

Evaluating CNN* Architectures and Training Paradigms for Visual Commonsense Reasoning Using the SVRT Dataset

Jasmine Bilir

jbilir@stanford.edu

Megan Dass

mdass9@stanford.edu

Riya Dulepet

riyadule@stanford.edu

Abstract

In this paper, we explore the domain of Visual Commonsense Reasoning (VCR) within Visual Question Answering (VQA) systems by leveraging and evaluating several prominent computer vision architectures and training paradigms. Utilizing the Synthetic Visual Reasoning Test (SVRT) dataset, which comprises various classification tasks designed to test visual reasoning capabilities, we implement and compare multiple approaches: a baseline Convolutional Neural Network (CNN), a hybrid CNN-Transformer model, contrastive learning techniques, and advanced feature extraction methods. Our goal is to ascertain which methodologies best enhance the model’s ability to interpret and respond to complex visual inputs in a manner akin to human cognitive processes. Preliminary results suggest varying levels of effectiveness across different models, but emphasize the value of CNN and feature extraction in the development of more intuitive and accurate VQA systems. This work not only benchmarks current strategies but also contributes to the ongoing discussion about the optimal integration of architectural innovations and learning paradigms in the field of computer vision.

1. Introduction

Visual question answering (VQA) encompasses the ability to distinguish and group relationally similar objects and images. For humans, visual reasoning and question answering is a critical part of general intelligence and fast reaction time, allowing us to identify, remember, and communicate about the differences and similarities in our environment. For Computer Vision, achieving state-of-the-art visual reasoning is essential and applicable to ongoing tasks like object detection, anomaly detection, or classification. Accordingly, better understanding of ‘Visual Question Answering’ (VQA) and ‘Visual Commonsense Reasoning’ (VCR) strategies can help AI algorithms better capture some of the complex visual and cognitive processes that humans do effortlessly.

Building off of work by Messina et. al., we run exper-

iments using images from the Synthetic Visual Reasoning Test (SVRT) dataset [2]. The black and white images from this dataset are used as input for our algorithms. For each experiment, we do a binary classification to identify a special type of spatial relationship. The output is 1 when the model(s) determines that the spatial relationship is present in the image and 0 when the model determines the relationship is not present. We run this for four different types of relationships as discussed in section 2.

Our motivation for this research is to identify the strengths and weaknesses of CNNs and hybrid-CNN architectures on the SVRT dataset and by extension VCR tasks more broadly. With our investigation, we can identify and optimize state of the VCR systems, focusing on resource efficiency and optimized complexity. Doing so can make way for a growing body of work in Computer Vision that considers problem-suitability and innovative architectural applications.

2. Dataset

For our dataset we will be using the Synthetic Visual Reasoning Test (SVRT) dataset [1]. This dataset is comprised of 23 classification based tasks using image frames (128 x 128, binary pixels) where multiple shape outlines are drawn. These classification tasks were originally administered to humans to test visual reasoning faculties and image associations. Each type of classification task is meant to target the compositional arrangements and relationships between the shapes in the image. For example, one type of classification task might be recognition that all the shapes in two images are the same, but are in a different orientation. Another might contain the same shapes but be positioned in a different sequential ordering. Another may be whether the shapes are concentric in the same way as the shapes in another image etc. For each problem, we have 400k in training, 100k in test, and 100k in val with even distribution between two labels.

2.1. Selected Problems and Classification

Of the 23 classification tasks, we chose to focus on the 4 prevalent SVRT problems that feature prominently in the

literature: Problem 1, Problem 5, Problem 20, and Problem 21. [2]. Descriptions of each task can be reviewed in Table 1. **Unless otherwise specified, we trained our models one problem at a time, using the set of positive sample images and negative sample images as input. As output, each sample received a 0 or 1 classification label.**






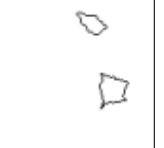


	Positive Sample	Negative Sample	Summary
1			True labels contain two identical shapes that are translations of each other.
5			True labels contain two sets of identical shape pairs.
20			True labels contain two sets of identical shapes that are reflections of each other.
21			True labels contain two shapes that are scaled, rotated, or translated versions of each other.

Table 1. SVRT Problems and Descriptions

2.1.1 Data Preparation and Features

In order to prepare our data, we ensured that all images were sized at 128 x 128 if they were not already. We then normalized images with a standard deviation of 0.5 and mean of 0.5 such that all pixel values fell into the range of $[-1, 1]$. In our model architectures we used an image to tensor conversion and then handled feature extraction at the architecture layer. For example, feature extraction is described in 4.2.3 as one of our studied methods while for CNN and CNN-Transformer Hybrid experiments the CNN layers operate as inherent feature extractors.

3. Literature

3.1. “Recurrent Vision Transformer for Solving Visual Reasoning Problems”, 2021

In this paper, Messina et. al. utilize the SVRT dataset to train a novel vision VQA architecture. The architecture proposed, the Recurrent Vision Transformer (RViT),

is a complex hybrid architecture meant to blend and reap the benefits of 1) Pre-trained ResNet50 layers, traditional upstream CNN layers, recurrent connections as one might find in an LSTM or GRU architecture, and a Vision Transformer (ViT) layers. The ResNet50 layers are intended to serve as a strong initialization method to ensure that complex features are extracted from the raw image data as well as ameliorate any vanishing gradient issues. Additionally, recurrent connections are meant iteratively refine observations and provide more stable conclusions. Therefore, the overall goal of this hybrid architecture is to refine its internal representations, while the spatial attention mechanisms enable it to focus on relevant parts of the input image, aiming to augment individual technologies by using them in combination. When compared to other structures, including ResNets and traditional vision transformers which achieve approximately 50% accuracy, RViT achieves much better results up to 99% for certain classifier groups. Taking much of our inspiration from Messina et. al., we attempt to build and benchmark more simplified VCR architectures to compare against their large and high-resource architecture. For training, Messina et. al. used Adam optimization, a learning-rate of 0.001 and trained for 200 epochs [2].

3.2. “Contrastive Pre-training and Representation Distillation for Medical Visual Question Answering Based on Radiology Images”, 2021

When focusing on VQA and VCA in applied contexts, particularly focusing on radiology tasks, it is difficult to accumulate large sets of classified labels. Thus, in order to promote VCA, Yang et. al. proposed a contrastive pre-training approach to achieve better results. The model architecture included contrastive learning as pre-training on three different classification types involving images of chest, brain, and abdomen. This contrastive pre-training was then distilled into a single student model that resulted in better VQA results when used in combination with ResNets and ViT style architectures. This likewise reduced the need for more specific annotations for specialized medical applications of VQA [3].

3.3. “Multiscale Feature Extraction and Fusion of Image and Text in VQA”, 2023

In this paper, Lu et. al. acknowledge that state-of-the-art VQA systems often struggle with accurately representing scene and object information from images and fully capturing textual information from questions. To address this, the paper proposes multi-scale feature extraction and fusion methods, which improve image representation. To resolve this problem, Lu et. al. summarizes existing literature and enumerates three primary multiscale feature extraction approaches, namely, image pyramid, feature pyramid, and the combined image-feature pyramid. Likewise, Lu et. al. also

proposes bilinear pooling within CNN feature extraction layers as an improvement. The ultimate goal of these modifications are to refine feature extraction techniques in VQA, aiming to balance model complexity with performance improvements. This balance would enhance the system’s ability to understand and reason about both visual and textual information. In experiments, these feature extraction techniques offered humble improvements increasing ResNet accuracy from approximately 74% to 75% [4].

4. Methods

4.1. Baseline

In the baseline approach, we utilize a Convolutional Neural Network (CNN) architecture. Specifically, the network is built using a series of convolutional layers followed by pooling layers to progressively reduce the spatial dimensions while increasing the depth of feature maps. The architecture leverages convolutional operations defined as:

$$\mathbf{X}_{l+1} = f(\mathbf{W}_l * \mathbf{X}_l + \mathbf{b}_l),$$

where \mathbf{X}_l and \mathbf{X}_{l+1} are the input and output feature maps at layer l , \mathbf{W}_l and \mathbf{b}_l are the weights and biases, and f is an activation function such as ReLU. The final layer is connected to a fully connected layer that maps the high-dimensional features to the answer space. The model is trained using a cross-entropy loss:

$$L = - \sum_{i=1}^N y_i \log(\hat{y}_i),$$

where y_i is the ground truth and \hat{y}_i is the predicted probability for each class.

4.2. Additional Approaches

4.2.1 CNN + Transformer

The second approach combines a Convolutional Neural Network (CNN) with a Vision Transformer (ViT) to enhance the model’s capability in capturing long-range dependencies within the image, based on the approach presented in Messina et. al’s paper [2]. Initially, a 5-layer CNN extracts feature maps from the input image, resulting in an $N \times N$ grid of D -dimensional visual tokens where $N = 4$ and $D = 512$. These tokens, along with a class token ([CLS]), are fed into a Vision Transformer encoder where the encoder weights are shared across all T layers where $T = 6$. The self-attention mechanism within the Transformer creates short paths between distant patches, facilitating robust feature extraction. In this architecture, we utilized 16 4×4 patches and 12 multi-attention heads. We likewise used query and key dimensions of 768. At each

time step t , the model produces an output y_t , and the loss is calculated using binary cross-entropy:

$$L_t = \text{BCE}(y_t, \hat{y}).$$

The total loss is aggregated as:

$$L_{\text{total}} = \sum_{t=1}^T \frac{1}{2} \left(\frac{1}{e^{s_t}} L_t + s_t \right),$$

where s_t are learnable parameters representing uncertainty. During inference, the final prediction $y = y_{\hat{t}}$ is selected based on the time step:

$$\hat{t} = \arg \max_t |y_t - 0.5|.$$

4.2.2 Contrastive Learning

In the third approach, contrastive learning is employed to pre-train models on large-scale unlabeled image datasets to capture diverse visual features. We adopt a self-supervised contrastive learning method, such as Momentum Contrast (MoCo), where pairs of augmented images \hat{x}_i and \hat{x}_i^+ are generated alongside a queue of negative samples \hat{q} . The feature representations $z_i = T_\theta(\hat{x}_i)$ and $z_i^+ = T_\theta^0(\hat{x}_i^+)$ are learned by minimizing the InfoNCE loss:

$$L = - \log \frac{\exp(z_i \cdot z_i^+ / \tau)}{\exp(z_i \cdot z_i^+ / \tau) + \sum_{j=1}^M \exp(z_i \cdot z_j^- / \tau)},$$

where τ is the temperature parameter. After obtaining the pre-trained models, a lightweight student model is trained to distill knowledge from these pre-trained models by optimizing a combined loss:

$$L_{\text{distill}} = \alpha(L_{\text{sim}} + L_{\text{dissim}}) + (1 - \alpha)L_{\text{class}},$$

where L_{sim} ensures the student’s features are similar to the pre-trained models’ features, L_{dissim} ensures features are dissimilar across different contexts, and L_{class} enables the model to classify different visual contexts effectively.

4.2.3 Feature Extraction + CNN

The fourth approach focuses on multiscale feature extraction and fusion using a pre-trained ResNet-152 model. The network extracts features from three different layers: conv3, conv4, and conv5, with output dimensions of $28 \times 28 \times 512$, $14 \times 14 \times 1024$, and $7 \times 7 \times 2048$, respectively. These multiscale features are fused using a top-down pathway similar to the Feature Pyramid Network (FPN). For instance, features from conv5 are upsampled and merged with conv4 features, followed by further upsampling and fusion with conv3 features. This hierarchical fusion is mathematically represented as:

$$F_{\text{conv3}} = F_{\text{conv3}} + \text{Upsample}(F_{\text{conv4}} + \text{Upsample}(F_{\text{conv5}})).$$

The combined features are then fed into a CNN to produce the final answer prediction (4).

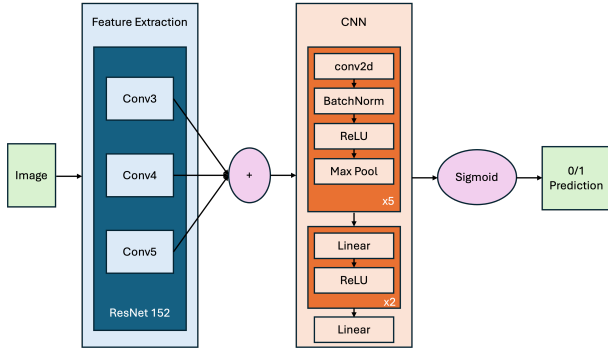


Figure 1. Feature Extraction + CNN architecture.

5. Results & Analysis

	Pblm. 1	Pblm. 5	Pblm. 20	Pblm. 21	Hyper-parameters
CNN	0.98	0.5	0.91	0.8	lr:0.0001, batch size:64, epochs:10
CNN, Contrastive Learning		0.5			lr:0.0001, batch size:64, epochs:5
Feature Extraction, CNN	0.9972	0.9986	0.9765	0.9458	lr:0.0001, batch size:64, epochs:5
CNN, Transformer	0.5	0.5	0.5	0.5	lr:0.0001, batch size:32, epochs:10

Table 2. Model Architecture vs. Test Accuracy Performance

	Pblm. 1	Pblm. 5	Pblm. 20	Pblm. 21	Test Acc.
CNN	✓		✓	✓	0.83

Table 3. CNN with combined training

5.1. CNN Baseline

The CNN architecture consists of 5 convolutional layers with ReLU activation and max-pooling, followed by 3 fully connected layers. The performance of CNN varies significantly across the four classification tasks as seen in Table 2. In Problem 1, which involves identifying two identical

shapes that are translations of each other, the CNN excels with a high accuracy of 98%, indicating its effectiveness in capturing simple geometric translations. Similarly in Problem 20, which requires identifying sets of shapes that are reflections of each other, sees a high accuracy of 91%, showing the model’s robustness in detecting reflective symmetry. We hypothesize that the convolutional layers are well-suited for detecting direct geometric transformation due to their ability to capture local patterns and features across different regions of the input image. However, in Problem 5, where the task is to identify two sets of identical shape pairs, the model’s accuracy drops to 50%, as evident in the confusion matrix of Figure 2a. This may suggest an increased task complexity, where the CNN needs to simultaneously identify and match multiple pairs of shapes. The relatively shallow design of the architecture may not provide sufficient hierarchical representation to capture the nuanced relationships between multiple shape pairs. Finally, Problem 21 is inherently more complex due to the variety of transformations involved, including scaling and rotation in addition to translation but still achieves a reasonable accuracy of 80%, most likely due to the similar learnings found in problems 1 and 20.

In addition to training the CNN architecture on each individual problem separately, we also trained the CNN architecture as seen in Table 3 on problems 1, 20, and 21 at the same time—100k for train, 12.5k for val, and 12.5k for test from each problem. The purpose of the combined training is to learn from a more diverse set of features occurring at the same time.

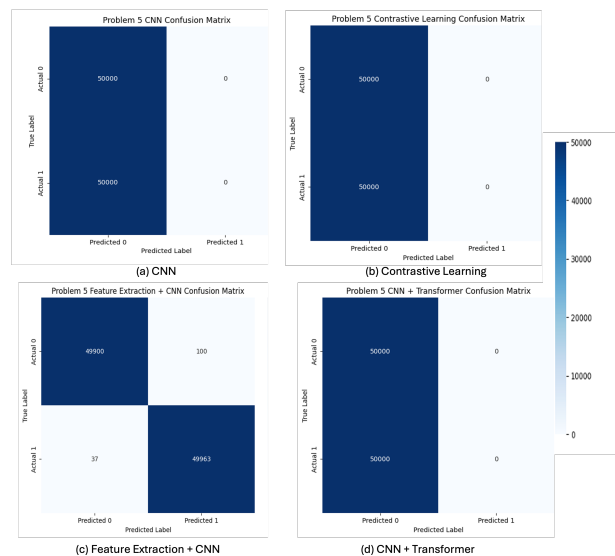


Figure 2. Confusion matrix for all experiments on Problem 5.

True Label	1	0
CNN Prediction	0	0
Feature Extraction + CNN Prediction	1	0

Figure 3. Example of a misclassification and classification for CNN and Feature Extraction + CNN on Problem 5.

True Label	1	1	0	0
CNN Prediction	1	0	1	0
Feature Extraction + CNN Prediction	1	1	0	0

Figure 4. Example of classifications for CNN and Feature Extraction + CNN on Problem 21.

5.2. CNN with Contrastive Learning

Given that we got relatively high-performing results of the CNN architecture on problems 1, 20, and 21, we sought to primarily improve the performance on problem 5. An initial approach we tried was contrastive learning such that the model could effectively learn the feature representations of similar vs. dissimilar shapes, which we thought was very well-suited for the dataset. However, as seen in Table 2, it did not yield better results, maintaining an accuracy of 50%. Since the effectiveness of contrastive learning relies heavily on the quality of the positive and negative pairs, it is possible that the pairs of identical shapes are not well-defined or there is ambiguity in distinguishing them from non-identical pairs. Consequently, the model might struggle to learn meaningful feature representations, leading to poor performance. Perhaps, robust data augmentation is necessary for the model to generalize well. Furthermore, a more complex, pretrained model such as ResNET with contrastive learning may be better suited to capture rich feature representations.

5.3. Feature Extraction with CNN

The feature extraction approach with CNN utilized a pretrained ResNet-152 model to extract a feature embedding vector to feed as input to the baseline CNN architecture.

Across all four problems, this approach outperformed the baseline and other experiments. Compared to the baseline, we saw the largest increase in performance of problems 5 and 21, which are of greater complexity than problems 1 and 20. We also see that overall, problem 5 had the highest accuracy using this approach, even though it had the lowest accuracy with the other approaches. The ability to merge features from different layers of the ResNet-152 model helps in recognizing and matching the pairs more effectively, showcasing the robustness of the method in handling increased task complexity with multiple pairs of shapes. For problem 21, we see a significant improvement from the baseline accuracy showing that multiscale feature extraction helps the model better capture the complexity of multiple transformations. Yet, the accuracy is still the lowest across the problems, indicating that the combined transformations are a more difficult task for feature extraction to generalize.

The hierarchical fusion of multiscale features significantly enhances the model’s ability to interpret and respond to complex visual inputs, making it a powerful technique for VCR. The accuracy improvements across the problems highlights the advantages of integrating features from multiple convolutional layers. This approach allows the model to capture both local and global patterns, which are crucial for understanding complex visual relationships.

5.4. CNN + Transformer

Across all four problems, the hybrid CNN + Transformer architecture showed the least improvement and the lowest accuracy relative to the other experiments. It was also the slowest architecture to train, taking between 2 – 2.25 hours to train an epoch. The following list describes the sub-experiments done to try and improve the results of this architecture (Note that combinations of the following were also attempted):

1. Including batch normalization layers after each of the 5 initial CNN layers
2. Including layer normalization layers before and after transformer layers in ViT
3. He initialization for CNN layer weights and Xavier initialization for Transformer and MLP weights
4. Gradient clipping during training
5. Various experiments with learning rates, batch sizes, and momentum
6. Training on smaller sets of data for more epochs (including 28k and 100k training set decreases, up to 100 epochs)

Ultimately, the best results are shown in table 2 and figure 5 show the hyperparameter selection of lr 0.0001, batch size 32, epochs 10. This architecture included both batch-norm and layer-norm as described above. While accuracy on its own was not much improved over the course of training, it should be noted that training and validation losses continued to decline over the course of training even though by small ($1e-3$ - $1e-4$) drops every epoch. Considerations for why the CNN-Transformer was not more performant are discussed at length in the discussion.

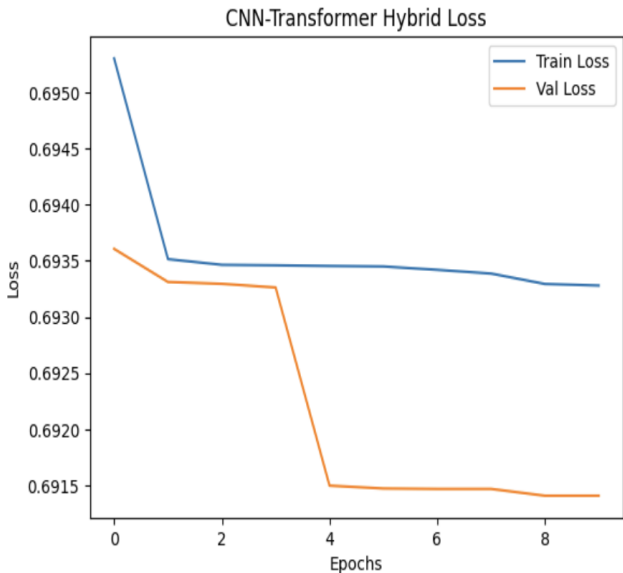


Figure 5. Describes micro-decline (order of $1e-3$ to $1e-4$) in loss function for CNN-Transformer Architecture

6. Discussion

6.1. Assessment CNN + Transformer

While the CNN-Transformer architecture did not yield results as promising as the pure CNN architecture or feature extraction architecture, we consider and compare this architecture relative to results in Messina et. al. to better understand why this might be the case [2]. Messina et. al. ran comparable experiments on ViT without any CNN layers and on RViT which included both pre-trained ResNet layers, traditional CNN layers, recurrent connections, and ViT. Our CNN-Transformer hybrid after 10 iterations is directly comparable to Messina et. al.’s ViT findings which achieved at best 50% accuracy on the four problem types. We would expect our hybrid structure to outperform the sole ViT experiment however it should be noted that our architecture was only capable of being trained for 10 epochs while the ViT was given 160 epochs. We discuss the reasons for the limited epochs as one of time and compute availability described at length in 6.2. The RViT structure however greatly

outperformed our CNN-Transformer hybrid, though was comparable to our CNN and Feature-Extraction architectures. One reason why the CNN-Transformer hybrid might have been less successful than RViT is because it did not include the pre-trained ResNet50 layers which were included before CNN and followed by transformer. Our thought is that these pre-trained layers might have provided a better initialization structure than CNN in isolation or CNN with He initialization. It is possible that our learning overall was slower and unable to compensate for this poor initialization through training. Additionally, in order to make this experiment novel and test simplified structures, we did not include any recurrent connections, which may have shown to add a much needed benefit during training. Lastly, RViT also had the benefit of training over 200 epochs while our CNN-Transformer hybrid only trained for 10 epochs.

As for why the CNN-Transformer architecture did not outperform the other experiments (CNN and Feature-Extraction + CNN), we have a few hypotheses. For one, the alternative two architectures are smaller and typically converge faster during training. Thus, if limitations were not at play, it may have been appropriate to compare after a longer duration of training with the CNN-ViT. That being said since strong results on the VCR task were achieved with these smaller structures, it also questions the theoretical need for a ViT at all for these types of problems. While different applications will require more bench-marking and research, it is possible that VCR tasks are ill-suited to large hybrid structures and both RViT and our CNN-Transformer hybrid may be overkill in these specific use-cases.

6.2. Limitations

Through most of this research, due to compute credits offered during the 231n class as well as general GPU shortages (especially for faster V100 and A100), we were only able to acquire NVIDIA T4 GPUs. Operating with this compute meant that training time for our architectures took anywhere between 1hr-2hrs per epoch. Moving at this pace made it harder to improve and fully train our models to our satisfaction. Especially, for the CNN-Transformer hybrid, we would have liked to assess and improve model performance with greater efficiency. (Note Colab VMs were not an option for us, as Colab VMs did not have enough persistent memory to store the SVRT dataset).

7. Future Work

If we had access to more computational resources, we would like to run extensive experiments with various hyperparameters. We want to explore advanced data augmentation methods to improve model robustness, especially in relation to improvements of contrastive learning. We would also like to explore how this dataset can be used for multi-task learning where a model can learn to classify as well as perform

object detection or segmentation to enrich feature representations, especially since such a dataset could be potentially useful for geographic segmentation.

8. Conclusion

Our study explores the effectiveness of various approaches to the SVRT dataset, including a baseline Convolutional Neural Network (CNN), a hybrid CNN-Transformer model, contrastive learning techniques, and advanced feature extraction methods. Overall, we found a feature extraction + CNN approach to be the most effective across the board, indicating that pre-trained models can significantly enhance the model's capability to interpret complex visual representations. In contrast, the hybrid CNN-Transformer model, while theoretically promising for capturing both local and global features, showed uniform but low performance across all tasks. This suggests a need for further tuning and possibly architectural adjustments to effectively harness the complementary strengths of CNNs and Transformers in this context. Overall our core contributions and revisions to the [2] paper include:

- A simplified CNN architecture compared to ResNet reached comparable results on problems 1, 20, and 21.
- A simplified CNN architecture trained on problems 1, 20, and 21 together vs. individually.
- Experimentation with contrastive learning.
- Advanced feature extraction with CNN architecture.

9. Contributions

We worked synchronously together on the data preparation. Riya worked on the basic CNN architecture and CNN + contrastive learning, Jasemine worked on the CNN + Transformer architecture, and Megan worked on the Feature Extraction + CNN. We all spent time reviewing each other's code and results.

References

- [1] F. et. al. Comparing machines and humans on a visual categorization test. *Proceedings of the National Academy of Sciences*, 108(43):234–778, 2011. [1](#)
- [2] M. et. al. Recurrent vision transformer for solving visual reasoning problems, 2021. [1](#), [2](#), [3](#), [6](#), [7](#)
- [3] Z. Liang, W. Jiang, H. Hu, and J. Zhu. Learning to contrast the counterfactual samples for robust visual question answering. In B. Webber, T. Cohn, Y. He, and Y. Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3285–3292, Online, Nov. 2020. Association for Computational Linguistics. [2](#)
- [4] S. Lu, Y. Ding, M. Liu, et al. Multiscale feature extraction and fusion of image and text in vqa. *International Journal of Computational Intelligence Systems*, 16:54, 2023. [3](#)