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. Previous 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 to our work is 2019 Masked Non-Autoregressive Image Captioning[], which used a BERT model as generator and involves 2 steps-refinement on the generated sequence. Masked Language Model 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 emperical 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

2015 deep caption with multimodal rnn[] proposed the mRNN model, which used CNN for image extraction and RNN for text generation. 2016 show attend tell[] employed the LSTM for text generation and exerimented on soft and hard

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

attention for early fusion between image feature and text feature. Based on this early fusion method, 2016 knowing where to look[] experimented the late fusion of image and text features, allowing model to attend on either image or text modality during generation. 2017 cascade recurrent nn[] experimented on reversing the generated caption, allowing its backend model to refine the former tokens based on later caption tokens. 2018 gla[] used attention model to combine local and global feature from images, so that captions can more accurately identify occluded objects. Similarly, 2019 stack vs[] 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. 2019 unsupervised image caption[] trained image caption in a GAN style, with a LSTM discriminator reproducing the original image feature from generated text sequence. Similrarly, 2019 mscap[] 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. 2019 Variational Autoencoder-Based Multiple ImageCaptioning Using a Caption Attention Map[] 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. 2019 image caption generation with pos[] 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. 2021 CLIP cap[] 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. 2022 mplug[] 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.

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. 2019 Masked Non-Autoregressive Image Captioning[] used BERT[] as text decoder and employed a 2 step generation method. Based on this work, Partially Non-Autoregressive Image Captioning [] and semi Non-Autoregressive Image Captioning[] partitioned the generated text in subgroups, words in the same group are generated non-autoregressively and different groups are generated in autoregressive way. Our method falls in this catogory and most close to the 2019 Masked Non-Autoregressive Image Captioning[]. The difference is we chose to use diffusion model as the non-autoregressive generation model. 2022 GRIT[] experimented changing the cross attention part of transformer decoder to use both Reginal feature from Faster RCNN and Grid features from swin transformer.

2.3 Diffusion models

Diffusion models aims at training a model that denoise a feature from Gaussial noise to original features. Ho, Jain and Abbeel[**?**] proposed the DDPM model 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. Based on DDPM, improved ddpm [] proposed several improvments, including setting variance to be learnable parameters, apply cosine instead of linear noise schedule, and speed up forward process by reducing forward steps. ddim[] reduced 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. Diffusion lm[] is a recent work on applying continuous diffusion model on text generation, this paper provides various techniques to improve the performance of continuous diffusion model on text generation. Diffusion model beat GAN[] 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. Noise on images are added as Gaussian noise to simulate the noise output in each step of diffusion generation. To guide the generation, 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, classifier free guidance technique is proposed Classifier-Free Diffusion Guidance[]. 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. Example of applying classifier free guidance includes GLIDE[], DALL-E2[], High-Resolution Image Synthesis With Latent Diffusion Models[]

By using diffusion model as text-to-image generation, DALL-E 2 and GLIDE model achieved significant image generation performance. Dall-e 2[] is a recent work on using CLIP and diffusion model for image generation task. The model used CLIP model for extracting feature from text, predict the corrisponding image CLIP feature through prior network, then use predicted image CLIP feature for final image generation. The model achieved significant novelty in generated images. The innovativity of generated of image from DALL-E 2 also provided us the inspiration 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 ... dataset. Further work on CLIP model also showed its transferability to other tasks, including image segmentation[], action recognition[], object detection[].

3 Background

3.1 Diffusion models

The training of denoise diffusion probabilistic model involves generation of noised samples (forward process), and denoising based on model's output (backward process). Let x_t represent the intermediate representation in the diffusion generation steps. 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 arbitary 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[gan, vae]. Multiple stetegies were proposed to accelerate the generation process. In Improved DDPM [] a subset of generation steps $\{s_0, ..., s_N | s_t < s_{t-1}\} \in (0, T]$ is selected. Model is trained to predict $x_{s_{t-1}}$ based on x_{s_t}. In diffusion-lm the model is trained directly predict the x_0 instead of the intermediate steps containing noise.

In addition, based on the objective function propose in[], we added an additional rounding term and a x_1 restoring loss term to L_{simple}. 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 term together with other restoring loss terms and refer to their sum as L'_{simple}. Rounding term L_R is parameterized by $E_{p_\theta(\hat{x}|x_t)}[-\log(p_\theta(w|\hat{x}))] = E_{p_\theta(\hat{x}|x_t)}[-\log(\prod_{i=1}^{l} p(w_i|\hat{x}_i))]$, 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:

$$L_{diffuseLM} = \sum_{t=1}^{N} E_{q(x_{s_t}|x_0)}[\|\mu_\theta(x_{s_t}, s_t) - \hat{\mu}(x_{s_t}, x_0)\|^2 - \lambda\log(p_\theta(w|\hat{x}))]$$

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 steps conditioned on CLIP feature extracted. In each step, CLIP-DiffusionLM receives input of $L \times D_{word}$-dim vectors x_t as embedding of a L length caption sequence, and two D_{CLIP}-dim vector for CLIP text and image features. Caption sequence embedding is either the output embedding x_t in the previous diffusion step, or the Gaussian noise x_T in case of the first diffusion step. Each of CLIP features is projected to D_{word} space by passing through a MLP layer, 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 each diffusion step, which consists of the prediction for x_{t-1}. After the final diffusion step, the model's prediction for x_0 is linear projected by weight `lm-head` and taken the softmax value to get the probability of the predicted word in each L length caption sequence position. 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. Finally, each position's word with maximum probability are concatenated to form the final caption text sequence.

?? caption latent embedding caption embedding ?? graph

5 Experiments

We trained our model on a single Nvidia A30 GPU. Our final ... model with ... took around ... hours. In the following experiments, unless otherwise specified, we performed training on Flickr8k dataset using the above model configuration.

5.1 fusion: concatenation or element-wise addition

We experimented with 2 fusion methods: concatenation or element-wise adding. Concatenation append the CLIP features as 2 additional tokens at the end of the caption latent embedding. An additional segment embedding layer is used to distinguish CLIP feature tokens with caption latent embedding tokens. 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 each timestep.

We conduct experiments by running both concatenation and element-wise addition fusion. Fig. shows their performance on Flickr8k dataset. Element-wise addition experience significantly less violent change in restoring loss, and converged to a lower loss before overfitting. Extending the training dataset to use both Flickr30k and Flickr8k showed similar result.

However, by testing the BLEU-4 score on generated captions, the ....

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

In this session we explain our choice of rounding coefficient hyperparameter. We experimented with $\lambda \in \{1.0, 0.5, 0.3, 0.2\}$ as shown in Fig. In general, the total loss $L_{diffuseLM}$ before converge decreased from around to ... In addition, decreasing λ to lower than 0.2 showed increase in L'_{simple} loss, which means model is maximizing the probability of each predicted word while not . Though we found the final L'_{simple} of $\lambda = 0.2$ lower that of $\lambda = 0.3$, BLEU score of the model shows the opposite Table....

Furthermore, to avoid the model only optimize on rounding term loss L_R, we proposed a dynamic λ scheduling method, The method update λ value after each gradient desend step as $\lambda = L'_{simple}/L_R * C$, where C is another hyperparameter defining the relative weight kept between L'_{simple} and L_R throughout the training. In our experiments, $C \in \{3, 1\}$ are chosen, due to the undynamic scheduled case showed ratio $\frac{L'_{simple}}{L_R}$ before converge range in []. We plot the dynamic scheduled loss in Fig.... The model did perform with constant loss ratio between L'_{simple} and L_R. However, the model didn't show improvements compared with the undynamic scheduled baseline. In our further experiments, the undynamic scheduled $\lambda = 0.3$ hyperparameter is used.

5.3 guidance free training

The effectiveness of applying classification free guidance is preformed 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 guidance baseline. We further test the 2 models' performance by evaluating BLEU score on validation set Table.... The simplier baseline model out performed...

5.4 learning rate

In case of learning rate defining and scheduleing, 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,

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.

5.5 x_0 prediction or x_{t-n} prediction

In our experiments, the ... showed better reproduce quality compared with... . 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 which is significantly less than autoregressive methods using an encoder-decoder structure. , which has generation steps propotional to the output sequense length. autoregressively apply the output of x_0 trained model further improved the performance.

5.6 number of x_t predictions

6 Conclusion

We present the application of diffusion in image caption task, and proved its validity in limited dataset. Particularly, we identified the certainty term to be an important 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: - 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. 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.

$$\xi_{ij}(t) = P(x_t = i, x_{t+1} = j|y, v, w; \theta) = \frac{\alpha_i(t) a_{ij}^{w_t} \beta_j(t+1) b_j^{v_{t+1}}(y_{t+1})}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i(t) a_{ij}^{w_t} \beta_j(t+1) b_j^{v_{t+1}}(y_{t+1})} \tag{1}$$

Paragraph Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

7 Examples of citations, figures, tables, references

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Donec odio elit, dictum in, hendrerit sit amet, egestas sed, leo. Praesent feugiat sapien aliquet odio. Integer vitae justo. Aliquam vestibulum fringilla lorem. Sed neque lectus, consectetuer at, consectetuer sed, eleifend ac, lectus. Nulla facilisi. Pellentesque eget lectus. Proin eu metus. Sed porttitor. In hac habitasse platea dictumst. Suspendisse eu lectus. Ut mi mi, lacinia sit amet, placerat et, mollis vitae, dui. Sed ante tellus, tristique ut, iaculis eu, malesuada ac, dui. Mauris nibh leo, facilisis non, adipiscing quis, ultrices a, dui. [**?**, **?**] and see [**?**].

The documentation for `natbib` may be found at

`http://mirrors.ctan.org/macros/latex/contrib/natbib/natnotes.pdf`

Of note is the command `\citet`, which produces citations appropriate for use in inline text. For example,

```
\citet{hasselmo} investigated\dots
```

produces

> Hasselmo, et al. (1995) investigated…

`https://www.ctan.org/pkg/booktabs`

7.1 Figures

Suspendisse vitae elit. Aliquam arcu neque, ornare in, ullamcorper quis, commodo eu, libero. Fusce sagittis erat at erat tristique mollis. Maecenas sapien libero, molestie et, lobortis in, sodales eget, dui. Morbi ultrices rutrum lorem. Nam elementum ullamcorper leo. Morbi dui. Aliquam sagittis. Nunc placerat. Pellentesque tristique sodales est. Maecenas imperdiet lacinia velit. Cras non urna. Morbi eros pede, suscipit ac, varius vel, egestas non, eros. Praesent malesuada, diam id pretium elementum, eros sem dictum tortor, vel consectetuer odio sem sed wisi. See Figure **??**. Here is how you add footnotes. [2] Sed feugiat. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Ut pellentesque augue sed urna. Vestibulum diam eros, fringilla et, consectetuer eu, nonummy id, sapien. Nullam at lectus. In sagittis ultrices mauris. Curabitur malesuada erat sit amet massa. Fusce blandit. Aliquam erat volutpat. Aliquam euismod. Aenean vel lectus. Nunc imperdiet justo nec dolor.

7.2 Tables

Etiam euismod. Fusce facilisis lacinia dui. Suspendisse potenti. In mi erat, cursus id, nonummy sed, ullamcorper eget, sapien. Praesent pretium, magna in eleifend egestas, pede pede pretium lorem, quis consectetuer tortor sapien facilisis magna. Mauris quis magna varius nulla scelerisque imperdiet. Aliquam non quam. Aliquam porttitor quam a lacus. Praesent vel arcu ut tortor cursus volutpat. In vitae pede quis diam bibendum placerat. Fusce elementum convallis neque. Sed dolor orci, scelerisque ac, dapibus nec, ultricies ut, mi. Duis nec dui quis leo sagittis commodo. See awesome Table **??**.

7.3 Lists

- Lorem ipsum dolor sit amet

[2] Sample of the first footnote.

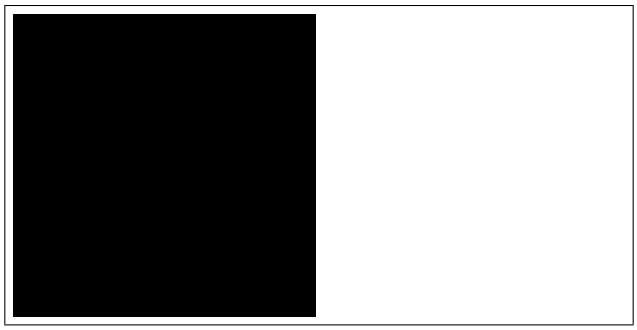

Figure 1: Sample figure caption.

Table 1: Sample table title

	Part	
Name	Description	Size (μm)
Dendrite	Input terminal	$\sim$100
Axon	Output terminal	$\sim$10
Soma	Cell body	up to 10^6

- consectetur adipiscing elit.
- Aliquam dignissim blandit est, in dictum tortor gravida eget. In ac rutrum magna.

8 Conclusion

Your conclusion here

Acknowledgments

This was was supported in part by.....