The logit lens on vision transformers for classification and infilling

Vicki Xu

December 15, 2023

1 Introduction

In recent years, the interpretability of machine learning (ML) models has become an increasingly hot field of study. Though ML models have historically been seen as "black boxes," as they have taken on increasingly important social tasks, there has been growing interest in understanding how such models ingest and process data.[16]

In particular, the transformer architecture, which was first developed for natural language processing tasks, has found applications in the computer vision field as well. Vision transformers (ViTs) require much fewer computational resources to train compared to state-of-the-art convolutional neural networks, which have made them important models for solving various computational tasks.[7] They are now employed in common tasks such as image segmentation,[12,18] image classification,[7] and object detection.[17] Because the variety and scale of their applications continue to grow, insight into their internal workings is important both for AI alignment and further technical development.

This report focuses on a method of interpreting intermediaries in a computation, the logit lens,[15] which is an early-exiting method that decodes hidden states of a transformer. The logit lens was originally developed for language transformers, but we explore here for ViTs due to the architectural similarity. We apply the logit lens on a base ViT for classification and infilling. We examine at what point the output of the hidden states begin to converge to the final output. We also provide further directions for this line of research.

2 The Transformer

A transformer model, developed first by Vaswani et. al,[19] is characterized by stacked encoder-decoder structures, relying entirely on self-attention instead of recurrent neural networks or convolution. The transformer was first used in language models, for which recurrence or CNNs were often used. A transformer with self-attention can be parallelized significantly more easily, and also requires significantly less training time than previous RNNs.

The encoder of the transformer is comprised of some amount of stacked identical layers, and each layer has two sub-layers, one a self-attention mechanism and one a position-wise fully-connected feed-forward network. The decoder is also comprised of some amount of stacked identical layers, and each layer in the decoder has the same sub-layers as the encoder plus a third sub-layer that performs multi-head attention over the output of the encoder stack.

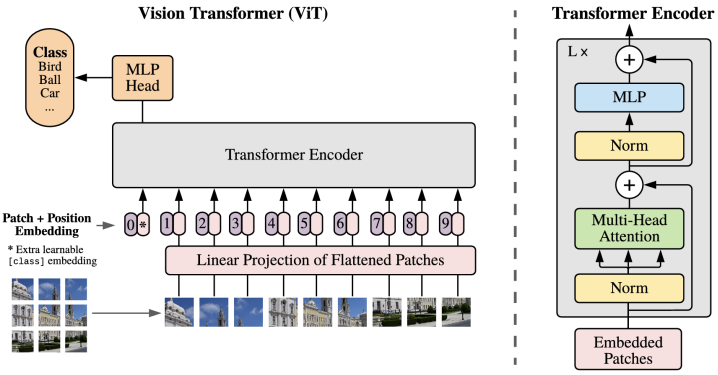


Figure 1: Architecture of a ViT model, as illustrated by Dosovitskiy et. al.[7]

The transformer works by splitting and encoding input text up into tokens (adjacent letters, syllables, or whole words). The tokens are converted to vectors according to a word embedding table and passed through the encoder.

2.1 The Vision Transformer

The base architecture of a vision transformer (ViT), proposed first by Dosovitskiy et. al,[7] is as illustrated in Figure 1. It is basically the same as a language transformer, except it works with images. ViT splits an image into fixed-size patches (analogous to the tokens in a language model), linearly embeds each of them, adds position embeddings, and feeds the vectors to a transformer encoder.

Based on the application, ViTs will have additional modifications on top of this architecture. `ViTForImageClassification` has a CLS token on top of the encoder, and `ViTMAE` for infilling tasks has a decoder.

3 Logit Lens

The "logit lens" for ML models was first proposed by user *nostalgebraist* on rationality blog LessWrong.[15] Applied to GPT-2 and GPT-3, the work was to essentially decode the model's internal state at various layer depths in its computation to see how the model's tokens shift at certain points in time. The reason why this works is that GPT's probabilistic predictions are a linear function of the activations of its final layers, so the results are intuitive to get when we apply the linear function after every layer. As opposed to other interpretability works, which focus on how a model updates its beliefs at each step, the logit lens depicts moreso an interpretation of the model's beliefs after each step.

nostalgebraist's results show that the distributions often converge closely to the final distribution somewhere in the middle of the computation – quite a few layers before the end. Essentially, GPT will start with a guess, but will have formed a fairly good guess by the middle. The rest of the layers are for fine-tuning the guesses.

We implemented a “logit lens" on two ViT models, one for classification and one for infilling. These are two separate but very common applications of ViTs that we felt would yield fruitful insights about how the architecture fares in certain tasks. In particular, infilling lends itself to very visual depictions, which could be useful as a qualitative lens for interpretability. Specifically, we look at the hidden states after each layer of the encoder.

We were interested in seeing whether ViTs would exhibit fundamentally different behavior than language transformers, which would indicate something about the nature of the data that is processed. If ViTs exhibited the same behavior, it would indicate that words and images are somehow analogous in some sense to the transformer architecture, which would be a nice corroboration.

4 Implementation and Results

We used `ViTForImageClassification` found on Huggingface,[10] and the PyTorch implementation of masked autoencoders, `ViTMAE`, found in the Facebook AI models repository.[14] We used the base architectures for both `ViTForImageClassification` and `ViTMAE`, with pretrained weights.

Both architectures had twelve encoder layers. We interpret the hidden state at each layer of the encoder. For `ViTForImageClassification`, we applied the classification vector at each layer of the encoder. For `ViTMAE`, we applied the infilling decoder at each level of the encoder, keeping the mask the same across all layers.

Other settings included a patch-size of 16, an embedding dimension of 768, 12 encoder attention heads, a decoder embed dimension of 512, a decoder depth of 8, 16 decoder attention heads, and an MLP size of 3072. These were the default settings for the base models. The mask ratio for `ViTMAE` is 0.75.

We ran `ViTForImageClassification` and `ViTMAE` on various images in the COCO Dataset.[13] The Colab notebooks for `ViTForImageClassification` and `ViTMAE`, respectively, can be found at the Github repository https://github.com/matchaginseng/fuzzy-palm-tree.

4.1 `ViTForImageClassification` Results

We tried `ViTForImageClassification` on images that (a) very obviously had a particular subject and (b) did not obviously have a singular subject. For (a), we chose a picture with sleeping cats[9] and a picture with a sandwich on a plate.[4] For (b), we chose a picture with soup and a sandwich with a spoon on a plate.[5] The images with their categorizations are depicted in Figure 2.

Additional results can be found in the results folder for classification in the Github repository.

4.2 `ViTMAE` Results

We tried `ViTMAE` on images with brighter colors or more significant contrast, so the model's inferences would be very apparent. We used a model with unnormalized pixels, which is better for downstream fine-tuning, rather than a model with normalized pixels, which is better for visualization.[11]

Figure 3 depicts the images, masks, and their final infilled versions. Figures 4, 5, and 6 depict the intermediaries for each image.

Additional results, including with a lower mask ratio of 0.4, can be found in the results folder for infilling in the Github repository.

```
Predicted class: hyena, hyaena
Predicted class: Band Aid
Predicted class: sunglasses, dark glasses, shades
Predicted class: sunglasses, dark glasses, shades
Predicted class: sunglasses, dark glasses, shades
Predicted class: maillot
Predicted class: velvet
Predicted class: wool, woolen, woollen
Predicted class: velvet
Predicted class: velvet
Predicted class: tabby, tabby cat
Predicted class: tabby, tabby cat
Predicted class: tabby, tabby cat
```

(a) Classification results on cats.

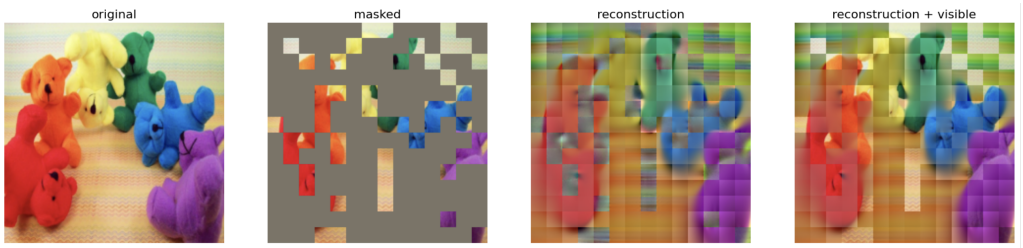
```
Predicted class: hyena, hyaena
Predicted class: plunger, plumber's helper
Predicted class: hammer
Predicted class: cowboy hat, ten-gallon hat
Predicted class: cowboy hat, ten-gallon hat
Predicted class: plate
Predicted class: plate
Predicted class: plate
Predicted class: plate
Predicted class: plate
Predicted class: plate
Predicted class: plate
Predicted class: plate
```

(b) Classification results on plate with sandwiches.

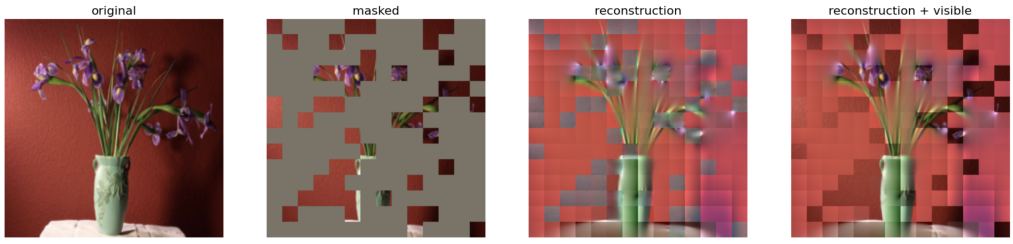
```
Predicted class: hyena, hyaena
Predicted class: buckle
Predicted class: wooden spoon
Predicted class: wooden spoon
Predicted class: wooden spoon
Predicted class: plate
Predicted class: plate
Predicted class: plate
Predicted class: plate
Predicted class: plate
Predicted class: plate
Predicted class: plate
Predicted class: guacamole
```

(c) Classification results on plate with sandwich and soup.

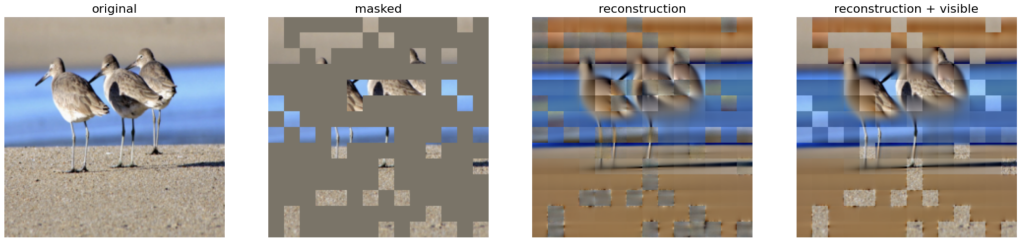
Figure 2: Images used for classification and their results.



(a) Teddy bear image with mask and reconstruction.



(b) Orchids image with mask and reconstruction.



(c) Birds image with mask and reconstruction.

Figure 3: Images used for infilling tasks and their masks and reconstructions: teddy bears,[6] orchids,[3] and birds.[2] The mask ratio is 0.75.

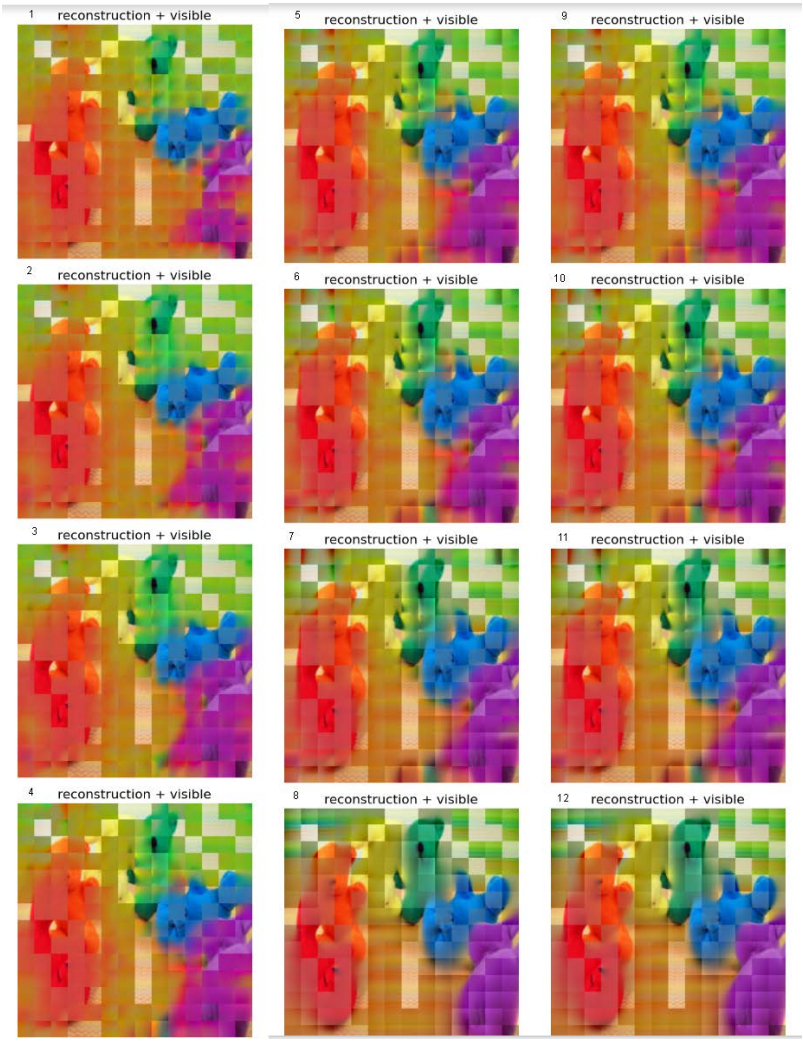


Figure 4: Infilling results for teddy bears, over the course of a model prediction, with mask as depicted in Figure 3. The first layer is the upper left, the fifth layer is the upper middle, the seventh layer is the upper right, and the last layer is the lower right.

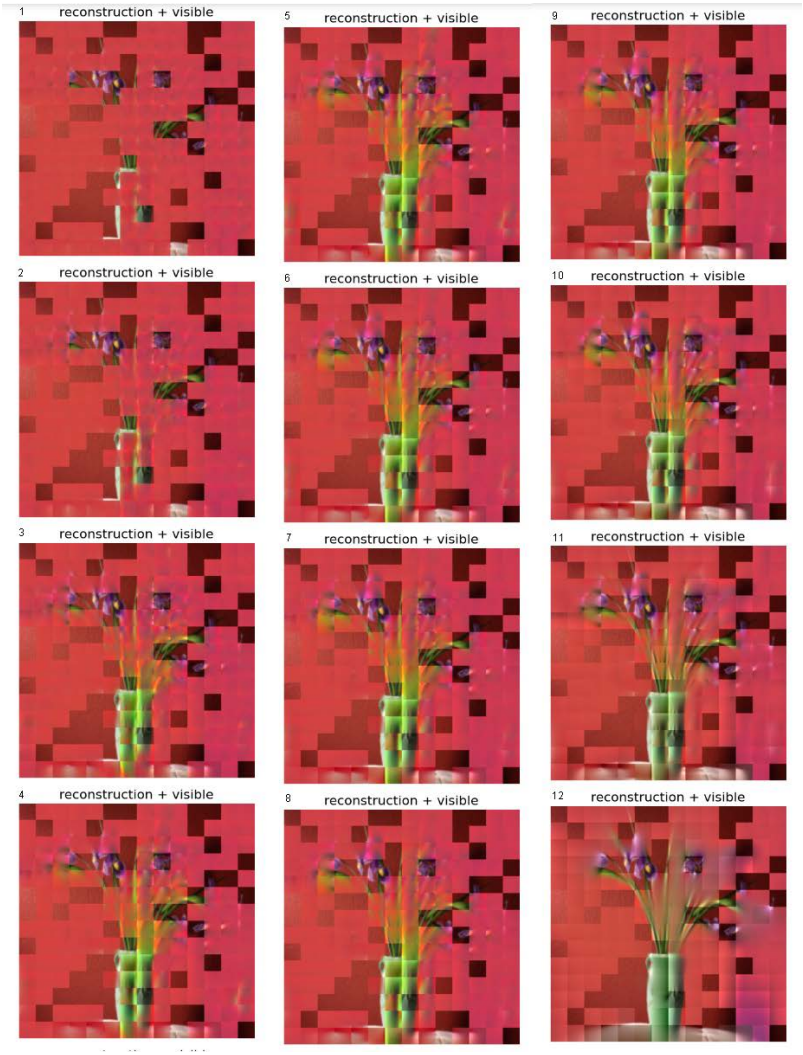


Figure 5: Infilling results for orchids, over the course of a model prediction, with mask as depicted in Figure 3.

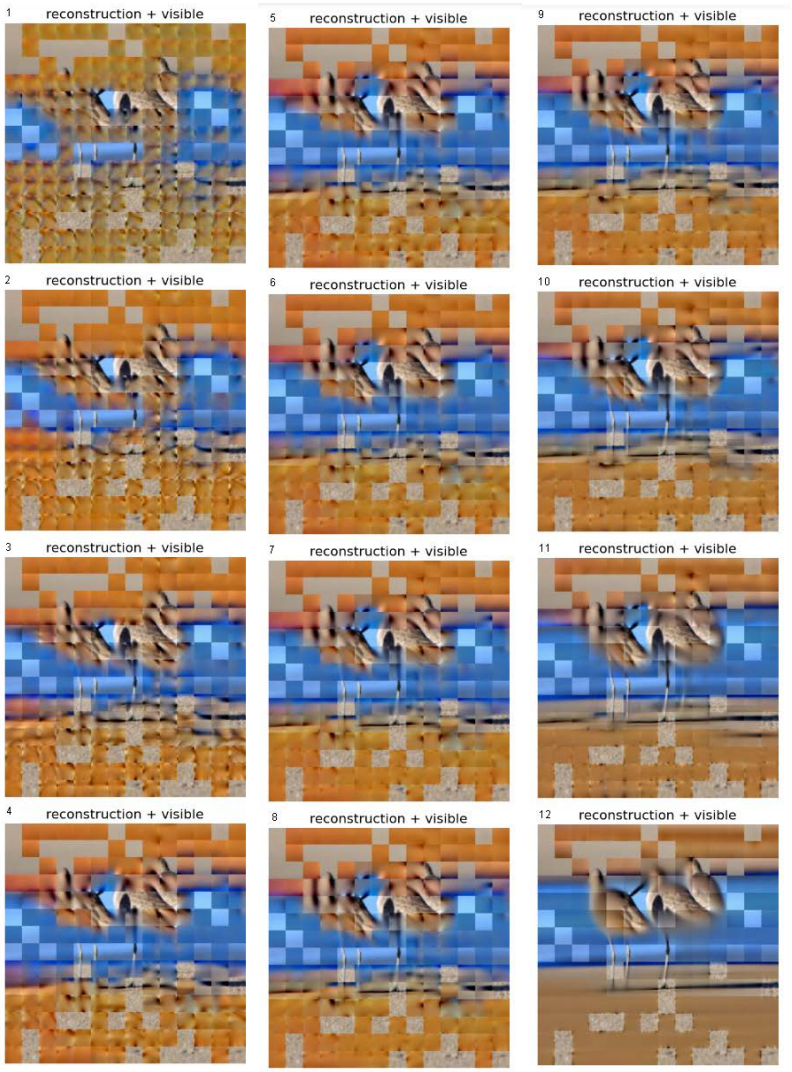


Figure 6: Infilling results for birds, over the course of a model prediction, with mask as depicted in Figure 3.

5 Discussion

5.1 `ViTForImageClassification` Discussion

The first layer is always the initialization of the weights, which interprets out to be `hyena`. In the following layers, the model sharpens its classifications. We see a similar early convergence pattern as with the language models with these images we tested, wherein the model comes to a conclusion midway through that seems close to the final prediction – with some exceptions.

With the cats, `ViTForImageClassification` performed as expected. The "band-aid" prediction on the second layer could be due to the shapes of the cats, or the remote controllers near the cats. We were unable to explain the "maillot" prediction, but the model also seems to notice texture with its predictions of "velvet" or "woolen", and is potentially paying attention to either the cats' furriness or the background (a sherpa blanket, but also cloth material). The model converges onto "tabby cat" as we would hope.

`ViTForImageClassification` surprised us with the sandwich. Because the sandwich was such a prominent feature of the image (and had been categorized in COCO as a "sandwich" image), we thought the model would conclusively determine the sandwich. However, none of the layers predicted "sandwich." The second layer perhaps found a plunger's shape in the shape of each sandwich, with the head of the plunger being the sandwich and the handle of the plunger being the toothpick. The handle-head form could also be related to the model's prediction of hammer. The cowboy hat prediction for the fourth and fifth layers could broadly be from the shape the toothpicks and the sandwich mark out (the crown of the hat) with the plate (the brim of the hat). Finally, it seems that the model predicted the plate in the final layers with the greatest confidence.

With the soup and sandwich image, `ViTForImageClassification` also gave interesting results. We were surprised to see the second layer predict "buckle." For the third to fifth layers, though the picture depicted a spoon, it was a metal one, not a wooden one; however, we thought perhaps the material was related to the relative predominance of brown throughout the rest of the image. The model then converges on the prediction of "plate" for a couple of layers, before finally – and unexpectedly – shifting to "guacamole," which we thought could be because of the presence of the avocado slices on the sandwich, and the circular soup bowl (since guacamole is often served in a bowl).

For food images, the model seemed to predominantly interested in the plate, even though a human might point out the soup and the sandwich first – indicating perhaps what humans notice about an image could also be socially conditioned, versus what the model might. It seems that each layer may also focus on different parts of the image, the result being that the model ends up with a rather counterintuitive prediction (like "cowboy hat" for a plate of sandwiches). With busier images in general, the model seems to look around the image throughout the layers before coming up with a guess. However, for some images, it seems to come up with a "big-picture" guess (like guessing "plate" for the sandwich image), while for others, it seems to zone in on a particular detail of the image (such as "guacamole" for the soup and sandwich image).

5.2 `ViTMAE` Discussion

We tested this with a couple of higher-contrast images. The first hidden state seems mostly to be color-patching. By the middle layer, for all pictures, the images will have converged to close to the final output. With the orchids, for example, by layer three the model had already come up with

a general sense that there should be a green stem connecting the purple flowers with the greenish vase.

This seems relatively intuitive given what we know about the logit lens applied to GPTs. Infilling results are a bit more challenging to interpret because though they lend themselves visually, we discovered that the blur in the reconstructed patches could make it difficult to determine more precisely around where the model starts getting more accurate. For future work, elaborated on in Section 6.1, imposing a quantitative metric on the infilling progress across layers could be useful.

6 Conclusion

In this report, we examined the logit lens method applied to vision transformers for classification and infilling. We used the base ViT transformer in both cases. We found that, similar to the logit lens applied to GPT, the model seems to converge on a relatively good guess in the middle of its layers and spend the rest of its layers fine-tuning – particularly for the infilling model. We offer a starting point for specific details about *what* the model notices between the layers.

6.1 Future Work

Future work takes a variety of directions.

Working with more complex ViTs, such as ViT-Large or ViT-Huge, with their additional encoder layers, may yield more specific or detailed results.

Coming up with a more quantitative way to evaluate the "closeness" of the intermediary layers to both the final distribution and to ground truth would also be informative. We evaluated this closeness quite qualitatively, as is apparent from our analysis. However, perhaps implementing a word association score on top of the classification predictions, or some kind of intersection-over-union metric for the infilling predictions, would give a more quantitative sense of each layer's "accuracy."

For the infilling model, a more comprehensive investigation of the influence of mask ratios on the output could also be interesting. Perhaps depending on the mask ratio, the model's intermediary layers behave differently.

Another direction is implementing the tuned lens on ViTs.[1] The tuned lens was previously applied to language models as a refinement of the logit lens, because the logit lens could be unreliable for certain GPT models due to representational drift. Applying a version of the tuned lens could produce more informative predictions – or could at least provide an interesting point of comparison.

Further on, we would be interested in investigating symmetry in attention heads in vision transformers. Combined with prior work on attention heads in vision transformers,[20] the logit lens on ViTs provides some foundational research in that direction. In particular, integrating a logit lens with transformer attention visualization is a great further direction. Knowing where the attention "is" at a certain point in a ViT, and then decoding the ViT's hidden state at that time, and seeing if the two seem to correspond rationally, could provide useful insights.

7 Acknowledgements

This report was written for the completion of CS299R: Reading and Research in Fall 2023. Thank you to Martin Wattenberg, Fernanda Viégas, and Catherine Yeh for your continued advising and

support.

References

[1] Nora Belrose, Zach Furman, Logan Smith, Danny Halawi, Igor Ostrovsky, Lev McKinney, Stella Biderman, and Jacob Steinhardt. Eliciting latent predictions from transformers with the tuned lens, 2023.

[2] COCO Dataset. Birds. [Online; accessed December 9, 2023].

[3] COCO Dataset. Orchids. [Online; accessed December 9, 2023].

[4] COCO Dataset. Sandwich on plate. [Online; accessed December 9, 2023].

[5] COCO Dataset. Soup and sandwich on plate. [Online; accessed December 9, 2023].

[6] COCO Dataset. Teddy bears. [Online; accessed December 9, 2023].

[7] 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. In *International Conference on Learning Representations*, 2021.

[8] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. 2021.

[9] Huggingface. Cats. [Online; accessed December 9, 2023].

[10] Huggingface. ViTForImageClassification. https://huggingface.co/docs/transformers/v4.35.2/en/model_doc/vit#transformers.ViTForImageClassification. Online; accessed December 9, 2023.

[11] Kaiminghe. mae_visualize models vs mae_pretrain_full models. https://github.com/facebookresearch/mae/issues/12. Online; accessed December 9, 2023.

[12] Xiangtai Li, Henghui Ding, Wenwei Zhang, Haobo Yuan, Jiangmiao Pang, Guangliang Cheng, Kai Chen, Ziwei Liu, and Chen Change Loy. Transformer-based visual segmentation: A survey, 2023.

[13] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015.

[14] Meta Research. mae. https://github.com/facebookresearch/mae. Online; accessed December 9, 2023.

[15] nostalgebraist. interpreting GPT: the logit lens. https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens. Online; accessed December 9, 2023.

[16] Cynthia Rudin, Chaofan Chen, Zhi Chen, Haiyang Huang, Lesia Semenova, and Chudi Zhong. Interpretable machine learning: Fundamental principles and 10 grand challenges, 2021.

[17] Tahira Shehzadi, Khurram Azeem Hashmi, Didier Stricker, and Muhammad Zeshan Afzal. Object detection with transformers: A review, 2023.

[18] Hans Thisanke, Chamli Deshan, Kavindu Chamith, Sachith Seneviratne, Rajith Vidanaarachchi, and Damayanthi Herath. Semantic segmentation using vision transformers: A survey, 2023.

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[20] Catherine Yeh, Yida Chen, Aoyu Wu, Cynthia Chen, Fernanda Viégas, and Martin Wattenberg. Attentionviz: A global view of transformer attention, 2023.