METRIC AS TRANSFORM: EXPLORING BEYOND AFFINE TRANSFORM FOR NEURAL NETWORKS

Suman Sapkota *
Independent Researcher

ABSTRACT

Artificial Neural Networks of varying architectures are generally paired with affine transformation at the core. However, we find dot product neurons with global influence less interpretable as compared to local influence of euclidean distance (as used in Radial Basis Function Network). In this work, we explore the generalization of dot product neurons to l^p-norm, metrics, and beyond. We find that metrics as transform performs similarly to affine transform when used in MultiLayer Perceptron or Convolutional Neural Network. Moreover, we explore various properties of Metrics, compare it with Affine, and present multiple examples where metrics seem to provide better interpretability.

1 INTRODUCTION

Artificial Neural Networks (ANN) are used end-to-end and generally as black-box function approximators. This is partly due to the vast number of parameters, the underlying function used, and the high dimension of input and hidden neurons. The backbone of Deep Networks including MLP, CNN Krizhevsky et al. (2012), Transformers Vaswani et al. (2017), and MLP-Mixers Tolstikhin et al. (2021) has been a linear transform of form $\boldsymbol{y} = \boldsymbol{x}\mathbf{W} + \boldsymbol{b}$ (or per neuron: $y_i = \boldsymbol{x}.\boldsymbol{w}_i + b$).

There have been explorations of other operations such as l^1-norm Chen et al. (2020) and l^2-norm Li et al. (2022) for transformations in Neural Networks much guided by the computational efficiency. However, matrix multiplication has won a hardware-lottery Hooker (2021), is highly optimized and other transformations are not explored much. In this work, we explore a form of norms and generalized measure of distance as a neuron, which we call metrics - of the form $y_i = f_{metric}(\boldsymbol{x}, \boldsymbol{w}) + b$. Here, $f_{metrics}$ can be norm function (e.g. l^p), a proper metric function, or any generalization. It is possible to use $f_{metrics}$ for measuring similarity as a negative or inverse distance.

Since metrics have a point of minima the lower contour set of metrics is generally bounded and the minima represents that two points are similar. Norm space can be easily extended to metric space if we take $d(\boldsymbol{x}, \boldsymbol{y}) = f_{norm}(\boldsymbol{x} - \boldsymbol{y})$. We can also generalize the metrics by relaxing different axioms (see Section 12).

In this paper, we explore various properties of metrics such as voronoi partitioning, metrics as transform, their generalization, invertibility and application in low dimensional embeddings.

2 ON ARTIFICIAL NEURONS BASED ON LINEAR AND RADIAL FUNCTION

The dot product is simple to understand but we find the dot product neuron, with non-linear activation is difficult to interpret. It represents a planar neuron rather than a local neuron as shown in Figure(1 *left*). Local neurons are generally found on Radial Basis Function (RBF) Network Broomhead & Lowe (1988) and generally consists of euclidean distance operation, for which the hardware and software are un-optimized, and such transforms are not generally applied in ANNs. RBF was also recognized as interpretable on a blog Olah (2014). This motivates us to explore the area of metrics that give a sense of distance between input and weights (or centers) in neural networks.

Moreover, if we compare the properties of linear neuron and radial neuron, we find strikingly different properties. The activation of linear neuron with relu activation is not bounded, and the maximum value is produced at infinity (∞, or very high values of input). A certain value of activation corresponds to a plane (a line in 2D) of possible inputs, which are again unbounded.

*Code available at: github. *Authors and funding not finalized.*

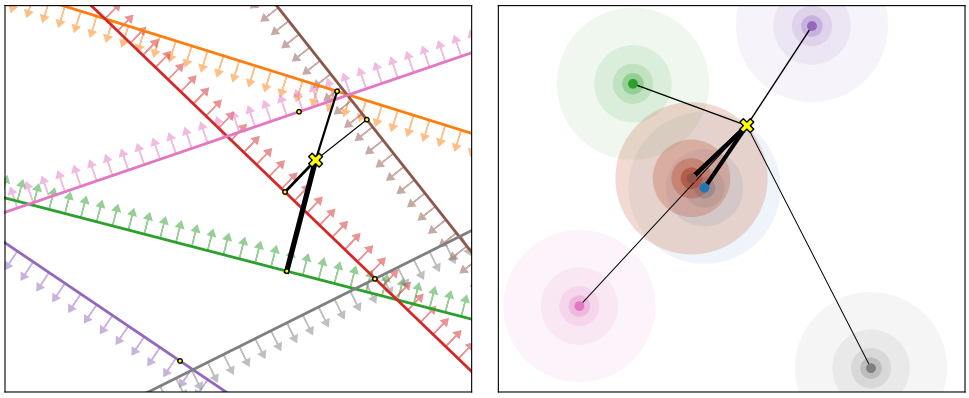

Figure 1: Neuron Interpretation: *(LEFT)* 2D ReLU neuron at $\boldsymbol{w}.\boldsymbol{x} + b \geq 0$, *(RIGHT)* 2D radial neuron at $\exp\left(-\|\boldsymbol{x} - \boldsymbol{w}\|^2\right)$. The lines or circles show region of neuron firing, $(\times)$ showing a data point and thickness of the lines indicating its activation magnitude. Zoom in for details.

Comparitively, the activation of radial neuron with gaussian activation is bounded, the maximum value is produced when center and input are same. A certain value of activation corresponds to a sphere (a circle in 2D) of possible inputs, which are bounded.

We can interpret the activations using similarity metric, which can be maximized by a single point only. We can visualize what a neuron represents by maximizing its activation. Linear neurons fail in that they do not have finite maxima. Previous works have also used neuron maximization to visualize what it represents Olah et al. (2017).

Moreover, we can use any function with finite minima as a generalized measure of distance. Figure 4 shows variation of uniform, convex Amos et al. (2017) and invex Sapkota & Bhattarai (2021) function. We are concerned with functions with a single finite global minima, which can be used as a generalized distance function. The distance from minima can also be interpreted as similarity(e.g. $f(x) = e^{-d^2}$) and produces maximum value (e.g. 1) for a particular point or region. We discuss more on generalized metrics in Section 6,11 and 12.

3 On use of Metrics for Convolution

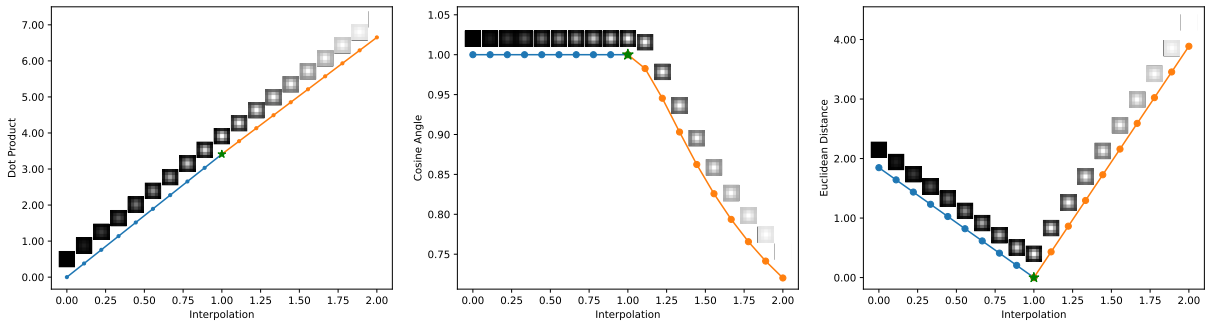

Figure 2: Dot product, angle and distance between patches of image when the values are interpolated between black-gaussian-white. The images just above the scatter-points represents the interpolated image. The star $(\star)$ symbol represents the reference point to which every interpolation is compared. Zoom in for details.

Similar to unstructured input space, convolution operation can be interpreted as a measure of similarity of reference patch (weight) and some input patch. We compare dot product and metrics, namely with l^2-norm and cosine angle on Figure 2. For individual activation, we find l^2 metric more interpretable as it gives minimum value if input and target match, and higher if they are farther away. Here, cosine angle does not differentiate between different intensity of the

image patch. Dot Product gives higher activation for larger intensity of input, more than dot product with reference input itself (i.e. alpha=1).

4 Voronoi Diagram of Transforms

In clustering or classification, the decision boundary of sets and regions of input are assigned to a cluster or a class. Generally, regions are based on distance or linear transforms which produce Voronoi partitioning Voronoi (1908); Fortune (2017). Each set or region is represented by unique neuron of the transform. In Figure 3, we compare Voronoi diagrams of linear transform and distance as transform, show the effect of bias use, and the effect of shift in parameters of the transformation.

Although both linear and distance transforms produce general Voronoi partitioning, centers in distance based transform lie inside their respective sets, which is more interpretable than linear Voronoi sets.

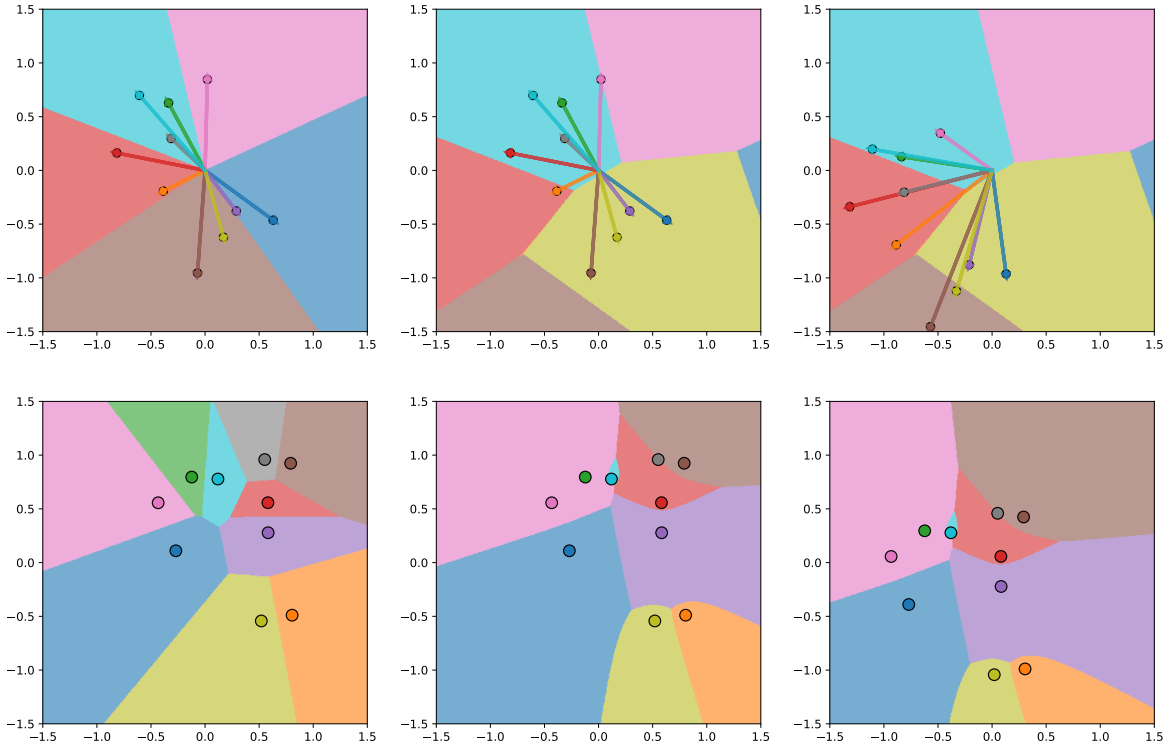

Figure 3: Voronoi Diagram of **(TOP)** Linear and **(BOT)** Distance for *LEFT*: without using bias, *MID*: using bias, and *RIGHT*: using bias and shifting the center/weights by $[-0.5, -0.5]$

5 Application of Metrics as Transform in Deep Neural Network

We try replacing linear layers in MLP with metrics and find that different types of metrics as transform simply work. We identify that normalization like BatchNorm Ioffe & Szegedy (2015) and LayerNorm Ba et al. (2016) or Softmax Bishop & Nasrabadi (2006) helps in optimization, without which we could not efficiently optimize the network. Moreover, normalization of the activation with uniform scale and shift doesn't change the Voronoi set of the neurons.

l^p-norm: We can use l^p-norm Wikipedia (a) with any p value as transform. The general equation of l^p-norm is : $\|\boldsymbol{x}\|_p = (|x_1|^p + |x_2|^p + \cdots + |x_n|^p)^{1/p}$. The norm is convex and valid norm induced metric for $p \geq 1$, however for $p < 1$, it is not convex. Regardless, we can use it as general a measure of distance.

Table 2: Maximum test accuracy among 4 runs with various transforms in ResNet-20.

Transform	Accuracy
l^2	93.06
i-stereo	92.71
linear	92.8

Table 1: Training MLP with different types of transformations at first layer on 2-layered MLP using different number of hidden neurons (H). The given measurements are test accuracy in format *mean*±*std*(max).

Method	$H = 5$	$H = 10$	$H = 20$	$H = 100$	$H = 500$
$l^{0.5}$	72.92±1.07 (74.46)	78.06±0.41 (78.79)	80.13±0.34 (80.84)	83.16±0.05 (83.22)	84.74±0.20 (85.13)
l^1	76.98±0.70 (77.89)	81.71±0.14 (81.92)	82.95±0.27 (83.43)	85.36±0.10 (85.52)	87.02±0.17 (87.35)
l^2	77.84±0.76 (78.92)	82.86±0.18 (83.22)	83.72±0.26 (84.16)	86.16±0.09 (86.31)	87.67±0.08 (87.74)
l^{20}	79.39±0.80 (80.45)	82.90±0.49 (83.62)	83.97±0.26 (84.34)	85.80±0.15 (86.02)	87.63±0.11 (87.79)
i-stereo	80.90±0.60 (81.79)	84.96±0.18 (85.25)	86.32±0.17 (86.59)	88.23±0.17 (88.43)	89.32±0.10 (89.52)
linear	81.17±0.33 (81.70)	84.92±0.21 (85.28)	86.47±0.11 (86.67)	88.42±0.09 (88.55)	89.72±0.07 (89.80)
convex	88.29±0.12 (88.48)	88.96±0.20 (89.30)	88.90±0.09 (88.99)	88.48±0.18 (88.75)	88.49±0.15 (88.67)
invex	88.54±0.26 (88.96)	89.29±0.12 (89.51)	88.18±0.81 (89.42)	88.19±0.26 (88.48)	89.11±0.29 (89.48)
ordinary	84.35±0.49 (84.88)	85.91±0.26 (86.26)	86.14±0.58 (87.24)	87.09±0.37 (87.55)	87.50±0.21 (87.92)

i-stereo: We can use inverse stereographic transform Wikipedia (d) to project N dimensional euclidean space into N+1 dimension Sphere. The the angular measure on the Sphere is a metric and invertible as shown in Section 10.

The experiments on Fashion-MNIST dataset in Table 1 shows that various metrics perform similar to dot product. Moreover, other metrics have not been explored and optimized much. With optimization (like AdderNet Chen et al. (2020)), we believe that the performance can increase. Here, we test on 2 layered MLP (with configuration as: *Layer1*→*BatchNorm*→*LayerNorm*→*ELU activation* Clevert et al. (2015)→*Linear*) with *Layer1* replaced with various metrics.

We were also able to use some of these metrics in CNN (ResNet-20) trained on on CIFAR-10 Krizhevsky et al. (2009) dataset as shown by Table 2. Here as well, we find metric convolution performing similar to the dot product.

6 Generalized notion of distance beyond Metrics

The generalized distances such as learnable convex function Amos et al. (2017), invex function Sapkota & Bhattarai (2021); Nesterov et al. (2022), or any function produce satisfactory results when used as transform in Neural Network. Although these functions are not truly metrics, they serve as generalized notion of distance. We discuss this on Section 12

In table 1, we compare these learnable function as distances in the form: $y = f_{metric}(\boldsymbol{x} - \boldsymbol{w}) + b$, where f_{metric} is itself specific type of neural network (depicted by Figure 4) with similar number of parameters for comparison. We use same settings as Section 5, replacing *Layer1* with generalized distance. The results show that convex and invex functions produce better accuracy than ordinary function. Although the experiment is limited, we can conclude that these notion of learnable distances work. Moreover, we may credit the low accuracy of ordinary neural network to it not being a generalized form of metric ; we need more experiments to rule out other factors like regularization.

7 Overfitting 2 layer Metric based MLP

We can overfit a metric-transform based 2-layer neural network by using M neurons for M data points, similar to k-Nearest Neighbour with $k = 1$. However, this is not feasible for a large dataset. We experiment with center initialization in $l^2 - norm$ based ANN to some random training samples in Table 3. We find that we can gain significant accuracy in MNIST and F-MNIST dataset without even training. We can initialize 2 layered MLP with data and gain some test performance higher than random accuracy.

Moreover, if we want to simplify our network interpretation, we have to normalize the activation to produce maximum value of 1 for neurons similar to input. Method like this has been explored earlier to intrepret MLP layer Sukhbaatar et al. (2015), but without resorting to metrics. In the experiments, we follow normalization method similar to UMAP McInnes et al. (2018). We take first layer to be modified metric to measure the similarity between input and centers or reference data point, i.e.

$$f_{similarity}(\boldsymbol{x}, \mathbf{C}) = \exp\left(\frac{-(\boldsymbol{d} - \min(\boldsymbol{d}))}{\sqrt{\mathrm{Var}(\boldsymbol{d})}}\right); \qquad \boldsymbol{d} = \{l^2(\boldsymbol{x}, \boldsymbol{c}_i)\}$$

for i represents index of all the centers.

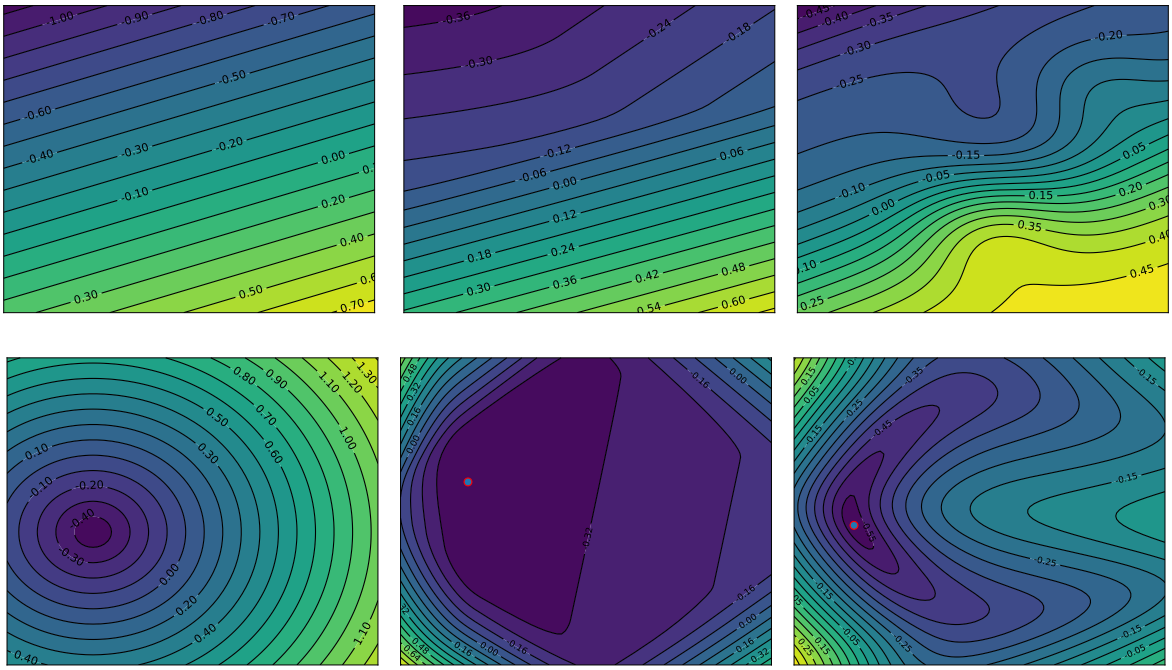

Figure 4: We compare bounded functions with their unbounded counterparts (*LEFT*) Uniform functions (*MID*) convex function and (*RIGHT*) invex function. For measure of distance, we are concerned with bounded functions (*BOT*) with minimum at a point. Here, we can use function bounded contour sets as measure of distance from the minima (x^*). To use the function itself as norm, we need to shift the minima to origin and the output of origin to zero.

Table 3: Initializing centers and labels with given $H = n$ hidden neurons which are initialized to random samples from the dataset. The accuracy is measured on test dataset with 20 different seeds.

Dataset	Acc	$H = 10$	$H = 50$	$H = 200$	$H = 1000$	$H = 5000$	$H = 20000$
MNIST	mean	36.48	60.76	76.40	85.61	88.56	89.43
	std	3.29	3.49	2.21	0.83	0.39	0.20
	max	42.69	67.86	81.0	87.03	89.57	89.79
F-MNIST	mean	37.75	58.08	67.75	72.60	73.96	74.43
	std	6.63	3.47	1.84	1.43	0.45	0.21
	max	50.23	62.10	70.20	75.08	74.74	74.94

The second layer is a linear map, corresponding to the value ($\boldsymbol{v}_i$) when similarity of i^{th} neuron is 1. The overall MLP is of the form: $\boldsymbol{y} = f_{similarity}(\boldsymbol{x}, \mathbf{C}) \cdot \mathbf{V}$. The initialization is done using input target pair, where, matrix $\mathbf{C}$ is initialized with input samples and $\mathbf{V}$ is initialized with the respective target values.

8 Interpretable Local Residual MLP

We interpret the residual network He et al. (2016) of form $\mathbf{y} = \mathbf{x} + \mathbf{S}\ .\ f_{similarity}(\mathbf{x}, \mathbf{C})\ .$ The similarity neuron f_{metric} produces peak activation of $\approx \mathbf{1}$ when the center and data are similar, like previous section. Here, $f_{similarity} = f_{softmax}(f_{metric}(\boldsymbol{x}, \mathbf{C})|t)$ Wikipedia (c) is a scaled and normalized metric. For low temperature (t), the $f_{softmax}$ decision boundary is hard, and Voronoi diagram is produced. The higher temperature (t) of $f_{softmax}$ is useful for training due to its smooth gradients.

The vector $\boldsymbol{s}_i$ for each i^{th} neuron represents the shift of $\mathbf{x}$-space after residual as shown in Figure(5). The shift vector $\boldsymbol{s}_i$ and centers $\boldsymbol{c}_i$ can be initialized with data or trained with backpropagation Rumelhart et al. (1986). In the experiment, we test for 1-block of Residual-MLP to correctly classify 2-spiral dataset, learned with backpropagation.

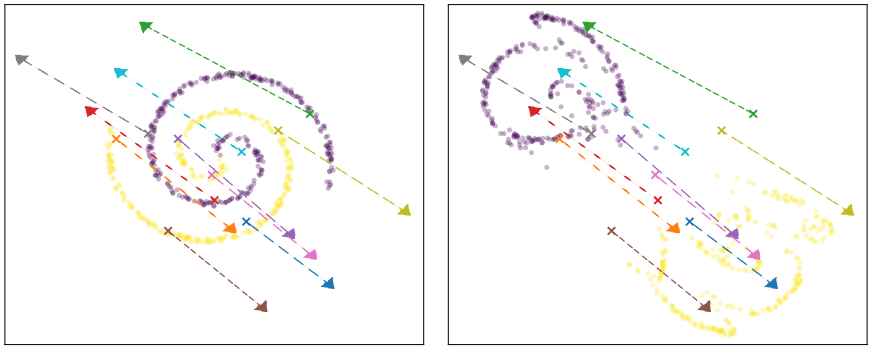

Figure 5: Local Residual Layer interpretation; visualization of the dataset $(\cdot)$ along with centers $(\times)$; Residual MLP moves the centers as shown by the arrow. *(LEFT)* $\mathbf{x}$-space *(RIGHT)* $\mathbf{x} + f_{res}(\mathbf{x}) = \mathbf{y}$ space. Zoom in for details.

9 Activation visualization on Embedding Space

Case Study on Dimensionality Reduction of Double-Helix: We compare UMAP McInnes et al. (2018) embeddings based on various metrics- angle and l^2-distance as compared to dot-product. Dot-product produces poor seperation in edge case of double-hellix as shown by Figure 6. We believe that it has to be a proper metric to produce good embedding. Dot-product on the other hand, does not follow the properties of metrics, as well as have both magnitude and angular component to it, making it difficult to disentangle the measurement.

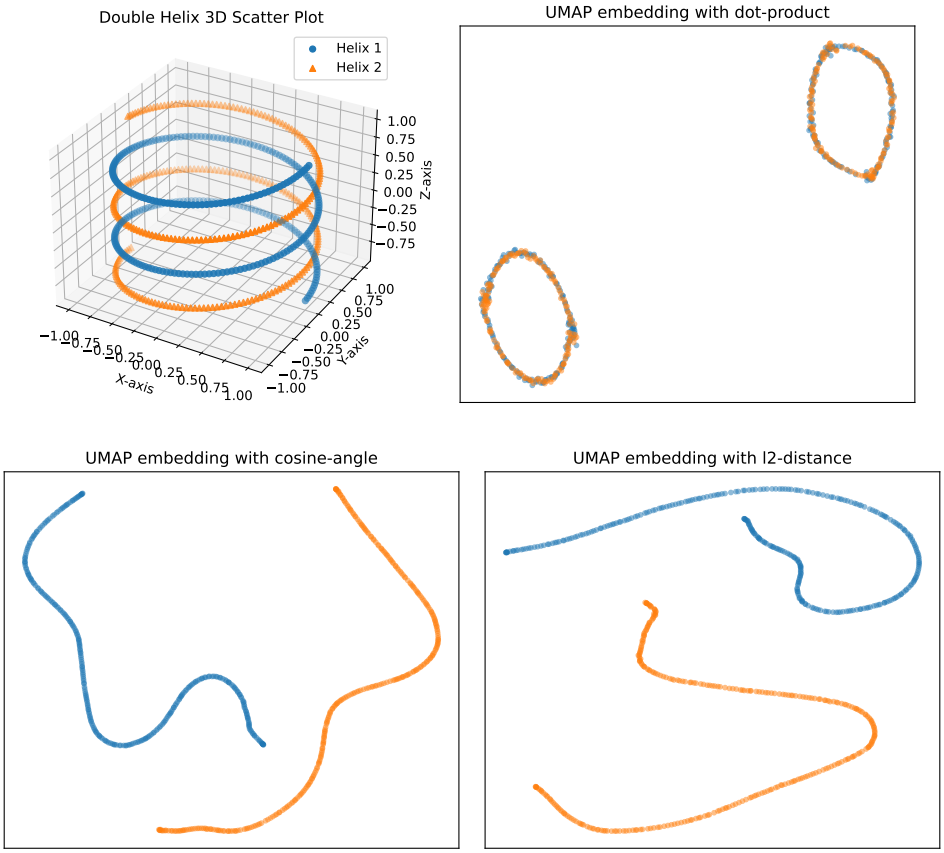

Figure 6: Mapping of double hellix using different measure of distance.

Figure 7: Data Interpretation using local activation visualization. Points represent the UMAP of weights/center and the size represents similarity to the test sample ($\times$), T represents the target class of the sample. *(TOP)* using l2-norm *(MID)* using cosine angle *(BOT)* using dot-product. We choose the plot for when dot-product does poor(left) and when distance does poor(right) among 32 activation samples. Zoom in for details.

Interpretation of High Dimensional Data: UMAP McInnes et al. (2018) has been used widely to visualize high dimensional dataset. However, it projects input to low dimension with not enough information to infer about the inputs from embeddings only. To increase information about input, we add extra dimension to the embedding points with local activation as shown in Figure 7. Here, we use negative exponential to get local similarity measure, a continuous version of k-nearest neighbour as used on UMAP. The mapping of centers in low dimension along with magnitude of activation allows us to interpret high-dimensional input space in embedding space. The dimension of local activation is higher than the embedding space, providing more information to infer about input. The activations are always local for metrics, i.e. angle and l^2-norm. For dot product activation, the embedding activations are not always local suggesting non-local activations on the high dimension itself.

10 Reversibility of Metrics as Transform

Metrics are reversible, i.e., we can reconstruct inputs given metrics from a set of points. Generally, N+1 metric values from unique centers intersect at unique point. There exists certain values of measures/distances, that do not intersect uniquely, such as scaled down l^2-distances, or when centers are collinear.

Invertibility of Linear Transform: Linear Transforms $f : \mathbf{R}^M \rightarrow \mathbf{R}^N$, where $N \geq M$, can be inverted using Moore–Penrose inverse (or Pseudoinverse). Scientific computing libraries provide library for this operation. The bias can be simply subtracted during the inversion.

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$$

This can be inverted as follows, where A^+ is the pseudoinverse.

$$\boldsymbol{x} = \boldsymbol{A}^+(\boldsymbol{y} - \boldsymbol{b})$$

Invertibility of "Inverse Stereographic Transform": We use inverse stereographic transform ($i-$stereo) to project nD space to (n+1)D space with nD sphere as data manifold. This transform is reversible, called as stereographic transform Wikipedia (d). Moreover, parametric linear transform or cosine angle on top of $i-$stereo is also reversible. Here, cosine($i-$stereo(x), $\boldsymbol{w}$) is a metric.

Invertibility of Angles: When we compute cosine angles, we need to normalize the vectors to unit magnitude of 1, and hence, the magnitude information is lost. Since, the cosine angles are the linear transform with unit vectors, the unit vectors are invertible. We can use extra angle, not from the origin but from extra known point to be able to reconstruct the magnitude information.

From the figure 8, we try to derive the inverse ($\boldsymbol{x}$, or length OB), with known A, α, $\hat{\boldsymbol{x}}$. Here, $\hat{\boldsymbol{x}}$ is calculated by pseudo-inverse of cosine-angles with the weights.

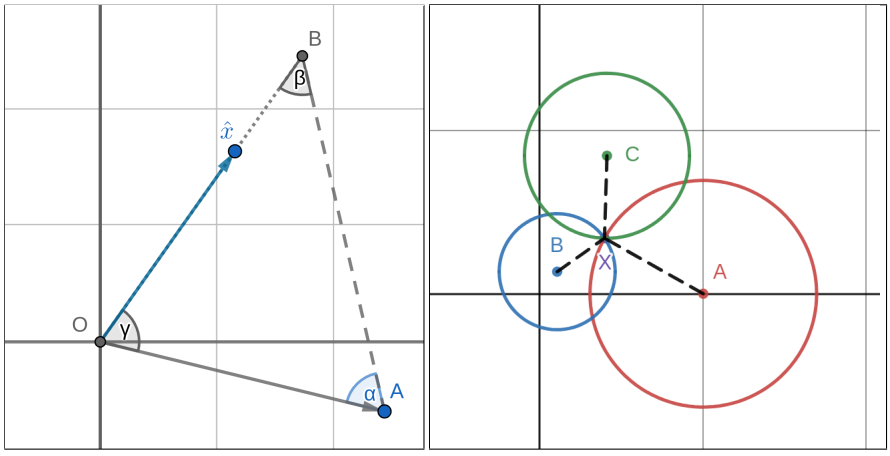

Figure 8: Inversion of input from angles (left) and distances(right).

Here, we can calculate angles: $\theta = \cos^{-1}(\hat{\boldsymbol{x}} \cdot \boldsymbol{a}/\|a\|)$ and $\beta = \pi - \theta - \alpha$. Now, using triangle law,

$$\frac{OB}{\sin\alpha} = \frac{OA}{\sin\beta}$$

$$OB = \frac{OA \sin \alpha}{\sin \beta}$$

Hence, with the magnitude OB known, reconstruction $\boldsymbol{x} = OB\hat{\boldsymbol{x}}$.

We consider this to be special case of triangulation, where N angles are calculated from input-origin-weight and, 1 extra angle calculated from origin-weight-input.

Invertibility of Euclidean Distance: Inverting a measure of euclidean distance from centers is inspired by the Global Positioning System (GPS) Bancroft (1985); Norrdine (2012). These system use distance from 4 known centers to locate a point in 3D space. We find euclidean distance can be inverted in any dimensions. If the dimension of point is N, then we need $N + 1$ distances from known centers to be able to reconstruct it. This could be done by solving the equation of n-Sphere for intersection.

The Algorithm 1 provides an implementation for inversion of euclidean distances. The derivation is left to the reader as an exercise. If the exact distance is not known, then scaled distances from N+2 points is needed for reconstruction, we find this with Gradient Descent based input reconstruction.

Algorithm 1 Multi-lateration: Euclidean Distance Inversion

```
def inverse_euclidean(C, d):
    ''' To find -> x: shape [N] -> given C: shape [N+1, N], d:[N+1]'''
    A = 2*(C[1:]-C[:-1])
    c2 = C**2
    Z = (c2[:-1]-c2[1:]).sum(dim=1, keepdim=True)
    invA = np.pinverse(A)

    d2 = d**2
    D = d2[:, :-1]-d2[:, 1:]

    x = np.matmul(invA, D.t()-Z).t()
    return x
```

11 On Reversibility of Generalized Metrics as Transform

Generalized distances such as convex and invex function does not have unique contour sets. The intersection of contours of different distances may intersect at non-unique points, i.e. the intersection of different contours of general convex function may not be unique.

Invertibility of Convex Contour Distance: We specialize the convex function to have same contour sets, which we call convex contour distance, i.e. the shape of the convex contour does not change Ma (2000) as shown in Figure 9. Such convex distance function have unique solution.

Similar to euclidean distance, convex distance produce 1-1 function. Even euclidean distance produce 1-1 function, however, any possible set of distances might not even intersect, and not have any input associated with it. Hence, euclidean along with convex contour distance produce 1-1 into function as shown in Figure 10, excluding the case when centers are collinear.

Reversibility of 1-Invex Contour Distance: Moreover, we can specialize the invex function as well to have same contour sets called invex contour distance. We find that if the contours can be joined with straight line from the center without intersection, then it has sensible contour sets (Figure 9 MID) which we call 1-Invex contour distance. If contours intersect with straight line from centers, the contour distance of such sets intersect with each other (Figure 9 RIGHT), unsuitable to use as general distance.

1-Invex contour distance can intersect at multiple points. Hence, the function produced is many-1 into as shown in Figure 10.

12 Properties of Metrics

Metrics: A metric space $(\mathbf{M}, d)$ is a set of input-space $(\mathbf{M})$ and distance function $d(\boldsymbol{x}, \boldsymbol{y})$ between any two elements in the space. The distance is measured by a function called a metric or distance function and satisfies following axioms for all $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathbf{M}$ Wikipedia (b).

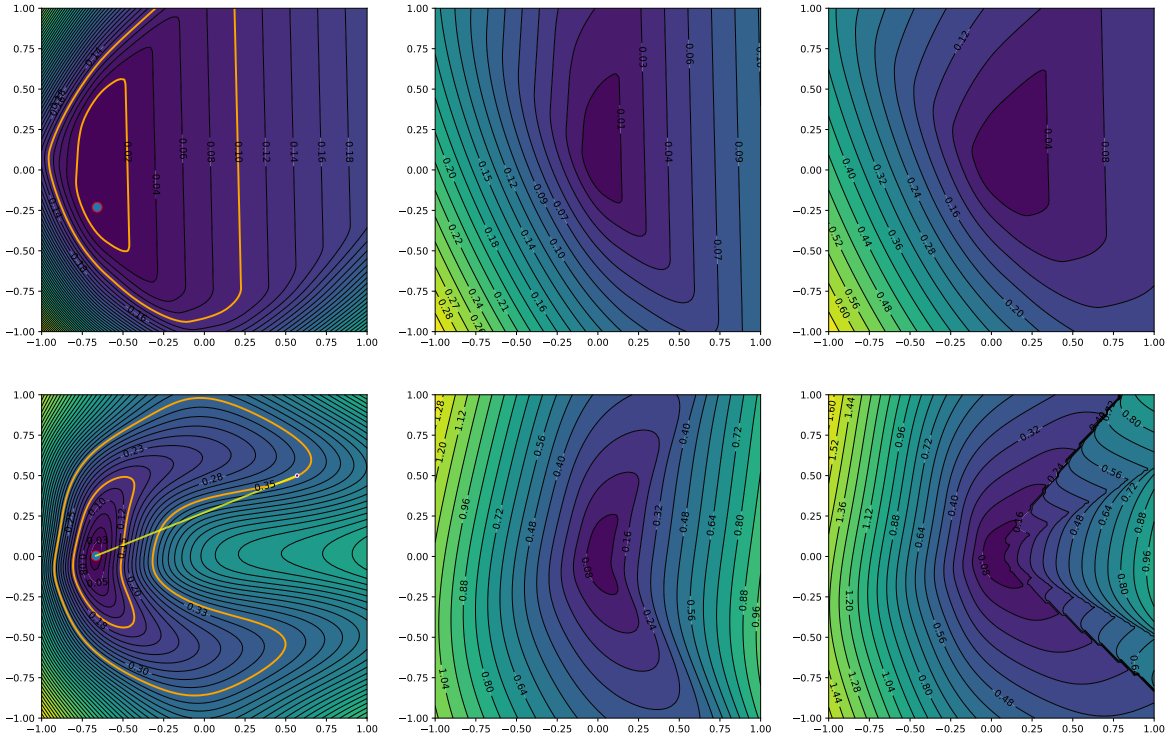

Figure 9: Contour Distance of **(TOP)** Convex and **(BOT)** Invex function. *LEFT*: General convex and invex functions with respective contour sets to create distance from, *MID*: contour distance from smaller set and *RIGHT*: contour distance from larger set. Here, invex contour distance does not create proper function as two contours from same set intersect with a straight line from center.

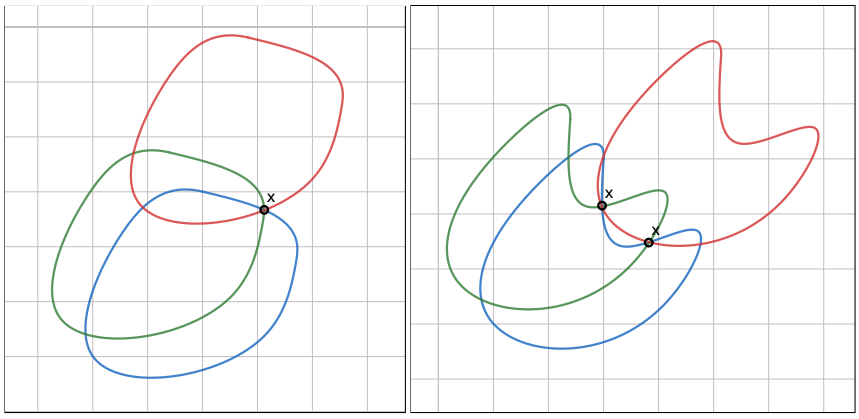

Figure 10: Intersection of convex-sets(*LEFT*) and invex-sets(*RIGHT*).

1. The distance from a point to itself is zero:
$$d(\boldsymbol{x}, \boldsymbol{x}) = 0$$
2. (Positivity) The distance between two distinct points is always positive:
$$\text{If } \boldsymbol{x} \neq \boldsymbol{y} \text{ , then } d(\boldsymbol{x}, \boldsymbol{y}) > 0$$
3. (Symmetry) The distance from x to y is always the same as the distance from y to x:
$$d(\boldsymbol{x}, \boldsymbol{y}) = d(\boldsymbol{y}, \boldsymbol{x})$$

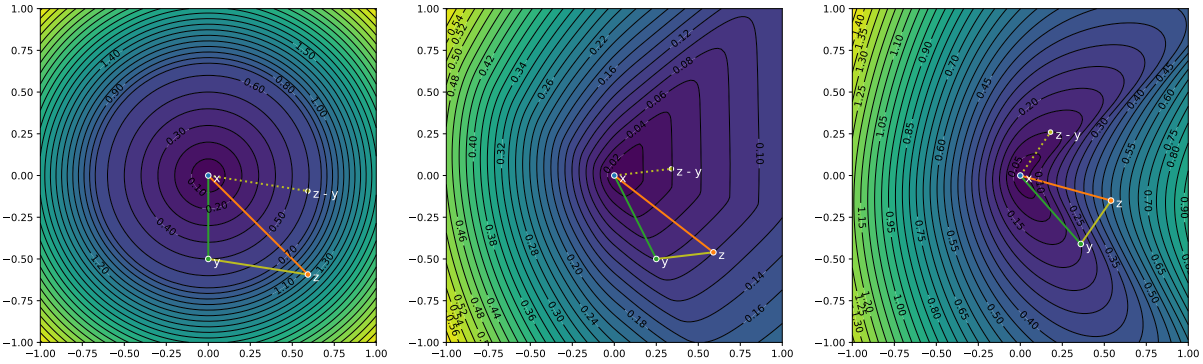

Figure 11: Triangular inequality in various types of metrics. We test if $d(\boldsymbol{x}, \boldsymbol{z}) \leq d(\boldsymbol{x}, \boldsymbol{y}) + d(\boldsymbol{y}, \boldsymbol{z})$ is $true$ or $false$ *(LEFT)*: Modified l^2-norm; $1.2 \leq 0.5 + 0.6$ is $false$, *(MID)*: Convex contour distance; $0.1 \leq 0.08 + 0.04$ is $true$ and *(RIGHT)*: 1-Invex contour distance; $0.5 \leq 0.2 + 0.1$ is $false$. Zoom in for details.

4. The triangle inequality holds:

$$d(\boldsymbol{x}, \boldsymbol{z}) \leq d(\boldsymbol{x}, \boldsymbol{y}) + d(\boldsymbol{y}, \boldsymbol{z})$$

Such metrics include l^p norm induced metrics and cosine-angle. However, we can generalize metrics by relaxing axioms $1 - 4$ with different types of measures.

Modified l^2-distance: Consider a convex piecewise function $f(x) = max(x, s * (x - b) + b)$ where, $s > 1$ is the slope of a piece and b is the location of intersection of the pieces. Then the overall distance function given by $f(l^2(x, y))$ is not a proper metric. It follows axiom $1, 2, 3$ except the triangle inequality (shown in Figure 11 LEFT). This generalization follows properties of Semimetrics.

General convex function: A general convex function like portrayed in Figure 9 does not have symmetric distance (axiom-3), and similar to previous convex piecewise modification, it also does not follow triangle inequality. It only follows axiom 1 and 2. This generalization follows properties of Premetrics.

General invex function: A general invex function also follows the same axioms of a general convex function.

Convex contour distance: A convex function with linear increase in distance from the minima is depicted by Figure 11 MID. Here, the convex contour distance follows the triangle inequality. However, it does not follow symmetry property, following axiom $1, 2$, and 3 only. This generalization follows properties of Quasimetrics.

1-invex contour distance: A 1-invex contour distance function with linear increase in distance from minima is dipicted by Figure 11 RIGHT. Such generalized metrics do not follow symmetry and the triangle inequality, following only axiom 1 and 2. This generalization also follows the properties of Premetrics.

For example, the function $f(\boldsymbol{x}, \boldsymbol{y}) = 0.9 + 0.1\cos(2d) - e^{-d^2}$; $d = \|\boldsymbol{x} - \boldsymbol{y}\|$ is symmetric but does not follow triangle inequality. The function is Semimetric, despite not being always increasing function of distance.

Overall, we find multiple types of Premetrics and Semimetrics unique in their construction. The generalized metrics have very small constrain that many function with different properties satisfy the axioms.

13 Discussion and Conclusion

We find that various metrics as transform work in the modern Neural Network architectures with use of proper normalization. Moreover, we find multiple cases where metrics provide better interpretation of neurons in neural network and provide way to initialize neurons with data. This work also studies generalization of metrics and their properties. We believe that metrics could increase intrepretability and allow for low-level modification of large models.

REFERENCES

Brandon Amos, Lei Xu, and J Zico Kolter. Input convex neural networks. In *International Conference on Machine Learning*, pp. 146–155. PMLR, 2017.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Stephen Bancroft. An algebraic solution of the gps equations. *IEEE transactions on Aerospace and Electronic Systems*, (1):56–59, 1985.

Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

D Broomhead and D Lowe. Multivariable functional interpolation and adaptive networks, complex systems, vol. 2, 1988.

Hanting Chen, Yunhe Wang, Chunjing Xu, Boxin Shi, Chao Xu, Qi Tian, and Chang Xu. Addernet: Do we really need multiplications in deep learning? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1468–1477, 2020.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

Steven Fortune. Voronoi diagrams and delaunay triangulations. In *Handbook of discrete and computational geometry*, pp. 705–721. Chapman and Hall/CRC, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Sara Hooker. The hardware lottery. *Communications of the ACM*, 64(12):58–65, 2021.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. pmlr, 2015.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

Xinlin Li, Mariana Parazeres, Adam Oberman, Alireza Ghaffari, Masoud Asgharian, and Vahid Partovi Nia. Euclidnets: An alternative operation for efficient inference of deep learning models. *arXiv preprint arXiv:2212.11803*, 2022.

Lihong Ma. *Bisectors and Voronoi diagrams for convex distance functions*. Citeseer, 2000.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

Vitali Nesterov, Fabricio Arend Torres, Monika Nagy-Huber, Maxim Samarin, and Volker Roth. Learning invariances with generalised input-convex neural networks. *arXiv preprint arXiv:2204.07009*, 2022.

Abdelmoumen Norrdine. An algebraic solution to the multilateration problem. In *Proceedings of the 15th international conference on indoor positioning and indoor navigation, Sydney, Australia*, volume 1315, 2012.

Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2017. doi: 10.23915/distill.00007. https://distill.pub/2017/feature-visualization.

Christopher Olah. Neural networks, manifolds, and topology. *Blog post*, 2014.

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

Suman Sapkota and Binod Bhattarai. Input invex neural network. *arXiv preprint arXiv:2106.08748*, 2021.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. *Advances in neural information processing systems*, 28, 2015.

Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in Neural Information Processing Systems*, 34:24261–24272, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Georges Voronoi. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. deuxième mémoire. recherches sur les parallélloèdres primitifs. *Journal für die reine und angewandte Mathematik (Crelles Journal)*, 1908(134):198–287, 1908.

Wikipedia. Lp-space. https://en.wikipedia.org/wiki/Lp_space, a. Accessed: Januray 2024.

Wikipedia. Metric-space. https://en.wikipedia.org/wiki/Metric_space, b. Accessed: Januray 2024.

Wikipedia. Softmax function. https://en.wikipedia.org/wiki/Softmax_function, c. Accessed: Januray 2024.

Wikipedia. Stereographic projection. https://en.wikipedia.org/wiki/Stereographic_projection, d. Accessed: Januray 2024.