ARL-TR-10239 ● Dec 2025

DEVCOM ARMY RESEARCH LABORATORY

# Using Large Language Models (LLMs) to Summarize Mission Data for Mission Planning and After Action Review (AAR)

by Benjamin T. Files and Kimberly A. Pollard

## NOTICES

### Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

ARL-TR-10239 ● DEC 2025


# Using Large Language Models (LLMs) to Summarize Mission Data for Mission Planning and After Action Review (AAR)

**Benjamin T. Files and Kimberly A. Pollard**
*DEVCOM Army Research Laboratory*

# REPORT DOCUMENTATION PAGE

| 1. REPORT DATE | 2. REPORT TYPE | 3. DATES COVERED | |
|---|---|---|---|
| | | START DATE | END DATE |
| December 2025 | Technical Report | October 2023 | September 2025 |

**4. TITLE AND SUBTITLE**
Using Large Language Models (LLMs) to Summarize Mission Data for Mission Planning and After Action Review (AAR)

| 5a. CONTRACT NUMBER | 5b. GRANT NUMBER | 5c. PROGRAM ELEMENT NUMBER |
|---|---|---|
| **5d. PROJECT NUMBER** | **5e. TASK NUMBER** | **5f. WORK UNIT NUMBER** |

**6. AUTHOR(S)**
Benjamin T. Files and Kimberly A. Pollard

| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) | 8. PERFORMING ORGANIZATION REPORT NUMBER |
|---|---|
| DEVCOM Army Research Laboratory<br>ATTN: FCDD-RLA-IB<br>Playa Vista, CA 90094 | ARL-TR-10239 |

| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) | 10. SPONSOR/MONITOR'S ACRONYM(S) | 11. SPONSOR/MONITOR'S REPORT NUMBER(S) |
|---|---|---|
| | | |

**12. DISTRIBUTION/AVAILABILITY STATEMENT**
DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.

**13. SUPPLEMENTARY NOTES**
ORCIDs: Benjamin T. Files, 0000-0002-1141-7886; Kimberly A. Pollard, 0000-0002-5849-1987

**14. ABSTRACT**
U.S. Army systems use sensors and computer algorithms to generate and capture large amounts of complex data. These data could support generation of reports, graphics, and summaries for after action reviews, team-to-team communication, and other purposes. Here, we describe a conceptual solution that would use automated mission summaries to convert relevant data into natural language text narratives. We propose a hierarchical approach that converts low-level events and observations into sentences and leverages the ability of large language models to effectively summarize text to produce narratives at an appropriate level. In pursuit of this goal, we explored candidate methods for generating first-level sentences from raw mission data gathered from an interactive simulator exercise. We focus on two key classes of mission data: firing events and vehicle formations. Results with firing events demonstrate the general principle that a language model can convert event data into natural language sentences with an accuracy that matches a human, although the nature of its errors differed from that of a human. Spatial reasoning to recognize formations is not a strength of language models. We arrived at a system that could reliably recognize categories of formations, but we caution that the system was neither flexible nor likely to generalize.

**15. SUBJECT TERMS**
large language model, LLM, transformers, machine translation, spatial reasoning, human/autonomy interaction, Military Information Sciences, Humans in Complex Systems

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES |
|---|---|---|---|---|
| a. REPORT | b. ABSTRACT | C. THIS PAGE | | |
| UNCLASSIFIED | UNCLASSIFIED | UNCLASSIFIED | UU | 39 |

| 19a. NAME OF RESPONSIBLE PERSON | 19b. PHONE NUMBER *(Include area code)* |
|---|---|
| Benjamin T. Files | (520) 691-6441 |

**STANDARD FORM 298 (REV. 5/2020)**
*Prescribed by ANSI Std. Z39.18*

## Contents

## List of Figures



## List of Tables

# 1. Introduction

U.S. Army systems use sensors and computer algorithms to generate and capture large amounts of complex data. For example, running an exercise on a military simulator generates gigabytes of data with a moment-by-moment record of the position and velocity of each entity on the simulated battlefield, potentially including data streams such as voice data recorded from communication, state information about every crew station and actuator, high-resolution physiological data from each human participant, firing events, hit events, damage events, and more. Real (as opposed to simulated) systems do/could generate and store similar amounts of data. These data could support generation of reports, graphics, and summaries for after action reviews (AARs), team-to-team communications, and other purposes.

To use and understand this high-volume, high-complexity data, Warfighters and researchers need specialized tools and knowledge. Custom-built data dashboards could be used to convey important elements of the data to a person with the expertise to interpret them. Simulators can provide a replay of the events recorded in the data, but the user would still need to select a relevant perspective from which to observe the results and would need to know what visual and nonvisual data are available for further scrutiny. Furthermore, if used in a real-world mission planning or command and control operation, the information presented in these computationally heavy visualizations may be unwieldy to pass along to lower or upper echelons. Here, we describe a conceptual solution that would instead use automated mission summaries to convert the relevant data into a natural language text narrative.

Summarizing complex data with natural language has many advantages. Natural language summaries can be read and edited by users without requiring any special skills. Natural language can abstract and compress information, allowing it to be efficiently transmitted. Many existing tools can work with natural language as inputs for the purposes of, for example, presentation, summary, linking, and search.

Automated text-based summary is enabled by the advent of so-called large language models (LLMs), which produce naturalistic text. Previous, non-LLM, approaches to natural language summaries have been successful; however, they require extensive handcrafting of rules and other processes to generate naturalistic text outputs (Gordon 2016; Gordon and Wang 2021). A benefit of an LLM-based summary system is that instead of demanding complex hand-coding, it requires only a natural language prompt.

In addition, LLMs can avoid the brittleness that often plagues rule-based, hand-coded algorithms. The flexibility and generalized capabilities of an LLM-based summary-generating system could be a major strength, allowing it to summarize events with novel elements. For example, a sufficiently flexible system could accommodate changes in doctrine, unit composition, mission, and capabilities without requiring a substantial reprogramming effort. This flexibility would allow the system to adapt to rapidly developing capabilities and tactics of both friendly and enemy forces.

Here, we report on efforts to take complex mission data gathered in a mounted platoon simulator exercise and develop candidate approaches for automatically generating useful summaries of these data. Once achieved, this foundational capability could enable several future applications. The need to flexibly team heterogenous systems at scale under communications limitations introduces a critical requirement for concise, understandable summaries of a given element's current situation and understanding. Automated LLM-driven summaries could lessen the burden on humans to generate these summaries and could effectively allow the machine-based systems to participate in a natural language dialogue with their human teammates. The proposed approach could be a step toward further enabling human–machine system integration.

Although the focus here is on generating narrative summaries for human use, the capability of machines to parse and extract information from text suggests that machine systems could use these summaries, too. In general, for systems to communicate with each other, they need some protocol for sharing information. Historically, machine–machine communication is accomplished through formalized protocols that facilitate sharing particular kinds of information in well-specified formats. Well-known formats typically seek some trade-off between human- and machine-readability (e.g., JavaScript object notation [JSON], extensible markup language [XML]). As technologies that enable machines to work with natural language mature, natural language could become the optimal communication protocol for both human and machine elements of mixed human–machine teams.

Our proposed approach is hierarchical, with the ultimate goal of at least two levels of textual representation. First, low-level data would be converted into sentences, resulting in a large collection of sentences with far more detail than is required or useful. Subsequently, this overly detailed text-based representation would be summarized into a shorter form. This summary process would then be iteratively repeated until a sufficiently short summary is produced. The strengths of this proposed method would be its flexibility to new types and categories of information

and its ability to be implemented without needing much—or any—hand-coding by an experienced computer programmer.

In pursuit of this hierarchical summarization goal, this technical report explores candidate methods for generating first-level sentences from raw mission data. We focus on two key classes of mission data: firing events and vehicle formations. These are mission-critical elements not only for AAR and future mission planning, but also for real-time reporting and successful execution of team missions.

## 1.1 Related Work

The advent of LLMs has created opportunities to use natural language where it was previously impractical. Several efforts have explored the use of LLMs for tasks related to mission summarization (Jin et al. 2024), including document summarization and timeline summarization.

Kurisinkel and Chen (2023) took a hierarchical approach to summarizing multiple documents, first using LLMs to extract the main message from each document, identifying an overall main message and context for the group of documents, and then passing the main message and context statements to a summarizing LLM. Evaluation was based on coherence ratings from four human experts, all finding that the hierarchical LLM approach worked well.

Sojitra et al. (2024) took a multistep hierarchical approach to timeline summarization with LLMs based on large sets of news articles, exploring sequential chunking, incorporating knowledge graphs, and preprocessing temporal statements. They found that these approaches had strengths relative to the state of the art, demonstrating the utility of LLMs for timeline summarization. Evaluation used ROUGE, which automatically evaluates summaries given one or more human-generated reference summaries.

These efforts and the ones reviewed in Jin et al. (2024), show that the general approach of hierarchical summary generation with LLMs is effective. These works take as their inputs documents (news articles, journal articles, blog entries, etc.), that were written by humans for humans to read. As such, the source documents might have structure and form, including emphasizing important information, that are missing from current machine-generated simulator event logs.

Related work explored using event logs, rather than human-written documents, as sources for building coherent narrative summaries from LLMs. One prominent example is in root cause analysis following a failure or error in a complex information technology ecosystem. Root cause analysis involves parsing through large volumes of computer-generated records to identify the events that caused

some security incident. An LLM-based agent that could use tools to query systems for relevant logs and then access summaries of those logs (also generated by an LLM) was found to be an effective tool for root cause analysis (Roy et al. 2024). This bears important similarities with the objective of the current work, as it involves concise summaries generated from large volumes of data that may not all be pertinent to the specific question at hand and are not already structured by a human author.

Other work has used LLMs to extract structure from computer event logs having an unknown structure. Ma et al. (2024) introduced an unsupervised process for discovering the structure in event logs to extract the relevant data into a structured form, using local-only LLMs. This process was effective, and the architecture could be used to discover structure and form in event logs for new kinds of events.

In summary, the emergence of LLMs and related technologies has led to a proliferation of work that can be leveraged to approach the problem of automatically summarizing mission data from military simulators and other contexts.

## 1.2 Enabling Technologies

Small, offline language models are a crucial component for the proposed approach. Current cloud-deployed language models have impressive capabilities, and many leading cloud-deployed models can and have been deployed in secure environments that maintain the privacy and security of sensitive data. Considering the longer-term, if summary tools are to be used in operational settings, the data would be even more restricted, and cloud computing resources could not be assumed to be available (i.e., due to bandwidth and electromagnetic limitations) or sufficiently secure. With that in mind, preliminary work with the proposed approach uses open-weight, smaller-scale, offline models that can be run on modest local hardware. Although developers continue to explore the effects of larger models, others seek to maximize the performance of smaller models (Wang et al. 2024). Bigger models tend to be better overall; however, smaller models can outperform larger ones, especially for a fixed training budget (Hoffmann et al. 2022). Smaller models can also be distilled from larger ones to achieve superior performance (Hsieh et al. 2023). Most relevant for the present work, smaller models can be deployed on a laptop with a consumer-grade graphics accelerator for real-time/interactive uses, making them suitable for austere and disconnected environments.

Retrieval-augmented generation (RAG) is another enabling technology for this work. Language models are computation- and data-intensive to train. RAG is a different approach that can induce language models to ground responses with

reference to specific documents without retraining the models (Lewis et al. 2020). RAG searches a library of documents for those that are relevant to a specific query and includes those documents in the context of the query to the language model. For the present work, we used RAG to include relevant examples of converting events into plain text. This necessitates a small number of handcrafted examples, which can be generated by a subject matter expert (SME). While the requirement for expert-labeled data may limit the immediate generalizability of the approach, these labeled examples can be generated ahead of time, and their use does not require the language model itself to be retrained or fine-tuned.

# 2. Assessments

As an initial assessment of the viability of the proposed approach, we conducted two small-scale experiments. The first generated sentences from firing events recorded during a simulator-based mounted platoon exercise. The second generated statements about formation movement from position data for a platoon of simulated vehicles.

## 2.1 Firing Events

### 2.1.1 Method

A convenience sample of 30 firing events were selected from a mounted platoon exercise that took place in an interactive simulator. Firing events were recorded as structured data with fields defined in Table 1. These identify the shooter, the weapon/ordnance, and what the shot hit—all using simulator-specific identifiers. An SME provided natural language sentences that corresponded to each event.

**Table 1. Fields and example data for a firing event.**

| Field name | Example |
|---|---|
| attacking_vehicle_class | RED_RCVLPawn |
| attacking_vehicle_name | RED_RCVLPawn |
| AttackingVehicle_AttackingGunnerType | RCVLGunnerCrewStationPawn |
| AttackingVehicle_AttackingWeaponName | IWSWeaponComponent |
| AttackingVehicle_ProjectileType | M2 |
| victim_vehicle_class | RCVMPawn |
| victim_vehicle_name | RCV5 |

Note: The provided interpretation for this example is, "An OPFOR RCV-Light hit a BLUFOR RCV called RCV5 with an M2 weapon."

Each event was used as a probe, resulting in 30 generated sentences. A prompt was generated by first retrieving the top-5 most similar examples among the 29 other events using the average normalized Levenshtein distance between the field values

of the entry in the library to those of the probe. These examples and the probe were inserted into the following prompt:

> I am going to give you some examples of data entries. Each one has an interpretation. Please look carefully at the examples:
>
> {examples}
>
> Now I will give you a challenge with a blank interpretation:
>
> {probe}
>
> Please provide an appropriate interpretation for this challenge entry.
>
> Your interpretation must identify the attacking vehicle and the type of weapon used. Your interpretation must also identify what was hit, always using the verb 'hit'. If the shot hit a BlockingVolume, Landscape, or StaticMeshActor, then the shot is considered a miss.
>
> Vehicle names, indicated by the fields 'attacking_vehicle_name' and 'victim_vehicle_name' that start with 'RED' are OPFOR vehicles. Vehicle names that do not start with 'RED' are BLUFOR vehicles. Terrain elements are neither OPFOR nor BLUFOR.
>
> Do not guess or infer information beyond what is in the data entry. Do not expand or define any acronyms or abbreviations.

This prompt was used along with system instructions, “You are an assistant who turns computer dicts into plain language.” For full text of an example prompt used, see Appendix A.

The prompt and instructions were provided to Mistral 7b Instruct v0.2, quantized to 5 bits. This model was used because it operated well on local hardware and was recently released at the time of this experiment. To keep outputs concise and easy to process, the model was given a JSON output format requirement consisting of a single field labeled “interpretation.” Generation used temperature 0.0 and frequency penalty 0.0 to encourage dry straightforward outputs.

### 2.1.2 Results

To evaluate the results, outputs were compared to the SME-produced sentences and to the source firing event data. In 24 out of 30 events, there was no material difference between the SME-produced and LLM-produced sentences, although they differed in word order and phrasing. The remaining six events were not necessarily errors, but they are worth considering further. In two cases, the LLM-produced sentence was better aligned to the event data due to errors in the SME-produced sentences. Two others differed because they introduced additional information that was not in the event data itself. In one of these cases, the LLM-sentence identified the section of a BLUFOR vehicle, even though section

information was not requested nor was it available in the input fields. The section identity of that vehicle was, however, given in the SME-generated sentence about a different firing event and was based on the SME's personal knowledge. In the other case of added information, the LLM-sentence added information about the weapon ("non-explosive ordnance"), which is correct but superfluous. The remaining two differences were clearly errors. In one, the LLM-sentence identifies the terrain as BLUFOR, which is nonsensical, and in the other the sentence identifies a BLUFOR vehicle as an OPFOR vehicle, which is a critical error. In summary, the LLM-generated sentences were accurate in 28/30 cases, which matches the human SME's accuracy. However, unlike the human's errors, one of the errors committed by the LLM was a critical error of team membership.

### 2.1.3 Discussion

These results, while small-scale, demonstrate the general principle that a small, local language model can convert event data into natural language sentences with an accuracy that matches a human, although it made errors a human would likely not make. These results are promising, but they are perhaps not surprising given the limited domain. Indeed, a handcrafted rule to generate these sentences might not be very difficult to create, as a simple template-filling approach would likely succeed. As such, this demonstration shows that the LLM-based method can handle this relatively simple, but important and widespread, use case.

One main motivation of the proposed approach to generating text-based narratives is the ability to adapt to new kinds of information and new systems without extensive hand coding. The prompt in this demonstration involves handcrafted text meant to guide the model to produce correct, grounded sentences. While we did not conduct a formal exploration of the effects of different prompts on the results, this prompt was crafted after several iterations of less specific prompts achieving less success. This may reduce the plug-and-play flexibility of this method for different contexts. However, the basic approach, followed by modest prompt engineering to introduce task-specific desiderata (e.g., always using the verb "hit" in the generated sentences) is likely still more flexible than a purely rule-based, hand-coded solution. While the programming skill to build a rule-based system to generate acceptable sentences might be uncommon, so-called prompt engineering is also a skill that goes beyond simply putting a request into text.

While the error rate of the LLM was the same as the error rate of the human SME, the type of error committed by the LLM differed from the type of errors committed by the human. Notably, the LLM hallucinated information not present in its specific input, and it mixed up grammatically similar and frequency-similar words that nonetheless had vastly different meanings. In this particular case, the LLM's

hallucinations (superfluous ordnance properties and unrequested section information) happened to be accurate and thus were not counted as errors. They nonetheless would qualify as contextual inconsistency hallucinations (Cossio 2025), wherein the model provides information not found in the input or context. Future efforts with larger samples would better elucidate how often the LLM's hallucinations happened to be correct versus incorrect. For each of the responses coded as clear errors, the LLM filled in a word that was the right part of speech, syntactically and semantically plausible, and often present in the training data. However, in both cases, the resulting sentence as a whole was inaccurate. This sort of error is common in LLMs, as their next-token–guesser behavior recognizes or mimics high-frequency patterns without deeper understanding (Wu and Deng 2025; Cossio 2025). Indeed, hallucination is considered inevitable in computable LLMs (Xu et al. 2025; Cossio 2025). In situations where accuracy is critical, the output of any LLM must be interpreted with care and reality-checked by knowledgeable humans. This is no less the case here than it is in other potentially high-stakes applications of LLMs, such as those in medical, legal, or financial domains (Cossio 2025).

Another caveat is that the results may not generalize beyond the context in which they arose. We used a quantized, relatively small model with 7b parameters, selected to run on available local hardware. A newer, larger model running on more capable local hardware could very well produce better results with less prompt engineering. The continuous emergence of new models with associated claims of new, better capabilities means that any finding becomes stale very quickly.

While the results were relatively accurate, this demonstration represents a relatively easy case. In Section 2.2, we address a more difficult task, one that challenges spatial reasoning.

## 2.2 Formation Recognition with Multimodal Models

Understanding the spatial relationships among vehicles in a section or platoon (i.e., their formation) provides important context about the unit's intents and beliefs, so formations are highly relevant for AAR. To support automated formation recognition as part of narrative summary, we endeavored to develop a pipeline for writing natural language sentences describing the relative positions of vehicles in a platoon in terms of identified formations.

As new vehicles and capabilities emerge, questions of how they should be incorporated in existing or new formations arise as well. One of the demonstration's goals was to be flexible to the emergence of potential new formations, such that

formation descriptions could be generated even if there were not existing doctrinal descriptions of the observed formations.

Understanding the relative positions of objects in space is a part of spatial reasoning, which is not a strength of language models (Bang et al. 2023; Cohn and Hernandez-Orallo 2023; Li et al. 2024). As such, this application represents a much greater challenge for LLM capabilities than the firing event translation. In this report, we discuss several approaches. Many of these were unsuccessful. We eventually arrived at a system that could reliably recognize formations, but we caution that the system was neither flexible nor likely to generalize.

### 2.2.1 Method and Results

The general approach was to define some prototypical formations and handcraft descriptions of the positions of the vehicles as they would be represented in simulator data, which includes heading and position information for each named vehicle. For initial efforts we used a four-vehicle platoon, in keeping with traditional armored platoons, but subsequently we focused on six-vehicle platoons, as were used in an available testbed simulator. The generated descriptions were then provided to the model to see if it could recover the correct formation name. Most approaches did not reliably provide the correct formation name and thus did not clear this initial evaluation. The one approach that did was then challenged with imperfect versions of the prototype formations before challenging it with real data.

### 2.2.2 Text-only Approach

For this approach, a human converted the spatial coordinates of the vehicles into definition statements about the positions of vehicles relative to the leader, which was defined as the vehicle farthest forward in the direction of travel. Position descriptions were relative to the leader with respect to the direction of travel, so a vehicle might be described as “behind and to the right” of the lead vehicle. The formations defined in this approach were four-vehicle formations in existing armored vehicle doctrine, and the descriptions reliably differentiated among column, staggered column, wedge, line, echelon left, and echelon right. These descriptions bear similarity to those used by Yamada et al. (2024), who found that even the most advanced modern LLMs have greater difficulty answering spatial questions about verbally described triangular arrangements of objects, or other diagonal line shape arrangements, than with square arrangements. The full prompt, including the descriptions used for each formation, appears in Appendix B.

In this approach, we used the same text model as in the firing event work; namely, Mistral 7b Instruct v0.2, quantized to 5 bits, with temperature 0.0, and frequency penalty 0.0. We used the system prompt, “You are a helpful assistant who answers

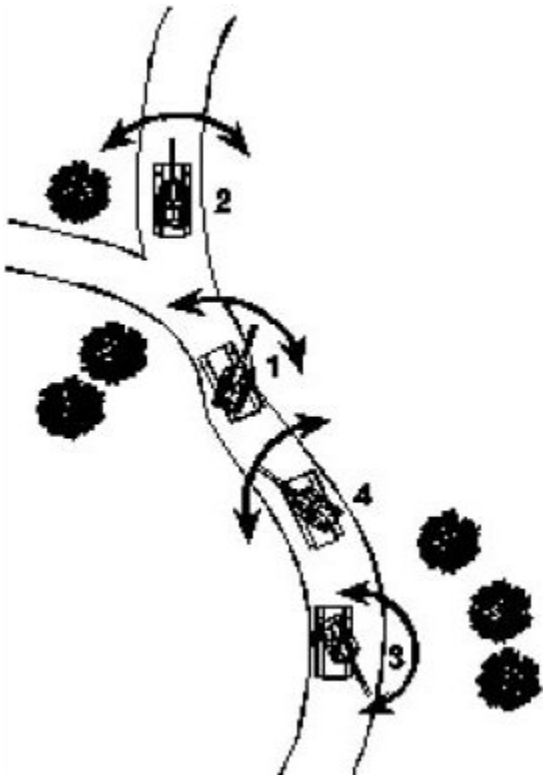
questions based only on the given information," and a JSON output format. The system was given a prompt with the definitions of the formations in terms of relative positions of the vehicles and a probe using one of those definitions with different vehicle names. This text pattern-matching approach was successful except that it consistently mislabeled staggered column formation as echelon left. This confusion is perhaps understandable, as the verbal definitions of these two formations were the most similar, with only the verbal descriptions of V3 and V4 differing—and differing via the presence of words and phrases that otherwise appear elsewhere within both definitions. The descriptions of the V3, for example, differed only in whether they were "to the left" or "far to the left." The subtle distinction between "far" and "[no qualifier]" appeared to be lost on the LLM, as did the presence of an additional "to the left" in the V4 description.

While some evidence suggests that careful verbal descriptions can lead an LLM to an internal approximation of spatial understanding for some shapes (Wu and Deng 2025), it should be understood that an LLM's behavior relies on statistical pattern matching rather than causal understanding or logical reasoning (Cossio 2025), so a preference for verbal similarities and nearby words (Huang et al. 2025) might be expected to overshadow any internal spatial representation. While different wordings for initial formation definitions could be attempted, it seemed likely that the approach would be brittle to novel formations that similarly differed in subtle ways. Therefore, we moved on instead to a more visual approach using a multimodal visual question answering model.

### 2.2.3 Multimodal Models

Visual question answering models accept an image and a text prompt as inputs and can generate responses that account for both the image and the text prompt. We tried several iterations of the general approach of asking the model to describe or name the formation appearing in an image.

As a first attempt using llava-v1.6-mistral-7b.Q5_K_M, we provided doctrinal diagrams of column and staggered column formations and asked for descriptions of the relative positions of the vehicles in the formations. This was intended as an initial check for spatial understanding as well as a candidate method to produce text descriptions of formations that might be more suited to the LLM's capabilities than those used in the text-only approach. The prompt included the name of the formation:




> "This image is a diagram of the armored vehicle formation called Column. It shows a top-down view of the vehicles traveling North (up in the diagram). Please describe the arrangement of vehicles in the diagram with reference to the direction of travel. Describe with enough detail that someone could recognize a column formation from your description in a different image. Ignore any terrain features in your description. Curved arrows indicate the firing arc and can be ignored."

The model generated a reasonable description of a column formation:

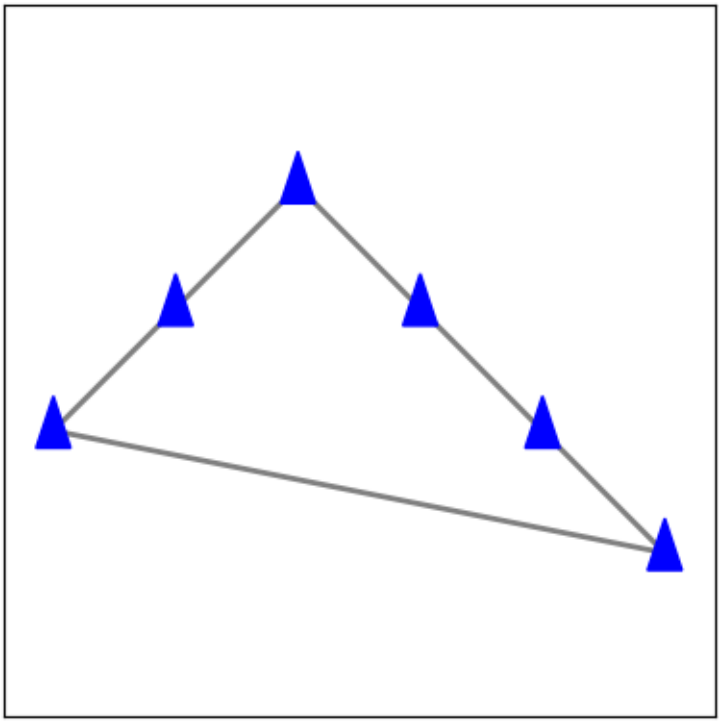
> "In the diagram, there are four armored vehicles arranged in a column formation. They are traveling in a North direction, as indicated by the arrows pointing upwards in the image. The first vehicle is positioned at the front of the formation, leading the way for the other vehicles. Following this vehicle is the second vehicle, which is slightly behind the first one. The third vehicle is positioned further back in the formation, maintaining a safe distance from the second vehicle. Finally, the fourth vehicle is at the rear of the formation, providing support and protection for the vehicles in front of it. This arrangement allows for efficient communication and coordination among the vehicles while maintaining a defensive formation against potential threats."

This text accurately describes the relative positions of the vehicles; however, it describes an arrow that is not in the image, and its comments regarding the benefits of the column formation miss the main point that column formation is primarily for fast movement on roads. Additional prompt refinement and a restrictive response format might have eliminated the extraneous information.

The model was able to produce a similar description for the staggered column formation, but it confused right and left in its descriptions of the vehicles' offsets. Because the spatial descriptions were inaccurate, this pipeline did not seem promising as a method for producing text descriptions an LLM could reliably identify. While right versus left offset in the staggered column might not be especially consequential, the confusion of right and left in differentiating echelon left and right would be a problem, as those formations imply an anticipated direction of contact that could be important for interpreting the movement of a platoon. Even rather large, state-of-the-art LLMs have demonstrated difficulty with distinguishing left and right, as well as showing greater difficulty with right than with left (Wu and Deng 2025). The reason for this unevenness may derive from greater prevalence of leftward relationships in natural language used as training data (Wu and Deng 2025). Another possibility is that it may be related to the prevalence of other uses (other meanings) of the words "right" and "left" found in the model's training or reference data.

### 2.2.4 Identifying the Formation in an Image

Finding that doctrinal diagrams did not provide a promising pipeline for generating LLM-friendly descriptions of formations, we then attempted to provide a more streamlined graphical representation of the formation along with a prompt to name the formation in the image. In this approach, we converted the relative positions of vehicles into an iconic image with small triangles indicating the vehicles' facing. In this example, the blue triangles are arranged as vehicles would be in a wedge formation.



This image shows a shape made of blue vertices and gray edges. Please describe this shape with enough detail that you could recognize it from only your description.

The response was unsatisfactory:

> The image displays a simple geometric shape consisting of a series of blue vertices connected by gray edges. The shape appears to be a two-dimensional figure with a rectangular base and a triangular top. The rectangular base is formed by four blue vertices, two on the left and two on the right, with a straight gray edge along the bottom. The triangular top is formed by three blue vertices, one at the top left, one at the top right, and one at the top center, with a straight gray edge along the top. The overall shape is flat and does not have any depth or shading, giving it a flat, two-dimensional appearance.

We attempted including multiple examples of a shape in a single image, as well as a multishot learning approach, but we did not achieve reasonable results with any of these approaches.

Although the small, offline model we used for firing event descriptions was sufficient to its task, one possibility we considered was that the small offline multimodal model was too small for a formation recognition task. As an informal check on that possibility, we submitted these abstract shape prompts to larger cloud-based multimodal models and found similarly unsatisfactory results.

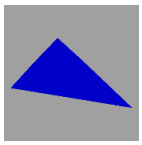
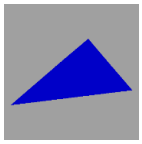
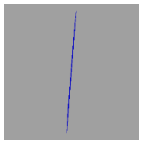
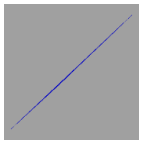
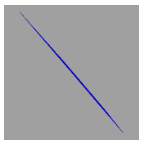
We also tried including multiple examples of each formation in a single image with a non-quantized model, llava-v1.6-mistral-7b-hf. With this model and approach, shape descriptions were mostly nonsensical.

### 2.2.5 Zero-Shot Image Classification

Zero-shot image classification is similar to visual question answering, but instead of generating a response or description, the model selects a label for the image among a set of candidate labels. We used a pretrained zero-shot classification model, siglip-so400m-patch14-384, on a single GPU. The model provides a score for each candidate label, with the label having the highest score as the chosen label. In this approach, we provided abstract geometric representations of formations (Table 2) to the model and provided formation names as candidate labels. This approach was not successful, with labels receiving mostly uniform weights. Presumably, formation names are not a part of the pretrained vocabulary, so the labels were not successfully applied.

As a second attempt, we used the same images and associated each formation with a common shape name describing the convex hull of the formation. This resulted in a many-to-one relationship, with, e.g., vee, wedge left, and wedge right all being described as a triangle (Table 2). Attempts to differentiate the orientation of the triangle were not successful; however, the linear formations of line, column, and echelon right/left were successfully mapped to a horizontal line, vertical line, or diagonal line, respectively.

**Table 2. Formation shapes.**

| Formation name | Example image | Shape name | Formation group |
|---|---|---|---|
| Line |  | Horizontal Line | Line |
| Wedge Right |  | Triangle | Wedge right or Wedge left or Vee |
| Wedge Left |  | Triangle | Wedge right or Wedge left or Vee |
| Vee |  | Triangle | Wedge right or Wedge left or Vee |
| Column |  | Vertical line | Column |
| Echelon Left |  | Diagonal Line | Echelon left or Echelon right |
| Echelon Right |  | Diagonal Line | Echelon left or Echelon right |
| Double U |  | Trapezoid | Double U |
| Staggered Left |  | Parallelogram | Staggered left or Staggered right |
| Staggered Right |  | Parallelogram | Staggered left or Staggered right |

The zero-shot formation classification approach reliably provided the correct label in all cases, with the caveat that right and left were not differentiated, and wedge and vee formations were not differentiated. An example result for a Column formation appears in Figure 1.

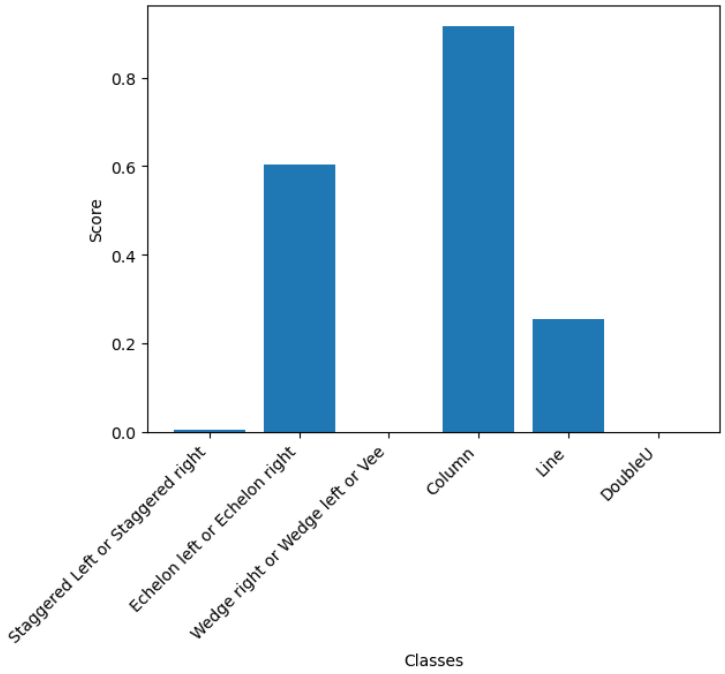



**Figure 1. Zero-shot image classifier scores for a Column formation. The correct label received the highest scores, with the other linear formations (Echelon and Line) receiving second- and third-highest scores, respectively.**

With the caveat that the system is now identifying groups of formations, it performs well. It also has the advantage that novel formations could be classified by giving a name to the shape that formation generates, and this would not require retraining or reprogramming.

### 2.2.6 Fine-tuned Image Labeling

To confirm that a fine-tuned model could, indeed, differentiate among these formations, we then fine-tuned a visual transformer model to classify formations. To support this, we fine-tuned a pretrained visual image transformer model, vit-base-patch16-224-in21k, to accurately label formation images. The model was trained on images generated using the abstract geometric approach already described (Table 2). To add variability to the images, we randomly displaced the position of each vehicle (spherical normal with standard deviation equal to 10% of the vehicle spacing distance) and the orientation of each vehicle (normal with standard deviation of 15° of vehicle heading) as well as the overall orientation of the formation (uniform on the circle). The convex hull of the formation was then

converted to a solid-colored polygon on a gray background. The model was trained on 800 example formation images (sampled uniformly among the 10 formation types). Training used a linear scheduler with the AdamW optimizer (Loshchilov and Hutter 2017) and a learning rate of 5e-5 with three epochs of 100 batches of 8 images. Evaluation on 800 new (held out) formation images had an accuracy of 99.6%.

Subsequently, we developed another image generator that produced images in a more expressive visual style that matched those used to communicate formations to human users of the simulator. This image generator also included field-of-view cones that varied with the orientation of the vehicle's main turret, with the correct/recommended turret orientation included for each vehicle in each formation. The generator also uses motion lines to convey the speed of each vehicle. At this point, we also revised the repertoire of formations to match those proposed for a six-vehicle mixed crewed/uncrewed platoon (i.e., vee, column, wedge, doubleyou, staggered column, line, and the stationary coil and herringbone). We used the same random perturbations of location and orientation described above and added random variation to speed and turret facing. Example images appear in Figure 2. To fine-tune the model, we generated labeled training examples as 10 examples each of the combinations of the 8 formations, 3 levels of position jitter (normal with standard deviation 1%, 5%, and 10% relative to vehicle spacing), facing (normal with standard deviation 5°, 15°, and 25°) turret orientation (5°, 15°, and 25°) and movement speed (5%, 15%, and 25%), for a total of 6,480 training examples. We used 30 training epochs and a batch size of 81 and otherwise kept training consistent with the previous experiment. Validation on a similarly balanced held-out set comprising four examples per randomization level for a size of 2,592 classified formations with accuracy 100%.

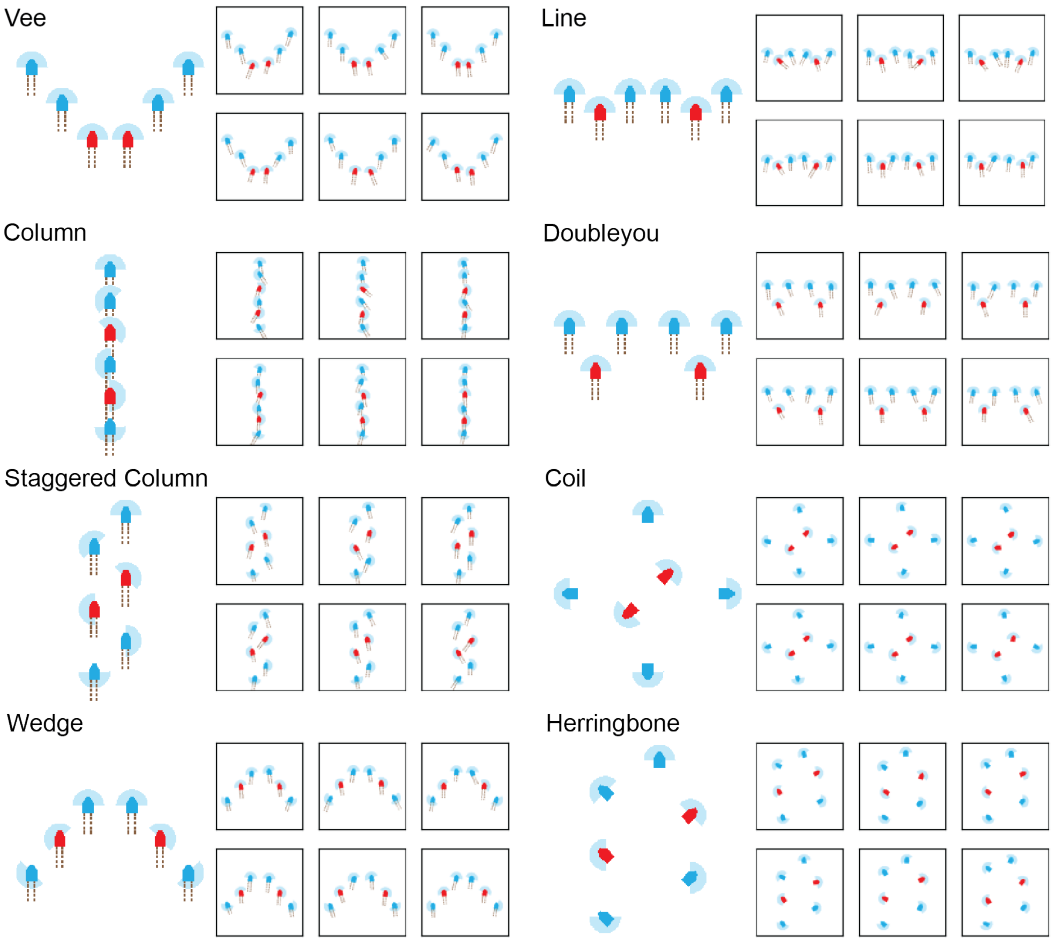



**Figure 2. Formation prototypes and examples of jittered formations. Jittered versions shown here represent the intermediate levels of jitter for position, orientation, turret facing, and speed.**

Initial efforts to apply this fine-tuned model to real data revealed that random displacement of vehicles failed to capture the naturalistic variability in formation movement, apparently due to disagreements between sections and one or more vehicle being occasionally far out of position, so we developed formation-specific variability to generate variations in the formation images that better reflected the underlying structure of the formation. For example, with herringbone and staggered column formations, we varied the amount of offset between the left and right vehicles. For column, we generated images of columns following a curved road. To add randomness to the generated formations, instead of a combination of low-, medium-, and high-variability on each dimension, we defined a single mixed normal-variability model, in which the offset had some high probability of being drawn from a normal distribution with a lower standard deviation and otherwise drawn from a normal distribution with a higher standard deviation. This approach generated a naturalistic distribution of formations with rare, large deviations. For examples of these images, see Figure 3.

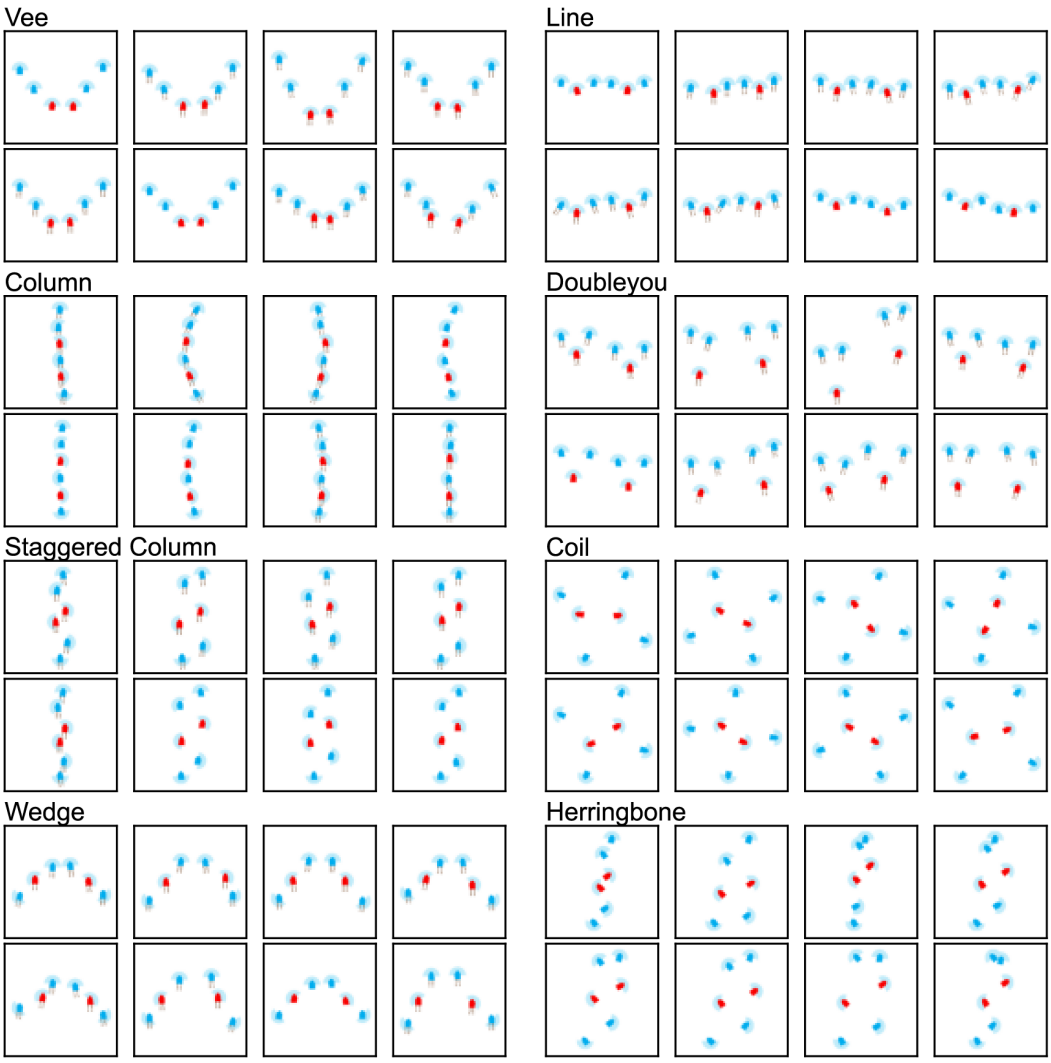



**Figure 3. Formation-specific variability. Prototype formations appear in Figure 2.**

Adjusting the noise in the synthetic formation image generator to match a target dataset risks the eventual model over-fitting to the target dataset. That possibility could be evaluated using new, unseen target data, but that work has not been done.

The same visual transformer model (vit-base-patch16-224-in21k) fine-tuned on 160 examples of each formation trained following the same strategy using these images classified a new held-out image set with 20 examples per formation with 100% accuracy.

### 2.2.7 Fine-tuned Labeling of Exercise Data

To validate that the fine-tuned image label approach functions with simulator data, we obtained data from a formation training exercise, in which a six-vehicle platoon was instructed to follow a route with a series of specific formations to use. The fine-tuned model provided a formation label for the positions of the vehicles sampled at

1-s intervals. Although the participants in the exercise were instructed to use specific formations at different phases of the exercise, this run came from relatively inexperienced crews, so their adherence to formations was not perfect. Nonetheless, the classifier identified formations that corresponded approximately to the instructed formations. Formation labels are summarized with respect to space in Figure 4 and with respect to time in Figure 5.

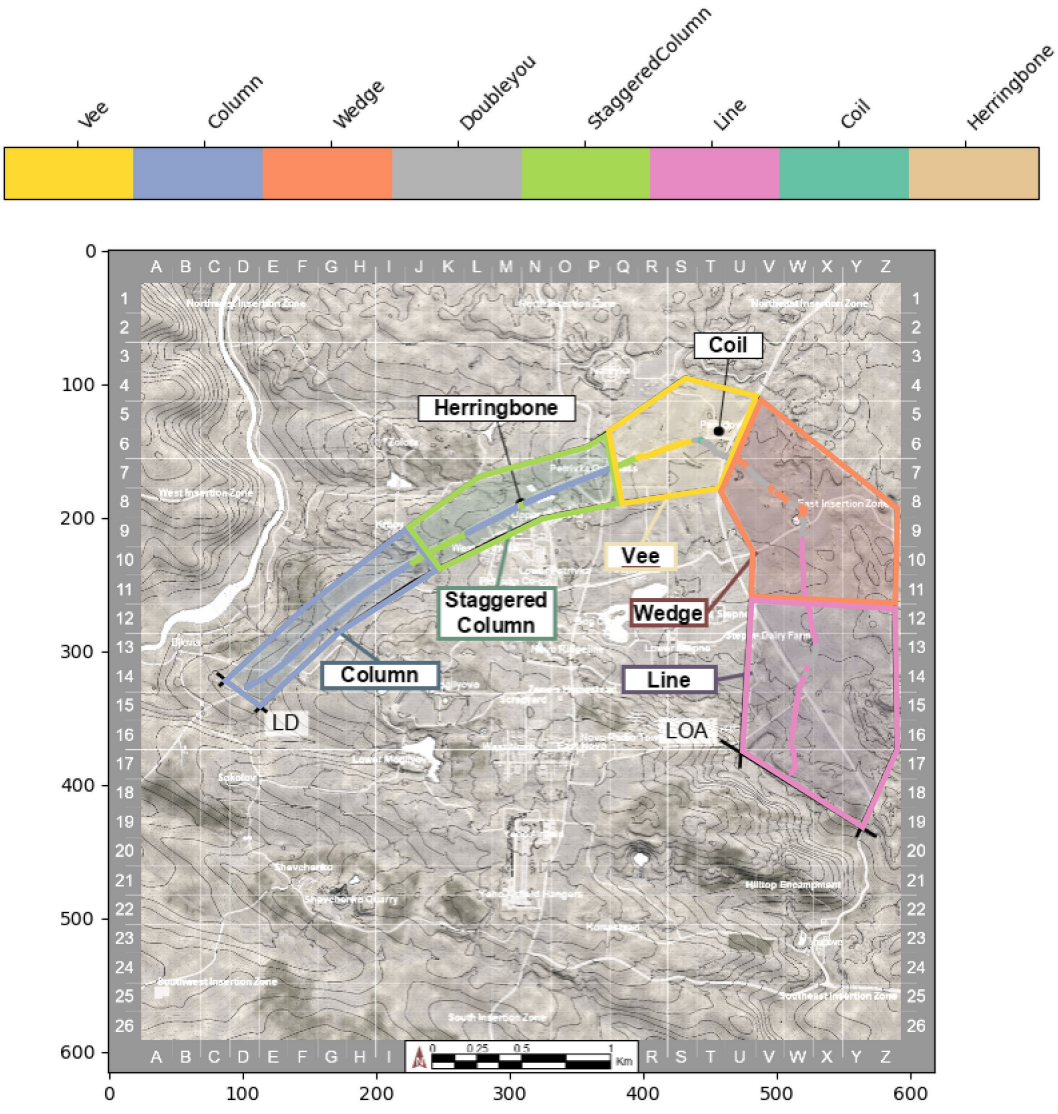



**Figure 4. Formation classification of formation training exercise. Annotations on the map show the instructions the platoon received indicating the route and regions in which they should use which formation. The colored line shows the route of the platoon over the course of the exercise, with the color of the line indicating the formation selected by the fine-tuned classifier. Note that Herringbone and Coil are stationary formations (indicated by black dots).**

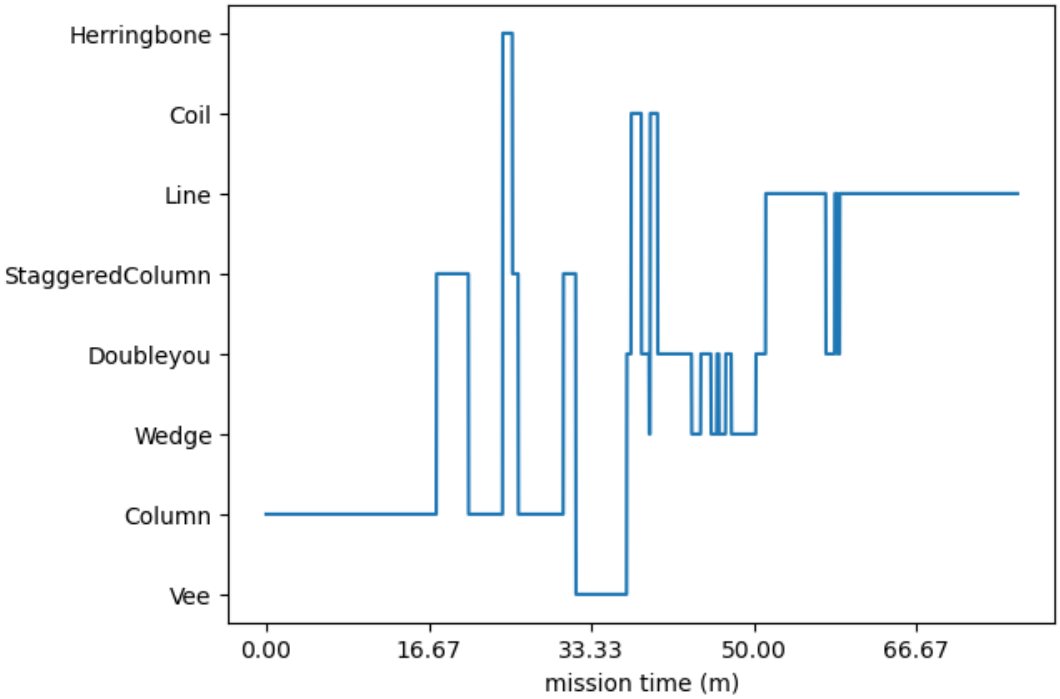



**Figure 5. Detected formation with respect to time. This is another view of the same data displayed in Figure 4 with respect to time instead of space. In this view, it is apparent that the stationary formations (Herringbone and Coil) were successfully detected.**

### 2.2.8 Discussion

Overall, the goal of automatically recognizing formations in a flexible, open-ended framework was only partially successful. A summary of these efforts appears in Table 3. Text-based spatial descriptions quickly appeared to be a dead end. The approach of converting position data into relative positions in an image as input to a multimodal model was also not especially successful. Multimodal models typically have many more parameters than text-only models, so the constraint of running a multimodal model on modest local hardware might have limited the performance of the overall approach. However, in some informal experimentation, we provided the same and similar prompts to large cloud-hosted models and did not achieve results that were much better.

One explanation for why spatial descriptions are such a challenge might come from how visual transformer models are typically trained. In a process called data augmentation, labeled images are rescaled, translated, rotated, and mirror reversed to generate irrelevant variability in the image while keeping the identity of the image subject constant (Bengio et al. 2011; Cubuk et al. 2020). This is an effective way to increase the robustness of image labels, but it might effectively destroy concepts of sidedness or global direction, since the training process is invariant to those properties.

Zero-shot image labeling of abstract shapes derived from the outline of the formation showed some promise, but the invariance to rotation and mirror-reversals was a problem. Fine-tuning a zero-shot image labeling model with rotation non-invariance and a military-relevant vocabulary could be a potential path forward, but that approach was not evaluated in this work.

A visual transformer model trained to recognize the specific formations used in a six-vehicle formation exercises accurately recognized those formations under plausible levels of random deformation. While this is a good solution to the specific problem of recognizing formations used in this use-case, it does not fulfill the goal of low- or no-programming solutions to flexibly and adaptably recognize formations. This solution required programming to generate synthetic formations to serve as training data. Alternatively, hand-labeled data from real operations could be used, but hundreds of examples would likely be needed.

**Table 3. Summary of formation labeling results.**

| Approach | Description | Finding |
|---|---|---|
| Text-only | Convert formation descriptions into text, use a text model to name the formation. | The text model confused formations that are not obviously similar. |
| Visual question answering: doctrine | Show doctrinal images of formations, request the model to describe the formation. | Descriptions confused left and right. |
| Visual question answering: abstract | Show abstracted images of formations, request the model to describe the shape. | Descriptions were nonsensical. |
| Zero-shot image classification: formation names | Show abstracted images of formations, have the model pick among formation names. | The model could not classify the images by formation. |
| Zero-shot image classification: shape names | Show abstracted images of formations and have the model pick among shape names. | The model identified the correct shape, but many-to-one relationship with formations mean limited utility. |
| Fine-tuned visual classifier: abstract | Fine-tune a visual transformer model to label formations in abstracted images. | The model correctly labeled formations with accuracy 99.6%. |
| Fine-tuned visual classifier: for six-vehicle platoon | Fine-tune a visual transformer model to label representations of formations used in a six-vehicle platoon. | The model perfectly labeled the held-out set, but performed poorly on real mission data (not shown). |
| Fine-tuned visual classifier: formation-specific noise | Fine-tune a visual transformer model to label representations of formations used in a six-vehicle platoon using formation-specific, hand-crafted variability. | The model perfectly labels held-out set and performs well on real mission data. |

# 3. Conclusion

The overall goal of this work was to propose and demonstrate proof-of-concept elements of an approach to generating narrative summaries or useful categorizations of mission data. The project was done under the constraint of using small models that could plausibly run locally, with no cloud-compute required, and to harness the flexibility of generative models and image classification models to perform well with little-or-no programming or fine-tuning. The overall idea is to generate an overly detailed collection of sentences that then would be iteratively summarized to achieve a concise summary of a mission. As a first step toward this approach, we identified two elements for proof-of-concept experimentation: the relatively easy task of translating firing events into sentences, and the relatively difficult task of recognizing formations from the location data of each individual vehicle.

This approach successfully used a local language model with a small number of manually generated examples to convert machine-native firing events into natural text sentences. While the error rate of these sentences was the same as the error rate of a human generating example sentences, the errors made by the model were qualitatively different from those generated by a human. While the human's errors are recognizable as typos, the text model errors were either critical or nonsensical (assigning a vehicle to the wrong team and assigning the terrain to a team, respectively).

Formation recognition proved to be a challenge for language models, visual question answering models, and zero-shot image classification. A fine-tuned visual transformer trained on synthetic formation images was able to recognize formations used in a training exercise. Although the fine-tuned visual transformer was the most successful formation recognition approach, this approach is inflexible and unable to adapt to changes in formations without retraining on new data. Future work should explore fine-tuning a zero-shot model with a military-relevant vocabulary and non-invariance to reflection and rotation. A zero-shot image classification model with a military-relevant vocabulary and sensitivity to spatial properties could potentially provide useful labels for novel formations and movement techniques. If successful, such a model would be a plausible element of an overall system for generating narrative summaries for mission data.

## 4. References


Bang Y et al. A multitask, multilingual, multimodal evaluation of ChatGPT on reasoning, hallucination, and interactivity. arXiv preprint; 2023. arXiv:2302.04023.

Bengio Y et al. Deep learners benefit more from out-of-distribution examples. Proceedings of the 14th International Conference on Artificial Intelligence and Statistics, in Proceedings of Machine Learning Research. 2011;15:164–172.

Cohn AG, Hernandez-Orallo J. Dialectical language model evaluation: an initial appraisal of the commonsense spatial reasoning abilities of LLMs. arXiv preprint; 2023. arXiv:2304.11164.

Cossio M. A comprehensive taxonomy of hallucinations in large language models. arXiv; 2025. arXiv:2508.01781. https://doi.org/10.48550/arXiv.2508.01781

Cubuk ED, Zoph B, Shlens J, Le QV. Randaugment: practical automated data augmentation with a reduced search space. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops; 2020. p. 702–703.

Gordon A. Commonsense interpretation of triangle behavior. Proceedings of the AAAI Conference on Artificial Intelligence; 2016;30(1). https://doi.org/10.1609/aaai.v30i1.9881

Gordon A, Wang T. Narrative text generation from abductive interpretations using axiom-specific templates. In: Mitchell A, Vosmeer M, editors. Proceedings of the 14th International Conference on Interactive Digital Storytelling, ICIDS 2021; 2021 Dec 7–10; Tallinn, Estonia. p. 71–79.

Hoffmann et al. Training compute-optimal large language models. arXiv preprint; 2022. arXiv:2203.15556.

Hsieh CY et al. Distilling step-by-step! Outperforming larger language models with less training data and smaller model sizes. arXiv preprint; 2023. arXiv:2305.02301.

Huang L et al. A survey on hallucination in large language models: principles, taxonomy, challenges, and open questions. ACM Transactions on Information Systems. 2025;43(2):1–55. https://doi.org/10.1145/3703155

Jin H, Zhang Y, Meng D, Wang J, Tan J. A comprehensive survey on process-oriented automatic text summarization with exploration of LLM-based methods. arXiv preprint; 2024. arXiv:2403.02901.

Kurisinkel LJ, Chen NF. LLM based multi-document summarization exploiting main-event biased monotone submodular content extraction. arXiv preprint; 2023. arXiv:2310.03414.

Lewis P et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. Advances in Neural Information Processing Systems. 2020;33:9459–9474.

Li F, Hogg DC, Cohn AG. Advancing spatial reasoning in large language models: an in-depth evaluation and enhancement using the StepGame benchmark. Proceedings of the AAAI Conference on Artificial Intelligence. 2024;38(17):18500–18507. https://doi.org/10.1609/aaai.v38i17.29811

Loshchilov I, Hutter F. Decoupled weight decay regularization. arXiv preprint; 2017. arXiv:1711.05101.

Ma Z, Kim DJ, Chen TH. OpenLogParser: unsupervised parsing with open-source large language models. arXiv preprint; 2024. arXiv:2408.01585.

Roy D et al. Exploring LLM-based agents for root cause analysis. In: Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering; 2024 July. p. 208–219.

Sojitra D, Jain R, Saha S, Jatowt A, Gupta M. Timeline summarization in the era of LLMs. Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval; 2024 July. p. 2657–2661.

Wang F et al. A comprehensive survey of small language models in the era of large language models: techniques, enhancements, applications, collaboration with LLMs, and trustworthiness. arXiv preprint; 2024. arXiv:2411.03350.

Wu W, Deng W. Spatial representation of large language models in 2D scene. Proceedings of the 4th Workshop on Generation, Evaluation and Metrics ($GEM^2$ 2025); 2025. p. 18–29.

Xu Z, Jain S, Kankanhalli M. Hallucination is inevitable: an innate limitation of large language models. arXiv; 2025. arXiv:2401.11817. https://doi.org/10.48550/arXiv.2401.11817

Yamada Y, Bao Y, Lampinen AK, Jungo K, Ilker Y. Evaluating spatial understanding of large language models. Transactions on Machine Learning Research. 2024.

## Appendix A. Firing Event Prompt Example

This appendix appears in its original form, without editorial change.

I am going to give you some examples of data entries. Each one has an interpretation. Please look carefully at the examples:

[{'victim_vehicle_class': 'Landscape', 'AttackingVehicle_ProjectileType': 'M2', 'AttackingVehicle_AttackingGunnerType': 'RCVLGunnerCrewStationPawn', 'AttackingVehicle_AttackingWeaponName': 'IWSWeaponComponent', 'attacking_vehicle_class': 'RED'INFORMS_RCVLPawn', 'attacking_vehicle_name': 'RED'INFORMS_RCVLPawn', 'victim_vehicle_name': 'Landscape', 'Interpretation': "An OPFOR RCV-Light missed it's target with an M2 weapon."}

{'victim_vehicle_class': 'Landscape', 'AttackingVehicle_ProjectileType': 'M17', 'AttackingVehicle_AttackingGunnerType': 'RCVLGunnerCrewStationPawn', 'AttackingVehicle_AttackingWeaponName': 'IWSWeaponComponent', 'attacking_vehicle_class': 'RED'INFORMS_RCVLPawn', 'attacking_vehicle_name': 'RED'INFORMS_RCVLPawn', 'victim_vehicle_name': 'Landscape', 'Interpretation': "An OPFOR RCV-Light missed it's target with an M17 weapon."}

{'victim_vehicle_class': 'RED'INFORMS_RCVMPawn', 'AttackingVehicle_ProjectileType': '3UBR8', 'AttackingVehicle_AttackingGunnerType': 'BTR82AGunnerCrewStationPawn', 'AttackingVehicle_AttackingWeaponName': '30mmWeaponComponent', 'attacking_vehicle_class': 'RED'INFORMS_BTR82APawn', 'attacking_vehicle_name': 'RED'INFORMS_BTR82APawn', 'victim_vehicle_name': 'RCV3', 'Interpretation': "An OPFOR BTR-82A hit RCV3 with it's 30mm weapon"}

{'victim_vehicle_class': 'Landscape', 'AttackingVehicle_ProjectileType': 'MK258', 'AttackingVehicle_AttackingGunnerType': 'RCVMGunnerCrewStationPawn', 'AttackingVehicle_AttackingWeaponName': 'MainGunWeaponComponent', 'attacking_vehicle_class': 'RED'INFORMS_RCVMPawn', 'attacking_vehicle_name': 'RCV2', 'victim_vehicle_name': 'Landscape', 'Interpretation': "RCV2 shot it's MK weapon and hit and enemy RCV-Light vehicle"}

{'victim_vehicle_class': 'Landscape', 'AttackingVehicle_ProjectileType': 'MK258', 'AttackingVehicle_AttackingGunnerType': 'RCVMGunnerCrewStationPawn', 'AttackingVehicle_AttackingWeaponName': 'MainGunWeaponComponent', 'attacking_vehicle_class': 'RED'INFORMS_RCVMPawn', 'attacking_vehicle_name': 'RCV5', 'victim_vehicle_name': 'Landscape', 'Interpretation': "RCV5, which is BLUFOR vehicle, shot it's MK258 and missed, hitting the terrain"}]

Now I will give you a challenge with a blank interpretation:

{'victim_vehicle_class': 'Landscape', 'AttackingVehicle_ProjectileType': '3UBR8', 'AttackingVehicle_AttackingGunnerType': 'BTR82AGunnerCrewStationPawn', 'AttackingVehicle_AttackingWeaponName': '30mmWeaponComponent', 'attacking_vehicle_class': 'RED'INFORMS_BTR82APawn', 'attacking_vehicle_name': 'RED'INFORMS_BTR82APawn', 'victim_vehicle_name': 'Landscape', 'Interpretation': "}

Please provide an appropriate interpretation for this challenge entry.

Your interpretation must identify the attacking vehicle and the type of weapon used. Your interpretation must also identify what was hit, always using the verb 'hit'. If the shot hit a BlockingVolume, Landscape, or StaticMeshActor, then the shot is considered a miss.

Vehicle names, indicated by the fields 'attacking_vehicle_name' and 'victim_vehicle_name' that start with 'RED"INFORMS' are OPFOR vehicles. Vehicle names that do not start with 'RED"INFORMS' are BLUFOR vehicles. Terrain elements are neither OPFOR nor BLUFOR.

Do not guess or infer information beyond what is in the data entry. Do not expand or define any acronyms or abbreviations.

## Appendix B. Text-only Formation Prompt

This appendix appears in its original form, without editorial change.

I am going to describe some formations of four-vehicle platoons. For each formation, one vehicle is designated the leader, and then each other vehicle's position is described relative to that leader. Consider these definitions carefully:

Formation Name: Column, Formation Description: V2 is the leader. V1 is behind the leader. V4 is far behind the leader. V3 is far behind the leader.

Formation Name: Staggered Column, Formation Description: V1 is the leader. V2 is behind and to the left of the leader. V4 is far behind the leader. V3 is far behind and to the left of the leader.

Formation Name: Wedge, Formation Description: V1 is the leader. V4 is behind and to the right of the leader. V2 is far behind and to the left of the leader. V3 is far behind and far to the right of the leader.

Formation Name: Line, Formation Description: V3 is the leader. V4 is to the left of the leader. V1 is far to the left of the leader. V2 is far to the left of the leader.

Formation Name: Echelon Left, Formation Description: V1 is the leader. V2 is behind and to the left of the leader. V4 is far behind and far to the left of the leader. V3 is far behind and far to the left of the leader.

Formation Name: Echelon Right, Formation Description: V1 is the leader. V2 is behind and to the right of the leader. V4 is far behind and far to the right of the leader. V3 is far behind and far to the right of the leader.

I will give you a formation description, and I want you to identify the correct formation name. The names of the vehicles will be different, and the descriptions might be in a different order, but the relationships described should match one of the defined formations. For example:

Description: RCV3 is the leader. MCV5 is behind and to the right of the leader. RCV4 is far behind and to the left of the leader. RCV8 is far behind and far to the right of the leader.

formation name: Wedge

Description: RCV6 is the leader. MCV1 is behind and to the left of the leader. RCV4 is far behind and far to the left of the leader. RCV2 is far behind and far to the left of the leader.

formation name: Echelon Left

Description: RCV3 is the leader. RCV6 is to the left of the leader. RCV9 is far to the left of the leader. MCV8 is far to the left of the leader.

formation name: Line

Now it is your turn.

Description: RCV8 is the leader. RCV7 is behind and to the left of the leader. RCV1 is far behind the leader. MCV9 is far behind and to the left of the leader.

formation name:

## List of Symbols, Abbreviations, and Acronyms

| | |
|---|---|
| AAR | after action review |
| JSON | JavaScript object notation |
| LLM | large language model |
| RAG | retrieval-augmented generation |
| SME | subject matter expert |
| XML | extensible markup language |

| | |
|---|---|
| 1 (PDF) | DEFENSE TECHNICAL INFORMATION CTR DTIC OCA |
| 1 (PDF) | DEVCOM ARL FCDD RLB CI TECH LIB |