CLIP, SAM, and Position: Exploring referring expressions with Vision Transformers.

Ben Peterson
Boise State University
benjaminpeterson@boisestate.edu

Abstract

Pretrained vision transformers have given us the power to harness the abilities of training on large amounts of data. We used this ability to create four different models to test aspects of grounded semantics using the refCOCO dataset. While the main goal was to determine the effectiveness of the Vision Transformers, we found that we misjudged some of the importance of the factors that lead to increased accuracy in the task of picking the correct objects given a referring sentence. We find that additional research into the context of an object will be a benefit to the future of training accurate models.

1 Introduction

Transformer models have had a revolutionary effect on the computer science ecosystem since their introduction in 2017 (Vaswani et al., 2017). One of the many outcomes of this revolution is the vision transformer (ViT). Several popular ViTs have been released by companies like Meta and OpenAI. They are able to use their considerable resources to train these models and then allow the public to utilize them.

Even with these advances, current models still only seem to have a superficial level of understanding as we perceive them. Children point to things, and then others give those objects descriptive words. This links the two together. Current models do not capture information in this same way (Kennington, 2023). Finding a way to incorporate physical objects into the ability of models to learn language could prove invaluable.

In this paper, we utilize two ViTs, logistic regression, and positional features to train a set of models. Each of these models is then used to test whether they can predict which object is being referenced given a descriptive sentence. We found that by combining these tools, we could achieve better accuracy than with any single tool. In addition, the use of the prebuilt ViTs allowed us to use a single GPU to train the models in a reasonable amount of time.

In the following sections, we will explain some of the history of semantic grounding and the role of ViTs. We will also touch on some of the current work that is happening in this area. We will then present our method and the results from our tests. Finally, we will offer some insights on what these results mean going forward.

2 Background and Related Work

Recent advancements in Natural Language Processing (NLP) have shown that, given enough text and parameters, you can create a Large Language Model (LLM) that imitates the ability to respond in a syntactically correct manner for the majority of responses. Still in question is what level of understanding LLMs actually have. Is there any actual learning taking place? In particular, when we communicate, our words have meaning and represent something. For those words that have a physical representation, we ground them in something that you can touch or display an image of. LLMs and computer programs do not do this the same way that we do, an issue known as the Symbol Grounding Problem (Harnad, 1990).

Current research still shows that even models that are trained to use both text and images still have issues with grounding tasks (Parcalabescu et al., 2020) (Marcus et al., 2022). The computing systems are missing important information since they cannot see the world the same way that we can (Kennington and Natouf, 2022). Computers can be trained to detect objects using transformers, (Carion et al., 2020) but they still lack the precision that would be achieved by a human subject. Suglia et al. (2023) concluded that interactive games may be the best way to learn about grounded language, as this is a method we often employ in learning ourselves. Despite all of the research, the issue of symbol grounding still exists.

Newer models, such as Segment Anything Model (SAM) (Kirillov et al., 2023) and Contrastive Language-Image Pre-Training (CLIP) (Radford et al., 2021) have provided the ability to manipulate and encode visual information without spending time training large models. This allows us to focus on the link between how a language is grounded and try out new methods.

3 Data

For our data, we used the refCOCO dataset (Kazemzadeh et al., 2014). The images come from the ImageCLEF IAPR image retrieval dataset (Grubinger et al., 2006). The referring expressions for this dataset were collected as part of a game where one player would create a referring expression based on an outlined area of an image that contained an object. A second player would then click on an object in the image. Points were provided if the player writing the referring expression was able to get the other player to select the correct object. This provided a robust set of referring expressions, as they were verified by the players of the game.

The refCOCO data set is conveniently divided into a train, validation, and test split. This same split was used for all four experiments. The refer API [1] was used to load and access the data. The refer API allows for easy access to information on each object. Each object contains the image it came from, a set of referring expressions, an annotation ID, and abounding box. All objects are given object ids for easy reference.

4 Model & Approach

Our model was trained using four different approaches. Each approach used a different way to represent the images as a feature vector. The summary of the model creation is shown in Figure 1. The general steps for each approach consisted of the following process:

1. Load the referring expressions and images from the training set.

2. Modify the images using the current approach. A separate model was created for each approach.

3. Compute a feature vector for each modified image.

4. Store the feature vector at the word level for each sentence that refers to the object. When additional feature vectors are found for each word, they are appended to the existing feature vectors. These are the positive image feature vector examples for each word.

5. Find negative feature vector examples for each word. A negative example is any word that is not the selected word. Negative examples were picked at random. The number of negative examples selected was a multiplier of the number of positive examples.

6. Create a new feature array for each word. The positive examples are paired with an array of ones, while the negative examples are paired with an array of zeros.

7. Create a logistic regression classifier for each word using the feature arrays. A Scikit Learn classifier[2] was used with an inverse regularization of 0.25 and maximum iterations of 1000.

This process is based on the words-as-classifiers model described by Schlangen et al. (2016). Each approach below is the image modification described in step 2 and is used to create the feature vector in step 3 of the above process.

4.1 Approach 1: Sub-image using CLIP and Position Features

For each set of referring expressions, a bounding box is provided that allows you to determine which object they are referring to in the image. This bounding box was used to crop the image to a sub-image just containing the referring object. The resulting image is then processed using CLIP. CLIP produces an encoded version of the image using its ViT-B/32 pretrained model. The CLIP vector is then concatenated with the results of a positional vector. The positional vector calculates a set of seven values that represent the sub-image location in the bigger image. The values gathered include the area of the sub-image, relative positions of the x and y coordinates, distance from the center, and ratio of the size of the sub-image to the larger image.

[1] `https://github.com/lichengunc/refer`

[2] `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html`

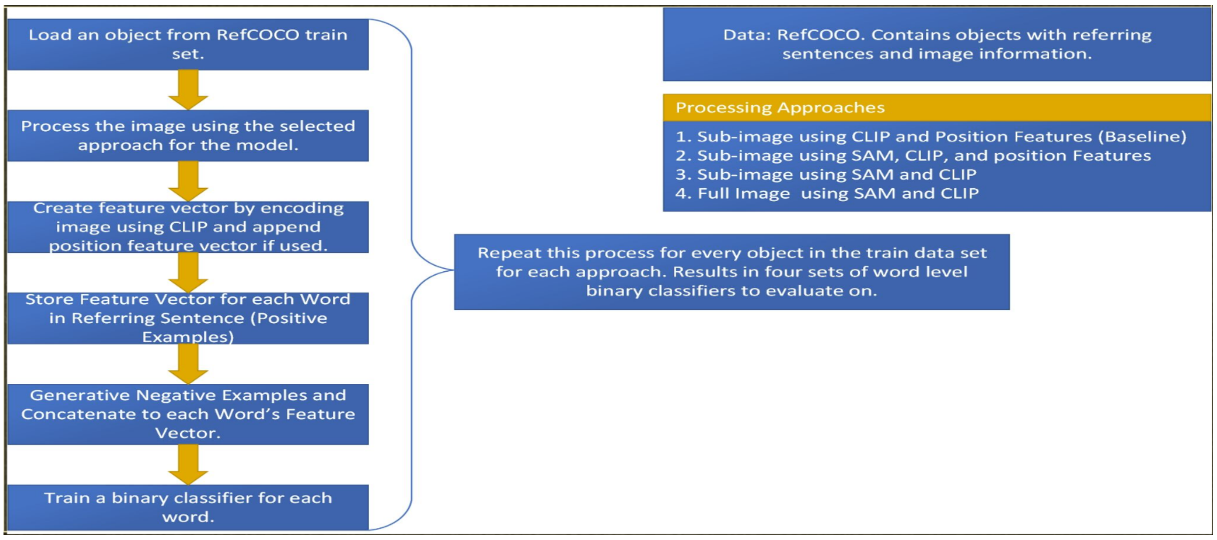
Load an object from RefCOCO train set.
Process the image using the selected approach for the model.
Create feature vector by encoding image using CLIP and append position feature vector if used.
Store Feature Vector for each Word in Referring Sentence (Positive Examples)
Generative Negative Examples and Concatenate to each Word's Feature Vector.
Train a binary classifier for each word.
Repeat this process for every object in the train data set for each approach. Results in four sets of word level binary classifiers to evaluate on.
Data: RefCOCO. Contains objects with referring sentences and image information.
Processing Approaches
1. Sub-image using CLIP and Position Features (Baseline)
2. Sub-image using SAM, CLIP, and position Features
3. Sub-image using SAM and CLIP
4. Full Image using SAM and CLIP

Figure 1: The process used to train each model. This process was repeated for each approach to produce four models.

4.2 Approach 2: Sub-image using SAM, CLIP, and Position Features

This approach uses the CLIP plus positional feature vector as described above, with one modification. Before the sub-image is created, the image is masked using SAM. SAM is trained to be able to create a mask around an object in an image based on a given prompt. For our prompts, we used a bounding box provided by the referring expression and a point at the center of the bounding box. The mask was then used to turn the picture white at all locations except where the mask was present. This new image was then used to create the feature vector using CLIP and the positional features. The process of how we determined to use this mask is described in Section 4.5.

4.3 Approach 3: Sub-image using SAM and CLIP

In this approach, we created the sub-image the same way as in approach 2, with the SAM-generated mask and CLIP. We did not add the positional features to each sub-image.

4.4 Approach 4: Full Image using SAM and CLIP

In this approach, we used SAM to create a mask, but we did not generate positional features or create a sub-image. The entire masked image was passed to CLIP to generate the feature vector.

4.5 Mask Selection

In creating the approach for each model, one of the main decision points was how to select the correct mask for the image using SAM. We were given an original image with a bounding box (figure 2). Our initial attempt used just a single point at the center of the object. When the resulting images were examined, we found that the mask was too focused and often only picked up a part of the object (Figure 3). We also tested using three points on the object, one at the center and two that were in separate corners of the object. This approach improved the masks, but also picked up extra noise, and initial tests showed it had poor accuracy (Figure 4). Since a bounding box was provided by the refCOCO data set, we used that box and a center point. We found that this produced the most accurate mask of all the methods (Figure 5).

5 Evaluation

After a logistic regression classifier was prepared for each word, we could then use those classifiers to make predictions. Accuracy was used as the single measure for predictions. All training and tasks were performed on a single GPU computer, and most of the training took less than 24 hours for each approach.

Task & Procedure The goal for our model is to be able to accurately pick an objects referring sentence, given all of the objects in an image. Each of the four approaches was picked to see the effect of SAM and positional features on accuracy.

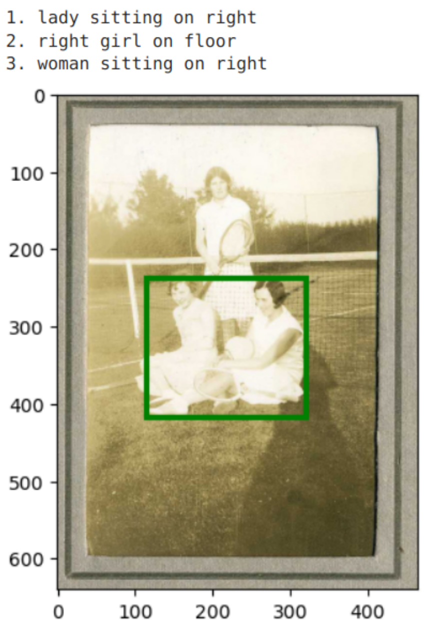

Figure 2: The original image with its bounding box and referring sentences.

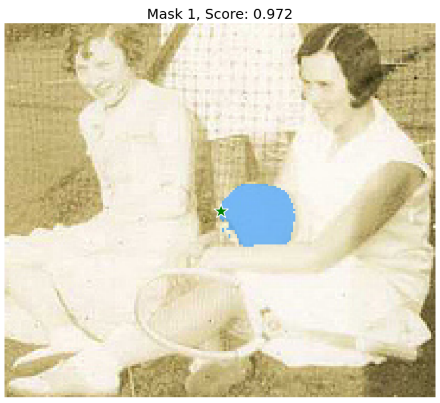

Figure 3: The masks selection using 1 central point, which does not capture the entire subject.

To test the accuracy of our model, we step through all of the objects. For each object, we store its annotation id as the gold standard and load its referring sentences. We then process all of the objects that are located in the larger image, including the object that we are currently referencing. To process each object, we create a feature vector using the same approach that was used to train the model.

With all of the objects processed, we step through each referring sentence. For each sentence, we step through all of the object feature vectors and calculate a probability for each word, using the logistic regression classifier and the feature vector. The probability for each word in the sentence is multiplied together and stored for each object. We then pick the object that has the highest probability. If the ID of the object selected by the prediction matches the gold standard, it is awarded an accuracy point.

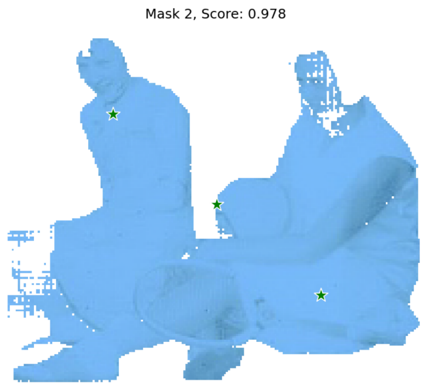

Figure 4: The masks selection using three points. We mask the subject but also include the neighboring subject.

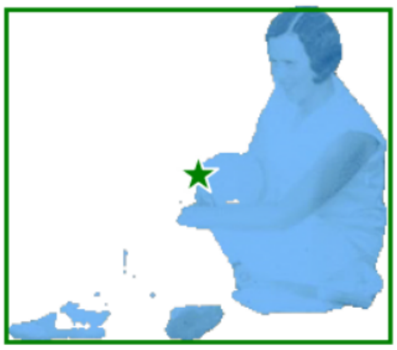

Figure 5: The masks selection using bounding box and center point. We see that this does the best job of masking the image.

Metrics & Baseline We report accuracy as the single measure for our tests. The ability to pick the correct object given a referring expression lends itself naturally to accuracy.

As our baseline, we picked approach 1 as described in Section 4.1. This approach has been used in the past in Boise State CS 536, and our implementation of this approach was verified by the instructor of the course. With a verified baseline, we could then safely compare it to the other methods. During our creation of Approach 1, we did calculate a random baseline. The random baseline had an accuracy of 15.7%. We felt that using a baseline with a more rigorous standard would allow us to better compare our results.

Approach	Accuracy
1 - Sec 4.1	64.86%
2 - Sec 4.2	**67.98%**
3 - Sec 4.3	47.88%
4 - Sec 4.4	58.26%

Table 1: Accuracy for each approach and the section where they are described. Approach 1 is the baseline using the sub-iamge with CLIP and position features. Approach 2 is a sub-image using SAM, CLIP, and position features. Approach 3 is a sub-image using SAM and CLIP without positional features. Approach 4 is the full image using SAM and CLIP.

Results The results of each test are found in table 1. We found that Approach 2, a sub-image using SAM, CLIP, and positional features, yielded the best accuracy of 67.98%. We can see that both SAM and the positional features have an effect on the accuracy. For SAM, we see that accuracy is improved as the difference between approach 1 and approach 2 is the addition of processing the image with SAM to remove the background noise. This was a small increase in accuracy, but it shows potential. While the mask selection process was reviewed for a handful of images, there were some images that did not have accurate masks, and better masks could increase the accuracy.

For positional features, we see a significant drop-off in approaches 3 and 4, where positional vectors are not added. For approach 3, the sub-image provides no reference to its position. This results in images that have a large area filled with a mask, and not enough distinguishing features are found. The large images in Approach 4 remedy this, as the mask is much smaller compared to the entire image and has some reference to where it is located in the image. Even with this, it is still unable to achieve the accuracy level of approach 2. In our tests, the positional vector had the greatest effect on accuracy.

6 Implications

In this project, we set out to learn about the effect of white space on an image and came up with some unexpected results. The one that was most interesting to us was the importance of positional features. We knew that they were important, but when removed, they made more difference than masking the object. This may be a factor due to the refCOCO dataset, as a lot of the referring expressions do use directional terms when referencing the objects. In the full image test (Section 4.4), we did not add the positional vector, but it did still approach the accuracy for the baseline. This appears to show that some positional information was gathered from its location in the full image. All of this points to the fact that a more careful consideration of the positional features could be beneficial in increasing accuracy.

While performing the task of finding the correct mask, we also considered how objects are considered in context. This was illustrated with the positional features in the sub-image tests and the object location in the large image test, as described above. There is one major difference between our tests and a real-world scenario. In the real world, the image is not just surrounded by white space. The things that are around the object give it context. Context is important. If someone is looking at a bear, their reaction to that bear is going to be very different depending on whether they are at a zoo or in the middle of the woods. With our SAM-processed images, that context is lost, and valuable information along with it. So while the location of the object is important, the context also matters. The small increase in accuracy gained from processing the image with SAM reinforces this concept. We need to understand how to incorporate context, which is no easy task.

The other observation worth noting is the information needed to create the model. A bounding box and central point for the mask were taken from the refCOCO dataset. We also needed that information for the positional features. While object detection is getting better, it still would not have provided as much information as the refCOCO dataset gave us. We also used two ViT models that had extensive training and our own model training. These models could not run in real time in the same way that we can process and select an object from an image. There is still a long way to go in this regard.

7 Conclusion

In this project, we tested two different ViTs and positional features. We found that the ViTs were invaluable, as the amount of training that was used would not be possible with our resources. It allowed us to present four different models that could be compared to one another. The ViTs did assist us in getting improved accuracy, but the thing that was not expected was how important the positional features would be in the determination of accuracy. It is worth taking an additional look at what other positional features could be included.

This leads us to the idea of the importance of context. Future work could include a close look at how we might be able to use more features that look at context. The removal of white space was one way to contextualize the object, but it was by subtraction. The results from the removal of the position features show what could be gained by adding more features that could enhance the model in an additive way.

References

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. *CoRR*, abs/2005.12872.

Michael Grubinger, Paul Clough, Henning Müller, and Thomas Deselaers. 2006. The iapr tc12 benchmark: A new evaluation resource for visual information systems. *Workshop Ontoimage*.

Stevan Harnad. 1990. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1):335–346.

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. 2014. ReferItGame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 787–798, Doha, Qatar. Association for Computational Linguistics.

Casey Kennington. 2023. On the computational modeling of meaning: Embodied cognition intertwined with emotion.

Casey Kennington and Osama Natouf. 2022. The symbol grounding problem re-framed as concreteness-abstractness learned through spoken interaction. In *Proceedings of the 26th Workshop on the Semantics and Pragmatics of Dialogue - Full Papers*, Dublin, Ireland. SEMDIAL.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. 2023. Segment anything. *arXiv:2304.02643*.

Gary Marcus, Ernest Davis, and Scott Aaronson. 2022. A very preliminary analysis of dall-e 2.

Letitia Parcalabescu, Albert Gatt, Anette Frank, and Iacer Calixto. 2020. Seeing past words: Testing the cross-modal capabilities of pretrained v&l models. *CoRR*, abs/2012.12352.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020.

David Schlangen, Sina Zarrieß, and Casey Kennington. 2016. Resolving references to objects in photographs using the words-as-classifiers model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1213–1223, Berlin, Germany. Association for Computational Linguistics.

Alessandro Suglia, Ioannis Konstas, and Oliver Lemon. 2023. Visually grounded language learning: a review of language games, datasets, tasks, and models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.