Dense Video Captioning

Srikumar Brundavanam[1] **Mukul Ganwal**[1] **David Ologan** [1] **Shang Shi**[1]
Carnegie Mellon University
Department of Mechanical Engineering
{vbrundav,mganwal,dologan,shangs}@andrew.cmu.edu

Abstract

In recent years, there has been a large research focus on dense video captioning. Video captioning has applications in many fields such as autonomous driving, video surveillance and creating captions for those with visual impairments. As such, a novel approach is proposed by employing language models (LM) to help to semantically align video features in an attempt to improve the overall performance of dense video captioning. The baseline model for comparison, End-to-end dense video captioning with parallel decoding (PDVC) [1], produced strong results compared to many state of the art video captioning frameworks. PDVC is trained on the ActivityNet and YouCook2 datasets but for this study only YouCook2 was used due to computational capacity limits. We propose the incorporation of semantic alignment through the addition of a tuner network before the video features are passed through the PDVC framework. Ablations were ran for different tuner architectures and overall, the modified PDVC framework outperformed the baseline PDVC in many evaluation metrics. Promising future extensions with Semantic alignment and Dense Video Captioning remain with its application to larger and more comprehensive data sets. The code base for this project can be found at: DenseVideoCaptioning

1 Introduction

Video captioning has been a large focus of research in recent years. An accurate video captioning model has wide-reaching implications in a variety of fields. It can aid those with visual impairments, improve self-driving cars, refine video surveillance [2] and even automatically label data. In order to improve the current state of the art (SOTA) dense video captioning models, the addition of semantic alignment holds much promise. Semantic alignment refers to the process of aligning the visual and caption features such that they are similar in the embedding space. In models like PDVC, only video features are passed into the framework during training and inference. Therefore, we propose a system that can instill caption information from a pretrained LM into the video features in order to semantically align them with their corresponding captions. To do so, a video feature tuner is trained through a contrastive loss with caption embeddings generated through a pre-trained LM. A more in-depth description of this training procedure is discussed in section 5. A similar idea is explored in CLIP [3] as discussed section 2.

2 Literature Review

SWINBERT: End-to-End Transformers with Sparse Attention for Video Captioning - A new end-end fully transformer-based architecture called SWINBERT [4] for video captioning is proposed. The SWINBERT model is a video-based pure-Transformer architecture designed for caption generation. It consists of two modules: Video Swine Transformer (VidSwin) and Multimodal Transformer Encoder. VidSwin is used to extract spatial-temporal video representations from raw

video frames, while the Multimodal Transformer Encoder takes the video representations and outputs a natural language sentence through seq2seq generation. The VidSwin module tokenizes the grid features of the raw video frames along the channel dimension to generate video tokens. During training, the Multimodal Transformer Encoder module performs seq2seq generation to form a natural language sentence, with textual and visual modalities inputs, and uses a causal self-attention mask to simulate a uni-directional seq2seq generation process. A percentage of word tokens are masked and the multimodal transformer is tasked with predicting the masked tokens with the help of video and other word tokens. All textual tokens have full attention to the video tokens. This helps with improved training and improves the performance of the longer videos. However, during inference, the SWINEBERT only takes in video inputs and generates an output natural language sentence. Not surprisingly, using multimodal transformer architecture, SWINEBERT achieves better performance than the previous state-of-the-art methods by a large margin.

End-to-end dense video captioning with parallel decoding - A dense video captioning framework with parallel decoding (PDVC) [1] is proposed. Previous methods separate the task into event localization and captioning forming a two-stage pipeline. This causes issues such as the downstream captioning task being highly dependent on the quality of the generated event proposals. PDVC avoids this by feeding the enhanced representations of event queries into the localization and captioning head in parallel, creating a deep connection between the two tasks allowing them to be mutually optimized. The overall model first adopts a pre-trained video feature extractor and transformer encoder to obtain frame-level features. The features are sent through a transformer decoder to output an event counter which ensures the correct amount of events predicted, localization head, and caption head. There exists room for potential improvement on top of PDVC such as improving the video features being passed into the transformer decoder with through semantic alignment.

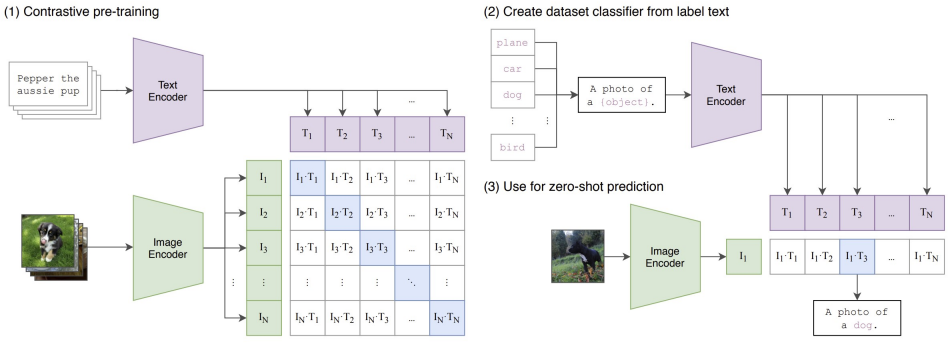

Figure 1: CLIP Framework

Contrastive Language-Image Pre-training - CLIP is a powerful model that combines a large-scale transformer-based language model with a convolutional neural network (CNN) for visual processing. CLIP is centered on learning perception from supervision contained in natural language [3]. Learning from natural language has several potential strengths over other training methods. As it does not require annotations to be in a classic "machine learning compatible format", it is much easier to scale natural language supervision for image classification compared to standard crowd-sourced labeling. A major motivation for natural language supervision is the large quantities of data available publicly on the internet. A new dataset of 400 million image-text pairs was collected from publicly available internet sources. Figure 1 below shows the summary of the current approach. CLIP jointly trains an image and text encoder for the occurrence of possible image-text pairings across a batch. During testing, the learned text encoder synthesizes a zero-shot linear classifier by embedding the names or descriptions of the target dataset's classes. Since CLIP trains a text and image encoder simultaneously to align their embeddings, its text encoder is ideal for our purposes.

3 Baseline Model

The baseline model selected for this project is PDVC [1]. PDVC decodes frame features extracted from a Vision Transformer, into an event set with their respective locations and captions. Prior dense video captioning methods utilize a two-stage "localize-then-describe" pipeline. However, the

performance of two-stage methods limit the mutual promotion of these two sub-tasks due to its reliability on the quality of the generated event proposals. To tackle this issue, PDVC aims to directly exploit inter-task association at the feature level by applying localization and captioning heads in parallel. PDVC is further improved by adding a novel event counter to estimate the number of events in the video enabling PDVC to precisely segment the video into several event pieces while avoiding the information missing along with unreliable event number estimation. This unreliable estimation is what causes redundant caption generation. These attributes make PDVC a novel method over existing dense captioning methods. When selecting a baseline, two main factors are considered. First, a competitive video captioning framework with SOTA performance is desired. Second, a framework where a LM can be introduced to perform semantic alignment is required. Through literature review, PDVC and SWINBERT stood out in particular. However, in SWINBERT, the caption labels are already being used during training alongside the video features. In PDVC on the other hand, only the video features are being used allowing for potential improvements on the framework.

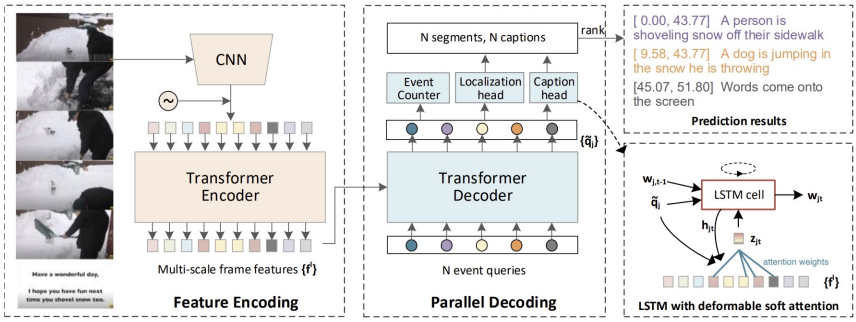

Figure 2: PDVC Framework [1]

Figure 2 shows the overview of the proposed PDVC. A deformable transformer with an encoder-decoder structure is adopted to capture the inter-frame, inter-event, and event-frame interactions by attention mechanism and produce a set of event query features. Next two parallel prediction heads predict the captions for each event query simultaneously. The transformer decoder is stacked with an event counter to further predict the number of final events. The final results are obtained by selecting events with high confidence to ensure a complete and coherent story.

3.1 Deformable Transformer

Deformable Transformer is an encoder-decoder architecture based on multi-scale deformable attention (MSDAtt). MSDAtt is useful for mitigating the slow convergence problem of self-attention in Transformers when processing image feature maps. For a given multi-scale set of feature maps, MSDAtt outputs a context vector by the weighted sum of sampling points across feature maps.

$$MSDAtt(q_i, p_j, X) = \sum_{l=1}^{L} \sum_{k=1}^{K} A_{jlk} W x^l_{\tilde{p}_{jlk}}$$

$$\tilde{p}_{jlk} = \phi(p_{ij}) + \Delta p_{jkl}$$

where $\tilde{p}_{jkl}$ and A_{jkl} are the position and attention weights of the k-th sampled key at the l-th scale for the j-th query element, respectively. W is the projection matrix for key elements. ϕ_l projects normalized reference points into the feature map at l-th level. δp_{jkl} are sampling offsets w.r.t. $\phi_l(p_j)$. By incorporating deformable attention modules, the Deformable Transformer supplants the self-attention modules in the Transformer encoder and the cross-attention modules in the Transformer decoder. This alteration facilitates improved representation ability and faster convergence rates for object detection.

3.2 Feature Encoding

Firstly, a pre-trained video feature extractor and transformer encoder are used to obtain a sequence of frame-level features. To extract rich spatiotemporal features from a video, PDVC uses a pre-trained

action recognition network, such as C3D or TSN, to extract features. Next, the feature map's temporal dimension is interpolated and rescaled to a fixed number for batch processing. Then, L temporal convolutional layers are introduced to generate feature sequences at multiple resolutions, ranging from T to $T/2^L$. To leverage multi-scale features in predicting multi-scale events, the multi-scale frame features are passed along with their positional embedding, into a deformable transformer encoder. This captures the frame-frame relationship across different scales.

3.3 Parallel Encoding

The decoding network consists of a deformable transformer decoder and three parallel heads, a captioning head that does caption generation, a localization head to predict event boundaries with confidence scores, and an event counter to predict a suitable event number.

Localization Head The Localization head is responsible for predicting the 2D relative offsets (center and length) of the ground-truth segment with respect to the reference point, as well as generating the foreground confidence of each event query through box prediction and binary classification respectively. Both these tasks are performed using a multi-layer perceptron. Once this process is complete, the Localization head outputs a set of tuples to represent the detected events.

Captioning Head PDVC provides both a lightweight and standard captioning head. The lightweight head feeds the event query $\tilde{q}_j$ into a vanilla LSTM at each timestamp. Then, the word w_{jt} is predicted by a fully connected layer followed by a softmax activation over the hidden state h_{jt}. In video captioning, Soft attention (SA) [5] is a popular component that can flexibly assign significance to each frame during the word generation process. The conventional two-stage approaches [6] utilizes event boundaries to align event segments and their corresponding captions, which poses a challenge for our captioning model which lacks access to such boundaries. Consequently, it becomes harder to establish connections between words and frames.

This issue is alleviated with the deformable soft attention (DSA) mechanism, which directs soft attention weights towards a limited region around reference points. The DSA generates K sampling points when generating $t-th$ word w_t. These sampling points are conditioned on both language query of hidden state h_{jt} and event query $\tilde{q}_j$. In soft attention, the key/value is represented by K×L sampling points, while the query is represented by $[h_{jt}, \tilde{q}_j]$. Since the sampling points are distributed near the reference point $\tilde{p}_j$, the output features z_{jt} of DSA are limited to attending on a relatively small region. The LSTM takes in the concatenation of the context z_{jt}, event query $\tilde{q}_j$, and the previous words $w_{j,t-1}$ as input. Then the probability of the next word w_{jt} is obtained through a FC layer with softmax activation. As the LSTM evolves, it generates a sentence S_j where each element is the word multiplied by the length of each sentence M_j.

Event Counter The event counter detects the number of events in a video, which is an essential indicator of dense captioning. Counting too many events causes redundant captions, poor readability and too few detected events means an incomplete story. The event counter comprises a max-pooling layer and an FC layer with softmax activation. It first compresses the most important information from the event queries $\{\tilde{q}_j\}$ into a global feature vector and predicts a fixed-size vector r_{len}, where each value corresponds to the likelihood of a particular number.

During inference, the predicted event number N_{set}, is obtained by selecting the highest value in r_{len}. To produce the final outputs, the event counter chooses the top N_{set} events with accurate boundaries and high-quality captions from the N event queries. As the captioning head tends to produce overestimated confidence for short sentences, the average word confidence is not a good measurement of sentence-level confidence. Thus, a modulation factor γ to rectify the influence of caption length and μ the balance factor is added to the confidence of each event query.

$$c_j = c_j^{loc} + \frac{\mu}{M_j^{\gamma}} \sum_{t=1}^{M_j} log(c_{jt}^{cap})$$

Set Prediction Loss During training, PDVC generates a series of N events along with their corresponding locations and captions. The Hungarian algorithm [7] is used to establish a correlation between the predicted events and ground truths in global schemes to obtain the most optimal bipartite matching results. The set prediction loss is the weighted sum of gIOU, classification, countering, and caption loss.

$$C = \beta_{giou} L_{giou} + \beta_{cls} L_{cls} + \beta_{ec} L_{ec} + \beta_{cap} L_{cap}$$

where L_{giou} represents the gIOU between predicted temporal and ground-truth segments, and L_{cls} represents the focal loss between the predicted classification score and the ground truth label, L_{cap} measures the cross-entropy between the predicted word probability and the ground truth, L_{ec} is also the cross-entropy loss between the predicted count distribution and the ground truth. The prediction heads to each layer of the transformer decoder and the final loss is the summation of the set prediction losses of all decoder layers.

3.4 Dataset

The effectiveness of PVDC is validated using the pre-existing large-scale ActivityNet Captions [8] and YouCook2 [9] datasets. The ActivityNet Captions dataset includes 20,000 movies of people engaging in various activities. Each movie lasts for roughly two minutes and contains four descriptive sentences. For training, validation, and testing, the dataset is divided into three sections. Similarly, the YouCook2 dataset contains 2,000 cooking films that are each about five minutes long and have around eight annotated description sentences. For training, validation, and testing purposes, this dataset is additionally divided into three sections. Due to computational constraints, only YouCook2 is used for the project. The overall train-test split is 75-25.

4 Baseline Evaluation

4.1 Evaluation Metrics

To evaluate the baseline model, the same four metrics are used as discussed in the PDVC paper. BLEU4 [10], METEOR [11], CIDEr [12] and SODA_c [13]. BLEU4, METEOR, and CIDEr calculate the average precision between generated captions and the ground truth labels. SODA_c evaluates how well the generated captions reflect the story line of a given video.

BLEU - The BLEU metric is based on precision measurement and ranges from 0 to 1 where a score of 1 means a perfect correspondence. The score is calculated between a generated caption (candidate) and its corresponding label (reference). For any n, all candidate n-gram counts and their corresponding reference counts are recorded. The candidate counts are then summed and clipped by their corresponding reference maximum value. This value is then divided by the total number of candidate n-grams. For a block of text, n-gram matches are first computed sentence by sentence. Then, the n-gram counts are summed over all candidate sentences and are divided by the number of candidate n-grams in the test corpus to calculate a modified precision score p_n. Overall, the BLEU score is calculated as:

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

BP is the brevity penalty that penalizes candidate translations for not matching the reference translations in length and w_n are the weights corresponding to each n-gram modified precision score.

METEOR - METEOR builds on top of the BLEU metric by incorporating recall into the calculations. The overall idea is based on matching unigrams and higher order n-grams not only through their surface forms but also stemmed forms (eg. plural forms) and synonyms. With all the generalized unigrams found, METEOR computes a score based on the unigram precision, recall, and a measure of fragmentation to capture how well the generated caption is ordered compared to the reference text. By incorporating recall, METEOR can capture correlation with human judgments. By taking a weighted average of the unigram recall (R) and precision (P), the F_{mean} is calculated with:

$$F_{mean} = \frac{10PR}{R + 9P}$$

To factor in longer matches, a penalty is calculated for a given alignment. To do so, all the unigrams in machine translation that are mapped to the reference are grouped into the fewest number of chunks such that the unigrams in each chunk are in the same positions as the reference. This implies a perfect translation would result in 1 chunk. The maximum of this penalty is 0.5 and the minimum is decided by the number of matched unigrams. Finally, combining the penalty and F_{mean} score, the METEOR score is calculated as:

$$S = F_{mean} \times (1 - Penalty)$$

CIDEr - The CIDEr metric is another popular automatic image descriptor evaluation metric. It measures the similarity of a generated sentence against a set of ground truth sentences written by humans. Similar to METEOR, CIDEr also incorporates both recall and precision. CIDEr aims to evaluate how well a candidate sentence matches the consensus of a set of image description sentences. Intuitively each sentence is split into a set of *n*-grams, and a consensus is drawn based on how often these set of *n*-grams are present in the reference sentences. Similarly,*n*-grams that are not present in the reference sentences are not in the candidate sentence, and *n*-grams that occur across all images provide little information and are given a lower weight. This is mathematically modeled through a Term Frequency Inverse Document Frequency (TF-IDF) weighting for each *n*-gram. the CIDEr score for *n*-grams of length n is computed using the average cosine similarity.

$$CIDEr_n(c_i, S_i) = \frac{1}{m} \sum_j \frac{g^n(c_i) \cdot g^n(s_{ij})}{\|g^n(c_i)\| \, \|g^n(s_{ij})\|}$$

where $g^n(ci)$ is a vector formed by $g^k(ci)$ corresponding to all *n*-grams of length n and $\|g^n(ci)\|$ is the magnitude of the vector $g^n(ci)$.

SODA_c - In the research community, the official dense video captioning scorer computes the averaged METEOR scores for matched pairs between generated and reference captions whose Intersection over Union (IoU) exceeds a specific threshold. However, it doesn't take into account the story or the context of the video and tends to allocate high scores to redundant captions as well. Story Oriented Dense Video Captioning evaluation framework (SODA) aims to find temporally optimal matches between generated and reference captions and penalize the redundant captions using F-scores from the METEOR scores for the matching captions. The F-measure scores for redundant captions are calculated using precision and recall, such that high amounts of captions tend to give low precision scores and high recall scores.

$$F_{measure}(G, J) = \frac{2 \times P(G, J) \times R(G, J)}{P(G, J) + R(G, J)}$$

where G is a set of manually-generated reference captions for a video and J is a set of captions generated by a system.

5 Implementation

On top of the existing PDVC baseline framework, modifications were introduced to improve the quality of video features being fed into the transformer encoder through semantic alignment. This was implemented using the pretrained text encoder to inject wisdom into the extracted video features. To achieve this, a framework as shown in Figure 3 was used. In this framework, the video captions are fed directly into CLIP's [14] frozen pre-trained text encoder. Simultaneously, the extracted video features are fed through a neural network called the video feature tuner. The output of this tuner and the text encoder are compared through a cosine similarity loss in order to align their outputs and this loss is used to update the video feature tuner weights. However, there is a dimension mismatch between the video and caption embeddings. CLIP's text encoder outputs caption embeddings with the dimensions of BxWx768 were W is the number of words in the caption and the video embeddings are of shape Bx200X3072 where 200 is the temporal dimension. Therefore to compare them using the aforementioned loss function, two steps are taken. First, an average is taken over the W dimension of the caption embedding to get the final shape of Bx768. Second, for the video embeddings, each are passed through a single layer network called the dimension matcher and an average is taken across the temporal dimension to also arrive at a shape of Bx768. In this way, the caption and video embeddings can be compared. This compression of the temporal dimension may result in the loss of information, an issue further discussed in section 8.

Once this training is over, the video feature tuner is frozen and inserted into the PDVC as shown in figure 4. Ideally with this trained tuner, it can incorporate some information from the CLIP text encoder into the video feature embeddings of unseen videos. With this addition, the overall PDVC framework is trained from scratch using YouCook2 to see whether some degree of semantic alignment was achieved. The performance of the model is evaluated with the same metrics as the PDVC baseline: BLEU4, METEOR, CIDEr, and SODA_c.

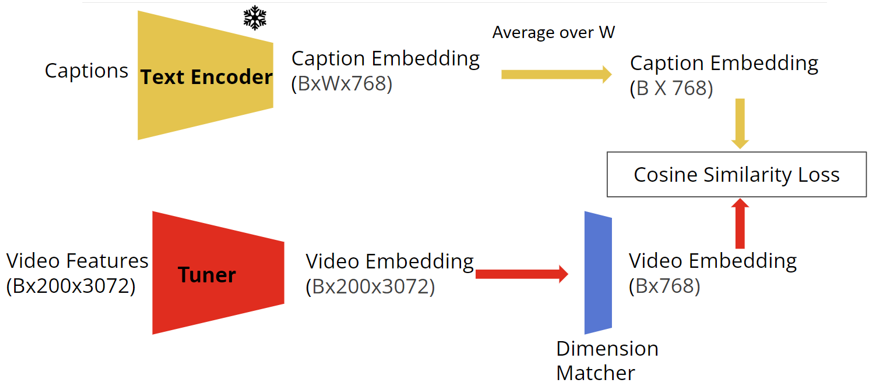

Figure 3: Tuner Training Process

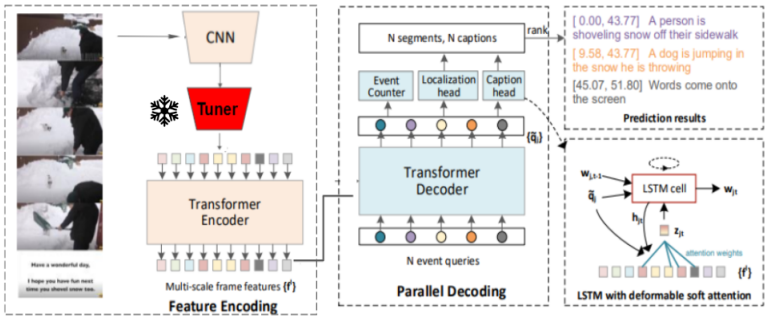

Figure 4: Tuner Integration into Existing PDVC Framework

Throughout this process, we experimented with a variety of tuner network architectures . The architecture details can be seen in Table 1. One note to make here is that Conv2 and Conv1 are just convolutions across different dimensions. Conv1 assumes the channel dimension to be the feature size and Conv2 assumes the channel dimension to be the temporal dimension. The results of each of these models can be found in section 6.

Table 1: Tuner Architecture Details

Architecture	Description
Linear	2 Linear(3072,3072) layers with BN and Gelu activation
Conv1	2 depth wise Conv1d(3072,3072,5,2) layers with BN and Gelu activation
Conv1 w/ linear	Conv1 with 1 BN and Linear(3072,3072) layer
Conv2	depth wise Conv1d(200,200,5,2) layers with BN and Gelu activation

6 Results

The baseline PDVC and the modified PDVC ablation results can be found in Table 2. Baseline implementation results are slightly different from those reported in Teng et al. [1]. Through our ablations, it was discovered that the results differ slightly every time the model is trained, meaning there is some variance with how the model may perform. However, the reported baseline results are similar enough to those reported by the paper that we felt comfortable proceeding with testing the modifications.

After training the tuner, it is frozen and placed within the PDVC framework. Subsequently, the rest of the PDVC pieline was retrained to evaluate the impact of semantic alignment on dense captioning. Again, the results are evaluated with BLEU4, METEOR, CIDEr and Soda_c. From Table 2, it is shown that Linear, Conv1 and Conv1 w/ Linear all outperform the baseline in at least three of the given metrics. In fact, Conv1 outperforms in all four metrics suggesting that it unilaterally improved

the precision, recall and also the overall story telling capabilities of PDVC. Although this is the case, Conv1 is still outperformed in the METEOR metric by Linear, so there definitely still remains room for improvement.

Conv2 model presented the lowest score across all evaluation metrics even compared with the baseline, indicating poor performance in generating captions for videos. This can be due to the low amount of model parameters compared to the other models. Fewer parameters may have limited its capability to perform proper semantic alignment and once the visual features are passed through it, no improvements are made in quality. In fact, the addition of Conv2 acted as a bottleneck to the overall framework.

Table 2: Ablations Study on Modified PDVC (bolded terms represent improved performance)

	BLEU4	METEOR	CIDEr	Soda_c
Baseline	0.76 ± 0.05	4.39 ± 0.07	20.68 ± 0.21	4.47 ± 0.87
Linear	**0.87 ± 0.06**	**4.74 ± 0.09**	**21.76 ± 0.04**	4.45 ± 1.13
Conv1	**0.90 ± 0.02**	**4.53 ± 0.07**	**22.32 ± 0.05**	**4.50 ± 1.48**
Conv1 w/ Linear	**0.77 ± 0.15**	**4.48 ± 0.02**	**21.07 ± 0.92**	4.47 ± 1.49
Conv2	0.40 ± 0.08	3.35 ± 0.01	14.34 ± 0.02	3.53 ± 0.71

7 Discussion

In conclusion, our study demonstrated the significant impact of semantic alignment on the performance of dense video captioning. Specifically, models trained with semantic alignment can achieve higher scores across all evaluation metrics, indicating better overall performance in caption generation. However, a high observed variance between the metric scores of models was noted, which could be attributed to the limited dataset size of YouCook2. This suggests that further research is needed to determine the optimal training data size and quality needed to achieve consistent performance across different metrics.

In addition, it is directly shown that an improperly trained and constructed tuner model such as Conv2 can hinder the performance of PDVC across all evaluation metrics, stressing the importance of investigating optimal tuner model architectures.

Nevertheless, this project highlights the importance of considering semantic alignment when generating video captions. Additionally, the results provide insight into the strengths and weaknesses of different architectures and evaluation metrics for dense video captioning, which could inform the development of more effective models in the future.

8 Future Work

Future work in dense video captioning could further evaluate the viability of the model and address its limitations. Possible extensions include training the model on larger and more diverse datasets, such as ActivityNet. Theoretically, this should help improve the model's generalizability to a wider range of videos and the overall stability of the model across different evaluation metrics. Additionally, other tuner architectures can be explored, such as transformer-based models or models with attention mechanisms, to further improve performance.

While we used CLIP as the primary text encoder in our study, other text encoders may provide better semantic alignment, and testing other text encoders could fine tune the performance of our model. Furthermore, the current method of matching dimensions between the video and text embeddings can result in the loss of information. Therefore, to achieve more sophisticated semantic alignment, other methods of dimension matching will be investigated. Possible solutions include scaling up the dimensions of the text embeddings instead of scaling down video embeddings. Instead of taking an average across the temporal dimensions, models where the temporal dimension is preserved could be explored as well.

Overall, the results of this study provide a foundation for further research on improving the performance of dense video captioning models, and demonstrate the impact of semantic alignment in models of this type.

References

[1] Teng Wang, Ruimao Zhang, Zhichao Lu, Feng Zheng, Ran Cheng, and Ping Luo. End-to-end dense video captioning with parallel decoding. *CoRR*, abs/2108.07781, 2021.

[2] Soheyla Amirian, Khaled Rasheed, Thiab R. Taha, and Hamid R. Arabnia. Automatic image and video caption generation with deep learning: A concise review and algorithmic overlap. *IEEE Access*, 8:218386–218400, 2020.

[3] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. 2021.

[4] 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.

[5] 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*, pages 4507–4515, 2015.

[6] Ji Wang, Wenbin Jiang, Lin Ma, Wei Liu, and Yong Xu. Bidirectional attentive fusion with context gating for dense video captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7190–7198, 2018.

[7] Nicolas Carion, Matthieu Francisco, Stefan Gabriel, Uwe Nicolas, Karsten Alexander, and Zukov Sergey. End-to-end object detection with transformers. In *Proceedings of the European Conference on Computer Vision*, pages 213–229, 2020.

[8] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-captioning events in videos. *CoRR*, abs/1705.00754, 2017.

[9] Linyi Zhou, Chenliang Xu, and Jason J. Corso. Towards automatic learning of procedures from web instructional videos. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7590–7598, 2018.

[10] 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*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

[11] Alon Lavie and Michael Denkowski. The meteor metric for automatic evaluation of machine translation. *Machine Translation*, 23:105–115, 09 2009.

[12] Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4566–4575, 2015.

[13] S.Fujita, T.Hirao, H.Kamigaito, M.Okumura, and M.Nagata. Soda: Story-oriented dense video captioning evaluation framework. *Proc. Eur. Conf. Comput. Vis.*, 2020.

[14] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021.