Attention Based Image Captioning in a Merge Architecture

Catherine Cao
August 2019

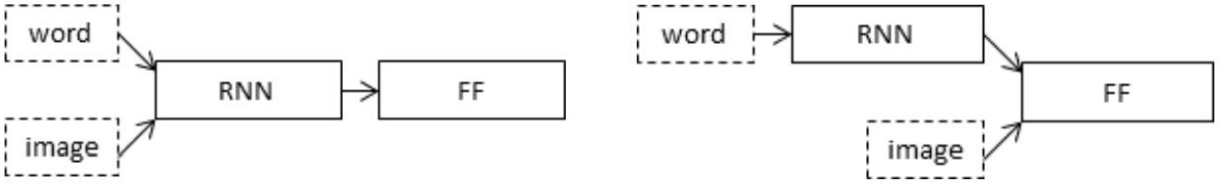
Abstract

There are a few different approaches that have achieved state-of-the-art results in image captioning. One of the main differentiating factors in the architecture is when the image as a thought vector is combined with the language model. When the image is used as input to the language decoder, it is called an inject architecture, whereas an image vector combined with the output of the RNN decoder is called a merge architecture. Building on previous work by others, this paper aims to explore the difference in performance of applying attention mechanism in an inject vs. merge architecture. Results of both inject and merge model with attention mechanisms are evaluated and compared in the result section. Overall, a merge architecture did not improve performance over an inject architecture.

Introduction

Image caption generation is the task of generating a natural language description of the content of an image (Bernardi et al., 2016). It has been viewed as a difficult problem as it requires both recognizing an image and generating a caption based on the content of that image. The state-of-the-art approach is to use a multi-model architecture, where an image model is built to generate a thought vector of the image, and a RNN type language model is used to generate the correct caption with input from the image model.

An interesting variation of this architecture is around where to provide the image information to the RNN model. Tanti et al. (2017) described this variation as the inject vs. merge model. In an inject model, the image vector is used as an input to the language model. This is the more dominant view. Another view is to build the language model independently of the image, then combine with the image at a later step. This is called a merge model. Figure 1 from Tanti et al. (2017) illustrates the difference between the two approaches.



(a) Inject model takes image vector as input to the RNN (b) Merge model combines image vector after the RNN

Figure 1: The inject and merge architectures for caption generation.

One practical advantage of the merge model is that it allows for transfer learning. Since the RNN is kept separate from the image model, it can be potentially pre-trained on a larger corpus. It has also been found that the merge model slightly outperforms the inject model with the same number of layers and dimensions, and the merge model tends to use more diversified words from a given vocabulary (Tanti et al. 2017).

Another exciting development of the image captioning task is incorporating attention mechanism into the model. Xu et al. in the paper *Show, Attention, and Tell: Neural Image Caption Generation with Visual Attention (2016)* incorporated attention mechanism into image captioning. With attention mechanism, the RNN model attends over the entire image to focus on the most relevant features. In addition, attention mechanism provides words/image alignment, which can be used to examine and understand the inner workings of the model. The original paper uses both soft and hard attention. This paper replicates the “soft attention” from the paper with alternative architectures.

This paper aims to combine the above two interesting concepts in image captioning, and experiment with applying attention mechanism in an inject and merge architecture.

Data

The data used for this paper is the COCO 2014 dataset. There is a training, validation, and test set in the original COCO data. This paper only looks at the training set. The training set contains 82,783 images, and an average of 5 different captions per image, with a total of 414,113 distinct image/caption combinations. In order to quickly iterate and experiment with different architectures, this paper uses 30,000 unique images, and randomly chosen captions as the training set. The validation and test set each contain 5,000 images and randomly chosen captions. The images in training, validation, and test sets do not overlap. In other words, the images in validation or test set do not exist in the training set.

The model is then built and validated on the training and validation set respectively. The final results are evaluated on the same test set.

Model

1. Baseline

First, two baseline models are built using a simple inject and merge model to familiarize with the architecture. In both models, the InceptionV3 model pre-trained on ImageNet data is used as the image encoder. In both models the last layer before the softmax layer with shape (2048,) was used as the image feature, and a single layer GRU is used as the language model. In the merge model, the image vector is added to the output from the GRU. In the inject model, the image vector is set as the initial state of the GRU.

2. Attention Mechanism

After the baseline models are working, attention mechanism is incorporated. There are three parts in the model with attention: encoder, decoder, and attention. This paper leverages the tensorflow implementation of the original *Show, Attention, and Tell: Neural Image Caption Generation with Visual Attention (2016)* paper, and experiments with different architectures.

The image encoder uses the third last layer in the InceptionV3 model pre-trained on ImageNet data. This third last layer has a shape of (8, 8, 2048), which allows us to apply attention over the 8x8 grid of the original image. The layer is further flattened to (64, 2048), where the 64 are the image features that attention will be given to, and each feature has 2048 dimensions. The encoder is simply a fully connected layer to map the 64 features to a different dimension.

The decoder is a GRU layer, where the input is each word in the caption. The GRU then attends over the 64 image features to obtain a weighted context vector from the image. The difference in the inject and merge model is where this attention happens.

2.1 Attention in Inject Architecture

In the inject architecture, the hidden state of GRU is sent back to attend over the 8x8 image thought matrix. An attention weight matrix over the 64 features are calculated, then multiplied by the original image features to obtain a weighted version of the features, called a “context vector”. This context vector helps bring out the more relevant area in the image. The context vector is then concatenated with the original hidden state of the GRU, then sent back to the GRU as the new hidden state. Figure 2 illustrates the architecture of an inject model with attention.

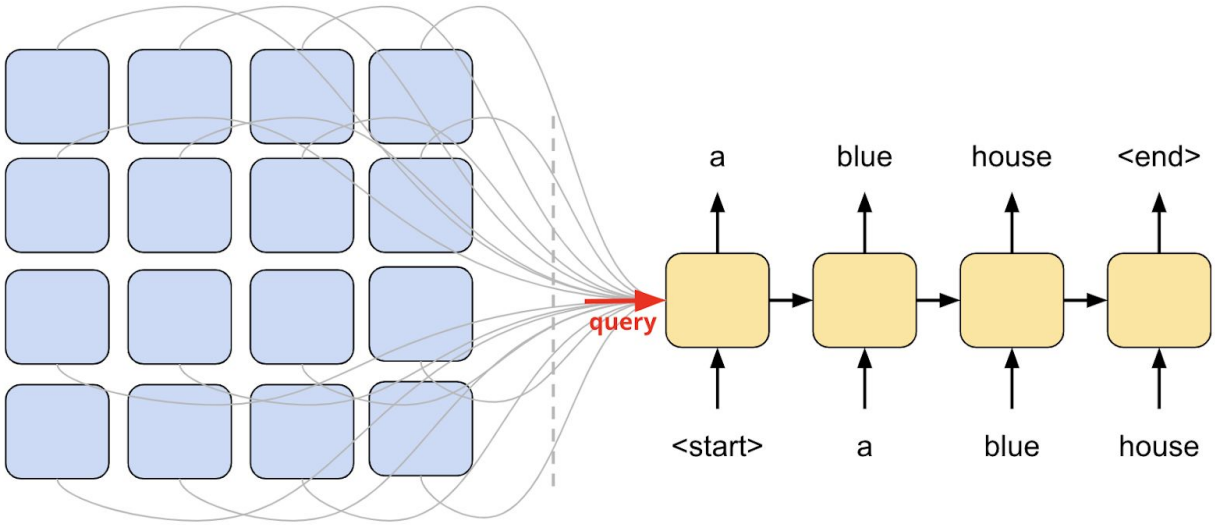


Figure 2: Attention mechanism in an inject architecture

2.2 Attention in Merge Architecture

Similar to the attention mechanism in an inject architecture, the merge architecture sends back a query that attends over the 8x8 image thought matrix. The resulting context vector is then combined with the original query. The difference is that in the merge model, the query is the output from the GRU layer. In this architecture, the GRU is essentially independent of the image vector, therefore can potentially be pre-trained on a larger corpus.

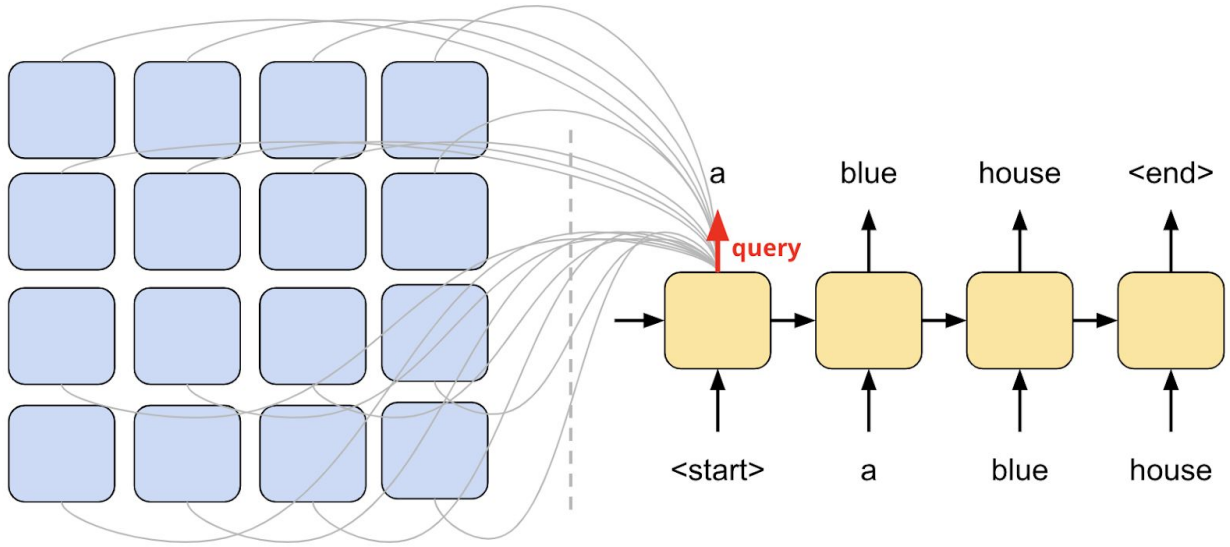


Figure 3: Attention mechanism in a merge architecture

In both models, the output from the GRU layer then runs through two additional fully connected layers with a softmax at the end to produce the final caption word. Three fully connected layers have also been tested on the merge model, but the result did not improve much based on the BLEU scores.

Each model was trained for 20 epochs with adam optimizer and minimal hyper-parameter tuning. Learning rates were decreased from 0.001 to 0.0005 after 10 epochs.

3. Pre-Trained Word Embeddings

The input captions are mapped to a word embedding matrix of 200 dimensions. Both trained and pre-trained embeddings are experimented. For pre-trained word embeddings, the GloVe embeddings with 6B tokens and 200 dimensions were used without further fine tuning. With the use of pre-trained embeddings, training time decreased by roughly 25-30%, from 1100 seconds per epoch to 800 seconds per epoch, and performance as measured by BLEU-(1,2,3,4), ROUGE-L, and CIDEr improved by an average of 27%[1] (20% for inject model, and 33% for merge model). Further details are shown in the results section.

[1] The 27% improvement also includes adding a ReLU activation of the first fully connected layer.

4. Beam Search

This paper also implements a length weighted beam search at inference to predict captions. The top k predictions for each word are kept, and the probabilities for the entire sequence is calculated as:

$$P_{sequence} = \frac{\sum_{i=0}^{n} log(P_i)}{n}$$

where i is the position of the word, and n is the length of the sequence. At the end of each word prediction, the top k sequences with the highest total probabilities are kept.

Beam search with k=3 had an average improvement of 8.3% over greedy search, as measured by the same 7 metrics and averaged between the inject and merge architectures. However, this improvement reached diminishing returns with higher value of k. Beam search with k=5 had an average improvements of 4.0% over greedy search, with an adverse effect of -1.2% for inject model and favorable effect of 9.2% for merge models. In general, a more favorable effect from beam search on the merge architecture has been observed. Further details are shown in the results section.

Results

Predicted captions on the test set are evaluated using the following standard evaluation metrics: BLEU-(1,2,3,4) (Papineni et al., 2002), METEOR (Denkowski and Lavie, 2014), ROUGE-L (Lin and Och, 2004), and CIDEr (Vedantam et al., 2015). The top ranking result on the COCO captioning leaderboard is as follows:

- BLEU-1: 0.795
- BLEU-2: 0.635
- BLEU-3: 0.485
- BLEU-4: 0.363
- METEOR: 0.277
- ROUGE-L: 0.573
- CIDEr: 1.196

The models in this paper achieved much lower scores than the top ranking results. One reason is that the model was only trained on 30,000 examples. As the original COCO 2014 dataset contains 414,113 examples in the training set, the models were trained on merely 1/14 of the data. Including all of the training set data would significantly improve the results. Nonetheless, we can still use the results to compare the performance of the two different architectures.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-L	CIDEr
Attention Inject Baseline	0.3531	0.2327	0.1459	0.0889	0.1621	0.3852	0.4225
Attention Inject w/ GloVe[2]	0.4797	0.3170	0.1969	0.1185	0.1648	**0.3940**	0.4124
Attention Inject w/ GloVe Beam-3	**0.4894**	**0.3211**	**0.2053**	**0.1272**	**0.1664**	0.3860	**0.4232**
Attention Inject w/ GloVe Beam-5	0.4737	0.3069	0.1955	0.1196	0.1649	0.3791	0.4112
Attention Merge Baseline	0.2582	0.1560	0.0914	0.0518	0.1216	0.3001	0.2614
Attention Merge w/ GloVe[3]	0.3295	0.2164	0.1356	0.0827	0.1395	0.3486	0.3319
Attention Merge w/ GloVe Beam-3	0.3937	0.2595	0.1679	0.1043	0.1432	0.3523	0.3549
Attention Merge w/ GloVe Beam-5	0.3787	0.2455	0.1583	0.0986	0.1393	0.3424	0.3392

Table 1. Model evaluation results

In general, the merge architecture fails to improve over the inject architecture in an attention framework. It is clear that there is a significant difference in performance between the two architectures, unlike what was found in the paper *What is the Role of Recurrent Neural Networks (RNNs) in an Image Caption Generator? (Tanti et al. 2017)*. One possible explanation is that the attention context vector in the merge architecture do not combine well with the output from the GRU, since it is not incorporated into the language generation process (the GRU). I have tried adding more fully connected layers after the concatenation to help with this, but it did not improve the results.

Another observation is that the merge model tends to produce captions with persistent phrases repeated until the maximum length of the sentence. This might come from a lack of training or data. However, it is interesting that this behavior, although observed, is not as frequent in the inject architecture. Beam search in this case helps remove the persistent patterns. This may be the reason why a bigger improvements from beam search is observed in the merge model.

Future Work

The models in this paper were trained on only 30,000 images due to limit on training time. It would greatly improve the results if the whole COCO training set is used for model building.

The pre-trained word embeddings used was the GloVe embeddings with 200 dimensions. A more advanced word embedding with context such as BERT could be used for future improvements.

The model was built without any hyper-parameter tuning (except a decrease of the learning rate after 10 epochs). Future work can experiment with hyper-parameter tuning, including the word embedding dimensions, number of fully connected layers, number of GRU layers, number of

[2] Includes adding a ReLU activation of the first fully connected layer
[3] Includes adding a ReLU activation of the first fully connected layer

output units in the layers, activation functions, optimizers, regularization such as batch normalization and drop out, etc.

In the merge model, the context vector and the output of the GRU are simply concatenated and fed through a few fully connected layers. More experiments can be done on other ways to combine.

Finally, the benefit of a merge architecture is that the RNN is kept separate from the image vectors and therefore can be pre-trained on a larger dataset. So experiments with using a pre-trained RNN can potentially improve the result and leverage the benefit of the merge architecture.

Conclusion

This paper experiments with adding attention mechanism in an inject and merge architecture. The result shows that an inject architecture is more conducive to the attention mechanism. It is also found that beam search and pre-trained word embeddings both improve the results in both architectures.

References

Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. (2015). Show and tell: A neural image caption generator. In Proc. CVPR'15.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R., and Bengio, Y. (2016). Show, Attend and Tell: Neural Image Caption Generation with Visual Attention.

Tanti, M., Gatt, A., and Camilleri, P. (2017). What is the Role of Recurrent Neural Networks (RNNs) in an Image Caption Generator?

Tanti, M., Gatt, A., and Camilleri, P. (2018). Where to put the Image in an Image Caption Generator.

Vedantam, R., Zitnick, C. L., and Parikh, D. (2015). CIDEr: Consensus-based image description evaluation. In Proc. CVPR'15.

Banerjee, S. and Lavie, A. (2005). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In Proc. Workshop on Intrinsic and extrinsic evaluation measures for machine translation and/or summarization, volume 29, pages 65–72.

Kilickaya, M., Erdem, A, Ikizler-Cinbis, N, and Erdem, E. (2016). Re-evaluating Automatic Metrics for Image Captioning. ACL Anthology.

DevlinF, J., Cheng, H., Fang, H., Gupta, S., Deng, L., He, X., Zweig, G., and Mitchell, M. (2015). Language Models for Image Captioning: The Quirks and What Works. ACL Anthology.

Appendix

1. Beam search improves results in some cases

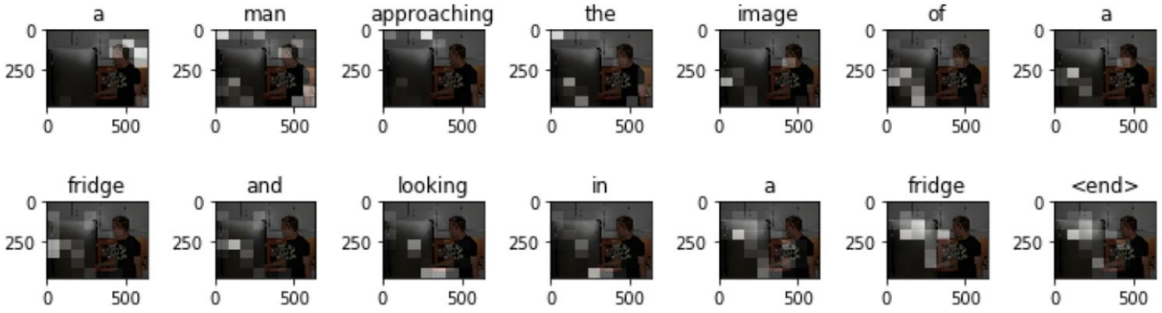
```
True Caption: giraffes big and small roaming around in their natural habitat
Greedy Search: two giraffes standing in a tree
Beam Search k=3: a group of giraffes standing in the savannah
Beam Search k=5: a group of giraffes standing in an open field
```

Real Caption: a man riding on the back of a motorcycle
Greedy Search: a person riding down a track
Beam Search k=3: a person riding down the street
Beam Search k=5: a person who is riding a motorcycle down the street

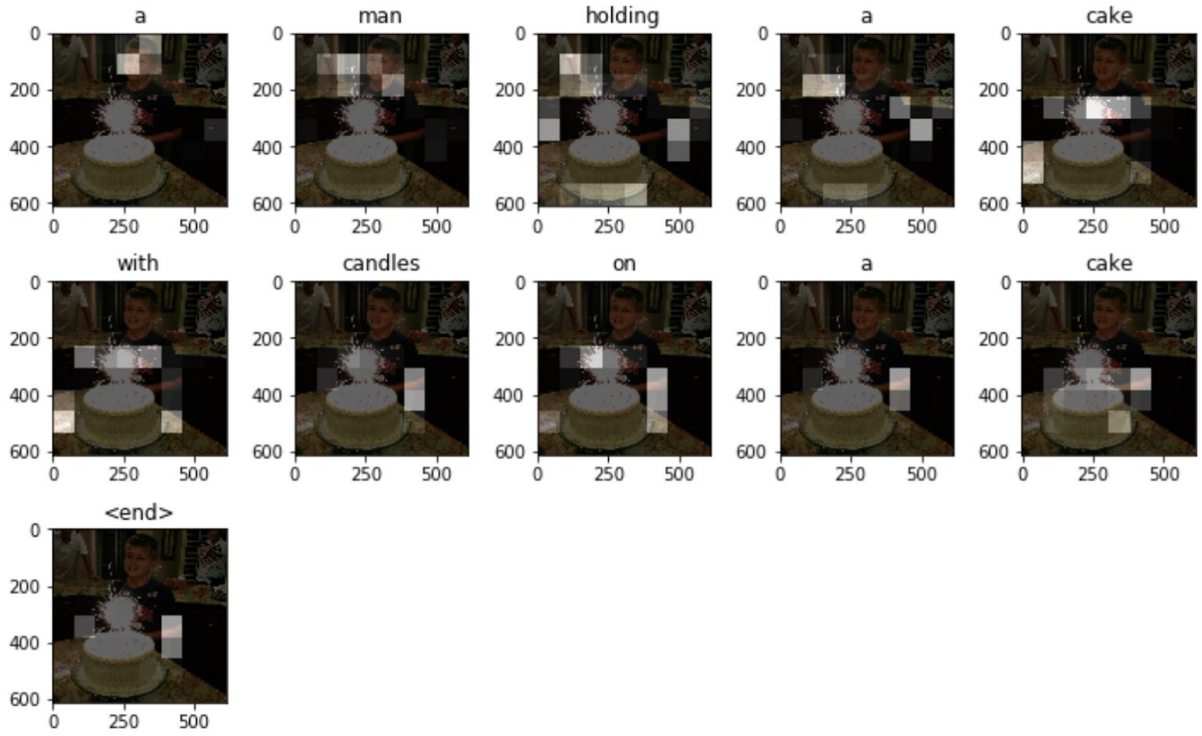
True Caption: a group of kids sit at a table eating
Greedy Search: a group of people eating packed and eating packed and eating packed and eating packed and eating packe
d and eating packed and eating packed and eating packed and eating packed and eating packed and eating packed and eat
ing packed and eating packed and eating packed and
Beam Search k=3: a group of people sitting at a table
Beam Search k=5: a group of people sitting at a table

2. Attention mechanism highlights salient object in image

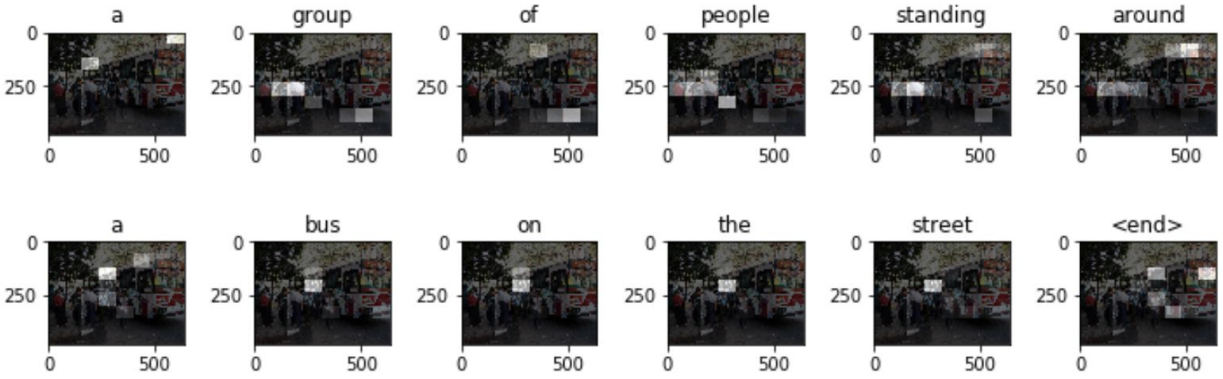
```
True Caption: <start> a man approaching refrigerator with arm stretched out <end>
Predicted Caption: a man approaching the image of a fridge and looking in a fridge <end>
```



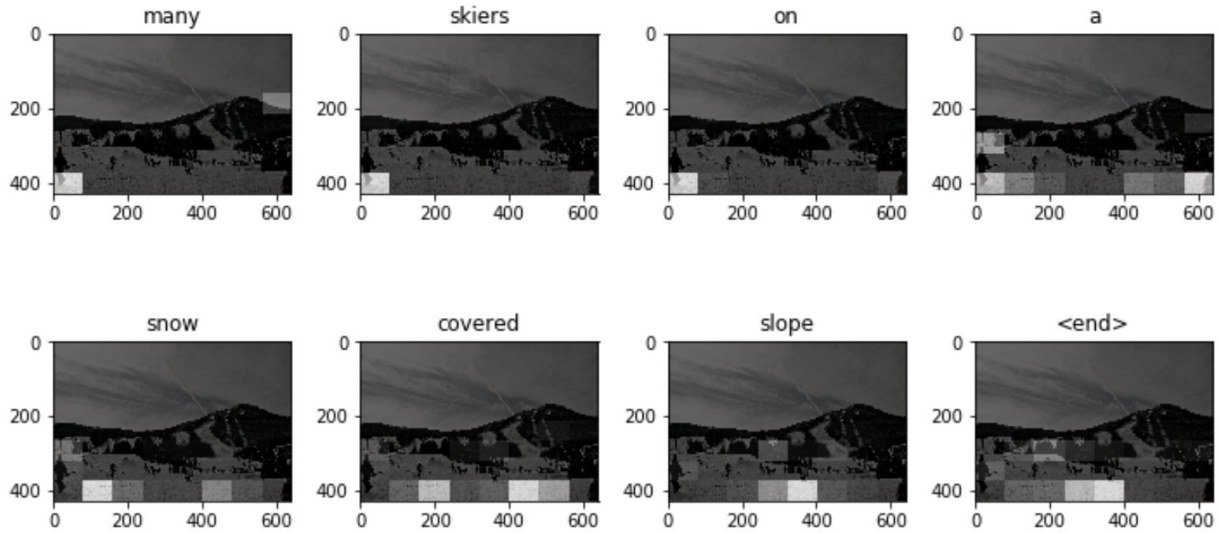
True Caption: <start> a boy smiling over a birthday cake on a kitchen counter <end>
Predicted Caption: a man holding a cake with candles on a cake <end>



True Caption: <start> a group of people standing beside a red and white bus <end>
Predicted Caption: a group of people standing around a bus on the street <end>



True Caption: <start> skiers walking around at a ski area <end>
Predicted Caption: many skiers on a snow covered slope <end>



```
True Caption: <start> a man doing skateboard sticks on the beach with headphones on <end>
Predicted Caption: a person riding a skateboard down a beach <end>
```