SubICap: Towards Subword-informed Image Captioning Supplementary Material

Naeha Sharif, Mohammed Bennamoun, Wei Liu
University of Western Australia
Western Australia
naeha.sharif@research.uwa.edu.au
{mohammed.bennamoun,wei.liu}@uwa.edu.au

Syed Afaq Ali Shah
Murdoch University
Western Australia
Afaq.Shah@murdoch.edu.au

1. Image Captioning Model

In this work, we adopt the image captioning model proposed by Herarde et al., [2]. It uses a Transformer based encoder-decoder model. The encoder consists of six layers, where each layer is composed of a multi-head self-attention layer and a feed-forward neural network. The multi-head self-attention layer consists of eight identical heads, each of which computes a query Q, key K and value V for the N token embeddings, given by:

$$Q = YW_Q, K = YW_K, K = YW_V \tag{1}$$

where, Y is the input token matrix containing the visual-appearance embeddings $\{y_1, y_2, .., y_N\}$ and W_Q, W_V, and W_K are the learned projections. The attention weight matrix θ for the visual features is formulated as:

$$\theta_A = \frac{QK^T}{\sqrt{d_k}} \tag{2}$$

where, θ_A is an attention weight matrix ($N \times N$), whose element ω_A^{lm} corresponds to the attention weights between the l^{th} and m^{th} tokens. θ_A is modified by incorporating relative geometric features $\{g_1, g_2, ..., g_N\}$ of objects. The visual appearance-based attention weights ω_A^{lm} between the l^{th} and m^{th} object are multiplied by geometric attention weights, given by:

$$\omega_G^{lm} = ReLU(Emb(\lambda(l,m))W_G) \tag{3}$$

where, $Emb(.)$ computes a positional embedding, W_G is a transformation matrix and $\lambda(l,m)$ is a displacement vector corresponding to objects *l* and *m*. $\lambda(l,m)$ for an object pair is calculated using geometric features, such as center coordinates (x_l, y_l), (x_m, y_m)), widths (w_l, w_m) and heights (h_l, h_m).

$$\lambda(l,m) = log\left(\frac{|x_l - x_m|}{w_l}\right), log\left(\frac{|y_l - y_m|}{h_l}\right), log\left(\frac{w_m}{w_l}\right), log\left(\frac{h_m}{h_l}\right)$$

The geometric and visual-appearance attention weights ω_G^{lm} and ω_A^{lm} respectively are combined to form visual-geometric attention wights ω^{lm}, which are computed as:

$$\omega^{lm} = \frac{\omega_G^{lm} \exp(\omega_A^{lm})}{\sum_{i=1}^{N} \omega_G^{li} \exp(\omega_A^{li})} \tag{4}$$

The output of each attention head is formulated as:

$$Head(Y) = softmax(\theta)V \tag{5}$$

where, θ is an NxN matrix, whose elements are the visual-geometric attention wights ω^{lm}. The outputs of all the attention heads (8 in our case) are concatenated and then multiplied to a learned projection matrix W_o, which is given by:

$$MultiHead(Q,K,V) = Concat(Head_1, .., Head_8)W_o \tag{6}$$

Next, the output of the self-attention layer (MultiHead) is fed to a point-wise feed-forward network (FFN). Similar to [2] and [6], the FFN consists of two linear projection layers with a ReLU activation function in between.

$$\text{FFN(z)} = max(0, zW_1 + b_1)W_2 + b_2 \tag{7}$$

The decoder then uses the output embeddings **q** = (q_1, q_2,... q_N) generated from the last encoder layer, to generate a sequence of subwords. We keep the encoder and decoder architecture of the transformer identical to [2] and refer the reader to [2] and [6] for more details on the transformer architecture.

2. Additional Pre-processing Details

The conventional method of forming vocabulary for the image captioning models is; to first segment the captions into tokens (words in this case) using whitespace, and then filtering off tokens that occur less than a threshold value

(generally 3-5). The filtered tokens are then used as vocabulary. Moreover, captions longer than a certain token length are truncated. We observe that a variation in vocabulary size and caption length impacts the model performance, and thus these parameter have to be chosen carefully. For our experiments, we truncate captions longer than 16 words, therefore, the maximum length of training captions in case of the baseline is 16 words. For the baseline model, each token corresponds to a word, whereas for *SubICap* which uses subword segmentation, each token might not correspond to a word, rather a subword. The maximum sequence length in case of subword modelling is longer than word-level modeling, as shown in Table 1

In our case of subword tokenization, we notice that larger vocabulary size, results into shorter sequences compared, shown in Table 1. Therefore, we adopt a filtering method to cater for overly long sequence of tokens (which are a small percentage of training examples). We select a token threshold τ such that 99% of the captions are unaltered, whereas the remaining 1% are truncated to length τ. The value of τ varies with the vocabulary size i.e., ranging between 19 to 38 for our models.

For training our captioning model, the tokenized captions are converted to sequence of ids using vocabulary to id mapping. We reserve vocabulary ids for special symbols such as EOS (End Of Sentence) and UNK (unkown). During inference the sequence of ids are mapped to a sequence of tokens, which are then detokenized to obtain the output captions.

De-tokenzation is used when the captioning model generates a sequence of tokens as output. Those tokens are fed to the detokenizer to transform them into a caption. Following is an arbitrary example of a tokenized and detokenized caption:

- **Tokenized Caption**: [the][_cat][_is][_sleep][ing]
- **De-tokenized Caption**: the cat is sleeping

Since white-space is preserved in the tokenized caption, it makes it very easy to de-tokenize and reproduce the raw caption with the following python script:

Models	Vocabulary Size	Max. Sequence Length (# of tokens)
Baseline	9,486	16
Ours-3k	3,078	19
Ours-2k	2,079	21
Ours-1k	1,085	24
Ours-500	579	31
Ours-300	335	38

Table 1. Shows the comparison between the models in terms of the maximum training sequence length.

Models	Unique Captions (%age)	Trainable Parameter	Max. Sequence Length (# of tokens)
Baseline	69.0	54.9M	16
SubICap-char	75.3	45.3M	82
SubICap-bpe	74.0	46.3M	25
SubICap-1k	74.8	46.3M	24

Table 2. Comparison between models which differ in terms of the tokenization of training captions.

```
detok = ''.join(tokens).replace('_', ' ')
```

3. Comparison between Various Tokenization Methods

Tokenization refers to the process of segmenting stream of characters into individual units, known as tokens. The sequence of tokens are then mapped to ids, in order to train the language model. Tokenization is one of the most important steps in language modeling because it impacts the way a model sees the textual input, i.e., as a sequence of words, subwords or characters.

Tokenization can help reduce the vocabulary size [3] and increase the training examples for each token in the vocabulary. Domingo et. al, [1] investigated the impact of various tokenizers on machine translation quality. They found that tokenizers had significant impact on the quality of translations. Here, we experiment with different tokenization methods: word-level, character-level and subword-level.

For word-level tokenization, we use a standard whitespace tokenizer, which segments the words in captions using whitespace as a separator. For character and subword-level tokenization, we use SentencePiece [4], and specify the modes in the tool. For example, in order to perform character-level segmentation we specify mode='char', and a vocabulary= 95 (since there are 95 printable ASCII characters, which include alphabets, numbers and punctuation marks etc.,). For subword segmentation we compare two algorithms, 1) BPE and 2) Unigram Language Model.

Results of our experiments are reported in Table 3 and Table 2, which show the impact of different tokenization methods on the metric scores, vocabulary size, trainable model parameters and percentage of unique captions. SubICap-1k, which uses Unigram Language Model for subword tokenization, achieves the best metric scores amongst all. SubICap-1k also strikes a balance between maximum sequence length and number of model parameters compared to character and word-level tokenization-based methods (Baseline and SubICap-char)

Models	Segment/token	Tokenizer	Vocab size	B1	B2	B3	B4	M	R	C	S
Baseline	Word	Standard Whitespace	9,486	75.2	58.8	44.6	33.7	27.5	55.5	111.0	21.0
SubICap-bpe	Subword	BPE	1,076	75.8	59.7	46.1	35.5	29.7	56.6	114.4	21.1
SubICap-char	Subword	Character-based	95	75.4	59.2	45.5	34.8	29.2	55.7	113.0	21.0
SubICap-1k	Subword	Unigram Language Model	1,085	**76.7**	**60.8**	**47.1**	**36.2**	**29.7**	**56.9**	**116.1**	**21.2**

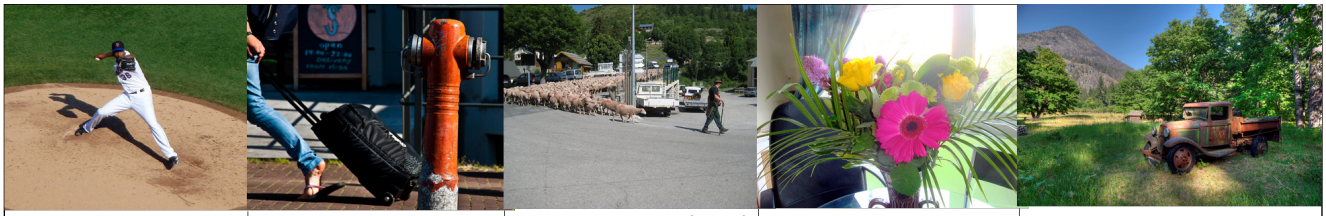
Table 3. Impact of various segmentation algorithms on the metric performance.

Baseline: 'two stuffed teddy bears sitting on a couch'	**Baseline:** 'two colorful feathers standing next to each other'	**Baseline:** 'a young boy holding a baseball bat at a ball'	**Baseline:** 'a group of people standing in an elephant in a city'	**Baseline:** 'a woman riding a horse jumping over two pink cones'
Ours: 'a baby sitting on a couch with two teddy bears'	**Ours:** 'two peacocks standing next to each other in a field'	**Ours:** 'a young boy holding a baseball bat in a batting cage'	**Ours:** 'a group of people standing around a cage in the street'	**Ours:** 'a woman riding a horse jumping over an obstacle'

Figure 1. Qualitative comparison of captions generated by our model (SubICap-1k) and baseline for images in the MSCOCO offline test set. Mistakes are highlighted in red color.

4. Qualitative Comparison

In order to perform a qualitative comparison between our proposed model (SubICap-1k) and the baseline, we provide the examples of captions generated by these models in Figure 1, Figure 2, Figure 3, and Table 4. Both models (baseline and SubICap) are fine-tuned for CIDEr-D score, and use a beam size of 5 during inference. Table 4 shows examples of captions which differ in lexical as well as semantic quality, however, still achieve an equal CIDEr-D score. Figure 2 and Figure 3 show comparison between captions which differ in terms of the lexical quality. Our model achieved a higher METEOR score than the baseline, which reflects that the captions generated by our model are lexically sound [5]. The examples shown in Figure 2 and Figure 3, further strengthen our point of view.



Baseline: 'a baseball player throwing a ball on a field'	**Baseline**: ''a person walking next to a red fire hydrant'	**Baseline**: 'a herd of sheep walking down a street'	**Baseline**: 'a group of colorful flowers sitting in a vase'	**Baseline**: 'a rusted truck parked in the middle of a field'
***Ours*: 'a baseball player pitching a ball on the mound'**	***Ours*: 'a person pushing a bag of luggage next to a fire hydrant'**	***Ours*: 'a man herding a herd of sheep down a road'**	***Ours*: 'a bunch of colorful flowers in a vase on a table'**	***Ours*: 'an old truck parked in the grass in a field'**

Figure 2. Qualitative comparison of captions generated by our model (SubICap-1k) and baseline for images in the MSCOCO offline test set. Captions generated by our model are lexically better compared to the ones generated by the baseline

Baseline: 'a black and white photo of a man riding a boat in the water'	**Baseline**: 'a blue train is sitting on the tracks in a building'	**Baseline**: 'a group of birds flying over a body of water'	**Baseline**: 'a small plane is taking off from a runway'	**Baseline**: 'a group of red umbrellas sitting under a table'
***Ours*: 'a black and white photo of a man windsurfing in the water'**	***Ours*: 'a blue and yellow train on the tracks in front of a building'**	***Ours*: 'a group of seagulls flying over a boat in the water'**	***Ours*: 'a small propeller plane taking off on a runway'**	***Ours*: 'a patio with a red umbrella and tables'**

Figure 3. More examples of captions generated by our model (SubICap-1k) and baseline for images in the MSCOCO offline test set. Captions generated by our model are lexically better compared to the ones generated by the baseline

Images	Captions and Scores	Ground Truth Captions
	**Ours**: 'a man on a horse herding a herd of sheep' [C: 179.1, M: 31.1, S: 34.4] **Baseline**: 'a man riding a horse next to a herd of sheep' [C: 179.1, M: 30.8, S: 20.6]	**GT1**: 'a herd of sheep walking across green grass' **GT2**: 'a man on a horse corralling sheep with his two dogs' **GT3**: 'a man and two dogs gathering a herd of sheep'
	**Ours**: 'a woman riding a horse jumping over an obstacle' [C: 154.3, M: 93.6, S: 20.6] **Baseline**: 'a woman riding a horse jumping over two pink cones' [C: 154.3, M: 93.6, S: 12.9]	**GT1**: 'a young person ridding a horse jumps a gate in a competition' **GT2**: 'a woman is riding a horse as it jumps over a bar' **GT3**: 'a woman riding a horse jumps over an obstacle'
	**Ours**: 'a pizza with peppers and olives in a box' [C: 93.6, M: 22.6, S: 17.1] **Baseline**:'a pizza with olives and cheese on a table' [C: 93.6, M: 13.8, S: 11.4]	**GT1**: 'large pizza covered in pepperoni, olives, peppers, onions and mushrooms' **GT2**: 'pizza with everything on it sitting on counter' **GT3**: 'a very big pizza that was just made to order'
	**Ours**: 'a young boy holding a baseball bat in a batting cage' [C: 152.9, M: 27.3, S: 19.1] **Baseline**: 'a young boy holding a baseball bat at a ball' [C: 152.9 , M: 21.1, S: 19.3]	**GT1**: 'young boy ready to bat in a little league uniform' **GT2**: 'a boy in a helmet and uniform holding a bat' **GT3**: 'a boy holding a baseball bat next to fence and wearing a baseball helmet'
	**Ours**: ' a clock on top of a mantle in front of a wall' [C: 102.3, M: 19.0, S: 26.3] **Baseline**: 'a clock sitting on top of a table with a statue' [C: 102.3 , M: 20.0, S: 17.6]	**GT1**: 'a golden clock rhino sculpture sitting on top of a fireplace' **GT2**: 'a clock on top of a rhino on a shelf' **GT3**: 'gold clock on a base shaped like a rhinoceros and a sign on a shelf in front of a painting'

Table 4. A comparison of captions generated by our model vs. baseline. Scores of few commonly used metrics such as CIDEr-D (C), METEOR (M), and SPICE (S) are provided with the generated captions. The captions are generated by models (baseline and SubICap) fine-tuned for CIDEr-D score, setting beam size to 5.

References

[1] Miguel Domingo, Mercedes Garcıa-Martınez, Alexandre Helle, Francisco Casacuberta, and Manuel Herranz. How much does tokenization affect neural machine translation? *arXiv preprint arXiv:1812.08621*, 2018.

[2] Simao Herdade, Armin Kappeler, Kofi Boakye, and Joao Soares. Image captioning: Transforming objects into words. In *Advances in Neural Information Processing Systems*, pages 11135–11145, 2019.

[3] Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. *arXiv preprint arXiv:1804.10959*, 2018.

[4] Taku Kudo and John Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*, 2018.

[5] Naeha Sharif, Lyndon White, Mohammed Bennamoun, Wei Liu, and Syed Afaq Ali Shah. Lceval: Learned composite metric for caption evaluation. *International Journal of Computer Vision*, 127(10):1586–1610, 2019.

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.