiting the desirable rotational equivariance. With [56], they generalise these concepts for images over a spherical shell, $\mathcal{S}^2$, and lift it to an $\mathrm{SO}(3)$ continuous symmetry equivariance, using a fourier transform-like method. Through these and others, networks are made equivariant to discrete group transforms and extended up to specific continuous group transforms. This subfield is rich with many other discoveries along the same vein; however, other further examples shall not be discussed since they diverge from a similar seeming construction of isotropic networks, and the key differences can be addressed with these existing examples.

Overall, the similarities between approaches are a tangential use of an equivariance relation as a core principle and a group-theoretic framework, particularly at its most similar, a continuous rotational equivariance of an aspect of deep learning. Similarly related are other geometrical concepts such as vector spaces, Lie groups and gauges. They may also both improve representations for physics-related tasks, where vector space construction can be crucial. Although Isotropic deep learning is of geometrical and deep learning construction, it does not sit cleanly into the current field of geometrical deep learning [57]. Instead, it is constructed around the geometry of embedded representations, as internal symmetries rather than a network-wide externally applied symmetry instilled by a predominantly task-dependent inductive bias.

The core of the differences between Equivariant networks and Isotropic deep learning is the wholly different contexts in which the symmetry arises and its consequences for the network.

For equivariant networks, the symmetry originates from the task itself: the data is lifted from a base space onto a symmetry group, which the network is then designed around to explicitly enforce the external symmetry of the data domain into its solutions. Meanwhile, for isotropy, symmetry is applied at the level of activation vector spaces through symmetry in the class of network functions. The latter originates from an argument of representational geometry not being arbitrarily deformed due to distinguished directions, typically incidentally, imposed by humans. It is an equivariance constraint on the network's functional forms rather than an equivariance of the network as a whole: a more local vs a global approach. A symmetry of data, which is task-necessitated, versus a general symmetry of internal representational geometry.

This highlights the differing motivations: the injection of highly specific task-aligned inductive bias, informed by the task, and hard-coded into the architecture for equivariant networks to increase efficiency, leverage symmetry structure for generalisation, and constrain the solution to a known hypothesis space. Whilst isotropy is the removal of a usually unintended inductive bias, the artificial basis-dependent anisotropies. Thus, isotropy is motivated as a minimal inductive bias as a new default for broader applicability, as a less arbitrary and arguably more natural geometry for representations by removing basis-dependence from functional forms - creating a task-agnostic inductive bias *unless* a priori task-specific knowledge is known for a problem where a strong constraint should be added.

A further difference is that the Isotropic symmetry is *only* enforced in the internal representation spaces, not even necessarily preserved through the transformations between the chain of vector spaces within a network. As a consequence, the symmetries themselves and the effect on activations also differ substantially. For isotropic networks, the equivariance is constructed from a family of symmetry classes, e.g. special orthogonal, where particular layers then acquire a specific instance of this family to be equivariant to. So a general principle of SO family symmetry enforcement on representations, which then functional forms have an equivariance to a specific symmetry from this family, for example: A layer's activations form an $\mathbb{R}^l$ linear vector space, which is transformed through a function $\mathbf{f}:\mathbb{R}^l\rightarrow\mathbb{R}^l$. As a consequence, the $\mathrm{SO}\left(l\right)$ is used in its equivariance to define $\mathbf{f}$'s functional form. Therefore, per layer, a specific symmetry from a symmetry family is used, whose functional forms are equivariant. An equivariant network is made equivariant to a specific instance of a symmetry class, which is task-necessitated. Hence, generally, isotropy is concerned with a symmetry family rather than a particular instance of a symmetry group. In isotropic deep learning, the network as a whole does not need to respect a specific symmetry[4].

Moreover, this has very different consequences for the vector spaces themselves. The equivariant network modifies the dimensionality, or construction of vector spaces, to enforce a global symmetry; whereas, isotropy leaves the vector space construction unchanged, affecting only certain transforms between them. The symmetry constraint in Equivariant networks produces an enlarged dimensionality of the activation spaces to accommodate the group structure in [53]. However, this differs for later irreducible representations like those in [54]. Both leave a stark difference in the structure of the activation space. Isotropy does not affect construction since a specific symmetry is not enforced globally, but a family of symmetries is applied only locally. Principally, isotropy is just elevating the existing discrete inductive bias in functional forms to a continuous one. It leaves the architecture topology and, consequently, vector space construction unchanged, giving it broad applicability.

[4]but may only if desirable

This is a consequence of isotropy's foundational derivation, which will only be briefly outlined as it is being substantially developed for future publications. Fundamentally, the isotropic symmetry can be argued to emerge from the architecture itself. In this work, isotropic deep learning is predominantly discussed in terms of fully connected feed-forward architectures, of an arbitrary number of hidden layers and an arbitrary number of neurons per hidden layer. However, in future work, it will be explained that these symmetries arise at the level of arbitrary graph structures. When one examines an arbitrary directed graph and endows its nodes with continuous activations that can be grouped and systematically divided up vector spaces, chained through continuous maps, one can apply group actions to this continuous-valued node topology, leaving the endowed directed graph unchanged. Here, the continuous isotropic symmetry arises and can be broken into discrete permutation symmetries characterising modern deep learning. Consequently, one can show that the isotropic symmetry originates from the arbitrary network architecture itself, rather than a task-dependent symmetry informing the development of a specific architecture, as in equivariant networks. From this, isotropy is applied to functional forms across the board: activation functions, initialisations, normalisations, regularisers, optimisers, etc., but not necessarily affecting architectures. This sets the approaches within two very different directions, regarding the origin of its symmetry and its relation to architecture. In this regard, isotropy can be viewed as a more fundamental and natural behaviour of the architectures than its broken symmetry counterpart of current anisotropic deep learning. Some overlap may occur, but isotropy is predominantly a symmetry from an architecture influencing functional forms. In contrast, equivariant networks are a symmetry of the underlying data structure that influences the architecture.

If one does alter the Isotropic deep learning architecture such that the family of isotropic symmetries is reduced to a specific instance of a symmetry, e.g. $\mathrm{SO}(3)$, then one could argue that the network is also globally equivariant, so a type of equivariant network. This establishes a tentative bridge between the two approaches. However, this is a very different symmetry to the $\mathrm{SO}(3)$ in [56]'s work. This is an $\mathrm{SO}(3)$ in the activation space vectors, not a symmetry in the coordinates of the base space, like [56] achieved, and must not be conflated. Therefore, since it remains a symmetry of representations, not data structure, one must broaden the definition of equivariant networks to establish any overlap with isotropic deep learning.

This is not to say one is better or more principled, or even parallel techniques; they are constructed for differing purposes. One for respecting a specific symmetry in solutions present in a particular task, such as in many physics-related problems, whilst one removes a basis dependence which may arbitrarily and anisotropically affect the distribution of representations in all problems, motivating it for universal adoption in deep learning. Isotropy is a proposal of basis independence and gauge invariance. These are differing proposals, an external-geometric-symmetric framework and an internal-algebraic-symmetric framework, both using equivariances as a core feature. This shows that isotropy is a framework with broader and more flexible applicability, justifying the assertion that one day it may be a better default inductive bias. Still, it does not displace the case-by-case application of a strong inductive bias in equivariant networks. As it stands, the substantial differences in approach to symmetry make the frameworks incompatible, seen through the restriction of pointwise nonlinearities in equivariant networks. This mutual exclusivity might be bridged within further work, only if a task-dependent inductive bias makes it desirable to do so.

Concluding: Isotropic deep learning and the various generalisations of Equivariant network share a similarity in their fundamental construction from an equivariance relation and philosophical focus on symmetry principles. Equivariant networks are made to enforce an end-to-end symmetry deduced from their data structure, enforced by architecture modifications like group-convolution [53]. This dramatically increases parameter efficiency and ensures physical solutions for specific problems. Isotropic deep learning promotes the existing discrete rotational symmetry to a continuous one in localised functional forms, whilst not necessarily making the network equivariant as a whole. The premise is to unconstrain embedded representations for general problems by primarily removing arbitrary basis dependence in functional forms. This is hypothesised to enable networks to form better-structured latent spaces. Therefore, equivariant networks are instilled with an a priori respect for a task-dependent symmetry, whereas isotropy is being developed as a universal new default for functional forms.

Hence, isotropic deep learning is both a framework of geometry and deep learning, but currently is a significant deviation from the foundational blueprint within Geometric deep learning [57], despite the use of equivariance and symmetries. They have substantial differences in how and which symmetries arise and how they affect the models, how activations are represented in networks and how parameters are constructed. Isotropic Deep Learning redefines the form for parameter initialisation, rather than restructuring parameters, so as not to affect the layer structure, which can remain dense.

Currently, the approaches are essentially mutually exclusive. However, as shown, specific instances of isotropic deep learning can be made to express some characterising properties of an equivariant network, so they retain some very minimal overlap. Isotropic deep learning more suitably falls within the class of geometries of representations, more closely related to the work of Elhage et al. [16], Olah et al. [22], Carter et al. [58]. This interdisciplinary approach, alongside work such as [16], may be more appropriately classified under a name such as ‘*representational geometry in deep learning*’.

G Anisotropy as a Historical Precedent

In this section, the historical precedence for anisotropic functions are explored, alongside how outside influences may have shaped the field in such a way. In particular, a discussion of how such functional form choices became near axiomatic to the definition of deep learning is provided. It requires a look into how coding schemes of neuroscience may have imparted this inductive bias from deep learnings[5] conception as a distinct discipline. However, there have been many influences throughout the course of the field's development and this section is only intended as a retrospective and speculative analysis of how neuroscientific coding schemes may have produced and reinforced the anisotropic deep learning to become the dominant paradigm.

Neural networks, of all forms, represent their information through a neural code which remains under debate [60, 61, 62]. Several codings have been proposed and it appears that initially anisotropic functions may have indirectly arisen from one of these: local coding.

Local coding predates deep learning [63] and was relatively more common around the advent of artificial neural networks, whereas it has been generally superceded by distributed codes, often sparse [20], as a model of biological neuronal activity [64] and to some extent in deep learning [22, 65, 16, 1]. Local coding is a neuropsychological hypothesis that one neuron's activation represents the presence of one real-world stimulus. This is also commonly known as the grandmother neuron interpretation [66, 67], and less so as gnostic neurons and gnostic fields [68]. This hypothesis was developed [63, 69, 68] and primarily debated [70] from the early 1940s through to the early millenium [66, 60, 67, 64], with a growing consensus of sparse coding in the present day. However, it will be argued that this somewhat outdated local coding scheme has left a lasting impact on the field of deep learning.

Crucially, McCulloch and Pitts later co-authored a paper supporting the presence of several differing pattern feature detectors in frog's retinas [71] expanding on Hartline's earlier work of similar findings [72]. Thus, it was shown that retinal ganglions can respond to distinct real-world patterns, so-called 'bug detectors' for the frog. Despite it being a ganglion, this work aligns closely with the description for grandmother neurons. These are the same McCulloch and Pitts who are earlier credited as the inventors of the binary threshold network [73] which laid the foundation for the first perceptron neural network being developed [59, 74]. In the very first sentence of [73] work, it states:

> *Because of the "all-or-none" character of nervous activity, neural events and the relations among them can be treated by means of propositional logic.*
>
> *–A Logical Calculus of the Ideas Immanent in Nervous Activity [73], p. 115.*

This quote could be considered to premise artificial neural networks on a framework of treating neuronal activity, and interacting representations, as a binary logic. This is arguably an earlier instance, of the same vein of work which led to their paper Lettvin et al. [71], indicating a growing interest in local coding at the time of deep learning's first developments. Particularly forming a link between McCulloch and Pitt's deep learning groundwork and their neuroscientific work, alongside computational convenience of the binary approach for early computers. A further indication of a pervasive local coding paradigm in one of McCulloch and Pitts [73]'s concluding statements:

> *[...] pushed to ultimate psychic units or "psychons," for a psychon can be no less than the activity of a single neuron. Since that activity is inherently propositional, all psychic events have an intentional, or "semiotic," character. The "all-or-none" law of these activities, and the conformity of their relations to those of the logic of propositions, insure that the relations of psychons are those of the two-valued logic of propositions. Thus in psychology, introspective, behavioristic or physiological, the fundamental relations are those of two-valued logic.*
>
> *–A Logical Calculus of the Ideas Immanent in Nervous Activity [73], p. 131.*

This "psychon", with semiotic qualities and two-valued logic, is suggestive of a local coding approach. This strongly aligns with the paradigm of binary operations on components of decomposed vectors in modern neural networks, through generalising the two-value logic. McCulloch and Pitts [73] may have normalised the brain-inspired, basis-dependent nature, of computation very early in deep

[5]While the term Deep learning did not have usage in the early stages of the field's development, it is adopted in this overview for clarity and continuity. This extends to early models such as [59].

learnings foundations based upon these neuroscienctific thoughts of the time. This likely influenced Rosenblatt whom later expanded upon McCulloch and Pitts [73] by implementing the Heaviside step function into his perceptron models [59].

These notions, inspired by the neuroscience of the time, appear to have implicitely encouraged the this adoption of the Heaviside step function in early deep learning through the foundational work of Rosenblatt [59]. This approach to activation functions has a continuous lineage to the vast array of elementwise functions utilised today. Connected through a series incremental modifications from discrete heaviside step functions, to adoption of differentiable sigmoid-based functions [75], to non-saturating ReLU [76] through to its variants. Which in-turn may have had some influence on the wider array of functions, like elementwise Dropout [3] or approximating hessians in a basis-dependent manner in adaptive optimisers [33, 34, 8] and many other instances. This is argued to have become more entrenched into the field through time as future developments continued to replicate the elementwise pattern and build off it.

Where this may have diverged with neuroscience's developements, is through a fundamental difference in the field. The brain has been incrementally shaped by natural selection, it is a natural structure, and modern interpretations of any specific coding do not produce a corresponding shift in its structure. Whereas for deep learning, they are artificial networks, shaped by choices. Thus, if coding interpretations have shaped the field, then they may induce any such biases into network characteristics. Local coding may have influenced early functions into their anisotropic form, the discretising effects of which then bias representations to retaining local coding [1]. Observations of this can then be used to implicitly justify the elementwise form. Moreover, the anisotropy may have circularly self-reinforced through performance metrics: anisotropic forms can produce symmetry broken representations [1], which is speculated to have benefit when using further anisotropic functional forms which operate along such redisposed discretised representations.

The effect may be two-fold: when deep learning models are analysed, they frequently display local-coding, implicitly suggesting that the corresponding elementwise functional form is most appropriate and not drawing skepticism about it. Furthermore, the discretised representations may in turn benefit the further continuation of these forms, since they may now outperform alternatives due to the preexisting effect on representations. Thus, the approach is speculated to have become reinforced through feedback loops. This may have also had some influence on the direction of processors developed for such tasks. All of this may contribute to why the choice has become a staple and generally unappreciated as a *choice*.

These decisions then produce a tendency towards local coding [1], summarised in a sentiment expressed by Olah et al. [22].

> Szegedy et al. found that random directions seem just as meaningful as the directions of the basis vectors. More recently Bau, Zhou et al. found the directions of the basis vectors to be interpretable more often than random directions. *Our experience is broadly consistent with both results; we find that random directions often seem interpretable, but at a lower rate than basis directions.*
>
> *–Feature Visualization, Distill blog publication. [22]*

This summarises the long series of conflicting observations within neural networks, some suggesting monosemanticity [77], some distributed [78] and some a mixture [79]. It is not to suggested that deep learning does not produce distributed representations, which are often observed throughout its development [59, 16], but its functional forms have an inductive bias towards a basis-aligned local coding. Hence, this wide variety of results can be reframed as a strong bias by functional forms, but partially overridden by more complex arrangements under strong task-dependent requirements. This frequently observed break from local coding may also indicate that alternative representation distributions are indeed optimal, but limited in observations due to these functional form biases.

It is felt by the authors, that this bias has been unintentionally but systematically obsfuscated through many years of mathematical notation suppressing explicit basis-dependence, leading to sentiments such as the following:

> *This internal structure has appeared in situations where the networks are not constrained to decompose problems in any interpretable way. The emergence of interpretable structure suggests that deep networks may be learning disentangled representations spontaneously*

–Network Dissection: Quantifying Interpretability of Deep Visual Representations [23], p. 1.

By virtue of the elementwise functional forms, artificial neural networks are loosely constrained to decompose problems in an interpretable way [1]. These inductive biases are injecting structure, despite the impression that no such constraints are applied. It is felt that this may be an explicit reference of a sentiment generally felt as a field, rather than one held solely by Bau et al.

This position paper as a whole is intended to make clear this underappreciated and implicit inductive bias. Hence, to prevent any self-reinforcement from such an implicit bias, one would require Isotropic deep learning to be a one-step overhaul or a very careful analysis of all functional forms, since any hybridised approaches may systematically reintroduce biases in representations and affect results.

Overall, this section has briefly overviewed how anisotropic deep learning can be traced back to a trajectory set in the early developments in the field, shaped by the biological discoveries of their time, and become pervasive through modern developments by research momentum and hypothesised self-reinforcement. The foundations of the field were defined by the choice of elementwise heaviside step function, implemented due to both computational neccessity and likely biological inspiration. Through a continuity of modifications it has emerged as a generalised elementwise form used nearly universally today. The cross-disciplinary approach between neuropsychology and artificial neural network development may have been an initial inspiration for this functional form, but has arguably now diverged due to practices. Yet, this has imparted an implicit inductive bias, entrenched into the mathematical structures of deep learning, which now lacks suitable a priori justification as neuroscience has made it outdated as a universal code. In the following subsection, it is discussed how Isotropic networks may better interact with distributed codes.

G.1 Isotropic Deep Learning for Distributed Codes

An alternative could be imagined. Where biology influences of the time are argued to have led to a preference for elementwise logic and functional forms, particularly through local coding interpretations.

This likely led to a brain-inspired computational model, assigning operations between scalars, as opposed to over a vector space. The latter is not possible within biology, due to constraints of individual neurons with localised responses; however, computation does not have such constraints. However, this trajectory arguably set by McCulloch and Pitts [73], may mean that the activation vectors within neural networks are now treated as just an array of numbers for each individual neuron, as opposed to promoted to a full vector space with direction and magnitude, despite the extensive use of general affine transforms.