Fast Inference for Vision-Language Model Image Captioning

Andrew Shi
Stanford University
acshi@stanford.edu

Taeuk Kang
Stanford University
taeuk@stanford.edu

Nash Brown
Stanford University
ncbrown@stanford.edu

Abstract

Autoregressive vision-language models for image captioning suffer from slow inference due to sequential token generation. We address this bottleneck by adapting the Medusa framework to image captioning, introducing speculative decoding heads to a ViT-GPT2 encoder-decoder architecture. Our approach augments a pretrained VisionEncoderDecoderModel (nlpconnect/vit-gpt2-image-captioning) with K lightweight Medusa heads, each predicting tokens $k+1$ steps into the future. We explore two training strategies: Medusa-1 freezes the 240M parameter backbone and trains only the Medusa heads, while Medusa-2 jointly fine-tunes all parameters. On Flickr8k, Medusa-1 with 4 heads achieves comparable validation loss to baseline (2.42 vs 2.51) while accelerating inference by 2.98× (77.6 vs 26.1 samples/s). Medusa-2 further improves quality (2.28 loss) while maintaining similar speedup (78.7 samples/s). Our results demonstrate that speculative decoding can effectively accelerate vision-language generation without sacrificing caption quality.

1. Introduction

Image captioning—the task of generating descriptive text for visual input—has wide-ranging applications, from assisting visually impaired users to indexing massive photo collections and enabling human–robot interaction. Recent advances in vision-language models, particularly those that pair a Vision Transformer (ViT) encoder with an autoregressive Transformer decoder, have achieved impressive accuracy on many benchmarks. However, their reliance on sequential token generation leads to slow inference, requiring one full forward pass through the decoder per token.

In many practical settings, such as live video captioning, mobile applications, or large-scale image indexing, reducing this decoding time is critical. The computational bottleneck arises from the autoregressive nature of the decoding process: each token must be generated sequentially, preventing parallelization across the sequence dimension during inference.

The input to our algorithm is a single RGB image of arbitrary resolution, which we preprocess to 224×224 pixels to match ViT requirements. We then use a ViT encoder coupled with a GPT-2 decoder augmented with multiple lightweight Medusa heads to output a predicted natural language caption describing the image content. Specifically, our architecture processes the input image through a pretrained ViT-base-patch16-224 encoder to extract visual features, which are then fed to a GPT-2 decoder enhanced with K additional Medusa heads. Each Medusa head k is designed to predict the token at position $t + k + 1$ in the caption sequence, enabling speculative multi-token generation. The final output is a variable-length text sequence (typically 10-20 tokens) providing a descriptive caption of the input image.

Our core technical contribution lies in adapting the Medusa speculative decoding framework to the vision-language domain. While the original Medusa work focused on pure language modeling tasks, we address the unique challenges of cross-modal generation where visual features must be effectively integrated with textual predictions across multiple future positions. This adaptation requires careful consideration of how visual context propagates through the decoder layers and influences each Medusa head's predictions.

We address this challenge by adapting the Medusa framework [1] to vision-language tasks. Medusa introduces speculative decoding by training multiple lightweight "heads" that predict future tokens in parallel, potentially allowing multiple tokens to be generated in fewer forward passes. We implement and compare two training strategies: backbone-frozen (Medusa-1) and joint fine-tuning (Medusa-2). We demonstrate 2.98× inference speedup on Flickr8k while maintaining or improving caption quality. We also provide empirical analysis of the trade-offs between number of heads, model performance, and inference speed.

2. Related Work

We organize related work into three categories: (1) vision-language captioning architectures, (2) speculative or

multi-token decoding, and (3) efficient generation techniques.

Vision-Language Models. Modern image captioning approaches predominantly adopt an encoder-decoder paradigm, where a visual encoder (e.g., a CNN or Vision Transformer) first extracts a compact representation of the image, and then an autoregressive text decoder generates a caption based on those features. Early methods like Show and Tell [12] combined a CNN encoder with an LSTM decoder, demonstrating that end-to-end training yields better captions than rule-based systems. However, CNN-LSTM models often produce repetitive or overly generic descriptions and can struggle to scale to large datasets without substantial tuning. More recently, Transformer-based architectures have become dominant. Dosovitskiy et al. [4] showed that splitting an image into patches and feeding them into a Vision Transformer (ViT) produces strong visual embeddings, but at the cost of higher compute compared to CNNs. Building on this idea, Kumar [6] illustrates how a ViT encoder can directly connect to a GPT-style decoder, yielding fluent captions that generalize better than earlier LSTM-based pipelines. Large-scale pretraining efforts such as VL-BERT [11] and BLIP [7] learn joint representations by combining image and text data, further boosting caption quality on benchmarks. While these large pretrained models achieve state-of-the-art accuracy, they still generate one token at a time. Our work builds on the VisionEncoderDecoderModel framework from Hugging Face—specifically using a ViT encoder with a GPT-2 decoder—and focuses on reducing inference time.

Speculative Decoding. Speculative decoding methods seek to overcome the "one token per forward pass" bottleneck of autoregressive generation by proposing multiple future tokens in parallel and then verifying them. Stern et al. [10] introduced blockwise parallel decoding: a model generates a short block of tokens and then checks them sequentially, reducing the total number of passes but still requiring repeated verification of each token in order. Xia et al. [13] proposed a more general speculative decoding framework for sequence-to-sequence models, where a small "draft" model proposes candidate tokens that a larger "verification" model confirms in fewer passes. While this two-model setup can achieve significant speedups, it introduces the overhead of maintaining and training a separate draft network. More recently, Chen et al. [2] presented speculative sampling, which similarly leverages a lightweight draft model to accelerate inference for large language models. These approaches are clever in reducing decoding rounds, but they often sacrifice some fluency or require careful balancing between draft and verification models. Cai et al. [1] introduced Medusa, a framework that adds multiple parallel decoding heads to a single large model. Each head k predicts the token at position $t + (k + 1)$ directly from the current hidden state, and a tree-based attention mechanism verifies these multi-step predictions together, which can cut the number of full decoder passes. Medusa's multi-head speculation is particularly practical because it retains a single decoder backbone and only requires a specialized verification step.

Efficient Vision-Language Generation. Beyond speculative decoding, various lines of research aim to make vision-language models smaller, faster, or more resource-efficient. Knowledge distillation techniques, such as those explored by Fang et al. [5], train a compact student model to mimic a larger teacher, preserving much of the original performance in a smaller footprint. While distillation can yield smaller models, it requires a large teacher network and additional training iterations. Quantization methods, such as Q8BERT [14], reduce numerical precision during inference, which can speed up computations on specialized hardware with negligible accuracy loss, but may degrade fine-grained visual reasoning. PaLI [3] demonstrates that jointly scaling model size and dataset size can improve both throughput and accuracy in multilingual vision-language tasks. However, massive scaling demands substantial compute resources. Other works explore lightweight adaptation layers (e.g., adapters or LoRA) to fine-tune only a small subset of parameters, reducing the inference footprint but still requiring architectural modifications. In contrast, our approach keeps the original ViT + GPT-2 architecture intact and simply attaches lightweight Medusa heads alongside the decoder, enabling multi-token proposals and inference time speedup without additional compression or quantization steps.

3. Data

We evaluate on the Flickr8k dataset, a standard benchmark for image captioning containing 8,000 images, each paired with five human-written captions. The dataset provides diverse descriptions that capture entities, actions, and spatial relationships in natural scenes, making it an ideal testbed for evaluating both caption quality and generation speed. The images span a wide range of scenarios, from indoor and outdoor scenes to various human activities and animal behaviors, ensuring our evaluation captures the model's ability to handle diverse visual content.

The five captions for the example image demonstrate the variation in descriptions: {"`A black dog and a spotted dog are fighting.`", "`A black dog and a spotted dog are fighting.`", "`A black dog and a white dog with brown spots are staring at each other in the street.`", "`Two dogs of different breeds looking at each other on the road.`", "`Two dogs on pavement moving toward each other.`"}. This variation highlights the challenge of cap-

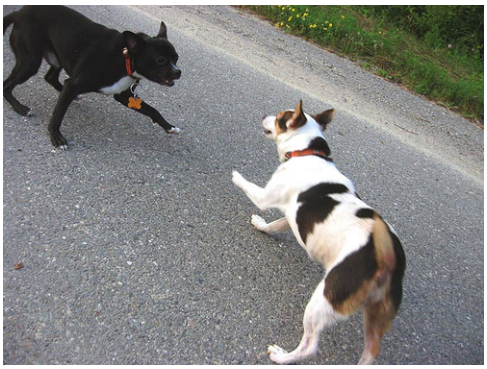

Figure 1. Example image from Flickr8k dataset showing two dogs facing each other on pavement.

tion generation, as there are multiple valid ways to describe the same visual scene, ranging from detailed descriptions of specific features to more general observations about the overall scene.

Our experimental setup uses an 80/20 train-test split, resulting in 6,400 training images and 1,600 test images. This split ensures sufficient training data while maintaining a robust evaluation set for measuring generalization performance. With five captions per image, the training set provides 32,000 image-caption pairs, while the test set contains 8,000 pairs. During training, we implement a random caption selection strategy where one of the five available captions is randomly chosen for each image in each epoch, introducing variability that improves model generalization across different linguistic expressions of the same visual content.

The dataset's caption diversity is particularly valuable for training robust models. Each image's five captions typically exhibit different levels of detail, focus on different aspects of the scene, and use varied vocabulary and sentence structures. This natural variation helps the model learn that multiple valid descriptions exist for any given image, encouraging the development of more flexible and creative caption generation capabilities.

Preprocessing. Our image preprocessing pipeline is designed to ensure compatibility with the pretrained ViT encoder while maintaining visual fidelity necessary for accurate caption generation. All images undergo a standardized transformation sequence implemented using PyTorch's transforms module [9]. The preprocessing begins with resizing all images to 224×224 pixels, which matches the input requirements of the ViT-base-patch16-224 architecture. This resizing maintains aspect ratio consistency while ensuring uniform input dimensions across the dataset.

Following resizing, images are converted to RGB format to ensure consistent color channel representation. The pixel values are then normalized using ImageNet statistics with mean values of $[0.485, 0.456, 0.406]$ and standard deviations of $[0.229, 0.224, 0.225]$ for the red, green, and blue channels respectively. This normalization aligns with the pretraining distribution of the ViT encoder, ensuring optimal performance from the pretrained weights and preventing distribution shift between pretraining and fine-tuning phases. This preprocessing ensures compatibility with the pretrained ViT encoder while maintaining the visual fidelity necessary for accurate caption generation.

Caption preprocessing employs the GPT-2 tokenizer, which uses byte-pair encoding (BPE) to handle the full vocabulary of 50,257 tokens. Each caption is tokenized with a maximum length constraint of 50 tokens, which accommodates the vast majority of captions in the Flickr8k dataset while maintaining computational efficiency. The tokenizer applies padding to the right side of sequences to ensure uniform length across batches, with padding tokens receiving special attention mask values to exclude them from loss computation. During training, we randomly sample one caption per image to introduce variability and improve generalization across different caption styles and vocabulary choices.

To maximize training efficiency and reduce computational overhead, we preprocess all images and cache the results as PyTorch tensor files (.pt format), eliminating the need for repeated image decoding and transformation during training. This preprocessing step converts images to tensors with shape [3, 224, 224] and applies all normalization.

4. Methods

4.1. Model

We use a pretrained backbone model that comprises of a Vision Transformer (ViT) encoder and a GPT-2 decoder. This allows us to build upon a capable yet lightweight model with learned weights for vision-language tasks. For the backbone model's encoder, it uses Google's vit-base-patch16-224-in21k model as the encoder, which is a ViT trained on 14 million ImageNet examples and outputs a probability distribution over 21,843 classes at resolution 224 x 224. The model is then fine-tuned on 1 million more ImageNet examples and outputs a probability distribution over 10,000 classes. For the backbone model's decoder, it uses GPT-2 as a lightweight language model that generates captions autoregressively based on the image features from the ViT encoder. The decoder uses both masked multi-head self-attention for the caption sequence and multi-head cross-attention to access the keys and values from the final hidden states of the ViT encoder. The backbone model consists of around 240M parameters.

4.2. Medusa Heads

The key innovation in our project is the integration of Medusa heads into the decoder architecture. Each Medusa

head is a lightweight feed-forward network designed to predict future tokens based on current decoder hidden states. The architecture consists of a first linear layer (`fc1`) that maps from 768 to 768 dimensions with SiLU activation, followed by a residual connection where the output is computed as `SiLU(fc1(input)) + input`. Finally, a second linear layer (`fc2`) maps from 768 to 50,257 dimensions to produce vocabulary predictions. This is done by copying weights from the main LM head of the pretrained GPT-2 decoder. This provides a strong initialization, as the heads start with the ability to generate reasonable next-token predictions. The first layer (`fc1`) uses standard Kaiming initialization. This initialization scheme and Medusa head architecture closely resembles the method described in [1].

The Medusa heads are implemented as `nn.ModuleList` containing K lightweight heads, each with identical architecture but independent parameters. Critical to proper functioning is the initialization strategy: fc2 layers are initialized by copying weights and biases from the pretrained GPT-2 language modeling head, ensuring each Medusa head starts with the ability to generate reasonable token predictions. The fc1 layers use normal initialization with standard deviation 0.02 and zero bias initialization for stable training dynamics.

During the forward pass of the Medusa head enhanced image captioning model, the standard encoder-decoder pass is first performed. Then, the hidden states from the decoder (after its last layer normalization) are retrieved. These hidden states are then passed through each of the Medusa heads in parallel to produce different logits at each head. Each Medusa head is trained to predict tokens in the future without seeing the intermediate ground truth tokens, where Medusa head k predicts token $k+1$ steps into the future.

At inference time, the base LM head and the Medusa heads propose a tree of candidate token sequences, and these candidates are verified in parallel using a mechanism similar to tree attention. The longest accepted candidate sequence is appended to the current generation, allowing multiple tokens to be decoded in fewer forward passes than the standard autoregressive next-token prediction mechanism. We also set a flag to allow the model to greedily decode the top-1 token from the base model's prediction for each Medusa head.

4.3. Training Setup

For the non-Medusa head architecture, we use the standard cross-entropy loss to calculate the loss for each mini-batch and update the model parameters. The loss function is given by

$$\mathcal{L}_{\text{BASE}} = -\frac{1}{N}\sum_{i=1}^{N}\log p_{t,i}\big(y_{i,t+1}\big), \tag{1}$$

where N is the mini-batch size.

For the Medusa head model, we use two training methods, following the methods discussed in [1]. For the first method, we freeze the backbone of the base model (ViT and GPT-2 decoder) and only train the parameters of the Medusa heads. In this setting, we use a revised training objective where the total loss is a sum of the cross-entropy losses for each Medusa head's predictions. The loss function for the Medusa head model is given by

$$\mathcal{L}_{\text{MEDUSA-1}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K}\lambda_k\ \log p_{t,i}^{(k)}\big(y_{i,t+k+1}\big), \tag{2}$$

where K is the number of Medusa heads and $\lambda_k = r^k$ for some decay constant r (e.g., $r = 0.8$).

In the second method, we perform a full fine-tune on the base model where the backbone and the Medusa heads are both trained. In this setting, the loss is calculated as a sum of the standard cross-entropy loss from the base GPT-2 model's LM head and the multi-token prediction from the Medusa heads. Namely, the loss function for the second method is given by

$$\mathcal{L}_{\text{MEDUSA-2}} = -\frac{1}{N}\sum_{i=1}^{N}\Big[\log p_{t,i}(y_{i,t+1})\ +\ \sum_{k=0}^{K-1}\alpha_k\ \log p_{t,i}^{(k)}(y_{i,t+k+1})\Big], \tag{3}$$

where α_k are the non-negative weights such that $\sum_k \alpha_k = 1$, $p_{t,i}$ is the baseline head's probability distribution for sample i at timestep t, and $p_{t,i}^{(k)}$ is the k-th Medusa head's distribution for the same sample. We use a cosine annealing learning rate scheduler to adjust the learning rate over epochs. We also use gradient clipping to stabilize training where the maximum norm is 1.0 to prevent exploding gradients. We selected AdamW as our optimizer due to its superior performance on transformer-based models, with weight decay set to 0.01 to prevent overfitting.

The Medusa loss implementation includes several critical technical details for training stability. Label shifting is carefully handled such that Medusa head k's logits at position t predict labels at position t+k+1, ensuring proper temporal alignment. We apply attention masking to exclude padding tokens (marked with -100) from loss computation, preventing the model from learning spurious patterns from padding. The loss function incorporates extensive numerical stability checks, filtering out non-finite losses and implementing proper loss averaging to prevent gradient explosions. For Medusa-2 training, we use different parameter groups with the AdamW optimizer: Medusa heads receive the full learning rate while backbone parameters use a 10× smaller learning rate to preserve pretrained knowledge.

The training loop implements robust error handling and monitoring. Validation is performed every 50 training steps to track model performance, with best model checkpointing based on validation loss. We implement gradient clipping with maximum norm 1.0 and include extensive logging

through Weights & Biases for experiment tracking. The data loading strategy randomly selects one caption per image during each epoch, introducing beneficial training variability across the multiple available captions per image.

4.4. Training Hyperparameters

Our hyperparameter choices were motivated by balancing training stability, computational efficiency, and model performance. For Medusa-1, we used a learning rate of 2e-3 for the Medusa heads, chosen to allow rapid adaptation of the lightweight heads while preserving the pretrained backbone knowledge. We set the lambda weights for the Medusa heads to 0.8^{k+1}. For Medusa-2, we employed a two-tier learning rate strategy: 2e-3 for Medusa heads and 2e-4 for backbone parameters, with the 10× ratio ensuring that pretrained features are fine-tuned gradually while allowing aggressive optimization of the new Medusa components. The batch size of 16 was constrained by our T4 GPU memory limitations while providing sufficient gradient stability. We implemented cosine annealing learning rate scheduling over 5 epochs, with gradient clipping at maximum norm 1.0 to prevent exploding gradients common in multi-head training scenarios. Training for 5 epochs provided sufficient convergence based on validation loss plateauing, while remaining computationally feasible for our resource constraints.

4.5. Evaluation

For evaluation, we run a forward pass of our model on the test set both at intervals during training and after the final model weights have been saved. We test for two metrics. First, we evaluate the loss that the models achieve on the test set and then evaluate the inference speed (measured in samples/seconds and tokens/second). A key characteristic of Medusa heads is that the performance of the model should not degrade due to the multi-token prediction mechanism. Thus, we expect that the loss on the test set using the Medusa-1 strategy remains comparable to the baseline backbone model and the loss for the Medusa-2 strategy to be lower than both the baseline backbone model and the Medusa-1 model since it performs a full fine-tune over the model weights.

For inference speed evaluation, we measure both samples per second and tokens per second over 100 batches from the validation set with a batch size of 16. This evaluation setup provides sufficient statistical power while maintaining reasonable computational cost. We compare the baseline model using standard autoregressive generation against our Medusa-enhanced models using their respective generation procedures. All timing measurements exclude data loading and preprocessing time to focus specifically on the model inference characteristics.

5. Experiments

5.1. Model Performance

First, we examine the performance of the trained models measured in cross-entropy loss. Figure 2 shows the training loss over batches, Figure 3 shows the training loss over epochs, and Figure 4 shows the validation loss at intervals of 50 iterations. We see that the curves look reasonable and well-behaved. We note that the validation loss shows a steady decrease across all 5 epochs; however, evaluating the shapes, we suggest that one can experiment with a higher learning rate to achieve better performance in the same number of epochs. There are two main observations from the loss curves. First, training with more Medusa heads improves model performance, as we observe a monotonic decrease in the validation loss curves as we increase the number of Medusa heads. Second, keeping the number of Medusa heads constant, performing a full fine-tune over all parameters allows the model to achieve better performance.

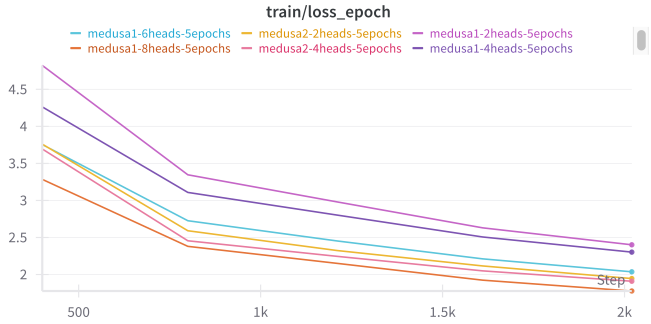

Figure 2. Training loss for different models over 5 epochs.

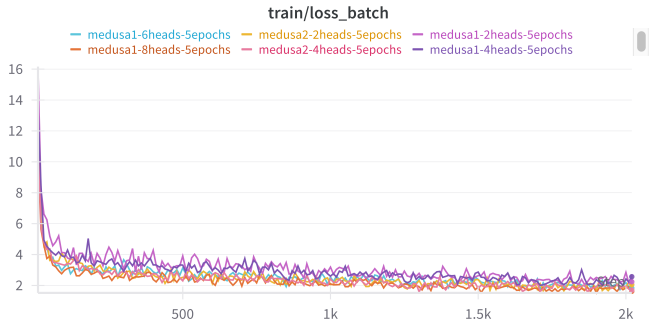

Figure 3. Training loss for different models over every minibatch.

To explain the first observation, increasing the number of Medusa heads may allow the model to gain a richer loss signal and may encourage the model to plan ahead when predicting future tokens for the caption. Furthermore, with an increased number of parameters due to the linear layers

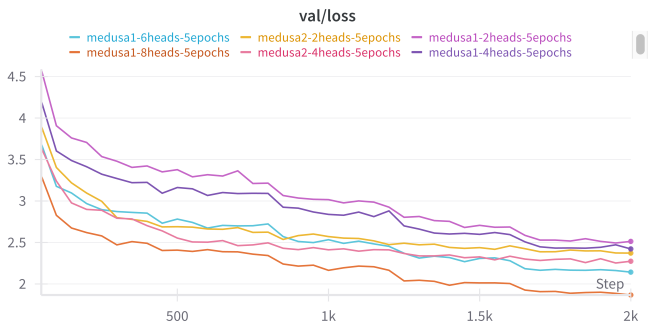

Figure 4. Validation loss for different models over 5 epochs, measured every 50 steps.

in each Medusa head, the model may have greater capacity to learn. The second observation is expected and intuitive; a full fine-tune over the model parameters is more effective as we tune the weights for our specific downstream task of interest.

After training the models and saving the checkpoints, we run a single forward pass on the 1,600 samples from the validation set and measure the cross-entropy loss. The results are shown in Table 1. We see that the performance of the Medusa-1 model is close to the baseline pretrained model, and the performance of the Medusa-2 model is better on the downstream task.

Model	# Medusa Heads	Validation Loss
Baseline	1	2.51
Medusa-1	2	2.52
Medusa-1	4	2.42
Medusa-2	2	2.37
Medusa-2	4	2.28

Table 1. Final validation loss for different models.

Our results reveal several important insights about applying Medusa to vision-language tasks. First, increasing the number of Medusa heads consistently improves validation loss, suggesting that multi-step prediction provides a richer training signal that encourages better planning in caption generation. The monotonic improvement from 2 to 4 heads indicates that the model benefits from longer prediction horizons, though computational constraints limited our exploration beyond 4 heads for Medusa-2.

Second, the comparison between Medusa-1 and Medusa-2 demonstrates different trade-offs for practical deployment. Medusa-1 offers the advantage of preserving the pretrained backbone entirely while achieving substantial speedup, making it ideal for scenarios where model stability and minimal training overhead are priorities. Medusa-2 provides superior generation quality through joint optimization but requires more computational resources during training.

6. Conclusion

We adapted the Medusa speculative decoding framework to vision-language image captioning, achieving significant inference acceleration without sacrificing caption quality. Our comprehensive evaluation demonstrates the effectiveness of this approach across multiple dimensions: the Medusa-1 strategy achieves competitive performance (2.42 vs 2.51 baseline loss) with 2.98× inference acceleration while requiring training of only the lightweight Medusa heads rather than the full model, making it highly practical for resource-constrained deployment scenarios. The Medusa-2 approach further improves caption quality to 2.28 loss while maintaining similar speedup characteristics, demonstrating the benefits of joint optimization when computational resources permit full fine-tuning.

Future work should explore several promising directions to enhance both the effectiveness and practical applicability of Medusa-based vision-language generation. The most immediate technical improvement involves implementing the complete tree-based verification mechanism described in the original Medusa work, which would unlock the full potential of multi-token acceptance and potentially achieve even greater speedup than our current simplified verification approach. Data augmentation represents another important avenue for improvement, as our current preprocessing pipeline deliberately avoids augmentation techniques such as random cropping, horizontal flipping, color jittering, and modern strategies like MixUp or CutMix that could significantly enhance model robustness and generalization across diverse visual conditions. Additionally, scaling experiments to larger vision-language models such as BLIP-2 or LLaVA would provide insights into how the Medusa framework scales with model size, while compression techniques like knowledge distillation or quantization could make the approach more suitable for mobile deployment scenarios.

Evaluation extensions are crucial for validating the broader applicability of the approach, including scaling to larger datasets like COCO Captions and incorporating more sophisticated caption quality metrics such as BLEU, CIDEr, and human evaluation studies that better capture semantic similarity and user preferences than cross-entropy loss alone. Cross-domain evaluation on datasets with different visual characteristics, such as medical images or artwork, would demonstrate generalizability beyond natural scene photography, while training methodology improvements like curriculum learning could further enhance both efficiency and performance.

The demonstrated success of Medusa for image captioning opens promising opportunities for accelerating other vision-language tasks, including visual question answering and image-text retrieval, where inference speed is equally critical for practical deployment. As vision-language models continue to grow in size and complexity, speculative de-

coding approaches like Medusa will become increasingly important for making these powerful models practical for real-time applications.

7. Contributions and Acknowledgements

Andrew Shi worked on setting up the model architecture, experimentation, and training. Nash Brown worked on hyperparameter tuning, performance tuning, and data evaluations. Taeuk Kang worked on the training infrastructure and GPU and preprocessing the data. We thank our project mentor, Sabri Eyuboglu, for their continued guidance and support throughout the quarter. We acknowledge [8] for the base model and code. We acknowledge Amazon Web Services for providing credits to train and run experiments.

References

[1] T. Cai, Y. Li, Z. Geng, H. Peng, J. D. Lee, D. Chen, and T. Dao. Medusa: Simple llm inference acceleration framework with multiple decoding heads, 2024. 1, 2, 4

[2] C. Chen, S. Borgeaud, G. Irving, J.-B. Lespiau, L. Sifre, and J. Jumper. Accelerating large language model decoding with speculative sampling, 2023. 2

[3] X. Chen, X. Wang, S. Changpinyo, A. Piergiovanni, P. Padlewski, D. Salz, S. Goodman, A. Grycner, B. Mustafa, L. Beyer, A. Kolesnikov, J. Puigcerver, N. Ding, K. Rong, H. Akbari, G. Mishra, L. Xue, A. Thapliyal, J. Bradbury, W. Kuo, M. Seyedhosseini, C. Jia, B. K. Ayan, C. Riquelme, A. Steiner, A. Angelova, X. Zhai, N. Houlsby, and R. Soricut. Pali: A jointly-scaled multilingual language-image model, 2023. 2

[4] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16×16 words: Transformers for image recognition at scale, 2021. 2

[5] Z. Fang, J. Wang, X. Hu, L. Wang, Y. Yang, and Z. Liu. Compressing visual-linguistic model via knowledge distillation, 2021. 2

[6] A. Kumar. The illustrated image captioning using transformers. *ankur3107.github.io*, 2022. 2

[7] J. Li, D. Li, C. Xiong, and S. Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation, 2022. 2

[8] NLP Connect. vit-gpt2-image-captioning (revision 0e334c7), 2022. 7

[9] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019. 3

[10] M. Stern, N. Shazeer, and J. Uszkoreit. Blockwise parallel decoding for deep autoregressive models, 2018. 2

[11] W. Su, X. Zhu, Y. Cao, B. Li, L. Lu, F. Wei, and J. Dai. Vl-bert: Pre-training of generic visual-linguistic representations, 2020. 2

[12] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator, 2015. 2

[13] H. Xia, T. Ge, P. Wang, S.-Q. Chen, F. Wei, and Z. Sui. Speculative decoding: Exploiting speculative execution for accelerating seq2seq generation, 2022. 2

[14] O. Zafrir, G. Boudoukh, P. Izsak, and M. Wasserblat. Q8bert: Quantized 8bit bert. In *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS Edition (EMC2-NIPS)*, page 36–39. IEEE, Dec. 2019. 2