ICTAM - Image-Captioning Tactical Advisor Model

Andrew Jeon
University of Washington ECE
Paul Allen Center, 185 E Stevens Way NE, Seattle, WA 98195
ajjeon2@uw.edu

Abstract

In this paper, I introduce a novel tactical advisor agent that uses image captioning to analyze StarCraft minimap images to provide strategic analysis in the form of word captions. Unlike Reinforcement learning approaches which focus more on actually solving the game, I focus more on the general problem of strategic, spatial-semantic understanding. I fine-tune, train GIT Generative Image-to-text Transformer, on 700 annotated StarCraft minimap image-caption pairs. This dataset was hand-annotated by myself and one other StarCraft player. I then evaluate ICTAM's (Image-Captioning Tactical Advisor Model) performance on 23 unseen test images. A quantitative evaluation will be done by calculating the % of how many test images ICTAM correctly identifies the winner or "Draw" for each frame of gameplay. A qualitative evaluation is also displayed by comparing GIT-base captions on test images and ICTAM's captions on test images.

1. Introduction

Real-time strategy (RTS) games such as StarCraft were one of Artificial Intelligence's first significant challenges after the "Deep Learning Explosion" of the last decade. These multi-agent, partially observable, fast-paced, complex environments represent unique challenges for agents to navigate. Research in this field had focused on achieving superhuman performance through reinforcement learning and imitation learning from expert replays. A significant milestone in this field is AlphaStar [6], which achieves grandmaster-level play by combining supervised learning and a colliseum of reinforcement learning agents competing against and learning from each other. CICERO [7], achieved human-level strategic negotiation in the board game Diplomacy, being one of the first works to integrate strategic reasoning with semantic understanding.

Recent advances in foundation models have sparked a new wave of interest in applying vision-language models to these games. The work *Large Language Models Play StarCraft II* [5] showed that pre-trained foundation models, guided with prompts and context, can generate competent strategic summaries and suggest tactical actions, showing the potential foundation models have as tactical advisor agents.

Inspired by all of this work, I propose a lightweight tactical advisor that operates directly on StarCraft minimap images. Instead of requiring full game-state access or structured inputs, my system applies Image-Captioning through a fine-tuned GIT-base [8] and a fine-tuned BLIP-base-image-captioning [4] to extract spatial-semantic patterns from raw minimaps to generate tactical analysis captions. This includes identifying who has more bases, who has map control, who is attacking who, and generally who has the advantage frame by frame.

In the real-world, tactical decision making, is often left to officers and commanders, sometimes aided by Command and Control systems. Needless to say combat is an incredibly stressful environment, which sometime impedes sound decision-making as seen throughout history time and time again. My approach bridges the gap between perception and lightweight tactical decision-making. Real combat scenes such as radar scans, drone images will be more complex visually. Additionally, transformer language models are pretrained on next-token prediction as we learned in class, so in a way next-token prediction isn't truly tactical "understanding". But what we consider artificially intelligent is still very much in flux and currently, generative decoder architectures represent the cutting-edge, and this is why ICTAM uses vision-language models as of now. If I can prove that a lightweight tactical AI analyst can understand StarCraft minimaps, then it would be a step towards proving that AI analysts can understand combat scenes visually and semantically. This in turn would be a step towards live tactical advising in real-world deployment. Real-world systems would likely incorporate among many other things, sensor fusion of cameras, radar, Lidar, and other sensors to have as accurate of a world state as possible. Real-world systems would certainly need to be robust across many modalities instead of only image and text. Nevertheless, ICTAM represents a first iteration towards artificially intelligent command and control systems to aid tactical decision-makers in highly stressful environments.

2. Related Works

GIT (Generative Image-to-Text Transformer for Vision-to-Language Tasks [8] was one of the first unified generative transformers that joined multiple vision-language tasks such as image captioning and VQA. The architecture was also simpler than other methods which often included external modules. GIT was pretrained with a Language Modeling loss where the objective is to predict the next token given the image and prior tokens. The model is essentially an image encoder ViT paired with a GPT text decoder, along with some sequence2sequence attention masking. It concatenates the vision encoder tokens to the text token, so the decoder can use the image tokens as context. I used Microsoft's git-base-coco which is a smaller variant of the original GIT [8] trained on 10 million image-text pairs and fine-tuned on the COCO Captions dataset [1]. Ultimately I chose git-base-coco due to its smaller size which is important for my use case due to the small size of my dataset and lightweight models being more practical for potential edge deployments in future tactical autonomous systems.

BLIP (Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation) [4] is a unified VLP framework that has a unimodal encoder, image-grounded text encoder, and a image-grounded text decoder. These 3 different "modes" train on 3 different pre-training objectives. Image-Text Contrastive Loss is for the unimodal encoder which uses cosine similarity to align positive image-text pairs, while distancing negative pairs. Image-Text Matching Loss is for the image-grounded text encoder which uses multimodal representation learning to help with fine-grained vision language alignment.Lastly, Language Modeling Loss is for the image grounded text decoder which generates a textual description given an image. I used Salesforce's blip-image-captioning-base which is a smaller variant of the original BLIP, tailored for image captioning.

Other options that were considered include BLIP-2 and VideoBLIP [3], these may be used later on if I am able to build more data and transition to video captioning as opposed to image captioning. BLIP-2 (Bootstrapping Language-Image Pretraining 2) [3] was a revolutionary pre-training strategy that bootstraps pre-training from existing frozen pre-trained image encoders and language models. Its main innovation was the Querying transformer Q-Former which is pre-trained twice. First the Q-Former learns to query visual features from a frozen image encoder using contrastive training. Second, The Q-Former output tokens are passed into a frozen LLM and the whole model is trained to generate matching text. This helps the Q-Former learn to extract useful visual features that aid in language generation while the language model remains frozen. BLIP-2 is basically an efficient pre-training strategy that bootstraps vision-language representation learning from a frozen image encoder and bootstraps vision-to-language generative learning from a frozen language model. The Q-Former is essentially learning "learnable queries" that say to the frozen encoder give me the x most important things about this image that will help the LLM describe it.

Overall, vision-language, image-video-captioning models are cutting edge architectures and what I believe to be the best architectures for the task of limited visual-semantic combat scene understanding. ICTAM will be an iterative work, where once new architectures come out and have potential to achieve true tactical "understanding," ICTAM can be reiteratively built with them.

3. Methods

3.1. Data Processing Pipeline

I utilized yt-dlp [9], an open-source command-line audio-video downloader to download 44 youtube videos, without audio, containing about 800 minutes of StarCraft 1vs1 gameplay. At first, I was considering using the audio of the commentators to help with annotation, but I quickly realized that the commentators often discussed other unrelated topics which made this not viable.

Next, I used [2] to sample one frame every 60 seconds from each video. I did this because ICTAM will only advise on minimap frames; for this high-level tactical advice we are analyzing gradual movements that happen over at least 10s of seconds, so I thought 1 frame was very roughly representative of the minimap state for 1 minute. Admittedly, this is a bit of stretch, and with more compute and annotation manpower, I would likely have sampled at least 2 or 4 frames per minute. Due to the extremely low sampling rate of 1 frame per 60 seconds, it is likely that some tactical movements will be missed even though this is very high level strategic modeling that we are doing. I also cropped the images to only include the minimap portion, as this sort of lightweight tactical advisor would likely not perform as well on the visual complexity of a full game state as opposed to the relatively simple visual complexity of just the minimap. This decision was made because, once again, StarCraft the game has more or less already been solved [6], and ICTAM is more focused on tactical language analysis of visual high-level movements. After this, I hand-cleaned strange null frames or frames where a commentator drew on the map as these samples would likely inhibit performance. After all this, I went from 800 frames to 723 frames of image-caption pairs.

On the caption side, I created an annotation.json file where myself and one other StarCraft player hand annotated

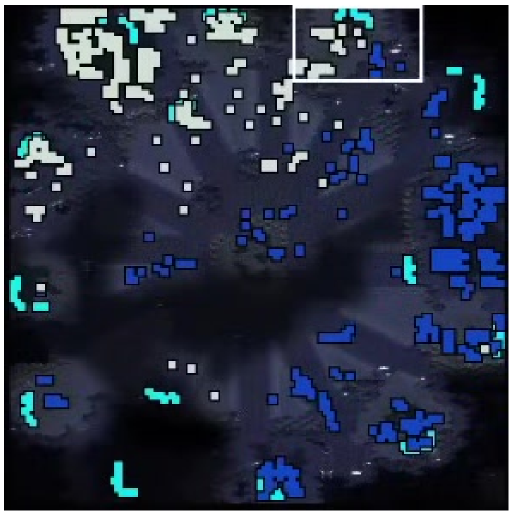

Figure 1. Example Image Sample

all 723 samples. Between the two of us we have a couple decades of StarCraft experience, and back when I played the game in my youth I achieved ranks that represented the top 20th percentile of players. I decided this level of expertise was sufficient to analyze and annotate these StarCraft minimaps. Captions follow a consistent structure where they are formatted as an answer to the question, ”Who is winning? Why?” Captions start with a player color or ”Draw” followed by a period. This acts as a determination of who is winning, this would allow ICTAM to learn not just to describe, but to make a tactical determination.
example ground-truth-caption: ”Red. Red has map control and has three bases while blue only has two.”

3.2. Training

To train ICTAM, I fine-tuned the GIT-base-coco model a variant of [8], pretrained on 10 million image-text pairs, and the blip-image-captioning-base model [4], pretrained on 130M image-caption pairs, using HuggingFace Transformers. The pre-training from the CLIP vision encoders is sufficient to give the models spatial and color recognition which is mostly all that is needed for our minimap tactical reasoning task (minimaps are not that complex, just colored dots on a map).

Due to the small size of my dataset (723). I use 700 image-text pairs to supervised train/finetune both these models and tested/evaluated them on the remaining 23 image-text pairs which I considered my test set. I used the HuggingFace built-in trainer to train my base models and save my finetuned models. I experimented with different batch sizes, epochs, and max encoding lengths. Epochs and batch sizes did not impact the result very much, while longer and shorter encoding lengths had varying results. Overall, a max encoding length of 50 tokens had good results and was most appropriate for our captions of 1-2 sentences. Doing a quick check with byte-pair encoding methodology would put this at 1-2 sentences (15-20 words - 50 tokens) which is reasonable.

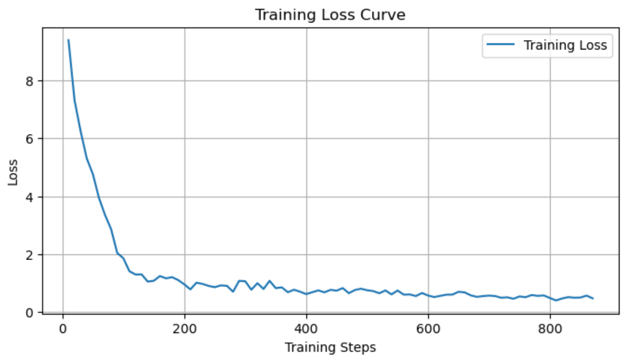

Figure 2. Caption for Git-finetuned

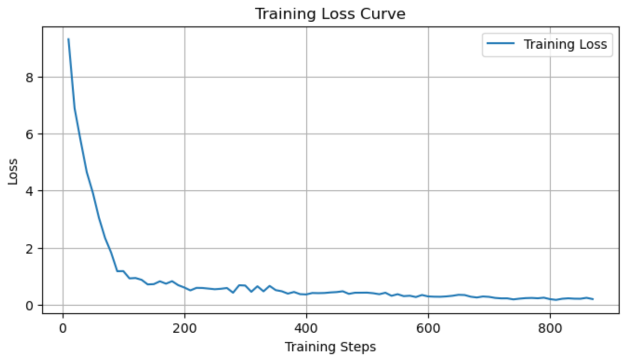

Figure 3. Caption for BLIP-finetuned

4. Experiments

4.1. Baseline Models

Of course, a baseline is necessary to validate that our additional fine-tuning training even accomplished anything! Thus, I conducted experiments where I generated a caption from ICTAM. Below, in figure 2 is the result of running git-base-coco and BLIP-base-image-captioning with no fine-tuning. As expected, the vision encoder is already able to correctly recognize shapes and colors but the language model caption output shows no sign of tactical reasoning or language similar to our GT captions.

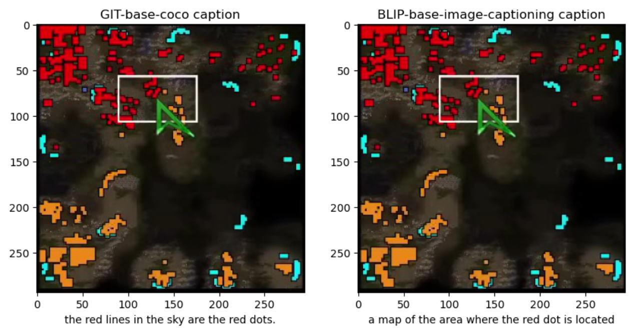

Figure 4. git-baseline and blip-baseline captioning results

4.2. Fine-Tuned Models

My finetuning of both git-base-coco and blip-image-captioning-base seems to have been moderately effective. The decoder generated captions now somewhat resemble my ground truth annotations. Overall, blip-finetuned performed much better than git-finetuned, which was expected. blip-image-captioning-base is $\tilde{2}$50M parameters while git-base-coco is $\tilde{1}$50M parameters. Nonetheless, both finetuned models went from baseline captions where only color, and spatial features were commented on to captions where bases are being counted and large armies in aggressive positions are being recognized. Overall, the captions sound reasonably similar to the ground truth captions. In some cases for GIT only, it went from gibberish to tactical-sounding gibberish. Qualitative examples of this improvement and accompanying images are show below:

git-base: ”the red lines in the sky are the red dots.”
blip-base: ”a map of the area where the red dot is located”
git-finetuned: ”. has red five while has, is a, attack, red s base is to up three. draw”
blip-finetuned: ”orange. orange has five bases to blue ’ s three and is attacking with a large army.”

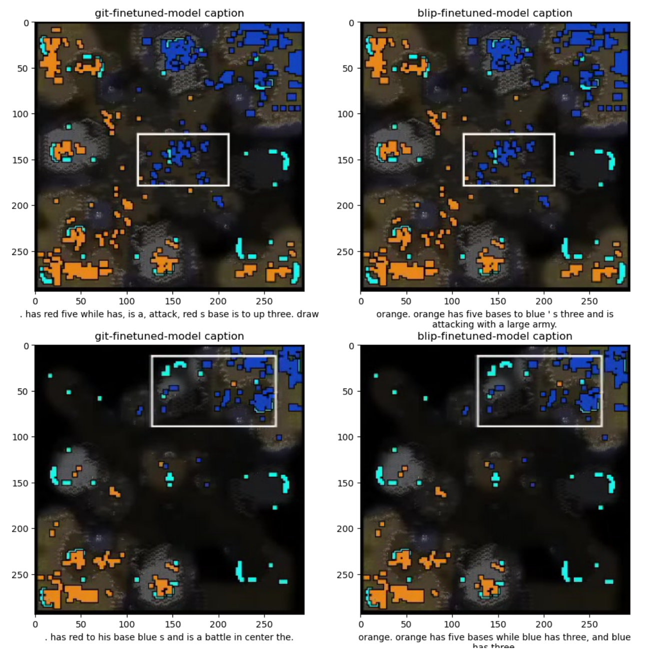

Figure 5. git-finetuned and blip-finetuned test captioning results

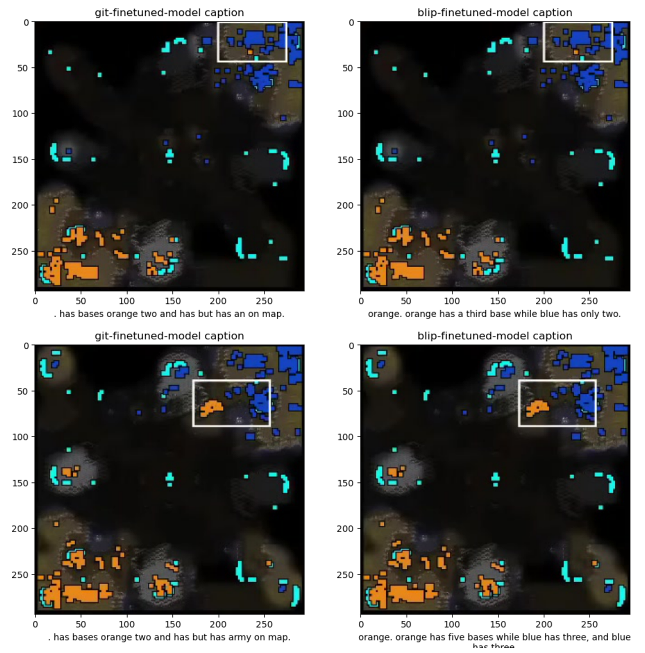

Figure 6. more git-finetuned and blip-finetuned test captioning results

4.3. Evaluation

$$WER = \frac{S + D + I}{N} = \frac{S + D + I}{S + D + C}$$

Model	WER (lower better)	TJA (higher better)
git-finetuned	0.946	0
blip-finetuned	0.922	0.65

Table 1. Evaluation of fine-tuned models on WER and TJA metrics

Symbol	Description
S	Number of substitutions
D	Number of deletions
C	Number of correct words
N	Number of words in GT

Table 2. WER terms

I first attempted evaluation with WER Word Error Rate as that is what is used in academia to evaluate image captioning tests. WER is defined by the equation above where basically all word-level errors are divided by the number of words in the GT. So, 0 is the best and 1 is the worst. I realized this metric is literally a 1 to 1 comparison between ICTAM's predicted caption and our ground truth annotation caption. This is not quite a fair comparison for ICTAM due to the obvious limits imposed by the scale of our dataset and the compute available to us. But more than this, our desired output is tactical reasoning, we don't want ICTAM to output the ground truth caption word for word, the ground truth captions are merely one human experts analysis of the scene, we want ICTAM to have its own analysis of the scene, not just output an exact copy of the ground truth caption every time. My models achieved high WERs, as our captions were not close word to word matches for our ground truth annotations, due to this another metric closer to our use case is needed.

In order to have a more reasonable quantitative error, I developed Tactical Judgement Accuracy or TJA to evaluate how well ICTAM is judging "who is the winner" of the current image or map scene. I did this by splitting the text output at the first "." for all 23 test image caption predictions and ground truth captions. So basically, I compared two dictionaries of {image: "draw", image2: "blue"...} and so on. I define accuracy as the number of matches or "correct judgements" divided by the number of test samples or 23. A formal derivation of TJA can be found below:

$$TJA = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}[\hat{y}_i = y_i] \tag{1}$$

Blip-finetuned made the correct tactical judgement 65% of the time and followed the caption structure 100% of the time. Git-finetuned made the correct tactical judgement 0% of the time due to not being able to follow the caption structure where the first word before the "." must be a color or draw. Blip-finetuned followed the caption structure on 100% of test images while git-finetuned followed caption structure on 0% of test images. This could be due to git-finetuned being a smaller, less powerful model and also our dataset not being large enough. Blip-finetuned, however showed an impressive amount of improvement and generated quite impressive captions. Despite all the aforementioned limitations in data and compute, blip-finetuned correctly recognized the colors of both players on every test frame, correctly recognized who had more presence on the map, and correctly recognized large clusters of dots as large armies. This is an impressive level of tactical understanding, given all the limitations ICTAM faced. However, blip-finetuned struggled to count exactly how many bases each player had but was able to determine who had "more" in a general sense on every test frame. Blip-finetuned also struggled with determining which players army or base it was talking about, but this was likely due to ground-truth caption wording which had some unclear use of possessive pronouns.

5. Discussion

A simple next step would be to scale up the dataset which would likely significantly improve performance on both variants of ICTAM. I could also do some additional hyperparameter tuning. Aside from this, the next step for this work is Video Captioning. Strategic reasoning benefits massively from having a temporal dimension. Strategic understanding of an image or a scene is valuable, but understanding over multiple frames or instances in time is an essential part of making an AI tactical advisor sufficiently capable in real-world deployment. This would enable an AI Tactical Advisor to understand how a combat scene is changing over time. A VQA iteration of Video-ICTAM or VCTAM could potentially answer questions such as "Who is winning?", "Why are they winning?", "Will they continue winning, or will the tides turn?" An approach I am considering is altering my data pipeline to divide the videos into 30-60 second clips, I would then sample 4-8 frames per clip to add in a temporal aspect to our data, and have one caption per every 4-8 frames describing the scene tactically similar to my ICTAM data. For video captioning, I would probably use VideoBLIP or just regular BLIP-2 on multiple frames. Beyond video-captioning, pairing minimap data with game camera images or videos attached with an object detection head would enable ICTAM to learn unit counts, different races, and semantically describe how a battle is going with these exact unit counts and race descriptions. This would would be the starcraft equivalent of pairing real life aerial, top-down, radar data with on-the-ground visual data from an autonomous robot. Real-life command and control systems likely have similar inputs such as this.

References

[1] Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollar, and C. Lawrence Zitnick. Microsoft coco captions: Data collection and evaluation server, 2015. 2

[2] FFmpeg Developers. FFmpeg. https://github.com/FFmpeg/FFmpeg, 2024. 2

[3] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models, 2023. 2

[4] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation, 2022. 1, 2, 3

[5] Weiyu Ma, Qirui Mi, Yongcheng Zeng, Xue Yan, Yuqiao Wu, Runji Lin, Haifeng Zhang, and Jun Wang. Large language models play starcraft ii:benchmarks and a chain of summarization approach. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 133386–133442. Curran Associates, Inc., 2024. 1

[6] Michaël Mathieu, Sherjil Ozair, Srivatsan Srinivasan, Caglar Gulcehre, Shangtong Zhang, Ray Jiang, Tom Le Paine, Richard Powell, Konrad Żołna, Julian Schrittwieser, David Choi, Petko Georgiev, Daniel Toyama, Aja Huang, Roman Ring, Igor Babuschkin, Timo Ewalds, Mahyar Bordbar, Sarah Henderson, Sergio Gómez Colmenarejo, Aäron van den Oord, Wojciech Marian Czarnecki, Nando de Freitas, and Oriol Vinyals. Alphastar unplugged: Large-scale offline reinforcement learning, 2023. 1, 2

[7] Meta Fundamental AI Research Diplomacy Team (FAIR), Anton Bakhtin, Noam Brown, Emily Dinan, Adam Fisch, Naman Goyal, Mike Lewis, Julian Michael, Tim Rocktäschel, Da Ju, et al. Human-level play in the game of diplomacy by combining language models with strategic reasoning. *Science*, 378(6624):1067–1074, 2022. 1

[8] Jianfeng Wang, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. Git: A generative image-to-text transformer for vision and language, 2022. 1, 2, 3

[9] yt-dlp Developers. yt-dlp: A youtube-dl fork with additional features and fixes. https://github.com/yt-dlp/yt-dlp, 2024. 2