FUSING VISION AND LANGUAGE MODELS TO GENERATE SEQUENCE OF RECIPE IMAGES FROM STEPS

Hshmat Sahak
Department of Computer Science
University of Toronto
Toronto, ON, CAN
hshmat.sahak@mail.utoronto.ca

ABSTRACT

In this work, we present RecipeVis, which generates an image for each step in a recipe by conditioning on the previously generated image and current step. RecipeVis leverages the power of the pretrained text-to-image Stable Diffusion model, as well as text and visual encoders that produce task agnostic embeddings for downstream applications. It uses an attention module to fuse the text and image embeddings. It also adds a cycle consistency loss to the standard diffusion loss.

1 INTRODUCTION AND BACKGROUND

The most well-known vision-language models fall under vision-language understanding, text generation with multimodal input, or multimodal output with multimodal input (Ghosh et al., 2024). Recipe image generation falls under image output from multimodal input. To account for this, we use the VLM architecture of Llava (Liu et al., 2024), but swap the LLM decoder with Stable Diffusion (Rombach et al., 2022), a state-of-the-art model for image generation. Our primary contributions include an empirical study to find a good image-text fusion layer design, demonstrating that VLMs incorporating previous image provides superior results over baseline text-to-image models, and introducing a cycle consistency loss term to ensure consistency between modes of output.

Diffusion Models (Ho et al., 2020; Sohl-Dickstein et al., 2015) have emerged as a class of generative models that have challenged the dominance of generative adversarial networks (GANs) by advancing the state of the art in text-to-image synthesis (Dhariwal and Nichol, 2021), as well as other domains like multi-modal modeling (Avrahami et al., 2022). They work by gradually adding noise to the data in a forward process and then learning to reverse this process to generate new samples. The forward process can be described as a Markov chain of length T, where noise is added to the data step by step. Here, α_t is a variance schedule that controls the amount of noise added at each step t.

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t}\mathbf{x}_{t-1}, (1-\alpha_t)\mathbf{I})$$

The reverse process aims to denoise the data, which is modeled using a neural network that predicts the mean and variance of the data at each step. There may also be a conditional signal $\mathbf{c}$.

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{c}) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t, \mathbf{c}), \Sigma_\theta(\mathbf{x}_t, t, \mathbf{c}))$$

During training, the model learns to reverse the noising process by minimizing a loss function that measures the difference between the true data distribution and the distribution of the generated samples. By iteratively applying the reverse process, the model can generate high-quality samples from random noise and the conditional signal $\mathbf{c}$.

$$\mathbf{x}_{t-1} = \mathbf{x}_t - \epsilon_\theta(\mathbf{x}_t, t)$$

where $\epsilon_\theta(\mathbf{x}_t, t, \mathbf{c})$ represents the model's prediction of the noise component at step t. This iterative denoising ultimately results in a coherent generated sample.

2 METHOD

The architecture of RecipeVis consists of an image and text encoder to produce embeddings (of previously generated image and current instruction) that are then fused in the image-text fusion

module to produce a condition vector that is passed to Stable Diffusion (see Figure 1). During training, the encoders are frozen, UNet is finetuned and fuser module is trained from scratch. This mimics common VLM architectures, replacing the language model with a generative model.

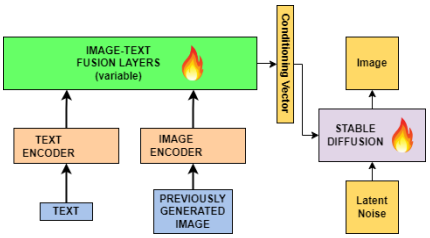

Figure 1: RecipeVis produces sequence of images by passing current instruction and previous image into frozen encoders, fusing the image and text representations, and passing it to a fine-tuned Stable Diffusion model.

Image-Text Fusion Module: We explore different mechanisms for fusing text and image embeddings to the latent space of Stable Diffusion. In **concatenation**, the embeddings are flattened and concatenated, then passed to a feed-forward neural net. Another strategy involves taking the pooled output embedding from the image encoder, projecting it to match the size of the text embedding, and add the image encoding to each embedding in the text sequence. We then pass this through a feed-forward network and reshape the output. We call this strategy **pooling image embeddings**. The reverse case, where we pool text embeddings, then project and add it to the image embeddings, is called **pooling text embeddings**. Finally, the attention module utilizes the cross attention between image and text as the conditional embedding. In the case of **attending to image**, the queries are text embeddings while keys and values are image embeddings. In **attending to text**, queries are image embeddings while keys and values are text embeddings.

Cycle Consistency Loss: We define the cycle consistency loss (details in A.4) as

$$L_{\text{cyc}} = 1 - \text{CLIP}(\text{Text}) \cdot \text{CLIP}(\text{VAE\_postprocess}(\text{VAE\_decoder}(\text{Pred\_}x_0)))$$

Here, "Text" is the description associated with an image, and Pred_x_0 is the diffusion model's prediction of the image given a noisy version of the image, the text and previous image.

3 Experimental Results

We use the YouCookII dataset (Zhou et al., 2018), prepared by the University of Michigan. It contains recipe videos, along with the instruction set and frame range for each step. We construct a dataset by assigning the middle frame of each section to the corresponding step, and collect all (previous image, instruction, image) triplets. In our experiments, we fine-tune the CompVis/stable-diffusion-v1-4 UNet and train the Fuser module. We use frozen Stable Diffusion's text encoder and openai/clip-vit-large-patch14 (Radford et al., 2021) as image encoder. We run training for 100000 steps with a learning rate of 3e-07 for both UNet and Fuser. For sampling, we use a DDPM scheduler, a guidance scale of 7.5, and 50 iterative refinement steps. Full details in Appendix A. The table below compares the performance of the different fusers on Clip L2 Comparison, PSNR and SSIM (Wang et al., 2004).

Table 1: Image comparison metrics (with/without cycle consistency loss).

Fuser	Clip Comparison Score ↓	PSNR ↑	SSIM ↑
Baseline (Stable Diffusion)	34.95	8.51	0.20
Concatenation	33.54/33.27	7.84/8.75	0.19/0.20
Text Pooling	34.13/33.55	8.28/8.38	0.18/0.18
Image Pooling	35.87/35.45	7.77/7.79	0.18/0.17
Attend to Text	34.95/33.62	7.06/7.06	0.17/0.18
Attend to Image	32.47 / **32.29**	9.11 / **9.19**	**0.21**/0.19

4 Conclusion

We present an empirical study of fusion methods for VLMs in the task of image sequence generation. We also propose a technical novelty in the training pipeline to ensure consistency between output modes. RecipeVis lays the foundation for more advanced neural architectures and loss functions to improve performance on image sequence generation from text. We leave more advanced fusion methods to further work, and hope to inspire further research in image sequence generation.

URM STATEMENT

Author Hshmat Sahak meets the URM criteria of ICLR 2024 Tiny Papers Track.

REFERENCES

Akash Ghosh, Arkadeep Acharya, Sriparna Saha, Vinija Jain, and Aman Chadha. Exploring the frontier of vision-language models: A survey of current methodologies and future directions. *arXiv preprint arXiv:2404.07214*, 2024.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18208–18218, 2022.

Luowei Zhou, Chenliang Xu, and Jason Corso. Towards automatic learning of procedures from web instructional videos. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. doi: 10.1109/TIP.2003.819861.

A APPENDIX

A.1 DATASET CURATION

YouCook2 is a very large video dataset aimed to be used by the vision community. It contains 2000 long untrimmed videos from 89 cooking recipes from around the world. On average, each recipe has 22 videos, and each video is an average of 5.26 minutes. Across all videos, there is 176 hours of data, with no video going past 10 minutes. Each video is annotated with its procedural steps (imperative English sentences) and their temporal boundaries. The videos are all downloaded from Youtube, and are in third-person viewpoint. An example of one annotated video is shown in Figure 2

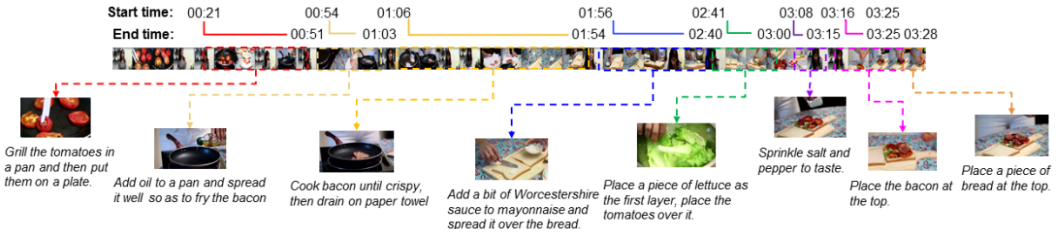

Figure 2: Sample YouCook2 data. Procedural steps are saved in the form of imperative English sentences and marked with their temporal boundaries.

Some relevant statistics about the data are recorded in Table 2

Table 2: Relevant dataset statistics for train and val sets.

Metric	Train	Validation
# of distinct recipes	89	89
Average # of videos per recipe	14.98	5.13
Total # of segments/steps	10337	3492

To construct our dataset, we use each procedural step as a data sample. We associate the frame corresponding to the middle of the temporal boundary of a step as the image for that step. All frames are resized to 256x256. Our dataset is then constructed by forming triplets of (procedural step, image, previous step's image). If a step is the first step in a procedure, the previous image does not exist, and thus is set to a blank image. During training, the procedural step and previous image are conditional signals to the Stable Diffusion model, and the desired output is the image associated with the step.

A.2 IMAGE-TEXT FUSION MECHANISMS

The architectures of the different image-text fusion mechanisms are shown in Figures 3 to 7. The output of the image encoder is 257x1024. The output of the text encoder is 77x768. In our experiments, we set the number of tokens in the condition signal to 3. So, the image generation module is ultimately conditioned on a sequence of 3 768-length vectors.

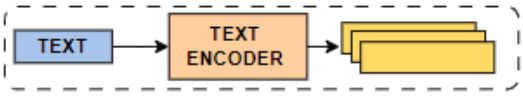

Figure 3: In the baseline Stable Diffusion model, there is no image-text fusion module as we only use text embeddings as the conditional signal.

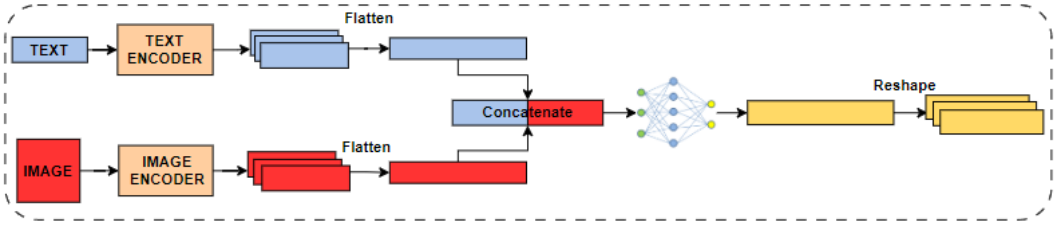

Figure 4: One method for fusion involves flattening the image and text embeddings, concatenating them, passing the concatenated vector through a neural network, and finally reshaping to sequence of tokens of appropriate conditioning size.

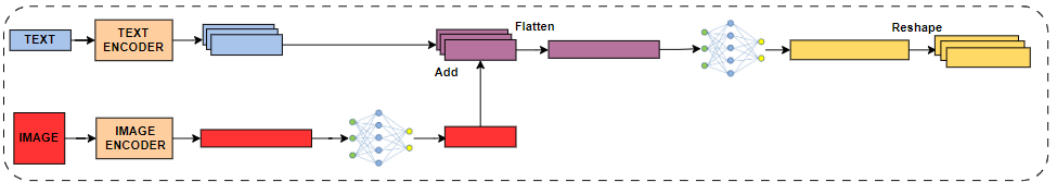

Figure 5: Image pooling involves extracting the pooled image embedding, passing it to a neural network to be of appropriate size, and adding the image vector to each token of the text encoder output. The result is then flattened and passed to a neural network, and finally reshaped to a sequence of tokens of appropriate conditioning size.

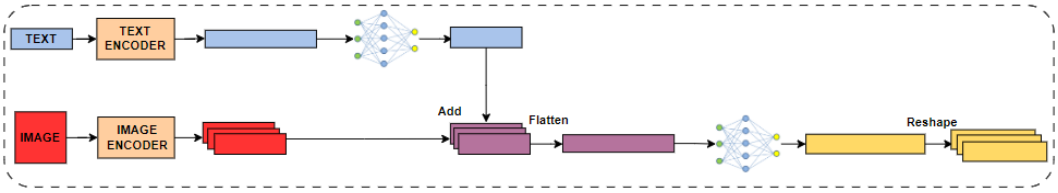

Figure 6: Text pooling involves extracting the pooled text embedding, passing it to a neural network to be of appropriate size, and adding the text vector to each token of the image encoder output. The result is then flattened and passed to a neural network, and finally reshaped to a sequence of tokens of appropriate conditioning size.

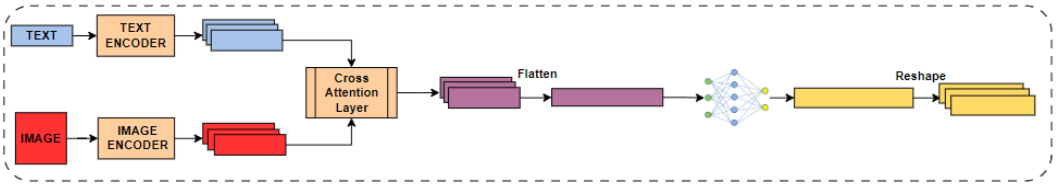

Figure 7: The attention module involves cross attention between the image and text embeddings. In attending to text, the query vectors are image embeddings, and keys/values are text embeddings. In attending to image, the query vectors are text embeddings, and keys/values are image embeddings.

A.3 Training and Inference Pipelines

Training: Each experiment utilized a single NVIDIA A40 GPU. The fuser modules were initialized with random weights, while the UNet parameters were initialized from the pretrained `CompVis/stable-diffusion-v1-4` model. The image encoder employed was the frozen OpenAI CLIP model (`openai/clip-vit-large-patch14`). The training configuration included a batch size of 32 and a total of 100,000 training steps. The learning rate followed a cosine decay schedule, starting at 3×10^{-7} for both the UNet and fuser modules. We use the DDPM

scheduler to add noise to the latents according to the noise magnitude at each timestep during the forward diffusion process.

Inference: During inference, we use 50 DDPM inference steps and guidance scale 7.5.

A.4 CYCLE CONSISTENCY

To ensure stronger alignment between image and text modalities, we incorporate an additional loss term to the standard denoising objective. This loss term penalizes the distance between text and image representations in the CLIP model's output space. Our method is visualized in Figure 8

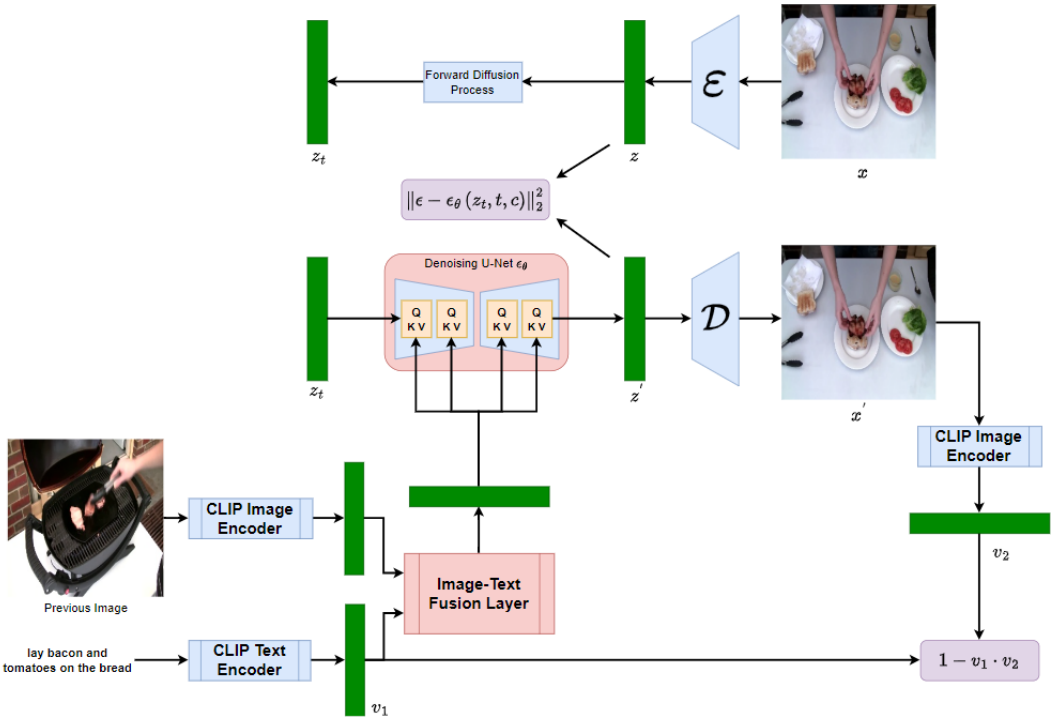

Figure 8: Our method optimizes the standard denoising objective and an auxiliary cycle consistency loss. The 2 losses are shown in purple.

A.5 SAMPLE RESULTS

Our best performing model across all metrics comes from attending to image as our text-image fusion module. Training with cycle consistency loss improves CLIP comparison score and PSNR slightly, but lowers SSIM score slightly. Qualitatively, samples are best when using the Attend to Image strategy.