HYPERBOLIC IMAGE-TEXT REPRESENTATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Visual and linguistic concepts naturally organize themselves in a hierarchy, where a textual concept "dog" entails all images that contain dogs. Despite being intuitive, current large-scale vision and language models such as CLIP (Radford et al., 2021) do not explicitly capture such hierarchy. We propose MERU, a contrastive model that yields hyperbolic representations of images and text. Hyperbolic manifolds have suitable geometric properties to embed tree-like data, so MERU can better capture the underlying hierarchy in image-text datasets. Our results show that MERU learns a highly interpretable and structured representation space while maintaining (or improving) CLIP's performance on standard transfer tasks like zero-shot classification, retrieval and resource constrained deployment.

1 INTRODUCTION

Visual-semantic hierarchy. It is commonly said that *'an image is worth a thousand words'* – consequently, images contain a lot more information than the sentences which we might typically use to describe them. For example, given the middle image in Fig. 1 one might describe it as *'itap (i took a picture) of a cat with a sleeping puppy'* or with a less specific sentence like *'tired doggo'* or *'itap of my cat'*. These are not merely diverse descriptions, but contain varying levels of detail about the underlying semantic contents of the image. As humans, we are able to reason about the relative detail in each caption, and are able to organize the concepts into a multimodal hierarchy (also called visual-semantic hierarchy (Vendrov et al., 2015)), namely, *'tired doggo'* $\rightarrow$ *'itap of a cat with a sleeping puppy'* $\rightarrow$ the image with the sleeping puppy and the cat. Providing multimodal models access to this inductive bias about vision and language has the potential to improve generalization (Radford et al., 2021), interpretability (Selvaraju et al., 2017) and enable better exploratory data analysis of large-scale datasets (Schuhmann et al., 2022; Radford et al., 2021).

Euclidean embeddings. Approaches such as CLIP (Radford et al., 2021) and ALIGN (Jia et al., 2021) have catalyzed a lot of recent progress in computer vision by showing that Transformer-based (Vaswani et al., 2017) models trained using large amounts of image-text data from the internet can yield transferable representations, and such models can perform *zero-shot* recognition and retrieval using natural language queries. All these models represent images and text as vectors in a high-dimensional euclidean, affine space and normalize the embeddings to unit L2 norm. However, such a choice of geometry can find it hard to capture the visual-semantic hierarchy.

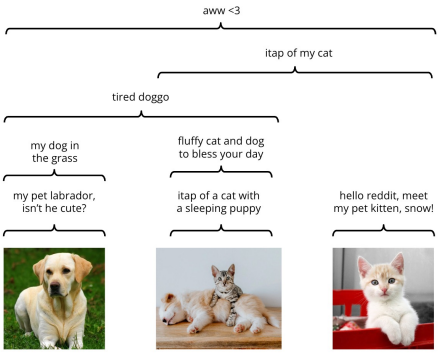

Figure 1: If we think of each image and text above as a *'concept'*, one can notice that text (*'tired doggo'*) is more generic than an image of a tired doggo (which might have more details such as a cat or grass). In such a view, vision and language can be thought of as being a part of a common visual-semantic hierarchy. Our work enables one to learn and infer this visual-semantic hierarchy purely from large amounts of image-text paired data.

An affine euclidean space treats all embedded points in the same manner, with the same distance metric being applied to all points (Murphy, 2013). Conceptually, this can cause issues when modeling hierarchies – a *generic* concept (closer to the *root node* of the hierarchy) is close to many other concepts compared to a *specific* concept (which is only close to its immediate neighbors). Thus, a euclidean space can find it hard to pack all the images that say a generic concept *'itap of my cat'* should be

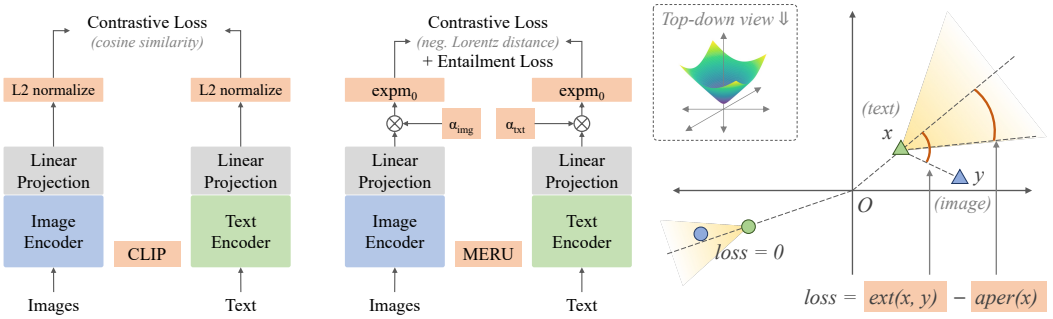

Figure 2(a): **Model design.** MERU shares similar architectural components as standard image-text contrastive models like CLIP. We process images and text using two separate encoders. While CLIP projects the embeddings to a unit hypersphere (via $L2$ normalization), we lift them onto the Lorentz hyperboloid using exponential map operator. We adapt the contrastive loss to use hyperbolic distance as a similarity metric, and use a special entailment loss to induce structure in learned hyperbolic representations.

Figure 2(b): **Entailment loss.** We enforce that an image embedding $\mathbf{y}$ lies inside a cone projected by the embedding of its paired text $\mathbf{x}$. This loss is implemented as the difference of exterior angle $\angle O\mathbf{xy}$ and half aperture of an imaginary cone at $\mathbf{x}$. No loss is applied if the image embedding is already inside the cone *(left quadrant)*.

close to while also respecting the embedding structure for *'itap of a cat with a sleeping puppy'*. Such issues are handled naturally by a hyperbolic space (Fig. 2b top-left) where the volume increases exponentially as we move away from the origin (Lee, 2019). This allows one to pack a lot of concepts close to a general concept (*'cat'*) by placing it close to the origin (Nickel & Kiela, 2017), and more specific concepts further away. Thus, dissimilar specific concepts (*'itap of a cat with a sleeping puppy'* and *'my pet kitten, snow'*) can be far away from each other while still being close to *'cat'*.

MERU embeddings. In this work, we train the first large-scale contrastive image-text models that embed data in a hyperbolic representation space (Nickel & Kiela, 2017) – MERU that captures the visual-semantic hierarchy (Fig. 1). Importantly, the visual-semantic hierarchy emerges with the hyperbolic structure, given access only to paired image-text data during training such models. Practically, MERU confers multiple benefits, including: (a) better performance on entailment tasks such as text-based image retrieval, (b) more efficient usage of the embedding space, enabling better performance in resource-constrained settings (c) a more interpretable latent space that allows one to reason about the generality *vs* specificity. Overall, our contributions are:

- We introduce MERU, the first implementation of deep hyperbolic representations we are aware of, training ViTs (Dosovitskiy et al., 2021) with 12M image-text pairs.
- We provide a strong re-implementation of CLIP that outperforms previous re-implementations (Mu et al., 2022) at comparable data scale, and systematically demonstrate the benefits of hyperbolic representations over this baseline on zero-shot classification and retrieval (Radford et al., 2021), and effectiveness for small embedding dimensions (Kusupati et al., 2022).
- We analyse the learned representations by MERU to demonstrate its potential for exploratory data analysis and visualization of large-scale multimodal datasets.

2 Approach

In this section, we discuss the modeling pipeline and learning objectives of MERU to learn hyperbolic representations of images and text. We use tools of hyperbolic geometry throughout our discussion, see Appendix A for a thorough discussion of the relevant topics.

Our model design is based on CLIP (Radford et al., 2021) due to its simplicity and scalability. As shown in Fig. 2a, we process images and text using two separate encoders, and obtain embedding vectors of a fixed dimension n. Beyond this step, we introduce two differences on CLIP: (1) instead of L2 normalization, we transfer the Euclidean embeddings from the encoder to the Lorentz hyperboloid, (2) we use the negative of geodesic distance in the contrastive loss, instead of cosine similarity.

We also use an additional textual entailment loss, illustrated for low-dimensions in Fig. 2b. See Appendix B for a detailed walkthrough of our model design and entailment loss.

Table 2: **Zero-shot image classification.** We train ViT models of varying sizes and transfer them *zero-shot* to 20 classification benchmarks. Best performances in for each task are in **bold**.

ViT		Food-101	CIFAR-10	CIFAR-100	CUB	SUN397	Cars	Aircraft	DTD	Pets	Caltech-101	Flowers	STL-10	EuroSAT	RESISC45	MNIST	PCAM	CLEVR	Country211	SST2	ImageNet	Average
S/16	CLIP (repro.)	74.5	60.1	24.4	34.2	27.5	11.0	1.4	14.5	73.7	63.9	47.0	88.2	18.4	31.3	10.3	50.2	**17.8**	5.2	50.1	34.3	31.1
	MERU (ours)	74.6	63.1	24.6	34.5	28.6	10.7	1.3	15.7	75.3	63.2	50.0	90.3	28.2	32.7	10.7	50.5	14.8	5.2	50.1	34.2	31.4
B/16	CLIP (repro.)	78.9	65.5	33.4	33.8	29.8	14.3	1.4	15.8	80.0	**68.5**	50.9	92.2	25.5	30.9	10.2	**54.1**	15.8	5.8	**51.5**	37.9	39.8
	MERU (ours)	78.8	67.7	32.7	35.3	30.9	14.6	1.7	**16.4**	**80.4**	**68.5**	**52.1**	92.5	**29.6**	34.4	**12.6**	49.7	14.0	5.6	50.0	37.5	40.3
L/16	CLIP (repro.)	80.3	**72.0**	**36.4**	36.6	32.0	**18.5**	1.1	16.0	79.7	68.3	48.6	**93.8**	27.1	35.3	11.0	51.2	14.8	6.1	51.0	38.4	40.9
	MERU (ours)	**80.6**	68.7	35.5	**37.8**	**33.0**	17.7	**2.2**	16.1	80.3	67.5	**52.1**	93.7	27.0	**36.5**	11.6	52.7	12.8	**6.2**	49.3	**38.8**	**41.0**

3 EXPERIMENTS

Our main objective in the experiments is to establish the competitiveness of hyperbolic representations from our MERU models in comparison with their Euclidean counterparts. We also probe the trained mdoels to assess the interpretability conferred by the hyperbolic structure.

Our primary comparison is with CLIP (Radford et al., 2021) which we re-implement and train using the RedCaps dataset (Desai et al., 2021). Similarly, we train MERU models with same training hyperparameters for fair and direct comparison. We train three models for CLIP and MERU, having Vision Transformers (Dosovitskiy et al., 2021) of varying capacity: ViT-S/B/L all with patch size 16. For quantitative evaluations, we perform zero-shot classification and retrieval as proposed by (Radford et al., 2021). See Appendix D for a description of training details and evaluation setup.

Enforcing structure improves zero-shot transfer. Promisingly, the improved performance of MERU on recall-based measures does not come at the expense of precision (Murphy, 2013). On standard zero-shot classification evaluations for vision and language models, we find that the hyperbolic representations from MERU are competitive with their euclidean counterparts (Tab. 2) and outperform them on average across 20 datasets and three model architectures, namely ViT-B/S/L with patch sizes of 16. This encouraging result shows that across different architectures incorporating the structure retains (and even slightly improves) performance in standard classification settings.

Table 1: **Zero-shot retrieval (Recall@5 on COCO and Flickr30K).** MERU yields better recall than CLIP. Best perf in each column shown in **bold**.

ViT		txt → img		img → txt	
		COCO	F30	COCO	F30
S/16	CLIP	29.2	39.9	29.5	36.6
	MERU	30.0	40.2	30.6	38.2
B/16	CLIP	31.9	44.1	31.9	40.6
	MERU	32.1	**44.7**	33.0	42.2
L/16	CLIP	31.0	40.6	32.0	42.2
	MERU	**32.1**	41.7	**33.5**	**43.0**

Text-based image retrieval. We next evaluate MERU and CLIP models the standard COCO and Flickr30K retrieval tasks (Karpathy & Fei-Fei, 2015; Hodosh et al., 2013) (Tab. 1). Hyperbolic representations from MERU consistently outperform CLIP on both datasets, for both *image* → *text* retrieval as well as *text* → *image* retrieval. This is encouraging evidence that the hyperbolic structure is useful for retrieval tasks.

3.1 PROBING THE HYPERBOLIC REPRESENTATION SPACE

In this section, we analyze the hyperbolic representations learned by our MERU models to understand the semantic hierarchy present in the training data. In all our analysis, we use the MERU ViT-L/16 models trained using RedCaps. Using these models, we embed 12M image-text pairs in RedCaps as ≈24M embedding vectors.

Image-to-image interpolations. We interpolate between two images containing different semantic concepts to reveal the hierarchy learned in the representation space. In order to interpolate from $\mathbf{x}$ → $\mathbf{y}$, we take the logarithmic map $\log_{\mathbf{y}}(\mathbf{x})$ (Eqn. (8)) to project $\mathbf{x}$ onto the tangent space at $\mathcal{T}_{\mathbf{y}}$. We then do linear interpolations between the projection and the $\mathbf{x}$, and then apply $\exp_{\mathbf{y}}(\cdot)$ to each point. We interpolate and find 30 equally spaced points along the geodesic on the hyperboloid. At

each interpolated points, we retrieve the nearest text representation from a pool of $\approx 500K$ captions from top-20 largest subreddits in RedCaps.

Fig. 3a shows *unique* captions encountered between selected images. Notice how the interpolation between *images* goes through more generic *textual* concepts. This shows that representations learned by MERU capture meaningful, abstract semantic structure underlying image-text datasets.

Image-to-origin interpolations. We interpolate from an image to origin, which represents the most generic concept. This interpolation would capture all concepts applicable to an image at different levels of abstraction. Notice in Fig. 3b, how the concepts become more abstract as we move closer to the origin of the hyperboloid.

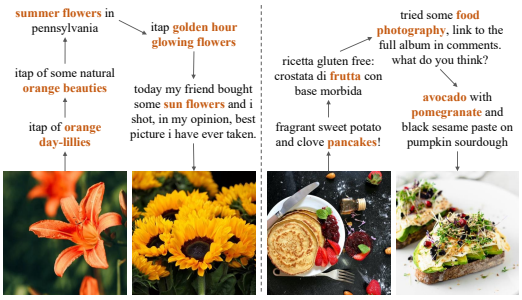

Figure 3(a): **Image-to-image interpolations with MERU.** We find that generic concepts like *'food photography'* (right) are encountered between specific concepts *'pancakes'* and *'avocado'* while interpolating and retrieving nearest-neighbor caption in the representation space. We highlight some visually salient objects within captions for better readability.

Figure 3(b): **Image-to-origin interpolations with MERU.** This is similar to Fig. 3a, wherein the second image is replaced by the origin. The concept depictions become more generic as we move towards the origin (e.g. third image $\rightarrow$ *labrador* $\rightarrow$ *best friend*, and fourth image $\rightarrow$ *espresso martini* $\rightarrow$ *cocktail*).

4 Related Work

Deep metric learning (Sohn, 2016; Song et al., 2015) has been used to embed vision and language data into a common semantic space (Frome et al., 2013; Karpathy & Fei-Fei, 2015). The motivations at the time included the possibility of improving vision backbones (Frome et al., 2013), enabling zero shot learning by expressing novel categories as sentences (Frome et al., 2013; Elhoseiny et al., 2013), and better ranking / retrieval of image-caption pairs (Karpathy & Fei-Fei, 2015; Young et al., 2014). More recent approaches utilizing large vision transformer models, contrastive metric learning and large-scale pretraining such as CLIP (Radford et al., 2021) and ALIGN (Jia et al., 2021) have helped better realize the motivations of those earlier works in practice. While these works learn euclidean embeddings, our MERU embeddings explicitly work in the hyperbolic space, that is conceptually better for embedding the tree-like structures in the visual-semantic hierarchy (Fig. 1) underlying vision and language data. Our results (Sec. 3) demonstrate that this yields similar or better performance (in terms of recall) as previous work, but lends structure and interpretability to the latent space as a benefit.

5 Conclusion

We learn large-scale image-text representations (MERU) that capture concept hierarchies underlying the two modalities. Our key innovation is in bringing the advances in learning hyperbolic representations to practical, large scale deep learning models with state-of-the-art transformer backbones (for both images and text). The resulting model is competitive or more performant than approaches such as CLIP while also capturing hierarchical knowledge which allows one to make powerful inferences such as reasoning about images at different levels of abstraction, and performing semantic interpolations between images. Beyond this, our model also provides clear performance gains for small embedding dimensions (which are useful in resource constrained settings). We hope this work catalyzes progress in representations for web-scale unstructured data.

REFERENCES

Mount Meru. https://en.wikipedia.org/wiki/Mount_Meru. 9

Mina Ghadimi Atigh, Julian Schoep, Erman Acar, Nanne van Noord, and Pascal Mettes. Hyperbolic image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4453–4462, June 2022. 18

Abeba Birhane, Vinay Uday Prabhu, and Emmanuel Kahembwe. Multimodal datasets: misogyny, pornography, and malignant stereotypes. *arXiv preprint arXiv:2110.01963*, 2021. 14

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – Mining Discriminative Components with Random Forests. In *ECCV*, 2014. 15

Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *CVPR*, 2021. 14

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015. 15

Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *ICCV*, 2021. 14

Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote Sensing Image Scene Classification: Benchmark and State of the Art. *Proceedings of the IEEE*, 105:1865–1883, 2017. 15

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing Textures in the Wild. In *CVPR*, 2014. 15

Adam Coates, Andrew Ng, and Honglak Lee. An Analysis of Single Layer Networks in Unsupervised Feature Learning. In *AISTATS*, 2011. https://cs.stanford.edu/~acoates/papers/coatesleeng_aistats_2011.pdf. 15

Karan Desai, Gaurav Kaul, Zubin Aysola, and Justin Johnson. RedCaps: Web-curated image-text data created by the people, for the people. In *NeurIPS Datasets and Benchmarks*, 2021. 3, 14

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2021. 2, 3, 14

Albert Einstein. Zur elektrodynamik bewegter körper. *Annalen der physik*, 4, 1905. 9

Mohamed Elhoseiny, Babak Saleh, and Ahmed Elgammal. Write a classifier: Zero-shot learning using purely textual descriptions. In *2013 IEEE International Conference on Computer Vision*, pp. 2584–2591, 2013. doi: 10.1109/ICCV.2013.321. 4

Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories. *CVPR Workshop*, 2004. 15

Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc Aurelio Ranzato, and Tomas Mikolov. DeViSE: A deep Visual-Semantic embedding model. In C J C Burges, L Bottou, M Welling, Z Ghahramani, and K Q Weinberger (eds.), *Advances in Neural Information Processing Systems 26*, pp. 2121–2129. Curran Associates, Inc., 2013. 4

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic entailment cones for learning hierarchical embeddings. April 2018. 11, 12

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022. 14

Patrick Helber, Benjamin Bischke, Andreas R. Dengel, and Damian Borth. EuroSAT: A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12:2217–2226, 2019. 15

Micah Hodosh, Peter Young, and Julia Hockenmaier. Framing image description as a ranking task. *Journal of Artificial Intelligence Research*, 2013. 3

Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, July 2021. URL https://doi.org/10.5281/zenodo.5143773. 14

Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision. In *ICML*, 2021. 1, 4

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross B. Girshick. CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning. In *CVPR*, 2017. 15

Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 3, 4, 15

Valentin Khrulkov, Leyla Mirvakhabova, E. Ustinova, I. Oseledets, and Victor S. Lempitsky. Hyperbolic image embeddings. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6417–6427, 2020. 18

Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3D Object Representations for Fine-Grained Categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, 2013. 15

Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. 15

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham M. Kakade, Prateek Jain, and Ali Farhadi. Matryoshka representation learning, 2022. URL https://openreview.net/forum?id=9njZa1fm35. 2, 18

Marc Teva Law, Renjie Liao, Jake Snell, and Richard S. Zemel. Lorentzian distance learning for hyperbolic representations. In *ICML*, 2019. 10

Matthew Le, Stephen Roller, Laetitia Papaxanthos, Douwe Kiela, and Maximilian Nickel. Inferring concept hierarchies from text corpora via hyperbolic embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3231–3241, Florence, Italy, July 2019. Association for Computational Linguistics. 11

Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010. 15

John M. Lee. *Introduction to Riemannian Manifolds*. Graduate Texts in Mathematics. Springer International Publishing, 2019. ISBN 9783319917542. URL https://books.google.com/books?id=UIPltQEACAAJ. 2, 9, 13

Yanghao Li, Haoqi Fan, Ronghang Hu, Christoph Feichtenhofer, and Kaiming He. Scaling language-image pre-training via masking. *arXiv preprint arXiv:2212.00794*, 2022. 14

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 14

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *ICLR*, 2019. 14

Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew B. Blaschko, and Andrea Vedaldi. Fine-Grained Visual Classification of Aircraft. *arXiv preprint arXiv:1306.5151*, 2013. 15

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. In *ICLR*, 2018. 14

Norman Mu, Alexander Kirillov, David Wagner, and Saining Xie. Slip: Self-supervision meets language-image pre-training. In *ECCV*, 2022. 2, 14, 17

Kevin P. Murphy. *Machine learning : a probabilistic perspective*. MIT Press, Cambridge, Mass. [u.a.], 2013. ISBN 9780262018029 0262018020. URL https://www.amazon.com/Machine-Learning-Probabilistic-Perspective-Computation/dp/0262018020/ref=sr_1_2?ie=UTF8&qid=1336857747&sr=8-2. 1, 3

Maximilian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In *NIPS*, 2017. 2

Maximilian Nickel and Douwe Kiela. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *ICML*, 2018. 10, 12, 18

M-E. Nilsback and A. Zisserman. Automated Flower Classification over a Large Number of Classes. In *ICVGIP*, 2008. 15

Omkar Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *CVPR*, 2012. 15

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 14

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning Transferable Visual Models From Natural Language Supervision. In *ICML*, 2021. 1, 2, 3, 4, 11, 14, 15, 16, 17

John G. Ratcliffe. *Foundations of Hyperbolic Manifolds*. Graduate Texts in Mathematics. Springer New York, 2006. ISBN 9780387331973. URL https://books.google.com/books?id=JV9m8o-ok6YC. 9

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115: 211–252, 2014. 18

Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021. 14

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022. 1, 14

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017. 1

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *ACL*, 2016. 14

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*, 2018. 14

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, A. Ng, and Christopher Potts. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *EMNLP*, 2013. 15

Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In D Lee, M Sugiyama, U Luxburg, I Guyon, and R Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. 4, 11

Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. November 2015. 4

Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. YFCC100M: The New Data in Multimedia Research. *Communications of the ACM*, 2016. 14

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021. 14

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 1, 14

Bastiaan S Veeling, Jasper Linmans, Jim Winkens, Taco Cohen, and Max Welling. Rotation Equivariant CNNs for Digital Pathology. *arXiv preprint arXiv:1806.03962*, 2018. 15

Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. Order-embeddings of images and language. *arXiv preprint arXiv:1511.06361*, 2015. 1

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge J. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. 2011. 15

Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. SUN database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010. 15

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014. doi: 10.1162/tacl_a_00166. URL https://aclanthology.org/Q14-1006. 4, 15

Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, André Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, Lucas Beyer, Olivier Bachem, Michael Tschannen, Marcin Michalski, Olivier Bousquet, Sylvain Gelly, and Neil Houlsby. A large-scale study of representation learning with the visual task adaptation benchmark. In *CVPR*, 2019. 15

Note on the method name: *Meru (or* Sumeru*) is a five-peaked mountain that holds high spiritual significance in Hinduism and other eastern religions like Buddhism and Jainism. It symbolizes the center of all physical, metaphysical, and spiritual universes (mou). We name our approach MERU, as the origin of hyperbolic space is a universal, unobserved concept entails everything, and plays a more vital role than in Euclidean (or generally, affine) spaces.*

A PRELIMINARIES

We begin with a brief overview of Riemannian manifolds (Appendix A.1) and discuss some concepts of hyperbolic geometry that are essential to our approach (Appendix A.2). For a detailed treatment of the topic, we recommend textbooks by Ratcliffe (Ratcliffe, 2006) and Lee (Lee, 2019).

A.1 RIEMANNIAN MANIFOLDS

A *smooth surface* is a two-dimensional sheet which is *locally Euclidean* – every point on the surface has a local neighborhood which can be mapped to $\mathbb{R}^2$ via a differentiable and invertible function. *Smooth manifolds* extend the notion of smooth surfaces to higher dimensions.

A *Riemannian manifold* $(\mathcal{M}, g)$ is a smooth manifold $\mathcal{M}$ equipped with a *Riemannian metric* g. The metric g is a collection of inner product functions $g_{\mathbf{x}}$ for all points $\mathbf{x} \in \mathcal{M}$, and varies smoothly over the manifold. At any point $\mathbf{x}$, the inner product $g_{\mathbf{x}}$ is defined in the *tangent space* $\mathcal{T}_{\mathbf{x}}\mathcal{M}$, which is a Euclidean space that gives a linear approximation of $\mathcal{M}$ at $\mathbf{x}$. Euclidean space $\mathbb{R}^n$ is also a Riemannian manifold, where g is the standard dot product.

Our main topic of interest are hyperbolic spaces, which are Riemannian manifolds with *constant negative curvature*. They are fundamentally different from Euclidean spaces that are *flat* (zero curvature). A hyperbolic manifold of n dimensions cannot be represented with $\mathbb{R}^n$ in a way that preserves both distances and angles. There are five popular models of hyperbolic geometry that either represent n-dimensional hyperbolic spaces either in $\mathbb{R}^n$ while distorting distances and/or angles (e.g. Poincaré ball model), or as a sub-manifold of $\mathbb{R}^{n+1}$ (e.g. the Lorentz model). We use the Lorentz model of hyperbolic geometry for developing MERU, which we briefly discuss next.

A.2 LORENTZ MODEL OF HYPERBOLIC GEOMETRY

The Lorentz model represents a hyperbolic space of n dimensions on the upper half of a two-sheeted hyperboloid in $\mathbb{R}^{n+1}$. See **??** (right, top-left quadrant) for an illustration of $\mathcal{L}^2$ in $\mathbb{R}^3$.

Hyperbolic geometry has a direct connection to the study of special relativity theory (Einstein, 1905). We borrow some of its terminology in our discussion – we refer to the hyperboloid's axis of symmetry as *time dimension* and all other axes collectively as *space dimensions*. Concretely, we can view every vector $\mathbf{x} \in \mathbb{R}^{n+1}$ in terms of its space and time components $[\mathbf{x}_s, x_t]$, where $\mathbf{x}_s \in \mathbb{R}^n$ and $x_t \in \mathbb{R}$.

Definition. The Lorentz model possessing a constant curvature $-c$ is defined as a following set of vectors:

$$\mathcal{L}^n = \{\mathbf{x} \in \mathbb{R}^{n+1} : \langle \mathbf{x}, \mathbf{x} \rangle_{\mathcal{L}} = -1/c, c > 0\} \tag{1}$$

where $\langle \cdot, \cdot \rangle_{\mathcal{L}}$ denotes the *Lorentzian inner product*. This inner product is induced by the Riemannian metric of Lorentz model. For two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{n+1}$, it is computed as:

$$\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}} = \sum_{i=0}^{n-1} x_i y_i - x_n y_n = \langle \mathbf{x}_s, \mathbf{y}_s \rangle - x_t y_t \tag{2}$$

The induced *Lorentzian norm* is $\|\mathbf{x}\|_{\mathcal{L}} = \sqrt{|\langle \mathbf{x}, \mathbf{x} \rangle_{\mathcal{L}}|}$. Every point on the hyperboloid satisfies the following constraint:

$$x_t = \sqrt{1/c + \|\mathbf{x}_s\|^2} \tag{3}$$

Geodesics. A *geodesic* is the shortest path between two points on the manifold. Geodesics in the Lorentz model are curves traced by the intersection of the hyperboloid with hyperplanes passing

through the origin of $\mathbb{R}^{n+1}$. Distance along the geodesic connecting two points $\mathbf{x}, \mathbf{y} \in \mathcal{L}^n$ is:

$$d_{\mathcal{L}}(\mathbf{x}, \mathbf{y}) = \sqrt{1/c} \cdot cosh^{-1}(-c\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}}) \tag{4}$$

Tangent space. The tangent space at any point $\mathbf{x} \in \mathcal{L}^n$ is a Euclidean space of vectors that are orthogonal to $\mathbf{x}$ according to the Lorentzian inner product:

$$\mathcal{T}_{\mathbf{x}}\mathcal{L}^n = \{\mathbf{v} \in \mathbb{R}^{n+1} : \langle \mathbf{x}, \mathbf{v} \rangle_{\mathcal{L}} = 0\} \tag{5}$$

Any vector in ambient space $\mathbf{u} \in \mathbb{R}^{n+1}$ can be projected to the tangent space $\mathcal{T}_{\mathbf{x}}\mathcal{L}^n$ via an orthogonal projection:

$$\mathbf{v} = \text{proj}_{\mathbf{x}}(\mathbf{u}) = \mathbf{u} + c\langle \mathbf{x}, \mathbf{u} \rangle_{\mathcal{L}} \mathbf{x} \tag{6}$$

When $\mathbf{x}$ is the origin of hyperbolic space ($[\mathbf{x}_s, x_t] = [\mathbf{0}, \sqrt{1/c}]$), this projection simplifies to $[\mathbf{v}_s, v_t] := [\mathbf{u}_s, 0]$.

Exponential and logarithmic maps. The *exponential map* provides a way to map vectors from tangent spaces onto the manifold. For a point $\mathbf{x}$ on the hyperboloid, it is defined as $\text{expm}_{\mathbf{x}} : \mathcal{T}_{\mathbf{x}}\mathcal{L}^n \rightarrow \mathcal{L}^n$ with the expression:

$$\text{expm}_{\mathbf{x}}(\mathbf{v}) = cosh(\sqrt{c}\|\mathbf{v}\|_{\mathcal{L}})\, \mathbf{x} + \frac{sinh(\sqrt{c}\|\mathbf{v}\|_{\mathcal{L}})}{\sqrt{c}\|\mathbf{v}\|_{\mathcal{L}}}\, \mathbf{v} \tag{7}$$

Intuitively the exponential map shows how $\mathcal{T}_x\mathcal{L}^n$ *folds* on the manifold. The inverse function is the *logarithmic map* ($\text{logm}_{\mathbf{x}} : \mathcal{L}^n \rightarrow \mathcal{T}_{\mathbf{x}}\mathcal{L}^n$), that is defined as:

$$\text{logm}_{\mathbf{x}}(\mathbf{y}) = \frac{cosh^{-1}(-c\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}})}{\sqrt{(c\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}})^2 - 1}}\, \text{proj}_{\mathbf{x}}(\mathbf{y}) \tag{8}$$

B Approach Details

Lifting embeddings onto the Hyperboloid. Consider a euclidean vector $\mathbf{v} \in \mathbb{R}^n$ from the image encoder or text encoder, after linear projection. We assume that this vector lies in the tangent space $\mathcal{T}_{\mathbf{O}}$ at the origin of the hyperboloid in $\mathbb{R}^{n+1}$, where $\mathbf{O} = [\mathbf{0}, \sqrt{1/c}]$. Note that here $\mathbf{0} \in \mathbb{R}^n$. Thus, our text and image encoders parameterize only the space-like components of the Lorentz model (Eqn. (3)).

We then apply the exponential map at origin $\text{expm}_{\mathbf{O}}$ to get space-like components of the point on hyperboloid. Here, Eqn. (7) reduces to (see Appendix C for derivation):

$$\mathbf{x}_s = \text{expm}_{\mathbf{O}}(\mathbf{v}) = \frac{sinh(\sqrt{c}||\mathbf{v}||)}{\sqrt{c}||\mathbf{v}||}\mathbf{v} \tag{9}$$

Note that norm above is the regular euclidean norm. We then compute the corresponding time-like component x_t from $\mathbf{x}_s$ using Eqn. (3). This yields a vector $\mathbf{x} = [\mathbf{x}_s, x_t] \in \mathcal{L}^n$, which is the lifted representation of the euclidean vector $\mathbf{v}$ produced by the image encoder or text encoder.

Our parameterization is simpler than previous work (Law et al., 2019; Nickel & Kiela, 2018) which parameterizes vectors in full ambient space $\mathbb{R}^{n+1}$. Since we parameterize only the space-like components $\mathbf{x}_s$, the resulting $\mathbf{x}$ *always* lies on the hyperboloid. This eliminates the need of projection (Eqn. (6)) and simplifies the expression of $\text{expm}_{\mathbf{O}}$.

Preventing numerical overflow. $\text{expm}_{\mathbf{O}}$ scales the euclidean embeddings $\mathbf{v}$ from encoders using an exponential operator. According to CLIP-style weight initialization, $\mathbf{v} \in \mathbb{R}^n$ would have an expected norm $= \sqrt{n}$. After exponential map, it becomes $e^{\sqrt{n}}$, which can be numerically large (*e.g.,* $n = 512$ and $c = 1$ gives $||\mathbf{x}_s|| \approx 6.7 \times 10^{10}$).

To fix this issue, we *scale* all vectors $\mathbf{v}$ in a batch before applying $\text{expm}_{\mathbf{O}}$ using two learnable scalars α_{img} and α_{txt}. We initialize them to $\sqrt{1/n}$ so that the euclidean embeddings have an expected unit

norm at initialization. We learn these scalars in logarithmic space to avoid collapsing all embeddings to zero. After training, they can be absorbed into the preceding projection layers.

Learning structured embeddings. Having lifted standard euclidean embeddings onto the hyperboloid, we next discuss the losses we use to enforce structure and semantics in representations learned by MERU. Recall that our motivation is to capture the visual-semantic hierarchy (Fig. 1) to better inform the generalization capabilities of vision and language models. For this, an important desiderata is a meaningful notion of distance between semantically similar text and image pairs (to provide better calibrated embeddings during zero-shot transfer). We also want the embeddings to respect the partial order imposed by the visual-semantic hierarchy to have better interpretability and structure. We do this with an entailment loss first proposed by (Le et al., 2019) which we rederive for arbitrary curvatures c.

B.1 Contrastive learning formulation

Given a batch of size B of image-text pairs and any j^{th} instance in batch, its image embedding $\mathbf{y}_j$ and text embedding $\mathbf{x}_j$ form a *positive* pair, whereas the remaining $B-1$ text embeddings in the batch $\mathbf{x}_i (i \neq j)$ form *negative* pairs.

In contrastive learning, we compute the negative of geodesic distance as a similarity measure (Eqn. (4)) for all B pairs in the batch. We divide these logits by a temperature τ and apply a softmax operator. Similarly, we also consider a contrastive loss for text, that treats images as negatives. The total loss $\mathcal{L}_{cont}$ is the average of these two losses computed for every image-text pair in the batch. Our implementation of the contrastive loss is same as the multi-class N-pair loss from (Sohn, 2016) used in CLIP (Radford et al., 2021) with the crucial difference being that we compute distances on the hyperboloid instead of cosine similarity.

B.2 Entailment loss

In addition to the contrastive loss, we adapt an entailment loss introduced in (Le et al., 2019; Ganea et al., 2018) [1] to enforce partial order relationships between related text and image pairs $\mathbf{x}$ and $\mathbf{y}$.

The key idea behind the entailment loss is to define an entailment cone for each text embedding $\mathbf{x}$, that narrows as we go farther from the origin (Fig. 2b). Formally, this cone is defined by the half-aperture (with a constant $K = 0.1$ used for setting boundary conditions (Ganea et al., 2018)):

$$\text{aper}(\mathbf{x}) = sin^{-1}\left(\frac{2K}{\sqrt{c}\|\mathbf{x}\|}\right) \tag{10}$$

Notice how the half-aperture is measured as half of the angle subtended by the cone at $\mathbf{x}$. Given this definition of the half-aperture, our next goal is to identify and penalize when the embedding of paired image $\mathbf{y}$ lies outside the entailment cone. For this, we measure the angle subtended by the arc from $\mathbf{y}$ to the axis of the entailment cone, shown as the exterior angle $\angle\mathbf{Oxy}$ in Fig. 2b:

$$\text{ext}(\mathbf{x}, \mathbf{y}) = cos^{-1}\left(\frac{y_t + x_t c\langle \mathbf{x}, \mathbf{y}\rangle_{\mathcal{L}}}{\|\mathbf{x}_s\|\sqrt{\left(c\langle \mathbf{x}, \mathbf{y}\rangle_{\mathcal{L}}\right)^2 - 1}}\right) \tag{11}$$

If the exterior angle is smaller than the aperture, then the partial order relation between $\mathbf{x}$ and $\mathbf{y}$ is satisfied and we need not penalize anything, while if the angle is greater, we need to reduce it. This is captured by the following loss function (written below for one example $\mathbf{x}$, $\mathbf{y}$):

$$\mathcal{L}_{entail}(\mathbf{x}, \mathbf{y}) = max(0, \text{ext}(\mathbf{x}, \mathbf{y}) - \text{aper}(\mathbf{x})) \tag{12}$$

We provide exact derivations of the above equations for half-aperture and exterior angle in Appendix B, which we generalize for arbitrary curvature compared to (Le et al., 2019) which uses a fixed curvature. Overall, our total loss is $\mathcal{L}_{cont} + \lambda\mathcal{L}_{entail}$ averaged over each minibatch.

[1] (Ganea et al., 2018) is more different to ours since they parameterize their representations according to the Poincaré ball model. (Le et al., 2019) use this loss with a fixed $c = 1$, which we extend to handle arbitrary, learned curvatures.

C APPROACH DERIVATIONS

We first provide the derivations for some of the expressions we provide in the main paper.

C.1 EXPONENTIAL MAP SIMPLIFICATION FOR SPACE-LIKE COMPONENTS

Recall that the output embedding from an encoder (image or text) $\mathbf{v}$ lies in a euclidean space. We lift it onto the hyperboloid to obtain the final hyperbolic representation $\mathbf{x}_s$. This usually comprises a projection on to the tangent space (Eqn. (6)) followed by an exponential map (Eqn. (7)).

We made two design choices in our modeling: (1) We only parameterize the space-like components of our hyperbolic representations $\mathbf{x}_s \in \mathbb{R}^n$. (2) We compute exponential map at the origin of hyperboloid, $\exp\mathbf{m}_{\mathbf{O}}$. Due to these choices, our expression of exponential map simplified from Eqn. (7) to Eqn. (9). We provide a derivation here.

First, we need to project the euclidean vector to the tangent space of the origin $\mathcal{T}_{\mathbf{O}}$. This simply amounts to setting the time-like component to zero. In other words, let the projected vector with space-time components be $\mathbf{v}_{st} \in \mathbb{R}^{n+1}; \mathbf{v}_{st} = [\mathbf{v}; 0]$. We find that $\mathbf{v}_{st}$ lies in the tangent space $\mathcal{T}_{\mathbf{O}}$, because if $\mathbf{O} = [\mathbf{0}; \sqrt{1/c}]$, then $\langle \mathbf{v}_{st}, \mathbf{O} \rangle_{\mathcal{L}} = 0$.

Also, notice that for $\mathbf{v}_{st} = [\mathbf{v}; 0]$ the Lorentzian norm simplifies to Euclidean norm of space-like components:

$$\|\mathbf{v}_{st}\|_{\mathcal{L}}^2 = -0^2 + v_1^2 + v_2^2 + \cdots + v_n^2 = \|\mathbf{v}\|^2 \tag{13}$$

Since $\mathbf{O} = [\mathbf{0}, \sqrt{1/c}]$, we can write the exponential map for the space-like components $\mathbf{v}$ as:

$$\begin{aligned} \exp\mathbf{m}_{\mathbf{O}}(\mathbf{v}) &= cosh(\sqrt{c}\|\mathbf{v}_{st}\|_{\mathcal{L}})\mathbf{0} + \frac{sinh(\sqrt{c}\|\mathbf{v}_{st}\|_{\mathcal{L}})}{\sqrt{c}\|\mathbf{v}_{st}\|_{\mathcal{L}}}\mathbf{v} \\ &= \frac{sinh(\sqrt{c}\|\mathbf{v}_{st}\|_{\mathcal{L}})}{\sqrt{c}\|\mathbf{v}_{st}\|_{\mathcal{L}}}\mathbf{v} \\ &= \frac{sinh(\sqrt{c}\|\mathbf{v}\|)}{\sqrt{c}\|\mathbf{v}\|}\mathbf{v} \end{aligned} \tag{14}$$

C.2 DERIVATION OF THE ENTAILMENT LOSS FOR ARBITRARY CURVATURE c

Half-aperture derivation. Previous work (Nickel & Kiela, 2018) learns hierarchies in the Poincaré ball model and derives the half-aperture and exterior angles for Poincaré embeddings. While (Ganea et al., 2018) use the Lorentz model, they provide derivations only for a fixed curvature $c = 1$. In our experiments, we treat curvature as a learnable parameter, which we found beneficial when scaling up, especially with ViT-L/16 models. Thus, we derive the half-aperture formula for the Lorentz model generalized to arbitrary curvatures $c > 0$.

We start with the expression introduced in (Ganea et al., 2018) – half-aperture for a point $\mathbf{x}_b$ on the Poincaré ball:

$$\text{aper}_b(\mathbf{x}_b) = \sin^{-1}\left(K\frac{1 - c\|\mathbf{x}_b\|^2}{\sqrt{c}\|\mathbf{x}_b\|}\right) \tag{15}$$

The Poincaré ball model and Lorentz model are isometric to each other – one can transform any point from the Poincaré ball ($\mathbf{x}_b$) to the Lorentz model ($\mathbf{x}_h$) using the following transformation:

$$\mathbf{x}_h = \frac{2\mathbf{x}_b}{1 - c\|\mathbf{x}_b\|^2} \tag{16}$$

The half-aperture of a cone should be invariant to the exact hyperbolic model we use, hence $\text{aper}_h(\mathbf{x}_h) = \text{aper}_b(\mathbf{x}_b)$. Substituting Eqn. (16) in Eqn. (15), we get the expression:

$$\text{aper}_h(\mathbf{x}_h) = \sin^{-1}\left(\frac{2K}{\sqrt{c}\|\mathbf{x}_h\|}\right)$$

Exterior angle derivation. We next derive the exterior angle on the Lorentz model for arbitrary curvatures $c > 0$, extending the formulation from (Ganea et al., 2018). Consider three points $\mathbf{O}$

(the origin), $\mathbf{x}$ (text embedding) and $\mathbf{y}$ (image embedding). Then, a hyperbolic triangle is a closed shape formed by pairwise geodesics connecting each pair of points. Similar to the euclidean plane, the hyperbolic plane also has its own law of cosines that allows us to talk about the angles in the triangle (Lee, 2019). Given pairwise distances between the points $o = d(\mathbf{x}, \mathbf{y})$, $x = d(\mathbf{O}, \mathbf{y})$ and $y = d(\mathbf{O}, \mathbf{x})$, we can write the interior angle, $\angle\mathbf{Oxy}$ as follows:

$$\angle\mathbf{Oxy} = \cos^{-1}\left[\frac{\cosh(o\sqrt{c})\cosh(q\sqrt{c}) - \cosh(p\sqrt{c})}{\sinh(o\sqrt{c})\sinh(q\sqrt{c})}\right] \tag{17}$$

correspondingly, the exterior angle is:

$$\text{ext} = \pi - \angle\mathbf{Oxy} \tag{18}$$

using the fact that $\pi - \cos^{-1}(t) = \cos^{-1}(-t)$, we get:

$$\text{ext} = \cos^{-1}\left[\frac{\cosh(p\sqrt{c}) - \cosh(o\sqrt{c})\cosh(q\sqrt{c})}{\sinh(o\sqrt{c})\sinh(q\sqrt{c})}\right] \tag{19}$$

Next, we note that:

$$p = d(\mathbf{O}, \mathbf{y}) = \frac{1}{\sqrt{c}}\cosh^{-1}(-c\langle\mathbf{O}, \mathbf{y}\rangle_{\mathcal{L}}) \tag{20}$$

$$q = d(\mathbf{O}, \mathbf{x}) = \frac{1}{\sqrt{c}}\cosh^{-1}(-c\langle\mathbf{O}, \mathbf{x}\rangle_{\mathcal{L}}) \tag{21}$$

and

$$o = d(\mathbf{x}, \mathbf{y}) = \frac{1}{\sqrt{c}}\cosh^{-1}(-c\langle\mathbf{x}, \mathbf{y}\rangle_{\mathcal{L}}) \tag{22}$$

For each of the inner products that feature the origin $\mathbf{O}$, the inner product has a very simple form in terms of the curvature and the time-like component. Concretely,

$$\langle\mathbf{O}, \mathbf{x}\rangle_{\mathcal{L}} = -\frac{x_t}{\sqrt{c}}$$

where as in the main paper x_t denotes the time-like component of $\mathbf{x}$. With this, denoting $h(t) = \cosh(t\sqrt{c})$, we get:

$$\begin{aligned} h(o) &= \cosh(o\sqrt{c}) \\ &= \cosh(\sqrt{c}\frac{1}{\sqrt{c}}\cosh^{-1}(-c \times -\frac{x_t}{\sqrt{c}})) \\ &= \cosh\cosh^{-1}(\sqrt{c}x_t) \\ &= \sqrt{c}x_t \end{aligned}$$

Similarly, $h(q) = \sqrt{c}y_t$. Next, we can write $h(p)$ as:

$$h(p) = -c\langle\mathbf{x}, \mathbf{y}\rangle_{\mathcal{L}} \tag{23}$$

With this, we can write Eqn. (24) as (noting that $\sinh(x) = \sqrt{\cosh^2(x) - 1}$):

$$\text{ext} = \cos^{-1}\left[\frac{h(p) - h(o)h(q)}{\sqrt{h(o)^2 - 1}\sqrt{h(q)^2 - 1}}\right] \tag{24}$$

We use the relationship between space-time components in Lorentz model (Eqn. (3)) and substitute everything into Eqn. (24) to get:

$$\text{ext}(\mathbf{x}, \mathbf{y}) = cos^{-1}\left(\frac{y_t + x_t c\langle\mathbf{x}, \mathbf{y}\rangle_{\mathcal{L}}}{\|\mathbf{x}_s\|\sqrt{\left(c\langle\mathbf{x}, \mathbf{y}\rangle_{\mathcal{L}}\right)^2 - 1}}\right) \tag{25}$$

D EXPERIMENTAL DESIGN

In this section, we give a detailed description of how we train our MERU models and CLIP baselines, along with the evaluation protocol for all models presented in Sec. 3.

D.1 TRAINING DETAILS

Baselines. We primarily compare with CLIP (Radford et al., 2021), that embeds images and text on a unit hypersphere in a Euclidean space. CLIP used a private training dataset of 400M image-text pairs. Several re-implementations of CLIP use publicly accessible datasets like YFCC (Thomee et al., 2016), Conceptual Captions (Sharma et al., 2018; Changpinyo et al., 2021), and LAION (Schuhmann et al., 2021; 2022); notable examples are OpenCLIP (Ilharco et al., 2021), SLIP (Mu et al., 2022), and FLIP (Li et al., 2022). We develop our CLIP baseline and train it using a single public dataset – RedCaps (Desai et al., 2021) – for simplicity, easy reproducibility, and reduced ethical risks (Birhane et al., 2021). Our smallest model trains using $8\times$ V100 (32GB) GPUs in *less than one day* and outperforms the recent re-implementatin of SLIP by a large margin. Appendix F walks through detailed development of this baseline.

Models. We use the Vision Transformer (Dosovitskiy et al., 2021) as image encoder, considering three models of varying capacity – ViT-S (Touvron et al., 2021; Chen et al., 2021), ViT-B, and ViT-L. All use a patch size of 16. The text encoder is same as CLIP – a 12-layer, 512 dimensions wide Transformer (Vaswani et al., 2017) language model. We use the same byte-pair encoding tokenizer (Sennrich et al., 2016) as CLIP, and truncate input text at maximum 77 tokens.

Initialization. We initialize the encoders exactly same as CLIP, except one change: we use a *fixed sine-cosine* position embedding in ViT, like (Chen et al., 2021; He et al., 2022). We initialize the softmax temperature as $\tau = 0.07$ and clamp it to a minimum value of 0.01. For MERU, we initialize the learnable projection scalars $\alpha_{img} = \alpha_{txt} = 1/\sqrt{512}$, the curvature as $c = 1.0$ and clamp it in interval $[0.1, 10.0]$ to prevent training instability. All scalars are learned in logarithmic space as $log(1/\tau)$, $log(c)$, and $log(\alpha)$.

Data augmentation. Similar to SLIP, we randomly crop 50–100% area of input images and resize them to 224×224. We randomly *prefix* the input captions with name of the associated subreddit using `wordsegment` Python library, similar to Desai et al. (2021).

Optimization. We use AdamW (Loshchilov & Hutter, 2019) with weight decay 0.2 and $\beta_2 = 0.98$. We disable weight decay for all gains, biases, and the scalars described above. All models train for $120K$ iterations (≈ 20 RedCaps epochs) with batch size 2048. The maximum learning rate is 5×10^{-4}, increased linearly for first $4K$ iterations, followed by cosine decay to zero (Loshchilov & Hutter, 2016). We use mixed precision training (Micikevicius et al., 2018) from PyTorch (Paszke et al., 2019) to accelerate training, however we compute $\exp_{\mathbf{o}}$ and hyperbolic losses in `float32` precision for numerical stability.

For MERU, we set the loss multiplier $\lambda = 0.2$ by running a small hyperparameter sweep, training ViT-B/16 models for one epoch. Quantitative performance is less sensitive to the choice of $\lambda \in [0.01, 0.3]$; however some non-zero value is required to induce partial order between text and images.

D.2 EVALUATION SETUP

We evaluate our trained models using a large set of downstream datasets covering a wide variety of visual concepts. All our evaluations primarily focus on zero-shot transfer, where we use the entire model for downstream task without any additional task-specific training.

Zero-shot image classification. We use 20 image classification datasets used by CLIP, SLIP, and other follow-up works. We select these datasets based on their ease of availability through open-source libraries like `torchvision` and Tensorflow datasets. We report top-1 mean per-class accuracy for all datasets, accounting for any label imbalance.

CLIP performs zero-shot classification by first creating text *prompts* using labels (*e.g.,* `a photo of a [label]`), followed by extracting their features to create the classifier weights. We use exactly same prompts for most datasets, except making a few changes to align with the linguistic style in training data (details in the appendix). When using multiple prompts per label, we ensemble them by

Table 3: List of all datasets used for zero-shot and linear probe evaluation. We evaluate zero-shot performance on the test splits. For linear-probe evaluations, we train classifiers using the train split and search hyperparameters using the validation split. Then we combine the training and validation splits and train a single classifier with the optimal hyperparameters, and report performance on the test split. Orange rows indicate datasets that do not have an official validation split; we constructed one by randomly holding out 10% of the official train split. EuroSAT and RESISC do not officially define any splits; we randomly sample instances to construct non-overlapping splits.

Dataset	Classes	Train	Val	Test	Metric
Food-101 (Bossard et al., 2014)	101	68175	7575	25250	accuracy
CIFAR-10 (Krizhevsky, 2009)	10	45000	5000	10000	accuracy
CIFAR-100 (Krizhevsky, 2009)	100	45000	5000	10000	accuracy
CUB-2011 (Wah et al., 2011)	200	5395	599	5794	accuracy
SUN397 (Xiao et al., 2010)	397	17865	1985	19849	accuracy
Stanford Cars (Krause et al., 2013)	196	7330	814	8041	accuracy
FGVC Aircraft (Maji et al., 2013)	100	3334	3333	3333	mean per-cls.
DTD (Cimpoi et al., 2014)	47	1880	1880	1880	accuracy
Oxford-IIIT Pets (Parkhi et al., 2012)	37	3312	368	3669	mean per-cls.
Caltech-101 (Fei-Fei et al., 2004)	102	2754	306	6084	mean per-cls.
Oxford Flowers (Nilsback & Zisserman, 2008)	102	1020	1020	6149	mean per-cls.
STL-10 (Coates et al., 2011)	10	4500	500	8000	accuracy
EuroSAT (Helber et al., 2019)	10	5000	5000	5000	accuracy
RESISC (Cheng et al., 2017)	45	3150	3150	25200	accuracy
MNIST (LeCun et al., 2010)	10	54000	6000	10000	accuracy
Patch Camelyon (Veeling et al., 2018)	2	262144	32768	32768	accuracy
CLEVR Counts (Johnson et al., 2017; Zhai et al., 2019)	8	1500	500	15000	accuracy
Country211 (Radford et al., 2021)	211	31650	10550	21100	accuracy
Rendered SST2 (Radford et al., 2021; Socher et al., 2013)	2	6920	872	1821	accuracy

averaging their embeddings prior to $\exp_{\mathbf{O}}$. These embeddings are from the encoder and lie in the tangent space (euclidean), hence vector averaging is a valid operation.

After applying exponential map, we compute the negative of geodesic distance, divide by softmax temperature, and apply a softmax classifier to obtain the final classification scores. In applications that do not require calibrated scores, using Lorentzian inner product as a similarity function is sufficient, since the geodesic distance is a monotonic function.

Cross-modal retrieval: We also perform retrieval evaluations using two image captioning datasets: COCO (Chen et al., 2015) and Flickr30K (Young et al., 2014). We report recall@5 for both, image retrieval and text retrieval. For COCO, we report results on `val2017` split, and for Flickr30K we use the Karpathy test split (Karpathy & Fei-Fei, 2015).

E Evaluation details

In Tab. 3, we list all the datasets we used for our zero-shot and linear probe transfer evaluations. We defined the training, validation, and test splits closely following the implementation details of CLIP (Radford et al., 2021). For zero-shot evaluation, we used text prompts for every dataset as listed in Tab. 4. We did not perform extensive prompt tuning, we simply checked the performance improvement on the held-out validation set using a baseline CLIP ViT-S/16 model trained using RedCaps (see Appendix F).

F Developing a strong CLIP baseline

One of our experimental contributions is to establish a lightweight, yet strong CLIP baseline. OpenAI's CLIP models are trained using a private dataset of 400M image-text pairs, across 128 GPUs for more than 10 days. We aim to maximize accessibility for future works – we decide our modeling

Table 4: We list all prompt templates used for zero-shot classification evaluation. Most of these prompts are same as (Radford et al., 2021). For some datasets, we observed the simple changes to prompts resulted in significant performance improvements; we highlight them as **(our prompts)**. These custom prompts are similar to the linguistic style of training captions ({`subreddit name`}: {`caption`}). **NOTE:** Some prompts use the word **'porn'** as it is included in the subreddit name. It does not indicate pornographic content but simply high-quality photographs.

Food-101 (our prompts):
– `food : {label}.`
– `food porn : {label}.`
CIFAR-10 and CIFAR-100:
– `a photo of a {label}.`
– `a blurry photo of a {label}.`
– `a black and white photo of a {label}.`
– `a low contrast photo of a {label}.`
– `a high contrast photo of a {label}.`
– `a bad photo of a {label}.`
– `a good photo of a {label}.`
– `a photo of a small {label}.`
– `a photo of a big {label}.`
– `a photo of the {label}.`
– `a blurry photo of the {label}.`
– `a black and white photo of the {label}.`
– `a low contrast photo of the {label}.`
– `a high contrast photo of the {label}.`
– `a bad photo of the {label}.`
– `a good photo of the {label}.`
– `a photo of the small {label}.`
– `a photo of the big {label}.`
CUB-2011 (our prompts):
– `bird pics : {label}.`
– `birding : {label}.`
– `birds : {label}.`
– `bird photography : {label}.`
Oxford-IIIT Pets:
– `a photo of a {label}, a type of pet.`
Caltech-101:
– `a photo of a {label}.`
– `a painting of a {label}.`
– `a plastic {label}.`
– `a sculpture of a {label}.`
– `a sketch of a {label}.`
– `a tattoo of a {label}.`
– `a toy {label}.`
– `a rendition of a {label}.`
– `a embroidered {label}.`
– `a cartoon {label}.`
– `a {label} in a video game.`
– `a plushie {label}.`
– `a origami {label}.`
– `art of a {label}.`
– `graffiti of a {label}.`
– `a drawing of a {label}.`
– `a doodle of a {label}.`
– `a photo of the {label}.`
– `a painting of the {label}.`
– `the plastic {label}.`
– `a sculpture of the {label}.`
– `a sketch of the {label}.`
– `a tattoo of the {label}.`
– `the toy {label}.`
– `a rendition of the {label}.`
– `the embroidered {label}.`
– `the cartoon {label}.`
– `the {label} in a video game.`
– `the plushie {label}.`
– `the origami {label}.`
– `art of the {label}.`
– `graffiti of the {label}.`
– `a drawing of the {label}.`
– `a doodle of the {label}.`

SUN397:
– `a photo of a {label}.`
– `a photo of the {label}.`
Stanford Cars (our prompts):
– `car porn : {label}.`
– `classic cars : {label}.`
FGVC Aircraft:
– `a photo of a {label}, a type of aircraft.`
– `a photo of the {label}, a type of aircraft.`
Describable Textures (DTD, our prompts):
– `mildly interesting : a {label} texture.`
– `mildly interesting : a {label} pattern.`
– `mildly interesting : a {label} thing.`
– `mildly interesting : a {label} object.`
Oxford Flowers:
– `flowers : {label}.`
STL10:
– `a photo of a {label}.`
– `a photo of the {label}.`
EuroSAT:
– `a centered satellite photo of {label}.`
– `a centered satellite photo of a {label}.`
– `a centered satellite photo of the {label}.`
RESISC:
– `satellite imagery of {label}.`
– `aerial imagery of {label}.`
– `satellite photo of {label}.`
– `aerial photo of {label}.`
– `satellite view of {label}.`
– `aerial view of {label}.`
– `satellite imagery of a {label}.`
– `aerial imagery of a {label}.`
– `satellite photo of a {label}.`
– `aerial photo of a {label}.`
– `satellite view of a {label}.`
– `aerial view of a {label}.`
– `satellite imagery of the {label}.`
– `aerial imagery of the {label}.`
– `satellite photo of the {label}.`
– `aerial photo of the {label}.`
– `satellite view of the {label}.`
– `aerial view of the {label}.`
MNIST:
– `a photo of the number: "{label}".`
Patch Camelyon:
– `this is a photo of {label}.`
CLEVR:
– `a photo of {label} objects.`
Country211:
– `a photo i took in {label}.`
– `a photo i took while visiting {label}.`
– `a photo from my home country of {label}.`
– `a photo from my visit to {label}.`
– `a photo showing the country of {label}.`
Rendered SST2:
– `a {label} review of a movie.`
ImageNet (our prompts):
– `i took a picture : itap of a {label}.`
– `pics : a bad photo of the {label}.`
– `pics : a origami {label}.`
– `pics : a photo of the large {label}.`
– `pics : a {label} in a video game.`
– `pics : art of the {label}.`
– `pics : a photo of the small {label}.`

Table 5: **Developing a strong CLIP baseline.** We start with CLIP baseline by SLIP (Mu et al., 2022) as a reference and modify it to facilitate fast training on a single 8-GPU machine while maintaining performance. We benchmark improvements by observing zero-shot classification performance on 20 datasets. All models in this table use ViT-S/16 as the image encoder. Our RedCaps-trained CLIP baseline (last row) is a significantly stronger baseline than its YFCC-trained counterparts.

	Images Seen	Food-101	CIFAR-10	CIFAR-100	CUB	SUN397	Cars	Aircraft	DTD	Pets	Caltech-101	Flowers	STL-10	EuroSAT	RESISC45	MNIST	PCAM	CLEVR	Country211	SST2	ImageNet	Average
YFCC15M-trained models																						
SLIP's CLIP	368M	43.4	61.0	29.9	31.1	43.9	3.1	4.7	17.9	25.0	53.3	47.8	86.8	22.3	16.1	9.8	64.8	14.7	8.7	49.5	32.7	32.5
Our implementation	368M	42.3	64.9	34.4	33.7	43.8	2.9	5.1	19.1	25.0	49.8	47.2	87.4	26.8	21.6	9.7	54.2	12.6	9.0	49.9	33.1	33.6
+ BS 4096→2048	184M	34.2	58.7	29.4	27.4	39.4	2.9	4.3	16.5	20.1	43.8	42.2	85.4	20.2	19.0	10.3	50.9	14.0	8.5	50.1	28.2	30.3
+ *sin-cos pos embed*	184M	34.2	67.3	33.6	25.4	41.1	3.1	4.2	17.8	21.0	44.3	43.6	86.4	18.6	19.6	10.5	50.1	9.0	8.3	49.3	28.7	30.8
RedCaps-trained models																						
+ YFCC→RedCaps	184M	71.5	61.4	25.6	29.9	27.5	10.1	1.5	14.3	72.7	62.8	42.2	88.0	23.4	30.5	10.1	50.3	17.4	4.9	50.1	32.6	36.3
+ 90K→120K iters.	246M	72.5	60.1	24.4	30.4	27.5	11.3	1.4	13.1	73.7	63.9	44.4	88.2	18.4	31.3	10.3	50.2	17.8	5.2	50.1	33.9	36.4
+ our prompts	246M	74.5	60.1	24.4	34.2	27.5	11.0	1.4	14.5	73.7	63.9	47.0	88.2	18.4	31.3	10.3	50.2	17.8	5.2	50.1	34.3	36.9

and optimization hyperparameters such that the smallest model (ViT-S/16) can be trained on a single 8-GPU machine in less than one day.

We start with a reference CLIP baseline from the recent work of SLIP. We experiment with the ViT-S/16 model, and carefully introduce one modification at a time and observe its impact on the baseline performance. All model variants are evaluated for zero-shot image classification on the 20 datasets used in the main paper; results are in Tab. 5.

CLIP baseline by SLIP. SLIP (Mu et al., 2022) developed a CLIP baseline that is trained using the publicly available YFCC15M subset released by (Radford et al., 2021). This baseline underperforms OpenAI's models as it uses an order of magnitude less training data, nevertheless facilitates fair comparison. We re-evaluate the publicly released ViT-S/16 checkpoint [2] using our evaluation code. This checkpoint obtains 32.5% average accuracy across all datasets.

Our re-implementation. We attempt a faithful replication of the CLIP baseline by SLIP. Our implementation obtains slightly higher performance (33.6%), and has three minor differences with SLIP's CLIP implementation.

- As per common practice, we gather image and text features across all GPUs to increase negatives for the contrastive loss. This gather operation is not recorded in the computation graph by SLIP, as per default implementation of PyTorch. We use an undetached gather operation to ensure proper gradient flow across devices.
- With the above change, we could use weight decay = $= 0.2$ like OpenAI's CLIP, unlike 0.5 used by SLIP's CLIP.
- During training and inference, we resize input images using bicubic interpolation. SLIP's CLIP resizes via bilinear interpolation. Bicubic interpolation helps on benchmarks with low-resolution images (e.g. CIFAR, STL).

Fitting the model to 8-GPUs. The baseline model described above requires $16\times$ V100 32GB GPUs with a standard batch size of 4096 and using automatic mixed precision. Techniques like gradient checkpointing can reduce memory requirements, but it comes at a cost of reduced training speed. Hence we avoid making it a requirement and simply reduce the batch size to 2048. This incurs a performance drop as the effective images seen by the model are halved. We use fixed *sine-cosine* position embeddings, so the model is not required to learn position-related inductive biases during training. This change slightly improves average accuracy ($30.3\% \rightarrow 30.8\%$ average accuracy).

Using RedCaps dataset for training. Finally, we switch the YFCC15M subset with the recent RedCaps dataset. RedCaps is a comparably sized dataset of 12M image-text pairs from Reddit. It offers higher quality of text, which significantly improves the baseline over previous YFCC15M trained variants ($30.8\% \rightarrow 36.3\%$ average accuracy). There are notable gains on datasets like Food-

[2] https://github.com/facebookresearch/slip

101, Pets, and Cars – these concepts have high coverage in RedCaps. To account for smaller size of RedCaps, we increase the training iterations from 90K up to 120K.

Finally, we adjust test-time prompts for a few evaluation datasets to match the linguistic style of RedCaps. For example, we use `food : {label}` for Food-101 as many captions on `r/food` simply mention the name of the dish in the corresponding image. We did not extensively tune these prompts, but we checked performance on the held-out validation sets to avoid cheating on the test splits.

With all these modifications, our CLIP ViT-S/16 baseline achieves 36.9% average zero-shot classification accuracy across 20 datasets, being trained on 8× V100 32 GB GPUs within ≈14 hours. We use these details exactly for larger image encoders and use them as strong baselines in our experimental comparisons.

G Additional results

Ablations. We next investigate all the important design choices in our construction of MERU embeddings (**??**). For MERU ViT-L/16 models, we find that the entailment loss is very important for achieving good performance on ZSL benchmarks (41.0 *vs* 40.7% on average, and 38.8 *vs* 33.9 % on ImageNet). Thus, the entailment loss not only adds structure but also improves performance [3]. Next, we fix the curvature to $c = 1$ instead of learning it during training. As far as we are aware, no prior work has learnt the curvature end to end (Atigh et al., 2022; Khrulkov et al., 2020; Nickel & Kiela, 2018). We find this design choice to be crucial for scaling, and found $c = 1$ models do not achieve good performance on convergence for ViT-L/16 models (**??** middle row). Finally, we experiment with using the Lorentz inner product Eqn. (2) directly instead of the distance on the manifold Eqn. (4). This inner product is numerically large and grows faster than hyperbolic distance (without the logarithmic form of $cosh^{-1}$) – we find that the training diverges due to numerical instability for ViT-L/16 models. Overall, we notice that these design choices are more crucial for the large ViT-L/16 models compared to the smaller ViT-B/16 models (**??**). We hope these ablations serve as guidelines for work in other domains that study hyperbolic geometry for deep representation learning.

Resource constrained deployment. We hypothesize that one of the advantages of imposing more structure that is naturally present the data (such as the latent visual-semantic hierarchy) yields embeddings that make more efficient use of the volume in the ambient embedding space. This is useful in various on-device deployment settings, where one might have runtime or memory constraints (Kusupati et al., 2022) necessitating low-dimensional embeddings. To verify this hypothesis, we sweep across the embedding dimension from 8 to 512 and train ViT-B/16 models that output the corresponding embedding dimension for image and text respectively. We then evaluate the resulting models on ImageNet (Russakovsky et al., 2014) for zero-shot classification accuracy. Our results (Tab. 7) show that MERU substantially outperforms CLIP at low embedding dimensions (64D to 8D) for zero-shot classification on ImageNet. This indicates that hyperbolic embeddings might be useful to use in resource constrained, on-device settings.

Table 7: **CLIP and MERU of different embedding dimensions.** We report zero-shot ImageNet accuracy; MERU consistently outperforms CLIP at very low embedding widths. **bold** shows best performance in each column.

Embed width ⇒	512	128	64	32	16	8
CLIP ViT-B/16	**37.9**	**37.4**	35.0	30.2	21.1	11.6
MERU ViT-/B16	37.5	**37.4**	**35.7**	**31.0**	**23.8**	**15.2**

[3] Note that this ablation is mathematically impossible for CLIP-style models as there is no obvious notion of entailment that can be defined when all the embeddings are normalized to unit norm.

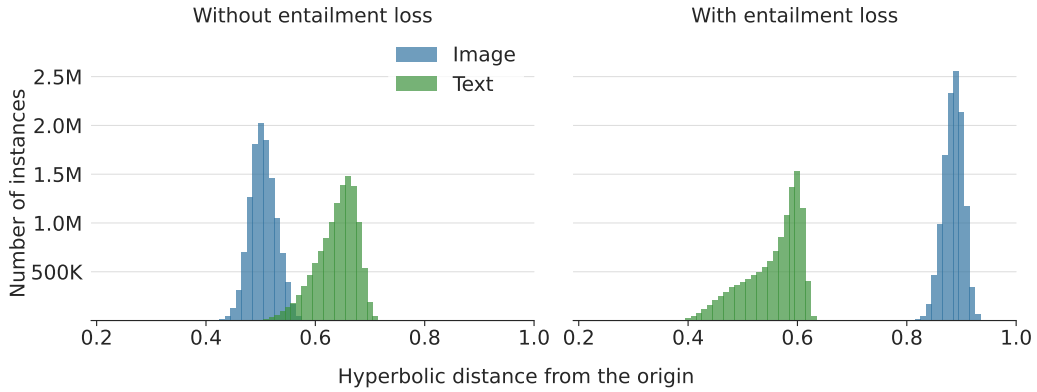

Figure 4: We extract hyperbolic embeddings of 12M image-text pairs from RedCaps using MERU ViT-L/16 models. **Left:** We plot the distribution of hyperbolic distances from origin of a model trained only with the contrastive loss (Appendix B.1). Notice that the distributions of the text and image embeddings overlap with each other. **Right:** We demonstrate the effect of using the entailment loss that enforces text $\rightarrow$ image for each datapoint. We see that using the entailment loss pulls text embeddings closer to the origin and pushes image embeddings further from the origin (and also leads to improved performance (Tab. 2)).

H More qualitative analysis

Effect of the entailment loss. We plot the effect of imposing the entailment loss on the distributions of distance between image-text representations and origin of the hyperboloid (Fig. 4). Recall that the entailment loss enforces a partial order between modalities as *'text entails image'*. The distributions overlap without this loss, but upon enforcing it, they become well separated – the text representations are placed closer to the origin. This suggests that the entailment loss is critical for imbuing the model with the inductive bias of the visual-semantic hierarchy (Fig. 1) and subsequent performance improvements (Sec. 3). We hypothesize that this effect will be less prominent as we increase the data scale, and that the inductive bias of the entailment loss will be required to a smaller extent (meaning that left side of Fig. 4 would look more similar to the right side).

More interpolations. In Fig. 5, we show more qualitative examples of the interpolations from images, going up the visual-semantic hierarchy, similar to Fig. 3b. Notice how images and text are found together in a common semantic hierarchy, with implicit, learnt structure between the different images and captions.

Which concepts are general for MERU? We also investigate the emergent hierarchy in the representation space by computing distance of various embeddings to the origin. Notice in Tab. 8 how generic captions (*'itap of nothing'*) are closer to origin than more detailed ones. Also notice how specific concepts such as a geographic location often end up further from the origin. Interestingly, this structure is being learnt purely through the grounding into image data, since the entailment loss only enforces text $\rightarrow$ image for individual pairs (and makes no assumptions about the relative generality of the concepts expressed within text).

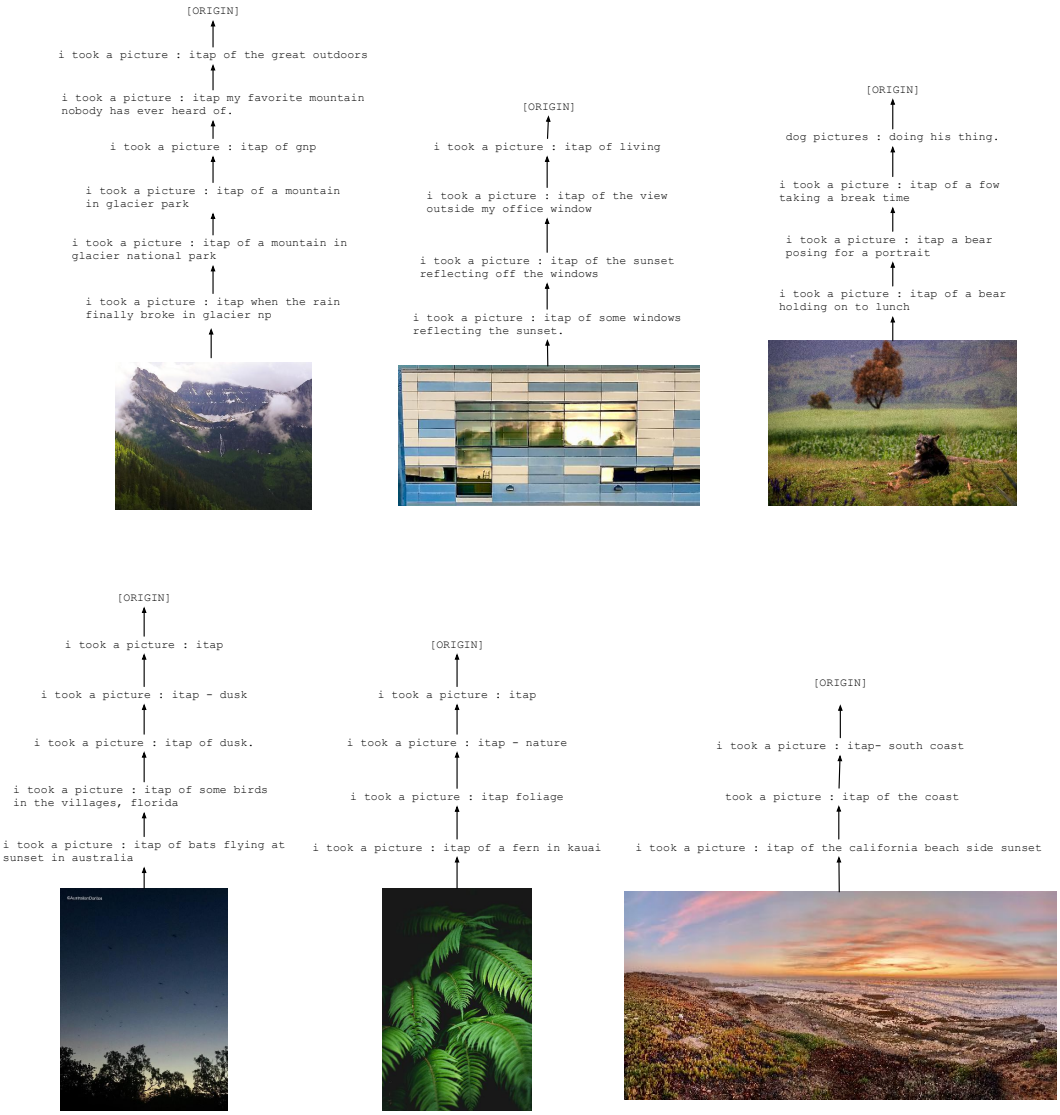

Figure 5: **Multimodal interpolations for MERU embeddings.** We project the query image embedding to the tangent space of the origin, take 50 linearly spaced steps, and lift all of them onto the hyperboloid. For each vector, we find the nearest neighbor text embedding using the Lorentz distance. We list all the (unique) texts encountered between the image and the `[ORIGIN]` (arrow shows the direction of interpolation, not entailment). `itap` means "I took a picture".

Table 8: **Generic and specific concepts. Top:** Captions from `r/itookapicture` whose representations are nearest to the origin, tend to be vague and generic. **Bottom:** Captions whose representations are furthest from the origin, tend to be long and descriptive, and often mention specific concepts like locations.

Nearest distance from the origin ($\approx$ generic):
– itap of nothing
– itap of where i live.
– itap of the place to be
– itap of another picture
– itap of the sea break
– itap of a place i found.
– itap of home.
– itap of somewhere i've never been before
– itap of another place
– itap of something random.
– itap of lovely scenery
Furthest distance from the origin ($\approx$ specific):
– itap of a zanate mexicano/clarinero
– itap of me navigating barges through a canal
– itap of a pink saline at formentera island
– itap of red stained double cream
– itap of a cliche ball at sunset
– itap of my red jasper.
– itap of a feather in a dark forest
– itap of my friend silhouetted, with desaturated red filters.
– itap of an orangey, purpley, red-ish sky.
– itap of gargoyles in a fountain
– itap of the silversmith's workbench in colonial williamsburg, va