Hyperbolic VQ-VAEs

Sam Acquaviva

Abstract

How can we construct meaning out of an image? Two key ingredients of human scene parsing are *hierarchy* and *symbols*. However, in neural networks that are tasked to find low-dimensional semantic embeddings, the representations are not necessarily symbolic or hierarchical. One neural network architecture, *Vector-quantized Variational Autoencoders* (VQ-VAEs), learns discrete representations. Another architecture, *Hyperbolic Autoencoders* (hVAEs), learns representations that are implicitly hierarchical. In this work, we combine the vector-quantization of VQ-VAEs with the hyperbolic embedding space of hVAEs to train a *Hyperbolic Vector-quantized Variational Autoencoder* in a step towards learning discrete hierarchical representations. Then, we implement and benchmark various training methods to improve training stability.

1 Introduction

Capturing the meaning of a photo is a difficult problem. In Figure 1, the left-most photo is more similar pixel-wise to the right-most photo, however, its meaning is more similar to the center photo: both depict a bird sitting on a branch. In other words, the two photos on the left have a more similar semantic representation. Although it is not perfectly understood how humans construct meaning or how this meaning is represented (Fodor, 1975; Margolis and Laurence, 1999; Medin and Schaffer, 1978; Rosch and Mervis, 1975; Smith and Medin, 2002), it is generally agreed upon that people use compositionality as part of the process (Carey, 2011; Fodor and Lepore, 1996; Kamp and Partee, 1995; Markman, 1991; Osherson and Smith, 1981; Smith and Osherson, 1984), meaning that people construct meaning of an object (e.g. bird) from its sub-parts (e.g. beak, feathers, wings, etc...) and their combination (e.g. the eyes are above the wings, the beak is in front of the head, etc...). Compositionality requires *discrete symbols* to determine what is a "part" and *hierarchy* to determine which parts are sub-parts.

In machine learning, finding semantic representations is an active area of research. Typically, this comes in the form of mapping images to points in

Figure 1: From left to right: a Kingfisher bird sitting on a branch, a Scarlet Macaw sitting on a branch, a Blue Hippo Tang fish. Although the left-most photo is more semantically similar to the center photo, it is actually more similar to the right-most photo using a pixel-wise comparison.

Euclidean space where images with similar semantic content are close to each other in the embedding space. There has been work to build in discreteness and hierarchy into these architectures, but they each have limits as discussed in Section 2.2.

We posit that incorporating compositionality in the neural network architectures will improve semantic representations for two reasons. For one, the meaning of images are inherently compositional so we can expect that building in this architectural prior will better fit the data. Additionally, encoding hierarchy can increase the computational efficiency of vector-quantized neural networks by enabling lower-dimensional representations as discussed in Section 3.1. This increased computational efficiency enables using more symbols for the same computational budget, which has been shown to increase performance (van den Oord et al., 2017).

In order to build-in compositionality, we will build-in both discreteness (through vector-quantization) and hierarchy (through hyperbolic embeddings). Our contributions are:

- Implementation and benchmarking of a vector-quantized variational autoencoder with hyperbolic embeddings and codebooks.[1]
- Analyses of training pathologies that do and do not improve stability of training and test performance.

[1]All code for implementation and experiments can be found at https://github.com/samacqua/hyperbolic-vqvae

2 Background

2.1 Hyperbolic Space

Euclidean space is the space that most aligns with people's intuitions about geometry; the sum of the internal angles of a triangle sum to 180 degrees, the distance between two points in 2-dimensions is $d(p,q) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2}$, 2 parallel lines are equidistant everywhere, and polygons of different areas can be similar.

All of these properties can be derived from 5 postulates called *Euclid's Postulates*. Of interest is the final postulate, which says that given a line l and a point p, there is only 1 line that goes through p and is parallel to l. This postulate is much less intuitive than the first four, and mathematicians have tried to derive this postulate from the first four without success (Niemiec and Pikul, 2022). So, many have explored what properties arise when you don't accept the final postulate.

If one rejects the fifth postulate and, instead, posits that there is more than 1 line that goes through p and is parallel to l, then we derive *hyperbolic space*. This space differs from people's geometric intuitions; the sum of the internal angles of a triangle sum to less than 180 degrees, the distance between two points is shown in Equation 1, 2 parallel lines are converge in 1 direction and diverge in the other, and there are no examples of polygons of different areas that are similar.

Hyperbolic space has the useful property that it is intrinsically better fit to represent tree structures than Euclidean space (Ganea et al., 2018). In Euclidean space, the circumference of a circle increases linearly with the circle's radius while in hyperbolic space, the circle's circumference increases exponentially with its radius. Since the number of nodes in a tree increases exponentially with the tree's depth, hyperbolic space allows one to lay out a tree without cluttering; placing a node far enough from its parent gives the node nearly the same amount of space as its parent for laying out its own children.

2.2 Autoencoders

One model class that attempts to learn semantic representations is the autoencoder as shown in Figure 2. Autoencoders find these representations by using a neural network encoder to compress an image to a lower-dimensional encoding and a neural network decoder to reconstruct the input from this lower-dimensional representation.

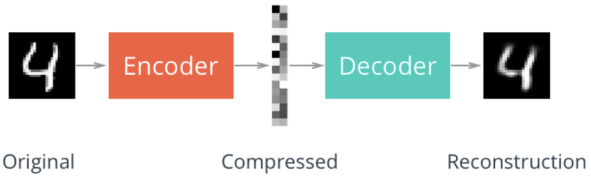

Figure 2: A depiction of an autoencoder architecture. The original image is fed through a neural network encoder to generate a lower dimensional representation in latent space. This compressed representation is fed through a neural network decoder to reconstruct the input image.

2.2.1 Variational Autoencoders

One drawback of vanilla autoencoders is that, after training, sampling from the semantic representation in order to generate a reconstructed image is difficult. Since the only restriction on the bottleneck is that it maximizes the reconstruction accuracy, the distribution over the latent space is difficult to determine (Bank et al., 2020). This is a drawback because then one cannot use the learned representation to generate semantically plausible images.

The Variational Autoencoder (VAE) deals with this issue by encoding the input as a distribution over the latent space instead of as a single point. Although one can use any distribution to model the latent space, let us assume we are using a normal distribution. In this case, the VAE's encoder will output the mean and standard deviation of a multivariate normal distribution. A sample from this distribution is then fed to the decoder to reconstruct the image. In addition to the regularization effect of enforcing a Gaussian latent space, this parameterization has the added benefit that we can now sample from the semantic space after training because we now know the distribution of the latent space.

2.2.2 Vector-Quantized Variational Autoencoders

Another variant of the autoencoder is the Vector-quantized variational autoencoder (VQ-VAE) (van den Oord et al., 2017). VQ-VAEs "quantize" the low-dimensional representation by maintaining a discrete list of vectors in Euclidean space, called the "codebook". Then, instead of feeding the output of the encoder directly to the decoder, the VQ-VAE uses the codebook vector that is closest to the encoding as the input to the decoder. For images, the image is usually divided into a grid of patches which are each quantized independently.

In this way, VQ-VAEs generate the representation of an image as a set of these discrete codes. This discreteness could in itself be an advantage since it is more explainable and more robust to adversarial examples (Huh et al., 2022). Additionally, one can train a prior over the latent codes after training in order to be able to sample from the codebook to generate semantically plausible images.

2.3 Hierarchical Autoencoders

Although the VQ-VAE effectively introduces discrete codes into the semantic representation, it does not address the issue of incorporating hierarchy. Here, we will go over two different approaches to incorporating hierarchy in autoencoders.

2.3.1 Hyperbolic VAEs

One way to incorporate hierarchy in VAEs is to use hyperbolic space at the bottleneck instead of Euclidean space. There has been work using hyperbolic distributions to parameterize and sample from the latent space instead of Euclidean distributions (Nagano et al., 2019). This architecture, called the Hyperbolic Variational Autoencoder (hVAE), derives a hyperbolic equivalent of the normal distribution called the wrapped normal. Experimentally, hVAEs outperform their Euclidean counterparts in tasks that require hierarchical representations, especially when the embedding dimension is small.

2.3.2 Hierarchical VQ-VAEs

Another way to incorporate hierarchy is through multiple quantization layers. Huh et al. found some success with this technique; earlier layers seem to encode lower-dimensional features while later layers incorporate higher dimensional features (Huh et al., 2022). This paper also presented different training pathologies that we use in order to improve training such as codebook initialization techniques, bounded measure spaces, ensuring smoothness, and grouped vector quantization.

3 hVQ-VAE

3.1 Motivation

The Hierarchical VQ-VAE is the only previously discussed architecture that incorporates both discrete symbols and hierarchy. They found that incorporating hierarchy on top of the discrete codes of VQ-VAEs, through the form of multiple quantization layers, was beneficial. This is one major aspect of the motivation as it confirms our intuition that incorporating a compositionality prior into the architecture will better fit the data.

Additionally, the work on hyperbolic VAEs found that incorporating hierarchy through hyperbolic space improved the performance of VAEs, especially in lower dimensions (Nagano et al., 2019). In VQ-VAEs, this low-dimensionality performance is important because quantization operation scales linearly with the size of the codebook and the size of each codebook vector. So, decreasing the codebook vector dimensionality allows us to increase the codebook size with the same computational budget. It has been shown that increasing the number of codebooks increases VQ-VAE performance (van den Oord et al., 2017).

So, we provide a different approach to incorporating hierarchy in VQ-VAEs; instead of using multiple quantization layers, we take inspiration from the success of hVAEs and use a hyperbolic embedding space to create the hyperbolic Vector-quantized Variational Autoencoder (hVQ-AE).

3.2 Architecture

The architecture of the hVQ-VAE is very similar to that of the VQ-VAE. The only difference is that the embeddings and codebooks are in hyperbolic space. Concretely, this means that the encoder and decoder for VQ-VAE and hVQ-VAE are identical. However, in the hVQ-VAE:

- the encoding is converted to hyperbolic space
- the codebooks are initialized using hyperbolic distributions
- the distance between the embedding and the codebook vectors are calculated using the hyperbolic distance function
- the hyperbolic codebook is converted to euclidean space before being passed to the decoder

For our experiments, we used a Resnet-style encoder and decoder, each with 3 layers.

3.3 Derivations

In order to train the hVQ-VAE, we needed to derive a few hyperbolic equivalents of Euclidean functions. First, since the embeddings and codebooks are in hyperbolic space, we need to use the hyperbolic distance function instead of the Euclidean distance function in order to implement the quantization function. In hyperbolic space, using the

Poincaré disk model, the distance between two points p and q is:

$$d(p,q) = \ln \frac{|aq|\ |pb|}{|ap|\ |qb|} \tag{1}$$

where a and b are the two ideal points where the unique hyperbolic line connecting them intersects the boundary, and $|xy|$ indicates the Euclidean length of the line segment connecting x and y in the model.

Additionally, we need to derive a hyperbolic equivalent of the traditional VQ-VAE loss function. In order to calculate the total loss, we need to calculate the reconstruction error and the distance between the un-quantized embedding and the quantized embedding.

The loss for the reconstruction error remains unchanged from the canonical VQ-VAE loss since the input and output are still in Euclidean space, as long as we correctly back-propagate through the hyperbolic embeddings. To do so, we use a geometric optimization library that preserves the gradients when translating between Euclidean and hyperbolic space. To calculate the distance component of the loss function, we simply use the hyperbolic distance function instead of the Euclidean distance function.

3.4 Optimization

VQ-VAEs are notoriously difficult to train stably. Additionally, hVAEs are experimentally more difficult to train than Euclidean VAEs. So, we incorporated many training methods from Hu et al. to improve performance of the hVQ-VAE.

3.4.1 Codebook Initialization

In order to improve the training of vector-quantized methods, we want to reduce the gradient error by reducing the difference between the embedding and the quantized embedding. To this end, we want to initialize the codebooks such that, even before training, the quantized embedding is close to the unquantized embedding. We reimplemented Hu et al.'s experiments on the effect of different codebook initialization techniques and confirmed their result that using data-dependent initialization both decreases KL-divergence between the embedding and the quantized embedding, and improves experimental performance.

3.4.2 Loss Function Changes

Another way to decrease the gradient approximation error is to ensure that the decoder function is smooth. We can ensure this by adding a smoothness term to the loss function. We found, contrary to Hu et al.'s results, that ensuring the smoothness of the decoder does not improve the experimental performance of hVQ-VAEs.

Another change to the loss function is to use a bounded similarity metric. However, this is problematic in hyperbolic space. In Euclidean space, one can use the cosine distance instead of the canonical Euclidean distance function to compare embeddings to codebooks with the added benefit of the cosine distance being bounded between 0 and 1. However, the hyperbolic analog of the cosine distance is lower-bounded by 0 but has no finite upper bound. We can trivially create a bounded measure by taking the inverse of this unbounded measure, but this measure loses its theoretical similarity to its Euclidean counterpart and experimentally provides no benefit.

3.4.3 Grouped Vector Quantization

Hu et. al provides a new quantization method called Grouped Vector Quantization (GCV) which helps to prevent under-training of the sub-vectors. In GVC, each vector is divided into a group of smaller vectors, called sub-vectors. Each sub-vector is independently quantized using a shared codebook. The quantized sub-vectors are then re-concatenated to create the final embedding vector. We found that GCV improves the performance of hVQ-VAEs.

4 Performance against VQ-VAE

We tested hVQ-VAEs on three tasks: image reconstruction, image classification, and a synthetically generated hierarchical representation task. For image reconstruction and classification, we used two datasets: MNIST and CIFAR-10. MNIST is a dataset of hand-written single digits, while CIFAR-10 is a dataset of photos of 10 objects. Both of these datasets are considered simple datasets which we focus on due to the lack of compute availability. We used both data-dependent codebook initialization and GVC as outlined in the previous section.

For the hierarchical representation task, we generated an artificial dataset of binary trees and then measured how similar the latent space distances were to the Hamming distance of the underlying tree representation. For this task, we used data-

dataset	d	**VQ-VAE**	**hVQ-VAE**
MNIST	2	0.97 $\pm$ 0.12	0.97 $\pm$ 0.04
	5	0.73 $\pm$ 0.15	0.84 $\pm$ 0.06
	10	0.72 $\pm$ 0.13	0.85 $\pm$ 0.09
	20	0.63 $\pm$ 0.07	0.74 $\pm$ 0.04
CIFAR-10	2	0.92 $\pm$ 0.29	0.88 $\pm$ 0.07
	5	0.95 $\pm$ 0.35	0.79 $\pm$ 0.04
	10	0.82 $\pm$ 0.25	0.74 $\pm$ 0.02
	20	0.82 $\pm$ 0.17	0.71 $\pm$ 0.06

Table 1: Quantitative comparison of Hyperbolic VQ-VAE (hVQ-VAE) against Vanilla VQ-VAE on the MNIST dataset in terms of mean-squared-error for a reconstructed test set, for different latent dimensionalities d. We calculated the mean and the $\pm$ 2 standard deviations with five different experiments.

dataset	d	**VQ-VAE**	**hVQ-VAE**
MNIST	2	0.77 $\pm$ 0.04	0.81 $\pm$ 0.05
	5	0.85 $\pm$ 0.02	0.85 $\pm$ 0.02
	10	0.84 $\pm$ 0.04	0.83 $\pm$ 0.09
	20	0.88 $\pm$ 0.02	0.87 $\pm$ 0.07
CIFAR-10	2	0.31 $\pm$ 0.01	0.33 $\pm$ 0.01
	5	0.39 $\pm$ 0.02	0.38 $\pm$ 0.03
	10	0.37 $\pm$ 0.03	0.36 $\pm$ 0.03
	20	0.42 $\pm$ 0.02	0.39 $\pm$ 0.04

Table 2: Quantitative comparison of Hyperbolic VQ-VAE (hVQ-VAE) against Vanilla VQ-VAE on the MNIST dataset in terms of classification accuracy, for different latent dimensionalities d. We calculated the mean and the $\pm$ SD with five different experiments.

dependent codebook initialization.

4.1 Reconstruction

We found that, on both MNIST and CIFAR-10, the hVQ-VAE and VQ-VAE performance was not statistically significant, as shown in Table 1. The hVQ-VAE consistently had a slightly lower mean-squared-error in lower dimensions than VQ-VAE ($d = 2$), but it was not statistically significant. Qualitatively, we found no distinctive difference between the generated images from the two architectures, as shown in Figure 3.

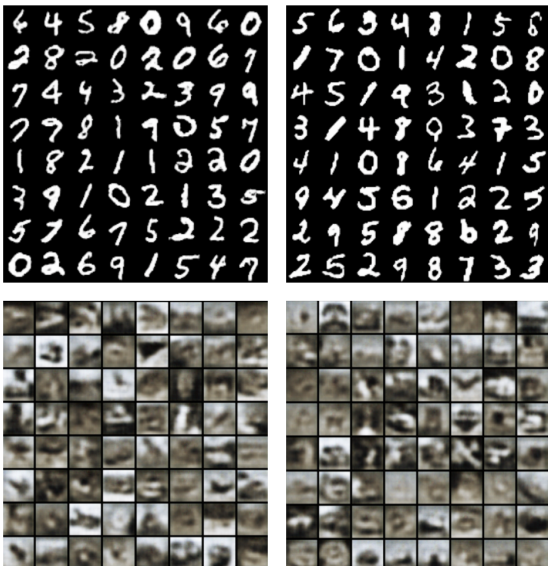

Figure 3: Unconditional image generation for MNIST (top) and CIFAR-10 (bottom), using a hVQ-VAE (left) and a VQ-VAE (right).

4.2 Classification

We found that, on both MNIST and CIFAR-10, the hVQ-VAE and VQ-VAE performance was not statistically significant, as shown in Table 2. The hVQ-VAE consistently had a slightly higher classification accuracy in lower dimensions than VQ-VAE ($d = 2$), but it was not statistically significant.

4.3 Hierarchical Representations

Model	Correlation	Correlation w/ Noise
VQ-VAE	0.446 $\pm$ 0.03	0.189 $\pm$ 0.01
hVQ-VAE	0.597 $\pm$ 0.10	0.271 $\pm$ 0.05

Table 3: Results of tree embedding experiments for the Hyperbolic VQ-VAE and Vanilla VQ-VAE. We calculated the mean and the $\pm$ 2 SD with five different experiments.

We used the dataset generation technique from the hVAE paper to quantitatively compare how hierarchical each architectures' codebooks are (Nagano et al., 2019). To construct the dataset, we first generate a binary tree of depth $d = 8$. Then, we obtain a binary representation for each node in the tree such that the Hamming distance between any pair of nodes is the same as the distance on the graph representation of the tree. Then, for each node, we randomly flip each coordinate value with probability $\epsilon = 0.1$.

We use a 3-layer MLP to map each binary set into the 2-dimensional latent space of the VQ-VAE or hVQ-VAE. After training, we compare the correlation between the distance in data-space (Ham-

ming distance of the underlying nodes) with the distance in embedding space (Euclidean distance between the quantized embeddings for VQ-VAE, hyperbolic distance between quantized embeddings for hVQ-VAE). For this experiment, we set the number of codebook vectors to 64.

We found that the hVQ-VAE had higher correlations with the underlying tree structure distances than the Euclidean VQ-VAE on both the un-noised binary tree and the noisy binary trees, as shown in table 3. This result gives us confidence that the hVQ-VAE does have some greater ability to represent hierarchy.

5 Conclusions

5.1 Discussion

We found that on both image construction and image classification tasks for two simple baselines, hVQ-VAE performs comparably to VQ-VAEs, even in low-dimensional settings. However, we found that hVQ-VAEs outperform VQ-VAEs in representing hierarchical structures in embedding space. We have three theories to join these two results:

1. The datasets we tested on (MNIST and CIFAR-10) are not that hierarchical. Although humans may use hierarchy to understand images (as in Figure 1), neural networks may exploit other properties of the images to learn useful semantic representations. In this case, enforcing a space that has the benefit of more space for hierarchy would not necessarily improve performance. In this case, the hyperbolic space adds no benefit since the network does not learn to exploit the implicit tree structure, and it adds the negative of difficult optimization.

2. The image patches that are quantized are too small to benefit from encoded hierarchy. It may be that, even if MNIST and CIFAR-10 could benefit from hierarchy, the current hVQ-VAE is unable to exploit it because it is encoding patches of each image and a given patch is too low resolution to benefit from hierarchy (see Figure 4). Our preliminary experiments compared the performance of VQ-VAEs and hVQ-VAEs with different patch sizes on MNIST and CIFAR-10, but found no statistically significant results. However, it may be that the image quality of the patches in these datasets are still too low.

3. MNIST and CIFAR-10 are more difficult datasets than the generated binary tree dataset that require optimization for hVQ-VAEs. As noted in the original paper on hVAEs, there are difficulties with hyperbolic optimization. It may be that hVQ-VAEs would provide a performance benefit as is, but it requires additional work in improving optimization.

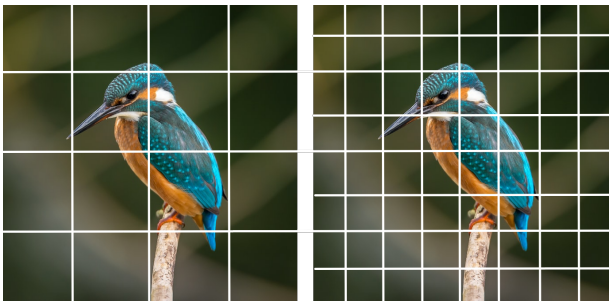

Figure 4: The same photo with two different patch sizes. In the photo on left, each patch is larger so it may benefit more from hyperbolic space. On the right, each patch is encoding a smaller region of the image, and the hierarchical benefit of hyperbolic space may not be exploited.

We also found that, for hVQ-VAEs, data-dependent initialization and grouped vector quantization improves performance, but a smooth decoder function and a bounded similarity measure does not. Although we provided justification for why the bounded similarity measure does not improve performance, it is still an open question why the smooth decoder function does not improve performance with a hyperbolic embedding space.

5.2 Future Work

If our proposed architecture, the hVQ-VAE, had shown improvement over euclidean VQ-VAEs, a natural extension would be to combine hVQ-VAEs with the work of Hu et al.. Concretely, this would entail using multiple hyperbolic vector quantization layers. However, work must be done to answer the question raised in 5.1 before trying this direction.

In future work, we can test each hypothesis. To test the first idea, we can generate image datasets that are explicitly hierarchical and train hVQ-VAEs on that dataset. To test hypothesis 2, we can use more compute on datasets with larger images to see if hVQ-VAEs outperform VQ-VAEs. If it is true that the hVQ-VAE does not outperform VQ-VAEs because the quantized patches are too low

resolution, then training with larger patches on higher-resolution images would show a benefit of hVQ-VAEs. To test the third idea, we can try more training changes, such as training with increased floating point precision as suggested in (Sa et al., 2018).

References

Dor Bank, Noam Koenigstein, and Raja Giryes. 2020. Autoencoders. *CoRR*, abs/2003.05991.

S. Carey. 2011. *The Origin of Concepts*. Oxford series in cognitive development. Oxford University Press.

J.A. Fodor. 1975. *The Language of Thought*. Language and thought series. Harvard University Press.

Jerry Fodor and Ernest Lepore. 1996. The red herring and the pet fish: why concepts still can't be prototypes. *Cognition*, 58(2):253–270.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. 2018. Hyperbolic neural networks. *CoRR*, abs/1805.09112.

Minyoung Huh, Brian Cheung, Pulkit Agrawal, and Phillip Isola. 2022. Improved techniques for hierarchical vector-quantization.

Hans Kamp and Barbara Partee. 1995. Prototype theory and compositionality. *Cognition*, 57(2):129–191.

Eric Margolis and Stephen Laurence. 1999. *Concepts: Core Readings*. MIT Press.

Ellen M Markman. 1991. *Categorization and Naming in Children: Problems of Induction*. The MIT Press.

Douglas L. Medin and Marguerite M. Schaffer. 1978. Context theory of classification learning. *Psychological Review*, 85:207–238.

Yoshihiro Nagano, Shoichiro Yamaguchi, Yasuhiro Fujita, and Masanori Koyama. 2019. A wrapped normal distribution on hyperbolic space for gradient-based learning.

Piotr Niemiec and Piotr Pikul. 2022. Hyperbolic geometry for non-differential topologists. *Mathematica Slovaca*, 72(1):165–184.

Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2017. Neural discrete representation learning. *CoRR*, abs/1711.00937.

Daniel N. Osherson and Edward E. Smith. 1981. On the adequacy of prototype theory as a theory of concepts. *Cognition*, 9(1):35–58.

Eleanor Rosch and Carolyn B Mervis. 1975. Family resemblances: Studies in the internal structure of categories. *Cognitive Psychology*, 7(4):573–605.

Christopher De Sa, Albert Gu, Christopher Ré, and Frederic Sala. 2018. Representation tradeoffs for hyperbolic embeddings. *CoRR*, abs/1804.03329.

Edward E. Smith and Douglas L. Medin. 2002. *Foundations of Cognitive Psychology: Core Readings*. The MIT Press.

Edward E. Smith and Daniel N. Osherson. 1984. Conceptual combination with prototype concepts*. *Cognitive Science*, 8(4):337–361.