Appendix

A Limitations and Future Work

In this paper we introduced three new datasets BLURD 3D, BLURD SD and BLURD Mask as well as a method for synthetically creating photo-realistic images by combining 3D renders and generative methods, called BLURD. In this work we only explored a limited scope of how BLURD can be used for representation learning and recognise that there is a near infinite list of other uses of BLURD we could have presented in this paper. Nevertheless here we discuss some of the promising future directions and extensions to BLURD as well as the limitations of our work.

A.1 Using the BLURD datasets to assess model biases

As discussed in section 4 the BLURD datasets can be a powerful tool for discovering biases in either pre-trained models or training datasets. An example discussed in section 4 was race, where we showed that some models performed poorly in the task of zero-shot accuracy on certain Hispanic races. While such evaluations can provide valuable insight into possible model biases or underrepresented attributes in training datasets, it is important to highlight their limitations. For one, differences in zero-shot accuracy could be explained by inherent biases in the BLURD datasets themselves, or a mismatch between the photo-realism of BLURD and real-life photography. In particular any data-driven generative approach can model a biased distribution of images, either introduced from the training data of the generative model or by some aspect of the training approach or model architecture. We therefore caution the reader on relying on the BLURD datasets and the evaluations presented in this paper as a categorical *proof* that a model or a training dataset is flawed. Similarly, BLURD cannot be used in isolation to categorically rule out that a model has no biases and therefore is safe to use in any application. However, investigating how BLURD can be extended or combined with other approaches to create more robust measures of model bias and limitations is a promising area of future research.

A.2 Evaluating representation spaces with BLURD

In section 4 we presented zero-shot accuracy, equivariance and the comparison between zero-shot accuracy and equivariance as methods to gain insight into the representation spaces of pre-trained models. By applying these evaluation methods, we provided a small set of example use cases for how the BLURD datasets can be employed. Nevertheless, it should be emphasized that these examples are not exhaustive in their representation of the full potential of the BLURD datasets, we leave other methods to future work. Furthermore, no evaluation method presented in this paper provides perfect insight into the structure of the representation space.

Here we note several limitations of the particular methods we selected. Zero-shot accuracy was measured via cosine distance to a caption representing a single factor or factors. As the contrastive loss objective of CLIP only considers vectors on the unit hypersphere and their angle separation, cosine distance was a natural choice [44]. However, two vectors can be close in angle yet far apart in Euclidean distance, causing zero-shot accuracy to be impacted by the choice of similarity measure. On the other hand, equivariance measures the parallelism between two vectors by cosine similarity. While providing insight into how the representation space is preserved under a factor change, Figure 6 provides some examples as to how wildly different scenarios can result in the same equivariance score. Additionally, parallelism is a property that is ill defined on a hypersphere, making it an analysis that assumes a zero curvature geometry. This assumption is in conflict with the assumption that the representations of CLIP models lie on the hypersphere [44]. Nonetheless, we can assume that on a small enough patch of the hypersphere, Euclidean geometry becomes a good approximation to that region of space. Clearly, that assumption fails if the angle between the two vectors under consideration is too large. In the case of BLURD, as we restrict ourselves to only human faces with a handful of factors of variation it is reasonable to assume all vectors are in relative proximity to each other. However, we leave to future work a thorough exploration of how making Euclidean assumptions when analyzing CLIP representations impacts the results on BLURD. Comparisons between zero-shot accuracy and equivariance also requires that Euclidean space is a reasonable approximation in the region of the hypersphere under consideration. Addtionally, in Figure 6 we provide examples of how equivariance and separability can differ. In our analysis we used zero-shot accuracy a proxy for separability, however we note that zero-shot accuracy is a poor measure of separability and is only loosely related. A better measure would be to find if there exists a hyperplane that separates the respective classes, while a closed form solution exists to finding such a hyperplane, this wasn't the focus of this study and we leave this analysis to future work.

A.3 Generative models for creating datasets for representation learning

One key criteria of a good dataset for representation learning is the minimization of additional confounding factors outside of the factors of variation of interest. Typically 3D renders achieve highly controllable environments due to their deterministic rendering pipeline, therefore for any given set of factors any two images generated with those factors will be identical. In contrast, generative methods often introduce too many confounding factors or mix factors when used to create datasets for representation learning [41, 6, 37, 13, 47, 49, 30, 15, 26, 24, 60, 43]. BLURD seeks to address this by applying strong conditions on the generative model, constraining the generated image to faithfully represent the desired factors. While by visual inspection BLURD achieves this, it is difficult to categorically rule out imperceptible but undesirable influences of one factor on another. One future extension to our work made possible by the masks we produce with each render (see Figure 10) is to use regional prompting (separately prompting each masked region with its associated factor). Nevertheless, we note that the slight intra-factor variation gained by using generative models, which are stochastic in nature is beneficial for representation learning. As the point estimate a deterministic 3D render provides can be of limited analytical value. In contrast a sample of multiple vectors corresponding to the same fixed set of factors allows for greater statistical analysis.

B BLURD Datasets

B.1 BLURD: The Making Of

In this section we describe major tools we used to create the dataset, as well as the major design choices and considerations.

3D Render Engine: Blender is a popular choice with researchers for creating synthetic datasets due to it being freely available, open source license and possessing an extensive API [9, 28, 23, 65]. Blender comes with two rendering engines, a real-time render engine Eevee, and an industry standard path-tracing render engine Cycles [4]. Real-time render engines, typically employed in game engines, are frequently used for synthetic datasets due to their quick render time and advanced graphics [6]. however due to the various heuristics and approximations involved in rendering in real-time, they introduce systematic error into the final image, making them ill suited for a photo-realism dataset. Cycles, on the other hand is an unbiased, physically based, path tracer [4]. As well as having superior realism, Cycles also has the added advantage of more accurate and flexible render passes, which we discuss our use of in Section 3.

In addition to Blender we take advantage of the CC0 licensed high dynamic range images (HDRI) from Poly Haven to create realistically lit scenes and backgrounds [58]. Furthermore, we use the Human Generator assets and Blender plugin, because they provides high quality human models with accurate skin materials and hair particles, allows for fine control of several factors of variation, the source code is licensed under the GNU General Public License and the plugin has a well documented python API [42]. However, we stress that while the BLURD method requires careful experimentation with the SD hyperparameters, our method does not rely on any particular assets per se and will work with other 3D environments of sufficient quality. Therefore we release the *blend* file and source code to create the BLURD dataset in the hopes other researchers extend the dataset with their own assets and 3D scenes.

Diffusion Model: Image diffusion models are a class of models that are designed to learn the process of progressively denoising images and generating samples within the training domain. The Stable Diffusion (SD) family of models has seen an explosion of research interest since its inception due to it being one of the few high quality freely available open source text-to-image diffusion models [13, 67, 47, 37, 41]. SD has multiple versions, including 1.0, 1.4, 1.5, 2.0, 2.1, 3.0 and XL. Among these, version 1.5 (V1.5) stands out due to its level of community engagement in creating high quality custom fine-tunes and ControlNets. Fine-tunes are important as they allow for greater control in the style and quality of imagery produced by SD [37, 49]. In particular, we were interested in fine-tunes

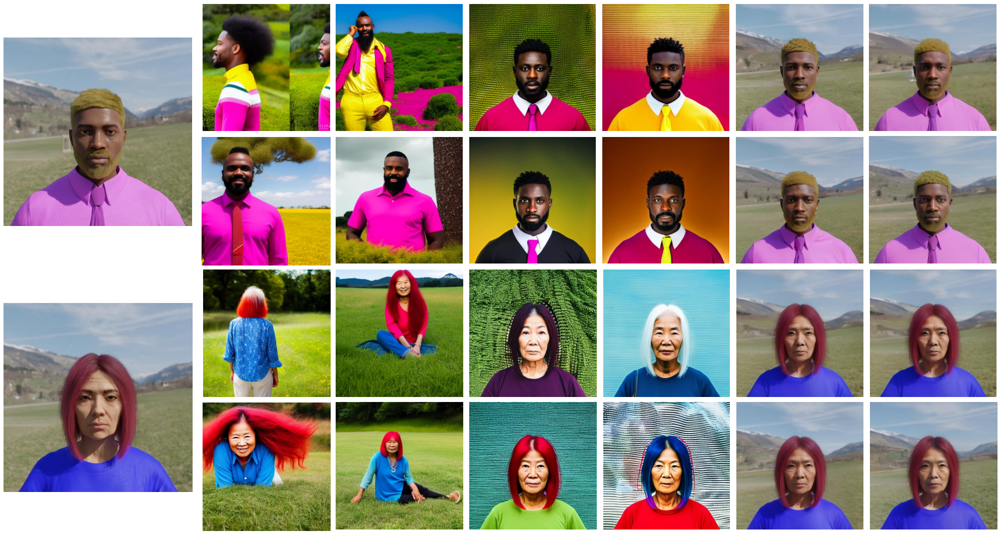

Figure 9: An example of creating **BLURD** dataset using the original SD 1.5 model instead of the Realistic Vision fine-tune. *Left:* the two images on the far left of the figure are from **BLURD 3D** as per normal. *Middle left:* The eight images are examples of **BLURD SD: Unleashed** generated by the original SD 1.5 model conditioned only on text generated from the same factors as the corresponding 3D render. *Middle right:* The next The eight images are generated identically to **BLURD SD: Unleashed** except with SD 1.5. *Right:* the last eight images are generated identically to **BLURD SD: Harbinger** except with SD 1.5. The difference in realism and quality is stark when compared to Figure 2

of SD that exhibited a high degree of realism in their outputs. After, testing various community made fine-tunes available on HuggingFace [63] we settled on Realistic Vision V5.1 due to its focus on photo realism and ability to generate high quality consistent imagery [19]. We note that at the sacrifice of quality, our method can be used on the base model of SD V1.5 and provide some examples in Figure 9. We leave evaluating other generative models such as SDXL, as replacement to SD V1.5 in BLURD to future work.

Conditioning: ControlNet is a method of adding conditional controls to image diffusion models [67]. ControlNet achieves this by learning to inject additional conditions into the network blocks of the Stable Diffusion Unet decoder during training[67]. In theory, a ControlNet model can learn to inject any type of condition into Stable Diffusion, however ControlNet typically adds conditional control via an input image[67]. ControlNets have been trained to inject various conditions, including canny edge, depth map, normal map, M-LSD lines, HED soft edge, ADE20K segmentation, openpose, user sketches and spatial palettes [67, 39, 66]. However, in this work we are interested in conditional images that can be efficiently generated with a 3D rendering engine, hence we use three ControlNets exclusively in this work, namely depth map, normal map and spatial palettes [67, 39, 66]. Additionally, SD is also capable of conditioning on a input image by using a partially noised image as the starting point for the diffusion process. Using SD in this way is termed image-to-image (img2img) and by carefully calibrating the level of added noise, SD can effectively be used for style transfer [13, 67, 47, 37, 41]. This is a property of SD that we exploit in our method.

B.2 BLURD Mask

Here we describe in more detail how we constructed BLURD Mask and modified the CelebAMask-HQ dataset to create the semantic segmentation test set. BLURD Mask follows a similar pipeline to BLURD 3D/SD for the creation of the 3D renders, and their photo-realistic counterparts. However, to achieve greater variation we add to the number of factors. We add pitch, elevation and focal length to the camera to allow for three degrees of freedom of camera rotation and zoom around the subject. Additionally we add environment background rotation, environmental lighting strength and environmental lighting warmth. Instead of producing a image for every possible factor combination, we simply uniformly sample a random set of factors and render a corresponding 3D render with the

Table 3: **ControlNet Settings:** The settings for each of the ControlNet models, where a ControlNet was used more than once, the same setting where used.

	ControlNet Model	Weight	Guidance End	Processor Resolution
Depth	v11f1p_sd15_depth	0.8	0.8	N/A
Normal	v11p_sd15_normalbae	0.8	0.8	N/A
Color	t2iadapter_color_sd14v1	1	1	2048

normal map, depth mask and masks. We then use the BLURD SD: Harbinger process described in section 3 to obtain the photo-realistic version, however we increase the resolution for better quality imagery and only produce one image per 3D render. In total we sample and render 17k new images for BLURD Mask. The masks created by BLURD during rendering are shown in Figure 10. As can be seen the masks from BLURD are dissimilar to the segmentation mask in CelebAMask-HQ as shown in Figure 11. To address this discrepancy we reduce the BLURD mask shown in Figure 10 to six categories; *skin*, *eyebrows*, *hair*, *eyes*, *lips* and *cloth*. The *skin* mask is created by merging the face, eyelash and beard masks from Figure 10, the beard mask undergoes several iterations of erosion before merging and then the final merged mask is dilated and holes are removed. The *eyes*, *eyebrows* and *hair* masks are created from the corresponding mask from Figure 10 with closing and hole removal operations applied. The *lips* mask is taken directly from Figure 10 without modification. Finally, *cloth* is the merging of the shirt and tie masks from Figure 10. Similarly, The original dataset CelebAMask-HQ includes a total of 18 different categories: Skin, Neck, Hat, Eyeglasses, Necklace, Hair, Earrings, Cloth, Eye (left), Eye (right), Brow (left), Brow (right), Nose, Ear (left), Ear (right), Mouth, Lip (upper), Lip (lower). To reduce these categories to the more manageable six we first eliminated masks for objects like eyeglasses, earrings, and hats. Then, we merged masks for left/right or upper/inner/lower anatomy of specific features such as eyes, ears, lips and mouth. Finally, we merged masks for specific features like the nose, ears, and neck into skin. Figure 7 contains the final result for both datasets. While the labels for the two datasets are not identical after this process, they are sufficiently close to evaluate the zero-shot domain adaption task.

We release BLURD Mask as a separate dataset containing the 3D rendered images, their synthetically created photo-realistic counter parts and the corresponding segmentation labels. Even though BLURD Mask shares a similar creation process to BLURD 3D/SD we separate it as a distinct dataset for the following reasons: The images in BLURD Mask are created at a higher resolution than BLURD 3D/SD; we sample from the possible factors rather than iterate over all factors; the factors pitch, elevation, focal length, background rotation, environmental lighting strength and environmental lighting warmth are continuously valued rather than discrete, with the latter six factors only introduced in BLURD Mask; and the addition of segmentation labels compatible with the segmentation masks of CelebAMask-HQ after merging.

B.3 Licence and Where to Learn More

The datasets BLURD 3D, BLURD SD and BLURD Mask are available under the CC BY-NC 4.0 license. Source code used to create all BLURD datasets can be found at `https://github.com/squaringTheCircle/BLURD`. The BLURD datasets are available for download at `https://www.blurd.xyz/`.

B.4 Additional Settings

The following section contains information on the settings used to create BLURD. Full details are available with the released source code at `https://github.com/squaringTheCircle/BLURD`. Figure 12 contains a visual representation of the effect of different noise levels and different SD samplers. Figure 13 and Figure 14 provide details on the construction of the shader and compositor node structure in Blender used to create the depth map, normal map and masks. For an example of the depth map, normal map and spatial palette created with each render refer to Figure 15. For an example of the masks created with each render see Figure 10.

Table 4: **BLURD Stable Diffusion Settings:** The Stable Diffusion settings for each ablation of the BLURD SD datasets. Only BLURD SD: Harbinger used Img2Img mode hence required a denoising parameter.

	Sampler	Batch Size	Steps	Cfg Scale	Denoising Strength
BLURD SD: Unleashed	Euler a	32	20	7	N/A
BLURD SD: Tempered	Euler a	8	20	7	N/A
BLURD SD: Harbinger	Euler a	8	20	7	0.20

Table 5: **BLURD Dataset Factors of Variation and Their Values:**. In total there are 9 factors of variation with $73,000$ possible combinations, not all factors are compatible with both genders, namely hair and beard styles are gender specific.

Factor	**Values**
Gender	female, male
Background	grassy meadow, indoor livingroom
Age	30, 70
Race	african, asian, caucasian, hispanic
Hair Style	afro dreads, bald top, bob long, buzzcut, none, short combed, undercut
Hair Color	black, blue, cyan, green, magenta, orange, red, violet, white, yellow
Beard Style	full, none
Clothes Color	black, blue, cyan, green, magenta, orange, red, violet, white, yellow
Angle	0, 5, 10, 15, 20

B.5 Datasheet

MOTIVATION	
For what purpose was the dataset created?	The BLURD 3D/SD datasets were created to evaluate and benchmark pre-trained vision and language models. They are intended for representation learning and representation space understanding. In particular, the focus on human subjects allows for the analysis of model biases or underrepresented human characteristics in pre-training datasets. Additionally the ablation of BLURD SD were created to progress the understanding of why SD has common failure modes and how they relate to the representation space of CLIP models. BLURD 3D/SD were also created to provide a dataset for representational learning that most closely mimics real world photography. BLURD 3D uses state of the art assets and a path-tracing render engine to achieve photo-realism and BLURD SD uses generative methods as well as 3D renders to create photo-realism that is visually indistinguishable from a real photo. Therefore, BLURD 3D/SD is intended for the study of the photo-realism gap that still exists between 3D render engines and real world photography. Finally, BLURD Mask is intended for benchmarking how generative methods can mimic the distribution of real world photography using semantic segmentation as a domain adaption task.
Who created the dataset and on behalf of which entity?	The BLURD datasets were created by the authors.
Who funded the creation of the dataset?	The University of Adelaide and the Australian Government Research Training Program Scholarship.
COMPOSITION	
What do the instances that comprise the dataset represent?	The instances comprise of images of human faces with associated factors or segmentation masks.
How many instances are there in total?	1.7 million images.
Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set?	We sample a subset of the possible factors, see section 3 for details.
What data does each instance consist of?	Image with associated factors or segmentation masks.
Is there a label or target associated with each instance?	Yes, the associated factors or segmentation masks.
Is any information missing from individual instances?	No.
Are relationships between individual instances made explicit?	Yes, individual instances are separated by factor changes.
Are there recommended data splits?	In BLURD Mask the BLURD 3D/SD generated images are for training, the test set is created from CelebAMask-HQ. See Appendix B for further details
Are there any errors, sources of noise, or redundancies in the dataset?	BLURD SD is generated with a partially stochastic process which generates noise and also contains several images per factor permutation.

Is the dataset self-contained, or does it link to or otherwise rely on external resources?	BLURD Mask requires CelebAMask-HQ. The BLURD 3D/SD dataset we distribute are self contained however external assets are required to generate addtional images. Refer to `https://www.blurd.xyz/` and `https://github.com/squaringTheCircle/BLURD` for additional details.
Does the dataset contain data that might be considered confidential?	No.
Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety?	No.

COLLECTION

How was the data associated with each instance acquired?	Each image in the BLURD 3D/SD is associated with the factors that generated the image as well as a depth map, normal map, pixel map and several masks. Each image in BLURD Mask is associated with segmentation masks.
What mechanisms or procedures were used to collect the data?	See section 3 and Appendix B for details on how the BLURD datasets were constructed. The assets used in this paper were manually selected or part of the Human Generator plugin.
If the dataset is a sample from a larger set, what was the sampling strategy?	A subset of the possible factors were manually chosen based on their contribution to the image. For example *shirt color* was chosen as one of the factors in the BLURD dataset as the shirt of the person is a large feature of the image, while *eye color* was not chosen and it comprises a much smaller portion of the image.
Who was involved in the data collection process and how were they compensated?	The authors of this paper.
Over what timeframe was the data collected?	The data were collected between 2023 and 2024
Were any ethical review processes conducted?	No.
Did you collect the data from the individuals in question directly, or obtain it via third parties or other sources (e.g., websites)?	Third parties, Blender Market Place and CelebAMask-HQ
Were the individuals in question notified about the data collection? If so, please describe (or show with screenshots or other information) how notice was provided, and provide a link or other access point to, or otherwise reproduce, the exact language of the notification itself.	N/A. No individuals outside the authors were involved
Did the individuals in question consent to the collection and use of their data? If so, please describe (or show with screenshots or other information) how consent was requested and provided, and provide a link or other access point to, or otherwise reproduce, the exact language to which the individuals consented.	N/A.

If consent was obtained, were the consenting individuals provided with a mechanism to revoke their consent in the future or for certain uses? If so, please provide a description, as well as a link or other access point to the mechanism (if appropriate).	N/A.
Has an analysis of the potential impact of the dataset and its use on data subjects (e.g., a data protection impact analysis) been conducted? If so, please provide a description of this analysis, including the outcomes, as well as a link or other access point to any supporting documentation.	N/A.
PREPROCESSING	
Was any preprocessing/cleaning/labeling of the data done?	N/A.
Was the "raw" data saved in addition to the preprocessed/cleaned/labeled data?	N/A.
Is the software that was used to preprocess/clean/label the data available?	N/A.
USES	
Has the dataset been used for any tasks already?	Yes, this paper presented a number of experiments and tasks.
Is there a repository that links to any or all papers or systems that use the dataset?	Yes, we will maintain a list at `https://github.com/squaringTheCircle/BLURD`.
What (other) tasks could the dataset be used for?	The BLURD datasets have utility in a number of research areas, Appendix A discusses some potential future work. Some examples are investigating disentanglement learning, causal reasoning and analysis, out of distribution research and improving generative models
Is there anything about the composition of the dataset or the way it was collected and preprocessed/cleaned/labeled that might impact future uses?	No.
Are there tasks for which the dataset should not be used?	Commercial purposes
DISTRIBUTION	
Will the dataset be distributed to third parties outside of the entity on behalf of which the dataset was created?	Yes, the dataset will be publicly available at `https://www.blurd.xyz/`.
How will the dataset will be distributed?	The dataset will be available for download as a tarball on `https://www.blurd.xyz/`. We are currently also engaging Huggingface to host the dataset
Will the dataset be distributed under a copyright or other intellectual property (IP) license, and/or under applicable terms of use (ToU)?	The license of the dataset is **CC BY-NC 4.0**.

Have any third parties imposed IP-based or other restrictions on the data associated with the instances?	Yes, some digital assets are subject to the Human Generator Asset License `https://help.humgen3d.com/FAQ/Human+Generator+Asset+License`, all other assets are under the CC0 license.
Do any export controls or other regulatory restrictions apply to the dataset or to individual instances?	N/A
MAINTENANCE	
Who will be supporting/hosting/maintaining the dataset?	The authors. See `https://www.blurd.xyz/` for additional details.
How can the owner/curator/manager of the dataset be contacted?	Please contact the corresponding author of this paper.
Is there an erratum?	No.
Will the dataset be updated?	Yes.
If the dataset relates to people, are there applicable limits on the retention of the data associated with the instances (e.g., were the individuals in question told that their data would be retained for a fixed period of time and then deleted)? If so, please describe these limits and explain how they will be enforced.	N/A.
Will older versions of the dataset continue to be supported/hosted/maintained? If so, please describe how. If not, please describe how its obsolescence will be communicated to dataset consumers.	Yes, excluding extenuating factors.
If others want to extend/augment/build on/contribute to the dataset, is there a mechanism for them to do so?	They can create an issue on the github `https://github.com/squaringTheCircle/BLURD`.

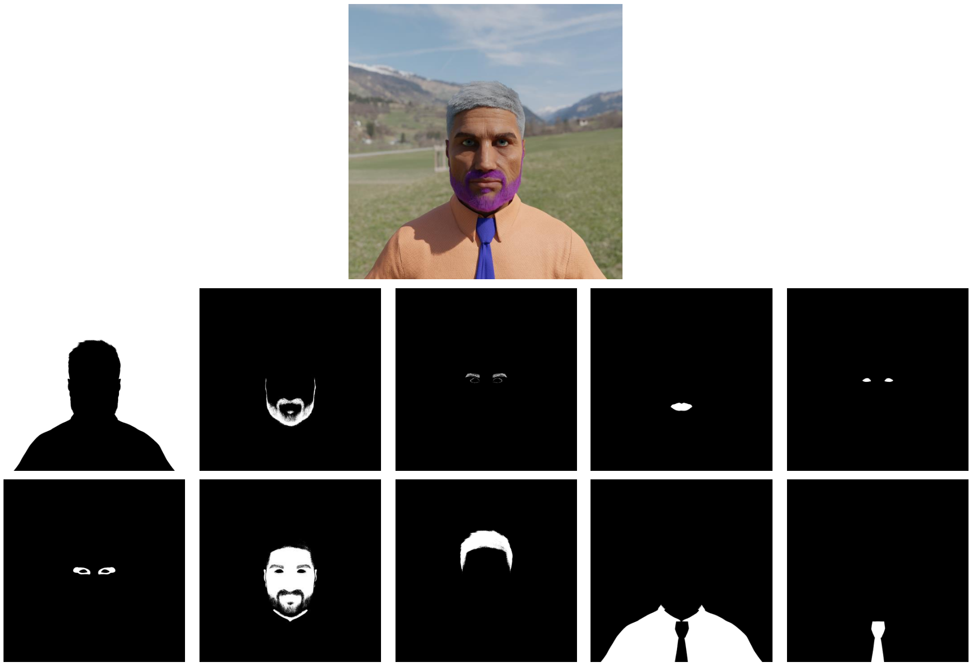

Figure 10: An example of the segmentation masks produced with each render. The top image is the 3D render, then reading from left to right is the background mask, beard mask, eyebrow and eyelash mask, lip mask, eyes mask, eyeshadow mask, face mask, hair mask, shirt mask and tie mask. The eyebrow and eyelash masks are combined here for brevity, however they are produced as separate masks.

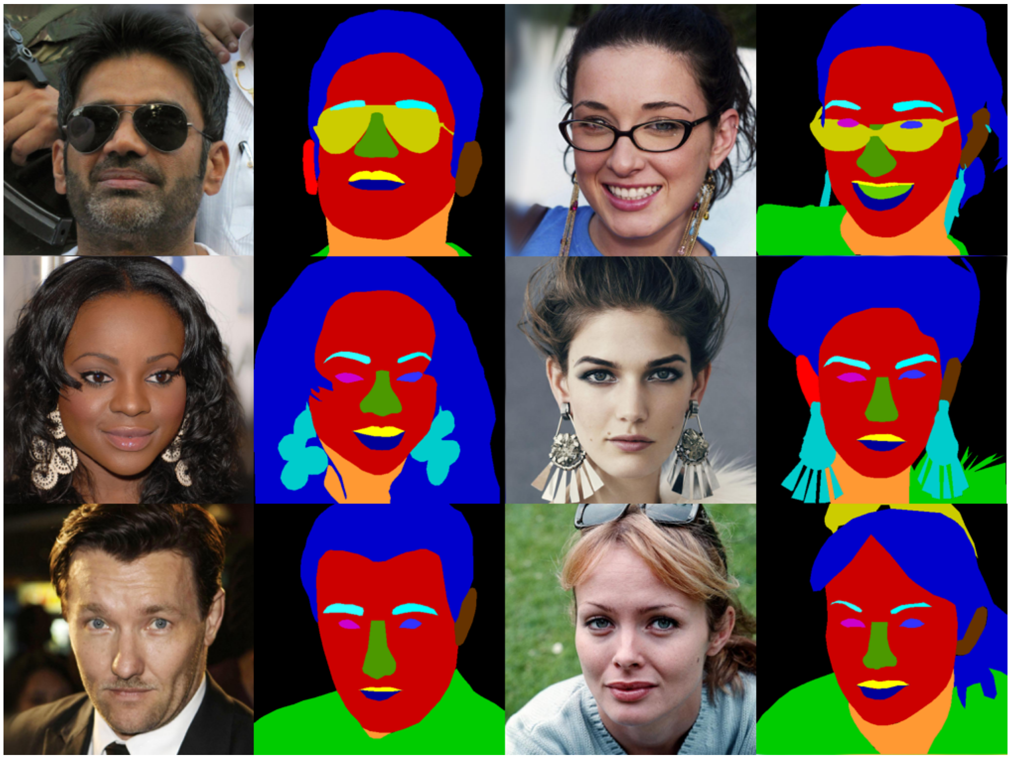

Figure 11: Examples of the segmentation masks from the **CelebAMask-HQ** dataset. **CelebAMask-HQ** contains 18 classes which we reduced down to 6. Masks from objects such as eyeglasses, earrings and hats were removed. Separate masks for left/right or upper/lower anatomy were merged. Finally seperate masks for the nose, ears and neck were merged.

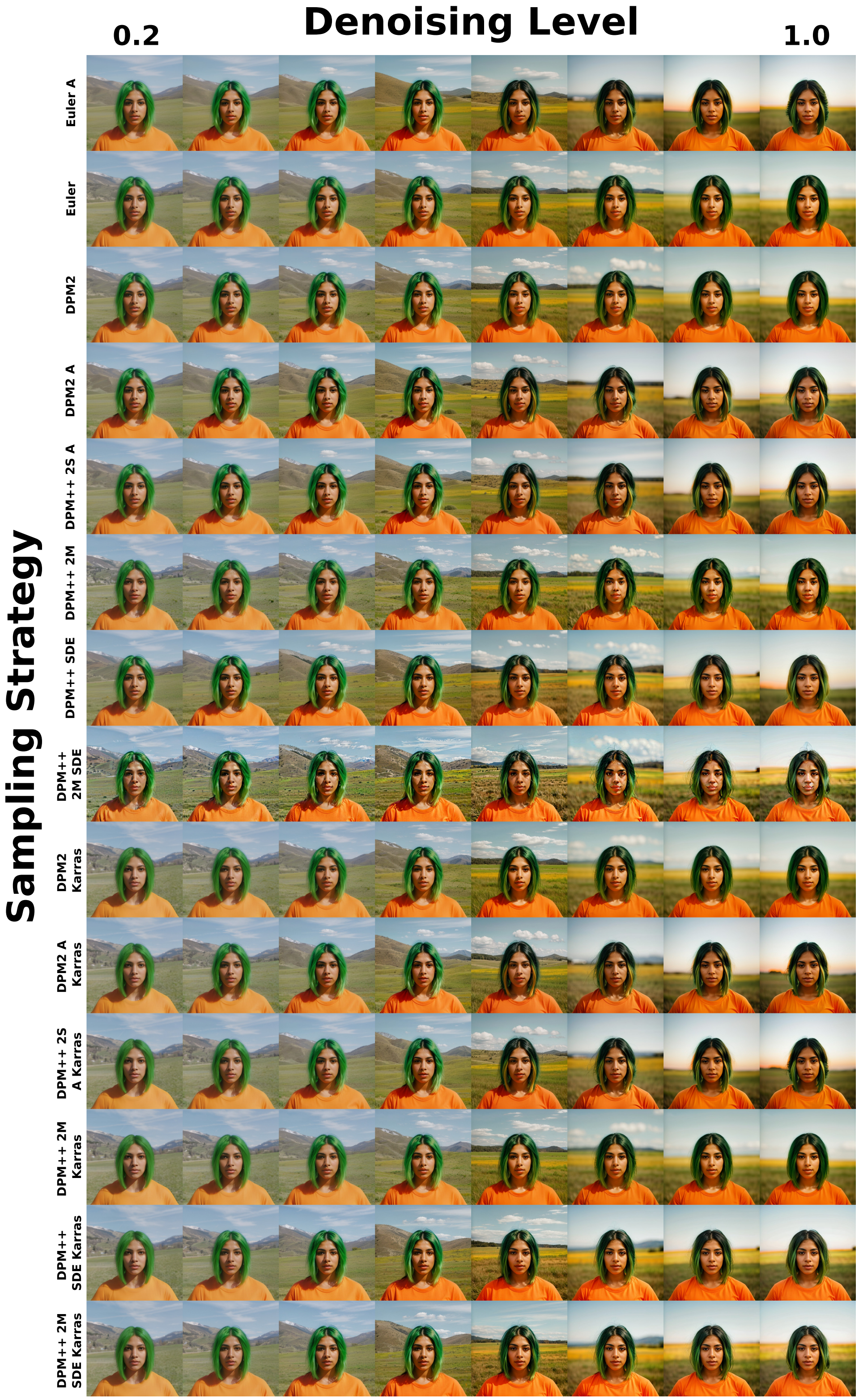

Figure 12: Experiments with different levels of denoising, and different samplers. We used visual inspection to select a denoising level that offered a good balance between adhering to the 3D render and generating realistic imagery.

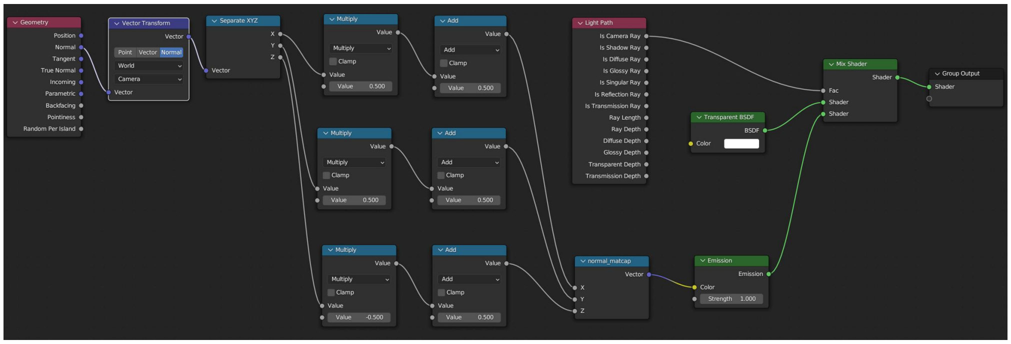

Figure 13: The normal map shader node tree. Built with Blender's node editor. The node tree defines a material, which we apply as a material override in a render layer. The custom normal map shader we built for **BLURD** is available in the *blend* file we release at `https://www.blurd.xyz/`

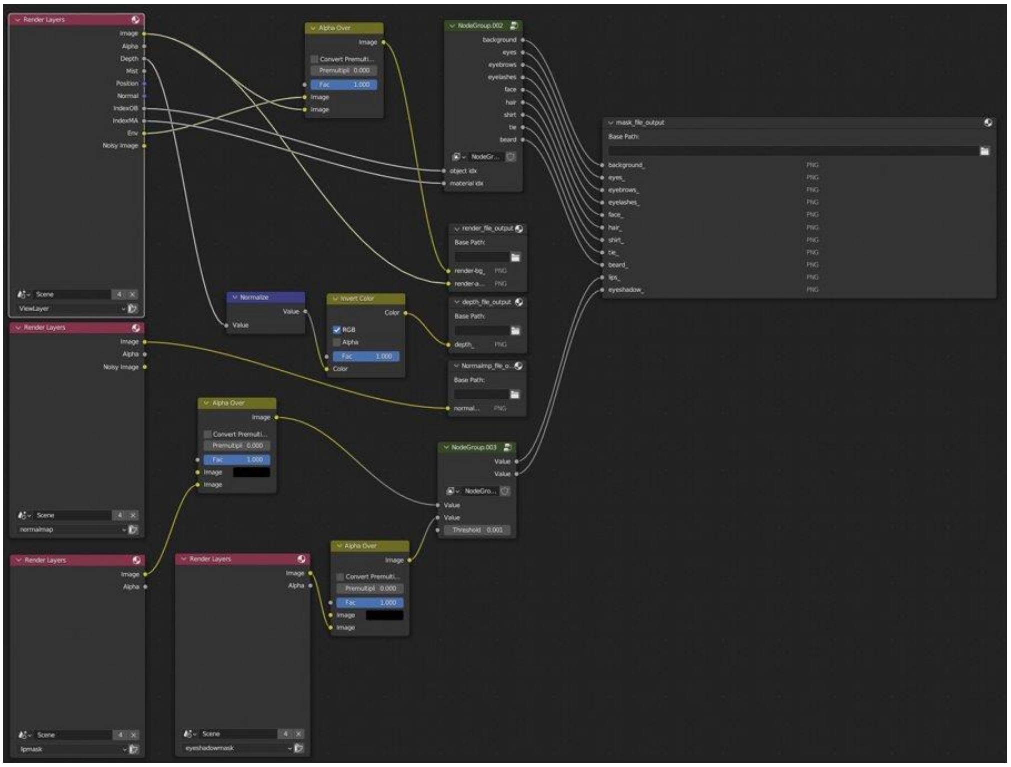

Figure 14: The compositor node tree setup. Here we show how we utilise Blender's render layers and passes for **BLURD**

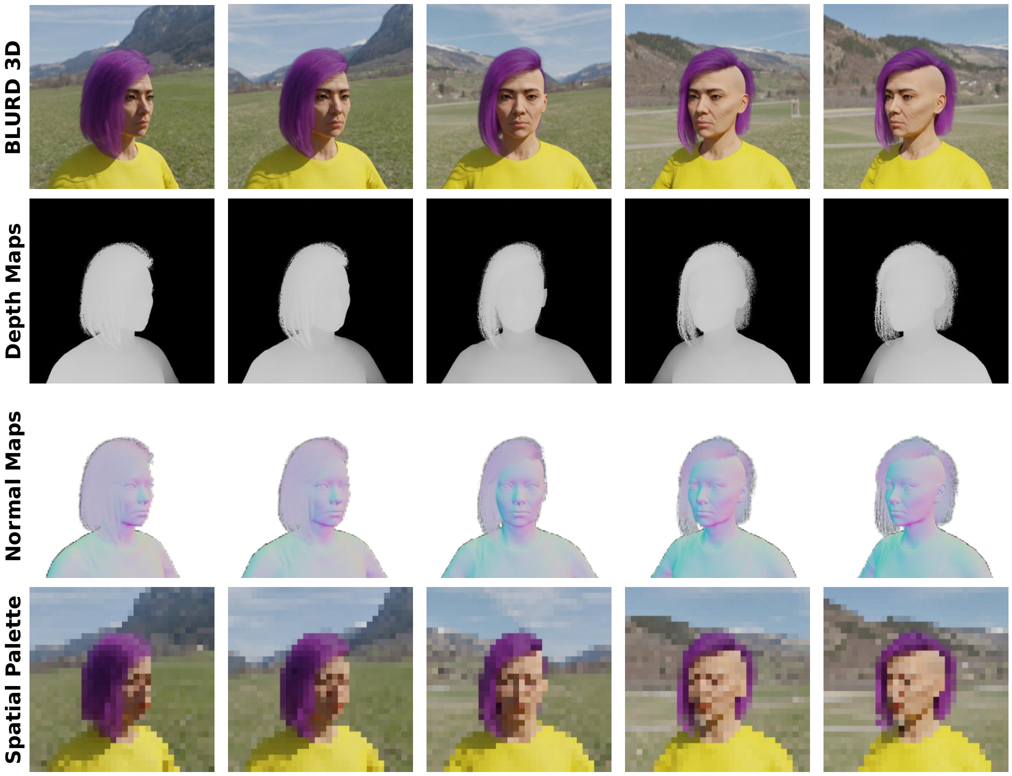

Figure 15: An example from the **BLURD 3D** dataset. Here all the factors of variation are held fixed except for the camera angle which varies between five values. The top row is the 3D renders, the middle top contains images of the depth maps, the middle bottom row contains images of the normal maps and lastly the bottom row consists of the spatial palette. The depth maps and normal maps are made using Blender while the pixel grids use a separate post processing step.

C Related Works Additional Details

Datasets created for representation learning have been extensively employed to study representation spaces of deep learning models [8, 23, 6, 46, 33, 31, 24, 38, 28]. Often creators of datasets for representation learning rely on creating synthetic toy datasets to solve simplified versions of the problem and extrapolate their findings to real-world scenarios [8, 23, 6, 46, 33, 31, 24, 38].

Early examples include dSprites, which consists of binary 2D images of hearts, ellipses, and squares in low resolution [36] and its variations Color-dSprites, Noisy-dSprites, Scream-dSprites [64]. Small-NORB, is a 3D dataset of 50 toys of 5 categories: four-legged animals, human figures, airplanes, trucks, and cars. Reed *et. al* introduced a 2D dataset of video game sprites and Cars3D [46]. Cars3D utilizes 199 CAD models to generate color renderings of cars from different rotation angles [46].

Later, datasets with a greater diversity of 3D shapes became for common [8, 28]. For example the 3dshapes dataset offers procedurally generated objects with ground truth independent latent factors such as floor color, wall color, object color, scale, shape, and orientation [8]. Futhermore, datasets like CLEVR, Biased Cars, and ShapeNet offered even greater experimentation with factors of change and controlled evaluation settings [28, 35, 12]. However, these datasets often lack realism, posing challenges for broader application [28, 35, 12, 6].

While simple 2D or 3D datasets offered great control they were highly limited in their ability to extrapolate to more complex scenarios. Soon researchers turned to large scale collections of real world dataset [34, 24, 10]. Liu *et al.* introduced CelebA and LFWA, requiring a costly professional labeling company to hand label forty face attributes and five key points [34]. In a larger effort Cao *et al.* used Google Image Search to create VGGFace2, a dataset consisting of 3.31 million images of faces with 9131 individuals [10]. However, a more controlled dataset is required for quantitative investigations of inductive biases, sample complexity, and the interplay between simulations and the real world in disentanglement models. Hence Gondal *et al.* introduced MPI3D [24] a dataset which consists of over one million images of physical 3D objects with seven factors of variation. MPI3D utilised a controlled recording setup and simple pendulum-like objects to create a truly controllable photo-realistic dataset [24]. While MPI3D achieved both realism and controllability the setup was highly contrived and limited to the simplest of geometric shapes.

To address the expense and limited nature of real world capture datasets, researchers again looked to the rapidly improving 3D rendering technology to obtain Photo-realism [6, 18]. Simulators have been explored in specific domains like autonomous driving, for example CARLA, a popular self-driving car simulator, offered researchers a highly realistic environment with controllable factors such as environmental conditions, static and dynamic actors, and map rendering [18].

Platforms like ThreeDWorld, based on the Unity game engine, provide interactive environments for creating datasets [21]. By customizing scene setup and data sampling through a low-level API, researchers could design simulations tailored to their needs [21]. The Synthetic Visual Concepts (SyVIC) dataset demonstrated the use of ThreeDWorld for vision-language model training [11].

In contrast, the authors of PUG proposed leveraging the Unreal Engine directly to create custom environments for representation learning [6]. By using the existing Epic Games marketplace, researchers can access a wide range of content and enjoy the flexibility of a powerful and widely used game engine [6]. This approach attempted to achieve photo-realism with high quality game assets and industry leading game graphics engines [6]. However, while PUG is useful in it's own right it's reliance on a real-time game engine causes it to fall far short of true real world photography[6].

Outside of representation learning, the field of facial recognition has been pushed to create highly synthetic realistic datasets to alleviate privacy concerns [60, 43, 15, 26, 37, 29]. Trigueros *et al.* used GANs that could disentangle identity-related attributes from non-identity-related attributes [60]. Similarly, Qiu *et al.* introduced SynFace mixing real faces with DiscoFaceGAN generated images [43], while Colbois *et al.* and Grimmer *et al.* independently showed you could use StyleGAN2 to create a controllable face dataset useful for facial recognition tasks, albeit with slightly different methods. [15, 26]. Finally, Melzi *et al* used GANs, Stable Diffusion and Dreambooth to create a controllable and realistic face dataset in their GANDiffFace approach [37]. By first using a GAN's latent space they could generate a controllable identity, then they trained a Stable Diffusion model with Dreambooth to achieve greater realism [37]. However this approach still suffers from relying on the GAN latent space with sacrifices too much control over a 3D synthetic environment [37]. Additionally, Melzi *et al* required an expensive fine-tuning of Stable diffusion per identity.

In summary, while synthetic toy datasets have been instrumental in studying disentanglement models, there is a need for more controlled datasets that bridge the gap between simulation and the real world. Photo-realistic datasets, simulators like CARLA, ThreeDWorld and PUG have impressive 3D graphics, yet still fall far short of true photo-realism. On the other hand, real world datsets such as MPI3D or CelebA are expensive to collect and are either highly limited to simple shapes or contain too many co-founding factors. Methods such as GANDiffFace seek to address these issue my using data-driven approaches, however their reliance on the latent space of GANs sacrifices the control over the factors of variation possible in 3D environments. In stark contrast, our proposed approach of using Blender together with Stable diffusion and ControlNet offers flexibility, tight control over the factors of variation and true Photo-realistic imagery far surpassing anything possible with computer graphics alone.

D Limitations and Biases in Pre-trained Models Additional Details and Figures

We use the BLURD 3D/SD datasets to assess the zero-shot accuracy of a number of pre-trained CLIP backbones trained on various datasets. We conduct experiments for both *single-factor* zero-shot accuracy and *multi-factor* zero-shot accuracy. For each experiment we employ several CLIP backbones, specifically ViT-B-32 trained on the OpenAI, Laion400m and Laion2b datasets, ViT-L-14 trained on the OpenAI, Laion2b and Datacomp XL datasets and finally ViT-H-14-CLIPA and ViT-g-14 trained on the Laion2b dataset[27, 14, 44, 52, 20, 51]. For zero-shot accuracy we measure the caption retrieval accuracy with a caption derived from the factors of variation. The captions are derived from the template “A picture of a *age race gender* with/wearing *factors*" where *factors* is a comma separated list of any remaining factors.

Factors not under consideration are replaced with the empty string with the exception of *gender* which is replaced with “person”. An example of a single factor caption is “A picture of person with red hair”. An example of a multi-factor caption is “A picture of a 70 y.o. asian female with a undercut hair style, wearing a blue shirt, with red hair”. A caption is considered retrieved if it has the highest cosine similarity out of all the captions considered. For *single-factor* zero-shot accuracy the set of captions is all captions generated by the single-factor template for every instance of that factor. For example, for the gender factor the captions would be “A picture of a male" and “A picture of a female".

In the *multi-factor* zero-shot classification accuracy case the multi-factor template is used with all factors excluding camera angle. Computing the accuracy over the classes of all combinations of factors would be too challenging of a zero-shot task for obtaining interpretable results. Therefore we simplify the task by instead uniformly sampling 100 unique combinations of factors and a single image corresponding to one of the combinations. We then compute the caption retrieval accuracy using the factor combinations to generate the caption. We repeat the process over 2000 trials per CLIP model, pre-training dataset and BLURD 3D/SD dataset.

Understanding where current pre-trained CLIP models perform well and where they struggle provides valuable insights into the limitatons and bias of training datasets and model architectures. In section 4 we discussed how the results of zero-shot accuracy experiments can be analyzed to reveal model limitations and biases. In subsection D.1 we provide additional confusion matrices to motivate future research. For instance, we note that the Laion400m dataset struggles to determine gender when used to pre-train the ViT-B-32 backbone, where the OpenAI and Laion2b perform well. In other cases certain factors seem difficult regardless of the pre-trained model or dataset use, for example most models struggled to determine the correct age of the image subject.

D.1 Additional Confusion Matrices for Single Factor Zero-Shot Classification Results

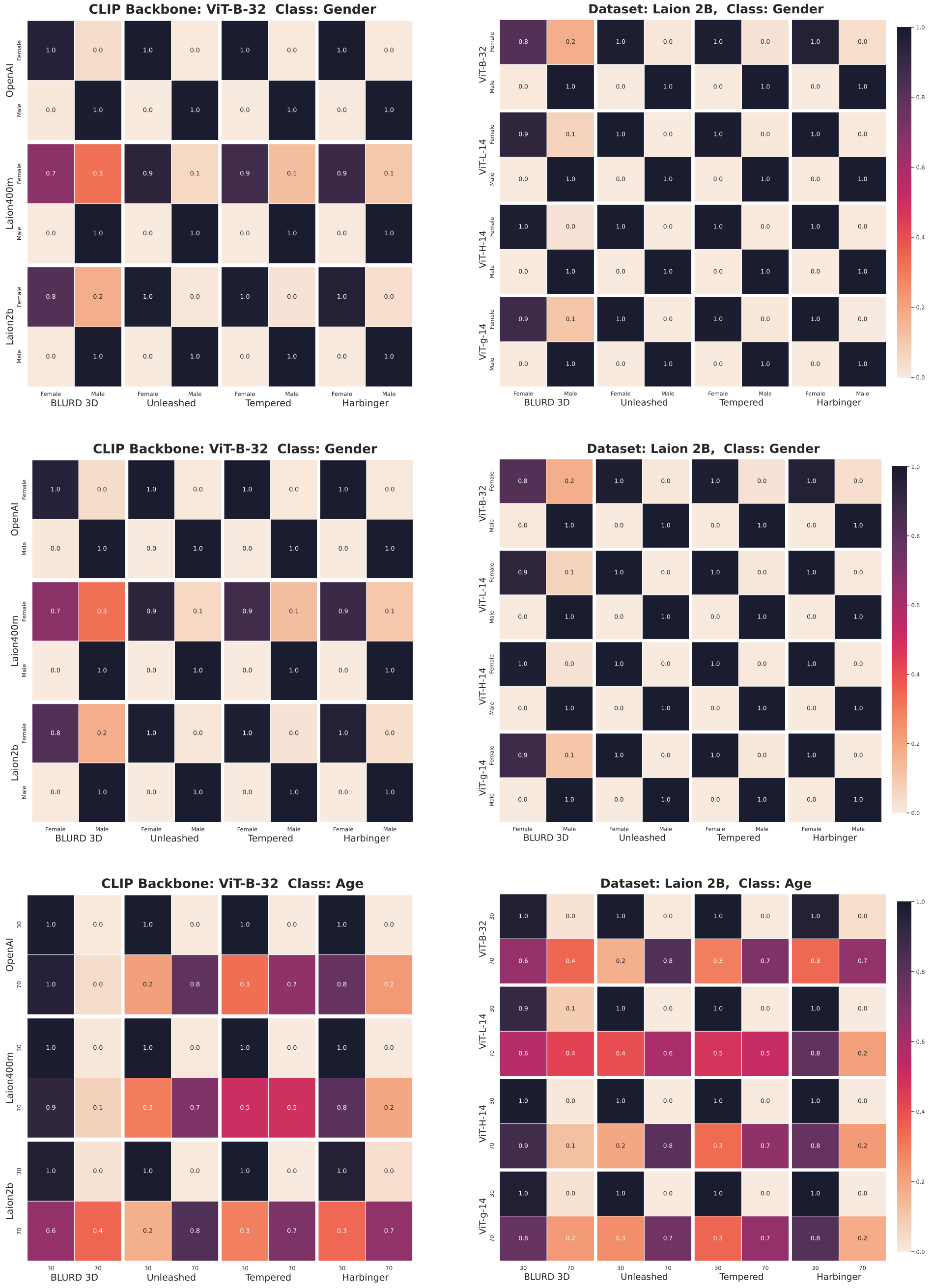

Figure 16: Zero-shot confusion matrices for the factor gender, race and age. *Left:* We fix the CLIP backbone to ViT-B-32 and provide a confusion matrix for each of the CLIP training datasets OpenAI, Laion400m and Laion2b evaluated each member of **BLURD**. *Right:* We fix the training dataset to Laion2b and provide a confusion matrix for the CLIP backbones ViT-B-32, ViT-L-14, ViT-H-14-CLIPA and ViT-g-14 evaluated each member of **BLURD**.

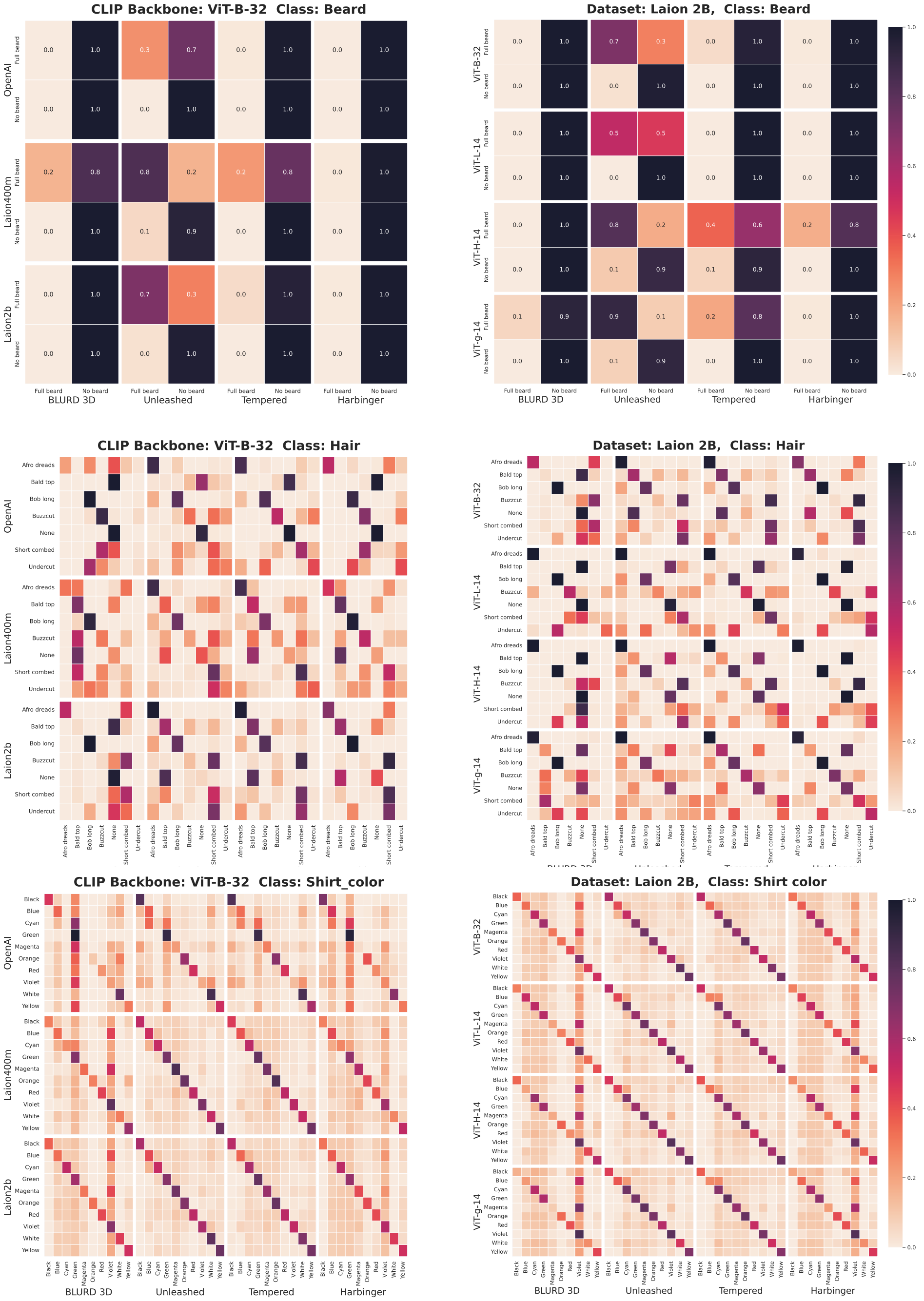

Figure 17: Zero-shot confusion matrices for the factor beard, hair and shirt color. *Left:* We fix the CLIP backbone to ViT-B-32 and provide a confusion matrix for each of the CLIP training datasets OpenAI, Laion400m and Laion2b evaluated each member of **BLURD**. *Right:* We fix the training dataset to Laion2b and provide a confusion matrix for the CLIP backbones ViT-B-32, ViT-L-14, ViT-H-14-CLIPA and ViT-g-14 evaluated each member of **BLURD**.

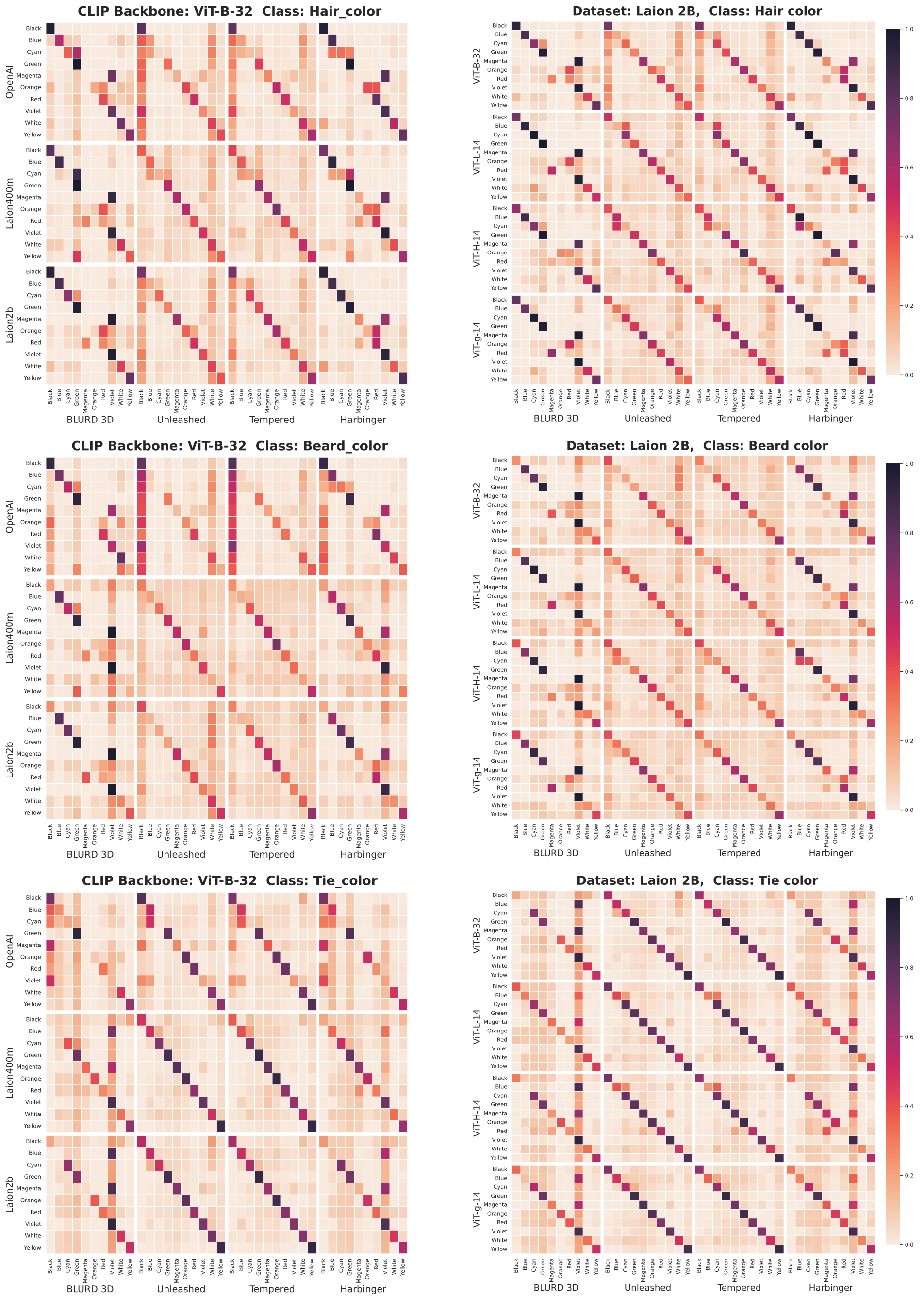

Figure 18: Zero-shot confusion matrices for the factor beard, hair and shirt color. *Left:* We fix the CLIP backbone to ViT-B-32 and provide a confusion matrix for each of the CLIP training datasets OpenAI, Laion400m and Laion2b evaluated each member of **BLURD**. *Right:* We fix the training dataset to Laion2b and provide a confusion matrix for the CLIP backbones ViT-B-32, ViT-L-14, ViT-H-14-CLIPA and ViT-g-14 evaluated each member of **BLURD**.

E Equivariance Additional Details and Figures

Equivariance is a property that describes how a representation changes predictably when a specific factor is altered, independently of any changes made to other factors [6]. In the ideal case, a change to a single factor will not alter the representation of any other factor. To measure image equivariance we computed the difference vector between the embeddings of two images that underwent a factor change (e.g a change in hair color), then sample another set of two images with different factor values, yet undergoing the same factor change to obtain a second difference vector. We then calculate the cosine similarity between the difference vectors. For text equivariance a difference vector between the embeddings of two images is calculated as before, then the underlying factor values of the images are used to generate two captions using the same template as in Appendix D. The two captions are used to obtain a second difference vector and the cosine similarity between the difference vectors is calculated. See Figure 6 for a visual representation of how equivariance is calculated. Finally for each possible factor change we sample 500 sets of displacement vectors and average the resulting equivariances, repeating this for every model and pre-training dataset.

In the case of the 3D render to photo-realism factor, we sampled images from BLURD 3D then retrieved the corresponding images with the identical factor values from the BLURD SD datasets. Equivariance can then be calculated using image equivariance as before. We repeat this calculation for 3000 samples for every model and pre-training dataset, averaging the result. To our knowledge BLURD provides the only measure of the equivariance of varying in the 3D render to photo-realism factor.

As equivariance measures the cosine similarity between two vectors its values range from $[-1, 1]$. A value of 1 indicates that every vector is parallel to every other in the sampled set, meaning that under a factor change the representation space is “translated" in some direction ignoring scale. A value of 0 indicates that on average there is no single direction that corresponds to the factor change or in the pathological case that every vector in the sample is orthogonal to every other vector. A value of -1 is similar to the 1 case however the translation occurs in the opposite direction to the initial vector. An example were the equivariance might be -1 would be when the image displacement vectors are all parallel and facing the same direction and the text displacement vectors are similarly all parallel, however they face the opposite direction. High equivariance is desirable as it indicates that a particular direction in the representation space encodes for the factor. Furthermore, if each factor corresponds to a different direction and has high equivariance, this indicates a highly disentangled representation space allowing for a great deal of interpretability [23, 6, 46, 33, 31, 24, 38]. In subsection E.1 we provide additional equivariance results to motivate future research.

E.1 Equivariance Results Additional Figures

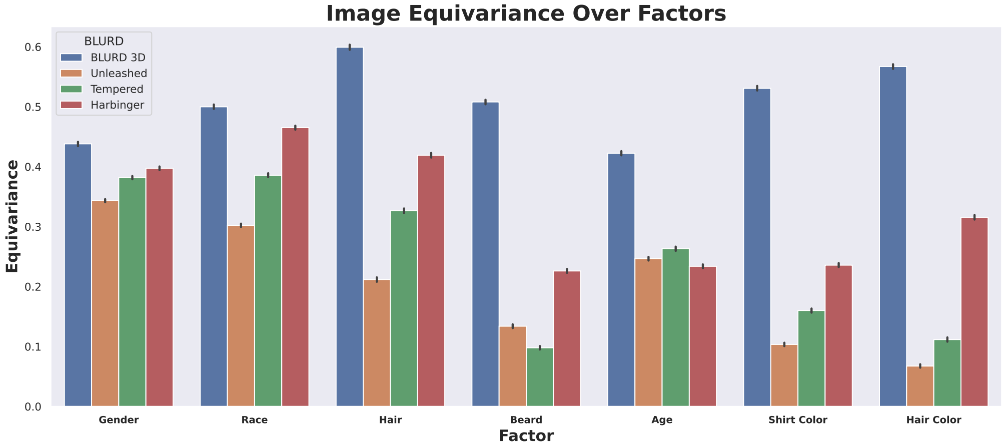

Figure 19: Image equivariance over seven factors. Image equivariance compares the cosine similarity between the displacement vectors of two different images undergoing the same factor change. Here we evaluate the same CLIP backbones and pretraining datasets from Figure 4. Plots for tie color and beard color are omitted as they are equal to shirt color and hair color respectively.

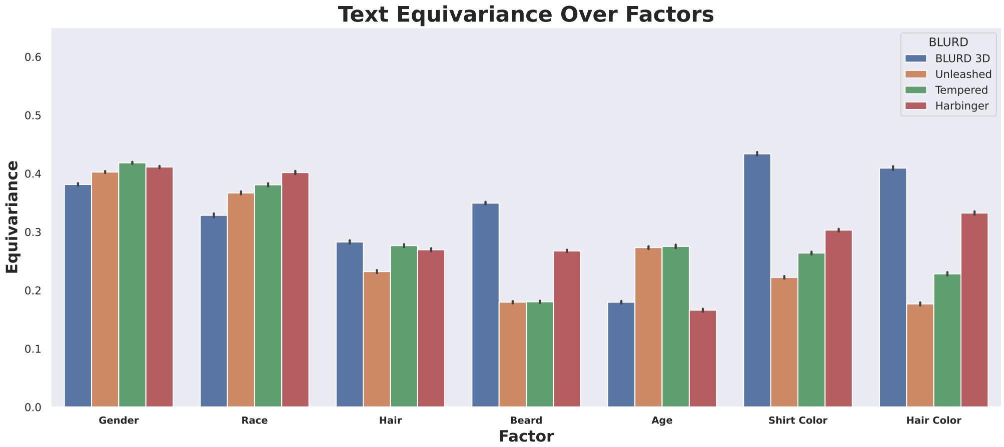

Figure 20: Text equivariance over seven factors. Text equivariance compares the cosine similarity between the displacement vectors of an images undergoing a factor change and a text caption undergoing the same factor change. Here we evaluate the same CLIP backbones and pretraining datasets from Figure 4. Plots for tie color and beard color are omitted.

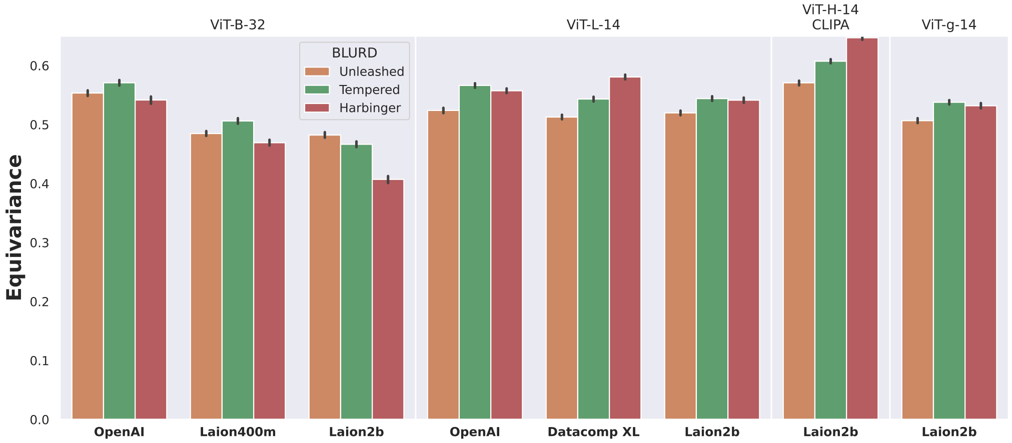

Figure 21: Image equivariance of a transformation from 3D render to photo-realism for each of the CLIP backbones ViT-B-32, ViT-L-14, ViT-H-14-CLIPA and ViT-g-14 trained on the datasets OpenAI, Laion400m, Laion2b and Datacomp XL datasets.

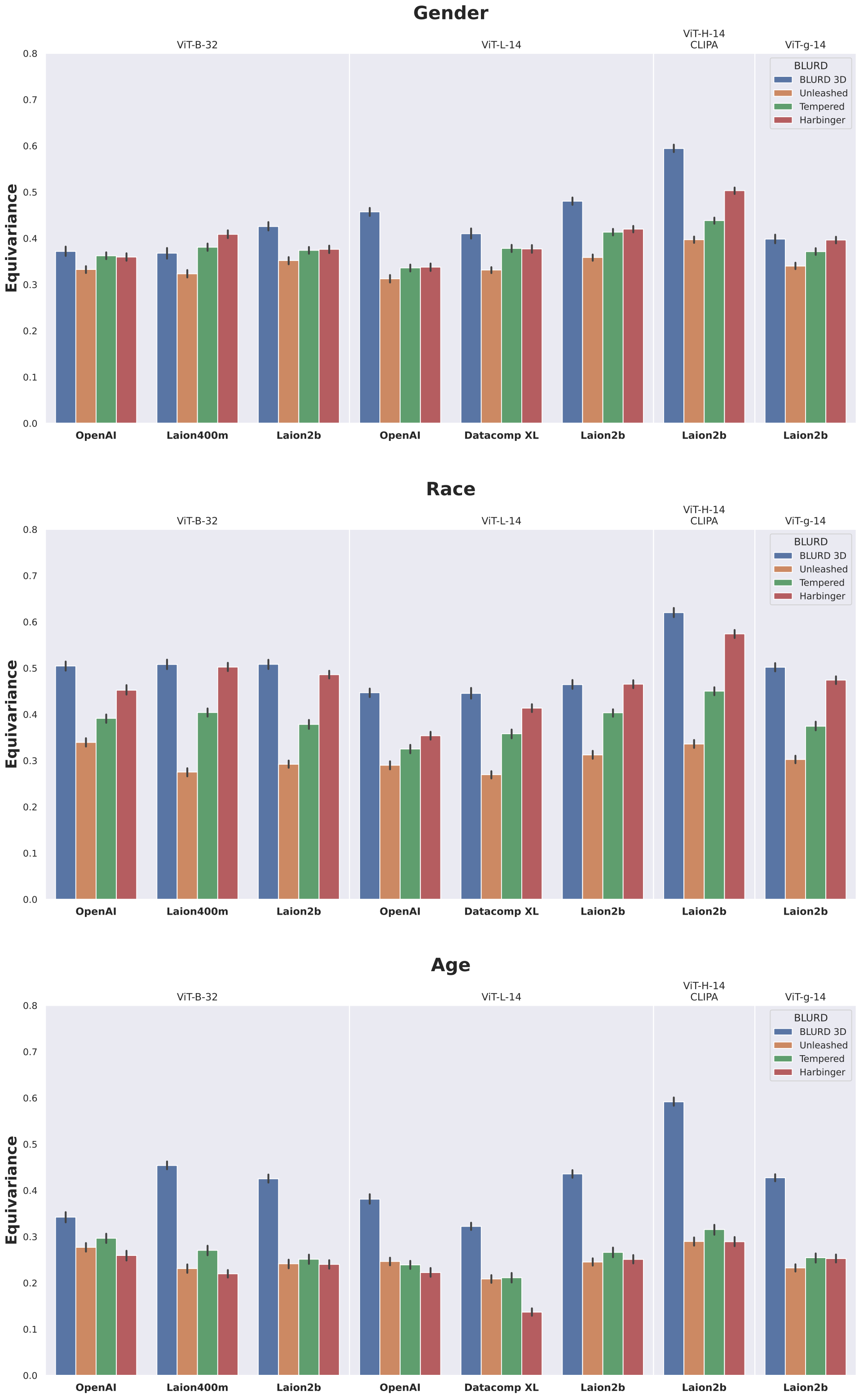

Figure 22: Image equivariance of a transformation in the gender, race and age factors for each of the CLIP backbones ViT-B-32, ViT-L-14, ViT-H-14-CLIPA and ViT-g-14 trained on the datasets OpenAI, Laion400m, Laion2b and Datacomp XL datasets.

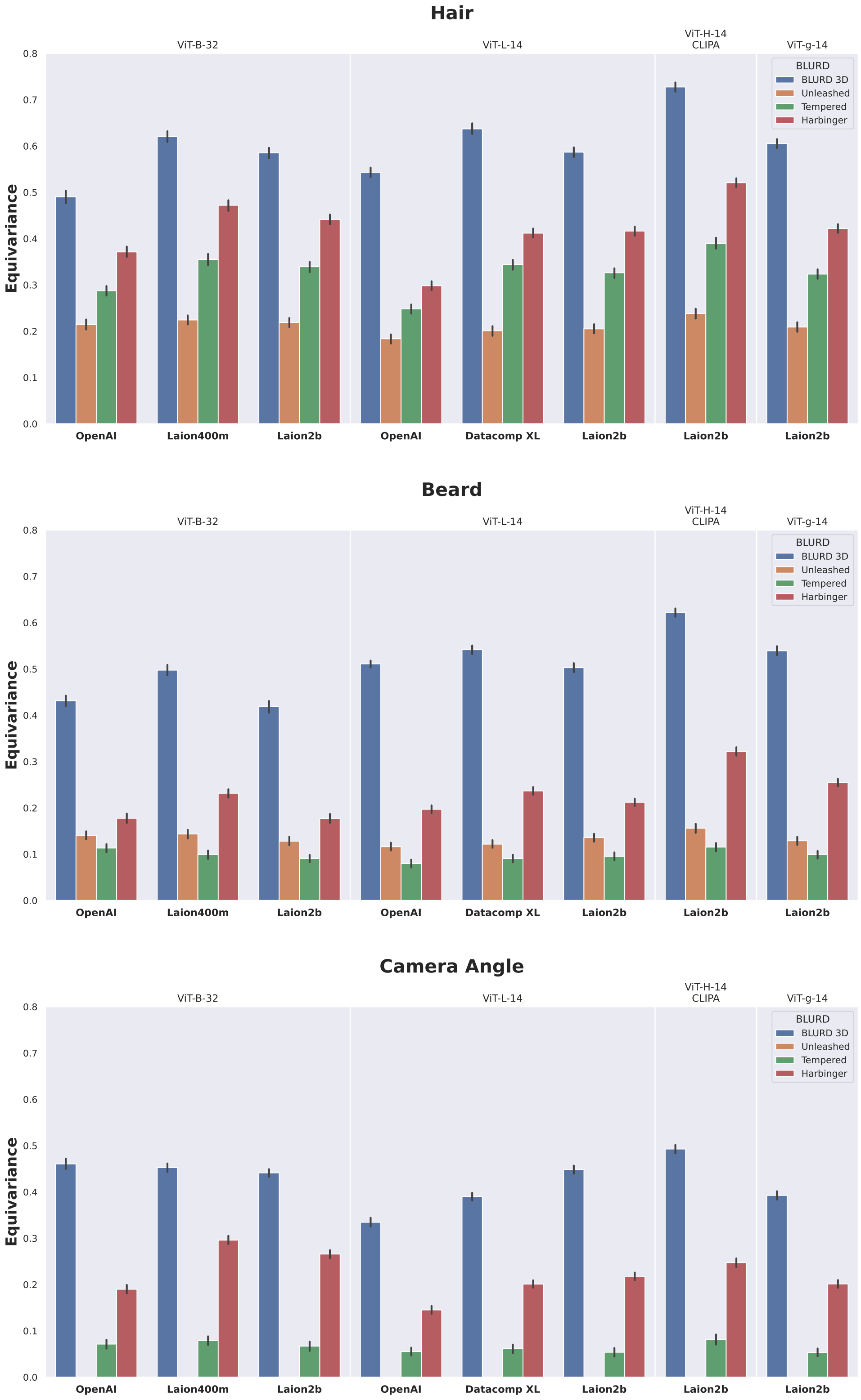

Figure 23: Image equivariance of a transformation in the hair, beard and camera angle factors for each of the CLIP backbones ViT-B-32, ViT-L-14, ViT-H-14-CLIPA and ViT-g-14 trained on the datasets OpenAI, Laion400m, Laion2b and Datacomp XL datasets.

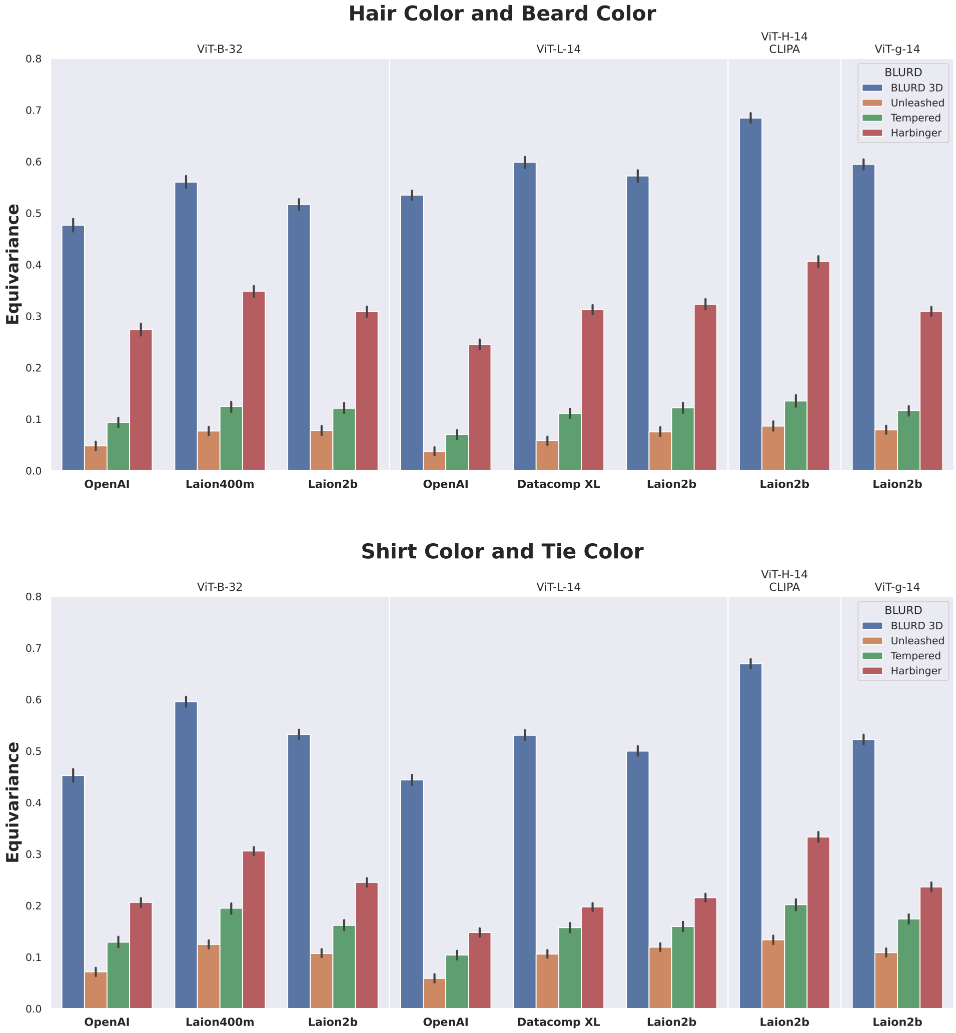

Figure 24: Image equivariance of a transformation in the hair, beard, shirt and tie color factors for each of the CLIP backbones ViT-B-32, ViT-L-14, ViT-H-14-CLIPA and ViT-g-14 trained on the datasets OpenAI, Laion400m, Laion2b and Datacomp XL datasets.

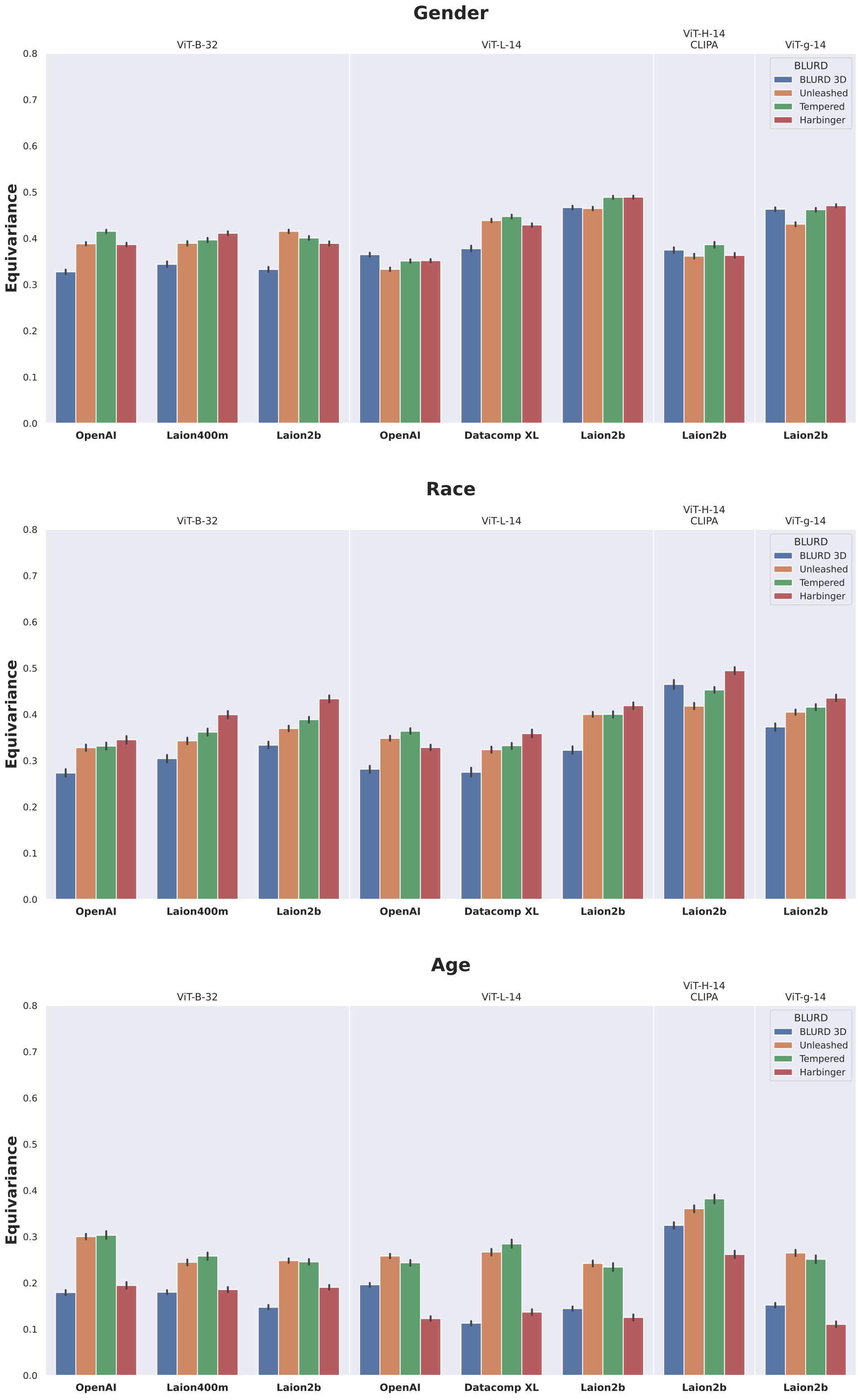

Figure 25: Text equivariance of a transformation in the gender, race and age factors for each of the CLIP backbones ViT-B-32, ViT-L-14, ViT-H-14-CLIPA and ViT-g-14 trained on the datasets OpenAI, Laion400m, Laion2b and Datacomp XL datasets.

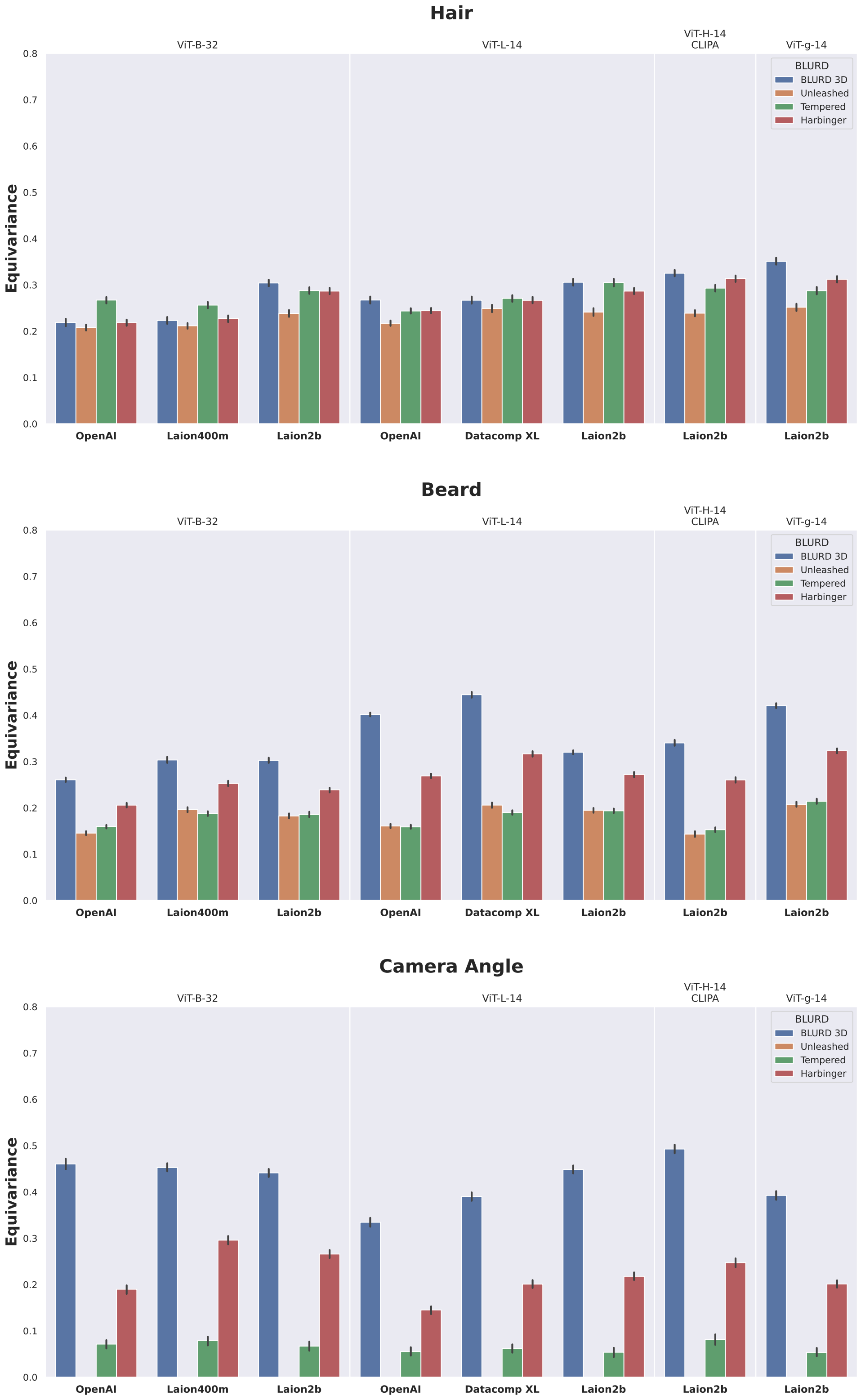

Figure 26: Text equivariance of a transformation in the hair, beard and camera angle factors for each of the CLIP backbones ViT-B-32, ViT-L-14, ViT-H-14-CLIPA and ViT-g-14 trained on the datasets OpenAI, Laion400m, Laion2b and Datacomp XL datasets.

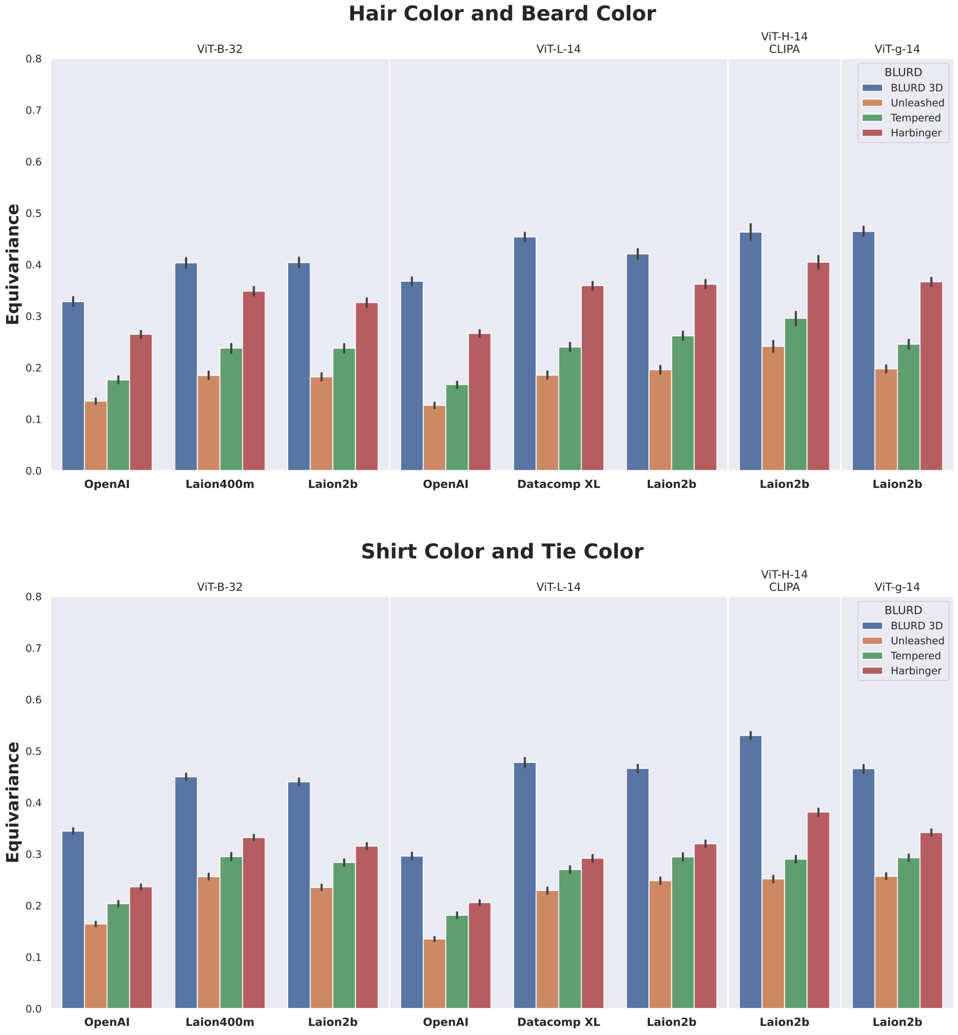

Figure 27: Text equivariance of a transformation in the hair, beard, shirt and tie color factors for each of the CLIP backbones ViT-B-32, ViT-L-14, ViT-H-14-CLIPA and ViT-g-14 trained on the datasets OpenAI, Laion400m, Laion2b and Datacomp XL datasets.

F Impact of Low-Poly Assets on the BLURD SD Pipeline Additional Details and Figures

Here we provide additional details to the experiments and analysis in section 4. Recall that we investigate the impact of using lower quality 3D models and textures in the proposed BLURD pipeline. To that end we replace the high-quality assets used in BLURD with easier to produce low-poly assets with simple single material textures. We acquired assets provided under the Creative Commons licenses (CC0 and CC-BY) from Blender Studio, MakeHuman, and the community [4, 57]. We then utilized either low-poly stylized versions or applied the decimate modifier in Blender to convert high-poly meshes into low-poly ones. Overall, we demonstrate that by adjusting the generation parameters, we can mitigate the shortcomings associated with low-quality 3D assets.

In the first experiment, we compared the performance of the particle-based hair system used in BLURD with a low-poly hair mesh variant, when varying the image-to-image (img2img) denoise level. The results revealed that the particle-based hair system is consistently more photo-realistic, even at low denoise levels. In contrast, the low-poly mesh hair only appears more realistic at very high denoise levels and retains artifacts due to the low-polygon geometry. Figure 28 displays a sample of images generated with denoise levels from 0.25 to 0.85.

In the second experiment shown in Figure 29, we analyzed the effect of changing the img2img output resolution from 512×512 to 1024×1024, comparing the particle-based hair and low-poly hair mesh. In this case, lower resolutions aided in making the low-poly mesh hair appear more photo-realistic within the same denoise level. Meanwhile, the particle-based hair system maintained its photo-realistic appearance even at higher resolutions.

Finally, in the third experiment, we replaced the high-quality assets used in our initial BLURD SD pipeline with a low quality analog. Specifically, we utilized a low-poly hair mesh from MakeHuman and a decimated human base mesh from Blender Studio, together with simple single-color materials. Figure 8 displays a sample of images with different weightings for the depth and normal map ControlNets at two different levels of denoising. We observed that the depth ControlNets and normal map ControlNets are highly sensitive to geometry, resulting in visible artifacts in the final image.

We varied the weights of both the depth ControlNets and normal map ControlNets from 1.0 to 0.0 across different denoise levels and found that either ControlNet will independently produce artifacts, demonstrating that both ControlNets preserve the low-polygon geometry well. However, altering the weights of the depth and normal ControlNets, or raising the denoise level, can alleviate this issue. Nonetheless, this comes at the cost of reducing controllability over the desired factors of variation.

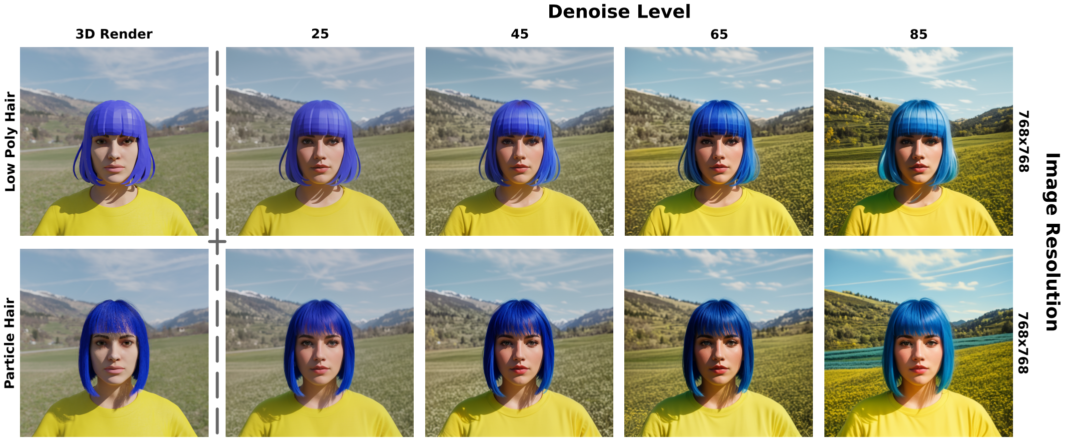

Figure 28: Impact of replacing the particle based hair system with a simple low-poly hair mesh[1]. Here we use the ***BLURD SD:** Harbinger* pipeline but vary the img2img denoise level from 0.25 to 0.85. As can be seen in the figure the more realistic particle based hair system obtains photo-realism even at low denoise levels. On the other hand the low-poly mesh hair only begins to appear more like real hair at very high denoise levels and retains some artifacts from the low-polygon geometry.

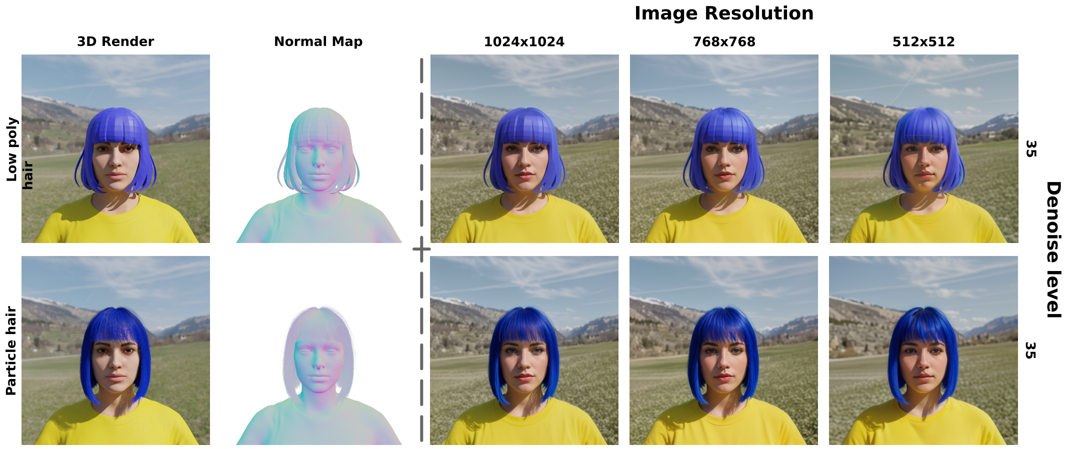

Figure 29: Impact of replacing the particle based hair system with a simple low-poly hair mesh[1]. Here we use the ***BLURD SD:*** *Harbinger* pipeline but vary the img2img image resolution from 512×512 to 1024×1024. In the case of the low-poly mesh lower resolutions aid in making the hair appear more photo-realistic, even within the same denoise level. However the particle based hair system is able to retain photo-realism even at high resolution levels.

[1] Low-poly hair mesh (culturalibre_hair_18) was provided by culturalibre under the terms of the of Creative Commons Attribution 4.0