Unified Deep Learning Model for Image and Video Caption Generation

Chapter 1: Introduction, Objectives, Background & Survey

1.1 Introduction  
Image and video captioning refers to the process of automatically generating descriptive textual annotations for visual content using machine learning and deep learning techniques. This has significant applications in multimedia retrieval, aiding visually impaired users, autonomous vehicles, surveillance systems, and content management. With the growing availability of multimedia data and the increasing sophistication of deep learning methods, unified models capable of handling both image and video inputs have become crucial.

This project presents a unified deep learning model that uses a shared architecture for captioning both images and videos. The system uses ResNet-50 as a visual encoder and an LSTM-based decoder to generate meaningful captions from features extracted from images or video frames.

1.2 Objectives  
- To develop a deep learning model capable of generating captions for both images and videos using a unified architecture.  
- To utilize ResNet-50 as an encoder for visual feature extraction.  
- To use an LSTM decoder for sequence generation from extracted features.  
- To preprocess and train on standard datasets (Flickr8k for images and MSR-VTT for videos).  
- To evaluate and demonstrate caption generation performance.

1.3 Background  
Conventional captioning models have typically focused on either images or videos, not both. Image captioning usually involves CNN-LSTM models, whereas video captioning requires handling temporal dynamics across frames.

Recent advances in deep learning, particularly the use of transfer learning with pre-trained CNNs (e.g., ResNet-50) and sequence learning using RNNs/LSTMs, have shown impressive performance in caption generation tasks. Combining these methods in a unified model allows shared training and reuse of architecture.

1.4 Literature Survey  
- Show and Tell (Vinyals et al., 2015): Introduced the encoder-decoder framework for image captioning using CNN + LSTM.  
- Sequence to Sequence - Video to Text (Venugopalan et al., 2015): Used sequence learning for video captioning by aggregating features across frames.  
- MSR-VTT Dataset (Xu et al., 2016): Provided a standard benchmark for video captioning.  
- Unified Models: Research increasingly emphasizes model unification to handle multimodal input (image/video) and reduce duplication in model development.

Chapter 2: Proposed System, Methodology, Hardware and Software Details

2.1 Proposed System  
The unified captioning system comprises:  
- Encoder: ResNet-50 extracts deep visual features from input images or sampled video frames.  
- Decoder: An LSTM generates a sentence word-by-word from encoded features.  
- Vocabulary Builder: Tokenizes and indexes captions using a word-to-index mapping.  
- Training Loop: Trains using cross-entropy loss with teacher forcing for better convergence.

2.2 Methodology  
2.2.1 Data Preprocessing  
- Images: Loaded and resized using image\_data\_processing.ipynb. Paired with captions from Flickr8k.  
- Videos: Processed using OpenCV to extract frames. Frames are sampled at intervals to reduce redundancy (video\_data\_processing.ipynb).  
- Captions: Tokenized, padded, and indexed using a custom vocabulary class (vocab.ipynb).

2.2.2 Model Architecture  
- Encoder: Pre-trained ResNet-50 (excluding final classification layer).  
- Decoder: Embedding layer → LSTM → Linear output layer mapping to vocabulary.  
- Training (update\_training.ipynb):  
 - Epoch-wise and batch-wise training loop  
 - Logging loss and accuracy  
 - Saving model checkpoints

2.2.3 Unified Handling  
- A shared training loop accepts either image or frame features, thus allowing both datasets to be used interchangeably.  
- Future extension can allow modality-specific heads for fine-tuning.

2.3 Hardware Requirements  
- Minimum: Google Colab GPU environment  
- CPU: Intel i5 or equivalent  
- RAM: 8 GB+  
- GPU: NVIDIA CUDA-enabled GPU (for local use)

2.4 Software Requirements  
- Python 3.10+  
- Libraries: PyTorch, torchvision, OpenCV, pandas, matplotlib, tqdm  
- Jupyter Notebook (Google Colab preferred)

Chapter 3: Cost Involved

| Resource | Description | Cost |  
|-------------------------|----------------------------------------|----------------|  
| Google Colab (Free GPU)| Used for training and testing | $0 |  
| Dataset Access | Flickr8k and MSR-VTT (open datasets) | $0 |  
| Development Tools | Python, PyTorch, Jupyter | $0 |  
| Local Resources (optional)| PC/laptop + GPU | $800 (approx) |

Total Estimated Cost: $0 (cloud-based); Optional local GPU: ~$800

Chapter 4: Results

The unified model was successfully trained on both image and video data.

4.1 Image Captioning Results  
Example:  
- Input: Image of a man riding a horse.  
- Output Caption: "A man is riding on a horse in the field."

4.2 Video Captioning Results  
Example:  
- Input: Short video showing a dog playing with a ball.  
- Output Caption: "A dog is playing with a ball in the yard."

4.3 Training Performance  
- Training loss consistently decreased across epochs.  
- Final model accuracy for image captioning: ~80%  
- Caption generation quality evaluated using BLEU score (future enhancement).

4.4 Model Features  
- Shared encoder-decoder for both modalities  
- Modular training with logging and checkpointing

Chapter 5: Conclusion

This project demonstrates the feasibility of using a unified deep learning architecture to perform captioning on both images and videos. Leveraging a ResNet-50 encoder and an LSTM decoder, the model effectively generates meaningful captions. Data preprocessing, vocabulary handling, and training were modularly implemented and integrated.

Future work can involve using transformer-based decoders, implementing attention mechanisms, and deploying the model as a real-time captioning API.

Chapter 6: Code Snapshots

Due to space limitations, only key snippets are included:

Encoder-Decoder Architecture:  
# Encoder  
resnet = models.resnet50(pretrained=True)  
modules = list(resnet.children())[:-1]  
self.resnet = nn.Sequential(\*modules)

# Decoder  
self.lstm = nn.LSTM(embed\_size, hidden\_size, num\_layers, batch\_first=True)  
self.linear = nn.Linear(hidden\_size, vocab\_size)

Training Loop:  
for epoch in range(num\_epochs):  
 for i, (images, captions) in enumerate(train\_loader):  
 features = encoder(images)  
 outputs = decoder(features, captions)  
 loss = criterion(outputs, captions[:, 1:])  
 loss.backward()  
 optimizer.step()

Chapter 7: References

1. Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and Tell: A Neural Image Caption Generator. CVPR.  
2. Venugopalan, S., Xu, H., Donahue, J., Rohrbach, M., Mooney, R., & Saenko, K. (2015). Sequence to Sequence - Video to Text. ICCV.  
3. Xu, J., Mei, T., Yao, T., & Rui, Y. (2016). MSR-VTT: A Large Video Description Dataset for Bridging Video and Language. CVPR.  
4. PyTorch Documentation - https://pytorch.org/docs/stable/index.html  
5. Flickr8k Dataset - https://forms.illinois.edu/sec/1713398  
6. MSR-VTT Dataset - http://ms-multimedia-challenge.com/2021/dataset