

## A Model Details

**Generative language model** We use a simple forward recurrent neural model with cross-entropy loss in all model variations:

$$P(w_{t+1}|w_{0:t}, c) = \hat{y}_{t+1} = F(w_{0:t}, c; \theta) \quad (2)$$

$$loss(k_{1:t}, \theta) = - \sum_{t=0}^T \log(\delta_{k_t} \cdot \hat{y}_t) \quad (3)$$

where  $F$  represents the neural network function with parameters  $\theta$ , inputs  $w_{0:t}$  the sequence of words with  $w_0$  the sentence marker ‘ $\langle s \rangle$ ’, and  $c$  to represent the image or with additional top-down knowledge.  $\hat{y}_{t+1} \in [0, 1]^{|V|}$  is a categorical distribution over the choices in vocabulary  $V$  for the conditional probability of the next word. The loss is calculated for each sample of word sequence  $[v_{k_0}, v_{k_1}, \dots, v_{k_T}]$ , which  $k_t \in \{1, \dots, |V|\}$  refers to the word index in the vocabulary, and  $\delta_{k_t}$  is its one-hot encoding.

**Simple encoder-decoder** An encoder-decoder architecture without spatial attention, similar to (Vinyals et al., 2015), is the most simple baseline for setting up the experiments and designing the foundation for fusing vision and language. The input to the model is an image and the start symbol  $\langle s \rangle$  for the language model decoder. The word embeddings  $e_t$  are concatenated with the scene visual features ( $\bar{v}$ ). The embeddings are randomly initialised and learned as a parameter set for the model. The visual vectors are produced by a pre-trained ResNet50 (He et al., 2016). Then,  $\bar{v}$  is made by a dense layer translating the visual vector to a unified tensor size for computational convenience. This layers also helps fine-tuning the visual features.

$$F_v(x) = \text{ReLU}(\mathbf{W}_v \cdot x + \mathbf{b}_v)$$

$$\bar{v} = \frac{\sum_{i=1}^k F_v(v'_i)}{k}$$

where  $F_v$  the function in Figure 2,  $v'_i \in \mathbb{R}^{2048}$  with ResNet50 dimensions,  $\mathbf{W}_v \in \mathbb{R}^{100 \times 2048}$  and  $\mathbf{b}_v \in \mathbb{R}^{100}$  are parameters to be learned as fine-tuning. The resulting vector is concatenated to a word embedding and fed to the Long-Short Term Memory (LSTM) network (Hochreiter and Schmidhuber, 1997) and its output to a multi-layer perceptron (MLP) with a softmax layer which predicts the next word, as it was described earlier in Equation 2. This function would be:

$$\hat{y}_{t+1} = \text{softmax}(\text{MLP}(\text{LSTM}([e_t; \bar{v}], \mathbf{h}_{t-1}))) \quad (4)$$

where  $e_t$  and  $\mathbf{h}_t$  respectively represent the word embedding and the hidden unit in recurrent cell at time  $t$  of the word sequence (Figure 3a). Ideally, the spatial features must be learned bottom-up in  $\bar{v}$  as other visual features in the deep layers of convolutions in ResNet.

**Adaptive attention** The simple encoder-decoder architecture relies on bottom-up learning of visual features and geometric arrangement of objects. However, it has been shown in recent image captioning models that a spatial attention mechanism to localise each word improves the language generation (Xu et al., 2015). Moreover, the attentions can be learned as an adaptation of modalities. Based on this assumption we will use the adaptive attention similar to (Lu et al., 2017). In generalisation of adaptive attention, the feature vectors including visual features from different locations as well as the contextual language features and other modalities  $\hat{\mathbf{f}} = [f_1, f_2, \dots, f_n]$  are fused with weighted sum according to their attention weight  $\hat{\alpha}$ .

$$\hat{c}_t = \sum_{i=1}^n \alpha_i f_i \quad (5)$$

where  $\hat{c}_t$  represents the fused vector after applying adaptive attention on  $n$  feature vectors. Knowing which features in what degree contribute to prediction of the next word is decided in a multi-layer perceptron ( $MLP_a$ ) with softmax as  $\hat{\alpha}$  in Figure 12. This module is formalised in a sequential process as follows:

$$z_t = \mathbf{W}_a^2 \tanh(\mathbf{W}_a^1 \cdot \hat{\mathbf{f}}_t)$$

$$\hat{\alpha}_t = \text{softmax}(z_t).$$

where  $\hat{\alpha}_t = [\alpha_{t,1}, \alpha_{t,2}, \dots, \alpha_{t,n}]$  is the output of the module in time  $t$ , and  $\mathbf{W}_a^1, \mathbf{W}_a^2$  are the parameters of the module which will be trained in the model.

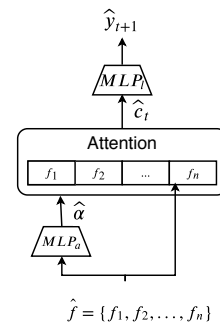


Figure 12: The generalised adaptive attention module.

**Bottom-up localisation** With visual feature representing each region of the image as in Figure 2, attention mechanism is going to work as localisation model. We designed the interaction between the attention mechanism and the language model more similar to (Anderson et al., 2018): two layers of stacked LSTM, the first stack ( $LSTM_a$ ) to produce features for attention model, then the second stack ( $LSTM_l$ ) to produce contextualised linguistic features to be fused with attended visual features (Figure 3b). This design makes it easier to be extended with top-down visual vectors.

$$\hat{c}_t = \sum_{i=1}^{49} \alpha_{t,i} v_i + \alpha_{t,50} h_t^l \quad (6)$$

where each  $v_i$  is a visual feature referring to one of the 49 locations in Figure 2, and  $h_t^l$  is the contextualised language feature from  $LSTM_l$ .

**Top-down localisation** Unlike the bottom-up localisation, the top-down method has a list of regions of interest pre-processed from other procedures. The process of region proposals can be part of a bottom-up process as in Anderson et al. (2018) or Johnson et al. (2016) which instead of the grids of regions in ConvNets in Figure 2 a Faster R-CNN (Ren et al., 2015) is used to extract all possible regions of interest. In this paper, we use the bounding box annotations on images as the top-down localisation knowledge, then we use ResNet50 to extract visual features from these regions Figure 4. At this stage the top-down visual representation only proposes visual vectors of two objects in random order without their spatial role in intended descriptions shown in Figure 3d.

$$\hat{c}_t = \alpha_{t,1} v_{obj_1} + \alpha_{t,2} v_{obj_2} + \alpha_{t,3} h_t^l \quad (7)$$

where each  $v_{obj_1}$  and  $v_{obj_2}$  are the visual features referring to two regions in Figure 4, and  $h_t^l$  is the contextualised language feature from  $LSTM_l$ .

**Top-down target-landmark assignment** Another top-down information is the assignment of one region as the target and another region as the landmark. This top-down knowledge is encoded as the order in the list of two object, first object is the target and the second object is the landmark in Equation 7.

$$\hat{c}_t = \alpha_{t,1} v_{TARGET} + \alpha_{t,2} v_{LANDMARK} + \alpha_{t,3} h_t^l \quad (8)$$

where each  $v_{TARGET}$  and  $v_{LANDMARK}$  are the visual features referring to two regions in Figure 4 and their semantic role is defined top-down.

**Top-down geometric features** With top-down localisation we may lose the relative location of two objects since they are processed separately in two disconnected convolutional neural networks. Therefore, the top-down geometric features are required for grounding of denotation of the locational words. Additionally, representing geometric knowledge can encode the frame of reference. For example, a simple geometric relation between two bounding boxes can be an arrow from the centre of one bounding box to the centre of the other, however the choice between the order of objects depends on the frame of reference (i.e.  $obj_1 \rightarrow obj_2$  or  $obj_1 \leftarrow obj_2$ ). We represent the geometric features by considering the top-down target-landmark assignment (i.e.  $TARGET \rightarrow LANDMARK$ ). Therefore with these feature vectors we encode the top-down frame of reference as well. This creates different variations of feature fusions (Table 2).

In order to find the best encoding of top-down geometric features, we considered two different vectorisation strategies to represent relation between two bounding boxes Figure 5.

- (*mask*) a concatenation of two mask vectors in 49 locations (Figure 5a).
- (*VisKE*) a dense representation with 11 geometric features according to (Sadeghi et al., 2015) (Figure 5b): where  $dx, dy$  are changes in coordinates of the centres,  $ov, ov_1, ov_2$  the overlapping areas (total, relative to the first, and the second bounding box),  $h_1, h_2$  heights,  $w_1, w_2$  widths and  $a_1, a_2$  areas.

Then, a feed-forward network with two layers ( $F_s$ ) is used to project geometric features into a 100-dimension vector to become comparable with other modalities.

$$F_s(x) = W_s^2 \tanh(W_s^1 \cdot x + b_s^1) \\ s = F_s(s')$$

where  $s$  represents the transformed geometric spatial features, and  $W_s^2 \in \mathbb{R}^{100 \times 100}$ ,  $W_s^1 \in \mathbb{R}^{100 \times 11}$  (or  $\mathbb{R}^{100 \times 98}$ ) are the set parameters regarding this module to be learned in the model.

## B Examples of generated descriptions

More examples of generated descriptions with beam search of depth 5 are shown in Figure 13.

Model name	Visual features	Attention
<i>bu49</i>	$[\mathbf{v}_1, \dots, \mathbf{v}_{49}]$	$\hat{c}_t = \sum_{i=1}^{49} \alpha_{t,i} \mathbf{v}_i + \alpha_{t,50} \mathbf{h}_t^l$
<i>bu49 + mask</i>	$[\mathbf{v}_1, \dots, \mathbf{v}_{49}]$	$\hat{c}_t = \sum_{i=1}^{49} \alpha_{t,i} \mathbf{v}_i + \alpha_{t,50} \mathbf{h}_t^l + \alpha_{t,51} \mathbf{s}$
<i>bu49 + VisKE</i>	$[\mathbf{v}_1, \dots, \mathbf{v}_{49}]$	$\hat{c}_t = \sum_{i=1}^{49} \alpha_{t,i} \mathbf{v}_i + \alpha_{t,50} \mathbf{h}_t^l + \alpha_{t,51} \mathbf{s}$
<i>td</i>	$[\mathbf{v}_{obj_1}, \mathbf{v}_{obj_2}]$	$\hat{c}_t = \alpha_{t,1} \mathbf{v}_{TARGET} + \alpha_{t,2} \mathbf{v}_{LANDMARK} + \alpha_{t,3} \mathbf{h}_t^l$
<i>td + mask</i>	$[\mathbf{v}_{obj_1}, \mathbf{v}_{obj_2}]$	$\hat{c}_t = \alpha_{t,1} \mathbf{v}_{obj_1} + \alpha_{t,2} \mathbf{v}_{obj_2} + \alpha_{t,3} \mathbf{h}_t^l + \alpha_{t,4} \mathbf{s}$
<i>td + VisKE</i>	$[\mathbf{v}_{obj_1}, \mathbf{v}_{obj_2}]$	$\hat{c}_t = \alpha_{t,1} \mathbf{v}_{obj_1} + \alpha_{t,2} \mathbf{v}_{obj_2} + \alpha_{t,3} \mathbf{h}_t^l + \alpha_{t,4} \mathbf{s}$
<i>td (order)</i>	$[\mathbf{v}_{TARGET}, \mathbf{v}_{LANDMARK}]$	$\hat{c}_t = \alpha_{t,1} \mathbf{v}_{TARGET} + \alpha_{t,2} \mathbf{v}_{LANDMARK} + \alpha_{t,3} \mathbf{h}_t^l$
<i>td + mask (order)</i>	$[\mathbf{v}_{TARGET}, \mathbf{v}_{LANDMARK}]$	$\hat{c}_t = \alpha_{t,1} \mathbf{v}_{TARGET} + \alpha_{t,2} \mathbf{v}_{LANDMARK} + \alpha_{t,3} \mathbf{h}_t^l + \alpha_{t,4} \mathbf{s}$
<i>td + VisKE (order)</i>	$[\mathbf{v}_{TARGET}, \mathbf{v}_{LANDMARK}]$	$\hat{c}_t = \alpha_{t,1} \mathbf{v}_{TARGET} + \alpha_{t,2} \mathbf{v}_{LANDMARK} + \alpha_{t,3} \mathbf{h}_t^l + \alpha_{t,4} \mathbf{s}$

Table 2: The visual features and their attention



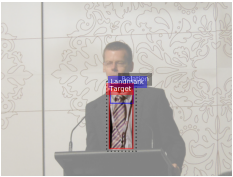
	<p><math>\langle</math> “<b>keyboard</b>”, “in front of”, “<b>computer</b>”<math>\rangle</math></p> <p><i>simple</i>                      computer</p> <p><i>bu49</i>                        keyboard on desk</p> <p><i>td</i>                             computer on top of desk</p> <p><i>td order</i>                    keyboard on computer</p> <p><i>td order + VisKE</i>        keyboard on computer</p>
	<p><math>\langle</math> “<b>mirror</b>”, “in side of”, “<b>semi</b>”<math>\rangle</math></p> <p><i>simple</i>                        truck</p> <p><i>bu49</i>                        truck has door</p> <p><i>td</i>                             door on truck</p> <p><i>td order</i>                    light on road</p> <p><i>td order + VisKE</i>        mirror on truck</p>
	<p><math>\langle</math> “<b>lanyard</b>”, “around”, “<b>neck</b>”<math>\rangle</math></p> <p><i>simple</i>                        tie</p> <p><i>bu49</i>                        man has hair</p> <p><i>td</i>                             tie around neck</p> <p><i>td order</i>                    tie around neck</p> <p><i>td order + VisKE</i>        tie around neck</p>

Figure 13: From VisualGenome: 2413204<sup>a</sup> 2417890<sup>b</sup> 2413371<sup>c</sup>

<sup>a</sup>Schmidt (2010): CC BY-NC-SA 2.0.

<sup>b</sup>Yap (2008): CC BY-NC 2.0.

<sup>c</sup>Coghlan (2011): CC BY-SA 2.0.