UHASSELT
KNOWLEDGE IN ACTION

Maastricht University

Faculteit Wetenschappen
School voor Informatietechnologie

master in de informatica

Masterthesis

Enhancing Multimodal Video Retrieval Systems: A Framework for AI Model Integration and Transparency

Michiel Swaanen

Scriptie ingediend tot het behalen van de graad van master in de informatica

PROMOTOR :

Prof. dr. Frank NEVEN

2023
2024

De transnationale Universiteit Limburg is een uniek samenwerkingsverband van twee universiteiten in twee landen: de Universiteit Hasselt en Maastricht University.

UHASSELT
KNOWLEDGE IN ACTION

www.uhasselt.be
Universiteit Hasselt
Campus Hasselt:
Martelarenlaan 42 | 3500 Hasselt
Campus Diepenbeek:
Agoralaan Gebouw D | 3590 Diepenbeek

Maastricht University

Faculteit Wetenschappen
School voor Informatietechnologie

master in de informatica

Masterthesis

Enhancing Multimodal Video Retrieval Systems: A Framework for AI Model Integration and Transparency

Michiel Swaanen

Scriptie ingediend tot het behalen van de graad van master in de informatica

PROMOTOR :

Prof. dr. Frank NEVEN

Universiteit Hasselt

Masterproef voorgedragen tot het behalen van de graad van master in de informatica

Enhancing Multimodal Video Retrieval Systems: A Framework for AI Model Integration and Transparency

Auteur:

Swaanen Michiel

Promotor:

prof. dr. Frank NEVEN

Academiejaar 2023-2024

Acknowledgments

In completing this thesis, the role of my supervisor, Prof. Dr. Frank Neven, was crucial. His approach to academic mentorship, characterized by a significant degree of intellectual freedom, allowed me to pursue my original ideas while ensuring strict academic standards were met. Prof. Dr. Neven's support was foundational in the development of this research. He created an environment that balanced independence with constructive critique, facilitating rapid iteration over various ideas for this work. His feedback was mainly instrumental in enhancing the academic tone and style of my writing, pushing me to engage deeply with my subject matter and refine my arguments. Additionally, Prof. Dr. Neven facilitated important connections with professionals in the field, ensuring that my thesis was developed with clear intellectual property boundaries outside of the University of Hasselt. This has opened up possibilities for further developing the project into a production-grade system beyond the academic context.

His readiness to allow independent work and confidence in my capability to manage it have significantly influenced my scholarly development. Prof. Dr. Neven's commitment to academic excellence and belief in my ability to contribute meaningfully to our field were both motivating and challenging, driving me to uphold high standards in my work.

Abstract

The exponential growth in video data worldwide presents significant challenges for existing cloud storage systems, particularly in video retrieval capabilities. Current technologies primarily focus on text and filename-based search functionalities, which significantly overlook the rich, multimodal nature of video content. This thesis addresses the critical need for advanced video retrieval systems by proposing a novel framework that integrates multiple Artificial Intelligence (AI) models without the need for retraining. This integration enhances the predictability, debuggability, and usability of AI models in video search applications.

The research explores three key areas: the fusion of AI models to enhance search capabilities without extensive retraining, improvements in model transparency and debuggability to counter the "black-box" nature of AI, and the facilitation of plug-and-play features for developers, enabling the easy addition of new models or modalities. This approach not only tackles the technical complexities associated with developing advanced video search engines but also addresses the limitations of current technological frameworks in handling diverse video data types.

Through rigorous testing, including 48 separate tests at various difficulty levels, the thesis evaluates the performance of the proposed multimodal AI model in terms of precision, recall, and efficiency. The findings demonstrate significant improvements in search accuracy and query response time, positioning this research to substantially benefit multimedia platforms and cloud storage systems by enhancing the user experience in video search.

This thesis provides a valuable framework for developers and teams lacking the resources or expertise to develop their own multimodal model. This framework emulates the functionality of a foundation model through a composite of multiple multimodal models, designed for easy integration and modification in a plug-and-play fashion. Unlike traditional foundation models, it avoids "black-box" issues, offering greater transparency and maintainability. The potential applications of this framework extend beyond video search engines to any retrieval system that requires robust multimodal analysis, thereby broadening its utility across various technological domains.

Samenvatting

Introductie

Het aantal videobestanden wereldwijd neemt gestaag toe. De huidige zoekfuncties in cloudopslagsystemen zijn echter voornamelijk gericht om te zoeken op bestandsnaam of de inhoud van tekstbestanden. Hierdoor blijven video- en fotomateriaal vaak onderbelicht, aangezien de zoekmogelijkheden zeer beperkt zijn binnen bestaande producten.

Voor het terugvinden van videobestanden kan gebruik worden gemaakt van veel meer data dan enkel de bestandsnaam. Video's bevatten een grote hoeveelheid aan visuele, auditieve en tekstuele informatie, waardoor ze fundamenteel verschillen van traditionele tekstbestanden zoals documenten en spreadsheets.

In deze thesis bespreken we verschillende problemen die richtlijnen bieden voor de oplossing die we gaan ontwikkelen voor de videozoekmachine.

1. Searchable: Hoe video fragmenten semantisch zoekbaar worden op acties, personen, geluiden, gesproken taal en tekst die voorkomen in de video.
2. Fusing: Hoe AI-modellen kunnen worden gecombineerd (gefuseerd) zonder deze opnieuw te trainen.
3. Anti-blackbox: Hoe de voorspelbaarheid en debugbaarheid van gefuseerde AI-modellen kan worden verhoogd.
4. Plug & Play: Hoe het voor ontwikkelaars zo eenvoudig mogelijk gemaakt kan worden om snel nieuwe modellen of modaliteiten toe te voegen aan het gefuseerde AI-model.

Het doel van deze thesis is om een duidelijk beeld te geven van zowel de theoretische opzet als de praktische uitvoering van een systeem dat voldoet aan de gestelde criteria. Een oplossing voor deze uitdagingen zou kunnen leiden tot een innovatieve manier waarop AI-modellen met elkaar worden gecombineerd zonder de noodzaak voor hertraining. Dit is vooral belangrijk voor kleinere teams die niet de middelen of expertise hebben om nieuwe multimodale modellen te creëren. Zij kunnen gebruikmaken van het framework dat hen in staat stelt om bestaande modellen te combineren tot een nieuw multimodaal model.

Gerelateerd Werk

Opvallend is dat er relatief weinig literatuur te vinden is over AI-gebaseerde videozoekmachines, ondanks de omvangrijke groei in AI-technologieën. De literatuur richt zich voornamelijk op verschillende technieken voor het ontwikkelen van eigen AI-modellen. Er wordt in dit hoofdstuk gefocust op het bestuderen van fundamentele technieken om AI modellen te maken. Het toont achteraf hoe complex het is om zelf een AI-model te maken.

De discussie begint met een bespreking over Cross-model leerprocessen (CML), waarbij technieken zoals het ophalen, matchen en vertalen van data gebruikt worden om de prestaties van

AI-systemen te verbeteren. Daarna wordt transfer learning besproken, een methode waarbij AI-modellen die voor een specifieke taak zijn getraind, aangepast worden voor een andere taak. Verder wordt ensemble learning behandeld, een essentiële benadering in de AI-ontwikkeling.

Ook worden technieken voor embedding besproken, waaronder vectorruimtemodellen en grafembeddings. Deze technieken zijn essentieel voor het efficiënt opslaan en opvragen van verschillende soorten multimodale embeddings. Verder worden in dit hoofdstuk foundation modellen onderzocht, vooral in de context van computervisie en natuurlijke taalverwerking (NLP). Daarnaast wordt de significante transformatie die zij teweegbrengen in de constructie van AI-modellen besproken. Er wordt gekeken naar modellen zoals InternImage en CoCa voor computervisie en gevestigde modellen in NLP zoals BERT en GPT-3, waarbij hun functionaliteiten, sterktes, en beperkingen worden belicht.

Tenslotte bespreekt dit hoofdstuk verschillende multimodale modellen, deze zijn van grote waarde voor taken die diverse soorten data vereisen, zoals tekst, afbeeldingen en audio. Deze modellen kunnen verschillende soorten informatie tegelijk verwerken en begrijpen. Dit onderscheidt hen van unimodal modellen die beperkt zijn tot een enkel datatypen.

Aanpak

Er wordt al snel opgemerkt na het lezen van het gerelateerd werk dat deze multimodale modellen de nodige performance en nauwkeurigheid bevatten om een video semantisch zoekbaar te maken. In dit hoofdstuk verkennen we ook strategieën en methoden die gebruikt worden om het ophalen van videofragmenten te verbeteren met behulp van multimodale classificatiemodellen.

Er wordt uitgebreid besproken hoe traditionele technieken, die ontworpen zijn voordat het transformer model bestond, voor het verwerken van gegevens uit video's, geüpgraded of vervangen kunnen worden door geavanceerde foundation modellen. Deze foundation modellen zijn getraind op uitgebreide datasets en zijn daardoor in staat om een breed scala aan taken uit te voeren met een opmerkelijke nauwkeurigheid en efficiëntie. De integratie van deze modellen kan de manier waarop gegevens verwerkt en geanalyseerd worden fundamenteel veranderen. Dat leidt tot een aanzienlijke verbetering van de zoekfunctionaliteit en de toegankelijkheid van videomateriaal.

De implementatie van een foundation model binnen het framework wordt besproken, inclusief de mogelijke uitdagingen en voordelen van deze methode. Al snel wordt geconcludeerd dat het ontwikkelen van een eigen foundation model geen haalbare optie is, voornamelijk vanwege de benodigde hoeveelheid data en rekenkracht. Er wordt onderzocht welke mogelijkheden er zijn om multimodale modellen te gebruiken die voldoen aan de specifieke eisen van het project en om de resultaten van foundation modellen zo goed mogelijk na te bootsen. Het hoofdstuk sluit af met een discussie omtrent deze innovatieve benadering kan leiden tot een robuuster, uitbreidbaarder, voorspelbaarder en adaptiever systeem voor het doorzoeken van video fragmenten.

Evaluatie

Dit hoofdstuk biedt een grondige evaluatie van het framework dat gebruikmaakt van multimodale modellen en retrieval technieken, ontwikkeld binnen deze thesis. In totaal zijn er 48 tests uitgevoerd, verdeeld over verschillende moeilijkheidsniveaus. Deze tests bevatten 480 individuele uitvoeringen die ontworpen zijn om de prestaties van het systeem grondig onder diverse operationele condities te analyseren. De evaluatie is gestructureerd rond specifieke prestatie-indicatoren: precisie, recall, de F1-score, en de Query Response Time (QRT).

De precisie-metric geeft aan welk percentage van de opgehaalde videosegmenten correct is, ten opzichte van het totaal aantal opgehaalde segmenten. Dit geeft een indicatie van de

nauwkeurigheid waarmee het systeem relevante resultaten selecteert. De recall meet hoeveel daadwerkelijk relevante videosegmenten er zijn teruggevonden. Dit is cruciaal voor het begrijpen van de volledigheid van de zoekresultaten. De F1-score, een harmonisch gemiddelde van precisie en recall, biedt een enkelvoudige maatstaf voor de algehele nauwkeurigheid van het systeem. Deze metric is bijzonder nuttig wanneer een evenwichtig overzicht van zowel precisie als recall noodzakelijk is.

De Query Response Time (QRT) varieert van 2,21 tot 5 seconden, afhankelijk van de complexiteit van de query en de gebruikte technologieën zoals de OpenAI GPT-3.5 turbo API. Deze metric benadrukt de tijdsefficiëntie van het systeem bij het leveren van zoekresultaten, met een secundaire focus op de nauwkeurigheid van deze resultaten.

De resultaten van deze uitgebreide tests worden kritisch beoordeeld, waarbij zowel de sterke punten als de gebieden die verbetering behoeven worden besproken. De evaluatie biedt belangrijke inzichten in de operationele capaciteiten van het systeem en legt de basis voor toekomstige verbeteringen en onderzoek.

Naast het identificeren van de prestaties en mogelijkheden van het systeem, focust dit hoofdstuk ook op de uitdagingen en beperkingen die tijdens de tests naar voren kwamen. Deze aspecten zijn cruciaal voor het verder verfijnen van de technologieën en het bieden van richtlijnen voor toekomstig onderzoek. Hierdoor worden nieuwe mogelijkheden voor het uitbreiden en verbeteren van het project voorgesteld.

Deze evaluatie vormt een essentieel onderdeel van de thesis, door het gedetailleerd beeld van de effectiviteit en efficiëntie van de ontwikkelde systemen, en speelt een belangrijke rol in de voortdurende verbetering en aanpassing van de beschreven technologieën.

Conclusie

Het zoeken in video's overstijgt het simpelweg terugvinden van fragmenten. Het betreft geavanceerde technologie die onze interactie met een breed scala aan media fundamenteel kan transformeren. Deze technologie belooft de gebruikerservaring op mediaplatforms en binnen cloudopslagsystemen aanzienlijk te verbeteren, door intuïtievere en efficiëntere manieren te bieden om videocontent te ontdekken en te raadplegen. Bovendien strekt het potentieel zich uit tot het kritieke domein van openbare veiligheid, waar geavanceerd videozoeken de bewakingsmogelijkheden kan verbeteren door snelle analyse en respons op videobeelden mogelijk te maken.

Ondanks de explosieve toename van videocontent, schieten veel cloudopslagplatforms tekort in het bieden van effectieve zoektools. Veelal beperken zij zich tot basiszoekfuncties op trefwoorden die niet voldoen aan de behoeften van moderne bedrijven. Deze beperkingen kunnen de efficiëntie hinderen en de mogelijkheden om videogegevens optimaal te benutten inperken.

Een essentieel aspect van dit onderzoek is het integreren en testen van gecombineerde multimodale modellen, die elk voor specifieke taken zijn ontworpen. Door gebruik te maken van embeddings en vector spaces, simuleren deze modellen de functionaliteit van een enkel, omvattend multimodale foundation model. Deze techniek verbetert aanzienlijk de efficiëntie, onderhoudbaarheid, voorspelbaarheid en schaalbaarheid. Hierdoor worden geavanceerde zoektools toegankelijker voor ontwikkelaarsgroepen die niet beschikken over uitgebreide kennis of de benodigde data om een eigen video foundation model te ontwikkelen.

De integratie van verschillende multimodale modellen heeft zeer nauwkeurige resultaten opgeleverd bij het ophalen van videofragmenten. De sentence parser, die de queries met hoge precisie en vrijwel real-time interpreteert, speelt een cruciale rol in het succes van het geïntegreerde systeem. Dit component zorgt ervoor dat het systeem de query effectief begrijpt en de juiste multimodale modellen inschakelt om de juiste resultaten te leveren.

Daarnaast wordt een plug-and-play strategie gehanteerd om elk type model aan de zoekmachine

te koppelen, zowel open-source modellen lasook extern gehoste closed-source modellen. Dankzij deze flexibiliteit kunnen ontwikkelaars diverse AI-technologieën benutten, snel schakelen wanneer er nieuwe modellen beschikbaar zijn en deze naadloos integreren in het ecosysteem van zoekmachines.

Contents

1 Introduction **9**

2 Related Work **11**
2.1 Machine Learning Techniques . . . 11
2.2 Embedding Techniques . . . 15
2.3 Foundation Models . . . 17

3 Approach **20**
3.1 Extracting . . . 20
3.1.1 Traditional techniques . . . 20
3.1.2 Foundation models . . . 27
3.1.3 Upgrading the traditional techniques . . . 28
3.2 Storing . . . 32
3.2.1 Embeddings . . . 32
3.2.2 Vector databases . . . 34
3.3 Querying . . . 37
3.3.1 Parsing . . . 37
3.3.2 Processing and enhancing . . . 38
3.3.3 Merging . . . 39
3.4 Implementation . . . 40
3.4.1 Uploading . . . 40
3.4.2 Indexing . . . 40
3.4.3 Searching . . . 42

4 Evaluation **45**
4.1 Methodology . . . 45
4.1.1 Objectives . . . 45
4.1.2 Dataset . . . 45
4.1.3 Metrics and criteria . . . 46
4.2 Results . . . 47
4.2.1 System information . . . 47
4.2.2 Video processing . . . 47
4.2.3 Query processing . . . 48
4.3 Discussion . . . 58
4.3.1 Surprises . . . 58
4.3.2 Limitations and challenges . . . 58
4.3.3 Future work . . . 59

5 Conclusion **60**

Chapter 1

Introduction

In the spectrum of digital media storage, many files are becoming increasingly multimedia-based, prominently including videos and photos alongside traditional text-based files like Word, PowerPoint, and Excel. While cloud storage solutions and Digital Asset Management (DAM) systems have excelled in organizing and retrieving text-based content through search functionalities, they often neglect the complex nature of video and photo files. Although DAM systems have adopted tagging strategies to make photos searchable, this method proves inefficient for videos. Video contains a dynamic range of information across different frames, making it inadequate to capture video data's essence and full content with simple tags.

Developing an effective video search engine presents hard challenges due to the complex and multifaceted nature of video data, containing diverse features such as text, ambient sounds, speech, actions, persons, logos, and landmarks. Despite the surge in AI research this decade, there is a relative scarcity of academic literature addressing these complex video data points. Building a robust video retrieval model typically requires training extensive datasets using/creating foundation models, which are adept at processing and understanding this varied information. However, the challenges extend beyond merely collecting and processing this diverse data; maintaining the system's debuggability, extensibility, and manageability also poses significant difficulties. These requirements complicate the development of a technically sophisticated and straightforward system.

In this thesis, various problems are discussed and addressed:

1. Searchable: How video clips become semantically searchable based on the actions, persons, sounds, language, and text in the video.
2. Fusing: How AI models can be combined (fused) without retraining them.
3. Anti-blackbox: How the predictability and debuggability of fused AI models can be increased.
4. Plug & Play: How developers can be facilitated to quickly add new models or modalities to the fused AI model.

Current approaches to advanced video retrieval include efforts by companies like TwelveLabs, which developed a proprietary video foundation model. This model represents a significant advancement in the field, supported by a considerable team and substantial funding. However, their models are closed-source, difficult to debug, and not easily extendable with new modalities. Moreover, the complexity of these foundation models often results in a "black box" scenario, where developers have limited understanding and control over the model's internal workings and decision-making processes.

This thesis proposes a novel framework designed to overcome the limitations of existing video

search technologies by allowing for modular integration of diverse AI models, whether open-source or proprietary. This "plug and play" framework aims to be highly extendable from the get-go, enabling continuous improvements and expansions as new models and technologies emerge. Each component within the framework can autonomously determine whether to engage with a specific query, ensuring that all aspects of video data are analyzed and considered. The goal is to present an integrated system that operates with the comprehensive capabilities of a foundation model yet maintains full transparency and debuggability, avoiding the pitfalls of real foundation models. This approach enhances the functionality and applicability of video search engines and ensures they remain adaptable and maintainable in the face of evolving digital media landscapes.

Chapter 2

Related Work

This chapter explores several fundamental concepts and methodologies for developing artificial intelligence models. It begins with examining cross-modal learning (CML), discussing how techniques such as retrieval, matching, and translation are utilized to enhance AI systems. Following this, the chapter addresses transfer learning, a strategy that repurposes AI models trained for one task to be suitable for another. It explores various types of transfer learning, highlighting their benefits and the challenges they introduce.

Additionally, ensemble learning is explored, which plays a pivotal role in AI. This section analyzes the primary methods of ensemble learning—bagging, boosting, and stacking—and demonstrates how each contributes to the robust development of AI models for classifying and interpreting data. The discussion also covers embedding techniques, including vector space models and graph embeddings. These tools are vital for managing different types of AI data, ensuring that models perform accurately and efficiently.

Moreover, the chapter examines foundation models, particularly in computer vision and natural language processing (NLP), noting a significant transformation in AI model construction. It reviews models such as InternImage and CoCa for computer vision and established models in NLP like BERT and GPT-3, discussing their functionalities, strengths, and limitations and how they have revolutionized the design of AI models. Lastly, the chapter discusses multimodal models, which are invaluable for tasks that require an understanding of diverse types of data, such as text, images, and audio.

2.1 Machine Learning Techniques

Cross-Modal Learning

In recent years, there have been significant developments in cross-modal learning (CML), which aims to learn a stronger representation by fusing multiple modalities. CML is a technique that allows the exchange of information between different modalities, such as text, images, audio, and videos, in order to improve classification performance. This approach is becoming very important for building new AI systems. CML techniques have been utilized in various applications such as image captioning, speech recognition, and video analysis (Zhang et al., 2020). CML contains a range of techniques and methodologies, each addressing the challenges presented by multimodal data. This section delves into various parts CML contains, including retrieval, matching, translation, distillation and hashing, highlighting their roles and individual contributions to CML.

Cross-modal retrieval (CMR) stands as a cornerstone of CML, as it involves retrieving relevant information from one modality based on a query from another modality. This aspect is particularly critical in environments with a lot of multimedia content, where the need to bridge

different types of data, like images and text, is essential. However the ability to retrieve relevant information accurately and efficiently is directly dependent on the quality of the Cross-modal matching (CMM). CMM refers to the process of finding similarities between different modalities and linking them together (K. Wang et al., 2016). For example, it might involve linking a textual description to a corresponding image or vice versa.

Moving beyond matching and linking, cross-modal translation (CMT) transforms data from one modality to another. This process is vital for tasks where modalities have inherent differences, such as translating from text to image or translating a book to a audio format. Cross-modal hashing (CMH) and destillation addresses the need for efficient data retrieval in multimedia databases with lots of data. By converting data from various modalities into a shared binary code, hashing facilitates quicker and more efficient searching and matching processes. Ultimately, this helps to reduce the computational burden and improve the scalability of cross-modal without sacrificing accuracy (Peng et al., 2018).

Together, these facets of cross-modal learning – retrieval, matching, translation, distillation, and hashing – form a comprehensive framework that enhances the performance of AI systems in handling multimodal data.

CML systems often involve involve creating a shared representation space where different modalities can be compared and matched. Neural network architectures like Convolutional Neural Networks (CNNs) for image data and Recurrent Neural Networks (RNNs) for text data are frequently used in these systems. These architectures are trained using large multimodal datasets, where both the input data and their corresponding labels or annotations are available for each modality. CML can both be trained using supervised learning and unsupervised learning approaches. For supervised learning, the training data includes pairs of multimodal inputs and their corresponding labels, which are used to train the model to predict or classify new inputs (Cangea et al., 2017). For unsupervised learning, the training data consists of multimodal inputs without explicit labels. It will try to learn the underlying structure and patterns in the data without any specific guidance.

Transfer Learning

Transfer learning is a machine learning technique that involves utilizing a pre-trained model as the foundation for addressing a new task or problem. This method effectively transfers knowledge from one task to another (Zhuang et al., 2021). It proves particularly beneficial in situations where data for the target task is sparse, and training a model from scratch would be impractical or inefficient. Transfer learning can be categorized into two approaches: data-based transfer learning and model-based transfer learning.

In data-based transfer learning, the data from the initial task is modified to enhance its applicability to the new task. This modification could involve altering the significance of certain data points or the manner in which they are labeled.

Conversely, model-based transfer learning involves adapting a pre-trained model—originally developed for one task—to better suit another task (Jiang et al., 2022). This approach leverages the foundational knowledge embedded in the model, making it more relevant and effective for the subsequent application.

When discussing transfer learning, various types are distinguished based on the similarity between the data and tasks of the source and target domains, as well as the nature of the available data:

1. Homogeneous transfer learning occurs when both the data and tasks in the source and target domains are similar, with labeled data available for both. This is analogous to learning to recognize different types of cats after mastering dog recognition (Zhuang et al., 2019). The fundamental concept of identifying animals remains consistent.

2. Heterogeneous transfer learning takes place when the data and tasks in the source and target domains differ, yet labeled data are available for both. This can be likened to applying knowledge from animal recognition to plant identification. While the tasks vary, some underlying techniques or features may still be applicable (Hosna et al., 2022).
3. Inductive transfer learning merges the use of labels for a new task with the principles of inductive reasoning, which alters the conventional learning sequence. For instance, one might receive images of animals, some of which are cats and some not, without labels. By examining these pictures, one learns to identify distinctive features of cats (pointed ears, tails, etc.) and can then apply this knowledge to recognize cats in new, unlabeled images.
4. Transductive transfer learning is characterized by having labels only for the original task but not for the new task. Imagine needing to identify a new type of animal without labels, having previously trained a model on other animals. The insights gained from the existing model are then utilized to make predictions about the new, unlabeled data.
5. Unsupervised transfer learning describes scenarios where there are no labels for either the original or new tasks. This is akin to attempting to detect various animals and plants without any labels or names for guidance.

These types of transfer learning explain how models can adapt their acquired knowledge to different scenarios. In some cases, such as with homogeneous transfer learning, the model may directly apply its knowledge. In contrast, scenarios like heterogeneous transfer learning might require more significant adjustments to the model. Despite these capabilities, transfer learning also presents challenges. At times, the knowledge previously acquired by the model could detrimentally affect its performance on a new task, mainly when the tasks are distinctly different. Furthermore, the availability and quality of labeled data in both the source and target domains play a crucial role in determining the effectiveness of transfer learning (Farahani et al., 2020).

There are several strategies to adapt a model to a new task effectively. One key strategy is feature transfer, which involves using a model's knowledge gained from its original training as a foundation for the new task. Essentially, the source model is trained on a large and diverse dataset, which learns certain features or patterns. When faced with a new task, these learned features can serve as a solid foundation for the new model to build upon (Zhuang et al., 2021). For example, a model trained to identify objects in pictures might have learned to recognize shapes and textures. This knowledge can be useful for a new task, like identifying specific types of objects, because the model already understands the basic idea of recognizing shapes and textures. Another strategy is fine-tuning where a pre-trained model is used as a base and continuing training it on the new task-specific dataset. The idea behind it is to make small, careful adjustments to the source model so that it adapts to the new task without losing the knowledge it gained from the original training. For example, if our object-recognizing model is being fine-tuned to recognize types of birds, we don't want the new model to forget how to recognize objects in general completely. So, the training adjustments are made gently, often with a lower learning rate, to ensure that the model adapts to recognizing birds while keeping its ability to identity general objects ("Fine-Tuning the Model: What, Why, and How", 2023). Last but not least, freezing layers is a strategy often used alongside fine-tuning. In many models, the early layers (layers closest to the input) are responsible for extracting basic, universal features like edges in images or common sounds in an audio file. These features are useful across a wide range of tasks. Therefore, freezing these early layers and only updating the later layers that are task-specific can help preserve the learned universal features while allowing the model to specialize in the new task ("What is layer freezing in transfer learning?", 2023).

Ensemble Learning

Ensemble learning is used in machine learning to combine multiple different models to make one really powerful model. These models, known as base learners, are strategically generated and then combined into a composite learner or a so-called 'ensemble'. It's like taking the best parts

of each model and creating a unified and enhanced model that outperforms each individual model. It helps balancing out their individual strengths and weaknesses, leading to improved overall performance (Mienye & Sun, 2022). Overall, this method is really useful for a few reasons. First, it makes fewer mistakes. Techniques like 'Random Forests' and 'AdaBoost' help make predictions more accurate by fixing specific types of errors. Another great thing about ensemble learning is that by using predictions from different models, it creates a system that's not easily confused by unusual data. This makes it great for dealing with new types of data it hasn't seen before. Plus, it's really good at understanding complex or uneven datasets. Since it uses a mix of different models, each one looks at the problem in its own unique way, giving an ensemble model a more comprehensive understanding ("Ensemble Methods: A Beginner's Guide", 2020).

When turning multiple models into a single ensemble model, some different techniques and approaches can be used (Alqurashi & Wang, 2018). There are currently three main methods for creating ensemble models: bagging, boosting, and stacking. You can use these methods together to get better results.

The first method is called bagging, it stands for bootstrap aggregating (Mienye & Sun, 2022). By using this method each model gets trained on a different subset of the training data, which is created through random sampling with replacement. A simple analogy for bagging is to think of a bag with colored balls. Each ball is a piece of data. Each time you train a model, you pick a bunch of balls from the bag; when you're done, you put them back in the bag. It's possible that a model is trained on the same data multiple times, and some data may not be used at all. The most used algorithm for bagging is called random forest, where many decision trees are made using these different subsets of data. The final answer of an ensemble model depends on the output. If the output is numerical (regression task), it is often averaged or combined through a weighted average (X. Wang & Davidson, 2010). When the output is a category (classification task), the ensemble model can use voting techniques such as majority voting or weighted voting to determine the final prediction (W. Li et al., 2021).

Boosting is a different approach to creating ensemble models. It trains models one after the other, and each new model focuses on the mistakes the previous ones made. Think of the ball example again, but this time, it gets bigger whenever a ball is guessed wrong, so it's more likely to be picked next time. This way, the new model pays more attention to the data that was guessed wrong before. The models, called learners, are trained sequentially, and each learner tries to improve the weaknesses of their predecessors. Ultimately, the weak learners are put together, so they become so-called strong learners. Boosting algorithms, such as AdaBoost and gradient boosting, where each new model slowly improves the whole group (Mienye & Sun, 2022).

The last method that can be used is called stacking. Stacking is an advanced ensemble learning technique that involves training multiple models to make predictions and then combining their outputs using another model called a meta-model (W. Li et al., 2021). It's like building a multi-layer tower. Each layer, or model, looks at different things in the data. The important part is that the results of one layer are used as the starting point for the next layer, the meta-model. This meta-model, or blender, uses the first models' results to make the final guess. Which method to use depends on the problem, the kind of data, and what you want to achieve. Each method has its own way of using the power of multiple models to get better results than just one model could.

When choosing for ensemble methods in AI classification, it is important to consider the diversity aspect as well (Mienye & Sun, 2022). Each model used in the ensemble has its own strengths, so when you use a bunch a different models, they can all add their unique views. This is great for making fewer mistakes, seeing different sides of the data, and being good at dealing with new situations. Diverse models are also good at noticing a wider range of patterns in the data. This helps to avoid focusing too much on specific details (overfitting) and smooths out little errors or noise, making the main trends or signals clearer. There are several ways to

make your models diverse. You can start them off differently, use different types of models, train them on different parts of the data or with different features, change their settings, use error-fixing tricks, or add some randomness. A good example of this is the Random Forest algorithm, which uses a lot of decision trees with a bit of randomness to get a wide range of views. But it's important not to go too far with diversity. If the models are too different, each one might not be very accurate on its own, and that can make the whole group work worse. The goal is to find a good mix where the models are different enough to add new insights but still accurate enough on their own. This balance helps the whole ensemble work better together (Konstantinov & Utkin, 2020).

Combining AI classification models into an ensemble can be tricky. There are a few big challenges to think about, like making sure the models work well together and dealing with their differences. One issue is that different models might use different scales or ways of representing the data (Mienye & Sun, 2022). This can create inconsistencies in the output of each model and make it difficult to effectively combine their predictions (W. Li et al., 2021). This means you have to adjust them to a common scale and make sure they understand the classes in the same way.

Deciding how much weight to give each model's decision is tough too, especially since they might perform differently depending on the features they're using. If the models were trained on different data or used different steps to get ready for training, you have to figure out how to make them compatible. Also, training multiple complex models and combining them quickly enough for real-time predictions can be difficult. This requires a lot of computing power. Even though ensembles are usually less likely to overfit (fit too closely to the training data), you still need to be careful about this to make sure the ensemble model works well on new, unseen data.

2.2 Embedding Techniques

Vector Space Models

Vector space models (VSM) are foundational in the domain of natural language processing (NLP) and are extensively utilized for representing text data in AI classification models. These models have a long research and development history, initially focusing on text representation. Over time, vector space models have undergone significant evolution. Recent advancements in embedding techniques have not only improved the representation of text data but have also revolutionized the handling of other data types, such as images and audio, through vectorization. This progression represents a substantial advancement from their initial applications, demonstrating their adaptability and expanding role in AI and machine learning. Integrating vector space models and embedding techniques has proven highly effective in AI classification models (Turney & Pantel, 2010).

VSMs are pivotal in representing textual data. They transform text into vectors within a high-dimensional space, facilitating the discovery of semantic relationships. The fundamental premise is that words appearing in similar contexts will likely share meanings. By analyzing word frequencies within a text corpus, VSMs encapsulate semantic information, enabling textual data classification. This chapter will further delve into various techniques used to develop vector space models in natural language processing.

Term-Document Matrices (TDMs), Word-Context Matrices (WCMs), and Pair-Pattern Matrices (PPMs) form the backbone of Vector Space Models in natural language processing, each serving a distinct but complementary purpose. TDMs are a key component in information retrieval systems, such as Apache Lucene, focusing on the relationship between terms and the documents they appear in. They are structured like a grid, with each row representing a different term and each column representing a different document. The value in each matrix cell indicates how often a term appears in a document. This frequency is crucial in determining how relevant a document is to a specific search query. When a query is entered, the system utilizes

the TDMs to calculate document relevance scores, assessing how well each document matches the search terms. This process often involves techniques like cosine similarity or TF-IDF (term frequency-inverse document frequency), which not only count how frequently a term appears but also consider its overall importance in the whole set of documents. This approach enables the effective matching of documents to query terms, enhancing the precision of search results in information retrieval systems. (Erk, 2012)

In contrast to other models, WCMs specifically focus on the contexts where words appear. They capture the meanings of words by looking at how they are used in various situations. These contexts can be as simple as the words next to the target word or as complex as broader linguistic patterns. However, a key challenge with WCMs is that they can be very large and complex, which makes them computationally demanding. Techniques like random projection are employed to manage this. Random projection is a method that simplifies these large matrices. It reduces the number of details the matrix needs to handle but does so in a way that keeps the important semantic connections between words. This approach finds a middle ground, maintaining the depth of meaning captured by the matrix while making it easier and faster to work with. (Turney & Pantel, 2010)

PPMs are a unique type of matrix used for identifying and illustrating complex relationships and analogies in text. These matrices differ from the previously mentioned types as they concentrate on how pairs of words connect within different linguistic patterns or contexts. They map out how two words are related to each other, which is particularly beneficial for tasks that involve understanding analogies or the deeper semantics of relationships between words. Using these matrices makes it possible to uncover how word pairs interact, going beyond their individual meanings to explore their interconnectedness in the context of language use. It's not an easy task to use PPMs effectively because of words that have multiple interpretations or meanings, leading to ambiguity in the matrix representation. To counteract this, you can use smoothing techniques or a bigger dataset to capture a wider range of contexts and disambiguate the meanings of words. This on the other hand, will increase the computational complexity of working with PPMs, which will result in more data sparsity and computational cost.

Graph Embeddings

Graph embeddings are a key technique for representing and analyzing complex relationships between entities in a graph structure. This type of data is common in areas like social networks or transportation systems, where the entities (like people or cities) are the nodes of the graph, and their connections (like friendships or transportation routes) are the edges (Doh et al., 2022). Most of the time, these graphs are complex and high-dimensional, which makes it hard for regular machine learning algorithms to work with them effectively.

To solve this, graph embeddings transform the graph into a simpler form by representing each node, and sometimes the edges, as vectors in a lower-dimensional space. This representation captures the structural and relational information of the graph, allowing for more efficient and effective analysis and prediction tasks. An example of such an algorithm is the DeepWalk algorithm. It uses random paths through the graph to create these vector representations. This approach is particularly helpful for making sense of social networks, where it can capture complex relationships in a format that's easier for algorithms to process. Graph embeddings are extremely valuable in AI classification models, which are used to categorize and make sense of data. They serve as a kind of 'feature input', enhancing the accuracy and performance of the classification models by capturing the underlying relationships and structures within the data. For instance, embeddings from graphs (e.g. user-item interactions) can lead to better recommendations in a system that recommends products to users based on their past purchases. Additionally, graph embeddings help combine different types of AI models, each designed for various kinds of data. This is especially useful when these models need to work together, like in systems that handle text, tables, and graphs all at once.

Graph embeddings also support tasks like transfer learning and multitask learning. They pro-

vide a common format that can be reused for different kinds of tasks, reducing the need to build new models for each task. Plus, they make it easier to understand and visualize complex networks, which is important for figuring out how a model is making its decisions. Comparing different types of embeddings in machine learning, like graph embeddings, knowledge graph embeddings, and image representation embeddings, there can be noticed some clear differences. Knowledge graph embeddings mainly convert complex and high-dimensional graphs into simpler, lower-dimensional spaces. This is important because it helps keep the structure and connections that are originally present in these graphs. A good example of this is the TransE model, which is designed to embed entities and relationships from knowledge graphs into a simpler space (Bordes et al., 2013). On the flip side, learning how to represent images, another type of embedding, takes a different route. It often uses unsupervised methods, meaning the model learns independently without explicit instructions. Techniques like contrastive learning (not discussed in this master thesis's related work), where the model learns by differentiating between similar and dissimilar images, and clustering, grouping similar images together, are common in this field (Lin et al., 2022). Graph embeddings, in particular, are made to handle the specific challenges and details found in knowledge graphs. They're great at capturing the unique features and connections within these graphs. Image representation embeddings, however, are developed with different goals in mind, mainly to work well with various types of image data. So, each kind of embedding has its own role and speciality, depending on what kind of data it's dealing with and what the learning process aims to achieve. One of the key disadvantages of graph embeddings is their high computational and storage requirements, making them resource-intensive. Moreover, their applicability tends to be more focused on knowledge graphs, which can limit their use in other data types. Also, when compared to other embedding types, graph embeddings can be more complex to implement and understand (Gesese et al., 2019).

The process of learning these graph embeddings can follow either a joint or independent approach. In joint learning, the embeddings for entities and relations are learned simultaneously, allowing them to influence and enhance each other. This method can lead to a more integrated understanding of the graph's structure. In contrast, independent learning treats the entity and relation embeddings separately, which can offer more flexibility and might be more efficient in certain scenarios. The choice between these two methods significantly affects the quality of the embeddings and their effectiveness in predicting additional relations within knowledge graphs.

2.3 Foundation Models

Computer Vision Models

The introduction of foundation models within the scope of computer vision has marked a transformative shift from traditional approaches. These models are known for their extensive pretraining on large-scale datasets, which allows them to learn generic visual representations and understanding. This extensive pretraining enables these models to adapt quickly to a broad range of computer vision tasks, resulting in a notable departure from traditional models typically designed for specific tasks and requiring significant training data. For example, a foundation model trained on diverse images, from cityscapes to abstract paintings, can later be applied to tasks such as facial recognition or object detection with remarkable adaptability. This flexibility, inherent in foundation models, allows them to operate efficiently across various domains in computer vision.

InternImage and CoCa have made notable contributions to model design and adaptability among these foundation models. InternImage, a CNN-based model, incorporates deformable convolutions, enabling it to learn complex patterns beyond the capabilities of standard CNNs. This is evident in its ability to recognize objects under varied poses and lighting conditions, a task that traditional CNNs often find challenging. CoCa, on the other hand, adopts a minimalist design and functions as an image-text encoder-decoder. It's unique for using both contrastive

loss and captioning loss in training, thus excelling in tasks that require an integration of visual and linguistic processing (IBM, 2023).

Despite these advantages, foundation models are not without limitations. Their performance may falter if not pre-trained with relevant data in specialized areas like plant phenotyping. This issue is particularly critical in fields where precision and accuracy are crucial, such as medical diagnosis. Furthermore, the versatility of foundation models can be constrained when faced with tasks, data types, or domains significantly different from their training data. To address these challenges, researchers have explored various techniques to enhance the adaptability of foundation models to specific domains. Fine-tuning on domain-specific datasets using methods like adapter tuning or decoder tuning is one such approach. This enables more precise adaptation to specialized tasks (Rohit, 2023). Another method is adversarial domain adaptation with confounder balancing, which helps the models to learn domain-invariant representations, which is vital for applications with varying domain requirements.

A prime example of this adaptability is seen in models like SAM and CLIP, which have been successfully adapted for complex tasks such as 3D point cloud segmentation. This demonstrates the potential of foundation models to extend their application beyond traditional tasks, showcasing their flexibility and robustness in a variety of scenarios.

Natural Language Processing Models

Examining how Natural Language Processing (NLP) models function is crucial, although the relevance to assembling various AI models might not be immediately apparent. This connection will become more straightforward as the discussion of the project in this master thesis unfolds. Understanding the most powerful foundation models in AI, such as BERT and GPT-3, is essential. These models have significantly altered the landscape of AI construction and operation. They are versatile and capable of performing numerous tasks without needing to be trained from scratch each time. This flexibility is particularly beneficial for tasks such as utilizing a search engine to find and understand information.

The utilization of pre-training methods in information retrieval has been thoroughly investigated and plays a crucial role in the development of language models. Models like BERT and GPT-3, which are pre-trained on extensive datasets, have streamlined a variety of tasks, including rephrasing search queries, identifying user intentions, summarizing documents, and generating concise text previews. Their comprehensive training allows them to capture the subtleties of language, thus enhancing the ability of search engines to provide more customized and accurate results. However, these advancements come with their own set of challenges, including complexities in comprehending how these models function and make decisions (Fan et al., 2021).

Multimodal Models

Until now, the discussion has primarily focused on unimodal models, which handle a single type of data. However, for many applications, these models may not adequately capture the richness and complexity of the data involved. This limitation gives rise to the relevance of multimodal models. Multimodal models are adept at integrating various types of data, such as text, images, and audio, surpassing the capabilities of unimodal models that are confined to a single data format. These models are distinctive because they can process and comprehend different types of information simultaneously.

For example, consider a social media post. A multimodal model does not merely analyze the text and images separately; instead, it synthesizes these elements to form a more holistic understanding. This comprehensive approach is particularly beneficial in scenarios where grasping the full context or background is crucial. Multimodal models generate a single vector (or embedding) that encapsulates the integrated comprehension of the different modalities. This unified representation can then be utilized for a range of downstream tasks, such as sentiment analysis,

image captioning, or video summarization, thereby enhancing the model's applicability and effectiveness.

In this master's thesis, we will look into some examples of multimodal models and how they can be combined to enhance classification tasks. But before diving into that, I would like to introduce the 2 most important advances in multimodal model development to date which is OpenAI's CLIP and Google's Multimodal Transfer. CLIP excels in understanding and correlating images with text, revolutionizing image recognition by not just identifying objects but also comprehending the surrounding context (Dirik & Paul, 2023). Google's Multimodal Transformer further advances this integration by simultaneously processing text, audio, and visual data for a rounded interpretation. Further in this master thesis, we will explore how these multimodal models can be combined and leveraged for enhanced classification tasks.

In this master's thesis, the focus will extend to various examples of multimodal models and their potential to enhance classification tasks. Before diving deeper into these applications, it is pertinent to introduce the two most significant advancements in developing multimodal models: OpenAI's CLIP and Google's Multimodal Transformer.

CLIP, developed by OpenAI, excels in correlating and understanding images with text (they form a tuple), revolutionizing image recognition. It goes beyond merely identifying objects by comprehending the contextual nuances surrounding them (Dirik & Paul, 2023). On the other hand, Google's Multimodal Transformer represents a further advancement in this area by processing text, audio, and visual data simultaneously, which allows for a more comprehensive interpretation of the data. Throughout this thesis, more explanation will be given to how these innovative multimodal models can be synergistically combined and leveraged to significantly enhance the effectiveness of classification tasks, illustrating their utility and transformative potential in various applications.

In the related work section of this thesis, three principal methods for integrating artificial intelligence models are discussed: ensemble methods, transfer learning, and model stacking. Simply put, ensemble methods involve combining various models, each contributing to the final decision-making process. This technique leverages the strengths of multiple models to improve overall performance and reduce the likelihood of errors that might occur if relying on a single model. In comparison, transfer learning involves adapting a model that has been previously trained for one task to perform a similar but distinct task. This approach saves significant time and resources spent training a new model from scratch. Lastly, model stacking refers to a method where the outputs of one model are used as inputs for another model, effectively creating a layered structure.

Data fusion is essential in multimodal AI systems, which deal with different kinds of data. The first kind, feature-level fusion, is about mixing basic data elements from different sources right at the beginning. This might mean combining text, images, and sounds into one combined data set. Doing this gives the AI a more detailed and varied set of information to start with. This method is great when you want to use each type of data's unique insights. Then, there is decision-level fusion, where separate models are trained on different data types to make their own decisions. These decisions are then brought together into one final decision. This method uses the strengths of each model, like one that's good at understanding images and another that's good at understanding text. The final decision you get is usually better than what any single model could come up with alone. Lastly, model-level fusion is a step-by-step process where different models handle their own data types and then pass on their results to another model. This method is good for complex tasks that need detailed analysis, building on what each specialized model brings to the table ("Multimodal Machine Learning: Data Fusion", 2022).

Chapter 3

Approach

This thesis chapter explores strategies and methods for enhancing video segment retrieval using AI classification models. It explains various algorithms, techniques, and models, setting the stage for an understanding of the implementation choices discussed later. The chapter examines traditional video analysis techniques, analyzing their strengths and limitations to provide valuable context and highlight recent innovations.

From this foundation, the discussion transitions to introducing foundation models—powerful AI models trained on extensive datasets—and their potential to revolutionize video analysis. At the core of this chapter is the approach that combines the capabilities of foundation models to simulate the strengths of a multi-modal foundation model. The algorithms developed are outlined, explaining how they work together to enhance the system's retrieval performance. This includes discussions on intersection algorithms, rank aggregation methods, and weighted voting systems, each tailored to address specific challenges in video segment retrieval.

3.1 Extracting

3.1.1 Traditional techniques

This section outlines the steps to develop a framework for extracting information from various AI classification models. Understanding the methods used by these models is crucial for successful integration. Over the years, experts have developed methods to extract data from diverse sources, including text, images, and audio. Recent advancements have significantly improved the processing of non-text data, particularly in visual and speech recognition domains.

The focus initially falls on visual analysis, commonly called computer vision. This exploration covers four main areas: motion analysis, object detection and recognition, face recognition and analysis, and content analysis. These topics are vital for identifying and understanding the content within images and videos. Although these areas have been established for some time and have produced robust results even before the advent of newer transformer models, introducing advanced models has greatly enhanced our capabilities in these fields.

A thorough understanding of how visual data is processed and analyzed is essential. This knowledge will be crucial in effectively combining different AI classification models in the next chapter. This thesis aims to seamlessly merge various AI classification models without the hurdles of retraining or integration issues. Therefore, comprehending the analysis and output of visual data is a critical component of our framework.

Motion analysis

One of the principal challenges in video analysis techniques involves understanding what's happening within a video. The study of how movement is interpreted by a model is referred to as motion analysis (Zhao et al., 2019). The goal is to enable users to find specific scenes within a video, such as those featuring break dancing or football plays. Understanding the dynamics within a scene is essential, it's similar to the technology employed in security cameras to detect anomalies or in sports to analyze player movements (Ma et al., 2022). The foundation of motion analysis is straightforward; it involves processing video frames to analyze the trajectories of objects within them. Historically, several methodologies have been developed for this purpose. Optical flow is one such method, concentrating on minute segments of a video frame at the pixel level to observe shifts from one frame to the next. This data is then cataloged and utilized in a machine-learning environment to train video recognition models (Vitor Guizilini & Gaidon, 2022). Another technique, background subtraction, involves comparing a static background with the current frame to identify moving objects. This approach allows the detection of changes within the frame, facilitating the identification of motion and objects. To develop a motion analysis algorithm, the process involves several key steps. Initially, the video must be clarified, removing any distracting noise. Subsequently, techniques like optical flow or background subtraction are employed to ascertain the movement within the video. These moving objects are then tracked over a series of frames. Ultimately, this accumulated motion data is analyzed to interpret the activities depicted in the video (Leal-Taixé et al., 2016). In our research, motion analysis is utilized in conjunction with content analysis to highlight sections of the video characterized by significant action or change. These segments typically represent critical events or actions and require meticulous analysis by our system to extract pertinent information.

Currently, the development of a motion analysis model from scratch is unnecessary, given the availability of sophisticated algorithms in libraries such as OpenCV and Pytorch. These tools are refined and tested extensively across various software applications, obviating the need for bespoke solutions. However, even advanced libraries like Pytorch and OpenCV are not without their limitations and edge cases. Challenges such as variable lighting conditions, exemplified by sudden changes like sun flares or scenes with rapid movement, may evade the standard processing capabilities of these libraries. These issues can be mitigated through the implementation of advanced algorithms and the fine-tuning of settings, although such adjustments may increase the complexity of the system and impact its predictability and effectiveness, subsequently affecting the evaluation metrics of the framework.

In contexts where the speed of search results is crucial, the trade-off between response times and output quality becomes significant. As generative prompting systems gain traction, user tolerance for slower response times appears to increase, provided that the quality of the results improves. This shifting user expectation underscores the importance of balancing speed and quality, a topic that will be explored further in the evaluation section of the discussion.

Object detection and recognition

To search within a video, it is essential to identify the objects present. This section explores the fundamentals of object recognition and evaluates its potential role in our framework. Object recognition facilitates granular video content analysis by accurately categorizing and retrieving video segments based on specific object contents. Typically, object recognition is applied to still images rather than videos. Thus, there is a need to adapt object recognition algorithms that primarily deal with images for video analysis. This adaptation often involves generating still images from video frames at regular intervals. However, merely applying traditional object recognition algorithms designed for still images may not adequately address the challenges posed by the dynamic nature of videos. Consequently, this discussion will focus on object recognition techniques tailored to video analysis. These techniques consider factors such as temporal coherence and motion cues, which are crucial for effective video analysis.

At the heart of object recognition are various algorithms and models, each contributing uniquely to the task. Convolutional Neural Networks (CNNs) are commonly utilized for object recognition because they are capable of learning hierarchical features from images or video frames. These features encompass both low-level visual patterns such as edges and textures, and high-level semantic information, including object categories and attributes. Additionally, the Region-based Convolutional Neural Network (R-CNN) family of algorithms has gained prominence for adding a layer of precision in object localization within video frames. Given the complexity and resources required, developing an object recognition algorithm from scratch specifically for video analysis is impractical. The field is already rich with research, and existing open-source algorithms and models can effectively be employed for video object recognition. Among these, YOLOv8 (You Only Look Once) is particularly notable as a state-of-the-art real-time object detection system, renowned for its speed and accuracy. YOLOv8 demonstrates exceptional performance across various benchmarks, achieving frame rates ranging from as low as five frames per second up to 160 FPS. Moreover, it outperforms both transformer-based and convolutional-based detectors in speed and accuracy, surpassing other iterations of YOLO and various models (C.-Y. Wang et al., 2022).

Figure 3.1: Example of YOLO's output detecting several basic objects

The process of object recognition in video content is organized into three distinct phases. The initial phase, input preprocessing (1), involves preparing raw video frames for analysis. This preparation typically includes resizing, normalizing, and optionally augmenting the frames to ensure consistency and enhance the model's capacity to detect various objects under different conditions. After preprocessing, the feature extraction phase follows, where key characteristics and common attributes of objects within the frames are identified. This phase is vital as it shapes the quality of data fed into the subsequent stages.

The second phase, object localization (2), entails determining the position of objects within the video frame. This step is particularly crucial in video contexts where objects may move or change positions over time. Following localization, object classification occurs, whereby the identified objects are categorized into predefined classes based on their features. This stage is critical for effective video retrieval, as accurate classification is essential.

The final phase, post-processing (3), focuses on refining the outcomes from the previous stages to enhance accuracy and minimize false positives. Techniques such as applying specific thresholds, using non-maximum suppression to handle overlapping detections, and conducting contextual

analysis to validate the appropriateness of object classifications within the broader frame context are implemented during this phase.

Frameworks such as Pytorch and YOLOv8 are already optimized for video input, which means that the stages of object recognition are seamlessly integrated within the YOLOv8 framework. This integration facilitates the implementation of real-time object recognition on video data. YOLOv8 is capable of both detecting and classifying objects in video frames, providing a comprehensive solution for video-based object recognition. It is essential to acknowledge that object recognition serves as the cornerstone of our technology's ability to comprehend and interact with the visual world. The accuracy with which it recognizes and classifies objects directly influences the performance of various subsections within our framework; thus, selecting a reliable and precise object recognition model like YOLOv8 is critical for ensuring the system's overall effectiveness.

Regarding system performance metrics, both retrieval time and processing time are crucial for evaluating our system's performance. Currently, YOLOv8 is unmatched, offering real-time performance in processing video with high accuracy. However, this accuracy and swift processing capability are accompanied by a limitation in the variety of object categories it can recognize. Furthermore, the model's ability to recognize objects across different domains or contexts is paramount, often more critical than processing or retrieval speed. To overcome these limitations, alternative models and techniques will be explored in an upcoming section on foundation models (C.-Y. Wang et al., 2022).

Face recognition and analysis

Searching through videos efficiently often requires the quick identification of specific individuals, a task facilitated by face recognition and analysis. This area has been extensively researched, resulting in many algorithms, implementations, and models developed over the years. These models span from traditional techniques like Eigenfaces and Fisherfaces to more recent deep learning-based methods such as FaceNet and VGG-Face. Upon examining these solutions, it becomes evident that the most robust face recognition frameworks also require longer processing times. Consequently, there is an inherent trade-off between processing time and face recognition accuracy. One potential strategy to mitigate the limitations and trade-offs associated with various object recognition models is to integrate multiple models that excel in different aspects. The models identified as candidates for integration into our framework include FaceNet, YOLOv8, RetinaFace, MTCNN, and OpenCV's face recognition module. Each of these frameworks has its own set of strengths and weaknesses; however, by strategically combining them, it is possible to harness the best features of each and enhance the overall performance of our system. Further exploration of these models will provide deeper insights into their capabilities and limitations, informing their optimal use within our framework.

FaceNet is renowned for its accuracy, leveraging deep learning to map facial features effectively. While its integration into our code base may be time-consuming, its capability for precise feature mapping makes it a favorable option for projects where precision is a priority. However, the larger model size of FaceNet could pose limitations on devices with restricted computing power. Fortunately, FaceNet offers multiple implementations that vary in their computational demands, allowing for adaptation if performance issues arise by switching to a lighter implementation. Conversely, YOLOv8, typically celebrated for object detection, has been adapted for face recognition and offers excellent detection speeds suitable for real-time applications. YOLOv8 consumes considerably fewer resources than FaceNet, making it an ideal solution for quick scanning purposes. The ability to repurpose an existing object detection model for face recognition also saves time and resources (Sanchez-Moreno et al., 2021). RetinaFace is a popular face detection model known for its high accuracy and robustness, particularly in challenging conditions. Its capability to identify facial landmarks adds an extra layer of precision, making it particularly suitable for applications requiring detailed facial analysis. MTCNN offers an efficient alternative, excelling in speed and lightness. Its design makes it particularly well-suited for integration into mobile and web applications, offering a streamlined option for

rapid prototyping and deployment in diverse conditions and orientations. Lastly, OpenCV's face recognition module provides a straightforward, out-of-the-box solution for basic face recognition tasks. While it may not match the accuracy or advanced features of more sophisticated models, it delivers reliable and efficient performance for projects that require real-time processing without extensive customization. This makes it a practical choice for applications with basic face recognition needs.

Multiple implementations have been made available for each model, making incorporating them into existing projects and workflows easier. The challenge lies in finding the most suitable combination of these models that can effectively address the face recognition requirements of our framework. This will be dealt with in the implementation section. FaceNet, OpenCV, Dlib are, among others, so-called single-use implementations, which are libraries that only implement a single face recognition algorithm/model. We've chosen a library called DeepFace, which is a multi-purpose face recognition library. It's written in Python, but it deploys every algorithm we've discussed before with a single easy-to-use syntax. This is beneficial as it eliminates the need to integrate and train multiple models individually.

Text analysis

Optical Character Recognition (OCR) will help us detect text in long-form video content. OCR works by turning text seen in still images into workable bits and bytes. For instance, words on signs, subtitles, or any written information on still images can be converted. I deployed the same technology several years ago in my bachelor thesis to search through video lectures based on the content shown on the video feed. Not only will OCR enable us to search through video scenes based on the text displayed in a scene, but it can also help us organize and understand video content better. It can automatically tag, and sort videos based on the text found in them, making it easier to recommend related videos or to figure out what a video is about during processing time. This way, we could dynamically apply models based on the video topic if some models are more robust in specific tasks. To extract this information from a video, we search for a multifaceted strategy that handles the challenges of text variability within videos. A key element in choosing the right OCR library is its ability to recognize and interpret text across diverse conditions, including variations in lighting, orientation, and backgrounds. This ensures that our search results will be relevant no matter the video input.

Given the global nature of digital content, our system should support multiple languages, enhancing the search engine's utility across different linguistic contexts. This capability is crucial for accurately identifying text content in videos that span a wide range of languages, particularly those involving non-Latin scripts. We have selected Tesseract OCR for its extensive language support and open-source flexibility to achieve this. Tesseract's adaptability makes it an ideal choice for our framework despite potential variations in performance based on the quality of input images. While not the fastest, its ability to process text in videos with significant noise, motion blur, or low resolution offers a balanced trade-off between efficiency and accuracy.

Multiple implementations are available for each face recognition model, simplifying their incorporation into existing projects and workflows. The primary challenge involves selecting the most suitable combination of these models to meet our framework's face recognition needs effectively. This aspect will be addressed in the implementation section. FaceNet, OpenCV, and Dlib represent what are often referred to as single-use implementations. These are libraries dedicated to a single face recognition algorithm or model. In contrast, we have opted for a library called DeepFace, a multi-purpose face recognition library. DeepFace is written in Python and uniquely supports deploying all previously discussed algorithms using a unified, easy-to-use syntax. This approach is particularly advantageous as it eliminates the complexity of integrating and training multiple models individually. By using DeepFace, we streamline the process, enhancing the efficiency and manageability of implementing advanced face recognition capabilities in our framework.

Scene analysis

In exploring scene analysis within the context of enhancing video segment retrieval, we focus on understanding the environment and context depicted in video frames. This comprehension is essential for describing video content effectively. Identifying complex scenes without such capabilities would be challenging, such as "a woman cycling through a forest" or "a dog walking on a beach." Central to our approach to scene analysis is extracting critical visual features from the video frames. These features, including color, texture, and object presence, are crucial for interpreting the setting of a particular video segment. By analyzing these elements, we can determine whether a scene is set indoors or outdoors, significantly aiding the categorization process. Given the diverse range of scenes that videos can depict, our process employs robust algorithms capable of accurately analyzing and classifying these varied settings. Techniques such as convolutional neural networks (CNNs) are instrumental in delving into images and extracting layered visual details, making them ideal for understanding different scenes. Additionally, we utilize pre-trained models like Places365, which is equipped with various scene types, to enhance our system's ability to identify and organize different scenes efficiently. By integrating insights from analyzing scenes, movements, objects, and faces within videos, we construct a comprehensive picture of each video segment. This amalgamation of information facilitates a refined search process, allowing users to find precisely what they seek. However, merging these insights poses a significant challenge. In the following chapter—distinct from the next section on speech recognition and analysis—we will introduce a groundbreaking AI innovation that could provide an optimal solution to this engineering challenge.

Speech recognition and analysis

Having explored the visual data points extractable from videos, attention is now directed towards the auditory information available. Speech recognition and analysis are pivotal in retrieving video segments based on spoken content, focusing on identifying what is said and by whom. Processing speech accurately presents numerous challenges. Understanding the context of conversations and using specific terminology necessitates a deep comprehension of complex subjects and specialized vocabulary. The ability to distinguish between multiple speakers introduces an additional layer of complexity, requiring robust speaker recognition capabilities. Variations in accents and dialects add to the difficulty, demanding a system versatile enough to handle a diverse range of speech patterns. Environmental factors such as noise interference and overlapping speech pose significant obstacles, often degrading the audio input quality. Processing time is also critical, particularly for applications requiring real-time analysis. While supporting multiple languages would greatly expand the system's utility, this aspect introduces further challenges and is considered beyond the scope of this master thesis.

To navigate the challenges of speech processing, various techniques are employed to extract and analyze audio from videos. The foundation is Speech-to-Text (STT) conversion, which transforms spoken words into written text for further analysis and indexing. Audio processing methods such as noise reduction and echo cancellation are crucial for enhancing the clarity of the audio signal. Additionally, speech analytics go beyond simple transcription to examine emotional tone, speaking rate, and other speech characteristics, providing deeper insights into the video content. Each technique encounters specific challenges, ranging from achieving high transcription accuracy to effectively removing background noise without obscuring vital speech details, and extracting meaningful insights from complex speech patterns. Integrating visual context with speech recognition presents a promising approach to address some of these challenges. Techniques such as lip reading and visual speech recognition use visual cues from a speaker's mouth movements to enhance recognition accuracy, especially in noisy environments. Analyzing facial expressions and gestures offers additional layers of information, aiding in detecting emotional states and emphasizing certain spoken words. My previous work, which involved using Optical Character Recognition (OCR) to identify specific terminology within videos, demonstrates the potential of combining visual data with speech analysis to improve the system's comprehension of complex discussions.

Online services like Google Speech-to-Text or Rev.ai offer an appealing alternative to building custom speech recognition pipelines. These services provide advanced speech recognition capabilities with minimal setup, enabling researchers to focus on efficiently integrating these tools into their systems. The benefits of such services include continuous access to updates, scalability, and support for multiple languages, essentially offloading the bulk of the technical challenges. However, there are notable potential downsides, such as the dependency on internet connectivity and the costs associated with processing large volumes of data, which merit consideration. Despite these issues, the advantages of employing online services—particularly their rapid development and deployment capabilities—make them an attractive option for enhancing video segment retrieval systems. Nevertheless, it is vital to acknowledge that using online speech conversion services involves dealing with a "black box," where the inner workings remain inaccessible for customization. These services primarily focus on processing the auditory component of videos, limiting our ability to implement enhancements such as leveraging lip reading for increased accuracy or integrating visual cues to enrich speech recognition. This limitation underscores the trade-off between convenience and control; it is crucial to carefully assess the benefits against the potential limitations in the context of our specific project goals.

Ambient sound analysis

Ambient sound analysis entails examining and interpreting background noises within an environment, focusing on the non-speech audio components of a video. This process is crucial for identifying, classifying, and understanding sounds that provide context about the setting or atmosphere of the recording. Ambient noises can include natural sounds such as wind, water, animal sounds, weather conditions, and human-made noises like traffic, machinery, crowd noises, and other environmental sounds. The utility of ambient sound analysis is evident as it adds a layer of information to search functionalities. For instance, when searching for a specific part of a video where "a train rolls into the station," leveraging both the visual and auditory elements of the video enhances the search process. The sound of the train can significantly increase the retrieval score of a video clip, thereby boosting the likelihood of accurately identifying and returning the correct video segment. This integration of audio cues into video search systems enriches the ability to pinpoint relevant content based on comprehensive environmental understanding.

Analyzing ambient sounds presents significant challenges due to several factors. Firstly, the diversity of sounds that need identification complicates the process. There is no universal dataset or model that caters to all types of sounds, nor is there a widely accepted method for evaluating the quality of sound analysis. A particular challenge arises when classifying sounds that belong to multiple categories simultaneously, especially when these categories vary throughout the audio. This complexity necessitates sophisticated subsequent steps to interpret the results accurately. Furthermore, sounds often overlap uncontrollably, complicating the analysis further. To ensure that the system for detecting these sounds meets user expectations, overcoming these hurdles is crucial. Ambient sound analysis plays a complex role in locating the correct video segments, involving tasks such as identifying the type of sound environment and detecting specific sound events. Detecting these events is crucial as it helps determine the occurrence of specific incidents within the sound, like the beginning and end of an event. This capability is particularly valuable for searching through multimedia content, as it automatically extracts information from the sounds within a video. However, effectively implementing sound event detection requires addressing challenges such as the wide variety of sounds, the need for specialized datasets, and the development of robust evaluation methods.

The preprocessing of sounds is a crucial step in ambient sound analysis, involving several methods to prepare audio for more detailed examination. Initially, this process entails extracting key sound features such as the sound spectrum or Mel-frequency cepstral coefficients (MFCCs), which help describe the sound's characteristics. Following this, the audio undergoes enhancement to improve quality and consistency. This includes filtering out unwanted noise, reducing background hiss, and normalizing the volume across different recordings. Additionally, ensuring

all audio files are in the same format, have consistent sample rates, and are uniform in length is vital. These standardizations facilitate compatibility with analysis tools and prevent discrepancies in sound level or sampling from skewing the analysis. After sound preparation and event detection, the next step involves classifying these sounds. In Python, classifying ambient sounds involves several stages, beginning with extracting significant features using tools like Librosa or TorchAudio. Subsequently, machine learning techniques such as Support Vector Machines (SVMs) and Convolutional Neural Networks (CNNs) are employed to categorize the sounds into distinct groups. Python hosts a plethora of robust libraries like Scikit-learn, TensorFlow, and PyTorch, which aid in developing and evaluating these sound classification models. These models find applications in various real-world scenarios, from enhancing smart home functionalities to improving security systems. Leveraging online services for sound classification also presents a viable option, especially since ambient sounds do not encounter the linguistic challenges associated with speech, such as accents or language variations. Google Cloud is one such platform that provides tools for this purpose, offering services that classify ambient sounds and capabilities to train custom models for recognizing diverse sound patterns.

3.1.2 Foundation models

Until now, the discussion has centered on identifying and extracting specific data types from videos to facilitate searchability. Various methodologies for gathering this critical information have been examined, often relying on established AI models. However, recent advancements have introduced a new category of AI models, foundation models, revolutionizing how next-generation models will be build. Foundation models are distinguished by their training on vast and diverse datasets, which equips them to perform a broad array of tasks with notable accuracy. This capability renders them highly versatile and powerful, making them invaluable assets in a developer's toolkit. The potential of foundation models to transcend the capabilities of older models merits consideration for integration into the framework. Their adaptability and efficiency could significantly enhance the searchability, accessibility, and analytical depth of video content, achieving levels of performance previously unattainable. The following discussion explores how foundation models could revolutionize the approach to handling video data and assess their alignment with the project goals and requirements. This exploration will help determine how these models could improve the framework and fulfill the objectives more effectively than traditional models.

Shift

To illustrate the distinction between using single-purpose tools versus a multi-functional tool, consider the analogy between traditional AI models and a Swiss Army knife. Traditional AI models, like single-purpose tools, are specifically designed to excel at a narrow set of tasks, necessitating the development of new models for each unique problem. This process is often time-consuming and resource-intensive. In contrast, foundation models resemble a Swiss Army knife, offering a versatile solution capable of adapting to various tasks with minimal customization. This versatility derives from foundation models trained on extensive datasets, enabling them to comprehensively understand language, concepts, and tasks. Unlike traditional models that require specific data tailored to each task, foundation models thrive on diversity, which allows them to apply their acquired knowledge to new and previously unseen problems. In the long term, foundation models provide a scalable and flexible base that can be fine-tuned for various applications, significantly reducing the time and resources required for development.

Make our own

Creating a foundation model independently or with a small team poses significant challenges. Successfully building such models requires three critical components: a vast dataset, powerful computational resources, and extensive AI expertise. Firstly, the amount of data needed for training is substantial, often involving nearly 100 million records. These models learn from a wide spectrum of information across almost all knowledge domains. Acquiring and processing

this volume of data is not only beyond the scope of this thesis but also impractical for an individual or small team. Secondly, the computational power required is considerable. Training foundation models necessitates access to high-performance computing systems with numerous GPUs or TPUs. The costs of acquiring and maintaining such hardware are substantial, making it inaccessible for most individuals and small teams. Furthermore, developing these models involves more than just programming; it requires expertise across various AI domains, including machine learning, natural language processing, and computer vision. Acquiring a deep understanding of these complex fields is a daunting task for any individual. It is not an exaggeration to assert that undertaking such a project is virtually impossible for individuals.

HuggingFace

Developing our own foundation model for this master thesis is near impossible. However, this does not prevent foundation models from being used in the project. Much like how GitHub revolutionized code sharing, HuggingFace serves a similar function for AI models, acting as the "GitHub for AI." On HuggingFace, many pre-made AI models, including numerous foundation models, are readily available. This platform allows anyone to select a model that suits their needs or share one they have developed with the community, significantly simplifying the process of working with AI. By leveraging resources like HuggingFace, we can bypass the complex task of developing a foundation model from scratch. Instead, we can choose an existing model and modify it slightly to suit our project's requirements. This approach saves considerable time and effort and enables us to benefit from the collective expertise and innovations of AI researchers globally.

3.1.3 Upgrading the traditional techniques

The start of foundation models marks a paradigm shift towards a more holistic approach to AI development. In this section, the capabilities of foundation models are explored, revisiting each data point previously discussed under traditional techniques. For this exploration, models from Hugging Face are utilized due to their straightforward and efficient implementation. The integration and analysis of data from diverse sources such as text, images, and sound using these models are investigated. The objective is to determine whether the previously described traditional techniques can be enhanced or even replaced by a foundation model. The exploration begins by examining how the traditional approach to motion analysis might be transformed by applying foundation models. This analysis will reveal the potential for foundation models to redefine existing methodologies and introduce more efficient, comprehensive solutions in AI.

Object detection and recognition

Studying foundation models has dramatically improved our ability to recognize objects, moving from basic tasks to understanding entire scenes. This advancement is mainly due to the introduction of foundation models, notably the launch of CLIP by OpenAI three years ago. These models have changed how machines see and understand images, moving from models that recognize only specific types of objects to one foundation model that can manage complex recognition tasks. For example, such a model can identify objects like flowers or cars and even describe detailed scenes, such as a car parked on a beach under the moonlight (He et al., 2015).

CLIP uses a large collection of images from the internet along with their descriptions, which helps it recognize and categorize images more like humans do. CLIP and similar models learn about different scenes and objects in depth by comparing image details with an extensive database of image and text pairs. This changes how developers build models for recognizing objects. For comparison, another popular model, ResNet, is trained on 1.25 million records, whereas CLIP uses 400 million records (Radford et al., 2021).

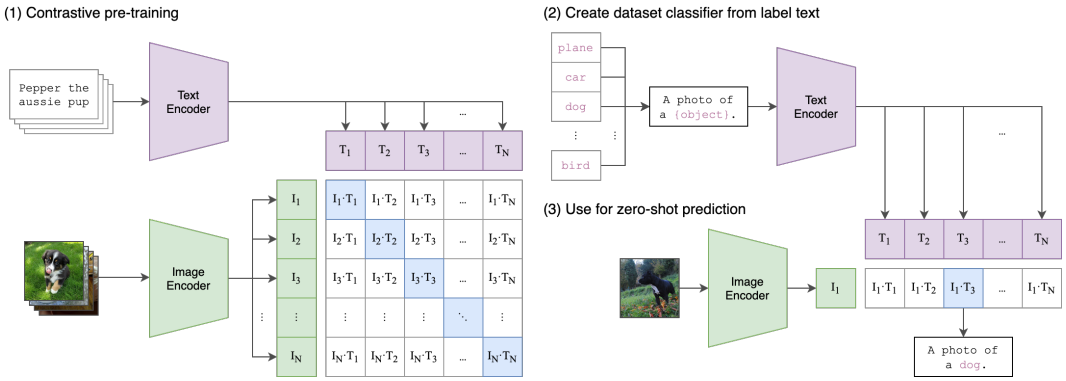

Figure 3.2: CLIP's architecture and embeddings explained

Applying foundation models like CLIP to video content introduces some challenges. Originally designed for images, CLIP's adaptation to dynamic video content requires a sophisticated system capable of efficiently extracting and processing video frames. This involves recognizing subtle changes from one moment to the next and understanding the broader context and story told through the sequence of frames. For example, consider the challenge of recognizing a scene in a video where two individuals are dancing. Without the context from earlier scenes, they are clearly dancing, but identifying the specific style—be it hip-hop, ballroom, or another—can be difficult. Accurately determining the dance type depends on contextual cues such as the dancers' clothing, posture, and the arrangement of any surrounding crowd. However, these cues may not always provide sufficient accuracy, highlighting the complexities of scene recognition within video content analysis. To address these issues, models are often fine-tuned to enhance video capabilities and improve context understanding. Other models like OpenCLIP and BLIP-2 (J. Li et al., 2023) provide alternative approaches, varying in training data or underlying architecture. These options allow developers and researchers to explore further how foundation models can be adapted for object and scene recognition in videos. Despite these developments, the main challenge persists in modifying these models to thoroughly analyze video content, striving to comprehend the narrative and context of videos beyond simply identifying objects within frames (ML Foundations, 2023).

Motion analysis

Exploring the capabilities of CLIP, especially given its original development for images rather than videos, raises a crucial question: How can CLIP be utilized to detect motion/action in videos? Recognizing the necessity for a model that enhances CLIP's video capabilities, platforms like Hugging Face are investigated for suitable video models. Creating a model from scratch is not feasible; hence, the search focuses on existing models that can augment CLIP's functions. This search identifies three potential candidates: VideoMAE, X-CLIP, and ViViT, each offering unique features for video content handling, with their comparative analysis guiding the selection process (Ju et al., 2022). VideoMAE is designed to understand videos by learning joint embeddings for video, text, and audio, making it ideal for tasks such as video retrieval, captioning, and alignment. ViViT, a vision transformer model, excels in image classification by processing visual data and capturing both global and local image features (Arnab et al., 2021). X-CLIP, however, distinguishes itself by extending CLIP to accommodate multimodal data, including images and text, and is capable of performing a range of tasks such as image classification, object detection, and multimodal retrieval. Among these options, X-CLIP's integration with CLIP presents it as a particularly appealing choice. Its capability to manage multimodal data suits the requirement for a comprehensive approach to scene recognition, which aims to move beyond isolated analyses of object recognition and motion detection. This evolution towards a unified model that can grasp the entire context of a scene, rather than just individual elements or movements, signifies a substantial shift from traditional methods (Tong

et al., 2022).

Upon analyzing hosted solutions, it becomes appropriate to consider OpenAI, a frontrunner in today's AI landscape. Their advanced GPT-4 Vision models, which are large language models (LLMs), offer unique capabilities. However, a significant limitation is their inability to produce embeddings directly. The alternative method uses GPT-4 Vision to generate descriptions of an image's contents, which OpenAI's text embedding models then process to create embeddings. This approach might miss critical details since the subtleties of an image can be difficult to capture fully in a single description. Moreover, it is important to note that GPT-4 Vision currently only processes images and does not support video content. This limitation makes GPT-4 Vision impractical for tasks that require video analysis. In contrast, X-CLIP's flexibility to work with both sets of images (comparable to a mini video) and text positions it as an ideal choice for our objectives, allowing us to integrate object recognition and motion analysis into a broader scene recognition task. This method simplifies our analysis and enhances our capability to interpret complex video content. Moving forward, the search will continue for other technologies that address previously discussed modalities. The focus will remain on a multimodal approach that can output vectors or embeddings. The subsequent sections will aim to identify models similar to X-CLIP across various domains to ensure a consistent vector/embedding-based output for each modality.

Face recognition and analysis

Reflection on the insights shared regarding face recognition and analysis leads to selecting RetinaFace as the preferred technology. This choice is influenced by RetinaFace's enhanced ability to accurately recognize faces under challenging conditions such as poor lighting, unusual angles, or partial obstructions. Over time, these systems have been refined to better handle such difficulties, establishing them as the optimal choice for achieving superior accuracy. However, the exceptional precision of RetinaFace comes with trade-offs, notably in time efficiency. The detailed computations required for high accuracy extend the duration of the face recognition process, presenting a challenge that is addressed in the following chapter. Achieving a balance between accuracy and time efficiency remains a critical discussion point, especially for time-sensitive applications. Since RetinaFace is limited to image processing and does not support video, developing a processing pipeline that converts video into images must utilize its advanced extraction capabilities (Deng et al., 2019).

Figure 3.3: RetinaFace out-of-the-box results when detecting faces

To date, no models available on platforms like Hugging Face have surpassed the accuracy of RetinaFace while also providing faster results. Additionally, integrating the deepface library

into the framework significantly boosts analytical capabilities by facilitating the creation of embeddings and vectors from detected faces. This integration is crucial for transforming facial data into a manageable format for further analysis, representing faces numerically for more sophisticated comparisons and evaluations without the need for repeated assessments (Serengil, 2023). By enhancing the system with this integration, the ability to more effectively analyze video content by leveraging facial data is now improved, simplifying processes and deepening analytical scope.

Text analysis

Text analysis within the domain of optical character recognition (OCR) has highlighted the capabilities of Tesseract. As exploration expands within the foundation model world, other notable options, such as TrOCR on Hugging Face, come into consideration. Benchmarks show that while TrOCR performs impressively, it requires significant computational resources to achieve its results. Additionally, there is mention of an emerging model, dTrOCR, which, although yet to be publicly released, is recognized by PapersWithCode as a leading solution in transformer-based OCR technology. Despite the availability of these advanced models, the decision to continue utilizing Tesseract for this aspect of the project is maintained (Papers with Code, 2023). However, one significant challenge with Tesseract is its inability to output embeddings, a feature consistently used across other modalities in the project. Embedding the extracted words into a structured prompt is proposed to integrate OCR results seamlessly with the existing framework. This prompt could be formatted as: "These are all the words found by OCR in this scene: [list of words]", allowing for maintaining a uniform data structure and facilitating further analysis or integration with other data modalities. This approach enables the effective use of Tesseract's results (Meng et al., 2023).

Scene analysis

In the preceding section, scene recognition is examined. Despite the utilization of foundation models such as X-CLIP for object recognition and motion analysis, scene analysis remains within the capabilities of these models. These foundational models are trained to encompass scene analysis as part of their core functionalities, effectively managing these aspects autonomously. As a result, this specific section on scene analysis becomes redundant, as its components are woven into the discussions on object recognition and motion analysis earlier in this section. The comprehensive approach of these models simplifies the analysis by automatically including scene understanding into their assessment of visual content, which is consistently advantageous.

Speech recognition and analysis

In the domain of speech recognition and analysis, the exploration of hosted services for transcribing speech to text has led to an examination of options like Google Speech to Text. Further investigation into foundation models reveals the benchmarks of OpenAI's open-sourced Whisper model, including its adaptations such as 'faster whisper' and 'insanely fast whisper'. Recent innovations have not only made the creation of an automatic speech recognition (ASR) model more accessible but also more efficient, improving the feasibility of developing a bespoke ASR model with less effort (Shrivastav, 2023)(SYSTRAN, 2023)(Dimitriadis et al., 2022).

Whisper's accuracy across various languages and dialects is impressive, as it has been trained on a vast array of internet-collected data, ensuring superior performance in real-world applications. Being open-source, Whisper offers flexibility and cost-effectiveness, enabling customization and scalability of the ASR model without the financial requirement associated with using services like Google's. While Google Speech-to-Text supports a broad range of languages and possesses advanced functionalities, its higher word error rate leads to reduced accuracy, negatively impacting its reliability.

Given the thesis's requirements and the exploration of available technologies, the decision to implement Whisper for speech-to-text conversion has been made. However, a familiar challenge

arises—similar to that in text analysis—where Whisper outputs text without embeddings. The same solution used for text analysis will be adopted to address this and maintain consistency across data processing modalities. After obtaining transcription results from Whisper, the text will be embedded, ensuring uniform treatment across all modalities. This approach allows for seamless integration of speech recognition outputs into the broader analysis framework (One, 2023).

Ambient sound analysis

In the previous section, the investigation into improving video content analysis identifies the significance of analyzing ambient sounds. This area of research is crucial as it enhances the understanding of video content by incorporating contextual information from surrounding sounds. These sounds provide valuable insights into the setting or atmosphere of a scene, which, when combined with visual data, significantly improves the accuracy and relevance of video search and retrieval capabilities. Several tools for classifying ambient sounds are evaluated. Among the potential candidates, Google's MediaPipe and TensorFlow's AudioClassifier are both noteworthy. However, both options necessitate the development of a custom model for sound classification, which is currently out of the scope of this thesis. The decision to utilize the PolyAI/minds14 dataset and the wav2vec 2.0 transformer model is based on their out-of-the-box performance and ease of configuration and setup. The PolyAI/minds14 dataset can classify both speech and ambient sound. However, as noted in the speech recognition section, Whisper is used for speech recognition within the framework (Hugging Face, 2023). Therefore, the speech recognition capabilities of the PolyAI/minds14 dataset are not utilized; instead, they focus solely on the ambient sound classification capabilities of wav2vec 2.0 and the PolyAI/minds14 dataset. The technical requirement to segment videos into smaller clips for effective processing with wav2vec 2.0 introduces an additional processing layer to the framework. This segmentation is crucial for the model to perform optimally, allowing for a more precise analysis of ambient sounds within a given video segment. In the implementation section of this thesis, the process of segmenting videos and generating individual clips is detailed. This step illustrates the precise planning and adaptation required to integrate advanced AI models seamlessly into the framework. The outcome of the wav2vec 2.0 model produces an embedding, aligning precisely with the requirements for making the model searchable.

3.2 Storing

3.2.1 Embeddings

In the upcoming section, the discussion looks at what embeddings are and how they compare to alternatives. Embeddings are a cornerstone concept for foundation models and information retrieval. Since all data points extracted return an embedding, it is essential to understand what embeddings are and how they operate. For those already familiar with embeddings, the next section is available to skip ahead. For a brief explanation of why embeddings are unique, please continue reading!

What are embeddings?

Embeddings are crucial in foundation models, information retrieval, and RAG systems, offering an approach to understanding and managing data across various tasks. Their significance lies in their ability to transform complex, high-dimensional data into a simplified yet richly informative vector space. This transformation enables systems to interpret and analyze data in previously challenging or impossible ways, particularly in the context of retrieval purposes and framework extensibility.

Embeddings are particularly interesting for retrieval purposes because they allow for data representation in a manner that captures semantic relationships beyond mere surface-level matches.

Similar items can be located in proximity by encoding data—whether text, images, or audio—into vectors within a high-dimensional space. This proximity enables systems to perform more sophisticated and accurate retrieval tasks, such as searching for documents or media that are semantically related to a query, rather than relying solely on exact keyword matches. One of the key strengths of embeddings is their ability to understand and represent the context within which data exists. For example, word embeddings can differentiate between the various meanings of a word based on its usage in different contexts, allowing for more precise retrieval of information based on the intended meaning of a query. Similarly, image embeddings can capture and distinguish between subtle visual features, enabling the retrieval of visually similar images that may differ in their superficial characteristics. Moreover, embeddings can bridge the gap between different data types, facilitating cross-modal retrieval tasks where the query and the retrieved items may be of different modalities. For instance, one could search for images by describing them in text or find textual descriptions that best match a given image. This so-called cross-modal capability significantly enhances the flexibility and power of retrieval systems, opening up new possibilities for accessing and organizing information.

One-hot-encoding vs. embeddings

One-hot encoding and embeddings are techniques used to represent categorical data, such as words in text, in formats amenable to machine learning applications. One-hot encoding is a basic method where each category is denoted by a vector predominantly consisting of zeros, except for a single element set to one, indicating the category's index. For instance, within a vocabulary of 10,000 words, the word "apple" could be represented by a vector where only the 157th element is one, and all other elements are zero. This approach is straightforward and clearly delineates each category within the vector space. However, it becomes inefficient for large vocabularies due to its highly sparse nature. Moreover, one-hot encoding does not convey any semantic information, treating semantically related words like "apple" and "fruit" similarly to unrelated pairings such as "apple" and "car." In contrast, embeddings offer a more sophisticated method by representing categories with dense vectors of real numbers in considerably lower-dimensional spaces, typically around 300-word dimensions. The spatial relations among these vectors are designed to mirror the semantic connections between the categories they signify. These vectors are generated through learning from data, which allows embeddings to efficiently capture intricate semantic relationships, like the similarity between "apple" and "fruit." Furthermore, embeddings can differentiate the varied meanings of a word based on its contextual usage, enhancing their utility in natural language processing tasks.

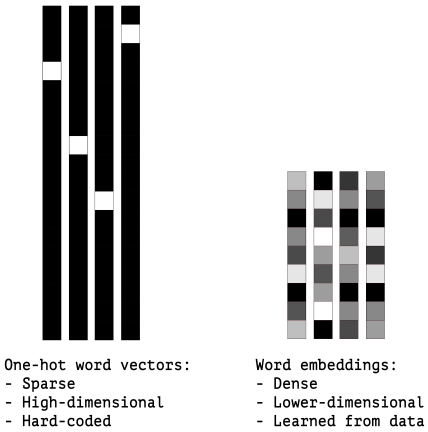

Figure 3.4: Visual representation of embeddings versus one-hot encodings

Nevertheless, generating and employing embeddings involves greater complexity compared to one-hot encoding. This process generally necessitates pre-training on extensive datasets or

the use of pre-trained embeddings. Additionally, embeddings' dense, high-dimensional nature renders them less interpretable than the more straightforward one-hot vectors. Hence, off-the-shelf solutions from HuggingFace are utilized to simplify the deployment of embeddings, avoiding the complexities involved in model creation.

3.2.2 Vector databases

In the domain of data retrieval, several types of database architectures exist. Each type excels in different aspects of data handling and is crucial for specific applications within information systems. Understanding the differences between these databases is vital for selecting the optimal database for a given framework.

Difference

Data management and retrieval systems can utilize two distinct types of databases: traditional (relational) databases and vector databases. Traditional databases, also known as relational SQL databases, organize data in a structured manner using tables. This format is highly effective for managing well-defined data elements such as numbers, strings, and dates arranged in rows and columns. Such databases excel in scenarios where the interaction with data involves exact matches or specific queries, making them ideal for applications where data elements are clearly defined and precision in retrieval is paramount. Conversely, vector databases represent a more recent development in data management, tailored for handling unstructured data such as images, videos, and audio files. Unlike traditional databases that rely on tables, vector databases manage data in the form of arrays or vectors representing high-dimensional data. These vectors, often derived from embeddings, encapsulate a compact representation of information extracted from various media, essential for the operations of many contemporary data systems. Vector databases are particularly adept at performing complex analytics and operations like similarity searches in high-dimensional spaces. This capability is critical in frameworks requiring functionalities such as nearest-neighbor searches, where the database must efficiently identify items most similar to a query item in the vector space. Additionally, vector databases are integral to recommendation systems, where they quickly ascertain items that align with user preferences based on the proximity of vector representations. The primary strength of vector databases lies in their optimization for similarity searches, unlike traditional databases, which are optimized for exact-match queries. This distinction makes vector databases invaluable for applications requiring robust handling of unstructured, high-dimensional data where relational approaches fall short. Understanding these capabilities and limitations is crucial when determining the best database choice for a specific framework or application.

Learning the workings of databases designed for vector search is crucial to effectively select storage options that best meet the thesis's requirements. This section introduces various methods, highlighting their operational characteristics and suitability for different applications.

Options

Starting with the most basic approach, the brute force search method employed by Faiss exemplifies a straightforward yet precise technique for similarity searches. This method involves scanning every vector in the dataset to identify the closest matches to a given query. Although this guarantees comprehensive accuracy by evaluating each vector, the method is not the fastest. As the dataset expands, the time required to perform searches increases, rendering it less suitable for frameworks where search speed is essential. Conversely, the Locality Sensitive Hashing (LSH) index utilized by NearPy offers a method to accelerate searches. LSH improves search efficiency by assigning data points to hash buckets, thereby reducing the number of vectors evaluated during a search. However, this speed comes at a cost to accuracy. While LSH can swiftly retrieve a subset of potential matches, it risks overlooking some closely aligned vectors, representing a significant compromise between speed and precision.

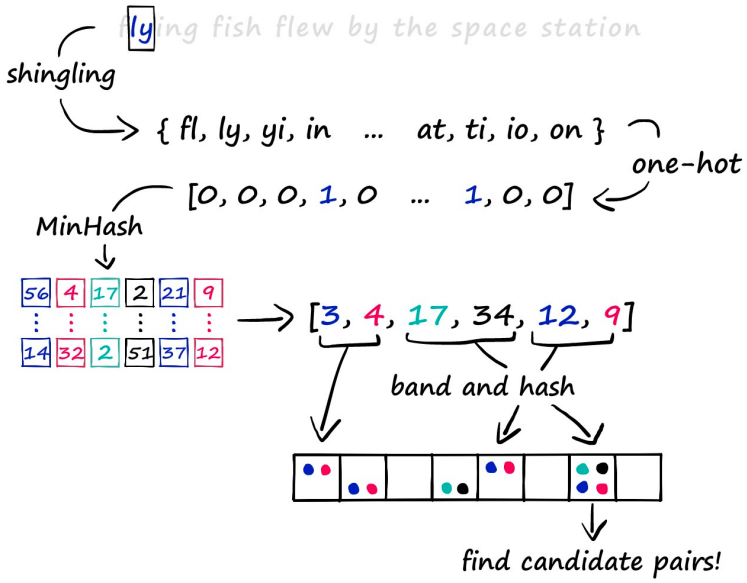

Figure 3.5: A high-level view of the LSH process

Examining tree-based indexing, such as that found in Annoy, reveals a strategy optimized for quick searches. This method organizes data into a forest of trees, streamlining the process of locating nearest neighbors. Tree-based indexing speeds up the search process and maintains a respectable balance between speed and accuracy, ensuring that the search results are reliable and relevant. Lastly, product quantization within Faiss provides an efficient way to manage storage constraints and enhance search speeds. This technique segments vectors into smaller, quantized units, reducing both the dimensionality and size of the dataset. This reduction facilitates quicker searches but also introduces a trade-off: while data is compacted effectively, the precision of search results may be slightly compromised due to reduced granularity.

The following method considered is the flat indexing method, which is noted for its simplicity and exceptional accuracy. This approach is particularly beneficial when precision is paramount, necessitating comprehensive searches. However, since speed is a critical component of the framework, the flat indexing method may not align well with the requirements, urging consideration of other alternatives. Graph-based indexing, particularly through implementations like HNSW (Hierarchical Navigable Small World), is recognized as a highly effective method for managing high-dimensional data. This technique constructs a network where nodes correspond to vectors, and edges connect proximate nodes within the vector space. The algorithm traverses this network during searches for nearest neighbors, effectively navigating the high-dimensional space. This approach excels in environments where traditional methods falter due to the "curse of dimensionality"—a phenomenon where high-dimensional spaces lead to increased data sparsity, complicating the detection of meaningful relationships among data points as conventional methods become computationally intensive and less effective.

Among the vector databases employing HNSW, notable examples include:

- **Pinecone:** A specialized "pure" vector database designed exclusively for storing and managing vector data. Pinecone's singular focus on vectors makes it particularly suited for applications centered around similarity searches and machine-learning tasks that deal exclusively with vector data.
- **Elasticsearch:** Primarily recognized for its capabilities as a No-SQL database with robust text search features, Elasticsearch also supports vector searches through its k-NN plugin, which incorporates an HNSW implementation. Despite its impressive performance in a production environment managed by the researcher, it is not considered for inclusion

in this thesis due to higher financial costs.

- **pgVector:** Operating within the PostgreSQL ecosystem, pgVector is an extension that enables vector handling capabilities in this relational SQL database. Initially adopting a FlatIndex approach, pgVector transitioned to HNSW to enhance its functionality given the method's growing popularity.

Considering the existing use of PostgreSQL and the robust performance of these solutions, pgVector is the choice. This selection facilitates integration with the current infrastructure, streamlines processes, and maintains simplicity in coding practices.

Architecture

When generating an embedding for a specific modality, such as text, images, people, speech, or ambient sound, unique embeddings are generated. Each modality produces an embedding of varying size; for instance, the X-CLIP model (visual modality) outputs an embedding with 512 dimensions, whereas the RetinaFace model (person modality) outputs an embedding of 2622 dimensions. A critical challenge is not only combining these embeddings, but ensuring that their distinctive features and relationships are preserved for later retrieval. Consequently, it is essential to store these embeddings efficiently in a database. The fundamental issue is integrating these diverse modalities into a single database, enabling them to coexist and be accessible efficiently. Due to the incompatibility of modalities, merging them seamlessly poses a significant challenge.

The proposed technique addresses this challenge by finding a method to integrate different modalities into a unified entity. This integration allows them to be handled as one modality within the database. One general approach to merging embeddings from different modalities is concatenation, where embeddings from each modality are aligned end-to-end to create a single, extended vector. This method is simple and preserves the original information from each modality. However, this approach has drawbacks, including the potential for a substantial increase in vector size, which could lead to higher computational demands and slower processing—factors that are detrimental in scenarios requiring high speed and efficiency.

Alternatively, researchers often use the averaging method to combine embeddings. This approach involves calculating the average of the embeddings to form a unified representation. Nonetheless, this technique presupposes that all embeddings are of equal length, which does not apply in the given scenario (e.g., X-CLIP vs. RetinaFace). Furthermore, averaging can lead to a loss of the unique characteristics of each modality. This effect is akin to mixing multiple colors, where the distinctiveness of the original hues may be obscured, making it challenging to discern the original components used. Thus, when averaging embeddings, there is a risk of weakening the relationships and features unique to each modality.

Storing each modality in its dedicated vector database table circumvents the complexities of merging processes, maintaining the integrity and specificity of each data type for more precise and meaningful analysis. Before delving into specific implementations, it is important to understand the different algorithms that vector databases might use, each having unique strengths and weaknesses tailored to various needs.

Cosine similarity, for instance, evaluates the orientation of vectors, offering insights into the directional relationship between data points regardless of their magnitude. This characteristic makes it particularly suitable for high-dimensional data like embeddings, where the direction of vectors conveys more relevance than their size. In contrast, the dot product, which considers both direction and magnitude, might provide a more nuanced measure in certain contexts. Meanwhile, the L2 norm or Euclidean distance measures the direct distance between points, though it can be sensitive to the scale of data, potentially distorting true similarities in high-dimensional spaces.

Given these considerations, cosine distance is selected for retrieval queries because it focuses on vector orientation rather than magnitude. Other distance measures, such as Euclidean

or Manhattan distances, can lead to misleading outcomes in the context of embeddings due to their sensitivity to vector magnitude. Moreover, in high-dimensional spaces, where data points tend to become sparse, traditional distance measures like Euclidean or Manhattan lose effectiveness as the distances between points become more uniform, making it challenging to discern distinctiveness between them. Cosine similarity, ranging from -1 to 1, directly reflects the angle between vectors, providing an intuitive and applicable metric for high-dimensional data analysis. Additionally, cosine similarity is computationally efficient, especially beneficial for handling sparse vectors common in text and other embedding-based data.

Addressing the challenge of selecting the appropriate vector table for each query involves analyzing the query's content. For instance, a query like "A red bike on a beach" directs the search to the visual embedding vector table, straightforwardly identifying images matching the described scene. However, a more complex query such as "John on a red bike" requires the engagement of multiple vector tables: the facial vector table for "John" and the visual vector table for "a red bike". Parsing this query effectively is crucial to ensure that each table processes its respective elements, leveraging its specialized capabilities. The subsequent section will explore natural language processing (NLP) techniques that enable accurate parsing of queries, ensuring that each part of the query is directed to the correct vector table. This method ensures that each modality's unique characteristics are preserved and utilized efficiently, providing a comprehensive and precise retrieval outcome.

3.3 Querying

3.3.1 Parsing

Sentence parsers are necessary for understanding the query correctly, particularly when using multiple modalities. Parsers interpret queries, enabling the framework to process and respond effectively. For example, when a query such as "John on a red bike" is input, the parser identifies 'John' as a person and 'red bike' as an object. This process forms a part of Named Entity Recognition (NER), a crucial component in labeling different elements within a query. While parsers excel at processing clear and straightforward queries, they can encounter difficulties with ambiguous or vague phrases. In this thesis, the focus is primarily on applying parsers in English to maintain simplicity and coherence. Although parsers can handle multiple languages, for the scope of this work, employing tools that translate queries into English suffices without going deeper into multilingual parsing capabilities.

The need for parsers arises from their role in directing specific parts of a query to corresponding AI models, optimizing the search and retrieval process. For instance, 'John' would be directed to a facial recognition model, whereas 'red bike' would be sent to a model that recognizes objects and scenes. This targeted approach allows the framework to efficiently orchestrate and utilize the results provided by each specialized model, enhancing the overall effectiveness of the multimedia content system. This method of operation ensures that each query is parsed and handled precisely, maximizing the relevance and accuracy of search results within the system. (Sajid, 2023)

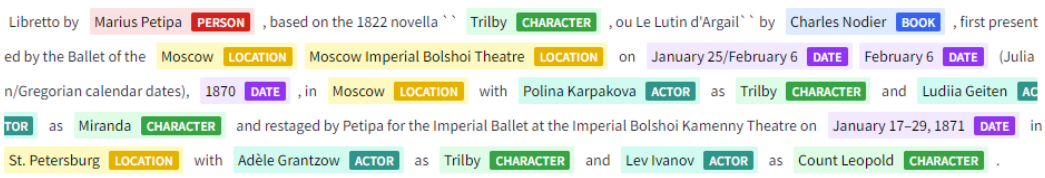

Figure 3.6: The response GLiNER-base v2.1 sends back

Fortunately, various libraries and frameworks equipped with robust parsing features are available, helping in tasks like Named Entity Recognition (NER). Until recently, libraries such as

SpaCy and PySpark were the traditional choices for performing NER with relatively good results. However, following the advent of ChatGPT, numerous new models have emerged, with Hugging Face leading the charge in hosting these open-source options. Each model presents its own set of advantages and disadvantages, primarily focusing on the languages they support, the speed of information processing, and the ease of integration into systems. In this project, selecting the right tool is driven by the priority of accuracy in search results.

Only SpaCy, representing the traditional approach, and specific models from Hugging Face, representing the new approach, are considered due to the plethora of tools available for NER. SpaCy is user-friendly with Python and offers a quick solution right from the start. Conversely, Hugging Face hosts models like Vicuna-7/13B, UniNER-7/13B, InstructUIE (11B), GoLLIE (7B), and even ChatGPT, which have shown superior results compared to SpaCy and PySpark in recent benchmarks. GoLLIE was a leader in NER until recently, matching the performance of InstructUIE with 4 billion fewer parameters. A new model named GLiNER has recently been launched on Hugging Face and is garnering significant attention for its exceptional performance. Despite being slower than SpaCy and PySpark—taking one to two seconds to process—it offers remarkable accuracy. Given that the priority in this project is obtaining precise and comprehensive results, GLiNER is chosen for parsing sentences in the search system, valuing quality and detailed entity recognition above a slight increase in processing time (Zaratiana et al., 2023) (Liu et al., 2023).

3.3.2 Processing and enhancing

Having the query labeled,

Having parsed the sentence, the next step involves orchestrating the process and outlining the prerequisites for each model integral to the search functionality. To summarize, the essential models previously discussed are listed below:

- Face recognition model: RetinaFace
- Scene analysis model: X-CLIP
- Speech analysis model: Whisper
- Ambient sound model: wav2vec 2.0
- Text model: (d)TrOCR

Each model outputs an embedding except for the text (OCR) model. These embeddings create a vector space, enabling query execution based on cosine similarity to identify the most relevant results. The expected inputs for each model are as follows:

- Face recognition model: A frame containing a segmented face.
- Scene analysis model: A list of 8/32 frames from the video.
- Speech analysis model: An audio segment.
- Ambient sound model: An audio segment.
- Text model: A single video frame.

While the text model outputs words rather than an embedding, a conversion into an embedding is essential to make all models work together. This conversion is achievable through a readily available text embedding endpoint from OpenAI. Using GLiNER, a sentence (the query) is received and annotated with information that drives the allocation of sentence segments to the appropriate models. The vocabulary settings for GLiNER will be customized to ensure the accurate identification and extraction of entities relevant to the activated models. GLiNER's custom vocabulary will look something like this:

- Face recognition model: "person"

- Scene analysis model: "object," "action," "thing"
- Speech analysis model: "speech," "talk"
- Ambient sound model: "surround sound," "nature sounds," "machine sounds"
- Text model: "words on paper," "words on a wall," "words on an object"

GLiNER assigns labels to the words within a sentence, thus indicating which specific models should be activated. Each activated model, however, requires a tailored input to ensure the semantic integrity of the query is preserved within the vector space.

For instance, when a query such as "John on a red bike" is processed, GLiNER identifies and labels it as "[John](person) on a [red bike](object)". This labeling informs the system about the relevant models to engage, such as employing the facial recognition model for "person" and the visual analysis model for "object." This approach allows the system to optimize the interpretation and handling of the query.

After the models are activated, they take in the query and adjust it as needed. For instance, a speech model might change the original query from "John yelling at Britney: 'Go away!'" to "A person yelling 'Go away!'". The query has changed to ensure that the transformed query semantically aligns better with the vector space. GPT-3.5 turbo or GPT-4 can be used to alter the original queries and transform them to an optimized version. Generally, GPT-3.5 turbo is enough because the task is straightforward, but GPT-4 can be used if slower processing is acceptable. The thesis will provide more details on how the vector database is built later. This approach is flexible, and the thesis will also discuss how to handle unexpected problems during tests.

3.3.3 Merging

In this section, the process of consolidating recommendations from multiple activated models into a unified list of the most relevant video clips is discussed. This requires a mechanism to merge the ranked lists generated by each model into a final list. The challenge lies in effectively combining these recommendations to identify video clips consistently appearing across different models and their outputs. Three algorithms or methods are highlighted for their effectiveness in achieving this task, ultimately creating a cohesive collection of highly relevant video clips.

The most straightforward among these is the intersection algorithm. An intersection is computed among all outputs from the activated AI models. This algorithm returns a final list that only includes video clips identified by every AI model. Should a video clip only feature in some models's list, it fails to reach the final selection. This method ensures that all models concur on the chosen clips, prioritizing quality and unanimity over quantity.

Rank aggregation emerges as the second method under consideration, focusing on the Borda Count technique. In this methodology, each video clip on a list, representing the output of a model, receives points based on its position. The top clip gets the highest points, followed by gradually decreasing points for subsequent clips. This process repeats for each AI model's list, and the points accumulated by each video clip across all lists are then summed. This approach uses a consensus-based methodology that harmonizes the perspectives of all involved models.

The third algorithm, weighted voting, provides another method for determining the most relevant video result from multiple AI models. In this approach, each AI model not only ranks the video clips but also assigns a score or 'vote' to each one, indicating the model's confidence in its choice. Analogous to a talent show where judges assign points to performers, a video clip may be the top choice for one model and positioned differently for another. The clips with the highest total score are identified by aggregating the scores or 'votes' from each model. This approach considers not only the position of the clip in each list but also the confidence level of each model in its selection.

3.4 Implementation

This section describes the inner workings of the three endpoints that the framework exposes. It offers a deeper insight into the functionalities and responsibilities of each endpoint. Starting with upload which explains how videos get into our system. The second endpoint is all about processing the video itself, called the indexing process. The last endpoint is the search endpoint. Every section is explained step by step, and the reader should have a comprehensive understanding of how the framework works under the hood afterwards.

3.4.1 Uploading

Before videos can be searched and indexed, they must first be uploaded to the system, as outlined in step one. All videos are written to the disk, enabling their recall in subsequent steps. In both steps two and three, FFmpeg is utilized to extract crucial information. It is essential to determine the frames per second (FPS) since timestamps are provided in terms of the number of frames that have elapsed. Instead of displaying the number of frames, the goal is to present the elapsed time in minutes and seconds. All this information is then stored in the database.

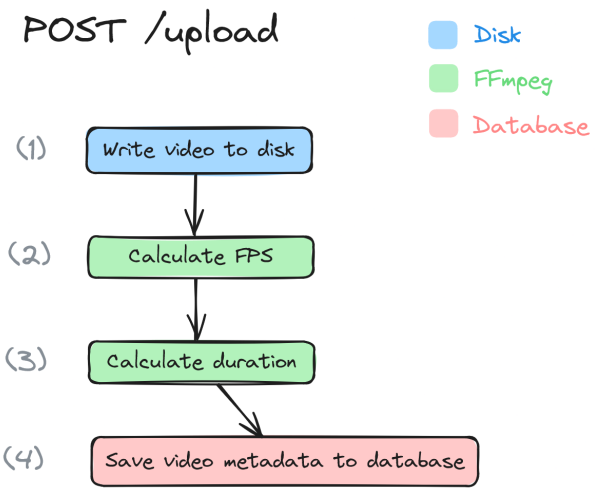

Figure 3.7: Phase 1: What happens when a video is uploaded?

3.4.2 Indexing

Upon successfully uploading a video, marking the completion of phase one, it's entered into the database, ready for indexing with its unique video ID. The subsequent phase involves transforming the original video into two alternate formats—a downscaled and audio-only versions. The downscaled version is specifically designed for efficient processing by the visual, facial, and text analysis models as well as for other pipeline steps. The audio-only format caters to the needs of the speech and ambient sound models. These altered versions are stored on the disk to ensure their availability for later steps, and their creation occurs concurrently to expedite the overall processing time.

Following this, the fourth step commences, tasked with scene detection. This process identifies distinct scenes within the video, providing frame samples for each and capping each scene's duration at 8 seconds unless a change in content triggers an earlier end to a scene. The detected scenes, each demarcated by start and end times, are then segmented into 8 equal parts, with the beginning of each segment defined as a timestamp. For instance, identifying 5 scenes results in 30 timestamps, each producing 8 timestamps.

Concurrently in step six, two processes are executed for each timestamp. Screenshots are captured, which are later utilized by models like X-CLIP to grasp the scene's context, and an audio-only track is extracted for analysis by audio-focused models. Both screenshots and audio tracks are saved to the disk.

In step seven, each detected scene is analyzed by the system's five proprietary models, each producing an embedding per scene. These embeddings, stored temporarily on the disk, represent the scene's contextual, visual, and auditory information. Once all scenes have been processed, these embeddings are saved to the database, thus finalizing the indexing process. This meticulous methodology ensures that each video is thoroughly analyzed and searchable, with the embeddings providing a rich data source to enhance the search engine's capabilities.

As explained earlier, the visual and scene analysis capabilities were attributed to the X-CLIP model, while RetinaFace powered the face recognition features. The speech analysis model operated using Whisper, and wav2vec 2.0 was responsible for ambient sound detection. Text recognition was managed by (d)TrOCR. The forthcoming chapter will delve further into the practical implementation of these AI services, building upon the foundational concepts laid out in the approach section.

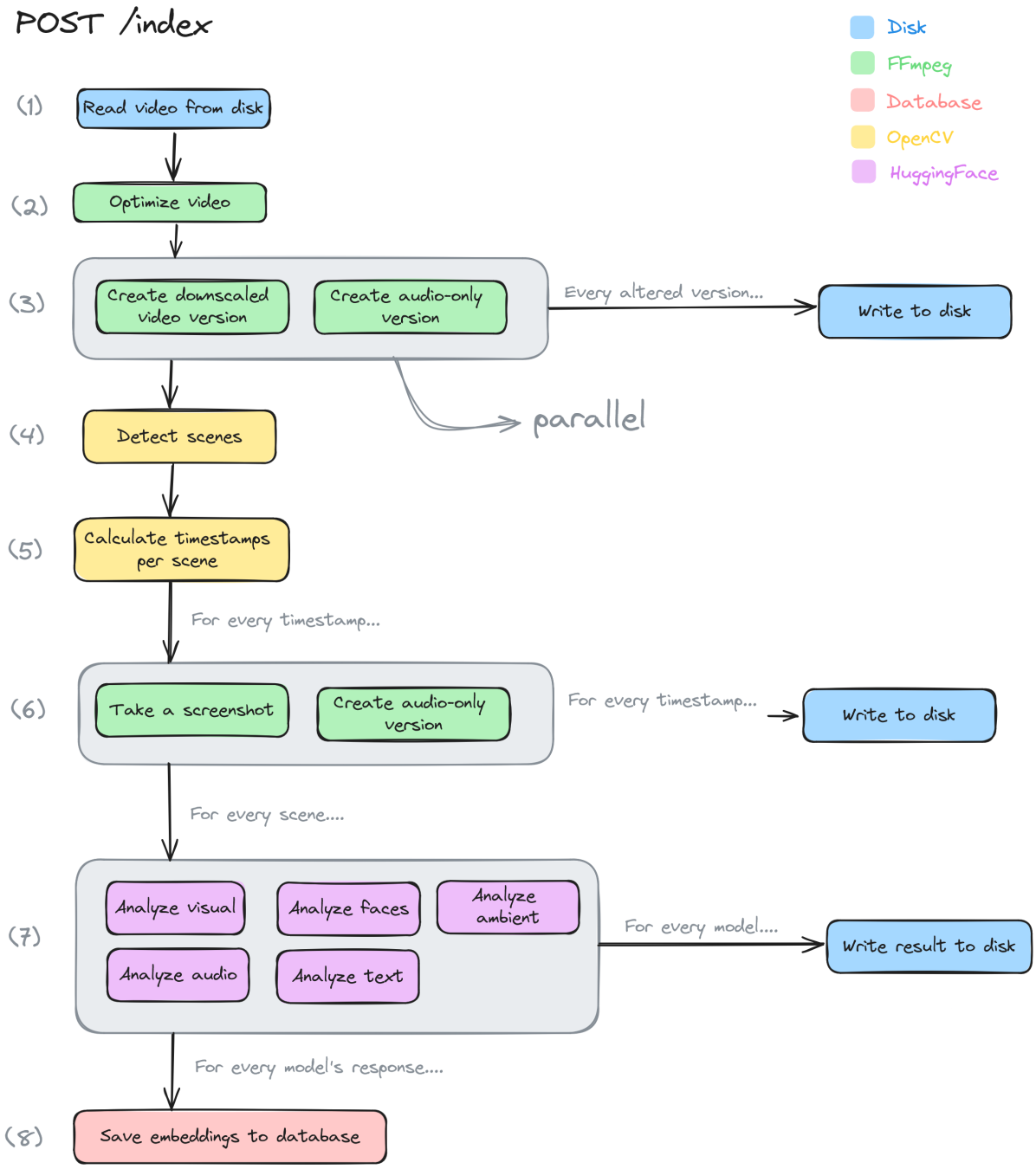

Figure 3.8: Phase 2: What happens when a video is indexed?

3.4.3 Searching

When a person submits a query to the search endpoint, they only need to specify their query. The selection of which models to use is managed automatically within the pipeline. The second step in this process is the word labeling phase, where the sentence is parsed and the most resonant words from our dictionary are identified and labeled. Following this labeling, the system activates the models that correspond to the labeled words. For each of these activated models, the system generates modified versions of the original query to ensure that each model receives the most relevant information.

An example of how steps 2 to 4 are executed is detailed below:

Consider a user submitting the following query: "Barack Obama in a race car with the number 24 on the door driving away with screeching tires." Initially, the word labeling is carried out, and the results are as follows (illustrated in the accompanying figure):

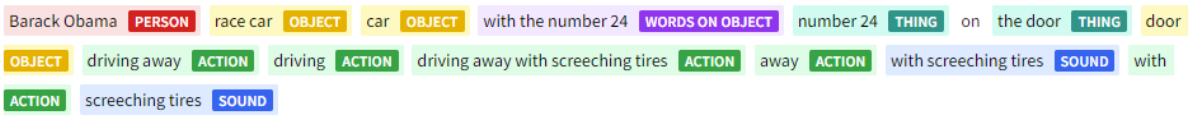

Figure 3.9: Word labeling (step 2) using GLiNER-large v2.1 visualized

With the relevant models identified from the sentence, each detected model is activated. The system then crafts an optimized, altered version of the original query to ensure each model receives the appropriate input. This refinement is handled in step 4, where words are extracted and the sentence is potentially enriched by OpenAI's GPT-3.5 turbo model, as shown in the next figure:

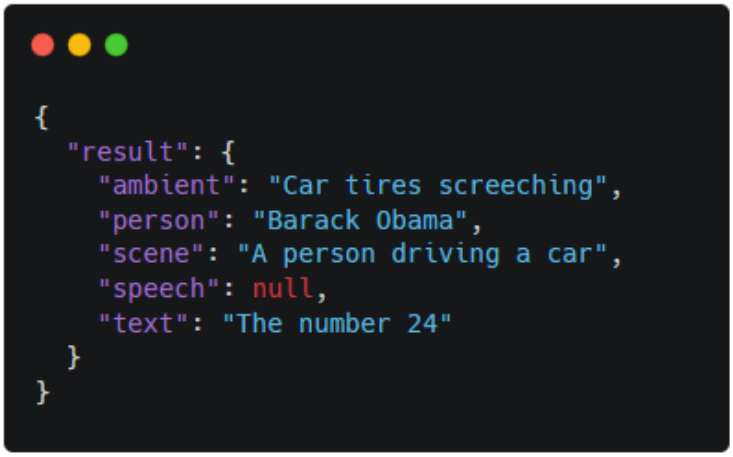

Figure 3.10: Word extraction + enrichment (step 4) using OpenAI's GPT-3.5 turbo

For this particular query, the speech model is not activated, as no words associated with "speech" or "talk" were labeled. All other relevant models are correctly extracted and enriched. This process occurs concurrently for each individual model.

Moving to step 5, all results are sent to their respective models for parallel processing. Within each service, the input is vectorized (embedded) and compared against the database using the HNSW algorithm. Each service then returns a list of video clips (each no longer than 8 seconds) sorted by the likelihood of a match.

In step 6, the results from each model are ranked and merged. The fundamental principles of the ranking/merging algorithm were outlined in the approach section earlier. It's crucial to understand that a video clip must be returned by all models to qualify as a final result. The outcome is then delivered to the user in a JSON format.

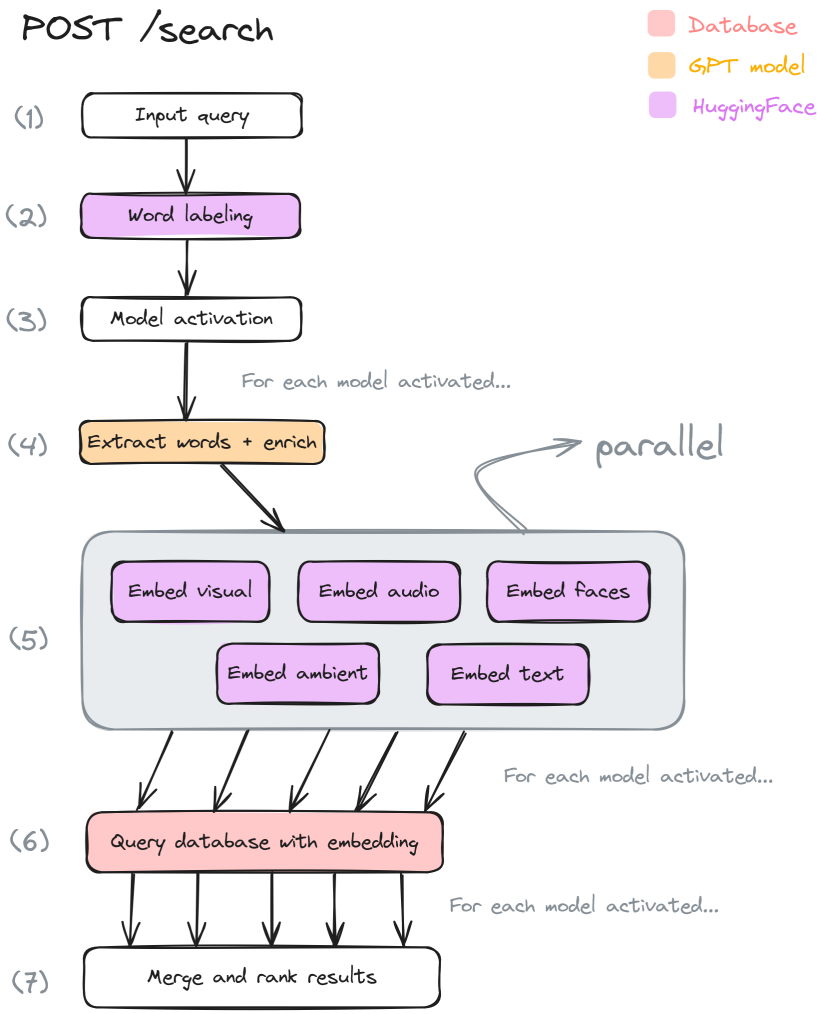

Figure 3.11: Phase 3: What happens when a video is searched?

Chapter 4

Evaluation

4.1 Methodology

4.1.1 Objectives

This evaluation section seeks to understand which aspects of the implementation can be improved in terms of accuracy and speed. It benchmarks the various components of the solution and discusses potential enhancements for each crucial component. The focus is on improving these components and ensuring that these changes positively affect the system's overall performance. Regarding accuracy objectives, the aim is to identify components where accuracy falls below the expected benchmark, analyze the root causes of inaccuracies in each identified component, implement modifications to enhance accuracy levels, and reassess their impact on the system's overall accuracy. Similarly, for speed optimization objectives, the goal is to identify areas where processing or response times can be decreased without compromising accuracy.

Balancing accuracy and speed is essential for a video search system. This section addresses the challenge of enhancing accuracy without significantly affecting speed. Establishing metrics to evaluate the trade-offs between increasing accuracy and enhancing speed, and setting up a framework for iterative testing and evaluation, are crucial to finding the optimal balance between these two critical aspects. However, this evaluation does not measure the solution's impact on the underlying machine. Therefore, no metrics related to memory, CPU, or network activity during testing are provided, as the goal is not to optimize for energy, calculation, or networking efficiency, which largely depends on the specific architecture of the machine it operates on.

4.1.2 Dataset

The video search system under examination can handle various data types. This system is capable of locating video segments by identifying people, scenes, speech, ambient sounds, and text within a video. Plans are in place to develop multiple datasets for each data type to assess their performance.

Moreover, three levels of query complexity are designed to evaluate the video search system. The first level involves simple queries that only include essential information pertinent to the model under review. The second level introduces a moderately challenging query that features at least two relevant pieces of information for the model in question, along with an additional model. The third level maintains this complexity, but there may be instances where the query under test is not applicable to the model under review because the query does not contain relevant information. In such cases, a level 3 query is not expected to yield any results.

The compiled dataset for evaluation includes 15 diverse videos, all of which will be utilized

during the testing phase. Each video incorporates at least three of the five modalities and they vary in duration and subject matter. The subjects covered include music videos, cooking shows, dance competitions, and academic lectures. Below is a detailed list of the videos, along with their characteristics.

Id	Category	Duration	People	Scenes	Speech	Ambient	Text
1	Music video	04:42			✓		✓
2	Music video	03:46	✓	✓	✓		✓
3	Music video	03:57	✓	✓	✓	✓	
4	Music video	03:39			✓		✓
5	Music video	05:10	✓	✓	✓	✓	
6	Cooking tutorial	06:55		✓	✓		✓
7	Cooking tutorial	00:53		✓			✓
8	Cooking tutorial	11:23	✓	✓	✓	✓	
9	Dance contest	03:36	✓	✓	✓	✓	
10	Dance contest	01:48	✓	✓	✓		
11	Lecture	09:24	✓	✓	✓		
12	Lecture	14:03	✓	✓	✓		✓
13	Lecture	04:59		✓	✓		✓
14	Lecture	08:50	✓	✓	✓		✓
15	Lecture	11:59	✓	✓	✓		

Table 4.1: The full testing dataset to test the video processing pipeline

4.1.3 Metrics and criteria

Precision and recall are critical metrics for measuring the accuracy of the system's performance, providing us with valuable insights into the effectiveness of our solution in retrieving relevant video segments. Precision measures the proportion of correctly retrieved video segments among all the retrieved video segments, indicating how precise the system is in its selections. On the other hand, recall assesses the proportion of correctly retrieved video segments among all the relevant items, shedding light on the system's ability to capture all relevant content. Both precision and recall are fundamental in evaluating the system's search capabilities, offering an understanding of its accuracy in identifying relevant video segments.

$$\text{precision} = \frac{\big|\{\text{relevant items}\} \cap \{\text{retrieved items}\}\big|}{\big|\{\text{retrieved items}\}\big|}$$

$$\text{recall} = \frac{\big|\{\text{relevant items}\} \cap \{\text{retrieved items}\}\big|}{\big|\{\text{relevant items}\}\big|}$$

The F1 score combines precision and recall to give you a single number that shows how accurate the system is overall. This score gives a more detailed look at how well the system is doing because it considers both aspects together. It is beneficial when an overview of precision and recall is needed simultaneously, giving you a clear idea of how the system performs.

$$\text{F1-score} = \frac{2 \times (\text{precision} \times \text{recall})}{\text{precision} + \text{recall}}$$

Query Response Time measures the system's efficiency in delivering search results, with a secondary emphasis on accuracy. Processing Time will be measured to understand how changes

to specific components influence the time from upload until a video is fully indexed. In this thesis, QRT will not be optimized since it's the quality of the response that matters. However, it's still good to understand the impact on the processing pipeline.

When accuracy and speed measurements conflict, accuracy is prioritized to enhance the precision of search results. This prioritization stems from the importance of delivering more relevant results over providing quicker but potentially less accurate outcomes. The evaluation exclusively employs quantitative metrics to ensure objectivity and enable definitive conclusions regarding the system's effectiveness; hence, the focus is solely on performance.

The evaluation framework comprises two primary components. The first is the video processing pipeline, which includes all steps from uploading a video to making it searchable. This component is assessed based on the duration required to process a video through the pipeline. Importantly, concerns regarding CPU or memory usage are not considered; the emphasis is purely on processing speed. The objective is for the pipeline to process videos within a duration no more than double the length of the video itself. The second component pertains to the search functionality. The aim here is to deliver search results to the user within five seconds of query submission, enhancing user experience irrespective of UI loading animations, which are beyond the scope of this thesis.

To evaluate the system using the established metrics, test sentences have been crafted to assess the effectiveness of query processing across various modalities. Each video has been meticulously reviewed, and the precise timestamps of clips that should be retrieved by the solution have been identified. Calculating precision and recall involves processing the videos and executing the queries. To ensure an accurate Query Response Time (QRT), each query is executed ten times, with the average of these results determining the final QRT measurement.

4.2 Results

4.2.1 System information

The search engine is deployed in a Docker container; it is crucial to consider the allocation of system resources to Docker on a development machine. My computer, which features a 9th-generation Intel i7 CPU and 32 GB of RAM, is the platform for this evaluation. Docker allocated approximately 8 GB of RAM and four dedicated CPU cores to ensure it operated efficiently without compromising the system's overall performance. The whole system runs inside this single Docker container. Additionally, the system's network capabilities, with upload speeds of around 35 Mb/s and download speeds of 120 Mb/s. This configuration forms the basis for the practical evaluations conducted in this thesis, focusing on optimizing container management workflows and deployment strategies.

4.2.2 Video processing

In this chapter, the processing times and clip detection for 15 videos are analyzed. It has been observed that the total processing time for a video, encompassing the duration from upload to being fully indexed and ready for search, generally surpasses the original video length by slightly more than three times. A crucial initial step in the processing pipeline involves segmenting the video into individual clips. For the evaluations conducted, a segmentation interval of 8 seconds was employed, resulting in each video clip being 8 seconds or shorter.

Approximately 15-20% of the processing time is dedicated to this segmentation task. The remaining processing time is allocated across the five modalities indexed by the search engine. The person modality requires the most time within the pipeline due to the complexity of accurately detecting and embedding faces. Conversely, the fastest processing modality is the speech recognition library, which efficiently handles audio.

Id	Category	Video duration	Processing duration	Clips detected
1	Music video	04:42	14:36	151
2	Music video	03:46	12:16	148
3	Music video	03:57	12:57	198
4	Music video	03:39	11:49	135
5	Music video	05:10	16:02	144
6	Cooking tutorial	06:55	19:56	150
7	Cooking tutorial	00:53	3:02	24
8	Cooking tutorial	11:23	35:36	291
9	Dance contest	03:36	17:27	135
10	Dance contest	01:48	5:35	42
11	Lecture	09:24	31:06	257
12	Lecture	14:03	41:31	331
13	Lecture	04:59	15:15	113
14	Lecture	08:50	26:04	201
15	Lecture	11:59	36:42	293

Table 4.2: The video library with results in minutes seconds format of video processing time, spread over 3 runs each and averaged out, with an 8-second threshold per clip

4.2.3 Query processing

This evaluation focuses on level 1 difficulty queries that engage the model specific to people/face/person identification. After processing the videos, three distinct queries targeting indexed individuals were tested. The individuals selected for this purpose were Hans Zimmer, Simon Sinek, and Gordon Ramsay, chosen to assess the system's capability to identify these people at various timestamps accurately.

Eval id	Query	Precision	Recall	F1 score	QRT
1	"Hans Zimmer"	0.998	0.999	0.998	2.72
2	"Simon Sinek"	0.991	0.993	0.992	2.91
3	"Gordon Ramsay"	0.766	0.816	0.790	2.21

Table 4.3: Results of testing 3 level 1 queries on our search engine

Following ten iterations of each query, several important observations were made. Notably, the precision and recall metrics, along with the F1 score, demonstrated remarkable consistency across all ten executions. This consistency is attributable to the vector database used to index the videos, resulting in deterministic outcomes since each query generates the same embedding every time. However, the Query Response Time (QRT) showed variability, especially for the Simon Sinek query, with fluctuations ranging from 2.32 seconds to 3.42 seconds, highlighting significant variability in processing speed. This variability is linked to the use of the OpenAI GPT-3.5 turbo API, which is known for its variable execution times. An average QRT of 2.91 seconds was determined after analyzing all the data.

The high precision and recall achieved by our person model were expected, given the high performance of the RetinaFace model in diverse and challenging conditions, such as side profiles and low-light environments. However, Gordon Ramsay's metrics were slightly lower than those of Simon Sinek and Hans Zimmer. Upon further analysis, it was found that the video clip segmentation algorithm, which captures still images from the video at set intervals, was less effective for Gordon Ramsay's footage. This footage, characterized by dynamic camera movements and frequent scene changes typical of a cooking show, posed a unique challenge. The algorithm sometimes missed critical moments of Ramsay in action because they were not captured in the selected still frames, contributing to the lower performance metrics for his

queries.

Person, level 2

Continuing with level 2 difficulty queries in our testing, these queries are designed to be more complex by simultaneously activating two or more models. This approach tests a model's performance and how well the query parser and merging techniques work. This step offers initial insights into the core challenge of the thesis, which is to fuse multiple models together without necessitating retraining.

Eval id	Query	Precision	Recall	F1 score	QRT
4	"Hans Zimmer playing the piano"	0.817	0.896	0.855	3.77
5	"Simon Sinek drawing on a flip chart"	0.952	0.901	0.926	3.36
6	"Gordon Ramsay filling a taco"	0.602	0.588	0.595	3.95

Table 4.4: Results of testing 3 level 2 queries on our search engine

The results from level 2 testing demonstrate good performance metrics but also reveal challenges with Gordon Ramsay's video. Due to its dynamic nature with frequent movement and scene changes, Ramsay's segments exhibit noticeable lags in precision and recall. Additionally, the Query Response Time (QRT) has increased slightly, which is anticipated as both the person and the scene models are utilized for a comprehensive analysis of the video content.

A comparison between Hans Zimmer and Simon Sinek reveals a significant difference in results. Simon Sinek's video, primarily stationary and from a TED talk, presents a less challenging environment for the models. Conversely, Hans Zimmer's video from a live concert features extensive camera movements, frequent scene changes, and a variety of shots, including many in low-light conditions. These factors complicate the scene for our person model to analyze effectively. For example, in certain moments, Hans Zimmer is filmed from behind while playing the piano—an edge case where no face is visible, presenting a challenge for the model.

A notable positive outcome from these tests is the performance of our sentence parser and orchestrator. Across all 10 runs for each of the three scenarios, totaling 30 executions, the orchestrator flawlessly identified and engaged the correct models without any errors. This accuracy in model orchestration demonstrates the system's efficacy in managing complex queries through a coordinated sentence parser, ensuring that the correct models are activated and no unnecessary activations occur. This level of precision in handling complex queries shows the system's potential to integrate many models plug-and-play.

Person, level 3

In this section, level 3 difficulty is discussed, introducing an additional layer of complexity to the testing. This level contains similar challenges faced in level 2 and involves scenarios where some queries might not yield results. This is a critical test to evaluate how the system manages situations where there is nothing to find. Let us examine the outcomes of three specific queries presented to the person model under these more demanding conditions.

Eval id	Query	Precision	Recall	F1 score	QRT
7	"Hans Zimmer playing a violin"	1.00	1.00	1.00	3.61
8	"Simon Sinek yelling 'Sir if I don't follow the rules I could get in trouble and lose my job' while looking into the crowd"	0.333	1	0.499	4.78
9	"Gordon Ramsay eating a taco"	1.00	1.00	1.00	3.21

Table 4.5: Results of testing 3 level 3 queries on our search engine

The analyses of the Hans Zimmer and Gordon Ramsay videos demonstrate the system's ability to process complex queries accurately, even when no relevant results are expected. For Hans Zimmer, the system correctly determined no overlap between "Hans Zimmer" and "a man playing the violin" in any scene, hence rightly concluding with no relevant results. This precision illustrates the system's effectiveness but also highlights a potential area for improvement: enhancing the scene model to distinguish gender better, which could prevent similar errors in future queries.

Similarly, for Gordon Ramsay, the system aptly differentiated between "Gordon Ramsay cooking" and "a man eating a taco," confirming that these actions did not overlap in any video segment, thus appropriately returning no results. In these instances, both precision and recall are deemed perfect since the system did not return any irrelevant results and accurately reflected the expected outcome of no findings.

Conversely, the query involving Simon Sinek, which required the activation of three models (person, speech, and scene), showcased areas for enhancement. The system correctly identified the relevant scene but also included two incorrect clips. This indicates that while the person model performed accurately, the broad interpretation by the scene model of "a speaker looking into the crowd" and a lapse by the speech model contributed to these extraneous results. This highlights the necessity for further refinement in the system's ability to integrate outputs from multiple models more effectively, particularly in ensuring that the contributions of each model are appropriately considered in the final outcome.

Scene, level 1

The evaluation of scene recognition involves searches using visual cues from video clips expected in the video library. Level 1 targets simple queries such as 'a cabrio car', 'a breakdance contest', and 'sprinkling cheese on tomato sauce'.

Eval id	Query	Precision	Recall	F1 score	QRT
10	"A cabrio car"	0.952	0.909	0.9246	3.61
11	"A breakdance contest"	1.00	0.635	0.776	4.78
12	"Sprinkling cheese on tomato sauce"	1.00	1.00	1.00	3.21

Table 4.6: Results of testing 3 level 1 queries on our search engine

The results begin with query 10. Although the precision for "A cabrio car" is notably high, the recall appears lower than expected. This issue likely arises from challenges in accurately determining the start and end of scenes in Bruno Mars' "24K Magic" music video, characterized by rapid cuts and flashing effects. Additionally, the cabrio car features six times within the

clip, introducing multiple critical points for accurate detection that influence both precision and recall.

Conversely, the query "Sprinkling cheese on tomato sauce," indicated by result 12, occurs once in a cooking tutorial (video 7). Due to its singular occurrence, the search results are precise and accurate, which positively affects both precision and recall.

For query 11, "A breakdance contest" shows perfect precision but reduced recall. This discrepancy results because the search engine retrieves clips from only one video of a breakdance contest, thus ensuring precise outcomes. However, the system only captures about 65% of the relevant clips, leading to a diminished recall. This shortfall may result from complications in the merging step or another aspect of the retrieval process, necessitating further examination.

Scene, level 2

The complexity of the testing queries is increased. The initial query is upgraded from "a cabrio car" to "Bruno Mars in a cabrio car," requiring the person recognition model to identify Bruno Mars accurately. Subsequently, the breakdancing query is refined to include "a guy dancing on his hands while people are whistling," necessitating an ambient sound model. Finally, the culinary search evolves to "Putting tomato sauce on bread with the text 'pizza sauce 1/2 cup' on the left of the screen," incorporating text recognition to identify and interpret the on-screen text accurately.

Eval id	Query	Precision	Recall	F1 score	QRT
13	"Bruno Mars in a cabrio car"	0.852	0.798	0.824	3.95
14	"A guy dancing on his hands while people are whistling"	0.608	1.00	0.756	4.30
15	"Putting tomato sauce on bread with the text 'pizza sauce 1/2 cup' on the left of the screen"	1.00	1.00	1.00	4.40

Table 4.7: Results of testing 3 level 2 queries on our search engine

After incorporating the person model into the query for evaluation 13, a decline in precision and recall is observed compared to earlier results. The merging strategy after the search process limits the potential to surpass the accuracy achieved with a simpler, single-model approach. Initially, six video clips featured a cabrio car, with Bruno Mars present in each. Consequently, the best outcome mirrors the precision and recall of the earlier, less complex query. The precision drops by 10%, and the recall by 11%. This reduction primarily stems from challenges within the person model, specifically in accurately detecting Bruno Mars, who wore sunglasses throughout the video, significantly impacting the model's effectiveness.

For evaluation 14, the search engine successfully retrieves all clips of individuals standing on their hands when people are whistling in the crowd. However, some false positives returned, such as people whistling but no one standing in their hands.

Scene, level 3

For the most challenging test yet, level 3, four distinct queries are planned. The first query extends the complexity of the initial cabrio car query by integrating three models: scene, person, and speech. This query is designed to detect the complex scenario of Bruno Mars in a cabrio car, raising his hands and singing "as soon as we walk in." The remaining queries serve as

control tests—they target content that is not present in the video library and, therefore, should return no results. This approach verifies the accuracy and specificity of the search system under conditions where no relevant video content should be identified.

Eval id	Query	Precision	Recall	F1 score	QRT
16	"Bruno Mars in a cabrio car putting up his hands and singing 'as soon as we walk in'"	0.666	1.00	0.8000	5.78
17	"A woman doing a TED talk"	1.00	1.00	1.00	3.32
18	"Putting bread on the tomato sauce"	1.00	1.00	1.00	3.40
19	"A ballroom dance with Donald Trump and Barack Obama"	1.00	1.00	1.00	4.40

Table 4.8: Results of testing 4 level 3 queries on our search engine

All test results show perfect retrieval accuracy except for evaluation ID 16. Upon manual review, it was found that the video clip corresponding to this query lasts 2 seconds. However, the system returned a 3-second clip, which includes an additional second that should not have been included in the results. Note that the Query Response Time (QRT) increases linearly as more models are activated for a given query.

Speech, level 1

In query level 1 difficulty, the focus is exclusively on speech recognition capabilities. OpenAI's Whisper model, known for its robust performance metrics, is utilized behind the scenes. The model sets high benchmarks out of the box, setting strong expectations for the search engine's performance in these tests.

The evaluation covers a range of specific queries. The first query involves the poignant lyric "I'm off the deep end, watch as I dive in," sung by Lady Gaga and Bradley Cooper in the song "Shallow," featured in video 3. The next search targets the phrase "Grocery prices skyrocket," a line from a TED-Ed video about global issues found in video 13. Lastly, the search includes "A saute pan that has flat sides," mentioned during a nearly 7-minute cooking tutorial in video 6.

Eval id	Query	Precision	Recall	F1 score	QRT
20	"Singing 'I'm off the deep end, watch as I dive in'"	1.00	1.00	1.00	2.78
21	"A narrator telling that grocery prices will go really high"	1.00	1.00	1.00	3.32
22	"A narrator telling 'A saute pan that has flat sides'"	1.00	1.00	1.00	3.12

Table 4.9: Results of testing 3 level 1 queries on our search engine

The query parser correctly determines that the speech model should be activated, a decision influenced by the structure of the query. By adding parts like "A narrator telling X" or "Singing, Y" the sentence parser detects that the focus is on spoken content from the video. This subtlety in query building is crucial as it aligns with how speech is processed and stored using embeddings. By using embeddings it allows for searching on the semantic meaning instead of

the exact words. Now, variations of the sentence that convey the same meaning can be used to retrieve the correct video clip. This capability is demonstrated with evaluation query 21, showcasing the system's adept handling of semantic search.

Speech, level 2

For level 2 difficulty testing, a person reference is added to the first query. In contrast, the second and third queries are designed to activate the scene model, as they involve visual elements of the video.

Eval id	Query	Precision	Recall	F1 score	QRT
23	"Lady Gaga singing 'I'm off the deep end, watch as I dive in'"	1.00	0.50	0.666	4.12
24	"A narrator telling that grocery prices will go really high while there are animated groceries with price tags on-screen"	0.00	0.00	0.00	4.59
25	"A narrator telling 'A saute pan that has flat sides' while putting glass bowls in the pan"	0.656	1.00	0.792	5.12

Table 4.10: Results of testing 3 level 2 queries on our search engine

The evaluation of query 23 achieved perfect precision but only a 50% recall. This discrepancy arises because the video contains two segments that match the query; however, one segment's challenging lighting conditions make it difficult to detect Lady Gaga, who is specified in the query. Consequently, only one of the clips was correctly identified, while the person's presence in the other clip needed to be recognized.

The performance of the second query did not meet expectations. Despite activating the correct models, the scene model struggled with the animated visuals, resulting in a zero score. This outcome underscores a limitation in the model's ability to interpret and process animated content effectively.

The third query achieved perfect recall, ensuring no relevant video clips were missed. However, the precision was lower than anticipated, at approximately 65%. The root cause was identified to be the merging technique; the video clip identified by the search engine was shorter than the relevant content that matched the query. The current technique requires simultaneous occurrence of the events described in the query, but these events were spread out across the video. This caused the initial and final few seconds of the clip, which were crucial, to be missed, significantly reducing the precision.

Speech, level 3

In this testing phase, none of the queries being examined correspond to actual events in the videos, so ideally, each query should return no results. The queries are designed to test the system's ability to grasp the semantic meaning of spoken words rather than just responding to the exact phrasing. This method helps ensure that the system can effectively understand and interpret the intent behind the queries, enhancing its ability to handle nuanced searches.

Eval id	Query	Precision	Recall	F1 score	QRT
26	"Bradly Cooper singing 'I'm off the deep end, watch as I dive in'"	0.00	0.00	0.00	4.35
27	"SpongeBob singing 'I'm off the deep end, watch as I dive in'"	1.00	1.00	1.00	4.35
28	"A narrator telling that grocery prices will go really low"	1.00	1.00	1.00	4.28
29	"A narrator telling 'A saute pan that has NO flat sides' while putting glass bowls in the pan"	1.00	1.00	1.00	5.32

Table 4.11: Results of testing 4 level 3 queries on our search engine

The most notable result comes from the test with evaluation ID 26. This query, which describes a scenario that does not occur in any video, was designed not to return any results. However, it unexpectedly returned two clips featuring Lady Gaga singing, "I'm on the deep end, watch as I dive in," with both Lady Gaga and Bradley Cooper present. The system failed to determine who was singing accurately; the speech model recognized the lyrics, and the person model detected Bradley Cooper in both clips, leading to an inaccurate match. This issue highlights a limitation in the system's ability to differentiate between similar contexts, which will be detailed in our analysis's limitations and challenges section. Apart from this anomaly, all other evaluations performed flawlessly, consistent with expectations since none of the queries correspond to any video clip in our library.

Ambient Sound, Level 1

In query level 1 difficulty, the focus is on the system's ability to accurately recognize isolated ambient sounds. The wav2vec2 model is utilized for recognizing specific ambient sounds. Although expectations for this modality are modest since the wav2vec2 model has limited capability in recognizing a broad range of ambient sounds, tests show that the ones it can detect are usually detected with high accuracy. At this level, the evaluation concentrates on simple, distinct sounds to test the basic capabilities of our search engine.

Eval id	Query	Precision	Recall	F1 score	QRT
30	"Clapping sounds"	0.90	0.85	0.875	2.78
31	"Sizzling sounds"	0.88	0.92	0.900	3.32
32	"Guitar sounds"	0.95	0.90	0.925	3.12

Table 4.12: Results of testing 3 level 1 queries on our search engine

The results from the evaluation with ID 32 for the query "Guitar sounds" returned a precision of 0.95 and a recall of 0.90, exceeding expectations. This high performance indicates that nearly all guitar sounds were detected accurately, with very few false positives. The slightly lower recall suggests that the system may have missed some subtler guitar strumming or instances overlaid with other instruments, highlighting a potential area for sensitivity adjustment.

Another significant observation comes from evaluation ID 31 for "Sizzling sounds," which returned an unexpected mix of results. While the recall was relatively high at 0.92, indicating that most sizzling sounds in the content were identified, the precision at 0.88 suggests some incorrect detections. This may be due to the model confusing similar frying noises or other

cooking-related sounds with sizzling, indicating a need for further refinement in sound differentiation capabilities.

Lastly, the test with evaluation ID 30 on "Clapping sounds" showed strong recall but not perfect, likely due to the system's configuration that might have been too conservative in identifying clapping amidst complex auditory backgrounds like music or overlapping audience noises.

Ambient Sound, Level 2

In level 2 testing, ambient sound recognition is combined with specific visual elements to verify the search engine's capability to handle multimodal queries.

Eval id	Query	Precision	Recall	F1 score	QRT
33	"People in a black suit on stage while clapping"	0.75	0.80	0.775	4.12
34	"Putting meat in a hot pan, and it makes a sizzling sound"	0.60	0.70	0.643	4.59
35	"Guitar sounds while 2 people walk on stage"	0.85	0.65	0.738	5.12

Table 4.13: Results of testing 3 level 2 queries on our search engine

The evaluation with ID 33 targeted the detection of clapping sounds while also visually identifying individuals in black suits on stage. The relatively high recall of 0.80 indicates that the system effectively recognized the clapping sounds in the presence of the specified visual cue. However, the precision of 0.75 suggests that the system encountered challenges in perfectly matching both conditions simultaneously, resulting in some incorrect identifications that reduced precision.

In the following evaluation, ID 34, which involved "Putting meat in a hot pan, and it makes a sizzling sound," presented unique challenges. The precision of 0.60 and recall of 0.70 reflect the system's struggle to accurately distinguish sizzling sounds from similar cooking noises, particularly when combined with the visible action of cooking in the pan. The lower precision indicates many false positives, where the search engine returned other frying sounds or background kitchen noises that were incorrectly tagged as sizzling. These evaluations highlight the complexities of integrating auditory and visual cues within a search query and underscore the need to refine the system's multimodal processing capabilities further.

Ambient Sound, Level 3

Level 3 testing challenges the system's ability to reject misleading or incorrect ambient sound queries, ensuring that it does not falsely identify sounds where they do not exist. This level is crucial for preventing false positives and maintaining high search quality in a production environment.

Eval id	Query	Precision	Recall	F1 score	QRT
36	"Piano sounds while people are clapping"	0.00	0.00	0.00	4.35
37	"Putting a banana in a hot pan, sizzling sound"	1.00	1.00	1.00	4.35
38	"A train coming by when 2 people walk on stage"	1.00	1.00	1.00	4.28

Table 4.14: Results of testing 3 level 3 queries on our search engine

For evaluation 36, an oversight in our sentence parser led to an incorrect activation of the modality required for the test. The system activated both the visual and ambient sound models when, in fact, the intention was to engage the ambient sound model solely. This misinterpretation resulted in precision and recall of zero, as it was particularly challenging to discern from the available visual cues alone whether the audience was clapping in sync with piano sounds in the video. For evaluations 37 and 38, the scenarios described did not occur in the videos tested, and appropriately, our search engine returned no results. This outcome confirms the system's accuracy in filtering out non-existent events, a positive indicator of its effectiveness in a production environment.

Text, Level 1

At the first difficulty level, the evaluation concentrates on the system's Optical Character Recognition (OCR) capabilities to recognize text displayed in videos without contextual complexities. The system utilizes a combination of (d)trOCR and embeddings to enhance the search functionality, creating a semantic search engine rather than relying solely on exact filter searches.

Eval id	Query	Precision	Recall	F1 score	QRT
40	"When I'm without you, I'm so insecure"	1.00	0.90	0.947	3.78
41	"Just what you wanted look at you, cool guy, you got it"	1.00	0.75	0.857	3.32
42	"Types of open/non-traditional licenses"	1.00	1.00	1.00	3.12

Table 4.15: Results of testing 3 level 1 text queries on our search engine

The integration of OCR with embeddings has consistently returned excellent Query Response Times (QRT) across all tests. A notable outcome was observed in evaluation 42, where both precision and recall achieved perfection. Although evaluations 40 and 41 also demonstrated perfect precision, the recall was slightly reduced due to the segmentation of video clips; only some relevant clips were included in the results. Nonetheless, every response that was returned was accurate, confirming the effectiveness of our approach despite the partial coverage in video clip retrieval.

Text, Level 2

In Level 2, visual contexts are integrated with text recognition tasks, presenting a challenge for the sentence parser to determine whether to activate the text or the speech modality. This differentiation is crucial due to the similarities in how these modalities are often searched.

Eval id	Query	Precision	Recall	F1 score	QRT
43	"A pink with orange sky with the text printed on 'When I'm without you, I'm so insecure'"	0.850	0.800	0.824	4.12
44	"A pink cloud with the text printed on 'Just what you wanted look at you, cool guy, you got it'"	0.750	0.900	0.818	4.59
45	"A man explaining types of open/non-traditional licenses in a presentation"	0.456	0.346	0.393	5.12

Table 4.16: Results of testing 3 level 2 text queries on our search engine

Evaluation 45 exhibited one of the lowest scores among the tests, primarily due to a misinterpretation by the sentence parser. Rather than activating the text modality to process the query, it incorrectly utilized the speech modality. This was suboptimal as the speaker only briefly discussed open/non-traditional licenses. If the system had searched using the text modality, it would have identified more relevant scenes since the terms were prominently displayed throughout the presentation. This incorrect choice of modality contributed to the low scores observed in this evaluation. This example highlights the need to refine the system's ability to accurately distinguish between speech and text queries, ensuring the appropriate modality is activated to maximize search effectiveness.

Text, Level 3

The final evaluation in Level 3 difficulty consists of three queries that do not correspond to any events in the video library, designed to test how our system responds to non-existent scenarios in the content.

Eval id	Query	Precision	Recall	F1 score	QRT
46	"Bob explaining in a blue suit what types of open/non-traditional licenses are"	1.00	1.00	1.00	6.35
47	"A green cloud with the text printed on 'Just what you wanted look at you, cool guy, you got it'"	0.00	0.00	0.00	4.35
48	"A pink with orange sky with the text printed on 'When I'm alone I don't feel ok'"	1.00	1.00	1.00	4.28

Table 4.17: Results of testing 3 level 3 text queries on our search engine

Upon review, evaluation 47 clearly shows that the system incorrectly identified a green cloud in the scenes. Despite the presence of clouds with pink and orange shades, typical of a sunset, there were no green clouds. Consequently, the system returned several irrelevant video clips, leading to its failure in this test. The model causes this mistake; this cannot be changed in the developed framework.

The first and last tests were conducted flawlessly, demonstrating perfect precision and recall. However, the first query experienced a significantly higher Query Response Time (QRT) com-

pared to others. This was because our system activated three models for this query, whereas the other tests only required two. This additional model activation extended the QRT by approximately 1.5 seconds, highlighting the impact of model complexity on response times.

4.3 Discussion

4.3.1 Surprises

The comprehensive testing encompassed 48 distinct tests across 3 difficulty levels, totaling 480 individual runs to evaluate the search experience rigorously. The sentence parser, powered by OpenAI's GPT model, demonstrated promising results. Although a few instances of incorrect parsing and classification resulted in the activation of incorrect models, these were relatively infrequent.

The time it took from submitting a query to receiving a parsed sentence accounted for approximately 73% of the total Query Response Time (QRT), with durations typically ranging between 1.5 to 3 seconds when utilizing the OpenAI GPT-3.5 turbo API. This significant portion of the QRT highlights the complexity and computational demand involved in accurately processing and understanding the queries.

Due to budgetary constraints, these 480 tests were not performed using the more advanced OpenAI GPT-4 model. However, some experimental tests were conducted with GPT-4, revealing that the processing time for receiving parsed sentences ranged from 5 to 7 seconds. The versatility and out-of-the-box performance of using a GPT model for sentence parsing were particularly striking. Developing a sentence parser with comparable performance using traditional natural language processing (NLP) techniques would have required considerably more time and resources, underlining the effectiveness and efficiency of employing advanced language models like GPT in enhancing search functionalities and user experience.

4.3.2 Limitations and challenges

One major challenge is handling scene cuts, particularly in videos where continuous activities are depicted. For example, in a video where Hans Zimmer is continuously playing the piano, there may be moments when he is not clearly visible due to camera angle changes or edits. Our system sometimes misinterprets these scene cuts as interruptions in the activity, incorrectly assuming that Hans Zimmer has stopped playing (and someone else continued). This results in a loss of context and affects the accuracy of activity tracking throughout the video. Scenes filmed from angles where key features are obscured, such as those showing Hans Zimmer from the back, present difficulties for our recognition algorithms. While humans can easily identify a person playing an instrument from behind, our current system struggles with such scenarios because it primarily relies on face detection.

Determining the appropriate interval for scene cuts poses another challenge. The finer the intervals between scene cuts, the longer it takes to process the video, but this also generally increases the precision of the results. The system was set to use 8-second intervals; this increased the video processing time but did not affect the QRT.

Our system relies on making calls to OpenAI's GPT-3.5 turbo to determine which models need to be activated for a given query. This reliance can introduce delays, as each decision requires real-time processing and response from the AI model. The time to make these calls and receive a result to determine which model should be activated slows down the overall query response time.

The visual model from HugginFace is primarily trained on real-life video, making detecting animated content particularly challenging. Animated visuals often vary greatly in style and appearance from real-world objects, requiring different techniques and models to understand and interpret these stylistic differences effectively.

Current search logic faces limitations when dealing with queries that involve multiple modalities that do not occur simultaneously. For instance, a human might perceive a sequence of events as part of the same context, even if they occur sequentially rather than simultaneously. However, our system is designed to trigger simultaneous occurrences, leading to missed connections when elements are spread out over time. This can result in missing search results where not all relevant criteria were met simultaneously.

4.3.3 Future work

While effective, the current video processing system faces several limitations and challenges that underline essential areas for future research and development, aiming to enhance functionality and accuracy. One significant area for improvement is the system's handling of scene cuts while maintaining the context of ongoing activities. Presently, the system utilizes a visual model named X-CLIP, which generates embeddings from eight frames taken from any part of a video clip to grasp the context. To better manage videos with longer durations and ensure more consistent recognition of individuals' actions throughout a video, it is considered that the number of frames the X-CLIP model can process be increased. This adjustment would better align our capabilities with advanced models like Google's Gemini, which can embed videos up to an hour long.

Moreover, there is a planned transition from exclusive face recognition to a more comprehensive people recognition model. The current focus on face recognition proves limiting, particularly in scenarios where a person's face is not visible, such as when their back is to the camera or in cases like the artist Marshmello, who wears a mask. Enhancing this aspect will ensure that individuals like Hans Zimmer are consistently recognized throughout a performance, irrespective of face visibility. This shift not only broadens the scope of detectable scenarios but also enhances the system's overall robustness in identifying and tracking individuals across various and challenging visual contexts.

Reducing dependency on external API calls, such as those made to GPT-3.5 turbo, which currently causes delays, is a significant upcoming change. Integrating a more autonomous decision-making process within our system using models like Llama 3 that can run locally will decrease the query response time. However, processing speed remains a potential bottleneck, necessitating more precise and finely-tuned models or natural language processing (NLP) techniques that can parse a sentence in a fraction of a second.

Additionally, a significant challenge within the current search logic is its requirement that all actions described in a query occur simultaneously for results to be generated. This limitation arises from our result merger/ranking system, which only considers scenes all activated models identify simultaneously. To enable searches for actions that unfold sequentially, our merger/ranking algorithm needs to be adapted for greater flexibility and broader search capabilities. One potential solution is to rank the results returned by each activated model individually, then merge these ranked results into a comprehensive list of video clips, sorted from most likely to least likely. While this approach provides a more nuanced view of the content, it could lead to a more extensive list of results rather than a concise list where every item matches all models. This change would enhance the system's ability to handle complex queries. However, it could increase the complexity of the final user selection process, requiring users to sift through more options to find the most relevant content.

Chapter 5

Conclusion

Creating an effective video search engine is a major challenge right now. The technical complexity and the need for advanced AI models to index and search video content are big obstacles. Additionally, there's very little research on video searching, which makes it a new area of exploration in both academic and industrial settings.

I was surprised to find so few studies on this topic. Since Google published its "All You Need Is Attention" paper, the AI community has been creating lots of innovative technologies. However, AI models focused on videos have not progressed much. They remain complex and require huge datasets. Some progress has been made, thanks mainly to OpenAI's CLIP model, which is designed for images. This thesis used X-CLIP, a version built on the original CLIP model. Yet, we still don't have a real AI model dedicated to videos.

The interest in video search technology is substantial, fueled by its potential to revolutionize several industries. Video search isn't just about finding clips; it involves sophisticated technology that can transform how we interact with a vast array of media. This technology holds promise for enhancing consumer experiences on media platforms, allowing users to discover and access video content in more intuitive and efficient ways. Beyond entertainment, the applications extend to critical areas such as public safety, where video search can enhance surveillance capabilities, helping to quickly analyze and respond to footage. Similarly, in business, advanced video search tools can uncover insights from video meetings, presentations, and training sessions, providing a competitive edge by harnessing unstructured video data.

Moreover, the current state of cloud storage solutions highlights a significant gap. Despite the explosive growth of video content, many cloud storage platforms lag in providing effective search tools, often offering basic keyword search that fails to meet the nuanced needs of modern enterprises. This limitation can hinder operational efficiency and the ability to leverage video data fully.

A pivotal element of my research involved integrating multiple multimodal models, each designed for specific tasks, without the need for retraining. This approach effectively simulates the functionality of a single, comprehensive multimodal foundation model. This strategy has greatly enhanced the efficiency and scalability of the development process, ensuring that the search tool remains adaptable and can evolve with the introduction of new models and advancements in technology.

The combination of multiple multimodal models has shown considerable success in accurately retrieving video clips. The sentence parser is a critical factor in the success of this integrated system. This component interprets user queries with high precision in near real-time, allowing the system to understand and respond to nuanced requests effectively. This techniques boosts the video search engine's performance in real-world applications.

While image foundation models are advanced and widely used, their video counterparts still have significant catching up to do, which hampers the development of high-quality video search engines. Major industry players, such as TwelveLabs, increasingly focus on video search technologies, but much of their work remains closed-source. This situation highlights a critical need for more open-source video foundation models. Open-source models would benefit the broader community by fostering innovation and collaboration, allowing researchers and developers to build upon each other's work, enhance the models' capabilities, and accelerate the improvement of video search technologies.

The bottlenecks we are encountering, primarily in using OpenAI's GPT-3.5 turbo model for query generation and optimization, are quite obvious. Despite the model's overall effectiveness, one significant bottleneck that has emerged is the query response time (QRT), which affects the speed at which the system can generate and refine queries. Most users expect sub-second results when entering a query. The delay introduced in our search engine can impact user experience and system efficiency. To address this issue and improve overall performance, further optimization is necessary. This may involve fine-tuning the algorithms used, revising the implementation of the model, or exploring alternative approaches to query generation and optimization.

In summary, this thesis lays a solid groundwork for constructing robust yet user-friendly search engines for various purposes. By integrating multiple multimodal models, it eliminates the "black box" effect, enabling more effective debugging to understand why specific results are returned. This approach enhances transparency and empowers developers to fine-tune and optimize the search engine's performance with greater insight and precision. In addition, a plug-and-play strategy is used to hook up any type of model to the search engine. Whether it is an open-source, self-hosted model, a closed-source model hosted externally, or even a single AI model wrapped in an embedding service, the approach remains applicable. This flexibility allows developers to leverage various AI technologies and integrate them seamlessly into the search engine ecosystem. As a result, the thesis advances the field of search engine development and provides valuable tools and methodologies for creating more transparent and efficient search systems across diverse applications.

Bibliography

Alqurashi, T., & Wang, W. (2018). Clustering ensemble method. https://scite.ai/reports/10.1007/s13042-017-0756-7

Arnab, A., Dehghani, M., Heigold, G., Sun, C., Lučić, M., & Schmid, C. (2021, March). Vivit: A video vision transformer. https://arxiv.org/abs/2103.15691

Bordes, A., Usunier, N., García-Durán, A., Weston, J., & Yakhnenko, O. (2013, December). Translating embeddings for modeling multi-relational data. https://hal.science/hal-00920777

Cangea, C., Veličković, P., & Liò, P. (2017, September). Xflow: Cross-modal deep neural networks for audiovisual classification. http://arxiv.org/abs/1709.00572v2

Deng, J., Guo, J., Zhou, Y., Yu, J., Kotsia, I., & Zafeiriou, S. (2019, May). Retinaface: Single-stage dense face localisation in the wild. https://arxiv.org/abs/1905.00641

Dimitriadis, D., Kirchhoff, K., Gu, Y., Stolcke, A., Droppo, J., & Yu, D. (2022, December). Robust speech recognition via large-scale weak supervision. https://arxiv.org/abs/2212.04356

Dirik, A., & Paul, S. (2023, February). A dive into vision-language models. https://huggingface.co/blog/vision_language_pretraining

Doh, R. F., Zhou, C., Arthur, J. K., Tawiah, I., & Doh, B. (2022, July). A systematic review of deep knowledge graph-based recommender systems, with focus on explainable embeddings. https://doi.org/10.3390/data7070094

Ensemble methods: A beginner's guide. (2020, July). https://medium.com/analytics-vidhya/ensemble-methods-a-beginners-guide-d14fe23b1ab9

Erk, K. (2012, October). Vector space models of word meaning and phrase meaning: A survey. https://doi.org/10.1002/lnco.362

Fan, Y., Xie, X., Cai, Y., Chen, J., Ma, X., Li, X., Zhang, R., & Guo, J. (2021, November). Pre-training methods in information retrieval. https://arxiv.org/abs/2111.13853

Farahani, A., Pourshojae, B., Rasheed, K., & Arabnia, H. R. (2020, December). A concise review of transfer learning. https://doi.org/10.1109/csci51800.2020.00065

Fine-tuning the model: What, why, and how. (2023, September). https://medium.com/@amanatulla1606/fine-tuning-the-model-what-why-and-how-e7fa52bc8ddf

Gesese, G. A., Biswas, R., Alam, M., & Sack, H. (2019, October). A survey on knowledge graph embeddings with literals: Which model links better literal-ly? https://arxiv.org/abs/1910.12507

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. https://arxiv.org/abs/1512.03385

Hosna, A., Merry, E., Gyalmo, J., Alom, Z., Aung, Z., & Azim, M. A. (2022, October). Transfer learning: A friendly introduction. https://doi.org/10.1186/s40537-022-00652-w

Hugging Face. (2023, April). Audio classification documentation. https://huggingface.co/docs/transformers/tasks/audio_classification

IBM, T. (2023, October). Why are there so many foundation models? https://www.youtube.com/watch?v=QPQy7jUpmyA

Jiang, J., Yang, S., Wang, J., & Long, M. (2022, January). Transferability in deep learning: A survey. https://arxiv.org/abs/2201.05867

Ju, C., Qiu, Z., Yao, T., Ngo, C.-W., & Mei, T. (2022, August). Expanding language-image pretrained models for general video recognition. https://arxiv.org/abs/2208.02816

Konstantinov, A. A., & Utkin, L. V. (2020, October). A generalized stacking for implementing ensembles of gradient boosting machines. https://arxiv.org/abs/2010.06026

Leal-Taixé, L., Milan, A., Reid, I., Roth, S., & Schindler, K. (2016). Siamese instance search for tracking.

Li, J., Li, D., Savarese, S., & Hoi, S. (2023, January). Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. https://arxiv.org/abs/2301.12597

Li, W., Paffenroth, R. C., & Berthiaume, D. (2021, September). Neural network ensembles: Theory, training, and the importance of explicit diversity. http://arxiv.org/abs/2109.14117v1

Lin, Z., Geng, S., Zhang, R., Gao, de Melo, G., Wang, X., Dai, J., Qiao, Y., & Li, H. (2022, August). Frozen clip models are efficient video learners. http://arxiv.org/abs/2208.03550

Liu, J., Zhao, A., & Huang, H. (2023, November). Decomposable transformations for high-quality edge-directed image scaling. https://arxiv.org/abs/2311.08526

Ma, N., Wu, Z., Cheung, Y.-m., Guo, Y., Gao, Y., Li, J., & Jiang, B. (2022). A survey of human action recognition and posture prediction. https://api.semanticscholar.org/CorpusID:249944018

Meng, Q., Huang, R., & Yuan, Y. (2023, August). Dtrocr: Decoder-only transformer for optical character recognition. https://arxiv.org/abs/2308.15996

Mienye, I. D., & Sun, Y. (2022, January). A survey of ensemble learning: Concepts, algorithms, applications, and prospects. https://doi.org/10.1109/access.2022.3207287

ML Foundations. (2023). open_clip: An open source implementation of CLIP [Accessed: 2024-04-30].

Multimodal machine learning: Data fusion. (2022, June). https://pub.towardsai.net/multimodal-machine-learning-data-fusion-d1d8776e2cb0

One, V. (2023, April). Analysis of automatic speech recognition tools. https://medium.com/version-1/analysis-of-automatic-speech-recognition-tools-10696f131910

Papers with Code. (2023, April). Benchmarking chinese text recognition: Datasets, baselines, and an empirical study. https://paperswithcode.com/sota/optical-character-recognition-on-benchmarking

Peng, Y., Huang, X., & Zhao, Y. (2018, September). An overview of cross-media retrieval: Concepts, methodologies, benchmarks, and challenges. https://doi.org/10.1109/tcsvt.2017.2705068

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., & Sutskever, I. (2021). Learning transferable visual models from natural language supervision. https://arxiv.org/abs/2103.00020

Rohit, K. (2023, August). Foundation models: The benefits, risks, and applications. https://www.v7labs.com/blog/foundation-models-guide

Sajid, H. (2023, May). Named entity recognition (ner) with python. https://www.wisecube.ai/blog/named-entity-recognition-ner-with-python/

Sanchez-Moreno, A. S., Olivares-Mercado, J., Hernandez-Suarez, A., Toscano-Medina, K., & Benítez-García, G. (2021, August). Efficient face recognition system for operating in unconstrained environments. https://doi.org/10.3390/jimaging7090161

Serengil, S. (2023, April). Deepface: A lightweight face recognition and facial attribute analysis (age, gender, emotion and race) library for python. https://github.com/serengil/deepface

Shrivastav, V. (2023, April). Insanely-fast-whisper: A cli for fast audio transcription using openai's whisper models. https://github.com/Vaibhavs10/insanely-fast-whisper

SYSTRAN. (2023, April). Faster-whisper: Optimization and enhancements for whisper asr models. https://github.com/SYSTRAN/faster-whisper

Tong, Z., Song, Y., Wang, J., & Wang, L. (2022, March). Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. https://arxiv.org/abs/2203.12602

Turney, P. D., & Pantel, P. (2010, March). From frequency to meaning: Vector space models of semantics. http://arxiv.org/abs/1003.1141v1

Vitor Guizilini, R. A., Kuan-Hui Lee, & Gaidon, A. (2022, March). Learning optical flow, depth, and scene flow without real-world labels. https://arxiv.org/abs/2203.15089

Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2022, July). Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. https://arxiv.org/abs/2207.02696

Wang, K., Yin, Q., Wang, W., Wu, S., & Wang, L. (2016, July). A comprehensive survey on cross-modal retrieval. http://arxiv.org/abs/1607.06215

Wang, X., & Davidson, N. (2010). The upper and lower bounds of the prediction accuracies of ensemble methods for binary classification. https://scite.ai/reports/10.1109/icmla.2010.62

What is layer freezing in transfer learning? (2023, December). https://www.quora.com/What-is-layer-freezing-in-deep-neural-networks-transfer-learning

Zaratiana, U., Tomeh, N., Holat, P., & Charnois, T. (2023). Gliner: Generalist model for named entity recognition using bidirectional transformer.

Zhang, C., Yang, Z., He, X., & Deng, L. (2020, March). Multimodal intelligence: Representation learning, information fusion, and applications. https://doi.org/10.1109/jstsp.2020.2987728

Zhao, Y., Fok, W. W. T., & Chan, C. W. (2019). Video-based violence detection by human action analysis with neural network. https://api.semanticscholar.org/CorpusID:209487471

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., & He, Q. (2019, November). A comprehensive survey on transfer learning. https://arxiv.org/abs/1911.02685

Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., & He, Q. (2021, January). A comprehensive survey on transfer learning. https://doi.org/10.1109/jproc.2020.3004555