MatchCLIP: Viewing Object-Relation Classification As A Matching Problem

Jiyuan Shen
SCRIPTS, Nanyang Technological University, Singapore
jiyuan001@e.ntu.edu.sg

Abstract

Knowing the apparent objects in the images are not enough. People always want the AI model to be much more intelligent. Just like Feifei Li once said [1] *'For machines to see the world, the hard part is not evolving the machine to what we humans are at three years old, but how to train the machine from three years old to what we humans are at thirteen years old'. Fortunately, the PSG classification task offers an excellent opportunity to explore the fine-grained and high-order semantic property in the images, which researchers used to pay less attention. In this report, I view the object-relation classification task as a matching problem and propose a model called MatchCLIP, which various object in the image can query its most corresponding relationship automatically with the help of explicitly encoding the relation tokens. Additionally, after making full use of the powerful CLIP pre-training model, it can achieve 34.60% mean recall@3 on the PSG classification competition leaderboard. I open-source our code at https://github.com/shenjiyuan123/CE7454_PSG.*

1. Introduction

Nowadays, most of the current computer vision tasks focus on the objects in the image. For example, image classification requires models to identify the object classes in the image. Object detection and image segmentation further require the model to identify objects in the image and find their locations (with different granularities). However, it seems that so many tasks ignore the semantic property like relation between objects inside the images. Therefore, in this assignment, the PSG Classification Challenge bridges this gap and requires us to explore how to train a computer vision model to well identify the relations in the images.

To be more specific, PSG Classification Challenge needs to establish a model that can output three most salient relations in the picture. It actually decreases the difficulty from pixel-level relation classification to image-level one. For example, in Fig. 1 when the model sees the photo, rather than providing the information of the objects (*e.g.* person, elephant), it only needs to output three salient relations such as feeding, leaning-on, and looking-at.

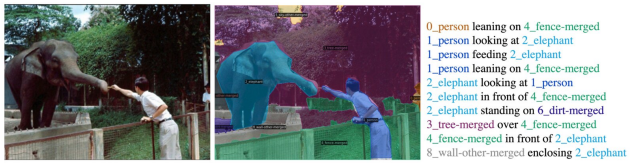
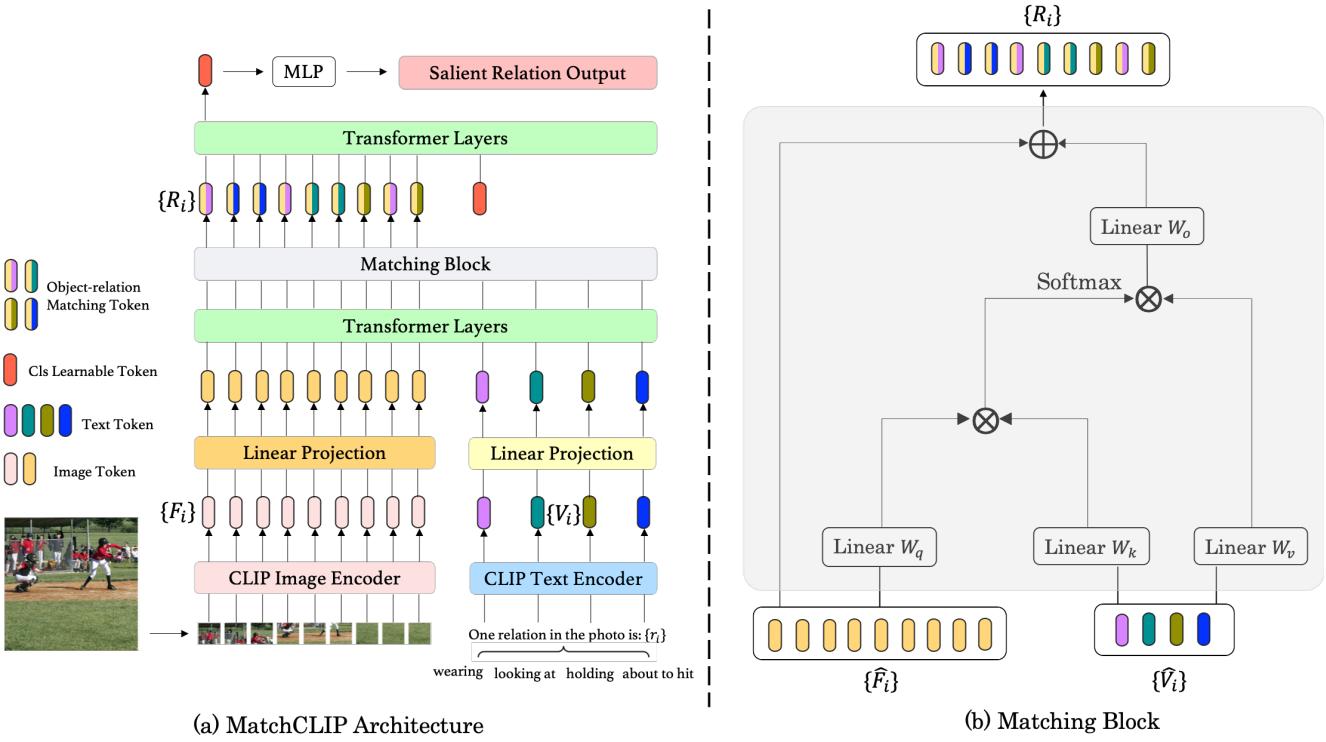


Figure 1. Example of the original PSG task requirement, which requires not only the object-relations, but also the panoptic segmentation to demonstrate exactly which two objects have the relationships. However, in this project, only three most salient relations in the image are required.

To solve this problem, a naive idea is treating it directly as a classification task. However, it definitely cannot work well (approximate at about 15.49% mean recall@3) because the objects relationship is not apparent and direct like objects categories at all. Some other existing methods like CLIP series [6, 9, 13] can model the object relationship to some extent despite that they are not designed to fulfill that characteristic straight-forward, in fact, their models are still not explicitly constructed objects relationship. CLIP implicitly learn about the relationship inside the images with the help of the large-scale datasets and contrastive Language-Image pre-training via natural language supervision. Although we can simply modify these models by fine-tuning only the last fully connected layer and freezing the former layers in order to predict the object relationships, the *cls* token cannot work well and make a good prediction (approximate at about 23.03% mean recall@3). The reasons are: 1) the relationship is implicit and output it from a object-based model is still too difficult; 2) the attention module only connect with different levels of feature maps, not directly with the relation tokens. However, the CLIP model is still attractive and powerful. If use the CLIP model as a feature extraction and then explicitly embed the relations into these, we possibly can make fully use of the CLIP's strong power. Additionally, due to that the re-

[1] https://www.ted.com/talks



(a) MatchCLIP Architecture

(b) Matching Block

Figure 2. (a) **MatchCLIP Architecture.** MatchCLIP contains two basic components: standard Transformer layers and Matching block. The raw input images and relation texts are both encoded by the CLIP pre-training model. After that, Transformer layers will fuse the object and relation; and Matching block will explicitly query the object with its most corresponding relationship. Finally, a learnable classification token is added to retrieve the most salient object-relations. (b) **The Architecture of Matching Block.** The Matching block takes the image tokens and text tokens as inputs. It matches all the image tokens to the corresponding relation token based on similarity in the embedding space. The assigned tokens are then fed into a linear layer and added with the original $\{\hat{F}_i\}$.

lationship is always based on two distinct objects, it can be viewed as a matching problem. In other words, one object can query its most corresponding relationship automatically with the help of explicitly encoding the relation tokens.

As a result, I propose a network called MatchCLIP inspired from GroupViT [14] and modify it to adapt the PSG Classification task. The proposed method is easy to implement and do not need much modifications from the naive ViT [4]. The main contributions in this report include:

- Use CLIP as an image feature extraction and propose a prompt engineering to assist the model converge.
- View object-relation classification as a matching problem and propose a Transformer-based model called MatchCLIP to explicitly query the relationship between different objects.
- Do a detailed ablation experiments and find the optimal training settings and hyper parameters. Achieve 34.60% mean recall@3 on the PSG classification competition leaderboard.

2. Method

In model design I am mainly inspired by the original Vision Transformer (ViT) [4] and GroupViT [14]. As shown in Fig. 2, I make some changes so that it can be suitable and straight-forward to the PSG classification task. Additionally, I don't want to waste the excellent performance of CLIP, especially its powerful capability of feature extraction and expression. So, I make fully use of the CLIP pre-training model and take the advantage of using its image and text encoder. All in all, the proposed method is not hard to understand and also simple to reproduce.

2.1. CLIP Pre-training Model

CLIP (Contrastive Language-Image Pre-Training) is a neural network trained on a variety of (image, text) pairs. It can be instructed in natural language to predict the most relevant text snippet, given an image, without directly optimizing for the task, similarly to the zero-shot capabilities of GPT-2 [10] and 3 [1]. It uses a very large-scale datasets (about 400 million image-text pairs collected from the internet) to train this model. Generally, according to the [9],

CLIP is good at zero-shot and transferring to downstream tasks, which is often competitive with a fully supervised baseline without the need for any dataset specific training. Given that excellent performance, I choose to use CLIP as a image and text feature extraction ,and rely on its powerful and rich expression capacity for the relationship classification task.

Image Encoder. The image encoder uses two types model during the training process that are ResNet [5] (includes EfficientNet-style [12]) and ViT. Due to that I cant to use image encoder as a feature extraction, I choose to freeze the parameters during the training process. Additionally, the original image encoder outputs only one token (similar with the *cls* token for the contrastive learning). However, from my perspective, it leads to the model to lose lots of fine-grained features so I maintain all the tokens (768 or 1024) and view these as the image tokens.

Text Encoder & Prompt Engineering. Similar to the image encoder setting, it also contains two types of basic model, ResNet and ViT. I also choose to detach the gradient back-propagation so that can maximize the ability of CLIP expression. And for the prompt, I try some examples and find the most simple and basic one can work well. So I just use " One relation in the photo is $\{V_i\}$."

2.2. MatchCLIP

Inspried by the GroupViT, which brings back the grouping mechanism into deep network and allows learning semantic segmentation tokens to emerge automatically with only text supervision, I proposed MatchCLIP that allows the network to automatically match the objects with the corresponding relationship. At the end of the model, I use a learnable classification token to retrieve the salient relations and add a MLP to get the final prediction.

Intuition. The motivation for building MatchCLIP is basically because that: 1) I am attracted by the power of CLIP and want to explore its potential. By using its pre-training model, I can easily extract high-level feature maps that helps construct relation-based model. 2) The relationship is always based on two distinct objects, so it can be viewed as a matching problem. The naive idea is that we can enumerate all possible cases and use some extra methods (like post-processing) to determine which one is most likely to happen. However, we can also convert the enumeration idea to query-value idea. With the help of attention module, I keep the minimum modifications to explicitly achieve object-relation matching problem.

Architecture. As Fig. 2(a) shows, the model has two branch at the beginning, Image Encoder and Text Encoder. Firstly, I split an input image into N non-overlapping patches and use the CLIP pre-training model to extract the image and text token, following a linear projection. After that, I feed them into the Transformer layers and Matching block. Typically, I repeat three standard Transformer layers and use one Matching block, which proves to have the best performance by the empirical experiments. If image token denotes F_i and text token denotes V_i, then the process can be fomulated as following:

$$\{\hat{F_i}\};\{\hat{V_i}\} = Transformer(concatenate([\{F_i\};\{V_i\}])) \tag{1}$$

$$\{R_i\} = MatchingBlock(\{\hat{F_i}\};\{\hat{V_i}\}) \tag{2}$$

When I get the $\{R_i\}$, I add only one learnable classification token to retrieve the most salient relations. During the experiments, I also test some other variations. For example, use hierarchical Transformer layers with multiple *cls* token so that the model can have multi-stage retrieval processes. Also, I explore the influence about adding or decreasing the number of *cls* tokens and Transformer layers. But at last, considering about the trade-off between accuracy and efficiency, it seems that using one learnable classification token and just three Transformer layers work well. Then, I just use the *cls* token and abandon the rest. After that, I feed it into a MLP and get the final output prediction. This process can be formulated as:

$$\hat{cls} = Transformer([\{R_i\}; cls_j])[j] \tag{3}$$

$$output = MLP(cls) \tag{4}$$

One more thing needs to be aware is that the first Transformer layers and the second one can have the same layers' number and architecture setting, but also can be different. In the later experiments, I just keep it the same.

Matching Block. As shown in Fig. 2(b), the Matching block takes the image tokens and text tokens as inputs. It matches all the image tokens to the corresponding text token based on similarity in the embedding space.

Formally, I compute the similarity matrix A^l between the image object tokens tokens $\{\hat{F_i}\}$ and relation $\{\hat{V_i}\}$ via a Softmax operation. The whole process can be described as following:

$$A^l = softmax(\frac{W_q \times {W_k}^T}{\sqrt{d_k}}) \tag{5}$$

$$\{\hat{R_i}\} = Linear(A^l \times W_v) + \{\hat{F_i}\} \tag{6}$$

3. Experiments

3.1. Setup

Model Variants. I build several different size and type of models, as shown in Tab. 1. The baseline contains two architecture: ResNet and linear probe after CLIP. The MatchCLIP series contain three different architectures. I find that

Model	Image size	Patch size	Transformer layers	Matching block	Hidden size D	MLP size	Heads	Params
ResNet50	224	/	/	/	/	/	/	25M
Linear probe CLIP-B/16	224	16	/	/	/	/	/	28T*
MatchCLIP-Base	224	16	3	1	512	1024	8	17M
MatchCLIP-Large	336	14	3	1	1024	4096	16	101M
MatchCLIP-Huge	336	14	3	1	1600	4096	16	186M

Table 1. Details of baseline model and MatchCLIP model variants. 28T* means that I only calculate the final fine-tuning layer's parameters. MatchCLIP-Huge is the model that I get the best mean recall@3.

the performance can be improved further with the increase of the model's size. However, for the trade-off between efficiency and accuracy, I use the MatchCLIP-Huge to get the best mean recall@3.

Training. I train the ResNet using SGD with momentum of 0.9 and learning rate of 0.00025. The MatchCLIP series follow the same setting, using AdamW (AMSGrad variant [11]) as the optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, a batch size of 32 and apply a smaller weight decay of 0.0005 (From the experiments, I find that AdamW converges much stabler and faster than the Adam and SGD. SGD needs a more delicate adjustment, it's annoying and unstable so I abandon it). Additionally, I set up a learning rate scheduler that follows linear warmup to 0.00025 and after some epochs, it will follows a cosine decay to zero at last. For all the models, I train it for 64 epochs.For the loss function, I try some different types like Focal loss [7] and BCE loss with label smoothing [8]. However, these methods that normally can work well doesn't perform effectively at all. It's confusing so I just maintain the original BCE loss at last.

Regularization. Basically, I use three different regularization methods. Firstly, I add dropout to the MatchCLIP model. For the Base and Large model, I set the dropout to 0.1. For the Huge model, I enlarge it to 0.3. Secondly, I use some dataset augmentation like auto-augment [2] and random augment [3]. Additionally, due to that the dataset labels are not balanced (some classes have more than 500 labels and other may only have 30) I try to use the over-sampled method. To be specific, for all the images in each batch size I guarantee that every class will have at least one image. That also means the batch size must larger than the class number.

3.2. Main results

As shown in Tab. 2, I set two baseline that are ResNet50 and Linear probe CLIP-B/16, which score is at 15.28% and 22.87% respectively. After using the regularization and some tricks mentioned in the Sec. 3.1, I get the highest score on test dataset at about 34.60%.

Additionally, from the Tab. 2, we can see that the proposed architecture MatchCLIP can actually work well, from 22.87% to 27.35% (+4.48) on test dataset, which is a huge increment. The reason is that after rich and high-level features extracted by CLIP, the network can explicitly match between object and its corresponding relationship through the Matching block. The text information gives the network more power to deal with the implicit relation classification task.

Model	Mean Recall@3 on Eval ↑	Mean Recall@3 on Test ↑
ResNet50	15.49	15.28
Linear probe CLIP-B/16	23.03	22.87
MatchCLIP-Base	27.03	27.35
MatchCLIP-Large	28.52	29.09
MatchCLIP-Large†	31.90	32.12
MatchCLIP-Huge†	**33.21**	**34.60**

Table 2. Main results of various model. † denotes that balance the class labels using over-sampled.

3.3. Ablation studies

There are some hyper-parameters can be selected and in the Tab. 3 I show different combinations' performance. I use the MatchCLIP-Large to test different hyper-parameters. For the model architecture, if I want to make the model deeper, the result will all drop to some extent. However, if I make the model much wider, increasing the hidden size dimension in the attention module to 1600, the score can increase at about 2.19%. Additionally, after balancing the dataset, there is a huge increase for 3.38%. As a result, I add both the over-sampled strategy and width increasing to the final model, and achieve the 33.21% on evaluation dataset and 34.60% on test dataset.

Additionally, I text whether the text encoder inside the MatchCLIP really work effectively. So I abandon the text encoder and use the image token alone, just like the architecture shown in the Fig. 3(a). From the Tab. 4, we can clearly see that there is a drop if we abandon the text encoder. Although the improvement may not so powerful, it really work and will assist the model to converge to a higher extent. For the future, I may consider exploring the use of text encoder, maybe adding a contrastive loss for model to understand the meaning of relationship rather than a series of probability.

Model	Transformer Layers	Matching Block	cls number	Data balance	Hidden size D	Mean Recall@3 on Eval
MatchCLIP-Large	3	1	1	/	1024	28.52
	4	1	1	/	1024	27.40 (-1.12)
	3	2	1	/	1024	27.95 (-0.57)
	3	1	5	/	1024	26.69 (-1.83)
	3	1	1	√	1024	31.90 (+3.38)
MatchCLIP-Huge	3	1	1	/	1600	30.71 (+2.19)
MatchCLIP-Huge†	3	1	1	√	1600	33.21 (+4.69)

Table 3. Ablation studies of different hyper-parameters and setting.

Model	Text encoder	Data balance	Hidden size D	Mean Recall@3 on Eval
MatchCLIP-Large	√	√	1024	31.90
MatchCLIP-Large‡	/	√	1024	31.73 (-0.17)
MatchCLIP-Huge	√	√	1600	33.21
MatchCLIP-Huge‡	/	√	1600	33.06 (-0.15)

Table 4. Ablation studies of text encoder performance.

4. Conclusion

In this paper, I explore the PSG classification task and view it as a object-relation matching problem. According to this intuition, I build the MatchCLIP based on CLIP and ViT to adapt the PSG problem. I use CLIP as a pre-training model to extract both image and text features, and due to explicitly encode the relation, it can query the relationship between different objects automatically. It achieves 34.60% mean recall@3 on the PSG classification competition leaderboard.

Although it achieves a pretty nice score finally, it still meets some problem partly because of the time-limited. 1) The input of CLIP text encoder is fixed. The idea behind that is because if I choose the groundtruth I cannot use the same model for evaluation and test. For this architecture, it degenerate to a weak-supervised task. However, fixing the text input space like I do (the text space is all the classes) may constraint the model performance to some extent. 2) The output is still a batch of probability and doesn't contain any reality information, which limit MatchCLIP's performance.

However, I still believe the attempt for explicitly encoding the text information into the network is meaningful and worthwhile. Until the last few days, I start realizing the limitation for my network. I have some ideas about how to deal with the above two points and attach my improved architecture in the Fig. 3.

References

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 2

[2] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018. 4

[3] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020. 4

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2020. 2

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[6] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10965–10975, 2022. 1

[7] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 4

[8] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? *Advances in neural information processing systems*, 32, 2019. 4

[9] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 1, 2

[10] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 2

[11] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019. 4

[12] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 3

[13] Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. Multimodal few-shot learning with frozen language models. *Advances in Neural Information Processing Systems*, 34:200–212, 2021. 1

[14] Jiarui Xu, Shalini De Mello, Sifei Liu, Wonmin Byeon, Thomas Breuel, Jan Kautz, and Xiaolong Wang. Groupvit: Semantic segmentation emerges from text supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18134–18144, 2022. 2

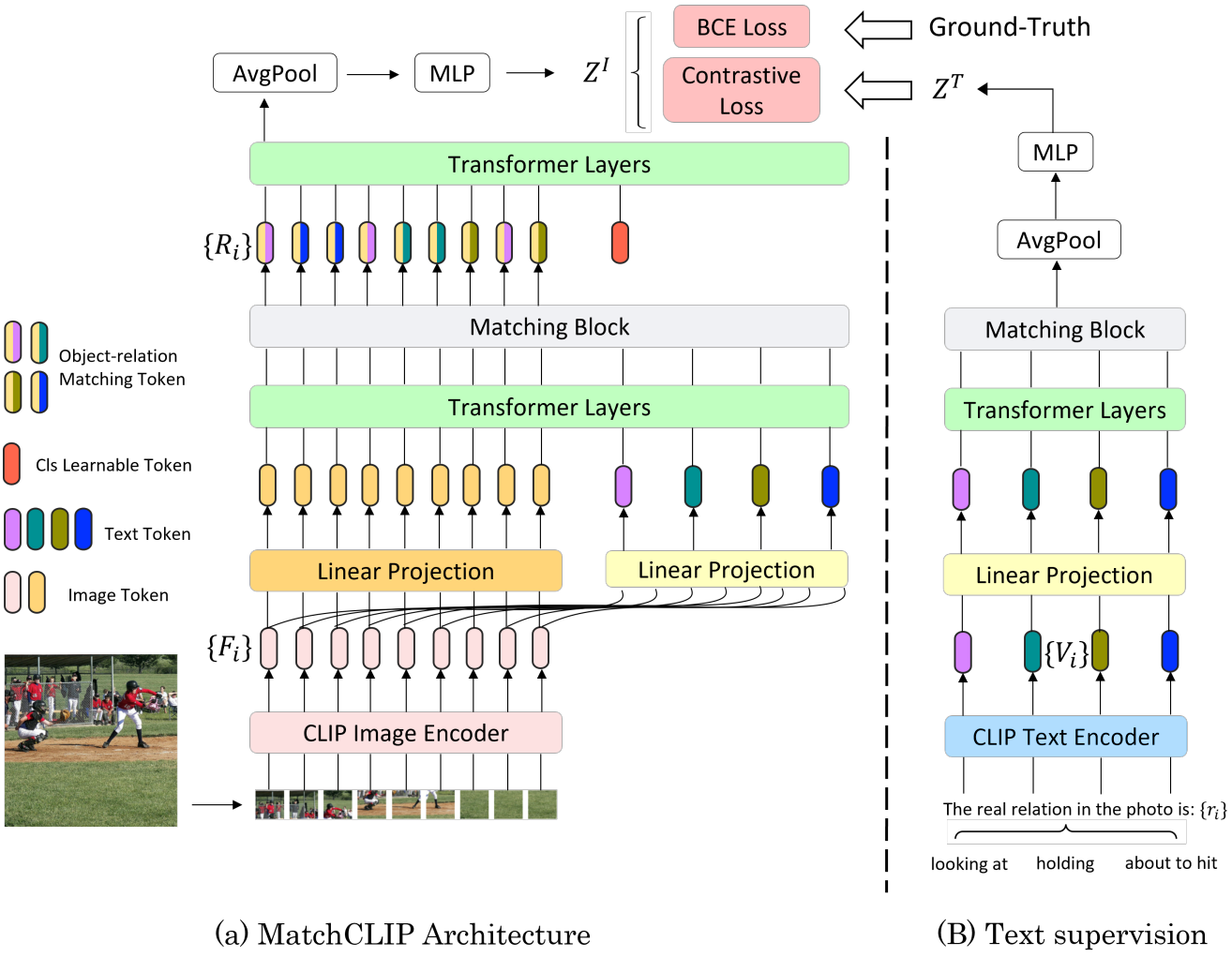


Figure 3. (a) **MatchCLIP Architecture.** It makes the minimum modification from the original one. I remove the text encoder to the outside so it only works during the training. The purpose is that I want the image to gradually learn the relation in several layers and I make the loss function more powerful including normal BCE loss and contrastive loss. The reason is that I want the output from the image can connect with the text and also not degenerate to a weak-supervised problem so I add the text supervision outside only in the training process. (b) **The Architecture of text supervision.** It follows the original text encoder at all. I want the input for the final MLP to be much richer so I use AvgPool to make full use of all the tokens rather than sample only one token.

21	G2201013J	14	10/12/22	34.59922 (20)

Figure 4. The screenshot of the leaderboard.