UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

AN EMPIRICAL STUDY OF THE EFFECT OF VIDEO ENCODERS ON TEMPORAL VIDEO GROUNDING

TESIS PARA OPTAR AL GRADO DE
MAGÍSTER EN CIENCIAS DE LA COMPUTACIÓN

IGNACIO ALEJANDRO MEZA DE LA JARA

PROFESOR GUÍA:
FELIPE BRAVO MARQUEZ

MIEMBROS DE LA COMISIÓN:
BENJAMÍN BUSTOS CÁRDENAS
ANDRÉS ABELIUK KIMELMAN
JUAN REUTTER DE LA MAZA

SANTIAGO DE CHILE
2024

Un estudio empírico del efecto de los codificadores de vídeo en la Localización Temporal de Vídeo

El núcleo de *computer vision* (CV) reside en la comprensión e interpretación de vídeos largos e inéditos, una tarea dificultada por el esfuerzo manual necesario para analizar la ingente cantidad de contenidos de vídeo que se generan diariamente. La tarea de *Temporal Video Grounding* (TVG) se perfila como una solución clave en este ámbito, con el objetivo de desarrollar modelos que identifiquen y marquen los límites temporales de las acciones en los vídeos mediante el lenguaje natural. El campo ha evolucionado significativamente, pasando de modelos basados en sugerencias a técnicas avanzadas que utilizan modelos basados en transformadores, mejorando notablemente la precisión y la eficiencia.

A pesar de estos avances, sigue habiendo una brecha en la exploración de la representación de vídeos en la TVG. La dependencia de características específicas de los modelos de clasificación tradicionales puede conducir a un ajuste excesivo y a una generalización limitada. Para abordar esta cuestión, esta tesis propone un estudio empírico detallado en el que se analizan diferentes representaciones de vídeo en conjuntos de datos de referencia como Charades-STA, ActivityNet Captions y YouCookII. Se desarrolla un marco exhaustivo para aislar el efecto de las distintas representaciones de vídeo, utilizando un conjunto diverso de más de 10 tipos de modelos preentrenados, centrados en clasificadores de acción basados en CNN y transformadores.

Los resultados de este estudio son reveladores. Demuestran que la optimización de la selección de estas representaciones de vídeo puede mejorar significativamente el rendimiento del modelo, lo que se logra preservando las estructuras del modelo y modificando al mismo tiempo el codificador de vídeo. Los resultados también indican complementariedad entre las representaciones de vídeo, sugiriendo que explotar esta complementariedad podría abrir nuevas vías de investigación y perfeccionamiento en TVG.

El estudio también revela diferencias significativas en el rendimiento del modelo con solo cambiar el codificador de vídeo. Descubre patrones y errores claros derivados del uso de determinadas características, lo que indica una posible complementariedad entre ellas. Esto sugiere que una selección cuidadosa de características puede conducir a modelos más robustos y generalizables en TVG.

En conclusión, este trabajo subraya la relevancia de las representaciones de vídeo en las tareas de TVG. Aporta al campo de CV proveyendo un *framework* para la extracción de representaciones de vídeos, así como ideas y métodos fundamentales que podrían orientar el desarrollo de modelos multimodales más robustos en TVG.

Abstract

The core of computer vision (CV) lies in the understanding and interpretation of long and unprecedented videos, a task made difficult by the manual effort required to analyze the vast amount of video content generated daily. The task of Temporal Video Grounding (TVG) is emerging as a key solution in this field, aiming to develop models that identify and mark the temporal boundaries of actions in videos using natural language. The field has evolved significantly, moving from models based on suggestions to advanced techniques using transformer-based models, notably improving accuracy and efficiency.

Despite these advances, there remains a gap in exploring video representation in TVG. The reliance on specific features of traditional classification models can lead to overfitting and limited generalization. To address this issue, this thesis proposes a detailed empirical study analyzing different video representations in benchmark datasets such as Charades-STA, ActivityNet Captions, and YouCookII. An exhaustive framework is developed to isolate the effect of different video representations, using a diverse set of more than 10 types of pre-trained models, focusing on CNN and transformer-based action classifiers.

The results of this study are revealing. They demonstrate that optimizing the selection of these video representations can significantly improve model performance, achieved by preserving the model structures and modifying the video encoder simultaneously. The results also indicate complementarity among video representations, suggesting that exploiting this complementarity could open new avenues for research and refinement in TVG.

The study also reveals significant differences in model performance with just a change in the video encoder. It uncovers clear patterns and errors derived from the use of certain features, indicating possible complementarity among them. This suggests that a careful selection of features can lead to more robust and generalizable models in TVG.

In conclusion, this work underscores the importance of video representations in TVG tasks. It contributes to the field of CV by providing a framework for the extraction of video representations, as well as fundamental ideas and methods that could guide the development of more robust multimodal models in TVG.

Para toda la gente que amo y me acompañó en este proceso.

Agradecimientos

Es un recuerdo vivo la postulación a beauchef y la montonera de trámites que tuve que hacer para que me aceptaran en la facultad. En su momento, tenía una visión idealizada de la universidad y tenía una ilusión de que este camino me daría un mejor pasar en mi vida, pero en verdad tenía poca claridad de mi futuro. A pesar de estos deseos iniciales, este camino idealizado me ha entregado varios tropezones y grandes alegrías que me permitieron conocer a grandes personajes y me permitieron visualizar nuevos objetivos en mi vida.

Una de las ideas más potentes de Sartre dice: “Cada hombre es lo que hace con lo que hicieron de él” y no puede estar más en lo cierto. La persona que soy hoy en día se lo debo en gran parte a lo que mi familia me ha entregado y entrega día a día, resaltó enormemente la paciencia y crianza que me dio mi madre, mi “mami” y mi “tata”, quienes me instaban a estudiar día a día. Si bien he recibido mucho amor de mi familia, quiero resaltar a mi madre, quien me ha apoyado y ha escuchado en los momentos más oscuros durante mi vida. En esos momentos quizás cuando el estudio no era mi prioridad, decidió batallar para que estudiara y quien iba a pensar que en las vueltas de la vida me gustaría tanto lo que estudio.

Quiero agradecer a mi pareja Vanesa Suing por el apoyo que me da día a día y en particular mis bajones que tengo y como me sabe sacar de los agujeros emocionales a través del optimismo que suelo carecer.

En general tengo problemas con la comunicación, se me hace algo denso y laborioso. Es por esto que resaltó aquellos amigos que me han acompañado, dando: apoyo, ideas, consejos y buenos momentos durante estos últimos años. En particular, la universidad me ha hecho conocer muy buenas personas y resaltó la amistad que me han entregado especialmente: Maximiliano Parot, Pablo Badilla, Sebastian Tinoco, Gabriel Ramos y Gabriel Iturra. La verdad me han entregado momentos únicos y me han dado un gran apoyo durante mi desarrollo.

Asimismo, deseo agradecer profundamente a Felipe Bravo, quien ha sido un mentor excepcional, guiándome por un camino que jamás habría imaginado y enseñándome enormemente en los años que hemos trabajado juntos.

Finalmente, agradecer a tres personas que junto a Felipe me han llevado a desarrollarme como estudiante. Primero que todo quiero agradecer a Amanda Williamson que fue un primer acercamiento a la investigación junto a Felipe, si bien esta investigación no converge a una publicación, generó instancias de conversaciones muy llenadoras. Por otro lado a Cristian Rodriguez y Edison Marrese, quienes son unas bestias y me han entregado un montón de conocimiento en el área, expandiendo la vista que tenía del campo a una mucho más profunda.

Table of Content

List of Tables

List of Figures

Relevant Acronyms

AI Artificial Intelligence

C3D 3D Convolutional Networks

CNN Convolutional Neural Networks

CV Computer Vision

GB Gigabyte

GloVe Global Vectors for Word Representation

I3D Inflated 3D ConvNet

ICCV International Conference on Computer Vision

K400 Kinetics-400

LGVI Local-Global Video-Text Interactions for Temporal Grounding

ML Machine Learning

MLP Multi Layer Perceptron

MViT Multiscale Vision Transformers

NLP Natural Language Processing

RAM Random Access Memory

Rev-MViT Reversible Multiscale Vision Transformer

Sota State of the Art

SS v.2 Something-Something Version 2

SVM Support Vector Machine

TMLGA Temporal Moment Localization of a Natural-Language Query in Video using Guided Attention

TSM Temporal Shift Module

TVG Temporal Video Grounding

tIoU Temporal Interval Over Union

VAC Video Action Classifier

mIoU Mean Intersection Over Union

Chapter 1

Introduction

Understanding and reasoning about long, untrimmed videos is at the core of Computer Vision (CV). As humans have the ability to intuitively identify relevant moments within videos. However, manual search in large videos or a continuous review of multiple videos can cause errors and demand considerable time in locating events of our interest. For these reasons, providing an effective solution to the challenging problem of The task of Temporal Video Grounding (TVG) is essential for this and several problems in the field of computer vision. The task of TVG appears as a fundamental effort in CV, aiming to develop models that recognize and determine temporal boundaries of action instances in videos [44, 18, 15] using natural language queries [11, 19].

The task of temporal localization specifically aims to pinpoint a moment of relevance within a video, striving to provide the precise timestamp at which the queried event occurs. In this context, TVG operates with two inputs: the first is an untrimmed video, and the second is a natural language query detailing the search parameters that a machine learning framework is tasked to locate. An illustrative example can be seen in Figure 1.1, where the natural language query requests the identification of the exact moment a basketball player scores. In such a scenario, an end-to-end TVG architecture should have the capability to accurately determine and deliver the timestamp marking the beginning and end of the action queried by the user.

As such, work on the TVG task is extensive and includes a wide variety of approaches and techniques. While the original models were mostly suggestion-based, more recent techniques have aimed at predicting the start and end temporal positions directly, or by regressing them from the input video. The recent advent of transformer-based models [50] has also brought new developments, where the integration of pre-training stages or the direct addition of pre-trained vision and speech models has led to significant performance improvements.

In early developments, proposal-based models are responsible for generating a ranking of predefined candidate frames for the localization of relevant moments, while proposal-free models suggest that through multimodal representations it is possible to predict the probability that a time interval contains the relevant moment sought. Although contributions in this field are still limited, models using reinforcement learning for the localization of relevant moments and weakly supervised methods, which focus on generating models that

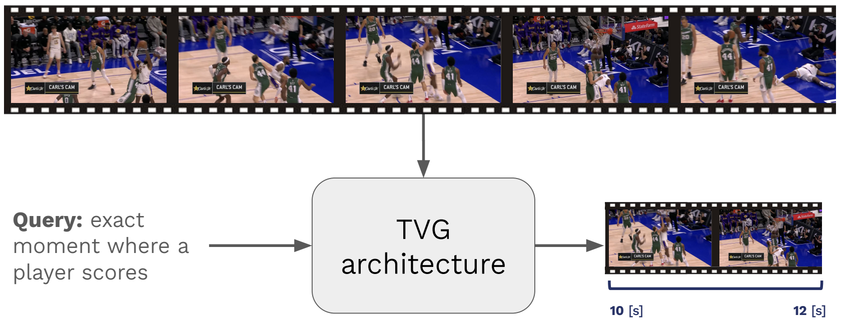


Figure 1.1: Representation of a general temporary video grounding framework.

mitigate the difficulty and cost associated with data annotation, stand out.

Despite the variety of methodologies employed in TVG tasks, there is a common element in the majority of studies: the use of vector representations to process both video and textual information integrated into the model. For textual representations, embeddings generated by word embedding algorithms or transformers like BERT are commonly used. Thanks to the standardization in the processes of extracting textual features, this step tends to be straightforward, easing the implementation of various variants without difficulty. In contrast, generating video representations often involves the use of video classifiers to create features. This process is considerably more challenging than handling text due to the absence of standardized frameworks for video feature extraction. Consequently, researchers are compelled to develop their own extraction pipelines, relying on pre-trained networks for video action classification.

Despite the large amount of prior work in the TVG task, we find that the role of the video representation has not been consistently investigated so far. Specifically, we observe that prior work has relied on features derived from action classification models such as C3D [48] or I3D [2], with alternatives remaining relatively unexplored. We speculate that this selection bias may lead to model designs that overfit or exploit spurious patterns in these features, without generalising in the long run. We suggest that the increased complexity of recent approaches in search of improved performance can be seen as indirect evidence of this point. This factor leads to two issues in the task of temporal localization: first, network overfitting to only one feature, and second, the potential selection of suboptimal features for problem resolution. This may result in the construction of overly complex networks to address the task when a more optimal solution could be achieved through feature permutation

In light of this issue, in this work we propose a comprehensive empirical study to shed light on the role of video representation in the TVG task. We consider three relevant benchmark datasets, Charades-STA, ActivityNet Captions, and YoucookII, and design a comprehensive framework that allows us to isolate the effect of different video representations by introducing minimal changes in the model architecture. We use a wide variety of pre-trained models to extract video representations, including more than 10 different types of models, resulting in more than 30 sets of extracted features for our data. These include, but are not limited to, well-known CNN and Transformer-based action classifiers.

Our results show that changes in the video representation can lead to substantial performance improvements by keeping the models as is, allowing a “classical” approach [41] to perform better by a large margin by simply changing the video encoder, findings that are consistent with similar observations recently made in the context of query representations [24]. Our experiments reveal clear pitfalls in the use of certain features and uncover complementarity between features, which could lead to further performance improvements if exploited. We hope that our work will provide the scientific community with additional tools to further improve performance on this task, and help to eliminate model biases towards the features that dominate today.

1.1 Research Problem

The field of temporal localization in video has received considerable attention in recent years, resulting in a number of diverse studies [60, 47, 59, 41, 42, 14, 58, 35, 30]. These investigations are characterized by the use of different techniques that aim to accurately identify specific events or actions within video timelines, while achieving efficient results in terms of storage and processing.

However, a common thread among these methodologies is their reliance on inputs to the end-to-end architecture that are dependent on both video and textual features. This critical reliance underscores the critical role of multimodal data in determining model performance, and highlights the indispensable synergy between visual content and linguistic information as the cornerstone of temporal localization efforts.

Despite the central role of video features in model input, the complexity of extraction and the lack of a standardized framework make video feature extraction difficult. This challenge, as shown in Table 1.1, leads several studies to prefer features extracted from the I3D model to feed the visual component of their models. This preference overlooks the importance of comparative analysis with alternative features and thus disregards the ”no free lunch” theorem, which states that ”there is no single model that fits all problems”.

In addition, a common practice in the TVG community is to pretrain video encoders on datasets identical to those on which they will be applied. This approach runs the risk of overfitting the model to the performance observed during training, potentially leading to overly optimistic results during testing phases. Such results, while initially promising, can be significantly degraded when the model is applied to samples from outside the training distribution, highlighting a critical area for improvement in the search for more robust and generalizable temporal localization models.

The highlighted issues pose significant challenges to progress in the TVG task, leading to stagnation in the field. This situation results in a constrained experimental scope, limited to a narrow set of traditional features and neglecting the potential of using state-of-the-art action classifiers for feature extraction. Such limitations not only stifle innovation, but also pave the way for suboptimal research practices, such as the development of overfitted models, thereby undermining the pursuit of more effective and robust TVG methodologies.

Table 1.1: Performance evaluation of different TVG architectures and the corresponding features used for the Charades dataset.

Model	Feature	Year	0.3	0.5	0.7	mIoU
PEARL [60]	I3D	2020	70.46	54.19	35.22	50.02
LOCFORMER [43]	I3D	2020	65.10	44.10	22.60	-
ACRMdt [47]	C3D/I3D	2020	-	53.09	31.75	-
VSLNet [59]	I3D	2020	72.96	59.46	35.48	51.38
TMLGA [41]	I3D	2020	67.53	52.02	33.74	-
DORi [42]	I3D	2020	79.25	68.41	50.56	60.29
ExCL [14]	I3D	2021	71.9	53.5	35.4	51.2
DRN [58]	I3D	2022	71.88	58.52	38.51	51.76
LGVI [35]	I3D	2022	73.47	57.53	38.33	53.01
UniVTG [30]	SlowFast	2023	72.63	60.19	38.55	52.17

1.2 Research Hypothesis

The success of Temporal Video Grounding (TVG) in Computer Vision is fundamentally influenced by video representation choices. Limited diversity in video features often leads to stagnation in performance and a lack of generalizability in TVG tasks. Addressing this, a diversified selection of video features is proposed to significantly boost the efficacy and flexibility of TVG models in varied real-world applications.

Research Questions

- Is the selection of video features beneficial for improving the performance of the task of temporally localizing a relevant moment?
- Is it possible to visualize the prediction biases inherent in the features used to train a TVG model?
- Is it feasible to observe performance benefits across various localization architectures without identifying specific effects attributable to a single network?
- Does training temporal localization models with different video features show a complementary component between the models?

1.3 Objectives

1.3.1 General Objective

This thesis focuses on the detailed analysis and comparison of various video features, extracted from a variety of action classifiers, with the goal of evaluating how the variation of these features affects the performance of temporal localization models. Inspired by methodologies from previous research, our study advances by proposing the use of features obtained from pretrained video classifiers. Additionally, it proposes the development of a framework designed to facilitate the efficient and user-friendly extraction of video features from an extensive range of pretrained action classifiers, currently considered state of the art.

Using the acquired feature set, we performed a thorough analysis that includes a performance comparison on temporal localization tasks between two different TVG models using different feature sets. The goal of this analysis is to uncover the effects of training models with different feature configurations. Consequently, this study aims to elucidate the impact of the selection of certain features on the performance of temporal localization tasks. In addition, it aims to determine whether there is complementarity in the video classification results derived from these features.

Through this proposal, the thesis aspires to offer significant contributions to the prevailing methodology in the design of temporal localization models, highlighting the importance of an appropriate selection of features and examining the potential complementarity and orthogonality among them to improve accuracy in temporal localization.

1.3.2 Specific Objectives

- Create a standardized framework for extracting video features using different types of action classifiers (based on temporality, convolutional networks, and transformers).
- To measure the performance impact of features obtained from action classifiers on the temporal localization task.
- Examine the potential bias introduced by using diverse features for training within the predictions generated by TVG models.
- Explore the consistency of the impact that variations in video features have on the performance of two models for temporal localization tasks.
- Determine whether there is complementarity in the predictions made by different Temporal Video Grounding models.

1.4 Results and Contributions

Our work introduces an innovative framework designed to streamline video feature extraction. Additionally, we conduct a comprehensive study showcasing the influence of diverse video features on TVG tasks and the orthogonal performance of these features. The contributions of this research are twofold:

1. Development of a Versatile Feature Extraction Framework: We propose a framework for effortlessly extracting a wide range of video features, parameterized by cutting-edge architectures in action classification. Built on the `pyslowfast` meta library, this framework simplifies generating feature vectors from videos and adapts to various hardware environments, allowing offline feature extraction for a multitude of video types.

2. Benchmarking Feature Performance in TVG: This contribution entails a thorough evaluation of features in TVG tasks across three major datasets. By training and comparing TVG models using these features, we uncover the distinct influences and biases introduced by varying the input features, highlighting their independent predictive capabilities.

In essence, this research offers a streamlined framework for video feature extraction and emphasizes the critical role of feature selection in temporal localization. A key recommendation is the exploration of feature orthogonality within the field, given their demonstrated independent predictive strengths.

1.5 Research Outcome

As a result of this work, we have published a paper in the workshop of the International Conference in Computer Vision (ICCV 2023), where part of the results shown in this document are exposed. On the other hand, the code of the framework built for the extraction of features is released [1], so that it can be used by anyone interested in extracting features from videos.

1.6 Thesis Structure

The development of this thesis is organized into four chapters, each addressing the following topics:

Chapter 2 aims to provide the reader with foundational concepts for a better understanding of the thesis development. It begins by explaining the knowledge areas involved in the task of temporal localization (Section 2.1). Subsequently, it delves into the Video Action Classifier (VAC) task (Section 2.3), outlining the objectives of VAC and introducing theoretical concepts defining some state-of-the-art networks. Finally, the chapter covers the task

[1] `https://github.com/Mezosky/VideoFeatures_TVG`

of temporal localization (Section 2.2), highlighting the two most famous methodologies to solve this problem, the metric used to measure performance, and key characteristics of three prominent datasets.

In Chapter 3, various experimental steps for developing this work are detailed. It begins by explaining how the data used in the study is preprocessed (Section 3.1), emphasizing the advantages of this preprocessing for subsequent processes. Next, it defines and delves into the framework constructed for feature extraction from Video Action Classifiers (Section 3.2). Subsequently, considerations for training two Temporal Video Grounding networks are discussed, along with the modifications required in annotations to ensure consistency with the extracted features (Section 3.3). The chapter concludes with the definition of experiments conducted with the features and their evaluation (Section 3.4).

Chapter 4 presents the results of each of the experiments described in section 3.4. Each experiment is accompanied by a comparative analysis, highlighting the differences between the features and TVG models used. Finally, Chapter 5 summarizes the conclusions derived from the work (Section 5.1) and provides a comprehensive list of topics for future research (Section 5.2).

Chapter 2

Background and Related work

To advance knowledge and enrich academic dialogue on the task of temporal video localization, it is essential to delve deeper into the topics involved in this task. For this reason, this chapter aims to immerse into the fundamental elements that underpin current research, exploring the background that characterizes it from the field of computer vision to the most specific details. Specifically, we will examine some of the main theoretical issues and empirical studies that form the backdrop against which this study is developed.

Firstly, in the opening section, we will briefly review definitions related to the areas of knowledge involved in this research: for this reason, definitions of computer vision, deep learning, and natural language processing are examined, in order to understand the purpose of these areas and comprehend how they relate to the task investigated in this work. Furthermore, due to the combination of visual and textual components presented in this work, the concept of multimodal tasks in machine learning is introduced. For this, an introduction to what the concept of multimodality refers to is provided, with the aim of understanding how the integration of these modalities can enhance the models' ability to tackle complex tasks involving information from multiple sources such as text, images, videos, or others.

Subsequently, the concepts that define the task of temporal localization are explored. To this end, a formal definition of the problem addressed by temporal video grounding architectures is provided, specifying the general structure of these networks and explaining how vector representations of text and videos form a crucial part within these architectures. Another aspect reviewed includes some of the most utilized approaches to address the problem. To understand how performance in temporal localization is measured, the metrics applied to evaluate the performance of TVG architectures are pointed out, specifying different definitions found in the literature. Finally, the inherent characteristics of the datasets used for both training and evaluation of the models are detailed. This multifaceted analysis aims to provide a comprehensive understanding of the methodologies that shape the landscape of temporal localization.

Finally, given the focus of this work on studying the impact of different video features on the task of temporal localization, we delve deeper into the element that enables feature extraction from videos: video action classifiers. Therefore, in this section, we formally define what constitutes the task of action classification, review the characteristics of the most commonly used datasets in this task, and deepen our exploration of the literature on action classifiers used in this research. The main focus is on explaining how these algorithms perform action recognition, how their structures can be advantageous for the task of temporal localization, and highlighting the differences between the different types of classifiers.

2.1 Scientific Disciplines

2.1.1 Machine Learning

Machine learning (ML) sits at the intersection of computational intelligence and computer science, dedicated to empowering computers to automatically acquire and learn meaningful patterns from data. In his seminal definition, Tom Mitchell succinctly captures the essence of this domain:

> "A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E."

– Tom Mitchell, 1997

The remarkable strength of ML algorithms lies in their ability to generate robust mathematical models that automatically extract and abstract relevant information from data, eliminating the need for explicit manual modeling. This stands in stark contrast to traditional rule-based programming, allowing the discovery of hidden, meaningful features from data and leading to the development of algorithms with far superior generalization capabilities.

During the training process, these models optimize internal parameters through sophisticated optimization algorithms to effectively solve the desired problem. The specific parameters chosen for optimization depend on the chosen ML model and the type of training employed.

A Landscape of Learning: Exploring Training Paradigms

ML techniques are categorized based on various factors, including the amount of data available and the level of supervision provided during training. Four main training paradigms dominate the landscape:

- Supervised Learning: Here, the model is trained on labeled data, meaning each data point has a known outcome. The model learns the relationship between the input features and the associated labels, enabling it to predict the outcome of unseen data. Examples include linear regression, support vector machines (SVMs), and decision trees.

- Unsupervised Learning: In contrast, unsupervised learning tackles unlabeled data, where no explicit outcomes exist. The model instead focuses on discovering hidden patterns and structures within the data, leading to applications like clustering, dimensionality reduction, and anomaly detection.
- Reinforcement Learning: This paradigm focuses on training an agent to learn by interacting with its environment. The agent receives rewards for desirable behavior and penalties for undesirable actions, allowing it to gradually learn optimal strategies for achieving its goals. Examples include deep Q-networks and policy gradient methods.
- Semi-supervised Learning: This hybrid approach utilizes a small set of labeled data alongside a larger set of unlabeled data during training. This framework leverages the informative power of labeled data while simultaneously exploiting the abundance of unlabeled data, leading to improved model performance.

The choice of training paradigm depends on the specific problem at hand and the available resources. Each paradigm offers unique capabilities and advantages, making ML a versatile and powerful tool for tackling a wide range of challenges across diverse domains.

Table 2.1: Comparison of Training Paradigms in Machine Learning

Paradigm	**Description**	**Examples**	**Application Areas**
Supervised Learning	Learns from labeled data to predict outcomes	Linear regression, SVMs, Decision trees	Classification, Regression, Prediction
Unsupervised Learning	Discovers hidden patterns in unlabeled data	Clustering, Dimensionality reduction, Anomaly detection	Data exploration, Customer segmentation, Fraud detection
Reinforcement Learning	Trains agents to learn through interaction with their environment	Deep Q-networks, Policy gradient methods	Robotics, Game playing, Autonomous vehicles
Semi-supervised Learning	Combines labeled and unlabeled data for improved performance	Co-training, Self-training, Label propagation	Text classification, Image segmentation, Natural language processing

2.1.2 Deep Learning

According to Kelleher [23], deep learning constitutes a subfield of artificial intelligence dedicated to crafting expansive neural network models capable of making precise, data-driven decisions. This approach proves particularly effective in scenarios characterized by intricate and unstructured data, leveraging the abundance of large datasets.

Deep learning addresses the challenge of extracting complex and abstract features from raw data, but the interesting thing is when variation factors such as a speaker's accent or the interpretation of an image require sophisticated understanding at an almost human level. If obtaining a representation is as difficult as solving the original problem, learning representations may not initially seem beneficial. However, deep learning overcomes this obstacle by introducing representations that are expressed in terms of simpler ones. It allows computers to build complex concepts by combining simpler elements.

The figure 2.1 shows how a deep learning system represents the concept of an image of a person by combining simpler concepts such as corners and contours, which are in turn defined in terms of edges. The example shows how the computer manages to understand the meaning of raw sensory input data, taking as input an image represented by a set of pixels. The complex mapping of functions from a set of pixels to the identity of an object is the challenge that deep learning addresses by decomposing the complex mapping into a sequence of simple nested mappings, each articulated by a different layer of the model. The visible layer, containing observable variables, receives the input, and a sequence of hidden layers extracts increasingly abstract features from the image. Called 'hidden' because their values do not appear in the data, these layers force the model to recognise which concepts are relevant to explain relationships in the observed data. The visualisations presented here show the types of features represented by each hidden layer. The first layer easily identifies edges based on the brightness of neighbouring pixels. Building on this edge description, the second hidden layer detects corners and extended contours. The third hidden layer uses the image description in terms of corners and contours to detect whole parts of specific objects. Ultimately, this description of the object parts of the image facilitates the recognition of the objects present in the image.

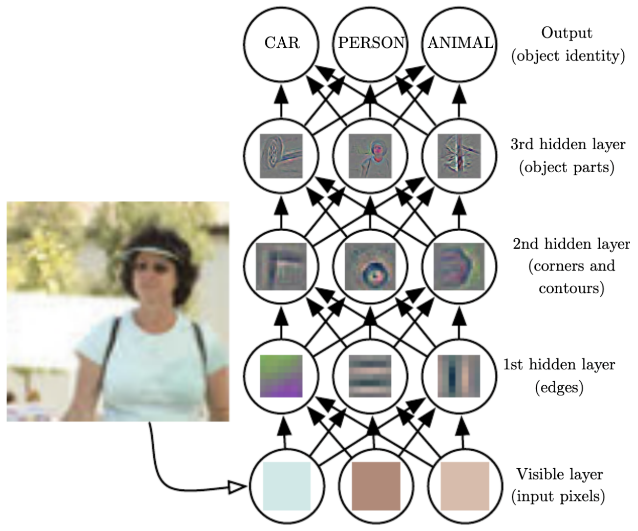

CAR
PERSON
ANIMAL
Output (object identity)
3rd hidden layer (object parts)
2nd hidden layer (corners and contours)
1st hidden layer (edges)
Visible layer (input pixels)

Figure 2.1: Representation of a deep learning model as shown in [16]. The figure illustrates the process of extracting features from a set of pixels, highlighting how each layer systematically extracts new features from the data, ultimately leading to the classification achieved in the final layer.

The classic example of a deep learning model is the feedforward deep network or multilayer perceptron (MLP). Operating as a mathematical function, a multilayer perceptron maps a set of input values to corresponding output values by composing many simpler functions. Each application of a different mathematical function serves as a transformative step, representing a new representation of the input. The notion of learning an optimal representation for the data provides one perspective on deep learning. Alternatively, the depth of deep learning enables the computer to acquire a multilevel computer program. Thinking of each layer of the representation as the state of the computer's memory after executing a parallel set of instructions illustrates this perspective. Deeper networks can execute more instructions in sequence, exploiting the power of sequential instructions that allow later steps to reference earlier results. In this deep learning paradigm, not all of the information in a layer's activations encodes factors that explain the input. The representation also contains state information, similar to a counter or pointer in a traditional computer program, that facilitates the execution of a program that understands the input. This state information, although unrelated to the content of the input, helps the model to structure its processing.

Today, deep learning, the engine of AI, powers a wide range of everyday services, from digital assistants and voice-enabled devices to fraud detection and self-driving cars. This transformative technology is improving healthcare through early disease detection, personalising online experiences, and making robots and industries more accurate and efficient. From language translation to environmental monitoring, deep learning is shaping the future of AI and impacting nearly every aspect of our lives.

2.1.3 Computer Vision

Computer vision is a multidisciplinary field that enables machines to interpret and understand visual information from the world, much like human vision. It involves developing algorithms and systems that allow computers to gain high-level understanding from digital images or videos. The ultimate goal of computer vision is to enable machines to perform tasks that typically require human visual abilities, such as object recognition, scene understanding, image and video analysis, and more. Actually this field, fueled by advancements in deep learning and artificial neural networks, has revolutionized various domains, from self-driving cars to medical imaging.

At its core, computer vision strives to bridge the gap between the digital realm and the physical world, enabling computers to extract meaningful information from images and videos. It encompasses a wide spectrum of tasks, including image and video classification, object detection, semantic segmentation, and instance segmentation.

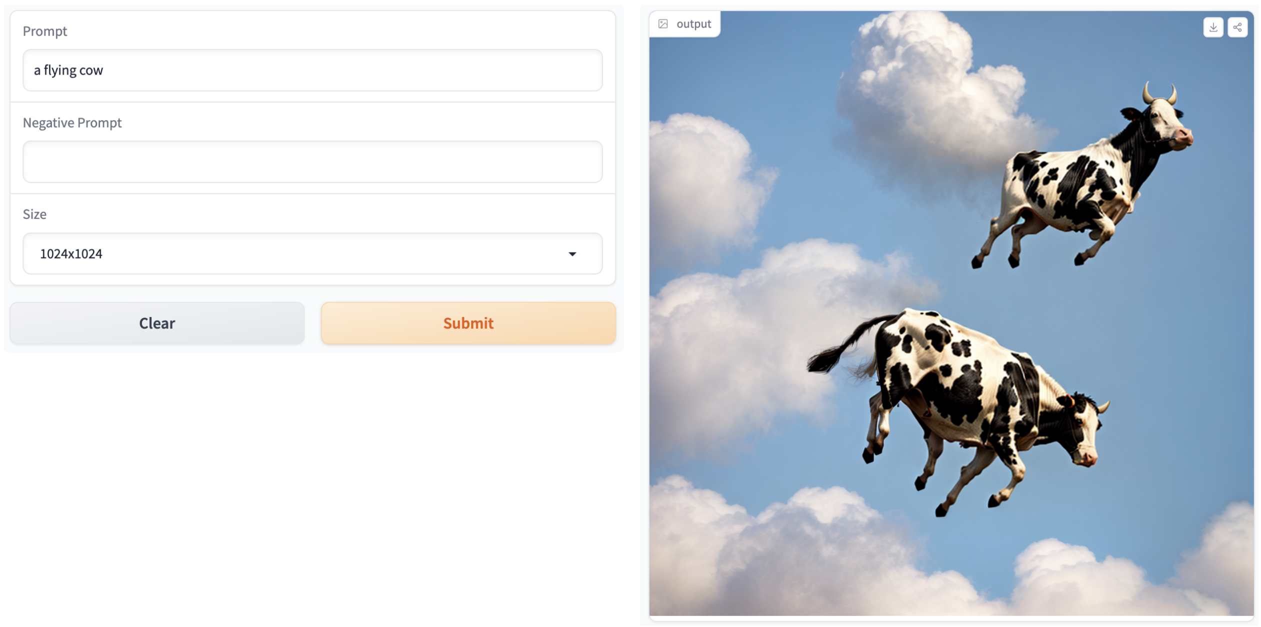
2.1.4 Natural Language Processing

One of the great advantages we possess as human beings is language. The ability to communicate provided our ancestors with competitive advantages in hunting, gathering, and community building compared to other species. Nowadays, humans have developed machines that use their own languages and can process vast amounts of information in a matter of seconds. This capability creates the need to develop methods to make human language accessible and/or understandable to computers, leveraging their computing power to solve language-related problems.

Natural Language Processing (NLP) is an interdisciplinary field that employs computational intelligence to address tasks involving human language in the form of text or speech. These tasks are well-defined and have knowledge of the expected output, including examples such as text translation, speech recognition software, etc. On the other hand, NLP should not be confused with Computational Linguistics, which seeks to understand human language through computation, posing deeper and more scientific questions about language. In other words, language is the subject of study for Computational Linguistics.

While NLP proves to be an interesting task, it is by no means a simple one. Despite humans being adept at learning languages, there is a poverty in understanding and describing the rules governing these systems. Therefore, working with human language becomes an ambiguous task, as changing a single word in a sentence can significantly alter its semantics. Another challenge is the dynamism of language, which undergoes a gradual evolution, causing variations in how we communicate our ideas based on factors such as culture, era, age range, etc.

2.1.5 Multimodal-Task


Prompt: a flying cow

Negative Prompt:

Size: 1024x1024

Figure 2.2: Example of a multimodal application in machine learning. For instance, the application enables image generation by simply providing a prompt that describes what one wishes to draw. In this case, the prompt is ”a flying cow,” and the result is the image created by a machine learning algorithm.

Humans utilize multiple stimuli to navigate the world and engage in various activities. The diverse sensory modalities we receive play a crucial role in identifying our environment in a multidimensional way. Among these, information from senses such as hearing and vision complements each other, enabling us to have a better understanding and perception of our surroundings.

The development of our brain has allowed us to establish optimal interaction between the different modalities we receive. This ability allows us to associate a sound with an animal or, conversely, associate an image with a sound, smell, or texture that we might experience upon interacting with the object in the image. This simultaneous interaction resulting from the combination of different channels is known as a ”multimodal” process.

In light of this, Multimodal Machine Learning is defined as the study of computational algorithms that incorporate multiple modalities within their architectures. In this context, multimodal tasks are often associated with the use of unstructured data such as images and text to solve a task collectively. A typical task is image generation, where the user inputs a query describing what they want to create, and a machine learning algorithm generates an image based on the provided prompt. Due to the utilization of multiple modalities, multimodal machine learning algorithms primarily require two aspects: effective representations for the input modalities and robust architectures that facilitate the correlation and comprehension of these representations during training.

Currently, multimodal models typically rely on deep neural networks due to their strong performance on unstructured data. In the realm of multimodal deep learning, the predominant modalities encompass inputs such as images, videos, text, and auditory elements like voice, sounds, and music. Nevertheless, less conventional modalities involve 3D visual data, LiDAR data, and depth sensor data.

2.2 Temporal Video Grounding

Temporal Video Grounding (TVG) presents a challenging task of aligning specific segments in a video with natural language queries. For instance, given the query ”persona take out a vacuum,” the objective is to identify the start and end times in the video where this moment occurs, as illustrated in Figure 2.3 (timestamps 24.9s and 42.4s).

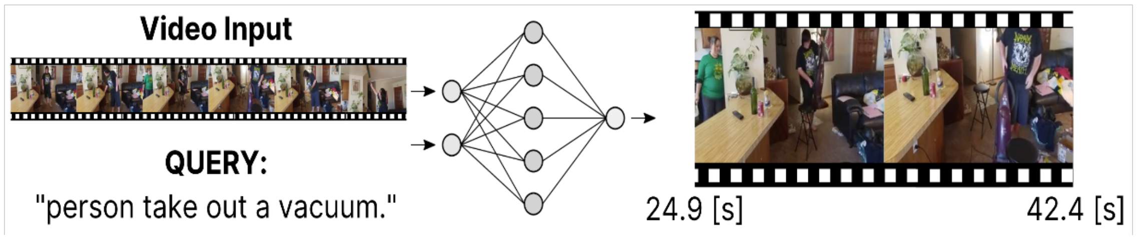


Figure 2.3: Illustration of Temporal Video Grounding Task

TVG presents a notable challenge as it requires not only the understanding but also the semantic alignment of video and language modalities. Additionally, TVG entails processing untrimmed videos, which often include irrelevant or noisy segments, adding complexity to the task. Despite these challenges, TVG offers tremendous potential for a variety of applications, including video summarization, editing, question answering, and captioning, making it a valuable asset in the field of multimedia analysis.

TVG Pipeline

Supervised learning in Temporal Video Grounding (TVG) requires a significant number of annotated samples, which presents challenges in terms of both the difficulty and cost associated with data annotation. To mitigate these challenges, recent research has been exploring weakly-supervised learning approaches, which learn from video-query pairs without needing detailed temporal event annotations. The traditional binary classification of TVG methods into proposal-based and proposal-free is increasingly seen as overly simplistic, prompting the need for a more nuanced taxonomy based on method architecture and learning algorithms.

In the realm of TVG, there is a lack of theoretical frameworks that delineate a standard structure or pipeline for TVG methods. Despite the presence of diverse and sophisticated architectures across various approaches, a TVG method conceptually typically comprises six key components, as depicted in Figure 2.4. It's important to note, as indicated by the dashed line in the figure, that the proposal generator is an optional component within these systems and its integration can vary in different stages of the methodology.

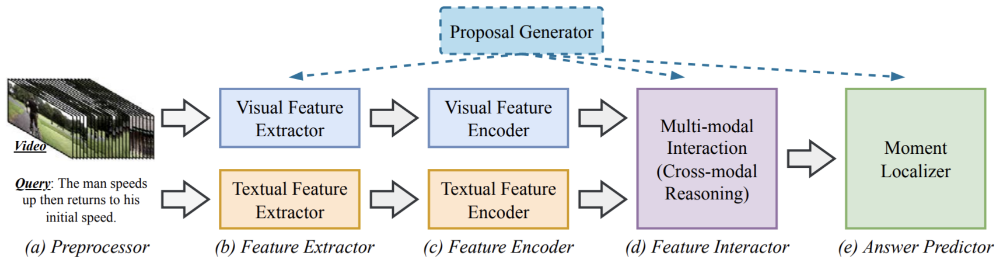


Figure 2.4: Temporal video grounding pipeline.

A TVG method takes a video-query pair as input, where the video, denoted as V, is a collection of consecutive image frames, and the query, denoted as Q, is a sequence of words. The preprocessor prepares inputs for feature extraction, such as downsampling and resizing image frames in the video and tokenizing words in the query sentence. The feature extractor converts the video frames and query words into their corresponding vector feature representations. The encoder module then maps the video and query features to the same dimension and aggregates contextual information to enhance the feature representation.

The interactor module, an essential component in TVG, learns multimodal representations by modeling the cross-modal interaction between video and query. Finally, the answer predictor generates moment predictions based on the learned multimodal representations. For proposal-based methods, the answer predictor makes predictions based on the proposals generated by the proposal generator. A proposal can be considered a candidate answer moment, generated at different stages. An example proposal is a video segment sampled from the input video. Proposal-free methods predict answers directly without the need for generating candidate answers.

2.2.1 Temporal Video Grounding Methods

The primary focus of solutions for Temporal Video Grounding (TVG) lies within the domain of supervised learning. In the early stages, methods relied on techniques such as sliding windows or segment proposal networks to pre-sample candidates. These candidates were then paired with queries, and answers were generated through cross-modal matching. However, the conventional "propose-and-rank" pipeline has been identified as inefficient due to the need for densely sampling candidates with overlap, resulting in redundant computations and reduced efficiency.

In response to these challenges, alternative approaches like anchor-based and proposal-free methods have emerged, embracing an "end-to-end" philosophy. These innovative methods encode the entire video sequence, preserving all relevant information within the model, progressively establishing themselves as the dominant solutions for TVG. Acknowledging the hurdles posed by supervised learning, recent studies have ventured into solving TVG through weakly-supervised learning. These approaches aim to alleviate the annotation burden by learning from video-query pairs without requiring detailed temporal event annotations.

Proposal-based Method

Proposal-based methods tackle the task of localizing a specific action instance in an untrimmed video by generating a set of candidate temporal segments, also known as proposals, and subsequently ranking these proposals to identify the one that best matches the target action. These methods offer a structured approach to TVG by explicitly identifying potential action instances before refining their temporal boundaries.

- Sliding Window-Based Methods: These methods employ a fixed-size window that slides across the video, generating proposals at each window position. This approach is simple and efficient but may miss subtle temporal cues beyond the window size.
- Proposal-Generated Methods: These methods utilize external modules or neural networks to generate proposal candidates based on the video content and query information. This approach offers more flexibility in capturing diverse temporal patterns but may introduce additional computational overhead.
- Anchor-Based Methods: These methods incorporate proposal generation directly into the model architecture, treating it as an integral part of the learning process. This approach achieves end-to-end learning and can dynamically adapt proposal generation based on the video and query.

Proposal-free Method

Proposal-free methods operate by directly predicting the start and end boundaries of the target moment within fine-grained video snippet sequences. Unlike their counterparts, proposal-free methods do not involve ranking an extensive array of proposal candidates. This streamlined approach not only enhances efficiency but also holds the potential to capture more nuanced temporal cues when contrasted with proposal-based methods.

Depending on the format of moment boundaries, proposal-free methods are categorized into two main types:

- Regression-based methods: These methods directly predict the start and end times of the target moment as continuous values. This approach is straightforward and can handle moments of varying lengths.
- Span-based methods: These methods predict the start and end indices of the target moment within the video snippet sequence. This approach is more flexible as it can handle overlapping or discontiguous moments.

2.2.2 Evaluation Metrics

Temporal Over Union

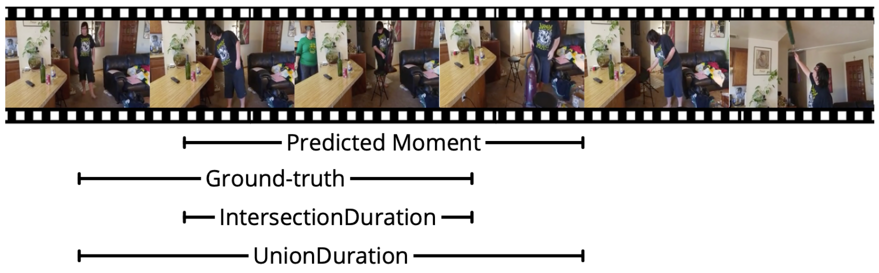


Figure 2.5: Example of how the interval is calculated over the union.

Temporal Interval Over Union (tIoU) is a widely used metric for evaluating the performance of temporal video grounding models. It measures the overlap between the predicted and ground-truth temporal extents of an action instance in a video. The metric takes values between 0 and 1, where 1 indicates perfect overlap and 0 indicates no overlap between the prediction and ground-truth. The equation is defined as following:

$$tIoU = \frac{\text{IntersectedDuration}}{\text{UnionDuration}}$$

In addition to the standard tIoU metric, various variants cater to specific contexts. For instance, the tIoU@0.5 metric employs a threshold of 0.5, considering only predictions with a tIoU score of 0.5 or higher as correct. Similarly, the tIoU@0.7 metric uses a threshold of 0.7.

The mIoU metric quantifies the average temporal Intersection over Union (IoU) across all annotations in the test set. Mathematically, it is expressed as:

$$mIoU = \frac{1}{N_q}\sum_{i=1}^{N_q} tIoU_i$$

An alternative method for computing mIoU involves not relying solely on a single prediction for each query. It's crucial to recognize that the top-ranked prediction from a TVG model may not consistently align with the ground truth for a given query. A more flexible evaluation approach is to consider the top-n retrieved moments for each query, denoted as $R@n, IoU@m$. Here, $R@n$ signifies the number of top predictions considered, and $IoU@m$ represents the threshold of IoU required to deem a prediction as correct.

Advantages of tIoU

The tIoU metric has several advantages for evaluating TVG models:

- It is a simple and intuitive metric that is easy to understand and interpret.
- It is a robust metric that is not sensitive to small changes in the predicted or ground-truth action extents.
- It is a versatile metric that can be used to evaluate the performance of TVG models on a variety of datasets and tasks.

Limitations of tIoU

The tIoU metric also has some limitations:

- It does not take into account the temporal ordering of action instances.
- It does not take into account the spatial location of action instances.
- It is not a fully normalized metric, as it can take on values greater than 1 for certain cases.
- Despite these limitations, tIoU is a valuable metric for evaluating TVG models and is widely used in the research community.

2.2.3 Dataset

A temporal video grounding dataset is a collection of videos and natural language descriptions that are used to train and evaluate models for the task of temporal video grounding. Temporal video grounding is the task of linking spoken language descriptions to specific video segments. In temporal video grounding, the model is given a video and a natural language description, such as a sentence or a caption, and its goal is to identify the specific segment of the video that corresponds to the description. This can involve tasks such as localizing the objects or actions mentioned in the description within the video, or associating a specific time interval with the description1.

Before explain in details some of relevant dataset, we define the following notations. Given a TVG dataset, we can denote its video corpus as $V = \{V_1, V_2, \ldots, V_N\}$ and its query set as $Q = \{Q_1, Q_2, \ldots, Q_M\}$, where N and M are the number of videos and queries, respectively. Note that multiple queries can be posed on the same video with different moments as answers, typically $M \geq N$ in TVG datasets. Given a video-query pair, a video V contains T frames $V = [f_1, f_2, \ldots, f_T]$, and a query Q has m words $Q = [q_1, q_2, \ldots, q_m]$, where the start and end times of the ground truth moment are denoted by τ_s and τ_e, $1 \leq \tau_s < \tau_e \leq T$. Here, we use the frame index to represent time points, based on a fixed frame rate or fps. Mathematically, TVG aims to retrieve the target moment starting from τ_s and ending at τ_e given a video V and query Q, i.e., $FT_{TVG} : (V, Q) \mapsto (\tau_s, \tau_e)$.

In summary, a temporal video grounding dataset should have the following characteristics to be relevant for the task:

- It should contain diverse and realistic videos that cover various domains, scenarios, and events.
- It should provide accurate and comprehensive annotations for the video segments and the natural language descriptions, such as the start and end time of the segments, the bounding boxes of the objects, and the semantic relations between the objects and the descriptions.
- It should have a sufficient number of videos and descriptions to enable effective learning and generalization of the models.
- It should have a balanced distribution of the video lengths, the segment durations, and the description complexities, to avoid biases and overfitting.

Charades-STA

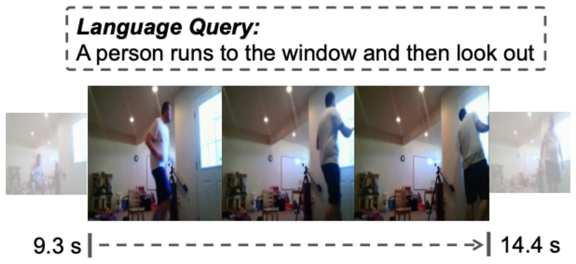


Figure 2.6: Example of the Charades-Sta dataset, in which you can see a video and the annotation associated with a moment.

Built upon the Charades dataset [46], which offers time-based annotations using a predefined set of activity classes and general video descriptions, Gao et al. [12] further enhanced the dataset by semiautomatically decomposing sentences describing the videos into smaller chunks and aligning them with the corresponding activity classes. These alignments were subsequently verified by human annotators, resulting in the effective association of original class-based activity annotations with their respective natural language descriptions, yielding a total of 13,898 pairs. Utilizing the predefined train and test splits, comprising 12,408 and 3,720 moment-query pairs, respectively, the videos in Charades-STA have an average duration of 31 seconds, containing an average of 2.4 moments, each spanning approximately 8.2 seconds on average. Charades-STA serves as a dataset for video moment retrieval, focusing on the task of locating a specific segment in a video that corresponds to a given natural language query. This dataset adds sentence-level temporal annotations to select videos from Charades, indicating the start and end times of the query segments. With queries ranging from simple to complex sentences with at least two clauses, the query segments cover less than half of the video length. Comprising 13,898 training samples and 4,233 test samples, including 1,378 complex sentences, Charades-STA presents a challenging dataset that demands proficiency in both natural language understanding and temporal activity localization.

ActivityNet Caption

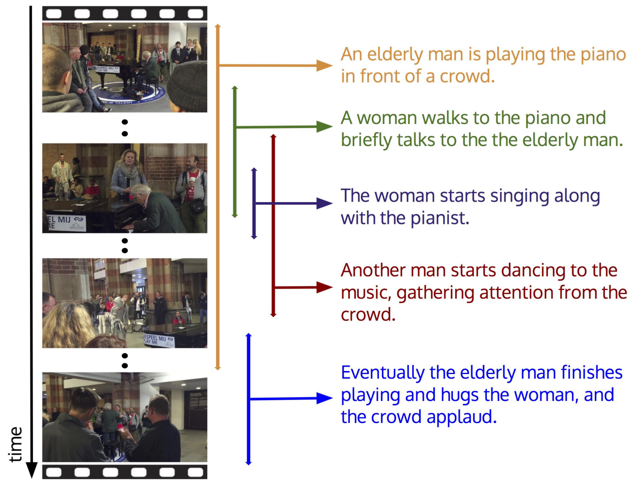

An elderly man is playing the piano in front of a crowd.
A woman walks to the piano and briefly talks to the the elderly man.
The woman starts singing along with the pianist.
Another man starts dancing to the music, gathering attention from the crowd.
Eventually the elderly man finishes playing and hugs the woman, and the crowd applaud.
time

Figure 2.7: Example of the ActivityNet dataset, where you can see that a video can contain multiple different annotations.

The ActivityNet Captions dataset extends the existing ActivityNet v1.3 by incorporating human-annotated captions for video segments, offering a rich resource for various video understanding tasks. These 100k captions, averaging 3.65 sentences per video and 13.48 words per sentence, provide precise temporal context and detailed descriptions of events, significantly enhancing model performance. Compared to Charades-STA, ActivityNet Captions boasts richer vocabulary, longer segments (36 seconds vs. 23 seconds), and more diverse content (200+ activity classes vs. 157). Notably, ActivityNet includes captions for the entire video, unlike Charades, providing comprehensive coverage for tasks like summarization and retrieval. These unique features make ActivityNet Captions invaluable for researchers and developers working on video captioning, retrieval, dense captioning, activity recognition, and other video understanding applications.

YourCook II

YouCookII encompasses 2000 extended, unedited videos featuring 89 distinct cooking recipes. Each video includes meticulously annotated temporal boundaries and imperative English sentences describing the procedural steps, as illustrated in Figure 2.8. These videos, sourced from YouTube and adopting a third-person viewpoint, are unconstrained and suitable for individual preparation at home with unfixed cameras. Offering an extensive array of recipe types and diverse cooking styles worldwide, YouCookII distinguishes itself with several key features. It constitutes a large-scale cooking dataset with 2000 annotated videos, allowing for unconstrained scenarios in individual home settings with unfixed cameras. The dataset comprises lengthy untrimmed videos lasting up to 10 minutes, providing an in-depth exploration of the cooking process. Notably, YouCookII temporally localizes procedure steps, describing them through English sentences, setting it apart from existing instructional video datasets.

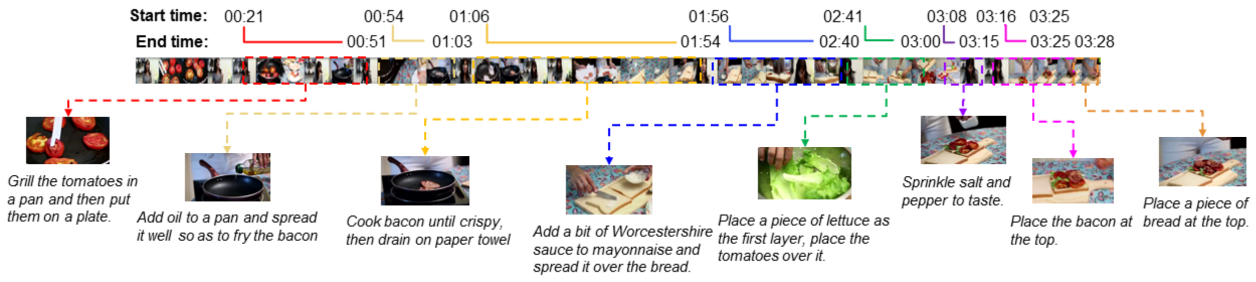

Start time:
End time:
00:21
00:51
00:54
01:03
01:06
01:54
01:56
02:40
02:41
03:00
03:08
03:15
03:16
03:25
03:25
03:28
Grill the tomatoes in a pan and then put them on a plate.
Add oil to a pan and spread it well so as to fry the bacon
Cook bacon until crispy, then drain on paper towel
Add a bit of Worcestershire sauce to mayonnaise and spread it over the bread.
Place a piece of lettuce as the first layer, place the tomatoes over it.
Sprinkle salt and pepper to taste.
Place the bacon at the top.
Place a piece of bread at the top.

Figure 2.8: Visual representation of the YouCookII dataset, where you can see the presence of longer videos and that multiple actions can appear in the dataset.

2.3 Video Action Classifiers (VAC)

The video action classifier task involves the application of computer vision and machine learning techniques to analyze and categorize human actions within video sequences. The primary objective is to develop algorithms that can automatically recognize, understand, and classify different types of actions performed by individuals or objects captured in a video. With this, there are three main approaches to video action classification:

- Holistic CNNs: These advanced networks leverage a unified 3D Convolutional Neural Network framework to meticulously process video data. By integrating three-dimensional convolutions, the holistic CNNs are specifically engineered to extract both spatial and temporal features directly from video frames, offering a nuanced understanding of the actions depicted. This method allows for a comprehensive analysis of the video content by learning feature representations that are inherently relevant to the dynamics and characteristics of the action being analyzed, thereby enhancing the accuracy and efficiency of video action classification tasks.

- Holistic Temporal Models: Characterized by their approach to mimic the three-dimensional convolutions used by holistic CNNs, these models aim to achieve a similar processing objective. However, they focus on minimizing memory consumption that is typically associated with 3D convolutions, by performing more efficient operations that still deliver comparable performance levels.
- Holistic Transformers: Following the emergence of transformer architectures, numerous studies have sought to incorporate this foundational model into action classification. These networks convert video frames into sequences of tokens and employ self-attention techniques to maximize the utilization of sequential content present in the frames. Distinguished by their ability to deeply integrate temporal components and establish meaningful relationships, holistic transformer models excel in capturing both spatial and temporal nuances, often resulting in enhanced performance.

This task plays a central role in a variety of applications such as video surveillance, human-computer interaction, content indexing, and automated video analysis, providing critical insights for a wide range of fields.

The journey begins with the collection of labeled video datasets that represent a wide variety of human actions. This diversity ensures that the classifier can accurately identify and distinguish between numerous activities by generalizing from the examples provided. It analyzes the video frames to extract key features, including spatial and temporal patterns, and uses these as the basis for its action predictions.

The pursuit of video action classification is not without its hurdles. It faces challenges such as lighting variations, background clutter, and the complex nature of human motion. To overcome these obstacles, a mix of strategies are employed, ranging from sophisticated deep learning models to temporal analysis to attention mechanisms, all aimed at increasing the accuracy of classifications.

At its core, the goal of video action classification is to provide automated, accurate, and instantaneous analysis of video content. This capability enables a wide range of applications, from enhancing security measures to deepening our understanding of human behavior, thus meeting the needs of various industries.

2.3.1 Dataset in VAC

Within a Video Action Classification (VAC) dataset, videos may encapsulate either a singular action or multiple actions. In the case of a single action, the video is tagged to denote the specific action that encompasses the entirety of the trimmed footage. For videos portraying multiple actions, meticulous annotations become imperative to specify the action occurring within each segment or frame. Take, for instance, a video depicting a person playing football, engaging in actions such as kicking the ball, running, jumping, and spinning. The corresponding annotations might detail that the segments from 0 to 5 seconds represent the action of kicking the ball, segments from 5 to 10 seconds embody the action of running, segments from 10 to 15 seconds capture the action of jumping, and segments from 15 to

20 seconds depict the action of turning. This level of granularity in annotations not only enhances the dataset's informational richness but also provides a nuanced understanding of actions within the temporal context of the video.

Concerning datasets, another pivotal aspect lies in their recording perspective, which can be categorized as either third person or egocentric. Third-person datasets are captured through external cameras, providing an external viewpoint, whereas egocentric datasets offer a first-person perspective, recorded from the viewpoint of the person engaging in the activity.

Some of the most relevant datasets include the following:

- **Kinetics:** kinetics is a large-scale dataset of 600 human action classes, with over 250,000 video clips. It is one of the most widely used VAC datasets due to its large size and diverse range of action classes. The dataset covers a wide variety of everyday activities, sports, and extreme sports.
- **AVA:** AVA is a dataset of 20 action classes, with over 16,000 video clips. It is a smaller dataset than Kinetics, but it contains more complex action classes that are more difficult to distinguish from each other. The actions in the AVA dataset are more intricate and require a more sophisticated understanding of motion to recognize.
- **Something-Something V2:** Something-Something V2 is an improved version of the original Something-Something dataset, addressing its limitations in terms of action diversity, annotation quality, and video length. It consists of 174,940 short videos of everyday actions, with each video approximately 3 to 7 seconds long. The dataset covers a broader range of action classes, including more subtle and nuanced actions, and features enhanced annotation quality and consistency.
- **Charades:** Charades is a dataset of 9,848 short videos of people acting out words and phrases. Each video is about 30 seconds long, and it is labeled with one of 157 words or phrases. The words and phrases in the Charades dataset are quite diverse, encompassing actions, emotions, and abstract concepts.
- **ActivityNet:** ActivityNet is a dataset of 16,000 long videos of human activities. Each video is about 1 to 5 minutes long, and it is labeled with one of 200 action classes. The action classes in the ActivityNet dataset are comprehensive, covering a wide range of everyday activities, sports, and social interactions.

2.3.2 Methodologies

Temporal Shift Module (TSM)

TSM stands out as a novel and efficient approach to temporal modeling for video understanding, seamlessly integrating into any two-dimensional convolutional neural network (CNN) architecture without introducing additional parameters or computational overhead. Its core mechanism involves shifting a portion of the channels along the temporal dimension, effectively facilitating information exchange between neighboring frames and enabling the network to capture temporal dependencies without the need for further convolutions or recurrent connections. TSM's operation entails dividing the input feature maps into two distinct parts:

- Base Feature Maps: These maps remain unaltered and undergo processing by the original 2D CNN architecture.
- Shifted Feature Maps: These maps undergo a temporal shift operation, where a specified portion of the channels is shifted along the temporal dimension by a predetermined offset. The resulting shifted feature maps are then concatenated with the base feature maps, providing the network with enriched temporal information.

TSM's compelling advantages over traditional temporal modeling methods include its remarkable parameter efficiency, maintaining the original 2D CNN architecture's parameter count. Its computational efficiency is also noteworthy, as TSM's operation involves simple shifting and concatenation operations, significantly reducing the computational cost of temporal modeling compared to 3D convolutions or recurrent neural networks (RNNs).

The Temporal Shift Module (TSM) represents a groundbreaking advancement in temporal modeling for video analysis. At its core, TSM introduces a dynamic temporal shifting mechanism that deviates from conventional temporal convolutions, aiming to enhance the model's ability to capture intricate temporal dependencies within video sequences.

The primary innovation of TSM lies in its integration of learnable temporal offsets into the convolutional operations. This introduces a dynamic temporal shift (Δt) to each spatial position (x, y) across the temporal dimension, facilitating adaptability to evolving temporal patterns. The mechanism is mathematically expressed as follows:

$$X'_{i,j,t} = X_{i,j,t+\Delta t},$$

where $X_{i,j,t}$ represents the input feature map at spatial position (i, j) and time step t, and $X'_{i,j,t}$ denotes the output feature map after the temporal shift.

The temporal shift (Δt) is learned during training and provides a mechanism for the model to dynamically adjust its temporal context for each spatial position. This adaptability is particularly valuable in scenarios where actions unfold over varying time scales within a video clip.

The Temporal Shift Module has demonstrated its efficacy in numerous applications, showcasing improved performance in tasks such as action recognition and temporal localization. By introducing dynamic temporal shifts, TSM offers a powerful tool for enhancing the temporal modeling capabilities of video analysis models, contributing to their accuracy and robustness.

I3D

The Inflated 3D ConvNet (I3D) represents a major advancement in video understanding, combining the benefits of 2D and 3D convolutions within a spatiotemporal architecture. This approach has led to state-of-the-art results in action recognition and detection tasks in video analysis.

I3D transforms 2D pre-trained convolutional networks into 3D by expanding the convolutional kernels through the temporal dimension, effectively 'inflating' them. This process involves replicating the 2D weights along the time axis, enabling the network to process spatial and temporal information simultaneously. The mathematical representation of this process is:

$$Y_{i,j,k,t} = \sum_{d=1}^{D} \sum_{h=1}^{H} \sum_{w=1}^{W} X_{i+d-1,j+h-1,k+w-1,t} \cdot W_{d,h,w},$$

where $Y_{i,j,k,t}$ is the output at spatial position (i, j, k) and time t, $X_{i,j,k,t}$ is the input feature map, and $W_{d,h,w}$ are the weights of the 3D convolution.

This inflation technique significantly boosts the model's capability to capture spatiotemporal features, enabling it to simultaneously learn about motion and visual appearance. I3D effectively maintains the computational efficiency of 2D convolutions while gaining the temporal insights provided by 3D convolutions.

The success of I3D is largely attributed to its ability to leverage knowledge from extensive 2D datasets and apply it to video content, creating a robust model for spatiotemporal feature extraction in video analysis. This amalgamation of 2D and 3D convolutional techniques makes I3D a highly effective and efficient tool for video understanding tasks.

X3D

X3D is a family of efficient video architectures that progressively expands a tiny 2D image classification architecture along multiple network axes, in space, time, width, and depth. This progressive expansion allows the model to achieve state-of-the-art performance while requiring significantly fewer multiply-adds and parameters compared to previous work.

- **Progressive Expansion:** X3D expands the network architecture gradually, starting with a small 2D image classification architecture and progressively increasing its complexity. This approach ensures that the network learns the most important features first, while also keeping the computational cost manageable.
- **Multi-Axis Expansion:** X3D expands the network along multiple axes, including spatial dimensions (height and width), temporal dimension (frame number), and network depth (number of layers). This multi-axis expansion allows the network to effectively capture information from both spatial and temporal patterns in videos.

- **Efficient Design:** X3D employs several design choices to further enhance its efficiency, such as using lightweight convolutions, reducing the number of parameters, and using efficient activation functions.

The spatial encoder extracts spatial features from the input video frames, while the temporal encoder extracts temporal features from the video frames. The two encoders are stacked to allow the network to learn both spatial and temporal dependencies in the videos.

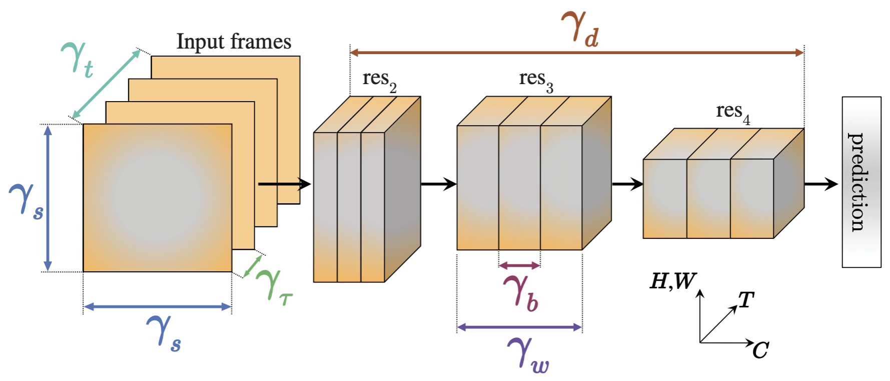


Figure 2.9: X3D architecture with the different gamma parameters to be learned: Temporal duration γ_t, frame rate γ_τ , spatial resolution γ_s, width γ_w, bottleneck width γ_b, and depth γ_d.

The temporal encoder is a novel component of the X3D architecture. It is designed to extract temporal features from the video frames. The temporal encoder consists of a series of temporal blocks. Each temporal block consists of a residual connection and a temporal attention mechanism. The residual connection allows the network to learn long-range temporal dependencies in the videos. The temporal attention mechanism allows the network to focus on the most important temporal features.

The output of the stacked spatial and temporal encoders is a set of feature maps that contain both spatial and temporal information about the video. These feature maps are then fed to the core encoder, which combines the spatial and temporal features to produce a final representation of the video.

The expansion of intermediate parameters in X3D results in different types of expanded networks, categorized as XS, S, M, L, and XL. Each expansion level introduces a specific increase in the network's capacity, corresponding to different levels of complexity and resource requirements.

XS and S networks represent the smallest and least complex configurations, suitable for resource-constrained devices or real-time applications. M and L networks provide a balance between performance and efficiency, making them suitable for a wider range of applications. XL networks represent the largest and most complex configurations, achieving the highest performance but requiring more computational resources.

The choice of expansion level depends on the specific application requirements and resource constraints. For instance, a mobile application might prioritize an XS or S network for real-time performance, while a high-performance video analysis system might utilize an L or XL network for maximum accuracy.

SlowFast

SlowFast is a novel architecture for video understanding that effectively integrates fast and slow pathways for temporal feature extraction, enabling accurate and efficient video analysis. The architecture employs a dual-stream architecture, consisting of a fast pathway and a slow pathway, to capture both rapid motion cues and detailed contextual information from video sequences.

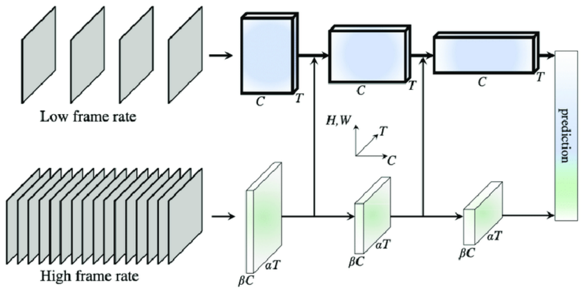


Figure 2.10: SlowFast network architecture. At the top is the slow path of the network, while below is the fast path.

The fast pathway processes video frames at a high frame rate, typically 64 frames per second (FPS), to capture rapid motion changes and temporal dependencies. It utilizes lightweight convolutional neural networks (CNNs) to extract low-resolution features efficiently, reducing computational cost while maintaining adequate temporal information.

In contrast, the slow pathway processes video frames at a lower frame rate, typically 8 FPS, to capture detailed contextual information and global relationships within the video. It employs more complex CNNs to extract high-resolution features, providing a comprehensive understanding of the video content.

To further enhance efficiency, SlowFast employs efficient temporal modeling techniques in both pathways. The fast pathway utilizes temporal convolutions with small kernel sizes to effectively capture short-term temporal dependencies while minimizing computational overhead. The slow pathway employs a temporal subsampling mechanism to reduce the number of frames processed, focusing on key frames that contain significant information. This reduces the computational burden while maintaining adequate temporal context.

The features extracted from both the fast and slow pathways are then fused together using an aggregation mechanism, enabling the network to combine the benefits of both temporal information sources.

A common aggregation approach is to concatenate the features along the feature dimension, creating a more comprehensive representation of the video's temporal dynamics. This allows the network to learn relationships between fast and slow temporal cues, leading to improved recognition and understanding.

Multiscale Vision Transformers - MViT

MViT is the first network that effectively integrates multi-scale feature extraction and transformer based temporal modeling to achieve state-of-the-art performance in video understanding tasks while maintaining computational efficiency.

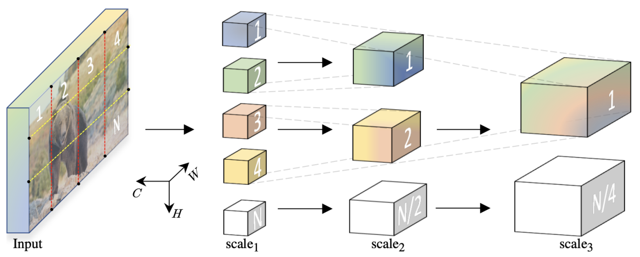


Figure 2.11: Multiscale Vision Transformers learn a hierarchy from dense (in space) and simple (in channels) to coarse and complex features. Several resolution-channel scale stages progressively increase the channel capacity of the intermediate latent sequence while reducing its length and thereby spatial resolution.

The network employs a hierarchical encoder-decoder architecture to extract features at multiple scales. This multi-scale approach enables MViT to capture both fine-grained details and global contextual information from video sequences.

The encoder consists of multiple stages, each with a different spatial resolution. At each stage, the video frames are passed through a series of convolutional layers to extract features. The features from each stage are then concatenated together to create a multi-scale feature representation. The decoder takes the multi-scale feature representation as input and reconstructs the video frames. The decoder also consists of multiple stages, each with a different spatial resolution. At each stage, the features are passed through a series of convolutional layers to upsample the resolution and predict the video frames.

MViT utilizes transformer blocks, inspired by natural language processing, to model temporal dependencies within videos. Transformers excel at capturing long-range temporal relationships and context, which is crucial for understanding video content.

The transformer blocks are applied to the multi-scale feature representation to capture temporal relationships between the features at different scales. This allows MViT to understand the overall temporal structure of the video and how the individual parts of the video relate to each other. MViT incorporates several design choices to enhance its efficiency, including:

- Using lightweight convolutions for feature extraction: Lightweight convolutions are used to extract features at each stage of the encoder and decoder. Lightweight convolutions have fewer parameters and FLOPs than standard convolutions, which reduces the computational cost of the network.

- Employing early fusion of multi-scale features: Early fusion of multi-scale features allows MViT to capture both fine-grained details and global context early in the processing pipeline. This reduces the computational cost of the network, as it does not need to process the features at each scale independently.

- Utilizing efficient transformer architectures: Efficient transformer architectures are used to model temporal dependencies within videos. Efficient transformer architectures have lower memory and computational requirements than standard transformer architectures.

Rev-MViT

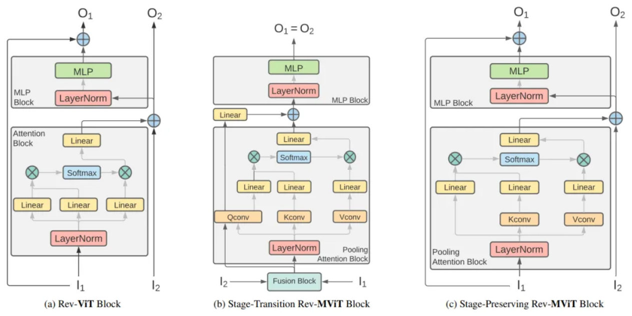


Figure 2.12: Representation of the blocks that make up Rev-MViT. In them, there is an attempt to simulate the operations performed in a traditional ViT block, but taking advantage of the use of inverse operations with $I_{1,2}$.

The Rev-MViT architecture represents a breakthrough in the evolution of vision transformers, addressing one of the most pressing challenges in deep learning: the high memory consumption of conventional transformers. By introducing a reversible mechanism, Rev-MViT allows for significant memory savings during the training process, enabling the deployment of deeper and more complex neural networks even on hardware with limited memory resources. This innovation not only makes advanced computer vision models more accessible, but also improves their performance in a variety of tasks, including image classification, object detection, and action recognition. Rev-MViT's design, which emphasizes memory efficiency without sacrificing accuracy, underscores its potential as a robust backbone for future vision-based applications.

The Reversible Vision Transformer (ViT) adopts a unique design that splits the model's operation into two parallel pathways, processing input data through a series of specialized blocks known as Reversible ViT blocks. These blocks maintain the core principles of the Vision Transformer architecture, yet modify them to allow for the backward calculation of gradients in a memory-efficient manner. Essentially, this setup manipulates two sets of input data, enabling the network to conserve memory by cleverly reusing computational steps when updating model weights during backpropagation.

In more detail, referring to Figure 2.12, the Reversible MViT framework employs block (a) for concurrently managing two distinct data streams, I1 and I2, leveraging Vision Transformer (ViT) principles but engineered for bidirectional transformation flow. This innovative setup capitalizes on a dual-pathway strategy, segmented into two primary blocks: Stage-Transition

Blocks (b): These components strategically connect the dual pathways, skillfully modulating data across dimensions by adjusting channel counts and image resolution. They act as dynamic conduits, facilitating seamless integration and transition of information between different processing stages. Stage-Preserving Blocks (c): As the architectural backbone, these blocks are pivotal for the continuity of the network's function, safeguarding the fidelity of input features' dimensions throughout their journey. They underscore the architecture's commitment to preserving data integrity while ensuring comprehensive processing and synthesis of information.

MViT-V2

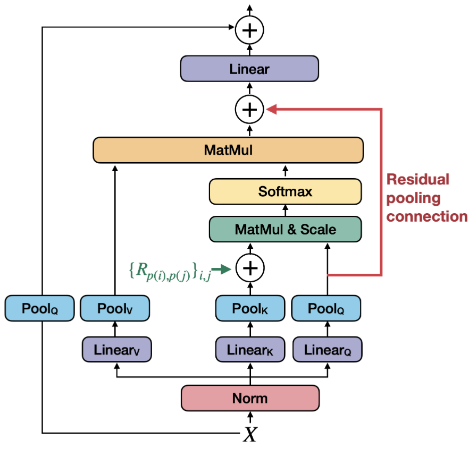


Figure 2.13: Visualization of the Enhanced Pooling Attention Mechanism: Integration of Decomposed Relative Position Embedding and Residual Pooling connection modules within the Attention Block.

The Multiscale Vision Transformers Version 2 (MViT-V2) represents a remarkable leap forward in the field of Vision Transformers, meticulously crafted to address the intricate challenges that its predecessors faced, especially in the nuanced realm of processing and deciphering visual data across a multitude of scales. This iteration heralds significant enhancements that considerably bolster its efficacy in various applications, notably in the realm of video action classification. Through its refined processing of multiscale representations, MViT-V2 achieves a more nuanced and efficient extraction of features at different resolutions. This nuanced capability is instrumental in capturing the intricate visual patterns and dynamics that pervade video content, offering a deeper understanding of the actions and interactions within.

At the heart of MViT-V2's innovations are its advanced self-attention mechanisms, which have been optimized not just for greater computational efficiency but also for a heightened ability to discern the long-range dependencies woven within the visual data. This evolution results in a pronounced improvement in the model's ability to recognize complex actions by keenly capturing both the minute details and the broader contextual strokes present in video sequences. Moreover, MViT-V2 is designed with an eye towards computational economy, maintaining a judicious balance between model complexity and operational performance. This strategic balance ensures that MViT-V2 is exceptionally suited for extensive video analysis tasks, delivering high levels of accuracy without placing undue demands on computational resources.

A noteworthy feature of MViT-V2 is its innovative implementation of decomposed relative positional embeddings. This approach intricately encodes the positional relationships of each patch in the input, segregating the embedding into horizontal and vertical offsets and scales. This method allows MViT-V2 to capture more refined spatial information and adapt more seamlessly to various input resolutions, enhancing its ability to understand and interpret visual content from diverse perspectives.

Additionally, MViT-V2 introduces residual pooling connections, which significantly enrich the flow of information across different scales of the transformer. This innovative approach, diverging from the singular pooling layer of its predecessor, integrates residual links between pooled and original patches. Such an integration preserves high-resolution details more effectively, enriching the feature representation of the transformer and thereby augmenting its analytical capabilities.

In comparative performance benchmarks, MViT-V2 has consistently demonstrated superior results, outpacing earlier models in terms of both accuracy and efficiency across a spectrum of tasks, including video action classification. Its robust and adaptable performance across a wide array of datasets highlights its potential as a powerful tool in the arsenal of computer vision technology. The development of MViT-V2 significantly contributes to the ongoing advancements in leveraging transformer architectures for visual data processing, representing a profound step forward in our quest to achieve a more nuanced understanding of visual content through the lens of machine intelligence. This enhancement in the technological landscape not only broadens the horizons of what's achievable within computer vision but also paves the way for future innovations in the field.

Chapter 3

Methodology

This chapter is dedicated to detailing the various processes applied to videos, elucidating the design of our feature extraction framework, and describing the experimental approaches undertaken on diverse datasets. The organization of this chapter unfolds as follows:

Initially, we introduce the video preprocessing steps designed to mitigate the computational demands in later phases. Following this foundational groundwork, we delve into the heart of our research—the development of a feature extraction framework. This section clarifies the framework's significance and its influence on the task of temporal localization. Subsequently, we discuss dataset annotation considerations and detail the training processes for the 11 features across various datasets. The chapter culminates with a comprehensive overview of the experimental setup, aimed at evaluating the effect of these features on the efficiency of temporal localization tasks.

3.1 Data Preprocessing

The preprocessing phase aims to adapt the data to a format compatible with subsequent processes. This stage consists of three fundamental steps: reducing the frames per second in the videos, decreasing dimensionality, and transforming the videos into the MP4 format.

To carry out these activities, the `ffmpeg` software (available at `https://ffmpeg.org/`) is employed to perform the necessary transformations on each of the datasets being worked on. Relevant considerations in these transformations include:

Reduction to 24 frames per second per video: This step aims to standardize the videos to a common framerate of 24 frames per second, enhancing coherence in subsequent processing phases. It is essential to note that, in the case of videos with less than 24 fps, `ffmpeg` performs frame interpolation to compensate for the difference. This can result in two situations: loss of information and/or generation of redundant information in sampling. However, the magnitude of these issues should be limited, as the eliminated information is in the order of milliseconds, and actions in the videos are defined in the range of seconds.

Cropping to 320x240 pixels in the videos: The purpose of this stage is to reduce the resolution of images in the videos to 320x240 pixels, aiming to obtain a standardized input in later processing stages and lighten the video load. Although this reduction in dimensionality implies a decrease in the quality of spatial information visualized in the videos, the level of compression should not be detrimental, as the nature of the datasets used does not require precise localization of small objects.

Transformation to the MP4 format: The format is standardized to ensure that reading and obtaining videos occur in a uniform context.

It is important to emphasize that the execution of this stage involves a reduction in the original features of the videos. Although this results in a decrease in quality and part of the information represented in the videos, it allows for a significant reduction in the computational resources required for generating features in videos. Among the benefited aspects are the lower amount of RAM needed to load videos into memory, and, on the other hand, a higher number of processes can be executed in parallel, thereby increasing the speed of the processes.

The reduction in the final weight of the datasets provides a clear example of the benefits of this reduction:

Dataset	Original Weight (GB)	Reduced Weight (GB)	Reduction (%)
Charades	55	9.5	82.72
ActivityNet	337	107	68.24
YouCookII	152	26	82.89

Table 3.1: Reduction in dataset weight.

The table demonstrates a reduction of over 80% in the original size of some datasets, implying a substantial decrease in computational resources for processing.

3.2 Feature Extraction

In the realm of computer vision, the practice of employing a pre-trained neural network initially designed for task X to extract features for addressing a different task Y is a well-established strategy. This concept, particularly the notion of a "backbone" as a feature extractor in deep learning, has been extensively deliberated and integrated into the fabric of contemporary research within the field. The inception of this idea can be traced back to the exploration of deep convolutional neural networks (CNNs) and the remarkable efficacy they demonstrated in learning hierarchical representations of data.

During the nascent stages, several early deep CNN models, such as AlexNet (2012) and VGGNet (2014), showcased the prowess of CNNs in feature extraction, particularly in tasks like image classification. Trained on extensive datasets of labeled images, these models adeptly learned to extract increasingly intricate features as they traversed from lower to higher layers of the network. This early success laid the foundation for the paradigm of employing pre-trained CNNs as feature extractors for a diverse array of downstream tasks.

The term "backbone" gained prominence concurrently with the evolution of transfer learning techniques. Researchers discerned that the knowledge acquired by a CNN on a substantial dataset could be effectively transferred to other tasks by repurposing its early layers as a feature extractor. This innovative approach empowered them to capitalize on pre-trained weights, circumventing the need to train the entire network anew. This not only significantly reduced training time but also contributed to enhanced overall performance.

In the work presented at [6], the term "backbone" gained prominence in tandem with the evolution of transfer learning techniques. Researchers astutely recognized that the knowledge acquired by a Convolutional Neural Network (CNN) on a large dataset could be effectively transferred to other tasks by repurposing its early layers as a feature extractor. This strategic approach empowered them to harness the benefits of pre-trained weights, circumventing the necessity to train the entire network from the ground up. The result was a substantial reduction in training time and a marked improvement in overall performance.

Given an untrimmed video $V \in \mathcal{V}$ that can be characterized as a sequence of frames such that $V = v_t$ with $t = 1, \dots, l$. Each video of V is annotated with a natural language sentence $S \in \mathcal{S}$, where S is a sequence of words $S = s_j$ with $j = 1, ..., m$, describing what happens in a certain time interval. Formally, this interval is defined by a time in seconds that marks the start t_s and end t_e of the annotation.

Following previous work, our pivot TMLGA model uses pre-trained action classification models to obtain a sequence of vectors to represent a given video. Critically, these input vectors are just passed through one projection layer before being fed into the localization module, which ensures that the amount of changes introduced to the model is minimized for our experiments. Although the original implementation relies on I3D [2] we instead consider a wide selection of pre-trained models for feature extraction, which we propose to group into three main categories. First, we test CNN-based video classification approaches, including C2D [48], I3D, SlowFast [10], X3D [9] and Non-Local variations in one classifier [53]. For these models, we modify the classification head of the last ResNet or ConvNet used, extracting the representation generated before dropout, final projections and classification layer of the network. Secondly, we look at Transformer-based action classification models, including MViT [8], MViT2 [27], and Rev-MViT [33]. Similarly to thee CNN-based case, for these models we represent the video using the vectors of the last layer of the transformer before the linear classifier. Finally, we study temporal reasoning models such as TSM [28], which provide a different angle in tackling the action classification task. In this model, the video data is divided into segments, each containing 8 buckets, where each bucket consists of 8 consecutive frames. This segmentation approach creates smaller video subsequences, allowing the temporal network to effectively permute frames and capture their temporal dependencies. Similarly to the other cases, the last projections of the temporal network are discarded to obtain the video feature representation.

To illustrate the generalised feature extraction process, Figure 3.1 shows the procedure used. The figure shows the creation of a bypass in the networks, omitting the neck and head layers. As mentioned earlier, this omission is mainly because the neck layers tend to introduce upscaling or concatenation of previously obtained features, and the head layers generate predictions using these features. Consequently, the outputs do not preserve the features in a raw format, but rather in a processed form. Each of the modified networks produces an output vector, the dimensions of which vary according to the features of the network and the video. It's important to note that, in practice, the bypass occurs before the application of dropout, MLP projections and/or softmax-like functions.

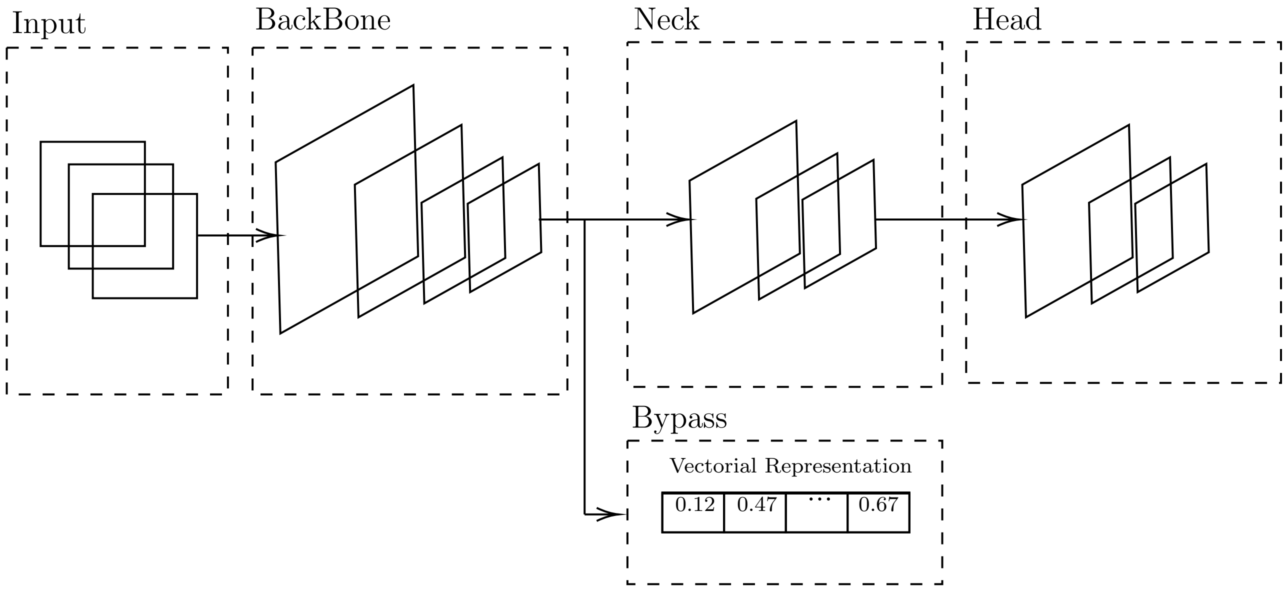


Figure 3.1: Representation of a video action classifier and the different blocks that make up an architecture.

We assume the existence of a video feature extractor F_V, capable of generating features from multiple encoders based on pre-trained action models. The feature extractor is responsible for mapping l frames of a video to l/s features, where s is the sampling rate of the pre-trained video model used for feature extraction. From the above, given a video V, $F_V(V)$ generates a set of vectors $G = g_i \in \mathbb{R}^{l/s \times f}$, where l/s represents the temporal dimension of the video and f represents the width of the information contained in the generated vector for a specific time slice is the dimenstionality of the space where the features lived.

Specifically, we adapted the forward process of each architecture used, incorporating small modifications depending on the requirements of the architectures. For the architectures C2D [48], I3D [2], SlowFast [10], MViT [8] (based on ViT [5]), MViT2 [27], and Rev-MViT [33], we modified the classification head of the last ResNet used, capturing the representation generated before dropout and final projections of the network in classification. As for X3D [9], a similar process was followed, with the main difference being that this architecture does not utilize a ResNet-based head but possesses its own distinctive characteristics.

For feature extraction, two action classification projects are used: one proposed by the meta research team [7], and the second developed by Lin et al. [28]. The choice of the meta team's project is based on their extensive development of action classifiers, which maintain similar logic in terms of feature extraction. This simplifies the coding of a general wrapper that can adapt to multiple architectures and their variations. As for the Lin et al. project, it is selected because it presents two features based on temporal reasoning and because they are built on the same architecture, it facilitates the creation of a wrapper that can span both architectures.

Following Rodriguez et al. [41], the video encoding procedure is done offline, performing rescaling to 320x240 when necessary, and sampling at 25 fps. Our implementation is based on *decord*[1], *pyslowfast*[2] and the official code release for TSM [28] [3]. Each video is sequentially traversed in buckets of a size given by the frame rate of the backbone model. In cases where the total number of extracted frames is not divisible by this number, we fill the last bucket by repeating the last frame.

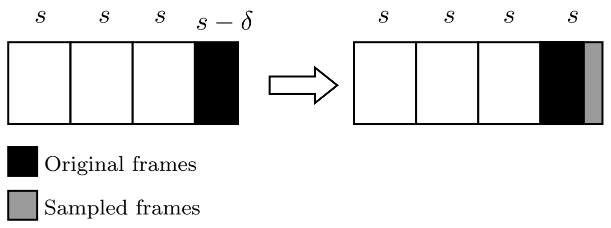


Figure 3.2: Preprocessing applied to videos with a frame count that is not a multiple of the target model's framerate.

Once the frames are loaded into memory, each video is sequentially traversed in buckets of the length of the frame rate, denoted as s, which is accepted by the different models. This procedure, employing the video encoder F_V, generates l/s buckets from which the features are extracted. However, an issue that may arise with videos is the lack of multiplicity with the frame rate s utilized by the pretrained networks from which features are extracted. To address this edge case, we propose extending the original video by filling the missing δ frames with a repetition of the last frame δ times, thus achieving the desired extension of s required by the utilized models.

Due to the potentially prolonged duration of a video, loading each frame into RAM could be highly resource-intensive, even after preprocessing the input video. An illustrative example is found in the case of __, the longest video in the YouCookII dataset, consisting of approximately ___ frames. Loading this video into RAM would surpass 120 GB, and even with sufficient memory, a substantial portion of the data would remain unused, adversely affecting overall code performance.

To address this challenge during video loading, a batch loading approach is proposed. Taking into account a video's framerate s, symmetric batches are generated to load into memory only the necessary $s \times N$ frames for a total of N features. This mitigates the strain on memory resources and enhances the overall efficiency of the code.

As previously mentioned, for the feature extraction, we utilized the architectures proposed in the works of Fan et al. [7] and Lin et al. [29]. Our approach involves leveraging the backbone of these architectures for action detection, and modifying the classification head

[1]`https://github.com/dmlc/decord`
[2]`https://github.com/facebookresearch/SlowFast`
[3]`https://github.com/mit-han-lab/temporal-shift-module#pretrained-models`

of each architecture to generate features that capture temporal information rather than performing classification.

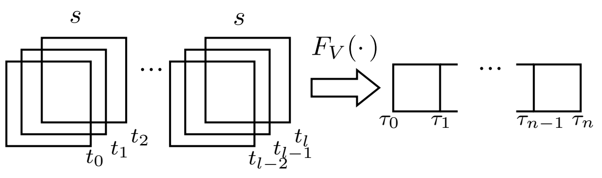


Figure 3.3: Temporal transformation obtained by applying an encoder on a video.

As illustrated in Figure 3.3, encoding a video introduces a change in temporal representation due to the processing in buckets of size s. Therefore, it is necessary to establish a mapping that enables the identification of which features correspond to natural language queries present in the datasets.

$$\tau = (t \cdot n \cdot fps)/l \tag{3.1}$$

We employed the mapping 3.1 to convert the frame/feature index to time. Specifically, an annotation starting at t_s and ending at t_e is converted to τ_s and τ_e, representing specific positions within the features such that $\tau_s, \tau_e \in [1, \dots, n]$.

While we have released the extracted features for benchmarking purposes, we also provide the extraction framework for readers to facilitate feature extraction for different datasets not included in this work.

3.3 Temporal Video Grounding

At the core of our experimental framework lies the TVG model, serving as a central element for evaluating video features. While various approaches exist for our task, not all are suitable, given that substituting video representations can often result in significant alterations to the model architecture and/or training dynamics. Following a thorough examination, we opted for the proposal-free approach introduced by Rodriguez et al. [41] (TMLGA) and the regression-based proposal-free method proposed by Jonghwan et al. [35] (LGVI). These selected methodologies align well with our experimental objectives and constraints, allowing us to test different features without applying drastic changes to the original networks.

TMLGA can be summarized in three parts: 1) a text encoder based on GloVe [37] embeddings on top of an LSTM [20], 2) a video encoder, which we discuss in detail below, and, 3) a localization module that combines information from both modalities using a dynamic filter [21] and predicts the start and end points of the segment. We consider this model to be a "*classic*" approach, as it has been widely adopted as a baseline by recent work, allowing for meaningful comparisons with a wide variety of approaches. We also note that TMLGA relies on a rather simple approach to query representation, which we believe allows for a cleaner result in terms of the contribution of the language encoder. Finally, we point out that TMLGA has an official implementation that has been publicly released, which we believe is essential to facilitate the replication of the original results and ultimately ensure the reproducibility of our experiments.

Much like TMLGA, LGVI introduces a nuanced approach to training temporal video grounding models. In this method, the authors introduce a function f_e designed to generate semantic phrases, acting as cues to identify relevant moments:

$$\arg\max_{\theta} \mathbb{E}[\log_{p_\theta}(C|V,Q)] \rightarrow \arg\max_{\theta} \mathbb{E}[\log_{p_\theta}(C|V,f_e(Q))]$$

To articulate the network specification, it is organized into four integral components. The initial segment pertains to the text and video encoders, while the second focuses on the query attention network, wherein the system establishes auxiliary constructs to enhance the identification of pertinent moments. The third component revolves around the local-global video-text interaction, and the fourth entails the temporal attention-based regression.

It's noteworthy that this method, distinct from TMLGA, employs a regression approach in the output. Specifically, it generates predicted timestamps for both the start and end, deviating from a distributional output that signifies the most probable prediction. We opt for LGVI due to its training simplicity and to demonstrate that potential behaviors observed in experiments with TMLGA are not exclusive to a specific architecture. The goal is to show that similar results can be achieved using a different model structure.

Regarding the utilization of architectures, as previously mentioned, our intention is to make modifications that do not alter the essence of the models. Since our objective revolves around comparing features, only the modules related to the use of video features in the employed models are adjusted. It is crucial to note that, in this study, the text features are not manipulated in any of the architectures. Therefore, the text features remain frozen throughout both training and inference, utilizing the representations generated in their respective original works.

3.3.1 Annotation Modification

From the preceding step, it is apparent that the size of each generated feature is contingent upon the sampling rate s and the architecture employed in the backbone. Variability in vector representations introduces a challenge related to annotations, where features extracted from different architectures create differences in the interpretation of temporal reference, thus marking each feature uniquely. Consequently, prior to training with any temporal localization network, it becomes imperative to modify the original annotations and establish coherent references for each dataset with respect to the features to be utilized.

This process is iteratively applied to each dataset, leveraging metadata from the videos such as duration, frames, and timestamps of annotation start and end. Subsequently, Equation 3.1 is employed to map the new annotation reference in the dataset. Ultimately, a comprehensive dataset is obtained, capturing a unified temporal context for further analysis.

3.4 Temporal Video Grounding Experiments

After training the TVG models with the diverse features obtained through our framework, the subsequent section proposes the execution of various experiments to examine the impact of these features on the specific task at hand. These experiments are primarily designed to scrutinize differences in performance, bias, and explore potential relationships between text and video embeddings.

Note that the experiment starts by replicating the original results of the TMLGA and LGVI networks. This serves two purposes: firstly, to compare the results obtained with the original results from the papers, and secondly, to set the hyperparameters of the network with a baseline configuration. Based on this configuration, the 13 features obtained for different data sets are then trained. It's worth noting that if replicating the original results proves difficult, efforts are made to achieve comparable values.

3.4.1 Features Performance Benchmark

In order to perform a comprehensive feature comparison, we have developed an extrinsic evaluation across the three proposed datasets. Our primary objective is to visualise the performance differences between the different features and highlight the variations in predictions based on the datasets and features employed. This experiment is expected to serve as a guide to how encoder characteristics can influence prediction generation, providing insight into the application of features in temporal tasks. In addition, the observed variation in predictions with different features underscores the importance of features and highlights the need for coherent feature extraction that is aligned with the temporal localisation model used.

The evaluation covers each of the proposed datasets and uses the TMLGA and LGVI architectures. For comparison, the tIoU metric is used, taking into account the best prediction made by the networks and incorporating IoU thresholds of 0.3, 0.5 and 0.7 (commonly used thresholds in the literature). In addition, the mIoU calculation is included to illustrate the general performance trends of the predictions.

3.4.2 Bias associated with predictions

To delve into potential differences arising from the use of different video classifiers in the task of temporal localization, we propose examining how predictions vary across various backbones. The objective is to observe distinct trends for each employed backbone, hypothesizing that certain backbones may perform better on specific video segments or types of datasets.

To facilitate this exploration, we propose creating distribution graphs similar to those presented in https://arxiv.org/pdf/2101.09028.pdf. Through these visualizations, we aim to illustrate how the normalized concentrations of the start (X-axis) and end (Y-axis) points of predictions made by the TMLGA model with different backbones vary across each dataset.

The intention is to highlight how backbone variation alone can lead to diverse predictions in the task of temporal localization.

To assess the correctness of predictions, various threshold levels of the interval over union metric will be employed. Similar to the performance benchmark, thresholds of 0.3, 0.5, and 0.7 will be used. This approach allows us to observe how different feature concentrations align with varying stringencies in the intersection of predictions with the ground truth, providing nuanced insights into the classifier's performance.

3.4.3 Orthogonality of predictions

Finally, we explore the orthogonality of features. The objective here is to visualize whether predictions made with different features are mutually exclusive or if they represent a subset of another feature derived from a more complex backbone. In essence, we aim to determine if one feature can accurately predict a relevant moment in a video that another feature fails to identify, or if the predictions made with one set of features are encompassed within the predictions of another feature with a similar architecture.

Similar to previous processes, various prediction thresholds are examined to define a correct video prediction. However, in contrast to prior cases, the analysis of thresholds involves comparing each prediction made by each feature, grouping them based on their associated nature. Subsequently, we assess whether the predictions made for each grouping are mutually exclusive or intersect. This approach generates Venn diagrams to identify congruencies among the features.

3.4.4 Relationship between predictions and annotations

Due to the utilization of diverse datasets for both training and evaluation, TVG models are tasked with identifying moments characterized by varying temporal aspects and spatial information. Consequently, an intriguing analysis involves exploring the relationship between created features and the detection of actions or moments associated with specific categories. For example, it would be interesting to examine scenarios where a CNN might excel in identifying relevant moments related to sports activities, while other features may enhance the detection of significant moments associated with cooking.

To execute this task, the primary action described in each annotation will be identified. Once the predominant action is determined, embeddings are generated to vectorially represent the video annotations. These embeddings are then projected onto a two-dimensional graph, enabling the visualization of potential groupings. Each representation is plotted in a two-dimensional graph, associating each point with the features that accurately classified the corresponding actions. The aim is to discern whether certain features demonstrate superior predictive capabilities for a set of actions compared to others.

For the purposes of this study, the analysis will commence with smaller datasets, starting with Charades.

Chapter 4

Results and Analysis

This chapter delves into an extensive comparative study conducted to assess various features in the task of temporal grounding, a crucial undertaking for enhancing video comprehension and retrieval efficiency in numerous practical applications.

The primary aim of this benchmark study is to systematically evaluate and compare the performance of different features in temporal grounding tasks. Through this, we strive to gain a clear understanding of the current state of the art in this field and identify potential improvements achievable by experimenting with features beyond those traditionally used.

Employing a robust and diverse dataset, our methodology encompassed rigorous testing, where we examined the impact of TVG models with different features using metrics, embeddings, bias analyses, and Venn diagrams. This approach was designed to offer a holistic view of the performance landscape in temporal video grounding tasks.

Our findings highlight performance improvements achieved solely by altering features, shedding light on the strengths and limitations of current methodologies. These insights are pivotal in guiding future research and enhancing practical applications in video analysis.

In this chapter, we present these findings in detail, starting with an exhaustive analysis of the comparative test results, followed by discussions on their implications and potential avenues for future research. Please note that the various results presented in this chapter were validated through publication at an ICCV 2023 workshop [4].

4.1 Features Performance Benchmark

The first experiment we conducted involved comparing the performance of Temporal Video Grounding models using different features. For this purpose, we tested two architectures: a classic architecture named TMLGA and a second architecture called LGI, which incorporates language guides to enhance predictions. The aim of this section is to determine, through these two architectures, whether varying features impacts the performance in temporal localization tasks. This could be a crucial factor for future research, as comparing and utilizing the most effective features might significantly improve the outcomes achieved.

Model			**Charades**		**ActivityNet**		**YouCookII**	
Encoder	f	Pretrain	0.5	0.7	0.5	0.7	0.5	0.7
MViT	8	K400	**50.27**	31.32	30.64	17.50	**27.78**	**15.21**
MViT-v2	16	K400	48.17	30.27	23.80	11.87	25.40	13.95
Rev-MViT	16	K400	49.89	**32.31**	28.94	15.97	23.71	13.00
X3D M	16	K400	43.60	27.96	26.39	14.14	22.97	12.80
X3D S	13	K400	41.37	25.16	30.23	17.54	23.00	12.60
SlowFast	8	K400	38.82	24.22	<u>23.34</u>	<u>11.70</u>	21.39	11.57
SlowFast	16	SS V.2	39.76	23.90	30.15	17.51	21.16	11.05
SlowOnly	8	K400	36.16	20.19	23.92	12.11	11.31	6.24
I3D NLN	8	K400	38.63	24.33	24.04	12.28	20.76	11.60
C2D	8	K400	<u>02.63</u>	<u>00.97</u>	24.33	12.39	<u>08.85</u>	<u>03.92</u>
TSM	8	K400	45.35	23.20	**31.81**	**18.06**	24.74	13.29
Mean	-	-	39.51	23.98	27.37	14.90	21.01	11.38
STD	-	-	13.41	06.65	07.04	05.46	06.76	03.37
I3D [42]	8	Charades	52.02	33.74	-	-	-	-
I3D [42]	8	K400	-	-	33.04	19.26	20.65	10.94

Table 4.1: Summary of our experimental results with TMLGA, where f denotes the frame rate used to extract features, while K400 and SS V.2 stand for the Kinetics-400 and Something-Something V.2 datasets, respectively. In the table, the best results for each dataset are indicated in **bold**, while the worst results are <u>underlined</u>.

Table 4.1 presents a summary of our experimental results, highlighting the performance variations of the TMLGA model with different feature sets. In the Charades-STA dataset, Transformer-based features demonstrate promising outcomes, notably outperforming our I3D features. This contrasts with the original results where I3D features surpassed ours by only 2.45 points. We attribute this discrepancy to the pre-training dataset: the original features, pre-trained on Charades, may have caused overfitting. Conversely, our features pre-trained on the K400 dataset show competitive performance, allowing for an unbiased selection and reducing the need for extensive training on task-specific datasets.

In the ActivityNet dataset, two key observations emerge. First, certain CNN-based features unexpectedly outperform newer Transformer-based models, with up to a 7-point difference in the 0.7 tIoU threshold. Second, the performance variance among features is smaller, suggesting a higher relevance of query representation. This aligns with data showing that ActivityNet queries are more complex in terms of vocabulary size and length compared to other datasets.

On the YouCookII dataset, the high visual complexity is evident, likely due to the longer average video length of about 5 minutes. Here, Transformer-based models excel, paralleling the performance of sophisticated architectures like DORi, which leverage spatial information.

These results underscore the critical role of feature selection in TVG tasks, demonstrating that strategic changes can yield performance improvements of up to 5 points. This highlights the risk of stagnation and suboptimal results when features are selected without fully understanding their impact on the dataset. Future research should focus on exploring the effects of diverse features and their broader applicability in video and speech tasks.

Model			**Charades**				**ActivityNet**			
Encoder	f	Pretrain	0.5-iid	0.7-iid	0.5-ood	0.7-ood	0.5-iid	0.7-idd	0.5-ood	0.7-ood
MViT	8	K400	27.34	11.91	18.99	0.07	45.33	27.51	23.04	10.62
MViT-v2	16	K400	**58.32**	32.69	45.10	21.93	44.30	27.28	22.15	9.95
Rev-MViT	16	K400	49.21	26.37	38.73	18.10	43.00	25.54	22.95	10.35
X3D M	16	K400	52.73	29.65	43.35	20.09	44.07	27.75	22.58	10.28
X3D S	13	K400	49.21	27.83	40.00	19.13	45.17	28.10	23.22	10.48
SlowFast	8	K400	54.68	**32.71**	46.55	**23.91**	43.48	27.87	22.01	9.97
SlowFast	16	SS V.2	55.16	31.83	**45.90**	23.56	43.91	25.38	21.79	9,97
SlowOnly	8	K400	48.36	27.83	39.61	18.61	43.75	26.09	22.56	10.31
I3D NLN	8	K400	52.49	30.98	43.59	21.84	44.19	27.16	22.86	10.66
C2D	8	K400	34.02	17.13	27.35	11.47	42.81	26.25	21.32	9.41
TSM	8	K400	45.39	24.54	35.08	13.57	42.69	26.85	22.74	10.36
Mean	-	-	47.95	26.68	42.29	38.57	43.88	26.85	22.74	10.36
STD	-	-	9.35	6.66	6.79	8.56	0.87	0.95	0.59	0.36

Table 4.2: Summary of our experimental results with LGI, where f denotes the frame rate used to extract features, while K400 and SS V.2 stand for the Kinetics-400 and Something-Something V.2 datasets, respectively. In the table, the best results for each dataset are indicated in **bold**, while the worst results are underlined.

In the case of the LGVI network 4.2, we observe a distinct grouping of results. Here, "iid" denotes the testing set with independent and identically distributed data compared to the training set, while "ood" refers to the out-of-distribution data relative to the training dataset. Simply put, the "iid" set represents videos with content similar to that of the training material, whereas the "ood" set includes content that significantly differs from the training set, for instance, if the training set contained only adults running, the "ood" set might feature children running.

Regarding the results, Table 4.2 indicates that for Charades, the network's performance varies with different features, similar to the patterns observed with TMLGA. The network achieves varying levels of performance, with differences of up to 5 points between the best and worst cases in the "iid" segment. For the "ood" sample, changing features also impacts predictions, suggesting an improvement in the generalization of the prediction problem.

An intriguing scenario arises with ActivityNet Captions, where no substantial improvement in performance is noted across all features. Furthermore, the examination of standard deviations and means suggests that the observed differences could be due to randomness in training. This lack of differentiation between features could imply that the network is not effectively utilizing video feature information for decision-making and may be influenced by biases in the training annotations.

To test whether the model's predictions are inherently linked to the input features, a testing replica of the RevMViT model was created using random features based on the dimensions of the original features. This experiment aimed to determine whether the network could make similar decisions with random features. The results are as follows:

Table 4.3: The results obtained in testing when applying a random feature to the LGVI model, using the network trained with RevMViT as a reference.

	0.5 iid	0.7 iid	0.5 ood	0.7 ood
Random Feature	38.86	21.52	17.24	6.38

Table 4.4: The results obtained in testing when applying a random feature to the TMLGA model, using the network trained with RevMViT as a reference.

	0.5	0.7
Random Feature	8.62	4.28

The results indicate minimal deterioration in the LGVI models despite the use of completely random features. This suggests that the network has a limited capacity to assimilate information from video features and tends to rely on predictions derived from annotations. Therefore, the minimal variation in predictive capability observed among different features could be attributed to the network's inability to effectively process the information contained in the features and its reliance on shortcuts to predict outcomes that maximize the metric.

Conversely, in the case of TMLGA, we do not observe the same behavior, as there is a significant deterioration in the model when random features are introduced. This finding suggests that the problem lies with the LGVI network during training, where it fails to effectively absorb information from the features.

In summary, it is generally observed that features have a direct impact on TVG networks, creating significant differences between one feature and another. However, training issues in TVG networks are also evident, where the networks are not addressing a multimodal problem but are instead heavily biased towards the annotations they encountered during training. This point is crucial, as it suggests the importance of verifying TVG networks with the randomness of features to ensure they are truly learning from them.

4.2 Bias associated with predictions

This section examines in detail how biases manifest themselves in TMLGA and LGVI models, affecting their ability to accurately interpret and respond to dynamic video content. As TVG algorithms increasingly rely on complex datasets and advanced neural networks, the tendency of these systems to inherit and enhance biases present in their training data emerges as a crucial challenge. Understanding and mitigating these biases is vital to ensure effective and equitable learning in TVG models. Thus, we can confirm that the network is not only assimilating biases from the data, but also learning differentially with variation in the features applied, which is a fundamental step towards developing more robust and fair artificial intelligence systems.

To better understand how different features influence our model's output, we visualized the 0-1 normalized predicted time intervals for each query, drawing inspiration from Otani et al. [34]. For this, we generated two sets of plots: one for correctly predicted intervals and another for incorrect predictions, both evaluated at a 0.7 tIoU threshold. In the figures 4.1, 4.2, 4.3 and 4.4, the colors represent the concentration of predictions across different start and end intervals. Herein, a darker color indicates a higher accumulation of predictions, while a lighter color signifies a lower concentration.

Starting with the analysis of the results obtained for the TMLGA network, in Figure 4.1, our findings on the Charades-STA dataset reveal two distinct clusters of correct predictions, located in the lower-left and upper-right sections of the plot. A notable observation is the model's propensity to predict intervals spanning the entire length of the video, a pattern not reflected in the training data. Additionally, the variation in prediction patterns across different features, evident in distinct accumulation zones, underscores the significant disparity in positive and negative predictions among models, suggesting an orthogonality in feature behavior.

In contrast, on the ActivityNet dataset, the model consistently defaults to degenerate solutions across all feature sets, often predicting intervals covering the full video duration. This could be attributed to the presence of similar patterns in the training data. However, similar to Charades-STA, we observe different accumulation points in the model's predictions, again indicating potential orthogonality between features.

Regarding the YouCookII dataset, similar to the other two datasets, different concentrations of predictions for each feature are observed. This analysis also reveals that YouCookII exhibits a unique behavior, with predictions clustered along the diagonal of the graph. An interesting aspect of this plot is that the more dispersed the concentrations are from the diagonal, the higher the error rate and lower the performance of the features. This is evident when comparing models such as MViT and I3D, where MViT shows less dispersion and thus potentially better performance than I3D. This observation underscores the importance of feature selection in model performance, particularly in the context of diverse datasets like YouCookII.

To further enhance the analysis, plots of the normalized duration distributions of the predictions for each feature are added 4.2. It is observed that for Charades, there is no single pattern in either the correct or incorrect predictions across the different features; in fact, each feature exhibits its own unique distribution. On the other hand, the situation changes with Activity-Net Captions. While variations are observed, particularly in the 0 - 0.5 range, a common trend across all features is the high concentration of errors in generating predictions throughout the same video. This pattern suggests that the network tends to predict the length of the video when unsure of the answer. This could be because the Temporal Intersection over Union (tIoU) metric still yields high values despite the error, potentially caused by the prevalent presence of 0-1 annotations in the dataset. Finally, with YouCook, while all distributions appear similar, this is likely due to the annotations of Activity-Net Captions falling within that range. However, this should not be considered a problem similar to what is observed with Activity-Net Captions. In the case of YouCook, the networks at least attempt to generate timestamps that are coherent with the problem and span the duration of the video. This nuanced understanding of the networks' behavior across different datasets highlights the complexity of temporal video grounding and the need for tailored approaches to different types of video content.

These observations highlight the complex interplay between feature selection and model performance. The varying accumulation points suggest distinct predictive behaviors, offering insights into the dataset-specific challenges and the potential for enhanced predictions through feature optimization.

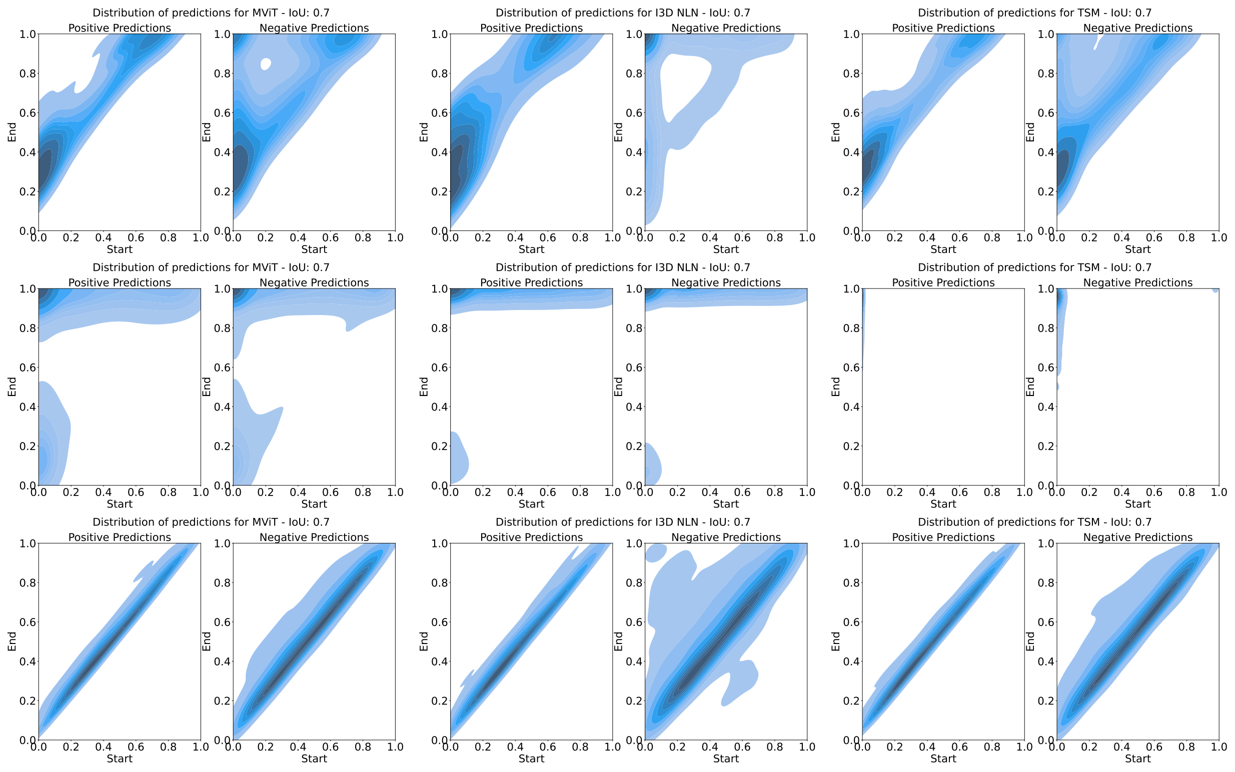


Figure 4.1: Normalized temporal prediction distributions across three datasets for TMLGA. Each row shows data for the Charades, ActivityNet, and YouCook II datasets (from top to bottom) comparing feature extraction methods: MViT, I3D, and TSM (from left to right). The x-axis denotes the start time of the prediction, and the y-axis denotes the end time, illustrating the spread and concentration of predictions within the datasets.

As observed in the previous experiment, there is a qualitatively low variation in predictions

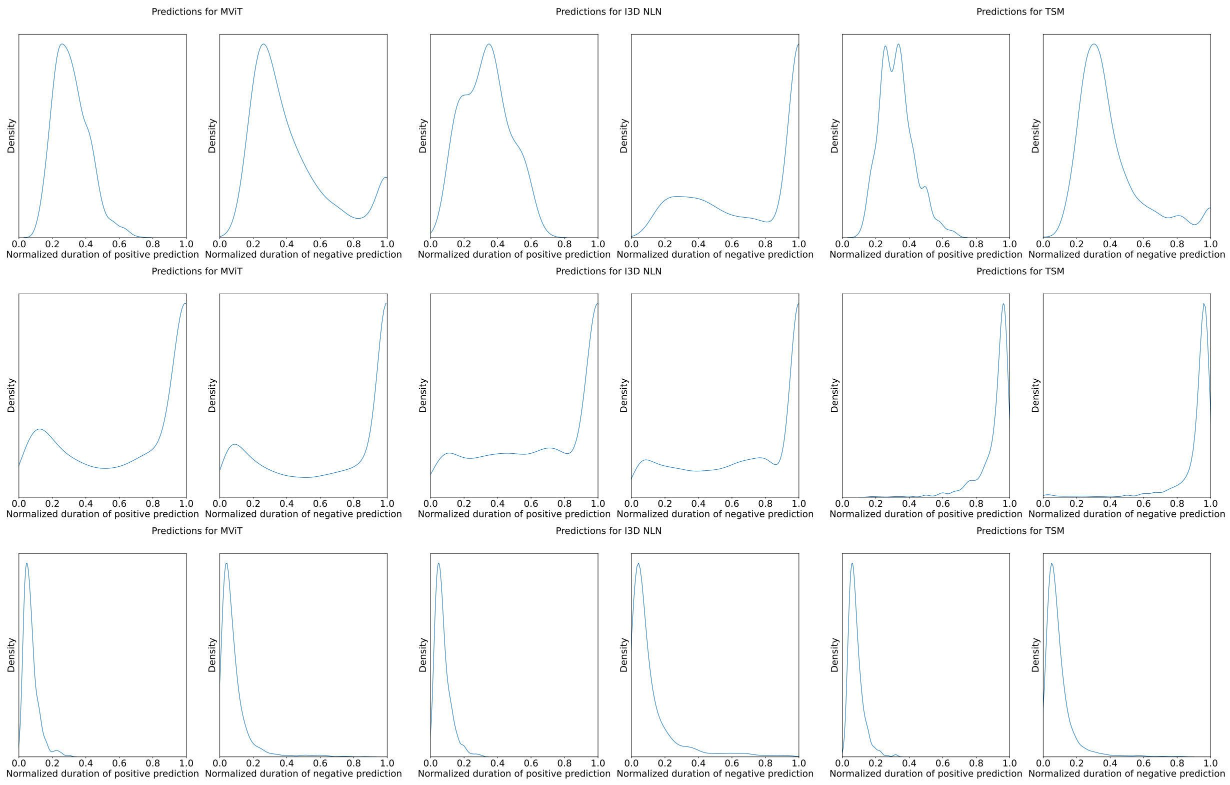


Figure 4.2: Comparison of normalized temporal prediction distributions across datasets for TMLGA. Each row visualizes the predictions for the Charades, ActivityNet, and YouCook II datasets (top to bottom), comparing the feature extraction methods MViT, I3D NLN, and TSM (left to right). The x-axis indicates the prediction density, and the y-axis marks the prediction end time, showcasing the variance in prediction patterns among the methods and datasets.

regardless of the features used. This reaffirms the outcomes obtained with the LGVI network in the plot 4.3, suggesting that regardless of the choice of features, there is minimal to no variation in the network's ability to assimilate information from the features, as evidenced by the similarity in the outputs. This reinforces the observation that the LGVI network, particularly for the Activity-Net dataset, appears to generate predictions based primarily on data biases, thereby overlooking the resolution of the intended multimodal task. This pattern indicates a potential limitation in the network's design or training process, where it fails to effectively leverage the diverse information that different features might offer, instead defaulting to pattern recognition based on training biases.

Finally, examining the distribution graphs for both the iid and ood testing sets of LGI 4.4, we notice distinct patterns. Unlike what was observed with TMLGA, Activity-Net exhibits a different behavior where the network does not saturate predictions at the length of the video when it is uncertain about the location of the relevant moment. Nevertheless, the curves, similar to the previous distributions, appear alike, indicating that the networks tend to yield similar predictions despite changes in features. This is in stark contrast to TMLGA, where greater variations were evident. This observation suggests a certain level of consistency in the predictive mechanisms of the networks across various features, highlighting a potential area of improvement in terms of feature sensitivity.

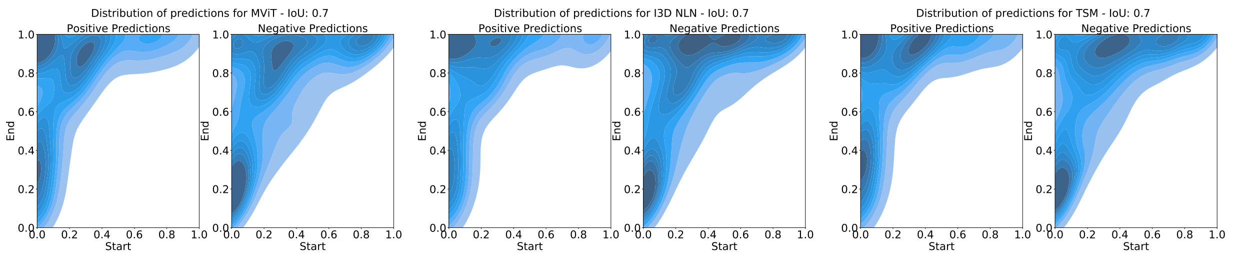


Figure 4.3: Distribution of normalized temporal predictions across the YouCook II dataset for LGVI, comparing feature extraction methods. From left to right: MViT, I3D NLN, and TSM. The x-axis shows the start time of predictions, and the y-axis shows the end time, illustrating how predictions are distributed temporally.

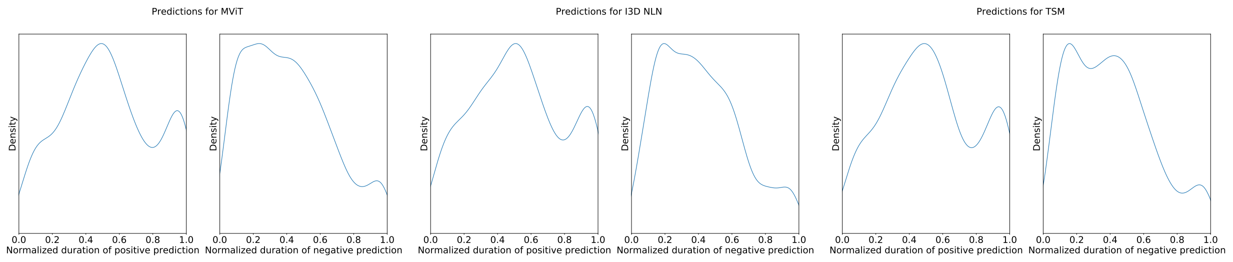


Figure 4.4: Normalized temporal prediction density for the YouCook II dataset for LGVI. The distribution compares the effectiveness of MViT, I3D NLN, and TSM (left to right) in predicting the end times of events, with the x-axis indicating prediction density and the y-axis indicating prediction end time.

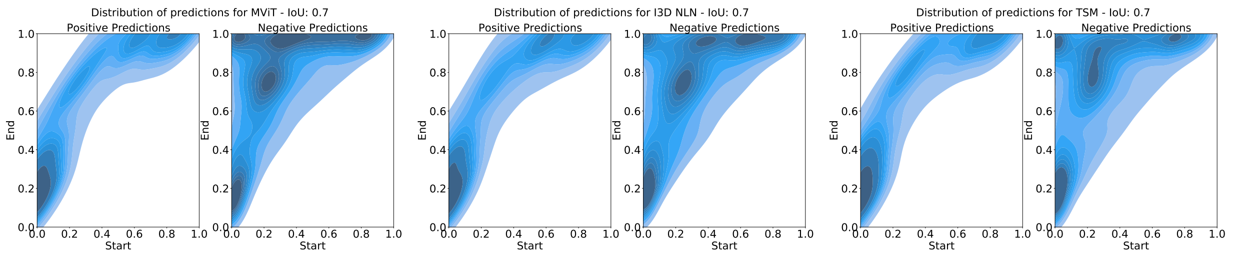


Figure 4.5: Distribution of normalized temporal predictions for out-of-distribution (OOD) samples in the YouCook II dataset for LGVI, using features from MViT, I3D NLN, and TSM. The x-axis shows prediction start time, while the y-axis shows the end time, highlighting the temporal prediction spread.

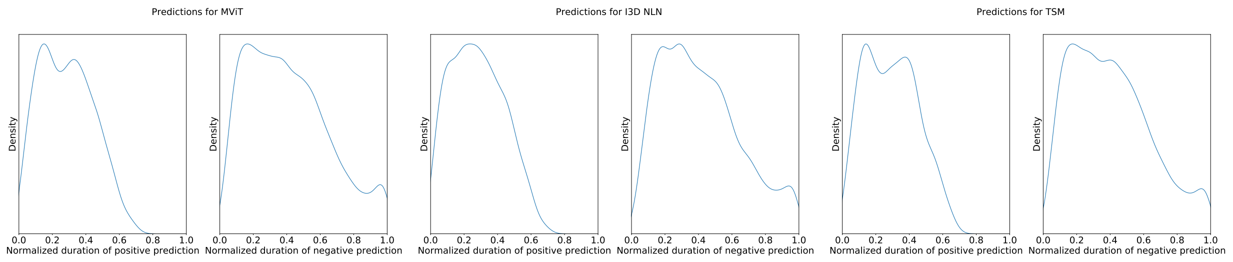


Figure 4.6: Normalized prediction density for out-of-distribution (OOD) events in the YouCook II dataset for LGVI. Comparing methods: MViT, I3D NLN, and TSM, the x-axis measures the density of predictions, and the y-axis measures the prediction end time.

4.3 Orthogonality of predictions

4.3.1 Venn diagrams on predictions

To demonstrate that the predictions are complementary, Venn diagrams of the predictions made by different features are plotted. Initially, features are categorized based on their nature: Temporal, CNN, and Transformer. After grouping the predictions, the correct ones made at different temporal interval over union thresholds are separated, allowing us to assess the predictive accuracy of each extracted feature.

To fully grasp these graphs, it's important to understand the following: each color represents a set of predictions. For video features of the transformer type, blue is used; for CNN features, green; and for temporal features, red. Additionally, the intersections between these different groups create new colors. Specifically, the intersection of all three feature subsets produces gray, indicating that all types of features are capable of making the specified predictions. This color-coding system is essential for interpreting the graphs, as it visually distinguishes between the predictive capabilities of different feature types and their combinations.

The first cases examined are the Venn diagrams for the TMLGA network. From these diagrams, it's observed that for each threshold, there are groups of videos that can only be predicted by specific types of features. This indicates that the features may capture different abstractions of information from the videos, leading to varied predictions in the final temporal model.

A closer look at the TMLGA results reveals that as the temporal threshold becomes more stringent, there is a smaller set of commonly predicted videos across the datasets (indicated by the grey segment in the Venn diagram). Moreover, it's noted that CNN features can predict the most videos independently, with temporal features predicting the least. This phenomenon might be due to the quantity and diversity of features available, with CNNs having the most.

For the LGVI network, results are similar to those observed with TMLGA in the independent and identically distributed (iid) sample. It's seen that the more stringent the threshold, the fewer the number of videos intersecting in the predictions. Additionally, the small difference in predictions for different thresholds in the ActivityNet dataset compared to TMLGA is notable, indicating a lower variance within the iid data analyzed. This could be attributed to the limited effectiveness of the features for this dataset with the LGVI network and the low variability in the iid data.

In the case of the out-of-distribution (ood) sample, greater differences among the features are observed, with a significant number of videos predicted only by CNN features. This observation aligns with the previously noted behavior where a higher threshold results in fewer intersecting predictions among different features.

In summary, the Venn diagrams for TMLGA and LGVI networks highlight two key points: (1) They underscore the significance of feature selection in localization tasks, emphasizing the importance of choosing the right features for optimal performance. (2) Secondly, they suggest the potential for a complementary use of features, which could lead to the development of more robust models in temporal localization tasks.

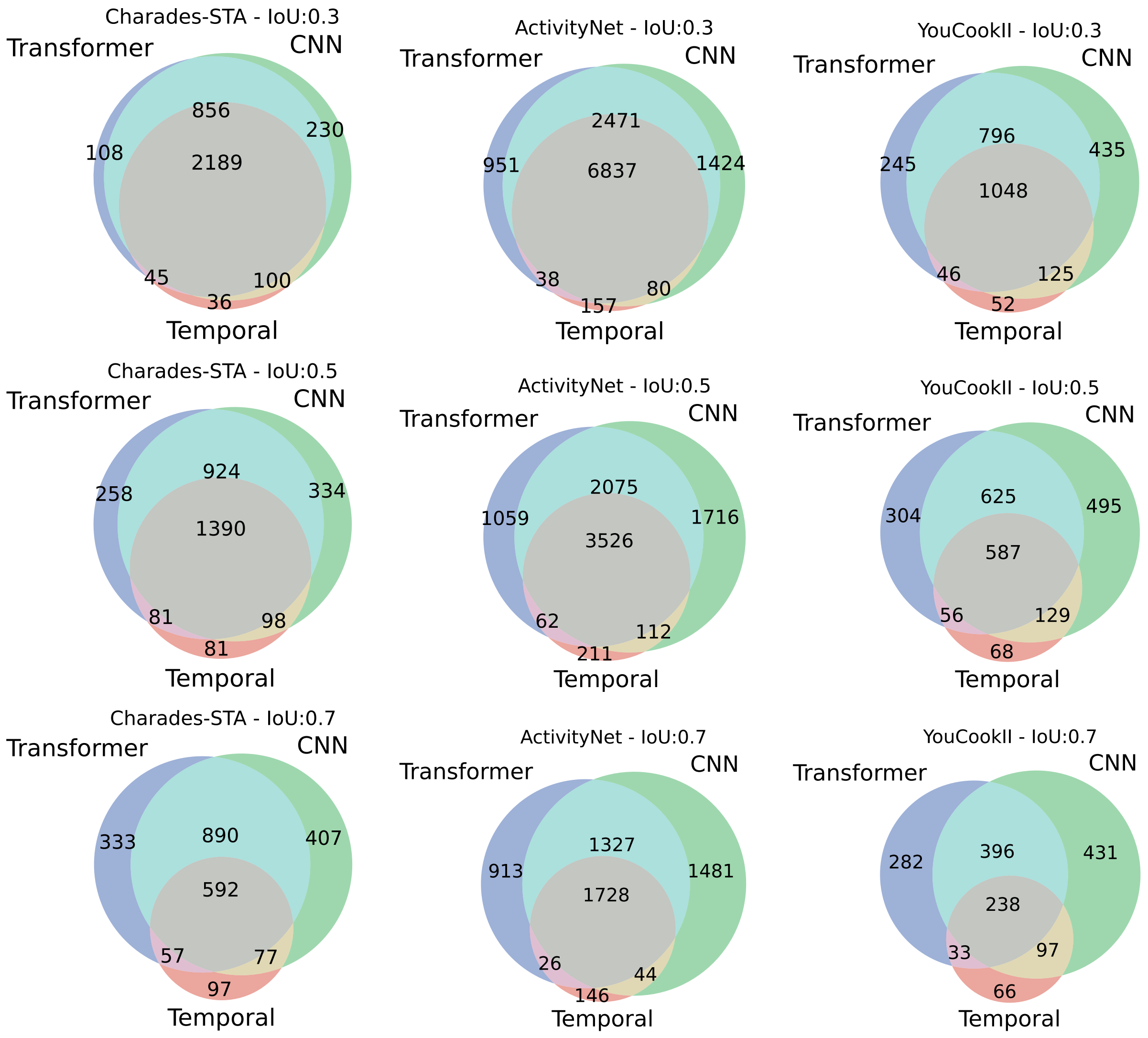


Figure 4.7: Overlap of the correct predictions, in terms of query-video pairs for $\alpha = 0.7$, in Charades-STA (left), ActivityNet (center) and YoucookII (right) benchmarks. We group features based on their nature.

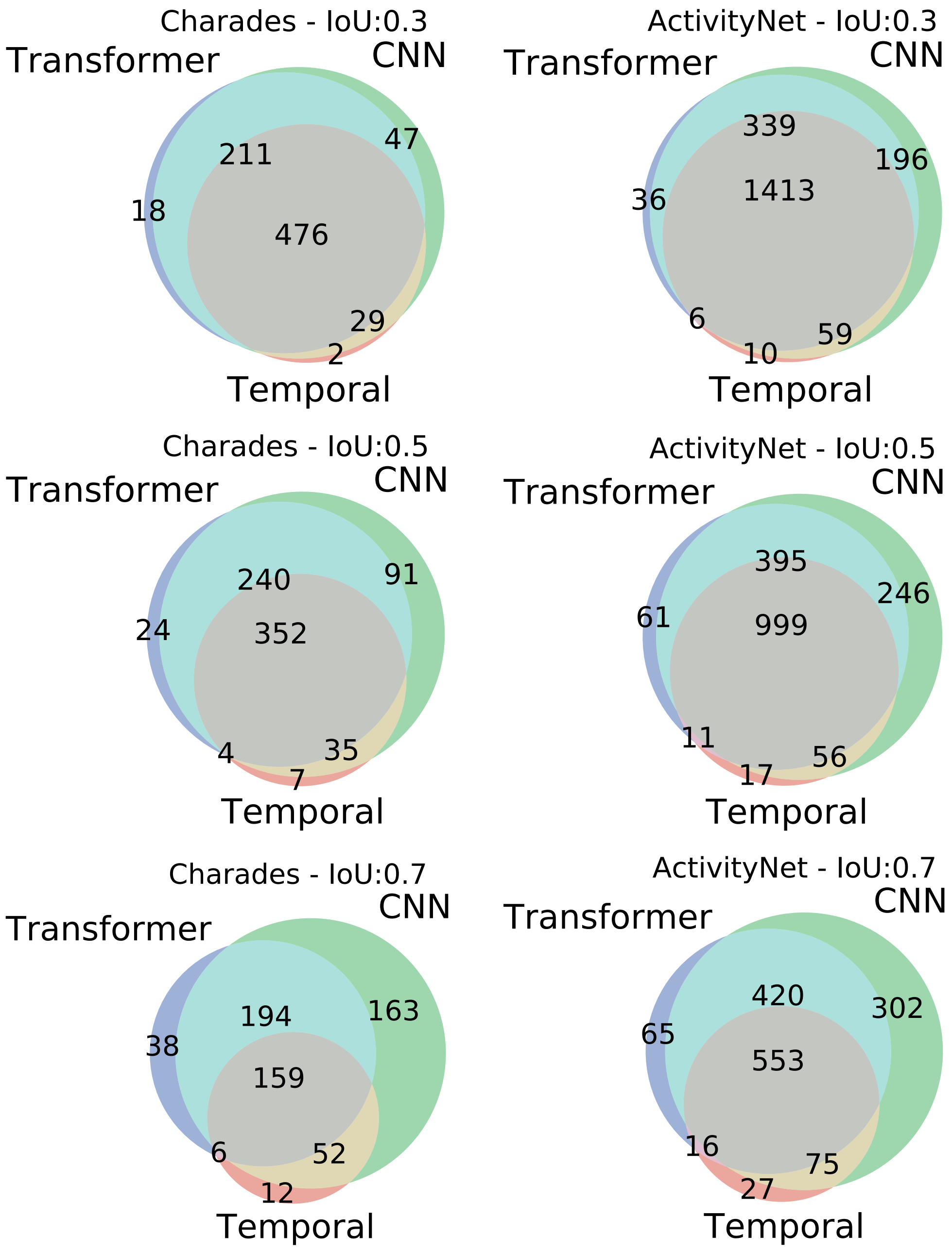


Figure 4.8: Overlap of the correct predictions, in terms of query-video pairs for $\alpha = 0.7$, in Charades-STA (left), ActivityNet (center) and YoucookII (right) benchmarks. We group features based on their nature.

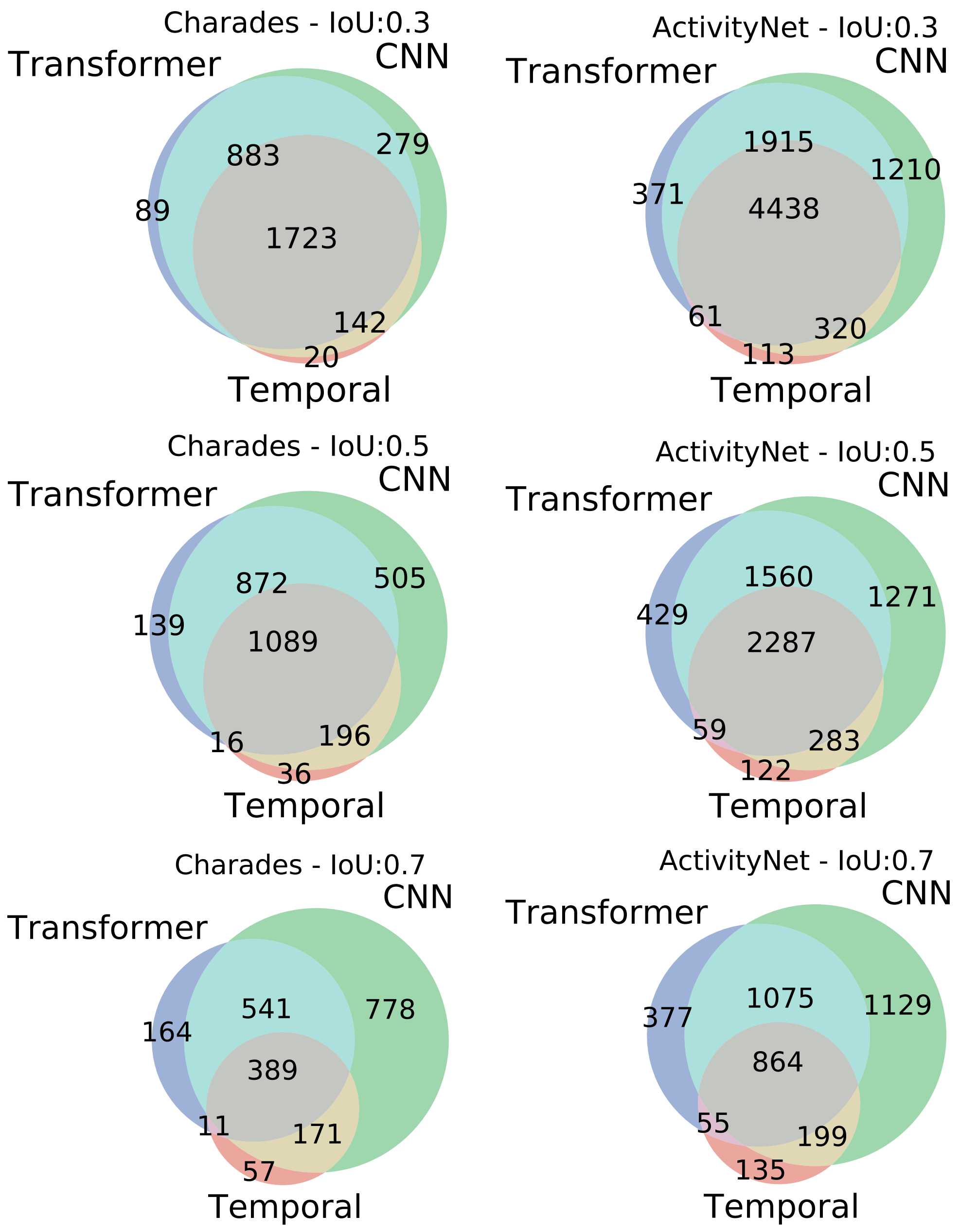


Figure 4.9: Overlap of the correct predictions, in terms of query-video pairs for $\alpha = 0.7$, in Charades-STA (left), ActivityNet (center) and YoucookII (right) benchmarks. We group features based on their nature.

4.3.2 Post-training Metamodel

To supplement and delve deeper into the potential completeness seen in the different features visualized in Venn diagrams; this subsection proposes selecting post-training models and generating an oracle, where we retain the best prediction made for a specific video among all the predictions made by the different features.

The goal of this task is to demonstrate that if we can identify the feature that best classifies a given video, we can achieve superior results compared to those currently obtained by TVG networks. Thus, our proposal does not seek to create a model, but rather a test with which we suggest a new direction for the task of temporal localization, where the focus should be on combining features to achieve better performance.

Table 4.5: Enhanced Performances Achieved by Merging Models with Various Features, Utilizing the Highest IoU Scores from the Training Set of TMLGA Models. The findings suggest that if we can adeptly select the appropriate features for prediction, the efficiency of TVG models can increase significantly (see 4.1).

Dataset	tIoU - 0.5	tIoU - 0.7	mIoU
Charades	84.46	65.40	74.63
ActivityNet	50.05	32.37	51.22
YouCookII	60.86	41.48	58.93

Table 4.6: Performance results from merging models with different features using the highest IoU scores achieved in the test set by LGVI models for both IID and OOD datasets. The results show significant improvements on both datasets compared to 4.2.

Dataset	Data	tIoU - 0.5	tIoU - 0.7	mIoU
Charades	iid	91.49	75.82	77.01
ActivityNet	iid	70.36	57.47	65.87
Charades	ood	84.53	62.55	70.66
ActivityNet	ood	49.7	31.7	50.36

From the results in 4.5 and , it is observable that for both TMLGA and LGI, improvements are seen in all thresholds and average performances that the networks possess. This indicates that each of the models trained with different features provides different information to the models, resulting in varied performances on different videos. A potential reason for this is the nature of the features, which encode different spatio-temporal information depending on the video being analyzed. This suggests that more modern models like transformers, while achieving better performance, do not capture all the information contained in videos that older networks might capture.

Finally, while the results are encouraging to declare that there is a potential complementarity among features, the question arises: is this completeness of the features a random factor in machine learning models? To answer this question, in section 4.3.3 we will explore the variation in the combinations of features and tests with different seeds in the TMLGA model.

4.3.3 Analyzing the Impact of Feature Combination and Randomness in Temporal Video Grounding Models

After presenting the potential for completeness in using multiple features when training a temporal video grounding model, the next step is to verify whether the construction of the proposed oracle could be due to randomness generated by the training itself. This will be tested through multiple seeds to see if this behavior persists or disappears when varying the random factor in the models.

The experiment conducted involves training models using the TMLGA network with four different features, each trained with four distinct seeds, resulting in a total of 16 models for experimentation. The features chosen are those with the highest performance in the Charades dataset: MViT, Rev MViT, MViT v2, and X3D M. Due to computational constraints and execution time, the experiment is only carried out for the TMLGA network and uses the Charades dataset, as it is the smallest dataset used.

The first task involves creating meta-models by combining each of the models with the same features but different training seeds. Our goal is to demonstrate that the randomness of the training does not contribute significantly, as seen when combining features of different natures. The following results were obtained:

Table 4.7: Results from Combining TMLGA Models with Identical Features but Four Distinct Seeds. The table shows improvements in outcomes compared to those achieved without combining models with different seeds, and a similar performance increase is noted across all four features.

Model	tIoU - 0.5	tIoU - 0.7	mtIoU
MViT	66.45	44.92	58.65
Rev-MViT	64.54	44.35	58.41
MViT-v2	64.87	45.46	58.93
X3D M	62.93	43.06	58.18
Mean	64.70	44.45	58.54
Std	1.44	1.03	0.32

From the table, it can be observed that combining models with the same features results in a slight improvement in performance compared to the results shown in 4.5, showing a similar contribution for the four combined features.

The second experiment consists of combining models with different features but the same seed. The aim is to verify that the improvement is greater than that seen by combining models trained with the same features, suggesting that the information in the features is different and can create a complementary behavior between them for final decision-making. The results of this experiment are as follows:

The results show a considerable improvement when combining models with different features. There is an average increase of 10 points by combining the same number of models,

Table 4.8: Enhanced Outcomes from Integrating Diverse Features: Mvit, X3D M, ReV-MViT, and MViT-v2 with Varied Seeds from 2012, 2015, 2017, and 2022. The observed performance metrics exceed the results in Table 4.7, indicating a more comprehensive and effective combination of features.

Seed	tIoU - 0.5	tIoU - 0.7	mIoU
2012	78.41	56.75	68.59
2015	78.23	55.99	68.64
2017	79.22	58.36	69.03
2022	78.49	57.58	68.73
Mean	78.58	57.17	68.75
Std	0.44	1,03	0,20

and it is observed that despite different seeds, the selections are stable, indicating that the increase is not related to a random factor that could have occurred in a particular training of the networks.

The outcomes of this experiment intrinsically demonstrate that there is a complementary contribution between the features. Although we have not been able to construct an automatic selector capable of effectively selecting features, it is evident that the behavior indicated by the oracle is not merely due to randomness present in deep learning models.

4.4 Relationship between predictions and annotations

In order to see if there's a tendency for certain groups of queries/annotations to be predicted more accurately in videos with different features, we propose to examine how predictions from the Charades meta-model cluster with TMLGA and determine if certain verbs are more effectively predicted by features of a particular type than others. It is important to note that the use of the metamodel is primarily because it identifies the most effective features for different examples in the test

To perform this experiment, we first identify the features and queries. Given the complexity of the content of a query to make comparisons, we simplify the text component to the verbs present in the queries associated with the videos. Specifically, to extract the verbs, we use SentenceTransformer, applying part-of-speech tagging to the queries to identify the predominant verb in the queries made to the videos. From the results, we obtain the following distribution for the 10 most frequent verbs in the queries:

From the graph 4.10 we can see that out of a total of 87 verbs identified with part-of-speech tagging in the test set, the majority of queries are concentrated around the actions: "open", "put", "continue" and "play", showing a significant difference compared to other verbs. This indicates a significant imbalance in the dataset, which could affect the performance of videos with actions that occur only once.

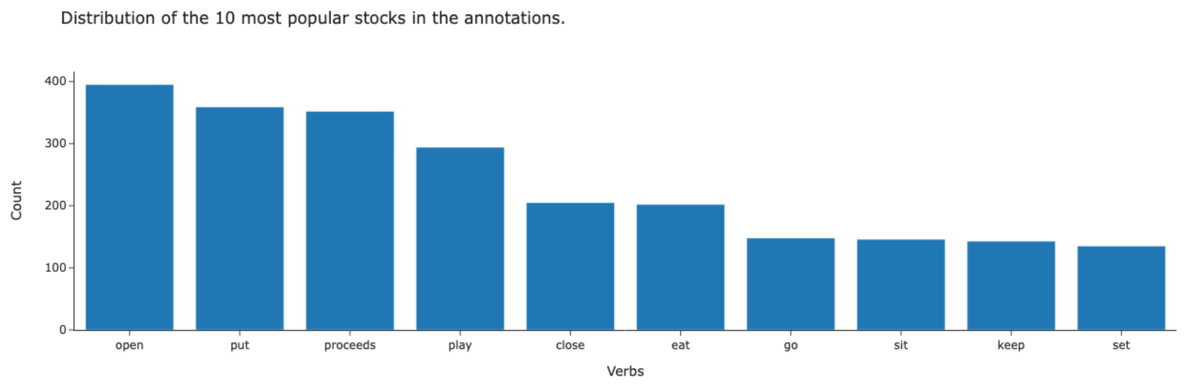


Figure 4.10: Distribution of the Top 10 Most Frequently Occurring Verbs in the Queries from the Charades Testing Dataset.

Having identified the verbs and the embeddings representing the queries, we project the queries into a two-dimensional space using T-sne. Our goal is to see if it's possible to observe clusters that group the verbs according to the embeddings of the most similar sentences, thereby identifying possible trends in the predictions made by the features (for example, we hope to see that some clusters have a predominance of one feature over another). A second point to consider in this experiment is that we only use predictions from the Charades metamodel of TMLGA that achieve performances in tIoU above 0.9, as this more demanding threshold should be able to highlight the true trends present in the features and queries.

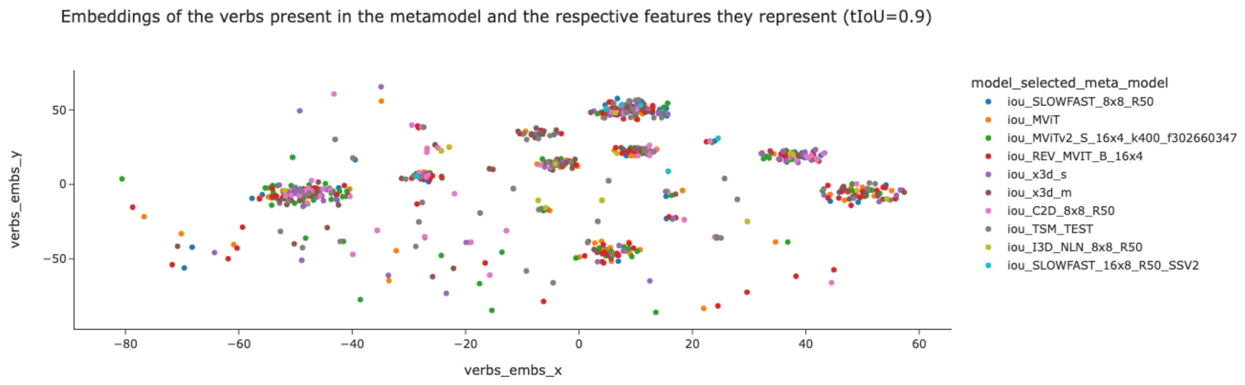


Figure 4.11: Projection of video queries from the Charades testing set with an IoU Exceeding 0.9. In the plot, each point represents a specific query, and the colors associated with each point indicate the features with the best performance for that particular case.

Looking at the graph 4.11, we can see that there are about 9 large clusters of queries within the projection. A closer look at each of these clusters reveals that they are all composed of the same verbs. A notable aspect of the graph is the presence of different colours on the dots, representing each of the features that achieved the best results on certain videos. This shows that there is no clear trend in the prediction of clusters, indicating random behaviour in the data cloud. In order to reduce the noise in the information projected by the features, a second graph 4.12 is presented where each feature is categorised according to its type: CNN, Temporal or Transformer. This approach aims to simplify the visual content and assess whether a trend can be more clearly identified.

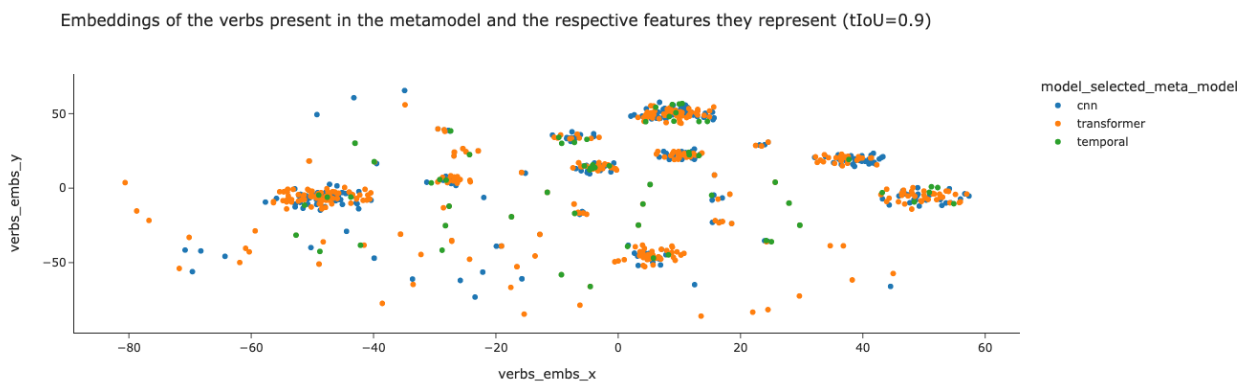


Figure 4.12: Projection of Video Queries from the Charades Testing Set with an IoU Exceeding 0.9. In the graph, each point symbolizes a distinct query, and the colors assigned to each point represent the nature of the features with the best performance for that specific case.

In 4.12, it is now possible to see just three types of colors that represent the features by their nature. Although this significantly reduces visual clutter, the pattern of randomness seen in 4.11 persists, making it impossible to discern any clear trend in the predictions made. Further analysis involves zooming in on the cluster that contains the verb 'play' (see image 4.13). Within this cluster, no discernible trend is observed in the predominance of one type of feature over another, merely highlighting the random behavior of the features.

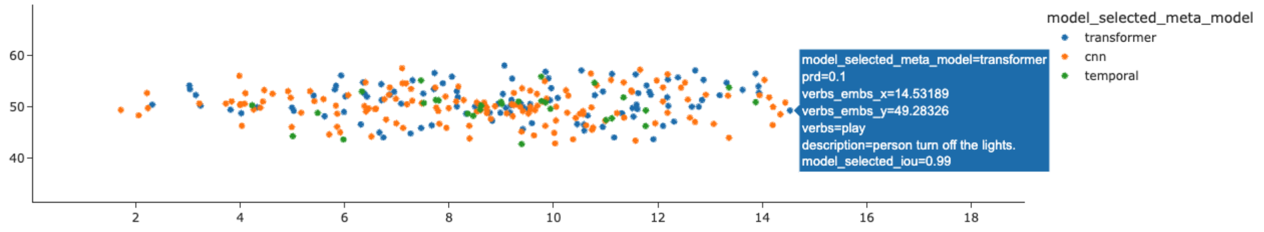


Figure 4.13: The image depicts a zoomed-in view of one of the clusters formed by the queries. Despite grouping by the nature of the features, it is impossible to identify a type of feature that predominates over others in this cluster, highlighting the absence of a discernible pattern associated with the predictions in this cluster.

In summary, from the projections and comparisons of video features with their respective queries, it is not possible to establish a clear trend between them, with only random patterns observed in the meta-model's predictions. This lack of a discernible pattern suggests that the difference in features is dictated by more complex behaviors than just accurately predicting a subset of actions in the dataset.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In this thesis, we have embarked on a comprehensive analysis of video features in the context of the video grounding task, leading to the development of a novel framework for video feature extraction. Our research has illuminated that using features with pre-trained encoders from datasets different from the target application can produce results comparable to those using finely-tuned, application-specific encoders. This approach effectively counters potential network overfitting and biases related to task proficiency.

The development of our video feature extraction framework serves as a practical application, offering a robust tool for researchers and professionals in the field of temporal localization. This work aims to provide the scientific community with new ideas and tools to enhance the performance of TVG models and reduce the reliance on dominant and potentially biased features in their selection.

However, while our framework allows for easy extraction of video features, its architecture is not easily compatible with developments outside of the Meta team and requires constant updating to incorporate new models. This means that maintenance is needed; otherwise, some of the backbones will become outdated compared to the state of the art.

Regarding our findings, we observed a substantial performance increase by altering video representations, significantly improving model performance even without modifying the underlying architecture. Our research also identified specific weaknesses in certain classical features, while discovering synergies between different features. This aspect of our research suggests potential pathways for optimization and performance improvement in temporal localization networks.

On the other hand, when comparing features with different TVG models, we encountered problems during network training. For instance, in the ActivityNet dataset, the LGVI network tends to learn annotation characteristics to optimize performance, neglecting the visual information provided through video features. This is concerning as it indicates a lack of rigor in the TVG task that could lead to models blind to the problem's multimodality due to the current approach and potential dataset inadequacies.

Another relevant aspect of our study is the revelation of unique prediction patterns among video features, suggesting that integrating these diverse characteristics into a unified network could lead to more accurate predictions. The variation of features has shown to have a profound impact on localization, leading to improvements comparable to those of more complex networks that consider spatial components.

Finally, this work provides an exhaustive demonstration of the complementary power of features. Using a meta-model and Venn diagrams, it is shown that different models trained with different features can generate exclusive predictions. This indicates that if we can automatically identify when to use certain features based on the input video, it suggests significant leaps in performance in temporal localization networks.

Regarding the limitations reflected in the study of feature completeness, the study is limited to the extrinsic verification of the contribution of features through the temporal localization task, due to the complexity of intrinsically comparing the information represented in the videos.

Looking to the future, applying these discoveries to modern architectural models opens exciting opportunities for advancement in temporal localization. The inherent orthogonality of the features we have explored is expected to promote the development of sophisticated and customized solutions in this dynamic field. With this thesis, we have not only shed light on the current state of video feature analysis but also laid the groundwork for future advances in this area.

5.2 Future Work

The exploratory and analytical work presented in this thesis opens several avenues for future research. Building upon the findings and methodologies employed, we propose the following directions for advancing the field of Temporal Video Grounding (TVG) and related areas:

Testing Features Across Diverse Architectures

- Future studies should focus on extending the current research by incorporating more modern and varied architectural frameworks. This will enable a comprehensive understanding of how different architectures influence feature performance in TVG models.
- A systematic assessment of feature performance across a spectrum of architectures, including both established and emerging models, is crucial. This approach will shed light on the adaptability and efficiency of features in diverse environments, providing insights into optimal model design.

Exploring Text and Video Encoder Variations in TVG

- An in-depth investigation into the impact of varying text and video encoders on TVG task performance is warranted. This research should aim to understand how encoder variations affect model accuracy and efficiency.
- A comparative analysis of different encoder architectures and their influence on TVG outcomes will provide valuable guidelines for selecting appropriate encoders for specific applications.

Enhancing TVG Models through Feature Combination

- Building on the foundation of this thesis, further research should delve into combining multiple features to boost TVG model generalization. This entails identifying and testing synergistic feature combinations that enhance overall model robustness.
- The focus should be on exploring combinations that not only improve accuracy but also contribute to the scalability and practical applicability of TVG models in real-world scenarios.

Bibliography

[1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-End Object Detection with Transformers. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, Lecture Notes in Computer Science, pages 213–229, Cham, 2020. Springer International Publishing.

[2] Joao Carreira and Andrew Zisserman. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[3] Shaoxiang Chen and Yu-Gang Jiang. Semantic proposal for activity localizaiton in videos via sentence query. *AAAI*, 2019.

[4] Ignacio M. De la Jara, Cristian Rodriguez-Opazo, Edison Marrese-Taylor, and Felipe Bravo-Marquez. An empirical study of the effect of video encoders on temporal video grounding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 2850–2855, October 2023.

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*, March 2022.

[6] Omar Elharrouss, Younes Akbari, Noor Almaadeed, and Somaya Al-Maadeed. Backbones-review: Feature extraction networks for deep learning and deep reinforcement learning approaches, 2022.

[7] Haoqi Fan, Yanghao Li, Bo Xiong, Wan-Yen Lo, and Christoph Feichtenhofer. Pyslowfast. `https://github.com/facebookresearch/slowfast`, 2020.

[8] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale Vision Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6824–6835, 2021.

[9] Christoph Feichtenhofer. X3d: Expanding architectures for efficient video recognition, 2020.

[10] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. SlowFast Networks for Video Recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6202–6211, 2019.

[11] Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. Tall: Temporal activity localization via language query. In *ICCV*, 2017.

[12] Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. Tall: Temporal activity localization via language query, 2017.

[13] Runzhou Ge, Jiyang Gao, Kan Chen, and Ram Nevatia. Mac: Mining activity concepts for language-based temporal localization. In *WACV*, 2019.

[14] Soham Ghosh, Anuva Agarwal, Zarana Parekh, and Alexander Hauptmann. ExCL: Extractive Clip Localization Using Natural Language Descriptions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[15] Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. Video Action Transformer Network. In *CVPR*, pages 244–253, 2019.

[16] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* MIT Press, Cambridge, MA, USA, 2016. `http://www.deeplearningbook.org`.

[17] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017.

[18] Chunhui Gu, Chen Sun, David A. Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul Sukthankar, Cordelia Schmid, and Jitendra Malik. AVA: A Video Dataset of Spatio-Temporally Localized Atomic Visual Actions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6047–6056, 2018.

[19] Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. Localizing moments in video with natural language. In *ICCV*, 2017.

[20] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[21] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc V Gool. Dynamic filter networks. *Advances in neural information processing systems*, 29, 2016.

[22] Ioannis Karatzas. and Steven E. Shreve. *Brownian Motion and Stochastic Calculus.* Springer, Berlin, 2nd edition, 2000.

[23] J.D. Kelleher. *Deep Learning.* MIT Press essential knowledge series. MIT Press, 2019.

[24] Erica Kido Shimomoto, Edison Marrese-Taylor, Hiroya Takamura, Ichiro Kobayashi, Hideki Nakayama, and Yusuke Miyao. Towards parameter-efficient integration of pre-trained language models in temporal video grounding. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13101–13123, Toronto, Canada, July 2023. Association for Computational Linguistics.

[25] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-Captioning Events in Videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 706–715, 2017.

[26] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations. *International Journal of Computer Vision*, 123(1):32–73, May 2017.

[27] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. MViTv2: Improved Multiscale Vision Transformers for Classification and Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4804–4814, 2022.

[28] Ji Lin, Chuang Gan, and Song Han. TSM: Temporal Shift Module for Efficient Video Understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7083–7093, 2019.

[29] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[30] Kevin Qinghong Lin, Pengchuan Zhang, Joya Chen, Shraman Pramanick, Difei Gao, Alex Jinpeng Wang, Rui Yan, and Mike Zheng Shou. Univtg: Towards unified video-language temporal grounding, 2023.

[31] Daizong Liu, Xiaoye Qu, Jianfeng Dong, and Pan Zhou. Adaptive Proposal Generation Network for Temporal Sentence Localization in Videos. *arXiv:2109.06398 [cs]*, September 2021.

[32] Meng Liu, Xiang Wang, Liqiang Nie, Xiangnan He, Baoquan Chen, and Tat-Seng Chua. Attentive moment retrieval in videos. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 15–24. ACM, 2018.

[33] Karttikeya Mangalam, Haoqi Fan, Yanghao Li, Chao-Yuan Wu, Bo Xiong, Christoph Feichtenhofer, and Jitendra Malik. Reversible vision transformers, 2023.

[34] Esa Rahtu Mayu Otani, Yuta Nakahima and Janne Heikkilä. Uncovering Hidden Challenges in Query-Based Video Moment Retrieval. In *The British Machine Vision Conference (BMVC)*, 2020.

[35] Jonghwan Mun, Minsu Cho, and Bohyung Han. Local-global video-text interactions for temporal grounding, 2020.

[36] Jonghwan Mun, Minsu Cho, and Bohyung Han. Local-Global Video-Text Interactions for Temporal Grounding. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10807–10816, Seattle, WA, USA, June 2020. IEEE.

[37] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[38] Philip Protter. *Stochastic Integration and Differential Equations*. Springer, 1990.

[39] Daniel Revuz and Marc Yor. *Continuous martingales and Brownian motion*. Number 293 in Grundlehren der mathematischen Wissenschaften. Springer, Berlin [u.a.], 3. ed edition, 1999.

[40] Cristian Rodriguez, Edison Marrese-Taylor, Basura Fernando, Hiroya Takamura, and Qi Wu. Memory-efficient temporal moment localization in long videos. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1901–1916, 2023.

[41] Cristian Rodriguez, Edison Marrese-Taylor, Fatemeh Sadat Saleh, Hongdong Li, and Stephen Gould. Proposal-free Temporal Moment Localization of a Natural-Language Query in Video using Guided Attention. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 2464–2473, 2020.

[42] Cristian Rodriguez-Opazo, Edison Marrese-Taylor, Basura Fernando, Hongdong Li, and Stephen Gould. DORi: Discovering Object Relationships for Moment Localization of a Natural Language Query in a Video. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1079–1088, 2021.

[43] Cristian Rodriguez-Opazo, Edison Marrese-Taylor, Basura Fernando, Hiroya Takamura, and Qi Wu. Locformer: Enabling transformers to perform temporal moment localization on long untrimmed videos with a feature sampling approach. *arXiv preprint arXiv:2112.10066*, 2021.

[44] Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[45] Gunnar A. Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in Homes: Crowdsourcing Data Collection for Activity Understanding. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science, pages 510–526, Cham, 2016. Springer International Publishing.

[46] Gunnar A. Sigurdsson, Gül Varol, Xiaolong Wang, Ivan Laptev, Ali Farhadi, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. *ArXiv e-prints*, 2016.

[47] Haoyu Tang, Jihua Zhu, Meng Liu, Zan Gao, and Zhiyong Cheng. Frame-wise cross-modal matching for video moment retrieval. *IEEE Transactions on Multimedia*, 24:1338–1349, 2022.

[48] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning Spatiotemporal Features With 3D Convolutional Networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4489–4497, 2015.

[49] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, \Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[51] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal Segment Networks: Towards Good Practices for Deep Action Recognition. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science, pages 20–36, Cham, 2016. Springer International Publishing.

[52] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks for action recognition in videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(11):2740–2755, 2019.

[53] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks, 2018.

[54] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[55] Zhenzhi Wang, Limin Wang, Tao Wu, Tianhao Li, and Gangshan Wu. Negative Sample Matters: A Renaissance of Metric Learning for Temporal Grounding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(3):2613–2623, June 2022.

[56] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking Spatiotemporal Feature Learning: Speed-Accuracy Trade-offs in Video Classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 305–321, 2018.

[57] Huijuan Xu, Kun He, L Sigal, S Sclaroff, and K Saenko. Multilevel language and vision integration for text-to-clip retrieval. In *AAAI*, 2019.

[58] Runhao Zeng, Haoming Xu, Wenbing Huang, Peihao Chen, Mingkui Tan, and Chuang Gan. Dense regression network for video grounding, 2020.

[59] Hao Zhang, Aixin Sun, Wei Jing, and Joey Tianyi Zhou. Span-based localizing network for natural language video localization. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6543–6554, Online, July 2020. Association for Computational Linguistics.

[60] Lingyu Zhang and Richard J. Radke. Natural language video moment localization through query-controlled temporal convolution. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2524–2532, 2022.

[61] Mingxing Zhang, Yang Yang, Xinghan Chen, Yanli Ji, Xing Xu, Jingjing Li, and Heng Tao Shen. Multi-Stage Aggregated Transformer Network for Temporal Language Localization in Videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12669–12678, 2021.

[62] Songyang Zhang, Houwen Peng, Jianlong Fu, Yijuan Lu, and Jiebo Luo. Multi-Scale 2D Temporal Adjacency Networks for Moment Localization with Natural Language. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2021.

[63] Minghang Zheng, Yanjie Huang, Qingchao Chen, Yuxin Peng, and Yang Liu. Weakly Supervised Temporal Sentence Grounding With Gaussian-Based Contrastive Proposal Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15555–15564, 2022.

[64] Luowei Zhou, Nathan Louis, and Jason J. Corso. Weakly-Supervised Video Object Grounding from Text by Loss Weighting and Object Interaction. *arXiv:1805.02834 [cs]*, May 2018.

[65] Luowei Zhou, Chenliang Xu, and Jason J. Corso. Towards Automatic Learning of Procedures From Web Instructional Videos. In *Thirty-Second AAAI Conference on Artificial Intelligence*, April 2018.