

Understanding How Computer Vision Models See Illusions-Illusions

Zach Salehe
Harvard University

zachsalehe@fas.harvard.edu

Abstract

This work aims to explore how modern computer vision models perceive illusions-illusions—images that are designed to look like common visual illusions while lacking their illusory components. While humans are easily fooled by illusions due to perceptual gaps in our visual system, vision models generally process entire images at once, and thus, need not rely on perceptual filling-in, making them unsusceptible to illusions in the same way. Prior work has shown that, despite this, vision-language models struggle to differentiate between illusions and illusions-illusions. I show that this is likely a result of illusions and illusions-illusions being close in their feature spaces, demonstrated by extensive feature comparisons on a variety of models. Additionally, I explore the possibility of generating novel illusions. Though my experiments in this regard were mostly unsupporting of this possibility, there is hope that future work may succeed given the right data and priors. Code and data are available at https://github.com/zachsalehe/neuro240_illusionsillusions.

1. Introduction

Understanding the human perception of visual illusions has been long-studied in fields such as neuroscience and psychology. Following the emergence of computer vision, understanding computers’ perception of illusions has become quite the topic of interest for many researchers as well. While this is still very much an unsolved problem, I would like to instead turn my attention to a neighboring problem.

Illusions-illusions [13] are, in a sense, fake illusions. That is, they are images designed to look like common visual illusions, but have been stripped of all their illusory components (see Fig. 1 for an example). By way of investigating the ensuing questions, I hope to provide an understanding of how computer vision models see illusions-illusions.

Are current computer vision systems susceptible to human visual illusions? I believe that current computer vi-

sion systems are not susceptible to many human visual illusions, mainly due to the inherent differences in how humans and computers process visual information. Due to biological constraints, only a small region of a human’s visual field is in sharp focus at any given time. As a result, the brain must continuously infer and fill in visual details in the peripheral regions in order to overcome such perceptual gaps [6]. Many genres of illusions take advantage of these visual shortcomings; they trick our brains into perceiving things differently than how they truly are.

On the other hand, computer vision systems have no perceptual gaps to fill, as they can take each individual pixel into consideration when reasoning about an image. For this reason, however, vision models can be susceptible to adversarial attacks, in which they are sensitive to extremely small perturbations at each pixel that would go unnoticed by any human. In a sense, adversarial attacks are like visual illusions for computers.

Prior work has shown evidence that current vision-language models can easily recognize illusions, but also falsely recognize illusions-illusions as their illusory counterparts [13]. Consequently, I do not believe these models to be susceptible to these illusions at all. Rather, I think they are just aware of what local features various illusions tend to contain. Given that vision-language models operate on image features, their confusion leads me to believe that illusions and illusions-illusions of the same type are very closely related in the models’ feature spaces. In this work, I show this to be true on open source vision-language models PaliGemma 2 [12] and Qwen2.5-VL [1].

How do CNNs see these images? Broadly speaking, CNNs see images as a large collection of features. The learned feature representations vary between CNNs, depending on what kind of information was most valuable for the task it was trained to perform. Though, pre-trained general-purpose feature representations do also exist [9].

The key thing to note here is that these features are local—that is, each feature roughly describes a single patch in the image. While the differences between illusions and illusions-illusions are quite noticeable to any human, they

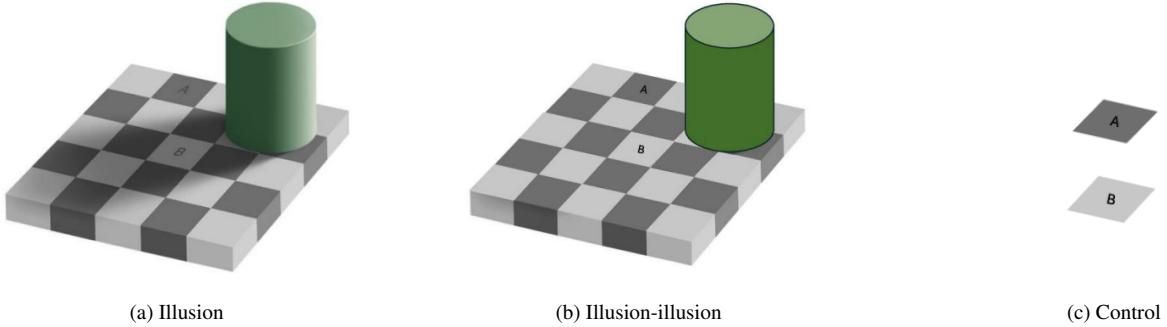


Figure 1. An example of an illusion, illusion-illusion, and control image given the famous Checker Shadow illusion. In the real illusion (a), tile A and tile B are the same color, but the human brain is fooled by the presence of the shadow, making it seem as though tile A is darker. The shadow is removed in the illusion-illusion (b), resulting in tile A and tile B actually being different colors. In the control (c), all that remains are the differently colored tile A and tile B.

are nearly identical to each other at a local scale. As a result, I believe illusions and illusions-illusions exhibit similar feature representations when passed through CNNs. In addition to the vision-language models, I also show this to be true with DINOv2 [9] and Stable Diffusion 3.5 [4], which are pre-trained in a more general-purpose fashion and exhibit much higher quality features as a result.

Can we create more such images automatically? Given the differences in how humans and computers see and process visual illusions, I think that state-of-the-art image generation models lack the ability to understand what truly makes an image illusory for humans. As a result, I do not believe that they are able to independently generate illusions of most varieties. I show this to be mostly true in practice, save for some exceptions. Additionally, past work [15] has shown that models trained solely on human visual illusions fail to generate anything of value either. However, with recent advancements in image generation, future work might be able to finally achieve such a feat given the right data and priors.

The majority of state-of-the-art image generation models utilize latent diffusion model architectures. Similar to feature space, I also show that illusions and illusions-illusions are close in latent space. Though this is the case, I found that generating new images from nearby latents produce non-illusory results.

2. Related Work

The reasoning behind my main hypotheses draw from prior work in neuroscience and machine learning, specifically with regards to how humans and computers visually process and understand illusions (and illusions-illusions).

Illusions Illusions have proven to be a powerful tool in neuroscience at helping researchers understand how human perception operates at a higher level. Chapter 3 of *Biological and Computer Vision* [6] summarizes many key aspects of human perception within the context of many different visual illusions. One major point is that there exists a disassociation between what the eyes see and what the brain perceives; the brain fills in much of what it perceives based off past information, biological priors, etc. Another noteworthy point is that spatial context heavily dictates what the brain actually perceives (the Checker Shadow illusion, Fig. 1, illustrates this phenomenon well). Many of the brain’s biological and psychological attributes that allow humans to be susceptible to various illusions, I believe, are lacking in modern computer vision systems. This is precisely why I do not think that vision models have the capacity to be susceptible to many human visual illusions.

At the intersection of neuroscience and machine learning, prior work [8] has taken to comparing the responses of the human brain and of RNNs on several types of illusions. Specifically, they evaluate using a pre-trained PredNet model [7] using illusory contours and flash-lag illusions. In the illusory contours experiments, a monkey and the PredNet are shown a sequence of images derived from Kanizsa figures. Their results show that certain neurons in the monkey’s brain and in the PredNet exhibit similar response patterns to the various images along the sequence. For the flash-lag experiment, the PredNet is shown a video of a rotating line, where a second line flashes for one frame, and is tasked with predicting each subsequent frame. At the point where the second line flashes, the PredNet predicts the second line to radially lag behind the first in the following frame. This is similar to how humans perceive this flash-lag effect as well. While this work demonstrates evidence that RNNs exhibit similar behavior to brains on two given types of illusions, I do not believe this necessarily equates

to modern computer vision systems being fully susceptible to illusions of most other varieties.

Several previous works [10, 15] have taken to creating datasets containing broad sets of human visual illusions. The older of the two [15] takes to training an image classifier and a generative model on their dataset. Their classifier, which was fine-tuned to classify the type of illusion from a set of possible categories, achieves above-chance accuracy on all categories, but is mediocre (< 50%) on most. Additionally, they train a GAN [5] on a subset of the dataset to generate new optical illusions. Unfortunately, the model had completely failed to generate anything of value.

The more recent of the two [10] instead takes to evaluating the visual comprehension of current vision-language models. Given some optical illusion, models were questioned on their understanding of the image and were given several multiple-choice answers to choose from. These experiments were conducted in both zero-shot and few-shot learning scenarios. Though most models achieved above-chance accuracy various types of illusions, the accuracies were small enough for the authors to conclude that current state-of-the-art models struggle to properly understand optical illusions.

Illusions-illusions Recent work [13] has introduced the idea of illusions-illusions as a means of evaluating the optical perception of current vision-language models. In this evaluation, models were prompted on illusions, illusions-illusions, and control images. In this case, the control images are obtained by taking illusions-illusions and removing any part of the image that remotely resembles the original illusion (see Fig. 1 for an example). The control images were not intended to fool the models—they were merely intended to ensure that the models can perform basic optical reasoning on the simplest of examples. Nonetheless, there were many cases where models had still confused control images for their corresponding illusions. Perhaps a bit less surprisingly, the models also performed poorly on illusions-illusions, often mistaking them for their illusory counterparts.

This is the key work I focus on throughout my project. Particularly, my aim is to extend this illusion-illusion evaluation in order to provide a better understanding of how vision models perceive illusions-illusions, primarily in hopes of providing explanation for the author’s findings.

3. Methods

My proposed experiments fall under one of two main categories: feature comparison and illusion generation. Given the high-dimensional nature of neural image features, I employ spectral clustering methods [11, 16, 17] in order to properly visualize them in an RGB space, which allows us

to qualitatively compare features. To quantitatively compare these features in a position-agnostic way, we compute a nearest-neighbor cosine similarity metric. As for attempting to generate new illusions given existing ones, we utilize recent latent interpolation techniques [14].

Algorithm 1 Nearest-neighbor cosine similarity

```

 $F_1, F_2 : \text{Mat}(n, m)$                                 ▷ Input
 $S_1, S_2 \leftarrow \text{Zeros}(n)$ 
for  $i \in \{1, \dots, n\}$  do
     $s_1 \leftarrow \text{CosineSimilarity}(F_1[i], F_2)$ 
     $S_1[i] \leftarrow \text{Max}(s_1)$ 
     $s_2 \leftarrow \text{CosineSimilarity}(F_2[i], F_1)$ 
     $S_2[i] \leftarrow \text{Max}(s_2)$ 
end for
return  $\frac{\text{Mean}(S_1) + \text{Mean}(S_2)}{2}$                                 ▷ Output

```

3.1. Spectral clustering

Spectral clustering is a graph-based method for clustering data in a lower-dimensional space spanned by eigenvectors, and is a long-studied and well-established problem within spectral graph theory. For several decades now, spectral clustering has also been used for image segmentation and perceptual grouping [11], and more recently, has been extended to visualizing extremely high-dimensional neural image features in lower-dimensional spaces (eg. RGB) [16, 17].

Let $F \in \mathbb{R}^{n \times m}$ be a matrix representing a set of n feature vectors, each one being m -dimensional. We first compute the adjacency matrix $W \in \mathbb{R}^{n \times n}$, where each entry W_{ij} is the cosine similarity between the i -th and j -th feature vectors, with

$$W = \frac{FF^T}{(\text{diag}(FF^T)^{1/2})(\text{diag}(FF^T)^{1/2})^T}.$$

We then compute the degree matrix $D \in \mathbb{R}^{n \times n}$, which is a diagonal matrix where each entry D_{ii} is the sum of cosine similarities of the i -th feature vector with all other features, with $D_{ii} = \sum_{j=0}^n W_{ij}$. The unnormalized graph Laplacian matrix is then given by $L = D - W$.

Normalized cuts is a method used to partition a graph into disjoint subsets, with the goal being to minimize the sum of edge weights between the subsets relative to the sum of edge weights within each subset. One can show that relaxing this normalized cuts problem given our feature graph is equivalent to finding the eigenvalues and eigenvectors of the normalized graph Laplacian matrix, $D^{-1/2}LD^{-1/2}$.

Assume that the eigenvectors $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$ are sorted from smallest to largest corresponding eigenvalues. The second eigenvector \mathbf{x}_2 (called the Fiedler vector) provides

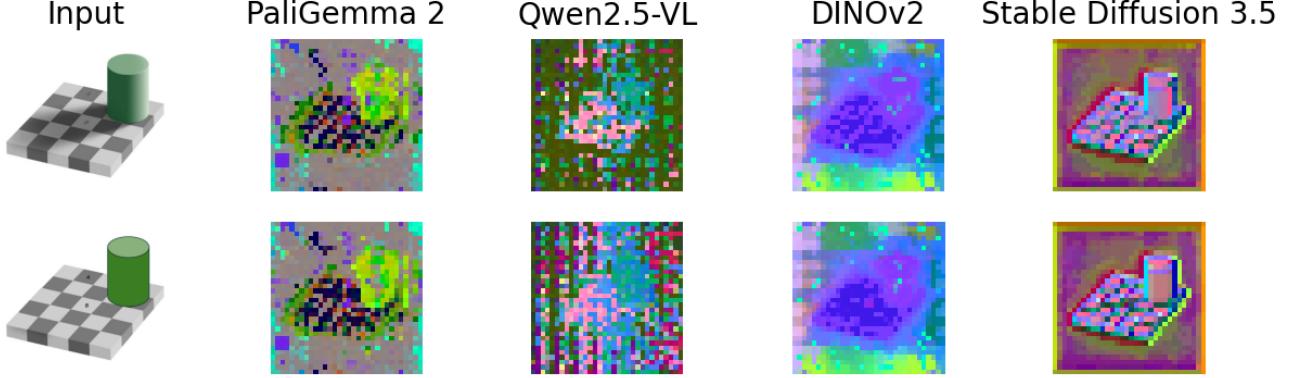


Figure 2. Spectrally clustered features of the Checker Shadow illusion and illusion-illusion given the PaliGemma 2 [12], Qwen2.5-VL [1], DINOv2 [9], and Stable Diffusion 3.5 [4] models. See Fig. 6 for features across the full dataset.

Model		Muller	Grid	Duck	Tichener	Checker	Shepard	Kanizsa	Cafe	Cornsweet	Confetti	Avg.
PaliGemma 2	✓	0.88	0.84	0.79	0.88	0.90	0.94	0.85	0.81	0.86	0.84	0.86
	✗	0.73	0.72	0.70	0.72	0.72	0.70	0.73	0.73	0.72	0.71	0.72
Qwen2.5-VL	✓	0.87	0.78	0.69	0.85	0.76	0.79	0.83	0.81	0.83	0.81	0.80
	✗	0.66	0.66	0.64	0.67	0.61	0.66	0.69	0.69	0.66	0.63	0.66
DINOv2	✓	0.94	0.95	0.85	0.96	0.95	0.98	0.93	0.90	0.94	0.88	0.93
	✗	0.71	0.68	0.64	0.70	0.65	0.64	0.71	0.73	0.63	0.66	0.68
SD 3.5 (Features)	✓	0.99	0.99	0.97	0.98	0.98	0.97	0.99	0.97	0.99	0.97	0.98
	✗	0.95	0.92	0.94	0.95	0.94	0.94	0.96	0.94	0.94	0.86	0.93
SD 3.5 (Latents)	✓	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.97	0.99
	✗	0.95	0.94	0.97	0.97	0.95	0.96	0.97	0.97	0.95	0.89	0.95

Table 1. Nearest-neighbor cosine similarity between features (and latents) of illusions and illusions-illusions given the PaliGemma 2 [12], Qwen2.5-VL [1], DINOv2 [9], and Stable Diffusion 3.5 [4] models. A higher score corresponds to more similar features. In each row, ✓ denotes that the similarity was computed between each illusion and its corresponding illusion-illusion, whereas ✗ denotes that the similarity was computed and averaged between each illusion and every other illusion.

a real-valued solution to this relaxed normalized cuts problem, and describes how this graph should be optimally divided into two clusters. Moreover, the rest of the eigenvectors $\mathbf{x}_3, \mathbf{x}_4, \dots, \mathbf{x}_n$ describe how to further hierarchically subdivided these clusters.

To visualize the first k eigenvectors $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_k$ ($k \ll n$ in practice) in RGB space, t-distributed stochastic neighbor embeddings [2] are used to assign RGB values at each pixel.

3.2. Nearest-neighbor cosine similarity

Similar to how spectral clustering provides a way to qualitatively assess cosine similarity between image features, we would like an effective way to quantitatively assess feature similarity as well. Performing cosine similarity directly between the same pixel location in two different images can be problematic, however. While we expect corresponding illusions and illusions-illusions to exhibit similar features, we do not always expect them to be in the same spacial locations, as in the Muller-Lyer illusion (Fig. 5), for example.

To compute cosine similarity in a position-agnostic fash-

ion, I employ a nearest-neighbor search over all possible pairs of features between the two images. More specifically, for every feature in the first image, we store the maximum cosine similarity between it and all other features in the second image. After repeating the same process for the second image, our final nearest-neighbor cosine similarity score is the average of all the stored maximum values. See Algorithm 1 for a pseudocode implementation.

3.3. Latent interpolation

Interpolating between two images—that is, creating an image that looks like some sort of reasonable combination of the two—is a non-trivial trivial task. Unfortunately, simply interpolating the images in pixel space is likely to result in visual garbage. However, with the rise of latent diffusion models for image generation [4], latent interpolation [14] aims to solve this problem by interpolating in a model’s latent space instead.

Given two images, we first encode them into latent space and add shared noise to their latents. We then interpolate

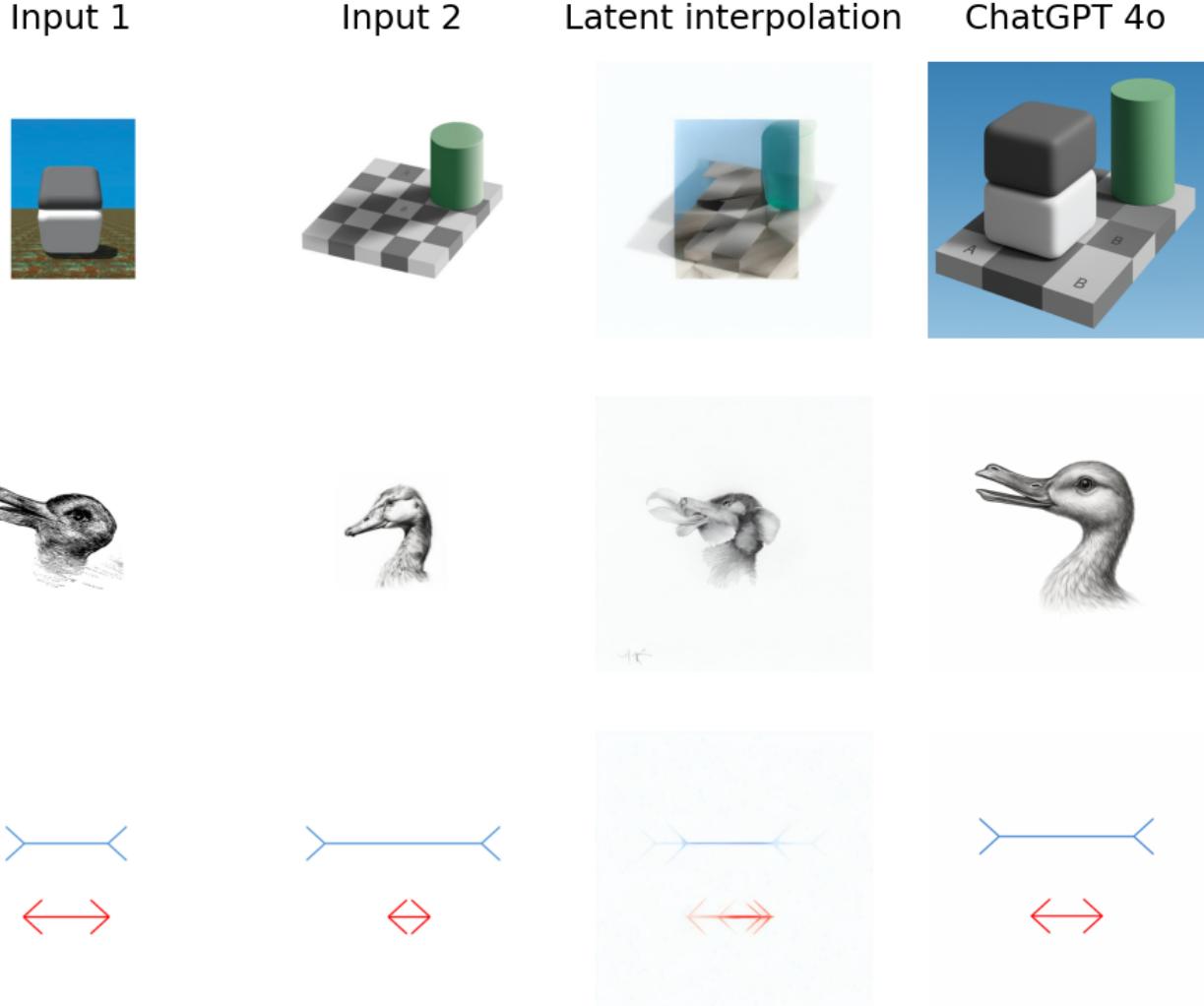


Figure 3. Illusion generation results given two illusions/illusions-illusions using latent interpolation [14] and ChatGPT 4o.

the noisy latents using spherical linear interpolation, which computes the midpoint along the shortest path on a sphere. The interpolated latent is then denoised like usual with the diffusion model, and the denoised latent is finally decoded back into pixel space.

The resulting image should, in theory, look like some reasonable combination between the two inputs. This approach, however, is known to fail when the two inputs deviate too much in terms of style, semantics, or layout.

4. Results

In this section, I introduce and discuss my experimental results using the methods discussed in the previous one. The code used for all my results can be found at https://github.com/zachsalehe/neuro240_illusionsillusions.

4.1. Feature comparison

I began by computing the image features across the full dataset (Fig. 5) from the work that first proposed illusions-illusions [13]. This dataset features ten famous optical illusions, each one varying in the kinds of tricks they are meant to play on the human brain. For each illusion, the dataset features the illusion itself, its corresponding illusion-illusion, and a control. For this experiment, I only consider the illusions and illusions-illusions.

Features were computed across four recent vision models: PaliGemma 2 [12], Qwen2.5-VL [1], DINOv2 [9], and Stable Diffusion 3.5 [4]. PaliGemma and Qwen are vision-language models, much like those that were tested in the original illusions-illusions work. Unlike those from the work (ChatGPT, Gemini, etc.), PaliGemma and Qwen are open source, which allows for us to actually access and

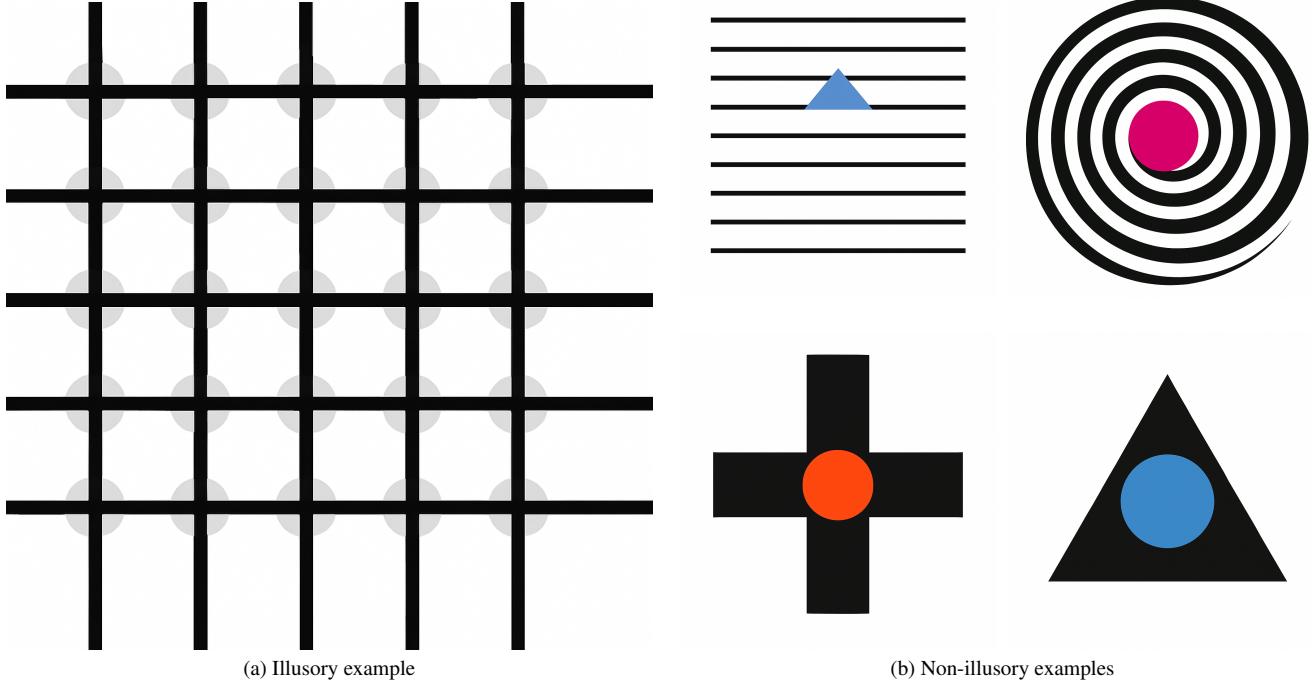


Figure 4. Illusion generation given the prompt “Can you generate me an illusion?” using ChatGPT 4o. Out of dozens of images, only one illusory example (a) was produced. This example has a similar effect to the Grid illusion (Fig. 5), where small gray squares appear where there are not any. The others (b) resemble something illusion-like, but are non-illusory.

extract the image features. DINO is a pretrained general-purpose feature extractor, which can be finetuned and applied to a wide variety of vision tasks, whereas, Stable Diffusion is an image generation model, and is the newest of the four. Like most modern computer vision models, these four also utilize vision transformer backbones [3].

Fig. 2 shows the spectrally clustered features (top $k = 20$ eigenvectors) of the Checker Shadow illusion across the four models (see Fig. 6 for full results). One thing that immediately stands out is the difference in feature quality between the vision-language models and the stand-alone vision models. I believe this phenomenon is likely a result of differences in their training objectives. PaliGemma and Qwen’s vision encoders were, at a high level, trained for aligning image embeddings with text embeddings. As a result, they do not necessarily learn rich and dense features, as much of the fine-grained details become irrelevant. On the other hand, DINO is trained specifically to produce rich and dense features, and Stable Diffusion is naturally learns very detailed features in order to generate high-quality images.

Despite their differences in feature quality, similar trends can be seen across the board; illusions and illusions-illusions map to very similar feature representations. Across most of the data pairs, the differences in image features are quite subtle, and sometimes even unnoticeable. Due to the higher quality of their features, it is no sur-

prise that these differences are more distinguishable with DINO and Stable Diffusion. However, given that the original inspiration for my work came from the fact that vision-language models, like PaliGemma and Qwen, cannot differentiate between illusions and illusions-illusions, the spectral clustering results strongly support my claim that this occurs due to similarities in feature space.

Tab. 1 shows the nearest-neighbor cosine similarity results across the full dataset. Additionally, metrics on Stable Diffusion’s latents are shown as well. Metrics were computed once between each illusion and its corresponding illusion-illusion, and again averaged between each illusion and every other illusion in the dataset. The later is intended to serve as a baseline. Comparing the results between the two cases, we see once again that illusions and illusions-illusions are very close to one another in feature space, further supporting my claim that this is the cause for vision-language models’ confusions.

4.1.1. Implementation details

I used Hugging Face to load each of the four models. I wrote the code to load each model and extract their features, which can be found in `compute_features.ipynb`. To spectrally cluster the features, I utilized the `ncut-pytorch` [16] Python package. I wrote the code to spectrally cluster the features and perform nearest-

neighbor cosine similarity on them, which can be found in `process_features.ipynb`. Additionally, I wrote the code to extract Stable Diffusion’s latents and perform nearest-neighbor cosine similarity on them, which can be found in `process_latents.ipynb`.

4.2. Illusion generation

Given this newfound knowledge that illusions and illusions-illusions are close in feature (and latent) space, I wanted to take advantage of this fact in attempting to generate new illusions given existing ones. Fig. 3 shows some results of generating new illusions given two illusions/illusions-illusions. At first, I had done so with latent interpolation [14]. The results were quite poor, in general, with the generated images not looking very reasonable at all. While latent interpolation is known to fail when the inputs differ in terms of style, semantics, or layout, I nevertheless expected slightly better results given that the inputs (at least in the second and third rows) do not deviate much. In desperation, I turned to ChatGPT 4o’s new image generation model to see if it can do any better. To my surprise, it generated very reasonable looking images when given the two inputs. While nice-looking, there was nothing inherently illusory about ChatGPT’s results.

Given my surprise with the quality of ChatGPT’s results, I wanted to see it could generate anything of value simply given the prompt “Can you generate me an illusion?” I had originally performed this experiment at the start of the term when I was first beginning this project, and had found the results to be utterly terrible. However, given its new-and-improved image generation model, the images produced (Fig. 4) were quite alright. Though, after generating dozens of images, only one of them turned out to be illusory. While the non-illusory examples are high-quality images, these led me to believe ChatGPT does not really know what makes an image illusory. The illusory example begs to differ, though. Given the rarity of this event, however, I am inclined to believe that the model may have gotten lucky, or was simply mimicking the grid-like patterns seen in the Grid illusion (Fig. 5), of which it has surely seen as part of its training dataset.

4.2.1. Implementation details

To perform latent interpolation, I used the codebase from their paper [14], which can be found in `controlnet.zip`. The short snippet I wrote to interact with this codebase can be found in `run_controlnet.ipynb`.

5. Discussion

In this work, I have shown that illusions and illusions-illusions are very close in several different models’ feature space. This, I hope, provides a concrete explanation and

follow-up to the original findings [13] that vision-language models struggle to differentiate between the two. There remains more work to be done in this regard, however. Namely, it seems likely that these vision-language models are performing a sort of nearest-neighbor search when prompted about illusions-illusions, and given that their features are near those to their corresponding illusions, just assumes that the two are equivalent. Perhaps this nearest-neighbor approach is not so practical after all, and future work could explore better ways of responding to such prompts. Additionally, future work could explore incorporating better image encoders into vision-language models. Though I found the features to be similar across all models I tested, the low quality of the vision-language models’ features became apparent, and such models could potentially benefit from more rich and dense feature representations, which might allow them to more easily differentiate between illusions and illusions-illusions.

In addition, this work also explores the possibility of generating illusions. Attempting to generate illusions given existing illusions/illusions-illusions showed little promise. Though I do not quite believe that ChatGPT fully understands what makes an image illusory, the single illusory example (Fig. 4) that ChatGPT had generated given the prompt “Can you generate me an illusion?” gives me a little hope. While previous work has attempted to train a GAN solely on illusions [15], which failed to generate anything of value, recent advancements in latent diffusion models might make this task more feasible. With a large and diverse dataset, and perhaps some illusion-specific priors, future work may succeed in creating a consistent illusion generation model.

References

- [1] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhao-hai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. [1](#), [4](#), [5](#)
- [2] T. Tony Cai and Rong Ma. Theoretical foundations of t-sne for visualizing high-dimensional clustered data, 2022. [4](#)
- [3] 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, 2021. [6](#)
- [4] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, Kyle Lacey, Alex Goodwin, Yan-nik Marek, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis, 2024. [2](#), [4](#), [5](#)

- [5] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. [3](#)
- [6] Gabriel Kreiman. *Biological and Computer Vision*. Cambridge University Press, 2021. [1](#), [2](#)
- [7] William Lotter, Gabriel Kreiman, and David Cox. Deep predictive coding networks for video prediction and unsupervised learning, 2017. [2](#)
- [8] William Lotter, Gabriel Kreiman, and David Cox. A neural network trained for prediction mimics diverse features of biological neurons and perception. *Nature machine intelligence*, 2(4):210–219, 2020. [2](#)
- [9] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024. [1](#), [2](#), [4](#), [5](#)
- [10] Haz Sameen Shahgir, Khondker Salman Sayeed, Abhik Bhattacharjee, Wasi Uddin Ahmad, Yue Dong, and Rifat Shahriyar. Illusionvqa: A challenging optical illusion dataset for vision language models, 2024. [3](#)
- [11] Jianbo Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000. [3](#)
- [12] Andreas Steiner, André Susano Pinto, Michael Tschannen, Daniel Keysers, Xiao Wang, Yonatan Bitton, Alexey Gritsenko, Matthias Minderer, Anthony Sherbondy, Shangbang Long, Siyang Qin, Reeve Ingle, Emanuele Bugliarello, Sahar Kazemzadeh, Thomas Mesnard, Ibrahim Alabdulmohsin, Lucas Beyer, and Xiaohua Zhai. Paligemma 2: A family of versatile vlms for transfer, 2024. [1](#), [4](#), [5](#)
- [13] Tomer Ullman. The illusion-illusion: Vision language models see illusions where there are none, 2024. [1](#), [3](#), [5](#), [7](#), [9](#)
- [14] Clinton J. Wang and Polina Golland. Interpolating between images with diffusion models, 2023. [3](#), [4](#), [5](#), [7](#)
- [15] Robert Max Williams and Roman V. Yampolskiy. Optical illusions images dataset, 2018. [2](#), [3](#), [7](#)
- [16] Huzheng Yang, James Gee, and Jianbo Shi. Alignedcut: Visual concepts discovery on brain-guided universal feature space, 2024. [3](#), [6](#)
- [17] Xiao Zhang, David Yunis, and Michael Maire. Deciphering ‘what’ and ‘where’ visual pathways from spectral clustering of layer-distributed neural representations, 2024. [3](#)

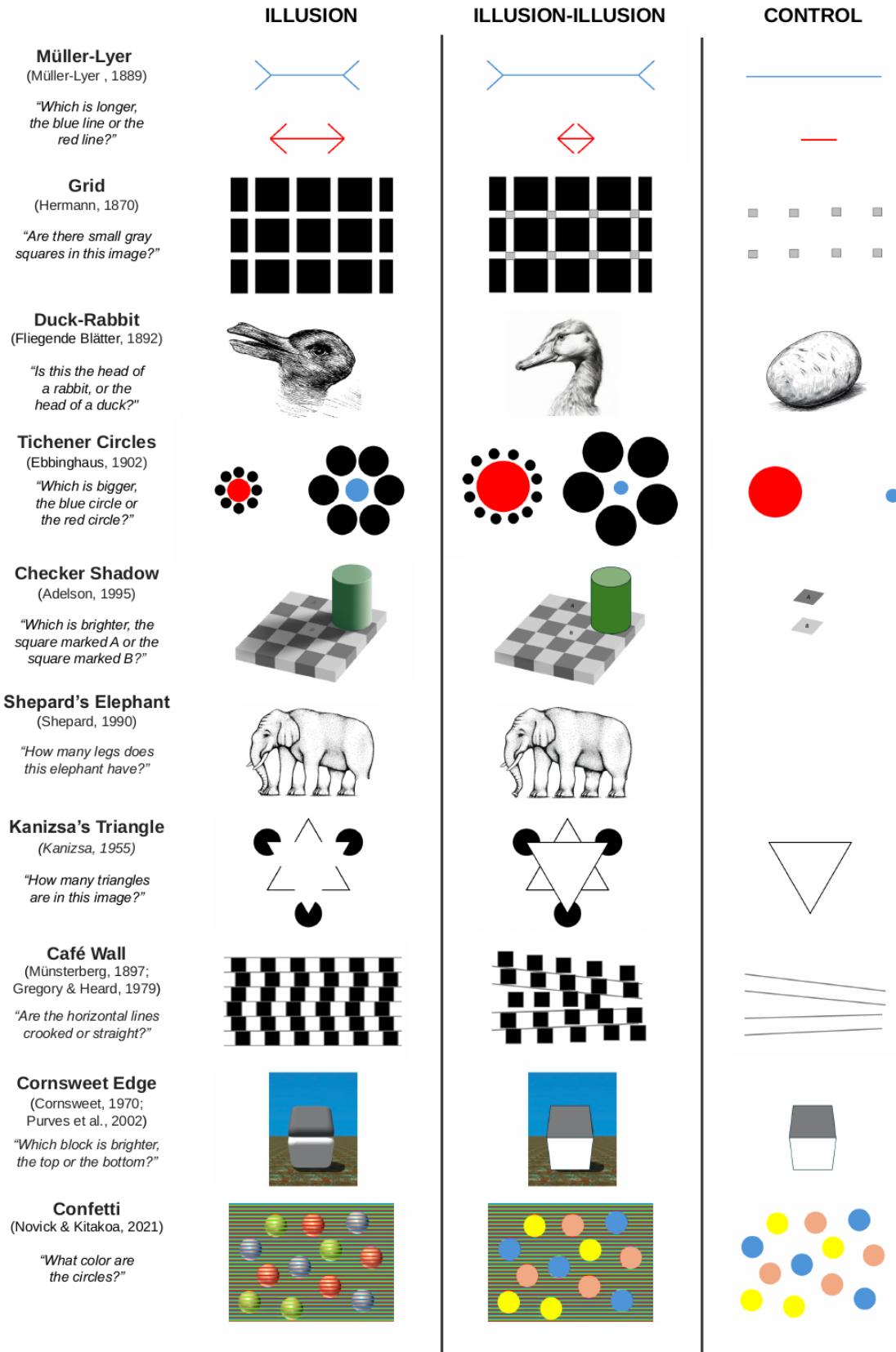


Figure 5. Full dataset from the original work on illusions-illusions [13].

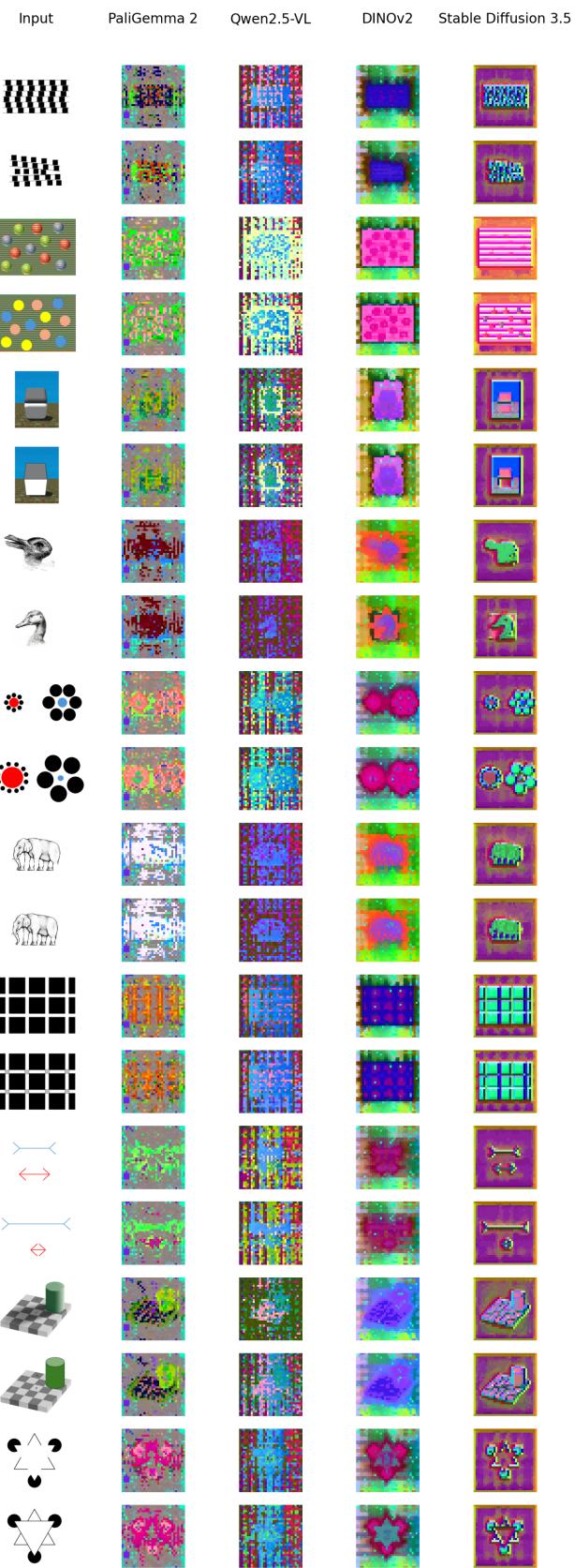
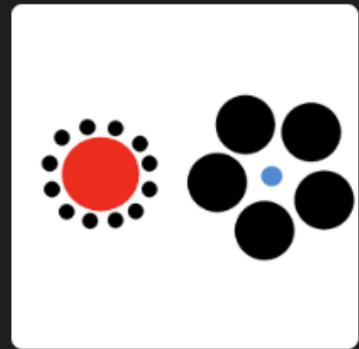


Figure 6. Feature comparison across the full dataset.



is the red or blue circle bigger



Although the red circle on the left **appears** larger than the blue circle on the right, they are actually **the same size**. This is an example of the **Ebbinghaus illusion**, where the perceived size of a central circle is influenced by the size and spacing of surrounding circles.

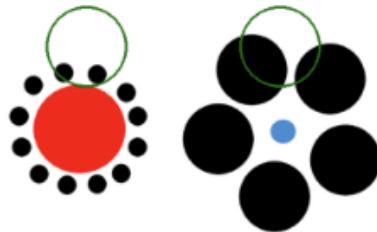
Would you like me to overlay the circles to prove they're identical?

yes

Overlay To Show Circles Are The Same Size

↓ ↵

Overlay to Show Circles Are the Same Size



As shown in the overlay above, both the red and blue circles are outlined with identical green circles—confirming they are **the same size**. The visual illusion is purely due to the context of the surrounding shapes.

Would you like to see how this illusion works with different shapes or arrangements? [.-]



Figure 7. Funny conversation I had with ChatGPT 4o regarding the Tichener Circles illusion-illusion.