International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

A Novel Approach to Semantic Search for Home Security Footage Using Generative AI and LLMs

Sibin Thomas

Tech Lead
sibin_thomas15@hotmail.com

Abstract
This paper looks at how generative AI and large language models (LLMs) could change the way image and video search works in home security systems [1]. We look at the problems with current methods, like manual tagging and simple machine learning-generated metadata, and suggest a new way to use deep learning models to create rich embeddings that show what visual material means semantically [2]. These embeddings, which are kept in vector databases, make natural language-based search easier, so users can ask complicated questions in a conversational way. We talk about how to train and fine-tune LLMs, using methods such as RLHF, context optimization, and RAG, to make searches more accurate and faster [3]. We also look into how to use trends of user access to do proactive indexing and caching of relevant results. This study opens the way for smarter and easier-to-use search tools in home security, making it easier for users to get useful information from their video data.

Keywords: Home security cameras, Image and video search generative AI, LLMs, Deep learning Embeddings, Vector databases, Natural language processing (NLP)

INTRODUCTION

Imagine being able to easily search through your home security footage by saying things like, "Show me any videos of people in my backyard yesterday afternoon." Using generative AI and large language models (LLMs) could change the way we connect with and get information from the huge amounts of video data that home security cameras are collecting all the time.

This paper looks at the problems with the usual ways of searching for images and videos, which depend on either labeling them by hand or using simple machine learning with labels that have already been set. These methods don't do a good job of capturing the subtleties of visual material and force users to use rigid search queries, which makes it harder for them to explore and analyze their footage.

Then, we look at how generative AI and LLMs could change things in this area. We can start a new era of natural language-based search by making rich embeddings that hold the semantic meaning of images and videos [4]. This lets users ask complicated questions in a natural way, making it easy to get very relevant answers.

This paper looks at the main steps needed to make such a system, from creating and saving embeddings in vector databases to teaching and fine-tuning LLMs to search accurately and quickly [5]. We look into advanced methods such as context optimization, retrieval augmented generation (RAG), and reinforcement learning from human feedback (RLHF) to improve performance and reduce problems.

We also talk about how user access trends and finding unusual events can be used to automatically index

and cache relevant results, which improves both the user experience and the efficiency of the system. The goal of this study is to encourage the creation of smarter and easier-to-use search features in home security camera systems. This will pave the way for a future where it will be as easy as asking a question to connect with and learn from video data.

BACKGROUND ON IMAGE AND VIDEO SEARCH WORKFLOW IN HOME SECURITY CAMERAS TODAY

This part goes into great detail about how picture and video search usually works in modern home security camera systems. It shows what's wrong with the old ways of doing things and makes it possible to look into how generative AI and LLMs might be able to change this process.

Event Detection and Video Recording

The workflow typically begins with the camera's event detection mechanism, often based on a Passive Infrared (PIR) sensor. When the PIR sensor detects motion, it triggers the camera to begin recording video and capturing images. This captured visual data forms the basis for subsequent search and retrieval operations.

Machine Learning-Based Metadata Generation

The video and images are then processed by ML models that have already been taught. These models usually run on the camera or in the cloud. These models are taught to recognize certain things or events that are interesting, like:

1. **Motion:** Detecting any movement within the camera's field of view.
2. **Person:** Identifying the presence of a human.
3. **Package:** Recognizing the delivery of a package.
4. **Vehicle:** Detecting the presence of a car, truck, or motorcycle.
5. **Pet:** Identifying pets like cats or dogs.

The output of these ML models is then attached to the video or image as metadata. This metadata may include labels indicating the detected objects or events, along with associated confidence scores and timestamps. In some cases, a mapping between video durations and the corresponding ML-generated metadata is maintained to facilitate searching within specific timeframes.

User Search and Retrieval

Most of the time, people use the home security camera app to look for a specific event or video clip. There are two main ways to search in the app:

1. **Chronological Browsing:** Users can scroll through a timeline of recorded events, either chronologically or within a specified duration.
2. **Basic Filtering:** In some cases, the app may offer basic filtering options based on the ML-generated metadata. For example, users might be able to filter events by category, such as "person," "package," or "vehicle."

LIMITATIONS OF TRADITIONAL IMAGE AND VIDEO SEARCH IN HOME SECURITY CAMERAS

Traditionally, searching image and video data in home security cameras has relied on either manual tagging or simple metadata created by ML [6]. These methods allow you to look in a basic way, but they have big problems that make them less useful and less enjoyable to use.

Limited Accuracy and Semantic Understanding

Manual tagging, in which people name photos and videos with relevant keywords by hand, takes a lot of time, effort, and can lead to mistakes [7]. It also doesn't always get the subtleties and background of visual

information. A "person" tag, for example, doesn't say whether the person is walking, running, holding something, or interacting with something else in the scene.

In the same way, basic ML-generated metadata, which usually includes object recognition models, doesn't give a lot of information about meaning. These models can spot things like "person," "car," and "package," but they often can't figure out what's going on, how things are connected, or what the scene is all about. This makes searches less accurate and specific, especially for questions that are long or complicated.

Restricted Search Capabilities

With traditional methods, users can only look using tags or categories that have already been set up. This makes things less flexible and forces users to rely on the labels that have already been set, which might not exactly reflect what they were looking for. Users can't ask complicated questions that involve many entities, acts, or relationships between time periods.

A person might want to look for "someone in a red shirt coming into the house through the back door between 10 pm and midnight," as an example. Traditional methods can't handle these kinds of complex questions, so users have to go through hours of footage by hand to find the event they're looking for.

Lack of Natural Language Understanding

In traditional search interfaces, users often have to go through menus, choose filters that are already set up, or type in specific terms. This can be hard to understand and use, especially for people who aren't tech-savvy. Users can't talk about their search intentions in a conversational way because the search engine doesn't understand normal language. This makes it harder and slower to find the information they need.

As an example, let's say a user has three outdoor cameras that are set up to look at different times of the day. Customers can't ask for things like "Find all the video clips from all 3 cameras that show a man in a red hoodie walking around my house between 2pm and 8pm today." with standard search interfaces.

Inability to Connect Events

Usually, each picture or video clip is handled separately, and it's not possible to link events or activities that happen in different recordings that are connected. Users have to manually put together pieces of information from different movies, which makes it hard to see the big picture of an event or spot patterns over time.

For example, if a user wants to keep an eye on a suspicious person as they move around their property, they would have to watch several video clips one by one to figure out the person's path and movements, which can take a lot of time and be useless.

Scalability and Maintenance Challenges

Manual tagging and simple ML-based metadata generation get harder to scale and keep up with as the amount of video data grows. It can be hard to make sure that naming or training ML models is done consistently and correctly for a lot of different cameras and situations. This can make search results less consistent and slow down the system as a whole.

Limited Personalization

Most of the time, traditional methods don't allow for customization, so they treat all people and cameras the same. This could lead to search results that aren't useful and a less satisfying experience for the user. The method isn't very useful or effective because search results can't be changed to fit the preferences of each user or the situation with each camera.

There are some problems with the way traditional image and video search works in home security cams that make them less useful and less enjoyable to use. Their low accuracy, limited search options, inability to understand normal language, inability to link events, problems with scalability, and limited ability to

customize make it clear that we need smarter and more advanced search solutions. The use of generative AI and LLMs could help get around these problems and completely change how people interact with and learn from their home security video.

ENHANCING IMAGE AND VIDEO SEARCH IN HOME SECURITY CAMERAS WITH GENERATIVE AI AND LLMS

Using basic ML generated metadata or manually tagging images and videos in home security cameras are two old ways to look for data that often aren't accurate or flexible enough [8]. These methods have trouble picking up on the subtleties of visual material and don't really make searching easy. This part goes into detail about how generative AI and LLMs can change the way picture and video search is done in this area by making it easier to use natural language for search and giving visual data a smarter, more semantic understanding.

Generating and Storing Embeddings

This method is based on making rich embeddings that catch the semantic meaning of images and videos [9]. These embeddings are dense vector representations that hold the most important parts of the visual material and are more than just labels or tags. These embeddings can be made with powerful deep learning models, like convolutional neural networks (CNNs) that have already been taught on huge sets of images. For example, models like CLIP (Contrastive Language-Image Pre-training) [2] can be used to make embeddings that hold both visual and semantic data. These models learn to connect images with textual descriptions. CLIP is taught by a huge set of image-text pairs. It learns to put images and text into the same embedding area. In this way, CLIP can make embeddings that catch images' semantic meaning, even if they haven't been labeled or tagged directly.

Then, these embeddings can be saved in a vector database, which is a special kind of database that is designed to store and explore high-dimensional vectors [10]. Vector databases are made to handle the large amounts and many dimensions of embedding data quickly. They use special search and indexing algorithms to quickly find useful embeddings based on how similar they are. Pinecone, Milvus, and FAISS are all well-known vector libraries.

Think about a picture of a "cat sitting on a mat." Instead of just labeling it "cat" and "mat," a CNN like CLIP would create an embedding that shows how the cat and the mat are connected, the type of mat, the cat's pose, and other information about the scene. This more complete picture lets you do more precise and detailed searches.

Processing User Input and Search

Input from the user is processed and turned into a question embedding so that natural language-based search can happen. In order to do this, an LLM must be used to figure out what the user wants and create a vector representation that captures the semantic meaning of the question. It then checks this query embedding against the picture and video embeddings that are already in the vector database to see which ones are the best matches. You can use a Sentence Transformer model that has already been trained [11]. It takes this text query as input and creates an embedding, which is a dense vector representation that captures the meaning of the question. This embedding shows the main thoughts and connections in the sentence as a list of numbers.

A user might ask, "Show me videos of the delivery person leaving a package at the front door yesterday." The LLM would look at this query, find the key entities and actions (delivery person, package, front door), and create a query embedding that holds all of this information. After that, this embedding would be used

to look through the vector database for videos that mostly match what the user asked for. Distance measures, such as cosine similarity, are used to figure out how similar the query embedding is to the video embeddings. As search results, the videos with the best similarity scores are shown.

The LLM is very important for getting the user's natural language query and turning it into an embedding space representation that makes sense. It can handle complicated searches that include many entities, actions, and time limits. The LLM can tell the difference between phrases like "a person leaving a package" and "a person taking a package," which makes it easier to find the right videos.

Training and Fine-tuning LLMs

To make an LLM good at this job, you need to use a mix of methods:

1. **Supervised Learning:** The LLM can be trained on a big dataset of paired images or videos and text descriptions [12]. It will learn to connect visual content with text descriptions over time. This dataset can be made by adding detailed text to images and videos by hand, or it can be used with existing datasets like LAION-5B, which has billions of image-text pairs.
2. **Reinforcement Learning from Human Feedback (RLHF):** RLHF can be used to fine-tune the LLM so that it responds in a way that people would want and so that search results are more accurate and relevant [3]. To do this, a reward model that mimics how people rate the quality of a search must be trained, and the LLM's learning must be guided by this model. Human annotators give the LLM feedback on its search results, telling it which ones are useful and which ones are not. This information is used to teach a reward model, which then tells the LLM how to make better, more accurate search results.

LLM Parameters and Hyperparameters

To train an LLM to search images and videos effectively for home security, you need to carefully tune a number of LLM-specific factors and hyperparameters. These factors affect how well the model works, how much it can learn, and how accurate its results are. To get a better idea of how to tweak these hyperparameters for home security cams, read on:

Model Size: This is about how many factors are in the LLM. It is possible for bigger models (with 175 billion parameters) to learn more complicated patterns and connections, but they need a lot more data and computing power to train. It's better to use smaller models because they use less energy and computer power, but they might not show as much information.

Tuning for Home Security: The best model size relies on what the home security system needs. Things to think about are:

1. **Complexity of search queries:** If users are expected to use natural language queries that are complicated and include many entities, actions, and time limits, then a bigger model may be needed. For instance, to say "show me videos of someone in a red shirt trying to open the back door last night between 10 pm and midnight," you would need to know more about words.
2. **Computational resources:** If the LLM is used on devices with limited resources, like the camera or a local hub, a smaller, more efficient model might be better.
3. **Accuracy requirements:** If accuracy is very important, even for simple questions, a bigger model might be needed.

Context Window

The context window shows the longest string of data that the LLM can handle at once, which is the user query plus the image or video description. A bigger context window lets the model look at more data, which could help it understand and be more accurate with long or complex questions.

Tuning for Home Security:

1. **Query length:** A bigger context window is helpful if users tend to use long, detailed searches. Say, "find the video where a small brown dog with a white patch on its chest entered the garden through the side gate."
2. **Description detail:** If you describe a picture or video in great detail (for example, by listing object properties, relationships, and actions), a bigger context window will help the LLM pick up on these small details.
3. **Computational constraints:** Bigger context windows require more processing power, which could be a problem for devices with limited resources.

Learning Rate

This parameter sets how many times the model's values are changed during each training round. A fast learning rate can make training go faster, but it could also make things less stable and cause them to go beyond the best answer. It is possible to get stuck in local minima and slow convergence if the learning rate is low.

Tuning for Home Security: Experimenting is a common way to find the best learning rate. Performance can be raised with methods such as learning rate scheduling, which changes the rate of learning while training. Keeping an eye on the training loss and validation precision can help you find the right learning rate.

Batch Size

The batch size tells the model how many training cases to run before its parameters are changed. Training can go faster with bigger batches, especially when GPUs are used, but they need more memory. Smaller batch numbers can help training be more stable and help it spread to more people.

Optimization Algorithm

This is the method that is used to change the model's settings while it is being trained. In terms of how fast they converge, how stable they are, and how much memory they use, each optimization method has its own pros and cons.

It takes a lot of computing power and careful choice of model design, training data, and hyperparameters to train LLMs. When it comes to fine-tuning LLMs, RLHF is one of the best methods because it lets the model learn from human input and get better over time.

Iterative Improvement and Experimentation

Through iterative training and experimentation, the picture and video search can keep getting better:

1. **A/B Testing:** When you put a parallel LLM model into experimentation mode, you can try different model architectures, parameters, or training methods against each other. This lets you directly compare performance and makes it easier to make decisions based on facts about which models to use and how to deploy them. For instance, two different LLM architectures can be put into use at the same time, and user comments and engagement metrics can be used to compare how well they work.
2. **User Feedback Integration:** Getting feedback from users on search results and using it in training can help improve the LLM's understanding of what users want and its ability to give them useful results [13]. You can get this feedback from users by asking them to rate something directly or by looking at things like click-through rates and how they narrow their searches.
3. **Fine-tuning:** Making small changes to the LLM on a regular basis using new data or new payment models can help it keep working well and adapt to changing user wants and needs. The LLM can be tweaked to make sure it stays correct and useful as new video data is added and user tastes change.

For the LLM to keep getting better and work better, it needs to be improved and tested over and over again. A/B testing lets you compare different models and setups based on data, and user feedback is very helpful for making the LLM better and more in line with changing user needs.

Context Optimization and Hallucination Reduction

Context optimization is one of the most important ways to make search results more accurate and useful. This means giving the LLM enough background information to understand the user's question and the image or video data that is important.

1. **User Profile and Preferences:** Adding information about the user, like past searches, most-used cameras, and chosen time frames, can help make the search experience more personal and the results more relevant. For instance, if a user often looks for videos taken by a certain camera in the evening, the system can give more weight to results from that camera and time period.
2. **Camera Metadata:** Adding camera-specific metadata, like location, orientation, and recording schedule, can help the LLM understand the visual material better. For instance, telling the LLM where the camera is ("front door" vs. "backyard") can help it tell the difference between events that were caught on different cameras that look a lot alike.
3. **Fact Checking:** Using what people say through the phone app to check the truth can help cut down on dreams and boost accuracy. For instance, if the LLM comes up with a description of the scene that doesn't match what the user knows about it, the user can tell the model what it's doing wrong and help it understand better. The LLM learns from this feedback process and gets better over time.

For accurate and useful search results, it's important to give the LLM a lot of information. This includes information that is unique to each user, metadata about the camera, and ways for users to fix any mistakes or hallucinations that the LLM makes.

Retrieval Augmented Generation (RAG)

The retrieval augmented generation (RAG) method can be used to make the generative AI system even better. RAG takes the best parts of both generative models and information retrieval techniques and puts them together. This lets the LLM get useful information from outside knowledge sources, like image databases or textual descriptions, and use it during the search process. This can help make search results more accurate and complete, especially for questions that are hard to understand.

For instance, if a user types in "videos of birds in my backyard," the LLM can use RAG to connect to an outside library of bird identifications. This lets the LLM not only find videos of birds but also give more detailed details about the species of birds seen, which improves the search results.

RAG lets the LLM use outside sources of information to improve the accuracy and completeness of search results, in addition to what it already knows.

This helps a lot for questions that need specific information or when the visual content isn't clear.

Fine-tuning Process for Home Security Cameras

1. **Start with a pre-trained LLM:** As a starting point, use a powerful LLM that has already been trained on a huge dataset (like CLIP). This gives you a solid base for understanding language and visual information.
2. **Create a home security-specific dataset:** Collect a dataset of images or videos, their natural language descriptions, and user feedback that is important to home security scenarios. This set of data should include things like "person walking," "car arriving," and "package delivered" that are often caught on home security cameras.

3. **Fine-tune the LLM:** Use the home security dataset to fine-tune the pre-trained LLM by changing the hyperparameters we talked about above (model size, context window, learning rate, batch size, optimization method).
4. **Employ RLHF:** To improve the LLM even more, use reinforcement learning from human input to make it respond more like people and make search results more accurate and relevant [14]. To do this, human annotators must give feedback on the search results, this feedback is then used to train a reward model, and the reward model is then used to guide the LLM's learning.
5. **Iterate and Experiment:** Keep an eye on how well the LLM is doing, try out different hyperparameter settings, and use user comments to make it better and better over time.
6. We can make very good LLMs for picture and video search in home security camera systems by fine-tuning these LLM parameters and hyperparameters and using methods like RLHF and iterative experimentation. Users can easily find the visual information they need by using natural language questions. This makes the search experience more powerful and easy to use.

Conclusion

In this study, we looked at how generative AI and LLMs could change the way image and video search works in home security camera systems. We can make the search experience smarter, easier to use, and more intuitive by going beyond the limits of old methods like human tagging and simple machine-generated metadata.

Strong deep learning models are used in the suggested method to create rich embeddings that capture the semantic meaning of visual content. These embeddings, which are kept in fast vector databases, make it possible for natural language-based search to work, letting users ask questions in a natural way.

It takes supervised learning, reinforcement learning from human feedback (RLHF), and careful tuning of model parameters and hyperparameters to train and fine-tune LLMs for this job. Iterative testing, A/B testing, and incorporating user comments can all help with continuous improvement.

Furthermore, context optimization, fact-checking tools, and retrieval augmented generation (RAG) improve the precision and breadth of search results, while lowering hallucinations and aiding the LLM's comprehension of visual content.

By looking at how users access the system and finding strange behavior, it can automatically organize and store relevant results, cutting down on wait times and making the best use of resources.

The results of this study show that generative AI and LLMs can make home security camera systems much better by making search functions smarter, faster, and easier to use. As these technologies keep getting better, we can expect even more advanced and easy-to-use ways to connect with and learn from the huge amounts of video data that home security cameras are collecting.

REFERENCES

1. Li, J., Li, L., & Sun, Q. (2023). A Survey on Large Language Models for Recommender Systems. arXiv preprint arXiv:2303.13386.
2. Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., ... & Sutskever, I. (2021). Learning transferable visual models from natural language supervision. [1] In International Conference on Machine Learning (pp. 8748-8763). [2] PMLR.

3. Stiennon, N., Ouyang, L., Wu, J., Ziegler, D., Lowe, R., Voss, C., ... & Christiano, P. (2020). Learning to summarize from human feedback. Advances in Neural Information Processing Systems, 33, 3008-3021.
4. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.
5. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
6. Chen, M. Y., & Lian, D. (2012, October). Efficient video event recognition using cloud computing. In 2012 IEEE 12th International Conference on Data Mining Workshops (pp. 826-831). IEEE.
7. Russell, B. C., Torralba, A., Murphy, K. P., & Freeman, W. T. (2008). Labelme: a database and web-based tool for image annotation. International journal of computer vision, 77(1),[1] 157-173.
8. Saravanan, C., & Sujatha, K. (2018, April). A state of art techniques on object detection using deep learning algorithms. In 2018 International Conference on Intelligent Computing and Control Systems (ICICCS) (pp. 1044-1047). IEEE.
9. Wang, H., Zhang, F., Xie, X., & Guo, M. (2015). Dl-sfa: Deeply-learned slow feature analysis for action recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2525-2532).
10. Amato, G., Falchi, F., Rabitti, F., & Savino, P. (2018, October). Yfcc100m-hnfc6: a large-scale deep features benchmark for similarity search. In Proceedings of the 2nd International Workshop on Similarity Search and Applications (pp. 1-8).
11. Reimers, N., & Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084.
12. Yu, J., Jiang, J., Xia, R., Chang, Y., & Hauptmann, A. G. (2014). Temporal action localization in untrimmed videos via multi-stage cnns. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1063-1070).
13. Liu, N. F., Amini, M. H., Yang, Y., & Uszkoreit, J. (2023). Webgpt: Browser-assisted question-answering with human feedback. arXiv preprint arXiv:2308.08893.
14. Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., ... & Lowe, R. (2022). Training language models to follow instructions with human feedback. [1] Advances in Neural Information Processing Systems, 35, [2] 27730-27744