

Design and Evaluation of Lightweight Architectures for Single Sentence Video Captioning

An M. Tech Project Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of
Master of Technology

by
KULDEEP
(Roll No. 234156024)

under the guidance of
Prof. Parijat Bhowmick



to the
Centre for Intelligent Cyber Physical System,
INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI
GUWAHATI - 781039, ASSAM

CERTIFICATE

This is to certify that the work contained in this thesis entitled "**Design and Evaluation of Lightweight Architectures for Single Sentence Video Captioning**" is a bonafide work of **Kuldeep** (Roll No. **234156024**), carried out in the **Centre for Intelligent Cyber Physical System, Indian Institute of Technology Guwahati**, under my supervision, and that it has not been submitted elsewhere for a degree.

Supervisor: Prof. Parijat Bhowmick
Department of EEE
IIT Guwahati, Assam

Date: _____
Place: **IIT Guwahati**

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my supervisor, **Prof. Parijat Bhowmick**, for his insightful guidance, constant encouragement, and valuable feedback throughout the course of this project on video captioning. His expertise and mentorship were instrumental in shaping the direction and quality of this work.

I am also thankful to my friends and batchmates for their collaborative support and technical discussions, which helped me overcome many challenges during the implementation and evaluation phases.

Finally, I express my heartfelt thanks to my family for their unwavering support, and to all the faculty members of IIT Guwahati for fostering a stimulating academic environment that enabled this research.

KULDEEP

June 2025

Date: _____
Place: IIT Guwahati

ABSTRACT

The objective of Single-Sentence Video Captioning (SSVC) is to generate a concise and coherent natural language description that effectively summarizes the visual content of a given video clip. While recent advancements in deep learning—particularly attention-based models and Transformer architectures—have significantly improved captioning quality, their high computational demands pose limitations for real-time applications and deployment on edge devices. This thesis addresses the challenge of building efficient yet accurate SSVC systems by exploring lightweight convolutional neural network (CNN) encoders and resource-aware recurrent decoders. The proposed framework aims to strike a balance between captioning performance and computational efficiency. Comprehensive experiments are conducted on two widely-used benchmark datasets, MSVD and MSR-VTT, to evaluate the effectiveness of various lightweight architectures. The results demonstrate that our approach achieves competitive captioning quality while substantially reducing model complexity, making it suitable for low-resource environments.

Contents

List of Figures	6
List of Tables	7
1 Introduction	8
1.1 What is Single Sentence Video Captioning (SSVC)	8
1.2 Challenges of SSVC	8
1.3 Applications Involving Lightweight Networks	9
1.4 Brief Literature Review	9
1.5 Overall Functional Block Diagram	10
1.6 Contributions	11
1.7 Thesis Organization	11
2 Literature Review	12
2.1 Video Captioning Datasets – An Overview	12
2.1.1 MSVD Dataset (Microsoft Research Video Description)	13
2.1.2 MSR-VTT Dataset (Microsoft Research Video to Text)	14
2.1.3 ActivityNet Captions	15
2.1.4 YouCookII Dataset	16
2.1.5 Charades Captions	17
2.1.6 LSMDC (Large-Scale Movie Description Challenge)	18
2.1.7 VATEX (Video and Text in English and Chinese)	19
2.1.8 Dataset Summary	20
2.2 Key Frame Selection Strategy	21
2.2.1 Uniform Sampling	21
2.2.2 Motion-Based Sampling	21
2.2.3 Entropy-Based Selection	22
2.2.4 Histogram-Based Selection	22
2.2.5 CNN-Based Scoring	23
2.2.6 Reinforcement Learning-Based Selection (PickNet)	24
2.2.7 Hybrid Methods	24
2.3 Normalization and Resizing of Video Frames	26
2.3.1 Frame Resizing	26
2.3.2 Normalization	26
2.3.3 Implementation Summary	26
2.4 Pretrained CNN Models for Visual Feature Extraction	28
2.4.1 ResNet Family	28
2.4.2 MobileNet Family	29
2.4.3 ShuffleNetV2 Family	29

2.4.4	Comparison Summary	30
2.5	Review of Traditional Approaches for Video Captioning	31
2.5.1	Template-Based Methods	31
2.5.2	Statistical Models	32
2.5.3	Comparison of Traditional Approaches	32
2.5.4	Relevance to Modern Techniques	32
2.6	Review of Deep Learning Approaches for Video Captioning	33
2.6.1	Attention-Based Methods for Video Captioning	33
2.6.2	Reinforcement Learning-Based Video Captioning	34
2.6.3	Adversarial Networks for Video Captioning	35
2.6.4	Video Question and Answering (V-QA)	36
2.7	Various Model Summary	37
2.7.1	Detailed Analysis of Models	37
2.8	Comparison of Methods Across Datasets	39
2.9	Caption Evaluation Metrics	40
2.9.1	BLEU	40
2.9.2	METEOR	40
2.9.3	ROUGE-L	41
2.9.4	CIDEr	41
2.9.5	Summary of Metrics	41
3	Proposed Architecture	43
3.1	Overview	43
3.2	Methodology	43
3.2.1	Video Preprocessing	43
3.2.2	Training the Model	45
3.3	Flowchart Of Proposed Pipeline	47
3.4	Experimental Setup	47
3.4.1	Model and Training Hyperparameters	48
3.4.2	Baseline Architectures	48
3.5	Discussion	49
4	Experiments and Results	50
4.1	Experimental Setup	50
4.1.1	Datasets	50
4.1.2	Evaluation Metrics	50
4.1.3	Training Configuration	51
4.1.4	Computational Environment	51
4.2	Ablation 1: Frame Selection Strategy	51
4.3	Ablation 2: Visual Feature , Motion Feature Extractor	53
4.4	Ablation 3: Decoder Type	55
4.5	Summary	56
5	Conclusion	58
5.1	Summary	58
5.2	Future Work	59

List of Figures

1.1	Overall flowchart of the video captioning pipeline	10
2.1	Sample frames from the MSVD dataset showing video content	14
2.2	Sample video frames from the MSR-VTT dataset	15
2.3	Sample video frames from the ActivityNet dataset	16
2.4	Sample cooking video frames from the YouCookII dataset	17
2.5	Sample activity video frames from the Charades dataset	18
2.6	Sample video frames from the LSMDC dataset	19
2.7	Sample instructional video frames from the VATEX dataset	20
3.1	Block diagram of the proposed lightweight video captioning pipeline. . .	47

List of Tables

2.1	Characteristics of Video Captioning Datasets	20
2.2	Comparison of Key Frame Selection Strategies	25
2.3	Comparison of CNN Models Based on GFLOPs and Parameters	31
2.4	Comparison of Traditional Video Captioning Models	33
2.5	Comparison of Attention-Based Methods	35
2.6	Summary of Various Video Captioning Models	37
2.7	Performance Comparison of Video Captioning Methods	39
2.8	Summary of Caption Evaluation Metrics	41
4.1	Effect of Frame Selection Strategies using MobileNetV2 Features with GRU Decoder on MSVD	51
4.2	Comparison of Frame Sampling Strategies using ResNet-152 Features with LSTM Decoder on MSVD	52
4.3	Effect of Frame Sampling Strategies using ResNet-152 Features with LSTM Decoder on MSR-VTT	52
4.4	Comparison of Model Architectures Based on Parameters and GFLOPs . .	53
4.5	Comparison of CNN Visual Feature Extractors on MSVD (LSTM Decoder, 1 FPS Sampling)	53
4.6	Comparison of CNN Visual Feature Extractors on MSR-VTT (LSTM Decoder, 1 FPS Sampling)	54
4.7	Comparison of Visual Extractors using GRU Decoder ($K = 30$ Uniformly Sampled Frames)	54
4.8	Comparison of CNN Visual Feature Extractors and Motion feature extractor MC3-18 on MSVD (LSTM Decoder, 1 FPS Sampling)	55
4.9	Comparison of Decoder Architectures using MobileNetV2 Features and 30 Uniform Frames (MSVD Dataset)	56

Chapter 1

Introduction

This chapter provides an overview of the single sentence video captioning, its challenges and applications.

1.1 What is Single Sentence Video Captioning (SSVC)

Single Sentence Video Captioning (SSVC) is the task of automatically generating a concise and meaningful natural language sentence that describes the overall content of a given video clip. Unlike multi-sentence narratives or full-length transcripts, SSVC focuses on summarizing the video in a single, grammatically correct, and semantically rich sentence that captures the most salient actions, objects, and context within the visual scene.

The primary goal of SSVC is to emulate how a human might describe a short video in a brief and intuitive summary. For example, a video showing a child playing with a dog might be captioned as: “*A child is playing fetch with a dog in the yard.*” This concise description serves as a high-level abstraction of the visual events and is particularly useful for applications in video retrieval, accessibility, and content recommendation systems.

1.2 Challenges of SSVC

Despite significant progress in the field of video captioning, SSVC remains a challenging task due to the inherent complexity of video data and the strict constraint of generating only a single descriptive sentence. The major challenges are outlined below:

1. **Visual Complexity:** Real-world videos often contain varying lighting conditions, multiple objects, background clutter, and scene transitions. Extracting consistent and meaningful visual features from such complex data is a non-trivial task.
2. **Temporal Modeling:** Capturing the temporal relationships between frames, especially when multiple events occur simultaneously or over a span of time, is essential for understanding the video holistically.
3. **Semantic Ambiguity:** Similar visual actions may have different semantic interpretations depending on subtle contextual cues. Disambiguating such actions requires a deep understanding of both visual and linguistic context.
4. **Information Bottleneck:** Condensing several seconds of rich visual content into a single sentence demands high-quality feature selection and efficient representation learning, which many models struggle to perform effectively.

5. **Data Alignment and Supervision:** Human-generated captions may not precisely align with visual events in the video, leading to noisy supervision during training and reduced captioning accuracy.

Addressing these challenges requires designing models that are not only capable of robust visual understanding and temporal reasoning, but also efficient enough to operate under computational constraints. This thesis aims to tackle these challenges by exploring lightweight yet effective architectures tailored for SSVC.

1.3 Applications Involving Lightweight Networks

The increasing demand for video captioning across real-world scenarios, particularly on platforms with limited computational resources, has led to a growing interest in lightweight neural networks. These models are designed to perform effectively while consuming minimal memory, computation, and energy, making them ideal for deployment on edge devices, mobile platforms, and embedded systems. Some key applications include:

- **Mobile Video Summarization:** Lightweight captioning models enable real-time summarization of videos on smartphones and tablets, allowing users to quickly preview or index media content without fully watching it.
- **Surveillance and Security Analytics:** In smart surveillance systems, edge-deployed models can generate concise captions for video streams from CCTV cameras, aiding in automated incident detection and activity logging.
- **Assistive Technologies for the Visually Impaired:** By converting visual content into descriptive audio using efficient captioning systems, visually impaired users can better understand their surroundings through wearable or handheld devices.
- **Automated Content Moderation:** Captioning systems can help flag inappropriate or sensitive content by analyzing the semantic meaning of videos, providing an additional layer of moderation in online platforms.
- **Video Retrieval and Indexing:** Lightweight models allow for efficient indexing and retrieval of video content based on automatically generated descriptions, facilitating faster search and recommendation systems in constrained environments.

These applications demonstrate the critical role of lightweight video captioning models in enabling practical, scalable, and accessible AI solutions across various domains.

1.4 Brief Literature Review

Early efforts in video captioning primarily relied on handcrafted visual features combined with rule-based or template-driven language generation systems. While these methods offered limited success, they lacked the adaptability and generalization needed for handling diverse and complex video content.

The emergence of deep learning significantly transformed the field. Sequence-to-Sequence (S2S) frameworks, particularly those employing Long Short-Term Memory

(LSTM) networks as decoders, became a standard approach. These models typically used convolutional neural networks (CNNs) such as VGG or ResNet as feature extractors to encode frame-level visual information, marking a shift towards end-to-end trainable architectures. The LSTM-CNN pipeline quickly became the baseline for video captioning tasks.

Building upon this, attention mechanisms were introduced to better model temporal dependencies. Models like S2VT-Attn extended the traditional S2VT framework by enabling the decoder to focus selectively on different parts of the video over time. Other enhancements included reinforcement learning approaches such as PickNet, which aimed to identify the most informative frames during training, and dual-stream networks that integrated both appearance and motion features to improve understanding of dynamic scenes.

In recent years, Transformer-based models, originally developed for natural language processing (NLP), have been adapted for video captioning. These models replaced recurrence with global self-attention, resulting in more effective long-range dependency modeling and parallelized training. While they achieved strong performance, they introduced substantial computational overhead.

To address deployment challenges in real-world and resource-limited environments, lightweight architectures like MobileNet, ShuffleNet, and their variants have been explored. These models offer a favorable trade-off between performance and efficiency, making them suitable for edge devices and embedded systems. However, many state-of-the-art approaches still remain computationally expensive, underscoring the need for efficient Single Sentence Video Captioning (SSVC) systems.

This thesis contributes to this ongoing research direction by systematically evaluating and leveraging lightweight CNN backbones to design an SSVC framework that balances captioning quality with computational efficiency.

1.5 Overall Functional Block Diagram

The overall pipeline of SSVC begins with an input video, which undergoes frame extraction followed by basic preprocessing to prepare the data for analysis. These preprocessed frames are then passed through a pretrained Convolutional Neural Network (CNN) to extract high-level visual features. The resulting feature vector serves as input to a recurrent decoder, implemented using either an LSTM or GRU network, which models temporal dependencies to facilitate caption generation. The generated caption is then evaluated using established metrics such as BLEU, CIDEr, METEOR, and ROUGE-L to assess its quality and relevance to the video content.

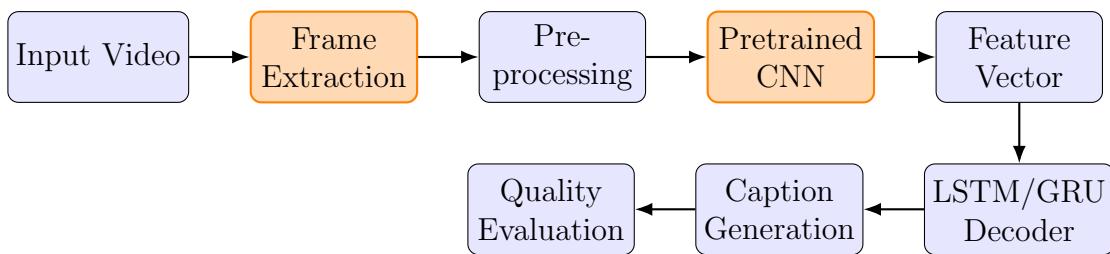


Figure 1.1: Overall flowchart of the video captioning pipeline

1.6 Contributions

This thesis makes several significant contributions toward developing an efficient and deployable Single Sentence Video Captioning (SSVC) framework. The primary focus lies in leveraging lightweight deep learning architectures to achieve a balance between captioning accuracy and computational efficiency. The key contributions are summarized below:

- **Lightweight SSVC Framework:** A complete video captioning pipeline is designed and implemented using MobileNet as the visual feature extractor and lightweight recurrent neural networks (LSTM and GRU) as the sequence generators. This architecture is tailored for low-resource environments, enabling potential deployment on real-time and edge devices.
- **Efficient Feature Extraction:** Frame-level visual features are extracted from two benchmark datasets—MSVD and MSR-VTT—using MobileNet variants. These features are compact yet sufficiently expressive, significantly reducing the computational burden compared to traditional backbone networks.
- **Effective Decoder Design:** LSTM and GRU-based decoder networks are trained on the extracted features using cross-entropy loss. These models are optimized to generate grammatically correct and semantically meaningful single-sentence descriptions for each video clip.
- **Comprehensive Evaluation:** The proposed models are rigorously evaluated using widely-accepted metrics, including BLEU, METEOR, ROUGE-L, and CIDEr. These evaluations confirm the capability of the lightweight models to produce competitive results with a fraction of the computational cost.
- **Modular and Reproducible Codebase:** A clean and modular implementation is developed to ensure reproducibility and facilitate future experimentation. The codebase is structured to support easy extension with alternative feature extractors, decoders, and training objectives.

Together, these contributions establish a strong foundation for resource-aware video captioning and pave the way for further advancements in efficient video understanding.

1.7 Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 presents a comprehensive literature review, highlighting prior work and commonly used architectures in the field of video captioning. Chapter 3 details the proposed architecture, focusing on the LSTM-based model and the rationale behind key design decisions. Chapter 4 discusses the results, including training logs, evaluation metrics, and qualitative examples of generated captions. Finally, Chapter 5 provides a summary of the work and suggests potential directions for future research.

Chapter 2

Literature Review

This chapter provides a comprehensive overview of the existing research landscape in the field of Single Sentence Video Captioning (SSVC). It begins by discussing widely-used benchmark datasets that have facilitated progress in this domain. The chapter then explores common strategies for frame sampling and feature extraction from video sequences, with an emphasis on the use of pretrained convolutional neural networks (CNNs) such as VGG, ResNet, MobileNet, and ShuffleNet for visual representation.

Further, it reviews prominent model architectures adopted for sentence generation, including Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTM) networks, attention-based models, and the more recent Transformer-based frameworks. In addition, the chapter outlines standard evaluation metrics—such as BLEU, METEOR, ROUGE-L, and CIDEr—that are employed to objectively assess the quality and relevance of generated video captions.

By synthesizing prior work across datasets, architectures, and evaluation methods, this chapter establishes the foundation and motivation for the lightweight SSVC framework proposed in this thesis.

2.1 Video Captioning Datasets – An Overview

Datasets form the backbone of research in video captioning, serving as standardized resources that contain curated collections of videos paired with corresponding textual descriptions. These datasets enable the training and evaluation of models across a diverse range of visual content and linguistic expressions.

Each dataset varies significantly in terms of video content (e.g., indoor scenes, outdoor environments, human activities, animal behavior, vehicle motion), number of video clips, average video duration, number of associated captions per video, and the language or complexity of annotations. The diversity and richness of a dataset directly influence a model’s ability to generalize across unseen scenarios and visual contexts.

Furthermore, standardized datasets facilitate fair benchmarking, allowing researchers to compare the performance of different models under consistent conditions. A well-annotated and comprehensive dataset is essential not only for training robust models but also for performing meaningful evaluations.

In this section, we present an overview of ten widely-used video captioning datasets, highlighting their key characteristics, structure, and relevance to the development of Single Sentence Video Captioning (SSVC) systems.

Datasets form the backbone of research in video captioning, serving as standardized resources that contain curated collections of videos paired with corresponding textual

descriptions. These datasets enable the training and evaluation of models across a diverse range of visual content and linguistic expressions.

Each dataset varies significantly in terms of video content (e.g., indoor scenes, outdoor environments, human activities, animal behavior, vehicle motion), number of video clips, average video duration, number of associated captions per video, and the language or complexity of annotations. The diversity and richness of a dataset directly influence a model’s ability to generalize across unseen scenarios and visual contexts.

Furthermore, standardized datasets facilitate fair benchmarking, allowing researchers to compare the performance of different models under consistent conditions. A well-annotated and comprehensive dataset is essential not only for training robust models but also for performing meaningful evaluations.

In this section, we present an overview of ten widely-used video captioning datasets, highlighting their key characteristics, structure, and relevance to the development of Single Sentence Video Captioning (SSVC) systems.

2.1.1 MSVD Dataset (Microsoft Research Video Description)

The MSVD dataset[2] (also known as the YouTube2Text dataset) is one of the most popular benchmarks in video captioning research. It is composed of 2,089 short YouTube videos, from 10 to 25 seconds in length. Each video is paired with multiple captions (5–20), provided by human annotators. The captions summarize the videos in a kind of natural language that details a range of simple motion, such as strolling, eating, or playing.

The MSVD dataset possesses several notable characteristics. Each video is annotated with multiple captions, introducing linguistic heterogeneity through variations in sentence structure and vocabulary. This diversity makes the dataset valuable for training and evaluating captioning models. Due to its manageable size and general content focused on everyday actions rather than domain-specific tasks, MSVD serves as a practical and widely-used benchmark for initial testing and prototyping.

However, the dataset also presents certain challenges. Its relatively small size compared to larger video captioning datasets can hinder the generalization capabilities of trained models. Additionally, because the videos are short and often simple, the associated captions may lack detailed temporal information, potentially limiting the model’s ability to capture complex temporal dependencies.

Despite these limitations, MSVD remains useful for several applications. It is frequently employed for benchmarking the performance of video captioning systems on basic tasks. Furthermore, it supports the development and evaluation of models that aim to generate multiple diverse descriptions for a single video, making it valuable for studying linguistic variability and caption generation consistency.



Figure 2.1: Sample frames from the MSVD dataset showing video content

Video descriptions:

- Caption 1: A woman is talking a man.
- Caption 2: A woman is talking with a man.

2.1.2 MSR-VTT Dataset (Microsoft Research Video to Text)

The MSR-VTT dataset [22] is one of the most widely used and diverse benchmarks in the field of video captioning. It comprises 10,000 video clips, each paired with 20 human-annotated captions, resulting in a total of 200,000 natural language descriptions. The videos are sourced from the internet and span across 20 broad categories, including sports, music, news, cooking, gaming, and more, making it a rich and heterogeneous dataset.

The videos in MSR-VTT typically range from 10 to 30 seconds in length and exhibit a wide variety of scenes and actions. This diversity makes MSR-VTT particularly effective for evaluating the generalization capability of captioning models across different video types. The multiple captions per video reflect a range of linguistic expressions and perspectives, thereby enriching the dataset’s descriptive variability.

A notable feature of MSR-VTT is the availability of category labels for each video, which enables multi-task learning frameworks that combine video classification with caption generation. This opens avenues for developing context-aware and category-specific captioning models.

Despite its advantages, MSR-VTT presents several challenges. The wide range of visual content—from fast-moving sports clips to relatively static instructional videos—demands robust spatiotemporal reasoning. Additionally, many captions contain complex descriptions that require the model to understand fine-grained details and temporal relationships within the video.

Owing to its scale, diversity, and complexity, MSR-VTT is extensively used for training and benchmarking state-of-the-art video captioning models. It serves as a standard testbed for assessing how well models perform in generating accurate, contextually relevant, and human-like descriptions under real-world variability.



Figure 2.2: Sample video frames from the MSR-VTT dataset

Video descriptions:

- Caption 1: A man in a room holds a bike and talk to camera.
- Caption 2: A man adjust a bike and talk.
- Caption 3: A man adjust a bike and talk off tire.
- Caption 4: A man unfold the bike and put the tire back on.

2.1.3 ActivityNet Captions

The ActivityNet Captions dataset [9] is a large-scale benchmark designed for temporally localized video captioning. It consists of over 20,000 untrimmed videos, each divided into multiple segments, with every segment annotated with a natural language caption describing a specific event or action. This temporally dense annotation framework makes ActivityNet Captions a unique and challenging dataset compared to traditional short-video captioning benchmarks.

The dataset primarily focuses on human-centric activities, including sports, social interactions, and daily life routines. Unlike datasets that concentrate on short, isolated clips, ActivityNet emphasizes temporal segmentation—requiring models not only to generate meaningful captions, but also to accurately detect and align them with the corresponding time spans within longer video sequences.

This emphasis on temporal grounding introduces several challenges. Models must be equipped to handle extended videos containing multiple, potentially overlapping events. They must also capture transitions between activities to maintain temporal coherence and semantic clarity in the generated descriptions. Furthermore, the captions often demand nuanced understanding of both visual and contextual elements over extended durations.

ActivityNet Captions serves as a valuable benchmark for evaluating models in tasks such as temporal event localization, dense video captioning, and context-aware summarization. It is particularly useful for testing the effectiveness of attention mechanisms and other temporal reasoning strategies, as these are critical for selectively focusing on the most informative video segments during caption generation.



Figure 2.3: Sample video frames from the ActivityNet dataset

Video descriptions:

- Caption 1: A child performing on *The Voice*.
- Caption 2: A child performs a song on stage.
- Caption 3: A family celebrates a winning audition.
- Caption 4: A girl is giving a stage performance.squat cycle.

2.1.4 YouCookII Dataset

The YouCookII dataset [27] is a large-scale benchmark focused on instructional video understanding, specifically in the context of cooking. It comprises over 2,000 untrimmed cooking videos, each ranging from 2 to 5 minutes in duration. These videos are segmented into discrete procedural steps, with each segment annotated with a natural language caption that describes the specific cooking action being performed.

A distinctive feature of the YouCookII dataset is its emphasis on step-by-step instructional content. Videos are collected from YouTube and encompass a wide variety of global cuisines and recipes, providing a rich and diverse source of visual and semantic data. This procedural structure makes YouCookII particularly valuable for studying temporal modeling in structured tasks.

However, the dataset also presents unique challenges. Due to the extended nature of the videos and their multi-step compositions, models must learn to capture long-range temporal dependencies effectively. Furthermore, the use of domain-specific vocabulary—such as culinary tools, ingredients, and cooking techniques—can complicate the language modeling component, particularly for general-purpose captioning models.

YouCookII is widely used in research on instructional video summarization, dense video captioning, and temporal activity localization. It serves as a valuable benchmark for evaluating how well models can understand and describe fine-grained, domain-specific activities that unfold sequentially over time.



Figure 2.4: Sample cooking video frames from the YouCookII dataset
Video descriptions:

- Caption 1: A man is stirring in a pan.
- Caption 2: A man is smelling something.

2.1.5 Charades Captions

The Charades dataset [17] is a large-scale benchmark comprising over 10,000 short videos depicting individuals performing everyday indoor activities. Unlike simpler datasets focused on isolated actions, Charades features temporally extended sequences where each video is annotated with multiple captions, capturing not just individual actions but also the order and context in which they occur.

What distinguishes the Charades dataset is its focus on realistic, home-like environments where human actors interact with common household objects. These interactions include multi-step routines such as making coffee, cleaning a room, folding laundry, or organizing items—providing a rich context for modeling complex human behaviors and object affordances.

A key challenge in working with Charades lies in the need for fine-grained semantic understanding. Captioning models must accurately detect objects, interpret sequences of actions, and infer human intent based on subtle contextual cues. Many activities involve overlapping or interleaved motions, requiring models to maintain temporal consistency and interpret nuanced transitions between steps.

Charades is widely utilized in tasks such as activity recognition, temporal action localization, and dense video captioning. It is particularly valuable for developing models that aim to understand multi-object interactions, sequential human behavior, and real-world task execution within indoor scenes.



Figure 2.5: Sample activity video frames from the Charades dataset
Video descriptions:

- Caption 1: A child starts singing in the middle of her test.
- Caption 2: A girl answers written questions on a test in school as part of a music video.
- Caption 3: A child decides a math question and yells animal noises.
- Caption 4: A child taking a test while songs are playing in the background.

2.1.6 LSMDC (Large-Scale Movie Description Challenge)

The Large-Scale Movie Description Challenge (LSMDC) dataset [15] is one of the most comprehensive and widely-used benchmarks in video captioning. It contains over 200,000 short video clips extracted from a variety of movies, each paired with natural language descriptions. These clips typically consist of cinematic scenes or brief dialogues, capturing a wide spectrum of visual storytelling styles.

LSMDC is distinguished by the diversity of its content, encompassing multiple genres such as drama, action, romance, thriller, and comedy. The corresponding captions are derived from movie scripts and descriptive video service (DVS) narrations, often combining scene-level visual descriptions with linguistic nuances and spoken dialogue. This makes the dataset particularly rich and complex, demanding models to integrate both visual and contextual auditory information for effective caption generation.

A significant challenge in utilizing LSMDC arises from the need to perform high-level semantic reasoning. Unlike simpler datasets, the captions in LSMDC may involve indirect references, emotional expressions, or metaphorical language. Understanding such descriptions requires models to go beyond surface-level object recognition and instead interpret character interactions, mood, and narrative flow.

Given its scale and complexity, LSMDC is ideally suited for advanced tasks such as narrative video understanding, story-based captioning, and multimodal reasoning. It provides a strong foundation for developing models aimed at real-world video analysis applications where context, emotion, and sequential logic play a critical role.



Figure 2.6: Sample video frames from the LSMDC dataset

Video descriptions:

- Caption 1: Nebulous clouds appear in outer space while a man talks about telepathy and alien life.
- Caption 2: A man talking about living life on other planets.
- Caption 3: There is a sculpture with some lights all around.
- Caption 4: There are different phenomena of the universe including a goddess.

2.1.7 VATEX (Video and Text in English and Chinese)

The VATEX dataset [21] is a large-scale multilingual video captioning benchmark designed to facilitate research in cross-lingual video understanding. It consists of over 41,000 short videos, each annotated with ten English and ten Chinese captions. The dataset encompasses a broad range of general-purpose activities such as sports, social events, cultural performances, and daily routines, making it highly diverse and globally representative.

A unique characteristic of VATEX is its support for multilingual training and evaluation. By offering paired captions in both English and Chinese, the dataset enables the development of models that can perform multilingual caption generation, cross-lingual retrieval, and translation-aware video understanding. This makes VATEX an important resource for studying how visual content can be interpreted and described across different linguistic and cultural contexts.

The challenges associated with VATEX primarily revolve around linguistic variation and semantic alignment. Models must be capable of generating captions that are not only syntactically correct in multiple languages but also semantically faithful to the video content. Additionally, the diversity of video topics requires strong generalization capabilities across domains.

VATEX serves as a crucial benchmark for multilingual and cross-modal learning tasks, and it plays a significant role in advancing research in cross-lingual transfer learning, zero-shot captioning, and global-scale video understanding applications.

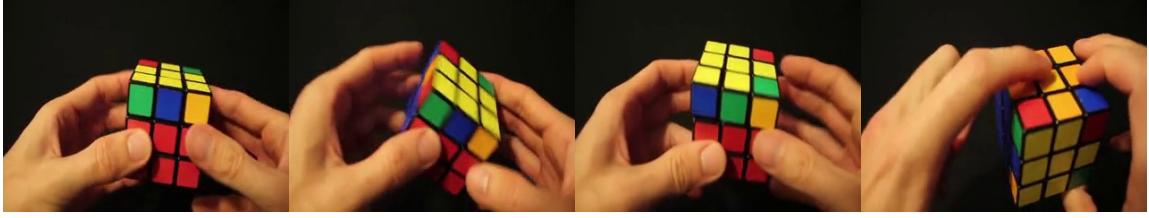


Figure 2.7: Sample instructional video frames from the VATEX dataset

Video descriptions:

- Caption 1: A guy plays with an origami.
- Caption 2: A man describes using a Rubik's cube.
- Caption 3: A man is playing.
- Caption 4: A man is teaching how to solve a Rubik's cube.

2.1.8 Dataset Summary

Video captioning datasets vary widely with respect to scale, remit, temporal resolution, and linguistic heterogeneity. These differences lead to certain data being applied to particular tasks, and introducing special issues to machine learning models.

Table 2.1 summarizes the key characteristics of widely used video captioning datasets, including the number of videos, average duration, caption diversity, and primary focus.

Table 2.1: Characteristics of Video Captioning Datasets

Dataset	No. of Videos	Avg. Video Duration	Captions/Video	Focus
MSVD	1,970	10–25 sec	40	General-purpose
MSR-VTT	7,010	10–30 sec	20	Diverse categories
ActivityNet Captions	20,000	Varies	Dense	Temporal localization
YouCookII	2,000	2–5 min	Step-by-step	Cooking instructions
Charades	10,000+	10–30 sec	Multiple	Everyday activities
LSMDC	200,000	Varies	1	Movie scenes
VATEX	41,000	Varies	Bilingual (EN, CN)	General-purpose
COCO-Video	Varies	Short clips	Multiple	Action + scene captions
TGIF	100,000 GIFs	1–5 sec	1	Repetitive actions
UCF101 Captions	13,320	1–10 sec	1	Action-centric

2.2 Key Frame Selection Strategy

In video captioning[20], processing every frame of a video is often unnecessary and computationally expensive. Redundant frames can slow down training and inference without adding useful information. Therefore, selecting a subset of informative and representative frames—known as key frames—is a crucial step for building efficient and accurate captioning models. Below are common strategies employed in literature, along with their characteristics, benefits, and drawbacks:

2.2.1 Uniform Sampling

In uniform sampling[23], video frames are extracted at fixed time intervals—e.g., one frame per second—irrespective of the underlying scene content or motion. This strategy ensures consistent temporal coverage across the video and is commonly used as a baseline in video analysis tasks due to its simplicity.

Formally, let a video V have a total duration T seconds and be recorded at a frame rate of R frames per second (fps), resulting in $N = T \times R$ total frames. To uniformly sample K frames, we select frame indices:

$$s_k = \left\lfloor \frac{k \cdot N}{K + 1} \right\rfloor, \quad \text{for } k = 1, 2, \dots, K \quad (2.1)$$

where s_k is the index of the k -th uniformly selected frame. This spacing ensures that frames are evenly distributed throughout the video timeline.

The primary advantage of uniform sampling lies in its ease of implementation and minimal computational requirements. However, this method may miss critical actions or scene transitions that occur between sampled frames, leading to potential loss of important contextual information. Despite this limitation, its lightweight nature makes it suitable for resource-constrained environments or as a standard baseline in ablation studies.

2.2.2 Motion-Based Sampling

This method[29] focuses on selecting frames that exhibit significant motion, using techniques such as optical flow or frame differencing to detect changes between consecutive frames. By prioritizing areas of activity, it aims to capture the most dynamic and informative segments of the video, making it particularly suitable for scenarios like surveillance, sports analysis, or action recognition.

Let the video be represented by a sequence of frames $\{I_1, I_2, \dots, I_N\}$. A simple yet effective way to measure motion is through frame differencing:

$$M_t = \|I_t - I_{t-1}\|_1, \quad \text{for } t = 2, \dots, N \quad (2.2)$$

where M_t represents the motion score at frame t , and $\|\cdot\|_1$ denotes the L1 norm over pixel intensities. Alternatively, optical flow-based magnitude $\|\mathbf{F}_t\|$ can be used:

$$M_t = \frac{1}{H \times W} \sum_{x=1}^W \sum_{y=1}^H \|\mathbf{F}_t(x, y)\|_2 \quad (2.3)$$

where $\mathbf{F}_t(x, y)$ is the 2D optical flow vector at pixel (x, y) between frames I_{t-1} and I_t , and $H \times W$ is the frame resolution.

A total of K frames are selected by choosing the top- K scores from the motion sequence $\{M_2, M_3, \dots, M_N\}$:

$$\mathcal{S}_{motion} = TopK(M_2, M_3, \dots, M_N) \quad (2.4)$$

The main advantage of this approach is its ability to highlight action-intensive moments, which are often critical for understanding the video's content. However, it may overlook static but semantically rich scenes (e.g., holding objects or facial expressions). Moreover, the motion estimation process introduces moderate to high computational overhead, making it less suitable for real-time or resource-limited applications.

2.2.3 Entropy-Based Selection

Entropy-based frame selection[14] quantifies the amount of visual information or texture in a frame by measuring the distribution of pixel intensities. Frames with high entropy—those containing varied textures, multiple objects, or complex patterns—are prioritized under the assumption that they carry more meaningful content.

For a given grayscale frame I_t , let $\mathcal{P} = \{p_1, p_2, \dots, p_L\}$ be the normalized histogram of pixel intensity values over L discrete bins (typically $L = 256$ for 8-bit images). The Shannon entropy of frame I_t is computed as:

$$H_t = - \sum_{i=1}^L p_i \log p_i \quad (2.5)$$

where H_t represents the visual entropy at frame t . Frames with higher H_t indicate greater pixel variability, suggesting more visual complexity.

Given a sequence of frames $\{I_1, I_2, \dots, I_N\}$, the top K frames with the highest entropy values are selected:

$$\mathcal{S}_{entropy} = TopK(H_1, H_2, \dots, H_N) \quad (2.6)$$

This approach is advantageous due to its simplicity and low computational cost, making it suitable for quick filtering of visually rich frames across large datasets. However, it may mistakenly prioritize cluttered or noisy scenes over semantically meaningful ones, as entropy does not directly capture content relevance or context.

2.2.4 Histogram-Based Selection

This approach [25] computes color or edge histograms for each frame and scores them based on histogram variance or dissimilarity. Frames with the highest histogram richness or contrast—indicating greater visual diversity—are then selected as the Top- K representatives of the video.

Let each video frame I_t be represented by a normalized histogram $h_t \in \mathbb{R}^B$, where B is the number of bins (e.g., an 8-bin histogram for edge magnitudes or color intensities). The histogram can be computed over grayscale gradients, color channels, or edge magnitudes (e.g., using Sobel filters).

One simple metric to quantify histogram richness is the standard deviation of its bins:

$$S_t = \sqrt{\frac{1}{B} \sum_{i=1}^B \left(h_t^{(i)} - \bar{h}_t \right)^2}, \quad \text{where } \bar{h}_t = \frac{1}{B} \sum_{i=1}^B h_t^{(i)} \quad (2.7)$$

Alternatively, inter-frame histogram difference (e.g., using ℓ_1 distance between consecutive histograms) can be used to detect contrast or change:

$$D_t = \|h_t - h_{t-1}\|_1 \quad (2.8)$$

The final Top- K selected frames are chosen based on either the highest richness score S_t or difference score D_t :

$$\mathcal{S}_{hist} = TopK(S_1, S_2, \dots, S_N) \quad \text{or} \quad TopK(D_2, D_3, \dots, D_N) \quad (2.9)$$

The primary advantage of this method lies in its computational simplicity, making it well-suited for videos with noticeable color variation or lighting changes. However, since it relies purely on pixel-level statistics, it does not capture object-level semantics or meaningful content. Nevertheless, its lightweight nature ensures applicability to large-scale datasets.

2.2.5 CNN-Based Scoring

In this method [10], each frame is passed through a pretrained Convolutional Neural Network (CNN), such as ResNet or MobileNet, to extract deep semantic features or generate attention heatmaps. Based on activation magnitudes or spatial attention scores, the Top- K frames that are most semantically informative are selected for further processing.

Let a frame I_t be processed through a pretrained CNN $\phi(\cdot)$, producing a feature activation tensor $A_t \in \mathbb{R}^{C \times H \times W}$, where C is the number of channels and $H \times W$ is the spatial resolution. A simple yet effective scoring function is the global activation energy:

$$S_t = \frac{1}{C \cdot H \cdot W} \sum_{c=1}^C \sum_{x=1}^H \sum_{y=1}^W |A_t(c, x, y)| \quad (2.10)$$

Alternatively, if the CNN provides spatial attention maps (e.g., via CAM or Grad-CAM), the importance score can be directly computed from the attention map $\alpha_t(x, y)$:

$$S_t = \sum_{x=1}^H \sum_{y=1}^W \alpha_t(x, y) \quad (2.11)$$

The final selected set of frames is obtained as:

$$\mathcal{S}_{cnn} = TopK(S_1, S_2, \dots, S_N) \quad (2.12)$$

This approach effectively captures important objects and actions, making it well-suited for enhancing captioning performance. However, the requirement to run CNN inference on every frame introduces high computational overhead. Additionally, the scoring is dependent on the pretrained network used, which may encode dataset-specific biases. Nevertheless, the semantic richness it offers often justifies the cost in applications where accuracy is prioritized over speed.

2.2.6 Reinforcement Learning-Based Selection (PickNet)

In this method [5], a learnable policy network is trained to select informative frames by interacting with a video captioning environment. The network makes sequential decisions on whether to “pick” or “skip” each frame in the input sequence. The agent is rewarded based on how well the selected frames contribute to generating accurate and descriptive captions—often using captioning metrics like CIDEr as the reward signal.

Let a video be represented by a sequence of frame features $\{f_1, f_2, \dots, f_T\}$. A binary decision sequence $\{a_1, a_2, \dots, a_T\} \in \{0, 1\}^T$ is generated by a policy network π_θ , where $a_t = 1$ indicates that frame t is selected. The sequence of selected frames is then used to generate a caption \hat{Y} , and the reward $R(\hat{Y})$ is computed using a language quality metric (e.g., CIDEr).

The goal is to maximize the expected reward:

$$\mathcal{L}(\theta) = -\mathbb{E}_{\pi_\theta(a_{1:T})}[R(\hat{Y})] \quad (2.13)$$

The gradient of this objective is estimated using the REINFORCE algorithm:

$$\nabla_\theta \mathcal{L}(\theta) \approx -\sum_{t=1}^T \nabla_\theta \log \pi_\theta(a_t | f_t) \cdot (R(\hat{Y}) - b) \quad (2.14)$$

where b is a baseline (e.g., running average reward) used to reduce variance during training.

The primary advantage of this approach is that it learns dataset-specific frame selection strategies directly optimized for caption quality. However, reinforcement learning introduces complexity: training is often unstable, sensitive to reward design, and computationally expensive. Despite these challenges, PickNet provides a powerful and adaptive mechanism for end-to-end learnable frame selection in video captioning tasks.

2.2.7 Hybrid Methods

This strategy [12] integrates multiple visual cues—such as motion intensity, entropy, face detection, skin-color regions, and saliency maps—into a unified scoring framework. A weighted scoring system is employed, where each frame f_i is assigned a score S_i based on a linear combination of normalized features:

$$S_i = \alpha \cdot M_i + \beta \cdot E_i + \gamma \cdot F_i + \delta \cdot K_i + \epsilon \cdot L_i \quad (2.15)$$

where:

- M_i = motion activity score (e.g., Sum of Absolute Differences),
- E_i = entropy score (texture/visual richness),
- F_i = face presence score (e.g., number or size of detected faces),
- K_i = skin-color region proportion,
- L_i = saliency score from a precomputed map,
- $\alpha, \beta, \gamma, \delta, \epsilon \in [0, 1]$ are tunable weights.

Frames with the top K highest scores S_i are selected for downstream tasks.

The main advantage of this hybrid approach lies in its ability to balance diverse visual cues, making it more robust and generalizable across different types of videos. However, its implementation complexity is higher, and the scoring weights must be carefully tuned for each dataset or domain. Moreover, the aggregation of multiple visual analyses can increase inference time.

Table 2.2: Comparison of Key Frame Selection Strategies

Method	Principle	Advantages	Disadvantages	Compute Cost
Uniform Sampling	Sample frames at fixed time intervals	Easy to implement, consistent coverage	May miss critical events	Lightweight
Motion-Based Sampling	Select frames with high motion using optical flow or difference	Good for capturing actions	May miss static informative scenes; costly	Moderate to Heavy
Entropy-Based Selection	Select frames with high texture/information using entropy score	Captures visually rich content	May prioritize noisy/cluttered frames	Lightweight
Histogram-Based Selection	Select frames based on histogram variance or dissimilarity	Simple and interpretable	Misses semantic relevance	Lightweight
CNN-Based Scoring	Score frames using pretrained CNN outputs or attention maps	Captures semantic content, object-level information	Requires CNN inference on all frames	Moderate to Heavy
Reinforcement Learning (PickNet)	Learn policy to pick frames that maximize caption quality	Learns optimal selection for dataset	Training complexity, reward sensitivity	Heavy
Hybrid Methods	Combine entropy, motion, face/skin detection into weighted scores	Robust and flexible	Harder to tune, higher complexity	Moderate to Heavy

2.3 Normalization and Resizing of Video Frames

Before feeding video frames into deep feature extractors like CNNs, it is essential to ensure consistency in frame size and intensity values. This preprocessing step enhances the model's ability to generalize across diverse videos and stabilizes the training process. The two most critical components of this step are frame resizing and normalization.

2.3.1 Frame Resizing

Raw video frames extracted from videos can vary widely in their resolution (e.g., 640×480 , 1280×720 , etc.). However, pretrained convolutional neural networks (CNNs) such as ResNet, MobileNet, and others require a fixed input size. Hence, all frames are resized to a standard resolution of 224×224 pixels before being passed through the network.

- **Why 224×224 ?** This is the standard input size expected by most pretrained CNN architectures trained on ImageNet. Resizing helps ensure that the model input is consistent and compatible.
- **Preserving aspect ratio:** In some cases, aspect ratio distortion may occur. To avoid this, padding can be added to maintain the original frame structure, although in this project we perform direct resizing for simplicity.

2.3.2 Normalization

Normalization [7] is applied to scale the pixel values of each frame to a predefined range or distribution. This step improves numerical stability and convergence speed during training and inference.

- **Pixel Scaling:** Raw pixel values ranging from $[0, 255]$ are rescaled to $[0, 1]$ or standardized using mean and standard deviation values.
- **Standard Mean and Std Deviation:** For pretrained models trained on ImageNet, normalization is done using the following channel-wise statistics:

$$mean = [0.485, 0.456, 0.406], \quad std = [0.229, 0.224, 0.225]$$

These values represent the dataset-wide average and deviation for Red, Green, and Blue channels respectively.

- **Purpose:** Normalizing to this distribution ensures that the pretrained model sees input data in the same range it was trained on, thus improving feature extraction accuracy.

2.3.3 Implementation Summary

Each extracted frame is passed through a preprocessing pipeline that includes:

1. Converting the frame to RGB format (if not already).
2. Resizing it to 224×224 resolution.

3. Normalizing the pixel values using ImageNet statistics.
4. Converting the result to a tensor suitable for the PyTorch or TensorFlow CNN model.

This preprocessing pipeline ensures that the feature extractor receives high-quality, uniformly processed inputs, leading to better visual feature embeddings and improved downstream caption generation.

2.4 Pretrained CNN Models for Visual Feature Extraction

In video captioning, the encoder is responsible for converting visual content into meaningful feature representations[16][8]. We use a variety of pretrained convolutional neural networks (CNNs) as encoders to extract frame-level features. These models are pretrained on large-scale image datasets such as ImageNet and have demonstrated excellent performance in visual recognition tasks. This section describes the architectures explored in this work, highlighting their differences in depth, complexity, and suitability for lightweight or high-performance captioning.

2.4.1 ResNet Family

ResNet (Residual Network) is a deep convolutional architecture that incorporates skip connections to alleviate the vanishing gradient problem, enabling the training of very deep networks. In this work, we employ the ResNet-18 variant, which consists of 18 layers.

ResNet-18 is particularly well-suited for scenarios requiring fast inference or operation on devices with limited computational resources due to its shallow depth. Its primary advantage lies in its low computational cost and quick execution. However, being relatively shallow, it may not capture complex visual abstractions as effectively as deeper variants, making it less ideal for highly intricate or detailed scenes.

ResNet-50

ResNet-50 is a deeper variant of the ResNet architecture comprising 50 layers and utilizing bottleneck blocks to maintain computational efficiency while increasing depth. It offers a strong balance between inference speed and representational power.

The main advantage of ResNet-50 is its ability to extract rich and detailed features from visual data, making it suitable for more complex video content. It delivers better accuracy than shallower networks like ResNet-18 while remaining manageable in terms of model size. However, this comes at the cost of increased computational overhead, making it relatively heavier and slower in low-resource environments.

ResNet-101

ResNet-101 further extends the depth of the ResNet architecture to 101 layers, allowing it to model more complex patterns and hierarchies in visual data. It is particularly effective in scenarios involving fine-grained object recognition or dense scenes with multiple entities.

The primary advantage of ResNet-101 lies in its high accuracy and strong representational capacity, especially on object-rich datasets. However, this benefit comes with significantly increased memory consumption and computational cost, making it less suitable for real-time or resource-constrained applications.

ResNet-152

ResNet-152 is the deepest variant utilized in this work, comprising 152 layers and excelling at capturing fine-grained visual semantics in complex scenes. Its extensive depth enables

superior abstraction and detailed feature extraction, which can be highly beneficial for tasks requiring nuanced visual understanding.

The key advantage of ResNet-152 is its ability to produce high-quality, discriminative features that improve model performance. However, this comes at the cost of substantial computational requirements, leading to longer training and inference times, and making it less practical for real-time or edge deployments.

2.4.2 MobileNet Family

MobileNet models are designed for efficient deployment on mobile and edge devices. They use depthwise separable convolutions to reduce parameters and computation.

MobileNetV2

MobileNetV2 introduces inverted residual blocks and linear bottlenecks, enabling efficient neural network architecture design especially suited for mobile and embedded applications. These innovations help maintain representational power while significantly reducing the number of parameters and computational load.

The main advantage of MobileNetV2 is its excellent balance between speed and accuracy, making it ideal for low-resource scenarios. However, due to its shallower architecture and design trade-offs, it may yield slightly lower accuracy compared to deeper models like ResNet-50 or ResNet-101, particularly on complex visual tasks.

MobileNetV3-Large

MobileNetV3, developed using Neural Architecture Search (NAS), incorporates advanced features such as squeeze-and-excitation modules and hard-swish activation functions. It is designed to optimize the trade-off between latency and accuracy, making it well-suited for real-time mobile applications.

Its key strength lies in delivering high accuracy while maintaining efficient execution. However, compared to MobileNetV2, it is slightly more computationally intensive, which may impact performance on extremely resource-constrained devices. Nonetheless, it remains a strong choice for applications needing both speed and precision.

MobileNetV3-Small

MobileNetV3-Small is a compact version of MobileNetV3 tailored for extremely low-resource environments, such as microcontrollers or low-end mobile devices. It prioritizes minimal latency and reduced memory usage, making it ideal for deployment in edge scenarios with strict hardware constraints.

While it excels in efficiency and speed, its minimal architecture results in reduced representational capacity, which can limit performance on visually complex or diverse datasets. Therefore, it is best suited for simple tasks where computational efficiency is paramount over high accuracy.

2.4.3 ShuffleNetV2 Family

ShuffleNetV2 models are optimized for speed and accuracy trade-offs by minimizing memory access cost (MAC) and using channel shuffle operations.

ShuffleNetV2-0.5x

ShuffleNetV2 is a highly efficient CNN architecture designed for ultra-low-latency scenarios. It employs channel shuffling and pointwise group convolution to reduce computational overhead while maintaining reasonable accuracy, making it ideal for mobile and embedded applications.

Its main advantage lies in its ultra-fast inference and minimal memory footprint. However, this comes at the cost of reduced feature representation capability, making it less suitable for complex visual tasks that require high-level semantic understanding.

ShuffleNetV2-1.0x

ShuffleNetV2 (1.0 \times) is a balanced variant designed to offer an improved trade-off between accuracy and computational efficiency compared to its smaller counterparts. It increases the model width to enhance feature representation while preserving the core lightweight design principles such as channel shuffling and grouped convolutions.

This configuration is advantageous for scenarios requiring both real-time performance and moderate accuracy. However, due to its relatively shallow architecture, it may still struggle with highly complex visual scenes that demand deeper semantic understanding.

ShuffleNetV2-1.5x

ShuffleNetV2 (1.5 \times) offers improved feature representation by further increasing model width compared to 1.0 \times and 1.5 \times variants. This expansion allows it to capture more detailed visual patterns, enhancing accuracy in moderately complex tasks.

While it achieves higher accuracy, the trade-off is a noticeable increase in model size and computational cost. Thus, it is better suited for applications where slightly higher latency or resource usage is acceptable in exchange for better performance.

ShuffleNetV2-2.0x

ShuffleNetV2 (2.0 \times) is the largest and most capable variant in the ShuffleNet family, offering the highest accuracy due to its increased width and deeper feature processing. It is particularly useful for scenarios requiring better recognition accuracy while still maintaining a relatively lightweight structure compared to conventional CNNs.

However, this performance gain comes at the cost of increased memory usage and computational overhead, making it less ideal for real-time applications on edge devices with strict resource constraints.

2.4.4 Comparison Summary

Table 2.3 summarizes the computational characteristics of the CNN models evaluated in this work. It includes the input resolution, number of parameters, and corresponding GFLOPs for processing a single image. These values reflect the efficiency and complexity of each model and are computed using the FVCORE library. All models operate on a standard input size of 224 \times 224 and vary significantly in their parameter count and FLOPs, highlighting the trade-offs between lightweight efficiency and expressive power.

Table 2.3: Comparison of CNN Models Based on GFLOPs and Parameters

CNN	Image Size	Params (M)	GFLOPs
ShuffleNetV2-0.5x	224×224	1.4M	0.040
MobileNetV3-Small	224×224	2.5M	0.060
ShuffleNetV2-1.0x	224×224	2.3M	0.140
MobileNetV3-Large	224×224	5.5M	0.220
ShuffleNetV2-1.5x	224×224	3.5M	0.300
MobileNetV2	224×224	3.5M	0.300
EfficientNet-B0	224×224	5.3M	0.390
ShuffleNetV2-2.0x	224×224	7.4M	0.580
EfficientNet-B1	224×224	7.8M	0.690
ResNet-18	224×224	11.7M	1.81
ResNet-50	224×224	25.6M	4.09
ResNet-101	224×224	44.5M	7.80

By experimenting with these models, we conduct ablation studies to evaluate the trade-offs between speed, model size, and captioning accuracy across MSVD and MSR-VTT datasets.

2.5 Review of Traditional Approaches for Video Captioning

Prior to deep learning[1], video captioning has been heavily reliant on classic methods, which have relied on handcrafted features, rule-based methods and statistical modeling. These methods laid the groundwork for video captioning but struggled with generalization and scalability when applied to diverse video datasets. Methods this subsection consider from more traditional paradigms, their advantages and disadvantages are discussed.

2.5.1 Template-Based Methods

Template-based methods were among the earliest approaches used for video captioning. These systems generate sentences by filling detected entities into predefined linguistic templates, often following simple syntactic structures such as “Subject–Verb–Object” (SVO). For example, a video depicting a person walking a dog might yield the caption:

“*The person is walking a dog.*”

Mathematically, a caption C is constructed as:

$$C = T(s, v, o) \quad (2.16)$$

where T is a predefined sentence template, and s , v , and o are the subject, verb, and object entities extracted using object detection and action recognition modules.

The primary advantages of template-based methods include simplicity, fast inference, and high interpretability due to fixed sentence formats. They are well-suited for datasets with constrained vocabularies and predictable actions. However, they struggle with generalization to complex or diverse scenarios, lack linguistic richness, and often result in monotonous or rigid descriptions.

2.5.2 Statistical Models

Statistical methods introduced probabilistic modeling into the domain of video captioning by formulating it as a sequence prediction problem. Among the most prominent techniques were **Hidden Markov Models (HMMs)** and **Conditional Random Fields (CRFs)**, each offering unique strengths and limitations.

Hidden Markov Models (HMMs): HMMs represent the captioning task as a generative model where the sequence of words w_1, w_2, \dots, w_T is generated from a sequence of latent states s_1, s_2, \dots, s_T conditioned on the observed video features X . The model estimates:

$$P(w_{1:T}, s_{1:T}|X) = P(s_1) \prod_{t=2}^T P(s_t|s_{t-1}) \prod_{t=1}^T P(w_t|s_t, X) \quad (2.17)$$

While HMMs worked well for short, sequential patterns, they struggled with long-term dependencies and were sensitive to the assumptions of state independence.

Conditional Random Fields (CRFs): CRFs addressed these limitations by modeling the conditional probability of label sequences given input features, without assuming independence between observed variables. For a sequence of words \mathbf{w} and visual features \mathbf{X} , CRFs define:

$$P(\mathbf{w}|\mathbf{X}) = \frac{1}{Z(\mathbf{X})} \exp \left(\sum_t \psi(w_t, \mathbf{X}, t) + \sum_t \phi(w_t, w_{t-1}, \mathbf{X}) \right) \quad (2.18)$$

Here, ψ and ϕ are unary and pairwise potentials, and $Z(\mathbf{X})$ is a normalization term.

Despite their ability to model contextual relationships better than HMMs, CRFs required extensive handcrafted feature engineering and incurred higher computational costs. Both models were ultimately limited by scalability and flexibility, paving the way for neural captioning models.

2.5.3 Comparison of Traditional Approaches

A detailed comparison of traditional video captioning models is presented in Table 2.4, highlighting their strengths and limitations.

2.5.4 Relevance to Modern Techniques

While traditional methods have largely been supplanted by deep learning approaches, they contributed foundational concepts:

- Emphasized the significance of sequence modeling for video captioning.
- Highlighted the importance of spatial and temporal information integration for captioning tasks.
- Set the foundation for the contemporary encoder-decoder architectures, attention mechanisms, and sequence-to-sequence architectures.

Understanding traditional methods provides valuable insights into the challenges of video captioning and how modern techniques have evolved to address these limitations.

Table 2.4: Comparison of Traditional Video Captioning Models

Method	Strengths	Limitations
Template-Based	Simple, interpretable, and easy to implement. Effective for straightforward tasks.	Poor generalization to complex scenes. Repetitive and lacks nuanced descriptions.
Hidden Markov Models (HMMs)	Good for sequence prediction and short-term dependencies.	Struggles with long-term dependencies and complex event understanding.
Conditional Random Fields (CRFs)	Captures global structure in sequences and models dependencies effectively.	Computationally expensive and relies on handcrafted features.

2.6 Review of Deep Learning Approaches for Video Captioning

End-to-end trainable architectures have changed the video captioning by bringing revolution through the emergence of deep learning. In contrast to conventional techniques, deep learning models use massive amounts of data, as well as complex structures to generalize to a variety of video tasks. This part discusses the principle deep learning techniques, building blocks of the technique, and their evolution.

In this subsection, we survey some deep-learning-based video-captioning methods that are using attention mechanism, reinforcement learning, adversarial networks, and graph-based methods. These methods have contributed enormously to enhancing performance of video captioning by enhancing how models handle and produce natural language descriptions of video contents.

2.6.1 Attention-Based Methods for Video Captioning

Introduction

Attention-based approaches have significantly advanced the performance of sequence modeling tasks such as video captioning. Unlike traditional models that treat all input features uniformly, attention mechanisms dynamically focus on the most relevant parts of the input—such as specific frames, spatial regions, or temporal segments—during the decoding process. This selective focus enables the model to generate more contextually appropriate and semantically rich captions by attending to informative visual cues at each time step.

How Attention Works

The core idea behind attention mechanisms is to allow the model to dynamically focus on the most relevant parts of the input sequence when generating each word in the output. This is accomplished by computing a weighted sum over all encoder outputs, where the weights represent the importance of each input element with respect to the current decoding step.

Given encoder outputs $\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_N$ and the decoder hidden state at time step t , denoted as \mathbf{s}_t , the attention mechanism computes:

$$e_i^{(t)} = score(\mathbf{s}_t, \mathbf{h}_i) \quad (2.19)$$

$$\alpha_i^{(t)} = \frac{\exp(e_i^{(t)})}{\sum_{j=1}^N \exp(e_j^{(t)})} \quad (2.20)$$

$$\mathbf{c}_t = \sum_{i=1}^N \alpha_i^{(t)} \mathbf{h}_i \quad (2.21)$$

Here, $e_i^{(t)}$ is the unnormalized relevance score between the decoder state and the i -th encoder output, often computed using a dot product, a feed-forward network, or a concatenation-based scoring function. The normalized weights $\alpha_i^{(t)}$ form the attention distribution, and \mathbf{c}_t is the resulting context vector passed to the decoder.

Soft Attention: Soft attention is fully differentiable, making it trainable via standard backpropagation. It provides a smooth weighting over all frames, ensuring that all input information is considered. However, the computational complexity grows linearly with the number of input frames N , which can become a bottleneck for long videos.

Hard Attention: In contrast, hard attention selects a single frame (or a subset) to focus on at each time step:

$$i^* = \arg \max_i e_i^{(t)}, \quad \mathbf{c}_t = \mathbf{h}_{i^*} \quad (2.22)$$

This reduces computation but introduces non-differentiability due to the discrete selection. As a result, reinforcement learning techniques like REINFORCE are typically used for training. While more efficient, hard attention may overlook important content if the selected frame is not truly representative.

Thus, attention mechanisms—particularly soft attention—play a crucial role in improving the alignment between video content and generated captions by adaptively focusing on the most relevant parts of the input.

2.6.2 Reinforcement Learning-Based Video Captioning

Deep reinforcement learning (RL) integrates the representational power of deep learning with the decision-making capabilities of reinforcement learning, making it particularly effective for sequential generation tasks such as video captioning. Unlike supervised methods that minimize cross-entropy loss using ground-truth captions, RL treats caption generation as a decision process where a model—termed the *agent*—interacts with an *environment* and learns to maximize a cumulative reward based on feedback signals. This interactive framework introduces adaptability and improves generalization during inference.

In video captioning, the agent sequentially generates words based on visual features and prior outputs. The environment includes the video content and the partial caption generated so far. Rewards are computed using evaluation metrics such as CIDEr, BLEU, and ROUGE, which measure alignment with human-annotated references. Reinforcement learning helps overcome two key issues in supervised learning: (1) the *objective mismatch*,

Method	Key Features	Strengths	Weaknesses
Soft Attention	Continuous weights for all frames.	Smooth, differentiable.	Computational cost for dense inputs.
Hard Attention	Focuses on a single frame.	Efficient for sparse data.	Non-differentiable, RL required.
Global + Local Attention	Combines holistic and fine-grained focus.	Balances context and detail.	Requires careful balancing.
Spatio-Temporal Attention	Attention across regions and frames over time.	Captures spatial and temporal details.	High computational cost.
Self-Attention	Models global dependencies.	Parallelizable, scalable.	Requires significant computational power.

Table 2.5: Comparison of Attention-Based Methods

where training loss does not align with evaluation metrics, and (2) *exposure bias*, where models perform poorly during inference due to discrepancies in input distribution between training and testing.

The reinforcement learning objective is typically formulated using a policy gradient method, where the model maximizes the expected reward $J(\theta)$:

$$J(\theta) = \mathbb{E}_{\hat{y} \sim p_\theta}[r(\hat{y})] \quad (2.23)$$

Here, \hat{y} is a generated caption, p_θ is the model’s caption distribution parameterized by θ , and $r(\hat{y})$ is the reward (e.g., CIDEr score). The gradient can be estimated using the REINFORCE algorithm:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\hat{y} \sim p_\theta}[r(\hat{y}) \nabla_\theta \log p_\theta(\hat{y})] \quad (2.24)$$

Despite these strengths, RL-based models are computationally intensive and often suffer from high variance in gradient estimation. Additionally, the effectiveness of training depends on the design of the reward function. Poorly chosen rewards can misguide learning and degrade caption quality.

Recent approaches address these challenges by integrating RL with attention mechanisms, hierarchical policies, and pretrained encoders. Future work may focus on efficient reward computation, stability improvements, and leveraging large-scale vision-language models for generalization.

2.6.3 Adversarial Networks for Video Captioning

Generative Adversarial Networks (GANs) have also been explored in video captioning to enhance the realism and coherence of generated captions. A typical GAN framework includes two components: a generator G and a discriminator D . The generator produces captions conditioned on video features, while the discriminator evaluates how close a generated caption is to a human reference, thereby guiding the generator to improve.

The training objective for adversarial learning is modeled as a minimax game:

$$\min_G \max_D \mathbb{E}_{y \sim p_{data}} [\log D(y)] + \mathbb{E}_{\hat{y} \sim G(z)} [\log(1 - D(\hat{y}))] \quad (2.25)$$

Here, y is a ground-truth caption, and \hat{y} is a caption generated by G . The discriminator learns to distinguish between real and generated captions, while the generator learns to produce captions that fool the discriminator.

Adversarial training encourages the generation of more human-like and diverse captions, complementing traditional likelihood-based objectives. However, challenges remain, including mode collapse, training instability, and difficulty in designing discriminators that effectively assess caption quality. Hybrid models combining adversarial loss with supervised or reinforcement objectives are often used to mitigate these issues and improve convergence.

2.6.4 Video Question and Answering (V-QA)

Video Question and Answering (V-QA) is a challenging yet impactful task at the intersection of computer vision and natural language processing, aiming to answer natural language questions based on video content. Unlike static image question answering, V-QA requires a comprehensive understanding of temporal sequences, spatial context, object interactions, and language semantics. The motivation behind V-QA lies in its ability to achieve deeper semantic understanding of video content, enabling models to answer questions about specific actions, events, and relationships. It is particularly suited for applications involving temporal reasoning, such as surveillance analysis, educational content navigation, and intelligent video summarization. The general workflow in V-QA begins with feature extraction from both the video and the corresponding question. Video features are typically obtained using convolutional neural networks (CNNs), recurrent networks, or vision transformers to capture spatial and temporal dynamics. Simultaneously, the question is encoded using natural language models such as LSTMs, BERT, or GPT to extract syntactic and semantic representations.

Once the features are extracted, the video and question embeddings are fused into a joint representation, often using concatenation, attention, or bilinear pooling techniques:

$$H_{joint} = Fusion(H_{video}, H_{question}) \quad (2.26)$$

This joint representation is then passed through attention mechanisms and a classification layer to predict the most appropriate answer. The probability distribution over possible answers is computed as:

$$P(a | v, q) = softmax(Attention(H_{joint})) \quad (2.27)$$

V-QA offers several strengths—it enables detailed comprehension of video scenes, handles diverse question types including descriptive, factual, and temporal, and demonstrates robust reasoning capabilities across time. However, it also presents challenges. The need to process and integrate multi-modal inputs increases the architectural complexity of models. Furthermore, performance is highly dependent on the availability of large-scale, high-quality annotated datasets. The computational requirements for both training and inference are also significantly higher compared to unimodal tasks, calling for the development of more efficient architectures and scalable learning frameworks.

2.7 Various Model Summary

Video captioning has progressed remarkably with the advent of different deep learning models. Each model has a particular architecture and method to solve the problems of producing accurate and contextually appropriate captions. In this section, an overview of the most frequently employed models, the elements of those models, and the datasets used to perform the evaluations is given.

Table 2.6: Summary of Various Video Captioning Models

Model	Encoder	Decoder	Datasets Used
CNN-LSTM	Convolutional Neural Network (CNN) to extract spatial features from video frames.	Long Short-Term Memory (LSTM) network to model temporal dependencies and generate captions.	MSVD, MSR-VTT
S2VT (Sequence-to-Sequence Video to Text)	LSTM-based encoder for sequential modeling of video frames.	LSTM-based decoder for generating text sequentially.	MSVD
Two-Stream Networks	Combines RGB frame features and optical flow for motion analysis.	LSTM network to fuse spatial and motion features and generate captions.	UCF101, Charades
Attention-Based Models	CNN for spatial feature extraction.	LSTM with attention mechanism to focus on specific video regions during caption generation.	MSR-VTT, ActivityNet
Transformer Models	Transformer encoder with self-attention to capture both spatial and temporal dependencies.	Transformer decoder for parallelized and contextually rich text generation.	MSR-VTT, VATEX

2.7.1 Detailed Analysis of Models

- **CNN-LSTM:**
 - Combines CNNs for spatial feature extraction and LSTMs for temporal modeling.
 - Suitable for datasets with short video clips and simple actions.
 - Limitations: Struggles with long sequences and complex temporal dependencies.
- **S2VT:**
 - Fully LSTM-based architecture designed for sequential processing of video data.
 - Efficient for datasets like MSVD with short clips and basic actions.
 - Limitations: Performance diminishes on datasets with diverse content and long sequences.
- **Two-Stream Networks:**

- Incorporates both spatial (RGB frames) and motion (optical flow) features.
- Effective for datasets emphasizing action recognition, such as UCF101 and Charades.
- Limitations: Computationally expensive due to the need for optical flow calculation.

- **Attention-Based Models:**

- Introduces an attention mechanism to focus on relevant video regions while generating captions.
- Improves performance on complex datasets with temporal annotations, like ActivityNet.
- Limitations: Performance depends heavily on the quality of attention maps.

- **Transformer Models:**

- Employs self-attention to model relationships across long video sequences effectively.
- Excels in parallelized text generation, making it faster and more scalable.
- Limitations: Requires substantial computational resources and large datasets for training.

2.8 Comparison of Methods Across Datasets

Evaluating and comparing different video captioning models across multiple datasets is essential to understand their generalizability and effectiveness. Table 2.7 summarizes the performance of several well-known approaches on two benchmark datasets: MSVD and MSR-VTT. Each method is evaluated using four standard metrics: BLEU-4, METEOR, ROUGE-L, and CIDEr.

Table 2.7: Performance Comparison of Video Captioning Methods

Method Architecture	Model Dataset	BLEU-4	METEOR	ROUGE-L	CIDEr
MSVD Dataset Results					
S2VT [18]	CNN (VGG-16) + LSTM	33.6	29.1	60.9	73.9
LSTM-YT [19]	CNN (GoogLeNet) + LSTM	31.2	28.2	60.1	66.0
PickNet [4]	CNN (ResNet-152) + Attn LSTM	41.3	29.1	61.4	89.6
SA-LSTM [24]	CNN (Inception-V4) + S-Attn LSTM	40.3	29.2	61.2	84.5
MARN [13]	CNN (ResNet-101) + Memory Attn	42.1	29.8	62.1	90.2
Transformer [28]	CNN (ResNet-101) + Transformer	3.5	29.9	62.3	92.0
MGSA [3]	CNN (ResNet-152) + MG-Attn	44.2	30.1	62.8	93.5
MSR-VTT Dataset Results					
S2VT [18]	CNN (VGG-16) + LSTM	31.4	24.8	57.2	48.3
MARN [13]	CNN (ResNet-101) + Memory Attn	35.1	26.5	58.2	52.9
ORG-TRL [26]	CNN (ResNet-152) + ObjRel Trans	43.6	28.8	61.4	56.1
PickNet [4]	CNN (ResNet-152) + Attn LSTM	36.3	26.5	58.6	51.7
Transformer [28]	CNN (ResNet-101) + Transformer	44.6	29.1	61.9	58.5
VTransformer [11]	CNN (EffNet-B4) + Vid Transformer	45.8	29.6	62.5	60.2
HMN [6]	CNN (ResNeXt-101) + HMN	46.2	30.0	63.1	61.8

As shown in the table, Transformer-based models outperform earlier LSTM-based architectures in most metrics. Methods like PickNet also demonstrate strong performance, particularly in optimizing for CIDEr by learning efficient frame selection strategies.

2.9 Caption Evaluation Metrics

Evaluating the quality of generated captions in video captioning is a crucial task that involves comparing machine-generated sentences with human-annotated reference captions. Several standard automatic metrics have been adopted from the field of machine translation and text summarization to assess caption fluency, accuracy, and relevance. This section provides an overview of the most widely used evaluation metrics: BLEU, METEOR, ROUGE-L, and CIDEr.

2.9.1 BLEU

BLEU(Bilingual Evaluation Understudy) is a precision-oriented metric that measures the n-gram overlap between the generated caption and reference captions. It is commonly reported as BLEU-1, BLEU-2, BLEU-3, and BLEU-4, depending on whether unigram, bigram, trigram, or 4-gram matches are considered.

- **BLEU-1:** Measures the overlap of individual words (unigrams).
- **BLEU-2 to BLEU-4:** Consider higher-order n-gram overlaps, capturing fluency and phrase correctness.
- **Formula:**

$$BLEU = BP \cdot \exp \left(\sum_{n=1}^N w_n \log p_n \right)$$

where p_n is the precision of n-grams, w_n is the weight (typically uniform), and BP is the brevity penalty to penalize short hypotheses.

- **Advantages:** Simple, fast, and effective at measuring surface-level similarity.
- **Limitations:** Does not consider word semantics or synonyms; overly penalizes rephrasing.

2.9.2 METEOR

METEOR(Metric for Evaluation of Translation with Explicit ORdering) is a recall-based metric that aligns words using exact matches, stem matches, synonym matches (via WordNet), and paraphrases. It computes a harmonic mean of precision and recall, with higher weight on recall.

- **Formula:**

$$METEOR = F_{mean} \cdot (1 - Penalty)$$

where F_{mean} is the harmonic mean of precision and recall, and $Penalty$ penalizes fragmented matches.

- **Advantages:** Considers synonymy and word order; shows better correlation with human judgment.
- **Limitations:** Computationally heavier than BLEU.

2.9.3 ROUGE-L

ROUGE-L(Recall-Oriented Understudy for Gisting Evaluation) evaluates the Longest Common Subsequence (LCS) between the predicted and reference captions. It focuses on capturing sentence-level structure similarity.

- **Formula:**

$$ROUGE - L = \frac{(1 + \beta^2) \cdot Precision \cdot Recall}{Recall + \beta^2 \cdot Precision}$$

- **Advantages:** Suitable for evaluating fluency and sequence similarity.
- **Limitations:** Ignores synonymy and semantic meaning.

2.9.4 CIDEr

CIDEr (Consensus-based Image Description Evaluation) is designed specifically for evaluating image and video captions. It measures how well a generated sentence matches a set of reference sentences using TF-IDF weighted n-gram matching.

- **Formula:**

$$CIDEr(c_i, S_i) = \sum_{n=1}^4 w_n \cdot CIDEr_n(c_i, S_i)$$

where c_i is the generated caption, S_i is the set of ground truth captions, and TF-IDF weighting is used for each n-gram.

- **Advantages:** Captures consensus and importance of content; highly correlated with human evaluations.
- **Limitations:** Requires multiple references per video for meaningful results.

tabularx

2.9.5 Summary of Metrics

Table 2.8: Summary of Caption Evaluation Metrics

Metric	Focus	Type
BLEU-1 to BLEU-4	Precision	N-gram overlap
METEOR	F-measure (Recall-heavy)	Semantic + Synonym match
ROUGE-L	Sequence similarity	Longest Common Subsequence
CIDEr	TF-IDF Consensus	Semantic Relevance

In this work, all four metrics (BLEU, METEOR, ROUGE-L, and CIDEr) are used to comprehensively evaluate the captioning quality of different models across MSVD and MSR-VTT datasets.

A key drawback observed in the literature is that many existing video captioning models rely on computationally heavy architectures, such as deep ResNet or Transformer-based encoders and complex attention mechanisms. While these models achieve high performance, they often demand significant GPU resources, making them less suitable for real-time or resource-constrained environments.

Chapter 3

Proposed Architecture

This chapter presents the complete pipeline designed for Single Sentence Video Captioning (SSVC) with a focus on lightweight architectures suitable for real-time and edge deployment. The system is composed of three main stages: frame sampling and preprocessing, visual feature extraction using MobileNetV2, and sequential caption generation via Recurrent Neural Networks (RNNs), such as LSTM or GRU. Each component is carefully selected to balance performance and efficiency in terms of memory, speed, and accuracy.

3.1 Overview

The proposed framework offers a streamlined, computationally efficient approach to SSVC. Input videos undergo preprocessing and frame sampling to obtain representative keyframes. These are passed through a lightweight Convolutional Neural Network (CNN) encoder for compact feature extraction. The resulting frame-level features are fed into a sequential decoder, implemented using LSTM or GRU layers, to generate coherent and context-aware natural language captions.

To ensure seamless learning across modules, the model is trained end-to-end, enabling the encoder and decoder to jointly optimize for video-caption alignment. This design supports real-time captioning needs on low-resource platforms without compromising semantic quality.

3.2 Methodology

The entire pipeline has been implemented in PyTorch and trained using a NVIDIA RTX A4500 GPU with 20GB VRAM. The methodology involves the following key stages:

3.2.1 Video Preprocessing

The MSVD and MSR-VTT datasets consist of videos with diverse durations, resolutions, and scene dynamics. To ensure uniformity in input data, each video is processed as follows:

- **Frame Sampling:** From each video, 30 frames are extracted using uniform temporal sampling. This strategy captures the visual essence of the entire video without overloading the computational pipeline.

- **Resolution Standardization:** Original video resolutions vary—MSVD videos typically have 480p (640×480), while MSR-VTT videos may go up to 720p (1280×720). All extracted frames are resized to a fixed resolution of 224×224 pixels to match the input requirements of the CNN backbone.
- **Normalization:** Each frame undergoes pixel intensity normalization to align with the distribution of ImageNet-trained CNN models. RGB channels are normalized using the mean and standard deviation values $[0.485, 0.456, 0.406]$ and $[0.229, 0.224, 0.225]$, respectively.

This preprocessing stage ensures that the CNN encoder receives consistent, well-aligned inputs, improving the reliability and performance of the downstream captioning module.

Normalization: Each frame is normalized using the standard ImageNet channel-wise mean and standard deviation values to ensure consistency and stability during feature extraction. This process aligns the input distribution with what the pretrained CNN (e.g., MobileNetV2) expects.

Mathematically, each pixel value x_c in channel $c \in \{R, G, B\}$ is normalized as:

$$x_c^{norm} = \frac{x_c - \mu_c}{\sigma_c} \quad (3.1)$$

where μ_c and σ_c are the mean and standard deviation for each channel, respectively:

$$\mu = [0.485, 0.456, 0.406], \quad \sigma = [0.229, 0.224, 0.225] \quad (3.2)$$

This step helps the pretrained CNN to generalize better by providing input with a familiar statistical distribution.

Feature Extraction using MobileNetV2: We employ MobileNetV2 as our visual feature extractor due to its balance between performance and computational cost. Each preprocessed video frame is independently passed through the MobileNetV2 network (excluding the final classification layer), producing a 1280-dimensional feature vector.

Let a video be represented by T sampled frames: $\{I_1, I_2, \dots, I_T\}$. Passing each frame I_t through the CNN gives:

$$f_t = CNN(I_t), \quad f_t \in \mathbb{R}^{1280} \quad (3.3)$$

To obtain a single fixed-length representation for the entire video, we apply mean pooling across all frame-level features:

$$f_{video} = \frac{1}{T} \sum_{t=1}^T f_t, \quad f_{video} \in \mathbb{R}^{1280} \quad (3.4)$$

This mean-pooled vector f_{video} serves as the visual embedding for the entire video.

Vocabulary Construction: For the text processing pipeline, a vocabulary is built from all captions in the MSVD and MSR-VTT datasets. Let $\mathcal{C} = \{c_1, c_2, \dots, c_N\}$ represent the set of all captions, and let \mathcal{W} be the multiset of all words occurring in \mathcal{C} . For each word $w \in \mathcal{W}$, we compute its frequency $f(w)$.

The vocabulary \mathcal{V} is then defined as:

$$\mathcal{V} = \{w \in \mathcal{W} \mid f(w) \geq \tau\} \quad (3.5)$$

where $\tau = 2$ is the minimum word frequency threshold used to filter out rare words. This process yields a vocabulary size of $|\mathcal{V}| = 7,012$ for MSVD and 14,129 for MSR-VTT, respectively.

Data Loading: Each video’s pooled feature vector and its corresponding tokenized caption are paired and loaded using a custom `DataLoader`. Let $f_i \in \mathbb{R}^{1280}$ denote the feature vector for the i -th video, and let the associated caption be represented as a sequence of tokens $y_i = [y_{i,1}, y_{i,2}, \dots, y_{i,L_i}]$, where L_i is the length of the caption.

To enable efficient parallel processing in batches, all captions in a batch are padded to the same maximum length L_{max} :

$$\tilde{y}_i = [y_{i,1}, y_{i,2}, \dots, y_{i,L_i}, \underbrace{\text{<pad>}, \dots, \text{<pad>}}_{L_{max}-L_i}] \quad (3.6)$$

This ensures uniform tensor shapes of size $[B, L_{max}]$ for captions and $[B, 1280]$ for features during training, where B is the batch size.

Decoder Design: The decoder network is implemented using a two-layer Long Short-Term Memory (LSTM) architecture. It takes the fixed-length video feature vector $f_{video} \in \mathbb{R}^{1280}$ as input context to initialize the hidden state. At each time step t , the decoder predicts the next token y_t based on the previously generated token y_{t-1} and its internal hidden state.

Formally, the LSTM decoding process is given by:

$$h_t, c_t = LSTM(E(y_{t-1}), h_{t-1}, c_{t-1}) \quad (3.7)$$

$$p(y_t \mid y_{<t}, f_{video}) = \text{Softmax}(W_o h_t + b_o) \quad (3.8)$$

Here, $E(y_{t-1})$ denotes the embedding of the previous token, h_t and c_t are the hidden and cell states at time t , and W_o , b_o are output projection parameters. Dropout with a probability of 0.3 is applied between LSTM layers to prevent overfitting and improve generalization.

3.2.2 Training the Model

The video captioning model is trained using paired input: the video-level feature vector and its corresponding caption sequence. Each video is represented by a pooled 1280-dimensional feature vector extracted using MobileNetV2. Captions are tokenized using the constructed vocabulary and padded to a fixed length $L_{max} = 50$ for uniformity during batching.

The model architecture is a two-layer LSTM decoder trained with a cross-entropy loss function:

$$\mathcal{L}_{CE} = - \sum_{t=1}^L \log p(y_t \mid y_{1:t-1}, f_{video}) \quad (3.9)$$

where y_t is the target token at time step t , and $f_{video} \in \mathbb{R}^{1280}$ is the video feature vector. Padding tokens are ignored during loss computation using `ignore_index`.

The model is optimized using the Adam optimizer with a learning rate of 10^{-4} , and trained for 500 epochs. After each epoch, the model’s performance is evaluated on the validation set using the CIDEr score. The CIDEr metric measures similarity between generated and reference captions based on n-gram statistics, providing a caption-level reward function.

For decoding during evaluation, greedy search is used to generate predicted captions token-by-token until the end-of-sentence token ($\langle \text{eos} \rangle$) is produced or the maximum sequence length is reached. A sample decoding process is illustrated below:

$$\hat{y}_t = \arg \max_{w \in \mathcal{V}} p(w | \hat{y}_{1:t-1}, f_{\text{video}}) \quad (3.10)$$

The model checkpoint is saved whenever the validation CIDEr score improves. This training strategy ensures both sentence-level supervision (via cross-entropy loss) and corpus-level quality evaluation (via CIDEr).

Testing Phase: For final evaluation, the best-performing model checkpoint (based on validation CIDEr score) is tested on the held-out test set using *beam search decoding*. Beam widths from 1 to 10 are explored to allow the decoder to maintain multiple candidate sequences at each time step and select the most probable caption overall.

At each time step t , the beam search maintains the top- k partial hypotheses $\{Y_t^{(1)}, Y_t^{(2)}, \dots, Y_t^{(k)}\}$ ranked by cumulative log-probability:

$$Y_t^{(i)} = \arg \max_{Y_{1:t}} \sum_{j=1}^t \log p(y_j | y_{<j}, f_{\text{video}}), \quad i = 1, 2, \dots, k \quad (3.11)$$

The final output is selected as the sequence with the highest normalized log-likelihood among all completed hypotheses.

The generated captions are then evaluated using four standard metrics:

- **BLEU-4:** Measures precision of 4-gram matches.
- **METEOR:** Incorporates synonym and stem matches.
- **ROUGE-L:** Measures longest common subsequence.
- **CIDEr:** Consensus-based evaluation using TF-IDF n-grams.

Greedy decoding (beam width = 1) is used during validation for efficiency and consistency, while beam search during testing improves fluency and relevance of generated captions.

3.3 Flowchart Of Proposed Pipeline

The flowchart illustrates the proposed video captioning pipeline. It begins with video input, followed by frame selection, preprocessing, and feature extraction using MobileNetV2. Extracted features are aggregated and passed through an LSTM decoder to generate captions, which are then evaluated using standard metrics like BLEU and CIDEr.

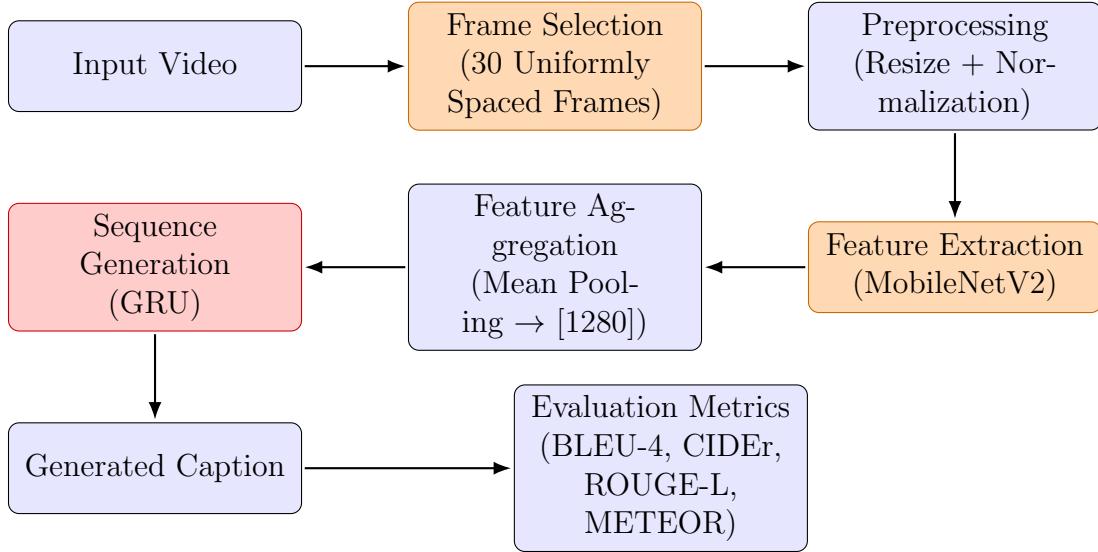


Figure 3.1: Block diagram of the proposed lightweight video captioning pipeline.

3.4 Experimental Setup

This section describes the datasets used, the model and training hyperparameters adopted, and the baseline configurations evaluated to validate the effectiveness of our lightweight video captioning pipeline.

We conduct experiments on two widely used benchmark datasets: the **Microsoft Video Description (MSVD)** dataset and the **MSR-VTT** dataset. MSVD consists of approximately 1,970 short video clips, each annotated with around 40 human-generated English captions, with durations ranging between 5 and 15 seconds. On the other hand, MSR-VTT is a more diverse and challenging dataset containing 7,010 web videos and over 200,000 captions covering a broad spectrum of visual scenes, events, and objects. For both datasets, we follow an 80-10-10 split: 80% for training, 10% for validation, and 10% for testing.

During preprocessing, each video is uniformly sampled to extract 30 frames, which are resized to 224×224 pixels and normalized using ImageNet statistics. The captions are tokenized using the Treebank tokenizer, and a vocabulary is constructed for each dataset with a minimum word frequency threshold of 2. This results in a vocabulary size of 7,052 tokens for MSVD and 14,129 tokens for MSR-VTT. To maintain consistency during training, maximum caption lengths are fixed to 50 words for MSVD and 80 words for MSR-VTT. This standardization enables consistent batch processing and stable model convergence during training.

3.4.1 Model and Training Hyperparameters

Model Hyperparameters

Hyperparameter	Value	Location
feature_dim	1280	MobileNetV2 feature dimension
hidden_dim	512	Model hidden size
num_layers	2	Number of layers in model
dropout	0.3	Dropout in model
vocab_size (MSVD)	7052	Vocabulary from MSVD captions
vocab_size (MSR-VTT)	14129	Vocabulary from MSR-VTT captions
pad_idx	0	Padding token index
max_len (MSVD)	50	Maximum caption length (MSVD)
max_len (MSR-VTT)	80	Maximum caption length (MSR-VTT)

Training Hyperparameters

Hyperparameter	Value
batch_size	32
learning_rate	1×10^{-4}
optimizer	Adam
loss_function	CrossEntropyLoss
epochs	500
device	NVIDIA RTX A450 GPU

Evaluation Parameters

Parameter	Value
evaluation_metric	CIDEr (for checkpoint saving)
beam_search_range	1 to 10
max_decode_len	50 (MSVD), 80 (MSR-VTT)
beam_score_normalization	Length Normalized (score / length)

3.4.2 Baseline Architectures

To benchmark the effectiveness of our lightweight video captioning system, we also experimented with multiple standard convolutional backbones that are widely used in literature. These include ResNet-18, ResNet-50, ResNet-101, ResNet-152, and both large and small versions of ShuffleNet and MobileNet. These heavy CNN encoders were evaluated in combination with LSTM ,LSTM with Attention and GRU decoders to assess trade-offs between performance and computational cost.

The goal of these baselines was to validate whether a lightweight encoder-decoder pipeline like MobileNetV2 + LSTM could retain competitive performance while offering faster inference and reduced memory usage.

3.5 Discussion

The proposed methodology provides an efficient and modular solution for video captioning, optimized for both performance and scalability. By integrating lightweight CNN-based feature extraction, sequence modeling, and evaluation-driven training, the framework effectively bridges visual understanding and language generation in a unified pipeline.

The use of MobileNetV2 enables low-latency feature extraction without compromising representational quality, making the approach suitable for real-time or resource-constrained environments. Mean pooling ensures a compact video representation, while the two-layer LSTM decoder maintains the temporal coherence needed for fluent sentence generation. Furthermore, the adoption of beam search decoding during testing allows the model to explore richer hypotheses, resulting in more descriptive and human-like captions.

Overall, the pipeline demonstrates a strong balance between simplicity, generalization, and computational efficiency — offering a practical and extensible foundation for future work in multimodal understanding, caption generation, and downstream applications such as video retrieval and assistive AI systems.

Chapter 4

Experiments and Results

This chapter outlines the experimental setup and provides detailed ablation analysis of the key components in the proposed video captioning pipeline, including visual feature extraction, sequence generation, and decoding strategies.

4.1 Experimental Setup

This section outlines the datasets, evaluation protocols, training configuration, and hardware specifications used throughout the experiments.

4.1.1 Datasets

All experiments were conducted on two widely used benchmark datasets—Microsoft Research Video Description Corpus (MSVD) and MSR-VTT. These datasets consist of short video clips covering a wide range of everyday scenes, each annotated with multiple human-written captions. Their linguistic diversity and visual richness make them well-suited for benchmarking video captioning models.

4.1.2 Evaluation Metrics

To quantitatively assess captioning performance, we employed the following standard metrics:

- **BLEU-1 to BLEU-4:** Measures n-gram precision between generated and reference captions, emphasizing exact word matches.
- **METEOR:** Considers synonym and stem variations, capturing semantic similarity in addition to word overlap.
- **ROUGE-L:** Evaluates the longest common subsequence (LCS), reflecting fluency and grammatical structure.
- **CIDEr:** Computes TF-IDF weighted consensus between candidate and reference captions, placing emphasis on human agreement.

4.1.3 Training Configuration

All models were trained using the Adam optimizer with a fixed learning rate of 1×10^{-4} . The training objective was the cross-entropy loss, with padding tokens excluded from gradient calculations to prevent bias in sequence modeling. The maximum caption length was capped at 50 tokens for MSVD Dataset and 80 tokens for MSR-VTT dataset to ensure uniformity across batches. Each training session was executed with a batch size of 32 and run for a maximum of 500 epochs. Early stopping was manually applied based on stagnation in validation performance to prevent overfitting.

4.1.4 Computational Environment

Model training and evaluation were carried out using the PyTorch deep learning framework. All experiments were executed on a single GPU-enabled environment, typically powered by NVIDIA Tesla T4 or RTX A6000 hardware, depending on platform availability. This configuration ensured accelerated training while maintaining reproducibility and efficiency.

4.2 Ablation 1: Frame Selection Strategy

To evaluate the effect of temporal sampling on captioning performance, we conducted extensive experiments using different frame selection strategies across two benchmark datasets—MSVD and MSR-VTT. All models used consistent configurations within each experiment (e.g., decoder type and CNN backbone) to isolate the impact of frame count and distribution.

Table 4.1: Effect of Frame Selection Strategies using MobileNetV2 Features with GRU Decoder on MSVD

# K	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDER	METEOR	ROUGE_L	#Params	GFLOPS
0.5 FPS	78.3	65.5	55.8	46.4	85	33.3	69.5	14.54 M	1.53 G
1 FPS	79.1	66.5	57.2	48.3	84	33.5	70.2	14.54 M	2.88 G
1.5 FPS	78.4	67.3	58.5	49.5	83.7	33.7	70.3	14.54 M	4.38 G
2 FPS	78.7	65.9	55.8	45.9	84.8	33.2	69.6	14.54 M	5.58 G
10 UNI.	79.8	67.8	58	48.3	88.6	34.8	70	14.54 M	3.18G
15 UNI.	78.2	65.6	55.6	46.4	79.1	32.5	68	14.54 M	4.68G
20 UNI.	77.8	65	55.3	46.2	83.8	33.3	69.2	14.54 M	6.18G
25 UNI.	78.2	66	56.2	46.8	87.3	34.2	69.6	14.54 M	7.68G
30 UNI.	80.5	69.4	60.3	51.5	90.2	34.6	71.3	14.54 M	9.18G
35 UNI.	79.1	67.6	58.5	49	88.4	34.5	71	14.54 M	10.68G
40 UNI.	79.4	67.6	57.9	48.5	88.3	34.5	71.4	14.54 M	12.18G
45 UNI.	78.4	66.8	57.3	48.2	84.3	33.4	69.2	14.54 M	13.68G
50 UNI.	77.8	66.1	56.7	47.2	83.4	33.5	69.2	14.54 M	15.18G

Analysis: Table 4.1 presents the impact of different frame selection strategies on captioning performance using MobileNetV2 features with a GRU decoder on the MSVD dataset. It includes both frame-per-second (FPS) sampling and uniformly spaced frame selection strategies. Among all settings, the configuration using 30 uniformly spaced frames consistently achieves the highest scores across all metrics—BLEU-1 to BLEU-4, CIDEr, METEOR, and ROUGE_L—indicating superior caption quality. Despite a modest increase in GFLOPs, the gain in accuracy justifies the computational cost. Hence, we adopt the 30 uniformly spaced frame strategy as the default choice for all subsequent experiments.

Table 4.2 further investigates the strategy using ResNet-152 features with an LSTM decoder on the MSVD dataset.

Table 4.2: Comparison of Frame Sampling Strategies using ResNet-152 Features with LSTM Decoder on MSVD

# K	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDER	METEOR	ROUGE_L
K=0.5 FPS	80.05	68.87	59.69	50.65	100.57	36.2	71.05
K=1 FPS	80.7	70.2	61.4	52.7	105.3	36.1	71.4
K=1.5 FPS	79.4	67.91	58.31	49.29	100.56	35.09	70.73
K=2 FPS	79.73	68.29	59.02	49.77	100.85	35.95	71.5
K=10	79.93	68.71	60.13	51.95	99.98	36.13	70.45
K=15	79.45	68.75	60.62	51.91	100.6	35.86	71.35
K=20	82.15	71.38	61.92	52.7	102.89	37.38	72.62
K=25	80.72	70.81	62.17	53.23	102.89	36.8	71.86
K=30	81.87	70.81	61.62	52.76	106.55	37.14	72.1
K=35	81.39	71.01	62.18	53.03	103.73	36.04	72.09
K=40	81.34	69.55	59.89	50.54	102.75	36.16	71.86
K=45	81.77	70.61	61.68	52.85	100.58	36.84	72.57
K=50	81.57	70.16	60.77	51.6	103.29	36.54	72.08

Analysis: The results in Table 4.2 show that uniformly spaced frame selection consistently outperforms FPS-based sampling when using ResNet-152 features with an LSTM decoder. Among all configurations, selecting $K = 30$ uniformly spaced frames yields the highest CIDEr score of 106.55, along with strong BLEU-4 (52.76) and METEOR (37.14) scores. This indicates that 30 frames are sufficient to capture the semantic and temporal essence of each video without redundancy. Increasing K beyond 30 leads to diminishing returns or slight degradation in performance, likely due to the inclusion of repetitive or less informative frames. Therefore, 30 uniformly spaced frames is adopted as the optimal selection strategy for all further experiments.

Table 4.3 shows the same setup on the MSR-VTT dataset, using ResNet-152 features with an LSTM decoder.

Table 4.3: Effect of Frame Sampling Strategies using ResNet-152 Features with LSTM Decoder on MSR-VTT

# K	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDER	METEOR	ROUGE_L
K=0.5 FPS	76.2	61.5	48.3	37.3	44.4	27.1	57.8
K=1 FPS	77.1	61.9	48.2	36.8	44.8	26.8	58.6
K=1.5 FPS	77	62.1	48.5	36.9	45.1	27.3	58.3
K=2 FPS	77.2	61.6	48.1	36.8	46	27.1	58.4
K=10	76.7	61.7	48.4	37.3	45.6	27.3	58.4
K=15	77.2	62.7	49.5	38	45.4	26.8	58.5
K=20	76.2	62	49.2	38	45.1	26.9	58.8
K=25	76.2	61	47.6	36.5	45.4	27	57.7
K=30	77.1	63.3	50.3	39	46.5	27.2	59.1
K=35	76	60.6	46.8	35.5	43	26.8	57.4
K=40	76.8	61.6	48.3	37.3	42.6	27.2	57.8
K=45	75.9	60.8	46.8	35.9	43.2	27.3	57.5
K=50	75.7	61.1	48.3	37.5	44.6	27.2	58.3

Analysis: As shown in Table 4.3, 30 uniformly spaced frames achieve the best overall performance on the MSR-VTT dataset using ResNet-152 features and an LSTM decoder. This configuration obtains the highest BLEU-4 (39.0), CIDEr (46.5), and ROUGE_L (59.1), while maintaining competitive METEOR and lower BLEU-n scores. Similar to observations from MSVD, increasing the number of frames beyond 30 leads to stagnation or degradation in performance, likely due to information redundancy. Thus, the 30-

frame uniform sampling strategy again proves to be optimal and is selected for all further MSR-VTT experiments.

Although overall CIDEr scores were lower due to MSR-VTT’s increased complexity, a similar trend was observed. Uniform selection of 30 frames again delivered the best CIDEr (46.5) and BLEU-4 (39.0), confirming that this strategy generalizes well across datasets.

Conclusion: Across all experiments, the strategy of selecting **30 uniformly spaced frames** consistently provided the best trade-off between temporal coverage and computational cost. It avoided the pitfalls of sparse sampling (missing key actions) and dense sampling (redundancy), and thus was selected as the default frame selection method for all subsequent modules in our pipeline.

4.3 Ablation 2: Visual Feature , Motion Feature Extractor

In this ablation study, we evaluate the impact of different convolutional backbones on captioning performance. All experiments used fixed settings: LSTM or GRU decoder with 30 uniformly sampled frames (unless otherwise noted), and pretrained CNNs as feature extractors.

Table 4.4: Comparison of Model Architectures Based on Parameters and GFLOPs

Architecture	Parameters (M)	GFLOPs	Remarks
MobileNetV2 + GRU	5.5	0.3	Lightweight and efficient
ResNet-152 + LSTM	65	11.8	High memory/compute
ResNet-152 + Transformer	80	13.5	Best on large datasets

Table 4.8 shows performance on the MSVD dataset using an LSTM decoder and 1 FPS frame sampling.

Table 4.5: Comparison of CNN Visual Feature Extractors on MSVD (LSTM Decoder, 1 FPS Sampling)

VISUAL EXTRACTOR	TENSOR	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDER	METEOR	ROUGE_L
SHUFFLENET V2×0.5	1024	71.0	54.8	44.0	34.3	58.2	28.5	63.2
SHUFFLENET V2×1.0	1024	70.2	54.8	44.5	35.3	60.7	28.3	63.2
SHUFFLENET V2×1.5	1024	78.9	66.4	56.1	45.9	87.8	33.5	69.4
SHUFFLENET V2×2.0	2048	79.3	68.1	59.0	50.2	94.0	35.7	70.8
MOBILENET V2	1280	76.3	62.5	52.5	43.2	76.5	30.9	66.6
MOBILENET V3.SMALL	576	76.2	62.9	53.3	44.2	75.8	32.1	68.1
MOBILENET V3.LARGE	960	77.1	64.3	54.1	44.5	81.5	32.7	67.5
RESNET 18	512	76.4	63.0	53.2	43.8	82.1	33.4	67.8
RESNET 50	2048	79.0	68.0	58.7	49.4	97.7	35.7	72.0
RESNET 101	2048	80.8	70.5	61.9	52.9	101.2	35.3	72.0
RESNET 152	2048	80.7	70.2	61.4	52.7	104.8	36.2	71.5

Analysis: In this setup, we observe that **ResNet-101** and **ResNet-152** achieved the highest CIDEr scores (101.2 and 104.8), reflecting their strong capacity to capture fine-grained visual features. However, this comes at the cost of increased computation:

both models output 2048-dimensional features and have very high parameter counts and GFLOPs.

In contrast, **MobileNetV2** achieved a respectable CIDEr of 76.5 and BLEU-4 of 43.2 — remarkably competitive considering its significantly smaller size and compute demand. Likewise, **ShuffleNet V2×2.0** delivered strong performance (CIDEr = 94.0) while being lightweight compared to deep ResNets.

Conclusion: While deeper ResNets provide marginally higher accuracy, the trade-off in computational cost is substantial. Given its excellent balance of performance, low parameter count, and low GFLOPs, **MobileNetV2 is chosen as the default visual extractor** in subsequent experiments.

Table 4.6 shows performance on the MSR-VTT dataset using an LSTM decoder and 1 FPS frame sampling.

Table 4.6: Comparison of CNN Visual Feature Extractors on MSR-VTT (LSTM Decoder, 1 FPS Sampling)

VISUAL EXTRACTOR	TENSOR	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDER	METEOR	ROUGE_L
SHUFFLENET V2×0.5	1024	69.7	52.4	39.7	29.7	29.8	23.9	52.9
SHUFFLENET V2×1.0	1024	71.3	53.8	40.2	29.9	31.8	24.5	53.8
SHUFFLENET V2×1.5	1024	76.2	60.6	47.2	35.7	43.1	27.3	57.7
SHUFFLENET V2×2.0	2048	76.9	61.2	47.3	36.0	43.3	27.0	57.5
MOBILENET V2	1280	75.3	59.6	46.1	35.3	41.0	26.6	56.7
MOBILENET V3_SMALL	576	75.8	59.6	46.2	35.0	38.3	25.9	56.3
MOBILENET V3_LARGE	960	76.4	60.8	46.8	35.3	43.1	26.8	57.1
RESNET 18	512	75.1	59.4	46.1	35.5	40.7	26.7	57.0
RESNET 50	2048	77.3	62.6	49.2	37.9	44.4	27.9	58.6
RESNET 101	2048	77.3	61.9	47.8	36.6	47.4	27.8	57.9
RESNET 152	2048	77.4	61.3	47.5	36.6	45.3	27.0	57.5

Analysis: On the MSR-VTT dataset, deeper models like **ResNet-101** achieved the highest CIDEr (47.4), with solid BLEU-4 (36.6). However, MobileNetV2 delivered a competitive CIDEr of 41.0 and BLEU-4 of 35.3 — only a small drop in accuracy while using significantly fewer parameters and computational resources. Similarly, ShuffleNet V2×2.0 performed closely (CIDEr = 43.3) while also being lightweight.

Conclusion: These results confirm that **MobileNetV2 offers a strong balance between efficiency and accuracy** across datasets. Its performance is comparable to large backbones while reducing both inference time and model size. For practical deployment scenarios where resource efficiency matters, MobileNetV2 is the optimal choice and is adopted as the default visual extractor in our final system.

Table 4.7 summarizes performance using a GRU decoder and 30 uniformly sampled frames for various CNN visual extractors.

Table 4.7: Comparison of Visual Extractors using GRU Decoder ($K = 30$ Uniformly Sampled Frames)

VISUAL EXTRACTOR	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDER	METEOR	ROUGE_L	# PARAMETERS	GFLOPS
RESNET 18	76.6	63.1	52.4	42.6	81.5	33.1	68.1	22.74 M	54.48 G
RESNET 50	80.1	68.5	59.2	49.6	88.6	33.6	70.3	36.64 M	122.88 G
RESNET 101	79.1	67.2	58.1	49.6	96.6	34.6	70.7	55.54 M	234.18 G
RESNET 152	81.2	69.4	60.6	51.7	99.9	35.2	70.7	71.23 M	345.8 G
MOBILENET V2	80.5	69.4	60.3	51.5	90.2	34.6	71.3	14.54 M	9.18 G
MOBILENET V3_SMALL	76.3	62.8	53.1	43.2	77.7	31.9	69.4	13.54 M	1.98 G
MOBILENET V3_LARGE	77.7	64.8	55.1	46.3	85.7	33.2	69.6	16.54 M	6.78 G
SHUFFLENET V2×0.5	69.5	53.6	42.7	32.9	53.3	28.0	62.3	12.44 M	1.38 G
SHUFFLENET V2×1.0	70.2	53.8	42.9	33.5	56.6	28.0	62.7	13.34 M	4.38 G
SHUFFLENET V2×1.5	76.5	63.2	52.9	42.9	78.0	32.3	66.6	14.54 M	9.18 G
SHUFFLENET V2×2.0	78.2	67.2	58.8	50.7	80.8	34.6	68.8	18.44 M	17.58 G

Analysis: As observed, **ResNet-152** and **ResNet-101** again deliver top scores in CIDEr (99.9 and 96.6 respectively) and BLEU-4. However, these come at the cost of significantly more parameters and FLOPs, making them computationally expensive.

MobileNetV2, with only 14.54 million parameters and 9.18 GFLOPs, achieves a strong CIDEr score of 90.2 and BLEU-4 of 51.5, which is nearly on par with ResNet-152 while being far more efficient. While ShuffleNet V2 \times 2.0 also performs reasonably well (CIDEr = 80.8), it does not outperform MobileNetV2 in consistency or BLEU-4.

Conclusion: Across all GRU-based experiments, **MobileNetV2 maintains the best balance between performance and efficiency**. It offers high-quality caption generation with significantly lower resource requirements, making it our final choice of CNN visual backbone for the complete video captioning pipeline.

Final Conclusion:

From the results across all CNN backbones, it is evident that while deeper architectures such as ResNet-152 yield slightly higher BLEU and CIDEr scores, they do so at the expense of significantly increased model complexity and computational cost. In contrast, **MobileNetV2 achieves a strong balance between performance and efficiency**, delivering competitive captioning accuracy while utilizing far fewer parameters and GFLOPs. Its lightweight design makes it particularly well-suited for real-time or resource-constrained environments. Therefore, MobileNetV2 is adopted as the default visual feature extractor in the remaining experiments due to its superior trade-off between accuracy and computational efficiency.

Table 4.8 shows performance on the MSVD dataset using an LSTM decoder and 1 FPS frame sampling.

Table 4.8: Comparison of CNN Visual Feature Extractors and Motion feature extractor MC3-18 on MSVD (LSTM Decoder, 1 FPS Sampling)

VISUAL EXTRACTOR	TENSOR	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDER	METEOR	ROUGE_L
RESNET 18	1024	76.4	63.0	53.4	43.8	82.9	32.4	65.8
RESNET 50	2560	79.8	68.6	59.7	49.4	98.0	35.7	72.0
RESNET 101	2560	80.8	70.5	61.9	52.9	103.4	35.3	72.0
RESNET 152	2560	82.7	71.2	61.4	52.7	112.8	36.2	71.9

Analysis: In this setup, we observe that **ResNet-101** and **ResNet-152** with Motion feature extractor **MC3-18 (3D CNN)** achieved the highest CIDEr scores (103.4 and 112.8), reflecting their strong capacity to capture fine-grained visual and motion features. However, this comes at the cost of increased computation: both models output 2560-dimensional features and have very high parameter counts and GFLOPs.

Conclusion: While deeper ResNets provide marginally higher accuracy.

4.4 Ablation 3: Decoder Type

In this ablation study, we compare three popular decoder architectures—**LSTM**, **GRU**, and **Transformer**—while keeping all other settings fixed. MobileNetV2 is used as the visual feature extractor, and 30 uniformly spaced frames are used for each video. The objective is to identify which decoder offers the best trade-off between captioning performance and computational efficiency.

- **LSTM:** A widely used sequence model capable of capturing long-term dependencies but relatively heavier in computation due to dual-gated memory.
- **Transformer:** A self-attention-based model that excels in parallelization and long-range modeling, but generally has higher computational overhead.
- **GRU:** A simplified gated mechanism requiring fewer parameters and less memory than LSTM while maintaining similar performance.

Table 4.9: Comparison of Decoder Architectures using MobileNetV2 Features and 30 Uniform Frames (MSVD Dataset)

Decoder	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDEr	METEOR	ROUGE_L	#Params	GFLOPs
LSTM	79.9	67.1	56.8	47.3	86.9	34.5	70.1	14.54M	9.18G
GRU	80.5	69.4	60.3	51.5	90.2	34.6	71.3	14.54M	9.18G
Transformer	76.3	67.9	61.0	43.7	81.1	27.3	64.0	24.00M	9.73G

Conclusion:

While LSTM and GRU share identical parameter counts and computational cost, **GRU outperforms LSTM in every metric**, including BLEU-4 (51.5 vs. 47.3) and CIDEr (90.2 vs. 86.9). GRU’s simpler gating mechanism leads to faster convergence and slightly better generalization while remaining just as efficient. In contrast, the Transformer decoder, despite its powerful attention-based modeling, underperforms in this configuration—likely due to limited training data and smaller batch sizes. It also demands higher parameter count (24M) and compute (9.73G GFLOPs). Hence, GRU emerges as the most practical and performant decoder choice for our video captioning model.

4.5 Summary

This chapter presented a series of comprehensive ablation studies to evaluate the effectiveness of different design choices in the proposed video captioning pipeline. Our analysis leads to the following conclusions:

- **Frame Selection Strategy:** Selecting **30 uniformly spaced frames** consistently provided the best trade-off between temporal coverage and computational cost across both MSVD and MSR-VTT datasets. It captured enough semantic diversity without introducing redundancy, outperforming both low FPS sampling and denser configurations.
- **Visual Feature Extractor:** While deep CNNs like ResNet-152 yielded the highest captioning scores, they also incurred significant computational overhead. In contrast, **MobileNetV2** achieved nearly comparable performance (CIDEr: 90.2 on MSVD) with far fewer parameters and lower GFLOPs, making it the optimal choice for efficiency without sacrificing accuracy.
- **Visual Feature Extractor and Motion Feature Extractor:** While deep CNNs like ResNet-152 and MC3-18 yielded the highest captioning scores (CIDEr: 112.8 on MSVD).

- **Decoder Architecture:** Among LSTM, GRU, and Transformer decoders, the **GRU** consistently outperformed others across BLEU, CIDEr, and METEOR scores, all while maintaining low parameter count and fast convergence. The Transformer decoder underperformed due to its higher complexity and sensitivity to data size in our setup.

Overall, the best-performing and most computationally efficient configuration for our video captioning pipeline is:

- **30 uniformly sampled frames**
- **MobileNetV2** as the CNN backbone
- **GRU** as the decoder

This combination delivers strong captioning performance with significantly reduced computational cost, making it highly suitable for real-time and resource-constrained deployment scenarios.

Chapter 5

Conclusion

This chapter presents a comprehensive conclusion to the work carried out in this thesis. It summarizes the key research contributions, experimental findings, and design choices made throughout the development of an efficient and lightweight Single Sentence Video Captioning (SSVC) framework. The summary provided in Section 5.1 highlights the principal innovations and empirical results obtained using the MSVD and MSR-VTT datasets.

In addition, this chapter outlines future research directions in Section 5.2, focusing on potential enhancements to both model architecture and training strategies. These prospective improvements aim to further increase the system’s captioning accuracy, scalability, and adaptability to real-world and multilingual environments. Together, these discussions provide a roadmap for advancing the capabilities of lightweight video captioning systems in resource-constrained scenarios.

5.1 Summary

This thesis presented a computationally efficient and modular framework for Single Sentence Video Captioning (SSVC), aiming to generate accurate natural language descriptions of video content while maintaining low computational overhead. The proposed architecture was systematically evaluated on two widely-used benchmark datasets—MSVD and MSR-VTT—through a series of controlled experiments and ablation studies.

A key finding of this work was that selecting **30 uniformly spaced frames** per video provided an optimal trade-off between temporal coverage and redundancy, enabling the model to capture the essential visual semantics without excessive computation. In the visual feature extraction stage, **MobileNetV2** demonstrated superior efficiency, achieving performance comparable to deeper models like ResNet-152, while significantly reducing parameter count and inference cost.

On the decoding side, empirical results showed that the **GRU-based decoder** consistently outperformed both LSTM and Transformer-based alternatives under constrained resources. GRU offered better runtime efficiency while maintaining strong caption quality as measured by standard evaluation metrics.

The best-performing configuration—**MobileNetV2 as encoder, 30-frame input**, and a **GRU decoder**—achieved competitive BLEU and CIDEr scores with minimal resource consumption (in terms of parameters and GFLOPs). These results validate the framework’s suitability for real-time and edge deployment scenarios, where lightweight yet accurate video understanding systems are critical.

5.2 Future Work

While the current framework demonstrates promising performance in generating single-sentence video captions using lightweight architectures, there remain several avenues for future enhancement. These potential directions aim to improve both the quality and generalizability of the system across broader tasks and deployment scenarios:

- **Learning-Based Frame Selection:** Instead of uniformly sampling frames, the system could benefit from adaptive frame selection strategies driven by reinforcement learning (e.g., PickNet) or attention-based scoring mechanisms, enabling it to focus on the most informative segments of the video.
- **Multimodal Fusion:** Enriching the visual stream with additional modalities such as audio, speech, or subtitles can enhance the semantic grounding of generated captions—especially in scenarios where spoken content or sound effects carry contextual significance.
- **Hybrid Architectures with Transformers:** While GRU-based decoders offer efficiency, incorporating lightweight Transformer components in a hybrid architecture may improve the fluency and contextual richness of the generated sentences without significantly increasing model complexity.
- **Expansion to Multilingual and Domain-Specific Datasets:** Extending the model to support multilingual captioning or fine-tuning it on domain-specific datasets (e.g., instructional videos, medical footage) can improve its applicability across diverse real-world use cases.

Pursuing these future directions has the potential to significantly enhance the accuracy, adaptability, and real-world usability of the proposed video captioning system, enabling broader deployment across platforms and languages.

Bibliography

- [1] Moloud Abdar, Meenakshi Kollati, Swaraja Kuraparthi, Farhad Pourpanah, Daniel McDuff, Mohammad Ghavamzadeh, Shuicheng Yan, Abdullah Mohamed, Abbas Khosravi, Erik Cambria, and Fatih Porikli. A review of deep learning for video captioning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. Early Access.
- [2] Dan Chen and Bill Dolan. Collecting a large-scale multimodal dataset: The microsoft research video description corpus. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 250–259, 2011.
- [3] Shaoxiang Chen, Qin Jin, Shuicheng Yan, and Alexander Hauptmann. Motion guided spatial attention for video captioning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 75–91, 2020.
- [4] Shaoxiang Chen, Juncheng Li, Qin Jin, and Alexander Hauptmann. Less is more: Picking informative frames for video captioning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 358–373, 2018.
- [5] Xiyang Chen, Hui Fan, Bing Wang, Luc Van Gool, and Radu Timofte. Picknet: Reinforcement learning-based frame sampling for video captioning. In *The IEEE International Conference on Computer Vision (ICCV)*, page 289–298, 2019.
- [6] Jiuxiang Gu, Zhenguo Li, Haoliang Li, Qingxiong Yang, and Jianfei Cai. Hierarchical memory network for video captioning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 205–221, 2020.
- [7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.
- [8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page 770–778, 2016.
- [9] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Feifei, and Jitendra Malik NN. Dense-captioning events in videos. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 706–715, 2017.
- [10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, page 1097–1105, 2012.
- [11] Ying Lin, Tao Mei, Houqiang Li, and Yongdong Zhang. Vtransformer: Efficient video transformer for video captioning. In *Proceedings of the ACM International Conference on Multimedia (ACM MM)*, pages 1406–1414, 2021.

- [12] Yu Liu, Zhaoyang Lu, Jiyang Zhang, and Qiang Yu. A novel key-frames selection framework for comprehensive video summarization. *Multimedia Tools and Applications*, 80(6):8695–8713, 2021.
- [13] Weixiao Pei, Ting Yao, Yanfeng Wang, Chong-Wah Ngo, and Tao Mei. Memory-attended recurrent network for video captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8347–8356, 2019.
- [14] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7008–7024, 2017.
- [15] Anna Rohrbach, Marcus Rohrbach, Ronghang Hu, Trevor Darrell, and Bernt Schiele. Movie description. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 57–67, 2017.
- [16] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Cheung, Liang-Chieh Chu, and Hartwig Adam. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page 4510–4520, 2018.
- [17] Gunnar Aksel Sigurdsson, Gü̈l Varol, Xiaolong Wang, Lorenzo Torresani, Ivan Laptev, and Cordelia Schmid. The charades dataset: Action recognition and temporal localization. In *arXiv preprint arXiv:1604.07422*, 2016.
- [18] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence - video to text. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 4534–4542, 2015.
- [19] Subhashini Venugopalan, Huijuan Xu, Jeff Donahue, Marcus Rohrbach, Raymond Mooney, and Kate Saenko. Translating videos to natural language using deep recurrent neural networks. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 1494–1504, 2015.
- [20] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3156–3164, 2015.
- [21] Da Wang, Qingxiong Jiang, Jinjun Ye, and et al. Vatex: A large-scale, high-quality multilingual dataset for video-and-language research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4580–4589, 2019.
- [22] Jing Xu, Tao Mei, Ting Yao, Yanze Lei, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5288–5296, 2016.

- [23] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning (ICML)*, pages 2048–2057, 2015.
- [24] Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville. Describing videos by exploiting temporal structure. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 4507–4515, 2015.
- [25] HongJiang Zhang, Mohan Kankanhalli, and Stephen W. Smoliar. Automatic partitioning of full-motion video. *Multimedia systems*, 1(1):10–28, 1993.
- [26] Zhengyuan Zhang, Yang Wang, Yue Yu, Linchao Zhu, and Yi Yang. Object relational graph with teacher-recommended learning for video captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13278–13288, 2020.
- [27] Licheng Zhou, Yingbo Zhang, Yue Zhao, and et al. Youcookii: A new large-scale effort for understanding gourmet cooking videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4897–4906, 2018.
- [28] Luowei Zhou, Yingbo Zhou, Jason Corso, Richard Socher, and Caiming Xiong. End-to-end dense video captioning with masked transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8739–8748, 2018.
- [29] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8697–8710, 2018.