Figure Captioning with Relation Maps for Reasoning

Charles Chen
Ohio University
lc971015@ohio.odu

Ruiyi Zhang
Duke University
rz68@duke.edu

Eunyee Koh Sungchul Kim Scott Cohen Ryan Rossi
Adobe Research
{eunyee,sukim,scohen,ryrossi}@adobe.com

Abstract

Figures, such as line plots, pie charts, bar charts, are widely used to convey important information in a concise format. In this work, we investigate the problem of figure caption generation where the goal is to automatically generate a natural language description for a given figure. While natural image captioning has been studied extensively, figure captioning has received relatively little attention and remains a challenging problem. A successful solution to this task has many potential applications, such as: 1) automatic parsing large amount of figures in PDF document; 2) improving user experience by allowing figure content to be accessible to those with visual impairment. To solve this problem, we introduce a dataset FigCAP and propose novel attention mechanism. In order to solve the exposure bias issue, we further train the captioning model with sequence-level policy based on reinforcement learning, which directly optimizes evaluation metrics. Extensive experiments show that the proposed method outperforms the baselines, thus demonstrating a significant potential for automatic generating captions for figures.

1. Introduction

Image understanding is an important area of investigation within computer vision and natural language processing. In recent years, excellent performance has been achieved in tasks for image understanding, such as, image captioning and Visual Question Answering (VQA). Figures, a specific type of images, convey useful trends, proportions and values in a concise format. Common examples include bar charts, pie charts, and line plots. They are widely used in documents, reports and talks to efficiently communicate ideas. Meanwhile, the rapid growth of PDF documents has resulted in a large number of figures for which automatic parsing is desired.

Figure captioning aims at generating natural language descriptions for figures, for example, high-level description (figure type, data labels, what this figure is about), and description with more details and insights (data trends, larger/smaller than relation).

With automatic caption generation, understanding large amount of figures rapidly is feasible. It facilitates automatic parsing for PDF documents by enhancing the text context with generated captions from figures: besides the text in PDF documents, the generated captions can be used as extra inputs to the PDF parser. Figure captioning can also enhance user experience for providing another way to access figure contents. For example, people with visible impairment can "read" figures via a text-to-speech system which essentially takes the generated captions as input.

Figure 1: Caption Generation for A Regular Image: "Giraffes in their wood and grass zoo enclosure". Example taken from COCO dataset [28].

Different from regular image captioning, i.e., generating captions for regular images (Figure 1), our task focuses on generating captions for figures (Figure 2). The major

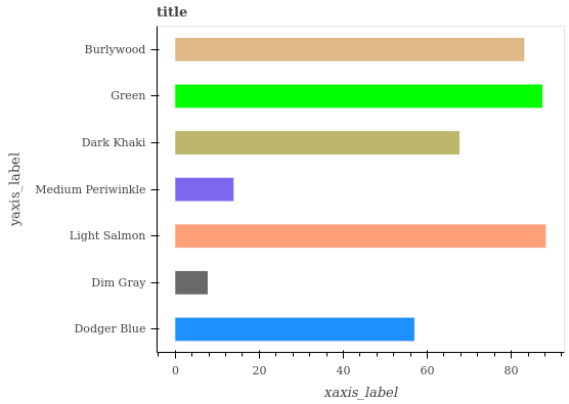


Figure 2: Caption Generation for A Figure: "There are seven different bars in this horizontal bar chart; they are DodgerBlue, DimGray, LightSalmon, MediumPeriwinkle, DarkKhaki, Green, Burlywood; LightSalmon is the maximum; LightSalmon is greater than DimGray." Example taken from our dataset FigCAP.

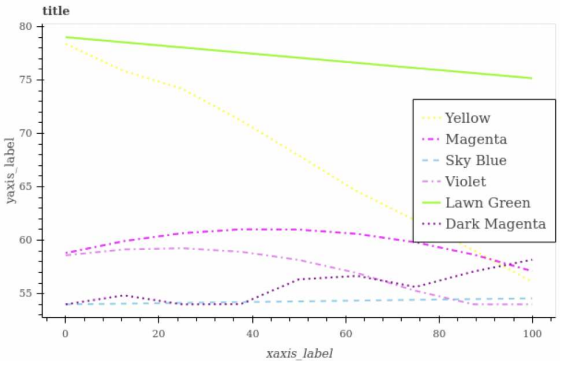


Figure 3: An example in FigCAP. We generate both high-level and detailed captions for the figure.

challenges include: 1) figures typically contain more "pivot" elements than regular images. For instance, in Figure 1, the pivot element is the "Giraffe" while in Figure 2 all the bars and their labels are pivot elements if the machine needs to learn "the bar with label LightSalmon has the maximum value". 2) a figure captioning system needs to determine how important a "pivot" object is compared to other "pivot" objects , especially without any additional guides.

In this work, we target at this problem and propose our methods to solve it. Our main contributions are:

- We collect a new figure captioning dataset FigCAP.
- We propose novel attention mechanisms to improve the performance of the captioning model.
- We train the captioning model under sequence-level policy with reinforcement learning such that the exposure bias issue is properly handled.
- Empirical experiments show that the proposed methods outperform baselines under several evaluation metrics.

2. Related Work

Image Captioning The existing approaches for image captioning largely fall into two categories: top-down and bottom-up. The bottom-up approaches first output key words describing different aspects of an image such as visual concepts, objects, attributes, and then combines them to sentences. [12, 25, 10, 26, 11] lie in this category. The successful application of deep learning in natural language processing, for example, machine translation, motivates the exploration on top-down methods, such as [31, 9, 18, 39, 42]. These approaches formulate the image captioning as a machine translation problem, directly translating an image to sentences by utilizing the encoder-decoder framework. The approaches based on deep neural networks proposed recently largely fall into this category.

Visual Question Answering Another related problem for image understanding is VQA [20], which is to answer queries in natural language about an image. However, figure captioning distinguishes itself from VQA in two important aspects. First, the input is different. The input to a VQA system consists of an image/figure to be queried and a question. The question can be regarded as a guide to select relevant image features while generating an answer. In contrast, the input to a figure captioning system is typically the figure only, which means there is no additional guides. The captioning model needs to learn what image features are relevant and what aspects the generated caption should focus on. Second, the output of a VQA system is the answer to the given question, typically with only a few words. On the other hand, a figure captioning system usually need to produce a few sentences to describe the information clearly.

Traditional approaches for VQA include [4, 14, 19, 44, 35], which train a linear classifier or neural network with the combined features from images and questions. Bilinear pooling or related techniques are further proposed to efficiently and expressively combine the image and question features [13, 23]. Spatial attention was used to adaptively modify the visual features or local features in VQA [41, 43, 17]. Bayesian models were used to discover the relationships

between the features of the images, questions and answers [30, 19]. Previous works [3, 2] also decompose VQA into several sub-problems and solve these sub-problems individually.

Figure VQA VQA has been used to answer queries in natural language about figures. Kahou *et al.* [22] introduced FigureQA, a novel visual reasoning corpus for VQA task on figures. On this dataset, relation network [36] has strong performance among several models. Kafle *et al.* [21] presented DVQA, a dataset used to evaluate bar chart understanding by VQA. On this dataset, multi-output model and SAN with dynamic encoding model have been shown to achieve better performances.

3. Background

3.1. Sequence-Generation Model

A sequence-generation model generates a sequence $Y = (y_1, \ldots, y_T)$ conditioned on an object X, where $y_t \in \mathcal{A}$ is a generated token at time t and $\mathcal{A}$ is the set of output tokens. The length of an output sequence is denoted as T, and $Y_{1,\ldots,t}$ indicates a subsequence $(y_1, \ldots, y_t)$. The data are given with (X, Y) as pairs to train a sequence-generation model. We denote the output a sequence-generation model as $\hat{Y}$.

Starting from the initial hidden state $\boldsymbol{s}_0$, a RNN produces a sequence of states $(\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_T)$, given a sequence-feature representation $(e(y_1), e(y_2), \ldots, e(y_T))$, where $e(\cdot)$ denotes a function mapping a token to its feature representation. Let $\boldsymbol{e}_t \triangleq e(y_t)$. The states are generated by applying a transition function $h : \boldsymbol{s}_t = h(\boldsymbol{s}_{t-1}, \boldsymbol{e}_t)$ for T times. The transition function h is implemented by a cell of an RNN, with popular choices being the Long Short-Term Memory (LSTM [16]) and Gated Recurrent Units (GRU [7]). We use LSTM in this work. To generate a token $\hat{y}_t \in \mathcal{A}$, a stochastic output layer is applied on the current state $\boldsymbol{s}_t$:

$$\hat{y}_t \sim \text{Multi}(1, \text{softmax}(g(\boldsymbol{s}_{t-1}))),$$
$$\boldsymbol{s}_t = h(\boldsymbol{s}_{t-1}, e(\hat{y}_t))$$

where $\text{Multi}(1, \cdot)$ denotes one draw from a multinomial distribution, and $g(\cdot)$ represents a linear transformation. Since the generated sequence Y is conditioned on X, one can simply start with an initial state encoded from X: $\boldsymbol{s}_0 = \boldsymbol{s}_0(X)$ [5, 7]. Finally, a conditional RNN can be trained for sequence generation with gradient ascent by maximizing the log-likelihood of a generative model.

3.2. Sequence-Level Training

Sequence-generation models are typically trained with Teacher-Forcing, which maximizes the likelihood estimation (MLE) of the next ground-truth word given the previous ground-truth word. This approach accelerates the convergence of training, but introduces exposure bias [33], caused by the mismatch between training and testing. The error will accumulate during testing, and this problem becomes more severe when the sequence become longer.

Sequence generation with reinforcement learning (RL) can alleviate exposure bias and improve the performance by directly optimizing the evaluation metrics via sequence-level training. Instead of training in word-level as MLE, sequence-level training is guided by the reward of the sequence. Variants of this method include adding actor-critic [5] or self-critical baselines [34, 1] to stabilize the training. Besides, [29] used image retrieval model to discriminate the generated and reference captions combined with sequence-level training.

4. FigCAP

FigureSeer [37], DVQA[21] and FigureQA [22] are figure understanding datasets proposed in the recent years. FigureSeer contains figures from research papers, while plots in both DVQA and FigureQA are synthetic. Due to the synthetic nature, one can generate as many figures, accompanied by questions and answers as he wants. Therefore, the size of FigureSeer is relatively small compared to DVQA and FigureQA, though its figures come from real data. In terms of figure type, FigureQA contains vertical and horizontal bar charts, pie charts, line plots, and dot-line plots while DVQA has only bar charts. Also, reasoning ability is important for captioning approaches to generate good quality captions. Note that FigureQA is designed for visual reasoning task. Considering these factors, we collect our dataset, FigCAP, with figures from FigureQA.

FigCAP consists of figure-caption pairs. We develop two datasets: **FigCAP-H** and **FigCAP-D** for two different use cases. FigCAP-H contains **H**igh-level descriptions for figure captions. In contrast, FigCAP-D contains **D**etailed descriptions for figure captions. The figures in both FigCAP-H and FigCAP-D are generated by the same method as FigureQA [22].

They have five different types: horizontal bar chart, vertical bar chart, pie chart, line plot and dotted line plot. The number for each type is the same. The captions in both datasets are obtained by converting the QA pairs in FigureQA. For example, given a line plot as shown in Figure 3 and a QA pair (*"Is Sky Blue less than Lawn Green?"*, *"Yes"*), we derive one sentence for describing the line plot: *"Sky Blue is less than Lawn Green"*. After converting all the QA pairs associated with the line plot and shuffling them, we obtain the high level captions:

"This figure is a line plot. It contains six categories: Yellow, Magenta, Sky Blue, Violet, Lawn Green and Dark Magenta."

and the detailed captions:

"[Dark Magenta has the lowest value. Lawn Green has the highest value. Sky Blue is less than Lawn Green. Yellow

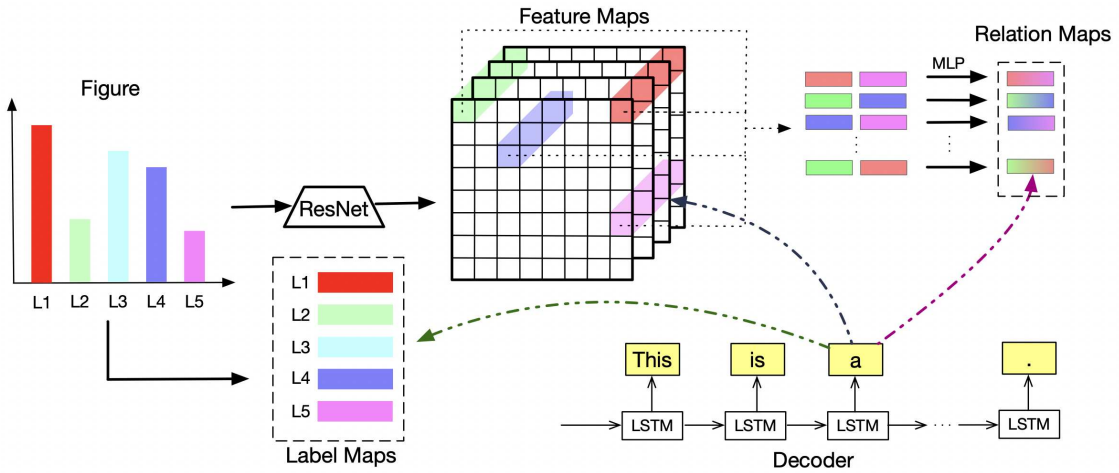


Figure 4: Model overview: Our model takes a figure image as input, encodes it with *ResNet*. Decoder is a LSTM with Attention Models *Att_F*, *Att_R* and *Att_L*. Solid arrow lines show data flows, and dash arrow lines show the attentions.

is greater than Violet. Sky Blue has the minimum area under the curve. Lawn Green is the smoothest. Yellow intersects Magenta.]".

Table 1 shows the numbers of figure-caption pairs for both datasets. Their sizes are similar to the setting in (Gan et al., 2017). The vocabulary size for captions in both of them is 126 and average lengths of the captions are 17 and 39, respectively. Note that we generate two versions for detailed captions: FigCAP-D1 and FigCAP-D2, with different sizes for testing. Also, since our datasets are synthetic, one can generate the figure-caption pairs as many as needed.[1]

Datasets	Training	Validation	Testing
FigCAP-H/D1	99,360	5,000	5,152
FigCAP-D2	99,360	5,000	19,857

Table 1: Statistics for FigCAP-H and FigCAP-D.

5. Captioning Model

Figure 4 shows the architecture of our proposed captioning model. The major components include an encoder, a decoder, Relation Maps, Feature Maps, and Label Maps. First, a Residual Network [15] encodes an input figure (the horizontal bar chart in Figure 4) into Feature Maps. Let $\boldsymbol{X}$ be the input figure. Then its Feature Maps $\boldsymbol{F}$ is the output of the encoder *ResNet*:

$$\boldsymbol{F} = ResNet(\boldsymbol{X})$$

Feature Maps $\boldsymbol{F}$ is then used to initialize the decoder, a LSTM [16]:

$$\boldsymbol{c}_0 = \sigma(\boldsymbol{W}_{Ic}\boldsymbol{F}), \boldsymbol{h}_0 = \sigma(\boldsymbol{W}_{Ih}\boldsymbol{F})$$

$\sigma(.)$ is the sigmoid function, and $\boldsymbol{W}_{Ic}$, $\boldsymbol{W}_{Ic}$ are trainable parameters. Two special tokens BOS and EOS represent the beginning and the end of each caption during training. We use the one-hot vector $\mathbf{1}_{\boldsymbol{y,t}}$ to represent the word y_t, and the encoding $\mathbf{1}_{\boldsymbol{y,t}}$ is further embedded by a linear embedding $\boldsymbol{E}$.

$$\boldsymbol{e}_t = e(y_t) = \boldsymbol{E}\mathbf{1}_{\boldsymbol{y,t}}, t > 0$$
$$\boldsymbol{e}_0 = \mathbf{0}, otherwise$$

At each step t, the LSTM is updated according to its input: both word vector $\boldsymbol{e}_t$ and context vector $\boldsymbol{d}_t$ (Section5.1.4). Eventually, the LSTM predicts the next word y_t according to the following:

$$\tilde{\boldsymbol{y}}_t = \sigma(\boldsymbol{W}_h\boldsymbol{h}_t + \boldsymbol{W}_d\boldsymbol{d}_t)$$
$$y_t \sim \text{Softmax}(\tilde{\boldsymbol{y}}_t)$$

Note that both the context vector $\boldsymbol{d}_t$ and $\boldsymbol{h}_t$ are inputs to the non-linear layer for computing $\tilde{\boldsymbol{y}}_t$. We illustrate details for computing context vector d_t with multiple attention mechanism in next section.

5.1. Attention Models for Figure Captioning

We employ LSTM [16] for decoding. With proposed attention models, the decoder may optionally attend to the label maps, feature maps and/or relation maps. The objective of figure captioning is to maximize likelihood or total rewards. The details of each component will be presented in the following subsections.

Attention mechanism has been widely used in the encoder-decoder structure to improve the decoding performance. We

[1] We will release our datasets and the generation code.

propose two attention models: Relation Maps Attention (*Att_R*), and Label Maps Attention (*Att_L*). We also introduce Feature Maps Attention (*Att_F*). Context vector $\boldsymbol{d}_t$ can be computed from one of them, or combination of them.

5.1.1 Feature Maps Attention *Att_F*

Feature Maps Attention Model takes Feature Maps $\boldsymbol{F}$ ($\boldsymbol{F}$ contains m feature vectors; $\boldsymbol{F} \in R^{m \times d}$) and the hidden state $\boldsymbol{h}_{t-1}$ of LSTM as input. For each feature $\boldsymbol{f}_j$ in $\boldsymbol{F}$, it computes a score between $\boldsymbol{f}_j$ and $\boldsymbol{h}_{t-1}$. With these scores as weights, it computes the context vector $\boldsymbol{c}_t$ as the weighted sum of all features in the feature maps. Equation 1 defines Feature Maps Attention Model:

$$\begin{aligned} e_{tj} &= Att\_F(\boldsymbol{h}_{t-1}, \boldsymbol{f}_j) \\ &= \boldsymbol{v}_a^T \tanh(\boldsymbol{W}_a \boldsymbol{f}_j + \boldsymbol{U}_a \boldsymbol{h}_{t-1}) \\ \alpha_{tj} &= \frac{\exp(e_{tj})}{\sum_{k=1}^{m} \exp(e_{tk})}, \quad \boldsymbol{c}_t = \sum_{j=1}^{m} \alpha_{tj} \cdot \boldsymbol{f}_j \end{aligned} \tag{1}$$

where $\boldsymbol{f}_j$ is the j-th feature in the feature maps $\boldsymbol{F}$, $\boldsymbol{c}_t$ is the context vector and α_{tj} is an attention weight.

5.1.2 Relation Maps Attention *Att_R*

Reasoning network, built upon the feature maps, produces relation maps which embed logical information in the given figure.

In order to generate correct captions describing relations among the labels (e.g. A is the maximum, B is greater than C, C is less than D.), it is essential to perform reasoning among labels in a given figure. Inspired by Relation Networks [36], we propose the Relation Maps Attention Model (*Att_R*). We consider each feature vector $\boldsymbol{f}_j \in R^d$ in the feature maps $\boldsymbol{F}$ as an object. For any two "objects", for example, $\boldsymbol{f}_i$ and $\boldsymbol{f}_j$, we concatenate them and feed the vector into a MLP, resulting in a relation vector $\boldsymbol{r}_{ij} \in R^{\hat{d}}$:

$$\boldsymbol{r}_{ij} = MLP(concat(\boldsymbol{f}_i, \boldsymbol{f}_j)), \boldsymbol{r}_{ij} \in R^{\hat{d}} \tag{2}$$

Therefore, the relation maps $\boldsymbol{R}$ contains m^2 relation vectors (m is the number of feature vectors in feature maps $\boldsymbol{F}$). Given the relation maps $\boldsymbol{R}$, at decoding step t, *Att_R* computes the relation context vector $\hat{\boldsymbol{c}}_t$ as follows:

$$\begin{aligned} \hat{e}_{tk} &= Att\_R(\boldsymbol{h}_{t-1}, \boldsymbol{r}_k) \\ &= \boldsymbol{v}_b^T \tanh(\boldsymbol{W}_b \boldsymbol{r}_k + \boldsymbol{U}_b \boldsymbol{h}_{t-1}) \\ \beta_{tk} &= \frac{\exp(\hat{e}_{tk})}{\sum_{l=1}^{m^2} \exp(\hat{e}_{tl})}, \quad \hat{\boldsymbol{c}}_t = \sum_{k=1}^{m^2} \beta_{tk} \cdot \boldsymbol{r}_k \end{aligned} \tag{3}$$

where $\boldsymbol{r}_k$ is the k-th relation vector in relation maps $\boldsymbol{R}$ and β_{tk} is an attention weight.

Note that more complex relationships can be induced from pairwise relations, e.g. $A > B$ and $B > C$ lead to $A > C$. The relation map $\boldsymbol{R}$ obtained from Reasoning Net represents abstract objects that implicitly represent object(s) in the figure, not explicitly represent one specific object like a bar or a line.

5.1.3 Label Maps Attention *Att_L*

We propose Label Map Attention Model (*Att_L*) where the LSTM attends to Label Map $\boldsymbol{L}$ for decoding. Label Map $\boldsymbol{L}$ is composed of embeddings of those labels appearing in the figure. If n is the number of labels in the figure, then $\boldsymbol{L}$ contains n vectors. Let $\boldsymbol{l}_j$ be the j-th vector in the label maps $\boldsymbol{L}$, we define *Att_L* as follows:

$$\begin{aligned} \tilde{e}_{tj} &= Att\_L(\boldsymbol{h}_{t-1}, \boldsymbol{l}_j) \\ &= \boldsymbol{v}_c^T \tanh(\boldsymbol{W}_c \boldsymbol{l}_j + \boldsymbol{U}_c \boldsymbol{h}_{t-1}), \\ \gamma_{tj} &= \frac{\exp(\tilde{e}_{tj})}{\sum_{j=1}^{n} \exp(\tilde{e}_{tj})}, \quad \tilde{\boldsymbol{c}}_t = \sum_{j=1}^{n} \gamma_{tj} \cdot \boldsymbol{l}_j \end{aligned} \tag{4}$$

where $\tilde{c}_t$ is the context vector at time step t.

Note that figure labels are also used as inputs. For example, in Figure 3, n is 6; Yellow, Magenta, Sky Blue, Violet, Lawn Green and Dark Magenta are extracted from it using state-of-the-art computer vision techniques such as Optical Character Recognition (OCR). Since labels appear in the caption of the input figure, instead to define a new set of vectors to represent the labels in the Label Maps $\boldsymbol{L}$, we use a subset of the word embeddings $\boldsymbol{E}$. In Figure 3, embeddings $\boldsymbol{e}$ for Yellow, Magenta, Sky Blue, Violet, Lawn Green and Dark Magenta compose its Label Map $\boldsymbol{L}$.

5.1.4 Context Vector $\boldsymbol{d}_t$

In the captioning model, the decoder can use any combination of *Att_F*, *Att_R* and *Att_L*, or it can use only one of them. For example, if we incorporate all three Attention Models (Eq.1,3,4) in the caption generation model, the final context vector $\boldsymbol{d}_t$, used as input to the decoder, is as follows:

$$\boldsymbol{d}_t = \text{concat}(\boldsymbol{c}_t, \hat{\boldsymbol{c}}_t, \tilde{\boldsymbol{c}}_t) \tag{5}$$

We explore different combinations of Attention Models for generating captions. More details are in Experimental Evaluations (Section 6).

5.2. Hybrid Training Objective

In the traditional method [40], "Teacher forcing" is widely used for the supervised training of decoders. Given an object X, it maximizes the likelihood of the target word y_t, given the previous target sequences Y_{t-1}:

$$L_{sl} = -\sum_{t=1}^{T} \log p(y_t | Y_{t-1}, x). \tag{6}$$

Due to the exposure bias and indirectly optimizing the evaluation metric, supervised training usually can not provide best results. Besides, the word-level training is difficult to handle the generation with different but reasonable word-orders. As a long-text-generation task, figure captioning will accumulate more errors as more words predicted and diversity will be undermined.

Sequence-level training with RL can effectively alleviate the mentioned problems, by directly optimizing the sequence-level evaluation metric. We use the self-critical policy gradient training algorithm in our model. Specifically, a sequence $\hat{Y}^b$ is generated by greedy word search, *i.e.*, selecting the word with the highest probability. Then, another sequence $\hat{Y}^s$ is generated by sampling next word $\hat{y}_t^s$ according to the probability distribution of $p(\hat{y}_t^s|\hat{Y}_{t-1}^s)$. The sampled sequence $\hat{Y}^s$ is an exploration of the policy for generating the caption, and the sequence $\hat{Y}^b$ obtained from greedy search is the baseline. We use CIDEr as the sequence-level evaluation metric and compute CIDEr for $\hat{Y}^s$ and $\hat{Y}^b$, respectively. The reward is defined as the difference of CIDEr between the sampled sequence $\hat{Y}^s$ and greedy sequence $\hat{Y}^b$. Let $r(Y)$ be the CIDEr of sequence Y. We minimize the sequence-level loss (i.e. maximizing the rewards):

$$L_{rl} = -(r(\hat{Y}^s) - r(\hat{Y}^b)) \sum_{t=1}^{T} \log p(\hat{y}_t^s|\hat{Y}_{t-1}^s, x) \qquad (7)$$

Our model is pretrained with MLE loss to provide more efficient policy exploration. Good explorations are encouraged while poor explorations are discouraged in future generation. However, we found that purely optimizing sequence-level evaluation metric, such as CIDEr, may lead to overfitting. To tackle this issue, we use hybrid training objective in our model, considering both word-level loss L_{sl} provided by MLE (Eq.6) and sequence-level loss L_{rl} computed by RL (Eq.7):

$$L_{hybrid} = \lambda L_{rl} + (1 - \lambda) L_{sl}, \qquad (8)$$

where λ is a scaling factor balancing the weights between L_{rl} and L_{sl}. In practice, λ starts from 1 and slowly decays to 0, then only reinforcement learning loss is used to improve our generator.

6. Experimental Evaluations

In this section, we validate our proposed models on the FigCAP-H and FigCAP-D. Specifically, we evaluate the models in two use cases: generating high-level captions and generating detailed captions for figures, respectively. We perform an ablation study on the improvements brought by each part of our proposed method.

6.1. Experimental Settings

We implement the following models with TensorFlow, and conduct experiments on a single nVidia Tesla V100 GPU. For any of them, *ResNet-50* pretrained on ImageNet [8] is used as the encoder and a 256-unit LSTM is the decoder.

- **CNN-LSTM**: This baseline model uses basic CNN-LSTM structure, without any Attention Model.
- **CNN-LSTM+*Att_F***: This model uses *Att_F* for decoding. Similar model is used in natural image captioning [42].
- **CNN-LSTM+*Att_F*+*Att_L***: This model uses both *Att_F* and *Att_L* for decoding.
- **CNN-LSTM+*Att_F*+*Att_L*+*Att_R***: This model uses *Att_F*, *Att_L* and *Att_R* for decoding.
- **CNN-LSTM+*Att_F*+*Att_L*+*Att_R*+RL**: The loss function of this model is described in Section 5.2. Training with RL can improve the model's performance when handling long captions, which is suitable for FigCAP-D.

All of them are optimized with Adam [24] on the training set and evaluated on the testing set. We tune hyperparameters on the validation set. Following [42] and [34], we use CIDEr [38], BLEU1-4 [32], METEOR [6] and ROUGEL [27] as evaluation metrics. Note that we only evaluate models containing *Att_R* on FigCAP-D since only long captions contain relation information.

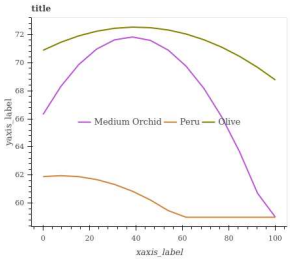
We evaluate the proposed models for the task of generating detailed captions and report experimental results in Table 2. We also report examples generated by our model *CapGen+Att_All* in Figure 5. The results in Table 2 demonstrate the importance of attention mechanisms, as the models with *Att_F+Att_L* and *Att_All* outperform the baselines on all datasets. The RL model also obtains substantially better performance than the MLE model.

6.2. Error Analysis and Future Work

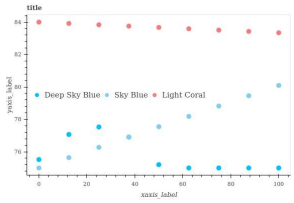
Experimental results show that the proposed Attention Models for figure captioning are capable of improving the quality of generated captions. Compared with the baseline model CNN-LSTM, we observe that models that use Attention Models achieve better performance on both FigCAP-H and FigCAP-D. This result indicates that attention-based models are useful for figure captioning. Second, we found that the effects of *Att_F* is higher in FigCAP-D than FigCAP-H. It indicates that generating high-level descriptions does not actually need complex Attention Models since it is more likely a classification task which can be accomplished based on general information of the figure. In addition, we find that Relation Maps $\boldsymbol{R}$ are useful if descriptions about relations

Models	Evaluation Metrics						
	CIDEr	BLEU1	BLEU2	BLEU3	BLEU4	METEOR	ROUGE
CNN-LSTM	0.158	0.055	0.050	0.044	0.038	0.115	0.244
CNN-LSTM+*Att_F*	0.868	0.215	0.200	0.181	0.159	0.200	0.401
CNN-LSTM+*Att_F*+*Att_L*	0.917	0.232	0.214	0.194	0.170	0.207	0.413
CNN-LSTM+*Att_All*	1.036	0.312	0.290	0.264	0.233	0.231	0.468
CNN-LSTM+*Att_All*+RL	**1.179**	**0.404**	**0.367**	**0.324**	**0.270**	**0.263**	**0.489**

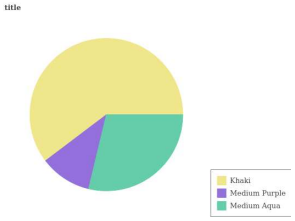
Table 2: Results for FigCAP-D: Detailed Caption Generation. *Att_All=Att_F+Att_L+Att_R*.



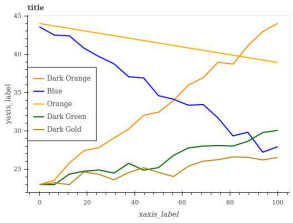
(a) This figure is a line plot; there are three different categories in it; their names are mediumorchid, olive, brown.



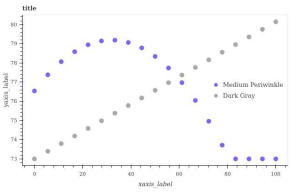
(b) This is a dot line plot; there are 3 different labels; their names are deepskyblue, lightcoral, skyblue.



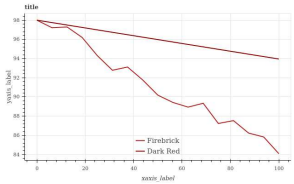
(c) There are 3 different labels in this pie chart; their names are khaki, mediumaqua, mediumpurple.



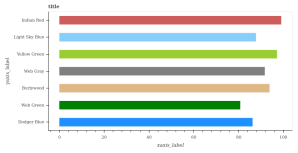
(d) It is a line plot, with five lines; their names are orange, **mediumblue, olive, orangered, greenyellow**.



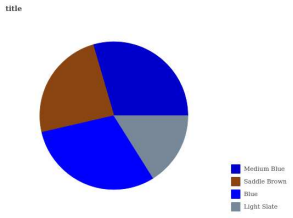
(e) Darkgray has the maximum area under the curve; darkgray is the smoothest; mediumperiwinkle is the roughest; darkgray has the lowest value; darkgray has the highest value; mediumperiwinkle intersects darkgray.



(f) Firebrick has the minimum area under the curve; darkred has the maximum area under the curve; darkred is the smoothest; firebrick is the roughest; firebrick has the **lowest** value; firebrick intersects darkred.



(g) Webgray is the minimum; **yellowgreen** is the maximum; webgreen is greater than yellowgreen; webgreen is less than indianred; webgray is the high median; webgray is the low median.



(h) **Mediumblue** is the maximum; mediumblue is greater than saddlebrown; saddlebrown is less than **saddlebrown**; mediumblue is the high median; saddlebrown is the low median.

Figure 5: Captions generated by *CapGen+Att_All* on FigCAP-H (top) and FigCAP-D (bottom).

of a figure's labels are desired (*e.g.*, Bar A is higher than Bar B; Bar C has the largest value).

Furthermore, with RL we can alleviate the exposure bias issue and directly optimize the evaluation metric used at the inference time. This enables us to achieve better performance in the generation of long captions.

Analysis of the generated captions revealed that one common error made by *CapGen+Att_All* is the generation of incorrect labels, as in Figure 5d, 5f and 5h. By comparing the true labels with the generated labels, we found that the model may generate a label that is close to the true label, e.g, use MediumBlue for Blue, and OrangeRed for Orange. An approach that we plan to investigate in future work is to incorporate a ranking model, which allows current models select the label with the highest score as the candidate from a set of similar labels.

Another error is the incorrect label relation. For example, in Figure 5f, YellowGreen is less than IndianRed and it is the second largest instead of the maximum. The Relation Maps $\boldsymbol{R}$ is built from the feature vectors in $\boldsymbol{F}$ currently, which leads to a fixed number of "object". A solution is to incorporate the label representation while computing the Relation Maps $\boldsymbol{R}$, such that $\boldsymbol{R}$ reflects the actual number of labels in the figure.

In addition, we plan to use more advanced sampling methods to generate the candidate sequence for reinforcement

learning, in order to achieve a better balance between exploration and exploitation. We also plan to conduct experiments on real dataset with proposed models.

7. Conclusion

In this work, we investigate the problem of figure captioning. We develop new datasets for different use cases. FigCAP-H contains high-level descriptions for figure, while FigCAP-D contains detailed descriptions such as label relations. We also propose two novel attention mechanisms. To achieve accurate generation of labels, we design Label Maps Attention. To discover the relations among labels, we propose Relation Maps Attention. In order to handle long sequence generation and alleviate the exposure bias issue, we utilize sequence-level training with reinforcement learning. Experimental results show that the proposed models, *CapGen+Att_F+Att_L*, *CapGen+Att_All*, and *CapGen+Att_All+RL*, effectively generate captions over figures under several metrics. A successful solution to this task allows figure content to be accessible to those with visual disabilities by providing input to a text-to-speech system; and enables automatic parsing of vast repositories of documents where figures are pervasive.

References

[1] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang. Bottom-up and top-down attention for image captioning and vqa. In *CVPR*, 2017.

[2] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Learning to compose neural networks for question answering. In *NAACL-HLT*, pages 1545–1554, 2016.

[3] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. Neural module networks. In *CVPR*, June 2016.

[4] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. Lawrence Zitnick, and D. Parikh. Vqa: Visual question answering. In *ICCV*, pages 2425–2433, 2015.

[5] D. Bahdanau, P. Brakel, K. Xu, A. Goyal, R. Lowe, J. Pineau, A. Courville, and Y. Bengio. An actor-critic algorithm for sequence prediction. In *ICLR*, 2017.

[6] S. Banerjee and A. Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.

[7] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. Ieee, 2009.

[9] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, pages 2625–2634, 2015.

[10] D. Elliott and F. Keller. Image description using visual dependency representations. In *EMNLP*, pages 1292–1302, 2013.

[11] H. Fang, S. Gupta, F. Iandola, R. K. Srivastava, L. Deng, P. Dollár, J. Gao, X. He, M. Mitchell, J. C. Platt, et al. From captions to visual concepts and back. In *CVPR*, pages 1473–1482, 2015.

[12] A. Farhadi, M. Hejrati, M. A. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth. Every picture tells a story: Generating sentences from images. In *ECCV*, pages 15–29. Springer, 2010.

[13] A. Fukui, D. H. Park, D. Yang, A. Rohrbach, T. Darrell, and M. Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. *arXiv preprint arXiv:1606.01847*, 2016.

[14] H. Gao, J. Mao, J. Zhou, Z. Huang, L. Wang, and W. Xu. Are you talking to a machine? dataset and methods for multilingual image question. In *NIPS*, pages 2296–2304, 2015.

[15] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016.

[16] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[17] I. Ilievski, S. Yan, and J. Feng. A focused dynamic attention model for visual question answering. *arXiv preprint arXiv:1604.01485*, 2016.

[18] X. Jia, E. Gavves, B. Fernando, and T. Tuytelaars. Guiding long-short term memory for image caption generation. *arXiv preprint arXiv:1509.04942*, 2015.

[19] K. Kafle and C. Kanan. Answer-type prediction for visual question answering. In *CVPR*, pages 4976–4984, 2016.

[20] K. Kafle and C. Kanan. Visual question answering: Datasets, algorithms, and future challenges. *Computer Vision and Image Understanding*, 163:3–20, 2017.

[21] K. Kafle, B. Price, S. Cohen, and C. Kanan. Dvqa: Understanding data visualizations via question answering. In *CVPR*, pages 5648–5656, 2018.

[22] S. E. Kahou, A. Atkinson, V. Michalski, Á. Kádár, A. Trischler, and Y. Bengio. Figureqa: An annotated figure dataset for visual reasoning. *arXiv preprint arXiv:1710.07300*, 2017.

[23] J.-H. Kim, S.-W. Lee, D. Kwak, M.-O. Heo, J. Kim, J.-W. Ha, and B.-T. Zhang. Multimodal residual learning for visual qa. In *NIPS*, pages 361–369, 2016.

[24] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[25] G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg. Baby talk: Understanding and generating image descriptions. In *CVPR*. Citeseer, 2011.

[26] R. Lebret, P. O. Pinheiro, and R. Collobert. Simple image description generator via a linear phrase-based approach. *arXiv preprint arXiv:1412.8419*, 2014.

[27] C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*, 2004.

[28] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014.

[29] R. Luo, B. Price, S. Cohen, and G. Shakhnarovich. Discriminability objective for training descriptive captions. In *CVPR*, 2018.

[30] M. Malinowski and M. Fritz. A multi-world approach to question answering about real-world scenes based on uncertain input. In *NIPS*, pages 1682–1690, 2014.

[31] J. Mao, W. Xu, Y. Yang, J. Wang, Z. Huang, and A. Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). *arXiv preprint arXiv:1412.6632*, 2014.

[32] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318. Association for Computational Linguistics, 2002.

[33] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba. Sequence level training with recurrent neural networks. In *ICLR*, 2016.

[34] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel. Self-critical sequence training for image captioning. In *CVPR*, 2016.

[35] K. Saito, A. Shin, Y. Ushiku, and T. Harada. Dualnet: Domain-invariant network for visual question answering. In *ICME*, pages 829–834. IEEE, 2017.

[36] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap. A simple neural network module for relational reasoning. In *NIPS*, pages 4967–4976, 2017.

[37] N. Siegel, Z. Horvitz, R. Levin, S. Divvala, and A. Farhadi. Figureseer: Parsing result-figures in research papers. In *ECCV*, pages 664–680. Springer, 2016.

[38] R. Vedantam, C. Lawrence Zitnick, and D. Parikh. Cider: Consensus-based image description evaluation. In *CVPR*, pages 4566–4575, 2015.

[39] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *CVPR*, pages 3156–3164. IEEE, 2015.

[40] R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

[41] H. Xu and K. Saenko. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. In *ECCV*, pages 451–466. Springer, 2016.

[42] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, pages 2048–2057, 2015.

[43] Z. Yang, X. He, J. Gao, L. Deng, and A. Smola. Stacked attention networks for image question answering. In *CVPR*, pages 21–29, 2016.

[44] B. Zhou, Y. Tian, S. Sukhbaatar, A. Szlam, and R. Fergus. Simple baseline for visual question answering. *arXiv preprint arXiv:1512.02167*, 2015.