CE7454 Project 1: Scene Graph Relation Classification

Jeremy Styborski (Matric: G2202711K)
School of Computer Science and Engineering
Nanyang Technological University
styb0001@e.ntu.edu.sg

October 10, 2022

Abstract

In this report, we investigate the ability of multiple deep learning networks to classify the scene graph relations present in images. Multiple variants of the ResNet architecture are considered, as well as a visual transformer. This is followed by multiple hyperparameter studies on input resolution, layer freezing, learning rate, and weight decay. Finally, we compare the optimal tuned models to a zero-shot CLIP implementation. Experimental results show that hyperparameter tuning is indeed effective for increasing validation recall, achieving a maximum score of 24.40, yet the tuned models still underperform relative to CLIP.

Figure 1. Example of the SGG process with sample image [1].

1. Introduction

Scene graphs for images depict the behaviors of and relations between objects in an image [1]. Scene Graph Generation (SGG) problems are primarily concerned with identifying objects, typically with a distinction between subjects and objects, as well as identifying relations between subject-object pairs. Figure 1 shows a simple example of an image with its corresponding scene graph. In this example, the network has selected 'dog' as the subject and 'surfboard' as the object. The relation between the object pair is 'ride'. Larger scene graphs may include many relations between multiple objects, creating a web of interactions.

Multiple techniques have been explored for generating scene graphs. As noted in [1], there are two main network architecture types for SGG: bottom-up and top-down. Bottom-up methods apply an object detection method, such as Faster R-CNN [11], to an image and then route the outputs to a relation recognition network, such as deep relational networks [3]. Top-down methods perform object detection and relation recognition simultaneously, such as in the fully convolutional SGG network [9]. A newer paradigm, Panoptic Scene Graph (PSG) generation [16] incorporates panoptic segmentation techniques into the SGG problem in order to avoid ambiguity associated with bounding boxes for object detection.

1.1. Deep Learning Project

The project presented to CE7454: Deep Learning is primarily concerned with identifying relations present within images and can be considered a sub-problem of SGG. The main goal of this relation identification project is only to identify relations or behaviors present in images, with no concern for the objects. Given a set of relations, the training images are tagged with the relations present within each. There is no information given regarding object labels or location. Clearly this relation identification task is reduced in scope relative to SGG. The problem can be considered as a type of image classification or action recognition.

Raw images for the reduced task were drawn from the MSCOCO 2017 Train/Val/Test repository [8]. Relation labels for each image were assigned manually from a set of 56 distinct relations, such as 'playing', 'walking on', and 'in front of' [16]. Each image includes at least three relation labels.

The training, validation, and testing datasets provided for the purposes of the Deep Learning Project are subsets of dataset mentioned above. Training and validation datasets contained 4,500 and 500 images, respectively, and contain both the raw images and object relations labels. A test set of 500 images was provided without labels. No information on bounding boxes, object labels, or segmentation was given.

The performance of the models is evaluated via mean

recall. Mean recall is evaluated by extracting the top three relation class predictions for each image and counting the class labels in common with the ground truth labels.

2. Prior Work

Although the source of the relation identification task is rooted in SGG, the task was treated as an image classification problem. Architectures from other image classification methods were adopted and explored.

ResNet [7] utilizes skip connections to enable very deep networks without exploding or vanishing gradients, allowing very deep image representations. ResNet gave state-of-the-art performance for image classification and object detection. Subsequent architectures explored modifications to the ResNet design. Wide ResNet [17] increased channel numbers and reduced network depth in order to reduce the number of forward and backward operations while still achieving state-of-the-art results on image classification. EfficientNet [13] coordinated scaling between channel number, network depth, and feature resolution to reduce overall network size while still achieving top accuracy on multiple image classification datasets.

The success of transformers in NLP [14] led to the introduction of visual transformers [4] that attend to visual "tokens" derived from image patches and perform competitively with convolutional networks.

Finally, the recent popularity of self-supervised training [2], [6], [5], has gained attention from companies with access to large server clusters and petabytes of training data. Developed by OpenAI and trained on more than 400 million images, CLIP [10] uses image/text pairs to learn image and text encodings in a self-supervised manner. Its encodings are versatile across multiple datasets and its pretrained models provide a robust basis for zero-shot evaluation.

3. Experiments

Five main experiments are conducted in this paper using the provided training and validation data. The first was a survey of image input resolutions using ResNet50 (RN50). Then multiple network architectures were examined, including RN18, RN50, WideResNet50 (WRN50), EfficientNet-S (EffN-S) and Visual Transformer Base 16x16 (ViT-B/16). Finally, RN50 and ViT-B/16 were evaluated for various levels of frozen layers, multiple minimum learning rates, and multiple weight decays.

All models were loaded with weights pretrained on ImageNet1K [12]. Accordingly, the final fully-connected layer of each model was replaced with a fully-connected (FC) layer appropriately sized for 56 classes. Loss was determined by a sum across Binary Cross Entropy losses for each class. Unless otherwise stated, experiments were conducted with SGD with a training batch size of 16, momentum at 0.9, weight decay of 0.0005, and cosine learning rate decay from a maximum of 1 to a minimum of 0.001. Additionally, unless otherwise stated, experiments were conducted to 300 epochs and with an image input resolution of 224x224.

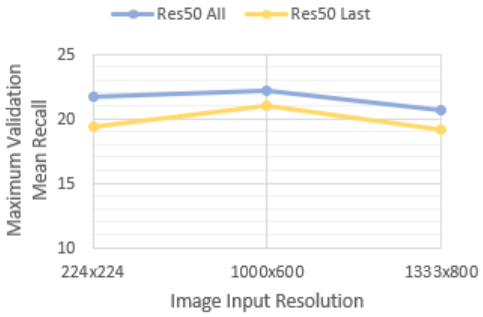

Figure 2. ResNet50 mean recall for three image resolutions, separate curves for all weights trainable and last layer trainable

Additionally, two zero-shot CLIP models, using RN50 and ViT-B/16 backbones, were evaluated on the relation classification problem.

Starter code for the relation identification project was provided on Github [15].

3.1. Image Resolution Sensitivity

The first experiment conducted was a simple survey on input image resolution. The starter code was given with images rescaled to a resolution 1333x800, which is considered highly detailed. There are three primary motivations for this experiment. First, 1333x800 images require a lot of memory to process. Second, the average image height and width for images contained in the training, validation, and testing sets are 585px and 481px respectively. Rescaling these images to 1333x800 is akin to generating redundant data. Only about 25 percent of the information in a rescaled 1333x800 image is independent; the rest is linearly interpolated. Third, most models examined are pretrained on 224x224 scaled ImageNet images. Therefore, convolutional filters in the pretrained network would expect image features that are scaled to 224x224 and may not function optimally otherwise. Relatedly, the ResNet-style architectures typically contain adaptive pooling layers that allow the network to process images of any size above the minimum resolution. Larger images forced through the network will return much larger resolutions right before the final adaptive pooling layer, meaning that much of the information is compressed out in the final pooling layer. Due to the absence of adaptive pooling layers in the ViT models, ViT cannot process images larger than 224x224.

Figure 2 shows the maximum validation mean recall for RN50 at three different image resolutions. The choices of 1333x800 and 224x224 have been discussed previously.

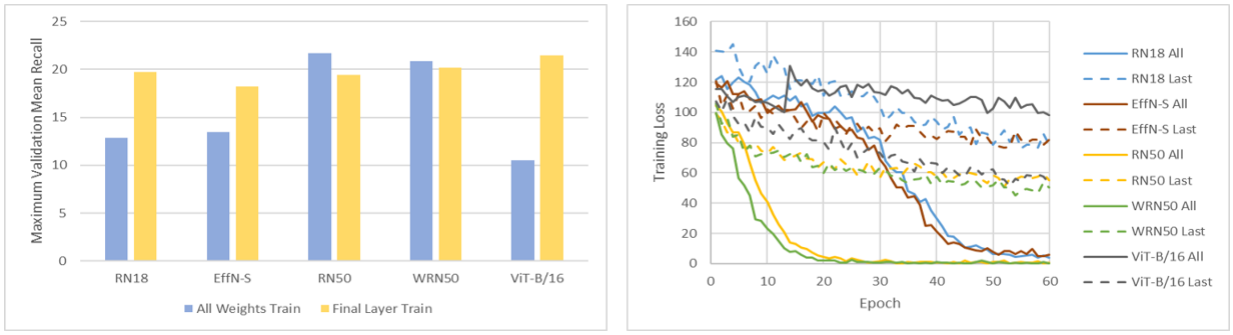

Figure 3. Maximum validation mean recall and training loss results across five network architecture types and two training schemes

The additional resolution of 1000x600 was chosen because it retains the aspect ratio of 1333x800 while reducing the total pixels by roughly 45 percent. RN50 was evaluated with all weights trainable and with only the final FC layer trainable. For both training methods, 1000x600 input image resolution gave the optimal mean recall. Moreover, the mean recall is higher for 1000x600 images compared to 224x224 images even when the convolutional layers are frozen. This could imply that the information lost due to rescaling images down to 224x224 is more detrimental relative to the benefit of using an image resolution expected by the convolutional filters. Of the three resolutions tested, 1000x600 rescaling provides the resolution closest to the native image sizes without losing information. This hypothesis is supported by the comparatively larger difference between recall results for 1333x800 for different training regimes. 1000x600 retains the original image's information without diluting it too much, and this dilution effect is reduced when the ResNet is allowed to finetune its filters for larger images.

3.2. Architecture Survey

The second experiment conducted was a survey of network architectures, RN18, RN50, WRN50, EffN-S, and ViT-B/16. Two methods of finetuning were used for each network: the first was finetuning the entire network and the second was finetuning the final FC layer only. All models were trained to 60 epochs in this experiment.

Figure 3 reports the maximum validation mean recall for each finetuning method and each network as well as the training loss for each. There are four main takeaways. First is the robust recall performance for all models finetuning the final FC layer. As all models have been pretrained on 224x224 ImageNet images (except EffN-S), this result suggests that the pretrained models can successfully extract and classify image features, including image features that correspond to relations rather than objects. Second, as shown in the training loss chart, training all layers of ResNet architectures leads to overfitting, achieving nearly zero training loss. Surprisingly, the larger models (RN50 and WRN50) converge to a low training loss much quicker than the smaller models (RN18 and EffN-S). Perhaps the larger ResNet architectures with more layers and skip connections to process images more deeply are more generalizeable to new tasks and are more readily finetuned to new datasets. Third, despite overfitting to the training dataset, the fully trainable large ResNet architectures (RN50 and WRN50) still achieve a high mean recall on the validation set. This is counterintuitive. Perhaps the validation and training sets are very similar. Additionally, as mentioned in the second takeaway, larger networks may be more generalizeable to new datasets. Fourth, the ViT model never appears to overfit, but its performance when all layers are trainable is unimpressive. Note the training loss increases dramatically around epoch 15 and then never achieves a better training loss. Perhaps the cosine annealing of the learning rate caused the optimizer to become stuck in a poor local minimum.

Due to their top performances on validation mean recall, both RN50 and ViT-B/16 were selected for further study.

3.3. Frozen Layers, Learning Rate, Weight Decay

Figure 4 shows the RN50 and ViT-B/16 models evaluated under multiple sensitivity studies. When testing each variable, all other variables were held at their default values. When testing the minimum learning rate and weight decay variables, only the final FC layer was finetuned.

The first sensitivity study looked at the number of trainable layers. Previous experiments have covered the difference between fully trainable and last layer trainable and were repeated out to 300 epochs here. Additionally, one data point was added that trained the final FC layer and final sequential block of each network. For RN50, this is the final 3 bottleneck layers. For ViT-B/16, this is the final 3 transformer modules. RN50 shows a slight performance boost when allowed to train the last sequential block as well. This is likely due to a balance between extra finetuning yet disallowing the overfitting seen previously with ResNet architectures. ViT-B/16 performs best when only training the final FC layer, confirming the trend seen in Figure 3.

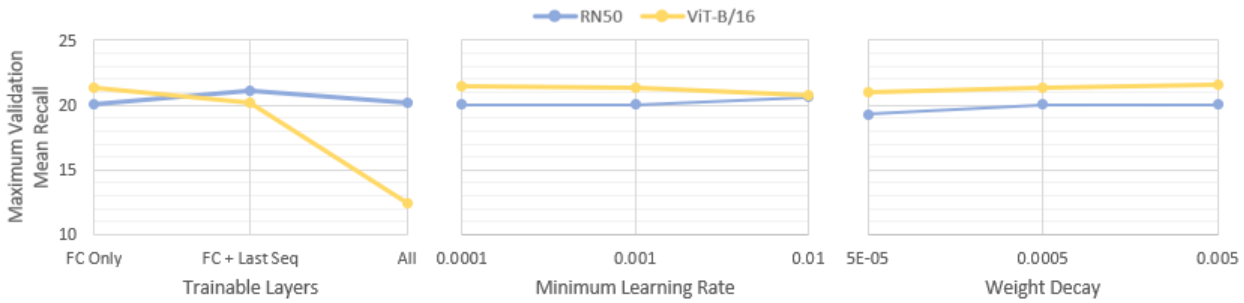

Figure 4. Maximum validation mean recall across three studies: number of trainable layers, minimum learning rate, and weight decay.

	Validation Recall	Test Recall
RN50	24.40	24.45
ViT-B/16	21.27	19.64
CLIP-RN50	25.92	25.44
CLIP-ViT-B/16	28.82	28.84

Figure 5. Performance of best RN50 and ViT-B/16 as well as CLIP models on validation and test recall.

The validation mean recall performance for both RN50 and ViT-B/16 is mostly flat with respect to variations in minimum learning rate and weight decay. Even so, we can discern a slight trend for both variables. The maximum mean recall decreases slightly as the minimum learning rate increases. This trend does not appear to hold for RN50, but upon closer inspection we found that the maximum mean recall value for RN50 at a minimum learning rate of 0.01 was highly anomalous, generally remaining below the mean recalls for RN50 at other minimum learning rates except at one epoch. Mean recall shows a slight positive correlation with the weight decay coefficient, maximizing at a weight decay of 0.005 for both networks. In general, the correlation between mean recall performance and minimum learning rate and weight decay is so slight that all trends are possibly statistical anomalies. It is more attractive to conclude that both RN50 and ViT-B/16 give robust performance across a range of learning rates and weight decays.

4. Test Set Performance

Due to their superior performance, both RN50 and ViT-B/16 models were selected for assessment on the test set. Lessons from the experiments in section 3 were incorporated. The final RN50 model was trained with 1000x600 input image resolution and trainable final sequential block and FC layer. The final ViT-B/16 model used a trainable FC layer. Both models were trained for 300 epochs with a minimum learning rate of 0.0001 and a weight decay of 0.005. Validation and test set mean recall performance are shown in Figure 5. The optimized RN50 model performs better than any previous model, achieving a mean recall of 24.40, and is robust when evaluated on test set. The optimized ViT-B/16 model underperforms the ViT-B/16 models examined previously and then performs notably worse on the test set. The reason for the drastic underperformance on the test set is uncertain, but it may be due to some degree of information leak; perhaps the ViT-B/16 was unwittingly overfit to the validation and training sets.

Borne out of simple curiosity, two zero-shot CLIP models, using RN50 and ViT-B/16 backbones respectively, were evaluated on the same validation and test sets. Neither CLIP model was finetuned on the relation identification datasets. Similarly, no classification layer was appended to the models to perform a classification task. Rather, all 56 relation text descriptions were tokenized and encoded using the native CLIP text encoder. Each validation and test set image was encoded using the native CLIP image encoder. The cosine similarity between encoded images and relation class encodings was computed and the three classes with the highest similarity to each encoded image were selected as the relation predictions. As shown in Figure 5, both zero-shot CLIP models outperformed our best finetuned RN50 model on both validation and test recall.

5. Conclusion

In this paper, we discussed the problem of Scene Graph Generation and how the project scope is derived from SGG. In experiments, we explored the effects of model architecture, image resolution, layer freezing, learning rate, and weight decay on recall performance. We demonstrated the robustness of models with only the final layer trainable across multiple models and that image resolution close to the native image resolution is a likely factor in recall performance. ResNet architectures benefit from having multiple layers finetuned whereas Visual Transformer models perform best with only the final layer trainable. We showed the invariance of model performance to learning rate and weight decay parameters. Finally, we demonstrated the dominance of zero-shot CLIP encodings over our models.

References

[1] X. Chang, P. Ren, P. Xu, Z. Li, X. Chen, and A. G. Hauptmann. A comprehensive survey of scene graphs: Generation and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2022.

[2] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations, 2020.

[3] B. Dai, Y. Zhang, and D. Lin. Detecting visual relationships with deep relational networks, 2017.

[4] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020.

[5] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko. Bootstrap your own latent: A new approach to self-supervised learning, 2020.

[6] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning, 2019.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

[8] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft coco: Common objects in context, 2014.

[9] H. Liu, N. Yan, M. S. Mortazavi, and B. Bhanu. Fully convolutional scene graph generation, 2021.

[10] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision, 2021.

[11] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2015.

[12] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[13] M. Tan and Q. V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. 2019.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, 2017.

[15] J. Yang. Openpsg. `https://github.com/Jingkang50/OpenPSG`, 2022.

[16] J. Yang, Y. Z. Ang, Z. Guo, K. Zhou, W. Zhang, and Z. Liu. Panoptic scene graph generation. In *ECCV*, 2022.

[17] S. Zagoruyko and N. Komodakis. Wide residual networks, 2016.

14	G2201013J	8	10/01/22		32.12768 (14)
15	G2201540L	12	10/06/22		31.76248 (15)
16	G2200706B	9	10/08/22		31.63082 (16)
17	G2204045F	5	10/07/22		31.00071 (17)
18	G2203858H	4	10/08/22		30.54685 (18)
19	G2203647J	8	10/10/22		30.37679 (19)
20	G2100655H	18	09/27/22		30.27323 (20)
21	G2202925L	7	09/25/22		30.25803 (21)
22	wang1753	11	10/06/22	G2200640K	30.20509 (22)
23	G2201488K	13	10/09/22		30.08131 (23)
24	G2204096D	6	10/07/22		29.43808 (24)
25	G2202712G	16	09/29/22		28.93513 (25)
26	G2202711K	6	10/03/22		28.84711 (26)
27	G2204168F	10	10/09/22		28.80775 (27)
28	G2204074C	6	10/09/22		28.76391 (28)
29	G2104441H	8	10/08/22		28.44111 (29)
30	G2204155F	5	09/15/22		28.02969 (30)
31	G2201371J	3	09/30/22		28.00320 (31)
32	G2203624L	28	10/03/22		27.89808 (32)
33	G2104574F	10	10/09/22		27.89012 (33)
34	G2202031C	1	10/09/22		27.59612 (34)
35	G2104967G	10	10/07/22		27.45658 (35)
36	G2200925E	12	10/08/22		27.40616 (36)
37	G2104242A	5	10/03/22		27.26830 (37)
38	G2203839J	8	10/04/22		27.10227 (38)
39	G2204057J	9	10/09/22		26.95346 (39)
40	G2202082A	7	10/09/22	断水流大师兄	26.46379 (40)

Figure 6. Screenshot of CodaLab results containing the final score for our submission. Note that the screenshot was taken before the end of the competition, such that the displayed rankings are not final.