CLIP-DIFFUSIONLM: APPLY DIFFUSION MODEL ON IMAGE CAPTIONING *

Shitong Xu
Imperial College London
shitong.xu19@imperial.ac.uk

ABSTRACT

Image captioning task has been extensively researched by previous work. However, limited experiments have focused on generating captions based on non-autoregressive text decoder. Inspired by the recent success of denoising diffusion model on image synthesize tasks, we applied denoising diffusion probabilistic models to text generation in image captioning tasks. We show that our CLIP-diffusionLM *** On the flickr8k dataset, the model showed. By combining samples from flickr8k and flickr30k dataset, our model showed ... performance. In addition, the model achieved ... zero shot performance in COCO 2015 image caption task. Our code is available at ...

contribution: experiment on learning rate, optimizer, adding mechanism, cosine schedule, using classifier free or not. using model to predict image feature or not, scale up to larger dataset

***K*eywords** Diffusion model · CLIP · Non-autoregressive text generation

1 Introduction

Image captioning has being a focus of research over the recent years. Among the previous proposed works, the text encoder used could be split to 2 general classes, i.e. autoregressive and non-autoregressive class. Most of the sota models falls in the autoregressive class[**?**, **?**, **?**, **?**, **?**]. However, autoregressive generation suffer from 1) the slow generation speed due to the generation step is token by token; and 2) not capable of refining prefix of sentences based on the later generated tokens. Multiple attempts have experimented using a non-autoregressive model in the text generation steps[**?**, **?**, **?**]. The closest work to ours is Masked Non-Autoregressive Image Captioning by Gao et al. [**?**], which used a BERT model as generator and involves 2 steps-refinement on the generated sequence. Masked Language Model (MLM) is used in their work to supervise the generation of captions.

In contrast to MLM, which is a language model based on discrete tokens embedding prediction, diffusion models based on continuous latent embedding has been thriving in image and audio generation tasks[**?**, **?**, **?**, **?**, **?**]. To the best of our knowledge, there has not been previous work on generating caption embedding based on diffusion language model. Our work aim at employing a model to refine generated token continuously on sequence embedding, and provide empirical insight on useful tricks to improve the generated captions. In particular, we used pretrained CLIP[**?**] model for extracting image and text features, and DistilBert[**?**] model based on diffusion-lm[**?**] for text sequence generation. Our contribution consists of proposing a diffusion based image captioning model. In addition, we experiment on effectiveness of multiple model design and hyperparameter setting on CLIP-DiffusionLM, including weight assign in loss function terms, feature fusion method, learning rate and classification free guidance.

2 Related Work

2.1 Autoregressive image captioning

Mao et al. proposed the mRNN model[**?**], which used CNN for image extraction and RNN for text generation. Xu et al. [**?**] employed the LSTM for text generation and experimented on soft and hard attention for early fusion between

* *Citation*: **Authors. Title. Pages.... DOI:000000/11111.**

image feature and text feature. Based on this early fusion method, Lu et al. [**?**] experimented with the late fusion of image and text features, allowing model to attend on either image or text modality in late phase of generation. Wu and Hu [**?**] experimented on reversing the generated caption, allowing its backend model to refine the former tokens based on later caption tokens. Le et al. [**?**] used attention model to combine local and global feature from images, so that captions can more accurately identify occluded objects. Similarly, Wei et al. [**?**] also used image features from both high and low generaliaty, and combined them using cross attention. Their work also involves multi-step refining of the generated text caption. Feng et al. [**?**] trained image caption in a GAN framework, with a LSTM discriminator reproducing the original image feature from generated text sequence. Similrarly, Guo et al. [**?**] proposed GAN based method to train model predicting stylized text. Multiple discriminators are used to supervise if generated text captured image related feature, in the desired style, and similar to a caption made by human. Kim et al. [**?**] used variational auto encoder for extracting image information, their model allows multi caption generation by sampling from the learned image feature distribution, thus produce various captions for a single image. He et al. [**?**] used POS tagging to help the generation of text. The image feature is used as additional input when the model is predicting tokens related to image-specific information, i.e. object, colour, relative position of objects. Mokady, Hertz and Bermano [**?**] experimented on using pretrained CLIP image feature for sequence generation. The CLIP features are transformed to a sequence of token and used as prefix for a GPT-2 model in generation. Li et al. [**?**] introduced skipped connections between transformer layers to address the information asymmetry between vision and language modality. The model achieved state of the art performance and strong zero shot ability on various tasks. Nguyen et al. [**?**] experimented changing the cross attention part of transformer decoder to use both Reginal feature from Faster RCNN and Grid features from swin transformer.

2.2 Non autoregressive image captioning

In contrast, non-autoregressive models benefits from the attention models' ability to pass textural information in both direction during generation. The text generated in former timesteps could adjust based on text in later timesteps, thus is expected to achieve better performance. Gao et al.[**?**] used BERT[**?**] as text decoder and employed a 2 step generation method. Based on this work, Partially Non-Autoregressive Image Captioning by Fei [**?**] and semi Non-Autoregressive Image Captioning by Xu et al. [**?**] partitioned the generated text in subgroups. Words in the same group are generated non-autoregressively and different groups are generated in autoregressive method. Our model falls in non-autoregressively category and is most close to the Masked Non-Autoregressive Image Captioning[**?**]. The difference is we chose to use diffusion model as the non-autoregressive generation model.

2.3 Diffusion models

Diffusion models aims at training a model that denoise Gaussial noise incrementally to reproduce original features. Ho, Jain and Abbeel[**?**] proposed the Denoising Diffusion Probabilistic Model (DDPM) to simplified the loss function by only letting models to predict the noise in generation steps, and proposed alternative loss functions by removing the weight terms. In the following explanation, we refer to DDPM as diffusion model for simplicity. Nichol and Dhariwal[**?**] proposed several improvements based on DDPM, including setting variance to be learn-able parameters, apply cosine instead of linear noise schedule, and speed up forward process by reducing forward steps. Song, Meng and Ermon[**?**] experimented on reducing the variance in forward process. The result showed that by reducing variance to 0, the deterministic model achieved higher FID score in image generation on both CIFAR10 and CelebA. DiffusionLM by Li et al. [**?**] is a recent work on applying continuous diffusion model on text generation. DiffusionLM explored various techniques to improve the performance of continuous diffusion model on text generation. Dhariwal and Nichol [**?**] proposed classifier guidance for improving generated image FID score. In a classifier guided diffusion model, a classifier model is pretrained to predict noised images' object class. During training, the classifier provides gradient on which direction to optimise the generated image, so that the generate image resembles an object closer to the target class.

To avoid training classifier for guiding model, Jonathan and Tim [**?**] proposed classifier free guidance model. In classifier free guidance, the difference in output of generative model when provided with both guided and unguided context information is used as implicit guidance. By using classifier free diffusion model as text-to-image generator, DALL-E2[**?**], GLIDE[**?**] and High-Resolution Image Synthesis With Latent Diffusion Models[**?**] model achieved significant image generation performance. In particular, DALL-E2 used CLIP model for extracting feature from text, predict the corresponding image CLIP feature through prior network, then use predicted image CLIP feature for final image generation. The model achieved significant novelty in generated images and also inspired us to train a image-to-text model with diffusion model in generation step.

2.4 CLIP model

CLIP model is a contrastive-learning-based model trained on WebImageText dataset. The WebImageText consists of 400 million image-text pairs collected from public available source on Internet. CLIP model demonstrated strong zero shot performance in their evaluation on ImageNet[] dataset. Further work on CLIP model also showed its transferability to other tasks, including image segmentation[**?**, **?**], objection detection[**?**, **?**], action recognition[**?**] and video-text retrieval[**?**].

3 Background

3.1 Diffusion models

The training of denoise diffusion probabilistic model involves generation of noised samples (forward process), and model denoising Gaussian noise back to the original feature (backward process). Let x_t represent the intermediate representation in the diffusion generation stages. In our context, x_t is the sequence of word embedding of the caption. The forward process incremently add noise to the ground truth embedding x_0 to generates T noised features $[x_1, ...x_T]$. Each x_t at step t is sampled from probability distribution $q(x_t|x_{t-1}) = N(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$ and the final step feature x_T is aiming at approximate a Gaussian noise. From reparameterization trick, the x_t at any step t could be directly sampled from x_0: $x_t = \sqrt{\bar{\alpha}_t}x_0 + (1-\bar{\alpha}_t)\epsilon$, where $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$ and ϵ follows a Multivariant Gaussian Normal distribution. The backward process involves training model with parameter θ to denoise the samples generated in the forward process. The training objective is to minimize the negative log-likelihood of generating x_0 from arbitrary Gaussian noise x_T as generated by q, that is to minimize

$$E_q[-\log(p_\theta(x_0))] = E_q[-\log(\int p_\theta(x_{0:T}), d(x_{1:T}))]$$

By optimizing the variational lower bound of $p_\theta(x_0)$ instead, and modeling the backward process as a Markov Process gives:

$$\begin{aligned} E_q[-\log(p_\theta(x_0))] &\leq E_q[\log(\frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)})] \\ &= E_q[\log(p(x_T)) + \sum_{t=1}^{T}\log(\frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})})] \end{aligned}$$

where $p_\theta(x_{t-1}|x_t) = N(x_{t-1}; \mu_\theta(x_t, t), \sum_\theta(x_t, t))$ and $\mu_\theta(x_t, t)$ is model's prediction on mean of x_{t-1} conditioned on x_t. From the work of [], expanding and reweighting each term of the negative log-likelihood gives a concise loss function

$$L_{simple} = \sum_{t=1}^{T} E_{q(x_t|x_0)}\|\mu_\theta(x_t, t) - \hat{\mu}(x_t, x_0)\|^2$$

where $\hat{\mu}(x_t, x_0)$ is the mean of posterior $q(x_{t-1}|x_t, x_0)$.

Due to the large generation step number (T = 1000 as proposed in [**?**]), and the generation step being autoregressive on the denoised feature in the previous step, the reverse diffusion is significantly slower than the other generative models[**?**, **?**]. Multiple stetegies were proposed to accelerate the generation process. Improved DDPM proposed by Nichol and Dhariwal[**?**] used a subset of generation steps $\{s_0, ..., s_N|s_t < s_{t-1}\} \in (0, T]$. Model is trained to predict $x_{s_{t-1}}$ based on x_{s_t}. DiffusionLM proposed by Li et al.[**?**] is trained directly to predict the x_0 instead of an denoised intermediate steps x_{t-1}.

In addition, based on the objective function propose by Li et al.[**?**], we added an additional rounding term and a x_1 restoring loss term. x_1 restoring loss $\|\mu_\theta(x_1, 1) - \hat{\mu}(x_1, x_0)\|^2$ is evaluating the performance of model on restoring x_1. For simplicity, in the following explanation and experiments, we evaluate the x_1 restoring loss together with L_{simple} and refer to their sum as L'_{simple}. Rounding term L_R is parameterized by $E_q[-\log(p_\theta(w|\hat{x}))] = E_q[-\log(\prod_{i=1}^{l} p(w_i|\hat{x}_i))]$, where l represents the generated sequence length, w represent the gound truth sentence and $\hat{x}$ is the predicted sequence embedding from the input x_t. $p_\theta(w_i|\hat{x}_i)$ follows the softmax distribution. In our experiments, we found the relative importance between rounding term and restoring embedding loss L'_{simple} significantly influence the model performance. The relative importance is addressed by the hyperparameter λ as rounding term coefficient and discussed in detail. Based on the above 2 modifications, our training objective is as follow:

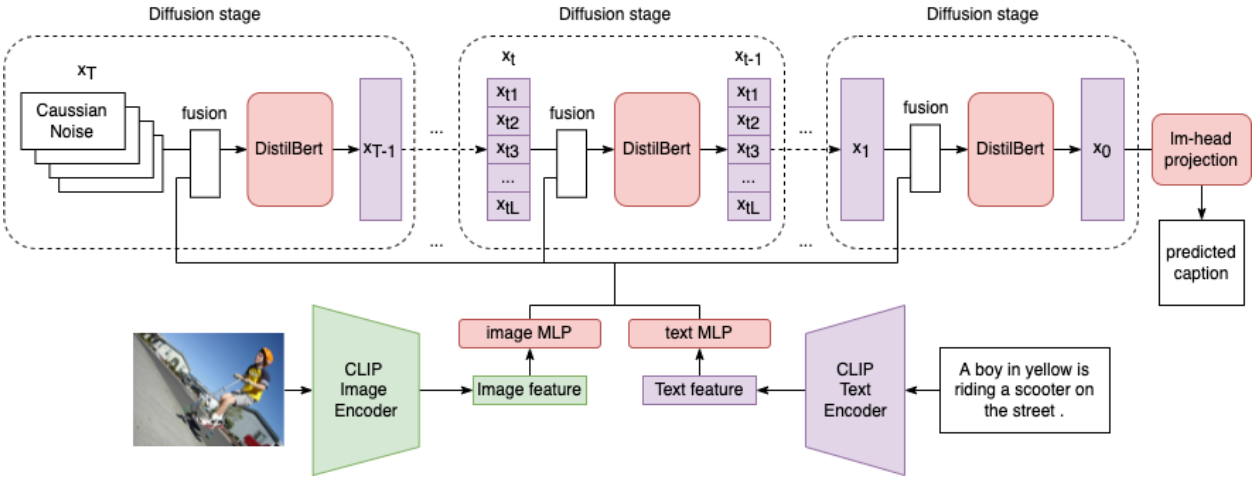

Figure 1: CLIP-DiffuseLM model

$$L_{diffuseLM} = L'_{simple} + \lambda L_R = \sum_{t=1}^{N} E_{q(x_{s_t}|x_0)}[\|\mu_\theta(x_{s_t}, s_t) - \hat{\mu}(x_{s_t}, x_0)\|^2 - \lambda \log(p_\theta(w|\hat{x}))]$$

4 CLIP-DiffusionLM

Our CLIP-DiffusionLM model consists of 2 parts. A pretrained CLIP (ViT-B/32) model is used for image and caption label feature extraction, and a DistilBert-base-uncased model for performing denoising diffusion in each diffusion step conditioned on CLIP image feature.

In a forward process, the CLIP model first extract image features using a ViT-Base/32 architecture[]. In this model, source RGB image is partitioned into 32 patches and prefixed with a <CLS> token to form the input of the 12 ViT layer model. Output feature at the <CLS> token position is treated as the global feature for the whole image. For classification free experiments, CLIP text feature extraction is also performed on the caption labels and used as guided context to improve the performance. In caption feature extraction, a modified transformer encoder model is used(Language Models are Unsupervised Multitask Learners, attention is all you need[]). Activation at the highest transformer layer in <EOS> token is used as the global caption feature. In our following experiments, both CLIP text and image branches parameters are not optimized in the backward propagation.

The generation of caption for a given image consists of a sequence of diffusion stages conditioned on CLIP feature extracted. In each stage, CLIP-DiffusionLM receives input of $L \times D_{word}$ dimension vectors x_t as embedding of a L length caption, and two D_{CLIP} dimension vector for CLIP text and image features. Input caption embedding is either the output embedding x_t in the previous diffusion stages, or the Gaussian noise x_T in case of the first diffusion stages. Each of CLIP features is projected by a MLP layer to D_{word} space, before fuse with the sequence embedding x_t. The fused sequence is the input of the DistilBert model, and output of the DistilBert model is the output of one single diffusion stage, which consists of the prediction for x_{t-1}. After the final diffusion stage, the model's prediction for x_0 is linear projected by weight matrix `lm-head` and taken the Softmax value to get the probability of the predicted word in each L length caption sequence position. Finally, each position's word with maximum probability are concatenated to form the output text caption sequence. Unless otherwise specified, both the embedding layer for extracting labeled caption embedding and the `lm-head` use pretrained DistilBert model embedding layer parameter and are not optimized in model training.

5 Experiments

We trained our model using Flickr8k[] and Flickr30k[] dataset on a single Nvidia A30 GPU. Our final model used ?? $\lambda = 0.3$, x_0 prediction, learning rate of 5e-5 and optimized by AdamW. The CLIP image feature is concatenated to each timestep of end of each diffusion step output for feature fusion. In the following experiments, unless otherwise specified, we performed training on Flickr8k dataset using the above model configuration.

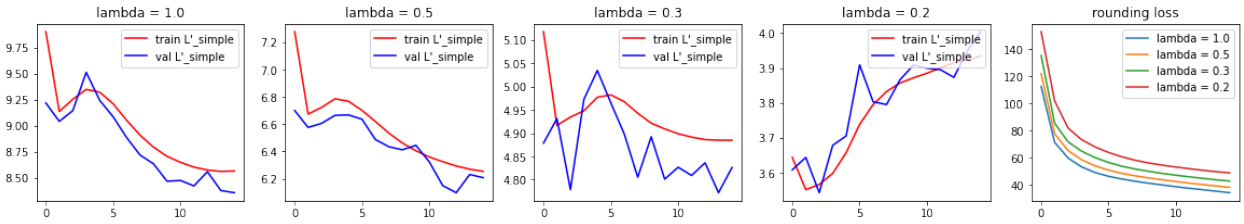

Figure 2: constant L_R coefficient loss

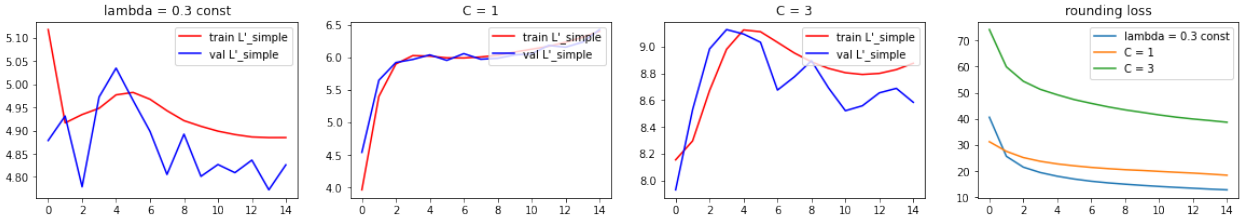

Figure 3: dynamic L_R coefficient scheduling loss

unique seq, inference step, add or concat, training time, dataset 30+8

w	p_{uncond}	L'_{simple}	L_R	BLEU-4
non classifier-free guidance		4.89	12.87	0.1549
0.3	0.2	4.92	13.49	0.1539
1.0	0.2	4.89	13.06	0.1558

Table 1: classifier free guidance

λ	BLEU-4
1.0	0.1635?
0.5	0.1664?
0.3	0.1549
0.2	0.1614?

Table 2: constant L_R coefficient BLEU

5.1 relative importance of Rounding term and Embedding-restoring loss term L_{simple}

In this session we explain our choice of rounding coefficient hyperparameter λ. We introduce the coefficient in the expectancy of controlling model to optimize primarily based on either the L'_{simple} or the L_R loss. $\lambda \in \{1.0, 0.5, 0.3, 0.2\}$ were experimented on as shown in Fig?. When λ decrease, the model showed a decreasing trend in its L'_{simple} loss (from 8.56 to 4.88) at the cost of increasing L_R loss (from 34.4 to 48.9). In addition, setting λ to lower than 0.3 showed increase in L'_{simple} loss, which means model is increasing the probability of output the ground truth sequence while not captioning the similarity between the predicted word and ground truth words. Though we found the final L'_{simple} of $\lambda = 0.2$ (3.93) lower that of $\lambda = 0.3$ (4.88), BLEU score of the model shows the opposite in Table?. In conclusion, $\lambda = 0.3$ showed highest BELU-4 score with relatively low L'_{simple} loss, and thus is chosen as default value in later experiments.

Furthermore, we explored if a dynamic λ scheduling method help the model adjust λ based on ratio between L'_{simple} and L_R. The dynamic scheduling method update λ value after each gradient descend step as $\lambda = L'_{simple}/L_R * C$, where C is a hyperparameter defining the relative weight kept between L'_{simple} and L_R throughout the training. To choose a reasonable C value, we examined the $\lambda = 0.3$ experiment final epoch's $\frac{L'_{simple}}{L_R}$ value, which is around 2.80. Based on this observation, we took $C = 1$ and expects both L'_{simple} and L_R loss decrease at the same rate and reach $\lambda \approx 0.3$ in the final epoch. Same reasoning applies to the choice of $C = 3$, which aims at reaching $\lambda \approx 1.0$ in the final epoch.

We plot the dynamic scheduled loss in Fig?. The relative ratio between L'_{simple} L_R did not follow the dynamic ratio defined as expected. A possible explanation is that the loss is taken before gradient decent is performed, and evaluation of loss is based on the next batch sample, result in the logged $\frac{L'_{simple}}{L_R}$ value not following the expected relative loss ratio. In both experiments, the model showed worse performance with dynamic λ scheduling enabled. In our further experiments, the non-dynamic scheduled $\lambda = 0.3$ is used.

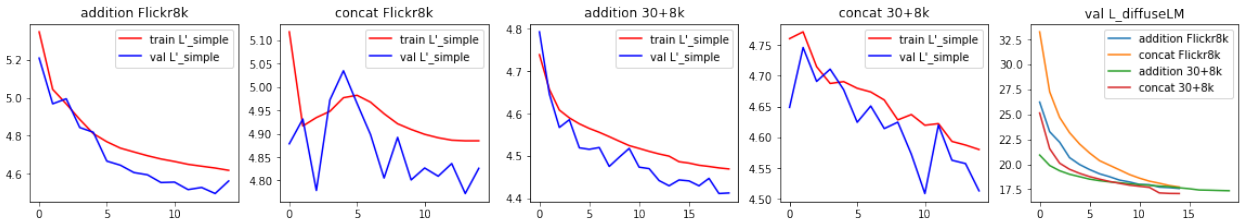

Figure 4: concatenation or element-wise addition fusion

5.2 classification free guidance

The effectiveness of applying classification free guidance is examined on our model. The guidance provided is the CLIP text feature of ground truth label caption. Fig? shows the comparison between the baseline and classification free guidance trained model. In contrast to previous success of applying classification free guidance, our model failed to improve significantly than none classification guided baseline. We further test the 2 models' performance by evaluating BLEU score on validation set Table?. The model trained using classifier free guidance hardly outperformed the simpler baseline model, with the parameter suggested in [**?**] even result in slight decrease in BLEU-4 score. As a result, classification free guidance was not used in final model.

5.3 learning rate

TODO: retry with bleu on 1e-4 running - 5e-5 use above result lin schedule, use same metrics as log which is 0.1632

In case of learning rate defining and scheduling, we experimented on constant learning rate of $lr \in \{5e-4, 1e-4, 5e-5, 1e-5\}$. In addition, we experimented on linear, log, cosine learning rate scheduling[] between the best-performing learning rate $1e-4$ and $5e-5$.

In constant learning rate experiments, $lr = 5e-5$ achieved the highest BLEU-4 score (), and $lr = 1e-4$ achieved the lowest l'_{simple} loss (4.69). In the following experiments, $lr = 5e-5$ was taken as default learning rate and $lr = 1e-4$

In linear and log learning rate scheduling, model learning rate is updated after every epoch. The 2 scheduling methods differes mainly on the , where log scheduling decrease learning rate at faster than linear scheduling but allow model to refine with low learning rate in the final epoches.

Based on the above experiments, model suprisingly perform better with higher learning rate and fail to converge to a lower loss value when a small . We further perform cosine scheduleing of learning rate, allow the learning rate decrease to a low value before reset to a high value every 5 epoches in the 15 epoch training. The performance is comparable with the baseline. To simply the process, we kept learning rate as a $1e-4$ constant in further experiments. A probable cause is model optimzie on information relate to L'_{simple} and L_R in different rate.

5.4 x_0 prediction or $x_{s_{t-1}}$ prediction

In DiffusionLM proposed by Li et al.[**?**], model is trained to predict x_0 instead of $x_{s_{t-1}}$ conditioned on x_{s_t}. In our experiments, the x_0-prediction method showed better BLEU-4 score (0.1632) and $L_{diffuseLM}$ loss (17.8) performance compared with $x_{s_{t-1}}$-prediction (0.1575 BLEU-4 score and 22.6 $L_{diffuseLM}$ loss). In addition, by following the x_0-prediction method, our model could converge to a reasonable output in less than 5 diffusion steps. This step number is significantly less than the autoregressive methods using an encoder-decoder structure, which involves generation steps proportional to the output sequence length.

5.5 fusion: concatenation or element-wise addition

We experimented with 2 fusion methods: concatenation or element-wise addition. Concatenation append the CLIP features as additional tokens to the end of the caption embedding x_t. An additional segment embedding layer is used to distinguish CLIP feature tokens with caption embedding. In contrast, element-wise addition method use CLIP image feature as position embedding. CLIP image feature is element-wise added to caption word embedding in each timestep. In case of Guidance free training, the CLIP text feature is also element-wise added. In later explanation, element-wise addition is referred to as addition method for simplicity.

We conduct experiments by running both concatenation and addition fusion. Fig? shows their performance on Flickr8k and Flickr 30+8k dataset. The addition method experience less violent change in L'_{simple} loss, and converged to a lower loss before overfitting. Extending the training dataset to use Flickr 30+8k showed similar result.

However, by testing the BLEU-4 score on both method trained on Flickr 30+8k dataset, the addition method showed 0.1948 BLEU-4 score, while concatenation method showed 0.2337 BLEU-4 score, indicating the addition method might method is less prone to overfitting. As a result, concatenation method is chosen despite its higher converging loss and instability.

6 Conclusion

We present the application of diffusion in image caption task, and proved its validity on Flickr8k and Flickr30k dataset. Particularly, we identified the importance of rounding term in loss function to help model converge, and introduced the adaptive ratio adjustment to balance its importance with other terms. There are various improvements to the model and training process: - For given predicted word, investigate on the region in image that resulted in the generation of the token. of focus on for given Experiment on output the attention graph of the model, to check the model did focus on the correct region of the image. - In various cases, the model failed to identify the correct object colour. For example, after correctly identify a girl wearing dress and a shirt, the model mistake the colour of shirt to the colour of the dress. - The output text sequence suffers from apparent grammar mistakes, for example, missing subject, repeated words. Additional supervision on the output text grammar might help model reduce such error. - We trained on raw image data of the dataset. However, image argumentation has proven to be a valid method to improve the performance[]. Performing data argumentation might improve the model's generalizability and help reduce the wrong colour alignment problem as discussed above. - Due to the computation limit, we are not able to train an model capable of generating caption longer than 16 characters. This also influenced the BLEU performance, since BLEU pelalize sequences with length in their ... coefficient. - in various cases bleu performance and loss does not match, suggest better loss function to define model's performance. We believe analysing and improving based on the above observations, diffusion as text generation step could achieve comparable or better performance than auto-regressive models.

Acknowledgments

We would like to thank Mu Li and Yi Zhu for sharing their insight in various models in vision and NLP field publicly online, Boyang Gu for providing advice in early stage of the research. The computation resource was supported by Imperial College London.