# 274C FINAL REPORT
# SEQ2SEQ: VIDEO CAPTIONING

**Jinhwa Kim, Jonathan Dedinata, Long Vu**
Department of Computer Science
University of California, Irvine
{86727408, 64003232, 37289444}
{jinhwak,jdedinat,xlvu}@uci.edu

## ABSTRACT

Video captioning is a multi-modal task with the goal of captioning a video with a sentence that matches what is happening within the video. To achieve this, first, we pre-process all the videos and text data, then, we train the model along with an NLP algorithm to get the desired output. We explored several Seq2Seq-based models with various types of encoders and decoders as well as implementing attention mechanism to improve the performance of the model. Various decoder algorithms are also experimented with, such as the greedy algorithm and the beam search algorithm. We conduct quantitative and qualitative analysis and show the performance results of the models. Finally, we present the problems of the evaluation metrics that are widely used in the video captioning task with examples from our analysis.

## 1 INTRODUCTION

Video captioning is one of multi-modal learning tasks which handles image features and text features at the same time. While image captioning, which is a very similar task to video captioning, shows a decent result, video captioning still shows a poor performance. The goal of video captioning is to produce a caption describing or summarising the content of any given videos. Multiple other research projects considerably overlap with video captioning include text-to-video retrieval, video captioning, video question answering, and video moment retrieval. Most of the related projects explored with the recurrent network architecture, so in our project, we will also utilize the same model. These types of projects start with feature extraction of the video inputs using pre-trained vision models and pre-trained language models, then apply a multi-modal fusion to produce an output. Current existing approaches using these types of architecture has achieved considerable success and we aim to replicate these types of models and compare their performance with our model.

In this paper, we implemented a Seq2Seq-based model, with various types of encoders and decoders for our architecture. Our architecture starts by parsing the input videos into frames and turning them into a latent representation with a feature extraction model. The outputs of the feature extraction model are then used as inputs for the encoder, and the output of the encoder are used as inputs for the decoder. In our architecture, we experimented with different models for feature extraction, encoders, and decoders. For our feature extraction models, we are using VGG16 Simonyan & Zisserman (2015) and ResNet He et al. (2015). As for our overall Seq2Seq architecture, we will be referring to architectures used in Venugopalan et al. (2015). As for the pre-trained model, we will be comparing the results produced from Lin et al. (2021) (state-of-the-art performance in 2021) with our own models. Initially, we tried to fine-tune pre-trained models such as UniVL Luo et al. (2020) and ClipBERT Lei et al. (2021) on our dataset but due to computational limitations, we were unable to do this. On each of these models, we will be evaluating their performance on the MSVD Chen & Dolan (2011) dataset. We aim to compare their performance on the BLEU metric and METEOR metric and see how robust the models are when evaluated on different metrics as well as how accurate the metric scoring system is.

## 2 Related Works

**Video Captioning** Early work on video captioning is based on the template and generates a description using the templates (Krishnamoorthy et al., 2013). Venugopalan et al. proposed a unified Seq2Seq model which aims to jointly learn both encoding and decoding step. The model uses a CNN model for extracting the image features and a LSTM-decoder decodes this information into a sequence of tokens that describes a video. Sun et al. shows a unified encoder-decoder model using BERT model that can pre-train the model with large-dataset and do transfer-learning depending on each task. Recently, the Microsoft Research Team (Luo et al., 2020) proposes a unified-task model that is based on the Transformer model and train with five different objectives to handle several different tasks with a same model such as video captioning and Action Segmentation. These projects are also based on other projects such as the COCO project (Lin et al., 2014). We will also be taking inspiration for our base model from Venugopalan et al. (2015).

**Multi Modal Learning** As the state of deep learning models advances, a lot of applications such as image recognition, recommendations, and Natural Language Processing (NLP) show huge success and get impressive outcomes not only on each domain but also across multiple domains(Summaira et al., 2021). Multimodal deep learning has been studied from various domain features and applied to several applications such as Image Captioning (Hu et al., 2021; Pan et al., 2020) and Video captioning (Luo et al., 2020; Ramanishka et al., 2017; Lei et al., 2021) which are based on both of the image features from images or frames of video and text features for the captioning. Furthermore, Lin et al. (2021) proposed a more elaborate version of architecture that uses multimodal models as well as well using a sparse attention mask instead of a full attention mask to improve the prediction result. These models have achieved successful results and can produce reliable captions for videos.

## 3 Data

There exists several datasets for video captioning task that we are trying to solve. The first and the most common one is Microsoft Research Video Description Corpus (MSVD) (Chen & Dolan, 2011). This dataset contains multiple video snippets, with each snippet having multiple short descriptions about the content of the video. Other common datasets include YouCook Zhou et al. (2017), which contains videos describing various cooking actions, and MSR-VTT dataset introduced by Xu et al. (2016), which contains a much larger scale dataset, with more than 200 thousands clip-description pairs. Due to the scope of our project and limited computational resources, we choose to focus only on MSVD dataset.

| Video | Ground-truth |
|---|---|
|  | (1) two bear cubs are digging in the dirt. (2) a man is watching two bears. (3) a man is sitting while two bears play around. |
|  | (1) a man is lifting weights. (2) doing gyms. (3) a man is doing exercise. |
|   | (1) the man is singing and playing the guitar. (2) a man is playing guitar. (3) a band is playing guitar on television. |

Table 1: Examples of MSVD dataset. Each video is paired with multiple captions.

### 3.1 MSVD DATASET

The MSVD dataset contains 1970 short Youtube clips that shows various types of activity, ranging from simple actions such as eating, cooking, or playing instruments, to more complex scenarios such as man falling off a bike, or a girl having an accident while doing gymnastics, etc. The clips have an average duration of less than 10 seconds, and each clip contains about 41 descriptions that describe the scenario shown in the clip.

We use MSVD dataset for both training and testing our model. Followed the data split of Venugopalan et al. (2015), we extract the first 1200 videos for the training set, the next 100 videos for the validation set, and the final 670 videos for the test set. Table 1 shows some examples of video and caption pairs in the MSVD dataset. As can be seen from the table, one video can have several different sentences describing it. While it makes the description to have more variety, it also makes the model learning all different descriptions for the same video more difficult.

### 3.2 DATA PREPROCESSING

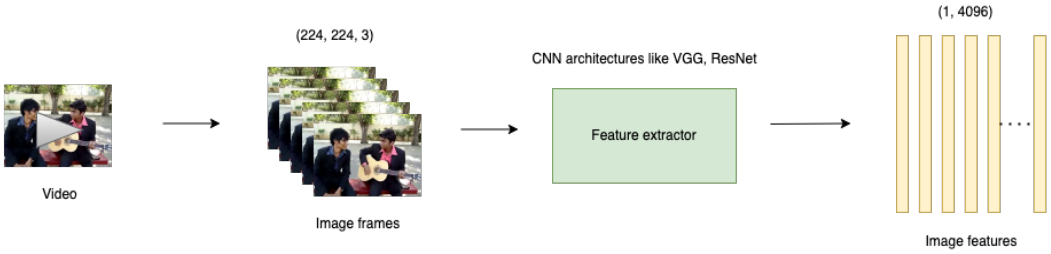



Figure 1: Video Preprocessing Pipeline

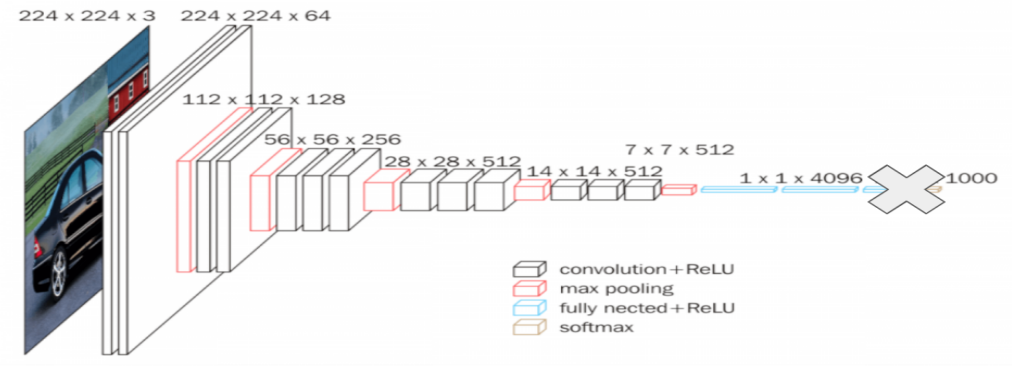



Figure 2: VGG Feature extraction. Figure from Simonyan & Zisserman (2015)

**Video Feature Extraction** Figure 1 shows our pipeline to preprocess the videos data for training. A video consists of several frames with the frequency of 24-25 frames per second. To extract the feature representation for each video, first, we split the video into a series of 80 frames. Based on the success of transfer learning, we use pre-trained CNN models like VGG and ResNet (Simonyan & Zisserman, 2015; He et al., 2015) to extract feature for each image frame. Figure 2 shows our approach for feature extraction.

The original pre-trained model is trained to classify 1000 ImageNet categories, with the last two layers mapping the all image features into different categories. Since we only care about image features, we can remove the last two layers of the pre-trained model, and extract the modified model output. For VGG, with each image of size 224x224x3 (HxWxC), the feature extracted will have the size of 1x4096. Since we have 80 frames in each video, the final feature representation for each video will have the size of 8x4096. All of this process are done before the training step: we save the video feature as a `numpy` array for each video and load the `numpy` array of corresponding video

feature at training time. This is used to reduce our training time to a reasonable level. Without this step, training the whole end-to-end model could take a few days (processing videos is much more computationally expensive than processing image).

**Text Preprocessing** To train a sequence to sequence model, the text caption has to be processed into a sequence of word embeddings. To do this, first, we convert the text caption to lowercase and tokenize the caption. The caption is then padded with beginning-of-sentence `<bos>` and end-of-sentence `<eos>` tokens. Each token is then converted into a list of indices, which are taken from the vocabulary of all words that appear in the captions. This list of indices, when went through the embedding layer, will output a sequence of embedding vectors for each corresponding token.

# 4 APPROACH

## 4.1 SEQ2SEQ MODEL

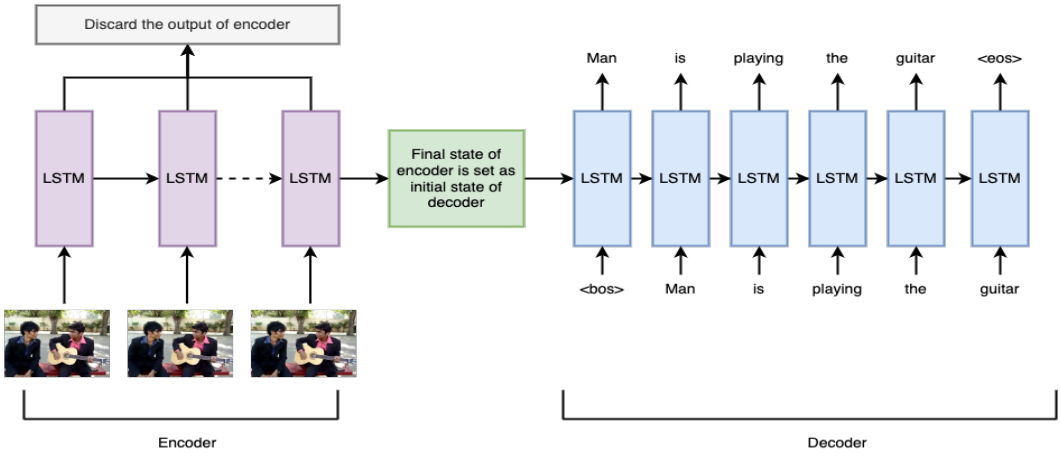



Figure 3: Baseline Model Architecture

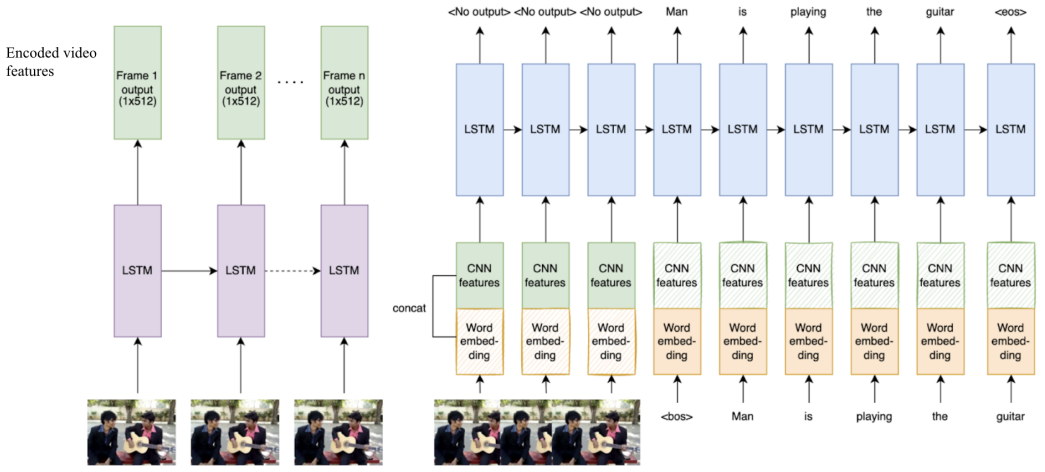
Because the video captioning has to capture the information from the video and generate captions based on the captured information, we use Seq2Seq model as our baseline model architecture. If we have a video with extracted features from the pre-trained model as $(V_1, ..., V_n)$, an encoder encodes the video features by $h_t = f(W^{(hh)}h_{t-1} + W^{(hx)}V_t)$ and finally generates a fixed-size vector that has the information of all frames. A decoder is initialized with this context vector first and generates captions that describe the video $(y_1, ..., y_m)$ by $h_t = f(W^{(hh)}h_{t-1}), y_t = softmax(Wh_t)$. Our task is to maximize the probability $P(y_1, ..., y_m|V_1, ..., V_n)$. Figure 3 shows the architecture of our baseline model.

## 4.2 ATTENTION MODEL

Because our baseline model encodes all the information of video frames into ome hidden state vector and discards all outputs of encoder, decoder considers all the information of a video equally and generates captions.As a video is consist of several different frames and each frame could have a different information, managing all the information equally might make the model confuse and limit the generations. We adapt a soft-attention mechanism (Yao et al., 2015) that allows the decoder to decide the weight of each frame and generates a context vector by doing weighted sum of temporal features. A context vector is calculated by $\sum_{n=1}^{n} \alpha_i^{(t)} V_i$ where $\sum \alpha_i^{(t)} = 1$

## 4.3 UNIFIED SEQ2SEQ MODEL

In the previous seq2seq model described in section 4.1, the hidden state of all layers for the video encoder are discarded. Only the hidden state of the last frame is used for the input of the decoder




(a) (b)

Figure 4: Unified Model Encoder-Decoder Architecture

state. This poses a problem: previous layers' hidden states containing important information might be discarded. Since different frames might contain different activities that could affect the final video description, we don't want this to happen.

Instead of having two independent LSTMs for encoder and decoder, inspired from Venugopalan et al. (2015), we use a single unified LSTM for both encoding and decoding stage. This is achieved by having having two LSTMs stacked on each other. The first LSTM primarily serves as the video encoder, which the main function is extracting the video representation at each time step (Figure 4a). Instead of discarding the output of all frames, these hidden states are concatenated with the word embedding to form the input for the unified LSTM (Figure 4b). The first several steps of the unified model receives the hidden representations from the first LSTM and concatenates it with an empty word embedding (null input word). After all frames of the video are encoded, now the LSTM enters the decoding stage. To mark the beginning of this stage, beginning-of-sentence token `<bos>` are added as the text input of the LSTM. At this stage, the model maximizes the likelihood of the generated sentence given the previous word and the video latent representation. The problem is formulated as:

$$\theta^* = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{n} \log p(y_t | h_{t-1} y_{t-1}; \theta) \tag{1}$$

with $y_t$ is the output at step $t$, $h_t$ is the hidden state of the LSTM at time step t, $\theta$ is the parameter of the model. We try to find the best $\theta$ that maximizes the output sentence probability.

## 5 Experiments

### 5.1 Evaluation

For the quantitative evaluation of the models, we use BLEU (Papineni et al., 2002) and METEOR (Denkowski & Lavie, 2014) metrics. These metrics calculate the overlapped tokens or n-grams between the generated sentences and ground-truth.

For BLEU, we used the method described on Papineni et al. (2002). Mathematically, it can be defined as:

$$BLEU = \min(1, \exp(1 - \frac{reference\_length}{output\_length}))(\Pi_{i=1}^{n} precision_i)^{1/n}$$
$$precision_i = \frac{\Sigma_{snt \in Cand-Corpus} \Sigma_{i \in snt} \min(m^i_{cand}, m^i_{ref})}{w^i_t = \Sigma_{snt' \in Cand-Corpus} \Sigma_{i' \in snt', m^{i'}_{cand}}} \tag{2}$$

An example of how a BLEU score is calculated can be seen below:

**Reference**: the cat is on the mat

**Candidate**: the the the cat mat

Using $i = 1$, showcasing $precision_1$, the total number total number of words that overlap is 5, on words "the" 3 times, "cat" once, and "mat" once. Then, we see the number of words that overlaps from the Candidate to the Reference, which is 4, "the" 2 times, "cat" once, and "mat" once. The $precision_1$ value would then be $\frac{4}{5} = 0.8$. Calculcating this way up to $n = 4$, we get the BLEU score $0.55$ for the above reference and candidate sentences.In our usage of BLEU metrics, we used the values $n = [1, 2, 3, 4]$.

Formula 3 represents the equation of METEOR score.

$$P = \frac{m}{w_t}, R = \frac{m}{w_r}, F_{mean} = \frac{10PR}{R + 9P}, p = 0.5 \left(\frac{c}{u_m}\right)^3$$
$$METEOR = F_{mean}(1 - p) \tag{3}$$

In the METEOR formula, $P$ is the unigram precision, $m$ is the number of unigrams in the candidate translation that are also found in the reference translation, and $w_t$ is the number of unigrams in the candidate translation. $R$ is the unigram recall and $w_r$ is the number of unigrams in the reference translation. $F_{mean}$ is the harmonic mean and $p$ is the penalty. Using the reference and candidate sentence examples from the BLEU section, we get a METEOR score of 34.0.

### 5.2 Experimental Setup

| Model | LSTM Hidden size | Batch size | Epoch | Loss | Optimizer |
|---|---|---|---|---|---|
| Seq2Seq | 512 | 320 | 30 | CrossEntropy | Adam |
| Seq2Seq + Attn | 512 | 30 | 30 | CrossEntropy | Adam |
| Unified Seq2Seq | 512 | 32 | 300 | CrossEntropy | Adam |

Table 2: Experimental Settings for Training

For our setup, we used Google Colab Pro as our main computational resource. Due to the limitations of Google Colab Pro, we were unable to run pre-trained models such as UniVL Luo et al. (2020) and SwinBERT Lin et al. (2021). For the basic Seq2Seq model and attention model, we used a TensorFlow/Keras implementation with the ADAM optimizer and the CrossEntropy loss function. For our unified Seq2Seq LSTM models, we used a PyTorch implementation and also used the ADAM optimizer and the CrossEntropy loss function. We also add DropOut layer to our architecture to prevent overfitting. Table 2 shows the detail of our training settings for each model type.

### 5.3 Quantitative Analysis

For the quantitative evaluation, we use our test dataset to calculate BLEU and METEOR score of each instance and average them. Table 3 shows the performance result of each model. Seq2Seq model shows a decent result while unified Seq2Seq model and attention model do not give improvements compared to baseline model even though these models have low training loss when compared to baseline model. For analyzing this result, we conduct qualitative analysis on Section 5.4 and check the quantitative result.

| Model | BLEU-1 | BLEU-2 | BLEU-3 | BLEU-4 | METEOR |
|---|---|---|---|---|---|
| Seq2Seq (baseline) | 78.8 | 57.2 | 42.1 | 19.5 | 56.3 |
| Seq2Seq + Attn (Beam) | 74.7 | 52.4 | 37.0 | 16.4 | 52.4 |
| Seq2Seq + Attn (topk) | 70.2 | 45.3 | 30.2 | 13.5 | 48.8 |
| Unified Seq2Seq (VGG) | - | - | - | - | 31.7 |
| Unified Seq2Seq (ResNet) | 47.5 | 28.8 | 17.9 | 10.6 | 38.9 |
| SwinBERT | - | - | - | 58.2 | 41.3 |

Table 3: Model Performance Results. Unified Seq2Seq VGG and SwinBERT's author didn't perform their evaluation with BLEU so we don't include them.

From our quantitative analysis, we can find the limitation of the METEOR and BLEU metrics that are widely used in a Video-Captioning task. Table 4 shows one of ground-truth sentences and generated sentences from our models and the METEOR score of each generation. In the first example, we can see a sentence "the men are playing" is quite different from the ground-truth, but a sentence "a man and woman are dancing" has information of "dancing". However, the second sentence has lower METEOR score than first sentence. In addition, the second example shows that METEOR score does not consider semantic relationship between ground-truth and generated sentences. Because the METEOR and BLEU only consider the number of overlapped words, if the model generates some sentences with a template (e.g. is -ing ), it gets higher score. Because the baseline model tends to generate the sentence with similar template with ground-truth, it could get higher score than other models.

| Model | Caption | METEOR |
|---|---|---|
| Ground-Truth | some monkeis are dancing | 100.0 |
| Model Generation | a man and woman are dancing | 56.0 |
| | the men are playing | 60.0 |
| Ground-Truth | a man and boy are rock climbing | 100.0 |
| Model Generation | a man is talking | 27.9 |
| | a man is going down | 27.0 |

Table 4: Model Caption Generation and Performance Result

In regards to current more advanced models for video captioning, Lin et al. (2021) achieved a BLEU-4 score of 58.2 and a METEOR score of 41.3 on testing data for their video captioning task on MSVD. Their BLEU score is significantly higher than ours, which we suspect is because their model uses more sophisticated methods for feature extraction and training, which in their case uses VidSwin. However, our models achieved higher METEOR scores than SwinBERT, which can be attributed to how unreliable METEOR scoring can be. Additionally, other state-of-the-art models also achieved high BLEU-4 scores ranging around 46 to 56 and METEOR scores ranging 33 to 36 Lin et al. (2021). Another explanation on why our model achieves higher METEOR score than other models is because of the difference in the METEOR score calculation methods. In our usage of METEOR, we used the NLTK library implementation of METEOR score, whereas the other models used the implementation from Denkowski & Lavie (2014).

### 5.4 Qualitative Analysis

For qualitative evaluation of our models, we check the cases that have lower/higher variance of performance over the models. Table 5 shows the example cases for that. We can find that all of models can generate a description for a video related to cooking. While the baseline model generates very simple and short sentence, other models tends to generate longer sentences and the unified model tends to generate much longer and elaborated sentences compared to the other models. Additionally, the generated sentences are not necessarily close to ground truth for the unified model. Further inspecting the generated caption shows that the unified model performs well on videos that has more complex scenarios and more complicated captions as their ground truth.

For evaluating our attention model, we check the weight matrix each time a token is generated and see whether the decoder actually pays attention to the key frames for generating the corresponding token. Figure 5 shows the examples of attention weight results. The figure shows each frame image

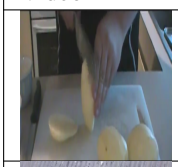
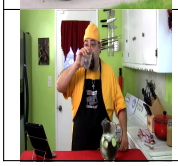
| Video | Model | Generation |
|---|---|---|
|  | Ground-truth | a man is cutting potatoes |
| | Baseline | a woman is cutting a potato |
| | Att-model (beam search) | a man is slicing a potato |
| | Att-model (topk) | a person is slicing a potato |
| | Unified-model | someone is chopping up a tomato into small pieces |
|  | Ground-truth | a woman and two girls are taking a horse cart ride |
| | Baseline | a girl is riding a horse |
| | Att-model(beam search) | a man is riding in the garden |
| | Att-model (topk) | the man went up on the grass |
| | Unified-model | a man driving a car runs into the side of a parked car in a parking lot |
|  | Ground-truth | a man drinks from a glass |
| | Baseline | a man is a a man |
| | Att-model (beam search) | a man is drinking |
| | Att-model (topk) | the man is drinking |
| | Unified-model | a man is drinking a clear liquid from a glass |

Table 5: Model Caption Generation Results

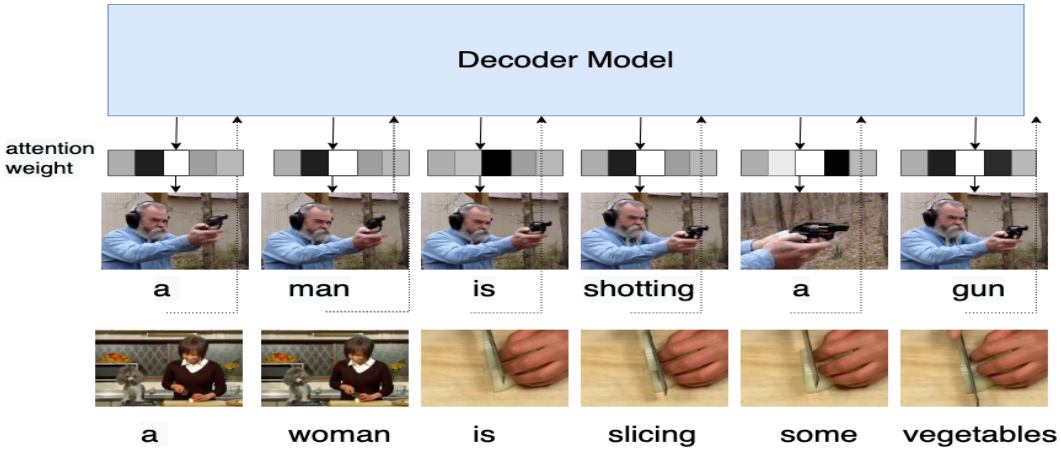



Figure 5: Attention Result Examples

that the model has a maximum attention weight for generating each token. In the first example, we can see that when generating a ”man” token, it gives more weight to frames that the man appears, while focusing on a frame that a gun appears when generating ”a” before a ”gun” token. Also, we can see that for generating tokens for ”a”, ”woman”, it has higher attention weight on the frames that the woman appears, while focusing on frames that the woman’s hand and slicing the onion for generating tokens of ”is”, ”slicing”, ”some”, ”vegetables”. These examples show that our attention model can focus more on important and related information and generates corresponding tokens.

## 6 Conclusion

We implemented several different models for the video captioning task and compare the results over the models. We found that basic Seq2Seq model shows higher performance on quantitative evaluation compared to other models. However, in the qualitative evaluation, we can find that Seq2Seq model has very similar format between the generations which is too simple to express the contents of video. On the other hand, other improved models such as attention model and unified model show more elaborated caption than baseline model. From these results, we can say that adding embedding and more LSTM layers or adding attention mechanism is helpful to generate better captions and the METEOR and BLEU score has some limitations to evaluate the quality of model well. Because we

only use LSTM model architecture as we have limit of computational resources, we can improve our result more if we can train our model with pre-trained model like BERT and Transformer. In addition, for checking the effect of attention mechanism, if we compare the results with long-sequence video, the effect of attention could be more effective.

## 7 ACKNOWLEDGEMENT

We divide our works like this: Jinhwa Kim is responsible for implementing baseline model and attention model. Jonathan Dedinata is responsible for surveying previous works and implementing pre-trained model. Long Vu is responsible for implementing unified model and pre-processing the data. We divided all the work equally and did everything together from the survey to the analysis of the results.

## REFERENCES

David L. Chen and William B. Dolan. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL-2011)*, Portland, OR, June 2011.

Michael Denkowski and Alon Lavie. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*, pp. 376–380, 2014.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385.

Xiaowei Hu, Zhe Gan, Jianfeng Wang, Zhengyuan Yang, Zicheng Liu, Yumao Lu, and Lijuan Wang. Scaling up vision-language pre-training for image captioning. *CoRR*, abs/2111.12233, 2021. URL https://arxiv.org/abs/2111.12233.

Niveda Krishnamoorthy, Girish Malkarnenkar, Raymond Mooney, Kate Saenko, and Sergio Guadarrama. Generating natural-language video descriptions using text-mined knowledge. In *Proceedings of the Workshop on Vision and Natural Language Processing*, pp. 10–19, Atlanta, Georgia, June 2013. Association for Computational Linguistics. URL https://aclanthology.org/W13-1302.

Jie Lei, Linjie Li, Luowei Zhou, Zhe Gan, Tamara L. Berg, Mohit Bansal, and Jingjing Liu. Less is more: Clipbert for video-and-language learning via sparse sampling. *CoRR*, abs/2102.06183, 2021. URL https://arxiv.org/abs/2102.06183.

Kevin Lin, Linjie Li, Chung-Ching Lin, Faisal Ahmed, Zhe Gan, Zicheng Liu, Yumao Lu, and Lijuan Wang. Swinbert: End-to-end transformers with sparse attention for video captioning. *CoRR*, abs/2111.13196, 2021. URL https://arxiv.org/abs/2111.13196.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. URL http://arxiv.org/abs/1405.0312.

Huaishao Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Jason Li, Taroon Bharti, and Ming Zhou. Univl: A unified video and language pre-training model for multimodal understanding and generation. *arXiv preprint arXiv:2002.06353*, 2020.

Yingwei Pan, Ting Yao, Yehao Li, and Tao Mei. X-linear attention networks for image captioning. *CoRR*, abs/2003.14080, 2020. URL https://arxiv.org/abs/2003.14080.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 311–318, 2002.

Vasili Ramanishka, Abir Das, Jianming Zhang, and Kate Saenko. Top-down visual saliency guided by captions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1409.1556.

Jabeen Summaira, Xi Li, Amin Muhammad Shoib, Songyuan Li, and Jabbar Abdul. Recent advances and trends in multimodal deep learning: A review. *CoRR*, abs/2105.11087, 2021. URL https://arxiv.org/abs/2105.11087.

Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. *CoRR*, abs/1904.01766, 2019. URL http://arxiv.org/abs/1904.01766.

Subhashini Venugopalan, Marcus Rohrbach, Jeff Donahue, Raymond J. Mooney, Trevor Darrell, and Kate Saenko. Sequence to sequence - video to text. *CoRR*, abs/1505.00487, 2015. URL http://arxiv.org/abs/1505.00487.

Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. 2016. doi: 10.1109/CVPR.2016.571.

Li Yao, Atousa Torabi, Kyunghyun Cho, Nicolas Ballas, Christopher Pal, Hugo Larochelle, and Aaron Courville. Describing videos by exploiting temporal structure, 2015. URL https://arxiv.org/abs/1502.08029.

Luowei Zhou, Chenliang Xu, and Jason J. Corso. Procnets: Learning to segment procedures in untrimmed and unconstrained videos. *CoRR*, abs/1703.09788, 2017. URL http://arxiv.org/abs/1703.09788.