Explicit Image Caption Editing

Zhen Wang[1*], Long Chen[2*], Wenbo Ma[1], Guangxing Han[2], Yulei Niu[2], Jian Shao[1], and Jun Xiao[1†]

[1]Zhejiang University [2]Columbia University
zju_wangzhen@zju.edu.cn, zjuchenlong@gmail.com, junx@cs.zju.edu.cn

Abstract. Given an image and a reference caption, the image caption editing task aims to correct the misalignment errors and generate a refined caption. However, all existing caption editing works are *implicit* models, *i.e.*, they directly produce the refined captions without explicit connections to the reference captions. In this paper, we introduce a new task: Explicit Caption Editing (ECE). ECE models explicitly generate a sequence of *edit operations*, and this edit operation sequence can translate the reference caption into a refined one. Compared to the implicit editing, ECE has multiple advantages: 1) Explainable: it can trace the whole editing path. 2) Editing Efficient: it only needs to modify a few words. 3) Human-like: it resembles the way that humans perform caption editing, and tries to keep original sentence structures. To solve this task, we propose the first ECE model: `TIger`. It is a non-autoregressive transformer-based model, consisting of three modules: $\text{Tagger}_{\text{del}}$, $\text{Tagger}_{\text{add}}$, and Inserter. Specifically, $\text{Tagger}_{\text{del}}$ decides whether each word should be preserved or not, $\text{Tagger}_{\text{add}}$ decides where to add new words, and Inserter predicts the specific word for adding. To further facilitate ECE research, we propose two ECE benchmarks by re-organizing two existing datasets, dubbed COCO-EE and Flickr30K-EE, respectively. Extensive ablations on both two benchmarks have demonstrated the effectiveness of `TIger`.

Keywords: Image Captioning, Caption Editing, Explicit Editing

1 Introduction

Image caption generation (*a.k.a.*, image captioning), is the task of generating natural language captions for given images. Due to its multimodal nature and numerous downstream applications (*e.g.*, human-machine interaction [7], content-based image retrieval [29], and assisting visually-impaired people [24]), caption generation has raised unprecedented attention from both CV and NLP communities. Thanks to the development of encoder-decoder frameworks (*e.g.*, CNN+ RNN [38] or Transformer [36]), current state-of-the-art image caption generation models can generate "reasonable" captions from scratch and achieve satisfactory performance. However, numerous studies [34,35] have revealed that these SOTA

[*]Zhen Wang and Long Chen are co-first authors with equal contributions.
[†]Corresponding author. Codes: https://github.com/baaaad/ECE.

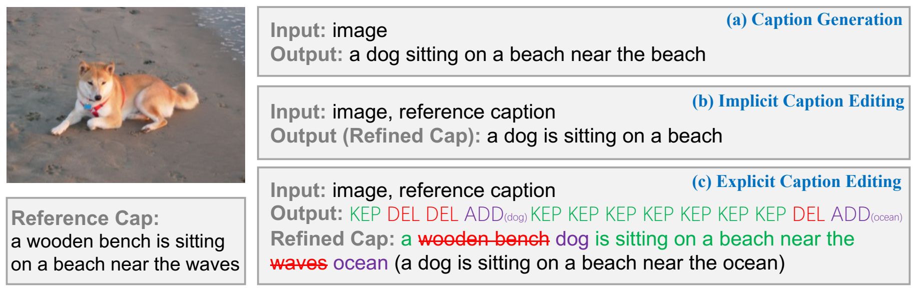

Fig. 1. Comparisons between our proposed ECE task (c) and existing caption generation (a) and implicit caption editing (b). The outputs are from the SOTA models [3,35].

models always suffer from severe bias issues and overlook some content details (*e.g.*, gender bias [14], object hallucination [33]). As shown in Fig. 1(a), given the input image, a SOTA captioning model [3] generates "`a dog sitting on a beach near the beach`". Thus, SOTA models can indeed generate a coherent sentence structure for the image (*i.e.*, "`a __ on a __ near the __`"), but fail to properly predict the correct details and even repeat the main object "`beach`".

To mitigate these problems and make the generated captions focus more on visually-grounded content details (beyond sentence structures), some pioneering works [34,35] have proposed a new task: Image Caption Editing (ICE). Different from captioning models which generate captions from scratch, ICE directly edits another reference caption and pays more attention to the misaligned details. For example in Fig. 1(b), ICE model takes an extra reference caption "`a wooden bench is sitting on a beach near the waves`" as input, and aims to generate a refined caption. Unfortunately, all existing ICE works are ***implicit*** editing models. By "implicit", we mean that they directly produce final refined captions, without explicit connections (editing process) to the reference captions.

Although ICE models can significantly improve the captions qualities, it is worth noting that there are still several drawbacks for this implicit manner: 1) **Unexplainable**: they fail to explain whether these words are copied from the reference caption or regenerated, and whether they truly recognize and modify errors or simply generate words by language priors [23]. 2) **Inefficient**: All words are regenerated, which is more like rewriting or re-captioning instead of editing. 3) **Structure-breaking**: They are easy to break the sentence structures of reference captions without focusing on details. For example in Fig. 1(b), the model roughly deletes part of the structure (*e.g.*, "`near the __`").

In this paper, we introduce a new image caption editing task: **Explicit Caption Editing** (ECE). By "explicit", we mean that ECE models explicitly generate a sequence of *edit operations*, and these edit operations translate the reference captions into the refined captions. Typically, the edit operations consist of `ADD`, `DELETE`, and `KEEP`[1]. As shown in Fig. 1(c), for each input word in the reference

[1] These are the most common edit operations in numerous text explicit editing tasks, such as simplification [9,26], fusion [25]. Of course, different ECE models can design or propose other edit operations, *e.g.*, `REORDER`. More discussion are left in appendix.

caption, the ECE model predicts `KEEP` or `DELETE` to decide whether this word needs to be preserved or not, and predicts `ADD` to add extra specific words. The predicted edit operation sequence is mainly composed with `KEEP` to preserve the main sentence structure and few `DELETE`/`ADD` to fix misalignment errors. Compared to existing implicit caption editing works, ECE avoids all mentioned weaknesses: 1) ECE traces the whole editing path, which is used to translate reference captions (**Explainable**). 2) ECE only needs to modify a few words (**Explicit Editing Efficient**). 3) ECE resembles the way that humans perform editing, and tries to keep the original sentence structures (**Structure-preserving**).

To solve this new task, we propose the first ECE model, a non-autoregressive transformer-based ECE model: `TIger` (**T**a**gger** and **I**nserter). Specifically, `TIger` consists of three modules: $\text{Tagger}_{\text{del}}$, $\text{Tagger}_{\text{add}}$, and Inserter. All three modules are built on top of the multimodal BERT architecture [22]. Given an input image and a reference caption, $\text{Tagger}_{\text{del}}$ decides whether each word should be preserved or not by predicting `KEEP` and `DELETE`. Then, $\text{Tagger}_{\text{add}}$ decides whether a new word should be added after each input word by predicting `KEEP` and `ADD`. A special token [`Mask`] is placed for each position with the `ADD` prediction. Subsequently, Inserter predicts the specific word for each [`Mask`] token. Since $\text{Tagger}_{\text{add}}$ only adds one new word after each input word once a time, we iteratively execute $\text{Tagger}_{\text{add}}$ and Inserter multiple rounds to guarantee enough words adding.

To further facilitate ECE research, we also propose two new ECE benchmarks by re-organizing MSCOCO [20] and e-SNLI-VE [44,17], dubbed **COCO-EE** and **Flickr30K-EE**, respectively. Particularly, we pair each reference caption with one ground-truth caption by several criteria and rules. Each ECE instance consists of an image, a reference caption, and a ground-truth caption. Compared to existing implicit editing works [35,34] which use machine-generated captions as reference captions, ours are all human-written sentences, *i.e.*, they are more natural and have no grammatical errors. Besides, we propose two supplementary metrics for ECE: Editing Steps (ES) and Gains Per Step (GPS), which consider not only the quality of captions, but also the efficiency of editing models.

In summary, we make three main contributions: 1) We propose a new visual-language task: ECE, *i.e.*, the caption editing model explicitly generates a set of edit operations on the reference captions. 2) For reliable benchmarking, we propose two new ECE datasets (COCO-EE and Flickr30K-EE), and new metrics for ECE evaluation. 3) We propose the first ECE model `TIger`. Extensive ablations have demonstrated the effectiveness of `TIger`. Moreover, `TIger` can serve as an off-the-shelf model to improve the quality of machine-generated captions.

2 Related Work

Image Caption Generation. With the release of advanced encoder-decoder frameworks, NN-based [27,16,38] methods have risen to prominence. They typically use an encoder to extract image features and a decoder to generate all words. Recent advances in captioning works focus on stronger architectures and better training procedures. To encoder visual context, numerous attention mech-

Fig. 2. Two examples from proposed ECE benchmarks: COCO-EE and Flickr30K-EE.

anisms are proposed to boost the performance [41,6,43,3,15,28,21,39], and they tend to focus on specific local features in the image when predicting each word in the caption. On the other side, current caption generation performance is dominated by reinforcement learning (RL) based methods [31,32,42], which directly optimize the sequence-level caption quality. Besides, to accelerate the decoding process, non-autoregressive methods [10,11,13] are proposed, which simultaneously generate words by discarding the sequential dependencies within sentence.

Image Caption Editing. ICE, *i.e.*, editing the existing reference caption paired with an image for refinement instead of re-generating from scratch, was first proposed by Sammani *et.al.* [34]. Specifically, they use a pre-trained deep averaging network to encode the reference caption, and design a gate mechanism to help the decoder to generate refined captions. Later, Sammani *et.al.* [35] proposed a new method for caption editing, which designs a selective copy memory attention to better encode the reference caption. As discussed above, they are all *implicit* caption editing models. In this paper, we propose the new explicit editing task, which can avoid the weaknesses in existing implicit works.

Explicit Text Editing. Explicit text editing, explicitly labeling the input reference caption with a sequence of edit operations, has been widely applied in different text editing tasks, such as text simplification [1,9], sentence fusion [26,25], grammatical error correction [4] and text generation [12]. Besides the basic edit operations like insertion and deletion, they tend to design different edit operations and edit mechanisms for their specific downstream tasks. In this paper, we extend three explicit text editing models (EditNTS [9], LaserTagger [26], and Felix [25]) into ECE, and compare them with our `TIger`. Specifically, EditNTS predicts edit operations by an LSTM sequentially. LaserTagger and Felix are all Transformer-based models, where LaserTagger predicts the edit operations restricted to a fixed phrase vocabulary and Felix uses extra reordering operations.

3 ECE and Benchmarks

3.1 Task Definition: Explicit Caption Editing (ECE)

In this section, we first formally define the ECE task. Given an image and a reference caption (Ref-Cap), ECE models aim to explicitly predict a sequence of edit operations (*e.g.*, `KEEP`/`DELETE`/`ADD`) on the Ref-Cap, which can translate the Ref-Cap close to the ground-truth caption (GT-Cap). Typically, Ref-Cap is slightly misaligned with the image. This task hopes the captioning models not only focus more on the visually-grounded content details, but also perform more explainable, explicit editing efficient[2], and human-like editing. As the example shown in Fig. 2(b), given Ref-Cap "`Motorcyclists are stopped at a stop sign`", the ECE models aim to explicitly predict a edit operation sequence: "$\texttt{KEEP}_{\texttt{Motorcyclists}}$ $\texttt{KEEP}_{\texttt{are}}$ $\texttt{DELETE}_{\texttt{stopped}}$ $\texttt{DELETE}_{\texttt{at}}$ $\texttt{ADD}_{\texttt{in}}$ $\texttt{KEEP}_{\texttt{a}}$ $\texttt{DELETE}_{\texttt{stop}}$ $\texttt{DELETE}_{\texttt{sigh}}$ $\texttt{ADD}_{\texttt{close}}$ $\texttt{ADD}_{\texttt{race}}$ $\texttt{ADD}_{\texttt{around}}$ $\texttt{ADD}_{\texttt{a}}$ $\texttt{ADD}_{\texttt{corner}}$"[3].

3.2 Explicit Caption Editing Benchmarks

Criteria. Based on the task definition of ECE and essential requirements of each ECE instance, each reference caption (Ref-Cap) and its corresponding ground-truth caption (GT-Cap) should be selected reasonably for each image. We argue that there are several criteria in developing high-quality ECE datasets:

c1. **Human Annotated Captions.** Both Ref-Cap and GT-Cap should be written by humans to avoid grammatical errors.

c2. **Image-Caption Similarity.** The scene described by the Ref-Cap should be similar to the scene in the image.

c3. **Caption Similarity.** Paired Ref-Cap and GT-Cap should have a certain degree of overlap and similar caption structure to avoid completely regenerating the whole sentence or roughly breaking the structure of Ref-Cap.

c4. **Caption Differences.** To ensure necessary editing operations, the differences between the Ref-Cap and GT-Cap shouldn't be just one (or few) words, which can be easily corrected by only language bias.

Existing ICE work [35,34] simply uses machine-generated captions as their Ref-Caps, which may mislead editing models to focus more on grammatical errors instead of content details. Meanwhile, each image has five GT-Caps, and these GT-Caps may have potential differences (caption structures or described events [5]). These training samples may confuse the editing model to break the sentence structures of Ref-Caps. To this end, we constructed two high-quality ECE benchmarks based on the aforementioned criteria. Details are as follows:

COCO-EE. We built COCO-EE based on dataset MSCOCO [20], which contains 123,287 images, and 5 ground-truth captions for each image. To ensure *c1*, we selected all Ref-Caps and GT-Caps in COCO-EE from MSCOCO captions. Since each image is labeled with 5 captions, we regard all 5 ground-truth captions as the GT-Cap candidates and filter Ref-Cap candidates from the rest captions based on image-caption similarity score to ensure *c2*. We then calculated several caption similarity scores to further filter the Ref-Cap candidates to ensure *c3* and *c4*. Finally, for each filtered Ref-Caps candidate, we selected the caption with the shortest edit distance[4] from corresponding GT-Caps candi-

[2]We emphasize efficient from the perspective of "explicit editing efficiency", as realizing more performance gains with less meaningful editing steps, which differs from other efficiency metrics (inference time and FLOPs). More details are left in appendix.

[3]Based on different basic edit operations used in each ECE model, the GT edit operation sequence can be different. This example uses `KEEP`/`DELETE`/`ADD` as operations.

[4]The shortest edit distance is the minimum number of edit operations (except the `KEEP` operation) to translate one sentence to the target sentence.

Dataset	COCO-EE			Flickr30K-EE		
	Train	Val	Test	Train	Val	Test
#Editing instances	97,567	5,628	5,366	108,238	4,898	4,910
#Images	52,587	3,055	2,948	29,783	1,000	1,000
Mean Reference Caption Length	10.3	10.2	10.1	7.3	7.4	7.4
Mean Ground-Truth Caption Length	9.7	9.8	9.8	6.2	6.3	6.3
Mean Edit Distance	10.9	11.0	10.9	8.8	8.8	8.9
Vocabulary	11,802	3,127	3,066	19,124	4,178	4,183

Table 1. Statistical summary of the COCO-EE and Flickr30K-EE benchmarks.

dates to form a Ref-GT caption pair. Following the above steps[5], we constructed COCO-EE, and divided it into training, val, and test sets following the "Karpathy" split [16]. The statistical summary about COCO-EE is shown in Table 1.

Flickr30K-EE. We built Flickr30K-EE based on dataset e-SNLI-VE [17]. e-SNLI-VE is a visual entailment dataset using the same image set as the image captioning dataset Flicrk30K [44]. For each image in e-SNLI-VE, there are three sentences (hypothesis), which have different relations with the image (premise): entailment, neutral, and contradiction. For each image and its textual hypotheses in e-SNLI-VE, we selected the contradiction and entailment hypothesis as a Ref-GT caption pair if they have the same text premise, which ensures *c2*. Since the paired contradiction and entailment hypothesis are human-annotated (*c1*) and have the same text premises, they tend to have a certain textual similarity (*c3*) while maintaining visual differences (*c4*) at the same time. Together with the image, each ECE instance contains one image from Flickr30K and one human-annotated Ref-Cap and GT-Cap pair. Finally, we obtained the Flickr30K-EE[5]. Similarly, we divided it into training, val, and test sets based on e-SNLI-VE splits. The statistical summary about Flickr30K-EE is shown in Table 1.

4 Proposed Approach

Overview. In this section, we introduce the proposed `TIger` for the ECE task. Specifically, the design of the `TIger` is inspired from the manner in which humans conduct caption editing, *i.e.*, *our humans would like to delete all the irrelevant or wrong words in the reference caption first, and then gradually add the missing words or details till enough.* Based on this motivation, we design three modules in `TIger`: **Tagger$_{\mathbf{del}}$**, **Tagger$_{\mathbf{add}}$**, and **Inserter**. The overview of the pipeline of the `TIger` is illustrated in Fig. 3, and the function of each module is as follows:

1) Tagger$_{\mathbf{del}}$: The Tagger$_{\text{del}}$ aims to predict whether to keep or delete each input word. For example in Fig. 3 (1-st Round), the words "`field`", "`with`" and "`ball`" in the reference caption ("`a person is on a` <u>`filed`</u> <u>`with`</u> `a` <u>`ball`</u>") are not related to the image content, and we hope the Tagger$_{\text{del}}$ module can predict "`DELETE`" for these words, and "`KEEP`" for the rest of the words.

2) Tagger$_{\mathbf{add}}$: The Tagger$_{\text{add}}$ aims to decide which words need to be added with a new word after them, and a special token [`Mask`] will be placed after

[5]More details about the dataset construction steps are left in the appendix.

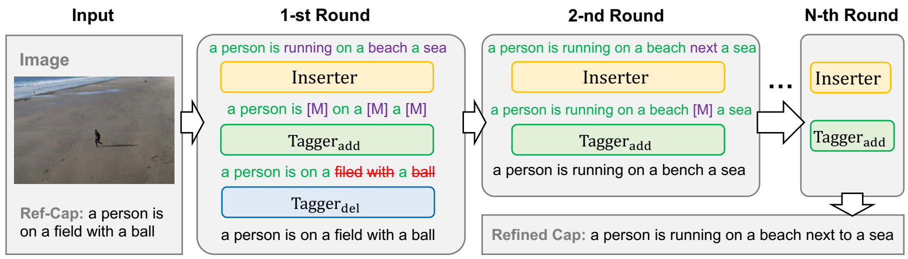

Fig. 3. Overview of the whole `TIger` pipeline. $\text{Tagger}_{\text{del}}$ is only used in the first round, $\text{Tagger}_{\text{add}}$ and Inserter are used in all rounds. In the first editing round, `TIger` aims to fix the main errors. Then, in the following rounds, `TIger` tries to add more details to generate more coherent and reasonable captions. [M] denotes the special [`MASK`] token.

these words. For example, given the input caption ("`a person is on a a`"), $\text{Tagger}_{\text{add}}$ thinks a new word should be added after "`is`", "`a`", and "`a`", *i.e.*, the output of $\text{Tagger}_{\text{add}}$ is "`a person is [Mask] on a [Mask] a [Mask]`".

3) Inserter: Given the output of $\text{Tagger}_{\text{add}}$, the Inserter aims to predict a specific word for each [`Mask`] token, *i.e.*, "`running`", "`beach`", and "`sea`".

Since the $\text{Tagger}_{\text{add}}$ and Inserter can only add one new word at each position for each round, we can easily run $\text{Tagger}_{\text{add}}$ and Inserter iteratively for multiple rounds to guarantee enough words adding. Instead, for the $\text{Tagger}_{\text{del}}$, we hope it directly detects all the wrong or unsuitable words in the first round.

4.1 Multimodal Feature Extraction

As shown in Fig. 4, all three modules $\text{Tagger}_{\text{del}}$, $\text{Tagger}_{\text{add}}$, and Inserter are all built on top of the multi-modal BERT [22,18], which applies a series of transformer blocks and co-attention layers to learn better multi-modal features of the images and texts. The input for each module is a sequence of multimodal tokens.

Visual Token Representations. For the given image, we first generate a set of image region features by extracting proposals and their corresponding visual features from a pre-trained object detector. We also encode the spatial location features of each proposal into a 5-d vector (normalized top-left and bottom-right coordinates, and fraction of the region area covered). A visual token feature is the sum of a region proposal feature and its spatial location feature. In addition, a special [`IMG`] token is placed at the beginning of the visual token sequence to represent the entire image. The token feature of [`IMG`] is the mean-pooled visual feature with a spatial encoding corresponding to the entire image.

Textual Token Representations. For the given reference caption, we first convert it into a sequence of tokens by tokenization [8]. Then, we put special [`CLS`] and [`SEP`] tokens at the start and end of textual token sequence, respectively. Meanwhile, for Inserter, another token [`MASK`] is used to indicate the position for new words adding. Same as [22], a textual token representation is the sum of token-specific learned embedding [40], position encoding, and segment encoding.

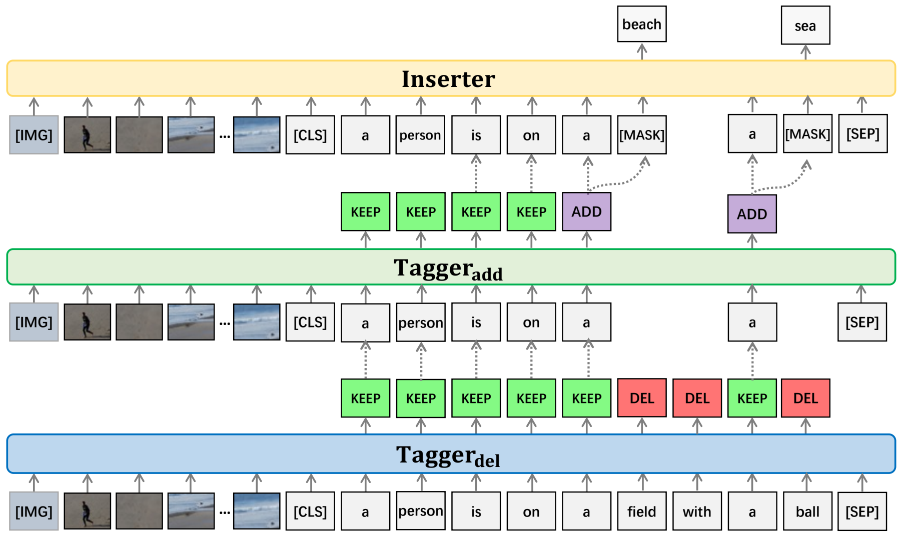

Fig. 4. Illustration of the input visual-language token sequences for each module. We take the first editing rounds in as the example.

Multimodal Input Token Sequence. Given the image and reference caption, we first encoder them into a sequence of visual tokens $\{v_1, \ldots, v_K\}$ and textual tokens $\{w_1, \ldots, w_L\}$, respectively. K and L is the number of visual and textual tokens, respectively. Then, the input token sequence for the three modules is $\{[\texttt{IMG}], v_1, \ldots, v_K, [\texttt{CLS}], w_1, \ldots, w_L, [\texttt{SEP}]\}$. The output representations for the visual and textual tokens are $\{h_{v_1}, \ldots, h_{v_K}\}$ and $\{h_{w_1}, \ldots, h_{w_L}\}$, respectively.

4.2 Model Description

Tagger$_{\text{del}}$ & Tagger$_{\text{add}}$ Modules. As shown in Fig. 4, given the visual-textual token sequence, Tagger$_{\text{del}}$ and Tagger$_{\text{add}}$ tag each textual token with a specific edit operation z. For each textual token, both Tagger$_{\text{del}}$ and Tagger$_{\text{add}}$ conduct a binary classification, *i.e.*, $z \in \{\texttt{KEEP}, \texttt{DELETE}\}$ for Tagger$_{\text{del}}$ and $z \in \{\texttt{KEEP}, \texttt{ADD}\}$ for Tagger$_{\text{add}}$. We pass the final representation of each textual token $\{h_{w_1}, h_{w_2}, \ldots, h_{w_L}\}$ into a two-layer MLP to make the binary prediction, *i.e.*, $z_{w_i} = \arg\max f(h_{w_i})$. Thus, the entire output of Tagger$_{\text{del}}$ and Tagger$_{\text{add}}$ is a sequence of edit operations corresponding to the sequence of input tokens, represented as $\{z_{w_1}, z_{w_2}, \ldots, z_{w_L}\}$. The output textual token sequence can be translated from the input textual token sequence and predicted edit operations.
Inserter Module. As shown in Fig. 4, the input tokens fed into the Inserter is a sequence of tokens including the word tokens and the [MASK] tokens, which is constructed from the Tagger$_{\text{add}}$ module. Given the image and the input tokens, the Inserter finishes the insertion by predicting the specific word from the vocabulary for each [MASK] token based on the observed tokens and visual information. Specifically, we pass the final representation of each [MASK] token $h_{w_{mask}}$ into a linear layer, mapping it to a distribution over the vocabulary. Lastly, all [MASK]

tokens can be replaced with the predicted word, and the output textual token sequence can be formed with the rest word tokens for following the procedures.
Multi-Rounds Editing. As the specific editing process shown in Fig. 4, `TIger` resembles the way that humans might perform caption editing, *i.e.*, considering what to keep, where to add, and what to add. By tracing these edit operations, the whole editing process is explainable and efficient. Meanwhile, since $\text{Tagger}_{\text{add}}$ only adds one new word after each input word once a time, there might not be enough details if we only apply $\text{Tagger}_{\text{add}}$ once. Thanks to this modular design, we can seamlessly use $\text{Tagger}_{\text{add}}$ and Inserter iteratively for multi-rounds to guarantee enough details. Instead, if we make the $\text{Tagger}_{\text{add}}$ can add more than one word once a time, it also needs to predict the number of new words to add at the same time. Meanwhile, the Inserter needs to predict words for multiple [`MASK`] tokens that may be placed consecutively. This significantly increases the difficulty of training, and empirically this single-round solution gets worse results.

4.3 Training Objectives

The $\text{Tagger}_{\text{del}}$ and $\text{Tagger}_{\text{add}}$ are essentially solving a binary classification task, and the Inserter is essentially solving a masked language modeling task. Thus, we train all three modules with the cross-entropy (XE) loss. Due to the modular nature, we train the three modules separately. In our experiments, we also emphasize the importance of predicting relative more `KEEP` operation. Specifically, for $\text{Tagger}_{\text{del}}$, it can preserve more words in the caption for the whole following editing process. For $\text{Tagger}_{\text{add}}$, it can offer more context words with relative fewer [`MASK`] tokens for Inserter, which makes the edit operation prediction much easier. Thus, We use different XE loss weights for the `KEEP` tokens and other tokens (`DELETE` or `ADD`). The loss weight ratio λ denotes the XE loss weights of edit token `KEEP/DELETE` for training $\text{Tagger}_{\text{del}}$ and `KEEP/ADD` for training $\text{Tagger}_{\text{add}}$, respectively. More detailed influence of λ is discussed in Sec. 5.3.

5 Experiments

5.1 Experimental Setup

Evaluation Datasets and Metrics. We evaluated our `TIger` on both COCO-EE and Flickr30K-EE datasets (cf. Sec. 3.2). For the caption quality evaluation, we followed existing caption generation works, and used four prevalent evaluation metrics: BLEU-N (B-N) (1-to 4-grams) [30], ROUGE-L (R) [19], CIDEr-D (C) [37] and SPICE (S) [2]. Particularly, we evaluated generated captions against its single ground-truth caption. Meanwhile, to evaluate the explicit editing efficiency of editing, we propose two supplementary metrics: Editing Steps (**ES**), and Gains Per Step (**GPS**). ES is the total number of meaningful editing steps, and GPS is the average performance gains per meaningful editing step, *i.e.*, we hope ECE models realize the most performance gains with the least number of meaningful editing steps. In this paper, since all baselines apply the same set of

	Model	Quality Evaluation							Efficiency Evaluation			
		B-1	B-2	B-3	B-4	R	C	S	ES	GPS(C)	D	A
	Ref-Caps	50.0	37.1	27.7	19.5	48.2	129.9	18.9	—	—	—	—
	UpDn [3]	49.9	35.3	25.5	18.8	48.3	159.2	31.2	—	—	—	—
ICE	UpDn-E [3]	54.0	40.1	30.2	22.9	52.8	182.0	33.2	19.22	2.71	10.14	9.08
	MN [34]	50.2	35.8	26.0	19.4	48.9	163.9	31.6	19.08	1.78	10.14	8.94
	ETN [35]	53.8	40.5	23.8	23.8	53.3	190.5	32.1	18.96	3.20	10.14	8.82
ECE	V-EditNTS [9]	49.2	36.5	27.4	20.5	49.8	149.0	26.2	5.90	3.24	3.76	2.14
	V-Felix [25]	36.9	28.2	21.6	16.2	49.7	139.5	25.3	5.51	1.74	4.57	0.94
	V-LaserTagger [26]	42.0	30.5	22.4	16.0	46.8	127.1	24.1	4.11	-0.68	3.54	0.57
	TIger (Ours)	**54.8**	**42.0**	**32.4**	**24.7**	**54.3**	**194.8**	**33.3**	7.74	**8.38**	4.59	3.15

Table 2. Performance of our model and other state-of-art models on COCO-EE. "Ref-Caps" denotes the quality of given reference captions. "D" and "A" denotes the number of editing step of `DELETE` and `ADD` operations, respectively.

edit operations (*i.e.*, `KEEP`, `DELETE`, and `ADD`), we regard the sum of `DELETE` and `ADD` operations as ES. Meanwhile, since CIDEr-D is regarded as the most important metric for caption evaluation as to its high agreements with humans, we use the improvements of CIDEr-D score to calculate GPS, denoted as GPS(C).
Baselines. We compared our `TIger` against state-of-the-art image caption editing models. Specifically, we compared three strong implicit caption editing models: **UpDn-E** [3], **MN** [34], and **ETN** [35]. They are all built on top of the widely-used UpDn architecture [3], and propose some extra modules to encode the reference caption. Meanwhile, for more complete comparisons, we further extended three text explicit editing models (EditNTS [9], LaserTagger [26], and Felix [25]) into ECE, denoted as **V-EditNTS**, **V-LaserTagger**, and **V-Felix**, respectively. For all these three models, their basic editing operations are `KEEP`, `DELETE` and `ADD`. Specifically, V-EditNTS predicts the edit operation sequence iteratively by an LSTM. V-LaserTagger and V-Felix are one-round Transformer-based editing models, which directly predict multiple `ADD` operations simultaneously. More details about these baselines are left in the appendix.
Implementation Details. The implementation details are left in appendix.

5.2 Comparisons with State-of-the-Arts

Settings. We evaluated `TIger` on COCO-EE and Flickr30K-EE by comparing with state-of-the-art methods. Since our target is to propose the ECE task and the first ECE model, we first compared `TIger` with simple ECE baselines which were extended by text explicit editing models (V-EditNTS, V-Felix, and V-LaserTagger). For completeness, we also reported the results of all existing ICE models (UpDn-E, ETN, and MN). Since all implicit models are built on top of the widely-used UpDn architecture, we only reported the results of the UpDn captioning model rather than all other SOTA captioning models (e.g., VLP[45]) as they actually don't belong to the caption editing task. Since V-Felix and V-LaserTagger are also Transformer-based architectures, we used the same ViLBERT pretrained weights as `TIger`. For the other baselines, we converted all the words in each dataset to lower cases and built their respective

	Model	Quality evaluation							Efficiency evaluation			
		B-1	B-2	B-3	B-4	R	C	S	ES	GPS(C)	D	A
	Ref-Cap	34.7	24.0	16.8	10.9	36.9	91.3	23.4	—	—	—	—
	UpDn [3]	25.6	16.1	10.4	6.3	30.1	71.0	21.4	—	—	—	—
ICE	UpDn-E [3]	33.9	24.7	18.3	12.5	41.1	129.1	29.8	12.00	3.15	7.41	4.59
ICE	MN [34]	30.0	20.0	13.6	8.6	34.9	91.1	25.2	12.09	-0.02	7.41	4.69
ICE	ETN [35]	34.8	25.9	19.6	13.7	41.8	143.3	31.3	12.06	4.31	7.41	4.65
ECE	V-EditNTS [9]	38.0	27.6	20.1	13.8	40.2	129.1	28.7	5.48	6.90	3.59	1.89
ECE	V-Felix [25]	21.1	16.7	13.5	10.1	38.0	127.4	27.8	5.54	6.51	4.92	0.62
ECE	V-LaserTagger [26]	30.8	20.8	15.0	10.5	34.9	104.0	27.3	3.37	3.77	3.35	0.02
ECE	**TIger (Ours)**	**38.3**	**28.1**	**21.1**	**14.9**	**42.7**	**148.3**	**32.0**	6.65	**8.58**	4.63	2.02

Table 3. Performance of our model and other state-of-art models on Flickr30K-EE. "Ref-Caps" denotes the quality of given reference captions. "D" and "A" denotes the number of editing step of `DELETE` and `ADD` operations, respectively.

vocabulary. All baselines were trained with XE loss. Since implicit models do not explicitly predict edit operations, we suppose they delete all the words in the reference caption first and add new words from scratch to output caption, *i.e.*, ES is calculated as the sum of words in reference and output caption. Meanwhile, we mainly focused on the efficiency evaluation of ECE models, so we have used gray font for efficiency evaluation of ICE methods. Results on COCO-EE and Flickr30K-EE are reported in Table 2 and Table 3, respectively.

Results on COCO-EE. From Table 2, we can observe: 1) For the quality evaluation, our model achieves the largest performance gains on all metrics (*e.g.*, 194.8 vs. 190.5 in ETN on CIDEr-D). 2) For efficiency evaluation, SOTA implicit models always outperform their explicit counterparts, but they require more editing steps. Instead, our model achieves the best GPS(C) score by predicting more `ADD` operations, instead of simply deleting or keeping the words in the reference captions. It also shows our ability to detect and fix detailed errors.

Results on Flickr30K-EE. From Table 3, we can observe: 1) For the quality evaluation, similar with COCO-EE, our model achieves the largest performance gains on all metrics (*e.g.*, 148.3 vs. 143.3 in ETN on CIDEr-D). 2) For efficiency evaluation, our model achieves the best GPS(C) score (*e.g.*, 8.58 vs. 6.90 in V-EditNTS). Compared to the weaknesses of implicit models (need more editing steps) and explicit models (marginal performance gains), our model achieves a decent balance between performance gains and editing steps, *i.e.*, we improved the quality of reference captions with quite a few meaningful editing steps.

5.3 Ablation Studies

In this section, we run a set of ablation studies to analyze the influence of different hyperparameter settings, and the influence of pre-trained ViLBERT weights.

Influence of Weighted XE Loss. As mentioned in Sec. 4.3, we used weighted XE loss for training. To explore the influence of different loss weights, we first run ablations by setting different loss weight ratios $\lambda \in \{1.0, 1.2, 1.5, 2.0\}$ on both $\text{Tagger}_{\text{del}}$ and $\text{Tagger}_{\text{add}}$. Results are reported in Table 4. Then, we explored the

	λ	B-1	B-4	R	C	S
COCO	1.0	54.1	24.0	53.9	190.0	33.4
	1.2	54.4	24.1	54.0	190.9	33.4
	1.5	**54.8**	**24.7**	**54.3**	**194.8**	**33.3**
	2.0	54.6	24.6	54.1	193.9	33.1
Flickr30K	1.0	34.2	13.4	41.2	137.0	30.9
	1.2	34.3	14.1	41.7	144.0	31.4
	1.5	**38.3**	**14.9**	**42.7**	**148.3**	**32.0**
	2.0	37.2	14.9	42.7	148.0	31.5

Table 4. Performance on COCO-EE and Flickr30K-EE with different XE loss weights λ.

	T_{del}	T_{add}	B-1	B-4	R	C	S
COCO			54.1	24.0	53.9	190.0	33.4
	✓		55.0	24.7	54.3	193.7	33.1
		✓	54.1	24.1	54.0	191.2	33.7
	✓	✓	**54.8**	**24.7**	**54.3**	**194.8**	**33.3**
Flickr30K			34.2	13.4	41.2	137.0	30.9
	✓		34.9	14.3	42.0	144.9	34.6
		✓	34.3	13.7	41.4	140.9	31.2
	✓	✓	**38.3**	**14.9**	**42.7**	**148.3**	**32.0**

Table 5. Influence of different modules with weighted XE loss ($\lambda = 1.5$). "T_{del}" and "T_{add}" denote Tagger_{del} and Tagger_{add}, respectively.

influence of weighted XE loss to a single Tagger module, *i.e.*, we run ablations by setting one of the Tagger with $\lambda > 1.0$, and the other with $\lambda = 1.0$. The results are reported in Table 5.3. Note that all Inserters were trained with $\lambda = 1.0$.

Results. From Table. 4, we have several observations: 1) For both the COCO-EE and Flickr30K-EE, `TIger` with weighted XE loss training always gets better performance than the baseline ($\lambda = 1.0$). 2) The model trained with $\lambda = 1.5$ gets the best performance, *i.e.*, it boosts the CIDEr-D score from 190.0 to 194.8 for COCO-EE and from 137.0 to 148.3 for Flickr30K-EE. This demonstrates the effectiveness of paying more attention to predicting the `KEEP` operation. We then used $\lambda = 1.5$ to train `TIger` in all experiments. From Table 5.3, we can observe that: 1) `TIger` with only one of the Tagger modules trained with $\lambda = 1.5$ alone can still achieve better performance than baseline. 2) The weighted XE loss has more impact on Tagger_{del} than Tagger_{add}. The possible reason is that `TIger` only applies Tagger_{del} once, which determines the basic caption for further adding.

Different Editing Rounds. Since `TIger` iteratively use Tagger_{add} and Inserter multiple rounds to add words, we run ablations to analyse the effect of different edit rounds. The maximum number of editing rounds was set to 5.

Results. From Table 6, we can observe that: 1) For COCO-EE, the performance of `TIger` keeps improving in the first 3 editing rounds. Then, the quality evaluation metrics reach the best scores and keep unchanged or even slightly drop with more editing rounds. For example, BLEU-1 keeps increasing with more editing rounds, CIDEr-D reaches the best score 194.8 in the 4-th round and drops to 194.6 in the 5-th round. Since most metrics reach their best scores in the 4-th round, considering the trade-off between model performance and editing efficiency, we used 4 editing rounds for the COCO-EE. 2) For Flickr30K-EE, the performance of `TIger` keeps improving in the first 3 editing rounds. Most quality evaluation metrics reach the best score in the 3-rd round, and then keep unchanged (14.9 for BLEU-4 and 148.3 for CIDEr-D) or drop slightly (SPICE) with more editing rounds. Thus, we used 3 editing rounds for the Flickr30K-EE.

Influence of the pre-trained Weights. Since we took advantage of the pre-trained weights to train `TIger`, we further ran ablations to examine the influence of the pre-trained ViLBERT weights. The results are reported in Table 7.

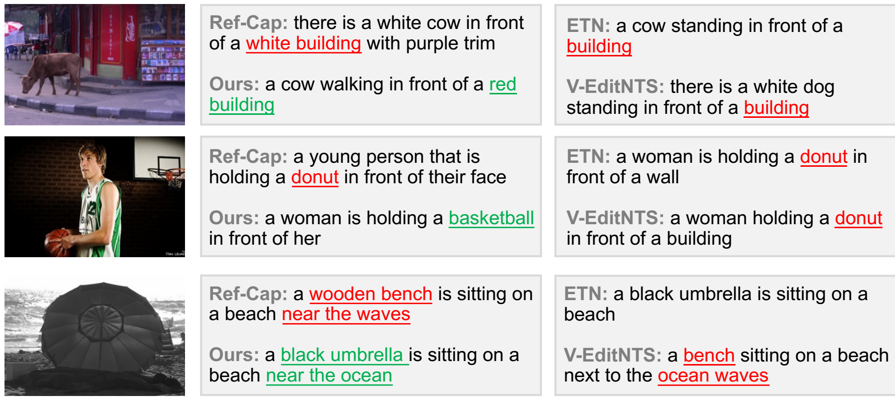

Fig. 5. Visualization results of our model compared to baselines in COCO-EE.

# Rounds	COCO-EE						Flickr30K-EE					
	B-1	B-4	R	C	S	GPS(C)	B-1	B-4	R	C	S	GPS(C)
1	49.3	22.2	54.2	180.2	31.6	8.01	33.5	13.9	42.2	142.8	31.0	8.79
2	52.8	23.8	54.3	189.8	32.7	8.41	36.8	14.8	42.5	148.1	31.9	**8.88**
3	54.2	24.5	**54.4**	193.9	33.1	**8.49**	38.3	**14.9**	42.7	**148.3**	**32.0**	8.58
4	54.8	**24.7**	54.3	**194.8**	**33.3**	8.38	38.4	14.9	**42.8**	148.3	31.9	8.45
5	**55.0**	24.7	54.3	194.6	33.3	8.26	**38.6**	14.9	42.8	148.3	31.9	8.42

Table 6. Performance of `TIger` with different editing rounds.

Results. From Table 7, we can observe that for both datasets, as the first ECE model, both TIger models with and w/o pre-trained weights all outperform other ECE baselines with same pre-trained weights in both quality and efficiency evaluation. Meanwhile, `TIger` trained with pretraind weight achieves better performance than the model trained from scratch. For example, in CIDEr-D, the pre-trained weights improve score from 178.1 to 194.8 for COCO-EE.

5.4 Transfering to Machine-Generated Captions

As mentioned before, machine-generated captions may be semantic coherent but suffer from severe bias issues, such as overlooking some content details and producing incorrect or repetitive content. To evaluate the generalization ability on machine-generated captions, we directly use the trained `TIger` to edit machine-generated captions without extra fine-tuning. To guarantee fairness and avoid data leakage, we first trained `TIger` and the ECE baselines on the same COCO-EE (and Flickr30K-EE) training set. Then, we apply the trained `TIger` to directly edit the captions generated from these ECE baselines (*i.e.*, as reference captions) on the test set. The results are reported in Table. 8.
Results. As shown in Tabel. 8, we can observe that: 1) For COCO-EE, our proposed `TIger` can significantly improve the quality of all the captions generated by ECE baselines (*e.g.*, CIDEr-D score from 149.0 to 172.7 for V-EditNTS). 2) For Flickr30K-EE, the average improvements are still remarkable (*e.g.*, 135.8

Models	COCO-EE					Flickr30K-EE				
	B-1	B-4	R	C	S	B-1	B-4	R	C	S
`TIger` w/o pretrain	53.6	23.3	52.8	178.1	31.1	35.0	14.0	41.7	140.8	30.9
`TIger`	**54.8**	**24.7**	**54.3**	**194.8**	**33.3**	**38.3**	**14.9**	**42.7**	**148.3**	**32.0**

Table 7. The Influence of the pretraind ViLBERT weight.

Models	COCO-EE					Flickr30K-EE				
	B-1	B-4	R	C	S	B-1	B-4	R	C	S
V-EditNTS	49.2	20.5	49.8	149.0	26.2	38.0	13.8	40.2	129.1	28.7
V-EditNTS+Ours	**51.9**	**21.6**	**51.7**	**172.7**	**32.3**	36.2	13.6	**40.9**	**135.8**	**30.3**
V-Felix	36.9	16.2	49.7	139.5	25.3	21.1	10.1	38	127.4	27.8
V-Felix+Ours	**51.2**	**21.7**	**51.9**	**175.3**	**32.3**	**30.2**	**12.8**	**39.6**	**133.8**	**29.5**
V-LaserTagger	42.0	16.0	46.8	127.1	24.1	30.8	10.5	34.9	104.0	27.3
V-LaserTagger+Ours	**50.7**	**20.4**	**50.9**	**166.4**	**31.7**	**32.4**	**10.9**	**36.7**	**110.4**	27.2

Table 8. The result of extending `TIger` for ECE baselines

vs. 129.1 in V-EditNTS on CIDEr-D score). This also demonstrates the robustness of `TIger` when given different reference captions (*e.g.*, these ECE baselines generated captions may erroneously delete or preserve some words).

5.5 Qualitative Evaluation

Fig. 5 shows some results generated by `TIger` compared to baselines (ETN [35] and V-EditNTS [9]). The three examples demonstrate that our model is capable of recognizing and correcting incorrect details (*i.e.*, "`white`" to "`red`", "`donut`" to "`basketball`", and "`bench`" to "`umbrella`"), while the baselines simply delete the wrong word "`white`" or fail to correct the object errors "`donut`" and '`umbrella`". Meanwhile, the last example demonstrates that our model can add new details to the captions, like attributes (color) of main objects (*e.g.*, `black`), while baselines may overlook them. Furthermore, our model can fix these details without breaking the structure of the caption (*e.g.*, `near the ocean`).

6 Conclusions and Future Work

In this paper, we proposed a new visual-language task: Explicit Caption Editing (ECE). To facilitate the ECE research, we also proposed two benchmarks by reorganizing two existing datasets MSCOCO and e-SNLI-VE, dubbed as COCO-EE and Flickr30K-EE, respectively. Meanwhile, we proposed the first ECE model `TIger`. We validate the effectiveness of `TIger` through extensive comparative and ablative experiments. Moving forward, we are going to 1) design stronger ECE models by introducing some advanced edit operations; 2) try to bridge the gap between explicit and implicit editing, and propose a unified model for both tasks.

Acknowledgement. This work was supported by the National Key Research & Development Project of China (2021ZD0110700), the National Natural Science Foundation of China (U19B2043, 61976185), Zhejiang Natural Science Foundation (LR19F020002), Zhejiang Innovation Foundation(2019R52002), and the Fundamental Research Funds for the Central Universities (226-2022-00051).

References

1. Alva-Manchego, F., Bingel, J., Paetzold, G., Scarton, C., Specia, L.: Learning how to simplify from explicit labeling of complex-simplified text pairs. In: IJCNLP. pp. 295–305 (2017) 4
2. Anderson, P., Fernando, B., Johnson, M., Gould, S.: Spice: Semantic propositional image caption evaluation. In: ECCV. pp. 382–398. Springer (2016) 9
3. Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., Zhang, L.: Bottom-up and top-down attention for image captioning and visual question answering. In: CVPR. pp. 6077–6086 (2018) 2, 4, 10, 11
4. Awasthi, A., Sarawagi, S., Goyal, R., Ghosh, S., Piratla, V.: Parallel iterative edit models for local sequence transduction. arXiv (2019) 4
5. Chen, L., Jiang, Z., Xiao, J., Liu, W.: Human-like controllable image captioning with verb-specific semantic roles. In: CVPR. pp. 16846–16856 (2021) 5
6. Chen, L., Zhang, H., Xiao, J., Nie, L., Shao, J., Liu, W., Chua, T.S.: Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In: CVPR. pp. 5659–5667 (2017) 4
7. Das, A., Kottur, S., Gupta, K., Singh, A., Yadav, D., Moura, J.M., Parikh, D., Batra, D.: Visual dialog. In: CVPR. pp. 326–335 (2017) 1
8. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv (2018) 7
9. Dong, Y., Li, Z., Rezagholizadeh, M., Cheung, J.C.K.: Editnts: An neural programmer-interpreter model for sentence simplification through explicit editing. In: ACL (2019) 2, 4, 10, 11, 14
10. Fei, Z.c.: Fast image caption generation with position alignment. arXiv (2019) 4
11. Gao, J., Meng, X., Wang, S., Li, X., Wang, S., Ma, S., Gao, W.: Masked non-autoregressive image captioning. arXiv (2019) 4
12. Gu, J., Wang, C., Zhao, J.: Levenshtein transformer. NeurIPS **32** (2019) 4
13. Guo, L., Liu, J., Zhu, X., He, X., Jiang, J., Lu, H.: Non-autoregressive image captioning with counterfactuals-critical multi-agent learning. arXiv (2020) 4
14. Hendricks, L.A., Burns, K., Saenko, K., Darrell, T., Rohrbach, A.: Women also snowboard: Overcoming bias in captioning models. In: ECCV. pp. 771–787 (2018) 2
15. Huang, L., Wang, W., Chen, J., Wei, X.Y.: Attention on attention for image captioning. In: ICCV. pp. 4634–4643 (2019) 4
16. Karpathy, A., Fei-Fei, L.: Deep visual-semantic alignments for generating image descriptions. In: CVPR. pp. 3128–3137 (2015) 3, 6
17. Kayser, M., Camburu, O.M., Salewski, L., Emde, C., Do, V., Akata, Z., Lukasiewicz, T.: e-vil: A dataset and benchmark for natural language explanations in vision-language tasks. arXiv (2021) 3, 6
18. Li, G., Duan, N., Fang, Y., Gong, M., Jiang, D.: Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 11336–11344 (2020) 7
19. Lin, C.Y.: Rouge: A package for automatic evaluation of summaries. In: Text summarization branches out. pp. 74–81 (2004) 9
20. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV. pp. 740–755 (2014) 3, 5
21. Liu, A.A., Zhai, Y., Xu, N., Nie, W., Li, W., Zhang, Y.: Region-aware image captioning via interaction learning. IEEE Transactions on Circuits and Systems for Video Technology (2021) 4

22. Lu, J., Batra, D., Parikh, D., Lee, S.: Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. NeurIPS (2019) 3, 7
23. Lu, J., Xiong, C., Parikh, D., Socher, R.: Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In: CVPR. pp. 375–383 (2017) 2
24. MacLeod, H., Bennett, C.L., Morris, M.R., Cutrell, E.: Understanding blind people's experiences with computer-generated captions of social media images. In: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems. pp. 5988–5999 (2017) 1
25. Mallinson, J., Severyn, A., Malmi, E., Garrido, G.: Felix: Flexible text editing through tagging and insertion. arXiv (2020) 2, 4, 10, 11
26. Malmi, E., Krause, S., Rothe, S., Mirylenka, D., Severyn, A.: Encode, tag, realize: High-precision text editing. arXiv (2019) 2, 4, 10, 11
27. Mao, J., Xu, W., Yang, Y., Wang, J., Huang, Z., Yuille, A.: Deep captioning with multimodal recurrent neural networks (m-rnn). arXiv (2014) 3
28. Mao, Y., Chen, L., Jiang, Z., Zhang, D., Zhang, Z., Shao, J., Xiao, J.: Rethinking the reference-based distinctive image captioning. In: ACMMM (2022) 4
29. Ordonez, V., Han, X., Kuznetsova, P., Kulkarni, G., Mitchell, M., Yamaguchi, K., Stratos, K., Goyal, A., Dodge, J., Mensch, A., et al.: Large scale retrieval and generation of image descriptions. IJCV pp. 46–59 (2016) 1
30. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: Bleu: a method for automatic evaluation of machine translation. In: ACL. pp. 311–318 (2002) 9
31. Ren, Z., Wang, X., Zhang, N., Lv, X., Li, L.J.: Deep reinforcement learning-based image captioning with embedding reward. In: CVPR. pp. 290–298 (2017) 4
32. Rennie, S.J., Marcheret, E., Mroueh, Y., Ross, J., Goel, V.: Self-critical sequence training for image captioning. In: CVPR. pp. 7008–7024 (2017) 4
33. Rohrbach, A., Hendricks, L.A., Burns, K., Darrell, T., Saenko, K.: Object hallucination in image captioning. In: EMNLP (2018) 2
34. Sammani, F., Elsayed, M.: Look and modify: Modification networks for image captioning. arXiv (2019) 1, 2, 3, 4, 5, 10, 11
35. Sammani, F., Melas-Kyriazi, L.: Show, edit and tell: A framework for editing image captions. In: CVPR. pp. 4808–4816 (2020) 1, 2, 3, 4, 5, 10, 11, 14
36. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. NeurIPS **30** (2017) 1
37. Vedantam, R., Lawrence Zitnick, C., Parikh, D.: Cider: Consensus-based image description evaluation. In: CVPR. pp. 4566–4575 (2015) 9
38. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell: A neural image caption generator. In: CVPR. pp. 3156–3164 (2015) 1, 3
39. Wang, Y., Xu, N., Liu, A.A., Li, W., Zhang, Y.: High-order interaction learning for image captioning. IEEE Transactions on Circuits and Systems for Video Technology (2021) 4
40. Wu, Y., Schuster, M., Chen, Z., Le, Q.V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al.: Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv (2016) 7
41. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., Bengio, Y.: Show, attend and tell: Neural image caption generation with visual attention. In: ICML. pp. 2048–2057 (2015) 4
42. Xu, N., Zhang, H., Liu, A.A., Nie, W., Su, Y., Nie, J., Zhang, Y.: Multi-level policy and reward-based deep reinforcement learning framework for image captioning. TMM **22**(5), 1372–1383 (2019) 4
43. Yang, Z., He, X., Gao, J., Deng, L., Smola, A.: Stacked attention networks for image question answering. In: CVPR. pp. 21–29 (2016) 4

44. Young, P., Lai, A., Hodosh, M., Hockenmaier, J.: From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. TACL **2**, 67–78 (2014) 3, 6
45. Zhou, L., Palangi, H., Zhang, L., Hu, H., Corso, J., Gao, J.: Unified vision-language pre-training for image captioning and vqa. In: AAAI. vol. 34, pp. 13041–13049 (2020) 10